# Autonomous Armada v1.2

*Server Readme*

*August 6, 2012*

*David Esposito and Kevin Johnstone*

# Table of Contents

Introduction ..... 3
Software Overview ..... 3
Starting the Server ..... 3
Command Line ..... 3
Eclipse ..... 3
Allowing Clients To Connect ..... 5
Local Network ..... 5
IP Assigned to Your Computer ..... 5
IP Assigned to Home Router ..... 6
Using The Server GUI ..... 6
Building Custom Maps ..... 6
Naming Convention ..... 7
Header ..... 7
The Map ..... 7

## Introduction

Autonomous Armada is built on top of the Battle Challenge AI Framework. The server allows remote players to connect through Java Sockets. The central server will maintain and lead the gameplay.

## Software Overview

Autonomous Armada uses Java sockets to support communication between client and server. The server will run in a remote location with a socket connection to each client.

## Starting the Server

The server only takes one optional command line argument, the port number. This number defaults to port 3000.

### Command Line

The following example uses the default port.

```
java battlechallenge.server.BattleshipServer
```

The following example uses the port 4000.

```
java battlechallenge.server.BattleshipServer 4000
```

### Eclipse

Choose "Run->Run Configurations". Right click "Java Application" and select "New".

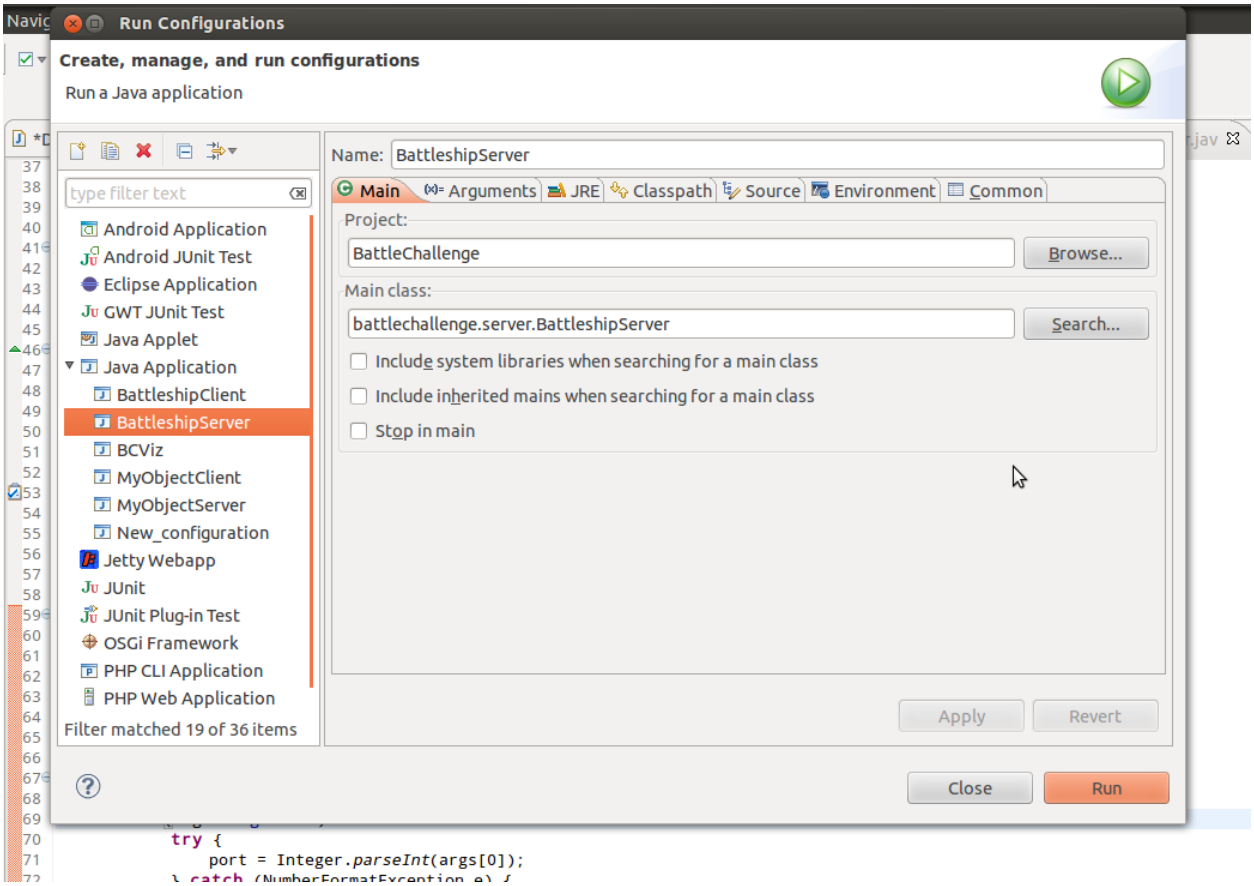
Main Tab:

