Object Bench Toolkit User Guide

# Introduction

The Object Bench Toolkit is a plugin designed to provide a set of features useful to new programmers. This plugin offers an Object Bench, Class Diagram view and Inspector functionality to assist new programmers in writing and testing object oriented systems. This plugin gives developers the opportunity to try their code without the need to implement a main method and figure out details of building and deployment.

# Prerequisites

Before installing the plugin it is important that the following prerequisites are met.

## Java

Java JDK 1.7+ should be installed.

## Eclipse

Eclipse Luna 4.4+ should be installed; the plugin may work on earlier versions but is not officially supported.

# Installation

The plugin can be installed in two ways, from the update site located online, or through a folder extracted from a provided zip file.

## Update Site

To install from the update site:

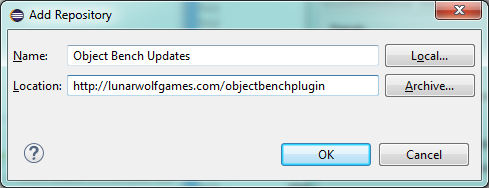
1. Launch Eclipse
2. Select **Help>Install New Software**
3. Press the **Add** button by the “**Work with**” field.
4. In the dialog enter:  
   **Name**: *Object Bench Updates***Location:** *http://lunarwolfgames.com/objectbenchplugin.*

Figure 1 - Setting the Update Site

1. Click “**OK”**
2. Mark the “Object Bench” component with a tick.
3. Click “**Next**”
4. Follow the on-screen prompts to install the software.
5. Restart Eclipse

## Local Repository

If installing from a zip archive:

1. Extract the archive to a desired location.
2. Launch Eclipse
3. Select **Help>Install New Software**
4. Press the “**Add**” button by the “**Work with**” field.
5. In the dialog press the “**Local**” button.
6. Navigate to the folder the zip was extracted to.
7. Click “**OK**”
8. Mark the “Object Bench” component with a tick.
9. Click “**Next**”
10. Follow the on-screen prompts to install the software.
11. Restart Eclipse

# Getting Started

## Perspective

To switch to the Object Bench Toolkit Perspective:

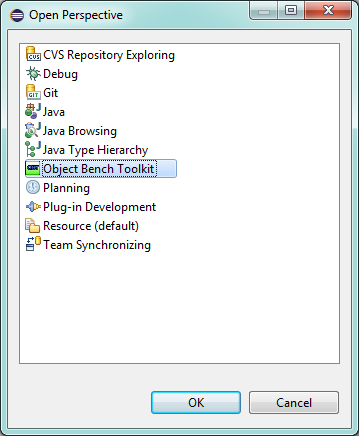
1. Select **Window>Open Perspective>Other**
2. Locate “**Object Bench Toolkit**”  
   

