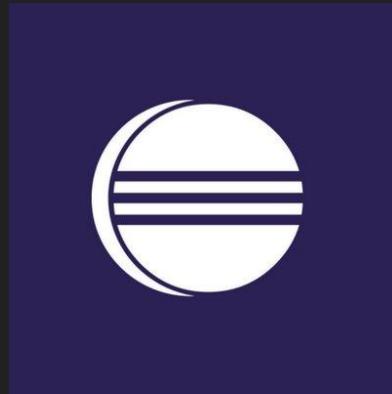


# IDE Eclipse



Sergio González Guerra  
Germán Alfonso Teixidó

# Index

- Download and configure Eclipse
- Create Java elements
- IDE programming tools
- Eclipse views
- Ejecute and debug
- FAQ
- Bibliografía

# 1

DOWNLOAD AND CONFIGURE ECLIPSE

# Where can we download Eclipse?



[www.eclipse.org](http://www.eclipse.org)

# How do we install Eclipse?

The image shows two screenshots of the Eclipse Installer application.

**Screenshot 1: Installation Selection Screen**

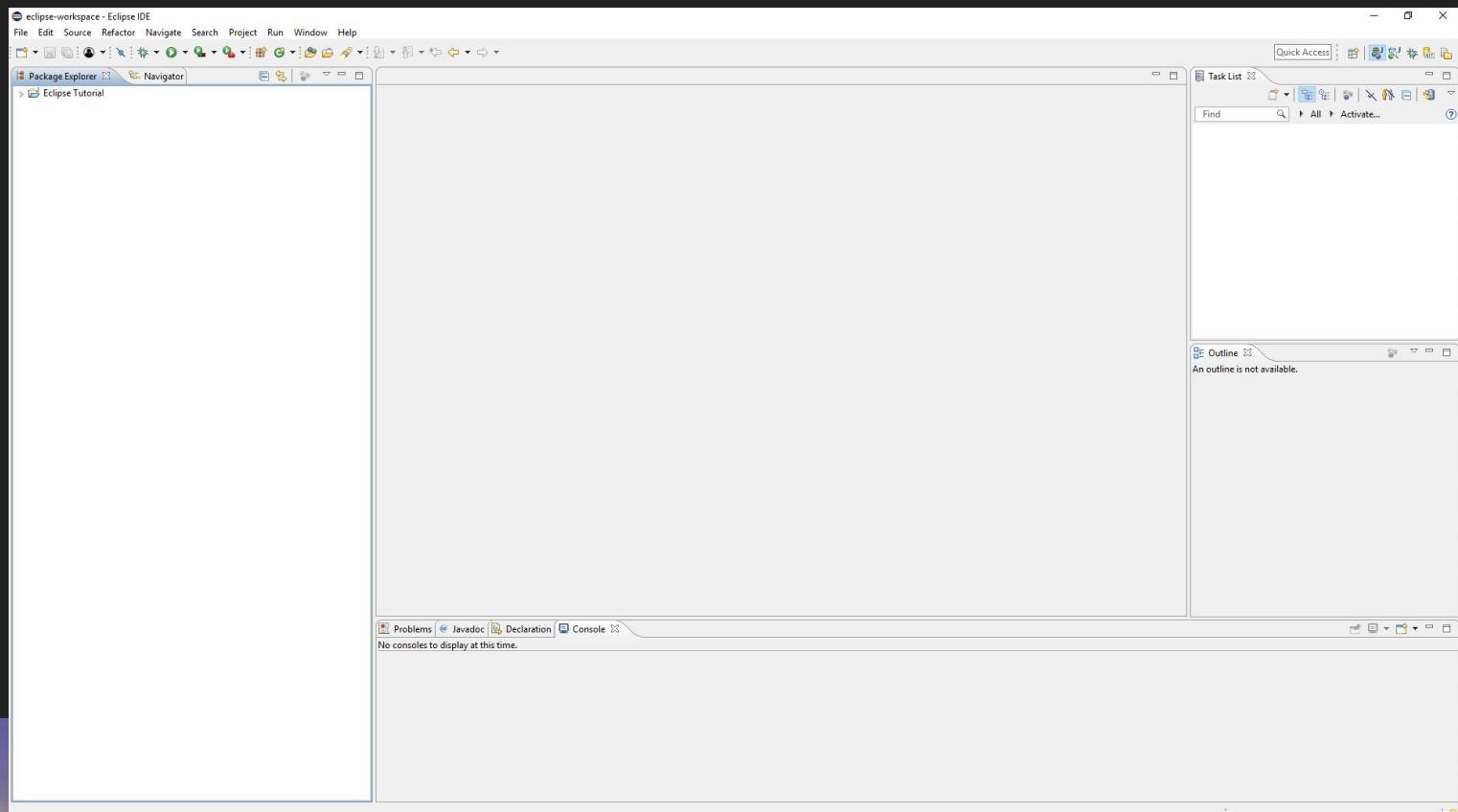
This screen displays five options:

- Eclipse IDE for Java Developers**: The selected option, indicated by a blue border. Description: "The essential tools for any Java developer, including a Java IDE, a Git client, XML Editor, Mylyn, Maven and Gradle integration."
- Eclipse IDE for Enterprise Java Developers**: Description: "Tools for Java developers creating Enterprise Java and Web applications, including a Java IDE, tools for Enterprise Java, JPA, JSF, Mylyn, Maven, Git and..."
- Eclipse IDE for C/C++ Developers**: Description: "An IDE for C/C++ developers with Mylyn integration."
- Eclipse IDE for JavaScript and Web Developers**: Description: "The essential tools for any JavaScript developer, including JavaScript, HTML, CSS, XML languages support, Git client, and Mylyn."
- Eclipse IDE for PHP Developers**: Description: "The essential tools for any PHP developer, including PHP language support, Git client, Mylyn and editors for JavaScript, HTML, CSS and XML."

**Screenshot 2: Configuration Screen for Eclipse IDE for Java Developers**

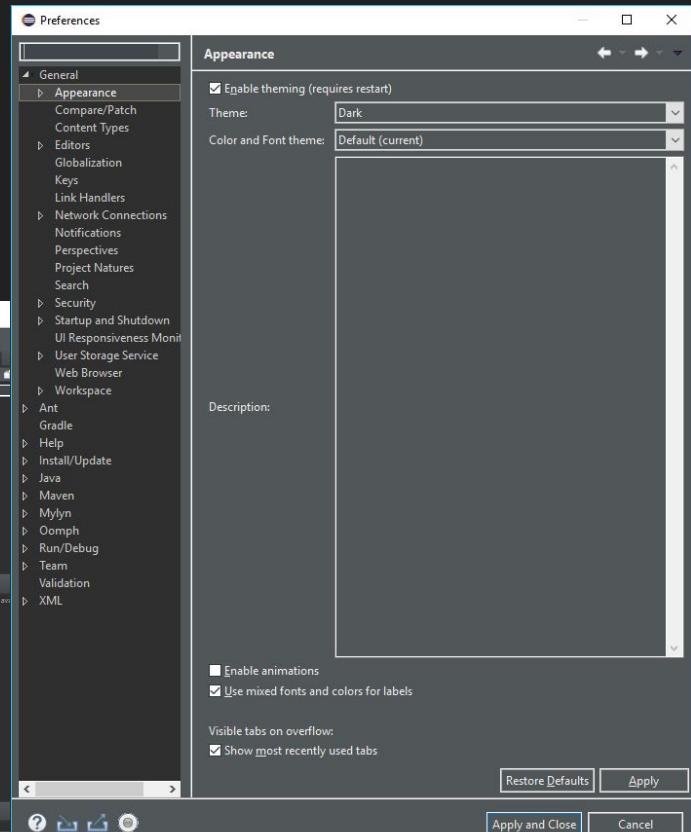
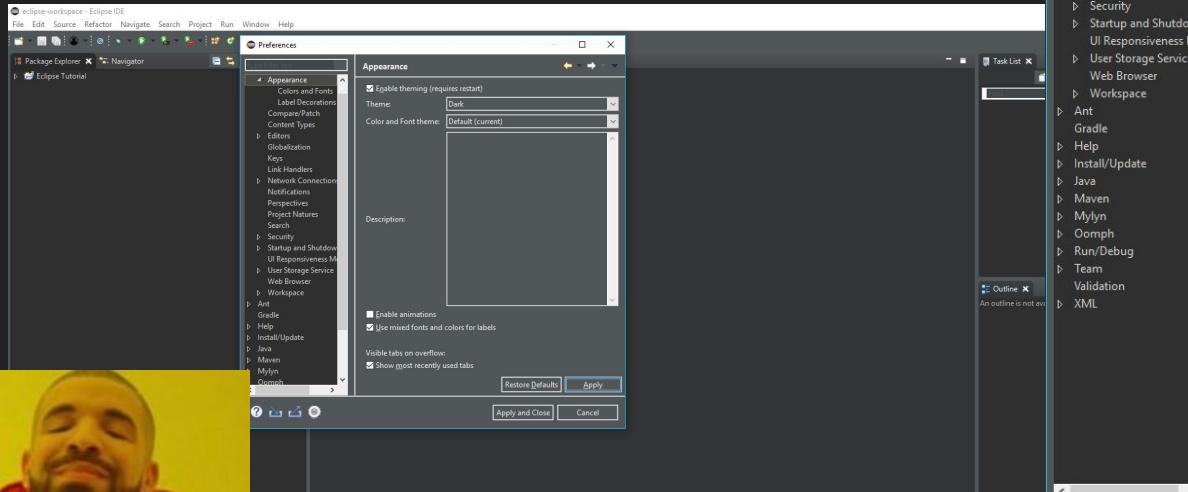
This screen shows the configuration for the selected "Eclipse IDE for Java Developers".

- Installation Folder**: Set to "F:\Users\Win10\eclipse\java-2018-12".
- Customization Options**:
  - create start menu entry**
  - create desktop shortcut**
- INSTALL** button at the bottom.



# Cambiando a modo oscuro

Window>Preferences  
Appearance->Themes [Dark]



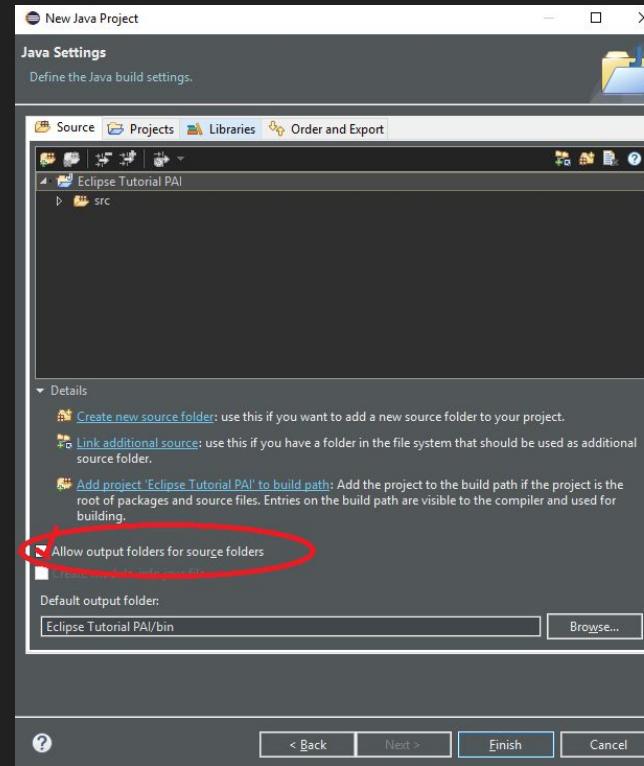
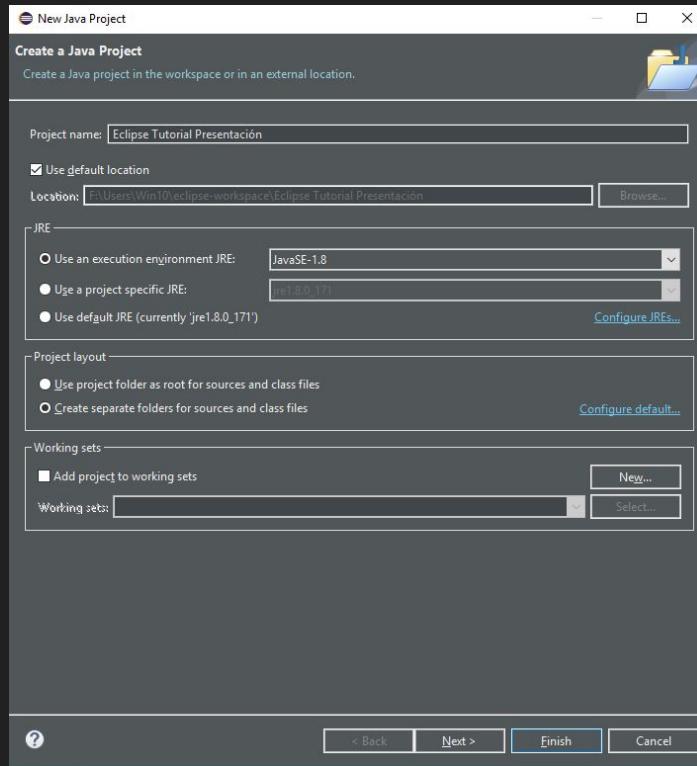
¡Recuerda aplicar  
los cambios!

