

# Processing

Skizzieren mit Software

## Der Editor

Hier siehst du die vier wichtigsten Bereiche des Editors `p5.js`, mit dem wir in diesem Kurs arbeiten.



Start / Stop



> sketch.js

```
1 function setup() {  
2   }  
3  
4 function draw() {  
5   }
```

Eingabefeld

Preview

Leinwand

Console


Clear

Fehlermeldungen

## Grundfunktionen der Arbeitsumgebung

1. Führe den Code aus. Klicke dazu auf Start (Play-Symbol) und schaue was passiert.
2. Versuche den Programmcode nachzuvollziehen: Es wird eine Leinwand (Englisch: Canvas) erstellt. Die Farbe der Leinwand ist grau.
3. Ändere die Zahlenwerte bei den Befehlen ***createCanvas*** und ***background*** und schau was passiert.
4. Lies dir die Infobox unten auf der Lernkarte durch.

**Hinweis:** Du kannst den gesamten Code sehen, indem du hier herunterscrollst.



fehlt ***createCanvas(Breite, Höhe)*** wird eine Leinwand erstellt.

Mit dem Befehl ***background(Farbwert)*** wird die Farbe der Leinwand bestimmt.

## Grundstruktur von Processing

Die beiden Grundbausteine **function setup()** und **function draw()** findest du zunächst in jedem Processing-Programm.

- **function setup()** wird einmalig beim Programmstart aufgerufen. Hier solltest du alle Anweisungen einfügen, die zur Vorbereitung der Arbeitsumgebung benötigt werden. Dazu zählt zum Beispiel die Erstellung der Leinwand mit dem **createCanvas**-Befehl.
- **function draw()** wird permanent aufgerufen, während das Programm läuft. Hier werden die Befehle eingefügt, mit denen du auf der Leinwand malen kannst. Welche Befehle das sind, lernst du auf den folgenden Folien. Mit dem **ellipse**-Befehl kannst du z. B. einen Kreis malen.

```
/* Einmalige Konfiguration der Arbeitsumgebung */  
  
function setup() {  
  
}  
  
/* Malen auf der Leinwand */  
function draw() {  
  
}
```

## Erstellen einer einfachen Form


1. Erstelle einen Kreis in der Mitte der Leinwand. Nutze dazu den **ellipse**-Befehl (s. orange Hinweis-Box). Als Hilfestellung kannst du dir die graue Infobox unten durchlesen.
2. Klicke auf Start und beobachte, was passiert.

### Befehle

ellipse(x, y, b, h)

Kreise und Ellipsen werden mit dem Befehl **ellipse(x-Koordinate, y-Koordinate, Breite, Höhe)** erstellt.

Beispiel: **ellipse(120, 120, 50, 60)**

Mitte der Leinwand kannst du bestimmen, indem du jeweils die Breite und Höhe der Leinwand durch zwei teilst. Z. B. hat unsere Leinwand eine Höhe von 300. Geteilt durch 2 ergibt sich 150 als Position für die y-Koordinate des Kreises.


**Tipp 2:** Ein Kreis ist eine Ellipse mit gleicher Breite und Höhe.

## Manipulieren von Formen

1. Schau dir den Code unten an. Hier siehst du eine mögliche Lösung der letzten Aufgabe.
2. Ändere nun die Farbe des Kreises zu schwarz. Nutze dazu den **fill**-Befehl (s. orange Hinweis-Box). Als Hilfestellung kannst du dir die graue Infobox unten durchlesen.

### Befehle

fill(farbcode)


 Die Farbe des Kreises ganz einfach verändern. Dazu brauchst du den **fill**-Befehl. In den Klammern legst du die Füllfarbe fest. Der **Farbcode** der Farbe **Schwarz** ist **0**, der Farbcode der Farbe **Weiß** ist **255**. Alle Zahlenwerte dazwischen sind unterschiedliche Graustufen.