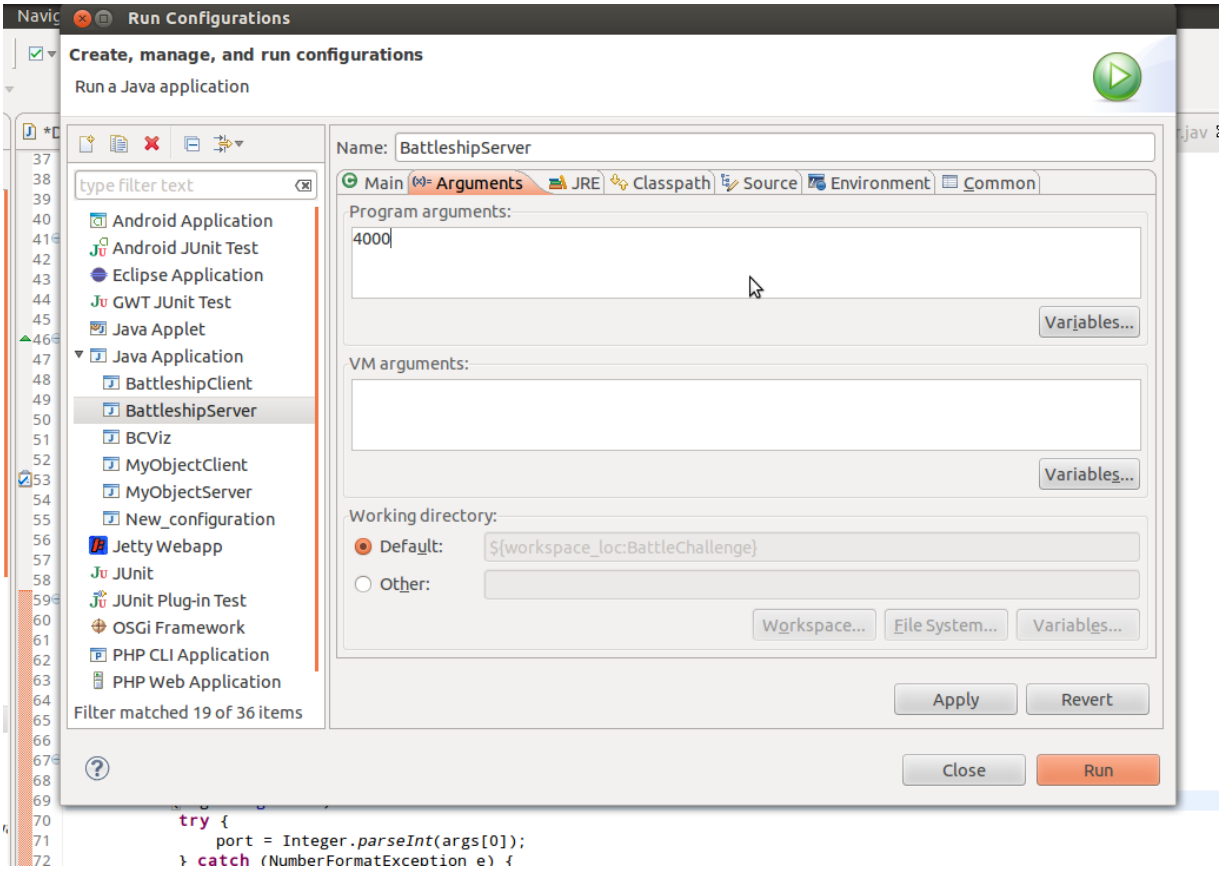
Main Class: "battlechallenge.client.BattleshipServer"

Argumants Tab:

Program Arguments: "4000"

NOTE: Do not use quotes when entering text fields.


