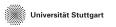
PSE – Vorkurs Tag 2

Tobias, Philipp, Linus, Tillmann

FIUS - Fachgruppe Informatik Universität Stuttgart

6. Juli 2025





am Anfang immer Vortages recap?	3
item oder bullet item?	4
mint style festlegen (z.B. friendly)	5





Recap Tag 1

am Anfang immer Vortages recap?





Was sind Verzweigungen?

► Programme müssen Entscheidungen treffen item oder bullet item?

► Beispiel: Links oder rechts gehen?





Boolean - Wahr oder Falsch

- Datentyp mit zwei Werten: true und false
- Wird für Bedingungen verwendet
- Z.B: heuteMontag = true;

mint style festlegen (z.B. friendly)



```
• a == b Wahr, wenn a gleich b ist
```

```
boolean result = (a == b); // result true, wenn a gleich b
```



```
a == b Wahr, wenn a gleich b ist
boolean result = (a == b); // result true, wenn a gleich b
a != b Wahr, wenn a ungleich b ist
boolean result = (a != b); // result true, wenn a ungleich b
```



```
a == b Wahr, wenn a gleich b ist
boolean result = (a == b); // result true, wenn a gleich b
a != b Wahr, wenn a ungleich b ist
boolean result = (a != b); // result true, wenn a ungleich b
a < b Wahr, wenn a kleiner als b ist</li>
boolean result = (a < b); // result true, wenn a kleiner als b</li>
```





```
a == b Wahr, wenn a gleich b ist
boolean result = (a == b); // result true, wenn a gleich b
a != b Wahr, wenn a ungleich b ist
boolean result = (a != b); // result true, wenn a ungleich b
a < b Wahr, wenn a kleiner als b ist</li>
boolean result = (a < b); // result true, wenn a kleiner als b</li>
analog bei a > b, a <= b, a >= b
```





ightharpoonup !a ightharpoonup nicht a





- ightharpoonup !a ightharpoonup nicht a
- ightharpoonup a && b ightharpoonup a UND b



- ightharpoonup !a ightharpoonup nicht a
- ightharpoonup a && b \rightarrow a UND b
- ightharpoonup a || b ightharpoonup a ODER b



- ightharpoonup !a ightharpoonup nicht a
- ightharpoonup a && b ightharpoonup a UND b
- ightharpoonup a || b ightarrow a ODER b

Klammern priorisieren:

▶ true || false && false →

- ightharpoonup !a ightharpoonup nicht a
- ightharpoonup a && b \rightarrow a UND b
- ightharpoonup a || b ightharpoonup a ODER b

Klammern priorisieren:

▶ true || false && false \rightarrow true



- ightharpoonup !a ightharpoonup nicht a
- ightharpoonup a && b ightharpoonup a UND b
- ightharpoonup a || b ightharpoonup a ODER b

Klammern priorisieren:

- ▶ true || false && false → true
- (true || false) && false \rightarrow





- ightharpoonup !a ightharpoonup nicht a
- ightharpoonup a && b \rightarrow a UND b
- ightharpoonup a || b ightharpoonup a ODER b

Klammern priorisieren:

- ▶ true || false && false → true
- ► (true || false) && false → false





if-Verzweigung

- Boolescher Ausdruck entscheidet
- Ausführung nur wenn true
- Syntax:

```
if (Bedingung) {
   // Code bei true
}
```





if-Verzweigung

- Boolescher Ausdruck entscheidet
- Ausführung nur wenn true
- Syntax:

```
if (Bedingung) {
    // Code bei true
}

Beispiel:
boolean a = true;
if (a) {
    System.out.println("a ist wahr");
}
```



if-Verzweigung

- Boolescher Ausdruck entscheidet
- Ausführung nur wenn true
- Syntax:

```
if (Bedingung) {
    // Code bei true
}

Beispiel:
boolean a = true;
if (a) {
    System.out.println("a ist wahr");
}

//Ausgabe: a ist wahr
```



if-else-Verzweigung

- Erweiterung der if-Anweisung
- Ausführung bei true oder false
- Syntax:

```
if (Bedingung) {
    // Code bei true
} else {
    // Code bei false
}
```



if-else-Verzweigung

- Erweiterung der if-Anweisung
- Ausführung bei true oder false
- Syntax:

```
if (Bedingung) {
2 // Code bei true
3 } else {
4 // Code bei false
  }
 Beispiel:
  boolean a = false;
  if (a) {
    System.out.println("a ist wahr");
  } else {
    System.out.println("a ist falsch");
  }
```



if-else-Verzweigung

- Erweiterung der if-Anweisung
- Ausführung bei true oder false
- Syntax:

```
if (Bedingung) {
     // Code bei true
3 } else {
   // Code bei false
 Beispiel:
  boolean a = false;
  if (a) {
     System.out.println("a ist wahr");
  } else {
     System.out.println("a ist falsch");
  //Ausgabe: a ist falsch
```



while-Schleife

- Wiederholt Anweisungen, solange eine Bedingung true ist
- ► Syntax: while (Bedingung) { /* Code */ }



while-Schleife

Wiederholt Anweisungen, solange eine Bedingung true ist

```
► Syntax: while (Bedingung) { /* Code */ }
  Beispiel:
   Scanner scanner = new Scanner(System.in);
   String eingabe = "";
   while (!eingabe.equals("ok")) {
     System.out.println("Bitte 'ok' eingeben:");
     eingabe = scanner.nextLine();
   }
1 // Ausqabe
2 // Bitte 'ok' eingeben:
3 // ...
4 // Bitte 'ok' eingeben:
5 // (Benutzer tippt "ok") → Schleife endet
```