# What kind of project can we do with Eclipse?

- Java Project
- Simple Project
- Plug-In Development Project

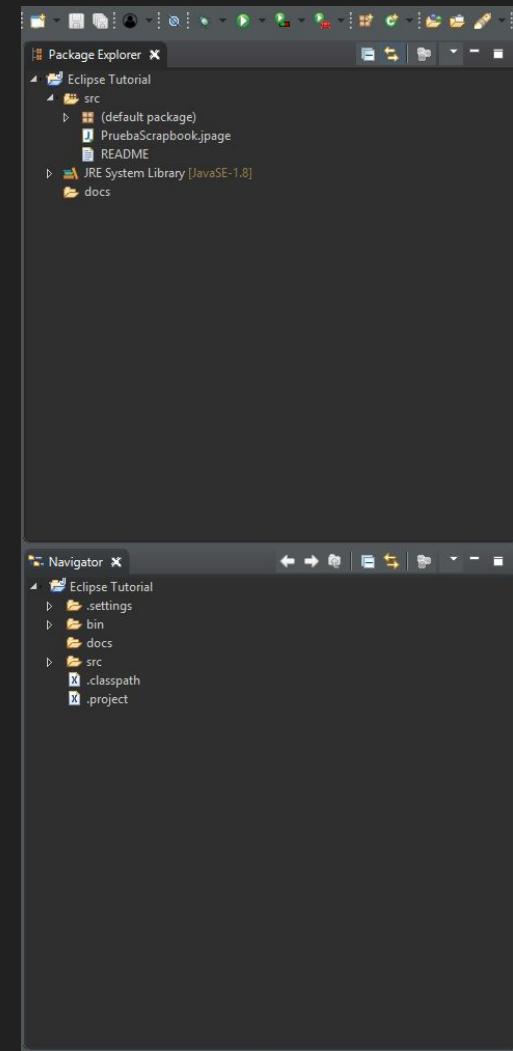
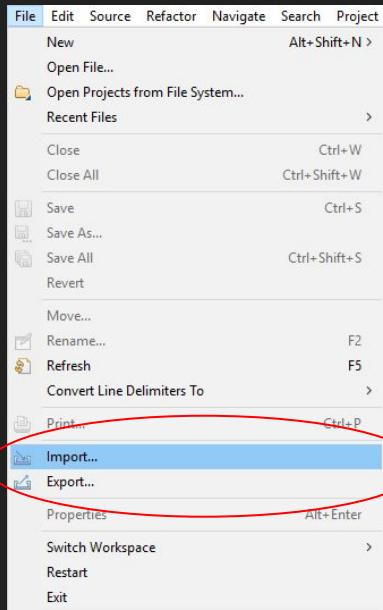
# How to create a Java Project

File > New > Java Project



# Import (and export) files with Eclipse?

- File > Import / Export
- Dragging files to the “Package Explorer”/“System Navigator”



# 2

CREATE JAVA ELEMENTS

# Add Packages

- Store and organize Java files
- Stored in /src
- Contains several parts separated by points
- Each point represent a directory
- Right click -> New -> Package



## Java Package



⚠ Discouraged package name. By convention, package names usually start with a lowercase letter

Creates folders corresponding to packages.

Source folder:

[Browse...](#)

Name:

Create package-info.java



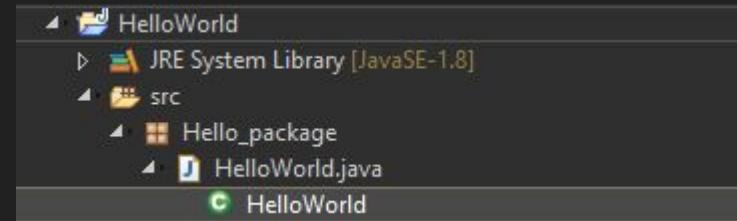
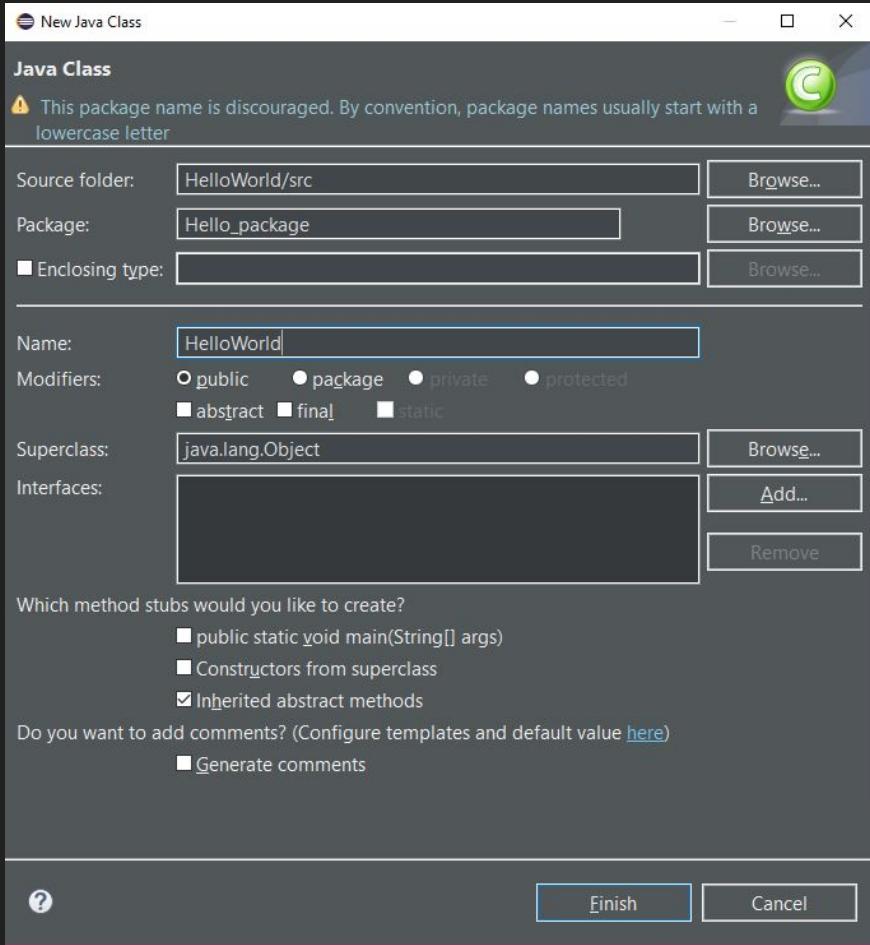
[Finish](#)

[Cancel](#)

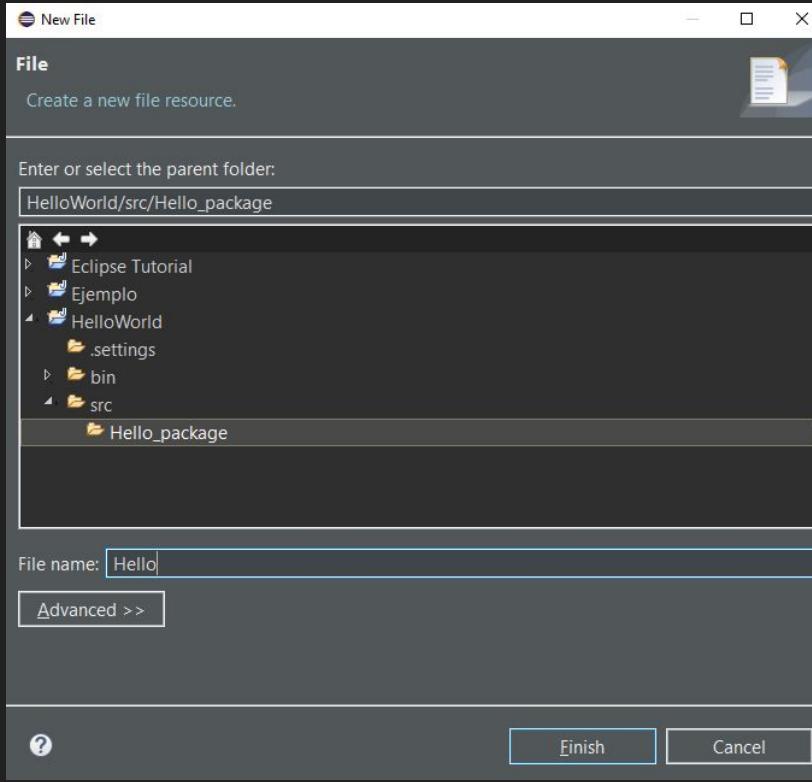
# Java Class

- ‘.java’ files -> Compiled on  
‘.class’ files.
- Stored in /src .
- Right click -> New -> Class, to  
create a new class.

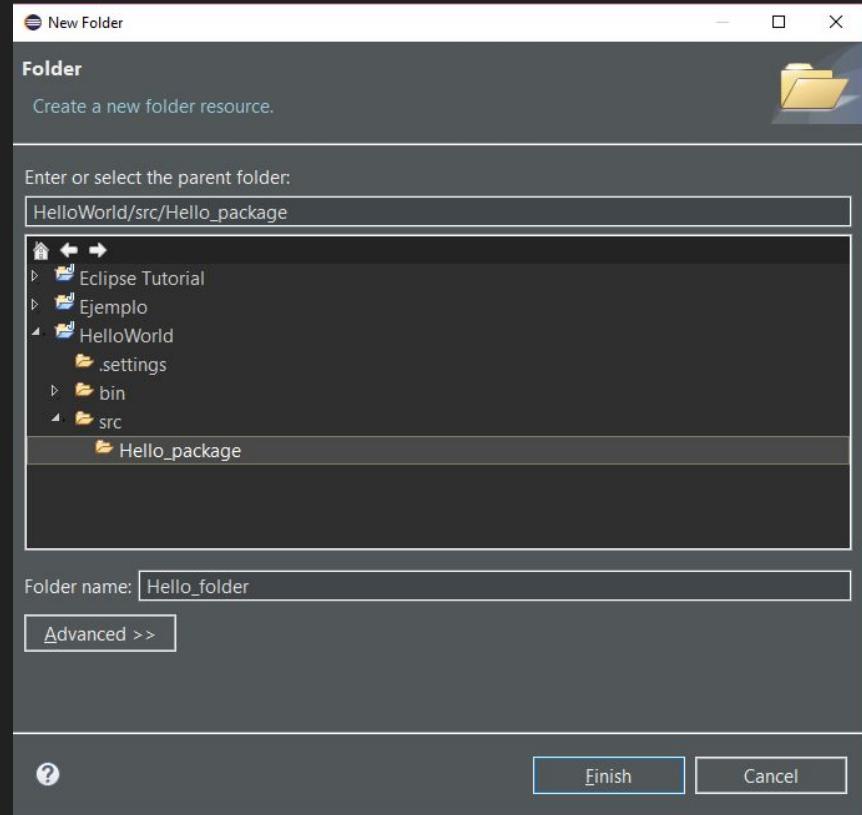




# Files



# Folder



# Interfaces

- Lack of implementation, other classes implement them.
- Tells to the implemented class, what to do.
- Can extend to another interface by inheritance.





## Java Interface

⚠ This package name is discouraged. By convention, package names usually start with a lowercase letter

Source folder:

HelloWorld/src

Browse...

Package:

Hello\_package

Browse...

 Enclosing type:

Browse...

Name:

Hello\_interface

Modifiers:

 public package private protected

Extended interfaces:

Add...

Remove

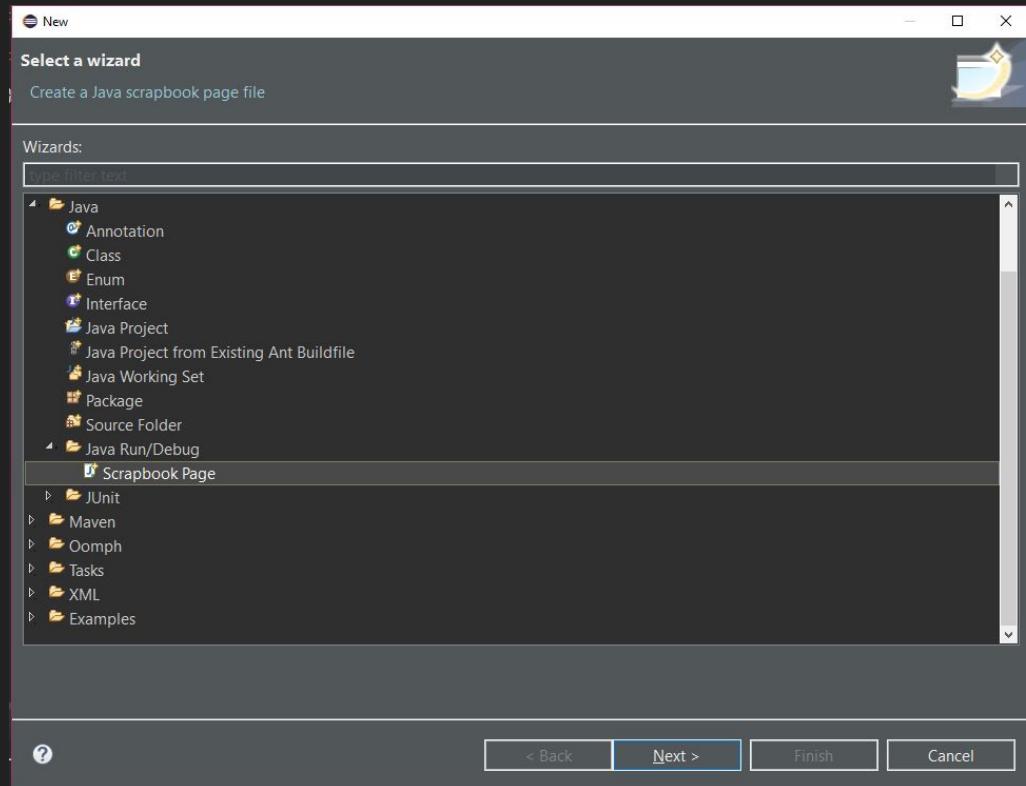
Do you want to add comments? (Configure templates and default value [here](#)) Generate comments

