

# GRUNDLAGEN DIGITALER GESTALTUNG 2

## 2. SEMESTER | SOMMERSEMESTER 2010 | KURS 2

---

### basics der programmierung: for-structure

---

Was den Code wesentlich einfacher und programmieren erst wirklich sinnvoll macht ist, Redundants zu vermeiden. Dadurch läßt sich viel unnötige Arbeit ersparen und der Code wird flexibler.

Beispiel:

```
void setup(){
  size(300, 300);
}
```

```
void draw(){
  line(0, 0, width, 0);
  line(0, 40, width, 40);
  line(0, 80, width, 80);
  line(0, 120, width, 120);
  line(0, 160, width, 160);
  line(0, 200, width, 200);
  line(0, 240, width, 240);
  line(0, 280, width, 280);
}
```

Der Code wiederholt ständig eine Funktion, nur die Parameter ändern sich, und zwar kommt pro Linie 40 beim x-Wert dazu. Ein Algorithmus, der sich einfacher schreiben läßt.

Hierfür läßt sich die For-Schleife verwenden,

angewand auf das vorhergehende Beispiel lautet sie wie folgt:

```
void setup(){
  size(300, 300);
}
```

```
void draw(){
  for(int i = 0; i < 8; i++){
    line(0, i*40, width, i*40);
  }
}
```

Was hier passiert ist folgendes: Die Ganzzahl i wird definiert, sie ist 0:

```
int i = 0;
```

...Zusätzlich wird festgelegt, daß sie kleiner als 8 ist:

```
i < 8;
```

...und immer eins mehr wird:

```
i++;
```

```
// i++ ist das gleiche wie i +=1;
```

Processing läuft so lange durch die For-Schleife durch, bis alle Zahlen berechnet, und dadurch auch alle Linien gezeichnet sind.

<http://www.processing.org/reference/for.html>

# GRUNDLAGEN DIGITALER GESTALTUNG 2

2. SEMESTER | SOMMERSEMESTER 2010 | KURS 2

.....  
basics der programmierung: for-structure  
.....

## **Aufgabe 2.1**

**Zeichne mit Hilfe der for-Schleife 10 Kreise, die alle auf der Bühne zentriert sind, und jeweils immer um 20 Pixel größer werden.**

# GRUNDLAGEN DIGITALER GESTALTUNG 2

## 2. SEMESTER | SOMMERSEMESTER 2010 | KURS 2

---

### basics der programmierung: for-structure

---

Eine for-Schleife kann gedoppelt werden, eine for-Schleife in einer for-Schleife.

```
for(int x = 0; x <= width; x+=20){
  for(int y = 0; y <= height; y+=20){
    ellipse(x, y, 10, 10);
  }
}
```

Damit kann z.B eine Fläche aus Kreisen generiert werden:

```
void setup(){
  size(200, 200);
  smooth();
  noStroke();
}
void draw(){
  background(255);
  fill(0, 40);
  stroke(255, 0, 0);
  for(int x = 0; x <=200; x+=10){
    for(int y = 0; y <=200; y+=10){
      ellipse(x, y, 10, 10);
    }
  }
}
```

oder auch Muster:

```
void setup(){
  size(200, 200);
  smooth();
  noStroke();
}
```

```
void draw(){
  background(200);
  fill(0, 40);
  for(int x = -20; x<= width; x+=10){
    for(int y = -20; y<= height; y+=10){
      ellipse(x+y/8.0, y+x/8.0, 15+x/2, 10);
    }
  }
}
```

**Aufgabe 2.2**  
**entwickle mit Hilfe der for-Schleife eine Muster**

# GRUNDLAGEN DIGITALER GESTALTUNG 2

## 2. SEMESTER | SOMMERSEMESTER 2010 | KURS 2

---

### basics der programmierung: if-structure

---

Eine wichtige Schleife für Interaktion ist die if-Schleife.

Die if-Schleife fragt Parameter ab, und führt anschließend eine Funktion aus, oder nicht.

Wie der Name „if“ schon sagt:  
WENN die Konditionen zutreffen, DANN...

```
if(hier werden die Konditionen definiert){
  //hier kommt hin was ausgeführt werden soll, wenn die
Konditionen zutreffen;
}
```

Zum Beispiel könnte die Kondition eine bestimmte Mouseposition sein:

Wenn die X-Position der Maus größer ist als die Hälfte der Bühne, dann verwende Rot als Füllfarbe.

Wenn die X-Position der Maus kleiner ist als die Hälfte der Bühne, dann verwende Schwarz als Füllfarbe.

```
void setup(){
  size(400, 400);
}
```

```
void draw(){
  background(100);
```

```
if(mouseX > width/2){
  fill(255, 0, 0);
}
```

```
if(mouseX < width/2){
  fill(0);
}
```

```
rect(mouseX, height/2, 20, 20);
}
```

Processing liest die Kondition und prüft ob sie stimmen. Wenn sie stimmen, wird ausgeführt was in den geschweiften Klammern steht. Wenn sie nicht stimmen, wird alles was in den geschweiften Klammern steht nicht gelesen.

