Java Tutorial

Inhaltsverzeichnis

[Änderungen 2](#_Toc475202211)

[Kapitel 0 – Der Anfang 3](#_Toc475202212)

[0 – Einleitung 3](#_Toc475202213)

[1 – Anmerkungen zu Java 3](#_Toc475202214)

[*1-1 Allgemein Informationen* 3](#_Toc475202215)

[1-2 Vom Quellcode zum ausführbaren Programm 3](#_Toc475202216)

[2 – Download 4](#_Toc475202217)

[Kapitel 1 – Hello-World 4](#_Toc475202218)

[1-1 – Ein neues Projekt kommt 5](#_Toc475202219)

[1-2 – Das Projekt läuft 6](#_Toc475202220)

[1-3 – Was kommt raus? 8](#_Toc475202221)

[Kapitel 2 – Grundlagen für Hello-World 9](#_Toc475202222)

[2-1 Aufbau einer Java-Datei 9](#_Toc475202223)

[Kapitel 3 – Variablen in Java 10](#_Toc475202224)

[3-1 Grundlagen 10](#_Toc475202225)

[3-2 Arbeiten mit Variablen 12](#_Toc475202226)

[3-3 Sondertyp String 13](#_Toc475202227)

[3-4 Verwendung von Variablen 13](#_Toc475202228)

# Änderungen

|  |  |
| --- | --- |
| Datum | Änderungen |
| 11.02.2017 | - Dokument initial erstellt  - Kapitel 0 erstellt  - Kapitel 1 erstellt |
| 12.02.2017 | - Kapitel 0 erweitert  - Kapitel 2 erstellt |
| 18.02.2017 | - Kapitel 0 erweitert  - Kapitel 3 erstellt |

# Kapitel 0 – Der Anfang

## 0 – Einleitung

Ich will dieses Dokument / Buch mit einer kurzen Einleitung anfangen.

Woher kommt die Idee noch eine Java Anleitung zu schreiben? Was ist an dieser Anleitung anders als an allen anderen? Was wird hier anders gemacht, als in den meisten anderen Anleitungen?...

Die Idee zu diesem Dokument und dem Projekt ist bei einem McDonalds besuch gekommen (Big Tasty Bacon im Menü) und dem Wunsch der anderen Person Java zu lernen. Auf meiner Seite bestand schon länger der Wunsch eine eigene Anleitung für Java zu schreiben, da alle die ich bisher gelesen habe nicht wirklich ausreichend sind und keinen roten Faden haben. Ich konnte mich aber nicht dazu motivieren damit anzufangen, da ich bereits Java beherrsche und damit keine Grundlagen Anleitung mehr brauchte.

Nun hatte ich aber einen Grund anzufangen und mein können in diesem Bereich zu erweitern.

## 1 – Anmerkungen zu Java

### 1-1 Allgemein Informationen

Zuerst ist wichtig zu wissen, dass Java eine objektorientierte Programmiersprache ist, d.h. alles in Java ist ein Objekt. Was kannst du dir unter einem Objekt vorstellen?

Das Konzept eines Objekts kannst du dir wie in der realen Welt vorstellen, also etwa ein Stuhl, ein Auto usw.

Welche Objekte es gibt, wird in so genannten Klassen definiert. Eine Klasse wäre dann z.B. das Auto und ein Objekt (oder Instanz der Klasse) wäre dann z.B. Martins Auto.

Zusätzlich Java hat das Ziel eine betriebssystemunabhängige Sprache zu sein und den gleichen Code auf jedem Gerät auszuführen, welches die Java Virtual Machine (JVM) unterstützt.

Dies wird als „Write once, run anywhere“ bezeichnet.

Die JVM stellt eine eigene Ablaufumgebung dar, welche dann betriebssystemspezifische Funktionen bereitstellt wie z.B. Zugriff auf das Dateisystem, Zugriff auf Netzwerkfunktionen oder Darstellung von Fenstern.

### 1-2 Vom Quellcode zum ausführbaren Programm

Alle Dateien die den eigentlichen Quellcode beinhalten haben die Dateiendung .java, diese werden dann mithilfe des Compilers (ein Programm welches aus den für Menschen lesbaren Befehlen die eigentlichen Maschinenbefehle erstellt) zu .class Dateien umgewandelt werden. Diese .class Dateien werden dann von der JVM ausgeführt.

