Hello World

1. Vorlesung, Javakurs 2010

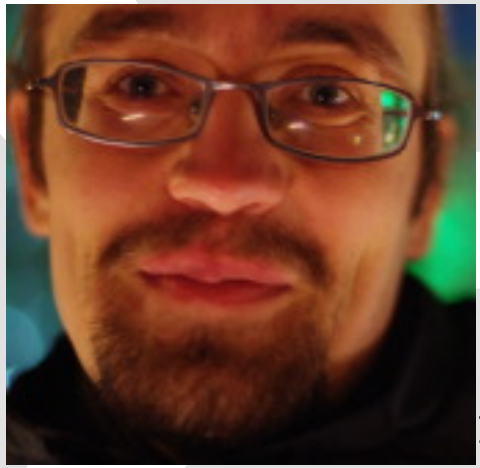
Mario Bodemann Sebastian Dyroff

www.freitagsrunde.org

21. März 2010

Vortragende

Sebastian



Mario [1]

- Studierende wie Ihr, Mitglieder der Freitagsrunde
- Informatik
- Diplom

[1] by Sebastian bzw. Mario

Freitagsrunde

- Studentische Initiative voller Studenten
- Gesamte Fak. IV: ET, TI, Info
- Organisiert Kurse, Kickerturniere, Gremienarbeit, Beamerverleihe, Keysignings, Einführungswochen, Klausurensammlungen, ...
- `www.freitagsrunde.org`
- Freitags, 13 Uhr im FR 5046

Inhaltsverzeichnis

1. Organisatorisches
2. Kompilieren und Ausführen
3. Hello World - das erste Programm
   - Ausführen
4. Abarbeiten von Befehlen
5. Variablen und einfache Typen
6. Operatoren
7. Fallunterscheidungen
8. Kommentare
9. Probleme und Fehlermeldungen
10. Zusammenfassung

Inhaltsverzeichnis

1. Organisatorisches
2. Kompilieren und Ausführen
3. Hello World - das erste Programm
   - Ausführen
4. Abarbeiten von Befehlen
5. Variablen und einfache Typen
6. Operatoren
7. Fallunterscheidungen
8. Kommentare
9. Probleme und Fehlermeldungen
10. Zusammenfassung

	Mo	Di	Mi	Do
10:15	*Hello World*	*Methoden*	*Aufgaben*	*OOP*
11:30	Übung	Übung	Übung	Übung
13:15	Mittag	Mittag	Mittag	Mittag
14:15	*Schleifen*	Übung	*Kapselung*	Übung
15:30	Übung		Übung	

H2032

TEL 106 / TEL 206

[2]

[2] CC BY by Thaddäus

Ziele des Kurses

- Für euch:
  - Javaprogramme schreiben
  - Grundlagen von Java erlernen
  - Probleme im Code finden und lösen
- Für uns:
  - Üben von Vorträgen
  - Lernen zu Sprechen
  - Resonanz zum Vortragsstil
- Zusammen: Viel, viel Spaß haben ...

[3]

[3] by Milan

Inhaltsverzeichnis

1 Organisatorisches

2 Kompilieren und Ausführen

3 Hello World - das erste Programm
- Ausführen

4 Abarbeiten von Befehlen

5 Variablen und einfache Typen

6 Operatoren

7 Fallunterscheidungen

8 Kommentare

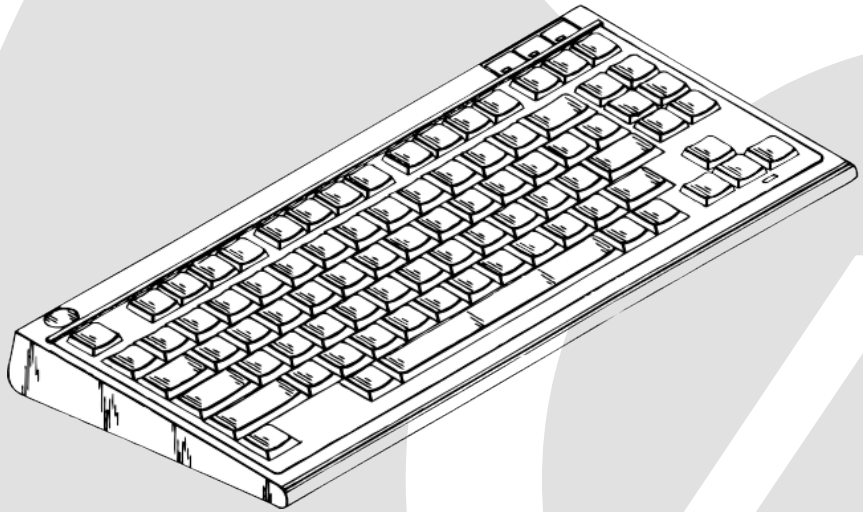
9 Probleme und Fehlermeldungen

10 Zusammenfassung

Arbeitsumgebung

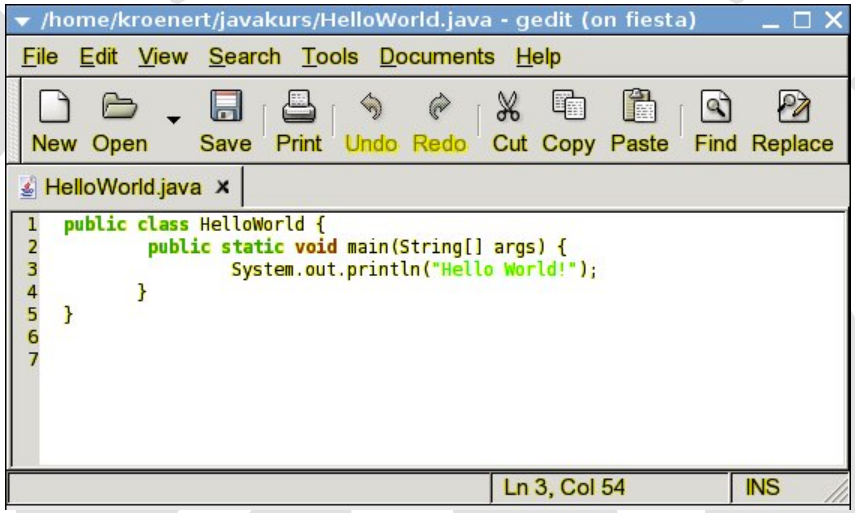
Arbeitsumgebung

- Einloggen



Arbeitsumgebung

- Einloggen
- einen Editor [4] öffnen



z.b. **Gedit** (Gnome) oder **kate** (KDE)

[4] Nein! Nicht Word!