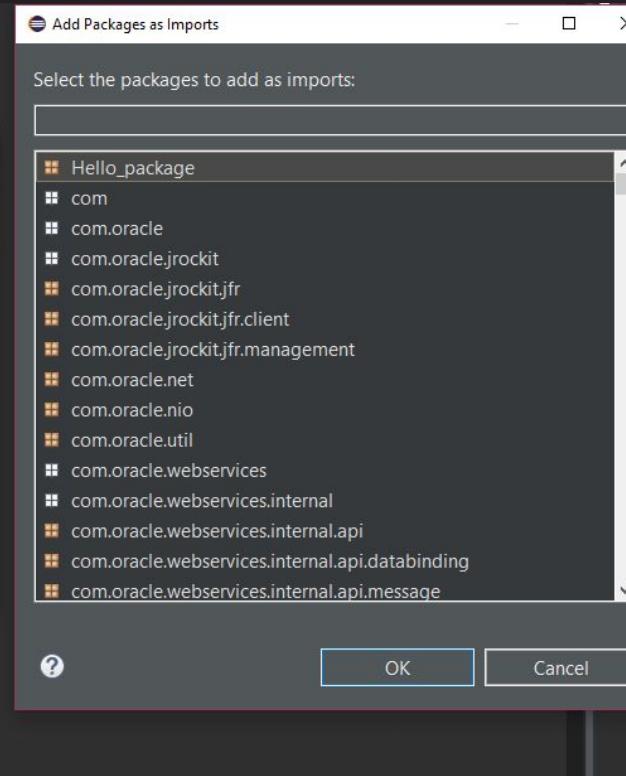
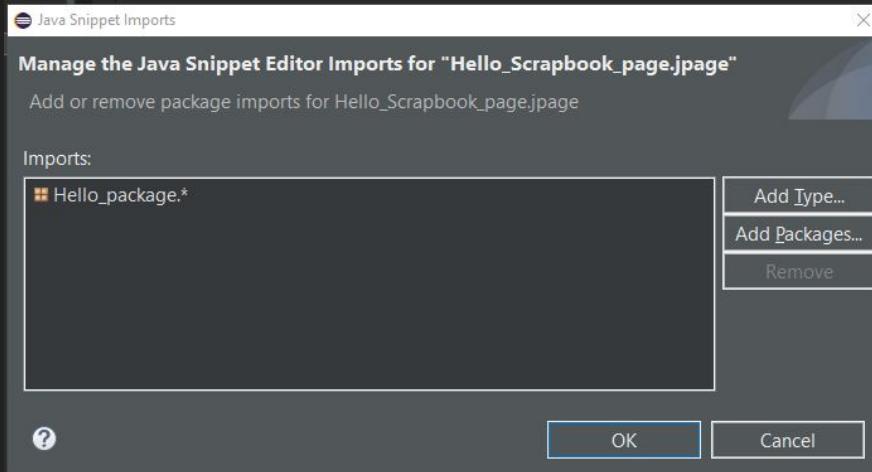
Finish

Cancel

# Scrapbook Page

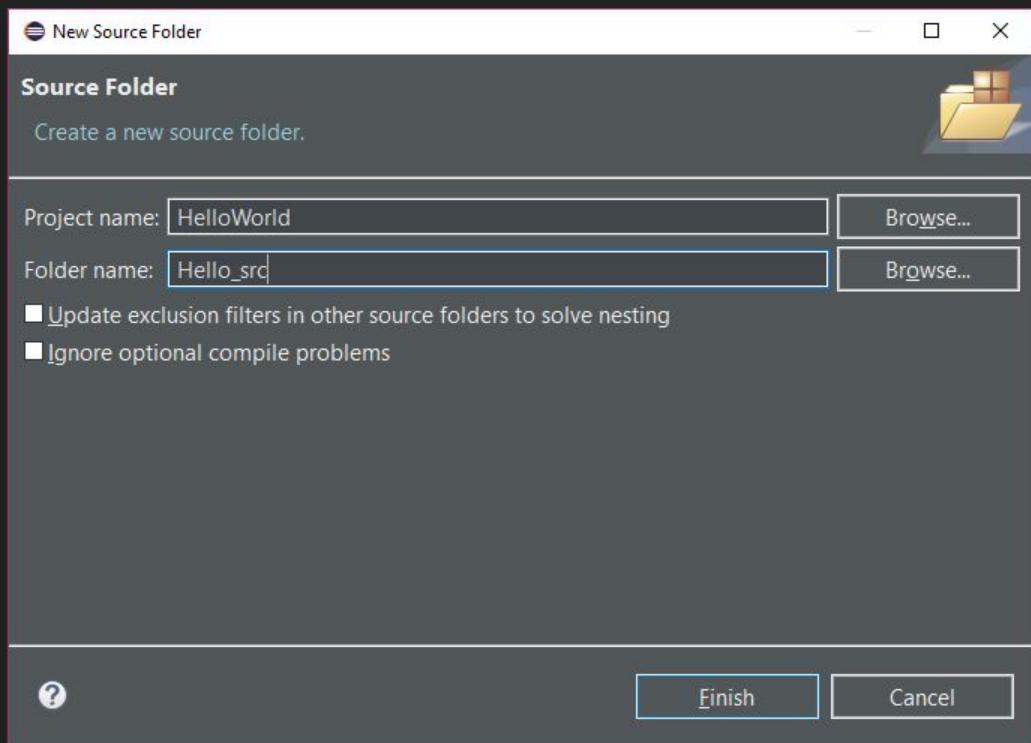
- Test code fragments.
- For creating a scrapbook page,  
Right click -> New ->  
Other..., or Ctrl+N
- For importing  
necesary class, Right  
click -> Set imports...





# Source Folder

- Intended to store .java source files.
- Automatically compiled in .class



# 3

## IDE'S PROGRAMMING TOOLS

# Debugger and compiler

The screenshot shows the Eclipse IDE interface with two code editors open:

- PruebaScrapbook.jpage**: Contains a multi-line string of lyrics from the song 'Eye of the Tiger'.
- ExampleClassTest.java**: Contains a Java class with a main method that prints "¡Hola, Eclipse!" to the console.

A red arrow points from the lyrics in the first editor to the `System.out.println("¡Hola, Eclipse!");` line in the second editor, highlighting the error. A tooltip box appears over the error, containing the following options:

- Change to 'out'
- Rename in file (Ctrl+2, R)

The status bar at the bottom displays the message: "Press 'Tab' from proposal table or click for focus".

At the bottom of the screen, there are tabs for **Problems**, **Javadoc**, **Declaration**, and **Console**. The **Problems** tab shows the status: "1 error, 0 warnings, 0 others". The **Errors** section lists one item:

Description	Resource	Path	Location	Type
System.out cannot be resolved or is not a field	ExampleClassTest.java		6: System.out.println("¡Hola, Eclipse!");	Error

# Autocomplete

Choose a lazy person to do a hard job,  
because that person will find a easy way to do it.

Be clever, be lazy.

# CTRL + [SPACEBAR]



# Class Names

A screenshot of an IDE showing Java code. The cursor is at line 11, column 5, where the identifier 'Obj' is being typed. A code completion dropdown menu is open, listing several suggestions:

- C Object - java.lang
- I ObjectInput - java.io
- I ObjDoubleConsumer - java.util.function
- I Object - org.omg.CORBA
- C ObjectAlreadyActive - org.omg.PortableServer.POAPacka
- A ObjectAlreadyActiveHelper - org.omg.PortableServer.POA
- I ObjectChangeListener - javax.naming.event
- I ObjectFactory - javax.naming.spi
- I ObjectFactoryBuilder - javax.naming.spi
- C ObjectHelper - org.omg.CORBA
- C ObjectHolder - org.omg.CORBA

The suggestion 'Object - java.lang' is highlighted with a green circle icon. The code editor shows a portion of a main method with lyrics from a song.

```
9
10    public static void main(String[] args) {
11        Obj|
13    }
14
15    System.out.println("Hello, World!");
16}
17
18public class Obj {
19    public static void main(String[] args) {
20        System.out.println("Hello, World!");
21    }
22}
23}
24
25/*
26 * We're no strangers to love
27 * You know the rules and so do I
28 * A full commitment's what I'm thinking of
29 * You wouldn't get this from any other guy
30 * I just wanna tell you how I'm feeling
31 * Gotta make you understand
32 * Never gonna give you up
33 * Never gonna let you down
34 * Never gonna run around and desert you
35 * Never gonna make you cry
36 * Never gonna say goodbye
37 * Never gonna tell a lie and hurt you
38 * We've known each other for so long
39 * Your heart's been aching but you're too shy to say it
40 * Inside we both know what's been going on
41 */
42
43
44
45
46
```

On the right side of the screen, there is a tooltip for the 'Object' class:

Class Object is the root of the class hierarchy. Every class has Object as a superclass. All objects, including arrays, implement the methods of this class.  
Since: JDK1.0  
See Also: [Class](#)

At the bottom of the code editor, there are two status messages:

Press 'Ctrl+Space' to show Java Non-type Proposals

Press 'Tab' from proposal table or click for focus

# Local variables & attributes

```
public static void main(String[] args) {  
    intPl  
    • intPrivate : int - ExampleClassTest  
    • intPublic : int - ExampleClassTest  
    System IntPredicate - java.util.function  
}  
  
public st  
  
}  
  
*  
I'e're no strangers to love  
'ou know the rules and so do I  
I full commitment's what I'm thinking of  
'ou wouldn't get this from any other guy  
I just wanna tell you how I'm feeling  
iotta make you understand  
Iever gonna give you up  
Iever gonna let you down  
Iever gonna run around and desert you  
Iever gonna make you cry  
Iever gonna say goodbye  
Iever gonna tell a lie and hurt you  
Ie've known each other for so long  
'our heart's been aching but you're too shy to say it  
Inside we both know what's been going on
```

# Methods & getters and setters

A screenshot of an IDE showing Java code completion. The code defines a class ExampleClassTest with a main method. Inside the main method, there is a line of code: `byte b= i.`. A tooltip appears over the dot, showing the method signature `byteValue(): byte - Integer` and a list of other methods from the Integer class. The list includes: compareTo(Integer anotherInteger), doubleValue(), equals(Object obj), floatValue(), getClass(), hashCode(), intValue(), longValue(), and shortValue(). Below the tooltip, the Java Language Specification is mentioned, specifically section 5.1.3 Narrowing Primitive Conversions.

```
1 public class ExampleClassTest {
2
3     public static void main(String[] args) {
4
5         Integer i= new Integer(1);
6         byte b= i.
7
8             byteValue(): byte - Integer
9                 • compareTo(Integer anotherInteger): int - Integer
10                • doubleValue(): double - Integer
11                • equals(Object obj): boolean - Integer
12                • floatValue(): float - Integer
13                • getClass(): Class<?> - Object
14                • hashCode(): int - Integer
15                • intValue(): int - Integer
16                • longValue(): long - Integer
17                • shortValue(): short - Integer
18
19 /**
20 * We're no strangers to love
21 * You know the rules and so do I
22 * Full commitment? • notify():void - Object
23 * You wouldn't get this from any other guy
24 * Press Ctrl+Space to show Java Non-type Proposals
25 * Just wanna tell you how I'm feeling
26 * Gotta make you understand
27 * Never gonna give you up
28 * Never gonna let you down
29 * Never gonna run around and desert you
30 * Never gonna make you cry
31 * Never gonna say goodbye
32 * Never gonna tell a lie and hurt you
33 * Never gonna tell a lie and hurt you
34 * We've known each other for so long
35 * Your heart's been aching but you're too shy to say it
36 * Inside we both know what's been going on
37 * We know the game and we're gonna play it
38 * And if you ask me how I'm feeling
39 * Don't tell me you're too blind to see
40 * Never gonna give you up
41 * Never gonna let you down
42 * Never gonna run around and desert you
43 * Never gonna make you cry
44 * Never gonna say goodbye
45 * We're no strangers to love
46 * You know the rules and so do I
```

A screenshot of an IDE showing Java code completion for the Integer constructor. The code defines a class ExampleClassTest with a main method. Inside the main method, there is a line of code: `Integer i= new Inte`. A tooltip appears over the dot, showing the constructor signature `Integer(int value) - java.lang.Integer` and a list of other constructor proposals. The list includes: Integer(String s), IntegerSyntax(int arg0), and IntegerSyntax(int arg0, int arg1, int arg2). To the right of the tooltip, a detailed description of the Integer constructor is provided, stating that it constructs a newly allocated Integer object that represents the specified int value. Parameters are described as value - the value to be represented by the Integer object.

```
1 public class ExampleClassTest {
2
3     public static void main(String[] args) {
4
5         Integer i= new Inte.
6
7             Integer(int value) - java.lang.Integer
8                 • Integer(String s) - java.lang.Integer
9                 • IntegerSyntax(int arg0) Anonymous Inner Type - javax.print.
10                • IntegerSyntax(int arg0, int arg1, int arg2) Anonymous Inner
11
12 /**
13 * We're no strangers to love
14 * You know the rules and so do I
15 * A full commitment's what I'm thinking of
16 * Press Ctrl+Space to show Java Non-type Proposals
17 * Just wanna tell you how I'm feeling
18 * Gotta make you understand
19 * Never gonna give you up
20 * Never gonna let you down
21 * Never gonna run around and desert you
22 * Never gonna make you cry
23 * Never gonna say goodbye
24 * Never gonna tell a lie and hurt you
25 * We've known each other for so long
26 * Your heart's been aching but you're too shy to say it
27 * Inside we both know what's been going on
28 * We know the game and we're gonna play it
29 * And if you ask me how I'm feeling
30 * Don't tell me you're too blind to see
31 * Never gonna give you up
32 * Never gonna let you down
33 * Never gonna run around and desert you
34 * Never gonna make you cry
35 * Never gonna say goodbye
36 * We're no strangers to love
37 * You know the rules and so do I
38 * A full commitment's what I'm thinking of
```

# Loops

The screenshot shows a Java code editor with the file "ExampleClassTest.java" open. The cursor is at the start of a new line, and a code completion dropdown menu is displayed, listing various loop constructs. The menu includes:

- for - iterate over array
- for - iterate over array with temporary variable
- for - iterate over collection
- foreach - iterate over an array or iterable
- for
- ForegroundAction - javax.swing.text.StyledEditorKit
- ForkJoinPool - java.util.concurrent
- ForkJoinTask - java.util.concurrent
- ForkJoinWorkerThread - java.util.concurrent
- ForkJoinWorkerThreadFactory - java.util.concurrent.ForkJoin

The main code area contains the following Java code:

```
public class ExampleClassTest {  
    public static void main(String[] args) {  
        for|  
    }  
}  
  
for (int i = 0; i < args.length; i++) {  
    String string = args[i];  
}
```

A status bar at the bottom right of the editor window displays the message: "Press 'Tab' from proposal table or click for focus".