Komplette Programme die dann eigenständig ausgeführt werden können, werden als .jar Datei komprimiert (ähnlich wie bei einer .zip-Datei). Diese .jar Datei entähl dann zuerst die .class Dateien die aus den .java Dateien erstellt wurden. Darüber hinaus können noch weitere Inhalte wie Bilder, Konfigurationen oder allgemeine andere Dateien in die .jar Datei aufgenommen werden.

## 2 – Download

In diesem Kapitel wird kurz erklärt welche Programme wir benötigen und wo diese im Internet zu finden sind.

Die Anzahl der Programme halten wir absichtlich klein, wir wollen relativ einfach arbeiten und dennoch effizient sein. Alle Programme sind auf dem aktuellsten Stand und wir werden uns nicht mit einer Ära beschäftigen die nicht mehr aktuell ist.

|  |  |  |
| --- | --- | --- |
| Programm | Link | Anmerkung |
| JDK (Java Development Kit) | [JDK 1.8](http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html) | Hier bitte die Version ohne Demos nutzen und für das passende Betriebssystem herunterladen.  (Windows Nutzer sollten Windows X64 herunterladen) |
| Netbeans | [Netbeans 8.2](https://netbeans.org/downloads/) | Zuerst muss die passende Plattform eingestellt werden (oben rechts), danach bitte eine der drei Versionen herunterladen:   * Java SE * Java EE * All |

Nach dem Download sollte zuerst das JDK installiert werden und danach Netbeans.

Bei der Installation ist nichts weiter zu beachten, alle Standard Einstellungen sind völlig ausreichend für den Anfang und werden ggf. in den entsprechenden Kapiteln geändert.

# Kapitel 1 – Hello-World

Bevor wir uns mit den langweiligen Dingen beschäftigen, steigen wir einfach direkt ein in die Materie!

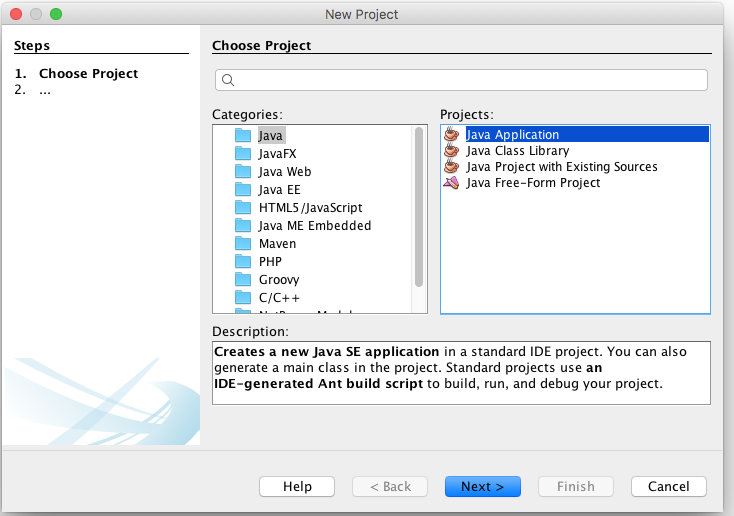
In diesem Kapitel werden wir unser erstes Projekt erstellen, ausführen und das Ergebnis sehen. Also los geht’s!