## Interaktive Animationen

1. Passe die Ellipse so an, dass sie der Position deines Mauszeigers folgt. Schreibe dazu "mouseX" dorthin, wo jetzt deine x-Position steht. In diesem Beispiel dort, wo 200 steht.
2. Gehe ebenso für die y-Koordinate vor.

### Befehle

```
ellipse(mouseX, mouseY, b, h)
```

 **mouseY** beschreiben die x- und y-Koordinaten der Mauszeiger-Position. Wenn wir möchten, dass unser Kreis dem Mauszeiger folgt, geben wir dem **ellipse**-Befehl, mouseX und mouseY als x- und y-Koordinate.

**Tipp:** Das funktioniert nur, wenn du deinen Mauszeiger über der Leinwand bewegst.

## Anwendung von Verzweigungen

1. Ändere die Hintergrundfarbe zu schwarz. Zur Erinnerung: Der Farbcode für Schwarz ist 0.
2. Wir wollen den Kreis jetzt unterschiedlich einfärben. Je nachdem, ob die Maus geklickt wird. Wenn die Maus geklickt wird, soll ein Kreis in weiß gezeichnet werden, ansonsten wie gehabt in schwarz. Gehe dazu wie folgt vor.
  1. Füge ***mouselsPressed*** als Bedingung im Code ein. Schau dir dazu die Struktur einer Verzweigung in der orangen Box an.
  2. Ergänze als Befehl A den ***fill***-Befehl mit dem Farbcode für Weiß (255).

### Befehle

```
if (BEDINGUNG) { Befehl A }  
else { Befehl B }
```



## Verzweigungen

Wir haben unser Programm soeben so angepasst, dass der Kreis unterschiedlich eingefärbt wird. Wenn die Maus geklickt wird, wird der Kreis weiß gezeichnet. Wenn die Maus nicht geklickt wird, wird der Kreis schwarz gezeichnet. Dazu haben wir eine Verzweigung genutzt. Nämlich **if-else**:

```
/* Überprüfe die Bedingung. Z. B., ob die Maustaste gedrückt ist */
if (BEDINGUNG) {
    /*Wenn die Bedingung zutrifft, führe diese Befehle (A) aus.*/
    //Befehle A, z. B. den Kreis weiß färben.

} else {
    /* Wenn die Bedingung nicht zutrifft, z. B. die Maustaste nicht geklickt
    führe diese Befehle (B) aus. */
    //Befehle B, z. B. den Kreis schwarz färben
}
```

Die Bedingung im **if-Block** wird auf Richtigkeit überprüft. Falls diese wahr ist wird **Befehl A** ausgeführt. Wenn die Bedingung nicht wahr ist, wird der **Befehl B** im **else-Block** ausgeführt.

Im folgenden Beispiel wird der Kreis weiß gefärbt, wenn der Mauszeiger gedrückt gehalten wird und sonst weiß:

```
if (mouseIsPressed) {

    fill(255)

} else {

    fill(0)

}
```

## RGB Farben

1. Weiß ist langweilig. Färbe den Kreis rot, wenn die Maus geklickt wird. Passe dazu den **fill**-Befehl an.
2. Verändere die Rot-, Grün- und Blau-Werte und schau was passiert.

### Befehle

fill(r,g,b)

Du kannst die Farbe des Kreises ganz einfach verändern. Dazu brauchst du den **fill**-Befehl. In den Klammern legst du die Füllfarbe fest. Wert zwischen 0 und 255 für Schwarz, Grau und Weiß, kannst du auch die **RGB-Farbcodierung** verwenden. RGB steht für **R**ot, **G**rün und **B**lau. Für jeden Kanal vergibst du wiederum einen Wert zwischen 0 und 255. So kannst du **alle Farben mischen**.