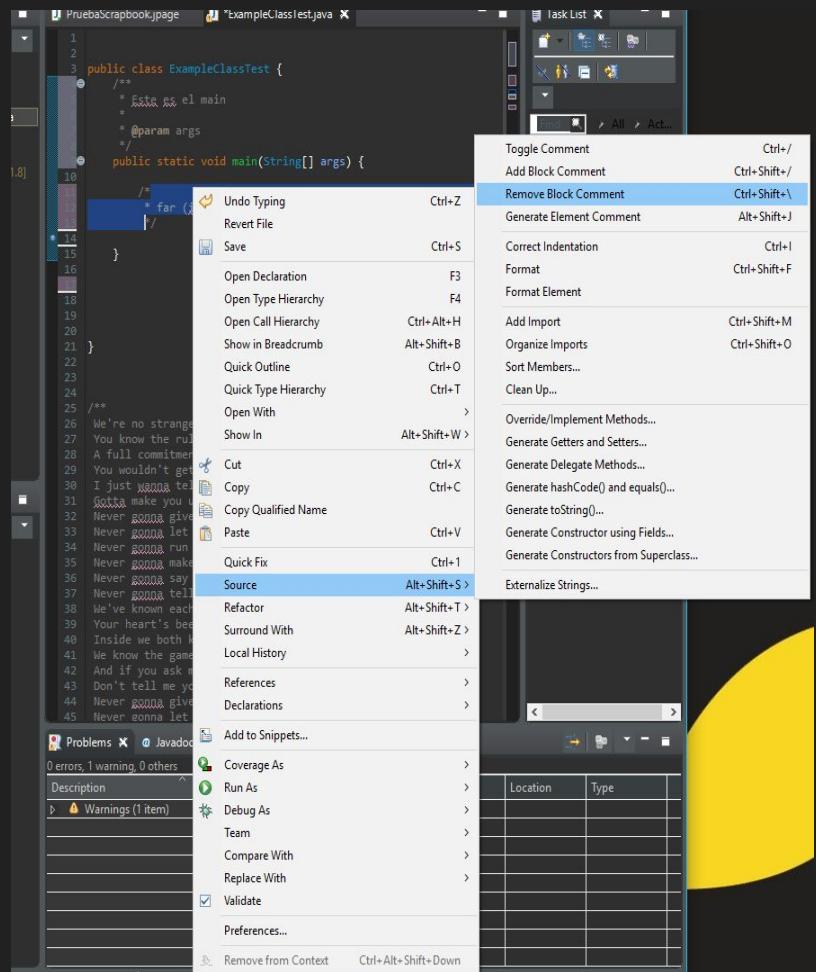
# Javadoc tags

The screenshot shows a Java code editor with the file `*ExampleClassTest.java` open. The code contains a class definition:public class ExampleClassTest {  
 /\*\*  
 \* Este es el main  
 \*/  
 public void main() {  
 System.out.println("Hello, World!");  
 }  
}A cursor is positioned at the end of the first line of the multi-line Javadoc comment. A completion dropdown menu is displayed, listing various HTML tags starting with '<>' followed by the tag name. The menu includes: <a>, <b>, <blockquote>, <br>, <code>, <dd>, <dl>, <dt>, <em>, <h1>, <h2>, and <h3>. At the bottom of the dropdown, a tooltip reads "Press 'Ctrl+Space' to show Java Non-Type Proposals".

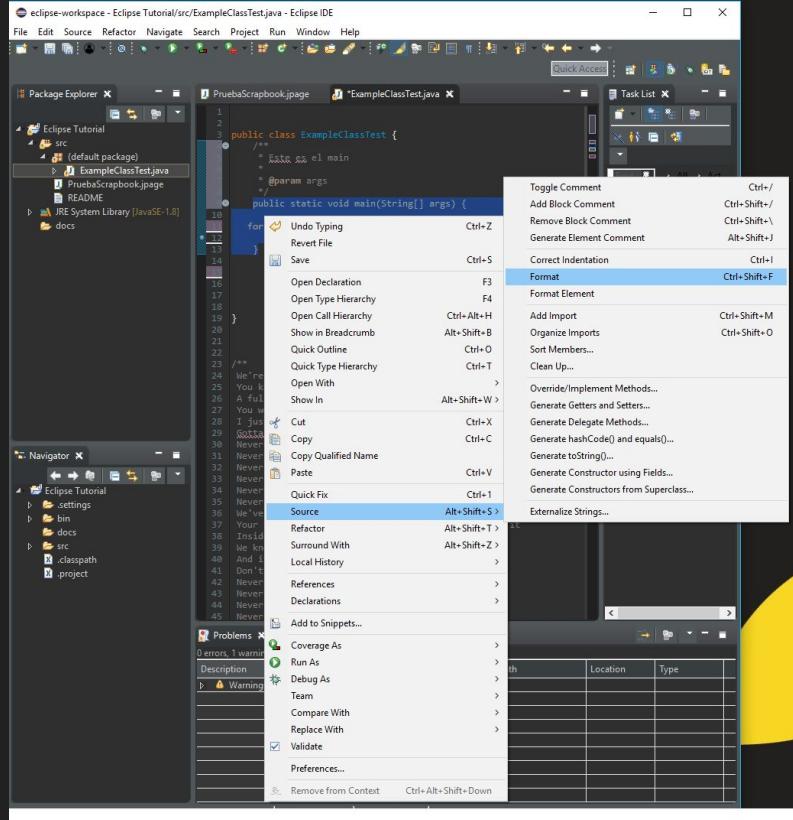
# Source Menu

Do you have any idea how fast i am?

# Comment/uncomment CTRL+/-



# Format

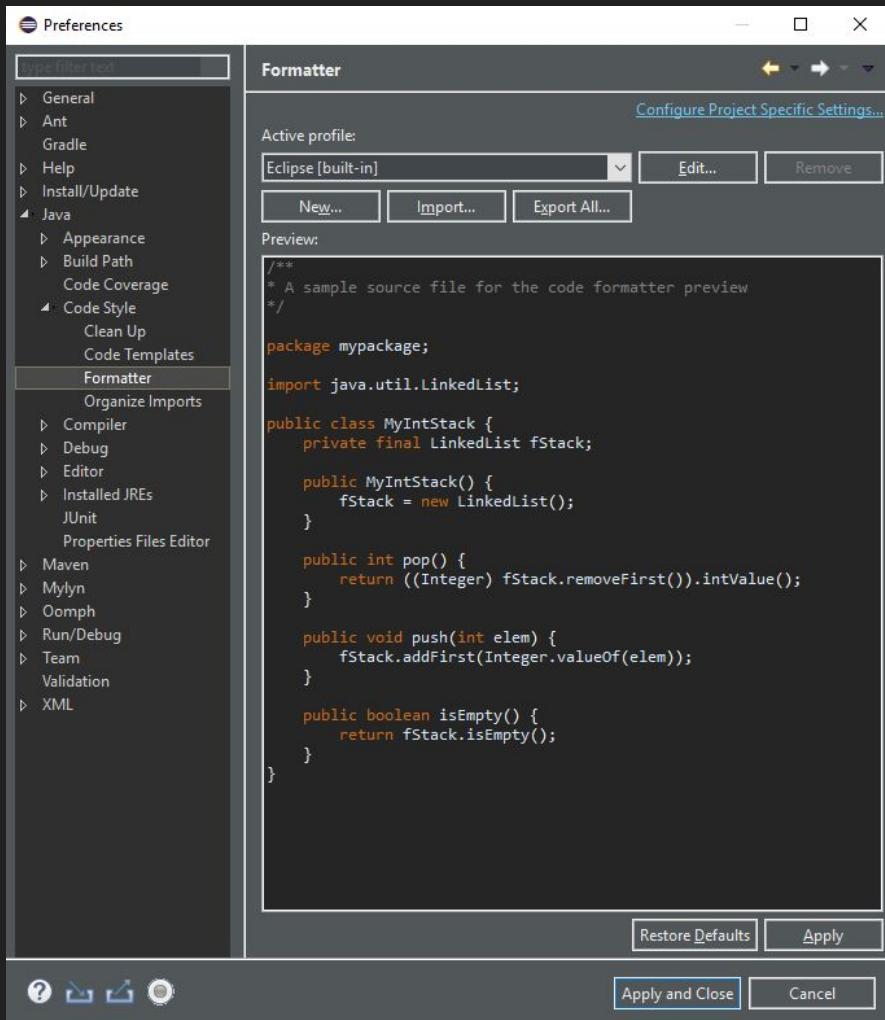


A screenshot of the Eclipse IDE interface showing the result of using the 'Format' command. The Java code has been re-indented and the comment '/\* cosas \*/' has been removed, resulting in:public static void main(String[] args) {  
 for (int i = 0; i < args.length; i++) {}  
}

