COMPTE RENDU LAB1

SUR TERMINAL

On installe le *JDK* et on vérifie que tous les outils sont bien installé.

```
romain@romain-X751LJ:~/Bureau/Elec4/Java/td1$ java -version
openjdk version "11.0.6" 2020-01-14
OpenJDK Runtime Environment (build 11.0.6+10-post-Ubuntu-1ubuntu119.10.1)
OpenJDK 64-Bit Server VM (build 11.0.6+10-post-Ubuntu-1ubuntu119.10.1, mixed mod
e, sharing)
romain@romain-X751LJ:~/Bureau/Elec4/Java/td1$ javac -version
javac 1.8.0_241
romain@romain-X751LJ:~/Bureau/Elec4/Java/td1$
```

On peut voir que **java** est installé sous sa version **11.0.6** et **javac** sous sa version **1.8.0_241**.

On écrit ensuite le code suivant.

HelloWorldDirect.java

```java
/*
*   Author: Romain Cocogne
*   Comments: Hello World for Direct Simple Compilation
*
*/

//  Class calling main
//  Print the "Hello World!" message on standard stream
public class HelloWorldDirect {

    //main method
    public static void main(String[] args) {
        System.out.println("Hello World!");

    }

}
```

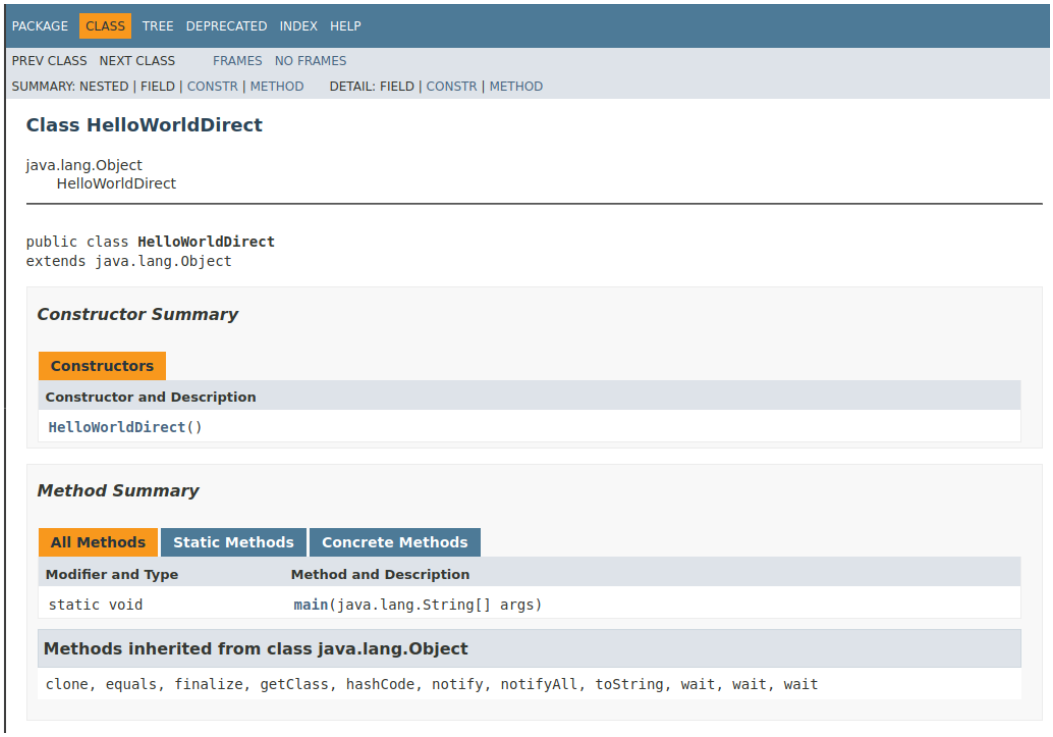
On compile avec **javac** et on obtient le fichier **HelloWorldDirect.class**, qui est le fichier exécutable par la commande **java**.

On peut ensuite lancer l'application **HelloWorldDirect**.

```
romain@romain-X751LJ:~/Bureau/Elec4/Java/td1$ javac HelloWorldDirect.java
romain@romain-X751LJ:~/Bureau/Elec4/Java/td1$ java HelloWorldDirect
Hello World!
romain@romain-X751LJ:~/Bureau/Elec4/Java/td1$
```