Hello World

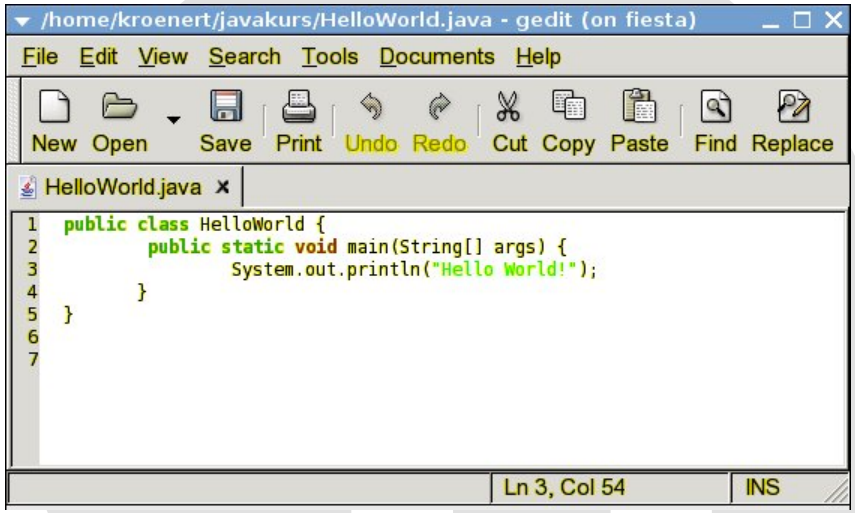
Listing 1: src/HelloWorld.java

```
1 public class HelloWorld {
2
3         public static void main(String[] args) {
4
5                 System.out.println("Hello World!");
6
7         }
8 }
```

Beim starten des Programs wird die “main” Methode ausgeführt

Arbeitsumgebung

- Einloggen
- einen Editor öffnen
- Programm schreiben



Arbeitsumgebung

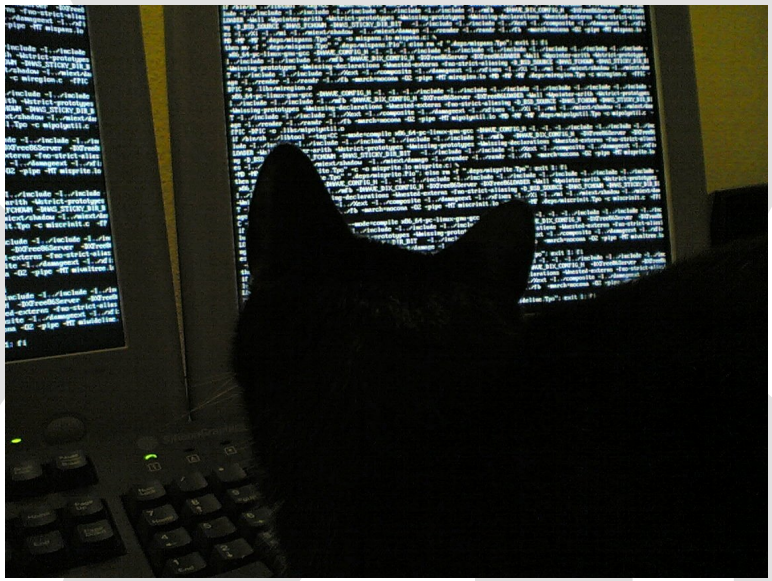
- Einloggen
- einen Editor öffnen
- Programm schreiben
- eine Shell [5] öffnen

[5] Konsole, Terminal, (MS-DOS-)Eingabeaufforderung, Kommandozeile

Kompilieren

Der Compiler übersetzt den Quellcode in ein ausführbares Programm.

javac ist der **Java C**ompiler.



```
$ javac HelloWorld.java
$
```

Komplieren eines Java Programms

Bytecode

```
$ ls -l
-rw------- 1 dyroff 426 Mar 22 10:30 HelloWorld.class
-rw------- 1 dyroff 106 Mar 22 10:29 HelloWorld.java
$
```

Ausgabedatei des Javakompilers

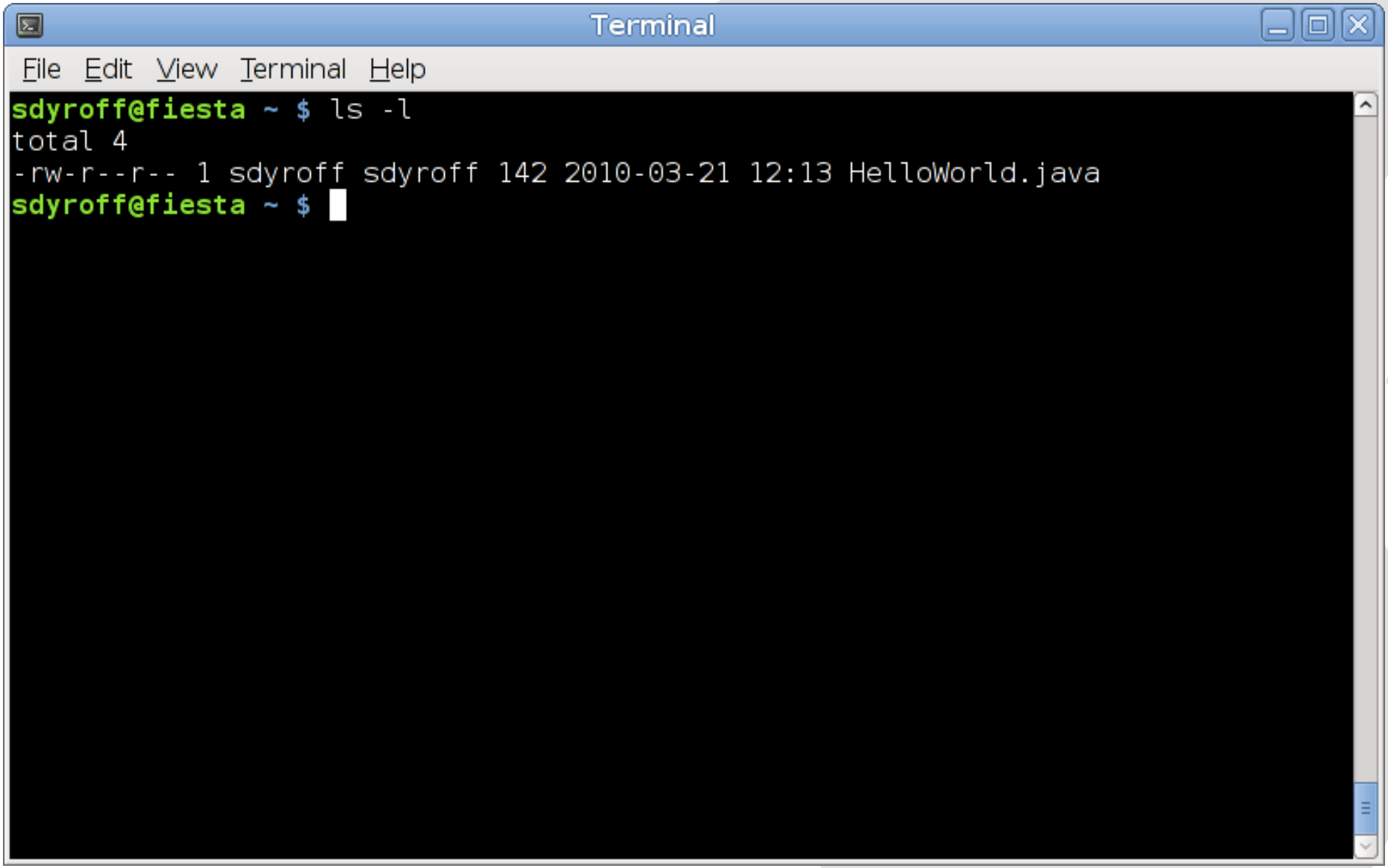
- Kompilieren erzeugt .class Dateien, sog. Bytecode
- Bytecode kann mit einer **Java Virtual Machine** ausgeführt werden
- Bytecode ist maschinen**un**abhängig

