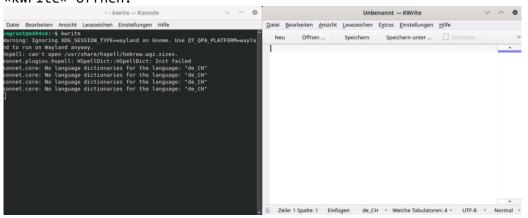
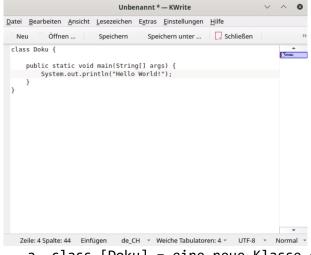
Lernjournal M404

Einfaches «Hello World» in Java (Terminal):

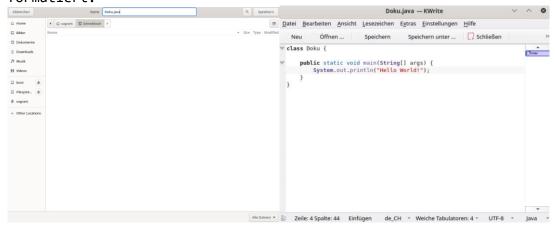
 Im Terminal (Ctrl+Alt+T) ein leeres Textdokument mit dem Befehl «kwrite» öffnen.



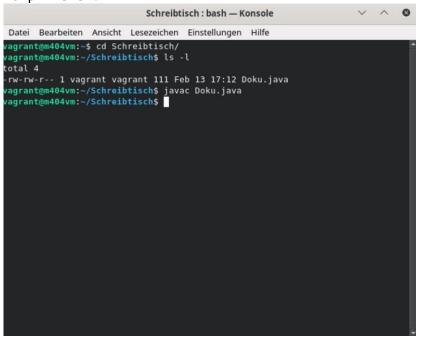
2. Im leeren Textdokument folgende Zeilen schreiben.



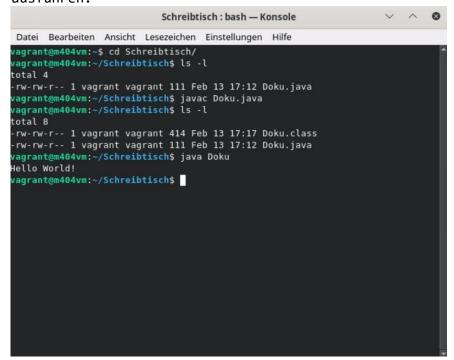
- a. class [Doku] = eine neue Klasse erstellen (muss gleich heissen, wie Java-File)
- b. public static void main(String[] args) = Hier wird das Programm
 geschrieben
- 3. File speichern (Name gleich Klasse). Das File wird danach richtig formatiert.



4. Ein neues Terminal öffnen und in das richtige Verzeichnis gehen. Im richtigen Verzeichnis das Java-File mit «javac [Filename]» kompilieren.

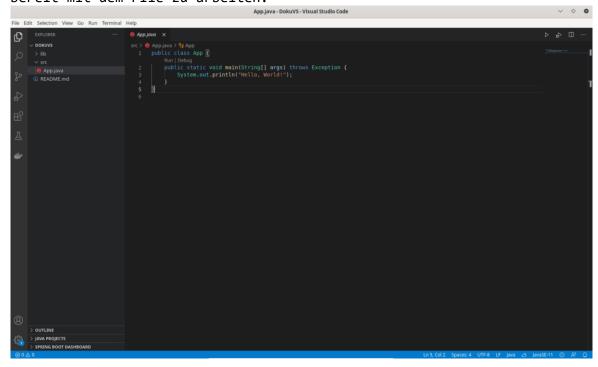


5. Nach dem Kompilieren wird ein neues File erstellt mit dem Namen «[Filename].class». Danach kann man es mit «java [Filename]» ausführen.



Einfaches «Hello World» in Java (VS-Code):

Ein neues VS-Code-File öffnen (Terminal -> «code»). In VS-Code
 Ctrl+Shift+P drücken und «java project» eingeben (Vorschlag anklicken
 -> No build tools). Speicherort wählen und einen Namen geben. Im
 Ordner «src» befindet sich das Java-File (App.java). Im File warten
 bis über «public static ...» «Run | Debug» steht. Danach ist VS-Code
 bereit mit dem File zu arbeiten.



