



JAVA TUTORIAL

Vernetzte Systeme - SS 06

Übersicht



- Warum Java?
 - Interoperabilität
 - grosse und gut dokumentierte Library
 - weit verbreitet
 - Syntax sehr nahe an C
 - Erfahrung: Java wird gegenüber Oberon / Eiffel / C bevorzugt
- Ziel dieser Einführung
 - Die wichtigsten Konzepte verstehen
 - Bestehenden Java-Code verstehen und erweitern können
 - ~~Kompletter Überblick über alle Java-Features?~~
 - ~~Ganze Java-Programme selber schreiben können?~~



Distributed Computing Group Java Tutorial 2

1 – Hello World



Muss in HelloWorld.java gespeichert werden!

Einstiegspunkt für jedes Programm

Übergebene Parameter vom Typ *Array of String*

```
class HelloWorld {  
    public static void main (String args[ ]) {  
        System.out.println("Hello World!");  
    }  
}
```

Ausgabestream. Erscheint in der Eclipse Konsole

Nicht beschränkt auf Strings!



2 – Fibonacci



Primitve Types:
char, byte, short,
int, long, float,
double, boolean,
void

plus Wrapper:
Character, Byte,
Short, Integer,
Long, Float,
Double, Boolean,
Void

```
public class Fibonacci {  
    public static void main(String[ ] args) {  
        // declaration of variables  
        int counter = 10;  
        int oldNumber = 0;  
        int newNumber = 1;  
  
        // the first two Fibonacci Numbers are predefined  
        System.out.println("1. Fibonacci Number: " + oldNumber);  
        System.out.println("2. Fibonacci Number: " + newNumber);  
  
        // generate the remaining numbers  
        for (int i=3; i<=counter; i++) {  
            int temp = oldNumber + newNumber;  
            oldNumber = newNumber;  
            newNumber = temp;  
            System.out.println(i + ". Fibonacci Number: " + newNumber);  
        }  
    }  
}
```

Kommentare wie in C:
// ... oder /* ... */

bekannt aus C:
for, while, if-else, switch

verschiedene Typen
mischbar!



3 – Fibonacci extended

```
public class Fibonacci {
    public static void main(String[] args) {
        // declaration of variables
        int counter = 10;
        int oldNumber = 0;
        int newNumber = 1;

        // parse counter value
        counter = Integer.parseInt(args[0]);
        System.out.println("Printing the first " + counter + " numbers:");

        // the first two Fibonacci Numbers are predefined
        System.out.println("1. Fibonacci Number: " + oldNumber);
        System.out.println("2. Fibonacci Number: " + newNumber);

        // generate the remaining numbers
        for (int i=3; i<=counter; i++) {
            int temp = oldNumber + newNumber;
            oldNumber = newNumber;
            newNumber = temp;
            System.out.println(i + ". Fibonacci Number: " + newNumber);
        }
    }
}
```

Da kann einiges schief gehen!!



3 – Fibonacci extended

```
...
// parse counter value
try {
    counter = Integer.parseInt(args[0]);
} catch (NumberFormatException e) {
    System.out.println("Sorry, first argument must be a number");
    return;
} catch (ArrayIndexOutOfBoundsException e) {
    System.out.println("Usage: java Fibonacci <number>");
    return;
}
System.out.println("Printing the first " + counter + " numbers:");

// the first two Fibonacci Numbers are predefined
System.out.println("1. Fibonacci Number: " + oldNumber);
System.out.println("2. Fibonacci Number: " + newNumber);
...

```

Exceptions aus dem try-Block



4 – Fibonacci even more extended

```
...
// parse counter value
System.out.print("How many Fibonacci Numbers ... ? ");
counter = readInteger();

// the first two Fibonacci Numbers are predefined
...
}

static int readInteger() {
    String line;
    BufferedReader input = new BufferedReader(
        new InputStreamReader(System.in));

    try {
        line = input.readLine();
        return Integer.parseInt(line);
    } catch (Exception e) {
        return 0;
    }
}
}
```

zusätzliche Methode readInteger() liest Integers aus der Konsole

InputStream: Zeichenbasiertes Lesen aus einer Quelle (Datei, Netzwerk, ...)

