OptiDocs

sp614x, WhoAteMyButter

Sep 24, 2023

CONTENTS

This is the documentation for OptiFine, a Minecraft optimization mod. It contains detailed tables, pictures, tutorials, and other information about OptiFine's features.

This documentation assumes you are using the latest version. Notes are not made for old versions.

CHAPTER

ONE

CAPES

OptiFine capes are a cosmetic given to OptiFine users who have donated.

The cape can be edited by opening the options menu in *Skin Customization* → *OptiFine Cape* → *Open Cape Editor*.

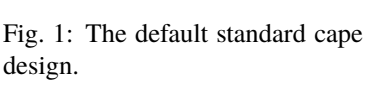

Fig. 1: The default standard cape design.

1.1 Styles

Capes can be customized and their style can be changed one of two styles:

Table 1: Cape styles

Name	Design	Options
Standard		None
White		None
Gray		None
Black		None
Red		None
Green		None
Blue		None
Yellow		None
Purple		None
Cyan		None
Custom		Top, Bottom, Text, Shadow
Banner		Top, Bottom, Shadow

1.2 Banner capes

Banners can be used as designs for OptiFine capes.

Legacy

Do not use needcoolshoes anymore for banner generation.

In the past, `https://www.needcoolshoes.com/banner` was used to generate OptiFine cape banners. However, needcoolshoes has often gone offline.

It is now recommended you use https://livzmc.net/banner/ **instead.**

Fig. 2: The LivzMC cape/banner editor.

Banner designs have some restrictions and limitations:

- Banners cannot have over 8 layers, *not including the base color*.
- Banners cannot contain the "Thing" pattern (resembles the Mojang logo).
- Banners with offensive patterns may be manually removed at any time.

1.3 Anniversary capes

Anniversary capes are cape styles that are only usable on certain times and dates.

1.3.1 10th anniversary

Most notably, the "10" cape replaced the Classic design during OptiFine's 10th anniversary.

Texture file	Texture resolution	Dates applied (YYYY/MM/DD)
	46px by 22px	2021/04/08 - 2021/04/15

April 8th, 2021 marked OptiFine's 10th birthday. To celebrate, all classic "OF" capes were changed to a "10" design. Colors were the same as the normal classic capes.

Fig. 3: How the "10" cape looks on a player

Table 2: Cape styles

Name	Design
Standard	
White	
Gray	
Black	
Red	
Green	
Blue	
Yellow	
Purple	
Cyan	

1.4 Special capes

Some players have "special" capes, given by sp614x. These players are often long-time contributors or supporters of OptiFine.

These capes **are not** purchasable, and are not given to regular donators.

As of August 2022, only 1 player has had their cape revoked.

Note

Cape names with an asterisk (*) are not official and are given only for categorization. A name with **(old)** means that player no longer wears that cape *currently*.

Table 3: List of special capes

Name	Design	Owner(s)	Cape purpose	Texture author
HD Grey 1*	Fig. 4: (source)	EskiMojo14 (old)	For testing HD capes.	EskiMojo14
HD Grey 2*	Fig. 5: (source)	EskiMojo14 (old)		Modified version of HD Grey 1.
HD Grey 3*	Fig. 6: (source)	EskiMojo14		Modified version of HD Grey 2.
Orange Heart*	Fig. 7: (source)	FakeRetroBot (old), trollim	For moderating the OptiFine Discord. Has since been revoked for being sold.	Kirbyrocket
Graph 1*	Fig. 8: (source)	filefolder3 (old)	For compiling graphs and statistics about OptiFine releases.	jckt
Graph 2*	Fig. 9: (source)	filefolder3		Modified version of Graph 1.
Code 1*	Fig. 10: (source)	jckt (old)	For moderating & managing the OptiFine Discord. Also for creating the OptiFine Discord bot.	jckt
Code 2*	Fig. 11: (source)	jckt		Modified version of Code 1.
Snow*	Fig. 12: (source)	Jiingy	For moderating & managing the OptiFine Discord.	Jiingy
Postmaster 1	Fig. 13: (source)	ZenW	For working on support e-mail.	Jiingy
Postmaster 2	Fig. 14: (source)	KaiAF (old), IrisJS (old), Fluffion (old), Jiingy		Modified version of Postmaster 1
Postmaster 3	Fig. 15: (source)	IrisJS, Fluffion		Modified version of Postmaster 2
Rainbow*	Fig. 16: (source)	KaiAF (old)		
Transgender*	Fig. 17: (source)	KyriaBirb	For moderation of OptiFine Discord.	ZenithKnight
Lego*	Fig. 18: (source)	MrCheeze445	For moderating & managing the OptiFine Discord.	Jiingy
HD Red*	Fig. 19: (source)	therealokin		Modified version of HD Grey 1, modified by sp614x
Lion*	Fig. 20: (source)	sp614x	Owned by the developer of OptiFine.	sp614x

1.5 Locking

Capes can be "locked", so they can only be moved to a different username after logging in through the OptiFine website. Once locked, they cannot be moved from the in-game Cape Editor.

Fig. 21: In the in-game editor, locked capes cannot be moved through the Minecraft account.

1.6 Technical details

The OptiFine client fetches capes for donators by querying the URL `http://s.optifine.net/capes/<NAME>.png`.

The native size of Minecraft's capes are 64px x 32px. The OptiFine cape server generally returns capes that are 46px x 22px, with the exception of *specialty capes* granted to specific users.

On load, a canvas is initialized with a size of 64x32. If the supplied cape image is larger in width or in height, it doubles the dimensions of the canvas repeatedly until it is large enough to fit the original image. The supplied cape image is then placed onto this larger canvas, ensuring that the resulting image inherits the proper 2:1 aspect ratio.

This has the added benefit of compatibility with the 44x34 legacy capes that did not include the elytra section.

Fig. 22: A labeled template of the cape UV map.

1.7 Unknown project capes

Warning

This information is not official and was gathered by ZenithKnight.

Two prototypes for new cape textures were presented by sp614x as part of an unknown project, but these were scrapped.

Fig. 23: The Standard cape design.

Fig. 24: A black and white variant.

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

CHAPTER

TWO

CUSTOM PLAYER MODELS

Custom Player Models (CPM; sometimes unofficially called Special Cosmetics) is a feature that can change or add to the player model.

CPMs are like *Capes*, in that they are loaded on join and can be associated with a specific player.

Fig. 2: Available CPMs can be chosen under the Cape Change menu.

CPMs cannot be bought and like *Special capes*, they are not available to normal donators.

Fig. 1: sp614x with a hat, available only to him.

2.1 List of available CPMs

2.1.1 MrCheeze

This cosmetic is titled `mrcheeze`. This is the username of an OptiFine Discord moderator. It is at http://s.optifine.net/items/mrcheeze/model.cfg.

It is meant to resemble the rings at the top of Lego blocks, as MrCheeze's cape is a reference to Legos, as well as his skin and profile.

Model render

Model JSON

```
{
        "type": "PlayerItem",
        "usePlayerTexture": true,
        "textureSize": [
                64,
                64
        ],
        "models": [
                {
                "id": "tail",
                "type": "ModelBox",
                "attachTo": "body",
```

(continues on next page)

(continued from previous page)

```
                "invertAxis": "z",
                "translate": [
                        0,
                        0,
                        0
                ],
                "submodels": [
                        {
                        "part": "tail",
                        "id": "tail",
                        "invertAxis": "z",
                        "translate": [
                                0,
                                3.5,
                                -7.2
                        ],
                        "rotate": [
                                -57,
                                0,
                                0
                        ],
                        "boxes": [
                                {
                                "coordinates": [
                                        -0.5,
                                        -3,
                                        -7.2,
                                        1,
                                        7,
                                        1
                                ],
                                "textureOffset": [
                                        60,
                                        20
                                ]
                                }
                        ]
                        },
                        {
                        "part": "tail2",
                        "id": "tail2",
                        "invertAxis": "z",
                        "translate": [
                                0,
                                -8.8,
                                -16.9
                        ],
                        "rotate": [
                                -102.5,
                                0,
                                0
                        ],
```

(continues on next page)

(continued from previous page)


```
                        "boxes": [
                                {
                                "coordinates": [
                                        -0.5,
                                        4.5,
                                        -22,
                                        1,
                                        6,
                                        1
                                ],
                                "textureOffset": [
                                        60,
                                        20
                                ]
                                }
                        ]
                        }
                ]
                }
        ]
}
```


2.1.2 Jiingy Hat

This cosmetic is titled `jiingy`. This is the username of an OptiFine Discord moderator. It is at http://s.optifine.net/items/jiingy/model.cfg.

It resembles a ushanka, a Russian hat.

Model render

Model JSON


```
{
        "type": "PlayerItem",
        "textureSize": [
                62,
                23
        ],
        "models": [
                {
                        "id": "Main",
                        "type": "ModelBox",
                        "attachTo": "head",
                        "invertAxis": "xy",
                        "translate": [
                                0,
```


(continues on next page)

(continued from previous page)

```
                0,
                0
            ],
            "rotate": [
                5,
                0,
                0
            ],
            "boxes": [
                {
                    "coordinates": [
                        -5,
                        4.5,
                        -6,
                        10,
                        5,
                        5
                    ],
                    "textureOffset": [
                        0,
                        0
                    ]
                },
                {
                    "coordinates": [
                        -4.5,
                        3,
                        -2.5,
                        9,
                        6,
                        7
                    ],
                    "textureOffset": [
                        0,
                        10
                    ],
                    "sizeAdd": "0.1"
                },
                {
                    "coordinates": [
                        3.5,
                        0,
                        -1.5,
                        1,
                        3,
                        4
                    ],
                    "textureOffset": [
                        30,
                        1
                    ],
                    "sizeAdd": "0.1"
```

(continues on next page)

(continued from previous page)

```
                                },
                                {
                                        "coordinates": [
                                                -4.5,
                                                0,
                                                -1.5,
                                                1,
                                                3,
                                                4
                                        ],
                                        "textureOffset": [
                                                30,
                                                1
                                        ],
                                        "sizeAdd": "0.1"
                                }
                        ]
                }
        ]
}
```

2.1.3 Jiingy Scarf

This cosmetic is titled `jiingy_scarf`. This is the username of an OptiFine Discord moderator. It is at http://s.optifine.net/items/jiingy_scarf/model.cfg.

It is a scarf.

Model render

Model JSON

```
{
        "type": "PlayerItem",
        "textureSize": [
                62,
                23
        ],
        "models": [
                {
                        "id": "Main",
                        "type": "ModelBox",
                        "attachTo": "body",
                        "invertAxis": "xy",
                        "translate": [
                                0,
                                0,
```

(continues on next page)

(continued from previous page)

```
                0
            ],
            "rotate": [
                0,
                0,
                0
            ],
            "boxes": [
                {
                    "coordinates": [
                        -5,
                        0,
                        -2.5,
                        10,
                        1,
                        5
                    ],
                    "textureOffset": [
                        32,
                        10
                    ]
                },
                {
                    "coordinates": [
                        -4,
                        -1,
                        -2.5,
                        8,
                        1,
                        5
                    ],
                    "textureOffset": [
                        32,
                        17
                    ]
                },
                {
                    "coordinates": [
                        1,
                        -6,
                        -2.5,
                        2,
                        5,
                        1
                    ],
                    "textureOffset": [
                        41,
                        2
                    ]
                },
                {
                    "coordinates": [
```

(continues on next page)

(continued from previous page)

```
                                        1,
                                        -7,
                                        -2.5,
                                        1,
                                        1,
                                        1
                                ],
                                "textureOffset": [
                                        48,
                                        6
                                ]
                        }
                ]
        }
    ]
}
```

2.1.4 Kai Ears

This cosmetic resembles cat ears, likely to be applied to KaiAF, an OptiFine Discord moderator. It is titled `kai_ears`. It is at http://s.optifine.net/items/kai_ears/model.cfg.

Model render

Model JSON

```
{
        "type": "PlayerItem",
        "usePlayerTexture": true,
        "textureSize": [
                64,
                64
        ],
        "models": [
                {
                        "id": "ears",
                        "type": "ModelBox",
                        "attachTo": "head",
                        "invertAxis": "z",
                        "translate": [
                                0,
                                0,
                                0
                        ],
                        "submodels": [
                                {
                                        "part": "leftEar",
```

(continues on next page)

(continued from previous page)

```
                "id": "leftEar",
                "invertAxis": "xy",
                "translate": [
                        0,
                        0,
                        0
                ],
                "rotate": [
                        0,
                        -180,
                        0
                ],
                "boxes": [
                        {
                                "coordinates": [
                                        2,
                                        7.7,
                                        -2,
                                        1,
                                        2,
                                        3
                                ],
                                "textureOffset": [
                                        0,
                                        0
                                ]
                        }
                ]
        },
        {
                "part": "rightEar",
                "id": "rightEar",
                "invertAxis": "xy",
                "translate": [
                        0,
                        0,
                        0
                ],
                "rotate": [
                        0,
                        0,
                        0
                ],
                "boxes": [
                        {
                                "coordinates": [
                                        2,
                                        7.7,
                                        -1,
                                        1,
                                        2,
                                        3
```

(continues on next page)

(continued from previous page)

```
                                                        ],
                                                        "textureOffset": [
                                                                0,
                                                                0
                                                        ]
                                                }
                                        ]
                                }
                        ]
                }
        ]
}
```

2.1.5 Kai Tail

This cosmetic resembles a cats tail, likely to be applied to KaiAF, an OptiFine Discord moderator. It is titled `kai_tail`. It is at http://s.optifine.net/items/kai_tail/model.cfg.

Model render

Model JSON

```
{
        "type": "PlayerItem",
        "usePlayerTexture": true,
        "textureSize": [
                64,
                64
        ],
        "models": [
                {
                        "id": "tail",
                        "type": "ModelBox",
                        "attachTo": "body",
                        "invertAxis": "z",
                        "translate": [
                                0,
                                0,
                                0
                        ],
                        "submodels": [
                                {
                                        "part": "tail",
                                        "id": "tail",
                                        "invertAxis": "z",
                                        "translate": [
                                                0,
```

(continues on next page)

(continued from previous page)

```
                                        3.5,
                                        -7.2
                                ],
                                "rotate": [
                                        -57,
                                        0,
                                        0
                                ],
                                "boxes": [
                                        {
                                                "coordinates": [
                                                        -0.5,
                                                        -3,
                                                        -7.2,
                                                        1,
                                                        7,
                                                        1
                                                ],
                                                "textureOffset": [
                                                        60,
                                                        20
                                                ]
                                        }
                                ]
                        },
                        {
                                "part": "tail2",
                                "id": "tail2",
                                "invertAxis": "z",
                                "translate": [
                                        0,
                                        -8.8,
                                        -16.9
                                ],
                                "rotate": [
                                        -102.5,
                                        0,
                                        0
                                ],
                                "boxes": [
                                        {
                                                "coordinates": [
                                                        -0.5,
                                                        4.5,
                                                        -22,
                                                        1,
                                                        6,
                                                        1
                                                ],
                                                "textureOffset": [
                                                        60,
                                                        20
```

(continues on next page)

(continued from previous page)

```
                                                        ]
                                                }
                                        ]
                                }
                        ]
                }
        ]
}
```

2.1.6 Back Sword

This cosmetic has an old-textured Iron Sword at the player's back. It is titled `back_sword`. It is at http://s.optifine.net/items/back_sword/model.cfg.

Model render

Model JSON

```
{
        "type": "PlayerItem",
        "textureSize": [
                16,
                16
        ],
        "models": [
                {
                        "id": "Level 1",
                        "type": "ModelBox",
                        "attachTo": "body",
                        "invertAxis": "yz",
                        "translate": [
                                -7,
                                2,
                                -4
                        ],
                        "rotate": [
                                0,
                                0,
                                -90
                        ],
                        "sprites": [
                                {
                                        "comment": "Axe",
                                        "textureOffset": [
                                                0,
                                                0
                                        ],
```

(continues on next page)

(continued from previous page)

```
                                        "coordinates": [
                                                0,
                                                0,
                                                0,
                                                16,
                                                16,
                                                1
                                        ]
                                }
                        ]
                }
        ]
}
```

2.1.7 Back Pickaxe

This cosmetic has an pre-JAPPA-textured Iron Pickaxe at the player's back. It is titled `back_pickaxe`. It is at http://s.optifine.net/items/back_pickaxe/model.cfg.

Model render

Model JSON

```
{
        "type": "PlayerItem",
        "textureSize": [
                16,
                16
        ],
        "models": [
                {
                        "id": "Level 1",
                        "type": "ModelBox",
                        "attachTo": "body",
                        "invertAxis": "yz",
                        "mirrorTexture": "u",
                        "translate": [
                                -7,
                                -14,
                                -4
                        ],
                        "rotate": [
                                0,
                                0,
                                0
                        ],
                        "sprites": [
```

(continues on next page)

(continued from previous page)

```
                                {
                                        "comment": "Axe",
                                        "textureOffset": [
                                                0,
                                                0
                                        ],
                                        "coordinates": [
                                                0,
                                                0,
                                                0,
                                                16,
                                                16,
                                                1
                                        ]
                                }
                        ]
                }
        ]
}
```

2.1.8 Back Axe

This cosmetic has an pre-JAPPA-textured Iron Axe at the player's back. It is titled `back_axe`. It is at http://s.optifine.net/items/back_axe/model.cfg.

Model render

Model JSON

```
{
        "type": "PlayerItem",
        "textureSize": [
                16,
                16
        ],
        "models": [
                {
                        "id": "Level 1",
                        "type": "ModelBox",
                        "attachTo": "body",
                        "invertAxis": "yz",
                        "mirrorTexture": "uv",
                        "translate": [
                                -8,
                                1,
                                -4
                        ],
```

(continues on next page)

(continued from previous page)

```
                        "rotate": [
                                0,
                                0,
                                -90
                        ],
                        "sprites": [
                                {
                                        "comment": "Axe",
                                        "textureOffset": [
                                                0,
                                                0
                                        ],
                                        "coordinates": [
                                                0,
                                                0,
                                                0,
                                                16,
                                                16,
                                                1
                                        ]
                                }
                        ]
                }
        ]
}
```

2.1.9 Back Quiver

This cosmetic has an old-textured Quiver at the player's back. Quivers are a remnant of Minecraft, the texture being removed in 1.9 snapshot 15w31a. It is titled `back_quiver`. It is at http://s.optifine.net/items/back_quiver/model.cfg.

Model render

Model JSON

```
{
        "type": "PlayerItem",
        "textureSize": [
                16,
                16
        ],
        "models": [
                {
                        "id": "Level 1",
                        "type": "ModelBox",
                        "attachTo": "body",
                        "invertAxis": "yz",
```

(continues on next page)

(continued from previous page)

```
                        "mirrorTexture": "u",
                        "translate": [
                                -8,
                                -13,
                                -4
                        ],
                        "rotate": [
                                0,
                                0,
                                0
                        ],
                        "sprites": [
                                {
                                        "comment": "Axe",
                                        "textureOffset": [
                                                0,
                                                0
                                        ],
                                        "coordinates": [
                                                0,
                                                0,
                                                0,
                                                16,
                                                16,
                                                1
                                        ]
                                }
                        ]
                }
        ]
}
```

2.1.10 Back Bow

This cosmetic has an old-textured Bow at the player's back. It is titled `back_bow`. It is at http://s.optifine.net/items/back_quiver/model.cfg.

Model render

Model JSON


```
{
        "type": "PlayerItem",
        "textureSize": [
                16,
                16
        ],
        "models": [
                {
                        "id": "Level 1",
                        "type": "ModelBox",
                        "attachTo": "body",
                        "invertAxis": "yz",
                        "mirrorTexture": "uv",
                        "translate": [
                                -8,
                                2,
                                -4
                        ],
                        "rotate": [
                                0,
                                0,
                                -90
                        ],
                        "sprites": [
                                {
                                        "comment": "Axe",
                                        "textureOffset": [
                                                0,
                                                0
                                        ],
                                        "coordinates": [
                                                0,
                                                0,
                                                0,
                                                16,
                                                16,
                                                1
                                        ]
                                }
                        ]
                }
        ]
}
```

2.1.11 Back Carrot on a Stick

This cosmetic has an old-textured Carrot on a Stick at the player's back. It is titled `back_carrotstick`. It is at http://s.optifine.net/items/back_carrotstick/model.cfg.

Model render

Model JSON

```
{
        "type": "PlayerItem",
        "textureSize": [
                16,
                16
        ],
        "models": [
                {
                        "id": "Level 1",
                        "type": "ModelBox",
                        "attachTo": "body",
                        "invertAxis": "yz",
                        "mirrorTexture": "u",
                        "translate": [
                                -7,
                                -14,
                                -4
                        ],
                        "rotate": [
                                0,
                                0,
                                0
                        ],
                        "sprites": [
                                {
                                        "comment": "Axe",
                                        "textureOffset": [
                                                0,
                                                0
                                        ],
                                        "coordinates": [
                                                0,
                                                0,
                                                0,
                                                16,
                                                16,
                                                1
                                        ]
                                }
                        ]
                }
```

(continues on next page)

(continued from previous page)

```
        ]
}
```

2.1.12 Back Fishing Rod

This cosmetic has an old-textured Fishing Rod at the player's back. It is titled `back_fishing_rod`. It is at http://s.optifine.net/items/back_fishing_rod/model.cfg.

Model render

Model JSON

```
{
        "type": "PlayerItem",
        "textureSize": [
                16,
                16
        ],
        "models": [
                {
                        "id": "Level 1",
                        "type": "ModelBox",
                        "attachTo": "body",
                        "invertAxis": "yz",
                        "mirrorTexture": "u",
                        "translate": [
                                -7,
                                -14,
                                -4
                        ],
                        "rotate": [
                                0,
                                0,
                                0
                        ],
                        "sprites": [
                                {
                                        "comment": "Axe",
                                        "textureOffset": [
                                                0,
                                                0
                                        ],
                                        "coordinates": [
                                                0,
                                                0,
                                                0,
                                                16,
```

(continues on next page)

(continued from previous page)

```
                                        16,
                                        1
                                ]
                        }
                ]
        }
]
}
```

2.1.13 Breasts

This cosmetic is titled `body_boobs`. It is at http://s.optifine.net/items/body_boobs/model.cfg.

Model render

Model JSON

```
{
        "type": "PlayerItem",
        "usePlayerTexture": true,
        "textureSize": [
                64,
                32
        ],
        "models": [
                {
                        "id": "Level 1",
                        "type": "ModelBox",
                        "attachTo": "body",
                        "invertAxis": "z",
                        "translate": [
                                -4,
                                3,
                                1.9
                        ],
                        "rotate": [
                                0,
                                0,
                                0
                        ],
                        "sprites": [
                                {
                                        "textureOffset": [
                                                20,
                                                23
                                        ],
                                        "coordinates": [
```

(continues on next page)

(continued from previous page)

```
                                                0,
                                                0,
                                                0,
                                                8,
                                                3,
                                                1
                                        ]
                                }
                        ]
                }
        ]
}
```

2.1.14 Body Sword

This cosmetic has an old-textured Iron Sword at the player's front. It is titled `body_sword`. It is at http://s.optifine.net/items/body_sword/model.cfg.

Model render

Model JSON

```
{
        "type": "PlayerItem",
        "textureSize": [
                16,
                16
        ],
        "models": [
                {
                        "id": "Level 1",
                        "type": "ModelBox",
                        "attachTo": "body",
                        "invertAxis": "yz",
                        "translate": [
                                -1,
                                -6,
                                -14
                        ],
                        "rotate": [
                                90,
                                -45,
                                90
                        ],
```

(continues on next page)

(continued from previous page)

```
                        "sprites": [
                                {
                                        "comment": "Axe",
                                        "textureOffset": [
                                                0,
                                                0
                                        ],
                                        "coordinates": [
                                                0,
                                                0,
                                                0,
                                                16,
                                                16,
                                                1
                                        ]
                                }
                        ]
                }
        ]
}
```

2.1.15 Body Hearth

This cosmetic has a heart at the player's chest. It is titled `body_hearth`. It is at http://s.optifine.net/items/body_hearth/model.cfg.

Hearth is likely to be a typo, intended to be *heart*.

Model render

Model JSON

```
{
        "type": "PlayerItem",
        "textureSize": [
                16,
                16
        ],
        "models": [
                {
                        "id": "Main",
                        "type": "ModelBox",
                        "attachTo": "body",
                        "invertAxis": "z",
                        "translate": [
                                -3,
                                3,
```

(continues on next page)

(continued from previous page)

```
                                2.1
                        ],
                        "rotate": [
                                0,
                                0,
                                0
                        ],
                        "sprites": [
                                {
                                        "textureOffset": [
                                                0,
                                                0
                                        ],
                                        "coordinates": [
                                                0,
                                                0,
                                                0,
                                                7,
                                                6,
                                                1
                                        ]
                                }
                        ]
                }
        ]
}
```

2.1.16 Arrow Hat

This cosmetic is intended to look like an arrow through the player's head, but it is bugged. The textures are not mirrored and the model parts are not positioned correctly. It is titled `hat_arrow`. It is at http://s.optifine.net/items/hat_arrow/model.cfg.

Model render

Model JSON

```
{
        "type": "PlayerItem",
        "textureSize": [
                16,
                16
        ],
        "models": [
```

(continues on next page)

(continued from previous page)

```
                {
                        "id": "SideA",
                        "type": "ModelBox",
                        "attachTo": "head",
                        "invertAxis": "yz",
                        "translate": [
                                0,
                                5,
                                0
                        ],
                        "rotate": [
                                -45,
                                0,
                                0
                        ],
                        "scale": 0.75,
                        "boxes": [
                                {
                                        "comment": "Tail",
                                        "textureOffset": [
                                                0,
                                                5
                                        ],
                                        "coordinates": [
                                                -13,
                                                -2.5,
                                                0,
                                                8,
                                                5,
                                                0
                                        ]
                                },
                                {
                                        "comment": "Head",
                                        "textureOffset": [
                                                0,
                                                0
                                        ],
                                        "coordinates": [
                                                5,
                                                -2.5,
                                                0,
                                                8,
                                                5,
                                                0
                                        ]
                                }
                        ]
                },
                {
                        "id": "SideB",
                        "baseId": "SideA",
```

(continues on next page)

(continued from previous page)

```
                        "rotate": [
                                45,
                                0,
                                0
                        ]
                },
                {
                        "id": "Back",
                        "type": "ModelBox",
                        "attachTo": "head",
                        "invertAxis": "yz",
                        "translate": [
                                0,
                                5,
                                0
                        ],
                        "rotate": [
                                45,
                                0,
                                0
                        ],
                        "scale": 0.75,
                        "boxes": [
                                {
                                        "comment": "Back (tex coord V is 5 instead of 10!
↪)",
                                        "textureOffset": [
                                                0,
                                                5
                                        ],
                                        "coordinates": [
                                                12,
                                                -2.5,
                                                -2.5,
                                                0,
                                                5,
                                                5
                                        ]
                                }
                        ]
                }
        ]
}
```

2.1.17 Axe Hat

This cosmetic is intended to look like an axe through the player's head. It is titled `hat_axe`. It is at http://s.optifine.net/items/hat_axe/model.cfg.

Model JSON


```
{
	"type": "PlayerItem",
	"textureSize": [
		16,
		16
	],
	"models": [
		{
			"id": "Level 1",
			"type": "ModelBox",
			"attachTo": "head",
			"invertAxis": "yz",
			"mirrorTexture": "",
			"translate": [
				-1,
				-1,
				-16
			],
			"rotate": [
				90,
				0,
				90
			],
			"sprites": [
				{
					"comment": "Axe",
					"textureOffset": [
						0,
						0
					],
					"coordinates": [
						0,
						0,
						0,
						16,
						16,
						1
					]
				}
			]
		}
	]
}
```

2.1.18 Bee Antenna

This cosmetic looks like a bee antenna coming out of the east side of the player's head. It is titled `hat_bee`. It is at http://s.optifine.net/items/hat_bee/model.cfg.

Model render

Model JSON


```
{
        "type": "PlayerItem",
        "texture": "optifine:textures/features/hat_bee.png",
        "textureSize": [
                16,
                16
        ],
        "models": [
                {
                        "id": "RightAntenna",
                        "type": "ModelBox",
                        "attachTo": "head",
                        "invertAxis": "yz",
                        "translate": [
                                2,
                                8,
                                0
                        ],
                        "rotate": [
                                0,
                                0,
                                0
                        ],
                        "boxes": [
                                {
                                        "comment": "V",
                                        "textureOffset": [
                                                0,
                                                0
                                        ],
                                        "coordinates": [
                                                -1,
                                                0,
                                                0,
                                                1,
                                                4,
                                                1
                                        ]
                                },
                                {
                                        "comment": "H",
```


(continues on next page)

(continued from previous page)


```
                "textureOffset": [
                        0,
                        5
                ],
                "coordinates": [
                        0,
                        4,
                        0,
                        2,
                        1,
                        1
                ]
        },
        {
                "comment": "Dot",
                "textureOffset": [
                        4,
                        0
                ],
                "coordinates": [
                        2,
                        3,
                        0,
                        1,
                        1,
                        1
                ]
        },
        {
                "textureOffset": [
                        4,
                        0
                ],
                "coordinates": [
                        3,
                        4,
                        0,
                        1,
                        1,
                        1
                ]
        },
        {
                "textureOffset": [
                        4,
                        0
                ],
                "coordinates": [
                        4,
                        3,
                        0,
                        1,
```


(continues on next page)

(continued from previous page)

```
                                        1,
                                        1
                                ]
                        },
                        {
                                "textureOffset": [
                                        4,
                                        0
                                ],
                                "coordinates": [
                                        3,
                                        2,
                                        0,
                                        1,
                                        1,
                                        1
                                ]
                        },
                        {
                                "comment": "DotCenter",
                                "textureOffset": [
                                        4,
                                        2
                                ],
                                "coordinates": [
                                        3,
                                        3,
                                        0,
                                        1,
                                        1,
                                        1
                                ]
                        }
                ]
        },
        {
                "id": "LeftAntenna",
                "baseId": "RightAntenna",
                "invertAxis": "xyz",
                "mirrorTexture": "u"
        }
    ]
}
```

2.1.19 Jingy Hat

This cosmetic looks like a re-textured Witch Hat. It is titled `hat_jingy`. The name *Jingy* is misspelled, the name is supposed to be *Jiingy*. It is a reference to Jiingy, an OptiFine Discord admin. It is at http://s.optifine.net/items/hat_jingy/model.cfg.

Model render

Model JSON


```
{
        "type": "PlayerItem",
        "texture": "optifine:textures/features/hat_jingy.png",
        "textureSize": [
                96,
                26
        ],
        "models": [
                {
                        "part": "witch_hat",
                        "type": "ModelBox",
                        "attachTo": "head",
                        "id": "witch_hat",
                        "invertAxis": "xy",
                        "translate": [
                                -2.372,
                                -23.5,
                                -8.5
                        ],
                        "rotate": [
                                90,
                                72.5,
                                89.5
                        ],
                        "submodels": [
                                {
                                        "id": "Tip",
                                        "invertAxis": "xy",
                                        "translate": [
                                                -2.3828,
                                                40.3367,
                                                2.1
                                        ],
                                        "rotate": [
                                                0,
                                                -90,
                                                50
                                        ],
                                        "boxes": [
                                                {
```


(continues on next page)

(continued from previous page)

```
                                        "coordinates": [
                                                -1,
                                                -1.5,
                                                -1,
                                                2,
                                                3,
                                                2
                                        ],
                                        "textureOffset": [
                                                75,
                                                21
                                        ]
                                }
                        ]
                },
                {
                        "id": "Top",
                        "invertAxis": "xy",
                        "translate": [
                                -0.2993,
                                37.9964,
                                2.1
                        ],
                        "rotate": [
                                0,
                                -90,
                                37.5
                        ],
                        "boxes": [
                                {
                                        "coordinates": [
                                                -2,
                                                -2,
                                                -2,
                                                4,
                                                4,
                                                4
                                        ],
                                        "textureOffset": [
                                                58,
                                                18
                                        ]
                                }
                        ]
                },
                {
                        "id": "Middle",
                        "invertAxis": "xy",
                        "translate": [
                                0.5898,
                                35.475,
                                2.1
```

(continues on next page)

(continued from previous page)

```
                ],
                "rotate": [
                        0,
                        -90,
                        14.5
                ],
                "boxes": [
                        {
                                "coordinates": [
                                        -2.8999,
                                        -2.3873,
                                        -3,
                                        6,
                                        4,
                                        6
                                ],
                                "textureOffset": [
                                        33,
                                        16
                                ]
                        }
                ]
        },
        {
                "id": "Bottom2",
                "invertAxis": "xy",
                "translate": [
                        0.9,
                        33.05,
                        2.1
                ],
                "rotate": [
                        0,
                        -90,
                        -3
                ],
                "boxes": [
                        {
                                "coordinates": [
                                        -4,
                                        -1,
                                        -4,
                                        8,
                                        2,
                                        8
                                ],
                                "textureOffset": [
                                        0,
                                        16
                                ],
                                "sizeAdd": 0.1
                        }
```

(continues on next page)

(continued from previous page)

```
                ]
            },
            {
                "id": "Bottom1",
                "invertAxis": "xy",
                "translate": [
                    0.775,
                    31.4594,
                    2.1
                ],
                "rotate": [
                    0,
                    -90,
                    -7.5
                ],
                "boxes": [
                    {
                        "coordinates": [
                            -4.9348,
                            -1,
                            -5,
                            10,
                            2,
                            10
                        ],
                        "textureOffset": [
                            56,
                            3
                        ]
                    }
                ]
            },
            {
                "id": "Base",
                "invertAxis": "xy",
                "translate": [
                    0.775,
                    30.9636,
                    2.1
                ],
                "rotate": [
                    0,
                    0,
                    -7.5
                ],
                "boxes": [
                    {
                        "coordinates": [
                            -7.00004,
                            -0.5136,
                            -7,
                            14,
```

(continues on next page)

(continued from previous page)

```
                                                        0,
                                                        14
                                                ],
                                                "textureOffset": [
                                                        0,
                                                        0
                                                ]
                                        }
                                ]
                        }
                ]
        }
```

2.1.20 Link Hat

This cosmetic looks like a prototype of the santa hat. It is unknown what the name is a reference to. It is titled `hat_link`. It is at http://s.optifine.net/items/hat_link/model.cfg.

Model render

Model JSON

```
{
        "type": "PlayerItem",
        "textureSize": [
                64,
                64
        ],
        "models": [
                {
                        "id": "Level1",
                        "type": "ModelBox",
                        "attachTo": "head",
                        "invertAxis": "y",
                        "translate": [
                                -5,
                                7,
                                -5
                        ],
                        "rotate": [
                                0,
                                0,
```

(continues on next page)

(continued from previous page)

```
                        0
                ],
                "boxes": [
                        {
                                "textureOffset": [
                                        0,
                                        0
                                ],
                                "coordinates": [
                                        0,
                                        0,
                                        0,
                                        10,
                                        3,
                                        10
                                ],
                                "sizeAdd": 0.1
                        }
                ],
                "submodel": {
                        "comment": "Level2",
                        "invertAxis": "y",
                        "translate": [
                                1,
                                3,
                                1
                        ],
                        "rotate": [
                                -20,
                                0,
                                0
                        ],
                        "boxes": [
                                {
                                        "textureOffset": [
                                                0,
                                                13
                                        ],
                                        "coordinates": [
                                                0,
                                                0,
                                                0,
                                                8,
                                                4,
                                                8
                                        ]
                                }
                        ],
                        "submodel": {
                                "comment": "Level3",
                                "invertAxis": "y",
                                "translate": [
```

(continues on next page)

(continued from previous page)


```
                1,
                4,
                1
            ],
            "rotate": [
                -25,
                0,
                0
            ],
            "boxes": [
                {
                    "textureOffset": [
                        0,
                        25
                    ],
                    "coordinates": [
                        0,
                        0,
                        0,
                        6,
                        4,
                        6
                    ]
                }
            ],
            "submodel": {
                "comment": "Level4",
                "invertAxis": "y",
                "translate": [
                    0.5,
                    4,
                    0.5
                ],
                "rotate": [
                    -40,
                    0,
                    0
                ],
                "boxes": [
                    {
                        "textureOffset": [
                            0,
                            35
                        ],
                        "coordinates": [
                            0,
                            0,
                            0,
                            5,
                            4,
                            5
                        ]
```


(continues on next page)

(continued from previous page)

```
                                                }
                                        ],
                                        "submodel": {
                                                "comment": "Level5",
                                                "invertAxis": "y",
                                                "translate": [
                                                        0.5,
                                                        4,
                                                        0.5
                                                ],
                                                "rotate": [
                                                        -40,
                                                        0,
                                                        0
                                                ],
                                                "boxes": [
                                                        {
                                                                "textureOffset":␣
↪[
                                                                        0,
                                                                        44
                                                                ],
                                                                "coordinates": [
                                                                        0,
                                                                        0,
                                                                        0,
                                                                        4,
                                                                        4,
                                                                        4
                                                                ]
                                                        }
                                                ],
                                                "submodel": {
                                                        "comment": "Level6",
                                                        "invertAxis": "y",
                                                        "translate": [
                                                                0.5,
                                                                4,
                                                                0.5
                                                        ],
                                                        "rotate": [
                                                                -35,
                                                                0,
                                                                0
                                                        ],
                                                        "boxes": [
                                                                {

↪"textureOffset": [
                                                                                                ␣
↪0,
                                                                                                ␣
```

(continues on next page)

(continued from previous page)

```
↪52
                                                                    ],

↪"coordinates": [
                                                                        
↪0,
                                                                        
↪0,
                                                                        
↪0,
                                                                        
↪3,
                                                                        
↪5,
                                                                         3
                                                                    ]
                                                            }
                                                    ],
                                                    "submodel": {
                                                            "comment":
↪"Level7",
                                                            "invertAxis": "y
↪",
                                                            "translate": [
                                                                    0.5,
                                                                    5,
                                                                    0.5
                                                            ],
                                                            "rotate": [
                                                                    -15,
                                                                    0,
                                                                    0
                                                            ],
                                                            "boxes": [
                                                                    {

↪"textureOffset": [
↪       32,
↪       13
↪],

↪"coordinates": [
↪       0,
↪       0,
↪       0,
```

(continues on next page)

(continued from previous page)

```
↪       2,
↪       5,
↪       2
                                                                ]
                                                        }
                                                ],
                                                "submodels": [
                                                        {
↪"comment": "Level8",
↪"invertAxis": "y",
↪"translate": [
↪       0.5,
↪       5,
↪       0.5
↪],
↪"rotate": [
↪       -3,
↪       0,
↪       0
↪],
↪"boxes": [
↪       {
↪               "textureOffset": [
↪                       40,
↪                       0
↪               ],
↪               "coordinates": [
↪                       0,
↪                       0,
```

(continues on next page)

(continued from previous page)

```
                                                                                                                    ␣
↪                       0,
                                                                                                                    ␣
↪                       1,
                                                                                                                    ␣
↪                       4,
                                                                                                                    ␣
↪                       1
                                                                                                                    ␣
↪                ]
                                                                                                                    ␣
↪        }
                                                                                                                    ]
                                                                                                                }
                                                                                                            ]
                                                                                                        }
                                                                                                    }
                                                                                                }
                                                                                            }
                                                                                        }
                                                                                    }
                                                                                }
                                                                            }
                                                                        }
                                                                    }
                                                                }
                                                            }
                                                        }
                                                    }
                                                }
                                            }
                                        }
                                    }
                                }
                            }
                        }
                    }
                }
            }
        ]
}
```

2.1.21 Pickaxe Hat

This cosmetic is intended to look like a pickaxe through the player's head. It is titled `hat_pickaxe`. It is at http://s.optifine.net/items/hat_pickaxe/model.cfg.

Model JSON

```
{
        "type": "PlayerItem",
        "textureSize": [
                16,
                16
        ],
        "models": [
                {
                        "id": "Level 1",
                        "type": "ModelBox",
                        "attachTo": "head",
                        "invertAxis": "yz",
                        "translate": [
                                0,
                                1,
                                -14
                        ],
                        "rotate": [
```

(continues on next page)

(continued from previous page)

```
                                0,
                                -90,
                                0
                        ],
                        "sprites": [
                                {
                                        "comment": "Axe",
                                        "textureOffset": [
                                                0,
                                                0
                                        ],
                                        "coordinates": [
                                                0,
                                                0,
                                                0,
                                                16,
                                                16,
                                                1
                                        ]
                                }
                        ]
                }
        ]
}
```

2.1.22 Reddit Hat

This cosmetic looks very similar to `hat_bee`, except that the antenna is more centered, and the yellow dot is white. It is intended to look like Snoo's single antenna, the alien-like mascot of Reddit. It is titled `hat_reddit`. It is at http://s.optifine.net/items/hat_reddit/model.cfg.

Model render

Model JSON

```
{
        "type": "PlayerItem",
        "texture": "optifine:textures/features/hat_reddit.png",
        "textureSize": [
                16,
                16
        ],
        "models": [
                {
                        "id": "Level 1",
                        "type": "ModelBox",
                        "attachTo": "head",
```

(continues on next page)

(continued from previous page)


```
                        "invertAxis": "yz",
                        "translate": [
                                0,
                                8,
                                0
                        ],
                        "rotate": [
                                0,
                                0,
                                0
                        ],
                        "boxes": [
                                {
                                        "comment": "V",
                                        "textureOffset": [
                                                0,
                                                0
                                        ],
                                        "coordinates": [
                                                -1,
                                                0,
                                                0,
                                                1,
                                                4,
                                                1
                                        ]
                                },
                                {
                                        "comment": "H",
                                        "textureOffset": [
                                                0,
                                                5
                                        ],
                                        "coordinates": [
                                                0,
                                                4,
                                                0,
                                                2,
                                                1,
                                                1
                                        ]
                                },
                                {
                                        "comment": "Dot",
                                        "textureOffset": [
                                                4,
                                                0
                                        ],
                                        "coordinates": [
                                                2,
                                                3,
                                                0,
```


(continues on next page)

(continued from previous page)


```
                                        1,
                                        1,
                                        1
                                ]
                        },
                        {
                                "textureOffset": [
                                        4,
                                        0
                                ],
                                "coordinates": [
                                        3,
                                        4,
                                        0,
                                        1,
                                        1,
                                        1
                                ]
                        },
                        {
                                "textureOffset": [
                                        4,
                                        0
                                ],
                                "coordinates": [
                                        4,
                                        3,
                                        0,
                                        1,
                                        1,
                                        1
                                ]
                        },
                        {
                                "textureOffset": [
                                        4,
                                        0
                                ],
                                "coordinates": [
                                        3,
                                        2,
                                        0,
                                        1,
                                        1,
                                        1
                                ]
                        },
                        {
                                "comment": "DotCenter",
                                "textureOffset": [
                                        4,
                                        2
```


(continues on next page)

(continued from previous page)

```
                                        ],
                                        "coordinates": [
                                                3,
                                                3,
                                                0,
                                                1,
                                                1,
                                                1
                                        ]
                                }
                        ]
                }
        ]
}
```

2.1.23 Reindeer Antlers

This cosmetic looks like two reindeer antlers coming out of the top of the player's head. It is titled `hat_reindeer`. It is at http://s.optifine.net/items/hat_reindeer/model.cfg.

Model JSON

```
{
        "type": "PlayerItem",
        "textureSize": [
                16,
                16
        ],
        "models": [
                {
                        "id": "Level 1",
                        "type": "ModelBox",
                        "attachTo": "head",
                        "invertAxis": "yz",
                        "translate": [
                                -8,
                                6,
                                0
                        ],
                        "rotate": [
                                0,
                                0,
                                0
                        ],
                        "sprites": [
                                {
                                        "textureOffset": [
                                                0,
                                                0
                                        ],
```

(continues on next page)

(continued from previous page)

```
                                        "coordinates": [
                                                0,
                                                0,
                                                0,
                                                16,
                                                6,
                                                1
                                        ]
                                }
                        ]
                }
        ]
}
```

2.1.24 Shovel Hat

This cosmetic has an old-textured Iron Shovel in the player's head. It is titled `hat_shovel`. It is at http://s.optifine.net/items/hat_shovel/model.cfg.

Model JSON

```
{
        "type": "PlayerItem",
        "textureSize": [
                16,
                16
        ],
        "models": [
                {
                        "id": "Level 1",
                        "type": "ModelBox",
                        "attachTo": "head",
                        "invertAxis": "yz",
                        "mirrorTexture": "",
                        "translate": [
                                -1,
                                -3,
                                -15
                        ],
                        "rotate": [
                                90,
                                0,
                                90
                        ],
                        "sprites": [
                                {
                                        "comment": "Axe",
                                        "textureOffset": [
                                                0,
                                                0
```

(continues on next page)

(continued from previous page)

```
                                        ],
                                        "coordinates": [
                                                0,
                                                0,
                                                0,
                                                16,
                                                16,
                                                1
                                        ]
                                }
                        ]
                }
        ]
}
```

2.1.25 Vanilla Witch Hat

This cosmetic looks like the Vanilla's Witch's hat. It is not the same as `hat_jingy`. It is titled `hat_witch`. It is at http://s.optifine.net/items/hat_witch/model.cfg.

