

# Entry Level Plugin API Guide

This guide explains the very basics of our plugin API, how to develop your first plugin and how to build and deploy it.

- [Prerequisites](#)
- [Maven Setup](#)

# Prerequisites

## What you will need:

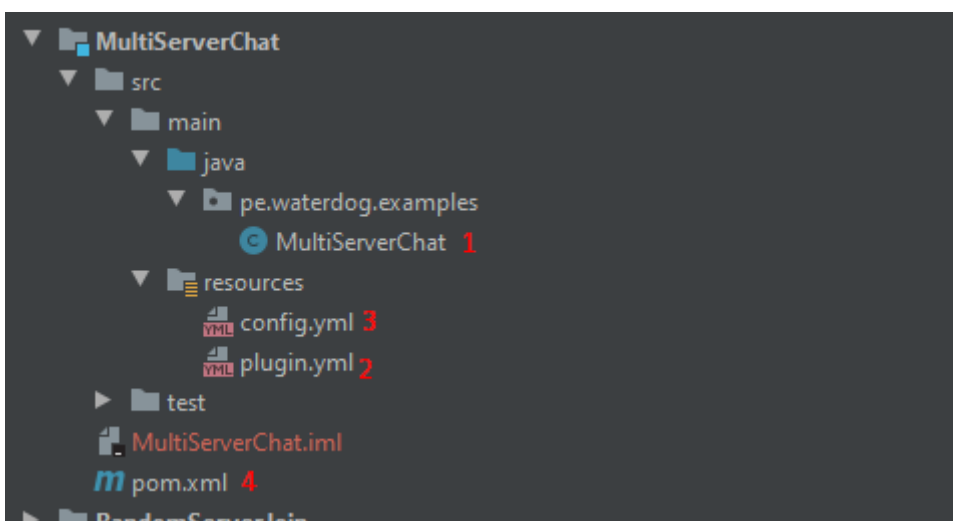
- An suitable IDE supporting maven. We recommend using either IntelliJ or Eclipse.
- An internet connection for downloading the Waterdog API Artifacts

## Structure

The base structure of a plugin is really simple.

Required parts:

- a plugin.yml file located in the programs resources folder.
- a base class extending the `Plugin` class and overwriting it's `onEnable()` method



- 1** - The Base class of the Plugin, mostly named after the plugin itself or from what the plugin is supposed to do.
- 2** - The plugin.yml file containing very basic information on the plugin.
- 3** - A configuration file that the user can use to easily change plugin settings.
- 4** - The base maven file. required to build the plugin and compile it to a runnable file.

# Maven Setup

## Base Setup

Maven is our choice for building and compiling plugins to actually runnable .jar files.

An example Maven compile file would look something like this:

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns="http://maven.apache.org/POM/4.0.0"
    xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>

    <groupId>pe.waterdog.examples</groupId>
    <artifactId>MultiServerChat</artifactId>
    <version>1.0-SNAPSHOT</version>

    <repositories>
        <repository>
            <id>waterdog-repo</id>
            <url>https://repo.waterdog.dev/artifactory/main</url>
        </repository>
    </repositories>

    <dependencies>
        <dependency>
            <groupId>dev.waterdog.waterdogpe</groupId>
            <artifactId>waterdog</artifactId>
            <version>1.0.0-SNAPSHOT</version>
            <scope>provided</scope>
        </dependency>
    </dependencies>

    <build>
```

```
<plugins>
  <plugin>
    <groupId>org. apache. maven. plugins</groupId>
    <artifactId>maven- compiler- plugin</artifactId>
    <version>3. 8. 0</version>
    <configuration>
      <release>11</release>
      <encoding>UTF- 8</encoding>
    </configuration>
  </plugin>
</plugins>
</build>
</project>
```

## Explanation

There are three relevant parts in the pom.xml for the beginning:

- **project configuration:** the `groupId` , `artifactId` and `version` are base configuration variables for your own plugin.
- **dependencies:** The java frameworks / applications your plugin depends on. By default, only the Waterdog dependency is required there.
- **build configuration:** the configuration targetting the build / compile process. Here we set the java version the plugin should compile to and the encoding all files should have.

**If you use more dependencies, shading will be required.**

## Shading

When using more than just the default waterdog dependency and included libraries, you will be required to *shade* your artifact. What that basically means is that we are including all the dependencies your plugin uses in one artifact (jar). This requires modifying our pom.xml.

```
<plugin>
  <groupId>org. apache. maven. plugins</groupId>
  <artifactId>maven- shade- plugin</artifactId>
```

```
<version>3.1.0</version>
  <executions>
    <execution>
      <phase>package</phase>
      <goals>
        <goal>shade</goal>
      </goals>
    <configuration>
      <createDependencyReducedPom>false</createDependencyReducedPom>
    </configuration>
  </execution>
</executions>
</plugin>
```

**Don't be surprised if your artifact file increases in size.** As we are now including all the libraries in the artifact, the file size is of course a bit larger.