On peut maintenant générer la documentation avec la commande **javadoc** et ouvrir le fichier généré **index.html**.

```
romain@romain-X751LJ:~/Bureau/Elec4/Java/td1$ javadoc HelloWorldDirect.java
Loading source file HelloWorldDirect.java...
Constructing Javadoc information...
Standard Doclet version 1.8.0_241
Building tree for all the packages and classes...
Generating ./HelloWorldDirect.html...
Generating ./package-frame.html...
Generating ./package-summary.html...
Generating ./package-tree.html...
Generating ./constant-values.html...
Building index for all the packages and classes...
Generating ./overview-tree.html...
Generating ./index-all.html...
Generating ./deprecated-list.html...
Building index for all classes...
Generating ./allclasses-frame.html...
Generating ./allclasses-noframe.html...
Generating ./index.html...
Generating ./help-doc.html...
```



PACKAGE CLASS TREE DEPRECATED INDEX HELP

PREV CLASS NEXT CLASS FRAMES NO FRAMES

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

Class HelloWorldDirect

java.lang.Object
    HelloWorldDirect

public class **HelloWorldDirect**
extends java.lang.Object

Constructor Summary

Constructors

Constructor and Description
HelloWorldDirect()

Method Summary

All Methods **Static Methods** **Concrete Methods**

Modifier and Type	Method and Description
static void	main(java.lang.String[] args)

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

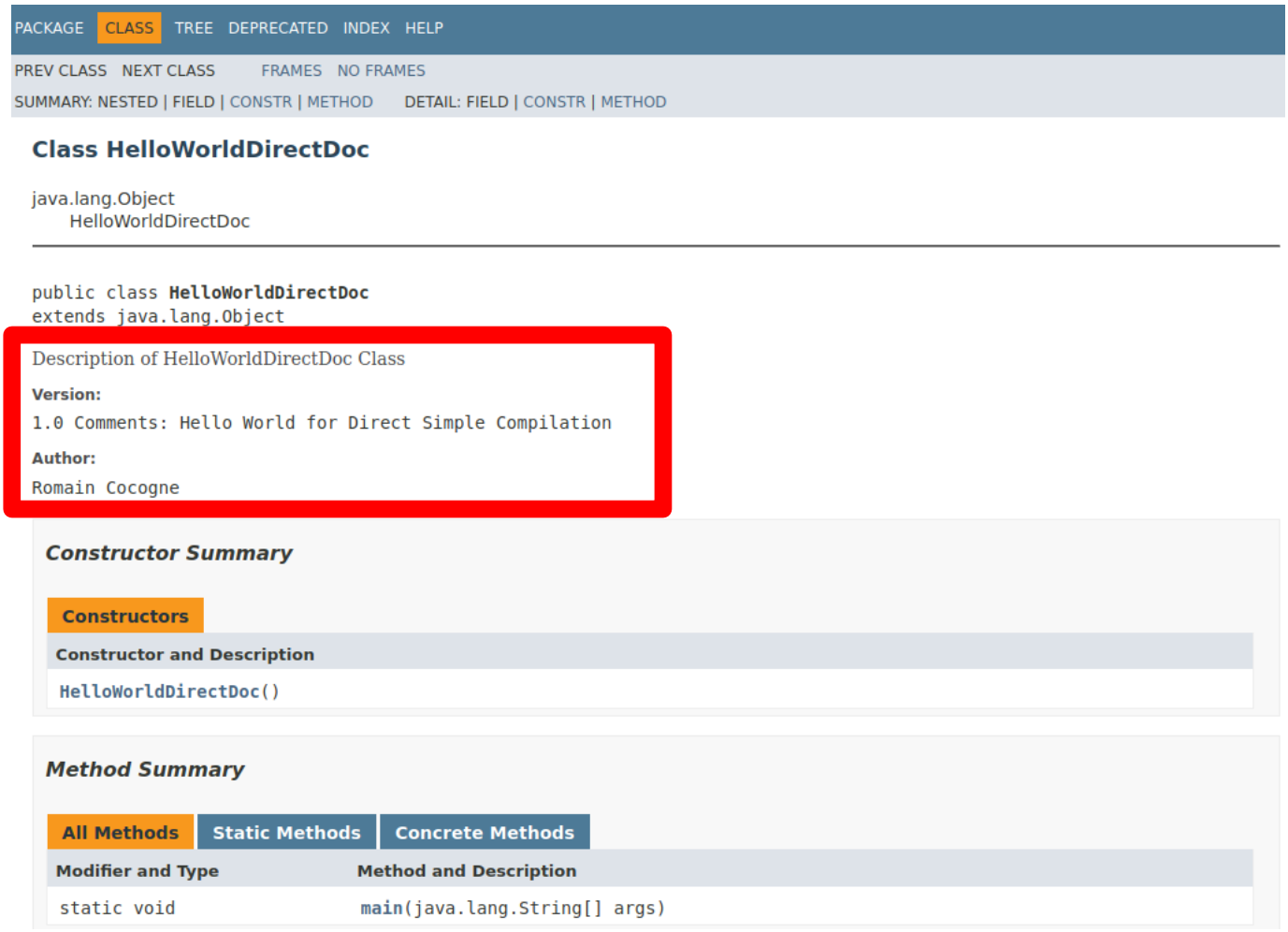
On va maintenant créer le fichier HelloWorldDirectDoc.java.