Model render

Model JSON

```
{
        "type": "PlayerItem",
        "texture": "optifine:textures/features/hat_witch.png",
        "textureSize": [
                64,
                64
        ],
        "models": [
                {
                        "id": "Level1",
                        "type": "ModelBox",
                        "attachTo": "head",
                        "invertAxis": "yz",
                        "translate": [
                                0,
                                7,
                                0
                        ],
                        "rotate": [
                                0,
                                0,
                                0
                        ],
```

(continues on next page)

(continued from previous page)

```
                        "boxes": [
                                {
                                        "textureOffset": [
                                                0,
                                                0
                                        ],
                                        "coordinates": [
                                                -5,
                                                0,
                                                -5,
                                                10,
                                                2,
                                                10
                                        ],
                                        "sizeAdd": 0.1
                                }
                        ]
                },
                {
                        "id": "Level2",
                        "type": "ModelBox",
                        "attachTo": "head",
                        "invertAxis": "yz",
                        "translate": [
                                0,
                                8.75,
                                0
                        ],
                        "rotate": [
                                -3,
                                0,
                                -1.5
                        ],
                        "boxes": [
                                {
                                        "textureOffset": [
                                                0,
                                                12
                                        ],
                                        "coordinates": [
                                                -3.5,
                                                0,
                                                -3.5,
                                                7,
                                                4,
                                                7
                                        ]
                                }
                        ]
                },
                {
                        "id": "Level3",
```

(continues on next page)

(continued from previous page)

```
                "type": "ModelBox",
                "attachTo": "head",
                "invertAxis": "yz",
                "translate": [
                        0,
                        12.5,
                        0
                ],
                "rotate": [
                        -9,
                        0,
                        -4.5
                ],
                "boxes": [
                        {
                                "textureOffset": [
                                        0,
                                        23
                                ],
                                "coordinates": [
                                        -2,
                                        0,
                                        -2,
                                        4,
                                        4,
                                        4
                                ]
                        }
                ]
        },
        {
                "id": "Level4",
                "type": "ModelBox",
                "attachTo": "head",
                "invertAxis": "yz",
                "translate": [
                        0,
                        16.25,
                        0
                ],
                "rotate": [
                        -21,
                        0,
                        -10.5
                ],
                "boxes": [
                        {
                                "textureOffset": [
                                        0,
                                        31
                                ],
                                "coordinates": [
```

(continues on next page)

(continued from previous page)

```
                                                -0.25,
                                                0,
                                                -1,
                                                1,
                                                2,
                                                1
                                        ],
                                        "sizeAdd": 0.25
                                }
                        ]
                }
        ]
}
```

2.1.26 Nose Up

It is unknown what this cosmetic is supposed to be. It has no texture. It is titled `head_nose_up`. It is at http://s.optifine.net/items/head_nose_up/model.cfg.

Model JSON

```
{
        "type": "PlayerItem",
        "usePlayerTexture": true,
        "textureSize": [
                64,
                32
        ],
        "models": [
                {
                        "id": "Level 1",
                        "type": "ModelBox",
                        "attachTo": "head",
                        "invertAxis": "yz",
                        "translate": [
                                -1,
                                3,
                                4
                        ],
                        "rotate": [
                                0,
                                0,
                                0
                        ],
                        "sprites": [
                                {
                                        "textureOffset": [
                                                11,
                                                13
                                        ],
```

(continues on next page)

(continued from previous page)

```
                                        "coordinates": [
                                                0,
                                                0,
                                                0,
                                                2,
                                                1,
                                                1
                                        ]
                                }
                        ]
                }
        ]
}
```

2.1.27 Nose Down

It is unknown what this cosmetic is supposed to be. It has no texture. It is titled `head_nose_down`. It is at http://s.optifine.net/items/head_nose_down/model.cfg.

Model JSON

```
{
        "type": "PlayerItem",
        "usePlayerTexture": true,
        "textureSize": [
                64,
                32
        ],
        "models": [
                {
                        "id": "Level 1",
                        "type": "ModelBox",
                        "attachTo": "head",
                        "invertAxis": "yz",
                        "translate": [
                                -1,
                                2,
                                4
                        ],
                        "rotate": [
                                0,
                                0,
                                0
                        ],
                        "sprites": [
                                {
                                        "textureOffset": [
                                                11,
                                                13
                                        ],
```

(continues on next page)

(continued from previous page)

```
                                        "coordinates": [
                                                0,
                                                0,
                                                0,
                                                2,
                                                1,
                                                1
                                        ]
                                }
                        ]
                }
        ]
}
```

2.1.28 Villager Nose

This cosmetic looks like a Villager's nose. In contrast to a Villager, the nose is at the back of the player's head, not the front. It is titled `head_nose_villager`. It is at http://s.optifine.net/items/head_nose_villager/model.cfg.

Model render

Model JSON

```
{
        "type": "PlayerItem",
        "textureSize": [
                16,
                16
        ],
        "models": [
                {
                        "id": "Level 1",
                        "type": "ModelBox",
                        "attachTo": "head",
                        "invertAxis": "yz",
                        "translate": [
                                -1,
                                -1,
                                4
                        ],
                        "rotate": [
                                0,
                                0,
                                0
                        ],
                        "boxes": [
                                {
```

(continues on next page)

(continued from previous page)

```
                                        "textureOffset": [
                                                0,
                                                0
                                        ],
                                        "coordinates": [
                                                0,
                                                0,
                                                0,
                                                2,
                                                4,
                                                2
                                        ]
                                }
                        ]
                }
        ]
}
```

2.1.29 Mouse Ears

This cosmetic looks like a mouse's ears. They are attached to the top of the player's head. It is titled `ears_mouse`. It is at http://s.optifine.net/items/ears_mouse/model.cfg.

Model render

Model JSON

```
{
        "type": "PlayerItem",
        "textureSize": [
                32,
                32
        ],
        "models": [
                {
                        "id": "Main",
                        "type": "ModelBox",
                        "attachTo": "head",
                        "invertAxis": "yz",
                        "mirrorTexture": "",
                        "translate": [
                                0,
                                0,
                                0
                        ],
                        "rotate": [
                                0,
```

(continues on next page)

(continued from previous page)

```
                                0,
                                0
                        ],
                        "boxes": [
                                {
                                        "textureOffset": [
                                                0,
                                                0
                                        ],
                                        "coordinates": [
                                                2,
                                                6,
                                                0,
                                                4,
                                                4,
                                                1
                                        ]
                                }
                        ]
                },
                {
                        "id": "Second",
                        "type": "ModelBox",
                        "baseId": "Main",
                        "invertAxis": "xyz",
                        "mirrorTexture": "u"
                }
        ]
}
```

2.1.30 Angel Wings

This cosmetic is intended to look like an angel's wings. It is titled `wings_angel`. It is at http://s.optifine.net/items/wings_angel/model.cfg.

Model JSON

```
{
        "type": "PlayerItem",
        "textureSize": [
                16,
                16
        ],
        "models": [
                {
                        "id": "Main",
                        "type": "ModelBox",
                        "attachTo": "body",
                        "invertAxis": "",
                        "translate": [
```

(continues on next page)

(continued from previous page)

```
                                1,
                                0,
                                3
                        ],
                        "rotate": [
                                0,
                                0,
                                0
                        ],
                        "sprites": [
                                {
                                        "textureOffset": [
                                                0,
                                                0
                                        ],
                                        "coordinates": [
                                                0,
                                                0,
                                                0,
                                                16,
                                                16,
                                                1
                                        ]
                                }
                        ]
                },
                {
                        "id": "Other",
                        "baseId": "Main",
                        "attachTo": "body",
                        "invertAxis": "x",
                        "mirrorTexture": "u"
                }
        ]
}
```

2.1.31 Wolverine Hand

This cosmetic looks like long "claws" coming out of the player's hand. It is titled `hand_wolverine`. It is at http://s.optifine.net/items/hand_wolverine/model.cfg.

Model JSON


```
{
        "type": "PlayerItem",
        "textureSize": [
                16,
                16
        ],
        "models": [
                {
                        "id": "LeftHand",
                        "type": "ModelBox",
                        "attachTo": "leftArm",
                        "translate": [
                                0,
                                6.5,
                                0
                        ],
                        "rotate": [
                                0,
                                0,
                                0
                        ],
                        "boxes": [
                                {
                                        "comment": "1",
                                        "textureOffset": [
                                                0,
                                                0
                                        ],
                                        "coordinates": [
                                                2.4,
                                                0,
                                                -2,
                                                1,
                                                9,
                                                1
                                        ],
                                        "sizeAdd": -0.3
                                },
                                {
                                        "comment": "2",
                                        "textureOffset": [
                                                0,
                                                0
                                        ],
                                        "coordinates": [
                                                2.4,
                                                0,
                                                -0.5,
                                                1,
                                                9,
                                                1
```


(continues on next page)

(continued from previous page)

```
                                        ],
                                        "sizeAdd": -0.3
                                },
                                {
                                        "comment": "3",
                                        "textureOffset": [
                                                0,
                                                0
                                        ],
                                        "coordinates": [
                                                2.4,
                                                0,
                                                1,
                                                1,
                                                9,
                                                1
                                        ],
                                        "sizeAdd": -0.3
                                }
                        ]
                },
                {
                        "id": "RightHand",
                        "baseId": "LeftHand",
                        "attachTo": "rightArm",
                        "invertAxis": "x"
                }
        ]
}
```

2.2 Technical details

Cosmetics are loaded alongside capes for all players. After trying to load a player's cape, OptiFine will also check a URL:

```
http://s.optifine.net/users/USERNAME.cfg
```

Where `USERNAME` is the player's **case-corrected** username. This URL must also be accessed under HTTP; HTTPS will fail.

If that player has any cosmetics, `s.optifine.net` will return a JSON file with the extension `.cfg`.

The file follows the format of:

```
{
        "items": [
                {
                        "model": "String, optional",
                        "texture": "String, optional"
                        "type": "String",
                        "active": "Boolean"
```

(continues on next page)

(continued from previous page)

```
                }, ...
        ]
}
```

Only "type" and "active" are required.

For example, see sp614x's model configuration.

OptiFine will iterate through each entry in the "items" array. If a model is not declared, OptiFine will load one based off of "type":

> Example: https://optifine.net/items/TYPE/model.cfg
> Santa Hat: https://optifine.net/items/hat_santa/model.cfg
> **Capes are exempt from this rule**

OptiFine will then load that model's texture by querying http://optifine.net/items/MODEL/users/USERNAME.png. The most reliable method of getting any model's texture is by querying `sp614x` as the USERNAME. If a texture cannot be loaded, red wool replaces it.

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

CHAPTER

THREE

DEBUG KEYS

Debug keys are a feature of Vanilla Minecraft that perform various debugging functions when a specific key combination is pressed, usually involving the `F3` button.

In addition to the Vanilla set of debug keys, OptiFine adds some of its own.

Keys	Result
`F3` + `V`	Load all visible chunks
`F3` + `O`	**Open shader options** New in version M5.
`F3` + `R`	Reload shaders

Table 1: Requires system property `chunk.debug.keys`

Keys	Result
`F3` + `E`	Show chunk path
`F3` + `L`	Smart cull
`F3` + `U`	Capture frustum
`Alt` + `F3` + `U`	Capture shadow frustum
`Shift` + `F3` + `U`	Release frustum
`F3` + `V`	Chunk visibility

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

CHAPTER

FOUR

EASTER EGGS

There are easter eggs in both the OptiFine website and in the OptiFine client. Some are only visible at certain dates, while others are still visible.

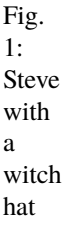

Fig. 1: Steve with a witch hat

4.1 X-Clacks-Overhead

The OptiFine website returns standard HTTP headers.

In addition to the normal headers, the website also returns: `x-clacks-overhead: GNU Terry Pratchett`. X-Clacks-Overhead is an unofficial HTTP header to memorialize Terry Pratchett.

Fig. 2: The header, shown in Firefox

4.2 sp614x's birthday

On the main menu, the splash message will say "Happy birthday, sp614x!" if the current computer date is August 14.

4.3 Cosmetics

4.3.1 Custom Player Models

Important

Please see *Custom Player Models* for more detail.

Custom Player Models is a feature that can change or add to the player model.

CPMs are like *Capes*, in that they are loaded on join and can be associated with a specific player.

Fig. 3: Available CPMs can be chosen under the Cape Change menu.

Santa Hat

Texture file	Texture resolution	Model file	Dates applied (YYYY/MM/DD)
Unknown	96 x 26	https://optifine.net/items/hat_santa/model.cfg	2020/12/25 - 2020/12/31

On December 25th, 2020, a Santa hat was added to all donators' heads. It was available until the end of year.

The same name (`hat_santa`) was reused for the Witch Hat, but the model and texture are different.

Fig. 4: How the santa hat looks on a player

Witch Hat

Texture file	Texture resolution	Model file	Dates applied (YYYY/MM/DD)	Notes
	96px x 26px	https://optifine.net/items/hat_santa/model.cfg	2021/10/30 - 2021/11/07	
	64px x 64px	https://optifine.net/items/hat_witch/model.cfg		Actual witch's hat, unused.

Fig. 5: How the hat looks on a player

The witch hat is a gray hat applied to all donators automatically.

Its name is actually reused from `hat_santa` (Santa Hat). The original Witch's Hat has an unknown use, but it is applied to sp614x.

It was first applied on October 30, 2021, and lasted until November 7, 2021; ~1 week.

4.3.2 Capes

Some special capes or event-related capes may be given to specific people or at specific times.

"10" Cape

See *Anniversary capes*.

Secret project capes

See *Unknown project capes*.

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

CHAPTER

FIVE

FAQ

Warning

This page is very long and may be uncomfortable to navigate.

5.1 OptiFine

5.1.1 Which OptiFine edition should I get?

Note

This only applies to 1.7.10 and below.

OptiFine Ultra is the most popular edition.

- OptiFine Ultra has the most optimizations and features that can increase FPS.
- OptiFine Standard is more compatible with other mods.
- OptiFine Light may be more helpful for underpowered laptops and notebooks, but it does not have any advanced features and is not compatible with ModLoader and Forge.

5.1.2 Do I have to configure OptiFine to get better FPS?

Usually no. OptiFine comes with many optimizations that should increase the FPS without any configuration. You can further increase the FPS by configuring the Video Settings and finding the trade-off between quality and performance that works best for you.

5.1.3 How can I configure OptiFine for best FPS?

Go to the Video Settings menu and look at the tooltip shown for every setting. It should show which value to set for better FPS and which for better quality.

5.1.4 Why are some world chunks flickering with OptiFine?

Most probably you have enabled "Chunk Loading: Multi-Core", but the graphics card driver does not support it properly. To fix it go to the graphics card control panel and set "Threaded Optimization"/"OpenGL Threading" to **OFF**. For best results disable it globally, not only for `java.exe` or `minecraft.exe`.

Fig. 1: An example of how to disable threaded optimization for NVIDIA cards.

5.1.5 Why am I getting the same or less FPS with OptiFine installed?

OptiFine generally increases the FPS *(200%+ is common)* and in the worst case it should be the same as Vanilla Minecraft. There are many reasons why in some cases the FPS with OptiFine may be the same or lower than Vanilla:

- You might have enabled some higher quality settings that can decrease the FPS (for example: Render Distance Extreme, Antialiasing, Anisotropic Filtering). To reset the Video Settings to their

default values go to *Video Settings* → *Other* → *Reset Video Settings*. * You might have enabled the FPS limiter in Video Settings * If you are using a HD texture pack, it most probably comes with some quality features that are not recognized by vanilla, but are active with OptiFine and can decrease the FPS. For example: Connected Textures, Custom Sky, Random Mobs, Custom Animations, etc. You can turn these OFF in the Video Settings if you do not need them. * OptiFine may not be installed properly. Make sure to install OptiFine last, so that no other mod is overwriting the OptiFine classes. Exceptions are mods which are specially designed to be installed after OptiFine. * There may be a mod conflict and Minecraft may be reporting a lot of errors in the error log. Check the Minecraft error log to see if this is the case (Magic Launcher can show the error log).

To test properly make sure that you are using identical test conditions; same texture pack, same render distance, same world, etc.

Even if the average FPS reported in the debug screen is the same as vanilla, the gameplay with OptiFine should feel smoother and more fluid. There are several OptiFine options (Chunk Loading, Smooth FPS, Smooth World, etc) that can further reduce the lag spikes and stabilize the framerate.

5.1.6 Can I include OptiFine in my modpack?

Generally no, unless you have an explicit permission from us.

5.1.7 How do I open the output log?

In your launcher settings, go to the `General` tab and tick the `Open the output log when Minecraft: Java Edition starts` button

Fig. 2: A tutorial with the steps circled in red

5.1.8 Why do I get a warning message when launching OptiFine?

A recent update was made to the vanilla Minecraft launcher, which now displays a warning message whenever the user tries launching a modified version of the game. This applies to **all** modified clients, including OptiFine, Forge, and Fabric.

This warning can be safely ignored.

5.1.9 I can't find OptiFine in my launcher!

Mojang updated the Minecraft launcher and added a toggle to show or hide modded versions of the game. Go to the "Installations" tab and make sure the Modded box is ticked.

Fig. 3: A red arrow pointing to the "Modded" checkbox

5.1.10 I have a powerful graphics card, but Minecraft keeps using integrated graphics.

Note

These instructions only work for modern versions of Windows 10.

1. Open the Settings app.
2. Navigate to `System` > `Display`, and scroll down until you see "Graphics Settings".
3. **Select "Browse", then locate and add `javaw.exe`.**
    - The location of this file will vary, but you can usually find it at `C:\Program Files (x86)\Minecraft Launcher\runtime\jre-x64\bin`.
4. Select "Java(TM) Platform SE Binary", and then select "Options".
5. Set the graphics preference to "High Performance", and then save.

5.1.11 Why isn't OptiFine open-source?

The core of OptiFine consists of many, various changes to Minecraft's rendering code. Rather than simple patches, these are significant reorganizations.

This means publishing the full source code of OptiFine would be a direct violation of Minecraft's EULA. Technically, it would be possible to extract the actual changes as patches, which can then be published to GitHub as source code.

However, there are a number of issues with this idea. First, OptiFine is built on a custom version of Mod Coder Pack. This non-standard version of MCP is used to allow OptiFine to start development on new versions of the game much, much earlier. During this time, official MCP mappings are either completely missing, or otherwise very unstable.

Unfortunately, according to the MCP license and Terms of Usage, modified versions of MCP scripts are not allowed to be distributed. This means that, even if OptiFine patches were released, nobody else would be able to collaborate on the code, defeating the entire purpose of going open-source. Even if somehow all of that was solved, this would still mean significantly changing how OptiFine development is handled. sp614x does not work on the mod using patches, which means he would have to either merge the patches manually, or completely change his entire workflow to use patches.

Changing the development process like this would not be a trivial task in the slightest.

In summary, due to a multitude of legal and technical complications, sp614x cannot make OptiFine open-source.

5.1.12 What happens if the developer disappears, or just gives up and quits?

Java programs are not difficult to decompile.

If sp614x ever went missing, virtually anyone with the right knowledge could decompile OptiFine in its entirety, compare it to decompiled vanilla Minecraft code, and extract the patches.

Alternatively, if sp614x ever decides to quit, he is willing to publish OptiFine's patches to GitHub. With that said, there are currently no plans to stop OptiFine development.

Either way, OptiFine can live again.

5.1.13 Where is OptiFine for Minecraft Bedrock Edition?

Danger

Anything claiming to be OptiFine for Bedrock, is completely unofficial, and most likely a scam or malware

---

The Bedrock Edition of Minecraft is not at all supported by OptiFine.

It uses an entirely different engine, and bears little resemblance to Java Edition.

5.1.14 What do the two numbers for FPS mean?

Average/Minimum. Minimum reflects the perceived framerate.

5.2 Donator Cape

5.2.1 How can I change the design of my cape?

You can login at the cape change page and select a new design. The cape change page has a timeout of 2 minutes and the cape has to be changed in this time.

5.2.2 How long does it take for the cape to get activated?

The cape is activated automatically when the donation payment is complete. Please note that some payment types (bank transfer, eCheck, etc) may take several days to complete. A notification email is sent to the donation email address when the cape is activated.

5.2.3 How can I check if the cape is active?

You can login at the cape change page. It will show the current cape design or a notification if the cape is not active. The cape change page can automatically correct upper/lowercase errors in the username.

5.2.4 I donated, but I did not receive a confirmation email and I do not see the cape. What can I do?

Most probably your donation was received without a username. You can login on the donator login page and assign a username for the cape. I got the confirmation email, but I do not see the cape on my player, what can I do? Please check that:

- The username is correctly spelled, **upper/lowercase matters**
- OptiFine is correctly installed (installation instructions)
- The option *Video Settings* → *Details* → *Show Capes* is **ON**
- The option *Skin Customization* → *Cape* is **ON**
- No firewall or router is blocking OptiFine from accessing the OptiFine server where the cape is located
- The cape is visible in single-player worlds. Some servers may modify the player name by adding tags or truncating it, which changes the player identity and the cape may not be displayed.

5.2.5 I gave the wrong username when donating, what can I do?

If the username capitalization is not correct (upper/lowercase wrong), then login on the cape change page and the capitalization should be automatically fixed.

If the username is still not correct, you can login on the donator login page where you can assign a new username.

5.2.6 How can I temporarily deactivate my cape?

You can deactivate and reactivate the cape on the cape change page.

5.2.7 How can I move my cape to another account?

You can move the cape to another username on the cape change page. After the cape is moved to a new username it can **only be modified from the new account!**

If you move the cape to a wrong username, you can recover it on the donator login page.

5.2.8 My cape was active, but it is now missing. How can I recover it?

You can recover the cape on the donator login page where you can assign a new username for the cape. The cape was most probably moved by someone who knows your Minecraft username (e-mail) and password. You should change your minecraft password to avoid the cape being moved again.

5.2.9 I can't access the Cape servers/Mojang servers!

There are many reasons why you could be blocked by the Cape servers or Mojang servers.

One of the reasons could be you have an antivirus installed that may be blocking the cape servers. You can close the antivirus to see if that helps.

Another one could be your ISP is blocking the cape servers. If this is the case, there is nothing we can do to fix this.

Or, you may have installed an app that is purposely blocking the cape servers. One of these apps is called **Mantle**. If you have this installed, open it, then press the unload button. If this doesn't help, uninstall Mantle. Press the Windows key, go to "Add or remove programs", and search for "Mantle". Then, press uninstall on "Mantle" or "Mantle Uninstaller". This should remove everything. Then restart your computer.

In some cases, you may have to manually remove a certain line in the host file. Here are the instructions:

1. Open Notepad with Administrator *(Search "Notepad" in your Windows Search Bar, right click on it and press Run as Administrator)*, accept the prompt that comes up.
2. Press `File |->| Open`
3. In the bottom right corner of the file window, change Text Documents (.txt) to All Files (.*)
4. Navigate to `C:\Windows\System32\drivers\etc`
5. Double click on `hosts`
6. Delete the **entire line** that contains `s.optifine.net` or **anything** that contains `mojang`.
7. Press `Control + S` to save.
8. Close Notepad and restart your computer.

5.2.10 Why can't I use the Mojang pattern for my banner cape?

The Mojang pattern *(referred in the game as "Thing")* is specifically disallowed to prevent impersonation of Mojang employees.

Fig. 4: The culprit; the resemblance is uncanny

5.2.11 Why do I get the "Invalid cape design" error?

If the banner has more than 8 layers, that's why. There is a maximum of 8 patterns, not including the base color.

Your cape must also **not include the** Mojang logo pattern.

5.2.12 Why can't donators have custom capes?

There are a couple of reasons why this isn't allowed. Namely, impersonation of Minecon Capes, Mojang staff Capes, or any other official Minecraft cape.

Moderation of NSFW and or offensive Images also becomes an issue.

Users who do have a custom (non-banner) cape were only given for very specific reasons.

Please do not ask for a custom cape; the creator of OptiFine, sp614x, has stated that there will be no more custom image capes.

5.2.13 Why does my OptiFine cape not show on PvP clients?

Some PvP clients block OptiFine capes from showing, to promote buying their own cosmetics.

In general, OptiFine doesn't provide support for PvP clients, as most of them illegally redistribute OptiFine without permission.

OptiFine highly recommends using either standalone OptiFine or Forge with individual mods.

5.2.14 My OptiFine cape was stolen!

Important

Administrators **can not** and **will not** move capes for you

---

Login at https://optifine.net/login, and simply update the username for the cape. Make sure it is also marked as **locked**. If you cannot move the cape, just lock it for now and wait 24 hours before trying to update the username.

If you're not the person who made the donation for the cape, you will have to ask the original donator to perform these actions. If your friend gave you the cape, ask them.

In a majority of cases, the following steps are probably not be necessary. However, if you want to be absolutely sure you're safe, this is the way to go:

1. Scan your computer, and change your passwords **immediately**. This should include your email account, your Minecraft/Mojang account, and your OptiFine.net account. Ideally, it should also be in that exact order. All of this is to ensure no strangers have access to any of these relevant accounts.
2. After the passwords have been updated, log in at https://optifine.net/login and follow the same steps above.

5.2.15 I moved my cape to another account, but it just disappeared!

All OptiFine capes are immediately disabled when transferred. This change was made recently to prevent abuse.

The cape can be activated again by the new cape owner by opening the game and navigating to `Options > Skin Customization > OptiFine Cape > Open Cape Editor`. From here, the cape can be activated once again.

5.2.16 Why isn't my cape showing?

There are several reasons that your donator cape may not display in-game. First, check that it's enabled in settings, at `Skin Customization... > Cape: ON`, and `Video Settings... > Details... > Show Capes: ON`. If it does not show then, check that the cape is actually activated **and** linked to your Minecraft username. You can do this by logging in at https://optifine.net/login, or opening the cape editor from in-game at `Skin Customization... > OptiFine Cape... > Open Cape Editor`.

If it seems that your cape is no longer linked to your username, see *this*.

Finally, if you still cannot see the cape at this point, there is a very slim chance that your internet service provider may be caching certain things *"for your convenience"*.

There have been a small number of cases where this was the issue, and the only solution was to temporarily switch to a mobile network. The cape should reappear on the normal connection within several days.

5.2.17 Will I have to update my cape if I change my Minecraft username?

No.

OptiFine will update your username automatically, but it may take up to 24 hours before this change goes into effect.

5.2.18 Can I change my donator e-mail?

Currently, **no**.

It is not known if this functionality will be added in the future.

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

CHAPTER

SIX

INSTALLATION

This page outlines the steps to install OptiFine for any version on any operating system.

6.1 Pre-requirements

Before installing OptiFine, you may need to install some things first to ensure the installer runs properly.

6.1.1 Java runtime

You may need to install a Java runtime (JRE) above version 8. It is not recommended to use Oracle's Java. Instead, use Adoptium's *(formerly AdoptOpenJDK)* Java runtime.

You can choose to install any JRE; all work with the OptiFine installer.

Windows

1. Go to https://adoptium.net/temurin/releases/
2. Under the *Operating System* filter box, select **Windows**.
3. Under the *Package Type* filter box, select **JRE**. You do not need the JDK.
4. Under the *Architecture* filter box, select **x64**.
5. Under the *Version* filter box, select any number; they are all >= 8.
6. Click the blue button labeled `.msi`.
7. Execute the downloaded MSI file.

Linux

1. Go to https://adoptium.net/temurin/releases/
2. Under the *Operating System* filter box, select **Linux**.
3. Under the *Package Type* filter box, select **JRE**.
4. Under the *Architecture* filter box, select **x64**.
5. Under the *Version* filter box, select any number; they are all >= 8.
6. Click the blue button labeled `.tar.gz`.
7. Extract the files inside the archive into your PATH; you may need to use `sudo` for this.

Mac

1. Go to https://adoptium.net/temurin/releases/
2. Under the *Operating System* filter box, select **macOS**.
3. Under the *Package Type* filter box, select **JRE**.
4. Under the *Architecture* filter box, select **x64**.
5. Under the *Version* filter box, select any number; they are all >= 8.
6. Click the blue button labeled `.pkg`.
7. Execute the downloaded PKG file.

6.1.2 Jarfix

If you have the Java runtime installed, but can't open `.jar` files with it, you may need to install Jarfix. Jarfix is an application that fixes the file association on Windows, so that JAR files will be executed as Java programs.

Fig. 1: A jarfox

Download it from https://johann.loefflmann.net/en/software/jarfix/index.html and run it.

Important

This is only for Windows. This issue does not occur in Linux or Mac.

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

6.2 Downloading

To download OptiFine, first determine what version you will want to download OptiFine for.

Visit https://optifine.net/downloads, and find the heading that has your version.

Note

The latest minecraft version will always be displayed. For older versions, click "Show all versions" at the bottom of the page.

Only download the most recent OptiFine version for your Minecraft version. Click the large blue "Download" button, or the "(mirror)" link.

If you want an older OptiFine version, click the "+ More" link under the latest OptiFine version for that Minecraft version. If you want a preview version of OptiFine, click the "+ Preview versions" link.

Important

If you are intending to use OptiFine with Forge, **ensure** that OptiFine version is compatible with your Forge version.

Fig. 2: Ensure this version is at least your Forge version!

You should now be downloading a `.jar` file.

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

6.3 Install With Vanilla Launcher

Important

To install OptiFine for a version, that same Vanilla version **must be installed** *and* ran beforehand. If the version is not installed, **or** if it has not been played before, OptiFine will not install.

1. Follow the instructions at *Downloading*.
2. Run the downloaded `.jar` file.
3. Follow the instructions and click "Install".
4. Open the Minecraft launcher and select the OptiFine profile, and click "Play".

Fig. 3: The OptiFine installer

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

6.4 Install With Forge

Note

This tutorial does not explain *how* to install Forge.

1. Download and install Minecraft Forge for the version you want
2. Follow the instructions at *Downloading*.
3. Open your Downloads folder in a file manager.
4. From here on, process depends on your OS:

Note

These steps are for the Vanilla Launcher. For MultiMC, find the mods folder for your specific Forge instance.

Windows

1. Press `Windows` + `R`, and inside of the Run window, type `%AppData%\.minecraft\`.

Mac

1. Open the Spotlight search by either:
   Clicking the Spotlight icon in the menu bar.
   Pressing `Command` + `Space`.
2. In the Spotlight window, type `~/library/Application Support/minecraft/`.

Linux

1. Open a file manager.
2. Change the path to `~/.minecraft/`.
5. If there is no "mods" folder in `.minecraft`, create one *(all lowercase; case-sensitive)*.
6. Drag the OptiFine jar from the Downloads folder into the "mods" folder.
7. Launch the game with Forge.

Important

If you are intending to use OptiFine with Forge, **ensure** that OptiFine version is compatible with your Forge version.

Fig. 4: Ensure this version is at least your Forge version!

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

6.5 Install With MultiMC

Note

This applies to derivatives of MultiMC, like PolyMC or Prism.

Read https://github.com/MultiMC/Launcher/wiki/MultiMC-and-OptiFine#as-a-json-patch.

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

If after any of these steps you run into a problem, see *Troubleshooting*.

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

CHAPTER

SEVEN

JVM ARGUMENTS

OptiFine supports running the game with arguments, some of which are not available in the options menu.

The system properties have to be added in the field "JVM Arguments" in the launcher profile. All arguments must be prefixed with `-D`.

Fig. 1: A lot of

Fig. 2: The profile's "edit" screen, with the **JVM ARGUMENTS** field selected.

Fig. 3: A list of the installations, with the "edit" drop-down option hovered.

Argument	Values	Meaning
`log.detail`	Boolean	Enables extended logging.
`saveTextureMap`	Boolean	Save the final texture map/atlas in the folder `debug/`.
`shaders.debug.save`	Boolean	Save the final shader sources in the folder `shaderpacks/debug/`.
`animate.model.living`	Boolean	Automatically animate all mob models. Useful for testing *Custom Entity Models*.
`player.models.local`	Boolean	Load the player models from the folder `playermodels/`.
`frame.time`	Boolean	**Show frame time in milliseconds instead of FPS.** New in version G6.
`gl.debug.groups`	Boolean	Enable OpenGL debug groups.
`gl.ignore.errors`	*List* of integers	Ignore OpenGL errors based on the comma seperated list of error IDs.
`cem.debug.models`	Boolean	CEM debug models. Automatically generate CEM models for all supported entities using different colors for each model part The part names and colors are written in the log.
`chunk.debug.keys`	Boolean	**Enables chunk debug keys, see *Debug Keys*.** New in version H3.

For example, to enable extended logging, add `-Dlog.detail=true` to the JVM arguments.

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

CHAPTER

EIGHT

TROUBLESHOOTING

You may run into a problem when installing or using OptiFine. Below listed are the most common problems and solutions. This is a non-exhaustive list; it does not contain every conceivable issue.

To get general support after confirming your solution is not here, join the Discord.

8.1 Downloading

8.1.1 Did not get a .JAR file

Danger

If you did not get a .JAR file, **do not** run it.

This occurs because you did not click the correct link when following the instructions on *Downloading*. Ensure that you skip ads, or click the "(mirror)" link.

8.2 Installing

8.2.1 FileNotFoundException (Access Denied)

Go to the file location `C:\Users\<your username here>\AppData\Roaming\.minecraft\libraries\optifine\OptiFine` and delete the folder corresponding to the OptiFine version you are trying to install.

8.2.2 There are errors in the following switches

This occurs because you did not follow the instructions in *Pre-requirements*. Scroll to the Jarfix section and follow the directions.

8.2.3 Cannot find Minecraft version

This occurs because you did not follow the instructions in *Install With Vanilla Launcher*. In the Vanilla Launcher, create a new profile (installation) with the requested Vanilla version. Run it, and then close it. Re-run OptiFine.

8.2.4 ZipException: error in opening zip file

Copy the below code and paste it into a file ending in `.bat`. Run it.

```
<# :

@echo off
echo Select the OptiFine.jar file to install
setlocal

for /f "delims=" %%I in ('powershell -noprofile "iex (${%~f0} | out-string)"') do (
java -jar %%~I
)
goto :EOF

: #>

Add-Type -AssemblyName System.Windows.Forms
$f = new-object Windows.Forms.OpenFileDialog
$f.InitialDirectory = pwd
$f.Filter = "JAR File (*.jar)|*.jar"
$f.ShowHelp = $true
$f.Multiselect = $true
[void]$f.ShowDialog()
if ($f.Multiselect) { $f.FileNames } else { $f.FileName }
```

Or, download the script `here`.

8.2.5 Could not find the main class

You did not follow the instructions in *Pre-requirements*. You need to install a Java runtime.

8.3 Launching

8.3.1 modName:tomatoGuy

This occurs because you have an old version of Complementary shaders. Either delete the shaderpack or update it.

8.3.2 Could not create the Java Virtual Machine

This occurs because you did not follow the instructions in *Pre-requirements*. You need to install a Java runtime, as well as Jarfix.

8.4 Using

8.4.1 Purple and black checkerboard textures

This happens because a required texture did not load. Normally, two things cause this:

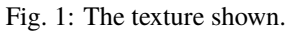

Fig. 1: The texture shown.

- Invalid path (check *File naming rules*).
- Missing file

Check your `latest.log` for more specifics on which textures are failing to load, and why.

Ensure all of your `.properties` files point to valid texture paths.

8.4.2 Warped shaders

Fig. 2: Half of all textures appear warped.

This occurs because of a known bug. Enabling and disabling shaders with Forge requires a restart.

If running on Forge, do not swap shaders while in a world.

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

CHAPTER

NINE

VERSIONING

OptiFine's versioning scheme follows most of Semantic Versioning.

- Letter: major version (A-Z)
- Number: minor version (1-9)
- `preX`: preview X (>=1)

Feature sets and Minecraft version changes warrant a major version increment.

Bugfixes and small changes warrant a minor version increment.

Preview versions can increment arbitrarily in order, and their feature set is mutable.

Minecraft versions are generally independent of the major version and are included inthe version identifier because multiple of the same major version may be ported to more than 1 Minecraft version.

Fig. 1: E3, D8, D7, B3, B2, B1

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

CHAPTER

TEN

ABOUT

This is the about page for the OptiFine **documentation**.

10.1 Is this official?

No.

This documentation itself is not official, but the source of most of it was created by sp614x and can be found at the GitHub.

The ReadTheDocs documentation is maintained by WhoAteMyButter, and sp614x does not edit it directly; hosted documentation is updated to the GitHub documentation later.

10.2 Is this legitimate?

Yes.

The *original* documentation here can be found at the GitHub. This project aims to make the source human-readable while also expanding on it with better explanations, details, examples, and more.

10.3 When was this made?

1. Check *the changelog* for the initial release date and for any further changes.
2. Go to the source and check the commit history.

10.4 I found a bug!

Quote

Use the source, Luke!

Head over to *Contributing* and see what you can do!

10.5 What's the license?

The Optifine Documentation both here and at the official GitHub repository are public domain. This documentation is explicitly licensed under CC0.

This means you are free to reproduce it, modify it, sell it, print it, translate it, among every other freedom (as in libre).

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

CHAPTER

ELEVEN

CHANGELOG

Important

This is the changelog for the ReadTheDocs documentation, **not** OptiFine. This changelog follows Keep A Changelog 1.1.0.

11.1 2023 September 24

11.1.1 Changed

- Changed all Minecraft Fandom links to the new Minecraft Wiki. (closes #22)

11.1.2 Fixed

- Hyphen typo in *Syntax*. (closes #19)

11.2 2023 September 13

11.2.1 Fixed

- `rot_y` and `rot_x` mistakes in *Entity parameters*.

11.3 2023 August 15

11.3.1 Fixed

- OpenGraph meta tags not showing.
- Small fixes and tab/space inconsistency issues.
- Tutorial links in *Shaders - Development*.
- Mistake in *Wildcards*.

11.4 2023 August 14

11.4.1 Added

- Updated to commit 8ed2130d.
- sp614x's birthday section in *Easter Eggs*.
- More examples to *Models*.

11.5 2023 August 6

11.5.1 Added

- Better *Style guide*.

11.5.2 Changed

- Rearranged section order in *Capes*.
- Renamed *Unknown project capes*

11.5.3 Fixed

- Mistakes and inaccuracies in *Special capes*.
- Dead links to *Custom Player Models*.

11.6 2023 August 4

11.6.1 Added

- **Reproducible builds.**
  - `.readthedocs.yaml` file.
  - Pinned requirements.
- A *contributor's guide*.
- More renders to *Custom Player Models*.
- Added more special capes to *Special capes*.
- More *Glossary* terms.

11.6.2 Changed

- Renamed Special Cosmetics to *Custom Player Models*.
- *Special capes* have been reorganized.
- *Easter Eggs* has been reorganized.

11.6.3 Fixed

- Tense and capitalization fixes.

11.6.4 Removed

- Top banner announcement.

11.7 2023 July 2

11.7.1 Changed

- Changed wording of overlay note in *Connected Textures*.

11.8 2023 June 26

11.8.1 Added

- Armor trims to *Armor trims*.

11.8.2 Changed

- Updated to commit 61a0581a.

11.8.3 Fixed

- Incorrect path syntax in *Syntax*.
- Broken cross-heading references.

11.9 2023 June 10

11.9.1 Added

- Re-added the accidentally removed "Complete file list" section from *Custom Colors*.

11.9.2 Fixed

- *Capes* having incorrect Technical details wording.

11.9.3 Removed

- Hovering tooltips, they don't work anymore.

11.10 2023 June 6

11.10.1 Added

- Checkboxes.
- Useful links at footer.
- Licensing terms to *About*.
- Most pages now have a "hero" subtitle (text at the header).
- *Shaders* page.
- All pages now have attached a JSON schema, if applicable.
- *Parts* page.
- *JSON Schemas* page.
- Social media cards (for embed links).
- More *troubleshooting problems*.
- More "version added" notes.

11.10.2 Fixed

- Link errors in *About*.
- Link errors in *Custom Player Models*.
- Typos in *Limitations*.
- Inconsistent articles and grammar in many pages.
- Bad menuselection syntax.

11.10.3 Changed

- Documentation theme is now Sphinx Immaterial.
- **Changelog future entries now follow Keep A Changelog 1.1.0.**
  - Deviation from standard in date-keeping; this is intentional.
- **"Table-style" properties have been overhauled and replaced with a better "header" system.**
  - This allows for better linking to specific sections.
  - Sections can contain better examples and some may have pictures *in the future*.
- Titles in Installation documents.
- Link at *Install With MultiMC*.
- CEM documents are now at the top-level (no subfolders).
- Updated **many** pages.
- Heavily overhauled *Custom Entity Models*.
- Location admonitions now use the glob format (* and **).
- Updated to commit 22f0481b.
- Changed code tab size to 2 (from 4).
- *JVM Arguments* is now under Information.
- *Custom Player Models* is now under Information.

11.10.4 Removed

- Tables for resource pack feature properties.
- Badges at top of index page; replaced by footer links on all pages.
- Smartquotes.

11.11 2023 April 12

- **Documentation is now open source, can be found at https://gitlab.com/whoatemybutter/optifinedocs**
  - Split into two branches, `master` and `dev`
  - `master` is the hosted version
  - `dev` contains the future documentation updates

11.12 2023 February 9

- Updated *Custom Sky*
- Added download button styling

11.13 2023 February 5

- Converted all images to lossless WebP (support)
- Reorganized table of contents
- Added raw:, range:, exists: to *Syntax*
- Overhauled installation instructions, see *Installation*
- Re=did changelog page, date format is now (year) (month) (day)
- Removed Broken Features
- Added *Troubleshooting*
- Added more information about Banner Capes to *Capes*

11.14 2022 December 8

- Overhauled Properties Files page, it is now *Syntax*
- Updated to commit b98f576e
- Changed dark theme colors
- Added *Glossary*

11.15 2022 August 27

- Added Broken Features
- Updated *Installation*
- Updated to commit 8410499f

11.16 2022 July 21

- Removed note about old versions; it is now expected for you to run an updated game
- Added *Special Cosmetics*
- Updated *CEM animations*
- Updated *CEM entity names*
- Updated to commit 8174f510

11.17 2022 April 6

- Replaced Values, Required, Default table options with one unified list under Values
- Updated theme, adds a *Back To Top* button when you begin to scroll up, and adds many other small improvements
- Fixed wrong name for option of Banner capes in *capes*
- Fixed egg naming error in *custom colors*
- Updated *CEM animations*
- Updated *CEM entity names*
- Updated to commit b07063a4

11.18 2021 December 11

- Added more *debug keys*
- Fixed various typos
- Adjust sidebar CSS

11.19 2021 November 20

- Expanded on *Paths*
- Added tutorials to *Custom Entity Models*
- Changed how the keyboard keys look
- Added *Installation*
- Added more examples to *NBT matching*
- Fixed *Spawner egg colors*
- Fixed color on tooltip hovers

11.20 2021 November 2

- Added the "10" anniversary capes to *Capes*
- Moved Extra Cosmetics to *Easter Eggs*
- Added more to *Easter Eggs*

11.21 2021 October 31

- Updated styling of notice at footer
- Added *extra_cosmetics*
- Updated *Capes*

11.22 2021 September 20

- Updated styling so TOC drawer doesn't overflow
- Removed copyright notice at footer
- Added *Versioning*
- Changed tab style

11.23 2021 August 30

- Updated to commit 431a82106b987229b2ab9f332dad20b560e67a5e

11.24 2021 August 29

- Updated to commit aedb5d527903882385b6953b887d0668860f447c
- Updated to commit 9b8f72b07f1f3229584ffa404c7f86f6e6bcf459

11.25 2021 August 22

- Updated *CIT*
- Compressed settings images
- Added styling to collapsible content
- Added more examples
- Changed some inline code to use GUI labels instead

11.26 2021 August 17

- Split CEM into section files
- Split shaders into section files
- Add documentation changelog
- Change spaces to tabs
- Compress *HD fonts icon*
- Add icons for most shader files

11.27 2021 July 14

- Move shaders
- Add previewable links
- Expand *Capes*

11.28 2021 July 7

- Add *Lagometer*
- Add *Capes*
- Fix CSS
- Fix content overflowing on edges
- Add *CEM* is_ridden entity parameter

11.29 2021 July 3

- Remove line numbers in code blocks
- Add better CSS
- More document icons
- More links & shortcuts
- Add internal template
- Use smaller image for icon and crisp-edge
- Fix Custom Animations glowsquid video

11.30 2021 July 2

- Update wording
- Add tabs
- Expand and clarify sections
- Add file-location admonitions
- Add keyword shortcuts
- Add CEM limitations
- Remove first-person centric language

11.31 2021 June 13

- Manually fix OpenGraph meta extension
- Merge fixes for absolutes in embeds
- Fix description headline

11.32 2021 June 12

- Add icons to each document
- Add metadata for embeds
- Add lead text to embeds
- Add theme color to embeds

11.33 2021 May 31

- Update to commit dbb3016
- Add *Random Entities*
- *FAQ* updates

11.34 2021 May 27

- Initial commit

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

CHAPTER

TWELVE

CONTRIBUTING

OptiDocs always welcomes contributions and changes. Whether you want to add a new page, add more to a section, or change an image, your contributions are invited; they are always welcome.