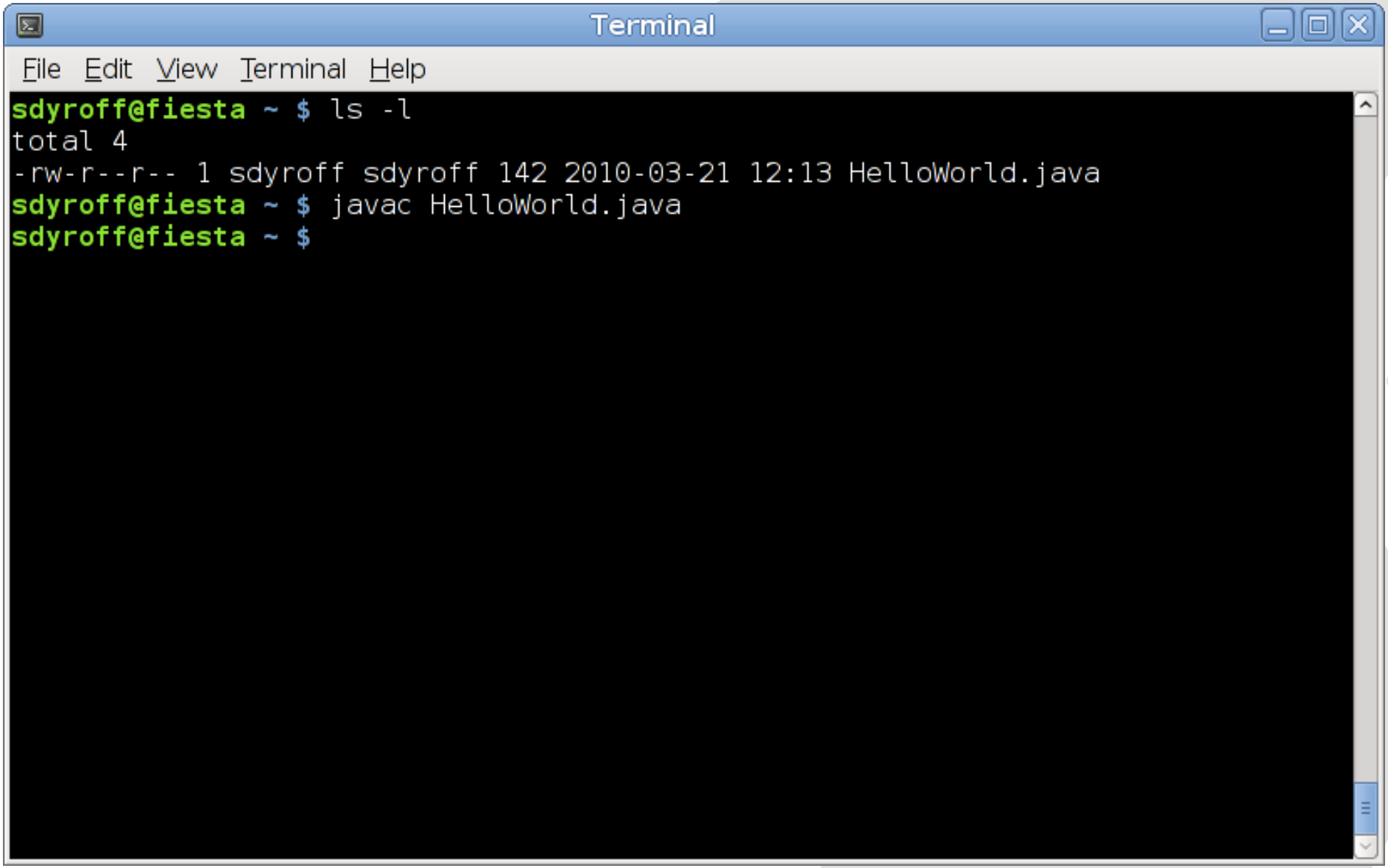
Ausführen

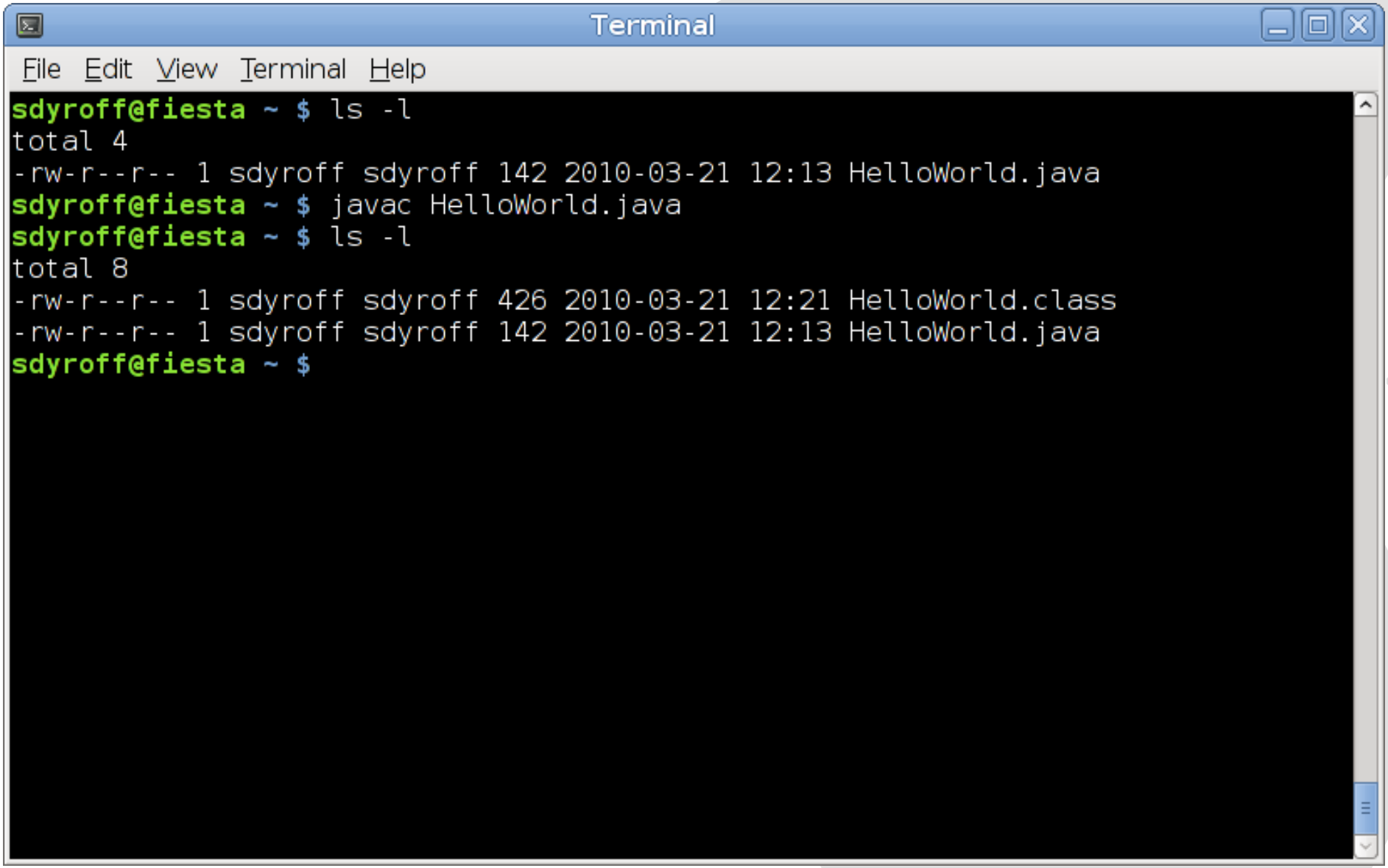
```
$ java HelloWorld
Hello World!
$
```

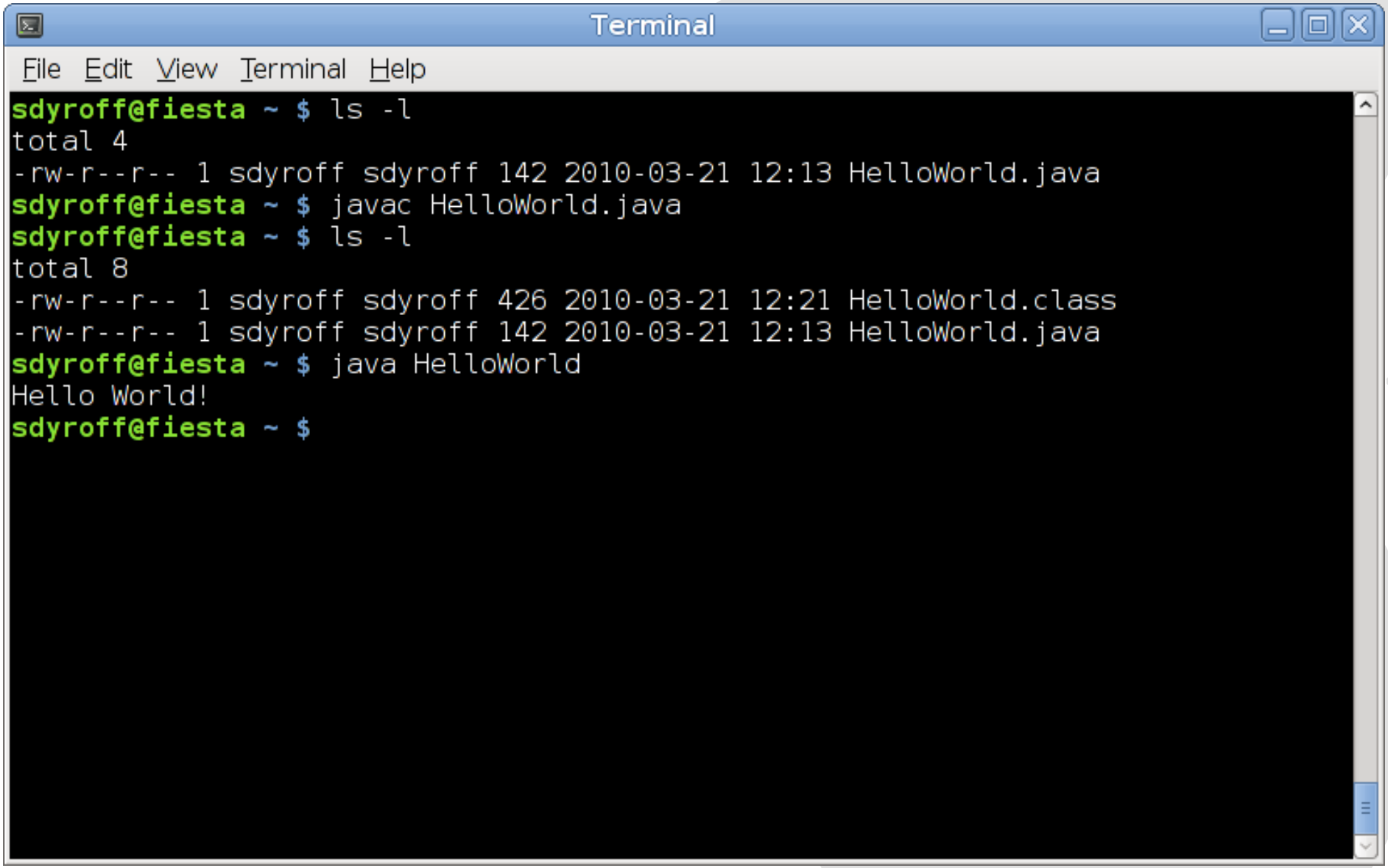
Ausgabe unseres Programms

- **java** ist die Java Virtual Maschine
- als Parameter wird der Klassenname übergeben
- die Ausgabe ist auf der Console zu sehen


