PortalNetwork

PortalNetwork Developers

Table of contents

Introduction

Portals

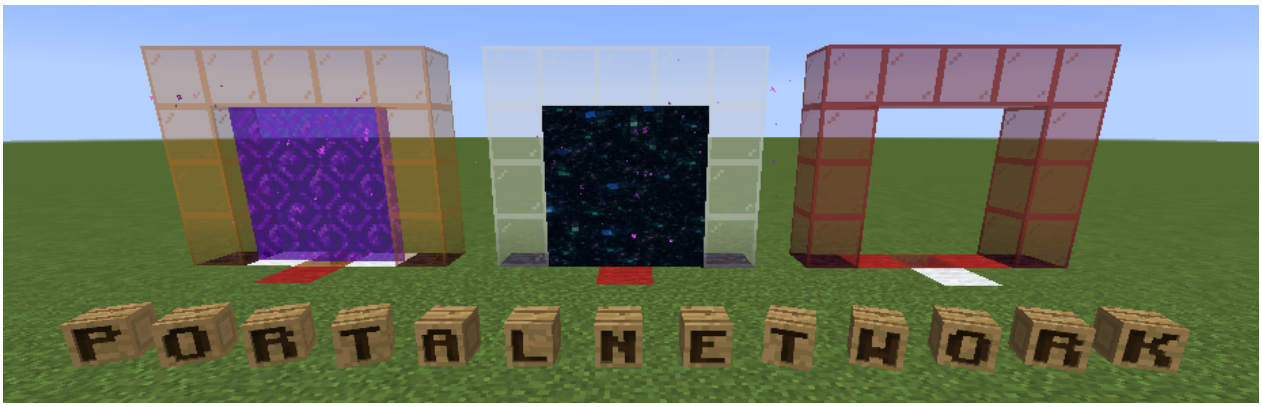
Configuration

Permissions and Commands

Contributing

API

Introduction



What is PortalNetwork?¶

PortalNetwork is a portal system that gives control back to the players to be able to create portals that can dial each other. It supports different types of portals and allows players to change the portal shape. Vehicles are also supported.

Features¶

- Players can create up to 256 networks of portals with up to 16 portals in each network. Any portal can dial any other portal in the same network.
- Three different looks of portals. Nether, End Gateway and Hidden (Invisible) Portals
- Portals can exist on different worlds and can dial each other across worlds.
- Vehicles like Minecarts, Horses, Pigs and boats will teleport as well and maintain their passengers.
- Blocks other than Obsidian can be placed in the portal area. This allows rails to be placed for example.
- Portals only need a base to be built. When they are dialled they will create their own frames if necessary and clean up after.
- Whatever enters one portal will maintain its relative orientation when existing another portal. This includes existing the back of the portal when entering the front
- Relative velocity is also maintained. This means flying through with an elytra works well.

- If a portal has one side blocked (for example it is against a wall) then the player existing it will be flipped 180 to exit out the other side. If both sides are blocked then you have a naughty player.
- An API to allow plugins to add additional portal types.

Quickstart¶

1. Get a `portalblock` using the command `/pn give`
2. Place the `portalblock` down and put white wool on either side of it. This defines the *network* the portal belongs to; in this case white/white.
3. Place a third piece of wool on the side you wish to be the front. This defines the *address* of the portal within its network. Altogether it should look like the shape of the T tetris block.