12.1 Things to know

12.1.1 rST

OptiDocs' source documents are written in reStructuredText (rST). rST is a markup system originally used in Python but has since been expanded to general use. In order to write or contribute to OptiDocs, you **must** be familiar with rST. rST is not Markdown and Markdown is not rST. You should start here; only use quick "cheatsheets" once you are comfortable with the syntax and caveats.

12.1.2 Sphinx

Sphinx is used for generating the documentation. Sphinx is widely used, from the Linux kernel to small project documentation. You should not need to be highly knowledgeable of Sphinx, but it is important to be aware of what it is and what it offers you. Sphinx is not rST. Sphinx takes in rST documents to generate cohesive and interconnected documentation.

Sphinx-Immaterial (pypi, github, kitchen sink) is the theme used for OptiDocs. This theme includes some unique fetures that OptiDocs does take advantage of, so you should learn the theme-specific options. Specifically, the `.. md-tab-set` and custom checkboxes styles.

12.1.3 git

OptiDocs uses git as the version control system. You must be familiar with how git works. You should be familiar with how merge requests (also called pull requests) work. There is no "develop" branch; only "master".

12.1.4 GitLab

OptiDocs' source is on GitLab, a source code hosting website. GitLab is not the same as GitHub. You will need a GitLab account to contribute. For help using GitLab, see https://docs.gitlab.com/.

12.2 Browsing the source

The source documents for OptiDocs are always open-source and are located at https://gitlab.com/whoatemybutter/optifinedocs.git. You can view any original rST document there. GitLab also offers a preview function when viewing rST files.

12.3 Creating issues

You can create an issue about anything you want, from a typo to an overhaul of a section of pages. Read about issues and how to create them here. You can also label your issues appropriately so they can be better organized and worked on.

12.4 Contributing directly

You can modify the OptiDocs source code and merge it back into the master branch yourself, instead of creating an issue and waiting for it to be completed by someone else. There are two main ways to do this:

1. Creating an isolated fork and merging your changes.
2. Making a merge request from an issue.

12.4.1 Standalone and merge

You should create an issue about the change you want to make first. This way, your merge request can be linked to it directly. Standalone merge requests are OK, but issues are **heavily** preferred.

First, you will need to make a clone of the OptiDocs source repository; this is called **forking**:

```
git clone https://gitlab.com/whoatemybutter/optifinedocs.git
```

The repository's name is `optifinedocs`, **not** `optidocs`.

Make the changes you would like to make to the repository. Remember, this is git, so you need to add and commit, too.

Important

You do not need to update the changelog document. That will be done manually after your MR is merged.

Upload your forked repository to a GitLab repository. Next, you will need to make a merge request (MR or PR). GitHub calls it *pull request*, GitLab calls it *merge request*; they are the same thing. You can first read GitLab's documentation on merge requests and https://docs.gitlab.com/ee/user/project/merge_requests/creating_merge_requests.html. Once your merge request has been made, please wait for a review.

12.4.2 Link to issue

Find or create the relevant issue to which you would like to contribute to. On the OptiDocs GitLab, merge request branches should have documentation built for them automatically by ReadTheDocs.

12.5 Style guide

For indentation, use either a tab character or 4 spaces. It does not matter, as long as it is consistent in 1 document.

12.5.1 Checkboxes

In *Syntax's string matching section*, you may notice some green checks and red crosses. To reproduce this, use this rST:

```
.. task-list::
    :class: cross-check
    :custom:

    - [x] yes
    - [x] check
    - [ ] no
    - [ ] cross
```

12.5.2 Admonitions

There are 4 custom admonitions you can use:

```
.. location::

    file location of relevant subject

.. notconfused::

    different subject that people may confuse with

.. legacy::

    this is a legacy property and should not be used; this does not require body text

.. default::

    this is the default value for this property key
```

12.5.3 Headers

From highest importance to lowest:

1. #
2. =
3. -
4. ~
5. ^
6. " (almost never used; if your nesting is this deep, you likely need to re-think your approach!)

Header underline length **must** equal length of header text.

Headers are in sentence case (Anniversary cape, Custom GUI, Special thingy). Do not start headers with articles (a, the).

For more examples, refer to the Wikipedia page MOS:HEADCAPS.

12.5.4 Text & prose

Be neutral and present information as if you are speaking to someone who has not used OptiFine before. Always link to specific features (CIT, CTM, etc). Link to glossary terms (using `:term:`myword``) if applicable.

12.5.5 Lists

Capitalize the first word and end entries with puncutation.

12.5.6 Tables

Use `.. csv-table::` or `.. list-table::` where possible. If you can, avoid the "=" based tables.

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

CHAPTER

THIRTEEN

GLOSSARY

Banner Cape
A type of OptiFine-specific *cape*. These capes use a banner pattern as their design.

Blending
A technique of combining two or more images.

Bone
A folder in which CEM entity elements are stored for both organisation, rotation, and animation purposes.

Boolean
A true or false value.

Cape
A purely-decorative cosmetic that is worn on the back of a player's body model. Capes may be given by Mojang (Minecon), or bought by donating to OptiFine.

CEM
Abbreviation for *Custom Entity Models*.

CEMA
Abbreviation for *Custom Entity Model Animations*.

CIT
Abbreviation for *Custom Item Textures*.

Colormap
A texture that defines how a block's tinting changes with biome and height.

Commit
A revision ID for use in source repositories. These IDs (hashes) are not sequential; commit 834FA0 may come before 43A3B9.

CPM
Abbreviation for *Custom Player Models*.

CTM
Abbreviation for *Connected Textures Mod*.

Debug keys
Key combination that executes some kind of debugging function in the game. See *Debug Keys*.

Dynamic Lights
Lighting that is emitted from hand-held items such as torches. See *Dynamic Lights*.

Element
An individual cube that makes up a model.

Emissive
Shorthand for *emissive texture*.

Emissive texture
A *texture* with full brightness that is overlayed on top of another texture. See *Emissive Textures*.

F2
Keyboard binding for taking a screenshot.

F3
Keyboard binding for opening the debug screen.

F5
Keyboard binding for changing player perspective.

F11
Keyboard binding for toggling fullscreen.

Face
A flat surface on a model.

Flag
An option added to a system command that modifies how it executes. See https://en.wikipedia.org/wiki/Command-line_interface#Command-line_option. This is not a Minecraft-specific term.

FPS
Abbreviation for frames per second.

Frame
A single texture in an animated texture.

Glint
The enchantment texture that is overlayed on top of enchanted items.

Interpolation
A smooth transition between two images. See https://en.wikipedia.org/wiki/Interpolation_(computer_graphics).

JSON
A format mainly used for storing *CEM*. See https://en.wikipedia.org/wiki/JSON.

JVM
Abbreviation for Java Virtual Machine. This is the application that runs Minecraft's code.

labPBR
A format of texture that defines how other objects look with *shaders*. See https://wiki.shaderlabs.org/wiki/LabPBR_Material_Standard.

Lagometer
A graph of the resources in rendering each frame. It is visible only when the debug screen (F3) is shown.

Layer
The second skin layer above the base player model. May also be a common mistranslation of *cape*.

Lightmap
A texture that defines how the lighting of various light sources change. See *Custom Lightmaps*.

Locking
A cape that cannot be moved to another username through the Cape Editor. Locked capes can only be moved to a different username after logging in through the OptiFine website.

NBT
Named Binary Tag, a tree data structure format that Minecraft uses to store information. Slightly similar to JSON in structure, it has a `{key=value, ...}` structure.

OF
An abbreviation for OptiFine.

Offhand
The player's second hand slot. Not present in 1.8.

OptiFine
A Minecraft optimization mod.

Panorama
A set of *textures* that create an illusion of distance three-dimensional surroundings. Mostly refers to the main menu panorama.

Parent Bone
The top-most bone in a list of bones; this bone is not inside any other bones

PBR
Shorthand for labPBR.

Pivot Point
A coordinate at which an *element* rotates around.

Read The Docs
Website where the unofficial documentation is hosted.

RTD
RTFD
Abbreviation for *read the docs*.

Shaders
A program that changes how Minecraft renders objects, such as the sun, water, blocks, the GUI, and more. For development, see *Shaders - Development*.

Skybox
Similar to a *panorama*, but for skies.

SNBT
Stringified *NBT*.

sp614x
The creator of OptiFine. The only developer of OptiFine.

Special Cape
A unique type of OptiFine-specific *cape*, given to a very few select of players by *sp614x*.

Special Cosmetics
Old name for additional cosmetics that OptiFine may load. These are available to only a very few select of players. See *Custom Player Models*.

Stack
The maximum amount of one item you can hold in one slot. Commonly 1, 8, 16, or 64.

String
A sequence of letters. OptiFine reads these in the ASCII encoding, not UTF-8.

Texture
A PNG file.

Tile
A *texture* applied to one *face* of a block.

UV
Shorthand for UV mapping.

Weight
: Number that states the relative priority of something, when compared to other things of the same order.

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

CHAPTER

FOURTEEN

JSON SCHEMAS

Throughout this documentation, many resource pack features pages have a corresponding JSON schema file attached at the bottom.

These files are intended to be used for validating resource packs that use OptiFine.

Features that use a `.properties` based system will need to have their files mapped to simple JSON files:

Listing 1: Properties in

```
a=4
b=somestring
c="quotes"
```

Listing 2: JSON out

```
{
    "a": 4,
    "b": "something",
    "c": "\"quotes\""
}
```

14.1 List

- `Better Grass` (*page*)
- `Block Render Layers` (*page*)
- `CEM Animation` (*page*)
- `CEM Model` (*page*)
- `CEM Part` (*page*)
- `CIT` (*page*)
- `CIT Global` (*page*)
- `Colormaps` (*page*)
- `CTM` (*page*)
- `Custom Animations` (*page*)
- `Custom Loading Screens` (*page*)
- `Custom Panoramas` (*page*)

- `Custom Sky` (*page*)
- `Dynamic Lights` (*page*)
- `Emissive Textures` (*page*)
- `HD Fonts` (*page*) *(deprecated)*
- `Natural Textures` (*page*)
- `Random Entities` (*page*)
- `Texture Properties` (*page*)

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

CHAPTER

FIFTEEN

BETTER GRASS

File location

/assets/minecraft/optifine/bettergrass.properties

Better Grass shows the side grass texture on grass blocks that are near other grass blocks.

If a **bottom** grass block is cardinally adjacent *(down one block, and 1 block forward north, east, south, or west)*, to a **top** block, the top-face grass block texture will replace the side texture of the **top** block.

Fig. 1: A grass block with grass on the side.

Fig. 2: A showcase of various grass block and path block combinations.

Fig. 3: Button and tooltip for the option, found in *Video Settings* → *Quality*.

15.1 Properties

15.1.1 `grass`

Values: Boolean
Optional
Default: `true`

Enables Better Grass for Grass Blocks.

15.1.2 `dirt_path`

Values: Boolean
Optional
Default: `true`

Enables Better Grass for grass path blocks.

15.1.3 mycelium

Values: Boolean
Optional
Default: `true`

Enables Better Grass for Mycelium.

15.1.4 podzol

Values: Boolean
Optional
Default: `true`

Enables Better Grass for Podzol.

15.1.5 crimson_nylium

Values: Boolean
Optional
Default: `true`

Enables Better Grass for Crimson Nylium.

New in version H5.

15.1.6 warped_nylium

Values: Boolean
Optional
Default: `true`

Enables Better Grass for Warped Nylium.

New in version H5.

Not to be confused with

Better Snow

15.1.7 grass.snow

Values: Boolean
Optional
Default: `true`

Enables Better Grass for Grass Blocks that have a snow layer or snow block on top of them.

15.1.8 mycelium.snow

Values: Boolean
Optional
Default: `true`

Enables Better Grass for Mycelium that has a snow layer or snow block on top of them.

15.1.9 podzol.snow

Values: Boolean
Optional
Default: `true`

Enables Better Grass for Podzol that have a snow layer or snow block on top of them.

15.1.10 grass.multilayer

Values: Boolean
Optional
Default: `false`

Allows a transparent grass texture to be used as an overlay for the grass block's side. If enabled, a transparent grass texture can overlay it.

If enabled:

- Layer 1: `grass_side`
- Layer 2: `grass` *(colored by biome)*

15.1.11 texture.grass

Values: String, *path to texture*
Optional
Default: `block/grass_block_top`

What texture to use for the **top** of a grass block which has Better Grass applied to it.

Note

This texture will be tinted (colored) by biome.

15.1.12 texture.grass_side

Values: String, *path to texture*
Optional
Default: `block/grass_block_side`

What texture to use for the **side** of a grass block which has Better Grass applied to it.

15.1.13 texture.dirt_path

Values: String, *path to texture*
Optional
Default: `block/dirt_path_top`

What texture to use for the **top** of a dirt path block which has Better Grass applied to it.

15.1.14 texture.dirt_path_side

Values: String, *path to texture*
Optional
Default: `block/dirt_path_side`

What texture to use for the **side** of a dirt path block which has Better Grass applied to it.

15.1.15 texture.mycelium

Values: String, *path to texture*
Optional
Default: `block/mycelium_top`

What texture to use for the **top** of a Mycelium block which has Better Grass applied to it.

15.1.16 texture.podzol

Values: String, *path to texture*
Optional
Default: `block/podzol_top`

What texture to use for the **top** of a Podzol block which has Better Grass applied to it.

15.1.17 texture.crimson_nylium

Values: String, *path to texture*
Optional
Default: `block/crimson_nylium`

What texture to use for the **top** of a Crimson Nylium block which has Better Grass applied to it.

New in version H5.

15.1.18 texture.warped_nylium

Values: String, *path to texture*
Optional
Default: `block/warped_nylium`

What texture to use for the **top** of a Warped Nylium block which has Better Grass applied to it.

New in version H5.

15.1.19 texture.snow

Values: String, *path to texture*
Optional
Default: `block/snow`

What texture to use for the **top** of a Snow block which has Better Grass applied to it.

15.2 Examples

15.2.1 Only Grass

```
mycelium=false
podzol=false
```

15.2.2 Not Grass

```
grass=false
dirt_path=false
crimson_nylium=false
warped_nylium=false
```

15.2.3 Texture for grass sides

```
texture.grass_side=block/redstone_block
texture.grass=block/emerald_block
```

15.2.4 Disable snowy blocks

```
grass.snow=false
mycelium.snow=false
podzol.snow=false
```

15.3 JSON schema

Note

Although this page is `.properties` based, it can be mapped to JSON.

```
{
        "$schema": "http://json-schema.org/draft/2020-12/schema",
        "$id": "https://gitlab.com/whoatemybutter/optifinedocs/-/blob/master/schemas/
↪better_grass.schema.json",
        "title": "Better Grass",
        "description": "Better Grass shows the side grass texture on grass blocks that␣
↪are near other grass blocks.",
        "type": "object",
        "properties": {
                "grass": {
                        "type": "boolean",
                        "description": "Enables Better Grass for Grass Blocks.",
                        "default": true
```

(continues on next page)

(continued from previous page)

```
                },
                "dirt_path": {
                        "type": "boolean",
                        "description": "Enables Better Grass for grass path blocks.",
                        "default": true
                },
                "mycelium": {
                        "type": "boolean",
                        "description": "Enables Better Grass for Mycelium.",
                        "default": true
                },
                "podzol": {
                        "type": "boolean",
                        "description": "Enables Better Grass for Podzol.",
                        "default": true
                },
                "crimson_nylium": {
                        "type": "boolean",
                        "description": "Enables Better Grass for Crimson Nylium.",
                        "default": true
                },
                "warped_nylium": {
                        "type": "boolean",
                        "description": "Enables Better Grass for Warped Nylium.",
                        "default": true
                },
                "grass.snow": {
                        "type": "boolean",
                        "description": "Enables Better Grass for Grass Blocks that have␣
↪a snow layer or snow block on top of them.",
                        "default": true
                },
                "mycelium.snow": {
                        "type": "boolean",
                        "description": "Enables Better Grass for Mycelium that has a␣
↪snow layer or snow block on top of them.",
                        "default": true
                },
                "podzol.snow": {
                        "type": "boolean",
                        "description": "Enables Better Grass for Podzol that has a snow␣
↪layer or snow block on top of them.",
                        "default": true
                },
                "grass.multilayer": {
                        "type": "boolean",
                        "description": "Allows a transparent grass texture to be used as␣
↪an overlay for the grass block's side.",
                        "default": false
                },
                "texture.grass": {
                        "default": "block/grass_block_top",
```

(continues on next page)

(continued from previous page)

```
                        "description": "What texture to use for the top of a grass block␣
↪which has Better Grass applied to it.",
                        "$ref": "common.schema.json#/$defs/resource"
                },
                "texture.grass_side": {
                        "default": "block/grass_block_side",
                        "description": "What texture to use for the side of a grass␣
↪block which has Better Grass applied to it.",
                        "$ref": "common.schema.json#/$defs/resource"
                },
                "texture.dirt_path": {
                        "description": "What texture to use for the top of a dirt path␣
↪block which has Better Grass applied to it.",
                        "default": "block/dirt_path_top",
                        "$ref": "common.schema.json#/$defs/resource"
                },
                "texture.dirt_path_side": {
                        "description": "What texture to use for the side of a dirt path␣
↪block which has Better Grass applied to it.",
                        "default": "block/dirt_path_side",
                        "$ref": "common.schema.json#/$defs/resource"
                },
                "texture.mycelium": {
                        "description": "What texture to use for the top of a Mycelium␣
↪block which has Better Grass applied to it.",
                        "default": "block/mycelium_top",
                        "$ref": "common.schema.json#/$defs/resource"
                },
                "texture.podzol": {
                        "default": "block/podzol_top",
                        "description": "What texture to use for the top of a Podzol␣
↪block which has Better Grass applied to it.",
                        "$ref": "common.schema.json#/$defs/resource"
                },
                "texture.crimson_nylium": {
                        "description": "What texture to use for the top of a Crimson␣
↪Nylium block which has Better Grass applied to it.",
                        "default": "block/crimson_nylium",
                        "$ref": "common.schema.json#/$defs/resource"
                },
                "texture.warped_nylium": {
                        "default": "block/warped_nylium",
                        "description": "What texture to use for the top of a Warped␣
↪Nylium block which has Better Grass applied to it.",
                        "$ref": "common.schema.json#/$defs/resource"
                },
                "texture.snow": {
                        "description": "What texture to use for the top of a Snow block␣
↪which has Better Grass applied to it.",
                        "default": "block/snow",
                        "$ref": "common.schema.json#/$defs/resource"
                }
```

(continues on next page)

(continued from previous page)

```
        },
        "additionalProperties": false
}
```

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

CHAPTER

SIXTEEN

BETTER SNOW

Shows a snow layer beneath specific blocks that have a snow layer or snow block near them.

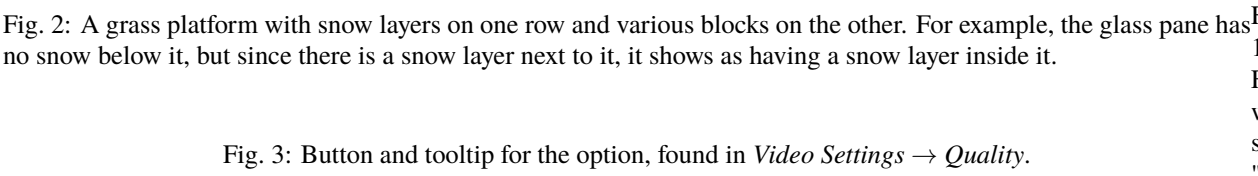

Fig. 2: A grass platform with snow layers on one row and various blocks on the other. For example, the glass pane has no snow below it, but since there is a snow layer next to it, it shows as having a snow layer inside it.

Fig. 1: Fence with snow "inside" it.

Fig. 3: Button and tooltip for the option, found in *Video Settings* → *Quality*.

16.1 Block list

- Glass panes
- Tall grass
- Grass
- Ferns
- Flowers
- Fences
- Fence gates
- Hoppers
- Saplings
- Iron bars
- Walls

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

CHAPTER

SEVENTEEN

BLOCK RENDER LAYERS

File location

(shaderpack)/shaders/block.properties

The custom block render layers are defined in `shaders/block.properties` included in a shader pack.

Warning

Blocks which are solid opaque cubes (stone, dirt, ores, etc) can't be rendered on a custom layer as this would affect face culling, ambient occlusion, light propagation and so on.

17.1 Properties

17.1.1 `layer.solid`

Values: *List* of blocks
Optional

No alpha, no blending (solid textures).

17.1.2 `layer.cutout`

Values: *List* of blocks
Optional

Alpha, no blending (cutout textures).

17.1.3 layer.cutout_mipped

Values: *List* of blocks
Optional

Alpha, no blending, mipmaps (cutout with mipmaps)

17.1.4 layer.translucent

Values: *List* of blocks
Optional

Alpha, blending, mipmaps (water, stained glass).

17.2 Example

```
layer.translucent=glass_pane fence wooden_door
layer.cutout=oak_stairs
layer.solid=stone dirt
```

17.3 JSON schema

Note

Although this page is `.properties` based, it can be mapped to JSON.

```
{
        "$schema": "http://json-schema.org/draft/2020-12/schema",
        "$id": "https://gitlab.com/whoatemybutter/optifinedocs/-/blob/master/schemas/
↪block_render_layers.schema.json",
        "title": "Block Render Layers",
        "description": "The custom block render layers are defined in ``shaders/block.
↪properties`` included in a shader pack.",
        "type": "object",
        "properties": {
                "layer.solid": {
                        "description": "No alpha, no blending (solid textures).",
                        "$ref": "common.schema.json#/$defs/item_id_list"
                },
                "layer.cutout": {
                        "description": "No alpha, no blending (cutout textures).",
                        "$ref": "common.schema.json#/$defs/item_id_list"
                },
                "layer.cutout_mipped": {
                        "description": "Alpha, no blending, mipmaps (cutout with␣
```

(continues on next page)

(continued from previous page)

```
↪mipmaps).",
                        "$ref": "common.schema.json#/$defs/item_id_list"
                },
                "layer.translucent": {
                        "description": "Alpha, blending, mipmaps (water, stained glass).
↪",
                        "$ref": "common.schema.json#/$defs/item_id_list"
                }
        },
        "additionalProperties": false
}
```

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

CHAPTER

EIGHTEEN

CUSTOM ENTITY MODELS

File location

/assets/minecraft/optifine/cem/**/*.jem /assets/minecraft/optifine/cem/**/*.jpm

Custom Entity Models (CEM) can completely change how an entity appears, animates, and moves.

The file format for CEM are JSON with the extensions `.jem` and `.jpm`.

Model files (`.jem`) contain a list of *entity part models*, which may be loaded inline or from a file `.jpm` file. These model files define a texture, and a list of submodels. These submodels define what part of an entity they attach to, how they attach, their *animations*, and more.

Fig. 1: A scary shark (source).

Fig. 2: Button and tooltip for the option, found in *Video Settings* → *Quality*.

18.1 Models

File location

/assets/minecraft/optifine/cem/**/*.jem

CEM model files contain the definition of a whole entity model. It is written in JSON.

They may be located anywhere inside the `cem` folder. The name of the file must match one of the entity names in *Entity names*, or be in the folder `/assets/minecraft/optifine/cem/<entity name>`.

Fig. 3: A 3D model in Blockbench.

18.1.1 Keys

`texture`

Values: String, file location
Optional

Texture used by entity model.

textureSize

Values: Array of 2 integers
Optional

Texture size in pixels; `[width, height]`.

shadowSize

Values: Decimal
Optional

Shadow size as a scale, from `0.0` to `1.0`.

models

Values: List of objects
Required

Array of model objects that make up the entity's full model.

baseId

Values: String
Optional

Model parent ID. If specified, all parent properties are inherited and do not need to be explicitly put.

model

Values: String, file location
Optional

Path to a *part model file* (`.jpm`) from which to load the part model definition.

If this is not specified, the items in a JPM can be specified inline to this object, the parent of "model".

id

Values: String
Optional

Model ID, can be used to reference the model as parent.

part

Values: String
Required

Entity part to which the part model is attached.

See *Entity names* for a list of part names for each entity.

Important

Make sure that the part names correspond with your model's applicable entity.

---

attach

Values: Boolean
Optional

How to handle replacing overlapping parts.

- True: Attach (add) to the entity part.
- False: Replace the entity part.

scale

Values: Float
Optional
Default: `1.0`

Render scale. `0.0` makes it "invisible".

Part model definitions

All of the items in a *CEM parts file* can be put here instead, if `model` is absent.

`animations`

Values: List of objects
Optional
Default: `[]`

Refer to *CEM animation* for what to place in each object in this list.

18.1.2 Randomized models

The alternative models use the same name as the main model with a number suffix.

For example:

- wolf.jem - main model (index 1)
- wolf2.jem - alternative model (index 2)
- wolf3.jem - alternative model (index 3)

The alternative models are selected randomly based on the entity ID.

To customize the use of the alternative models, add a `<model_name>.properties` file in the folder where the models are located.

The properties file works identically to the properties file used by *Random Entities*. The models to be used are selected with the setting `models.<n>=<list>` instead of `textures.<n>=<list>`. The index of the current matching rule is available as the *animation* parameter `rule_index`, and can be used to cutomize the model depending on entity properties.

For more details, see *Random Entities*.

Examples

Listing 1: creeper.properties, creeper.jem, creeper2.jem

```
models.1=2
name.1=James
```

Listing 2: boat.properties, boat.jem, boat2.jem, boat3.jem

```
models.1=2
nbt.1.Type=spruce

models.2=3
nbt.2.Type=birch
```

Listing 3: bed.properties, bed.jem, bed2.jem, bed3.jem

```
models.1=2
models.1=2
name.1=James
blocks.1=black_bed

models.2=3
blocks.2=orange_bed
```

18.1.3 JSON schema

```
{
        "$schema": "http://json-schema.org/draft/2020-12/schema",
        "$id": "https://gitlab.com/whoatemybutter/optifinedocs/-/blob/master/schemas/cem_
↪model.schema.json",
        "title": "Custom Entity Models Model",
        "description": "CEM model files contain the definition of a whole entity model.",
        "type": "object",
        "properties": {
                "texture": {
                        "$ref": "common.schema.json#/$defs/resource",
                        "description": "Texture used by entity model."
                },
                "textureSize": {
                        "type": "array",
                        "minItems": 2,
                        "maxItems": 2,
                        "items": {
                                "type": "integer"
                        },
                        "description": "Texture size in pixels; [width, height]."
                },
                "shadowSize": {
                        "type": "number",
                        "minimum": 0,
                        "maximum": 1,
                        "description": "Shadow size as a scale, from 0.0 to 1.0."
                },
                "models": {
                        "type": "array",
                        "description": "Array of model objects that make up the entity's␣
↪full model.",
                        "items": {
                                "type": "object",
                                "properties": {
                                        "baseId": {
                                                "type": "string",
                                                "description": "Model parent ID. If␣
↪specified, all parent properties are inherited and do not need to be explicitly put."
                                        },
```

(continues on next page)

(continued from previous page)

```
                                        "model": {
                                                "type": "string",
                                                "$ref": "common.schema.json#/$defs/
↪resource",
                                                "description": "Path to a JPM from which␣
↪to load the part model definition. If this is not specified, the items in a JPM can be␣
↪specified inline to this object, the parent of \"model\"."
                                        },
                                        "id": {
                                                "type": "string",
                                                "description": "Model ID, can be used to␣
↪reference the model as parent."
                                        },
                                        "part": {
                                                "type": "string",
                                                "description": "Entity part to which the␣
↪part model is attached."
                                        },
                                        "attach": {
                                                "type": "boolean",
                                                "description": "How to handle replacing␣
↪overlapping parts. If true, attach. If false, replace."
                                        },
                                        "scale": {
                                                "type": "number",
                                                "minimum": 0,
                                                "description": "Render scale. 0.0 is␣
↪invisible."
                                        },
                                        "animations": {
                                                "$ref": "cem_anim.schema.json#/
↪properties/animations"
                                        }
                                },
                                "required": [
                                        "part"
                                ],
                                "allOf": [
                                        {
                                                "$ref": "cem_part.schema.json"
                                        }
                                ]
                        }
                }
        },
        "required": [
                "models"
        ],
        "additionalProperties": false
}
```

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

18.2 Parts

File location

/assets/minecraft/optifine/cem/**/*.jpm

CEM part files contain the definition of one model part to be referenced in a CEM model file.

18.2.1 Keys

`invertAxis`

Values: String, permutation of "xyz"
Optional

Axes to invert: "xyz" inverts the X, Y, Z axes. An empty string inverts none of them and is equal the key being absent.

`translate`

Values: Array of 3 integers
Optional

Translate (move) the texture by the 3 integers `x`, `y`, and `z`.

`rotate`

Values: Array of 3 integers
Optional

Rotation (spin) the texture by the 3 integers `angle_x`, `angle_y`, and `angle_z`.

mirrorTexture

Values: String permutation of "uv"
Optional

Texture axis to mirror. `"uv"` will mirror both the U and V axis. An empty string mirrors along no axis and is equal the key being absent.

boxes

Values: Array of objects
Optional

Array of part model boxes.

textureOffset

Values: Array of 2 integers
Required

Texture offset for the box format; `[x, y]`.

uv<FACE>

Values: Array of 4 strings
Required

<FACE> is one of `Down`, `Up`, `North`, `South`, `West`, or `East`.

UV for face. Ordered as [`u1`, `v1`, `u2`, `v2`].

coordinates

Values: Array of 6 integers
Required

Box position and dimensions. Ordered as [`x`, `y`, `z`, `width`, `height`, `depth`].

sizeAdd

Values: Integer
Optional

Size increment added to all dimensions; can be used for asymmetric scaling.

sprites

Values: Array of objects
Optional

List of 3D sprite models; depth 1.

textureOffset

Values: Array of 2 integers
Required

Texture offset; `[x, y]`.

coordinates

Values: Array of 6 integers
Required

Box position and dimensions. Ordered as [`x`, `y`, `z`, `width`, `height`, `depth`].

sizeAdd

Values: Integer
Optional

Size increment added to all dimensions; can be used for asymmetric scaling.

submodel

Values: Self, CEM part object
Optional

A sub-model part; attached to the parent, moving and rotating with it.

submodels

Values: List of CEM part objects
Optional

A list of sub-model parts; attached to the parent, moving and rotating with it.

18.2.2 Texture UV

Warning

Texture UV cannot have both specifications, either "textureOffset" or uv<face>, not both.

Texture UV can be specified in box format with:

- "textureOffset", or
- "uvDown", "uvUp", "uvNorth", "uvSouth", "uvWest", and "uvEast".

Fig. 4: The box format UV mapping.

18.2.3 JSON schema

```
{
        "$schema": "http://json-schema.org/draft/2020-12/schema",
        "$id": "https://gitlab.com/whoatemybutter/optifinedocs/-/blob/master/schemas/cem_
↪part.schema.json",
        "title": "Custom Entity Models Part",
        "description": "CEM parts contains definition of one model part for a whole CEM␣
↪model.",
        "type": "object",
        "properties": {
                "texture": {
                        "type": "string",
                        "description": "Texture used by entity model.",
                        "$ref": "common.schema.json#/$defs/resource"
                },
                "textureSize": {
                        "type": "array",
                        "items": {
```

(continues on next page)

(continued from previous page)

```
                                "type": "integer"
                        },
                        "minItems": 2,
                        "maxItems": 2,
                        "description": "Texture size (width, height) in pixels."
                },
                "invertAxis": {
                        "type": "string",
                        "maxLength": 3,
                        "pattern": "^([xyz])?((?!\\1)[xyz])?((?!\\1)(?!\\2)[xyz])?((?!\\
↪1)(?!\\2)(?!\\3))?$",
                        "description": "Axes to invert; \"xyz\" inverts the X, Y, and Z␣
↪axes, \"\" inverts none of them."
                },
                "translate": {
                        "type": "array",
                        "minItems": 3,
                        "maxItems": 3,
                        "items": {
                                "type": "integer"
                        },
                        "description": "Translate texture by [0], [1], [2]."
                },
                "rotate": {
                        "type": "array",
                        "minItems": 3,
                        "maxItems": 3,
                        "items": {
                                "type": "integer"
                        },
                        "description": "Rotate texture by [0], [1], [2]."
                },
                "mirrorTexture": {
                        "type": "string",
                        "minLength": 2,
                        "maxLength": 2,
                        "pattern": "^([uv])?((?!\\1)[uv])?((?!\\1)(?!\\2)[uv])?$"
                },
                "boxes": {
                        "type": "array",
                        "items": {
                                "type": "object",
                                "properties": {
                                        "textureOffset": {
                                                "$ref": "#/$defs/textureOffset"
                                        },
                                        "coordinates": {
                                                "$ref": "#/$defs/coordinates"
                                        },
                                        "sizeAdd": {
                                                "$ref": "#/$defs/sizeAdd"
                                        }
```

(continues on next page)

(continued from previous page)

```
                                },
                                "patternProperties": {
                                        "^uv(Down|Up|North|South|West|East)$": {
                                                "type": "array",
                                                "minItems": 4,
                                                "maxItems": 4,
                                                "items": {
                                                        "type": "string"
                                                },
                                                "description": "UV for face."
                                        }
                                }
                        },
                        "description": "List of part model boxes."
                },
                "sprites": {
                        "type": "array",
                        "items": {
                                "type": "object",
                                "properties": {
                                        "textureOffset": {
                                                "$ref": "#/$defs/textureOffset"
                                        },
                                        "coordinates": {
                                                "$ref": "#/$defs/coordinates"
                                        },
                                        "sizeAdd": {
                                                "$ref": "#/$defs/sizeAdd"
                                        }
                                }
                        }
                },
                "submodel": {
                        "type": "object",
                        "$ref": "#",
                        "description": "Submodel; attached to the parent, moving and␣
↪rotating with it."
                },
                "submodels": {
                        "type": "array",
                        "items": {
                                "type": "object",
                                "$ref": "#"
                        },
                        "description": "List of submodels; attached to the parent,␣
↪moving and rotating with it."
                }
        },
        "allOf": [
                {
                        "anyOf": [
                                {
```

(continues on next page)

(continued from previous page)

```
                                        "required": [
                                                "textureOffset"
                                        ]
                                },
                                {
                                        "required": [],
                                        "patternProperties": {
                                                "^(?!(uv(Down|Up|North|South|West|East))
↪$).*$": {}
                                        }
                                }
                        ]
                },
                {
                        "required": [
                                "coordinates"
                        ]
                }
        ],
        "$defs": {
                "textureOffset": {
                        "type": "array",
                        "items": {
                                "type": "integer"
                        },
                        "minItems": 2,
                        "maxItems": 2,
                        "description": "Texture offset for the box format."
                },
                "coordinates": {
                        "type": "array",
                        "minItems": 6,
                        "maxItems": 6,
                        "items": {
                                "type": "integer"
                        },
                        "description": "Box position and dimensions. x, y, z, width,␣
↪height, depth."
                },
                "sizeAdd": {
                        "type": "number",
                        "minimum": 0,
                        "maximum": 65535,
                        "description": "Size increment added to all dimensions; can be␣
↪used for asymmetric scaling."
                }
        },
        "additionalProperties": false
}
```

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

18.3 Animation

This is the reference configuration for animating OptiFine's Custom Entity Models (CEMA).

Important

These apply inside of *Models* and this document is kept separate for organizational purposes **only**. It is **not** a separate file.

Fig. 5: Bendy creeper.

Each model variable which is to be animated is assigned a mathematical expression. The expression is evaluated every time the model is rendered and its value is assigned to the variable. This value controls the positions, scales, rotations, etc. of the parts of the model.

Important

Animations **must be in the parent bone**, *not* any sub-bone.

The variables and expressions are defined in the "animations" section of the JSON entity model **(JEM)**:

```
{
    "animations": [
        {
            "variable1": "expression1",
            "variable2": "expression2"
        }
    ]
}
```

18.3.1 Keys

Variables make up the keys of the items in the objects in the "animations" list. They are strings that refer to different values that control an entity.

For example, in the object `{"a": 5}`, "a" is the **key** and 5 is the **value**.

Model variables

Model variables are specified in the format `<model>.<variable_name>`.

The `<model>` can be one of:

- `this`: the current custom model.
- `part`: the original part model to which the custom model is attached.
- `<part>`: the original model by part name.
- `<id>`: the custom model by an assigned ID.

- `<part>:<sub_id>:<sub_sub_id>:...`: (hierarchical) start with the original model by part name, then find its children by ID.
- `<id>:<sub_id>:<sub_sub_id>:...`: (hierarchical) start with the original model by ID, then find children its by ID.

The first model found by part name or ID is used if there are duplicates. The model search by ID is deep, and is also deep when used in a hierarchical specification.

The hierarchical specification allows model groups (JSON part models) to be reused for different parts. For example, one hand model (`shoulder:upper_arm:elbow:forearm:palm:finger[1.5]`) can be used for both left and right hands; `left_hand:finger1` can be for the left thumb and `right_hand:finger1` for the right thumb.

The intermediate parents in the hierarchical specification can be skipped.

Model variable names

- `tx`, `ty`, `tz`: **translation** x, y, z (movement).
- `rx`, `ry`, `rz`: **rotation** x, y, z (spin, yaw, pitch).
- `sx`, `sy`, `sz`: **scale** x, y, z (size).
- `visible`: show model and submodels (boolean).
- `visible_boxes`: show model only; this does not affect submodels (boolean).

New in version H9: `visible` and `visible_boxes`

Entity variables

Warning

Entity variables are not supported for block entities.

Entity variables can be specified in 2 formats: `var.<name>` for floats (decimals, numbers), or `varb.<name>` for booleans (true, false). The `<name>` can be any string; for example `var.xyz`, `var.last_rx`, `var.cookies_in_the_cookie_jar`, etc.

New in version H9: `varb`

The variable is attached to the rendered entity and has a default value or 0 or false for `var` and `varb`, respectively.

Entity variables are useful for storing animation data between frames or storing constants for animation configuration.

Render variables

New in version H9.

- `render.shadow_size`: The size of the entity's shadow; this is a float from `1.0` (opaque) to `0.0` (invisible).
- `render.shadow_opacity`: The opacity (solidness) of the entity's shadow.
- `render.leash_offset_x`: When leashed, where the leash on the entity attachs to.
- `render.leash_offset_y`
- `render.leash_offset_z`
- `render.shadow_offset_x`: An offset of the entity's shadow position.

- `render.shadow_offset_z`

18.3.2 Values

Expressions make up the **values** of the items in the objects in the "animations" list. They are general mathematical expressions with brackets, constants, variables, operators, parameters, and functions.

Hint

They most closely resemble the AsciiMath format, although they are not identical.

Optionally-grouped variables, constants, parameters, and functions are separated by operators and evaluated as normal math expressions. For example, `sin(35)+max(5,50,500)` evaluates to `500.5735764`.

Constants

Constants never change and always evaluate to the same value.

Name	Type	Meaning
`pi`	Float, constant	Floating point, equal to `3.1415926`.
`true`	Boolean, constant	Truthy boolean.
`false`	Boolean, constant	False boolean.

Variables

Variables change and are most often used to change something with time.

Not to be confused with

Keys; these are expression variables.

Name	Type	Meaning
`<model>.<var>`	Any	A model variable's value, see the *Model variables* section.
`time`	Integer, ticks	The total game time in ticks (`0..720720`); not related to the daylight cycle.
`day_time`	Integer, ticks	**The current day time in ticks (`0..24000`).** New in version I3.
`day_count`	Integer	**The current day count.** New in version I3.

Render parameters

Name	Type	Meaning
`limb_swing`	Float	Counts up in ticks from 0 as the entity continues to move.
`limb_speed`	Float	The current speed of the entity's limbs. Ranges from `0.0` (still) to `1.0` (sprinting).
`age`	Float	How long the entity has existed in the world, in ticks.
`head_yaw`	Float	Head yaw; x rotation.
`head_pitch`	Float	Head pitch; y rotation.
`player_pos_x`	Float	**Player's X position (not necessarily the same entity).** New in version H8.
`player_pos_y`	Float	**Player's Y position (not necessarily the same entity).** New in version H8.
`player_pos_z`	Float	**Player's Z position (not necessarily the same entity).** New in version H8.
`player_rot_x`	Float	**Player's yaw (left-right).** New in version H8.
`player_rot_y`	Float	**Player's pitch (up-down).** New in version H8.
`frame_time`	Float	**Time in seconds since the last frame.** New in version H9.
`dimension`	Integer	**Dimension ID, -1 = Nether, 0 = Overworld, 1 = End.** New in version H9.
`rule_index`	Integer	**The index of the current matching *random models* rule. Defaults to 0.** New in version I1.

Entity parameters

Name	Type	Meaning
	Floats	
`health`	Float	Entity's current health.
`hurt_time`	Float	Time stage of when entity has been hurt once. Counts down from 10 to 0.
`death_time`	Float	**Time stage on entity's death. Counts up from 0 to 20.** New in version H8.
`anger_time`	Float	**The time the entity has been angry. 0 while neutral, 720 while aggressive, counts down to 0 when the target is lost.** New in version H9.
`max_health`	Float	Entity's maximum health.
`move_forward`	Float	How much entity is moving forwards-backwards, currently broken.
`move_strafing`	Float	How much entity is moving left-right, currently broken.
`pos_x`	Float	Entity's X position.
`pos_y`	Float	Entity's Y position.
`pos_z`	Float	Entity's Z position.
`rot_x`	Float	**Entity's X-axis rotation.** New in version H8.
`rot_y`	Float	**Entity's Y-axis rotation.** New in version H8.
`swing_progress`	Float	How far through the attack animation the entity is; counts up from `0.0` to `1.0`.
`id`	Float	**A unique numeric identifier.** New in version H8.
	Booleans	
`is_aggressive`	Boolean	**If the entity is aggressive towards another entity.** New in version H9.
`is_alive`	Boolean	If the entity is alive; not dead.
`is_burning`	Boolean	If the entity is burning.

continues on next page

Table 1 – continued from previous page

Name	Type	Meaning
is_child	Boolean	If the entity is a child.
is_glowing	Boolean	If the entity has the Glowing status effect.
is_hurt	Boolean	If the entity is taking damage.
is_in_hand	Boolean	If the entity **(items)** is being held in your hand.
is_in_item_frame	Boolean	**If the entity (items) is in an item frame.** New in version H7.
is_in_ground	Boolean	If the entity is embedded into a block (arrows, tridents).
is_in_gui	Boolean	If the entity is inside the GUI
is_in_lava	Boolean	If the entity is touching lava.
is_in_water	Boolean	If the entity is touching water.
is_invisible	Boolean	If the entity has the Invisibility status effect.
is_on_ground	Boolean	If the entity is on the ground; not flying.
is_on_head	Boolean	**If the entity (items) is on an armor head slot.** New in version H7.
is_on_shoulder	Boolean	**If the entity is on a player's shoulder (parrots).** New in version H9.
is_ridden	Boolean	If the entity is being ridden by another entity.
is_riding	Boolean	If the entity is riding atop of another entity.
is_sitting	Boolean	**If the entity is sitting (cat, wolf, parrot).** New in version H9.
is_sneaking	Boolean	If the entity (cats) is crouching/sneaking.
is_sprinting	Boolean	If the entity (cats) is sprinting.
is_tamed	Boolean	**If the entity is tamed (dogs, cats).** New in version H9.
is_wet	Boolean	If the entity is under rain or is inside a water block.

Operators

Name	Meaning
+, -, *, /, %	add, subtract, multiply, divide, modulo
!, &&, ||	negate, logical AND, logical OR
>, >=, <, <=, ==, !=	greater than, greater than or equal to, less than, less than or equal to, is equal to, is not equal to

Numerical functions

These functions return a number.