Terminal
File Edit View Terminal Help
sdyroff@fiesta ~ $ ls -l
total 4
-rw-r--r-- 1 sdyroff sdyroff 142 2010-03-21 12:13 HelloWorld.java
sdyroff@fiesta ~ $


Terminal
File Edit View Terminal Help
sdyroff@fiesta ~ $ ls -l
total 4
-rw-r--r-- 1 sdyroff sdyroff 142 2010-03-21 12:13 HelloWorld.java
sdyroff@fiesta ~ $ javac HelloWorld.java
sdyroff@fiesta ~ $


Terminal
File Edit View Terminal Help
sdyroff@fiesta ~ $ ls -l
total 4
-rw-r--r-- 1 sdyroff sdyroff 142 2010-03-21 12:13 HelloWorld.java
sdyroff@fiesta ~ $ javac HelloWorld.java
sdyroff@fiesta ~ $ ls -l
total 8
-rw-r--r-- 1 sdyroff sdyroff 426 2010-03-21 12:21 HelloWorld.class
-rw-r--r-- 1 sdyroff sdyroff 142 2010-03-21 12:13 HelloWorld.java
sdyroff@fiesta ~ $


Terminal
File Edit View Terminal Help
sdyroff@fiesta ~ $ ls -l
total 4
-rw-r--r-- 1 sdyroff sdyroff 142 2010-03-21 12:13 HelloWorld.java
sdyroff@fiesta ~ $ javac HelloWorld.java
sdyroff@fiesta ~ $ ls -l
total 8
-rw-r--r-- 1 sdyroff sdyroff 426 2010-03-21 12:21 HelloWorld.class
-rw-r--r-- 1 sdyroff sdyroff 142 2010-03-21 12:13 HelloWorld.java
sdyroff@fiesta ~ $ java HelloWorld
Hello World!
sdyroff@fiesta ~ $

Abarbeiten von Befehlen

Listing 5: src/HelloSolarSystem.java

```
1  public class HelloSolarSystem {
2
3          public static void main(String[] args) {
4
5                  System.out.println("Hello Mercury!");
6                  System.out.println("Hello Venus!");
7                  System.out.println("Hello Earth!");
8                  System.out.println("Hello Mars!");
9                  System.out.println("Hello Jupiter!");
10                 System.out.println("Hello Saturn!");
11                 System.out.println("Hello Uranus!");
12                 System.out.println("Hello Neptune!");
13
14         }
15
16 }
```

Abarbeiten von Befehlen

Listing 6: src/HelloSolarSystem.java

```
1  public class HelloSolarSystem {
2
3          public static void main(String[] args) {
4
5                  System.out.println("Hello Mercury!");
6                  System.out.println("Hello Venus!");
7                  System.out.println("Hello Earth!");
8                  System.out.println("Hello Mars!");
9                  System.out.println("Hello Jupiter!");
10                 System.out.println("Hello Saturn!");
11                 System.out.println("Hello Uranus!");
12                 System.out.println("Hello Neptune!");
13
14         }
15
16 }
```

- Befehle werden der Reihe nach abgearbeitet
- Klassennamen und Dateiname (ohne .java) müssen übereinstimmen

Inhaltsverzeichnis

1 Organisatorisches

2 Kompilieren und Ausführen

3 Hello World - das erste Programm
- Ausführen

4 Abarbeiten von Befehlen

5 Variablen und einfache Typen

6 Operatoren

7 Fallunterscheidungen

8 Kommentare

9 Probleme und Fehlermeldungen

10 Zusammenfassung

Datentypen – Integer

Listing 7: src/Variables.java

```
1  public class Variables {
2          public static void main(String[] args) {
3
4                  // Deklaration einer Variablen
5                  int number;
6                  // Initialisierung einer Variablen
7                  number = 23;
8                  System.out.println(number);
9
10         }
11 }
```

- **int** steht für **Integer**, eine ganze Zahl
- = weist den rechten Wert der Variablen auf der Linken zu

Datentypen – Integer

```
$ javac Variables.java
$ java Variables
23
$
```

Ausgabe Variables

- Kompilieren und Ausführen
- Der Wert der Variablen wird auf die Konsole geschrieben

Datentypen – String

Listing 9: src/VariablesII.java

```
1  public class VariablesII {
2      public static void main(String[] args) {
3
4          int age = 20;
5          int number = 3;
6
7          age = age + number;
8          String message;
9          message = "My age is: ";
10
11
12         System.out.println(message + age);
13
14     }
15
16 }
```

Datentypen – String

```
$ javac VariablesII.java
$ java VariablesII
My age is: 23
$
```

Ausgabe VariablesII

Datentypen – Double

Listing 11: src/VariablesIII.java

```
1 public class VariablesIII {
2     public static void main(String[] args) {
3
4         double height = 1.75;
5         String message = "My height is ";
6         System.out.println(message + height);
7
8     }
9 }
```

- **double** ist eine Fließkommazahl

Datentypen – Double

```
$ javac VariablesIII.java
$ java VariablesIII
My height is 1.75
$
```

Ausgabe VariablesIII

Datentypen – Boolean

Listing 13: src/VariablesIV.java

```
1  public class VariablesIV {
2      public static void main(String[] args) {
3
4          boolean amISmart = true;
5          boolean amIAJavaHacker = false;
6
7          boolean result = amISmart && amIAJavaHacker;
8
9          String message = "Am I a smart Java hacker? ";
10         System.out.println(message + result);
11     }
12 }
```

Datentypen – Boolean

```
$ javac VariablesIV.java
$ java VariablesIV
Am I a smart Java hacker? false
$
```

Ausgabe VariablesIV

Konventionen

- Variablennamen werden im sogenannten camelCase geschrieben
- Der erste Buchstabe ist aber klein.

Zum Beispiel: **mySolarSystem**

Empfehlung:

- kurze und aussagekräftige Namen verwenden

Datentypen im Überblick

Datentypen

```
1 boolean result = false;
2 int age = 20;
3 double height = 1.75;
4 String message = "Javakurs";
```

Typ	Wertebereich
boolean	{true;false}
int	{-2147483648 ... 2147483647}
double	$\{\pm 4,9 \cdot 10^{-324} \ldots \pm 1,7977 \cdot 10^{+308}\}$
String	{ “”... endlich }

Es gibt zwar noch mehr Datentypen, aber dies sind erstmal die wichtigsten.

