# Planet Server­ – Version 1 Documentation

## Introduction

Planetserver is used to develop multi-user virtual environments by providing a middle layer between the client and server. It uses sockets so any messages sent from one client can be relayed to all other clients. Planetserver supports android/ios/windows/max/linux platforms at the moment. It uses a custom protocol on TCP and we will soon be supporting UDP as well.

## Directories

Planetserver has a two main directories you should know about

**Lib** – Contains all dependencies needed by the server

**Extension** – Contains your game extension jar, and any other custom dependencies needed by your extension

**Conf** – Contains a configuration file for the server, and also a configuration file for log4j(logging)

## Configuration

PlanetServer contains a **config.properties** file in the **conf** folder. This file can be edited to set up specific settings for your server.

**server.tcp.port** setting is used to configure the port your server is listening on

**server.threads.executor** setting is used to configure the number of threads in the threadpool. You will want to increase this amount based on the amount of activity for performance reasons.

There is also a **log4j.properties** file where you can configure the logging for your server. Refer to the log4j documentation for more details about this.

Inside of the extension folder there is a **world.properties** file. This file contains properties specific to your game and are accessible via your game code. The only needed property here is the **extension.class.name** which points to the main entry point of your application. The example provided has this configured so for your own extension just change the class name.

# Extensions

When creating your application you create an extension to planetserver. The extension is simply a jar file that is placed inside of the extension folder with all other necessary jar files for your application. You must create a main class for the extension for the entry point. The main class must extend from PSExtension and override the init and destroy methods.

**For example**

public class WorldExtension extends PSExtension {

@Override

public void init()

{

}

@Override

public void destroy()

{

super.destroy();

}

}

Place any custom functionality into the init and destroy methods. This can contain any custom code you need for your application.

# Extension Requests

In order to send custom commands to the server you must register a custom class for a request handler.

For example, lets say you have a player and want to send a request to do something for the player.

First add the request handler addRequestHandler(“player”, PlayerRequest.class);

Then Create the PlayerRequest Java file which extends from BasicClientRequestHandler. You override the handleClientRequest method and then you can parse the request that was sent.

All commands are sent using a dot notation. For example anything that starts with player. will be handled by the request handler that is registered for the string “player”

Some example commands might be player.save, player.load, player.move etc…

A good way is to have an enum of the commands and then you can write a switch statement to handle the commands separately per java file. You can view the example in the sample java file.

# Future Plans For Next Major Version

Version 1.1 will support UDP messaging. We believe this is necessary for some higher message rate games and it will be a free update for all users of planetserver