

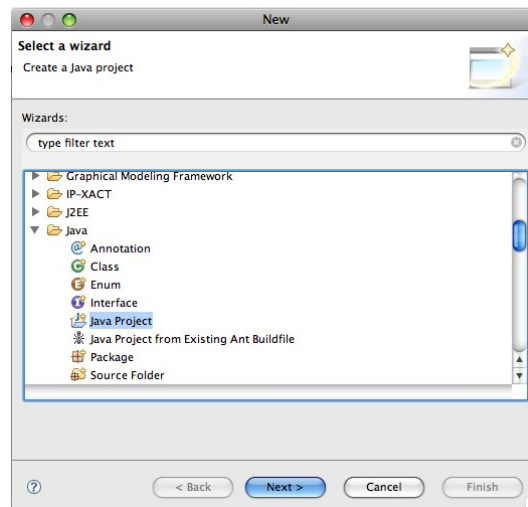
Hello World!

Es ist nicht unüblich, in einer neu zu lernenden Programmiersprache als erstes ein Progrämmchen zu schreiben, das den Text „Hello World!“ auf dem Bildschirm ausgibt. Das ist natürlich nicht weltbewegend, zeigt aber schon die grundsätzliche Vorgehensweise in der betreffenden Programmiersprache auf.

Ein neues Programm landet in Eclipse in einem neuen Projekt: Mit der folgenden Schaltfläche – ganz links oben – kann ein neues Projekt begonnen werden:

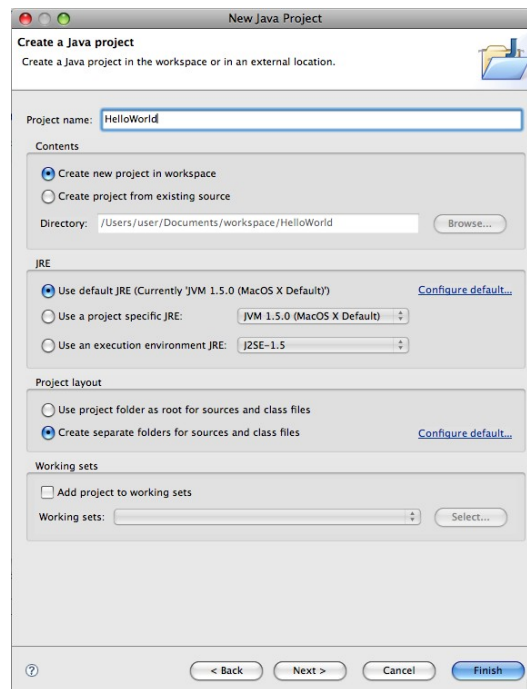


Es erscheint das folgende Fenster:



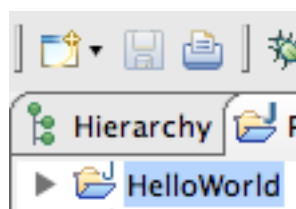
Hello World!

Dort wird wie oben gezeigt „Java Project“ ausgewählt, mit „Next >“ geht es weiter:



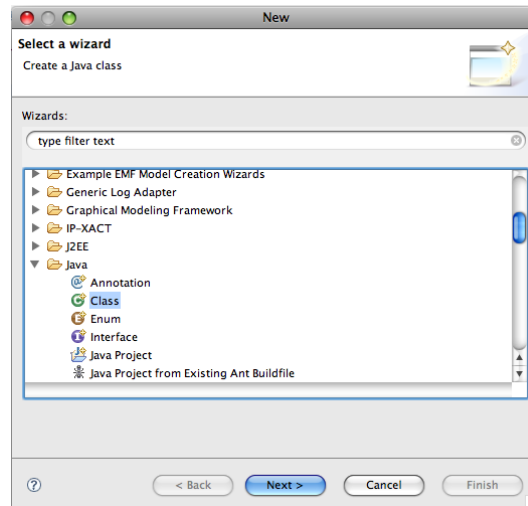
Hier muss nur bei „Project name“ der Name des Projektes eingetragen werden, in diesem Falle „HelloWorld“. Man sollte hier auf die Verwendung von Leerzeichen und deutschen Sonderzeichen verzichten. Weitere Angaben sind nicht erforderlich, mit „Finish“ kann das Projekt nun angelegt werden.