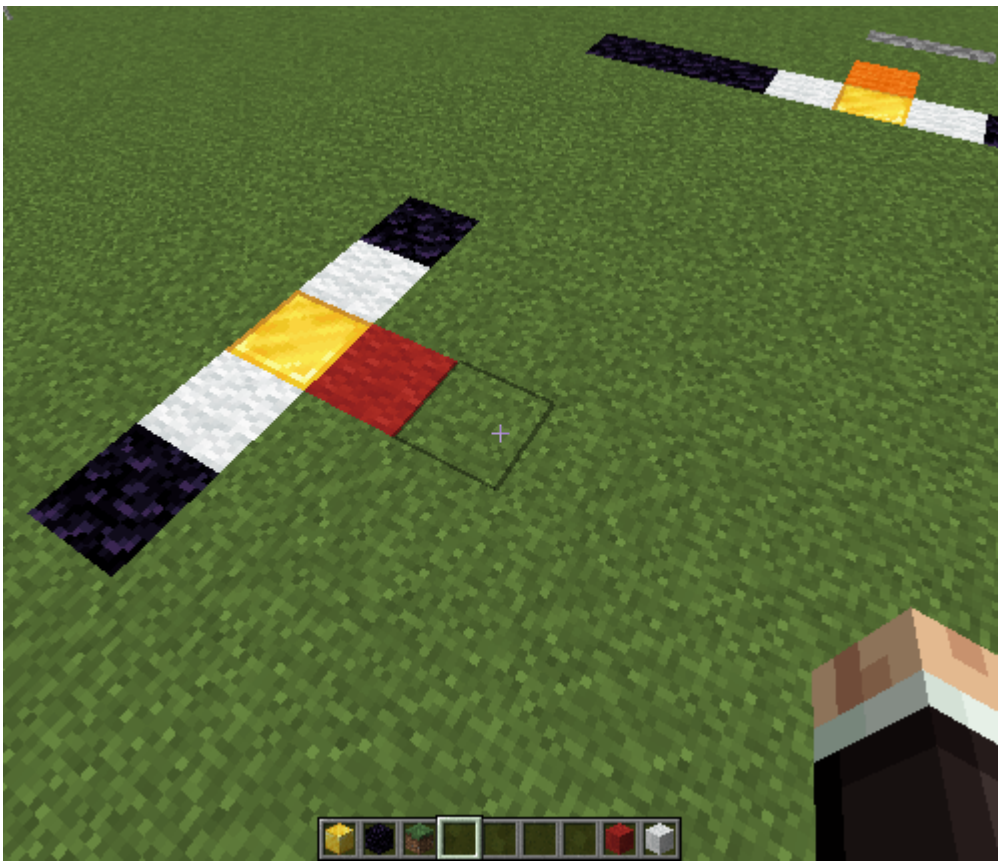
Portal Base

4. Optionally place obsidian on either side of the *network* wool blocks to make the portal wider

Portal Base made wider

5. Create another portal using steps 1-4 somewhere else. Make sure the `address` colour is different to your first portal.

Second Portal



6. Right click anywhere on the base to dial the other portal and step through. Note that the frame will turn the colour of the portal being dialled and the other portal will be dialled back.

Dialled Portal

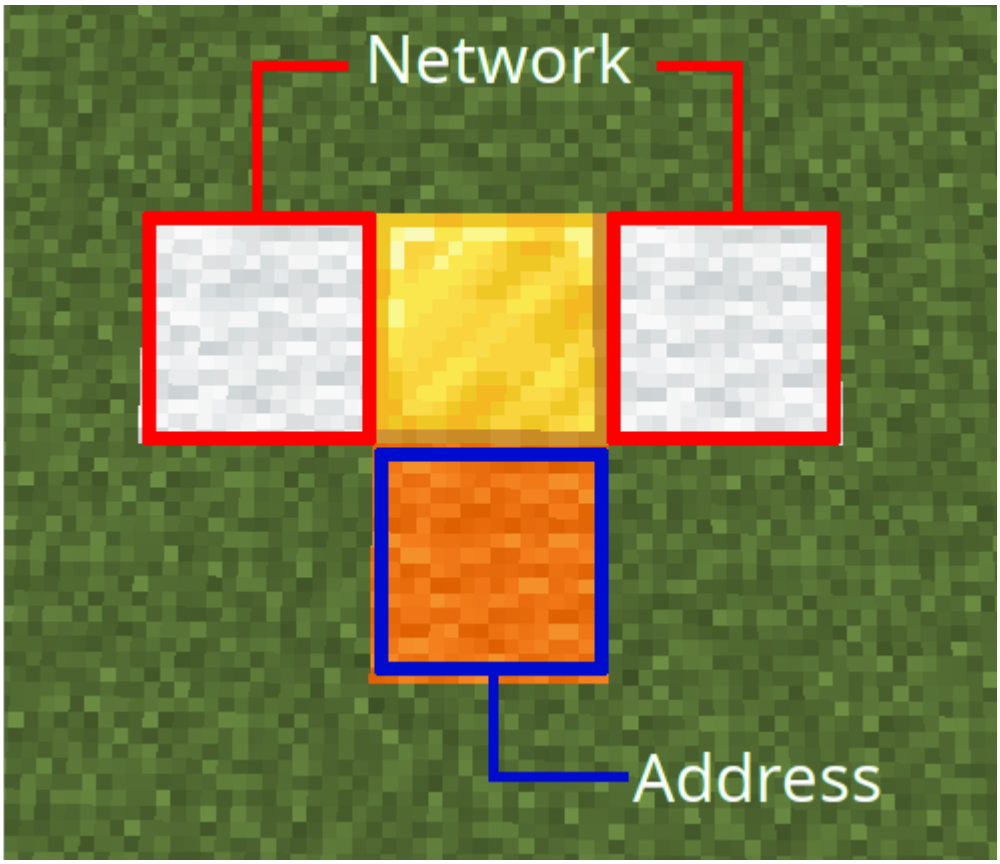
7. Right click again to dial the next colour. In this case as there is no next colour the portals will deactivate.