Figure 2 - Selecting the Perspective

1. Press “**OK**”
2. The pers pective will open as shown below.

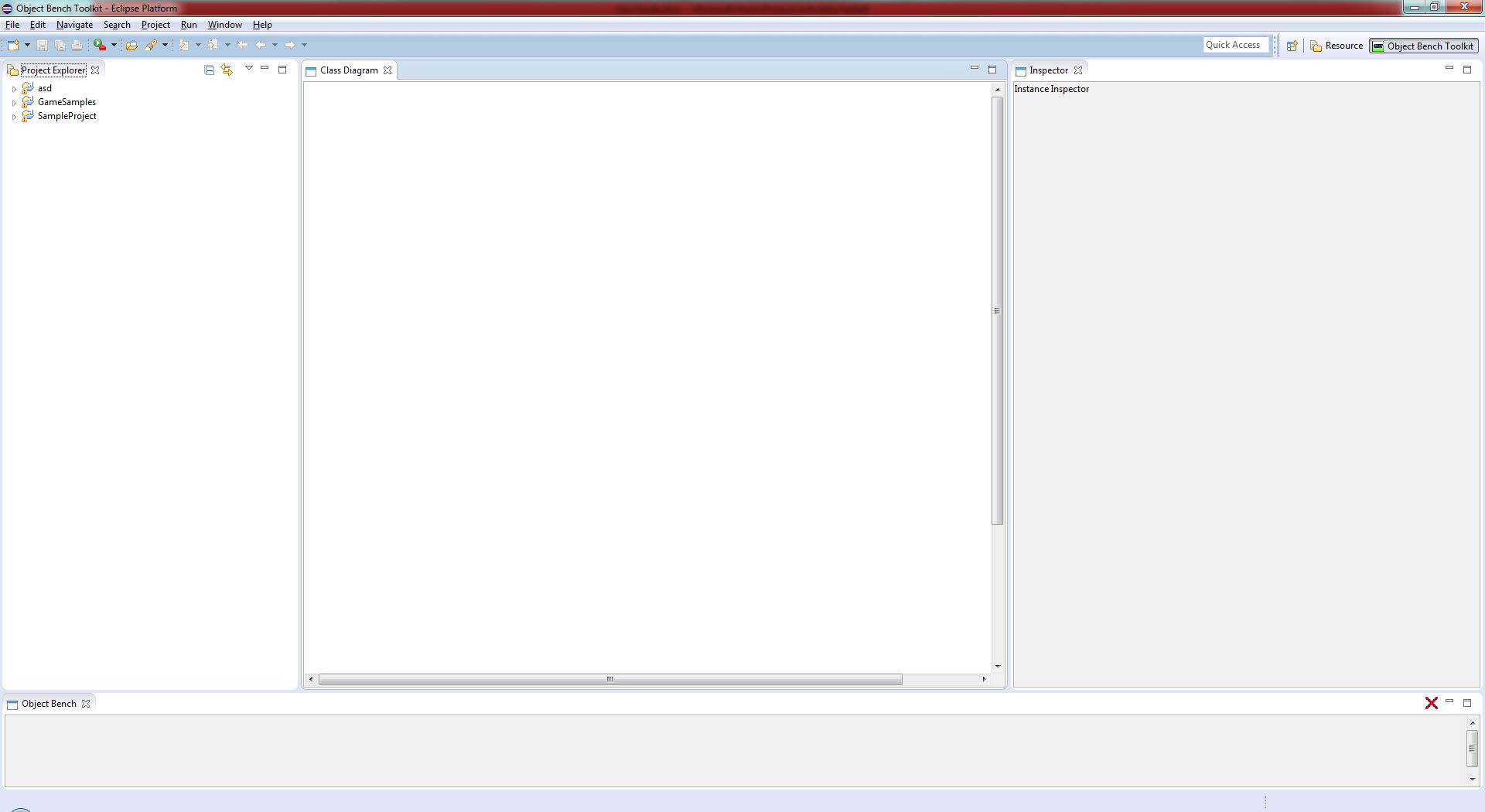


Figure 3 - The Object Toolkit Perspective

## Preferences

To edit preferences for the plugin:

1. Select **Window>Prefences**
2. Select “**Object Bench Tool Preferences**” from the menu.
3. Set preferences as desired. It is recommended to enable empty packages initially.

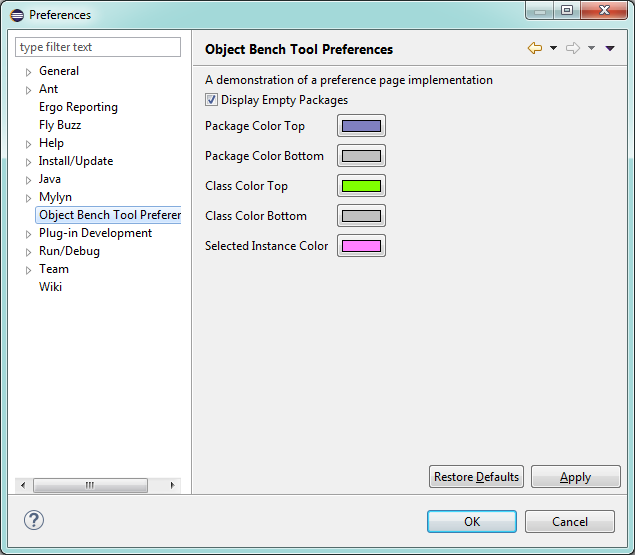


Figure 4 - Setting preferences.

## Components

### Class Diagram

The class diagram renders a UML styled overview of the classes in your project. From this view new classes and packages can be viewed, created and removed. Inheritance is represented by solid lines, and dependence is represented by a dashed line.