Name	Parameters	Return
`sin(x)`	`x`: any number	Sine of degrees `x`.
`cos(x)`	`x`: any number	Cosine of degrees `x`.
`asin(x)`	`x`: any number	Arcsine of degrees `x`.
`acos(x)`	`x`: any number	Arccosine of degrees `x`.
`tan(x)`	`x`: any number	Tangent of degrees `x`.
`atan(x)`	`x`: any number	Arctangent of degrees `x`.
`atan2(y, x)`	`y`: any number, `x`: any number	Two-argument arctangent of `y` and `x`.
`torad(deg)`	`deg`: any degree	Convert degrees to radians.
`todeg(rad)`	`rad`: any radian	Convert radians to degrees.
`min(x[, y...]`	`x...`: any number	The minimum of all given parameters.
`max(x[, y...]`	`x...`: any number	The maximum of all given parameters.
`clamp(x, min, max)`	`x, min, max`: any number	`x`, guaranteed to be between `min` and `max` values; if `x > max`, `x = max`, `x < min`, `x = min`.
`abs(x)`	`x`: any number	The absolute value of `x`; `abs(-5) == 5`.
`floor(x)`	`x`: any number	The floor of `x`; `floor(2.9) == 2`.
`ceil(x)`	`x`: any number	The ceiling of `x`; `ceil(2.1) == 3`.
`exp(x)`	`x`: any number	*e* (Euler's constant) raised to the power of `x`; `exp(4) == 54.598150033144236`.
`frac(x)`	`x`: any decimal	The decimal of `x`; `frac(11.4) == 0.4`.
`log(x)`	`x`: any number	The logarithm of `x`; `log(50) == 3.912023005428146`.
`pow(x, y)`	`x`: any number, `y`: any positive number	Raise base `x` to the power `y`; `pow(5, 2) == 25`
`random(see`	`seed`: any integer	**A random number between 0.0 and 1.0. `seed` is optional and if given, this returns the same result.** New in version H8.
`round(x)`	`x`: any decimal	Rounded `x`; `round(5.4) == 5`.
`signum(x)`	`x`: any number	The sign of `x`; `signum(0) == 0, signum(-5253) == -1`.
`sqrt(x)`	`x`: any positive number	The square root of `x`; `sqrt(25) == 5`.
`fmod(x, y)`	`x, y`: any number	Similar to Java's floorMod function, `x - (floorDiv(x, y) * y)`; if the signs of the arguments are the same, the results of `fmod` and the `%` operator are the same, but if the signs of the arguments are different, the results differ: `floorMod(+4, -3) == -2, (+4 % -3) == +1`
`lerp(k, x, y)`	`k, x, y`: any number	**The linear interpolation of `x` and `y`; `(1 - k) * x + k * y`.** New in version H9.
`if(cond, val[, cond2, val2, ...], val_else)`	`cond`: condition string, `val`: any value, `val_else`: any value	Select a value based on one of more conditions: return `val` if `cond` is true, return `val_else` if `cond` is false
`print(id, n, x)`	`id`: ?, `n`: n-th frame, `x`: value to print	**Prints `x` every `n`-th frame. Prints are sent to the output log, which can be enabled in the launcher settings.** New in version H8.
`printb(id, n, x)`	`id`: ?, `n`: n-th frame, `x`: boolean to print	**Prints `x` boolean every `n`-th frame. Prints are sent to the output log, which can be enabled in the launcher settings.** New in version H9.

Boolean functions

These functions return either true or false.

Name	Parameters		Tests
`between(x, min, max)`	`x, min, max`: number	any	Is `x` between `min` and `max`?
`equals(x, y, epsilon)`	`x, y, epsilon`: number	any	Is the difference of `x` and `y` within error margin `epsilon`? `abs(x-y) < epsilon`
`in(x, val1[, val2...])`	`x, val1...`: number	any	Is `x` equivalent to any of of `val1...`?

18.3.3 Examples

Basic structure

```
{
    "animations": [
            {
                    "this.rx": "clamp(-0.5 * part.rx, 0, 90)",
                    "this.tx": "3 * sin(limb_swing / 4) - 2",
                    "this:Hoof.rx": "if(leg4:Hoof.rx > 90, leg4:Hoof.rx - 90, 0)"
            }
    ]
}
```

Walking animation

`x` is a multipler to control how fast the leg swings back and forth, and `y` is a multiplier to control how far it swings back and forth.

```
"left_leg.rx": "sin(limb_swing * x) * limb_speed * y"
```

Attack animation

`x` is a multipler for how much it rotates.

```
"head.rx": "sin(swing_progress * pi) * x"
```

Hurt animation

x is a multipler for how much it rotates.

```
"head.rx": "-sin(hurt_time / pi) * x"
```

Custom counter

This is a counter that will count up while an entity is in water, and count down again when it leaves.

```
"var.counter": "if(is_in_water, min(20, var.counter + 0.1 * frame_time * 20), max(0, var.
↪counter - 0.1 * frame_time * 20))"
```

If statements

The leg will rotate by 45 degrees when the entity is not on the ground, otherwise it will stay at 0 deg.

```
"left_leg.rx": "if(!is_on_ground, torad(45), 0)"
```

The body will tilt forwards once the entity hits a certain movement speed.

```
"body.rx": "if(limb_speed > 0.7, torad(20), 0)"
```

18.3.4 Tutorial

18.3.5 JSON schema

```
{
        "$schema": "http://json-schema.org/draft/2020-12/schema",
        "$id": "https://gitlab.com/whoatemybutter/optifinedocs/-/blob/master/schemas/cem_
↪anim.schema.json",
        "title": "Custom Entity Models Animation",
        "description": "CEM Animations change how a custom entity model walks, swims,␣
↪idles, etc.",
        "type": "object",
        "properties": {
                "animations": {
                        "type": "array",
                        "items": {
                                "type": "object",
                                "patternProperties": {
                                        "^(this|part|[0-9a-zA-Z_]*((:[0-9a-z-A-Z_])*)?)\\
↪.(t[xyz]|r[xyz]|s[xyz])$": {
                                                "$ref": "#/$defs/expression"
                                        },
                                        "^(this|part|[0-9a-zA-Z_]*((:[0-9a-z-A-Z_])*)?)\\
↪.(render\\.(shadow_size|shadow_opacity|leash_offset_[xyz]|shadow_offset_[xz]))$": {
```

(continues on next page)

(continued from previous page)

```
                                                "$ref": "#/$defs/expression"
                                        },
                                        "^(this|part|[0-9a-zA-Z_]*((:[0-9a-z-A-Z_])*)?)\\
↪.(visible|visible_boxes)$": {
                                                "type": "boolean"
                                        },
                                        "^varb?\\.(.+)$": {
                                                "$ref": "#/$defs/expression"
                                        }
                                }
                        }
                }
        },
        "$defs": {
                "expression": {
                        "type": "string",
                        "minLength": 1,
                        "anyOf": [
                                {
                                        "pattern": "pi|true|false|time|day_time|day_count
↪"
                                },
                                {
                                        "pattern": "(this|part|[0-9a-zA-Z_]*((:[0-9a-z-A-
↪Z_])*)?)\\.(.*?)"
                                },
                                {
                                        "pattern": "limb_swing|limb_speed|age|head_
↪yaw|head_pitch|player_pos_[xyz]|player_rot_[xyz]|frame_time|dimension|rule_index"
                                },
                                {
                                        "pattern": "(max_)?health|(hurt|death|anger)_
↪time|move_(forward|strafing)|pos_[xyz]|rot_[xy]|swing_progress|id"
                                },
                                {
                                        "pattern": "is_
↪(aggressive|alive|burning|child|glowing|hurt|in_(hand|item_
↪frame|ground|gui|lava|water)|invisible|on_
↪(head|shoulder|ground)|ridden|riding|sitting|sneaking|sprinting|tamed|wet)"
                                },
                                {
                                        "pattern": "\\+|-|\\*|\/|
↪%|add|subtract|multiply|divide|!|&&|\\|\\||[<>]=?|[!=]="
                                },
                                {
                                        "pattern":
↪"(sin|cos|asin|acos|tan|atan|to(rad|deg)|abs|floor|ceil|exp|frac|log|random|round|signum|sqrt)\
↪\((-?\\d+(.\\d+)?)\\)"
                                },
                                {
                                        "pattern": "(atan2|pow|fmod)\\((-?\\d+(.\\d+)?),␣
↪(-?\\d+(.\\d+)?)\\)"
```

(continues on next page)

(continued from previous page)

```
                                },
                                {
                                        "pattern": "printb?\\((.*?), ?(-?\\d+(.\\d+)?), ?
↪(.*?)\\)"
                                },
                                {
                                        "pattern": "i[fn]\\(((.*?), ?(.*?))+\\)"
                                },
                                {
                                        "pattern": "lerp|clamp|between|equals\\((-?\\d+(.
↪\\d+)?), ?(-?\\d+(.\\d+)?), ?(-?\\d+(.\\d+)?)\\)"
                                },
                                {
                                        "pattern": "m(in|ax)\\(((-?\\d+(.\\d+)?)(?:,\\
↪s*(-?\\d+(.\\d+)?))?)\\)"
                                },
                                {
                                        "pattern": "-?\\d+(.\\d+)?"
                                }
                        ]
                }
        },
        "additionalProperties": false
}
```

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

18.4 Entity names

This is a table of entity and part names. Part names must be matched with the entity they will apply to.

18.4.1 Table

Fig. 6: Too many names.

Entity name	Part name
allay	head, body, left_arm, right_arm, left_wing, right_wing
armor_stand	head, headwear, body, left_arm, right_arm, left_leg, right_leg, right, left, waist, base
axolotl	head, body, leg1 ... leg4, tail, top_gills, left_gills, right_gills
banner	slate, stand, top
bat	head, body, right_wing, left_wing, outer_right_wing, outer_left_wing

continues on next page

Table 2 – continued from previous page

Entity name	Part name
bee	body, torso, right_wing, left_wing, front_legs, middle_legs, back_legs, stinger, left_antenna, right_antenna
bed	head, foot, leg1 ... leg4
bell	body
blaze	head, stick1 ... stick12
boat	bottom, back, front, right, left, paddle_left, paddle_right, bottom_no_water
camel	body, hump, tail, head, left_ear, right_ear, back_left_leg, back_right_leg, front_left_leg, front_right_leg, saddle, reins, bridle
cat	back_left_leg, back_right_leg, front_left_leg, front_right_leg, tail, tail2, head, body
cat_collar	back_left_leg, back_right_leg, front_left_leg, front_right_leg, tail, tail2, head, body
cave_spider	head, neck, body, leg1 ... leg8
chest	lid, base, knob
chest_boat	bottom, back, front, right, left, paddle_left, paddle_right, bottom_no_water, chest_base, chest_lid, chest_knob
chest_large	lid_left, base_left, knob_left, lid_right, base_right, knob_right
chest_mineca	bottom, back, front, right, left, dirt
chest_raft	bottom, paddle_left, paddle_right, chest_base, chest_lid, chest_knob
chicken	head, body, right_leg, left_leg, right_wing, left_wing, bill, chin
cod	body, fin_back, head, nose, fin_right, fin_left, tail
com-mand_block_	bottom, back, front, right, left, dirt
conduit	base, eye, cage, wind
cow	head, body, leg1 ... leg4
creeper	head, armor, body, leg1 ... leg4
creeper_charg	head, body, leg1 ... leg4
deco-rated_pot	neck, front, back, left, right, top, bottom
dragon	head, spine, jaw, body, left_wing, left_wing_tip, right_wing, right_wing_tip, front_left_leg, front_left_shin, front_left_foot, back_left_leg, back_left_shin, back_left_foot, front_right_leg, front_right_shin, front_right_foot, back_right_leg, back_right_shin, back_right_foot
donkey	<same as horse>, left_chest, right_chest
dolphin	body, back_fin, left_fin, right_fin, tail, tail_fin, head
drowned	head, headwear, body, left_arm, right_arm, left_leg, right_leg
drowned_outc	head, headwear, body, left_arm, right_arm, left_leg, right_leg
el-der_guardian	body, eye, spine1 ... spine12, tail1 ... tail3
enchant-ing_book	cover_right, cover_left, pages_right, pages_left, flipping_page_right, flipping_page_left, book_spine
ender_chest	lid, base, knob
end_crystal	cube, glass, base
enderman	head, headwear, body, left_arm, right_arm, left_leg, right_leg
endermite	body1 ... body4
evoker	head, hat, body, arms, left_leg, right_leg, nose, left_arm, right_arm
evoker_fangs	base, upper_jaw, lower_jaw
fox	head, body, leg1 ... leg4, tail
frog	head, body, eyes, tongue, left_arm, right_arm, left_leg, right_leg, croaking_body
fur-nace_minecar	bottom, back, front, right, left, dirt
ghast	body, tentacle1 ... tentacle9

continues on next page

Table 2 – continued from previous page

Entity name	Part name
giant	head, headwear, body, left_arm, right_arm, left_leg, right_leg
glow_squid	body, tentacle1 ... tentacle8
goat	head, body, leg1 ... leg4, left_horn, right_horn, nose
guardian	body, eye, spine1 ... spine12, tail1 ... tail3
hanging_sign	board, plank, chains, chain_left1, chain_left2, chain_right1, chain_right2, chains_v
head_dragon	head, jaw
head_creeper	head
head_piglin	head
head_player	head
head_skeleto	head
head_wither_	head
head_zombie	head
hoglin	head, right_ear, left_ear, body, front_right_leg, front_left_leg, back_right_leg, back_left_leg, mane
hopper_minecart	bottom, back, front, right, left, dirt
horse	body, neck, back_left_leg, back_right_leg, front_left_leg, front_right_leg, tail, saddle, head, mane, mouth, left_ear, right_ear, left_bit, right_bit, left_rein, right_rein, headpiece, noseband, child_back_left_leg, child_back_right_leg, child_front_left_leg, child_front_right_leg
horse_armor	body, neck, back_left_leg, back_right_leg, front_left_leg, front_right_leg, tail, saddle, head, mane, mouth, left_ear, right_ear, left_bit, right_bit, left_rein, right_rein, headpiece, noseband, child_back_left_leg, child_back_right_leg, child_front_left_leg, child_front_right_leg
husk	head, headwear, body, left_arm, right_arm, left_leg, right_leg
illusioner	head, hat, body, arms, left_leg, right_leg, nose, left_arm, right_arm
iron_golem	head, body, left_arm, right_arm, left_leg, right_leg
lead_knot	knot
lectern_book	cover_right, cover_left, pages_right, pages_left, flipping_page_right, flipping_page_left, book_spine
llama	head, body, leg1 ... leg4, chest_right, chest_left
llama_decor	head, body, leg1 ... leg4, chest_right, chest_left
llama_spit	body
magma_cube	core, segment1 ... segment8
minecart	bottom, back, front, right, left, dirt
mooshroom	head, body, leg1 ... leg4
mule	<same as horse>, left_chest, right_chest
ocelot	back_left_leg, back_right_leg, front_left_leg, front_right_leg, tail, tail2, head, body
panda	head, body, leg1 ... leg4
parrot	head, body, tail, left_wing, right_wing, left_leg, right_leg
phantom	body, left_wing, left_wing_tip, right_wing, right_wing_tip, head, tail, tail2
puffer_fish_b	body, fin_right, fin_left, spikes_front_top, spikes_middle_top, spikes_back_top, spikes_front_right, spikes_front_left, spikes_front_bottom, spikes_middle_bottom, spikes_back_bottom, spikes_back_right, spikes_back_left
puffer_fish_m	body, fin_right, fin_left, spikes_front_top, spikes_back_top, spikes_front_right, spikes_back_right, spikes_back_left, spikes_front_left, spikes_back_bottom, spikes_front_bottom
puffer_fish_s	body, eye_right, eye_left, tail, fin_right, fin_left
pig	head, body, leg1 ... leg4
pig_saddle	head, body, leg1 ... leg4

continues on next page

Table 2 – continued from previous page

Entity name	Part name
piglin	head, headwear, body, left_arm, right_arm, left_leg, right_leg, left_ear, right_ear, left_sleeve, right_sleeve, left_pants, right_pants, jacket
piglin_brute	head, headwear, body, left_arm, right_arm, left_leg, right_leg, left_ear, right_ear, left_sleeve, right_sleeve, left_pants, right_pants, jacket
pillager	head, hat, body, arms, left_leg, right_leg, nose, left_arm, right_arm
polar_bear	head, body, leg1 ... leg4
rabbit	left_foot, right_foot, left_thigh, right_thigh, body, left_arm, right_arm, head, right_ear, left_ear, tail, nose
raft	bottom, paddle_left, paddle_right
ravager	head, jaw, body, leg1 ... leg4, neck
salmon	body_front, body_back, head, fin_back_1, fin_back_2, tail, fin_right, fin_left
sheep	head, body, leg1 ... leg4
sheep_wool	head, body, leg1 ... leg4
shulker	head, base, lid
shulker_box	base, lid
shulker_bulle	bullet
sign	board, stick
silverfish	body1 ... body7, wing1 ... wing3
skeleton	head, headwear, body, left_arm, right_arm, left_leg, right_leg
skeleton_horse	<same as horse>
slime	body, left_eye, right_eye, mouth
slime_outer	body, left_eye, right_eye, mouth
sniffer	body, back_left_leg, back_right_leg, middle_left_leg, middle_right_leg, front_left_leg, front_right_leg, head, left_ear, right_ear, nose, lower_beak
snow_golem	body, body_bottom, head, left_hand, right_hand
spawner_min	bottom, back, front, right, left, dirt
spider	head, neck, body, leg1, ... leg8
squid	body, tentacle1 ... tentacle8
stray	head, headwear, body, left_arm, right_arm, left_leg, right_leg
stray_outer	head, headwear, body, left_arm, right_arm, left_leg, right_leg
strider	body, right_leg, left_leg, hair_right_top, hair_right_middle, hair_right_bottom, hair_left_top, hair_left_middle, hair_left_bottom
strider_saddle	body, right_leg, left_leg, hair_right_top, hair_right_middle, hair_right_bottom, hair_left_top, hair_left_middle, hair_left_bottom
tnt_minecart	bottom, back, front, right, left, dirt
tadpole	body, tail
trader_llama	head, body, leg1 ... leg4, chest_right, chest_left
trader_llama_	head, body, leg1 ... leg4, chest_right, chest_left
trapped_chest	lid, base, knob
trapped_chest	lid_left, base_left, knob_left, lid_right, base_right, knob_right
tropical_fish_a	body, tail, fin_right, fin_left, fin_top
tropical_fish_patte	body, tail, fin_right, fin_left, fin_top
tropical_fish_b	body, tail, fin_right, fin_left, fin_top, fin_bottom
tropical_fish_patte	body, tail, fin_right, fin_left, fin_top

continues on next page

Table 2 – continued from previous page

Entity name	Part name
turtle	head, body, leg1 ... leg4, body2
vex	head, body, left_arm, right_arm, left_wing, right_wing
villager	head, headwear, headwear2, body, bodywear, arms, left_leg, right_leg, nose
vindicator	head, hat, body, arms, left_leg, right_leg, nose, left_arm, right_arm
wandering_trader	head, headwear, headwear2, body, bodywear, arms, left_leg, right_leg, nose
warden	body, torso, head, left_leg, right_leg, left_arm, right_arm, left_tendril, right_tendril, left_ribcage, right_ribcage
witch	head, headwear, headwear2, body, bodywear, arms, left_leg, right_leg, nose, mole
wither	body1 ... body3, head1 ... head3
wither_armor	body1 ... body3, head1 ... head3
wither_skelet	head, headwear, body, left_arm, right_arm, left_leg, right_leg
wither_skull	head
wolf	head, body, leg1 ... leg4, tail, mane
wolf_collar	head, body, leg1 ... leg4, tail, mane
zoglin	head, right_ear, left_ear, body, front_right_leg, front_left_leg, back_right_leg, back_left_leg, mane
zombie	head, headwear, body, left_arm, right_arm, left_leg, right_leg
zombie_horse	<same as horse>
zombie_pigman	head, headwear, body, left_arm, right_arm, left_leg, right_leg
zombie_villager	head, headwear, body, left_arm, right_arm, left_leg, right_leg
zombified_piglin	head, headwear, body, left_arm, right_arm, left_leg, right_leg, left_ear, left_sleeve, right_sleeve, left_pants, right_pants, jacket

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

18.5 Limitations

18.5.1 Parent bones

On each CEM model, the model is limited to using the parent bones that that entity has by default. A full list of these can be found *here*. Every single element added must be inside one of these parent bones. Adding a new parent bone will cause the model to fail to load in-game.

18.5.2 Pivot points

The pivot points of the entities **cannot be modified in Vanilla**. An example of something that cannot be modified is moving the leg of a cow, as that would require moving its leg's pivot point.

However, this can be done using *CEM Animation*. While modelling the entity, pivot points on bones will behave as expected, allowing elements to rotate around a point, however this is not the case for parent bones.

When a template model is loaded, all the parent bones will already have their pivot points set up correctly. Do not touch these, or the model will break when loaded in-game.

If, for whatever reason, the pivot points need to be moved in the model, this is how it works:

```
The elements are tied to the pivot point in game.
Increasing the gap between the pivot point and the elements␣
↪will increase the gap between the actual pivot point and the elements in game.

For example, if the pivot point is moved␣
↪12 pixels east inside the model, the elements will appear 12 pixels west in game.
This is␣
↪because, as the pivot point cannot be moved in game, the elements will move instead.
The elements render relative to the pivot point.
```

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

CHAPTER

NINETEEN

CUSTOM ITEM TEXTURES

Custom Item Textures (CIT) can change items to different textures based on their properties, such as enchantments, name, or NBT rules.

Fig. 2: Button and tooltip for the option, found in *Video Settings* → *Quality*.

Fig. 1: The amount changes the texture.

19.1 Global properties

File location

/assets/minecraft/optifine/cit.properties

This file contains global properties for CIT and should be in the `optifine/cit` folder of the resource pack.

For individual item textures, see the *Properties* section.

Note

All paths are relative to `/assets/minecraft/` unless otherwise specified.
`average` and `layered` methods with `cap=1` are equivalent and will both show only the "dominant" enchantment on an item.

Warning

Not implemented: `method`, `cap`, `fade`.

19.1.1 method

Values: `average`, `layered`, or `cycle`
Optional
Default: `average`

Specifies how to apply multiple effects on the same item. Depending on the method chosen, multiple effects can be rendered with different intensities from `0` (invisible) to `1` (fully visible).

- `average`: Weighted average by enchantment level: `intensity = enchantment_level / sum(enchantment_levels)`.
- `layered`: Similar to `average`, but `max()` is used instead of `sum()`: `intensity = enchantment_level / max(enchantment_levels)`.
- `cycle`: Cycle through each effect in turn. The duration of each effect can be set via the `duration` property. The `[group]` value (if present) allows multiple sets of effects to be cycled through independently.

19.1.2 cap

Values: Positive integer
Optional

Specifies how many layers can render for `average` and `layered` methods. The top-most layers have priority over bottom-most layers as determined by the layer value of each effect.

19.1.3 fade

Values: Positive float
Optional
Default: `0.5`

The speed at which one effect will transition into another in a cycle. This does not affect the duration of the actual effect when displayed. For that, use the effect's `duration` property.

19.1.4 useGlint

Values: Boolean
Optional
Default: `true`

Use default `glint.png` enchantment texture or not.

- If **true**: `glint.png` is used if no other custom enchantment effect is available.
- If **false**: the default `glint.png` enchantment stops rendering completely.

This is important for items that have no specific enchantment, but have an enchantment effect - such as potions and golden apples.

Danger

Past 1.12, this no longer works.

19.2 Properties

File location

/assets/minecraft/optifine/cit/**/*.properties

For each item to override with a custom texture, create a `.properties` file in the `/assets/minecraft/optifine/cit/` folder of the resource pack. Properties files can be organized into subfolders of any depth, as long as everything is within the top-level `optifine/cit` folder.

Each properties file specifies:

- A list of matching item IDs or names.
- A replacement texture or model.
- An **optional** set of rules specifying when this CIT will apply to the item.

Note

All property names are case-sensitive.
All paths are relative to `/assets/minecraft/` unless otherwise specified.
For best compatibility with tag matching, use escape sequences for characters outside the ASCII range; `\u0107` instead of `ć`.
These properties apply to all CIT types.

19.2.1 `type`

Values: `item`, `enchantment`, `armor`, or `elytra`
Optional
Default: `item`

Type of texture replacement.

item

Simple item texture replacement. Applies to items in GUI, held in hand, and in the world. If multiple properties files match the same item, only the first is used (sorted by weight, then by file name).

enchantment

Overlay texture for enchantments *(replaces misc/glint.png)*. If multiple properties files match the same item, they are blended together using rules specified in *Global properties*.

Danger

Past 1.12, this no longer works.

---

armor

Armor texture replacement. Applies to armor models worn by players and mobs. If multiple properties files match the same item, only the first (sorted by weight, then by file name) is used.

elytra

Elytra texture replacement. Applies to elytra model worn by players and mobs. If multiple properties files match the same item, only the first (sorted by weight, then by file name) is used.

19.2.2 items

Values: *List* of items
Optional

List of items to apply the CIT to.

May be infinitely long, or a lone item. Elements act as a union of a set, meaning that the CIT will apply to *any* item in the list.

19.2.3 texture

Values: String, *path to texture*
Optional
Default: *Name of properties file:* `x.properties -> x.png`

Path to replacement texture.

Can be a full path or just a name:

- `mytextures/excalibur.png` → `mytextures/excalibur.png`
- `excalibur` → `optifine/cit/excalibur.png`

19.2.4 model

Values: String, *path to texture*
Optional

Path to replacement model. Model must be in vanilla format.

- `item/mymodel` → `/assets/minecraft/models/item/mymodel.json`
- `./mymodel` → `mymodel.json` from the same folder as the properties file

Model may reference textures from the same folder as where the calling properties file is, for example `./mytexture`.

19.2.5 damage

Values: Integer from `0` to `65535`, integer *range* from `0` to `65535`, or percentage range
Optional

Damage values. Replacement texture is used only when the item damage is a certain value or range.

For items with durability, damage starts at 0 for a new item and increases as it gets damaged. A brand-new item's damage will always be 0. The maximum damage an item can take varies

For 1.12- items, `damage` represents different properties like potion type. See this page for specifics.

Warning

Using `damage` to detect wool color is deprecated and should not be done.

19.2.6 damageMask

Values: Integer bitmask
Optional
Default: `0`

Binary bitmask applied to the item's damage before checking it against the list of eligible damage values

Examples:

- Match any Fire Resistance potion: `damage=3 damageMask=15`
- Match any non-splash Fire Resistance potion: `damage=3 damageMask=16399`
- Match non-splash Fire Resistance I potion only: `damage=3 damageMask=16447`
- Match splash Fire Resistance II potion only: `damage=16403 damageMask=16447`
- For a simpler way, see *Potions*.

Danger

This is an extremely unreliable and largely-unused method of checking properties. Do not use it.

19.2.7 stackSize

Values: Integer from `0` to `65535`, or integer *range* from `0` to `65535`
Optional
Default: `0-65535`

Required amount of item that must be in 1 inventory slot.

Can be a range of an example number. Although the maximum legitimate amount is 64, values up to 65535 are allowed.

Note

Values above 64 are useless.

19.2.8 enchantments

Values: *List* of strings
Optional
Default: *Any*

List of enchantment names to match.

The enchantment names may be short (`flame`) or in full (`minecraft:flame`). For example: `enchantments=minecraft:silk_touch sharpness smite`.

Note

If `enchantmentLevels` is not specified, this rule matches **any** enchantment level.

19.2.9 enchantmentIDs

Legacy

Alias to `enchantments`.

19.2.10 enchantmentLevels

Values: *List* of integers from `0` to `255`
Optional
Default: *Any*

List of enchantment levels.

Also allows ranges. For example: `enchantmentLevels=1 3 5 10`, or `enchantmentLevels=5-`.

Note

If `enchantments` is not specified, this rule matches **any** enchantment type.

19.2.11 `hand`

Values: `any`, `main`, or `off`
Optional
Default: `any`

Hand in which the item is placed onto (main hand, offhand). When rendered in the inventory GUI, the item is considered to be in the `main` hand.

Fig. 3: CIT can apply conditionally if you're holding the item in the left or right hand.

19.2.12 `nbt`

Values: Any valid *NBT matching rule*
Optional

NBT-based rule.

Replacement texture is used only when an NBT tag on the item has a specific value. See *NBT*. You can have infinitely-many NBT rules in a CIT.

19.3 Type-specific properties

19.3.1 Items

Note

Implies `type=item`.

`texture`

Values: String, *path to texture*
Required

Item replacement textures are stitched into `items.png`, and thus follow the same rules as normal item textures.

In particular, this means that animations **must** use Mojang's system of `.mcmeta` files for frame order and timing.

texture.<name>

Values: String, *path to texture*
Optional

Replacement for alternate textures.

For items with more than one texture, this allows specifying replacements for each texture separately. These textures depend on the item's model. For example: the vanilla bow has four possible textures depending on its state: `bow_standby`, `bow_pulling_0`, `bow_pulling_1`, or `bow_pulling_2`.

To replace **all four**, this can be used:

```
texture.bow_standby=my_special_bow_standby
texture.bow_pulling_0=my_special_bow_pulling_0
texture.bow_pulling_1=my_special_bow_pulling_1
texture.bow_pulling_2=my_special_bow_pulling_2
```

Potions also have two textures. To replace them, use:

```
texture.potion_overlay=...
texture.potion_bottle=...
```

Note

If no `texture.<name>` property matches, the generic texture property is used instead.

model.<name>

Values: String, *path to texture*
Optional

Replacement for alternate models.

For items with more than one model, this allows specifying replacements for each model separately.

For example: the vanilla bow has four possible textures depending on its state: `bow_standby`, `bow_pulling_0`, `bow_pulling_1`, `bow_pulling_2`. To replace **all four**, this can be used:

```
model.bow_standby=my_special_bow_standby
model.bow_pulling_0=my_special_bow_pulling_0
model.bow_pulling_1=my_special_bow_pulling_1
model.bow_pulling_2=my_special_bow_pulling_2
```

weight

Values: Positive integer
Optional
Default: `0`

If multiple properties files match the same item, the highest weighted one is used (biggest weight number). In the event of a tie, the properties filenames are sorted and compared alphabetically.

19.3.2 Enchantments

Note

Implies `type=enchantment`.

Note

`duration` only works for `cycle` enchantments.

Danger

Past 1.12, this no longer works.

texture

Values: String, *path to texture*
Required

The enchantment texture can be any resolution.

To animate an enchantment, use the `anim/*.properties` method with `to=full path to enchantment texture`

blend

Values: String
Optional
Default: add

Blend method when applying texture to the texture below it.

See *Blending methods* for a list of valid blending methods.

speed

Values: Positive integer
Optional
Default: 1

Scrolling speed of texture.

0 means no scrolling.

rotation

Values: Positive integer from 0 to 360
Optional

Angle of texture (in degrees) relative to the item.

If speed is non-zero, the texture will also scroll in this direction.

layer

Values: Positive integer
Optional
Default: 0

Specifies a unique layer and the ordering of the layers as they overlap each other.

If two or more effects use the same layer, weight next determines which effect is rendered (the other is not rendered).

weight

Values: Positive integer
Optional
Default: 0

Relative priority of the enchantment within a layer.

Of the matching effects, only the highest weighted one within a layer is rendered. In other words:

- The layer property determines the **ORDER** in which effects are rendered.
- The weight property determines **WHICH** effect is rendered for each layer.

If two effects have the same weight and layer, the properties filenames are sorted and compared alphabetically.

duration

Values: Positive integer
Optional
Default: `0`

Duration in seconds of the enchantment glint in a cycle.

19.3.3 Armor

Note

Implies `type=armor`

texture.<name>

Values: String, *path to texture*
Required

Replacement textures.

A replacement for each texture is needed in `minecraft:textures/models/armor/` for that armor type.

For diamond armor *(2 layers total)*:

```
texture.diamond_layer_1=my_diamond_armor_1
texture.diamond_layer_2=my_diamond_armor_2
```

For leather armor *(4 layers total)*:

```
texture.leather_layer_1=my_leather_armor_1
texture.leather_layer_1_overlay=my_leather_armor_1_overlay
texture.leather_layer_2=my_leather_armor_2
texture.leather_layer_2_overlay=my_leather_armor_2_overlay
```

The texture should match the format of the corresponding armor texture.

For animated textures, use the `anim/*.properties` method with `to`.

19.4 Potions

Note

While there is no specific potion-related tag, `nbt` should be used.

Potions with custom effects can be matched using their NBT `Potion` string, or with `nbt.CustomPotionEffects.*.Id`.

NBT Potion string

```
type=item
items=potion
nbt.Potion=minecraft:strength
```

CustomPotionEffects ID

```
type=item
items=potion
nbt.CustomPotionEffects.*.Id=20
```

Stronger versions of potions can be matched by prefixing `strong_` to the `Potion` NBT tag match. Longer versions of potions can be matched by prefixing `long_` to the `Potion` NBT tag match. Lingering and Splash potions can be matched with the same method by simply changing the `items` tag appropriately.

19.4.1 Shortcut

Note

Everything described here **can** be done via CIT properties files; this is a **shortcut**.

Note

No properties files are necessary for this method.

As an alternative to listing potion damage values or testing NBT, replacement textures for potions can be specified using a file name-based system.

See this page for the data-values.

There are **three** directories for potions:

1. `optifine/cit/potion/normal`: drinkable potions.
2. `optifine/cit/potion/splash`: splash potions.
3. `optifine/cit/potion/linger`: lingering potions.

Within any of these directories, create a PNG file with the name of the potion effect:

Note

Effect names in *italic* means they are obtainable in-game.
Effects not in *italic* can only be created via commands.

Important

This replaces **both** `potion.png/potion_splash.png` and `potion_contents.png` from the standard potion rendering.

Warning

Be sure to include the *colored* liquid in the replacement textures. Tint is not applied.

Effect name	File name
Absorption	`absorption.png`
Blindness	`blindness.png`
Confusion	`confusion.png`
Damage Boost	`damageboost.png`
Mining Fatigue	`digslowdown.png`
Haste	`digspeed.png`
Fire Resistance	`fireresistance.png`
Harming	`harm.png`
Healing	`heal.png`
Health Boost	`healthboost.png`
Hunger	`hunger.png`
Invisibility	`invisibility.png`
Leaping	`jump.png`
Slowness	`moveslowdown.png`
Speed	`movespeed.png`
Night Vision	`nightvision.png`
Poison	`poison.png`
Regeneration	`regeneration.png`
Resistance	`resistance.png`
Saturation	`saturation.png`
Water Breathing	`waterbreathing.png`
Weakness	`weakness.png`
Wither	`wither.png`

The names are the same as the potion. The replacement texture will automatically be used for that potion type; no properties file is required. Note that this replaces **both**.

Similarly, textures can be replaced for the various "no effect" potions. These have drinkable versions **only**, the rest are in the code and are listed here only for completeness.

Note

Effect names in *italic* means they are obtainable in-game
Effects not in *italic* can only be created via commands

Effect name	File name
Artless	`artless.png`
Awkward	`awkward.png`
Bland	`bland.png`
Bulky	`bulky.png`
Bungling	`bungling.png`
Buttered	`buttered.png`
Charming	`charming.png`
Clear	`clear.png`
Cordial	`cordial.png`
Dashing	`dashing.png`
Debonair	`debonair.png`
Elegant	`elegant.png`
Fancy	`fancy.png`
Flat	`flat.png`
Foul	`foul.png`
Gross	`gross.png`
Harsh	`harsh.png`
Milky	`milky.png`
Mundane	`mundane.png`
Odorless	`odorless.png`
Potent	`potent.png`
Rank	`rank.png`
Sparkling	`sparkling.png`
Stinky	`stinky.png`
Suave	`suave.png`
Thick	`thick.png`
Thin	`thin.png`
Uninteresting	`uninteresting.png`

If a single texture for all "no effect" potions is preferred, `/assets/minecraft/optifine/cit/potion/normal/other.png` is used as a fallback for any that do not have a specific replacement as listed above.

Two additional textures *(drinkable only)* can also be provided:

- `optifine/cit/potion/normal/water.png`: water bottle
- `optifine/cit/potion/normal/empty.png`: empty glass bottle

19.5 Examples

19.5.1 Stacked coins

```
type=item
items=iron_nugget
stackSize=61-64
texture=iron_coins_16
```

19.5.2 Splash potion

/assets/minecraft/optifine/cit/fire_resistance_splash.properties

```
type=item
items=splash_potion
model=item/fire_resistance_splash
nbt.Potion=minecraft:fire_resistance
```

/assets/minecraft/models/item/fire_resistance_splash.json

```
{
    "parent": "item/generated",
    "textures": {
        "layer0": "item/fire_resistance_overlay",
        "layer1": "item/fire_resistance_splash"
    }
}
```

19.6 JSON schema

Note

Although this page is `.properties` based, it can be mapped to JSON.

```
{
        "$schema": "http://json-schema.org/draft/2020-12/schema",
        "$id": "https://gitlab.com/whoatemybutter/optifinedocs/-/blob/master/schemas/cit.
↪schema.json",
        "title": "Custom Item Textures",
        "description": "Custom Item Textures (CIT) can change items to different␣
↪textures based on their properties, such as enchantments, name, or NBT rules.",
        "type": "object",
        "properties": {
                "type": {
                        "enum": [
                                "item",
                                "enchantment",
                                "armor",
                                "elytra"
                        ],
                        "default": "item",
                        "description": "Type of texture replacement."
                },
                "items": {
                        "type": "string",
                        "$ref": "common.schema.json#/$defs/item_id_list",
                        "description": "String of a space-separated list of items to␣
```

(continues on next page)

(continued from previous page)

```
↪apply the CIT to."
                },
                "texture": {
                        "type": "string",
                        "$ref": "common.schema.json#/$defs/resource",
                        "description": "Path to replacement texture."
                },
                "model": {
                        "type": "string",
                        "$ref": "common.schema.json#/$defs/resource",
                        "description": "Path to replacement model."
                },
                "damage": {
                        "type": [
                                "integer",
                                "string"
                        ],
                        "minimum": 0,
                        "maximum": 65535,
                        "description": "Damage values. Replacement texture is used only␣
↪when the item damage is a certain value or range."
                },
                "damageMask": {
                        "type": "integer",
                        "minimum": 0,
                        "maximum": 65535,
                        "description": "Binary bitmask applied to the item's damage␣
↪before checking it against the list of eligible damage values."
                },
                "stackSize": {
                        "type": [
                                "integer",
                                "string"
                        ],
                        "minimum": 0,
                        "maximum": 65535,
                        "description": "Required amount of item that must be in 1␣
↪inventory slot."
                },
                "enchantments": {
                        "type": "string",
                        "$ref": "common.schema.json#/$defs/enchantment_list",
                        "description": "List of enchantment names to match."
                },
                "enchantmentIDs": {
                        "type": "string",
                        "$ref": "common.schema.json#/$defs/enchantment_list",
                        "description": "List of enchantment names to match. Legacy␣
↪property.",
                        "deprecated": true
                },
                "enchantmentLevels": {
```

(continues on next page)

(continued from previous page)

```
                        "type": "string",
                        "description": "Space-separated list of enchantment levels, from␣
↪0 to 255."
                },
                "hand": {
                        "enum": [
                                "any",
                                "main",
                                "off"
                        ],
                        "default": "any",
                        "description": "Hand in which the item is placed onto (main hand,
↪ offhand)."
                }
        },
        "patternProperties": {
                "^nbt\\.([a-zA-Z0-9_\\-.+]+|\".*?\")$": {
                        "type": [
                                "number",
                                "string"
                        ],
                        "description": "NBT-based rule."
                }
        },
        "anyOf": [
                {
                        "if": {
                                "properties": {
                                        "type": {
                                                "const": "item"
                                        }
                                }
                        },
                        "then": {
                                "patternProperties": {
                                        "texture\\.([a-z0-9_.]+)": {
                                                "$ref": "common.schema.json#/$defs/
↪resource",
                                                "description": "Replacement for␣
↪alternate textures. For items with more than one texture, this allows specifying␣
↪replacements for each texture separately."
                                        },
                                        "model\\.([a-z0-9_.]+)": {
                                                "$ref": "common.schema.json#/$defs/
↪resource",
                                                "description": "Replacement for␣
↪alternate models. For items with more than one model, this allows specifying␣
↪replacements for each model separately."
                                        }
                                },
                                "properties": {
                                        "weight": {
```

(continues on next page)

(continued from previous page)

```
                                                "type": "integer",
                                                "minimum": 0,
                                                "default": 0,
                                                "description": "If multiple properties␣
↪files match the same item, the highest weighted one is used (biggest weight number)."
                                        }
                                }
                        }
                },
                {
                        "if": {
                                "properties": {
                                        "type": {
                                                "const": "enchantment"
                                        }
                                }
                        },
                        "then": {
                                "properties": {
                                        "blend": {
                                                "$ref": "common.schema.json#/$defs/
↪blending_method_enum"
                                        },
                                        "speed": {
                                                "type": "integer",
                                                "minimum": 0,
                                                "default": 1,
                                                "description": "Scrolling speed of␣
↪texture."
                                        },
                                        "rotation": {
                                                "type": "integer",
                                                "minimum": 0,
                                                "maximum": 360,
                                                "description": "Angle of texture (in␣
↪degrees) relative to the item."
                                        },
                                        "layer": {
                                                "type": "integer",
                                                "minimum": 0,
                                                "description": "Specifies a unique layer␣
↪and the ordering of the layers as they overlap each other."
                                        },
                                        "weight": {
                                                "type": "integer",
                                                "minimum": 0,
                                                "description": "Relative priority of the␣
↪enchantment within a layer."
                                        },
                                        "duration": {
                                                "type": "integer",
                                                "minimum": 0,
```

(continues on next page)

(continued from previous page)

```
                                                "description": "Duration in seconds of␣
↪the enchantment glint in a cycle."
                                        }
                                }
                        }
                },
                {
                        "if": {
                                "properties": {
                                        "type": {
                                                "const": "armor"
                                        }
                                }
                        },
                        "then": {
                                "patternProperties": {
                                        "texture\\.([a-z0-9_.]+)": {
                                                "$ref": "common.schema.json#/$defs/
↪resource",
                                                "description": "Replacement for␣
↪alternate textures. For armors with more than one texture, this allows specifying␣
↪replacements for each texture separately."
                                        }
                                }
                        }
                }
        ],
        "additionalProperties": false
}
```

Listing 1: For global CIT only

```
{
        "$schema": "http://json-schema.org/draft/2020-12/schema",
        "$id": "https://gitlab.com/whoatemybutter/optifinedocs/-/blob/master/schemas/cit_
↪global.schema.json",
        "title": "Custom Item Textures Global",
        "description": "Global properties for CIT that apply for all CIT files.",
        "type": "object",
        "properties": {
                "method": {
                        "enum": ["average", "layered", "cycle"],
                        "default": "average",
                        "description": "Specifies how to apply multiple effects on the␣
↪same item."
                },
                "cap": {
                        "type": "integer",
                        "minimum": 0,
                        "description": "Specifies how many layers can render for average␣
↪and layered methods."
                },
```

(continues on next page)

(continued from previous page)

```
                "fade": {
                        "type": "number",
                        "minimum": 0.0,
                        "default": 0.5,
                        "description": "The speed at which one effect will transition␣
↪into another in a cycle."
                },
                "useGlint": {
                        "type": "boolean",
                        "default": true,
                        "description": "Use default \"glint.png\" enchantment texture or␣
↪not."
                }
        },
        "additionalProperties": false
}
```

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

CHAPTER

TWENTY

COLORMAPS

File location

/assets/minecraft/optifine/colormap/**/*

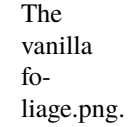

Fig. 1: The vanilla foliage.png.

Colormaps modify a texture's tint based off its biome and height.

A custom colormap can consist of either a PNG file, a `.properties` file, or both, depending on what is intended. OptiFine greatly expands this functionality to other blocks and to ambient sky and fog colors.

Artists can use this to great effect to give each biome its own feel. Custom colormaps can be applied to any one block or to a set of blocks. They can also be applied to ambient fog, sky, and underwater colors.

20.1 Formats

20.1.1 "vanilla" format

See also

See the wiki page on tint for more details.

Warning

This format is difficult to manipulate and is not recommended.

The format used by vanilla Minecraft is a 256px by 256px PNG, with the axes representing temperature and humidity, respectively. Each biome has fixed base temperature and humidity values corresponding to a **single** pixel in the colormap. As the **y** coordinate increases, the position in the colormap slowly moves toward the lower-right.

Fig. 2: An approximation of how Vanilla colormaps work.

A forum post by khanador illustrates how this works.

Note

The vanilla format is used for all custom colormaps as well, **unless this behavior is overridden**!

Biome colormaps use a triangular gradient by default. However, only the colors in the lower-left half of the image are used, even though the upper-right side of `foliage.png` is colored.

Fig. 3: The Vanilla `foliage.png` file. The upper-right side of `foliage.png` is colored is **entirely unused**.

Furthermore, a select few pixels are considered when the colormap is read by the game, and are determined by the code below.