Last update:

Portals

Building a Portal¶

At a minimum a portal is built using just 4 blocks put together in the shape of a T shape Tetromino. A `portalblock` and 3 coloured wool blocks. Right clicking any of these blocks will attempt to dial the next colour portal in the network.



Network¶

The blocks to either side of the portal designate the *network* to which it belongs. If you think of a phone number the *network* would be like an area code. Any portal can dial any other portal that belongs to the same network. As there are 16 colours this means that there is a total of 256 networks available.

Address¶

The third coloured wool defines the portal *address*. The address must be unique amongst all portals that belong to the same *network*. As there are 16 colours this means that you can have a total of 16 portals able to dial each other per network.

Bug

At the moment it is undefined what happens if 2 portals have the same address. This will be rectified shortly.

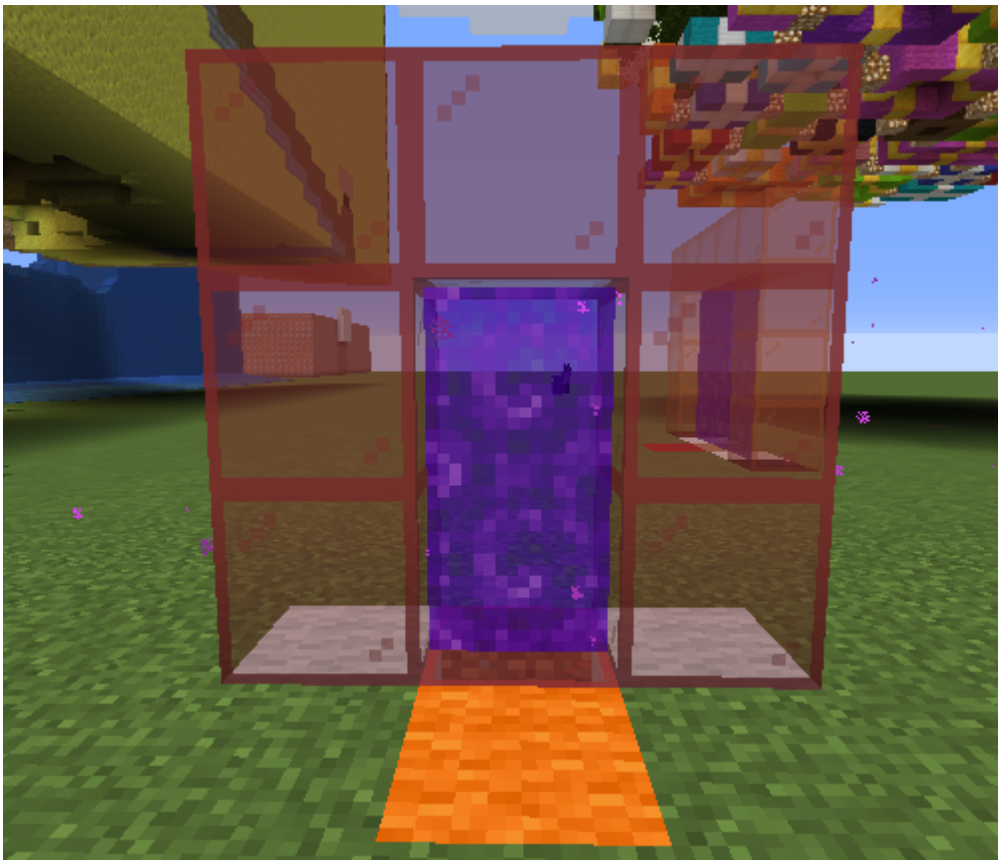
Portal Types¶

The `portalblock` in the middle will define what type of portal will be generated. This is purely cosmetic and any portal, regardless of its type, can connect to any other on the same network.

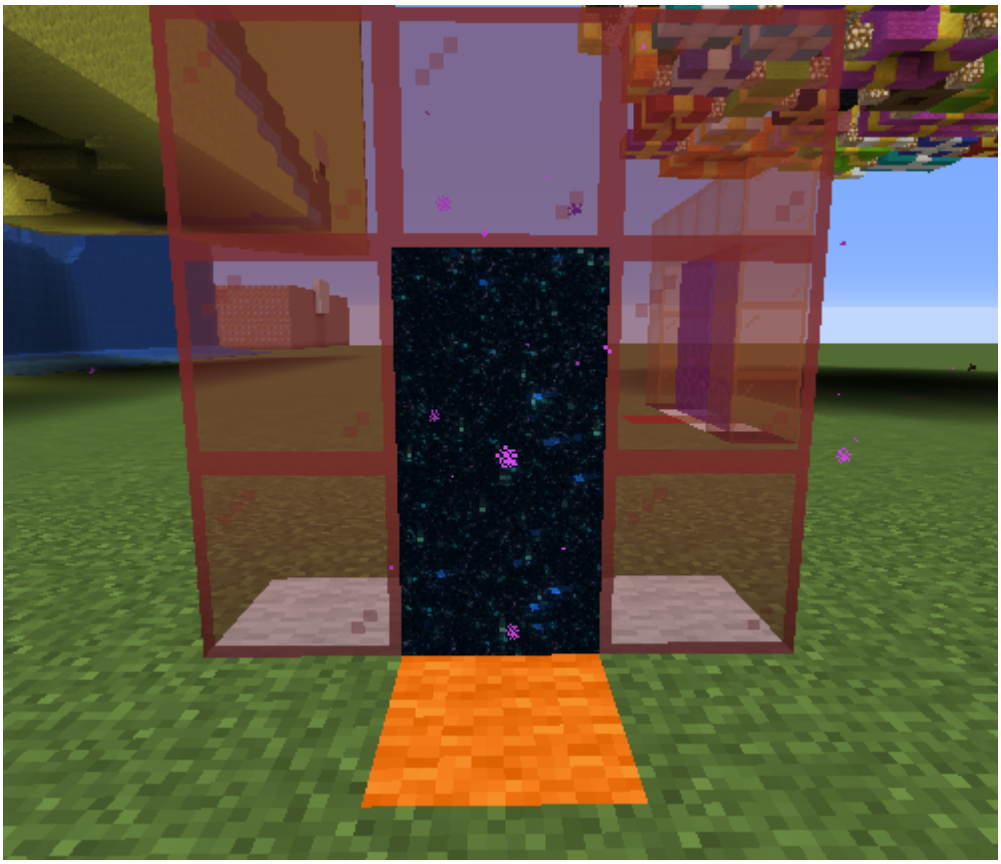
The three built-in portal types are:

1. Nether
2. End
3. Hidden

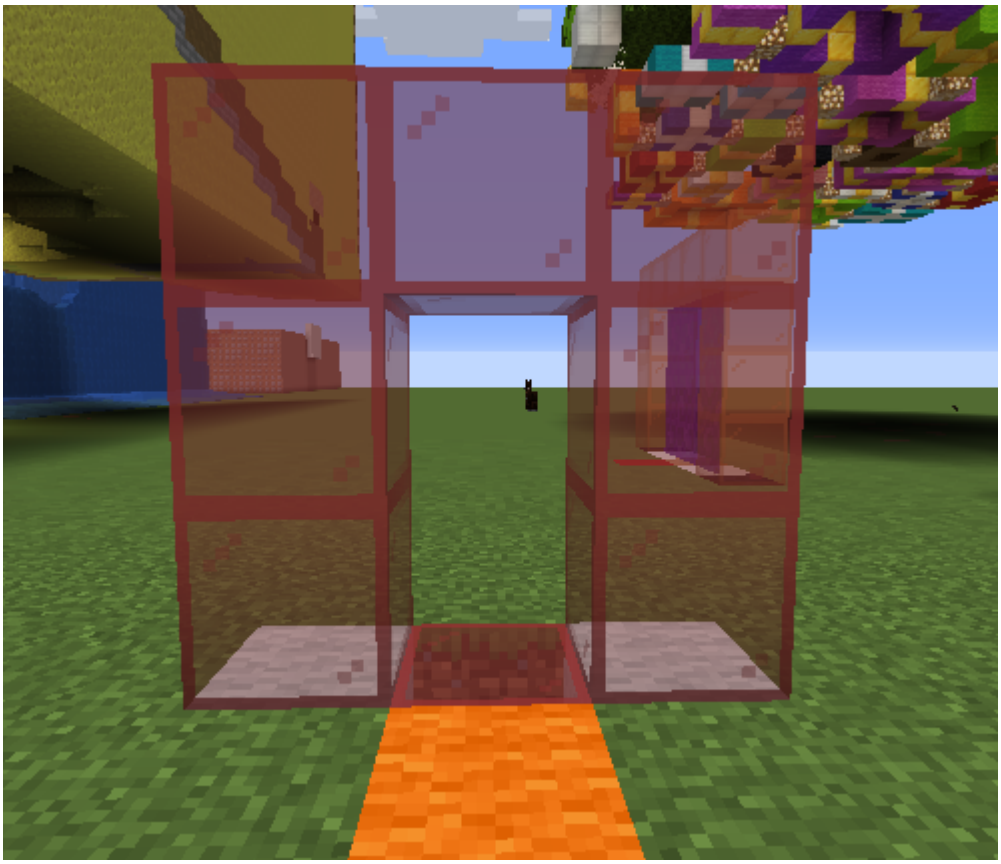
Nether¶



End¶

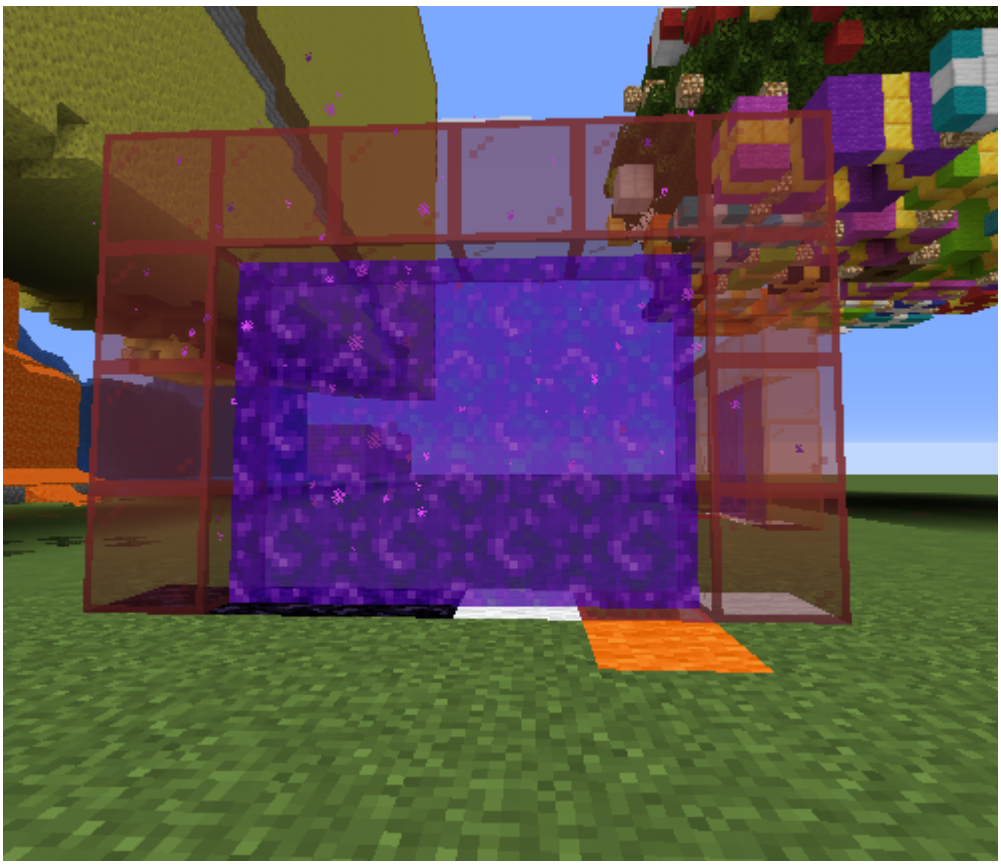


Hidden¶



Portal Size¶

The size of the portal is determined by how wide it is. You can make a portal wider by adding obsidian blocks to one or both sides of the network blocks. The height is half the width + 2 blocks. This includes the portal base and frame.



Building in Portal¶

When a portal activates it will create a frame for itself of the colour of the remote portal address and will fill its body with the type of portal being generated.

If any blocks are encountered in this area the portal will ignore them and generate itself around them. This means that any blocks placed in its frame will hide the frame and any blocks placed inside its body will not have a portal generated (but will still trigger a teleport if it is passable). It also means you can place blocks to hide the base layer of the portal and place rails for a minecart.

The only block that has special meaning is an obsidian block. Any obsidian block found in the portal area will prevent the portal generating any higher than just under that block and above that block will not trigger a teleport. This allows you to control the actual height of the portal.

Pressing sneak when placing a block on the base will not trigger a dial.

Direction and Velocity¶

When a player steps through a portal they will exit the destination portal with the same relative direction and velocity they entered. This means if they enter through the front left of a portal they will exit at the back left of the destination portal (relative to its direction). Velocity it maintained so flying through on an elytra will exit at the same height and velocity.