CTRL+SHIFT+F

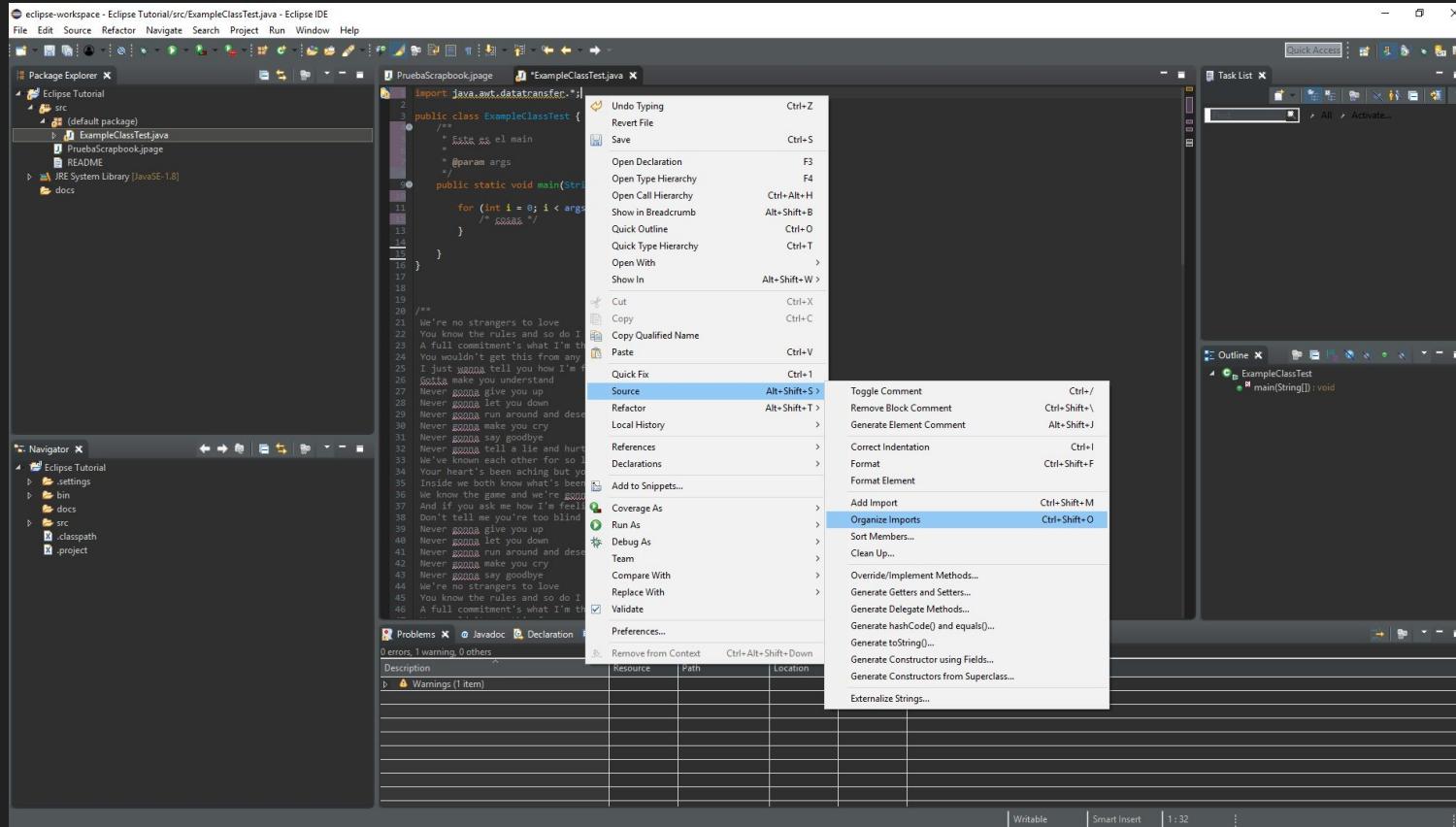
# Edit format

Window> Preferences  
Java> Code Style> Formatter



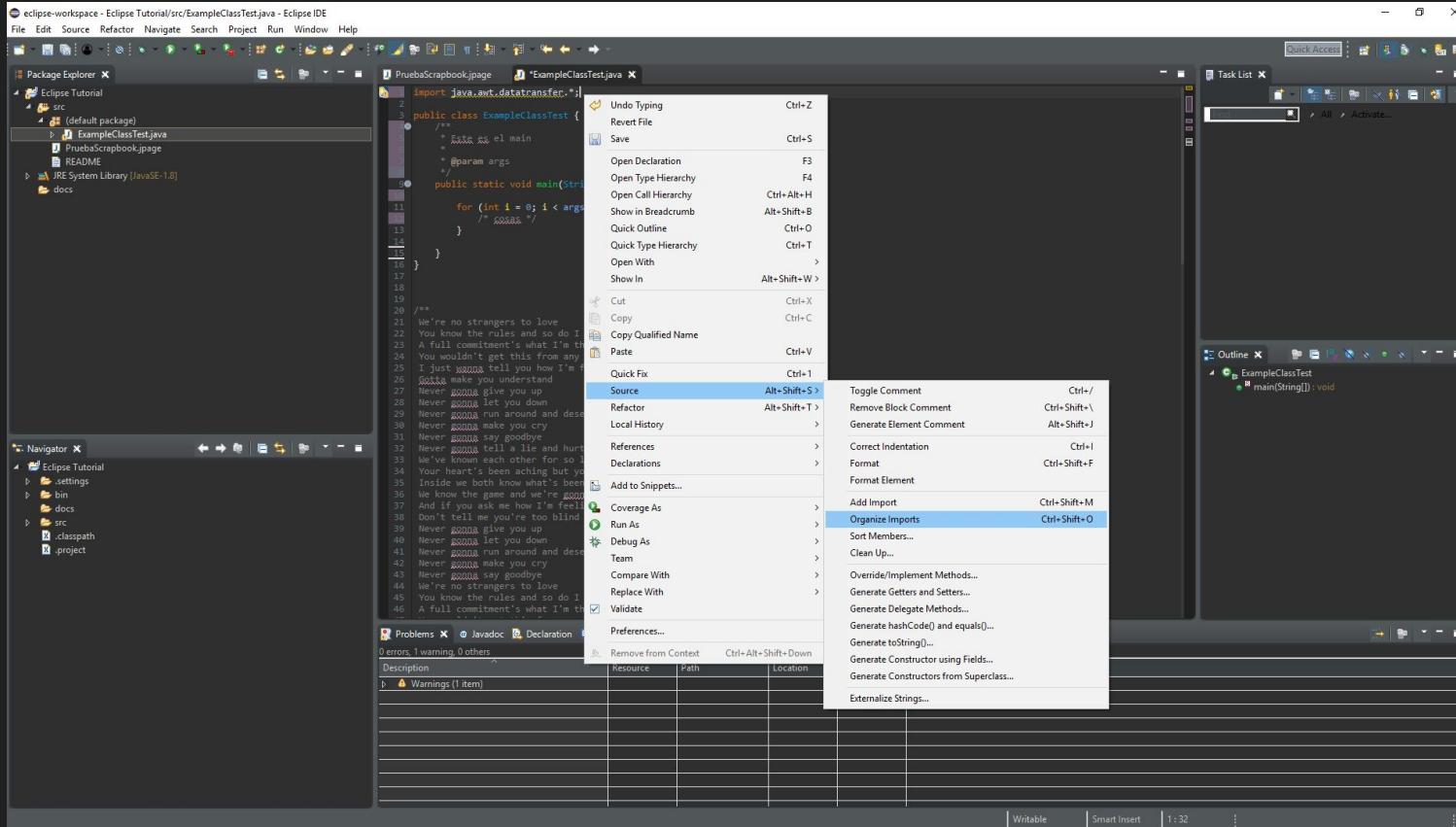
# Organize imports

CTRL+SHIFT+O



# Add imports

CTRL+SHIFT+M



# Override/Implement Methods

The screenshot shows an IDE interface with two main windows. On the left is a code editor titled "PruebaScrapbook.java" containing Java code. The code includes a main method with a for loop and several comments. On the right is a modal dialog titled "Override/Implement Methods". The dialog lists methods from the Object class that can be overridden or implemented. It includes checkboxes for "Select All" and "Deselect All", an "Insertion point:" dropdown set to "First member", and a checkbox for "Generate method comments". A note at the bottom states: "The format of the method stubs may be configured on the [Code Templates](#) preference page." At the bottom of the dialog are "OK" and "Cancel" buttons.

```
import java.awt.datatransfer.*;  
public class ExampleClassTest {  
    /*  
     * Este es el main  
     * @param args  
     */  
    public static void main(String[] args) {  
        for (int i = 0; i < args.length; i++) {  
            /*  
             */  
        }  
    }  
    /**  
     * We're no strangers to love  
     * You know the rules and so do I  
     * A full commitment's what I'm thinking of  
     * You wouldn't get this from any other guy  
     * I just wanna tell you how I'm feeling  
     * Cotta make you understand  
     * Never gonna give you up  
     * Never gonna let you down  
     * Never gonna run around and desert you  
     * Never gonna make you cry  
     * Never gonna say goodbye  
     * Never gonna tell a lie and hurt you  
     * We've known each other for so long  
     * Your heart's been aching but you're too shy to say it  
     * Inside we both know what's been going on  
     * We know the game and we're gonna play it  
     * And if you ask me how I'm feeling  
     * Don't tell me you're too blind to see  
     * Never gonna give you up  
     * Never gonna let you down  
     * Never gonna run around and desert you  
     * Never gonna make you cry  
     * Never gonna say goodbye  
     * We're no strangers to love  
     * You know the rules and so do I  
    */
```

Select methods to override or implement:

- Object
  - clone()
  - equals(Object)
  - finalize()
  - hashCode()
  - toString()

Insertion point:  
First member

Generate method comments

The format of the method stubs may be configured on the [Code Templates](#) preference page.

OK Cancel

# Generate Getters/Setters

The screenshot shows an IDE interface with two tabs: 'PruebaScrapbook.jpage' and 'ExampleClassTest.java'. The code editor displays Java code for a class named 'ExampleClassTest'.

```
1 import java.awt.datatransfer.*;
2
3
4 public class ExampleClassTest {
5     private int intPrivate=6;
6
7
8
9
10 /**
11 * Esta es el main
12 *
13 * @param args
14 */
15 public static void main(String[] args) {
16     for (int i = 0; i < args.length; i++) {
17         /* cosas */
18     }
19 }
20
21 }
22 }
23
24
25 /**
26 We're no strangers to love
27 You know the rules and so do I
28 A full commitment's what I'm thinking of
29 You wouldn't get this from any other guy
30 I just wanna tell you how I'm feeling
31 gotta make you understand
32 Never gonna give you up
33 Never gonna let you down
34 Never gonna run around and desert you
35 Never gonna make you cry
36 Never gonna say goodbye
37 Never gonna tell a lie and hurt you
38 We've known each other for so long
39 Your heart's been aching but you're too shy to say it
40 Inside we both know what's been going on
41 We know the game and we're gonna play it
42 And if you ask me how I'm feeling
43 Don't tell me you're too blind to see
44 Never gonna give you up
45 Never gonna let you down
46 Never gonna
```

A modal dialog titled 'Generate Getters and Setters' is open. It lists a single field 'intPrivate' with its corresponding getter 'getIntPrivate()' and setter 'setIntPrivate(int)'. There are buttons for 'Select All', 'Deselect All', 'Select Getters', and 'Select Setters'. Below the list, there are several configuration options:

- Allow setters for final fields (remove 'final' modifier from fields if necessary)
- Insertion point: After 'intPrivate'
- Sort by: Fields in getter/setter pairs
- Access modifier:
  - public
  - protected
  - package
  - private
- final
- synchronized
- Generate method comments

The message at the bottom states: 'The format of the getters/setters may be configured on the [Code Templates](#) preference page.'

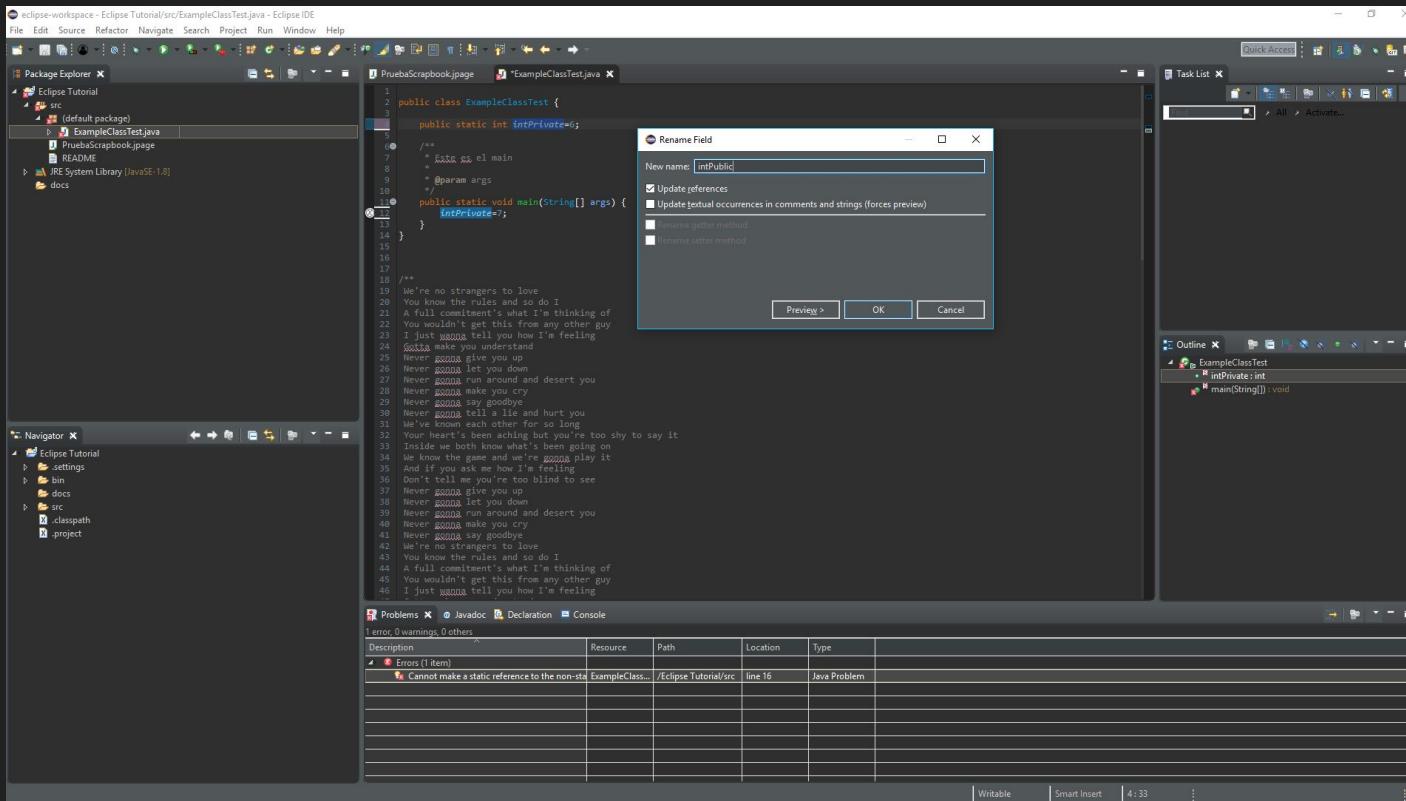
At the bottom of the dialog are 'Generate' and 'Cancel' buttons.

# Refactor Menu

Don't use this to copy anyone's code. It won't work.

# Rename

ALT+SHIFT+R



Another interesting stuff we can do with the Refactor Menu:

- Move (folders)
- Change Method Signature
- Pull Up and Push Down

# Documentation

No, your code isn't self explanatory, moron.

# External Javadoc doc.

- Use Mayus + F2 to check Javadoc documentation for anything.
- Javadoc has also a Eclipse view:  
Window > Show View... > Java > Javadoc
- We can import JAR files on Properties > JBP -> Libraries -> +Add External JARs.

# 4

## ECLIPSE VIEWS

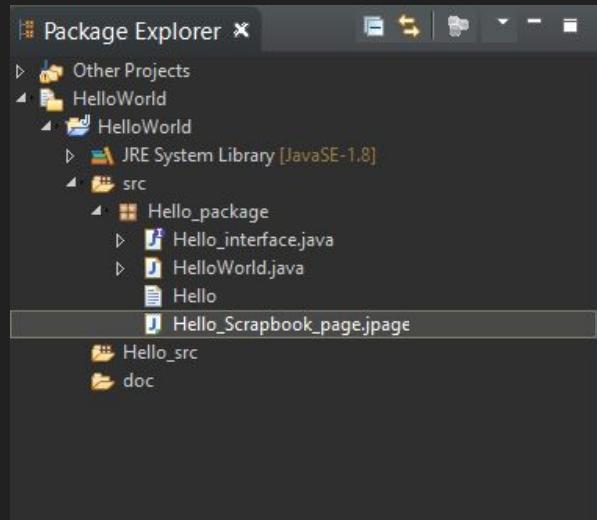
# Views of Eclipse

The user interface of Eclipse consists of two types:

- Editors -> Perform a complete Task
- Views -> Support functions

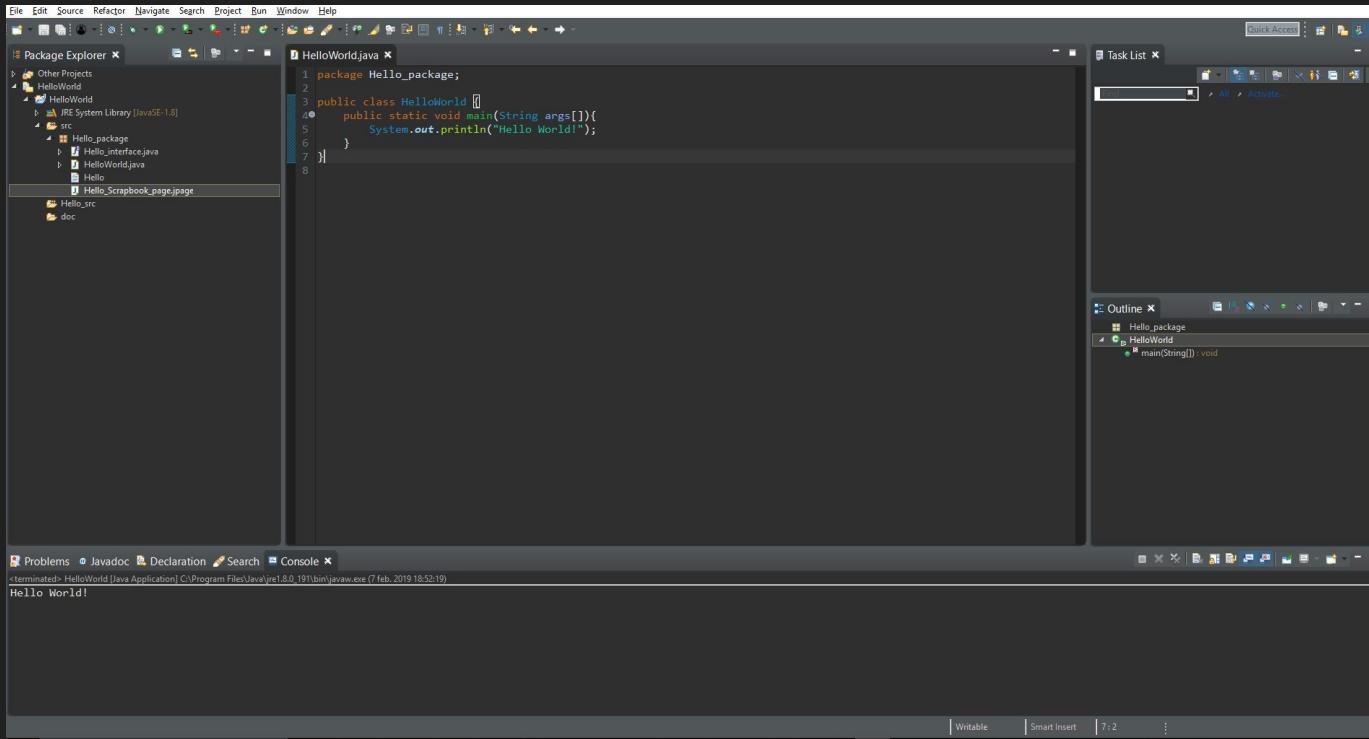
A screenshot of the Eclipse Java Editor. The active tab is "HelloWorld.java". The code displayed is:

```
1 package Hello_package;
2
3 public class HelloWorld {
4     public static void main(String args[]){
5         System.out.println("Hello World!");
6     }
7 }
8
```