The adjusted temperature and adjusted rainfall values are used when determining the biome color to select from the colormap. Treating the bottom-right corner of the colormap as `Temperature = 0.0` and `Rainfall = 0.0`, the adjusted temperature increases to 1.0 along the X-axis, and the adjusted rainfall increases to 1.0 along the Y-axis. The values used to retrieve the colors are computed as follows:

```
new_temperature = clamp(temperature, 0.0, 1.0)
new_rainfall = clamp(rainfall, 0.0, 1.0) * new_temperature
```

20.1.2 "grid" format

See also

The MCPatcher source

File location

/assets/minecraft/optifine/colormap/<ANY NAME>.png

An alternative format that offers finer control over each biome.

This format is similar to Vanilla's 256by by 256px format, but the **x** coordinate represents the biome ID number, and the **y** coordinate represents the height.

This allows complete separation between biomes, and gives full control from minimum to maximum build height.

Each column in the colormap represents a single biome.

Please note that the above image is "flipped" vertically:

- The **bottom** of the world (`y=0`) is at the **top** of the image.
- The normal maximum build height (`y=255`) is at the **bottom**.
- Sea level is `y=64`.

Forward compatibility

Unused columns in the map represent unassigned biome IDs that may be used by either by future Minecraft versions or by mods. Color schemes can be created for particular modded biomes if the IDs that they use are known.

If the IDs **aren't** known, it is best to at least pick a neutral-looking gradient for unused columns so that new biomes will have a reasonable default appearance, even if the pack isn't updated.

Backward compatibility

The vanilla `grass.png` and `foliage.png` maps in `/assets/minecraft/textures/colormap` are always in the vanilla format, regardless of any properties file setting.

This preserves compatibility for non-OptiFine users.

To use the grid format with grass or leaves, a custom colormap must be present in `/assets/minecraft/optifine/colormap/blocks` and be applied to the appropriate block(s). For OptiFine users, the custom colormap overrides the vanilla one; for non-OptiFine users, only the vanilla one will be used.

Resolution

While colormaps in this format are generally 256px by 256px, there is no strict requirement as there is with the vanilla format.

Minecraft 1.7 introduced rare variants of many biomes. For example "Birch Forest M" (ID 155) is the rare version of "Birch Forest" (ID 27).

Conveniently, the rare is always common + `128`. This fact can be utilized (or, exploited) if all rare biomes should use the same color schemes as the corresponding non-rare ones.

Simply make the colormap 128 pixels wide instead of 256, and OptiFine will "wrap" it in the `x` direction when assigning columns to biomes. Similarly, a 1-pixel wide colormap gives the same height-based color gradient across **all** biomes.

In the `y` direction, if more than 256 pixels is provided, OptiFine will use them if the server's build height is higher than 256, as is the case with 1.17 onward. Similarly, if the colormap is shorter than 256 pixels, it will simply "top out" at that height giving all blocks above that the same color as the top-most pixel of the map.

In particular, a height of 64 pixels allows for variation underground and a fixed color above sea level.

A height of 192 pixels combined with a property of `yOffset=64` gives just the opposite: variation above ground and a fixed color below. A height of 1 pixel allows for variation across biomes **but not by height**.

20.1.3 "fixed" format

OptiFine offers a simple "fixed" colormap format.

This format **does not** require an image; it is simply a single color applied to all blocks regardless of location. Its primary purpose is to override certain hardcoded block colors like sugar cane.

20.2 Properties

File location

/assets/minecraft/optifine/colormap/*.properties

Note

All values here are optional.

Note

The `format` property does not affect the vanilla "grass.png" and "foliage.png" files in `/assets/minecraft/textures/colormap`; those are **always interpreted in the vanilla format** in order to preserve compatibility for non-OptiFine users.

Note

The `yVariance` property can also be set globally in `/assets/minecraft/optifine/color.properties`.

Note

The `format` property can be omitted if it's set globally in the `/assets/minecraft/optifine/color.properties` file: `palette.format=grid`.
This makes all custom colormaps use the `grid` format, so ensure that this is intentional.

If the format is not `fixed`, the location of this file should sit in the same folder as the colormap image it will apply to. It should have the same base name as the texture.

20.2.1 `format`

Values: String, one of: `grid`, `vanilla`, `fixed`
Optional
Default: `vanilla`

The format to use for this colormap. If not specified, uses `vanilla` format. Has exceptions, see **NOTE** above.

20.2.2 blocks

Values: *List* of blocks
Optional

List of blocks to apply colormap to. For colormaps applied to terrain (as opposed to fog, sky, and underwater), this is a list of blocks and optional property values to apply the map to If this property is not specified, the block name is taken from the filename: `cobblestone.properties blocks=minecraft:cobblestone`

Example: `blocks=stone minecraft:sand minecraft:lever:face=wall:facing=east,west`. See *Blocks and items* for more information.

20.2.3 source

Values: String, *path to texture*
Optional

File path to colormap texture.

Note

This is for `vanilla` and `grid` colormaps only.

If this property is omitted, colormap defaults to a PNG with the same name and directory as the properties file itself: `stone.properties` <-> `source=stone.png`.

20.2.4 color

Values: String, hexadecimal RGB value without leading #
Optional
Default: `ffffff`

Differing behavior depending on `format`:

format=fixed

This color will be applied to all matching blocks.

`format=vanilla` or `format=grid`

This color is used for held and dropped blocks.

20.2.5 `yVariance`

Values: Integer
Optional
Default: `0`

If set, this property adds a random integer to the Y coordinate before sampling from the colormap, giving flat areas a more varied appearance. A value of `2` causes the game to pick a Y coordinate of `y + randomInteger(0, 2)`.

Note

This only applies to the `grid` format.

20.2.6 `yOffset`

Values: Integer
Optional
Default: `0`

Subtracts a fixed value from the block's **Y** coordinate in the world before sampling from the colormap.

For example, a value of `64` will use the pixel at `y=0` for blocks on Y-level `64`. A block at `y=65` will use pixel `1`. A block whose `y=66` uses pixel `2`, and so on.

20.3 Applying a colormap

Block-based colormaps can be applied in one of two ways:

As a list in `assets/minecraft/optifine/color.properties`

Warning

If the player is using multiple resource packs, only the *first* `color.properties` file will be read by the game.

Key	Values	Meaning	
`palette.block.<colormap image>`	**Values:** List of blocks *Optional*	Assigns each block it's colormap	*None*

For example, the below assigns Oak Leaves and tall grass their own colormaps:

- `palette.block.colormap/oak.png=oak_leaves`
- `palette.block.colormap/tall_grass_up.png=tall_grass:half=upper`
- `palette.block.colormap/tall_grass_low.png=tall_grass:half=lower`

As separate files under `assets/minecraft/optifine/colormap/blocks`

Important

This is assuming "oak.png", "tall_grass_up.png", and "tall_grass_low.png" are all in the same folder.

Subfolders are allowed and are useful to make organization easier. The left-side tabbed example could also be done this way:

- In `assets/minecraft/optifine/colormap/blocks/oak.properties`: `blocks=oak_leaves`
- In `assets/minecraft/optifine/colormap/blocks/tall_grass_up.properties`: `blocks=tall_grass:half=upper`
- In `assets/minecraft/optifine/colormap/blocks/tall_grass_low.properties`: `blocks=tall_grass:half=lower`

20.4 Grass and foliage

Custom colormaps will override the vanilla `grass.png` and `foliage.png`. This means vanilla maps can be left in place for compatibility, creating custom ones for OptiFine users.

Biome grass and foliage colors are selected from two 256px by 256px colormap images: `grass.png` and `foliage.png`. Both colormaps, shown below, can be found in `/assets/minecraft/textures/colormap/`.

Fig. 4: The Vanilla `grass.png` file, sets the colors for the grass block top and sides (along with other types of grass, such as tall grass, ferns, double tall grass, etc.).

Fig. 5: A template for foliage colormaps, created by Rodrigo Al.

Fig. 6: The Vanilla `foliage.png` file, sets the colors for tree leaves (with the exception of spruce and birch).

Note

`blocks=grass` property is not needed since it is in the filename.

In `assets/minecraft/optifine/colormap/blocks/grass.properties`:

```
format=grid
yVariance=2
```

In `assets/minecraft/optifine/colormap/blocks/oak.properties`:

```
format=grid
blocks=oak_leaves
```

20.5 Swamp & badlands colors

Vanilla Minecraft has no support for colormaps on swamp and badlands/mesa biomes. This is intentional, but OptiFine can override this behavior:

Fig. 7: Button and tooltip for the option, found in *Video Settings* → *Quality*.

20.6 Fixing sugar cane in 1.7+

Note

This only applies to 1.7 and above

From 1.7 onward, Minecraft applies the `grass.png` color to sugar cane.

A `fixed` colormap of `ffffff` (white) effectively reverts to the 1.6- behavior. A 256px by 256px all-white colormap would accomplish the same thing, but this method is more efficient.

The simplest way to do this is to create a properties file containing just one line:

In `assets/minecraft/optifine/colormap/blocks/reeds.properties`:

```
format=fixed
```

This works because the blocks property defaults to the filename (reeds) and the color property defaults to `ffffff` for fixed colormaps.

20.7 Other colors

Note

These behave like terrain-based ones, except that they do not care about the blocks property.

Fig. 8: A template for water colormaps, created by Rodrigo Al.

These specifically-named colormaps **override** the default fixed ambient colors:

Key	Meaning
~/colormap/redstone.png	16px x 1px redstone colors (0: fully off, 15: fully on).
~/colormap/pumpkinstem.png	8px x 8px pumpkin stem colors (0: sprout, 7: fully grown).
~/colormap/melonstem.png	8px x 8px melon stem colors (0: sprout, 7: fully grown).
~/colormap/lavadrop.png	<T> px x 1px lava drop colors (<T>: age of particle in ticks).
~/colormap/myceliumparticle.png	Any size, random mycelium particle colors.
~/colormap/xporb.png	Any size, array of experience orb colors.
~/colormap/durability.png	Any size, array of item durability colors.
~/colormap/swampgrass.png	256px x 256px swamp grass color palette.
~/colormap/swampfoliage.png	256px x 256px swamp foliage color palette.
~/colormap/pine.png	256px x 256px **spruce** tree color palette.
~/colormap/birch.png	256px x 256px birch tree color palette.
~/colormap/water.png	256px x 256px water color palette.
~/colormap/underwater.png	256px x 256px underwater color.
~/colormap/underlava.png	256px x 256px underlava color.
~/colormap/fog0.png	256px x 256px fog color for the overworld.
~/colormap/sky0.png	256px x 256px sky color for the overworld.

Each file can have a corresponding properties file to specify the format or other settings.

20.8 Examples

20.8.1 Single block

This is the most simple case. Since a custom colormap is applied to a single block type, **it does not need a properties file**.

For example, `assets/minecraft/optifine/colormap/blocks/sand.png` applies to sand blocks without the need to specify `blocks=sand` in the colormap properties file.

20.8.2 Multiple blocks

Note

The `source` property is unneeded if the colormap is *also* named the same.

To apply the same colormap to all stone and ore blocks, use a properties file:

In `assets/minecraft/optifine/colormap/blocks/stone_and_ore.properties`:

```
blocks=stone gold_ore iron_ore coal_ore lapis_ore diamond_ore redstone_ore redstone_
↪ore:lit=true emerald_ore
```

Use `format=grid` if using the `grid` format.

In `color.properties` this can also be written as:

```
palette.block.~/colormap/custom/stone.png=stone gold_ore iron_ore coal_ore lapis_ore␣
↪diamond_ore redstone_ore redstone_ore:lit=true emerald_ore
```

Add `palette.format=grid` to use grid format for all the custom colormaps (except the vanilla "grass.png" and "foliage.png").

20.9 Custom biome palettes

Note

This does not affect the vanilla foliage and grass colormaps in `/assets/minecraft/textures/colormap`.
This can be overridden per-colormap in each individual properties file in `~/colormap/custom`.

Custom biome palettes may be assigned to any standard block (one that does not already have its own special color multiplier method). Each custom colormap should have a `.properties` file in `~/colormap/custom`.

In Vanilla Minecraft, the grass and leaf textures vary in color depending on the climate of the surrounding biome. This is controlled by two files:

1. `/assets/minecraft/textures/colormap/grass.png`
2. `/assets/minecraft/textures/colormap/foliage.png`

Each file is a 256px x 256px colormap applied to the base grass or leaf texture (which is usually greyscale).

20.10 JSON schema

Note

Although this page is `.properties` based, it can be mapped to JSON.

```
{
        "$schema": "http://json-schema.org/draft/2020-12/schema",
        "$id": "https://gitlab.com/whoatemybutter/optifinedocs/-/blob/master/schemas/
↪colormap.schema.json",
        "title": "Colormap",
        "description": "Colormaps modify a texture's tint based off its biome and height.
↪",
        "type": "object",
        "properties": {
                "format": {
                        "enum": ["grid", "vanilla", "fixed"],
                        "default": "vanilla",
                        "description": "The format to use for this colormap."
                },
                "blocks": {
                        "$ref": "common.schema.json#/$defs/item_id_list",
                        "description": "List of blocks to apply colormap to."
                },
                "source": {
```

(continues on next page)

(continued from previous page)

```
                        "$ref": "common.schema.json#/$defs/resource",
                        "description": "File path to colormap texture."
                },
                "color": {
                        "type": "string",
                        "pattern": "^[0-9a-fA-F]{6}$",
                        "description": "Color will be applied to all matching blocks␣
↪(fixed) or held and dropped blocks (else).",
                        "default": "ffffff"
                },
                "yVariance": {
                        "type": "integer",
                        "default": 0,
                        "description": "Add a random number to the Y coordinate before␣
↪sampling from the colormap. Only for grid."
                },
                "yOffset": {
                        "type": "integer",
                        "default": 0,
                        "description": "Subtracts a fixed value from the block's Y␣
↪coordinate before sampling from the colormap."
                }
        },
        "dependentRequired": {
                "yVariance": ["grid"]
        },
        "allOf": [
                {
                        "if": {
                                "properties": {
                                        "format": {
                                                "enum": [
                                                        "vanilla",
                                                        "grid"
                                                ]
                                        }
                                }
                        },
                        "then": {
                                "not": {
                                        "required": [
                                                "source"
                                        ]
                                }
                        }
                }
        ],
        "additionalProperties": false
}
```

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

CHAPTER

TWENTYONE

CONNECTED TEXTURES

File location

/assets/minecraft/optifine/ctm/**/*.properties

Connected Textures (CTM) connects matching blocks together, making them appear unified.

For each block or terrain tile to override with connected or random textures, create a `.properties` file in the `/assets/minecraft/optifine/ctm` folder of the resource pack. Properties files can be organized into subfolders of any depth.

Fig. 1: The textures appear to "connect".

Fig. 2: Button and tooltip for the option, found in *Video Settings* → *Quality*.

Different types of connected texture methods are available with different requirements and restrictions.

21.1 General properties

Note

See *Paths* before continuing, or else the documentation may be confusing.

Note

`matchTiles` and `matchBlocks` can be omitted if they can be inferred from the filename instead:

- `~/ctm/xxx/<name>.properties` assumes `matchTiles=<name>`
- `~/ctm/xxx/block_<name>.properties` assumes `matchBlocks=<name>` (unless either property is specified explicitly; defined keys override inferred filenames)

21.1.1 method

Values: `ctm`, `ctm_compact`, `horizontal`, `vertical`, `horizontal+vertical`, `vertical+horizontal`, `top`, `random`, `repeat`, `fixed`, `overlay_ctm`, `overlay_random`, `overlay_repeat`, or `overlay_fixed`
Optional

Method to use when choosing a block's replacement texture:

- `ctm`: Standard 8-way method, 47 tiles.
- `ctm_compact`: Compact 8-way method, uses 5 tiles. Cannot be combined with any `overlay` method.
- `horizontal`: Connect to blocks on left and right only.
- `vertical`: Connect to blocks above and below only.
- `horizontal+vertical`: Connect horizontally, then connect vertically.
- `vertical+horizontal`: Connect vertically, then connect horizontally.
- `top`: Connect to block above only.
- `random`: Pick a tile at random.
- `repeat`: Repeat a fixed pattern over large areas.
- `fixed`: Use a single fixed tile, equivalent to `random` with only one tile.
- `overlay`: Overlay for block transitions, uses 17 tiles.
- `overlay_ctm`: Overlay variant of `ctm` method.
- `overlay_random`: Overlay variant of `random` method.
- `overlay_repeat`: Overlay variant of the `repeat` method.
- `overlay_fixed`: Overlay variant of `fixed` method.

Note

The `overlay` method can be combined with other methods if it comes **before** the other methods in the order alphabetically.

Warning

The `ctm_compact` method cannot be combined with any `overlay` method.

21.1.2 tiles

Values: *List* of tiles
Required

List of replacment tiles to use. Each tile must be a separate image, just like terrain and item textures Tiles can be specified in several ways:

- `0` -> `0.png`

- `8-11` -> `8.png, 9.png, 10.png, 11.png`
- `name` -> `name.png`
- `name.png` -> `name.png`
- `full/path/name.png` -> `full/path/name.png`
- <`skip`>: Skip this tile, continue with next CTM properties.
- <`default`>: Use the default texture for that block/tile.

In all cases except the last (<`default`>), the PNG file must be in the same directory as the properties file itself.

The formats can be mixed and matched: `tiles=0-4 5 some/other/name.png`.

Note

The `overlay` methods may use the special name <`skip`> for empty tiles to be skipped; `overlay` methods cannot use the <`default`> special name.

21.1.3 matchTiles

Values: List of strings
Optional

List of tiles this method should apply to.

Multiple `.properties` file can refer to the same block/tile and they will be processed in alphabetical order by filename. All tile-based entries are checked before block ID-based ones; the first match wins.

21.1.4 matchBlocks

Values: *List* of blocks + optional properties
Optional

List of blocks this method should apply to.

To refer to a tile from vanilla Minecraft, simply use its name in `textures/block`: `matchBlocks=dirt` To refer to a tile from a mod, its name must be known: `matchBlocks=botania:blazeblock`

Tiles output by CTM rules can also be matched by another rule; the tile name is simply the full path to the tile: `matchBlocks=optifine/ctm/mygrass/1.png`

Block format: (optional parts are in `[]`) `[namespace:]name[:property1=value1,value2...:property2=value1,value2...]` For example:

- Short name: `oak_stairs`
- Full name: `minecraft:oak_stairs`
- Full name + properties: `minecraft:oak_stairs:facing=east,west:half=bottom`

21.1.5 weight

Values: Integer
Optional
Default: `0`

If multiple properties files match the same block, the highest weighted one is used.

In the event of a tie, the properties filenames are compared next.

21.1.6 connect

Values: `block`, `tile`, `material`, or `state`
Optional
Default: `block` for blocks, `tile` for tiles

The conditions under which two blocks should connect. For methods that connect to adjacent blocks, this rule specifies how the game should decide if two blocks should be connected:

- `block`: Connect if this block's name == neighbor block's name.
- `tile`: Connect if this block's tile texture == neighbor tile's texture.
- `material`: Connect if this block's material (stone, dirt, etc.) == neighbor block's material.
- `state`: Connect if this block's full state (block + properties) == neighbor block's state.

Fig. 3: A diagram of the block matching. From top to bottom, `block`, `material`, `state`.

21.1.7 connectTiles

Values: *List* of tiles
Optional

Connect only to blocks which are using the specified tiles.

Note

This rule only applies to `overlay` methods.

21.1.8 faces

Values: List of strings
Optional

Limit CTM to certain faces of the block:

- `bottom`: Bottom face (negative Y).
- `top`: Top face (positive Y).
- `north`: North face (negative Z).
- `south`: South face (positive Z).
- `east`: East face (positive X).
- `west`: West face (negative X).
- `sides`: Shorthand for `north south east west`.
- `all`: All sides.

Important

This property is ignored on non-cube blocks like signs and fences.

21.1.9 biomes

Values: *List* of biomes
Optional

Biome restrictions. Modded biomes also can be used.

21.1.10 heights

Values: Integer *range*
Optional

Height restriction, no limit. Since 1.18, negative values may be specified for height. When used in a range they have to be put in parenthesis `( )`.

See *Ranges*.

21.1.11 minHeight

Legacy

Legacy property for `heights`.

21.1.12 maxHeight

Legacy

Legacy property for `heights`.

21.1.13 ctm.<ctm_index>

Values: Tile index
Optional

Compact CTM tile replacement. Allows definition of replacement tile for a specific CTM case.

<ctm_index> is the index of the CTM case from the template (`0..46`), Tile index is the index of the tile as defined in `tiles`, **not the tile name**.

With `ctm_compact`, more than 5 tiles can be defined and they can use the additional tiles as replacements.

Important

This rule is only for the `ctm_compact` method.

21.1.14 tintIndex

Values: Integer
Optional
Default: `-1`, disabled

Tint index, only for `overlay` method. Tint index is for the tile's texture. `-1` means it is disabled.

Important

This rule only applies to the `overlay` methods.

21.1.15 tintBlock

Values: Block ID
Optional

The block used for the tile texture tinting.

Different blocks use different colors for the same tint index.

Important

This rule only applies to the `overlay` methods.

21.1.16 layer

Values: String of `cutout_mipped`, `cutout`, or `translucent`
Optional
Default: `cutout_mipped`

The layer on which the overlay texture should be rendered.

Values: * `cutout_mipped`: Transparent textures with mipmaps. * `cutout`: Transparent textures without mipmaps. * `translucent`: Translucent textures with mipmaps.

Important

This rule only applies to the `overlay` methods.

21.1.17 name

Values: *List* of block IDs
Optional

Only for blocks with have corresponding nameable tile entities. Generally, this means containers that can be renamed.

For example: Beacon, Brewing Stand, Enchanting Table, Furnace, Dispenser, Dropper, Hopper, and Command Blocks.

See *Custom GUIs* for the syntax.

21.2 Method properties

21.2.1 `ctm`: standard 8-way

Note

Implies `method=ctm`.

Important

48th tile is unused.

`tiles`

Values: *List* of 47 tiles
Required

List of the 47 tiles to use when connecting.

`innerSeams`

Values: Boolean
Optional
Default: `false`

Whether to show seams on inner edges when connecting to adjacent blocks.

21.2.2 `ctm_compact`: compact 8-way

Note

Implies `method=ctm_compact`,

`tiles`

Values: *List* of 5 tiles
Required

List of the 5 tiles to use when connecting.

`innerSeams`

Values: Boolean
Optional
Default: `false`

Whether to show seams on inner edges when connecting to adjacent blocks.

`ctm.N`

Values: Integer
Optional

Indexes of replacement tiles for specific CTM cases. `N` is a tile index.

Important

This is generally only used for special cases where you want to override the default behavior.

21.2.3 `horizontal`: horizontal only

Note

Implies `method=horizontal`.

`tiles`

Values: *List* of 4 tiles
Required

List of the 4 tiles to use when connecting.

21.2.4 `vertical`: vertical only

Note

Implies `method=vertical`.

`tiles`

Values: *List* of 4 tiles
Required

List of the 4 tiles to use when connecting.

21.2.5 `top`: top face only

Note

Implies `method=top`.

`tiles`

Values: String
Required

The single tile to use when connecting.

21.2.6 `random`: random connect

Note

Implies `method=random`.

`tiles`

Values: *List* of tiles
Required

List of the tiles to use when connecting. Can be infinitely long or short.

weights

Values: *List* of integers
Optional

List of weights to apply to the random choice.

For example, for `tiles=1 2 3 4; weights=10 1 10 5`, tile 1 has weight 10, 2 has 1, 3 has 10, and 4 has 5. Weights do not have to total any value; in the above example, tiles 1 and 3 will each be used ~38% of the time.

Important

This rule must have the same number of elements as the `tiles` rule.

randomLoops

Values: Integer *range* from `0` to `9`
Optional
Default: `0`

Repeats the random function by this amount to increase randomness. Can be used to make different textures use different random patterns.

Warning

A high `randomLoops` value may decrease the chunk loading speed.

symmetry

Values: `none`, `opposite`, or `all`
Optional
Default: `none`

Desired level of symmetry for the faces of each block.

Applies to standard 6-sided blocks only (dirt, glass, not fences).

- `none`: All 6 faces are textured independently.
- `opposite`: 2-way symmetry; opposing faces have the same texture, but each pair can potentially have a different texture.
- `all`: All 6 faces have the same texture.

linked

Values: Boolean
Optional
Default: `false`

Whether to link textures between related blocks.

If true, OptiFine uses the same random number seed for all parts of a multi-block object. For example, the top and bottom halves of tall grass. This allows randomized textures that will remain consistent within each set of blocks.

If false, the two halves will be scrambled, chosen independently. This property currently only applies to plants, double plants (rose bushes, peonys, etc.), and doors.

Important

For `linked` to work, multiple properties files with `linked=true` and the same number of replacement textures and same set of `weights` must be present. For example, `double_plant_top.properties`:

```
method=random
tiles=grass_top1 grass_top2 grass_top3
weights=1 2 3
```

21.2.7 `repeat`: repeated

Note

Implies `method=repeat`.

width

Values: Integer
Required

The width of the repeating pattern.

height

Values: Integer
Required

The height of the repeating pattern.

tiles

Values: *List* of tiles
Required

A list of tiles. The number of elements must equal `width` * `height`.

symmetry

Values: `none`, or `opposite`
Optional
Default: `none`

Desired level of symmetry for the faces of each block.

Applies to standard 6-sided blocks only (dirt, glass, not fences).

- `none`: All 6 faces are textured so that the pattern tiling looks the same from all sides
- `opposite`: 2-way symmetry; opposing faces have the same texture, which means that tiling on the south and east faces will be mirrored left-to-right when comapared to the north and west faces

21.2.8 `fixed`: one texture

Note

Implies `method=fixed`.

tiles

Values: String
Required

Single tile to use.

21.3 JSON schema

Note

Although this page is `.properties` based, it can be mapped to JSON.

```
{
        "$schema": "http://json-schema.org/draft/2020-12/schema",
        "$id": "https://gitlab.com/whoatemybutter/optifinedocs/-/blob/master/schemas/cit.
```

(continues on next page)

(continued from previous page)

```
↪schema.json",
        "title": "Connected Textures",
        "description": "Connected Textures (CTM) connects matching blocks together,␣
↪making them appear unified.",
        "type": "object",
        "properties": {
                "method": {
                        "enum": [
                                "ctm",
                                "ctm_compact",
                                "horizontal",
                                "vertical",
                                "horizontal+vertical",
                                "vertical+horizontal",
                                "top",
                                "random",
                                "repeat",
                                "fixed",
                                "overlay_ctm",
                                "overlay_random",
                                "overlay_repeat",
                                "overlay_fixed"
                        ],
                        "description": "Method to use when choosing a block's␣
↪replacement texture."
                },
                "tiles": {
                        "type": "string",
                        "description": "Space-separated string of replacment tiles to␣
↪use."
                },
                "matchTiles": {
                        "$ref": "common.schema.json#/$defs/item_id_list",
                        "description": "Space-separated string of tiles this method␣
↪should apply to."
                },
                "matchBlocks": {
                        "$ref": "common.schema.json#/$defs/item_id_list",
                        "description": "Space-separated string of blocks this method␣
↪should apply to."
                },
                "weight": {
                        "type": "integer",
                        "default": 0,
                        "description": "If multiple properties files match the same␣
↪block, the highest weighted one is used."
                },
                "connect": {
                        "enum": [
                                "block",
                                "tile",
                                "material",
```

(continues on next page)

(continued from previous page)

```
                                "state"
                        ],
                        "description": "The conditions under which two blocks should␣
↪connect."
                },
                "connectTiles": {
                        "$ref": "common.schema.json#/$defs/item_id_list",
                        "description": "Connect only to blocks which are using the␣
↪specified tiles."
                },
                "faces": {
                        "type": "string",
                        "pattern": "(bottom|top|north|south|east|west|sides|all) ?",
                        "description": "Limit CTM to certain faces of the block."
                },
                "biomes": {
                        "type": "string",
                        "description": "Space-separated string of biome restrictions."
                },
                "heights": {
                        "type": "string",
                        "description": "Height restriction ranges."
                },
                "minHeight": {
                        "type": "integer",
                        "minimum": -65535,
                        "deprecated": true,
                        "description": "Legacy key for heights."
                },
                "maxHeight": {
                        "type": "integer",
                        "maximum": 65535,
                        "deprecated": true,
                        "description": "Legacy key for heights."
                },
                "tintIndex": {
                        "type": "integer",
                        "minimum": -1,
                        "default": -1,
                        "description": "Tint index, only for overlay method."
                },
                "tintBlock": {
                        "$ref": "common.schema.json#/$defs/item_id",
                        "description": "The block used for the tile texture tinting."
                },
                "layer": {
                        "enum": [
                                "cutout_mipped",
                                "cutout",
                                "translucent"
                        ],
                        "default": "cutout_mipped",
```

(continues on next page)

(continued from previous page)

```
                        "description": "The layer on which the overlay texture should be␣
↪rendered."
                },
                "name": {
                        "$ref": "common.schema.json#/$defs/item_id_list",
                        "description": "Only for blocks with have corresponding nameable␣
↪tile entities."
                }
        },
        "patternProperties": {
                "^ctm\\.\\d+$": {
                        "type": [
                                "string",
                                "integer"
                        ],
                        "minimum": 0,
                        "description": "Compact CTM tile replacement. Allows definition␣
↪of replacement tile for a specific CTM case."
                }
        },
        "required": [
                "tiles"
        ],
        "additionalProperties": false,
        "allOf": [
                {
                        "if": {
                                "properties": {
                                        "connectTiles": {}
                                }
                        },
                        "then": {
                                "properties": {
                                        "method": {
                                                "const": "overlay"
                                        }
                                }
                        }
                },
                {
                        "if": {
                                "patternProperties": {
                                        "^ctm\\.\\d+$": {}
                                }
                        },
                        "then": {
                                "properties": {
                                        "method": {
                                                "const": "ctm_compact"
                                        }
                                }
                        }
```

(continues on next page)

(continued from previous page)

```
                },
                {
                        "if": {
                                "properties": {
                                        "tintIndex": {}
                                }
                        },
                        "then": {
                                "properties": {
                                        "method": {
                                                "const": "overlay"
                                        }
                                }
                        }
                },
                {
                        "if": {
                                "properties": {
                                        "layer": {}
                                }
                        },
                        "then": {
                                "properties": {
                                        "method": {
                                                "const": "overlay"
                                        }
                                }
                        }
                },
                {
                        "if": {
                                "properties": {
                                        "method": {
                                                "const": "ctm"
                                        }
                                }
                        },
                        "then": {
                                "properties": {
                                        "innerSeams": {
                                                "type": "boolean",
                                                "default": false,
                                                "description": "Whether to show seams on␣
↪inner edges when connecting to adjacent blocks."
                                        }
                                }
                        }
                },
                {
                        "if": {
                                "properties": {
                                        "method": {
```

(continues on next page)

(continued from previous page)

```
                                                "const": "ctm_compact"
                                        }
                                }
                        },
                        "then": {
                                "properties": {
                                        "innerSeams": {
                                                "type": "boolean",
                                                "default": false,
                                                "description": "Whether to show seams on␣
↪inner edges when connecting to adjacent blocks."
                                        }
                                },
                                "patternProperties": {
                                        "^ctm\\.\\d+$": {
                                                "type": [
                                                        "string",
                                                        "integer"
                                                ],
                                                "minimum": 0,
                                                "description": "Indexes of replacement␣
↪tiles for specific CTM cases."
                                        }
                                }
                        }
                },
                {
                        "if": {
                                "properties": {
                                        "method": {
                                                "const": "random"
                                        }
                                }
                        },
                        "then": {
                                "properties": {
                                        "weights": {
                                                "type": "string",
                                                "description": "Space-separated string␣
↪of weights to apply to the random choice."
                                        },
                                        "randomLoops": {
                                                "type": "integer",
                                                "minimum": 0,
                                                "maximum": 9,
                                                "description": "Repeats the random␣
↪function by this amount to increase randomness."
                                        },
                                        "symmetry": {
                                                "enum": [
                                                        "none",
                                                        "opposite",
```

(continues on next page)

(continued from previous page)

```
                                                        "all"
                                                ],
                                                "default": "none",
                                                "description": "Desired level of␣
↪symmetry for the faces of each block."
                                        },
                                        "linked": {
                                                "type": "boolean",
                                                "default": false,
                                                "description": "Whether to link textures␣
↪between related blocks."
                                        }
                                }
                        }
                },
                {
                        "if": {
                                "properties": {
                                        "method": {
                                                "const": "repeat"
                                        }
                                }
                        },
                        "then": {
                                "properties": {
                                        "width": {
                                                "type": "integer",
                                                "minimum": 1,
                                                "maximum": 16384,
                                                "description": "The width of the␣
↪repeating pattern."
                                        },
                                        "height": {
                                                "type": "integer",
                                                "minimum": 1,
                                                "maximum": 16384,
                                                "description": "The height of the␣
↪repeating pattern."
                                        },
                                        "symmetry": {
                                                "enum": [
                                                        "none",
                                                        "opposite"
                                                ],
                                                "default": "none",
                                                "description": "Desired level of␣
↪symmetry for the faces of each block."
                                        }
                                },
                                "required": [
                                        "width",
                                        "height"
```

(continues on next page)

(continued from previous page)

```
                                ]
                            }
                        }
                ]
}
```

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

CHAPTER

TWENTYTWO

CUSTOM ANIMATIONS

File location

/assets/minecraft/optifine/anim/**/*.properties

Custom Animations allows all textures to be animated, like GUIs and entities.

Important

For block and item textures, including *CTM* and *CIT* replacements, continue using Mojang's `mcmeta` method instead.

Fig. 1: Animation is in a sheet.

In Minecraft 1.5, Mojang added the ability to animate any block or item texture *(originally a feature provided by MCPatcher)*. However, there is yet no way to animate other textures like mob skins or GUIs. OptiFine fills the gap enabling any rectangular area of any non-block or item texture to be animated.

This includes even textures specific to other OptiFine features such as random mob skins or skyboxes.

To build an animation, first choose a texture and determine the X and Y coordinates, and width and height of the area to animate. Create the animation as a vertical strip of frames.

The **width** should be the same as the width of the area to animate. The **height** should be a multiple of the animation area height.

Multiple non-overlapping parts of the same texture can be animated by using the same to value with different `from`, `x`, `y`, `w`, and `h` values. They can even have independent timing and frame order information.

Emissive animation is also possible.

22.1 Properties

Note

`duration`, `interpolate`, `skip`, `tile`, `duration` are optional, rest are **required**

22.1.1 from

Values: String, *path to texture*
Required

Path to source texture of the animation to display.

22.1.2 to

Values: String, *path to texture*
Required

Path to destination texture to replace and animate.

22.1.3 x, y

Values: Positive integers
Required

Coordinates of top-left corner of the **destination** texture to animate to. Normally, this is 0, 0.

22.1.4 w, h

Values: Positive integer
Required

Width and height of an individual animation frame.

22.1.5 duration

Values: Positive integer
Optional

Duration of each individual frame, in ticks. For reference, there are 20 ticks in 1 second.

22.1.6 interpolate

Values: Boolean
Optional

Whether to interpolate between each animated frame.

This furnace has an interpolated animation; focus on the fire inside.

22.1.7 skip

Values: Positive integer
Optional

What frame number to skip/ignore during animation.

Important

Frame numbers start at 0, not 1.

22.1.8 tile.N

Values: Positive integer, `N` is positive integer
Optional

What frame number (starting from 0) to display at the `N`-th tick. N can be greater than the number of frames.

22.1.9 duration.N

Values: Positive integer, `N` is positive integer
Optional

Duration in ticks to display tile `N` for. This only applies to tiles for which a `tile.N` is declared.

22.2 Example

22.2.1 Rainbow squid

```
from=./glow_squid_glow.png
to=textures/entity/squid/glow_squid.png
x=0
y=0
w=64
```

(continues on next page)

(continued from previous page)

```
h=32
duration=1
interpolate=true
skip=2
```

Note

See the *Syntax* document for how to specify paths to texture files.

This creates an interpolating animation that plays each frame in order from top to bottom once for one tick *(1/20th second)* each and then loops infinitely.

22.3 Frame order and timing

Each custom animation may specify its animation speed and frame order. In the properties file, add a series of entries:

```
tile.X=Y
duration.X=Z
```

`X` starts at 0 and represents the order of animation frames based on tick. `Y` is the tile number in the animation frames, the first tile being 0, the second 1, etc. `Z` is the duration that frame should be displayed, in game ticks *(1 tick = 1/20 second)*.

If omitted, duration is assumed to be the default frame duration, or `1` if not configured.

For example, suppose the animation file is 16px x 48px, at 3 frames. To make it run on a 5-frame cycle with a pause in the middle, the properties file might look like this:

```
tile.0=0
tile.1=1
tile.2=2
duration.2=5
tile.3=1
tile.4=0
```

The animation happens in this order:

1. Frame 0: Display animation tile 0 for 1 tick (default duration).
2. Frame 1: Display animation tile 1 for 1 tick (default duration).
3. Frame 2: Display animation tile 2 for 5 ticks (duration=5).
4. Frame 3: Display animation tile 1 for 1 tick (default duration).
5. Frame 4: Display animation tile 0 for 1 tick (default duration).
6. Go back to frame 0.

Total: 5 frames over 9 ticks.

22.4 JSON schema

Note

Although this page is `.properties` based, it can be mapped to JSON.

```
{
        "$schema": "http://json-schema.org/draft/2020-12/schema",
        "$id": "https://gitlab.com/whoatemybutter/optifinedocs/-/blob/master/schemas/
↪custom_animations.schema.json",
        "title": "Custom Animations",
        "description": "Custom Animations allows all textures to be animated, like GUIs␣
↪and entities.",
        "type": "object",
        "properties": {
                "from": {
                        "$ref": "common.schema.json#/$defs/resource",
                        "description": "Path to source texture of the animation to␣
↪display."
                },
                "to": {
                        "$ref": "common.schema.json#/$defs/resource",
                        "description": "Path to destination texture to replace and␣
↪animate."
                },
                "x": {
                        "type": "integer",
                        "minimum": 0,
                        "description": "X coordinate of the top-left corner of the␣
↪destination texture to animate to."
                },
                "y": {
                        "type": "integer",
                        "minimum": 0,
                        "description": "Y coordinate of the top-left corner of the␣
↪destination texture to animate to."
                },
                "w": {
                        "type": "integer",
                        "minimum": 0,
                        "description": "Width of an individual animation frame."
                },
                "h": {
                        "type": "integer",
                        "minimum": 0,
                        "description": "Height of an individual animation frame."
                },
                "duration": {
                        "type": "integer",
                        "minimum": 0,
                        "description": "Duration of each individual frame, in ticks."
```

(continues on next page)

(continued from previous page)

```
                },
                "interpolate": {
                        "type": "boolean",
                        "description": "Whether to unterpolate between each animated␣
↪frame."
                },
                "skip": {
                        "type": "integer",
                        "minimum": 0,
                        "description": "What frame number to skip/ignore during␣
↪animation."
                }
        },
        "patternProperties": {
                "^tile\\.\\d+$": {
                        "type": "integer",
                        "minimum": 0,
                        "description": "What frame number to display at the n-th tick."
                },
                "^duration\\.\\d+$": {
                        "type": "integer",
                        "minimum": 0,
                        "description": "Duration in ticks to display the tile for."
                }
        },
        "additionalProperties": false,
        "required": [
                "from",
                "to",
                "x",
                "y",
                "w",
                "h"
        ]
}
```

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

CHAPTER

TWENTYTHREE

CUSTOM COLORS

File location

/assets/minecraft/optifine/color.properties

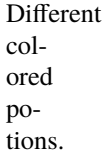

Fig. 1: Different colored potions.

Custom Colors can modify the hardcoded colors for various particles, fogs, and miscellanea.

Values only need to be provided for the properties that need to be changed.

Fig. 2: Button and tooltip for the option, found in *Video Settings* → *Quality*.

The default Minecraft values for each property are given below *for convenience*.

Note

All property names are case~sensitive. All colors are in hexadecimal RGB *(red, green, blue)* format without the #, `000000` to `ffffff`. All paths are relative to `assets/minecraft` unless otherwise stated.

23.1 Properties

23.1.1 Particles

Key	Meaning	Default
`particle.water`	Base water particle color *(splashes, bubbles, drops)*. Biome water color multiplier is applied to this value. The value should match the color of the resource pack's base water texture. If the base water texture is grey, in which coloring is via `misc/watercolor#.png`, this should be set to `ffffff`	`334cff`
`particle.portal`	Base portal particle color. A random multiplier between `0.4` and `1.0` is applied to all three R, G, B values	`ff4ce5`

23.1.2 Fogs, skies

Key	Meaning	Default
`fog.nether`	Fog used in the Nether dimension	`330707`
`fog.end`	Fog used in The End dimension	`181318`
`sky.end`	Color of the sky in The End dimension	`282828`

23.1.3 Lilypads

Key	Meaning	Default
`lilypad`	Single color, used across **all** biomes	`208030`

23.1.4 Potions

For potions with more than 1 effect, the final color is the average of the applicable colors, weighted by the level of each potion effect.

Note

`potion.water` is a plain bottle of water

Key	Default
potion.absorption	2552a5
potion.blindness	1f1f23
potion.confusion	551d4a
potion.damageBoost	932423
potion.digSlowDown	4a4217
potion.digSpeed	d9c043
potion.fireResistance	e49a3a
potion.harm	430a09
potion.heal	f82423
potion.healthBoost	f87d23
potion.hunger	587653
potion.invisibility	7f8392
potion.glowing	94a061
potion.jump	786297
potion.levitation	ceffff
potion.luck	339900
potion.moveSlowdown	5a6c81
potion.moveSpeed	7cafc6
potion.nightVision	1f1fa1
potion.poison	4e9331
potion.regeneration	cd5cab
potion.resistance	99453a
potion.saturation	f82423
potion.unluck	c0a44d
potion.waterBreathing	2e5299
potion.weakness	484d48
potion.wither	352a27
potion.water	385dc6

23.1.5 Spawner egg colors

Fig. 3: Red are the shell, blue are the spots.

Key	Meaning	Default
egg.shell.<entity>	Change the color of the shell of the egg	*None*
egg.spots.<entity>	Change the color of the spots on the egg	*None*

<entity> controls what spawn egg the colors apply to Colons must be escaped: egg.spots.minecraft\:creeper=000000", "*None*"

23.1.6 Map colors

Blocks

Key	Meaning	Default
map.air	Void, unrendered blocks	000000
map.grass	Grass block	7fb238
map.sand	Yellow sand	f7e9a3
map.cloth	Any wool	c7c7c7
map.tnt	TNT block	ff0000
map.ice	Ice, packed ice, blue ice	a0a0ff
map.iron	Iron blocks	a7a7a7
map.foliage	Grass, tall grass, flowers, ferns	007c00
map.clay	Clay	a4a8b8
map.dirt	Dirt, coarse dirt, rooted dirt	976d4d
map.stone	Stone	707070
map.water	Water source, water flowing	4040ff
map.wood	Any planks	8f7748
map.quartz	Any quartz block	fffcf5
map.gold	Gold block	faee4d
map.diamond	Diamond block	5cdbd5
map.lapis	Lapis block	4a80ff
map.emerald	Emerald block	00d93a
map.obsidian	Obsidian block	815631
map.netherrack	Netherrack block	700200

General colors

Key	Default
map.white	ffffff
map.orange	d87f33
map.magenta	b24cd8
map.light_blue	6699d8
map.yellow	e5e533
map.lime	7fcc19
map.pink	f27fa5
map.gray	4c4c4c
map.light_gray	999999
map.cyan	4c7f99
map.purple	7f3fb2
map.blue	334cb2
map.brown	664c33
map.green	667f33
map.red	993333
map.black	191919

Terracotta

Key	Default
map.white_terracotta	d1b1a1
map.orange_terracotta	9f5224
map.magenta_terracotta	95576c
map.light_blue_terracotta	706c8a
map.yellow_terracotta	ba8524
map.lime_terracotta	677535
map.pink_terracotta	a04d4e
map.gray_terracotta	392923
map.light_gray_terracotta	876b62
map.cyan_terracotta	575c5c
map.purple_terracotta	7a4958
map.blue_terracotta	4c3e5c
map.brown_terracotta	4c3223
map.green_terracotta	4c522a
map.red_terracotta	8e3c2e
map.black_terracotta	251610

Nether blocks

Key	Default
map.crimson_nylium	bd3031
map.crimson_stem	943f61
map.crimson_hyphae	5c191d
map.warped_nylium	167e86
map.warped_stem	3a8e8c
map.warped_hyphae	562c3e
map.warped_wart_block	14b485

23.1.7 Sheep coats

Key	Default
sheep.white	e6e6e6
sheep.orange	ba6015
sheep.magenta	953a8d
sheep.light_blue	2b86a3
sheep.yellow	bea22d
sheep.lime	609517
sheep.pink	b6687f
sheep.gray	353b3d
sheep.light_gray	757571
sheep.cyan	107575
sheep.purple	66258a
sheep.blue	2d337f
sheep.brown	623f25
sheep.green	465d10
sheep.red	84221c
sheep.black	151518

23.1.8 Collar colors

Used on wolf and cat collars.