HelloWorldDirectDoc.java

```java
/** Description of HelloWorldDirectDoc Class
*   @author Romain Cocogne
*   @version 1.0
*   Comments: Hello World for Direct Simple Compilation
*/
//  Class calling main
//  Print the "Hello World!" message on standard stream
public class HelloWorldDirect {

    //main method
    public static void main(String[] args) {
        System.out.println("Hello World!");

    }

}
```

La différence avec précédemment se trouve dans l'entête du code. Cela nous est utile pour la génération de la documentation. Si on lance la commande suivante, on peut voir la nouvelle forme de la doc.

```
romain@romain-X751LJ:~/Bureau/Elec4/Java/td1$ javadoc -version -author  HelloWorldDirectDoc.java
Loading source file HelloWorldDirectDoc.java...
Constructing Javadoc information...
Standard Doclet version 1.8.0_241
Building tree for all the packages and classes...
Generating ./HelloWorldDirectDoc.html...
Generating ./package-frame.html...
Generating ./package-summary.html...
Generating ./package-tree.html...
Generating ./constant-values.html...
Building index for all the packages and classes...
Generating ./overview-tree.html...
Generating ./index-all.html...
Generating ./deprecated-list.html...
Building index for all classes...
Generating ./allclasses-frame.html...
Generating ./allclasses-noframe.html...
Generating ./index.html...
Generating ./help-doc.html...
```



PACKAGE CLASS TREE DEPRECATED INDEX HELP

PREV CLASS NEXT CLASS FRAMES NO FRAMES

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

Class HelloWorldDirectDoc

java.lang.Object
HelloWorldDirectDoc

public class **HelloWorldDirectDoc**
extends java.lang.Object

Description of HelloWorldDirectDoc Class

Version:

1.0 Comments: Hello World for Direct Simple Compilation

Author:

Romain Cocogne

Constructor Summary

Constructors	
Constructor and Description	
HelloWorldDirectDoc()	

Method Summary

All Methods	Static Methods	Concrete Methods
Modifier and Type	**Method and Description**	
static void	main(java.lang.String[] args)	

On remarque que la version capturée comprend aussi le commentaire. Il faut donc faire attention : les tags ne prennent pas en compte le retour à la ligne.

On peut voir les options de **javadoc** disponibles avec **javadoc -help**.

On veut maintenant lire le *bytecode* java de notre programme. Il s'agit du code qui va être lu par la *JVM*. On utilise **javap** pour désassembler le *bytecode* et le lire. On peut voir dans le main, les différentes commandes qui sont appelées.

Par exemple ici on peut remonter les appels qui permettent d'afficher « Hello World ! » :

```
3: ldc           #3                  // String Hello World!
```

```
#3 = String             #18            // Hello World!
```

```
#18 = Utf8              Hello World!
```

