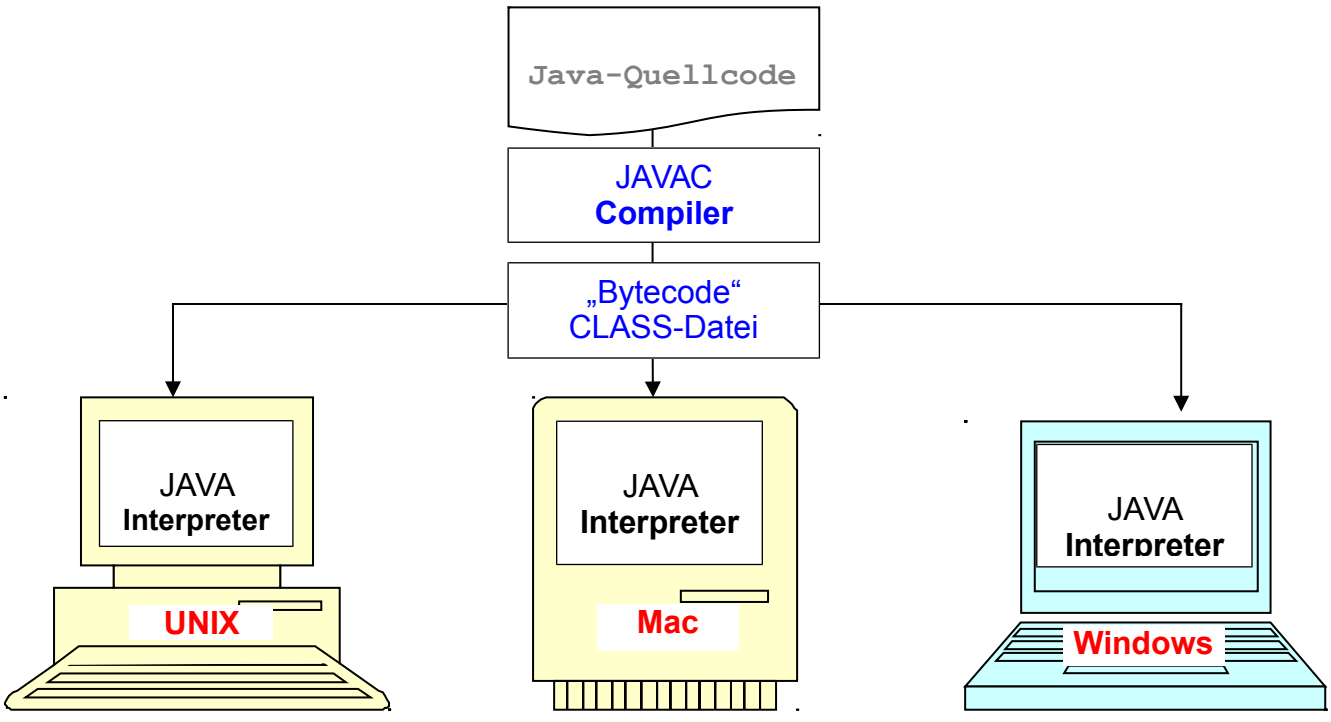
## Prinzipien der Programmerstellung mit Java

Java gehört zu den Programmiersprachen, die als plattformunabhängige Framework-Sprachen bezeichnet werden. Die Erstellung und Ausführung eines Java-Programms verläuft in einem zweistufigen Prozess.

- Quellcode wird vom **Java-Compiler** zu Bytecode kompiliert,
- Der Bytecode wird vom **Java-Interpreter** ausgeführt.

a.) Zuerst wird der Quellcode durch den Java-Compiler in Bytecode (CLASS-Dateien) verwandelt. Dies geschieht in der Regel auf dem Computersystem des Programmierers. Beim Kompilieren wird kein ausführbarer Maschinencode erzeugt, sondern ein Zwischencode (*Bytecode*), der sich wegen seiner byteorganisierten Binärform besonders zur Übertragung in Netzwerk-Umgebungen eignet. *Bytecode* ist ein spezieller Assemblercode, der Befehle für die *Java Virtual Machine (JVM)* enthält und deshalb nur innerhalb einer auf dem Zielrechner installierten Java-Laufzeit-Umgebung ausgeführt werden kann. Der Bytecode enthält die hardware-unabhängige Repräsentation jeder einzelnen Klasse des ursprünglichen Java-Quellcodes.