<http://www.processing.org/reference/if.html>

# GRUNDLAGEN DIGITALER GESTALTUNG 2

## 2. SEMESTER | SOMMERSEMESTER 2010 | KURS 2

---

### basics der programmierung: if-structure

---

Diese Zeilen:

```
if(mouseX > width/2){
  fill(255, 0, 0);
}
if(mouseX < width/2){
  fill(0);
}
```

lassen sich auch wie folgt schreiben:

Es wird solange Rot gemalt, solange die X-Position größer ist als die Hälfte der Bühnenbreite.  
Ansonsten wird Schwarz gemalt.

```
if(mouseX > width/2){
  fill(255, 0, 0);
}else{
  fill(0);
}
```

Konditionen können Kombiniert werden, dafür gibt es folgende Operator:

```
&& // und
||  // oder
```

Zum Beispiel kann die X-Position mit der Y-Position der Maus kombiniert werden:

```
void setup(){
  size(400, 400);
  noCursor();
}

void draw(){
  background(255, 0, 0);
  fill(0);
  rect(width/2, height/2, width/2, height/2);
  if(mouseX > width/2 && mouseY > height/2){
    fill(255, 0, 0);
  }else{
    fill(0);
  }
  rect(mouseX, mouseY, 20, 20);
}
```

# GRUNDLAGEN DIGITALER GESTALTUNG 2

## 2. SEMESTER | SOMMERSEMESTER 2010 | KURS 2

---

### basics der programmierung: if-structure

---

Die Konditionen können auch nach dem Negativ-Zustand abgefragt werden.

*!* // *nicht*

Wenn die X-Position der Maus NICHT größer ist als die Hälfte der Bühnengröße, dann..

```
if(!(mouseX > width/2)){  
  fill(255, 0, 0);  
}else{  
  fill(0);  
}
```

beachte: dabei muß die gesamte Kondition in Klammern gesetzt werden.

```
void setup(){  
  size(400, 400);  
  noCursor();  
}
```

```
void draw(){  
  background(255, 0, 0);  
  if(!(mouseX > width/2)){  
    fill(255, 0, 0);  
  }else{  
    fill(0);  
  }  
  rect(mouseX, width/2, 20, 20);  
}
```

# GRUNDLAGEN DIGITALER GESTALTUNG 2

## 2. SEMESTER | SOMMERSEMESTER 2010 | KURS 2

---

### basics der programmierung: if-structure

---

Folgende Operatoren können verwendet werden, um die Bedingung zu formulieren:

```
x == y // x ist gleich y
x != y // x ist nicht gleich y
x < y // x kleiner als y
x > y // x größer als y
```

Beachte:

= ist etwas anderes als ==

```
x = mouseX;
```

hier wird die Variable x gleich gesetzt mit der X-Position der Maus

```
if(x == mouseX){
}
```

hier wird die Frage gestellt, ob x gleich der X-Position der Mouse ist.

Im folgenden Code seht ihr wie eine if-Schleife, eine Variable ändert:

```
int x = 100;
int xspeed = 1;
void setup() {
  size(600, 200);
  smooth();
  background(255);
}

void draw() {
  noStroke();
  fill(255, 10);
  rect(0, 0, width, height);

  // Addiert die Geschwindigkeit zu der x-position
  x = x + xspeed;

  // der Richtungswechsel
  if ((x > width) || (x < 0)) {
    xspeed = xspeed * -1;
  }

  // der Kreis verwendet die x-Position
  stroke(0);
  fill(175);
  ellipse(x, height/2, 16, 16);
}
```

# GRUNDLAGEN DIGITALER GESTALTUNG 2

## 2. SEMESTER | SOMMERSEMESTER 2010 | KURS 2

---

### basics der programmierung: boolean

---

Boolean ist ein Datentyp, der nur zwei Zustände haben kann: true oder false.  
Er funktioniert wie ein Lichtschalter, der entweder an oder aus ist:

```
boolean goRight; //ein Boolean wird erstellt
int myXpos;
void setup(){
  size(200, 200);
  goRight = true; //anfangs hat er den Zustand true
  myXpos = 0;
}
void draw(){
  background(0);
  if(myXpos < 0){
    goRight = true;
  } //wenn myXpos kleiner als 0 ist, schaltet er auf true
  if(myXpos > width-20){
    goRight = false;
  } //wenn myXpos größer als die Breite ist, schaltet er auf false
  if(goRight == true){
    myXpos++;
    fill(255, 0, 0);
  }
  if(goRight == false){
    myXpos--;
    fill(255);
  }
  rect(myXpos, height/2, 20, 20);
}
```

# GRUNDLAGEN DIGITALER GESTALTUNG 2

## 2. SEMESTER | SOMMERSEMESTER 2010 | KURS 2

.....  
aufgaben bis zum 26.4  
.....

### **Aufgabe 2.1**

**zeichne mit Hilfe der for-Schleife 10 Kreise, die alle auf der Bühne zentriert sind, und jeweils immer um 20 pixel größer werden.**

### **Aufgabe 2.2:**

**entwickle mit Hilfe der for-Schleife eine Muster**

### **Aufgabe 2.3**

**entwickle mit Hilfe der if-Schleife einen Kreis, der sich wie eine endlos rollende Billardkugel über die Bühne bewegt, und dabei alle Seiten der Bühne anrollt.**

### **Aufgabe 2.4**

**entwickle mit Hilfe der if-Schleife eine interaktive Grafik.**

```
if(überfordert){  
  don't_worry(be_patient, keep_trying);  
}else{  
  checkout(, http://www.processing.org/learning/pvector/ "  
};
```