Erstellen eines Objekts der Klasse BufferedReader

Gegenstück zu System.out

Supertyp aller Exceptions

Distributed Computing Group Java Tutorial 7

5 – Bookmark Verwaltung

Bookmarks:

- Enthält eine Liste von Websites
- addBookmark() fügt eine Website hinzu
- deleteBookmark() entfernt eine Website
- toString() zum Ausgeben der Bookmarks

Erlaubt die Ausgabe des Inhaltes eines Objekts o durch System.out.print(o)

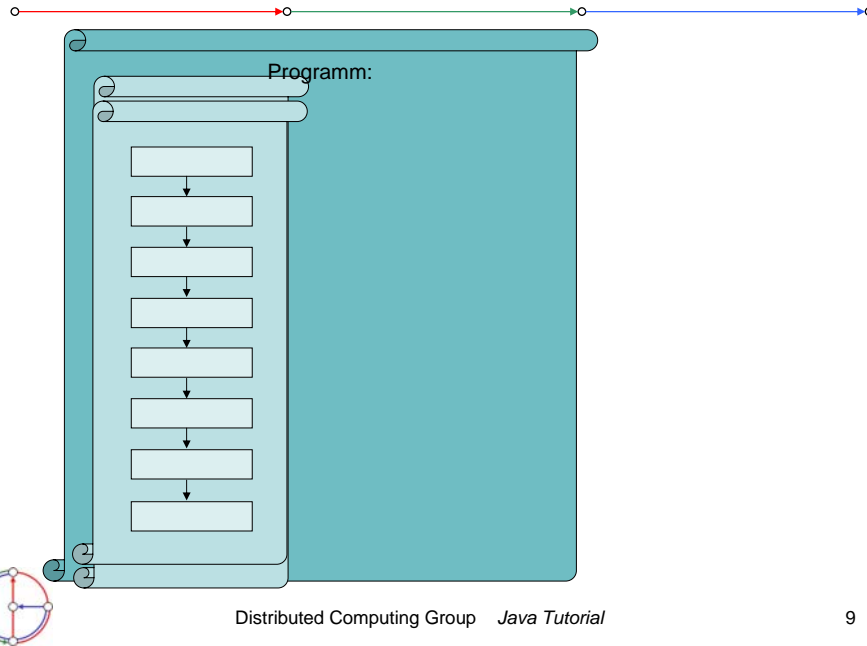
Website:

- name
- url
- description

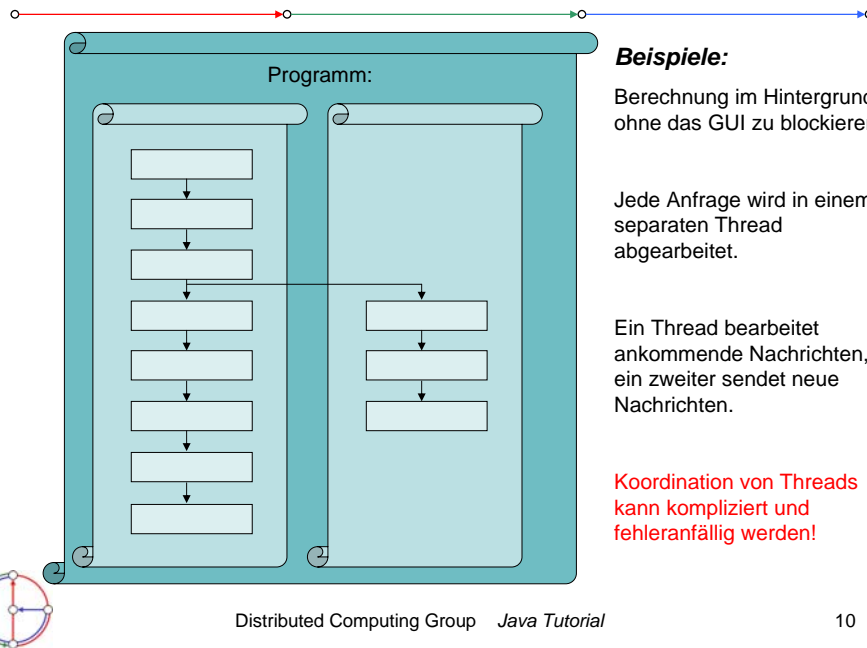
(alles Strings)

Distributed Computing Group Java Tutorial 8

6 – Threads



6 – Threads



Beispiele:

Berechnung im Hintergrund, ohne das GUI zu blockieren.

Jede Anfrage wird in einem separaten Thread abgearbeitet.

Ein Thread bearbeitet ankommende Nachrichten, ein zweiter sendet neue Nachrichten.

Koordination von Threads kann kompliziert und fehleranfällig werden!

That's it!



Fragen?

Freitag, 13.15-15.00, ETZ F76.1