## 1-1 – Ein neues Projekt kommt

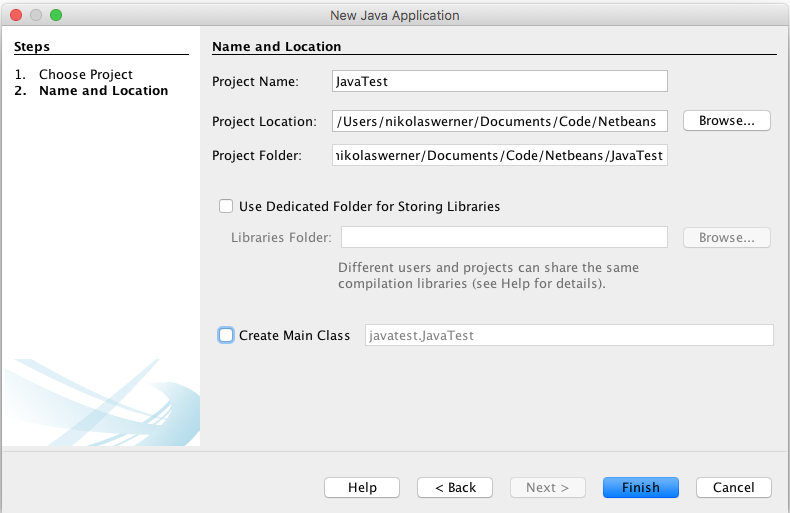
Zuerst wird Netbeans gestartet, die erste Startseite interessiert uns nicht allzu sehr.

Um ein neues Projekt zu erstellen, klicken wir auf File > New Project.

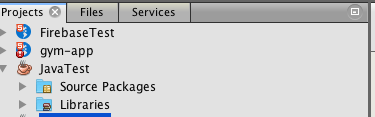
Im neu geöffneten Fenster, wählen wir Java aus und Java Application:



Danach geht es auf Next und der Wunschname des Projekts wird ausgewählt, wir wollen keine Dateien erstellen lassen, daher wählen wir „Create Main Class“ ab



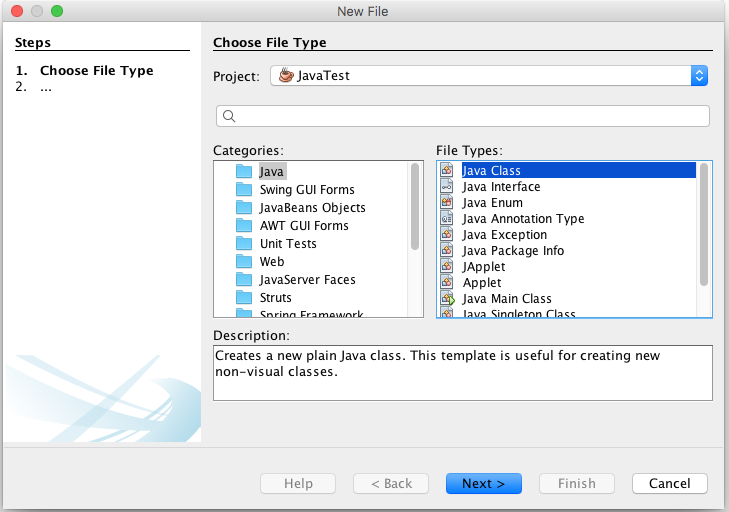
Mit einem Klick auf Finish erstellen wir nun das Projekt. Es sollte nun links in der Projects-Liste erscheinen



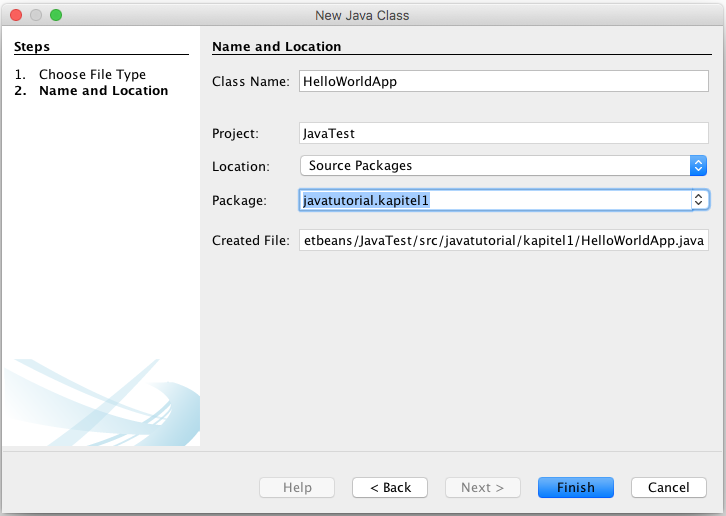
## 1-2 – Das Projekt läuft

Nun wurde das Projekt erstellt, mit dem in diesem und folgenden Kapiteln gearbeitet wird. Als nächstes wird nun die erste Datei erstellt die wir danach ausführen.