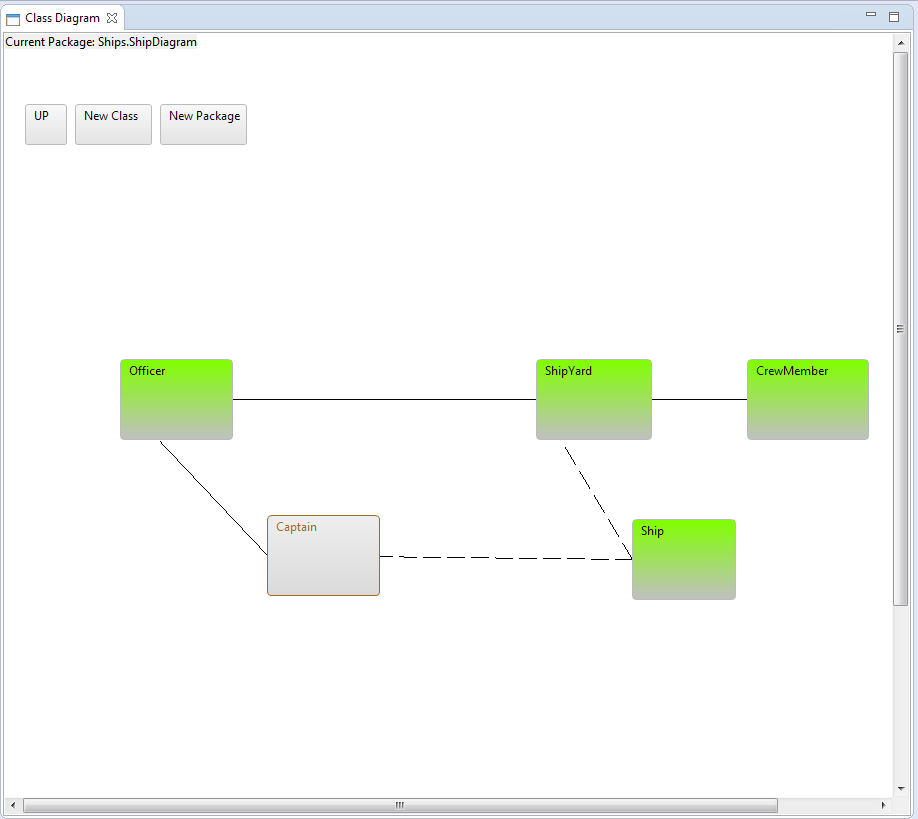


Figure 5 - The class diagram.

### Object Bench

The object bench is where all created instances are placed ready to be interacted with.

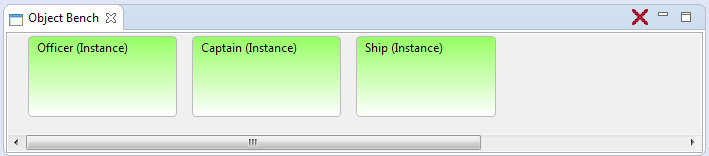


Figure 6 - The Object Bench

### Inspector

The inspector view takes a selected instance and allows a user to view it’s fields, the inspector is capable of looking at both public and private fields and set the values within as required.

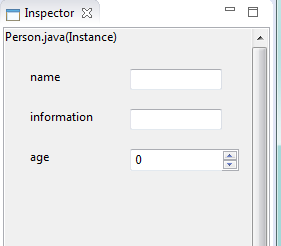


Figure 7 - The Inspector

## Selecting an Active Project

To select a project to work with:

1. Ensure that the “**Project Explorer**” is open.
2. From the “**Project Explorer**” simply select a project, any file/folder within a project selected will mark the parent of these as the active project.
3. The active project is set.

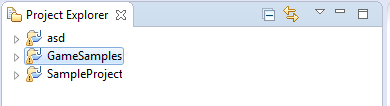


Figure 8 - Active project here is "GameSamples"

# Navigating The Class Diagram

## Opening Packages

We can open a package by simply double clicking the package icon. To return to the previous screen the “**UP**” button can be pressed.

## Moving Classes

Classes within the class diagram can be moved at will, to move a class simply hold down a mouse button and drag to a desired location.

## Opening A Class In Code Editor

To open a class in the code editor you can either:

* Double click the class icon.

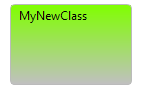


Figure 9 - A class icon

* Right click the class icon and select “**Open in Editor**”.