Run Configurations
Create, manage, and run configurations
Run a Java application
type filter text
Android Application
Android JUnit Test
Eclipse Application
GWT JUnit Test
Java Applet
Java Application
BattleshipClient
BattleshipServer
BCViz
MyObjectClient
MyObjectServer
New_configuration
Jetty Webapp
JUnit
JUnit Plug-in Test
OSGi Framework
PHP CLI Application
PHP Web Application
Filter matched 19 of 36 items
Name: BattleshipServer
Main
Arguments
JRE
Classpath
Source
Environment
Common
Project:
BattleChallenge
Browse...
Main class:
battlechallenge.server.BattleshipServer
Search...
Include system libraries when searching for a main class
Include inherited mains when searching for a main class
Stop in main
Apply
Revert
Close
Run
try {
port = Integer.parseInt(args[0]);




# Allowing Clients To Connect

## Local Network

For games on your own machine. Just use the default settings for IP and port. If you changed the port on the server, you will have to match that with the client.

Sever:

```
java battlechallenge.server.BattleshipServer
```

Client:

```
java -cp ../libs/jewelcli-0.7.6.jar:. battlechallenge.client.BattleshipClient --n=David --b=DavidBot
```

```
java -cp ../libs/jewelcli-0.7.6.jar:. battlechallenge.client.BattleshipClient --n=David2 --b=DavidBot
```

## IP Assigned to Your Computer

Provide each client with your IP address.

Sever:

```
java battlechallenge.server.BattleshipServer
```

Client:

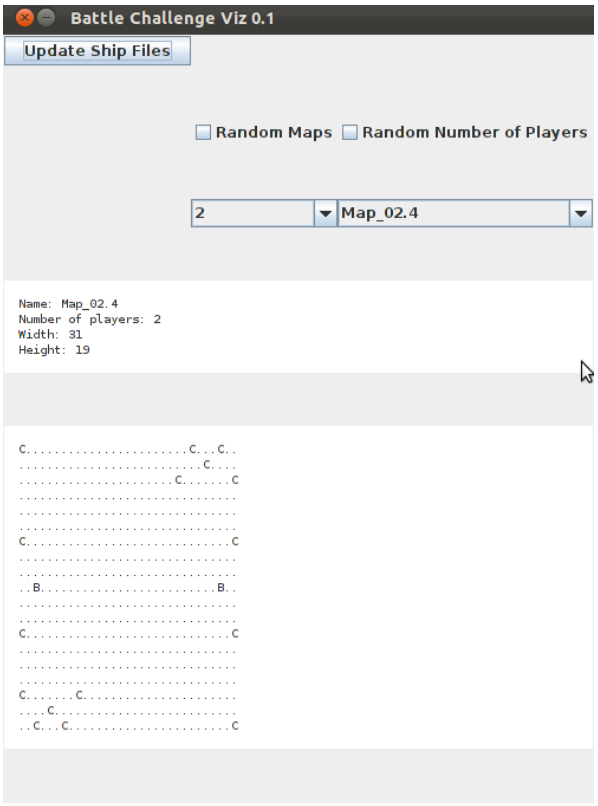
```
java -cp ../libs/jewelcli-0.7.6.jar:. battlechallenge.client.BattleshipClient --n=David --
b=DavidBot --i=143.215.110.249

java -cp ../libs/jewelcli-0.7.6.jar:. battlechallenge.client.BattleshipClient --n=Loser --
b=StarterBot --i=143.215.110.249
```

## IP Assigned to Home Router

Coming soon… this is a pain in the ass…

## Using The Server GUI

At time of wring… I don't care and am tire of writing… The GUI should be self-explanatory. Also, the random check boxes do not actually work yet. Coming Soon!




## Building Custom Maps

The map file is broken in to two sections: headers and map

## Naming Convention

New maps should be named “Map_(number of players).(index or version)” ex. Map02.0, Map04.3 and Map_08.1c.

## Header

Three comma separated values make up the header.

1. Width: Number of columns
2. Height: Number of rows
3. Number of Players:  Number of bases inside the map

## The Map

The map is made up of ASCII characters:

- “.” – Ocean or open space
- “B” – Base, where a players ships will be placed when enough minerals have been earned.
- “C” – Island (formally cities). Players need to place their ships on these locations to earn minerals.
- “W” – Walls, boundaries which are not destructible. This causes the bot to navigate around them.

Example:

```
...B...
...C...
B.CWC.B
...C...
...B...
```