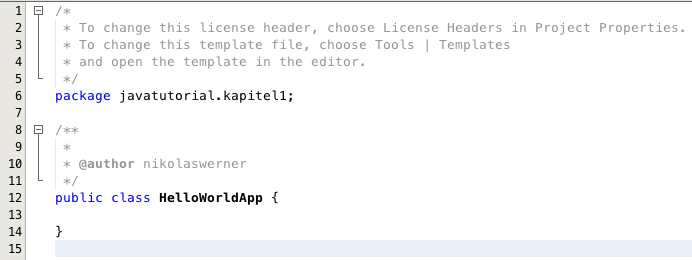
Hierzu klicken wir wieder auf File > New File, in dem neuen Fenster wählen wir das erstellte Projekt aus, Java und Java Class



Mit einem Klick auf Next können wir nun weitere Einstellungen vornehmen, die sollten wie folgt gesetzt werden



Danach bestätigen wir alle Einstellungen und erhalten folgende Datei

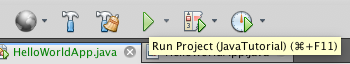


Als nächstes wird die Datei so geändert, so dass diese den folgenden Inhalt hat

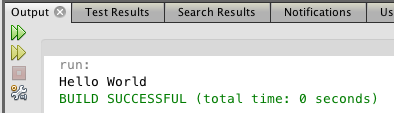


## 1-3 – Was kommt raus?

Um den geschrieben Code nun auszuführen, klicken wir in der Menüleiste von Netbeans auf den grünen Pfeil



Danach wird das geschriebene Programm ausgeführt und die Ausgabe erscheint unten im Netbeans Fenster im Reiter Output



Somit haben wir nun das erste lauffähige Java Programm geschrieben, es macht nichts mehr, als den Text „Hello World“ auszugeben.

Im nächsten Kapitel schauen wir uns das Programm genauer an und sehen, dass das Programm zwar simpel ist, aber bereits alles Wichtige beinhaltet, was benötigt wird um in Java sinnvoll zu starten.

# Kapitel 2 – Grundlagen für Hello-World

## 2-1 Aufbau einer Java-Datei

Nachdem unser erstes Java Programm bereits lauffähig ist und die Ausgabe sichtbar ist, schauen wir uns nun genauer an, welche Bestandteile eine Java-Datei hat.

Wir rufen uns zuerst einmal den Quellcode in Erinnerung:



Gehen wir also die Datei Zeile für Zeile durch und schauen uns an, was genau diese tun.



In dieser Zeile wird das package definiert, das package stellt so etwas wie den eindeutigen Bezeichner über alle Java Anwendungen dar.

Typischerweise wird das package mit einer umgedrehten URL angegeben, etwa:

com.nikolaswerner.javatutorial.kapitel1

Dies würde auf die URL nikolaswerner.com deuten und auf das Projekt „javatutorial“.

Das package ist keine Pflichtangabe, gehört aber zu jeder guten Datei dazu und sollte daher schlicht immer genutzt werden. Im weiteren Verlauf wird deutlich, wieso das Package so wichtig ist zur eindeutigen Identifizierung einer Klasse.



In dieser Zeile definieren wir die HelloWorldApp-Klasse welche wir letztendlich ausführen wollen. In jedem Projekt sollte ein Klasse einen eindeutigen Namen haben. Zur eindeutigen Identifizierung einer Klasse wird die package Angabe hinzugezogen, der volle Name unserer neuen Klasse wäre also

javatutorial.kapitel1.HelloWorldApp



Dies ist der eigentliche Code der ausgeführt wird. Wenn eine Klasse wie HelloWorldApp eine solche Methode enthält, dann ist diese grundsätzlich in einer JVM ausführbar.

