

# GRUNDLAGEN DIGITALER GESTALTUNG 2

## 2. SEMESTER | SOMMERSEMESTER 2010 | KURS 1

---

### das werkzeug „processing“

---

Processing ist eine Programmiersprache und Programmierumgebung, die auf der Programmiersprache Java basiert. Processing versteht sich als offenes Projekt initiiert und entwickelt von Ben Fry und Casey Reas am MIT in Boston und am Interaction Design Institute Ivrea. Processing ist eine „open source“ Programmiersprache, und untersteht der GNU General Public License, das heißt in Kurzform:

- Die Software (d. h. der Quelltext) liegt in einer für den Menschen lesbaren und verständlichen Form vor
- Die Software darf beliebig kopiert, verbreitet und genutzt werden
- Die Software darf verändert und in der veränderten Form weitergegeben werden

[mehr über GNU General Public License:](http://de.wikipedia.org/wiki/GNU_General_Public_License)  
[http://de.wikipedia.org/wiki/GNU\\_General\\_Public\\_License](http://de.wikipedia.org/wiki/GNU_General_Public_License)

Processing war ein Projekt an der M.I.T in dem kreatives Coding eine wesentliche Rolle spielt. Das Ziel war es, einen Syntax zu entwickeln, der es Menschen einfach macht, ihre Ideen in Programmiersprache einfach umzusetzen, obwohl sie keine Programmierer sind. Vorallem für den Kreativbereich wie Designer, Künstler, Architekten, Forscher etc.

Es gibt bereits Programme in dieser Richtung die für den Kreativbereich entwickelt wurden wie zum Beispiel Flash oder Director. Der wesentliche Vorteil von Processing ist die bessere Performance und die Verknüpfung von Code und Design. Außerdem sind Flash und Director kommerzielle Programme. Der Syntax von Processing hat Ähnlichkeiten mit den gängigen Sprachen wie zum Beispiel Java, Actionscript, Lingo, der Scriptsprache Javascript (nicht zu verwechseln mit Java) und der statische Auszeichnungssprache HTML. Der Umstieg auf andere Computersprachen fällt somit leichter.

Processing verbindet die Paradigmen der verschiedenen Tools und Programmiersprachen. Es vereinheitet die Einfachheit von Flash und Director in der Programmerstellung mit der klareren Programmstruktur und besseren Performance von Java, C++ und OPENGL.

[mehr über Processing:](http://www.processing.org/about)  
<http://www.processing.org/about>

[zum Download von Processing:](http://www.processing.org/download)  
<http://www.processing.org/download>

# GRUNDLAGEN DIGITALER GESTALTUNG 2

## 2. SEMESTER | SOMMERSEMESTER 2010 | KURS 1

---

### links

---

processing                      [www.processing.org/exhibition/](http://www.processing.org/exhibition/)

#### people:

ben fry                              [www.benfry.com](http://www.benfry.com)

casey reas                         [www.reas.com](http://www.reas.com)

daniel shiffman                  [www.shiffman.net](http://www.shiffman.net)

golan levin                        [www.flong.com](http://www.flong.com)

zachary liebermann              [www.thesystemis.com](http://www.thesystemis.com)

#### organisation:

processing                         [www.processing.org](http://www.processing.org)

infosthetics                       [www.infosthetics.com](http://www.infosthetics.com)

m.i.t                                 <http://web.mit.edu/>

# GRUNDLAGEN DIGITALER GESTALTUNG 2

## 2. SEMESTER | SOMMERSEMESTER 2010 | KURS 1

.....  
und so einfach ist es zu handhaben  
.....



### Run

Kompiliert den Code, öffnet ein Fenster, und läßt das Programm darin ablaufen. Wenn die Hochstelltaste dabei gedrückt ist, wird der Präsentiermodus aufgerufen.



### Stop

beendet das laufende Programm



### New

erstellt eine neue Skizze (Projekt) im selben Fenster. Um eine neue Skizze in einem neuen Fenster zu erstellen gehe zu File/New



### Open

öffnet ein Menü mit Optionen um Skizzen und Beispiele zu öffnen.



### Save

Sichert das Programm. Wenn es unter einem anderem Namen gespeichert werden soll, gehe zu File/Save as



### Export

exportiert die Skizze in ein Java Applet.