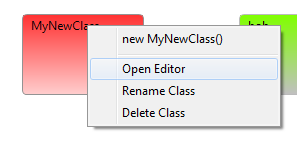


Figure 10 - Open the editor

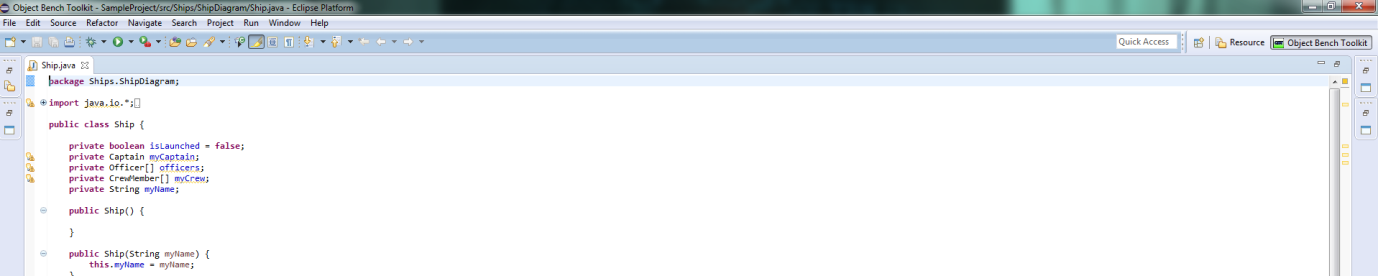
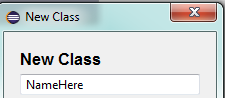
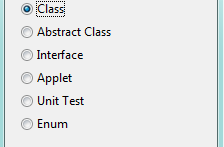
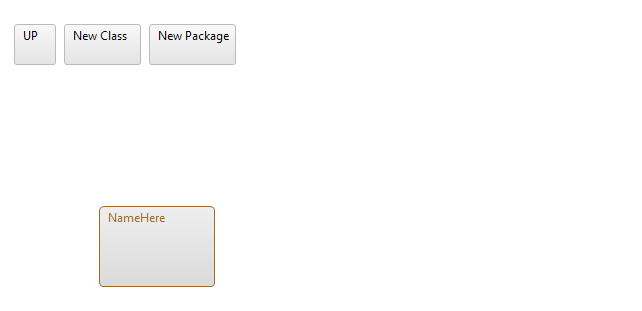


Figure 11 - The default code editor is opened.

# Adding/Removing Classes

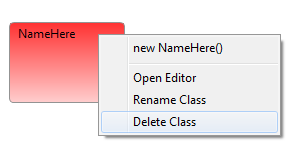
## Adding a New Class

To add a new Class:

* Within the class diagram press the “**New Class**” button.   
  http://i.gyazo.com/2c5e2687a9074717e27daa16ae02abd6.png
* Enter a name within the name field in the dialog that opens.  
  
* Select a template (**For more information on templates see the Templates Section**)  
  
* Click the OK button  
  http://i.gyazo.com/daac4a0814058bdf0d4baa3bef8a8ee9.png
* Your new class will be created within the active package.   
  

## Removing a Class

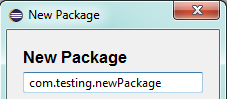
To remove a class simply:

1. Right click the class icon from the class diagram.
2. Select “**Delete Class**”.  
   
3. The class will be deleted

# Adding/Removing PAckages

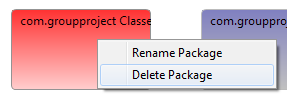
## Adding A Package

To create a new package:

1. From the class diagram select “**New Package**”  
   http://i.gyazo.com/2bfa944dfde80951204509e4122e2e03.png
2. Enter a name for the package.  
   
3. Press “**OK**”.  
   http://i.gyazo.com/14875cb776e368935cddea80072c1308.png
4. The new package will be created.

## Removing A Package

To remove a package:

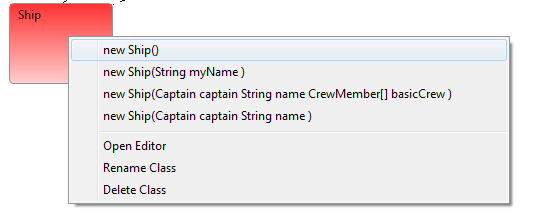
1. Right click the package icon from the class diagram.
2. Select “**Delete Package**”.  
   
3. The package will be deleted.

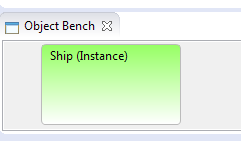
# Object Bench Instances

Object Bench instances are created from within the class diagram view. The system is capable of calling any constructor that is defined by a user.