**Tipp:** Ein möglicher RGB-Code für Rot ist (255,0,0).

## Zufällige Farben

Nun wollen wir den Zufall entscheiden lassen, welche Farbe die Kreise annehmen. Konkret: Falls die Maustaste gedrückt ist, soll ein zufälliger Rot-Wert vergeben werden. Gehe dazu wie folgt vor:

1. Erstelle eine Variable r.
2. Vergebe der Variablen r einen zufälligen Wert zwischen 0 und 255. Nutze dazu den **random**-Befehl.
3. Ersetze den aktuellen Rot-Wert (255) durch r.

### Befehle

```
var "variablenName" = "grundWert"  
"variablenName" = random(untersterWert, obersterWert)  
fill(r,g,b)
```



http:// Der **random**-Befehl wählt einen zufälligen Wert. Du musst lediglich den Wertebereich definieren. Für einen zufälligen Farbwert zum Beispiel 0 bis 255 oder z. B. **random(200, 255)**.

## Variablen

Variablen sind Platzhalter, die beliebige Inhalte haben können. Sobald du etwas in den Variablen gespeichert hast, merkt sich der Computer den Wert. Du kannst später im Programm wieder darauf zugreifen oder den Wert verändern. Variablen sind also flexible Werte.

Wenn wir eine Variable erstellen nennt man das initialisieren. Wenn wir der Variablen einen Wert geben, nennt man das deklarieren.

### Beispiel

- In Zeile 1 definieren wir den Buchstaben "r" als Variable (var). Wir speichern den Wert "0" in der Variablen r.
- In Zeile 12 greifen wir auf die Variable zu und ändern den Wert auf einen zufälligen Wert zwischen 0 und 255.
- In Zeile 13 fügen wir für den Rot-Wert des Kreises den zufälligen Wert der Variablen ein.

```
1  var r = 0
2
3  function setup() {
4    createCanvas(400, 300)
5    background(0)
6  }
7
8  function draw() {
9    ellipse(mouseX, mouseY, 80, 80)
10
11    if(mouseIsPressed) {
12      r = random(0,255)
13      fill(r,0,0)
14    } else{
15      fill(0)
16    }
17  }
```





## Extra-Challenge



Bist du etwa schon fertig? Toll! Dann überlege dir nun, wie du dein Programm erweitern könntest.

Hier sind 2 Anregungen:

- Wenn die Maus geklickt wird, verändere auch den Grün-Wert und Blau-Wert des Kreises zufällig.
- Zeichne anstelle des Kreises ein Rechteck. Nutze dazu den **rect**-Befehl.

### Befehle

fill(r, g, b)

random(untersterWert, obersterWert)

rect(x, y, b, h)



Bist du etwa schon wieder fertig? Toll! Folgend eine weitere Anregung, wie du dein Programm erweitern kannst. Erstelle eine Animation. Gehe dazu wie folgt vor:

1. Erstelle eine Variable `x` mit dem Wert 0.
2. Erstelle ein Rechteck mit dem ***rect***-Befehl. Setze die *Variable* `x` als x-Koordinate.
3. Gebe dem Kreis eine zufällige Farbe. Nutze dazu den ***fill***-Befehl und die *Variablen* `r,g` und `b`.
4. Verändere den Wert der Variable `x`. Addiere immer 5 zum aktuellen Wert hinzu.

### Befehle

```
var "variablenName" = "grundWert"
```

```
rect(x, y, b, h)
```

```
fill(r, g, b)
```



## Super-Extra-Challenge



Hier findest du den gesamten Code mit Erklärungen.

1. Gehe den Code Schritt-für-Schritt durch. Hast du alles verstanden?
2. Tausche dich mit deinem Programmierpartner aus und erklärt euch gegenseitig das Programm.
3. Passt das Programm nach Belieben an. Fügt z.B. den Befehl ***background(0)*** ganz oben in die draw-Funktion ein und schaut, was passiert.

Lasst eurer **Kreativität** freien Lauf.