# GRUNDLAGEN DIGITALER GESTALTUNG 2

## 2. SEMESTER | SOMMERSEMESTER 2010 | KURS 1

---

### grundregeln zum syntax: anweisungen und funktionen

---

Der Syntax von Processing ist „Case-Sensitive“ das heißt es muß Groß- und Kleinschreibung berücksichtigt werden.  
 // zwei Schrägsriche markieren, daß die Zeile vom  
 // Programm nicht gelesen werden soll.

Anweisung:

Die Anweisung ist die elementarste Programmeinheit. Eine Anweisung wird immer mit einem Semikolon beendet. Codiert könnte die Anweisung für einen Milchkaffee wie folgt aussehen:

*Milchkaffee = zusammenschütteninbecher(Kaffee,Milch);*

Funktionsaufrufe sind einfache Anweisungen im folgenden Format:

*funktionsname();*

*funktionsname(parameter);*

*funktionsname(parameter1, parameter2, parameter3);*

Processing hat bereits eine große Anzahl an Funktionen, die im folgenden näher erläutert werden.

Um zu zeichnen brauchen wir zum einen eine „Bühne“, das Fenster, was sich öffnet, wenn wir den Code abspielen. Die Funktion um die Größe des Fenster in Pixeln festzulegen lautet:

*size(Breite, Höhe);*

Wenn die Größe des Fensters nicht definiert wird, ist die Größe des Fensters in der Grundeinstellung:  
*size(100, 100);*

Eine weitere Funktion ist der Hintergrund:

*background(rot, grün, blau);*

Standardmäßig sind die Farbangaben in Processing in RGB-Werten angegeben. Für Rot, Grün und Blau wird ein Zahlenwert zwischen 0 und 255 angegeben.

Die Grundeinstellung des Hintergrunds ist Hellgrau.

Um eine Linie zu zeichnen:

*line(x1, y1, x2, y2);*

Über Koordinatenangaben wird die Position der Linie angegeben. Processing verwendet das kartesische Koordinatensystem. Der Nullpunkt ist oben links.

Um einen Kreis zu zeichnen:

*ellipse(x, y, Breite, Höhe);*

Um ein Rechteck zu zeichnen:

*rect(x, y, Breite, Höhe);*

# GRUNDLAGEN DIGITALER GESTALTUNG 2

## 2. SEMESTER | SOMMERSEMESTER 2010 | KURS 1

.....  
grundregeln zum syntax: anweisungen und funktionen  
.....

Es gibt verschiedene Möglichkeiten die Füllung zu definieren:

*fill(grauwert);*

oder:

*fill(grauwert, alphawert);*

oder:

*fill(rot,grün, blau);*

oder:

*fill(rot,grün, blau, alphawert);*

Die Funktionen werden von oben nach unten gelesen, deswegen muß die Funktion der Farbe eine Zeile vor dem was mit dieser Farbe gezeichnet werden soll stehen.

### **Aufgabe 1.1**

**mache eine Zeichnung (z.B einfache Figur, Gesicht, oä)  
auf Papier und zeichne es in Processing nach.**

# GRUNDLAGEN DIGITALER GESTALTUNG 2

## 2. SEMESTER | SOMMERSEMESTER 2010 | KURS 1

---

### grundregeln zum syntax: wirkung der reihenfolge

---

Processing liest die Zeilen von oben nach unten, und führt die Anweisungen in dieser Reihenfolge aus.

```
rect(10, 10, 50, 50);
rect(20, 20, 50, 50);
rect(30, 30, 50, 50);
```

und nun in einer anderen Reihenfolge:

```
rect(30, 30, 50, 50);
rect(20, 20, 50, 50);
rect(10, 10, 50, 50);
```

Wie man sieht ist das Ergebniss ein anderes.

Processing ist eine Zustandsmaschine. Man setzt verschiedene Zustände diese beeinflussen den Zeichenprozess bis sie geändert werden. Als Beispiel die Funktion der Farbe:

```
fill(rot, grün, blau);
```

Beispiel:

```
fill(255, 0, 0);
rect(10, 10, 50, 50);
rect(20, 20, 50, 50);
rect(30, 30, 50, 50);
fill(0, 0, 255);
rect(40, 40, 50, 50);
```

Die Rechtecke werden so lange rot gemacht, bis eine andere Anweisung folgt.