Jedes Wort hat hier eine Bedeutung, diese ist wie folgt:

|  |  |
| --- | --- |
| Wort | Bedeutung |
| public | Zeichnet aus, dass wir die Methode öffentlich machen, d.h. diese ist von außen zugreifbar und kann von anderen Objekten oder der JVM aufgerufen werden. |
| static | Die Methode wird statisch definiert, hierdurch wird nicht für jedes Objekt die Methode einzeln aufgerufen, sondern für alle Objekte zentral nur eine Methode. |
| void | Die Methode gibt nichts zurück, sie ist nur ein Auftrag. |
| main(String[] args) | Der Name der Methode und ihre Parameter. Der Inhalt der Klammern kann an dieser Stelle ignoriert werden, er ist nur nötig um die Konvention der JVM einzuhalten.  Dieser Inhalt wird in einem späteren Kapitel behandelt. |

Mit diesem Wissen haben wir nun unser erstes Java Programm geschrieben welches komplett lauffähig ist. Im nächsten Kapitel wollen wir dieses Programm nun variable machen, damit wir nicht immer den gleichen Text auf dem Bildschirm sehen.

# Kapitel 3 – Variablen in Java

Wie bereits im nullten Kapitel geschrieben ist alles in Java ein Objekt, welches auf einer Klasse basiert.

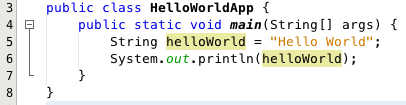
Bisher haben wir nicht sehr viele Klassen oder bereits definierte Objekte kennengelernt und mit noch keinem gearbeitetet, das wollen wir in diesem Kapitel nun ändern

## 3-1 Grundlagen

Eine Variable in Java stellt die Möglichkeit dar, Daten eines bestimmten Typs zu speichern und immer wieder darauf zuzugreifen, egal ob wir den Inhalt auslesen wollen oder schreiben wollen.

Zuerst beschäftigen wir uns mit einer lokalen Variablen, damit wir unseren Text aus dem Programm einfacher anpassen oder erweitern können.

Wir fangen damit an, dass wir die Nachricht die das Programm ausgibt, auslagern in eine eigene Variable:



Wir haben nicht sehr viel im Code geändert, aber die Änderung zeigt uns dennoch sehr viel.

Eine Variable in Java wird immer nach dem folgenden Muster definiert:

(Klasse) {variablenName} = [Inhalt]

Was bedeutet dies konkret für unsere Code-Zeile?

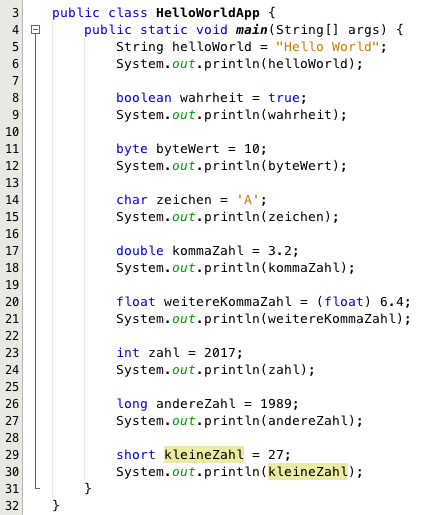
Wir definieren eine Variable vom Typ (Der Klasse) String, dies ist eine allgemeine unlimitierte Ziechenkette, mit dem Namen „helloWorld“, der Inhalt wird auf „Hello World“ gesetzt.

Danach geben wir die Variable einfach in die Ausgabefunktion, nun lassen wir dafür das Programm wieder laufen und sehen, dass die Ausgabe 1:1 die ist, die wir auch vorher hatten.

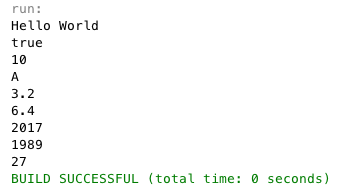
Folgende Typen sind die primitivsten Datentypen:

|  |  |  |
| --- | --- | --- |
| Typ | Beschreibung | Wertebereich |
| boolean | Wahrheitswert (Wahr oder Falsch) | true oder false |
| byte | Ein einzelner Datenwert, selten Benutzt | -128 bis 127 |
| char | Ein einzelnes Zeichen | Entweder eine Zahl: 0-65535  Oder das direkte Zeichen wie ‚A’ |
| double | Gleitkommazahl mit hoher Genauigkeit | +/- 2^64 Ziffern |
| float | Gleitkommazahl mit geringer Genauigkeit | +/- 2^32 Ziffern |
| int | Ganzzahliger numerischer Wert | -2147483648 bis +2147483647 |
| long | Großer ganzzahliger numerischer Wert | +/- 2^64 Ziffern |
| short | Kleiner ganzzahliger numerischer Wert | -128 bis 127 |

Um genau zu sehen wie die Typen in der Praxis aussehen, schauen wir uns diese einmal in der nächsten Stufe unseres Programms an:



Nachdem wir das Programm wieder ausführen, erhalten wir die folgende Ausgabe:



So wie wir die Variablen definiert haben, so bekommen wir auch die Ausgabe.