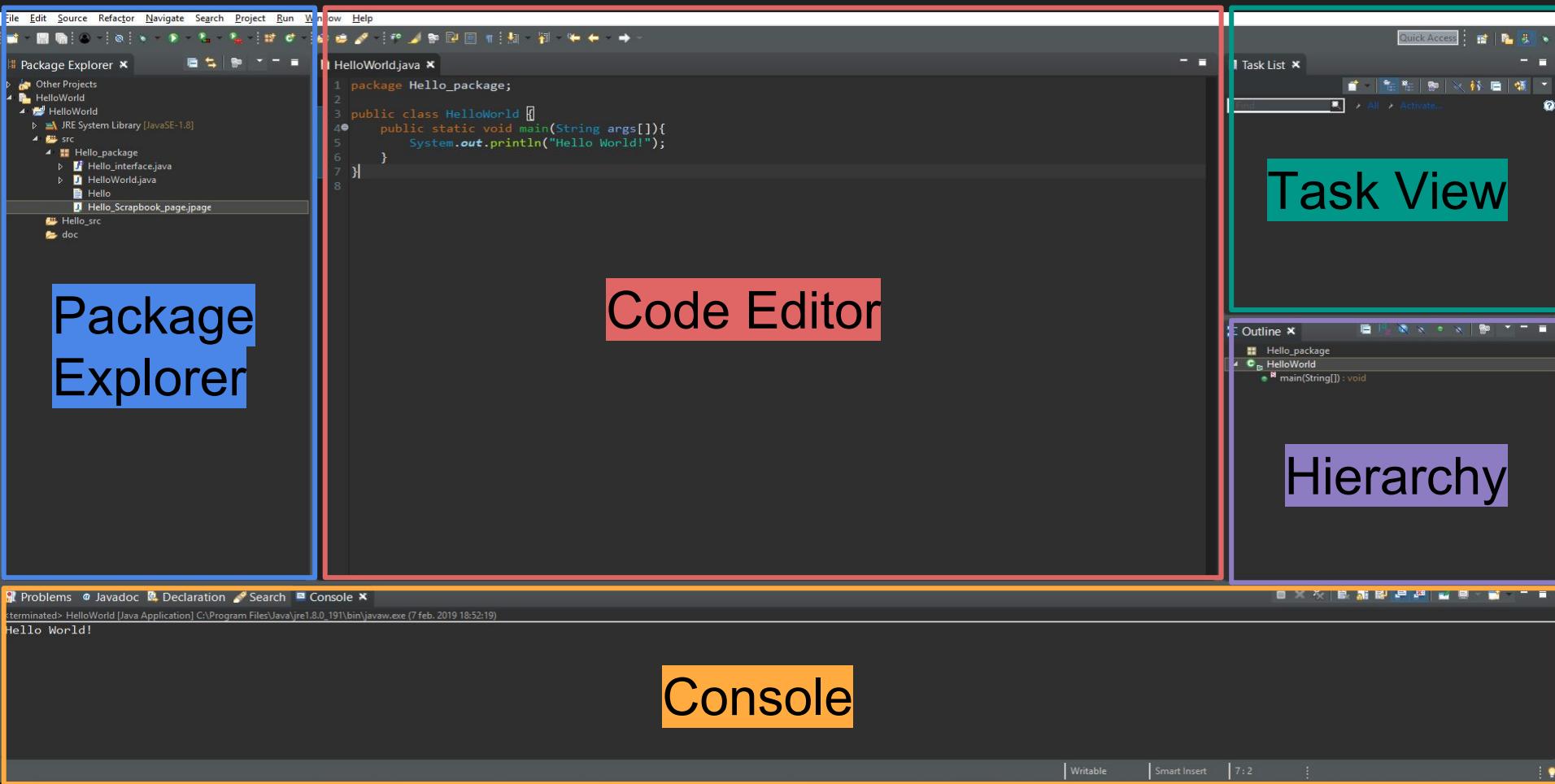


# Perspectives

- Set of editors and views to support an activity



# Java



# Resource

# Outline

The screenshot shows a Java code editor interface with two tabs open: `HelloWorld.java` and `Círculo.java`. The `HelloWorld.java` tab is active, displaying the following code:

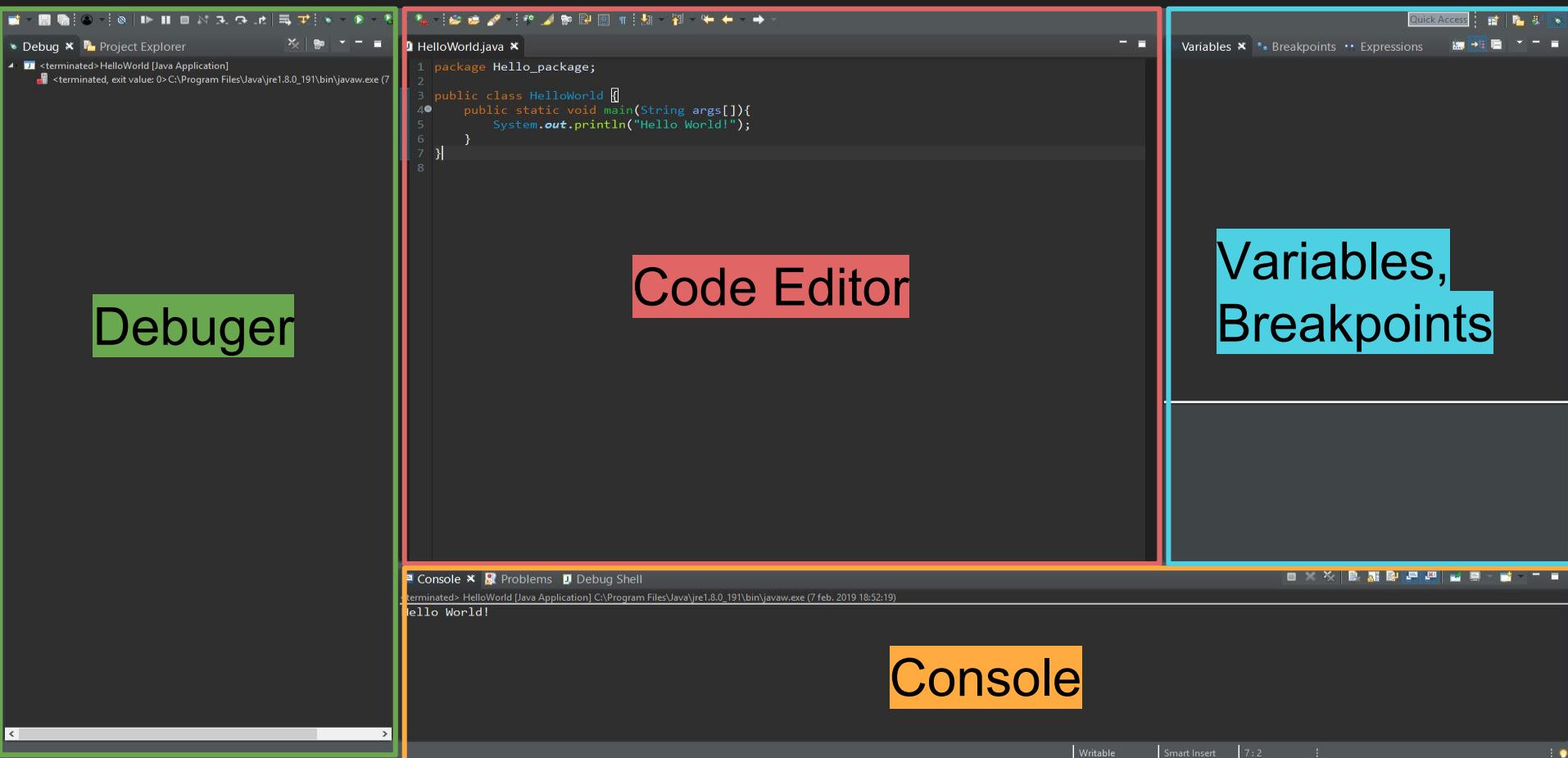
```
1 package Hello_package;
2
3 public class HelloWorld {
4     int x = 0;
5     private int z= 0;
6     protected int w = 0;
7     public int y = 0;
8     public static void main(String args[]){
9         System.out.println("Hello World!");
10        int a = 1;
11        int b = 2;
12        int c = 0;
13        c = a+b;
14        System.out.println(c);
15    }
16
17    public int getZ() {
18        return z;
19    }
20
21    public void setZ(int z) {
22        this.z = z;
23    }
24 }
```

A large red rectangular box covers the right side of the screen, containing the text "Code Editor". The status bar at the bottom indicates "Code Editor".

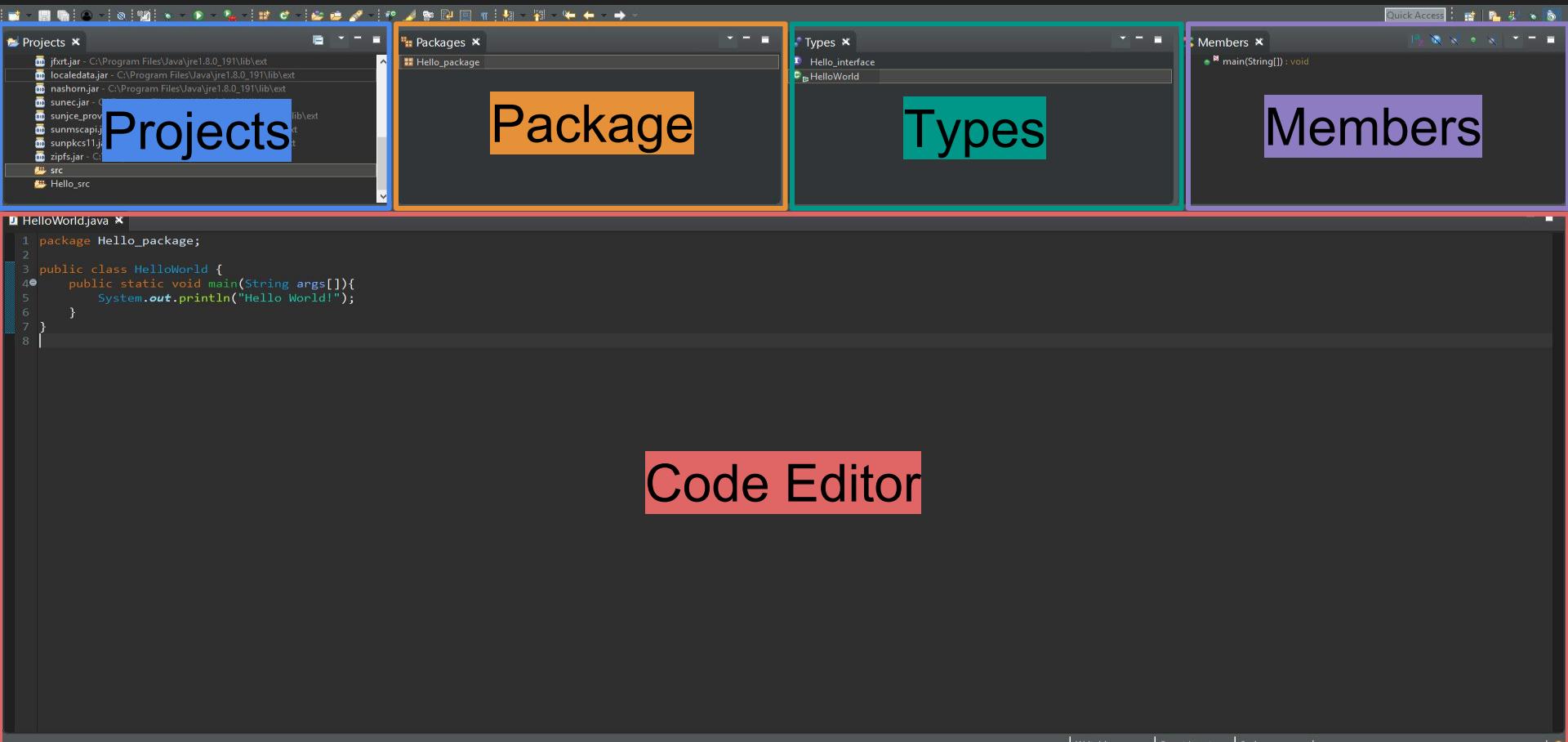
# Code Editor

# Task View

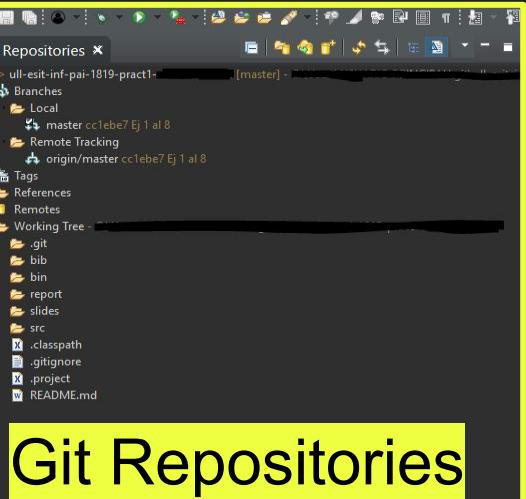
# Debugger



# Java Browsing



# Git



A screenshot of a code editor showing Java code for a "Pair" class. The code includes methods for creating a pair, getting its components, comparing pairs, reflecting, and swapping components. A red box highlights the word "Code Editor".

```
1 public class Pair {
2     int first, second;
3     public Pair(int first, int second) {
4         this.first = first;
5         this.second = second;
6     }
7     public String toString() {
8         return "(" + first + "," + second + ")";
9     }
10    public boolean compare(Pair a, Pair b) {
11        if ((a.first == b.first) && (a.second
12            return true;
13        else
14            return false;
15    }
16    public void reflex () {
17        int z = first;
18        first = second;
19        second = z;
20    }
21    public void swap(Pair q) {
22        Pair temp = q;
23    }
}
```