b.) Im zweiten Schritt ist die *JRE (Java-Runtime-Environment*, Java-Laufzeit-Umgebung, Java-Interpreter) dafür zuständig, den Bytecode auf einem konkreten Computersystem zur Ausführung zu bringen. Für jeden gängigen Prozessortyp bzw. Betriebssystem gibt es eine eigenständige Interpreter-Version (Windows, Mac, Unix...), so dass Java-Bytecode unter gewissen Voraussetzungen auf jedem Computersystem ausgeführt werden kann. Die Java-Laufzeitumgebung enthält neben dem Interpreter auch eine komplette API-Systembibliothek, die alle Standard-Klassen enthält.



Die Trennung von Programmerstellung und Programmausführung entspricht den heutigen Anforderungen bei verteilten Anwendungen im Internet (Client-Server-Prinzip). Der Server ist derjenige, der einen Service (Dienst oder Programm) anbietet, der Client (Kunde) ist derjenige, der einen Service in Anspruch nimmt (oder ein Programm ausführt).

Übungsaufgaben werden nicht zwingend in einer Netzwerkumgebung erstellt, sondern auf einem lokalen Rechnersystem sowohl kompiliert als auch ausgeführt. Der mehrstufige Erstellungsprozess verkompliziert die Programmentwicklung gegenüber Sprachen wie C oder C++. Die Verwendung einer integrierten Entwicklungsumgebung (IDE) wie zum Beispiel Eclipse fasst die nötigen Ausführungsschritte zusammen.

