LJC Java Workshop 25 Oct 2016 - preparation

To make sure that you are ready for the workshop, please run through the following steps beforehand.

These instructions are based on the Eclipse development environment. If you are proficient in another IDE such as IntelliJ or Netbeans then please feel free to use that instead.

1. Install software

- Install the following software:
  - Java Development Kit from Java download page
    (Choose your platform then tick "Accept License Agreement")
  - Eclipse (or your favourite IDE)
    Select the Java configuration, Java EE won't be needed at present.
  - Git (or gitbash on Windows) from Git download page
    Leave all installation options set to default.
  - Environment
    Add your JDK's bin folder to the beginning of your PATH, as per the JDK install location from the above step. For more details Google for: `java path setup`
- Copy the starter files
  - Open a Terminal or Command Prompt window and `cd` into the folder where you want to keep your work, such as C:\ or your usual home folder. In Windows Gitbash use `cd /c/` to change to the C: drive if necessary.
  - Copy the starter files from github by typing: `git clone https://github.com/LondonJavaCommunity/JavaWorkshop.git`
- Check your Java JDK and Eclipse
  - At the command prompt type `java -version` then `javac -version` These should both print a suitable message. If not, check the above Java installation and PATH. You can check the PATH variable by typing `echo %PATH%` on Windows, or `echo $PATH` on Mac & Linux. If you make any further changes to the PATH variable, you will need to open a new command window for them to take effect. Also make sure that you downloaded a JDK and not JRE.
  - Start Eclipse. If Eclipse asks for a workspace choose the `JavaWorkshop` folder that you cloned from Guthub earlier (eg `~/dev/JavaWorkshop`). If Eclipse doesn't ask for a workspace, then do `File> Switch Workspace` .

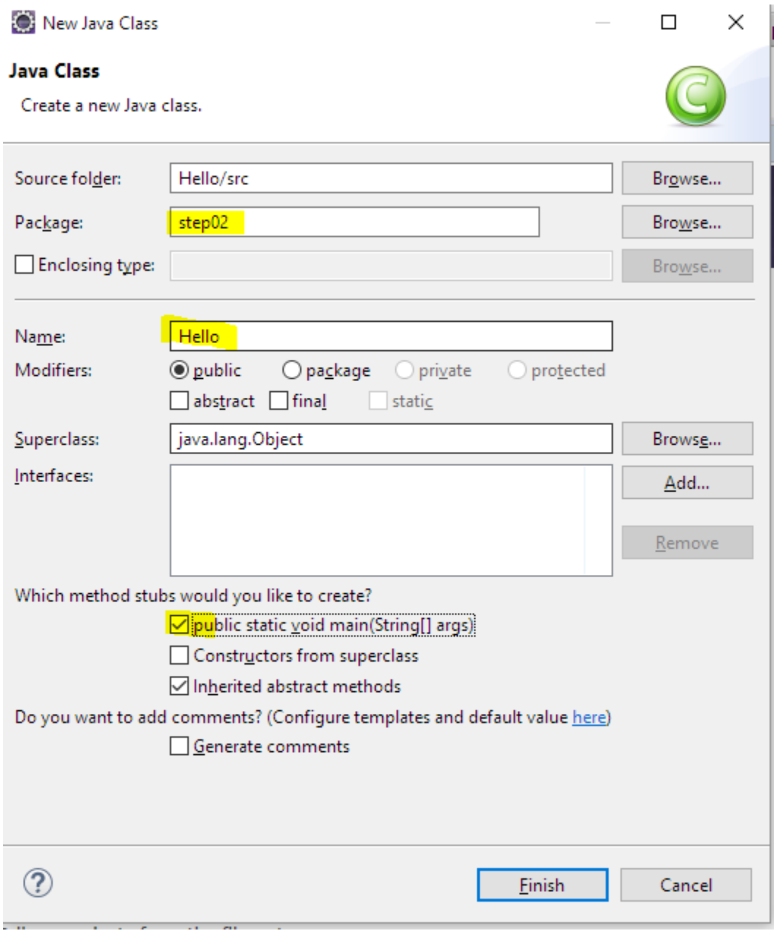
2. Review simple Java app

- Open a Terminal or Command Prompt window and `cd` into the `JavaWorkshop` folder (which was created by the `git clone` command above). Then `cd` into the sub-folder called `step01`
- Notice the file `Simple.java` . Display its contents using the `cat` command (or `type` on Windows)
- Compile it by typing `javac Simple.java`
- Run it by typing `java -cp . Simple`
  The `-cp .` option adds your current directory to Java's `classpath` where it searches for .class files. This option may or may not be needed depending on how you have set up your Java environment.

3. Create a simple Eclipse Java project

- In Eclipse do `File> New> Other> Java Project` . You may need to expand the categories, or type `java` into the `filter text` box. After you have used `New Java Project` for the first time it will appear directly in the `File> New` menu, so you won't need to take the longer route via the `Other` dialog.

- In the `Project name` box, type `Hello` and press `Finish`
- Create a new class using `File> New> Other> Class`. Again you may need to expand the categories or filter text. In the `Package` box type `step02`, and in the `Name` box type `Hello`. Tick the box to create the method `public static void main`, and click `Finish`. The dialog for this should look something like the following:



New Java Class

Java Class

Create a new Java class.

Source folder: Hello/src

Package: step02

□ Enclosing type:

Name: Hello

Modifiers: ● public ○ package ○ private ○ protected

□ abstract □ final □ static

Superclass: java.lang.Object

Interfaces:

Which method stubs would you like to create?

☑ public static void main(String[] args)

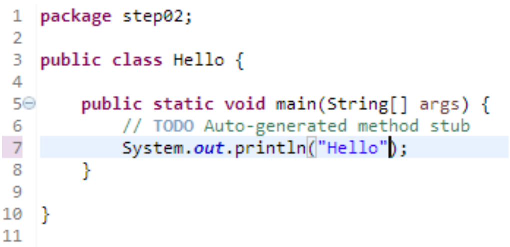
□ Constructors from superclass

☑ Inherited abstract methods

Do you want to add comments? (Configure templates and default value here)

□ Generate comments

- If at this stage you see the Eclipse Welcome page, close it.
  Open your Hello.java file by expanding the `Hello` item within the left-hand `Package Explorer` window.
  If you forgot to tick the box above to create the `main` method, click inside the Hello class (ie just below `public class Hello`), type `main` and press `CTRL+SPACE`. This code-completion template can create a `main` method for you.
- Click inside the `main` method, type `syso` and press `CTRL+SPACE` as a shortcut template to print something. Inside the brackets type `"Hello"` (including the quotes.)
  Line numbers are handy, so in Eclipse Preferences type `numbers` in the search box and tick `Show line numbers`
  Your Hello.java file should now look something like the following:



```
package step02;

public class Hello {

    public static void main(String[] args) {
        // TODO Auto-generated method stub
        System.out.println("Hello");
    }

}

```

- In the left-hand `Package Explorer` view, right-click at the top level of your `Hello` project and choose `Run As > Java Application`. You should see some output in the lower `Console` view.

Once you have done the above you're good to go, and we look forward to welcoming you at the workshop.