Inhaltsverzeichnis

1 Organisatorisches

2 Kompilieren und Ausführen

3 Hello World - das erste Programm
- Ausführen

4 Abarbeiten von Befehlen

5 Variablen und einfache Typen

6 Operatoren

7 Fallunterscheidungen

8 Kommentare

9 Probleme und Fehlermeldungen

10 Zusammenfassung

Operatoren

Listing 16: src/Operators.java

```
1  public class Operators {
2      public static void main(String[] args) {
3
4          int a, b;
5          a = 10;
6          b = 2;
7
8          int multi = a * b;
9          int average = (a + b) / 2;
10
11     }
12 }
```

- es gelten die üblichen Rechenregeln

Operatoren

Logische Operatoren

&&	und
||	oder
!	Negation

Arithmetische Operatoren

+	Addition
−	Subtraktion
/	Division
*	Multiplikation
%	Modulo

Inhaltsverzeichnis

1 Organisatorisches

2 Kompilieren und Ausführen

3 Hello World - das erste Programm
- Ausführen

4 Abarbeiten von Befehlen

5 Variablen und einfache Typen

6 Operatoren

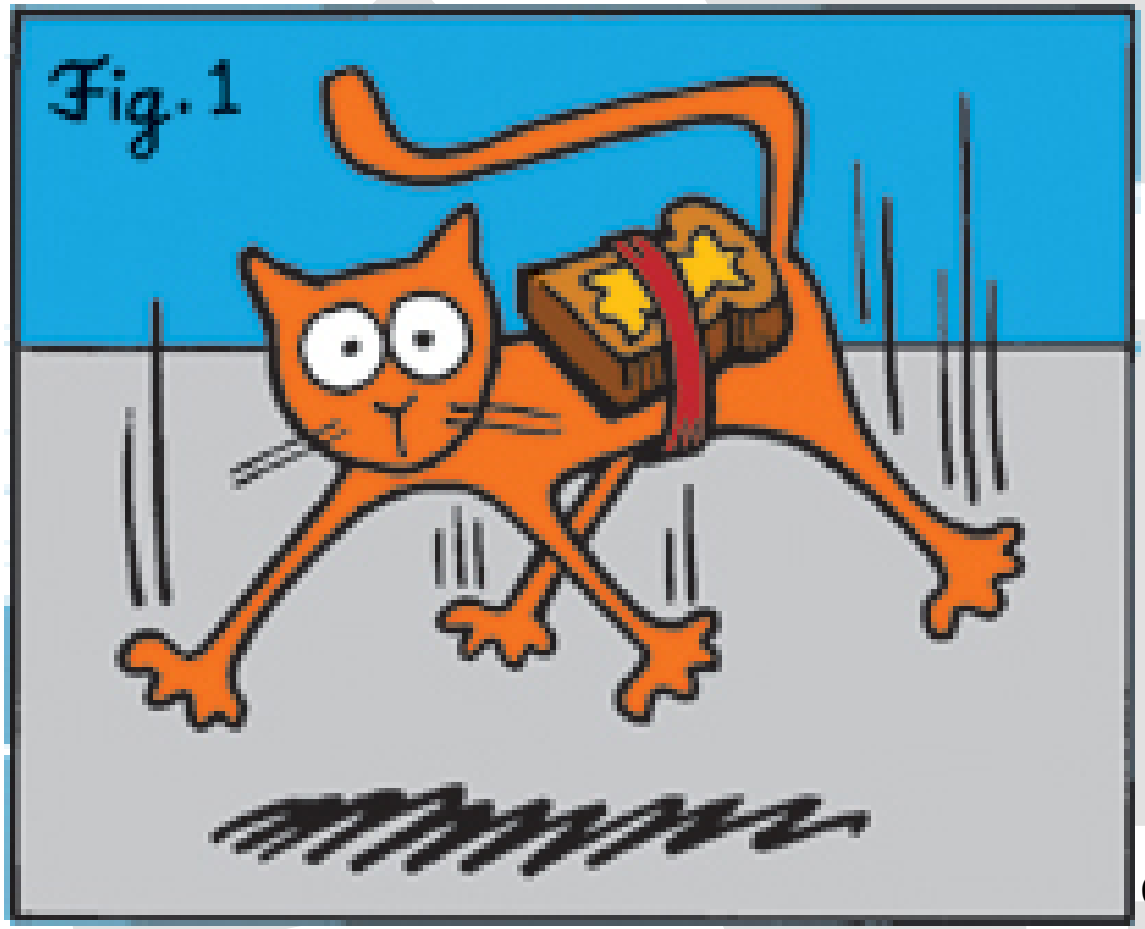
7 Fallunterscheidungen

8 Kommentare

9 Probleme und Fehlermeldungen

10 Zusammenfassung

Fallunterscheidung



6

[6] by Greg Williams

Einfaches Beispiel

Listing 17: src/IfSimple.java

```
4       boolean catDown = true;
5       if( catDown == true ) {
6         System.out.println("Toast wird Katze drehen ...");
7       }
8       if( catDown == false ) {
9         System.out.println("Katze wird Toast drehen ...");
10      }
```

Einfaches Beispiel (mit else)

Listing 18: src/IfSimpleCorrect.java

```
4       boolean catDown = true;
5       if( catDown == true ) {
6         System.out.println("Toast wird Katze drehen ...");
7       } else {
8         System.out.println("Katze wird Toast drehen ...");
9       }
```

Einfaches Beispiel (mit else)

Listing 19: src/IfSimpleCorrect.java

```
4       boolean catDown = true;
5       if( catDown == true ) {
6         System.out.println("Toast wird Katze drehen ...");
7       } else {
8         System.out.println("Katze wird Toast drehen ...");
9       }
```

Ausgabe

```
[mario src]$ java IfSimpleCorrect
Toast wird Katze drehen ...
```

Bedingungen mit logischen Operatoren

Listing 20: src/IfGrades.java

```
4       int points = 77;
5       if( points <= 50 ) {
6         System.out.println("Leider nicht bestanden ...");
7       } else {
8         System.out.println("Bestanden !!");
9       }
```

Bedingungen mit logischen Operatoren

Listing 21: src/IfGrades.java

```
4       int points = 77;
5       if( points <= 50 ) {
6         System.out.println("Leider nicht bestanden ...");
7       } else {
8         System.out.println("Bestanden !!");
9       }
```

Ausgabe

```
[mario src]$ java IfGrades
Bestanden !!
```

Bedingungen kombiniert (mit else if)