Fig. 4: The red collar on a tamed wolf.

Fig. 5: The red collar on a tamed cat.

Key	Default
collar.white	f9fffe
collar.orange	f9801d
collar.magenta	c74ebd
collar.light_blue	3ab3da
collar.yellow	fed83d
collar.lime	80c71f
collar.pink	f38baa
collar.gray	474f51
collar.light_gray	9d9d97
collar.cyan	169c9c
collar.purple	8932b8
collar.blue	3c44aa
collar.brown	835432
collar.green	5e7c16
collar.red	b02e26
collar.black	1d1d21

23.1.9 Dyes

Base color for banners, beacon beam, tropical fish, wolf and cat collars if unspecified.

Key	Default
dye.white	f9fffe
dye.orange	f9801d
dye.magenta	c74ebd
dye.light_blue	3ab3da
dye.yellow	fed83d
dye.lime	80c71f
dye.pink	f38baa
dye.gray	474f52
dye.light_gray	9d9d97
dye.cyan	169c9c
dye.purple	8932b8
dye.blue	3c44aa
dye.brown	835432
dye.green	5e7c16
dye.red	b02e26
dye.black	1d1d21

23.1.10 Text

Miscellaneous

Key	Meaning	Default
text.xpbar	Experience bar number color	80ff20
text.boss	“Boss Health” text color	ff00ff
text.sign	Sign text color by default	000000

Color codes

Note

Colors below `text.code.15` are for *text shadows*, if enabled in options

Key	Default
text.code.0	000000
text.code.1	0000aa
text.code.2	00aa00
text.code.3	00aaaa
text.code.4	aa0000
text.code.5	aa00aa
text.code.6	ffaa00

continues on next page

Table 1 – continued from previous page

Key	Default
text.code.7	aaaaaa
text.code.8	555555
text.code.9	5555ff
text.code.10	55ff55
text.code.11	55ffff
text.code.12	ff5555
text.code.13	ff55ff
text.code.14	ffff55
text.code.15	ffffff
text.code.16	000000
text.code.17	00002a
text.code.18	002a00
text.code.19	002a2a
text.code.20	2a0000
text.code.21	2a002a
text.code.22	2a2a00
text.code.23	2a2a2a
text.code.24	151515
text.code.25	15153f
text.code.26	153f15
text.code.27	153f3f
text.code.28	3f1515
text.code.29	3f153f
text.code.30	3f3f15
text.code.31	3f3f3f

23.1.11 Resource loading screen

Not to be confused with

Custom Loading Screens

Fig. 6: Red is `screen.loading`. Blue is `screen.loading.outline`. Green is `screen.loading.progress`. Purple is `screen.loading.background`.

Key	Meaning	Default
screen.loading	Background color	ffffff
screen.loading.bar	Loading bar background color	ffffff
screen.loading.outline	Loading bar outline color	000000
screen.loading.progress	Loading bar foreground color	e22837
screen.loading.blend	Logo blending mode	None

screen.loading

Default value

ffffff

Background color of the loading screen.

screen.loading.bar

Default value

ffffff

Loading bar background color. This is behind the progress bar.

screen.loading.outline

Default value

000000

Loading bar outline color. This is the outline around the bar's background.

`screen.loading.progress`

Default value

`e22837`

Loading bar foreground color. This is the progress bar itself.

`screen.loading.blend`

Default value

None for all 4 fields

Logo blending mode. Defined as 4 values split by a space.

Important

It is unclear what specifically these values do. If you know, please make an Issue on the repository!

In order, the fields are `src`, `dst`, `dstA`, and `dstB`.

All of these fields may be any of:

- `ZERO`
- `ONE`
- `SRC_COLOR`
- `ONE_MINUS_SRC_COLOR`
- `DST_COLOR`
- `ONE_MINUS_DST_COLOR`
- `SRC_ALPHA`
- `ONE_MINUS_SRC_ALPHA`
- `DST_ALPHA`
- `ONE_MINUS_DST_ALPHA`
- `SRC_ALPHA_SATURATE`

23.1.12 Other

`clouds`

Overrides cloud type. Must be a single value of either:

- `fast`
- `fancy`
- `none`

`xporb.time`

Default value

`628`

Experience orb animation duration, in milliseconds.

23.2 Aliases

- `map.snow`: `map.white`
- `map.adobe`: `map.orange`
- `map.silver`: `map.light_gray`
- `map.lightBlue`: `map.light_blue`
- `collar.silver`: `collar.light_gray`
- `collar.lightBlue`: `collar.light_blue`
- `dye.silver`: `dye.light_gray`
- `dye.lightBlue`: `dye.light_blue`
- `sheep.silver`: `sheep.light_gray`
- `sheep.lightBlue`: `sheep.light_blue`

23.3 Miscellaneous colormaps

Custom Colors allows the tints of different blocks, entities, and enviornments to be changed with a texture.

Because of their location, you can find this list at *Colormaps* and at *Lightmaps*.

Important

Although Custom Colors manages both of the files in the two separated lists, they are kept separate because of their file location.

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

CHAPTER

TWENTYFOUR

CUSTOM GUIS

File location

/assets/minecraft/optifine/gui/container/*.properties

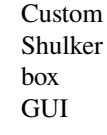

Fig. 1: Custom Shulker box GUI

Custom GUIs can define a texture for each GUI, and apply them based on different criteria, such as the entity, biome, height, and more.

For each container GUI texture to override, create a `.properties` file in the `/assets/minecraft/optifine/gui/container` folder of the resource pack. Properties files can be organized into subfolders of any depth, as long as everything is within the top-level `/assets/minecraft/optifine/gui/container` folder.

Fig. 2: Button and tooltip for the option, found in *Video Settings* → *Quality*.

Important

Different container types have different requirements and restrictions.

24.1 General properties

24.1.1 `container`

Values: `anvil`, `beacon`, `brewing_stand`, `chest`, `crafting`, `dispenser`, `enchantment`, `furnace`, `hopper`, `horse`, `villager`, `shulker_box`, `creative`, or `inventory`
Required
Default: None

Type of container GUI to apply to.

- `creative` refers to the creative inventory with the tabs.
- `inventory` refers to the normal survival inventory, with the player in a window.

24.1.2 texture, texture.PATH

Values: String, *path to texture*
Required
Default: None

The replacing texture for the GUI.

The `texture` property replaces the default GUI texture. The `texture.PATH` property can be used to replace any GUI texture; `PATH` is relative to `/assets/minecraft/textures/gui`.

Important

The creative inventory GUI does not have a default texture, so it must use PATH textures.

Example for creative inventory:

Listing 1: /assets/minecraft/optifine/gui/container/creative/creative_desert.properties

```
container=creative
biomes=desert
texture.container/creative_inventory/tab_inventory=tab_inventory_desert
texture.container/creative_inventory/tabs=tabs_desert
texture.container/creative_inventory/tab_items=tab_items_desert
texture.container/creative_inventory/tab_item_search=tab_item_search_desert
```

Important

At least one `texture` or `texture.PATH` is required.

24.1.3 name

Values: String
Optional
Default: None

Custom entity or block entity name.

This will apply the replacement GUI only when the container matches this rule.

See *Regular expressions* for details.

24.1.4 biomes

Values: *List* of biomes
Optional
Default: None

Biomes where this replacement applies.

Biomes added by mods can also be used with the same syntax.

24.1.5 heights

Values: Integer, or range of integers
Optional
Default: None

Heights where this replacement applies.

Since 1.18, negative values may be specified for height. When used in a range they must be put in parenthesis: `(-3)-64`.

24.2 Specific properties

These additional properties do not need any separate files. They are just properties that apply only when `container` equals their required value.

24.2.1 Chests

Note

Implies `container=chest`.

large

Values: Boolean
Optional

Whether to use the replacement GUI on a large (double) chest.

trapped

Values: Boolean
Optional

Whether to use the replacement GUI on a trapped chest.

christmas

Values: Boolean
Optional

Whether to use the replacement GUI on any Christmas chest. Christmas chests appear from December 24 to 26 of any year.

ender

Values: Boolean
Optional

Whether to use the replacement GUI on an Ender Chest.

24.2.2 Beacons

Note

Implies `container=beacon`.

levels

Values: Integer, or range of integers.
Optional
Default: None

What levels of beacon power to apply the replacement to; how many bases of blocks.

Fig. 3: The levels from left to right: 4, 3, 2, 1.

24.2.3 Villagers

Note

Implies `container=villager`.

professions

Values: `none`, `armorer`, `butcher`, `cartographer`, `cleric`, `farmer`, `fisherman`, `fletcher`, `leatherworker`, `librarian`, `mason`, `nitwit`, `shepherd`, `toolsmith`, or `weaponsmith`, along with an optional level experience format
Optional

List of villager professions with an optional level specifier.

Entry format: `<profession>[:level1,level2,...]`

Examples:

- Professions farmer (all levels) or librarian (levels 1,3,4): `professions=farmer librarian:1,3-4`
- Professions fisher, shepard, nitwit: `professions=fisherman shepherd nitwit`

24.2.4 Horse

Note

Implies `container=horse`.

variants

Values: `horse`, `donkey`, `mule`, `llama`
Optional

What specific horse type to apply replacement to.

24.2.5 Dispenser, dropper

Note

Implies `container=dispenser`. Dropper applies as well.

variants

Values: `dispenser` or `dropper`
Optional
Default: `dispenser`

What specific block to apply the replacement GUI to.

24.2.6 Llama, shulker box

Note

Implies `container=shulker_box` or `container=horse`.

Important

Despite the container being `horse`, this property will apply to Llamas instead.

colors

Values: `white`, `orange`, `magenta`, `light_blue`, `yellow`, `lime`, `pink`, `gray`, `light_gray`, `cyan`, `purple`, `blue`, `brown`, `green`, `red`, `black`
Optional

Shulker box color or llama carpet color to apply the replacement texture to.

24.3 JSON schema

Note

Although this page is `.properties` based, it can be mapped to JSON.

```
{
        "$schema": "http://json-schema.org/draft/2020-12/schema",
        "$id": "https://gitlab.com/whoatemybutter/optifinedocs/-/blob/master/schemas/
↪custom_guis.schema.json",
        "title": "Custom GUIs",
        "description": "Custom GUIs can define a texture for each GUI, and apply them␣
↪based on different criteria, such as the entity, biome, height, and more.",
        "type": "object",
        "properties": {
                "container": {
                        "enum": [
                                "anvil",
```

(continues on next page)

(continued from previous page)

```
                                "beacon",
                                "brewing_stand",
                                "chest",
                                "crafting",
                                "dispenser",
                                "enchantment",
                                "furnace",
                                "hopper",
                                "horse",
                                "villager",
                                "shulker_box",
                                "creative",
                                "inventory"
                        ],
                        "description": "Type of container GUI to apply to."
                },
                "texture": {
                        "$ref": "common.schema.json#/$defs/resource",
                        "description": "The replacing texture for the GUI."
                },
                "name": {
                        "type": "string",
                        "description": "Custom entity or block entity name."
                },
                "biomes": {
                        "type": "string",
                        "description": "Space-separated string of biomes where this␣
↪replacement applies."
                },
                "heights": {
                        "type": [
                                "string",
                                "integer"
                        ],
                        "description": "Heights where this replacement applies."
                }
        },
        "patternProperties": {
                "^texture\\.[/0-9a-z._]+$": {
                        "$ref": "common.schema.json#/$defs/resource",
                        "description": "The replacing texture for the GUI."
                }
        },
        "allOf": [
                {
                        "if": {
                                "properties": {
                                        "container": {
                                                "const": "chest"
                                        }
                                }
                        },
```

(continues on next page)

(continued from previous page)

```
                        "then": {
                                "properties": {
                                        "large": {
                                                "type": "boolean",
                                                "description": "Whether to use the␣
↪replacement GUI on a large chest."
                                        },
                                        "trapped": {
                                                "type": "boolean",
                                                "description": "Whether to use the␣
↪replacement GUI on a trapped chest."
                                        },
                                        "christmas": {
                                                "type": "boolean",
                                                "description": "Whether to use the␣
↪replacement GUI on any Christmas chest."
                                        },
                                        "ender": {
                                                "type": "boolean",
                                                "description": "Whether to use the␣
↪replacement GUI on an Ender Chest."
                                        }
                                }
                        }
                },
                {
                        "if": {
                                "properties": {
                                        "container": {
                                                "const": "beacon"
                                        }
                                }
                        },
                        "then": {
                                "properties": {
                                        "levels": {
                                                "type": [
                                                        "string",
                                                        "integer"
                                                ],
                                                "minimum": 1,
                                                "maximum": 4,
                                                "description": "What levels of beacon␣
↪power to apply the replacement to; how many bases of blocks."
                                        }
                                }
                        }
                },
                {
                        "if": {
                                "properties": {
                                        "container": {
```

(continues on next page)

(continued from previous page)

```
                                                "const": "villager"
                                        }
                                }
                        },
                        "then": {
                                "properties": {
                                        "professions": {
                                                "type": "string",
                                                "pattern":
↪"(none|armorer|butcher|cartographer|cleric|farmer|fisherman|fletcher|leatherworker|librarian|mason|ni
↪\d+(-\\d+)?(,\\d+(-\\d+)?)*)",
                                                "description": "Space-separated string␣
↪of villager professions with an optional level specifier."
                                        }
                                }
                        }
                },
                {
                        "if": {
                                "properties": {
                                        "container": {
                                                "const": "horse"
                                        }
                                }
                        },
                        "then": {
                                "properties": {
                                        "variants": {
                                                "enum": [
                                                        "horse",
                                                        "donkey",
                                                        "mule",
                                                        "llama"
                                                ],
                                                "description": "What specific horse type␣
↪to apply replacement to."
                                        },
                                        "colors": {
                                                "enum": [
                                                        "white",
                                                        "orange",
                                                        "magenta",
                                                        "light_blue",
                                                        "yellow",
                                                        "lime",
                                                        "pink",
                                                        "gray",
                                                        "light_gray",
                                                        "cyan",
                                                        "purple",
                                                        "blue",
                                                        "brown",
```

(continues on next page)

(continued from previous page)

```
                                                        "green",
                                                        "red",
                                                        "black"
                                                ],
                                                "description": "Llama carpet color to␣
↪apply the replacement texture to."
                                        }
                                }
                        }
                },
                {
                        "if": {
                                "properties": {
                                        "container": {
                                                "const": "dispenser"
                                        }
                                }
                        },
                        "then": {
                                "properties": {
                                        "variants": {
                                                "enum": [
                                                        "dispenser",
                                                        "dropper"
                                                ],
                                                "default": "dispenser",
                                                "description": "What specific block to␣
↪apply the replacement GUI to."
                                        }
                                }
                        }
                },
                {
                        "if": {
                                "properties": {
                                        "container": {
                                                "const": "shulker_box"
                                        }
                                }
                        },
                        "then": {
                                "properties": {
                                        "colors": {
                                                "enum": [
                                                        "white",
                                                        "orange",
                                                        "magenta",
                                                        "light_blue",
                                                        "yellow",
                                                        "lime",
                                                        "pink",
                                                        "gray",
```

(continues on next page)

(continued from previous page)

```
                                                        "light_gray",
                                                        "cyan",
                                                        "purple",
                                                        "blue",
                                                        "brown",
                                                        "green",
                                                        "red",
                                                        "black"
                                                ],
                                                "description": "Shulker box color to␣
↪apply the replacement texture to."
                                        }
                                }
                        }
                },
                {
                        "if": {
                                "properties": {
                                        "colors": {}
                                }
                        },
                        "then": {
                                "properties": {
                                        "variants": {
                                                "enum": [
                                                        "shulker_box",
                                                        "llama"
                                                ]
                                        }
                                },
                                "required": [
                                        "colors"
                                ]
                        }
                }
        ],
        "required": [
                "container"
        ],
        "additionalProperties": false
}
```

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

CHAPTER

TWENTYFIVE

CUSTOM LIGHTMAPS

File location

/assets/minecraft/optifine/lightmap/**/*.png

Custom Lightmaps can change the color of light from light sources. OptiFine allows customizing the lighting in-game.

Fig. 1: Daylight lightmap.

25.1 Vanilla lighting

Every block has two light values from `0` to `15` assigned to it, one for sky brightness and one for torch brightness.

A block in direct sunlight has a sky value of `15`. A block in the shade directly adjacent to it has a value of `14` and so on.

Blocks deep underground far from any block that can see the sky have sky brightness `0`. Similarly for torches. A torch block has light value `14` *(15 for glowstone)* and the light value drops by `1` each block away from it, in a diamond shape.

Fig. 2: A visualization of the light levels a torch gives off.

To generate the lighting actually seen in game, Minecraft uses a 16px x 16px lightmap image. The image's axes correspond to the 16 light levels of each type. If a block has torch brightness `x` and sky brightness `y`, then the point (`x`, `y`) is used for its lightmap.

Important

The lightmap is not in any of the game's assets.

Two variables affect the lightmap: the time of day, and the torch flicker. Minecraft implements dusk/dawn transitions and torch flicker by making the entire lightmap darker or lighter as a whole. rather than by adjusting the sky/torch brightness values.

25.2 Other lightmaps

To create custom lighting, a lightmap palette needs to be created for each world:

- Nether: `/assets/minecraft/optifine/lightmap/world-1.png`
- Overworld: `/assets/minecraft/optifine/lightmap/world0.png`
- The End: `/assets/minecraft/optifine/lightmap/world1.png`

For the overworld, **optional** rain and thunder palettes may also be specified:

- Overworld rain: `/assets/minecraft/optifine/lightmap/world0_rain.png`
- Overworld thunder: `/assets/minecraft/optifine/lightmap/world0_thunder.png`

The rain and thunder palettes are only active when the main Overworld palette is defined.

Each palette can be any width, but must be 32 or 64 pixels tall. If it's 64, the bottom half is used for night vision; see *Night vision*.

Of the 32 rows of pixels, the top 16 represent sunlight and the bottom 16 represent torchlight.

Two columns (16 pixels from the top half, and 16 pixels from the bottom half) are chosen to form the axes of the final 16px x 16px lightmap used.

Fig. 3: Blue: Night. Orange: Dusk/dawn. Cyan: Day. Yellow: Lightning.

In the top half, the left-hand side represents night and the right-hand side represents day, with the dusk/dawn transitions in between. The very far right of the palette represents lightning flashes.

Again, there is no specified width for the palette, but more width means more room for detail in the transitions.

Torches work similarly, but in this case the `x` coordinate is simply a random value simulating torch flicker. The variation along the `x` dimension will determine how noticable torch flicker is. To have completely steady torchlight with no flicker, make all pixels along each row the same color.

Lightmaps work the same in all three worlds, but since there is no night or day in Nether and The End, the "time of day" value is constant. For these worlds, simply give rows `0` through `15` the same color all the way across.

25.3 Night vision

In Vanilla, the night vision effect is computed by scaling the RGB values by `1.0 / max(R,G,B)`. For example, `(0.2, 0.3, 0.6)` would brighten to `(0.333, 0.5, 1.0)` after dividing by `0.6`.

This behaviour can be overridden with a custom lightmap by making the height 64 pixels instead of 32. Provide four palettes instead of two: normal sun, normal torch, night vision sun, and night vision torch.

Lightmap generation works exactly the same way but uses rows 32-47 and 48-63 instead.

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

CHAPTER

TWENTYSIX

CUSTOM LOADING SCREENS

Custom Loading Screens define the screen when changing worlds, loading a world, starting the game, or reloading datapacks.

File location

/assets/minecraft/optifine/gui/loading/loading.properties

Controls the behaviour of the world loading screen.

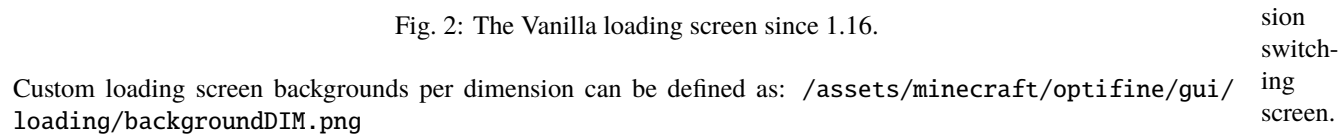

Fig. 2: The Vanilla loading screen since 1.16.

Fig. 1: The dimension switching screen.

Custom loading screen backgrounds per dimension can be defined as: `/assets/minecraft/optifine/gui/loading/backgroundDIM.png`

Where `DIM` is the dimension ID:

- 1: End
- 0: Overworld
- -1: Nether

Note

Modded dimensions can also be configured in this way.

26.1 Properties

Note

The properties `scaleMode`, `scale`, and `center` can also be configured per dimension:

```
dim<dim>.scaleMode=<fixed|full|stretch>
dim<dim>.scale=2
dim<dim>.center=<true|false>
```

26.1.1 scaleMode

Values: `fixed`, `full`, or `stretch`
Optional
Default: `fixed`

Custom scale mode for the background texture:

- `fixed`: use fixed scale, pixel for pixel *(default)*.
- `full`: fullscreen, keep aspect ratio.
- `stretch`: fullscreen, stretch picture.

26.1.2 scale

Values: Integer
Optional
Default: `scaleMode=fixed`: `2`, `scaleMode=full`: `1`

Custom scale for the background texture.

For scale mode `fixed`, it defines the pixel scale to use. This is combined with the curent GUI scale.

For scale modes `full` and `stretch`, it defines how many tiled textures should fit on the screen.

26.1.3 center

Values: Boolean
Optional
Default: `false`

Defines if the background texture should be centered on the screen.

26.2 JSON schema

Note

Although this page is `.properties` based, it can be mapped to JSON.

```
{
        "$schema": "http://json-schema.org/draft/2020-12/schema",
        "$id": "https://gitlab.com/whoatemybutter/optifinedocs/-/blob/master/schemas/
↪custom_loading_screens.schema.json",
        "title": "Custom Loading Screens",
        "description": "Custom Loading Screens define the screen when changing worlds,␣
↪loading a world, starting the game, or reloading datapacks.",
```

(continues on next page)

(continued from previous page)

```
        "type": "object",
        "properties": {},
        "additionalProperties": false,
        "patternProperties": {
                "^(dim-?[0-9]+\\.)?scaleMode$": {
                        "enum": [
                                "fixed",
                                "full",
                                "stretch"
                        ],
                        "description": "Custom scale mode for the background texture.",
                        "default": "fixed"
                },
                "^(dim-?[0-9]+\\.)?scale$": {
                        "type": "integer",
                        "minimum": 0,
                        "description": "Custom scale for the background texture.",
                        "default": 1
                },
                "^(dim-?[0-9]+\\.)?center$": {
                        "type": "boolean",
                        "description": "Defines if the background texture should be␣
↪centered on the screen.",
                        "default": false
                }
        }
}
```

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

CHAPTER

TWENTYSEVEN

CUSTOM PANORAMAS

File location

/assets/minecraft/optifine/gui/background.properties

Custom Panoramas control the behaviour of the main menu panorama.

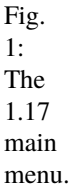

Fig. 1: The 1.17 main menu.

27.1 Alternative panorama folders

Note

This is optional.

Alternative panorama folders can include a `background.properties` file to define custom properties for each panorama.

For example:

```
/assets/minecraft/optifine/gui/background1
    /panorama_0.png
    /panorama_1.png
    /panorama_2.png
    /panorama_3.png
    /panorama_4.png
    /panorama_5.png
```

27.2 Properties

27.2.1 `weight`

Values: Integer
Optional
Default: 1

Weights of selections, in descending order; **higher** weights will be selected more often.

27.2.2 Blurs

The main menu background uses **3** types of blur prior to 1.12.

Warning

Higher blur levels may decrease the main menu FPS.

Danger

This feature does not work past 1.12.

`blur1`

Values: Integer, `1` through `64`
Optional
Default: `1`

`blur2`

Values: Integer, `1` through `3`
Optional
Default: `1`

`blur3`

Values: Integer, `1` through `3`
Optional
Default: `1`

27.3 Overlay colors

Note

When the top and bottom colors are both `0`, that overlay is disabled.

If enabled, two gradient overlays can be drawn on top of the background panorama.

The color format is **ARGB** (alpha [transparency], red, green, blue), in hexadecimal.

To read it, convert each interval of two values to decimal from hexadecimal (`AABBCCDD` == `0xAA`, `0xBB`, `0xCC`, `0xDD` == `170`, `187`, `204`, `221`).

27.3.1 overlay1.top

Values: AARRGGBB
Optional
Default: `80FFFFFF`

First overlay, top gradient color.

27.3.2 overlay1.bottom

Values: AARRGGBB
Optional
Default: `00FFFFFF`

First overlay, bottom gradient color.

27.3.3 overlay2.top

Values: AARRGGBB
Optional
Default: `00000000`

Second overlay, top gradient color.

27.3.4 overlay2.bottom

Values: AARRGGBB
Optional
Default: `80000000`

Second overlay, bottom gradient color.

27.4 JSON schema

Note

Although this page is `.properties` based, it can be mapped to JSON.

```
{
        "$schema": "http://json-schema.org/draft/2020-12/schema",
        "$id": "https://gitlab.com/whoatemybutter/optifinedocs/-/blob/master/schemas/
↪custom_panoramas.schema.json",
```

(continues on next page)

(continued from previous page)

```
        "title": "Custom Panoramas",
        "description": "Custom Panoramas control the behaviour of the main menu panorama.
↪",
        "type": "object",
        "properties": {
                "weight": {
                        "type": "integer",
                        "minimum": 0,
                        "default": 1,
                        "description": "Weights of selections, in descending order;␣
↪higher weights will be selected more often."
                },
                "blur1": {
                        "type": "integer",
                        "minimum": 1,
                        "maximum": 64,
                        "default": 1
                },
                "blur2": {
                        "type": "integer",
                        "minimum": 1,
                        "maximum": 3,
                        "default": 1
                },
                "blur3": {
                        "type": "integer",
                        "minimum": 1,
                        "maximum": 3,
                        "default": 1
                },
                "overlay1.top": {
                        "type": "string",
                        "pattern": "^[0-9a-fA-F]{8}$",
                        "default": "80FFFFFF",
                        "description": "First overlay, top gradient color."
                },
                "overlay1.bottom": {
                        "type": "string",
                        "pattern": "^[0-9a-fA-F]{8}$",
                        "default": "00FFFFFF",
                        "description": "First overlay, bottom gradient color."
                },
                "overlay2.top": {
                        "type": "string",
                        "pattern": "^[0-9a-fA-F]{8}$",
                        "default": "00FFFFFF",
                        "description": "Second overlay, top gradient color."
                },
                "overlay2.bottom": {
                        "type": "string",
                        "pattern": "^[0-9a-fA-F]{8}$",
                        "default": "00000000",
```

(continues on next page)

(continued from previous page)

```
                        "description": "Second overlay, bottom gradient color."
                }
        },
        "additionalProperties": false
}
```

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

CHAPTER

TWENTYEIGHT

CUSTOM SKY

File location

/assets/minecraft/optifine/sky/world*/*.properties /assets/minecraft/optifine/sky/**/*.png

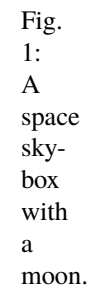

Fig. 1: A space skybox with a moon.

Custom Sky changes the skybox texture and can apply different sets of skies depending on the time, biome, heights, weather, and more.

Place the file in:

- `~/sky/world0/` for Overworld.
- `~/sky/world1/` for End. (since version G9)
- `~/sky/world-1/` for Nether.

in the resource pack.

Each file represents one layer of the sky. OptiFine will load them in order by their number, applying one on top of the previous.

Additionally, two special properties files are applied to the sun and moon if present. This is mainly intended to allow overriding the blend method used by the sun and moon:

- `~/sky/world0/sun.properties`: replaces `sun.png`.
- `~/sky/world0/moon_phases.properties`: replaces `moon_phases.png`.

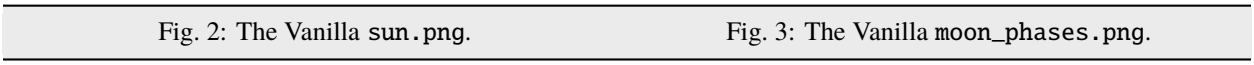

Fig. 2: The Vanilla `sun.png`.

Fig. 3: The Vanilla `moon_phases.png`.

Instead of a full skybox, the source texture should match the layout of `sun.png` or `moon_phases.png`.

Important

The "world0" in the path refers to the overworld. If there were other worlds with skies *(the Nether and End do not use the standard sky rendering# methods)*, their files would be in `~/sky/world<world number>`.

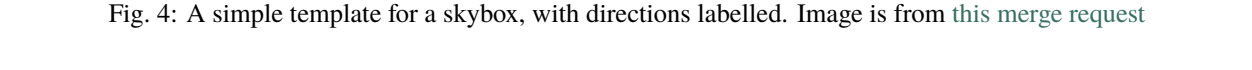

Fig. 4: A simple template for a skybox, with directions labelled. Image is from this merge request

Fig. 5: An example of how a skybox might look.

28.1 Properties

Note

`speed` does not affect the fading in and out; this always occurs on a 24-hour cycle.

28.1.1 `startFadeIn`

Values: String, hh:mm 24-hour format
Optional

Fade in/out times. All times are in hh:mm 24-hour format. See *Time format*.

If no times are specified, the layer is always rendered.

28.1.2 `endFadeIn`

Values: See above
Optional
Default: None

28.1.3 `endFadeOut`

Values: See above
Optional
Default: None

28.1.4 `source`

Values: String, *path to texture*
Optional
Default: `skyN.png` in same directory, N is properties' file name N.

Path to source texture.

Multiple properties files can reuse the same source.

28.1.5 blend

Values: add, subtract, multiply, dodge, burn, screen, replace, overlay, or alpha
Optional
Default: add

The blending method between fading of layers.

Here, 'previous layer' can refer to the default sky or to the previous custom sky defined by sky<N-1>.properties. Supported blending methods are:

- add: Add this sky bitmap to the previous layer. In case of overflow, white is displayed.
- subtract: Subtracts this sky bitmap to the previous layer. In case of negative values, black is displayed.
- multiply: Multiply the previous RGBA values by the RGBA values in the current bitmap.
- dodge: Lightens the sky bitmap.
- burn: Darkens the sky bitmap.
- screen: Inverts both layers, multiplies them, and then inverts that result.
- replace: Replace the previous layer entirely with the current bitmap. There is no gradual fading with this method; if brightness computed from the fade times is >0, the full pixel value is used.
- overlay: RGB value > 0.5 brightens the image, < 0.5 darkens.
- alpha: Weighted average by alpha value.

28.1.6 rotate

Values: Boolean
Optional
Default: true

Whether or not the source texture should rotate with the time of day.

28.1.7 speed

Values: Positive float
Optional
Default: 1.0

Rotation speed as a multiple of the default of one 360-degree cycle per game day.

A value of 0.5 rotates every two days.

Info

Irrational values can be useful to make clouds appear in different positions each day.

28.1.8 axis

Values: *List* of three floats
Optional
Default: `0.0 0.0 1.0`

The axis of rotation of the skybox. If a player is looking in the given direction, the skybox will appear to be rotating clockwise around the line of sight.

Default rotation is along the southern axis (rising in the east and setting in the west).

For reference, the vectors corresponding to the six cardinal directions are below. However, the rotation axis can be any vector except `0.0 0.0 0.0`:

```
down  =  0 -1  0
up    =  0  1  0
north =  0  0 -1
south =  0  0  1
west  = -1  0  0
east  =  1  0  0
```

28.1.9 days

Values: *List* of integers, or integer *ranges*
Optional

The days for which the layer is to be rendered.

Days are numbered from 0 to `daysLoop`-1, for example: `days=0 2-4 6`.

28.1.10 daysLoop

Values: Positive integer
Optional
Default: `8`

Number of days in a loop, see above.

28.1.11 weather

Values: `clear`, `rain`, or `thunder`
Optional
Default: `clear`

Under what weather for which the layer is to be rendered. Several values can be specified separated by spaces, for example `weather=clear rain thunder`.

28.1.12 biomes

Values: *List* of biome IDs
Optional
Default: None

Limit the sky to only certain biomes.

28.1.13 heights

Values: *List* of integers or integer *range*
Optional

Limit the sky to only certain heights.

Since 1.18, negative values may be specified for height. When used in a range they have to be put in parenthesis `( )`: `(-3)-64`.

28.1.14 transition

Values: Integer
Optional
Default: 1

Transition time *(in seconds)* for the layer brightness. It is used to smooth sharp transitions, for example between different biomes.

28.2 Time format

All times are in hh:mm 24-hour format.

For reference,

Sunrise	6:00	`/time set 0`
Noon	12:00	`/time set 6000`
Sunset	18:00	`/time set 12000`
Midnight	0:00	`/time set 18000`

The fade times control the brightness when blending.

- between startFadeIn and endFadeIn: 0 up to 1
- between endFadeIn and startFadeOut: always 1
- between startFadeOut and endFadeOut: 1 down to 0
- between endFadeOut and startFadeIn: always 0

Important

`startFadeOut` does not need to be specified; its value is uniquely determined by the other three.

28.3 Blender model

Note

This Blender model was contributed by usernamegeri.

You can generate the vector coordinates that are required for the axis of rotation of the skybox by using a pre-made tool for Blender. It can be found `here`.

28.4 JSON schema

Note

Although this page is `.properties` based, it can be mapped to JSON.

```
{
        "$schema": "http://json-schema.org/draft/2020-12/schema",
        "$id": "https://gitlab.com/whoatemybutter/optifinedocs/-/blob/master/schemas/
↪custom_sky.schema.json",
        "title": "Custom Sky",
        "description": "Custom Sky changes the skybox texture and can apply different␣
↪sets of skies depending on the time, biome, heights, weather, and more.",
        "type": "object",
        "properties": {
                "startFadeIn": {
                        "type": "string",
                        "pattern": "^\\d{1,2}:\\d{1,2}$",
                        "description": "Start time to fade in."
                },
                "endFadeIn": {
                        "type": "string",
                        "pattern": "^\\d{1,2}:\\d{1,2}$",
                        "description": "End time to fade in."
                },
                "endFadeOut": {
                        "type": "string",
                        "pattern": "^\\d{1,2}:\\d{1,2}$",
                        "description": "End time to fade out."
                },
                "source": {
                        "$ref": "common.schema.json#/$defs/resource",
                        "description": "Path to source texture."
```

(continues on next page)

(continued from previous page)

```
                },
                "blend": {
                        "$ref": "common.schema.json#/$defs/blending_method_enum",
                        "default": "add",
                        "description": "The blending method between fading of layers."
                },
                "rotate": {
                        "type": "boolean",
                        "default": true,
                        "description": "Whether or not the source texture should rotate␣
↪with the time of day."
                },
                "speed": {
                        "type": "number",
                        "minimum": 0.0,
                        "default": 1.0,
                        "description": "Rotation speed as a multiple of the default of␣
↪one 360-degree cycle per game day."
                },
                "axis": {
                        "type": "string",
                        "pattern": "^-?\\d+(\\.\\d+)? -?\\d+(\\.\\d+)? -?\\d+(\\.\\d+)?$
↪",
                        "default": "0.0 0.0 1.0",
                        "description": "The axis of rotation of the skybox."
                },
                "days": {
                        "type": "string",
                        "description": "The days for which the layer is to be rendered."
                },
                "daysLoop": {
                        "type": "integer",
                        "minimum": 0,
                        "default": 8,
                        "description": "Number of days in a loop."
                },
                "weather": {
                        "enum": ["clear", "rain", "thunder"],
                        "default": "clear",
                        "description": "Under what weather for which the layer is to be␣
↪rendered."
                },
                "biomes": {
                        "type": "string",
                        "description": "Limit the sky to only certain biomes."
                },
                "heights": {
                        "type": "string",
                        "description": "Limit the sky to only certain heights."
                },
                "transition": {
                        "type": "integer",
```

(continues on next page)

(continued from previous page)

```
                        "default": 1,
                        "minimum": 0,
                        "description": "Transition time in seconds for the layer␣
↪brightness."
                }
        },
        "additionalProperties": false
}
```

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

CHAPTER

TWENTYNINE

DYNAMIC LIGHTS

File location

/assets/minecraft/optifine/dynamic_lights.properties /assets/<MOD>/optifine/dynamic_lights.properties

Dynamic Lights allows hand-held and dropped light-emitting items such as torches to illuminate the blocks around them in the world.

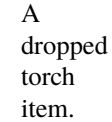

Fig. 1: A dropped torch item.

Fig. 2: Button and tooltip for the option, found in *Video Settings* → *Quality*.

Fig. 3: An example of dynamic lighting; the held-item torch illuminates the blocks around the player, and the dropped glowstone item also does the same.

This configuration file allows **mods** to define dynamic light levels for entities and items.

29.1 Properties

29.1.1 `entities`

Values: String, "<entity:light level>"
Optional

Entity light levels. The entity name is automatically expanded with the mod's ID, if applicable.

The light level should be between `0` and `15`. For example: `entities=basalz:15 blitz:7`.

Important

This does not work for `minecraft:` entities.

29.1.2 items

Values: String, <item:light level>
Optional

Item light levels. The item name is automatically expanded with the mod's ID, if applicable.

The light level should be between `0` and `15`. For example: `items=florb:15 morb:7`.

Important

This does not work for `minecraft:` items.

29.2 JSON schema

Note

Although this page is `.properties` based, it can be mapped to JSON.

```
{
        "$schema": "http://json-schema.org/draft/2020-12/schema",
        "$id": "https://gitlab.com/whoatemybutter/optifinedocs/-/blob/master/schemas/
↪dynamic_lights.schema.json",
        "title": "Dynamic Lights",
        "description": "Dynamic Lights allows hand-held and dropped light-emitting items␣
↪such as torches to illuminate the blocks around them in the world.",
        "type": "object",
        "properties": {
                "entities": {
                        "type": "string",
                        "pattern": "(.*?:\\d{1,2}) ?",
                        "description": "Entity light levels."
                },
                "items": {
                        "type": "string",
                        "pattern": "(.*?:\\d{1,2}) ?",
                        "description": "Item light levels."
                }
        },
        "additionalProperties": false
}
```

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

CHAPTER

THIRTY

EMISSIVE TEXTURES

File location

/assets/minecraft/optifine/emissive.properties

Emissive Textures add a second texture on top of a block which will render with no darkness and will be unaffected by lightmaps.

Important

They do not change the actual lighting around them. This does not affect actual lighting, just the pixel's brightness.

It is possible to add overlays to block textures, which will always rendered with full brightness. This can simulate light emitting parts of the textures.

Fig. 1: Emissive diamond ore.

Fig. 2: An example of emissive ores in a dark water cave.

Fig. 3: Button and tooltip for the option, found in *Video Settings* → *Quality*.

The emissive overlays have the same name as the base texture + custom suffix. For example:

- `bedrock.png`: base texture
- `bedrock_e.png`: emissive overlay

The emissive overlays are rendered in the same block layer as the base texture, except overlays for textures from layer `SOLID`, which are rendered as `CUTOUT_MIPPED`. The overlays can also be used for items, mobs and block entities.

30.1 Properties

30.1.1 `suffix.emissive`

Values: String
Optional
Default: _e

The suffix a file must have to be registered as emissive.

30.2 Armor trims

Emissive armor trim textures are defined by adding one of the following material suffixes to the trim base name:

- `amethyst`
- `copper`
- `diamond`
- `diamond_darker`
- `emerald`
- `gold`
- `gold_darker`
- `iron`
- `iron_darker`
- `lapis`
- `netherite`
- `netherite_darker`
- `quartz`
- `redstone`

For example (if `suffix.emissive` is `_e`):

```
coast_amethyst_e.png, host_iron_darker_e.png, dune_leggings_netherite_e.png
```

30.3 JSON schema

Note

Although this page is `.properties` based, it can be mapped to JSON.

```
{
        "$schema": "http://json-schema.org/draft/2020-12/schema",
        "$id": "https://gitlab.com/whoatemybutter/optifinedocs/-/blob/master/schemas/
↪emissive_textures.schema.json",
        "title": "Emissive Textures",
        "description": "Emissive Textures add a second texture on top of a block which␣
↪will render with no darkness and will be unaffected by lightmaps.",
        "type": "object",
        "properties": {
                "suffix.emissive": {
                        "type": "string",
                        "default": "_e",
                        "description": "The suffix a file must have to be registered as␣
↪emissive."
                }
```

(continues on next page)

(continued from previous page)

```
        },
        "additionalProperties": false
}
```

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

CHAPTER

THIRTYONE

HD FONTS

File location

/assets/minecraft/optifine/font/**/*.{png,properties}

Danger

This feature is obsolete. **Do not use it.** Minecraft's font system has fixed the issues HD fonts was created to resolve.

Warning

Characters outside the ASCII range are not supported.

Fig. 1: The default font texture snippet.

HD Fonts can define custom widths for characters.

OptiFine first looks for fonts in the `assets/minecraft/optifine/font` folder. This allows having a custom font that works in vanilla and a higher resolution font that requires OptiFine to display properly.

- Default font: `assets/minecraft/optifine/font/ascii.png`
- SGA *(enchanting)* font: `assets/minecraft/optifine/font/ascii_sga.png`

To allow for more control over the widths of individual characters, OptiFine offers a way to specify them manually. Create a properties file corresponding to the font to customize:

- `/assets/minecraft/optifine/font/ascii.properties`
- `/assets/minecraft/optifine/font/ascii_sga.properties`

31.1 Properties

Each line in this file specifies the width of a character:

Note

Each character must be in it's **decimal** codepoint, not hexadecimal; this is why `SPACE U+0020` is `32`.

```
# Custom width
width.<ascii value 0-255>=<width 0-8>
```

Fig. 2: The ASCII table

For example, to specify the widths of capital A, B, and C, the following might be used:

```
# A
width.65=5.9

# B
width.66=5

# C
width.67=5.25
```

Values can be floating point numbers (Vanilla only supports integer widths) and range from 0-8 regardless of the resolution of the font. The widths for all characters do not need to be specified, only those that should override the default width.

The space character default width is `4.0`. This can be overridden by setting `width.32` to a custom value.

31.2 JSON schema

Note

Although this page is `.properties` based, it can be mapped to JSON.

```
{
        "$schema": "http://json-schema.org/draft/2020-12/schema",
        "$id": "https://gitlab.com/whoatemybutter/optifinedocs/-/blob/master/schemas/hd_
↪fonts.schema.json",
        "title": "HD Fonts",
        "description": "HD Fonts can define custom widths for characters.",
        "type": "object",
        "patternProperties": {
                "^width.(?:\\b(?:25[0-5]|2[0-4][0-9]|[01]?[0-9]{1,2})\\b\\s?)+$": {
                        "type": "number",
                        "minimum": 0,
                        "maximum": 8,
                        "description": "The width of a character.",
                        "deprecated": true
                }
        },
        "additionalProperties": false,
        "deprecated": true
}
```

Assumes latest OptiFine version.
Updated to commit 8ed2130d.

Open source at https://gitlab.com/whoatemybutter/optifinedocs.

CHAPTER

THIRTYTWO

LAGOMETER

The lagometer is a feature in OptiFine that displays a graph of the resources in rendering each frame. It is visible only when the debug screen (`F3`) is shown.

It replaces the Vanilla frame time graph.

Fig. 1: , , , ,

Fig. 2: Button and tooltip for the option, found in *Video Settings* → *Other*.

Fig. 3: A usual meter with mostly white

C	Meaning
Orange	Memory garbage collection
Cyan	Tick
Blue	Scheduled executables
Purple	Chunk upload
Red	Chunk updates
Yellow	Visibility check
Green	Render terrain
White	Anything not in above categories

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

CHAPTER

THIRTYTHREE

NATURAL TEXTURES

File location

/assets/minecraft/optifine/natural.properties

Natural Textures randomly rotate a block's texture based off of its position.

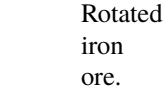

Fig. 1: Rotated iron ore.

Fig. 2: Button and tooltip for the option, found in *Video Settings* → *Quality*.

Fig. 3: An wall of iron ore, with the two different *(rotated)* corners circled in red.

Values:

- `4`: Rotate x 90° (4 variants).
- `2`: Rotate x 180° (2 variants).
- `F`: Flip texture horizontally (2 variants).
- `4F`: 4 + Flip (8 variants).
- `2F`: 2 + Flip (4 variants)

Important

This list is the default, but any block texture can be used.