In objektorientierten Programmiersprachen spielt sich alles in Klassen ab – wir benötigen also in unserem Projekt nun eine Klasse:

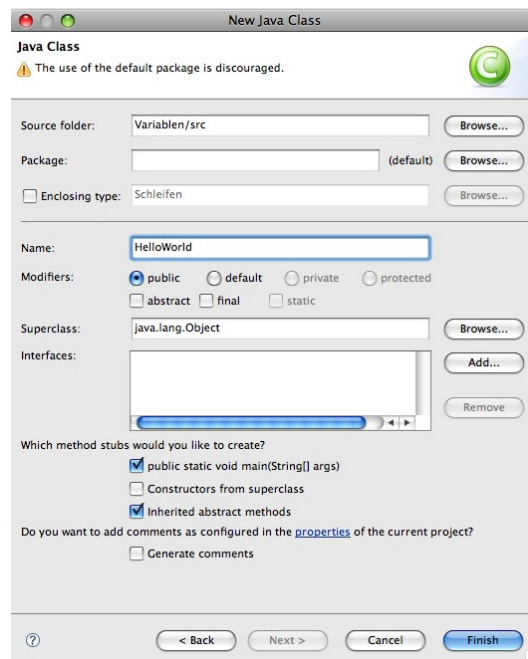


Zunächst markieren wir das Projekt, es ist dann blau unterlegt – dann landet die Klasse, die wir gleich anlegen werden, in diesem Projekt. Mit der schon bekannten Schaltfläche ganz links oben legen wir nun die gewünschte Klasse an:

Hello World!



Wir wählen hier „Class“ aus, mit „Next >“ geht es weiter:



Zwei Angaben sind in obigem Fenster zu machen: Bei „Name“ muss der Name der Klasse angegeben werden, in diesem Falle „HelloWorld“; auch hier sollte wieder auf Leerzeichen und deutsche Sonderzeichen verzichtet werden.

Die eigentliche Programmfunktionalität liegt bei objektorientierten Programmiersprachen in Methoden. Die Programmausführung beginnt in Java bei einer Methode mit dem Namen „main“ - diese muss es daher in jedem Java-Programm genau einmal geben. Wenn wir ein Häkchen bei „public static void main“ setzen, dann legt Eclipse diese Methode gleich an.

Mit „Finish“ geht es weiter:

Hello World!

```
1 |
2 | public class HelloWorld
3 | {
4 |
5 |     /**
6 |      * @param args
7 |      */
8 |     public static void main(String[] args)
9 |     {
10 |         // TODO Auto-generated method stub
11 |
12 |     }
13 |
14 | }
15 |
```

Wir sehen nun die gerade frisch angelegte Klasse und darin die main-Methode, die natürlich noch leer ist. In Java werden Programmanweisungen durch Einschließen in geschweifte Klammern zu einem Block zusammengefasst. Warum die Methode main genau so aussehen muss, sehen wir zu einem späteren Zeitpunkt.

Wir müssen dort nur eine einzige Zeile ergänzen, um „Hello World!“ ausgeben zu lassen – dann sieht das Programm so aus:

```
1 |
2 | public class HelloWorld
3 | {
4 |
5 |     /**
6 |      * @param args
7 |      */
8 |     public static void main(String[] args)
9 |     {
10 |         // TODO Auto-generated method stub
11 |         System.out.print("Hello World!");
12 |     }
13 |
14 | }
15 |
```

System.out.print ist eine in Java eingebaute Methode, die das, was sie als Argument bekommt, auf der Konsole ausgibt. Beim Argument handelt es sich um eine Zeichenkette, die der Computer weder verstehen kann noch auswerten soll, daher wird diese Zeichenkette in doppelte Anführungszeichen eingeschlossen. Die Befehlszeile wird mit einem Strichpunkt abgeschlossen.

Das Programm starten wir durch Klick auf die folgende Schaltfläche:



Im Konsolenfenster unten erscheint dann:

```
Console Problems
<terminated> HelloWorld [Java Appl
Hello World!
```