## 3-2 Arbeiten mit Variablen

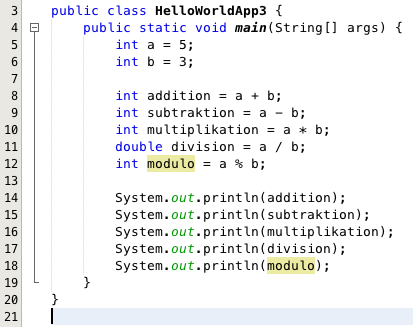
Mit Variablen kann natürlich auch gearbeitet werden, die allgemeinsten Funktionen die wir auf Variablen anwendenden können sind mathematische Funktionen.

Alle Funktionen die wir uns in diesem Kapitel ansehen werden können auf die vorher eingeführten primitiven Datentypen angewandt werden.

Die folgenden Mathematischen Funktionen sind in Java vorhanden:

|  |  |
| --- | --- |
| Funktion (als Zeichen) | Beschreibung |
| + | Addition |
| - | Subtraktion |
| \* | Multiplikation |
| / | Division |
| % | Modulo |

Als Beispiel ändern wir nun unser erstes Programm ab und fügen den folgenden Inhalt ein:



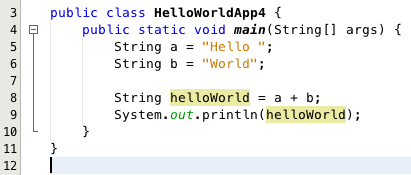
An dieser Stelle ist es nun empfohlen die Typen der Variablen auf die vorher kennengelernten Typen anzupassen und das Programm mehrmals laufen zu lassen. Somit wird deutlich, was genau mit den ganzen Typen bei den verschiedenen Operationen passiert.  
Sehr genau sollte auch auf die Hinweise von Netbeans geachtet werden, hier wird deutlich, dass nicht jeder Typ frei zuzuweisen ist, sondern bestimmte Konventionen eingehalten werden müssen.

## 3-3 Sondertyp String

Der erste Datentyp den wir kennengelernt haben war „String“. Dieser hat eine gewisse Sondersituation in Java, da es sich hier eigentlich nicht um einen primitiven Typ handelt, allerdings die Klasse überall genutzt wird und sogar im gleichen package definiert wird, wie die primitiven Typen.

Des Weiteren kann auch, obwohl es sich bei einem String um eine Zeichenkette handelt, mit diesen „gerechnet“ werden.

Wie wir einen String definieren haben wir bereits im letzten Kapitel gesehen, im folgenden Programm sehen wir nun, wie wir zwei Strings addieren können.



Die Ausgabe ist nun genau die gleiche wie vorher, d.h. wir sehen wieder „Hello World“ in der Ausgabe.

## 3-4 Verwendung von Variablen

Als letzter Punkt zu Variablen ist natürlich noch wichtig zu wissen, wo genau Variablen eingesetzt werden.

Zuerst einmal werden Variablen, wie wir bereits gesehen haben, genutzt um Werte zu speichern oder zu manipulieren. Diese Werte können an unterschiedlichen Stellen sein.

* Innerhalb einer Methode, wie wir es bisher immer gesehen haben
* Innerhalb einer Klasse, dies sehen wir im nächsten Kapitel

Beide Stellen sind sehr relevant und werden quasi in jedem Programm genutzt.

Des Weiteren können Variablen als Parameter für Methoden genutzt werden (was eine Methode ist, lernen wir im nächsten Kapitel). Dies haben wir bereits im ersten Kapitel gesehen und auch in allen weiteren in unserem Programm.

Ein Beispiel für einen Parameter ist die Zeichenfolge „String[] args“ nach dem Namen der main-Methode.