If a portal is blocked on one side, for example if it is built against a wall, then a player will always exit out of the unblocked side no matter which side they entered the source portal.

If a vehicle enters the portal then it and all its passengers will be teleported to the destination portal, maintaining relative direction and velocity. Yes you can have minecart on a rail go between portals.

Size Differences¶

Generally the player will exit out in the same relative position of the destination portal that they entered in the source portal.

If the destination portal is shorter then they will be clipped to its height if entering too high in the source portal.

Likewise if they are too far right or left in the source portal compared to the destination portal they will be clipped to the corresponding side.

Destinations¶

Portals can dial between worlds on the same server. This allows a portal in the Nether to dial a portal in the overworld or indeed any other custom work.

> **Info**
>
> Multiserver bungeecord support will come in due course.

Last update:

Configuration

Configuration File¶

File: `config.yml`

No configuration is necessary (yet)

default `config.yml`

```
## Don't touch ##
version: 1
```

Portal Data¶

File: `portaldata.yml`

This file is autogenerated and stores data about any portals created in the world. Do not touch this unless you know what you are doing.

Last update:

Permissions and Commands

Permission Summary¶

Permission	Description
portalnetwork.admin	Access to all Admin commands
portalnetwork.command.give	Access to `give`
portalnetwork.command.list	Access to `list`
portalnetwork.command.reload	Access to `reload`

No permission is necessary to build a portal.