A screenshot of a Git history interface. It shows a single commit entry:

ID	Message	Author	Authored Date	Committer	Committed Date
cc1ebe7	* [master] origin/master  HEAD  Ej 1 al 8	[REDACTED]	2 days ago	[REDACTED]	2 days ago

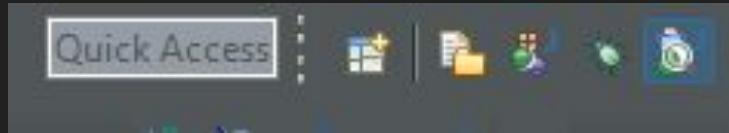
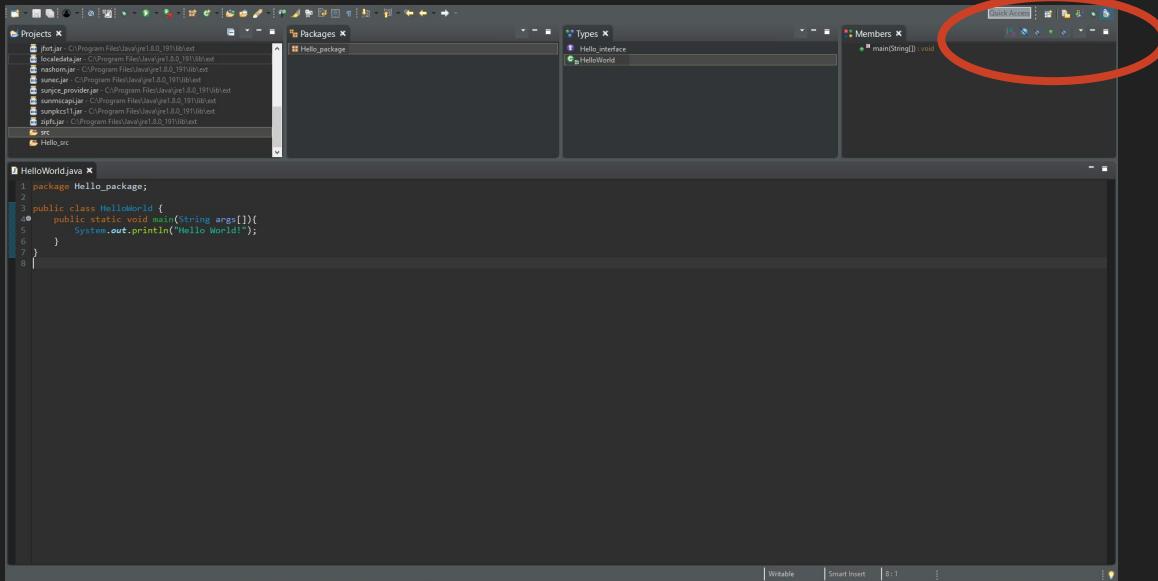
The commit message is "Ej 1 al 8". Below the table, there's a commit log showing the diff between the file "src/Pair.java" in the index and the working tree. A purple box highlights the word "History".

```
commit cc1ebe75d94000bbe96f9090173fc0277db0e881
Author: [REDACTED]
Committer: [REDACTED]
Branches: master, origin/master

Ej 1 al 8

diff --git a/src/Pair.java b/src/Pair.java
new file mode 100644
index 0000000..bd82220
--- /dev/null
+++ b/src/Pair.java
@@ -0,0 +1,28 @@
```

# Quick Access



# Tasks

- Allows quick management of pending tasks.
  - Window -> Show View -> Tasks
  - Right click -> Add Task
  - Priorities can be assigned.
  - Errors and warnings can appear here too.

 Add Task

Description:

Priority:

  Completed

On element:

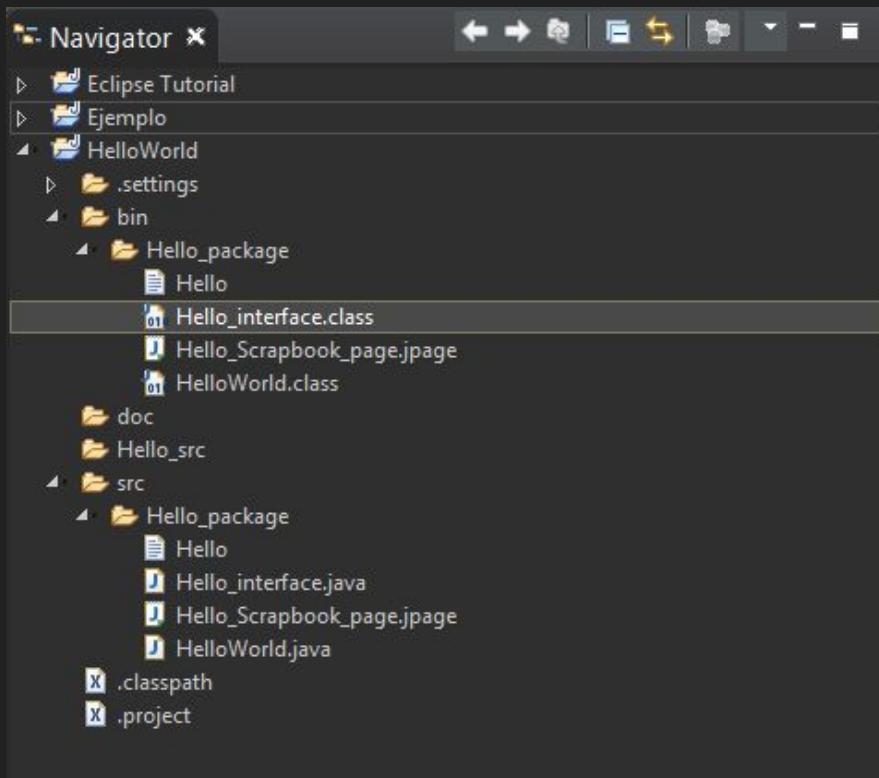
In folder:

Location:



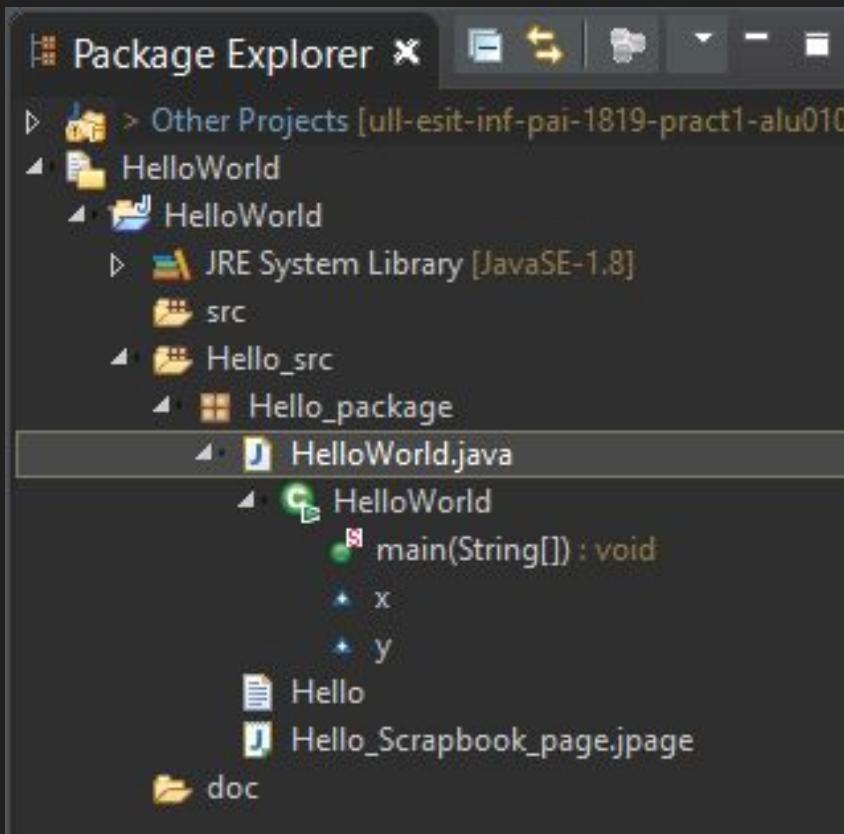
# Navigator

- We can see the structure of defined files.
- The only one that shows the folder '/bin'.



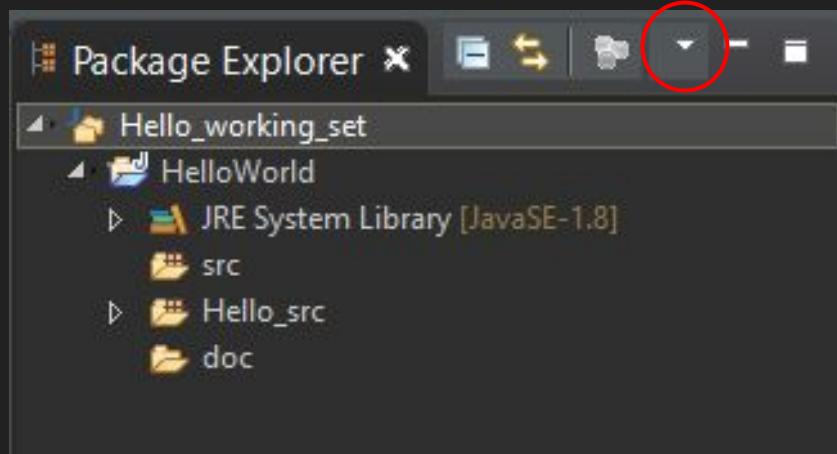
# Package Explorer

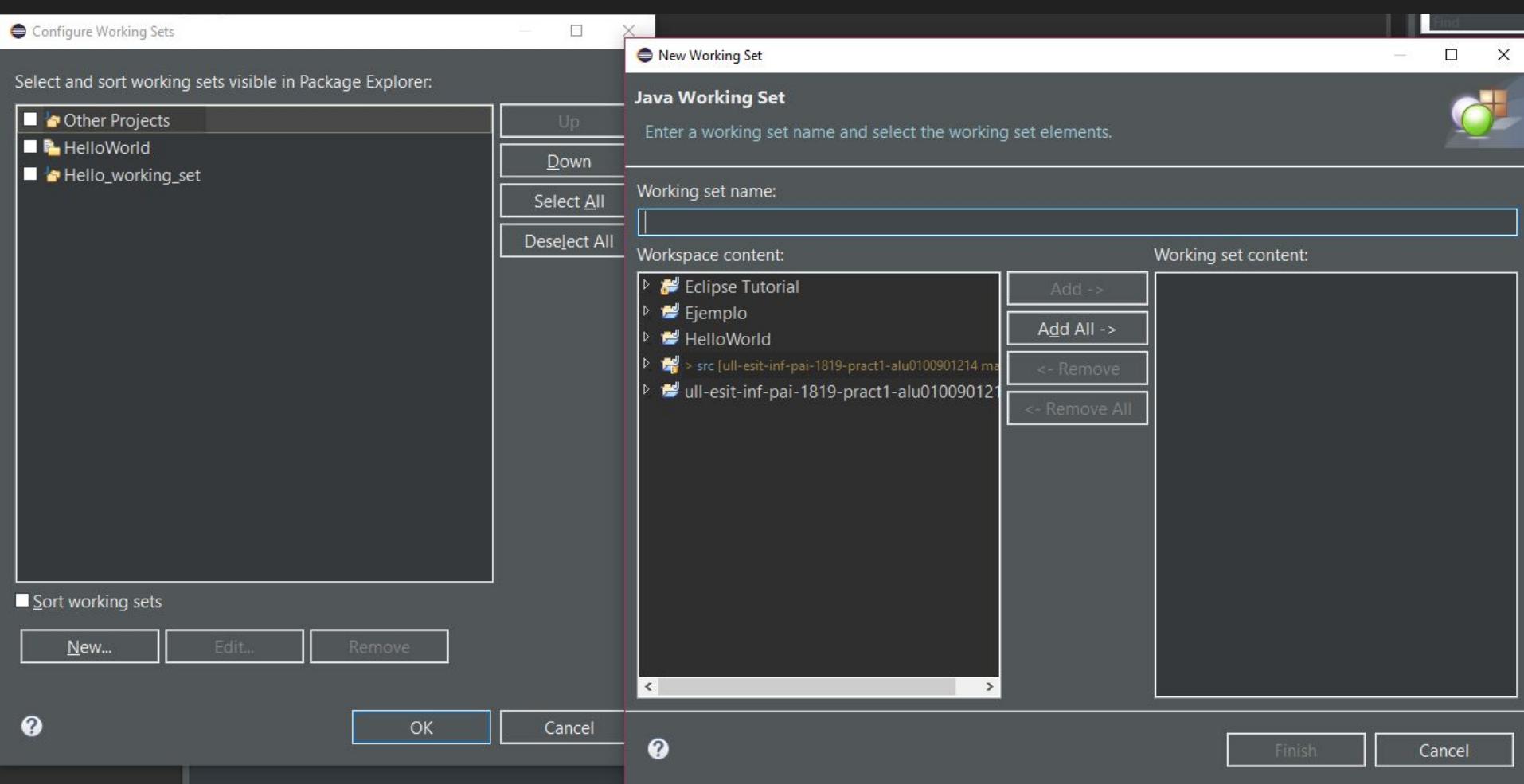
- Shows the logical structure of packages and java classes.
- .java files can be expanded to show their attributes and methods



# Working set

- They are used to separate the different projects that are being worked on.
- Useful when you are working on differentiated projects.
- View Menú -> Configure Working sets...





# Outline View

- A quick way to see the methods and attributes of a .java class.
- Icons give additional information about visibility.