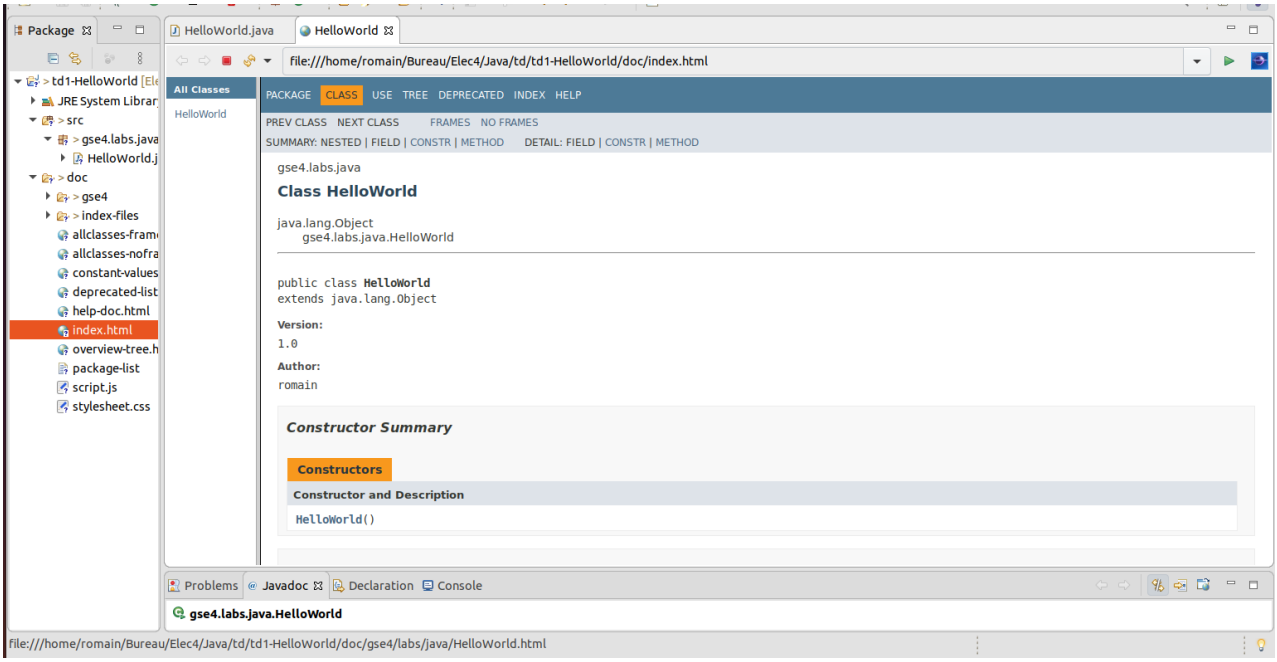
Ici on a le *bytecode* complet.

```
romain@romain-X751LJ:~/Bureau/Elec4/Java/td1$ javap -v HelloWorldDirectDoc.class
Classfile /home/romain/Bureau/Elec4/Java/td1/HelloWorldDirectDoc.class
  Last modified 7 avr. 2020; size 444 bytes
  MD5 checksum 6dedf17357640f2e7b8211055c32f126
  Compiled from "HelloWorldDirectDoc.java"
public class HelloWorldDirectDoc
  minor version: 0
  major version: 52
  flags: ACC_PUBLIC, ACC_SUPER
Constant pool:
   #1 = Methodref          #6.#15         // java/lang/Object."<init>":()V
   #2 = Fieldref           #16.#17        // java/lang/System.out:Ljava/io/PrintStream;
   #3 = String             #18            // Hello World!
   #4 = Methodref          #19.#20        // java/io/PrintStream.println:(Ljava/lang/String;)V
   #5 = Class              #21            // HelloWorldDirectDoc
   #6 = Class              #22            // java/lang/Object
   #7 = Utf8               <init>
   #8 = Utf8               ()V
   #9 = Utf8               Code
  #10 = Utf8               LineNumberTable
  #11 = Utf8               main
  #12 = Utf8               ([Ljava/lang/String;)V
  #13 = Utf8               SourceFile
  #14 = Utf8               HelloWorldDirectDoc.java
  #15 = NameAndType        #7:#8          // "<init>":()V
  #16 = Class              #23            // java/lang/System
  #17 = NameAndType        #24:#25        // out:Ljava/io/PrintStream;
  #18 = Utf8               Hello World!
  #19 = Class              #26            // java/io/PrintStream
  #20 = NameAndType        #27:#28        // println:(Ljava/lang/String;)V
  #21 = Utf8               HelloWorldDirectDoc
  #22 = Utf8               java/lang/Object
  #23 = Utf8               java/lang/System
  #24 = Utf8               out
  #25 = Utf8               Ljava/io/PrintStream;
  #26 = Utf8               java/io/PrintStream
  #27 = Utf8               println
  #28 = Utf8               (Ljava/lang/String;)V
{
  public HelloWorldDirectDoc();
    descriptor: ()V
    flags: ACC_PUBLIC
    Code:
      stack=1, locals=1, args_size=1
         0: aload_0
         1: invokespecial #1                  // Method java/lang/Object."<init>":()V
         4: return
      LineNumberTable:
        line 1: 0

  public static void main(java.lang.String[]);
    descriptor: ([Ljava/lang/String;)V
    flags: ACC_PUBLIC, ACC_STATIC
    Code:
      stack=2, locals=1, args_size=1
         0: getstatic     #2                  // Field java/lang/System.out:Ljava/io/PrintStream;
         3: ldc           #3                  // String Hello World!
         5: invokevirtual #4                  // Method java/io/PrintStream.println:(Ljava/lang/String;)V
         8: return
      LineNumberTable:
        line 4: 0
        line 6: 8
}
SourceFile: "HelloWorldDirectDoc.java"
```

SUR ECLIPSE

On suit les instructions du sujet pour créer le projet puis la classe **HelloWorld** sur *Eclipse*.

On génère maintenant la documentation. On remarque que les tags se sont automatiquement ajoutés. Le nom du package est différent que précédemment vu qu'ici nous l'avons spécifié lors de la création de la classe.



CONCLUSION

Nous pouvons développer en java avec un éditeur de texte lambda et les outils de *JDK*, mais l'*IDE Eclipse* nous propose des outils d'automatisation qui vont améliorer notre *workflow*.