Commands¶

Execute commands with `/portalnetwork <command>` or `/pn <command>`.

`give`¶

Give a player a `PortalBlock`

`/pn give [-type <portal type>] [<player name>]`

Where:

- `-type <portal type>`: Type of portal block to give. Defaults to `nether`. Built in types are `nether`, `end`, `hidden`.
- `<player name>`: Who to give it to. Defaults to yourself.

Permissions (any of)

- portalnetwork.admin
- portalnetwork.command.give

Examples

/pn give -type end Bob

/pn give

/pn give -type hidden

list¶

List all portals

`/pn list`

Permissions (any of)

- portalnetwork.admin
- portalnetwork.command.list

Examples

/pn list

reload¶

Reload all configuration.

`/pn reload`

Permissions (any of)

- portalnetwork.admin
- portalnetwork.command.reload

Examples

/pn reload

Last update:

Contributing¶

Here are some ways that you can help contribute to this project.

New ideas or Bug Reports¶

Need something? Found a bug? Or just have a brilliant idea? Head to the Issues *(https://github.com/Bundabrg/PortalNetwork/issues)* and create new one.

Contributing Code¶

If you know Java then take a look at open issues and create a pull request.

Do the following to build the code:

```
git clone https://github.com/Bundabrg/PortalNetwork
cd PortalNetwork
mvn clean package
```

Contributing Documentation¶

If you can help improve the documentation it would be highly appreciated. Have a look under the `docs` folder for the existing documentation.

The documentation is built using `mkdocs`. You can set up a hot-build dev environment that will auto-refresh changes as they are made.

Requirements¶

- python3
- pip3
- npm (only if changing themes)

Install dependencies by running:

```
pip3 install -r requirements.txt
```

Dev Environment¶

To start a http document server on `http://127.0.0.1:8000` execute:

```
mkdocs serve
```

Change PDF Theme¶

Edit the PDF theme under `docs/theme/pdf`. Rebuild by doing the following:

```
cd docs/theme/pdf
npm install
npm run build-compressed
```

This will update `pdf.css` under `docs/css/pdf.css`. Rebuilding the docs will now use the new theme.

Last update:

API

Maven¶

Add the following repository to your `pom.xml`

```
<!-- Bundabrg's Repo -->
<repository>
    <id>bundabrg-repo</id>
    <url>https://repo.worldguard.com.au/repository/maven-public</url
    <releases>
        <enabled>true</enabled>
    </releases>
    <snapshots>
        <enabled>true</enabled>
    </snapshots>
</repository>
```

Add the following dependency to your `pom.xml`

```
<dependency>
    <groupId>au.com.grieve</groupId>
    <artifactId>PortalNetwork</artifactId>
    <version>1.2.0</version>
    <scope>provided</scope>
</dependency>
```

New Portal Type¶

When creating a new Portal Type you will want to override either portals.BasePortal *(https://github.com/bundabrg/PortalNetwork/blob/master/src/main/java/au/com/grieve/portalnetwork/portals/BasePortal.java)* or one of the existing portal types.

The main methods you are interested in are:

- activate - Called when a portal activates.
- deactivate - Called when a portal deactivate
- getPortalIterator - Returns an iterator over the blocks that make up the body of a portal
- getPortalBaseIterator - Returns an iterator over the blocks that make up the base of a portal

• getPortalFrameIterator - Returns an iterator over the frame of a portal

Registering Portal Type¶

Use PortalManager#registerPortalClass *(https://github.com/bundabrg/PortalNetwork/blob/master/src/main/java/au/com/grieve/portalnetwork/PortalManager.java)* to register the new portal type.

For example if you have a new PortalType class called "DiamondPortal" that creates a diamond shaped portal you would do the following:

```
PortalNetwork.getInstance().getPortalManager()
    .registerPortalClass("diamond", DiamondPortal.class);
```

Then use the following command to get the portalblock ingame:

```
/pn give -type diamond
```

Last update: