

Variablen sind Platzhalter für Werte (z.B. Zahlen, Farben, Texte). Der tatsächliche Wert einer Variablen kann während der Ausführung des Programms **gelesen und auch überschrieben** werden. Du kennst bereits die Variablen `mouseX` und `mouseY`, die von Processing automatisch mit der aktuellen Mausposition überschrieben werden. Auch die Platzhalter `mousePressed`, `keyPressed` und `key` sind Variablen.

Scratch



Variablen müssen vor der ersten Verwendung **deklariert** und mit einem **Bezeichner benannt** werden. In Java muss man dabei einmalig den **Datentyp** angeben. Dieser legt fest, welche Art von Werten in der Variablen gespeichert werden kann.

Variablen kann man während des Programms einen **neuen Wert zuweisen**. In Java steht links der Bezeichner der Variablen und rechts vom Gleichzeichen der neue Wert. Das kann auch eine Rechnung sein.

Du kannst den Wert deiner Variablen nun für Parameter oder Bedingungen verwenden.

Java

```
int position;
```

Bezeichner
Datentyp (int = Ganzzahl)

oder Deklaration mit Zuweisung:
`int position = 200;`

```
// Zuweisung: Setzt den Wert auf 50
position = 50;
```

```
// Zuweisung: Erhöht den Wert um 10
position = position + 10;
```

```
// Verwendung als Parameter
square(position, 50, 10);
```

```
// Verwendung in Bedingungen
if (position > 20) {
// ...
}
```

1 Der Sketch links verwendet die Variable `durchmesser`.

Sketch Kap4_A1

```
1
void setup() {
  2 size(400, 400);
}

void draw() {
  3 background(255);
  circle(200, 200, diameter);
  if (mousePressed) {
    diameter = diameter + 1;
  }
}
```

- Mache eine Vorhersage, wie sich der Sketch verhält.
- Damit der Sketch funktioniert, muss die Variable `durchmesser` zuerst **deklariert** werden. Da wir den Durchmesser in Pixel messen, ist der Datentyp `int` sinnvoll. Damit lassen sich ganze Zahlen speichern. Finde die richtige Stelle (1, 2 oder 3) für die Deklaration `int diameter = 10;` im Programmcode.
- Ergänze die Verzweigung, dass der Kreis kleiner wird, wenn die Maustaste nicht gedrückt ist.
- Ergänze weitere Verzweigungen, dass der Durchmesser nicht negativ werden kann und der Kreis nicht größer als der Sketch werden kann.

Sketch Kap4_A2

```
void setup() {
  size(500, 500);
}

void draw() {
  background(255);
  circle(250, 250, 50);
}
```

2 Ping-Pong-Animation

- Übernehme den Sketch und deklariere die Variable `circleX` (Datentyp `int`) als globale Variable. Ersetze den ersten Parameter in der `circle`-Anweisung mit dieser Variablen.
- Ergänze eine Anweisung im `draw`-Block, dass sich der Kreis in jedem Frame um einen Pixel nach rechts bewegt. Wie könnte man die Bewegung beschleunigen? Wie könnte man die Bewegungsrichtung umkehren? Entscheide dich für eine Richtung und Geschwindigkeit.
- Überprüfe mit einer Verzweigung, ob der Kreis am Rand angekommen ist. Setze ihn dann wieder auf die gegenüberliegende Seite.

Wahrheitswerte (Datentyp boolean)

Eine Variable mit dem Datentyp `boolean` kann nur die Werte `true` („wahr“) oder `false` („falsch“) speichern. Dieser Datentyp ist oft geschickt, wenn ein Zustand gespeichert werden soll.

Bsp.: In einem Sketch soll eine Zeichnung mit Linksklick an- und mit Rechtsklick ausgeschaltet werden.