 Danach im Terminal ins richtige Verzeichnis gehen und das File kompilieren und ausführen (siehe Schritt 4 in «Einfaches «Hello World» in Java (Terminal)»).

```
Datei Bearbeiten Ansicht Lesezeichen Einstellungen Hilfe

vagrant@m404vm:~$ code

vagrant@m404vm:~$ cd Schreibtisch/DokuVS/src/
vagrant@m404vm:~/Schreibtisch/DokuVS/src$ ls -l

total 4
-rw-r--r- l vagrant vagrant 134 Feb 13 17:27 App.java
vagrant@m404vm:~/Schreibtisch/DokuVS/src$ javac App.java
vagrant@m404vm:~/Schreibtisch/DokuVS/src$ ls -l

total 8
-rw-rv-r-- l vagrant vagrant 461 Feb 13 17:41 App.class
-rw-r---- l vagrant vagrant 134 Feb 13 17:27 App.java
vagrant@m404vm:~/Schreibtisch/DokuVS/src$ java App
Hello, World!
vagrant@m404vm:~/Schreibtisch/DokuVS/src$

**Total Research

**Total Research
```

JavaFX:

- Neues Java Projekt (Ctrl+Shift+P)
- 2. «Maven» auswählen und auf «More» klicken

```
Select the project type

No build tools

Work with source code directly without any build tools

Maven create from archetype
Provided by  Maven for Java

Spring Boot
Provided by Spring Initializr Java Support

Quarkus
Provided by Quarkus

MicroProfile
Provided by MicroProfile Starter
```

3. Danach «JavaFX» eingeben und ein Packet von «org.» auswählen

```
igavafx

client-archetype-javafx com.gluonhq
graniteds-tide-javafx-spring-jpa-hibernate org.graniteds.archetypes
Base project with JavaFX 2.2, Spring 3.1 and Hibernate 3.6 using GraniteDS with the Tide API.
griffon-javafx-groovy-archetype org.codehaus.griffon.maven
Griffon JavaFX Java Archetype
griffon-javafx-java-archetype org.codehaus.griffon.maven
Griffon JavaFX Java Archetype
javafx ca.dsolutions
Basic setup for JavaFX projects with setup for deployment outside of IDE
javafx org.codehaus.mojo.archetypes
Archetype for creating a JavaFX application
javafx-archetype-fxml com.raelity.jfx
javafx-archetype-fxml org.openjfx
javafx-archetype-fxml-netbeans com.raelity.jfx
javafx-archetype-simple com.raelity.jfx
javafx-archetype-simple org.openjfx
javafx-archetype-simple org.openjfx
```

- 4. Version wählen (am besten immer die Neuste)
- 5. Speicherort wählen
- 6. groupId = [DNS rückwärts, z.B. ch.bztf], artifactId = [Projektname],
 3x überspringen

```
Define value for property 'groupId': ch.bztf
Define value for property 'artifactId': doku
Define value for property 'version' 1.0-SNAPSHOT: :
Define value for property 'package' ch.bztf: :
[INFO] Using property: javafx-maven-plugin-version = 0.0.4
[INFO] Using property: javafx-version = 13
Confirm properties configuration:
groupId: ch.bztf
artifactId: doku
version: 1.0-SNAPSHOT
package: ch.bztf
javafx-maven-plugin-version: 0.0.4
javafx-version: 13
Y: :
```

- 7. Ordner öffnen
- 8. Hier auf «Yes» klicken (oder auf Rakete)



9. In der «App.java» die «main»-Methode ausführen

```
## JavaFX App
## public class App extends Application {

private static Scene scene;

@Override
public void start(Stage stage) throws IOException {
    scene = new Scene(loadFXML("primary"), 640, 480);
    stage.setScene(scene);
    stage.show();

}

static void setRoot(String fxml) throws IOException {
    scene.setRoot(loadFXML(fxml));
}

private static Parent loadFXML(String fxml) throws IOException {
    FXMLLoader fxmlLoader = new FXMLLoader(App.class.getResource(fxml + ".fxml"));
    return fxmlLoader.load();
}

Run Debug

public static void main(String[] args) {
    launch();
}

and

large fixed was a fixed fix
```