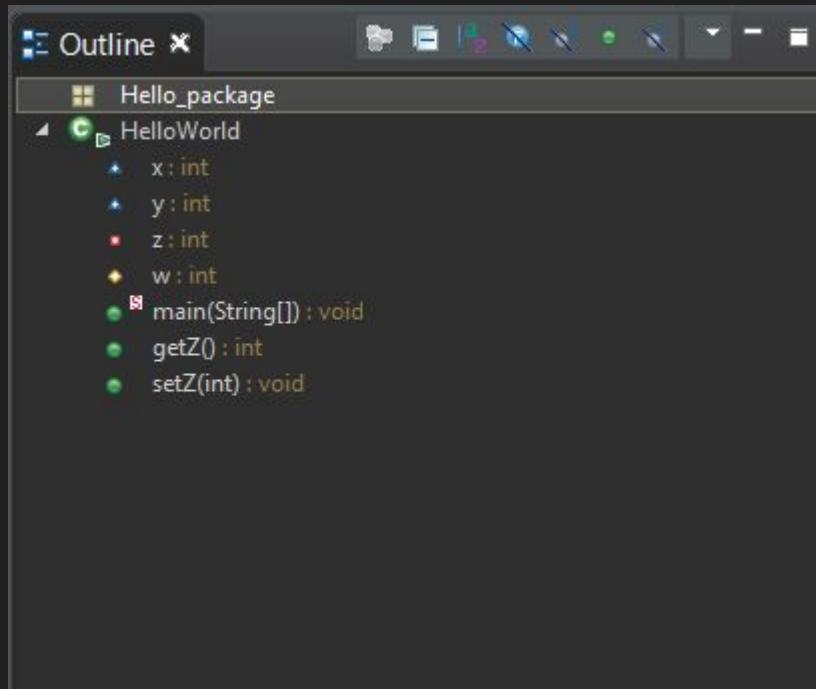


Package

Private

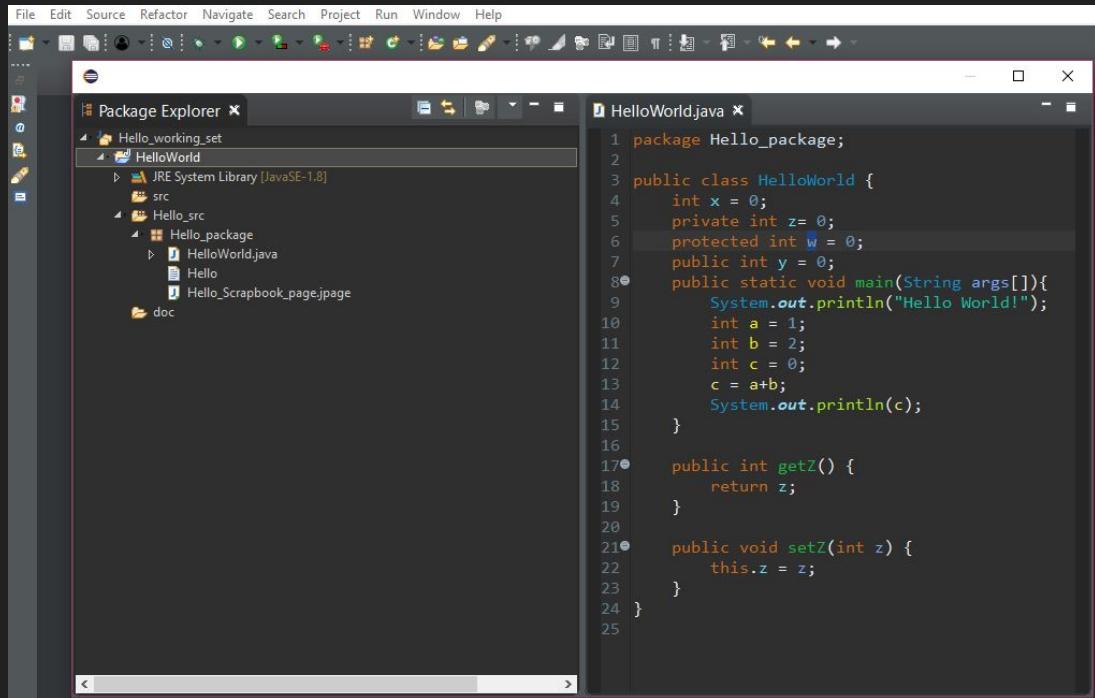
Protected

Public



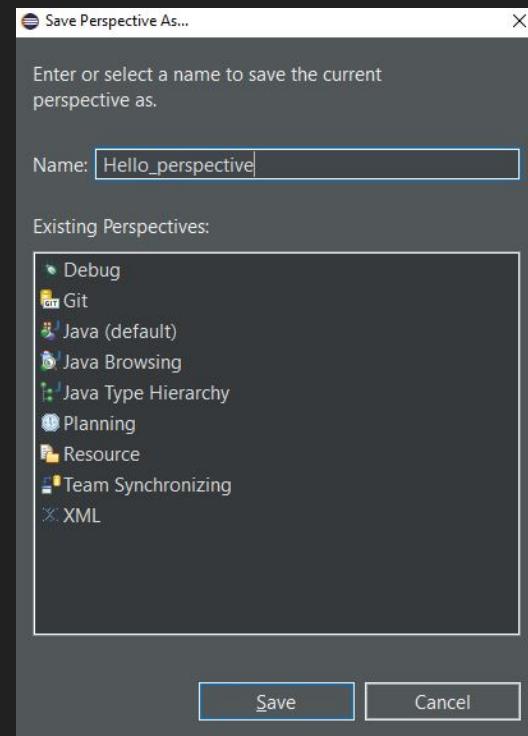
# Fast views

- Used to separate the view in a new window



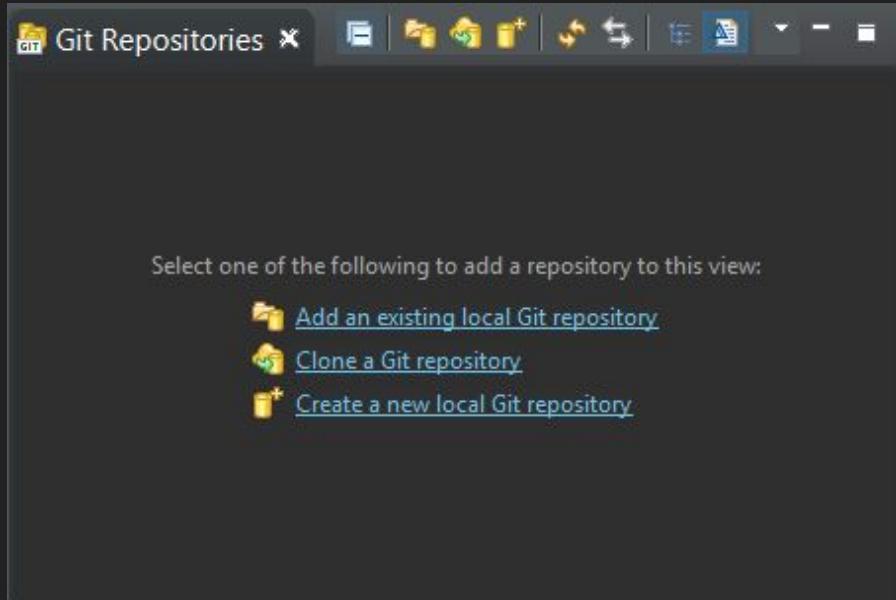
# Save a perspective

- Window -> Perspective... -> Save Perspective as ...
- Window -> Perspective -> Reset Perspective



# Git Repositories

- We can create a new repository.
- Add an existing one.
- Or Clone a repository





## Source Git Repository

Enter the location of the source repository.

Create new file

Upload files

Find file

Clone or download ▾

## Clone with HTTPS

Use SSH

Use Git or checkout with SVN using the web URL.

<https://github.com/ULL-ESIT-INF-PAI-1819>[Open in Desktop](#)[Download ZIP](#)

## Location

URI:

Local File...

Host:

Repository path:

## Connection

Protocol: Port: 

## Authentication

User: Password:  Store in Secure Store

&lt; Back

Next &gt;

Finish

Cancel

## New Java Project

### Create a Java Project

Create a Java project in the workspace or in an external location.



Project name:

Use default location

Location:  [Browse...](#)

#### JRE

Use an execution environment JRE:

Use a project specific JRE:

Use default JRE (currently 'jre1.8.0\_201') [Configure JREs...](#)

#### Project layout:

Use project folder as root for sources and class files

[Configure default...](#)

Create separate folders for sources and class files

#### Working sets

Add project to working sets

[New...](#)

Working sets:

[Select...](#)

**i** The wizard will automatically configure the JRE and the project layout based on the existing source.

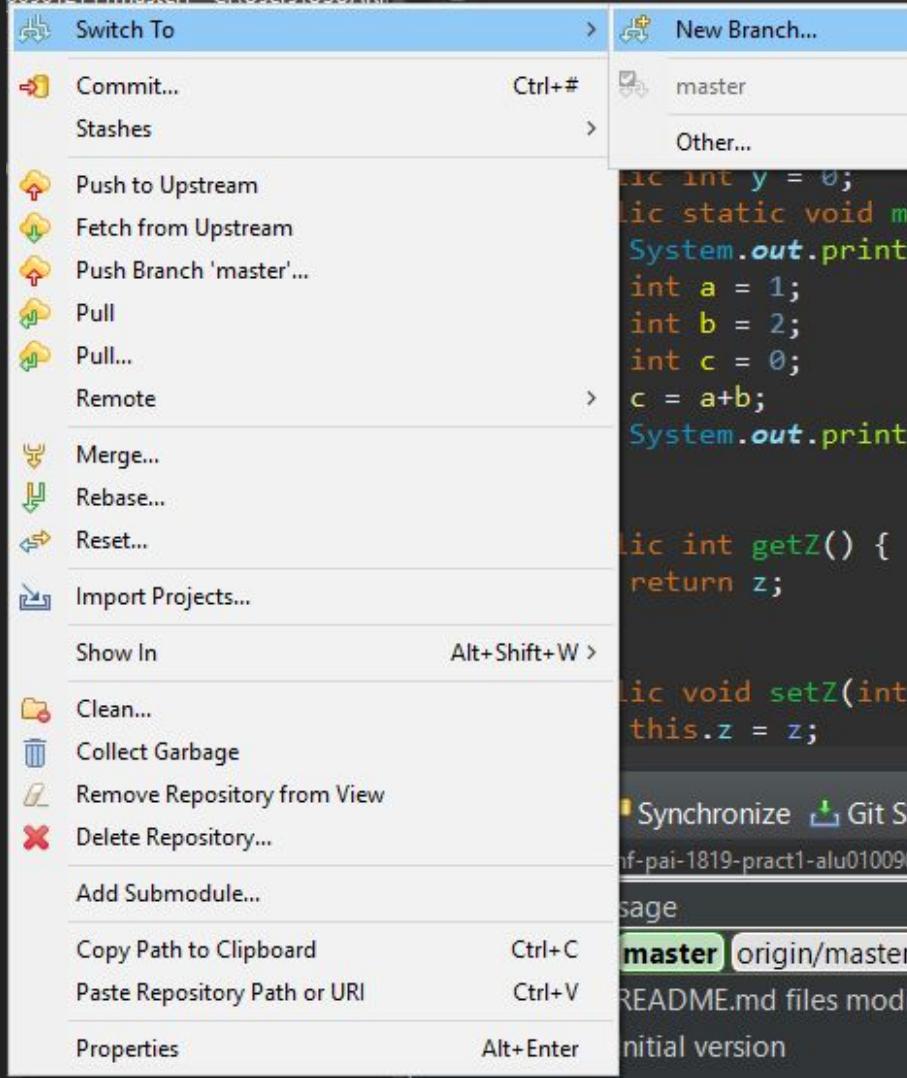


< Back

Next >

Finish

Cancel



# 5

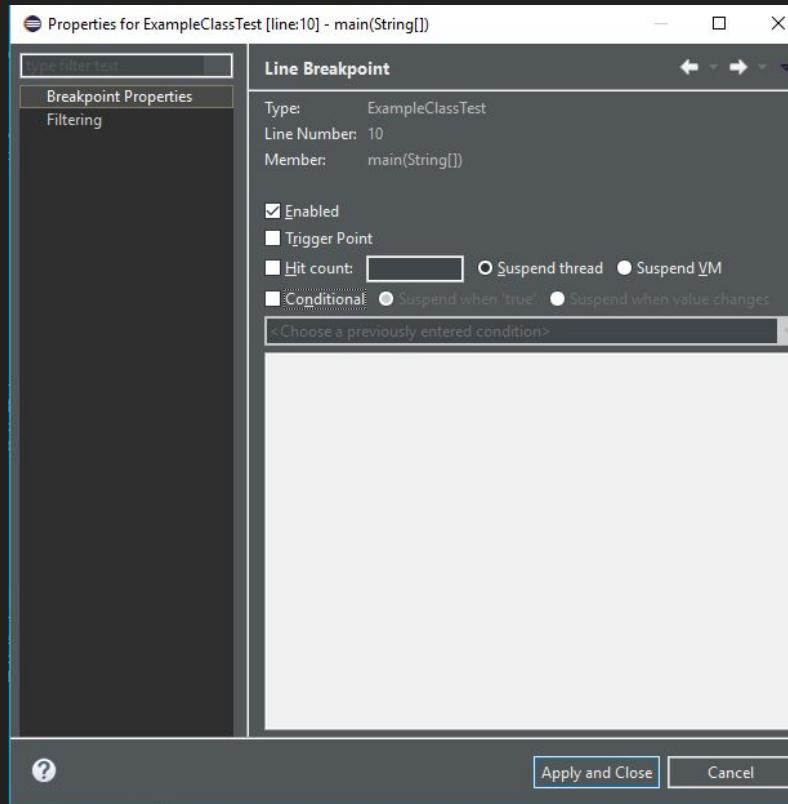
## EJECUTE AND DEBUG

# Shortcuts

- Run: CTRL +F11
- Debug: F11

¡Use breakpoints for debugging!

# Breakpoints Properties



- Hit count
- Conditional

# Debug steps



- RUN-RESUME (F8)
- SUSPEND
- TERMINATE
- STEP INTO
- STEP OVER
- STEP RETURN

Any Questions?