In einer Variablen muss gespeichert werden, ob die Zeichnung gerade angezeigt wird. Wir nennen diese Variable `kreisAnzeigen` und vergeben den Datentyp `boolean`, da nur zwei Zustände möglich sind. Damit der Kreis zu Beginn zu sehen ist, wird der Startwert `true` vergeben.

Die `boolean`-Variable kann als Bedingung verwenden. Die Bedingung gilt als erfüllt, wenn die Variable den Wert `true` hat. Du kennst diese Schreibweise schon von den Variablen wie `mousePressed` – sie sind auch vom Typ `boolean`.

Wir benötigen nun noch Bedingungen der Variablen einen Wert zuzuweisen, wenn eine Maustaste gedrückt wird. Eine Möglichkeit dafür siehst du rechts.

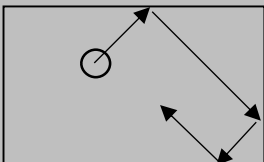
Beispiel mit Datentyp Boolean (Bsp1)

```
boolean kreisAnzeigen = true;

void setup() {
  size(200, 200);
}

void draw() {
  background(255);
  if (kreisAnzeigen) {
    circle(100, 100, 100);
  }

  if (mousePressed && mouseButton == LEFT) {
    kreisAnzeigen = true;
  }
  if (mousePressed && mouseButton == RIGHT) {
    kreisAnzeigen = false;
  }
}
```



„DVD-Logo-Animation“

Bei dieser Animation bewegt sich die Form gleichzeitig in x- und y-Richtung und prallt an jedem Rand ab.

Wovon hängt es ab, ob der Kreis jemals genau in die Ecke gelangt?

3 Ping-Pong-Animation (Fortsetzung von Aufgabe 2)

Passen die Animation des Kreises so an, dass er vom linken und rechten Rand abprallt. Du benötigst dazu eine Variable `nachRechts` vom Typ `boolean`, die festlegt, in welche Richtung sich der Kreis gerade bewegen soll.

4 Erstelle ein Sketch mit der „DVD-Logo-Animation“. (siehe links)

Bonus: Informiere dich über die `image`-Anweisung in Processing und ersetze den Kreis durch ein DVD-Logo.

5 Übungs-Sketches mit Variablen

Siehe Online-Vorschau. Bei allen Sketchen musst du Klicken!

Variablen machen Rechnungen übersichtlicher!

Beispiel mit Variablen zur Übersichtlichkeit (Bsp2)

```
void setup() {
  size(100, 300);
}

void draw() {
  int kreisY;
  color farbe;

  // Oberes Segment (rot)
  if (mouseY < 100) {
    kreisY = 50;
    farbe = color(255, 0, 0);
  }
  // Mittleres Segment (gelb)
  else if (mouseY < 200) {
    kreisY = 150;
    farbe = color(255, 255, 0);
  }
  // unteres Segment (grün)
  else {
    kreisY = 250;
    farbe = color(0, 255, 0);
  }
}
```

Mit dem Datentyp `color` kann man Farbwerte speichern.

Bisher hast du Rechnungen direkt in den Zeichenanweisungen notiert. Bei komplizierten Programmen kann man den Programmcode übersichtlicher gestalten, indem man zuerst die benötigten Koordinaten und Farben bestimmt und in Variablen speichert und erst dann alle Zeichenanweisungen ausführt.

Bsp.: Ampel

Im linken Teil werden noch keine Zeichnungen durchgeführt, sondern anhand der Mausposition die Variablen `kreisY` und `farbe` mit passenden Werten befüllt. Beim Zeichnen unten werden dann die Variablen `kreisY` und `farbe` als Parameter verwendet.

Die Programmteile sind mit hilfreichen Kommentaren versehen.

```
// Zeichnen: Weiße Kreise
fill(255);
circle(50, 50, 100);
circle(50, 150, 100);
circle(50, 250, 100);
// Zeichnen: Farbiger Kreis der Ampel
fill(farbe);
circle(50, kreisY, 80);
}
```