Listing 1: Example for obsidian texture, which can rotate with 4 variants and flip: `obsidian = 4F`

```
# Grass
grass_block_side = F
grass_block_side_overlay = F
grass_block_snow = F
mycelium_side = F
mycelium_top = 4F
dirt_path_top = 4
dirt_path_side = F

# Snow
snow = 4F
```

(continues on next page)

(continued from previous page)

```
# Dirt
coarse_dirt = 4F
podzol_top = 4F
podzol_side = F
farmland = 2F
farmland_moist = 2F

# Stone
granite = 2F
diorite = 2F
andesite = 2F
sandstone_top = 4
sandstone_bottom = 4F
stone_slab_top = F
end_stone = 4

# Gravel
gravel = 2
clay = 4F

# Logs
oak_log = 2F
spruce_log = 2F
birch_log = F
jungle_log = 2F
acacia_log = 2F
dark_oak_log = 2F
oak_log_top = 4F
spruce_log_top = 4F
birch_log_top = 4F
jungle_log_top = 4F
acacia_log_top = 4F
dark_oak_log_top = 4F

# Leaves
oak_leaves = 2F
spruce_leaves = 2F
birch_leaves = 2F
jungle_leaves = 2
dark_oak_leaves = 2F
acacia_leaves = 2F

# Ores
gold_ore = 2F
iron_ore = 2F
coal_ore = 2F
diamond_ore = 2F
redstone_ore = 2F
lapis_ore = 2F

# Nether
```

(continues on next page)

(continued from previous page)

```
netherrack = 4F
nether_quartz_ore = 2
soul_sand = 4F
glowstone = 4

# Redstone
redstone_lamp_on = 4F
redstone_lamp = 4F

# Prismarine
prismarine = 4F

# Misc
obsidian = 4F
cactus_side = 2F
```

33.1 JSON schema

Note

Although this page is `.properties` based, it can be mapped to JSON.

```
{
        "$schema": "http://json-schema.org/draft/2020-12/schema",
        "$id": "https://gitlab.com/whoatemybutter/optifinedocs/-/blob/master/schemas/
↪natural_textures.schema.json",
        "title": "Natural Textures",
        "description": "Natural Textures randomly rotate a block's texture based off of␣
↪its position.",
        "type": "object",
        "patternProperties": {
                "^([0-9a-f_]:)?[0-9a-f_]+$": {
                        "enum": ["4", "2", "F", "4F", "2F", "F4", "F2"],
                        "description": "How to rotate and flip the block's texture."
                }
        },
        "additionalProperties": false
}
```

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

CHAPTER

THIRTYFOUR

RANDOM ENTITIES

File location

/assets/minecraft/optifine/random/**/*.png /assets/minecraft/optifine/random/**/*.properties

Random Entities changes an entity's texture based on its qualities, such as position, NBT rules, the day time, and more.

The textures and configurations in `/assets/minecraft/optifine/mob/` are also supported.

Fig. 1: Rabbits can be frogs.

Fig. 2: Button and tooltip for the option, found in *Video Settings* → *Quality*.

Fig. 3: Random entities applied to a Piglin

Vanilla textures are assigned a texture number of 1.

34.1 Files

This file can be placed in the `optifine/random/` folder of the resource pack, parallel to the vanilla texture in `textures/`:

Primary (vanilla) texture

- `/assets/minecraft/textures/entity/creeper/creeper.png`

Alternates

- `/assets/minecraft/optifine/random/entity/creeper/creeper2.png`
- `/assets/minecraft/optifine/random/entity/creeper/creeper3.png`
- `/assets/minecraft/optifine/random/entity/creeper/creeper4.png`
- *etc.*

Properties *(optional)*

- `/assets/minecraft/optifine/random/entity/creeper/creeper.properties`

For paintings:

Primary (vanilla) texture

- `/assets/minecraft/textures/painting/burning_skull.png`

Alternates

- `/assets/minecraft/optifine/random/painting/burning_skull2.png`
- `/assets/minecraft/optifine/random/painting/burning_skull3.png`
- `/assets/minecraft/optifine/random/painting/burning_skull4.png`
- *etc.*

Properties *(optional)*

- `/assets/minecraft/optifine/random/painting/burning_skull.properties`

Textures ending with a digit use the separator `..`

Primary (vanilla) texture

- `/assets/minecraft/textures/entity/warden/warden_pulsating_spots_2.png`

Alternates

- `/assets/minecraft/optifine/random/entity/warden/warden_pulsating_spots_2.2.png`
- `/assets/minecraft/optifine/random/entity/warden/warden_pulsating_spots_2.3.png`
- `/assets/minecraft/optifine/random/entity/warden/warden_pulsating_spots_2.4.png`

Properties (optional)

- `/assets/minecraft/optifine/random/entity/warden/warden_pulsating_spots_2.properties`

34.2 Matching order

Each rule specifies a range of entity textures to use and one or more conditions under which to use them.

The entity coordinates when it spawns *(single player)* or when it is first seen by the client *(multiplayer)* are checked against each rule in sequence.

The first rule that matches wins. If no rule matches, the default texture *(e.g. creeper.png)* is used.

If no `.properties` *file* is present for an entity, then all available textures are used for that type of entity.

Entities with multiple textures will use the `.properties` file for the base texture.

In other words, all of these do not need to be created explicitly:

- `wolf.properties`
- `wolf_tame.properties`
- `wolf_angry.properties`

Just `wolf.properties` will work for all three, provided there are the same number of textures for each. Similarly for `"_eyes"` and `"_overlay"`.

34.3 Properties

Important

`N` starts at 1. Do not skip numbers; n=1 n=2 n=5 in sequence is invalid.

Important

For all of these, `N` is a number that links the individual properties together to form a rule.

34.3.1 `textures.N`

Values: List of texture indexes
Required

List of entity textures to use.

The texture index `1` is the default texture from `/assets/minecraft/texture`. Alternatively, the legacy property `skins.N` can be used.

34.3.2 `skins.N`

Values: See `textures.N`
Optional

34.3.3 `weights.N`

Values: *List* of integers
Optional
Default: None

List of weights to apply to the random choice.

Important

Weights do not have to total 100, or any other particular value. The number of weights should match the number of textures.

34.3.4 biomes.N

Values: List of strings
Optional
Default: None

List of biomes.

The vanilla biome names are listed here. Biomes added by mods can also be used.

34.3.5 heights.N

Values: *Range* of integers
Optional

Height ranges.

Replaces legacy `minHeight` and `maxHeight` properties. Since 1.18, negative values may be specified for height. When used in a range they have to be put in parenthesis `( )`: `(-3)-64`.

34.3.6 name.N

Values: String
Optional

Entity name.

Uses the *Strings* syntax.

34.3.7 professions.N

Values: `none`, `armorer`, `butcher`, `cartographer`, `cleric`, `farmer`, `fisherman`, `fletcher`, `leatherworker`, `librarian`, `mason`, `nitwit`, `shepherd`, `toolsmith`, and `weaponsmith`, along with an optional level experience format
Optional

List of villager professions with optional levels.

Entry format: `<profession>[:level1,level2,...]`.

Example:

- Professions farmer (all levels) or librarian (levels 1,3,4): `professions=farmer librarian:1,3-4`.
- Professions fisher, shepard, nitwit: `professions=fisherman shepherd nitwit`.

34.3.8 colors.N

Values: `white`, `orange`, `magenta`, `light_blue`, `yellow`, `lime`, `pink`, `gray`, `light_gray`, `cyan`, `purple`, `blue`, `brown`, `green`, `red`, and `black`
Optional

List of wolf/cat collar colors or sheep/llama/shulker box/bed colors.

Example: `colors.2=pink magenta purple`.

The legacy property `collarColors` is also recognized.

New in version I1: Bed color detection

New in version I3: Shulker box color detection

34.3.9 baby.N

Values: Boolean
Optional

If entity is a baby. Only valid for mobs.

34.3.10 health.N

Values: List of integers, or *ranges*, or percents
Optional

Range of health values; can also be given in percent. Only valid for mobs.

Example:

```
health.1=10
health.2=5-8 10-12
health.3=0-50%
```

34.3.11 moonPhase.N

Values: List of integers, or ranges
Optional

List of moon phases, from 0 to 7.

Example:

```
moonPhase.1=3
moonPhase.2=0 1 2
moonPhase.1=0-2 4-7
```

34.3.12 dayTime.N

Values: List of integers, or ranges
Optional
Default: None

List of day times in ticks (0-24000)

Example:

```
dayTime.1=2000-10000
dayTime.2=0-1000 18000-24000
```

34.3.13 weather.N

Values: `clear`, `rain`, and `thunder`
Optional

Weather conditions.

Several values can be specified, separated by spaces. For example: `weather=clear rain thunder`.

34.3.14 sizes.N

Values: Integer or *list* or integers or integer *range*
Optional

New in version H7.

Size of entity, if applicable.

Slimes and magma cubes **naturally** spawn in three sizes: 0=small, 1=medium, 3=big. **Naturally** spawing phantoms only spawn in one size: `0`.

Important

This is only valid for mobs with multiple sizes (0-255 for slimes and magma cubes, and 0-64 for phantoms).

Example:

```
textures.1=3
textures.2=0 1 3
textures.3=0-2 4-7
```

34.3.15 nbt.N.TAG

Values: Any valid *NBT matching rule*
Optional
Default: None

NBT-based rule.

Applies only when an NBT tag on the entity has a specific value. See *NBT*. You can have infinitely-many NBT rules.

Warning

This only works with NBT data that is client-side. NBT data that is only server-side will not work. Client-side data is data that the client would see and be able to know. Server-side only data is data that are not sent to the client because the Vanilla client has no use for them.

34.3.16 blocks.N

Values: Block ID
Optional
Default: None

For entities, this checks the block on which the entity is standing. For block entities, this checks the block of the block entity.

34.4 Examples

34.4.1 Creeper

Uses `creeper10.png` through `creeper14.png` for all underground creepers. `creeper13.png` will be used 7.3% *(3/(10+10+10+3+10))* of the time.

```
textures.1=10-14
weights.1=10 10 10 3 10
heights.1=0-55

# Use 5, 7, 9 in high, hilly areas.
textures.2=5 7 9
biomes.2=windswept_hills desert forest badlands jagged_peaks stony_peaks
heights.2=80-255

# Fallback rule if nothing else matches.
# Remember, if no rule matches, only the base creeper/creeper.png will be used.
textures.3=1-4 6 8 15-20
```

34.4.2 Omit Vanilla texture

Uses `slime2.png` through `slime16.png` for all slimes, not including the vanilla texture.

```
textures.1=2-16
```

34.5 JSON schema

Note

Although this page is `.properties` based, it can be mapped to JSON.

```
{
        "$schema": "http://json-schema.org/draft/2020-12/schema",
        "$id": "https://gitlab.com/whoatemybutter/optifinedocs/-/blob/master/schemas/
↪random_entities.schema.json",
        "title": "Random Entities",
        "description": "Random Entities changes an entity's texture based on its␣
↪qualities, such as position, NBT rules, the day time, and more..",
        "type": "object",
        "patternProperties": {
                "^textures\\.\\d+$": {
                        "type": "string",
                        "description": "Space-separated string of entity textures to use.
↪"
                },
                "^skins\\.\\d+$": {
                        "type": "string",
                        "description": "Legacy property of textures.N."
                },
                "^weights\\.\\d+$": {
                        "type": "string",
                        "description": "Space-separated string of weights to apply to␣
↪the random choice."
                },
                "^biomes\\.\\d+$": {
                        "type": "string",
                        "description": "Space-separated string of biomes."
                },
                "^heights\\.\\d+$": {
                        "type": "string",
                        "description": "Height ranges."
                },
                "^name\\.\\d+$": {
                        "type": "string",
                        "description": "Entity name to match."
                },
                "^professions\\.\\d+$": {
                        "pattern":
↪"(none|armorer|butcher|cartographer|cleric|farmer|fisherman|fletcher|leatherworker|librarian|mason|ni
```

(continues on next page)

(continued from previous page)

```
↪\d+(-\\d+)?(,\\d+(-\\d+)?)*)",
                        "description": "Space-separated string of villager professions␣
↪with an optional level specifier."
                },
                "^colors\\.\\d+$": {
                        "type": "string",
                        "pattern": "(white|orange|magenta|light_
↪blue|yellow|lime|pink|gray|light_gray|cyan|purple|blue|brown|green|red|black) ?",
                        "description": "Space-separated string of wolf/cat collar colors␣
↪or sheep/llama/shulker box/bed colors."
                },
                "^baby\\.\\d+$": {
                        "type": "boolean",
                        "description": "If entity is a baby."
                },
                "^health\\.\\d+$": {
                        "type": "string",
                        "description": "Range of health values; can also be given in␣
↪percent."
                },
                "^moonPhase\\.\\d+$": {
                        "type": "string",
                        "description": "List of moon phases, from 0 to 7."
                },
                "^dayTime\\.\\d+$": {
                        "type": "string",
                        "description": "List of day times in ticks, from 0 to 24000."
                },
                "^weather\\.\\d+$": {
                        "type": "string",
                        "pattern": "(clear|rain|thunder) ?",
                        "description": "Weather conditions."
                },
                "^sizes\\.\\d+$": {
                        "type": "string",
                        "description": "Size of mob, if applicable"
                },
                "^nbt\\.([a-zA-Z0-9_\\-.+]+|\".*?\")$": {
                        "type": [
                                "number",
                                "string"
                        ],
                        "description": "NBT-based rule."
                },
                "^blocks\\.\\d+$": {
                        "$ref": "common.schema.json#/$defs/item_id_list",
                        "description": "What block the entity is standing on, or what␣
↪block the entity is."
                }
        },
        "additionalProperties": false
}
```

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

CHAPTER

THIRTYFIVE

SHADERS

File location

$minecraftfolder/shaderpacks/*.zip $minecraftfolder/shaderpacks/*/

Fig. 1: Shaders in action.

Shaders are programs that manipulate the rendering pipeline of the game to create realistic lighting, shadows, reflections, and other visual effects. They can significantly transform the game's appearance, making it more immersive and visually appealing.

OptiFine alone does not include shaderpacks. Shaderpacks must be downloaded from other sources and installed in order to be used.

Fig. 2: A large example of shaders in the Overworld. Shown here is FastPBR.

35.1 Downloading

In game, the Download button in the Shaders menu opens the URL https://optifine.net/shaderPacks, which currently redirects to https://wiki.shaderlabs.org/wiki/Shaderpacks. This is the officially supported list of shaderpacks, and includes last updates, download links, platforms, Minecraft version, and more. It is recommended you download shaderpacks from this website.

Shaderpacks may also be downloaded from anywhere, and some of the most known websites are:

- PlanetMinecraft
- Modrinth
- CurseForge
- Shader developers' personal websites

Regardless of where a shaderpack is from, it is **always** a `.zip` file. If you did not get a `.zip` file, it is **not** a shaderpack.

35.2 Installing

Navigate to your Minecraft folder. If you do not know how to do this, click here.

If it does not already exist, create the `shaderpacks` folder.

Move the shaderpack file into this `shaderpacks` folder.

Re-open your Minecraft's shaders menu and it should appear. Click on it to enable it.

35.3 Configuring

Different shaders expose different configuration options.

Unless you know what you are doing, you should not change the right-hand settings at the shader screen (Normal Map, Old Lighting).

Shader options are accessed through the bottom-right button *Shader Options...*.

35.4 Internal shaders

Internal shaders are debug shaders for purposes of debugging the shader pipeline.

If you are not a shader developer, **you should never enable this.**

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

CHAPTER

THIRTYSIX

SHADERS - DEVELOPMENT

Shaders add a powerful system of rendering light, the elements of the world, and more.

Danger

This document is exceedingly long.

Caution

This document assumes good knowledge of how shaders work. If not, see below.

See also

See https://learnopengl.com/Introduction and https://open.gl/ for tutorials on OpenGL.

Not to be confused with

Shaders

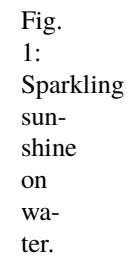

Fig. 1: Sparkling sunshine on water.

36.1 Color attachments

The data is passed from shader to shader using color attachments.

There are at least 4 for all machines. For machines that can support it, there are up to 16.

Warning

MacOS is limited to 8 color attachments, even with a modern GPU

In the deferred, composite, and final shaders, these are referenced by the `gcolor`, `gdepth`, `gnormal`, `composite`, `gaux1`, `gaux2`, `gaux3`, and `gaux4` uniforms.

Note

`colortex0` to `colortex15` can be used instead of `gcolor`, `gdepth`, etc.

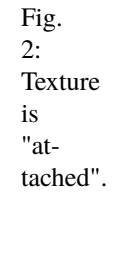

Fig. 2: Texture is "attached".

Despite the naming, `gnormal` and onward are the same and can be used for any purpose, `gcolor` and `gdepth` have meaning:

- The first one, `gcolor`, has its color cleared to the current fog color before rendering.
- The second one, `gdepth`, has its color cleared to solid white before rendering and uses a higher precision storage buffer suitable for storing depth values.
- The rest have their color cleared to black with 0 alpha.

Each color attachment uses 2 buffers (`A` and `B`) with logical names "`main`" and "`alt`", which can be used as ping-pong buffers. When the buffers are flipped, the mapping between `main`/`alt` and `A`/`B` is reversed.

Gbuffer programs always read from `main` (only `gaux1-4`) and write to `main` buffers *(they shouldn't read and write to the same buffer at the same time)*.

Deferred/composite programs always read from `main` and write to `alt` buffers. After a deferred/composite program is rendered, the buffers that it writes to are *flipped* so the next programs can see the current output as input.

The property `flip.<program>.<buffer>=<true|false>` can be used to enable or disable the flip, independent of the buffer write. The virtual programs "`deferred_pre`" and "`composite_pre`" can be used for buffer flipping before the deferred/composite pass.

Output color attachments are configured with the "`/* DRAWBUFFERS:XYZ */`" or "`/* RENDERTARGETS: X,Y,Z */`" comment, placed in the fragment shader. Gbuffers, deferred and composite programs can write to any color attachment, **but no more than 8 at the same time**.

If the output color attachments are not configured, then the program will **write to the first 8** color attachments.

When writing to the color attachments in the composite shader, *blending is disabled*. Writing to color attachments that the composite shader also reads from will generate artifacts, *unless the original contents are simply copied*.

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

36.2 Configurations

36.2.1 Vertex configuration

Fig. 3: Slide them around.

Source	Effect	Comment
`in vec3 mc_Entity;`	useEntityAttrib = true	
`in vec2 mc_midTexCoord;`	useMidTexCoordAttrib = true	
`in vec4 <type> at_tangent;`	useTangentAttrib = true	
`const int countInstances = 1;`	when `countInstances > 1` the geometry will be rendered several times, see uniform `instanceId`	

36.2.2 Geometry configuration

Source	Effect	Comment
`#extension GL_ARB_geometry_shader4 : enable`	Enable GL_ARB_geometry_shader4	
`const int maxVerticesOut = 3;`	Set GEOMETRY_VERTICES_OUT_ARB for GL_ARB_geometry_shader4	

36.2.3 Fragment configuration

Source	Effect	Com
`uniform <type> shadow;`	shadowDepthBuffers = 1	
`uniform <type> watershadow;`	shadowDepthBuffers = 2	
`uniform <type> shadowtex0;`	shadowDepthBuffers = 1	
`uniform <type> shadowtex1;`	shadowDepthBuffers = 2	
`uniform <type> shadowcolor;`	shadowColorBuffers = 1	
`uniform <type> shadowcolor0;`	shadowColorBuffers = 1	
`uniform <type> shadowcolor1;`	shadowColorBuffers = 2	
`uniform <type> depthtex0;`	depthBuffers = 1	
`uniform <type> depthtex1;`	depthBuffers = 2	
`uniform <type> depthtex2;`	depthBuffers = 3	
`uniform <type> gdepth;`	if (bufferFormat[1] == RGBA) bufferFormat[1] = RGBA32F;	
`uniform <type> gaux1;`	colorBuffers = 5	
`uniform <type> gaux2;`	colorBuffers = 6	
`uniform <type> gaux3;`	colorBuffers = 7	
`uniform <type> gaux4;`	colorBuffers = 8	
`uniform <type> colortex4;`	colorBuffers = 5	
`uniform <type> colortex5;`	colorBuffers = 6	
`uniform <type> colortex6;`	colorBuffers = 7	
`uniform <type> colortex7;`	colorBuffers = 8	
`uniform <type> centerDepthSmooth;`	centerDepthSmooth = true	
`/* SHADOWRES:1024 */`	shadowMapWidth = shadowMapHeight = 1024	
`const int shadowMapResolution = 1024;`	shadowMapWidth = shadowMapHeight = 1024	
`/* SHADOWFOV:90.0 */`	shadowMapFov = 90	
`const float shadowMapFov = 90.0;`	shadowMapFov = 90	
`/* SHADOWHPL:160.0 */`	shadowMapDistance = 160.0	
`const float shadowDistance = 160.0f;`	shadowMapDistance = 160.0	
`const float shadowDistanceRenderMul = -1f;`	shadowDistanceRenderMul = -1	Whe
`const float shadowIntervalSize = 2.0f;`	shadowIntervalSize = 2.0	
`const bool generateShadowMipmap = true;`	shadowMipmap = true	
`const bool generateShadowColorMipmap = true;`	shadowColorMipmap = true	
`const bool shadowHardwareFiltering = true;`	shadowHardwareFiltering = true	
`const bool shadowHardwareFiltering0 = true;`	shadowHardwareFiltering[0] = true	
`const bool shadowHardwareFiltering1 = true;`	shadowHardwareFiltering[1] = true	
`const bool shadowtexMipmap = true;`	shadowMipmap[0] = true	
`const bool shadowtex0Mipmap = true;`	shadowMipmap[0] = true	
`const bool shadowtex1Mipmap = true;`	shadowMipmap[1] = true	

Table 1 – continued from previc

Source	Effect	Com
`const bool shadowcolor0Mipmap = true;`	shadowColorMipmap[0] = true	
`const bool shadowColor0Mipmap = true;`	shadowColorMipmap[0] = true	
`const bool shadowcolor1Mipmap = true;`	shadowColorMipmap[1] = true	
`const bool shadowColor1Mipmap = true;`	shadowColorMipmap[1] = true	
`const bool shadowtexNearest = true;`	shadowFilterNearest[0] = true	
`const bool shadowtex0Nearest = true;`	shadowFilterNearest[0] = true	
`const bool shadow0MinMagNearest = true;`	shadowFilterNearest[0] = true	
`const bool shadowtex1Nearest = true;`	shadowFilterNearest[1] = true	
`const bool shadow1MinMagNearest = true;`	shadowFilterNearest[1] = true	
`const bool shadowcolor0Nearest = true;`	shadowColorFilterNearest[0] = true	
`const bool shadowColor0Nearest = true;`	shadowColorFilterNearest[0] = true	
`const bool shadowColor0MinMagNearest = true;`	shadowColorFilterNearest[0] = true	
`const bool shadowcolor1Nearest = true;`	shadowColorFilterNearest[1] = true	
`const bool shadowColor1Nearest = true;`	shadowColorFilterNearest[1] = true	
`const bool shadowColor1MinMagNearest = true;`	shadowColorFilterNearest[1] = true	
`/* WETNESSHL:600.0 */`	wetnessHalfLife = 600 (ticks)	
`const float wetnessHalflife = 600.0f;`	wetnessHalfLife = 600 (ticks)	
`/* DRYNESSHL:200.0 */`	drynessHalfLife = 200 (ticks)	
`const float drynessHalflife = 200.0f;`	drynessHalfLife = 200 (ticks)	
`const float eyeBrightnessHalflife = 10.0f;`	eyeBrightnessHalflife = 10 (ticks)	
`const float centerDepthHalflife = 1.0f;`	centerDepthSmoothHalflife = 1 (ticks)	
`const float sunPathRotation = 0f;`	sunPathRotation = 0f	
`const float ambientOcclusionLevel = 1.0f;`	ambientOcclusionLevel = 1.0f	0.0f =
`const int superSamplingLevel = 1;`	superSamplingLevel = 1	
`const int noiseTextureResolution = 256;`	noiseTextureResolution = 256	
`/* GAUX4FORMAT:RGBA32F */`	buffersFormat[7] = GL_RGBA32F	
`/* GAUX4FORMAT:RGB32F */`	buffersFormat[7] = GL_RGB32F	
`/* GAUX4FORMAT:RGB16 */`	buffersFormat[7] = GL_RGB16	
`const int <bufferIndex>Format = <format>;`	bufferFormats[index] = <format>	See *I*
`const bool <bufferIndex>Clear = false;`	gbuffersClear[index] = false	Skip
`const vec4 <bufferIndex>ClearColor = vec4();`	gbuffersClearColor[index] = vec4(r, g, b, a)	Clear
`const bool <bufferIndex>MipmapEnabled = true;`	bufferMipmaps[index] = true	Only
`const int <shadowBufferIx>Format = <format>;`	shadowBufferFormats[index] = <format>	See *S*
`const bool <shadowBufferIx>Clear = false;`	shadowBuffersClear[index] = false	Skip
`const vec4 <shadowBufferIx>ClearColor = vec4();`	shadowBuffersClearColor[index] = vec4(r, g, b, a)	Clear
`/* DRAWBUFFERS:0257 */`	drawBuffers = {0, 2, 5, 7)	Only
`/* RENDERTARGETS: 0,2,11,15 */`	drawBuffers = {0, 2, 11, 15}	Buffe

Assumes latest OptiFine version.

Updated to commit 8ed2130d.

Open source at https://gitlab.com/whoatemybutter/optifinedocs.

36.3 Formats

36.3.1 Texture formats

Fig. 4: RGB BGR GBR RBG BRG.

8-bit

Normalized	Signed normalized	Integer	Unsigned integer
R8	R8_SNORM	R8I	R8I
RG8	RG8_SNORM	RG8I	RG8I
RGB8	RGB8_SNORM	RGB8I	RGB8I
RGBA8	RGBA8_SNORM	RGBA8I	RGBA8I

16-bit

Normalized	Signed normalized	Float	Integer	Unsigned integer
R16	R16_SNORM	R16F	R16I	R16UI
RG16	RG16_SNORM	RG16F	RG16I	RG16UI
RGB16	RGB16_SNORM	RGB16F	RGB16I	RGB16UI
RGBA16	RGBA16_SNORM	RGBA16F	RGBA16I	RGBA16UI

32-bit

Float	Integer	Unsigned integer
R32F	R32I	R32UI
RG32F	RG32I	RG32UI
RGB32F	RGB32I	RGB32UI
RGBA32F	RGBA32I	RGBA32UI

Mixed

Attention

This documentation does not information about these

´R3_G3_B2 RGB5_A1 RGB10_A2 R11F_G11F_B10F RGB9_E5

36.3.2 Pixel formats

Normalized

- RED
- RG
- RGB
- BGR
- RGBA
- BGRA

Integer

- RED_INTEGER
- RG_INTEGER
- RGB_INTEGER
- BGR_INTEGER
- RGBA_INTEGER
- BGRA_INTEGER

36.3.3 Pixel types

- BYTE
- SHORT
- INT
- HALF_FLOAT
- FLOAT
- UNSIGNED_BYTE
- UNSIGNED_BYTE_3_3_2
- UNSIGNED_BYTE_2_3_3_REV
- UNSIGNED_SHORT
- UNSIGNED_SHORT_5_6_5
- UNSIGNED_SHORT_5_6_5_REV
- UNSIGNED_SHORT_4_4_4_4
- UNSIGNED_SHORT_4_4_4_4_REV
- UNSIGNED_SHORT_5_5_5_1
- UNSIGNED_SHORT_1_5_5_5_REV
- UNSIGNED_INT
- UNSIGNED_INT_8_8_8_8

- UNSIGNED_INT_8_8_8_8_REV
- UNSIGNED_INT_10_10_10_2
- UNSIGNED_INT_2_10_10_10_REV

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

36.4 ID mapping

36.4.1 Block ID mapping

Fig. 5: One of many numbers.

The block ID mapping is defined in `shaders/block.properties` included in the shader pack.

Forge mods may add custom block mapping as `assets/<modid>/shaders/block.properties` in the mod JAR file. The `block.properties` file can use conditional *Compilation directives (#ifdef, #if, etc.)*

For more details see section *Macros* A to I. Option macros are also available. Format: `block.<id>=<block1> <block2> ...` The key is the substitute block ID, the values are the blocks which are to be replaced. Only one line per block ID is allowed.

See *Syntax* for the block matching rules.

```
# Short format
block.31=red_flower yellow_flower reeds

# Long format
block.32=minecraft:red_flower ic2:nether_flower botania:reeds

# Properties
block.33=minecraft:red_flower:type=white_tulip minecraft:red_flower:type=pink_tulip␣
↪botania:reeds:type=green
```

36.4.2 Item ID mapping

The item ID mapping is defined in `shaders/item.properties` included in the shader pack. Forge mods may add custom item mapping as `/assets/<modid>/shaders/item.properties` in the mod JAR file.

See *Block ID Mapping* for the overview.

```
# Short format
item.5000=diamond_sword dirt

# Long format
item.5001=minecraft:diamond_sword botania:reeds
```

36.4.3 Entity ID mapping

The entity ID mapping is defined in `shaders/entity.properties` included in the shader pack. Forge mods may add custom entity mapping as `assets/<modid>/shaders/entity.properties` in the mod JAR file.

See *Block ID Mapping* for the overview.

```
# Short format
entity.2000=sheep cow

# Long format
entity.2001=minecraft:pig botania:pixie
```

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

36.5 Indexes

36.5.1 Draw buffer index

Prefix	Index
`colortex<0-15>`	0-15
`gcolor`	0
`gdepth`	1
`gnormal`	2
`composite`	3
`gaux1`	4
`gaux2`	5
`gaux3`	6
`gaux4`	7

36.5.2 Shadow buffer index

Prefix	Index
`shadowcolor`	0
`shadowcolor<0-1>`	0-1

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

36.6 Options screen

Shader options are parsed from the `.fsh` and `.vsh` files located in the folder `shaders`. The line comment located after the option is shown as a tooltip.

Tooltip lines are split on sentence end `". "` *(period space)*. Tooltip lines ending with `!` are automatically shown red.

One option can be present in several shader files and it will be switched simultaneously in all of them. Ambiguous options *(different default values found)* are **disabled and can not be changed**.

Fig. 6: Pushy pushy.

- Left-clicking on an option button selects the next value,
- Right-clicking selects the previous value,
- Shift-clicking resets the option to default value

Boolean, default ON:

```
#define SSAO       // Screen space ambient occlusion. High performance impact.
```

Boolean, default OFF:

```
// #define SSAO    // Screen space ambient occlusion. High performance impact.
```

The boolean variables are recognized only if the matching "#ifdef" or "#ifndef" is also found in the same file.

List:

```
#define SHADOW_DARKNESS 0.10   // Shadow darkness levels [0.05 0.10 0.20]
```

The allowed values are given as a list `[v1 v2 v3]` in the comment. The default value is automatically added if not present in the list.

Some `const` variables are also recognized *(for backwards compatibility with the Shaders Mod)*. They use a structure similar to the macro variables, for example:

```
const int shadowMapResolution  1572;  // Shadowmap resolution [1024 1572 2048]
const float shadowDistance     64.0;  // Draw distance of shadows [32.0 64.0 128.0 256.0]
```

Const variables without allowed values are, by default, not visible, unless used in a profile or configured on a screen.

The recognized `const` variables are:

```
shadowMapResolution
shadowDistance
shadowDistanceRenderMul
shadowIntervalSize
generateShadowMipmap
generateShadowColorMipmap
shadowHardwareFiltering
shadowHardwareFiltering0
shadowHardwareFiltering1
shadowtex0Mipmap
shadowtexMipmap
shadowtex1Mipmap
shadowcolor0Mipmap
shadowColor0Mipmap
shadowcolor1Mipmap
```

(continues on next page)

(continued from previous page)

```
shadowColor1Mipmap
shadowtex0Nearest
shadowtexNearest
shadow0MinMagNearest
shadowtex1Nearest
shadow1MinMagNearest
shadowcolor0Nearest
shadowColor0Nearest
shadowColor0MinMagNearest
shadowcolor1Nearest
shadowColor1Nearest
shadowColor1MinMagNearest
wetnessHalflife
drynessHalflife
eyeBrightnessHalflife
centerDepthHalflife
sunPathRotation
ambientOcclusionLevel
superSamplingLevel
noiseTextureResolution
```

36.6.1 Labels and tooltips

User friendly **option** labels can be loaded from language files in `/shaders/lang/`. Example from "/shaders/lang/en_us.lang":

```
option.SHADOW_FILTER=Shadow Filter
option.SHADOW_FILTER.comment=Smooth out edges of shadows. Very small performance hit.
```

User friendly **value** labels can be loaded from language files in `/shaders/lang/`. Example from "/shaders/lang/en_us.lang":

```
value.SHADOW_FILTER.0.4f=Normal
value.SHADOW_FILTER.0.9f=Soft
```

Value formatting can be added with:

```
prefix.SHADOW_FILTER=(
suffix.SHADOW_FILTER=)
```

Profile tooltips can be loaded from language files in `/shaders/lang/`. Example from "/shaders/lang/en_us.lang":

```
profile.comment=Low - low. Medium - medium. Standard - standard. High - high. Ultra -␣
↪ultra.
```

36.6.2 Profiles

Profiles allow a set of options to be switched together The current profile is detected based on the selected option values If no profile matches the current option values, the profile "Custom" is selected

It is recommended that all profiles use the same list of options and only the values differ. Disabled programs are special options and only disabling *(prefix !)* is recognized for them.

Format: `# profile.<name>=<list of options>`

Profile options

- `OPTION:value`: set value
- `OPTION`: set boolean option **ON**
- `!OPTION`: set boolean option **OFF**
- `profile.NAME`: copy all options from another profile
- `!program.name`: disable program `name`. The program name may include dimension: `world-1/gbuffers_water`

The following program names are recognized:

```
gbuffers_basic
gbuffers_textured
gbuffers_textured_lit
gbuffers_skybasic
gbuffers_skytextured
gbuffers_clouds
gbuffers_terrain
gbuffers_terrain_solid
gbuffers_terrain_cutout_mip
gbuffers_terrain_cutout
gbuffers_damagedblock
gbuffers_water
gbuffers_block
gbuffers_beaconbeam
gbuffers_item
gbuffers_entities
gbuffers_armor_glint
gbuffers_spidereyes
gbuffers_hand
gbuffers_weather
composite
composite1
composite2
composite3
composite4
composite5
composite6
composite7
final
shadow
shadow_solid
```

(continues on next page)

(continued from previous page)

```
shadow_cutout
deferred
deferred1
deferred2
deferred3
deferred4
deferred5
deferred6
deferred7
gbuffers_hand_water
```

Profile examples

```
profile.LOW=SSAO:false GOD_RAYS:false SHADOW_DIST:40 !program.composite1
profile.MED=profile.LOW GOD_RAYS SHADOW_DIST:80
profile.HIGH=SSAO GOD_RAYS SHADOW_DIST:120
```

User friendly profile labels can be loaded from language files in "/shaders/lang" Example from "/shaders/lang/en_us.lang":

```
profile.LOW=Low
profile.LOW.comment=Low quality. Intel and Mac compatible. No God Rays and SSAO
profile.MED=Medium
profile.MED.comment=Medium quality. Nvidia or AMD graphics card recommended
profile.HIGH=High
profile.HIGH.comment=High quality. Modern Nvidia or AMD graphics card required
```

36.6.3 Screen

- Main screen: `screen=<list of options>`
- Sub-screen: `screen.<NAME>=<list of options>`

Screen options

Option	Option name
[NAME]	link to sub-screen NAME
<profile>	profile selection
<empty>	empty slot
*	the rest of the options not configured on any of the screens

Screen columns

By default the options are shown in two columns:

1	2
3	4
5	6
...	...

When more than 18 options are present, the screen switches to 3 or more columns. The option names are automatically shortened to avoid text overflow outside the button.

- Main screen: `screen.columns=2`
- Sub-screen: `screen.<NAME>.columns=2`

36.6.4 Example

```
screen=<empty> <empty> <profile> ABOUT <empty> <empty> [VISUAL] [POST_PROCESS] [SHADOWS]␣
↪[COLOR] [SKY] [WATER] [TERRAIN] [WORLD]
screen.VISUAL=<empty> <empty> AO AO_STRENGTH LIGHT_SHAFT LIGHT_SHAFT_STRENGTH␣
↪DESATURATION DESATURATION_FACTOR REFLECTION REFLECTION_TRANSLUCENT ADVANCED_MATERIALS␣
↪[ADVANCED_MATERIALS] BLACK_OUTLINE PROMO_OUTLINE TOON_LIGHTMAP WHITE_WORLD
screen.ADVANCED_MATERIALS=<empty> <empty> MATERIAL_FORMAT REFLECTION_SPECULAR REFLECTION_
↪RAIN REFLECTION_ROUGH REFLECTION_PREVIOUS ALBEDO_METAL <empty> <empty> PARALLAX␣
↪PARALLAX_DEPTH SELF_SHADOW SELF_SHADOW_ANGLE PARALLAX_QUALITY PARALLAX_DISTANCE␣
↪DIRECTIONAL_LIGHTMAP DIRECTIONAL_LIGHTMAP_STRENGTH
screen.SUN_EFFECTS=SUN_EFFECTS GOD_RAYS LENS_FLARE RAINDROPS
screen.WAVING_GRASS=WAVING_GRASS WAVING_LEAVES WAVING_VINES
screen.MISC=*
```

User friendly screen labels can be loaded from language files in `/shaders/lang/`. Example from "/shaders/lang/en_us.lang":

```
screen.DOF=Depth of field
screen.DOF.comment=Depth of field effect. Adds blur to out of focus objects.
screen.WAVING=Waving grass and leaves
screen.WAVING.comment=Waving grass, leaves, fire and entities
```

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

36.7 Preprocessor directives

36.7.1 Compilation directives

The shaders configuration parsing is affected by the preprocessor conditional compilation directives. The following preprocessor directives are currently recognized:

```
#define <macro>
#undef <macro>
#ifdef <macro>
#ifndef <macro>
#if <int>
#if defined <macro>
#if !defined <macro>
#elif <int>
#elif defined <macro>
#elif !defined <macro>
#else
#endif
```

The current shaderpack can be reloaded by pressing "`F3+R`" or using the command "`/reloadShaders`".

Note

The "`/reloadShaders`" command has no output, and is not sent to a server when run.

36.7.2 Properties

Note

Because there are so many properties, some keys have their own section.

Important

The table does not list all the available keys, see the sections below, too.

This file can use conditional *compilation directives (#ifdef, #if, etc.)* For more details see *Macros*, A to I. The settings version, `oldLighting`, `separateAo`, sliders, profiles, and screen are parsed without option macros.

Fig. 7: The shader settings screen, with every option visible.

Key	Values	Meaning
`clouds`	fast|fancy|off	Set clouds type, also controlled by `Video Settings > Details > Clouds` with higher priority

continues on next page

Table 2 – continued from previous page

Key	Values	Meaning
oldHandLight	Boolean	Enable or disable old hand light When enabled: uses the handheld item with higher light value for the main hand The old hand light is also controlled by `Video Settings > Shaders > Old Hand Light` with higher priority
dynamicHandLight	Boolean	Enable or disable the dynamic hand light from Dynamic Lights This option can be used to disable the dynamic hand light from Dynamic Lights if the shader implements its own hand light.
oldLighting	Boolean	Enable or disable old block lighting with fixed multiplier The old block lighting is also controlled by `Video Settings > Shaders > Old Lighting` with higher priority
shadowTerrain	Boolean	Enable or disable rendering of terrain *(solid, cutout, cutout_mipped)* in the shadow pass
shadowTranslucent	Boolean	Enable or disable rendering of translucent blocks *(water, stained glass)* in the shadow pass
shadowEntities	Boolean	Enable or disable rendering of entities in the shadow pass
shadowBlockEntities	Boolean	Enable or disable rendering of block entities in the shadow pass
underwaterOverlay	Boolean	Enable or disable underwater screen overlay
sun	Boolean	Enable or disable sun rendering
moon	Boolean	Enable or disable moon rendering
vignette	Boolean	Enable or disable vignette rendering
backFace.solid	Boolean	Enable back-face rendering per render layer, default is false See *Block render layers*
backFace.cutout	Boolean	See above
backFace.cutoutMipped	Boolean	See above
backFace.translucent	Boolean	See above

continues on next page

Table 2 – continued from previous page

Key	Values	Meaning
rain.depth	Boolean	Enables rain and snow to write to the depth buffer
beacon.beam.depth	Boolean	Enables beacon beam to write to the depth buffer
separateAo	Boolean	When enabled the AO brightness *(smooth lighting)* is separated from color.rbg and put in color.a.
frustum.culling	Boolean	Enable or disable frustum culling
shadow.culling	Boolean	Enable or disable shadow culling
version.<mc_ver>	<of_edition> The minimum OptiFine version which is required by the shader pack Each Minecraft version has to be specified separately. For example: version.1.12.2=D1, version.1.10.2=F1, version.1.8=J1	
texture.noise	<path>	Allows the noise texture to be loaded from the shader pack
sliders	<list of options>	Options with multiple allowed values can be shown as sliders
alphaTest.<program>	<off|func ref> The alpha test can be configured per program Where: * func is one of: NEVER, LESS, EQUAL, LEQUAL, GREATER, NOTEQUAL, GEQUAL, GL_ALWAYS * ref: float value	
blend.<program>	<off|src dst srcA dstA> The blend mode can be configured per program. Where src, dst, srcA, and dstA are one of: ZERO, ONE, SRC_COLOR, ONE_MINUS_SRC_COLOR, DST_COLOR, ONE_MINUS_DST_COLOR, SRC_ALPHA, ONE_MINUS_SRC_ALPHA, DST_ALPHA, ONE_MINUS_DST_ALPHA, SRC_ALPHA_SATURATE	

continues on next page

Table 2 – continued from previous page

Key	Values	Meaning
`blend.<program>.<buffer>`	<off|src dst srcA dstA>	Blend mode per buffer The blend mode can be configured per program and buffer Where `src`, `dst`, `srcA`, and `dstA` are one of: `ZERO`, `ONE`, `SRC_COLOR`, `ONE_MINUS_SRC_COLOR`, `DST_COLOR`, `ONE_MINUS_DST_COLOR`, `SRC_ALPHA`, `ONE_MINUS_SRC_ALPHA`, `DST_ALPHA`, `ONE_MINUS_DST_ALPHA`, `SRC_ALPHA_SATURATE`
`scale.<program>`	<scale|scale offsetX ofsetY>	Composite render scale Defines a custom viewport to be used when rendering composite and deferred programs. The `scale`, `offsetX`, and `offsetY` should be between `0.0` and `1.0`.
`flip.<program>.<buffer>`	<true|false>	Ping-pong buffer flip Enable or disable ping-pong buffer flip for a specific buffer name in a specific composite or deferred program. When buffer flip is disabled the next composite program will use the same input and output buffers for this buffer name. The last program that writes to the buffer should have flip enabled so that the following programs can read from the buffer. This can be used with composite render scale to allow several composite programs to write to different regions in the same buffer. Forced buffer flip can be used to read from both ping-pong buffers.

continues on next page

Table 2 – continued from previous page

Key	Values	Meaning
`size.buffer.<buffer>`	width height	Buffer size Define custom fixed size for a specific buffer. Only `prepare`, `deferred`, and `composite` programs can render to fixed size buffers. Rendering to fixed size and normal buffers at the same time is not possible. When rendering to several fixed size buffers all of them must have the same size. When width and height are floating point values, then the buffer size will be relative to the render size. For example: `size.buffer.colortex2=0.5 0.5` will create the buffer with width and height that is half of the render width and height.
`program.<program>.enabled`	<expression>	Enable or disable programs depending on shader options Disabled progams are processed as not defined and instead their fallback programs will be used. The program name can contain dimension folder, for example: `program.world-1/composite2.enabled=BLOOM` The expression is a boolean expression which can use shader options of type switch (on/off), for example: `program.composite.enabled=BLOOM && !GODRAYS`

36.7.3 Macros

The standard macros are automatically included after the `#version` declaration in every shader file.

A. Minecraft version

```
#define MC_VERSION <value>
```

The value is in format 122 (major 1, minor 2, release 2) For example: 1.9.4 -> 10904, 1.11.2 -> 11102, etc.