## Zero-Parameter Constructor

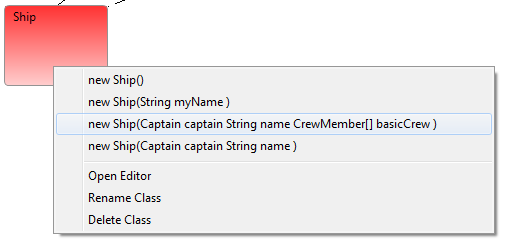
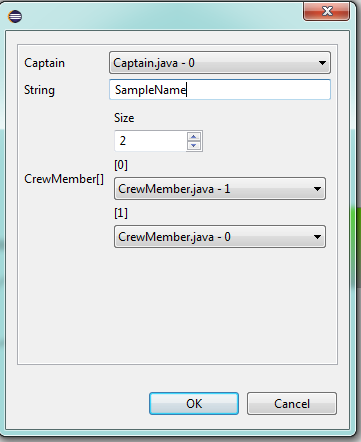
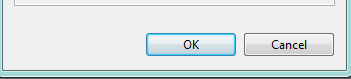
To create an instance of a class with a zero-parameter constructor:

1. Right click on the desired class icon from the class diagram.
2. Select the Zero-Parameter constructor from the menu.  
   
3. The instance will be created and placed on the object bench.



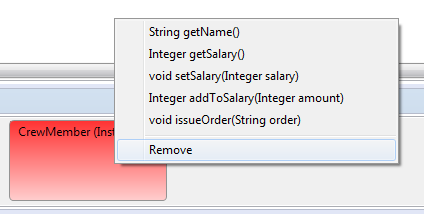
## Multi-Parameter Constructor

To create an instance of a class with a constructor with multiple parameters:

1. Right click on the desired class icon from the class diagram.
2. Select the desired constructor from the menu.  
   
3. Set the fields for the parameters to values desired.  
   
4. Click “**OK**”  
   
5. The instance will be added to the Object Bench.

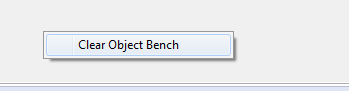
## Removing an Instance

To remove an instance from the object bench:

1. Right click on an instance icon in the object bench.
2. Select **“Remove**”   
   
3. The instance is removed from the object bench.

## Clearing The Object Bench

To clear the object bench either:

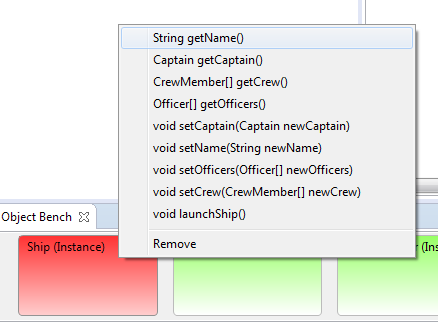
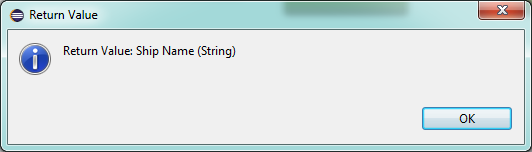
* Click the “**Clear Bench**” icon found in the top right corner of the view.  
  http://i.gyazo.com/31af4c4708b45b002b6e8424143fd332.png
* Right click anywhere in the object bench as select “**Clear Bench**”.  
  

# Calling Methods

Methods can be called from the object bench as required; methods are read from those defined within the classes of the instances on the object bench. When a method is called within the system a dialog will be presented if the call was successful. The dialog will also display any return value from the method.

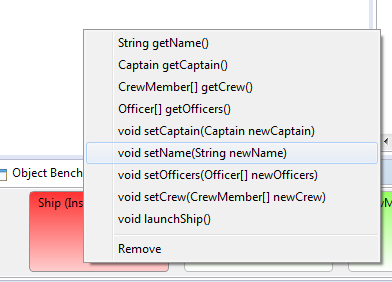
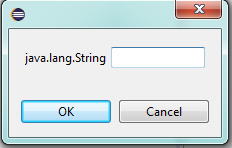
## Zero-Parameter Methods

To call a method with zero parameters:

1. Right click the desired instance from the object bench.
2. Select the desired method with zero parameters.   
   
3. The method will be called.   
   

## Multi-Parameter Methods

To call a method with parameters:

1. Right click the desired instance from the object bench.
2. Select the desired method.  
   
3. Enter values into the generated fields in the dialog.   
   
4. Press “**OK**”, the method will be called.

## Return Dialogs

When a method is called a return dialog will be displayed. This will show the value that is returned by a method defined in user code. This dialog will show even if the return type is void to provide assurance that the method has been successfully called.