**Aufgabe 1.2:**  
**wende das Prinzip der Reihenfolge von Parametern auf Linien in Kombination mit der Strichfarbe und Strichstärke an.**

```
stroke(Rot, Grün, Blau);
strokeWeight(Strichstärke);
line(x1, y1, x2, y2);
```

Linien und Füllung kann auch ausgeschaltet werden:

```
noStroke();
noFill();
```

# GRUNDLAGEN DIGITALER GESTALTUNG 2

## 2. SEMESTER | SOMMERSEMESTER 2010 | KURS 1

.....  
grundregeln zum syntax: variable  
.....

Zwei der grundlegenden Variablen sind:  
int (ganze Zahlen: int steht für integer) und  
float (Fließkommazahlen).

```
int meineInteger = 5;  
float meineFloat = 8.99; //Komma wird als Punkt geschrieben
```

Beispiel:

```
int mywidth = 50;  
int myheight = 50;  
int xPos = 20;  
int yPos = 30;
```

```
rect(xPos, yPos, mywidth, myheight);
```

Variablen haben den Vorteil einmal nur deklariert werden zu müssen, und immer wieder verwenden zu können.

### **Aufgabe 1.3:**

**mache eine Zeichnung auf Papier von Kreisen und Rechtecken und zeichne es in Processing nach.  
Lege über Variablen die Farbe fest.**

# GRUNDLAGEN DIGITALER GESTALTUNG 2

## 2. SEMESTER | SOMMERSEMESTER 2010 | KURS 1

---

### grundregeln zum syntax: active-mode

---

Bisher liest Processing die Zeilen nur jeweils einmal, und führt sie aus, und das Programm ist beendet.  
Erst wenn Processing im Active-Modus ist, kann Interaktion stattfinden. Der Active-Modus gliedert sich in die zwei Abschnitte: void setup() und void draw()

void setup():

hier wird alles festgelegt, was nur einmal gelesen werden muß, zum Beispiel die Größe der Bühne (size())

Die Code-Zeile um den Bereich zu definieren lautet:

```
void setup(){
  //hier kommen die statischen Ansagen hin
}
```

void draw():

Der Draw Bereich Funktioniert wie eine Endlosschleife, Processing liest grundsätzlich alle Zeilen pro Frame einmal. Wenn es zum Ende gekommen ist, springt es wieder zu der ersten Zeile innerhalb des void draw() Bereichs, und liest alle Zeilen nochmal, solange, bis das Programm beendet wird. Jetzt besteht die Möglichkeit, zu interagieren, und das gezeichnete zu generieren.

```
void draw(){
  //hier kommen die dynamischen Ansagen hin
}
```

Faustregel:

Jede normale Klammer und jede geschweifte Klammer muß wieder geschlossen werden, sonst gibt es eine Fehlermeldung, und das Programm funktioniert nicht.

Beispiel einer einfachen interaktion:

```
int myXpos;//Variablen werden
int myYpos;//benannt
```

```
void setup(){
  size(400, 400);
  smooth();//das gezeichnete wird weichgezeichnet
}
```

```
void draw(){
  myXpos = mouseX;//die Variablen werden pro Frame
  myYpos = mouseY;// immer wieder neu definiert
  ellipse(myXpos, myYpos, 20, 20);
}
```

Wenn in die Draw() Funktion ein Background gezeichnet wird, wird die Fläche immer wieder gelöscht, bei jedem Frame gibt es eine neue Zeichenfläche.

# GRUNDLAGEN DIGITALER GESTALTUNG 2

## 2. SEMESTER | SOMMERSEMESTER 2010 | KURS 1

.....  
aufgaben bis zum 19.4  
.....

### **Aufgabe 1.1**

**mache eine Zeichnung (z.B einfache Figur, Gesicht, oä)  
auf Papier und zeichne es in Processing nach.**

### **Aufgabe 1.2**

**wende das Prinzip der Reihenfolge von Parametern auf  
Linien in Kombination mit der Strichfarbe und Strichstärke an.**

### **Aufgabe 1.3**

**mache eine Zeichnung auf Papier von Kreisen und  
Rechtecken und zeichne es in Processing nach.  
Lege über Variablen die Farbe fest.**

### **Aufgabe 1.4**

**verwende die Mouseposition um Rechtecke und Kreise  
in ihrer Farbe, Position und oder Größe zu generieren.**

Weitere Erläuterung zu der Sprache von Processing findet  
ihr zusätzlich unter:

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