B. Maximum supported GL version

```
#define MC_GL_VERSION <value>
```

The value is an integer, for example: `210`: 2.1, `320`: 3.2, `450`: 4.5

C. Maximum supported GLSL version

```
#define MC_GLSL_VERSION <value>
```

The value is an integer, for example: `210`: 2.1, `320`: 3.2, `450`: 4.5

D. Operating system

One of the following:

```
#define MC_OS_WINDOWS
#define MC_OS_MAC
#define MC_OS_LINUX
#define MC_OS_OTHER
```

E. Driver

One of the following:

```
#define MC_GL_VENDOR_AMD
#define MC_GL_VENDOR_ATI
#define MC_GL_VENDOR_INTEL
#define MC_GL_VENDOR_MESA
#define MC_GL_VENDOR_NVIDIA
#define MC_GL_VENDOR_XORG
#define MC_GL_VENDOR_OTHER
```

F. GPU

One of the following:

```
#define MC_GL_RENDERER_RADEON
#define MC_GL_RENDERER_GEFORCE
#define MC_GL_RENDERER_QUADRO
#define MC_GL_RENDERER_INTEL
#define MC_GL_RENDERER_GALLIUM
#define MC_GL_RENDERER_MESA
#define MC_GL_RENDERER_OTHER
```

G. OpenGL extensions

Macros for the supported OpenGL extensions are named like the corresponding extension with a prefix `MC_`. For example, the macro `MC_GL_ARB_shader_texture_lod` is defined when the extension `GL_ARB_shader_texture_lod` is supported.

Only the macros which are referenced and supported are added to the shader file.

H. Options

```
#define MC_FXAA_LEVEL <value>              // When FXAA is enabled, values: 2, 4
#define MC_NORMAL_MAP                      // When the normal map is enabled
#define MC_SPECULAR_MAP                    // When the specular map is enabled
#define MC_RENDER_QUALITY <value>          // Values: 0.5, 0.70710677, 1.0, 1.4142135, 2.0
#define MC_SHADOW_QUALITY <value>          // Values: 0.5, 0.70710677, 1.0, 1.4142135, 2.0
#define MC_HAND_DEPTH <value>              // Values: 0.0625, 0.125, 0.25
#define MC_OLD_HAND_LIGHT                  // When Old Hand Light is enabled
#define MC_OLD_LIGHTING                    // When Old Lighting is enabled
#define MC_ANISOTROPIC_FILTERING <value>   // When anisotropic filtering is enabled
```

I. Textures

See also

See *Texture Properties*.

```
#define MC_TEXTURE_FORMAT_LAB_PBR       // Texture format LabPBR (https://github.com/
↪rre36/lab-pbr/wiki)
#define MC_TEXTURE_FORMAT_LAB_PBR_1_3   // Version 1.3
```

J. Render stages

Constants for the uniform "`renderStage`". The constants are given in the order in which the stages are executed.

```
#define MC_RENDER_STAGE_NONE <const>                    // Undefined
#define MC_RENDER_STAGE_SKY <const>                     // Sky
#define MC_RENDER_STAGE_SUNSET <const>                  // Sunset and sunrise overlay
#define MC_RENDER_STAGE_CUSTOM_SKY <const>              // Custom sky
#define MC_RENDER_STAGE_SUN <const>                     // Sun
#define MC_RENDER_STAGE_MOON <const>                    // Moon
#define MC_RENDER_STAGE_STARS <const>                   // Stars
#define MC_RENDER_STAGE_VOID <const>                    // Void
#define MC_RENDER_STAGE_TERRAIN_SOLID <const>           // Terrain solid
#define MC_RENDER_STAGE_TERRAIN_CUTOUT_MIPPED <const>   // Terrain cutout mipped
#define MC_RENDER_STAGE_TERRAIN_CUTOUT <const>          // Terrain cutout
#define MC_RENDER_STAGE_ENTITIES <const>                // Entities
#define MC_RENDER_STAGE_BLOCK_ENTITIES <const>          // Block entities
#define MC_RENDER_STAGE_DESTROY <const>                 // Destroy overlay
#define MC_RENDER_STAGE_OUTLINE <const>                 // Selection outline
#define MC_RENDER_STAGE_DEBUG <const>                   // Debug renderers
#define MC_RENDER_STAGE_HAND_SOLID <const>              // Solid handheld objects
#define MC_RENDER_STAGE_TERRAIN_TRANSLUCENT <const>     // Terrain translucent
#define MC_RENDER_STAGE_TRIPWIRE <const>                // Tripwire string
#define MC_RENDER_STAGE_PARTICLES <const>               // Particles
#define MC_RENDER_STAGE_CLOUDS <const>                  // Clouds
#define MC_RENDER_STAGE_RAIN_SNOW <const>               // Rain and snow
#define MC_RENDER_STAGE_WORLD_BORDER <const>            // World border
#define MC_RENDER_STAGE_HAND_TRANSLUCENT <const>        // Translucent handheld objects
```

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

36.8 Programs

Name	Render	When not defined, use
<none>	gui, menus	<none>

Fig. 8: A shiny glistening cone.

36.8.1 Shadow map

Name	Render	When not defined, use
shadow	everything in shadow pass	<none>
shadow_solid	<not used>	shadow
shadow_cutout	<not used>	shadow

36.8.2 Shadow composite

Name	Render	When not defined, use
shadowcomp	<shadowcomp>	<none>
shadowcomp1	<shadowcomp>	<none>
...		
shadowcomp15	<shadowcomp>	<none>

36.8.3 Prepare

Name	Render	When not defined, use
prepare	<prepare>	<none>
prepare1	<prepare>	<none>
...		
prepare15	<prepare>	<none>

36.8.4 GBuffers

Name	Render	When not defined, use
gbuffers_basic	leash, block selection box	<none>
gbuffers_textured	particles	gbuffers_basic
gbuffers_textured_lit	lit particles, world border	gbuffers_textured
gbuffers_skybasic	sky, horizon, stars, void	gbuffers_basic
gbuffers_skytextured	sun, moon	gbuffers_textured
gbuffers_clouds	clouds	gbuffers_textured
gbuffers_terrain	solid, cutout, cutout_mip	gbuffers_textured_lit
gbuffers_terrain_solid	<not used>	gbuffers_terrain
gbuffers_terrain_cutout_mip <not used>	gbuffers_terrain	
gbuffers_terrain_cutout	<not used>	gbuffers_terrain
gbuffers_damagedblock	damaged blocks	gbuffers_terrain
gbuffers_block	block entities	gbuffers_terrain
gbuffers_beaconbeam	beacon beam	gbuffers_textured
gbuffers_item	<not used>	gbuffers_textured_lit
gbuffers_entities	entities	gbuffers_textured_lit
gbuffers_entities_glowing	glowing entities, spectral effect	gbuffers_entities
gbuffers_armor_glint	glint on armor and handheld items	gbuffers_textured
gbuffers_spidereyes	eyes of spider, enderman and dragon	gbuffers_textured
gbuffers_hand	hand and opaque handheld objects	gbuffers_textured_lit
gbuffers_weather	rain, snow	gbuffers_textured_lit

GBuffers translucent

Name	Render	When not defined, use
gbuffers_water	translucent	gbuffers_terrain
gbuffers_hand_water	translucent handheld objects	gbuffers_hand

36.8.5 Deferred

Name	Render	When not defined, use
deferred_pre	<virtual> flip ping-pong buffers	<none>
deferred	<deferred>	<none>
deferred1	<deferred>	<none>
...		
deferred15	<deferred>	<none>

36.8.6 Composite

Name	Render	When not defined, use
`composite_pre`	<virtual> flip ping-pong buffers	<none>
`composite`	<composite>	<none>
`composite1`	<composite>	<none>
...		
`composite15`	<composite>	<none>

36.8.7 Final

Name	Render	When not defined, use
`final`	<final>	<none>

Important

The programs `shadow_solid`, `shadow_cutout`, `gbuffers_terrain_solid`, `gbuffers_terrain_cutout` and `gbuffers_terrain_cutout_mip` are not used

Attention

TO-DO: Separate programs for world border, entities (by id, by type), cape, elytra, wolf collar, etc.

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

36.9 Textures

36.9.1 Custom textures

Fig. 9: Stone bricks, actually.

Allows custom textures to be bound to the available shader units. Format: `texture.<stage>.<name>=<path>`

`<stage>`:

- `gbuffers`: gbuffers and shadow programs
- `deferred`: deferred programs
- `composite`: composite and final programs

<name> is the texture unit name.

The textures can be loaded from different places:

1. **Shader pack**
    - The texture path is relative to the folder shaders/: texture.composite.colortex1=textures/noise.png
2. **Resource pack**
    - The texture path should start with minecraft:: texture.composite.colortex2=minecraft:textures/font/ascii.png
3. **Dynamic (lightmap, texture atlas)**
    - texture.composite.colortex3=minecraft:dynamic/lightmap_1
    - texture.composite.colortex4=minecraft:textures/atlas/blocks.png

The suffix _n and _s can be used to load the normal/specular variant of the texture: minecraft:textures/atlas/blocks_n.png

Raw textures *(binary dump)* can also be loaded: texture.<stage>.<name>=<path> <type> <internalFormat> <dimensions> <pixelFormat> <pixelType>

Where:

- <type> is one of: TEXTURE_1D, TEXTURE_2D, TEXTURE_3D, TEXTURE_RECTANGLE
- <internalFormat> is the texture format, see *Texture formats* for the available names
- <dimensions> is a list of texture dimensions, depends on the texture type
- <pixelFormat> is the pixel format, see *Pixel formats* for the available names
- <pixelType> is the pixel type, see *Pixel types* for the available names

It is possible to bind several textures with different types to one texture unit. The shaders can differentiate between them based on the sampler type: sampler1d, sampler2d, sampler3d...

In one program only one sampler type can be used per texture unit. The suffixes .0 to .9 can be added to <name> to avoid duplicate property keys.

Wrap and filter modes can be configured by adding standard texture .mcmeta files, for example: textures/lut_3d.dat.mcmeta.

36.9.2 GBuffers textures

ID	Name	Legacy name
0	gtexture	texture
1	lightmap	
2	normals	
3	specular	
4	shadowtex0	shadow, watershadow
5	shadowtex1	shadow (when watershadow used)
6	depthtex0	
7	gaux1	colortex4 <custom texture or output from deferred programs>
8	gaux2	colortex5 <custom texture or output from deferred programs>
9	gaux3	colortex6 <custom texture or output from deferred programs>
10	gaux4	colortex7 <custom texture or output from deferred programs>
12	depthtex1	
13	shadowcolor0	shadowcolor
14	shadowcolor1	
15	noisetex	
16	colortex8	<custom texture or output from deferred programs>
17	colortex9	<custom texture or output from deferred programs>
18	colortex10	<custom texture or output from deferred programs>
19	colortex11	<custom texture or output from deferred programs>
20	colortex12	<custom texture or output from deferred programs>
21	colortex13	<custom texture or output from deferred programs>
22	colortex14	<custom texture or output from deferred programs>
23	colortex15	<custom texture or output from deferred programs>

36.9.3 Shadow textures

ID	Name	Legacy name
0	gtexture	texture
1	lightmap	
2	normals	
3	specular	
4	shadowtex0	shadow, watershadow
5	shadowtex1	shadow (when watershadow used)
7	gaux1	colortex4 <custom texture>
8	gaux2	colortex5 <custom texture>
9	gaux3	colortex6 <custom texture>
10	gaux4	colortex7 <custom texture>
13	shadowcolor0	shadowcolor
14	shadowcolor1	
15	noisetex	
16	colortex8	<custom texture>
17	colortex9	<custom texture>
18	colortex10	<custom texture>
19	colortex11	<custom texture>
20	colortex12	<custom texture>
21	colortex13	<custom texture>
22	colortex14	<custom texture>
23	colortex15	<custom texture>

36.9.4 Composite and deferred textures

ID	Name	Legacy name
0	colortex0	gcolor
1	colortex1	gdepth
2	colortex2	gnormal
3	colortex3	composite
4	shadowtex0	shadow, watershadow
5	shadowtex1	shadow (when watershadow used)
6	depthtex0	gdepthtex
7	colortex4	gaux1
8	colortex5	gaux2
9	colortex6	gaux3
10	colortex7	gaux4
11	depthtex1	
12	depthtex2	
13	shadowcolor0	shadowcolor
14	shadowcolor1	
15	noisetex	
16	colortex8	
17	colortex9	
18	colortex10	
19	colortex11	
20	colortex12	
21	colortex13	
22	colortex14	
23	colortex15	

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

36.10 Uniforms

36.10.1 General uniforms

Source	Value
`uniform int heldItemId;`	held item ID (main hand), used only for items defined in `item.properties`
`uniform int heldBlockLightValue;`	held item light value (main hand)
`uniform int heldItemId2;`	held item ID (off hand), used only for items defined in `item.properties`
`uniform int heldBlockLightValue2;`	held item light value (off hand)
`uniform int fogMode;`	GL_LINEAR, GL_EXP or GL_EXP2
`uniform float fogDensity;`	0.0-1.0
`uniform vec3 fogColor;`	r, g, b

Table 3 – continued from previous page

Source	Value
`uniform vec3 skyColor;`	r, g, b
`uniform int worldTime;`	<ticks> = worldTicks % 24000
`uniform int worldDay;`	<days> = worldTicks / 24000
`uniform int moonPhase;`	0-7
`uniform int frameCounter;`	Frame index (0 to 720719, then resets to 0)
`uniform float frameTime;`	last frame time, seconds
`uniform float frameTimeCounter;`	run time, seconds (resets to 0 after 3600s)
`uniform float sunAngle;`	0.0-1.0
`uniform float shadowAngle;`	0.0-1.0
`uniform float rainStrength;`	0.0-1.0
`uniform float aspectRatio;`	viewWidth / viewHeight
`uniform float viewWidth;`	viewWidth
`uniform float viewHeight;`	viewHeight
`uniform float near;`	near viewing plane distance
`uniform float far;`	far viewing plane distance
`uniform vec3 sunPosition;`	sun position in eye space
`uniform vec3 moonPosition;`	moon position in eye space
`uniform vec3 shadowLightPosition;`	shadow light (sun or moon) position in eye space
`uniform vec3 upPosition;`	direction up
`uniform vec3 cameraPosition;`	camera position in world space
`uniform vec3 previousCameraPosition;`	last frame cameraPosition
`uniform mat4 gbufferModelView;`	modelview matrix after setting up the camera transformations
`uniform mat4 gbufferModelViewInverse;`	inverse gbufferModelView
`uniform mat4 gbufferPreviousModelView;`	last frame gbufferModelView
`uniform mat4 gbufferProjection;`	projection matrix when the gbuffers were generated
`uniform mat4 gbufferProjectionInverse;`	inverse gbufferProjection
`uniform mat4 gbufferPreviousProjection;`	last frame gbufferProjection
`uniform mat4 shadowProjection;`	projection matrix when the shadow map was generated
`uniform mat4 shadowProjectionInverse;`	inverse shadowProjection
`uniform mat4 shadowModelView;`	modelview matrix when the shadow map was generated
`uniform mat4 shadowModelViewInverse;`	inverse shadowModelView
`uniform float wetness;`	rainStrength smoothed with wetnessHalfLife or drynessHalfLife
`uniform float eyeAltitude;`	view entity Y position
`uniform ivec2 eyeBrightness;`	x = block brightness, y = sky brightness, light 0-15 = brightness 0-240
`uniform ivec2 eyeBrightnessSmooth;`	eyeBrightness smoothed with eyeBrightnessHalflife
`uniform ivec2 terrainTextureSize;`	not used
`uniform int terrainIconSize;`	not used
`uniform int isEyeInWater;`	1 = camera is in water, 2 = camera is in lava, 3 = camera is in powdered snow
`uniform float nightVision;`	night vision (0.0-1.0)
`uniform float blindness;`	blindness (0.0-1.0)
`uniform float screenBrightness;`	screen brightness (0.0-1.0)
`uniform int hideGUI;`	GUI is hidden
`uniform float centerDepthSmooth;`	centerDepth smoothed with centerDepthSmoothHalflife
`uniform ivec2 atlasSize;`	texture atlas size (only set when the atlas texture is bound)
`uniform vec4 spriteBounds;`	sprite bounds in the texture atlas (u0, v0, u1, v1), set when MC_ANISOTROPIC
`uniform vec4 entityColor;`	entity color multiplier (entity hurt, creeper flashing when exploding)
`uniform int entityId;`	entity ID
`uniform int blockEntityId;`	block entity ID (block ID for the tile entity, only for blocks specified in `block.p`
`uniform ivec4 blendFunc;`	blend function (srcRGB, dstRGB, srcAlpha, dstAlpha)
`uniform int instanceId;`	instance ID when instancing is enabled (countInstances > 1), 0 = original, 1-N =

Table 3 – continued from previous page

Source	Value
`uniform float playerMood;`	player mood (0.0-1.0), increases the longer a player stays underground
`uniform int renderStage;`	render stage, see *Macros*, *J. Render stages*
1.17+ Options Below	
`uniform mat4 modelViewMatrix;`	model view matrix
`uniform mat4 modelViewMatrixInverse;`	modelViewMatrixInverse
`uniform mat4 projectionMatrix;`	projectionMatrix
`uniform mat4 projectionMatrixInverse;`	projectionMatrixInverse
`uniform mat4 textureMatrix = mat4(1.0);`	textureMatrix = mat4(1.0)
`uniform mat3 normalMatrix;`	normal matrix
`uniform vec3 chunkOffset;`	terrain chunk origin, used with attribute `vaPosition`
`uniform float alphaTestRef;`	alpha test reference value, the check is `if (color.a < alphaTestRef) dis`
`uniform float darknessFactor`	strength of the darkness effect (0.0-1.0)
`uniform float darknessLightFactor`	lightmap variations caused by the darkness effect (0.0-1.0)

36.10.2 GBuffers uniforms

Note

Programs: basic, textured, textured_lit, skybasic, skytextured, clouds, terrain, terrain_solid, terrain_cutout_mip, terrain_cutout, damagedblock, water, block, beaconbeam, item, entities, armor_glint, spidereyes, hand, hand_water, weather

Source	Value
`uniform sampler2D gtexture;`	0
`uniform sampler2D lightmap;`	1
`uniform sampler2D normals;`	
`uniform sampler2D specular;`	3
`uniform sampler2D shadow;`	waterShadowEnabled ? 5 : 4
`uniform sampler2D watershadow;`	4
`uniform sampler2D shadowtex0;`	4
`uniform sampler2D shadowtex1;`	5
`uniform sampler2D depthtex0;`	6
`uniform sampler2D gaux1;`	7 <custom texture or output from deferred programs>
`uniform sampler2D gaux2;`	8 <custom texture or output from deferred programs>
`uniform sampler2D gaux3;`	9 <custom texture or output from deferred programs>
`uniform sampler2D gaux4;`	10 <custom texture or output from deferred programs>
`uniform sampler2D colortex4;`	7 <custom texture or output from deferred programs>
`uniform sampler2D colortex5;`	8 <custom texture or output from deferred programs>
`uniform sampler2D colortex6;`	9 <custom texture or output from deferred programs>
`uniform sampler2D colortex7;`	10 <custom texture or output from deferred programs>
`uniform sampler2D colortex8;`	16 <custom texture or output from deferred programs>
`uniform sampler2D colortex9;`	17 <custom texture or output from deferred programs>
`uniform sampler2D colortex10;`	18 <custom texture or output from deferred programs>
`uniform sampler2D colortex11;`	19 <custom texture or output from deferred programs>
`uniform sampler2D colortex12;`	20 <custom texture or output from deferred programs>
`uniform sampler2D colortex13;`	21 <custom texture or output from deferred programs>
`uniform sampler2D colortex14;`	22 <custom texture or output from deferred programs>

continues on next page

Table 4 – continued from previous page

Source	Value
`uniform sampler2D colortex15;`	23 <custom texture or output from deferred programs>
`uniform sampler2D depthtex1;`	11
`uniform sampler2D shadowcolor;`	13
`uniform sampler2D shadowcolor0;`	13
`uniform sampler2D shadowcolor1;`	14
`uniform sampler2D noisetex;`	15

36.10.3 Shadow uniforms

Note

Programs: shadow, shadow_solid, shadow_cutout

Source	Value
`uniform sampler2D tex;`	0
`uniform sampler2D gtexture;`	0
`uniform sampler2D lightmap;`	1
`uniform sampler2D normals;`	
`uniform sampler2D specular;`	3
`uniform sampler2D shadow;`	Is waterShadowEnabled true? Yes: 5, no: 4
`uniform sampler2D watershadow;`	4
`uniform sampler2D shadowtex0;`	4
`uniform sampler2D shadowtex1;`	5
`uniform sampler2D gaux1;`	7 <custom texture>
`uniform sampler2D gaux2;`	8 <custom texture>
`uniform sampler2D gaux3;`	9 <custom texture>
`uniform sampler2D gaux4;`	10 <custom texture>
`uniform sampler2D colortex4;`	7 <custom texture>
`uniform sampler2D colortex5;`	8 <custom texture>
`uniform sampler2D colortex6;`	9 <custom texture>
`uniform sampler2D colortex7;`	10 <custom texture>
`uniform sampler2D colortex8;`	16 <custom texture>
`uniform sampler2D colortex9;`	17 <custom texture>
`uniform sampler2D colortex10;`	18 <custom texture>
`uniform sampler2D colortex11;`	19 <custom texture>
`uniform sampler2D colortex12;`	20 <custom texture>
`uniform sampler2D colortex13;`	21 <custom texture>
`uniform sampler2D colortex14;`	22 <custom texture>
`uniform sampler2D colortex15;`	23 <custom texture>
`uniform sampler2D shadowcolor;`	13
`uniform sampler2D shadowcolor0;`	13
`uniform sampler2D shadowcolor1;`	14
`uniform sampler2D noisetex;`	15

36.10.4 Composite and deferred uniforms

Note

Programs: composite, composite1, composite2, composite3, composite4, composite5, composite6, composite7, final, deferred, deferred1, deferred2, deferred3, deferred4, deferred5, deferred6, deferred7

Source	Value
`uniform sampler2D gcolor;`	0
`uniform sampler2D gdepth;`	1
`uniform sampler2D gnormal;`	2
`uniform sampler2D composite;`	3
`uniform sampler2D gaux1;`	7
`uniform sampler2D gaux2;`	8
`uniform sampler2D gaux3;`	9
`uniform sampler2D gaux4;`	10
`uniform sampler2D colortex0;`	0
`uniform sampler2D colortex1;`	1
`uniform sampler2D colortex2;`	2
`uniform sampler2D colortex3;`	3
`uniform sampler2D colortex4;`	7
`uniform sampler2D colortex5;`	8
`uniform sampler2D colortex6;`	9
`uniform sampler2D colortex7;`	10
`uniform sampler2D colortex8;`	16
`uniform sampler2D colortex9;`	17
`uniform sampler2D colortex10;`	18
`uniform sampler2D colortex11;`	19
`uniform sampler2D colortex12;`	20
`uniform sampler2D colortex13;`	21
`uniform sampler2D colortex14;`	22
`uniform sampler2D colortex15;`	23
`uniform sampler2D shadow;`	Is waterShadowEnabled true? Yes: 5, no: 4
`uniform sampler2D watershadow;`	4
`uniform sampler2D shadowtex0;`	4
`uniform sampler2D shadowtex1;`	5
`uniform sampler2D gdepthtex;`	6
`uniform sampler2D depthtex0;`	6
`uniform sampler2D depthtex1;`	11
`uniform sampler2D depthtex2;`	12
`uniform sampler2D shadowcolor;`	13
`uniform sampler2D shadowcolor0;`	13
`uniform sampler2D shadowcolor1;`	14
`uniform sampler2D noisetex;`	15

36.10.5 Custom uniforms

Define custom variables and uniforms using general mathematical expressions with brackets, constants, variables, operators and functions. The uniforms are sent to the shaders, the variables can be used in other variables or uniforms. The custom uniforms are updated on program change.

Important

Lightmap texture matrix, 1.17+:

```
const mat4 TEXTURE_MATRIX_2 = mat4(
    vec4(0.00390625, 0.0, 0.0, 0.0),
    vec4(0.0, 0.00390625, 0.0, 0.0),
    vec4(0.0, 0.0, 0.00390625, 0.0),
    vec4(0.03125, 0.03125, 0.03125, 1.0)
);
```

Table 6: Constants

Name	Type	Meaning
`pi`	Float, constant	Floating point, `3.1415926`
`true`	Boolean, constant	Truthy boolean
`false`	Boolean, constant	False boolean

The available biome ids, categories and precipitation types are defines as constants. For example: `BIOME_PLAINS`, `BIOME_DESERT`, `BIOME_EXTREME_HILLS`, etc.

Table 7: Parameters (float)

Constant	Meaning
biome	biome id
biome_c	0 to 16 (CAT_NONE, CAT_TAIGA, CAT_EXTREME_HILLS, CAT_JUNGLE, CAT_MESA, CAT_PLAINS, CAT_SAVANNA, CAT_ICY, CAT_THE_END, CAT_BEACH, CAT_FOREST, CAT_OCEAN, CAT_DESERT, CAT_RIVER, CAT_SWAMP, CAT_MUSHROOM, CAT_NETHER)
biome_p	0 to 2 (PPT_NONE, PPT_RAIN, PPT_SNOW)
temperature	0.0 to 1.0
rainfall	0.0 to 1.0 (humidity)

Rain/snow is rendered for `biome_precipitation != PPT_NONE`. If `temperature >= 0.15`, rain is rendered, otherwise snow.

The fixed scalar uniforms are also available as parameters. For example: `heldItemId`, `worldTime`, `moonPhase`, etc.

- **Vector** elements can be accessed with suffix `.x`, `.y`, and `.z`. For example: `sunPosition.y`
- **Color** elements can be accessed with suffix `.r`, `.g`, and `.b`. For example: `skyColor.r`
- **Matrix** elements can be accessed by row and column index. For example: `gbufferModelView.0.1`

Important

The dynamic uniforms `entityColor`, `entityId`, `blockEntityId`, `fogMode`, and `fogColor` can not be used as parameters as they may change many times per program.

See also

Parameters (boolean), operators, functions are in *Values*.

Table 8: :header: "Name", "Return" :widths: 20, 40, 100

`smooth([id], val, [fadeInTime, [fadeOutTime]])`	Smooths a variable with custom fade-in time | The `[id]` must be unique; if not specified, it is generated automatically | Default fade time is 1 sec
Vector functions:	

- `vec2(x, y)`
- `vec3(x, y, z)`
- `vec4(x, y, z, w)`

Example

```
variable.bool.isBiomeDark=in(biome, BIOME_RIVER, BIOME_FOREST)
variable.float.valBiomeDark=smooth(1, if(isBiomeDark, 1, 0), 5)
variable.float.valHurtDark=smooth(2, if(is_hurt, 1.3, 0), 0, 10)
variable.float.valSwordDark=smooth(3, if(heldItemId == 276, 1, 0), 0.5, 0.5)
uniform.float.screenDark=max(valBiomeDark, valHurtDark, valSwordDark)
uniform.vec3.screenDark3=vec3(screenDark, heldItemId, biome)

# More generally,
uniform.<float|int|bool|vec2|vec3|vec4>.<name>=<expression>
variable.<float|int|bool|vec2|vec3|vec4>.<name>=<expression>
```

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

36.11 Usages

36.11.1 Depth buffers usage

Name	Usage
`depthtex0`	everything
`depthtex1`	no translucent objects (water, stained glass)
`depthtex2`	no translucent objects (water, stained glass), no handheld objects

36.11.2 Shadow buffers usage

Name	Usage
`shadowtex0`	everything
`shadowtex1`	no translucent objects (water, stained glass)

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

36.12 Overview

The Shaders mod makes use of a deferred rendering pipeline.

The `gbuffer` shaders come first in the pipeline. They render data to textures that will be sent to the composite shaders. Optional composite shaders can be added after the shadow map (shadowcomp), before terrain (prepare) and before water rendering (deferred). The composite shaders then render to textures that will be sent to the final shader.

The final shader renders directly to the screen.

36.13 Dimension shaders

Shaders can be separated by world dimension by placing them in folder `/shaders/world<id>`, where `id` is the world dimension. When the world folder is present, the shaders will be loaded only from there; they ignores the default folder. Creating an empty world folder effectively disables the shaders for that world dimension. Mod world dimensions should also work.

Only `.vsh` and `.fsh` files are loaded from the dimension folder.

Example:

- `/shaders`: default shaders
- `/shaders/world-1`: nether shaders
- `/shaders/world1`: end shaders

Dimension folders are also scanned for options. The options in dimension folders may be given different names to avoid conflict with default values.

The `#include` directive found in `.vsh` and `.fsh` files is replaced with the contents of the included file:

Relative, look in same folder

```
#include "const.inc"
```

Absolute, start from base folder `shaders/`

```
#include "/world-55/lib.inc"
```

Included files may include other files.

Caution

The maximum include depth is limited to **10**.

To avoid code duplication on nested inclusions, the following can be used:

```
// File A

#ifndef INCLUDE_A
#define INCLUDE_A
...
#endif
```

Note

When Minecraft is started with *argument*, `-Dshaders.debug.save=true`, then the final shaders will be saved in `shaderpacks/debug/`.

36.14 Files

All shader files are placed in the folder `shaders/` of the shader pack. The shader source files use the name of the program in which they are to be used with extension depending on their type.

Extension	Type
`.csh`	Compute shader
`.vsh`	Vertex shader
`.gsh`	Geometry shader
`.fsh`	Fragment shader

Geometry shaders need either OpenGL 3.2 with layout qualifiers or the extension `GL_ARB_geometry_shader4` (`GL_EXT_geometry_shader4`) with configuration "`maxVerticesOut`".

36.15 Compute shaders

A list of compute shaders can be attached to every program except `gbuffers` programs. They are named like the program with optional suffix, for example "`composite.csh`", "`composite_a.csh`" ... "`composite_z.csh`".

Compute shaders run before the program and can read from all buffers using texture samplers. They can read and write to `colortex0-5` and `shadowcolor0-1` buffers as images using the aliases `colorimg0-5` and `shadowcolorimg0-1`, for example: `layout (rgba8) uniform image2D colorimg0;`.

Compute shaders need at least "`#version 430`" and local size definition, for example: `layout (local_size_x = 16, local_size_y = 16) in;`. Work groups are defined either fixed via `const ivec3 workGroups = ivec3(50, 30, 1);` or relative to render size via `const vec2 workGroupsRender = vec2(0.5f, 0.5f);`. The default configuration is `const vec2 workGroupsRender = vec2(1.0f, 1.0f);`, which executes the compute shader once per pixel.

36.16 Image access

All programs can read and write to `colorimg0-5` and `shadowcolorimg0-1` using `imageLoad()` and `imageStore()`.

36.17 Attributes

Source	Value	Comment
`in vec3 vaPosition;`	position (x, y, z)	1.17+, for terrain it is relative to the chunk origin, see 'chunkOffset'
`in vec4 vaColor;`	color (r, g, b, a)	1.17+
`in vec2 vaUV0;`	texture (u, v)	1.17+
`in ivec2 vaUV1;`	overlay (u, v)	1.17+
`in ivec2 vaUV2;`	lightmap (u, v)	1.17+
`in vec3 vaNormal;`	normal (x, y, z)	1.17+
`in vec3 mc_Entity;`	xy = blockId, renderType	'blockId' is used only for blocks specified in 'block.properties'
`in vec2 mc_midTexCoord;`	st = midTexU, midTexV	Sprite middle UV coordinates
`in vec4 at_tangent;`		xyz = tangent vector, w = handedness
`in vec3 at_velocity;`	vertex offset to previous frame	In view space, only for entities and block entities
`in vec3 at_midBlock;`	offset to block center in 1/64m units	Only for blocks

36.18 Block render layers

See *Block Render Layers*.

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

CHAPTER

THIRTYSEVEN

SYNTAX

This page details the rules and formats for specifying how OptiFine features work in a resource pack.

Fig. 1: A snippet of a properties file.

37.1 File naming rules

For any file in a resource pack to be visible to the game, it must match the regular expression `^[a-z0-9_.]+$`.

Hint

Must be lowercase, no spaces, a through z, 0 through 9, and _ only.

Textures must be in PNG format with the `.png` extension.

All text files must be encoded in UTF-8 without BOM. Do not use an ASCII encoding.

JSON files must follow the JSON specification strictly and have the `.json` extension.

37.2 File structure

Many OptiFine features use **properties files** to control how OptiFine-specific elements will work within a resourcepack.

Properties files are simple text files similar to the Windows INI format. Each line is a property, specified as `name=value`.

```
# Comments begin with a hashtag and are ignored.
property1=value
property2=some_other_value

# Blank lines are allowed.
property3=yet_another_value
```

All property names are case-sensitive: `renderpass` **is not the same** as `renderPass`. The order of properties within the file does not matter. Many properties have default values and can be omitted, and *in some cases the entire properties file is optional*.

Certain types of objects are used within properties files by different OptiFine features. Rather than describe these common types separately in each feature section, they are summarized here instead.

37.3 Paths

Often, OptiFine requires specifying a path to a texture or other file within the resource pack. A path is a "location" of where a file is.

The folder structure within a resource pack is deeply nested, so OptiFine has some shortcuts to make paths easier to write. Any of these options can be used to specify the same path.

Always use forward slashes "/" to separate folders.

Caution

Regardless of operating system, do not use backslashes "\", or the game will not properly recognize the path.

The below table summarises path shortcuts:

Symbol	Resolves to
None	`/assets/minecraft/optifine/`
`./`	`/assets/minecraft/optifine/`
`../`	**Not valid syntax**
`~`	`/assets/minecraft/optifine/`
`namespace:`	`/assets/namespace/`

37.3.1 Bare filename

Bare filenames with no slashes will refer to the file relative to `/assets/minecraft/optifine/`, **ignoring subfolders**.

```
texture=texture.png
```

37.3.2 Dot and dot-dot

You can use `./` to denote the current directory, regardless of location. **This does work in subfolders**. `..` can be used to travel up a folder, into the parent directory.

```
texture=./texture.png
texture=../texture.png
texture=../../subfolder/texture.png
```

37.3.3 Tildes

The tilde character `~` can be used to refer to `/assets/minecraft/optifine/`.

```
texture=~/texture.png
texture=~/subfolder/texture.png
```

37.3.4 Namespaces

An optional "namespace" prefix can be added. This example refers to exactly the same "creeper.png" file as default:

```
texture=minecraft:textures/entity/creeper/creeper.png
```

For textures used by other mods, the namespace will be something other than `minecraft`:

```
texture=MODID:subfolder/texture.png
```

This refers to `/assets/MODID/subfolder/texture.png`, **not to** `/assets/minecraft/MODID/subfolder/texure.png`.

Namespaces can also apply to blocks, items, and biome IDs.

37.4 Biomes

See also

See this page for the 1.13 biome changes

For features that call for a list of biomes, use this page. Biomes added by mods can also be used.

```
biomes=ocean deep_ocean river beach
```

37.5 NBT

OptiFine supports conditions based off of an item's NBT.

If multiple NBT rules are provided, **all** of them must match.

A value starting with `!` **negates** the match *(match all EXCEPT this)*.

List indexes start at 0; 0 is the first element, 1 is the second. All Unicode codepoints can be escaped with `\uXXXX`, but this is not required and is recommended for non-visible characters or characters above U+007E TILDE ~.

Some examples include:

- Match if Y is in list X: `nbt.X.*=Y`
- Match specific value: `nbt.X=Y`
- Match number in range: `nbt.X=range:0-100`
- Match if Y is first entry of list X: `nbt.X.0=Y`
- Match if X exists: `nbt.X=exists:true`
- Match item display name: `nbt.display.Name=My Sword`
- Match item display name with escapes: `nbt.display.Name=\u00a74\u00a7oMy Sword`
- Match item lore *(first line only)*: `nbt.display.Lore.0=My Lore Text`
- Match item lore *(any line)*: `nbt.display.Lore.*=My Lore Text`

37.5.1 Exists

You can check for the presence of an NBT value for a given tag by using the `exists:` prefix. It takes only two values: `true` and `false`.

`true` will apply if the NBT key given has any value associated with it. `false` will apply if the NBT key given is not present in the item's NBT.

```
# Apply to items that have any lore
nbt.display.Lore=exists:true

# Apply to items that have no lore at all
nbt.display.Lore=exists:false
```

New in version I1.

37.5.2 Raw

See also

Familiarize yourself with (S)NBT data types.

Raw exact values may be matched by using the `raw:` prefix. All values for `raw:` must be typed exactly as they are shown in the game's output log.

Raw can be used to match types explicitly, such as bytes, shorts, floats, doubles, etc.

```
nbt.display.Name=raw:'[{"text":"Dark red italics","italic":true,"color":"dark_red"}]'
nbt.3.OnGround=raw:1b
nbt.1.UUID=raw:'[I;-1668711424,-1434628111,-1613745989,1749596493]'
nbt.someRandomShort=raw:64s
```

The value must match exactly as the game reports it.

`raw:` can also be combined with all other prefixes. For example: `raw:pattern:`, `raw:regex:`, etc.

New in version I1.

37.5.3 Blocks and items

See also

See this page for the ID to name conversions
See this page for the 1.13 block & item changes

Danger

Do not use 1.7- IDs in 1.13+ packs!

In 1.13, many variant blocks were "flattened" to several simple blocks, and the block metadata was removed from their ID.

The block name format is `[namespace:]name[:property1=value1,...:property2=value1,...]`. Optional parts are in angle brackets `[ ]`. The default namespace is `minecraft`.

- Before 1.7, items can **only** be specified by ID.
- Since 1.7, items can **also** be specified by name, alongside ID.
- Since 1.13, items can **only** be specified by name, **not** ID.

The block IDs continue to exist within the game internally, but can no longer be specified in the configuration files as they are unstable. For example, Stone used to be ID `1`, but is now named `minecraft:stone`.

As with textures, the "`minecraft:`" prefix is optional, so just "`stone`" will also work, as "`minecraft:`" is the default.

In the event of mods, they will likely use a namespace other than `minecraft:`, so the prefix will be required for referring to modded items.

```
blocks=minecraft:oak_stairs:facing=east,west:half=bottom
blocks=oak_stairs:facing=east,west:half=bottom
```

The above will apply to oak stairs that are east or west facing, and are the bottom half. The `minecraft:` namespace is optional, so it can also be omitted.

37.5.4 Strings

This section is about matching strings and values using different matching methods. Strings can be matched in several ways.

Important

Any backslashes must be doubled. Matching backslashes within a regular expression or wildcard must be quadrupled.

- `nbt.display.name=regex:\\d+`
- `nbt.display.name=regex:\\\\`
- `nbt.display.name=/\\/\\`
- `nbt.display.name=regex:\d+`
- `nbt.display.name=regex:\\` (for matching \\)
- `nbt.display.name=/\/\\` (missing a backslash)

Exact value

For strings, you may either type the string directly: `Letter to Herobrine` matches the exact string `Letter to Herobrine` and nothing else. You may also use the *Raw* syntax.

Wildcards

Wildcards are shorter versions of regular expressions, in that they only support two unique constructs:

- The symbol `?` matches any text, as long as it exists.
- The symbol `*` matches any text, regardless of its presence.

The wildcard `*` is equivalent to the regular expression `.*` (0 or more).
The wildcard `?` is equivalent to the regular expression `.+` (1 or more).

Wildcard patterns must start with either `pattern:` or `ipattern:`.

Note

The meanings of `?` and `*` are swapped in contrast to regular expressions. It is unknown if this is a bug.

pattern

`pattern` is case-sensitive.

Example: `pattern:*Letter to ?`

- `Letter to Herobrine`
- `Letter to a creeper`
- `My Letter to John` (note the uppercase L)
- `letter to Herobrine` (case sensitivity)
- `Letter from Herobrine` (doesn't match "to")
- `Letter to` (doesn't match "?")

ipattern

`ipattern` is case-insensitive, meaning that "ABC" and "abc" are viewed the same.

Example: `ipattern:*Letter to ?`

- `Letter to Herobrine`
- `My letter to Herobrine`
- `Letter to a creeper`
- `letter to Herobrine`
- `letter to STEVE!`
- `A letter from CJ`
- `Letter from Herobrine`

Regular expressions

Regular expressions are strings that create "patterns" that other strings can be matched against. Patterns can be as simple as `.`, to IP address validation. OptiFine supports two types of regular expressions, depending on the case-sensitivity. In addition, OptiFine **does not** support expression flags.

The syntax understood by OptiFine is the Java syntax. OptiFine regular expressions are not multiline and are not global ("/gm").

Regular expressions must start with either `regex:` or `iregex:`.

regex

A regular expression.

Example: `regex:Letter (to|from) .*`

- `Letter to Herobrine`
- `Letter from Herobrine`
- `letter to Herobrine` (letter case)
- `A Letter to Herobrine` (`A` is not in expression)

iregex

`iregex` is similar to `regex`, except that it is case-insensitive.

Example: `iregex:Letter (to|from) .*`

- `Letter to Herobrine`
- `Letter from Herobrine`
- `letter to Herobrine`
- `LETTER TO HEROBRINE`
- `A Letter to Herobrine` (*A* is not in expression)
- `LETTER TOFROM HEROBRINE`

37.5.5 Numbers

Numbers can be matched simply by typing the number. Additionally, you can match more than one number as well as a range of numbers with lists and ranges.

Ranges

Important

There is no range for less than or equal to; use a *full range*: `0-100`, **not** `-100` to match 100.

Inclusive ranges between numbers are defined with a `-` between those digits. If there is no number present on the **right** side of the `-`, the range will match to positive infinity.

Ranges can be combined and intermixed with lists.

For example,

```
# 1, 2, 3
numbers=1-3

# Multiple ranges
# 1 through 3, or 6, or 8, or 10 through 15
# 1, 2, 3, 6, 8, 10, 11, 12, 13, 14, 15
numbers=1-3 6 8 10-15

# Greater than or equal to
# 100, or 200, or 5340, or 25902, etc.
numbers=100-

# Negative number, not a range
# Only matches negative 100, not -4, -7, or -101
numbers=-100
```

Since 1.18 (with OptiFine H5), negative values may be specified also. When used in a range, they must be surrounded by parentheses.

To use negative numbers in a range, parentheses must be around those negative numbers.

```
list=(-3)-(-1)
```

These can be combined to create vast ranges of possible numeric values.

Range

Note

range: was added in 1.19.2 I1 pre1.

Ranges of numbers can also be matched by using the *range:* prefix. They use the same syntax as the above, except they explicitly have *range:*.

```
nbt.N=range:A-B
nbt.number=range:0-100
```

Lists

Lists are defined with a space between each number. Multiple values are listed separately, split with spaces.

Each "entry" in the list can be either a single number, or a range.

For example,

```
numbers=1 2 3 4 5 6
numbers=10 70 23 -6 210
numbers=(-100)-200 500 900-
```

37.5.6 Colors

Note

There is no support for HSL, CMYK, etc.

Color values are specified in hexadecimal RGB format, without the leading hashtag:

```
# White
color=ffffff

# Black
color=000000

# Red
color=ff0000

# Green
color=00ff00

# Blue
color=0000ff
```

37.6 Blending methods

See also

See Blend modes on Wikipedia for some illustrations.

When two or more textures are combined, OptiFine offers several options for specifying the blending operation.

"This" or "current" texture refers to the texture currently being applied. "Previous" refers to whatever has been rendered so far, which could be a single texture or the result of an earlier blending operation.

`replace`	Replace the previous layer entirely with the current bitmap. No blending and only simple on/off transparency.
`alpha`	Blend the two textures using this texture's alpha value. This is the most common type of blending.
`overlay`	RGB value > 0.5 brightens the previous image, < 0.5 darkens. color is a synonym for this method.
`add`	Add this texture's RGB values multiplied by alpha to the previous layer.
`subtract`	Subtract this texture's RGB values from the previous layer.
`multiply`	Multiply the previous RGB values by this texture's RGB values
`dodge`	Add this texture's RGB values to the previous layer.
`burn`	$RGB_{new} = (1 - RGB_{current}) * RGB_{previous}$
`screen`	$RGB_{new} = 1 - (1 - RGB_{current}) * (1 - RGB_{previous})$

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

CHAPTER

THIRTYEIGHT

TEXTURE PROPERTIES

File location

/assets/minecraft/optifine/texture.properties

Texture Properties define how a texture is to be interpreted.

See also

See *Shaders - Development*: *Macros*, and *Textures*.

See also

See https://github.com/rre36/lab-pbr/wiki for information on what labPBR is.

Fig. 1: PBR for pillow fluff.

38.1 Properties

38.1.1 `format`

Values: `lab-pbr` or `1.3`
Optional

The texture format used for normal and specular shader textures.

New in version G6: lab-pbr

38.2 JSON schema

Note

Although this page is `.properties` based, it can be mapped to JSON.

```
{
        "$schema": "http://json-schema.org/draft/2020-12/schema",
        "$id": "https://gitlab.com/whoatemybutter/optifinedocs/-/blob/master/schemas/
↪texture_properties.schema.json",
        "title": "Texture Properties",
        "description": "Texture Properties define how a texture is to be interpreted.",
        "type": "object",
        "properties": {
                "format": {
                        "enum": ["lab-pbr", "1.3"],
                        "description": "The texture format used for normal and specular␣
↪shader textures."
                }
        },
        "additionalProperties": false
}
```

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

Assumes latest OptiFine version.
Updated to commit 8ed2130d.
Open source at https://gitlab.com/whoatemybutter/optifinedocs.

INDEX

U

W