Listing 22: src/IfGradesMixedElseIf.java

```
4     boolean testPassed = true;
5     int points = 35;
6
7     if( points > 50 && testPassed ) {
8       System.out.println("Klausur bestanden!");
9     } else if( testPassed ) {
10      System.out.println("Klausur wiederholen");
11    } else {
12      System.out.println("Durchgefallen ...");
13    }
```

Bedingungen kombiniert (mit else if)

Listing 23: src/IfGradesMixedElseIf.java

```
4       boolean testPassed = true;
5       int points = 35;
6
7       if( points > 50 && testPassed ) {
8         System.out.println("Klausur bestanden!");
9       } else if( testPassed ) {
10        System.out.println("Klausur wiederholen");
11      } else {
12        System.out.println("Durchgefallen ...");
13      }
```

Ausgabe

```
[mario src]$ java IfGradesMixedElseIf
Klausur wiederholen
```

Ganzzahlige Bedingungen

Listing 24: src/IfSwitch.java

```
3     int number = 51231;
4
5     System.out.print("Zahl endet auf ");
6     switch( number % 10 ) {
7       case 0: System.out.println("Null");
8         break;
9       case 1: System.out.println("Eins");
10        break;
11      // ...
12      default: System.out.println("Fehler!");
13    }
```

Ganzzahlige Bedingungen

Listing 25: src/IfSwitch.java

```
3     int number = 51231;
4
5     System.out.print("Zahl endet auf ");
6     switch( number % 10 ) {
7       case 0: System.out.println("Null");
8         break;
9       case 1: System.out.println("Eins");
10        break;
11      // ...
12      default: System.out.println("Fehler!");
13    }
```

Ausgabe

```
[mario src]$ java IfSwitch
Zahl endet auf Eins
```

Inhaltsverzeichnis

1 Organisatorisches

2 Kompilieren und Ausführen

3 Hello World - das erste Programm
- Ausführen

4 Abarbeiten von Befehlen

5 Variablen und einfache Typen

6 Operatoren

7 Fallunterscheidungen

8 Kommentare

9 Probleme und Fehlermeldungen

10 Zusammenfassung

Kommentare

- Warum mache ich das Folgende?
- Was gibt es für Randbedingungen?
- Warum sollte ich dies nicht anders lösen?
- Wichtig: Kommentar vor dem Code!
  - Hilft beim Denken
  - Hilft beim Nachvollziehen

Beweis durch Gegenbeispiel

Listing 26: src/CommentCounterProof.java

```
1 public class CommentCounterProof { public static void
    main(String argv[]){ System.out.println("Not a good
    Example"); }}
```

- Kann das jemand lesen?
- Was bewirken die Variablen?
- Wie machen dies andere?

Lesbarer Beweis durch Gegenbeispiel

Listing 27: src/CommentedCounterProof.java

```
1  /*
2    Samplecode for "correct" commenting.
3
4    This code will demonstrate how to use comments,
5    either in a multiline fasion(like this one), or
6    by single line
7   */
8  public class CommentedCounterProof {
9    // main entry point of class
10   public static void main(String argv[]){
11     // Supply our user with some basic
12     // information about this sample.
13     System.out.println("A better Example");
14   }
15 } // done with this sample
```

Lesbarer Beweis durch Gegenbeispiel

Listing 28: src/CommentedCounterProof.java

```
1  /*
2    Samplecode for "correct" commenting.
3
4    This code will demonstrate how to use comments,
5    either in a multiline fasion(like this one), or
6    by single line
7   */
8  public class CommentedCounterProof {
9    // main entry point of class
10   public static void main(String argv[]){
11     // Supply our user with some basic
12     // information about this sample.
13     System.out.println("A better
14   }
15 } // done with this sample
```