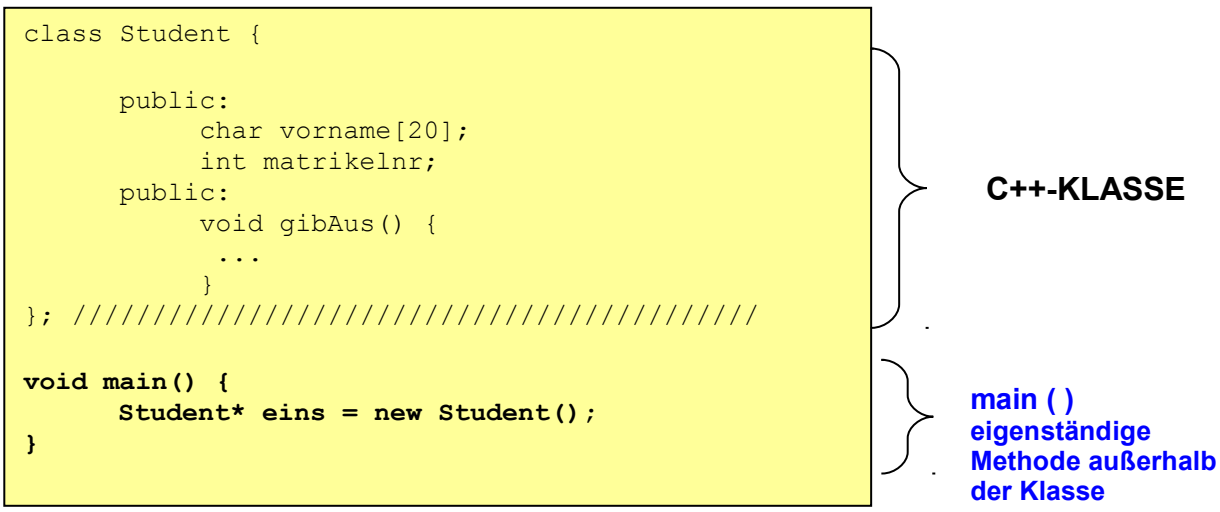
## Aufbau eines Java-Programms

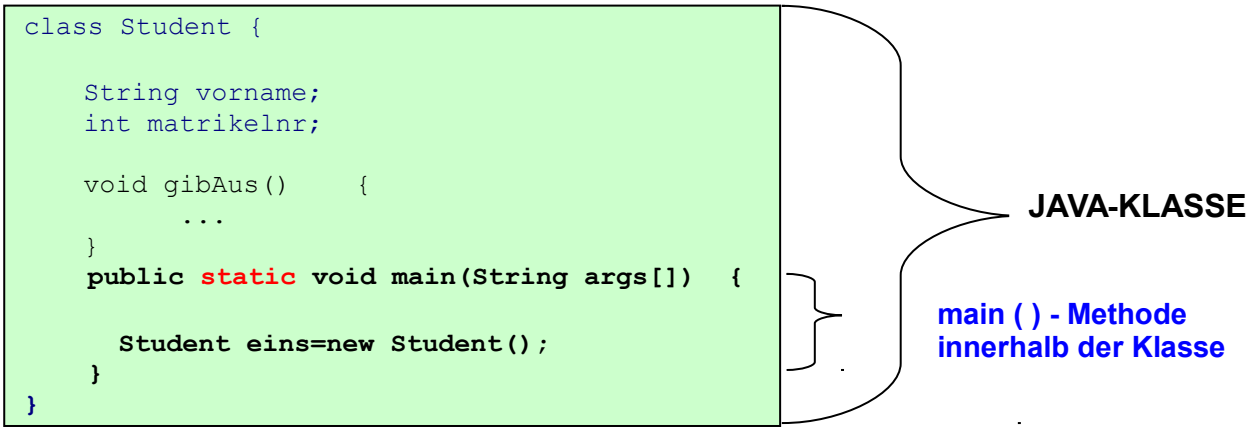
Die strikte Klassenorientiertheit von Java ist schon beim Erstellen des Quellcodes für ein Programm maßgebend. Da die Klassenebene die höchste Hierarchieebene innerhalb eines Programms ist, *müssen alle Programmbestandteile (Anweisungen, Methoden und Definitionen) innerhalb einer Klasse stehen, außerhalb einer Klasse sind nur ganz wenige Anweisungen erlaubt.*

### main

Ebenso wie in C und anderen davon abgeleiteten Programmiersprachen beginnt die Ausführung eines Java-Programms mit der Start-Funktion `main()`, die nach dem genannten Prinzip innerhalb einer Java-Klasse implementiert sein muss. Da die main( )-Funktion zur Ausführung kommen muss, *bevor überhaupt ein Objekt ihrer Klasse erstellt werden konnte*, muss sie als `static` deklariert sein. Dann benötigt sie kein Objekt, um vom Betriebssystem aufgerufen werden zu können.

Auch strikt objektorientierte Sprachen wie Java brauchen prozedurale Funktionen, die an kein Objekt gebunden sind. Static-Methoden innerhalb einer Klasse sind Ersatz für die in Java nicht erlaubten externen (außerhalb einer Klasse stehenden) Funktionen.






Die main-Methode darf prinzipiell in jeder beliebigen Klasse angesiedelt sein. Es darf sogar mehrere main-Methoden geben – allerdings dann in verschiedenen Klassen. Ausgeführt wird immer nur diejenige main-Methode, die sich in der Klasse befindet, die dem Java-Interpreter namentlich zur Ausführung übergeben wird.

Die main-Methode muss eine bestimmte Syntax einhalten. Sie ist `public` (öffentlich), `static` (nicht zu einem Objekt gehörig) und `void` (hat keinen Rückgabewert). Sie kann vom Betriebssystem Parameter übernehmen und muss deshalb in runden Klammern einen (beliebig benannten) String-Vektor deklarieren. Dieser braucht zwar nicht benutzt werden, muss aber immer angegeben werden (genauere Erklärung zu statischen Arrays und deren Attributen / Methoden im Kapitel Arrays).

```
public static void main(String [] args)
```

Signatur der main-Methode

```
// Beispiel-Programm: main mit Parameterübernahme

class ParameterTest {

    public static void main(String[] args) {

        for (int i = 0; i < args.length; i++)

            System.out.println(args[i]);
    }

}
```

Aufruf.

```
java ParameterTest Hallo!
```

Ausgabe:

```
Hallo!
```