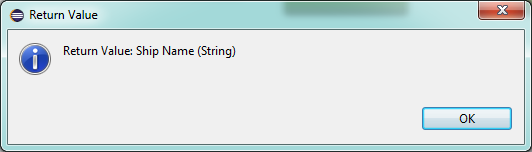


Figure 12 - A return dialog for a method.

# Inspector

## Selecting an Instance

To select an instance for viewing in the inspector simply click the desired instance in the object bench, the instance will be highlighted and displayed in the inspector.

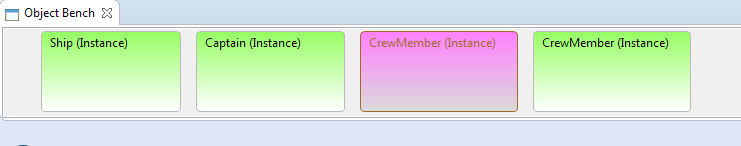


Figure 13 - A selected instance.

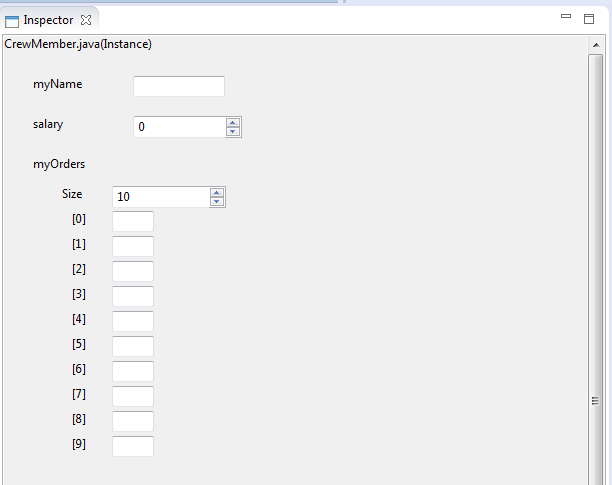


Figure 14 - The inspector for a selected instance.

## Editing Fields

To edit fields (variables) in the inspector:

1. Select an instance on the object bench.
2. Edit the various fields that are generated to values that you wish to use.

# Templates

When creating a new class there are several templates that can be selected.

## Standard Class

This creates a standard class with a single field, constructor and method. This is the best class to use to start with.

## Abstract Class

This creates an abstract class for classes to base implementation from.

## Interface

This template creates a basic interface which helps to understand the creation of interfaces.

## Applet

This template provides a bare-bones applet structure for creating simple Java applets.

## Unit Test (Currently Unsupported)

Provides a basic unit test set up, this doesn’t work at the time of writing but can be used to gain understanding in the structure and creation of unit tests.

## Enum

This template provides a basic enumerable styled class, useful to learn how to create custom enumerable classes.

# Supported Types

Here we list the known and supported types by this plugin.

## Numeric

* Integer
* Double
* Float
* Long

### Generated Control

The generated control for numeric types is a spinner.

http://i.gyazo.com/a3b8ab8b76e4bed5634f541882b24ed7.png

Figure 15 - A spinner for numeric types

## Text

* String
* Char

### Generated Control

The generated control for text types is a text box:

http://i.gyazo.com/d0a5a8f03012607cfa98949575698458.png

Figure 16 - A text field for text types.

## Other

* Boolean

### Generated Control (Boolean)

The generated control for a Boolean value is a checkbox.

http://i.gyazo.com/17ddc7ee02177414b986d39d723a3ee2.png

Figure 17 - A checkbox for Boolean values

## Arrays

* All Numeric
* All Text
* All Custom Classes
* Boolean

### Generated Control

Arrays will generate a custom control with fields for **size** and an individual control for the individual elements within the array dependent on the base type of the array.

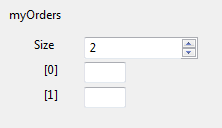


Figure 18 - An example array control for a string array.

## Custom Classes

Any class defined by a user can be used and will be supported by this system.

### Generated Control

When a user makes use of one of their own classes a drop down control is created, this will look on the **Object Bench** for any instances of the user type.

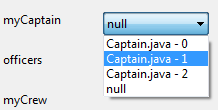


Figure 19 - A dropdown menu for a defined user class.

# Contact

The creators of this software are students at the University of Huddersfield.

## Blake Kendrick

**Email:** [u1055491@unimail.hud.ac.uk](mailto:u1055491@unimail.hud.ac.uk)

## Ashad Syed

**Email:** [u1055376@unimail.hud.ac.uk](mailto:u1055376@unimail.hud.ac.uk)