Erklärung

```
// Einzeiliger Kommentar
/* Kommentar bis */
```

Inhaltsverzeichnis

1 Organisatorisches

2 Kompilieren und Ausführen

3 Hello World - das erste Programm
- Ausführen

4 Abarbeiten von Befehlen

5 Variablen und einfache Typen

6 Operatoren

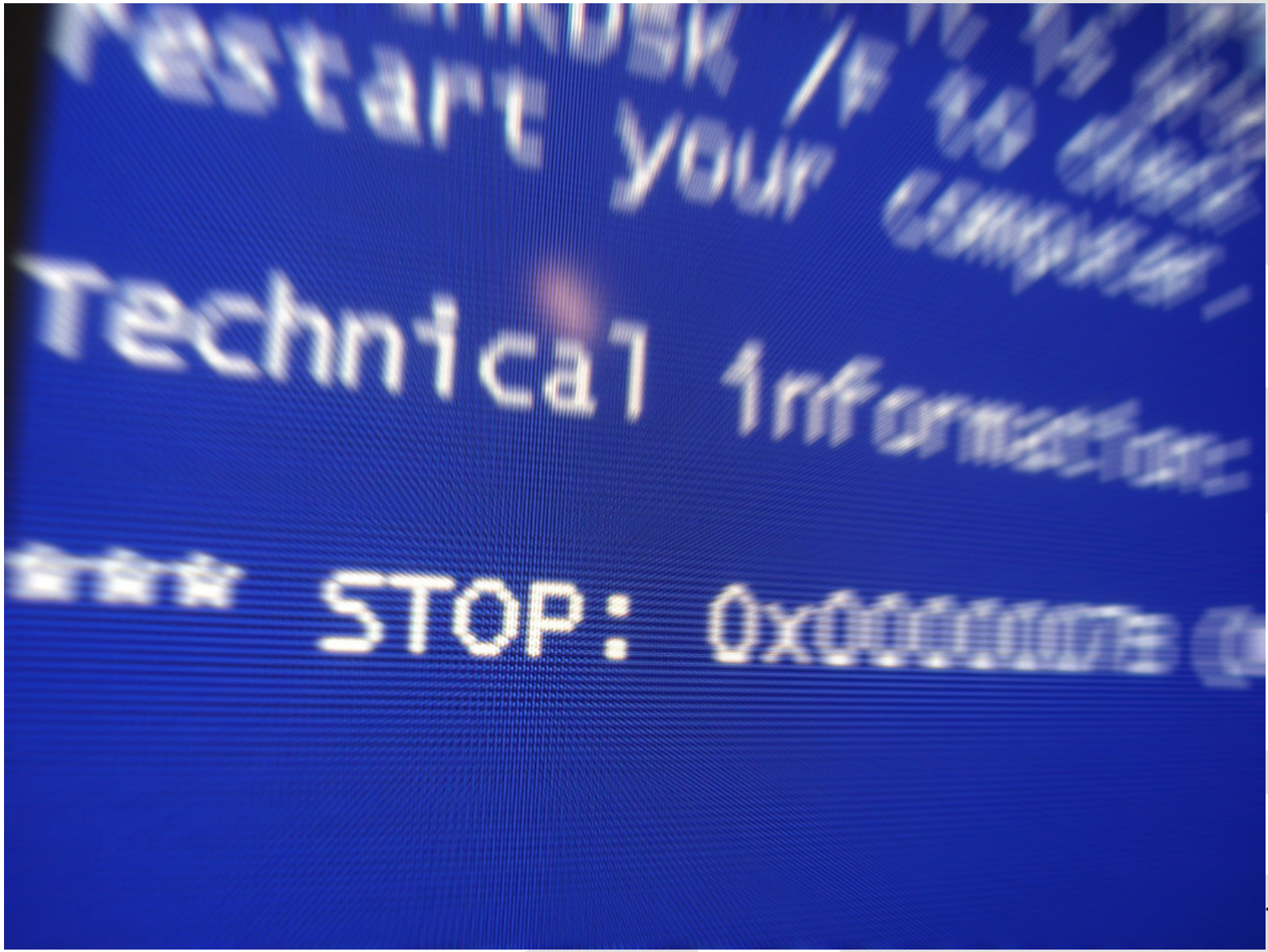
7 Fallunterscheidungen

8 Kommentare

9 Probleme und Fehlermeldungen

10 Zusammenfassung

Fehler



[7]

[7] by Justin Marty

Fehler I

Listing 29: src/no–compile/ErrorOne.java

```
1 public class ErrorOne {
2   public static void main(String[] argv) {
3     System.out.println("Hello World")
4   }
5 }
```

Fehler I

Listing 30: src/no–compile/ErrorOne.java

```
1 public class ErrorOne {
2   public static void main(String[] argv) {
3     System.out.println("Hello World")
4   }
5 }
```

[mario src]$ javac src/no-compile/ErrorOne.java

```
----------
1. ERROR in src/no-compile/ErrorOne.java (at line 3)
System.out.println("Hello World")
                                  ^
Syntax error, insert ";" to complete BlockStatements
----------
1 problem (1 error)
```

Fehler II

Listing 31: src/no–compile/ErrorTwo.java

```
1 public class ErrorTwo {
2   public static void main(String[] argv) {
3     int solution = 3 + 5;
4     System.out.println("3 + 5 =" + soluton);
5   }
6 }
```

Fehler II

Listing 32: src/no–compile/ErrorTwo.java

```
1 public class ErrorTwo {
2   public static void main(String[] argv) {
3     int solution = 3 + 5;
4     System.out.println("3 + 5 =" + soluton);
5   }
6 }
```

[mario src]$ javac src/no-compile/ErrorTwo.java

```
----------
1. ERROR in src/no-compile/ErrorTwo.java (at line 4)
System.out.println("3 + 5 =" + soluton);
                               ^^^^^^^

soluton cannot be resolved
----------
1 problem (1 error)
```

Fehler III

Listing 33: src/no–compile/ErrorThree.java

```
3       int dayOfWeek = 1;
4       switch(dayOfWeek) {
5         case 1:
6           String dayName = "Monday";
7         break;
8         // ...
9       }
10      System.out.println("Weekday = " + dayName);
```

Fehler III: Compilermeldung

[mario src]$ javac src/no-compile/ErrorThree.java

```
----------
1. ERROR in src/no-compile/ErrorThree.java (at line 10)
System.out.println("Weekday = " + dayName);
                                  ^^^^^^^

dayName cannot be resolved
----------
```

Fehler III: Fehlerstelle

Listing 34: src/no–compile/ErrorThree.java

```
4       switch(dayOfWeek) {
5          case 1:
6             String dayName = "Monday";
7          break;
8          // ...
9       }
10      System.out.println("Weekday = " + dayName);
```

Fehler III: Fehlerstelle

Listing 35: src/no–compile/ErrorThree.java

```
4       switch(dayOfWeek) {
5          case 1:
6             String dayName = "Monday";
7          break;
8          // ...
9       }
10      System.out.println("Weekday = " + dayName);
```

- Variablen gelten nur innerhalb ihres Blockes { }
- dayName gilt nur innerhalb von switch
- Lösung: Deklaration von dayName ausserhalb

Fehler III: Lösung

Listing 36: src/ErrorThreeSolution.java

```
3       int dayOfWeek = 1;
4       String dayName = "";
5       switch(dayOfWeek) {
6          case 1:
7             dayName = "Monday";
8          break;
9          // ...
10      }
11      System.out.println("Weekday = " + dayName);
```

Fehler III: Lösung

Listing 37: src/ErrorThreeSolution.java

```
3       int dayOfWeek = 1;
4       String dayName = "";
5       switch(dayOfWeek) {
6          case 1:
7             dayName = "Monday";
8          break;
9          // ...
10      }
11      System.out.println("Weekday = " + dayName);
```

Ausgabe

```
[mario src]$ java ErrorThreeSolution
Weekday = Monday
```

Fehler IV

Listing 38: src/ErrorFour.java

```
3     System.out.println( 1 / 0 );
```

Fehler IV

Listing 39: src/ErrorFour.java

```
3     System.out.println( 1 / 0 );
```

Kein Compilerfehler

```
[mario src]$ javac ErrorFour.java
[mario src]$
```

Fehler IV

Listing 40: src/ErrorFour.java

```
3     System.out.println( 1 / 0 );
```

Kein Compilerfehler

```
[mario src]$ javac ErrorFour.java
[mario src]$
```

Aber: Laufzeitfehler

```
[mario src]$ java ErrorFour
Exception in thread "main" java.lang.
ArithmeticException: / by zero
at ErrorFive.main(ErrorFour.java:3)
```

Inhaltsverzeichnis

1 Organisatorisches

2 Kompilieren und Ausführen

3 Hello World - das erste Programm
- Ausführen

4 Abarbeiten von Befehlen

5 Variablen und einfache Typen

6 Operatoren

7 Fallunterscheidungen

8 Kommentare

9 Probleme und Fehlermeldungen

10 Zusammenfassung

Zusammenfassung

- Erzeugen und Kompilieren eine Javaprogrammes
- Einfache Konstrukte
  - Variablen
  - Bedingungen
  - Kommentare
- Kompilerfehler und deren Lösung

Was kommt jetzt?

- Feedback hier vorne abgeben
- 1. Übung im TEL106/TEL206
- Danach: Essen
- Danach: 2. Vorlesung
- Danach: 2. Übung

4!


4!