

Unsere bisherigen Sketche bestehen aus nur einer Sequenz, deren Anweisungen von oben nach unten ausgeführt werden. Bei einem interaktiven Sketch können während der Ausführung des Programms weitere Zeichnungen ergänzt werden oder eine komplett neue Zeichnung erstellt werden.

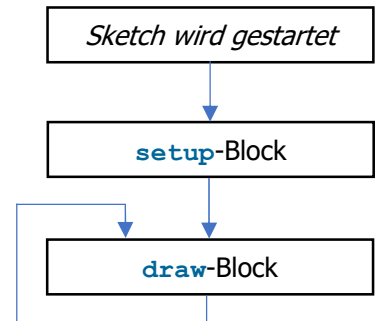
Die Grundstruktur eines interaktiven Sketches ist in Processing vorgegeben:

```
void setup() {
  size(400, 600);
  noStroke();
  fill(255, 0, 0);
}

void draw() {
  circle(mouseX, mouseY, 20);
}
```

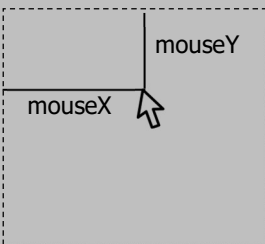
Der **setup**-Block wird beim Programmstart genau einmal ausgeführt. Hier werden Einstellungen vorgenommen, die den gesamten Sketch betreffen.

Der **draw**-Block wird während der Ausführung automatisch immer wieder wiederholt. Ein solcher Durchlauf wird auch „Frame“ genannt.



`size(breite, hoehe);`

Die Anweisung `size` kann nur im `setup`-Block verwendet werden und legt die Fenstergröße des Sketches fest.



Die Variablen (Platzhalter) `mouseX` und `mouseY` enthalten die aktuelle Position des Mauszeigers im Programmfenster.

1 a) Übernehme den Sketch oben und starte ihn.

b) Die Anweisung `background(255);` übermalt das gesamte Fenster weiß. Die Wirkung dieser Anweisung hängt von der Position im Programmcode ab. Probiere nacheinander die *markierte Position* ① bis ④ aus.

Was passiert jeweils?

c) Erweitere das Programm um weitere Zeichenanweisungen mit den Variablen `mouseX` und `mouseY`.

Sketch Kap2\_A1

```
void setup() {
  size(400, 600);
  noStroke();
  fill(255, 0, 0);
  ①
}

④

void draw() {
  ②
  circle(mouseX, mouseY, 20);
  ③
}
```

2 Im Beispiel oben hast du bereits die Variablen `mouseX` und `mouseY` kennengelernt.

Erstelle jeweils einen interaktiven Sketch bei dem sich die *markierte Position* mit dem Mauszeiger mitbewegt. Verwende die Vorlage rechts und ergänze an der markierten Stelle .

Sketch Kap2\_A2

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

void draw() {
  background(255);
  
}
```

a)

b)

c)

An der Position des Mauszeigers ist ein Kreis. Von dort geht eine Linie senkrecht nach unten.

d)

Die Linie verbindet die Position des Mauszeigers mit der Fenstermitte. An beiden Stellen ist ein Kreis.

e)

Die Form bewegt sich mit dem Mauszeiger mit.  
Tipp: Verwende auch negative Werte für die Breite und Höhe bei der `rect`-Anweisung.

## Grundrechenarten mit Werten und Variablen

Bisher haben wir für **Parameter** von Anweisungen eine **feste Zahl** oder eine **Variable** (`mouseX`, `mouseY`), die von Processing zur Verfügung gestellt wird, verwendet. An allen solchen Stellen kann man auch **Rechnungen** verwenden.

Bsp.: 

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

void draw() {
  background(255);
  circle(mouseX - 36, mouseY - 36, 100);
}
```

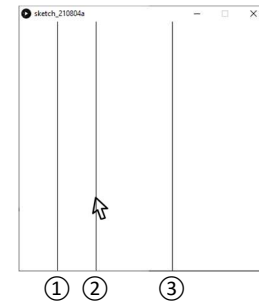


Der Mittelpunkt des Kreises ist durch die Rechnung von der (`mouseX` | `mouseY`) Position etwas nach links oben verschoben.

Bsp.: 

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

void draw() {
  background(255);
  line(mouseX / 2, 0, mouseX / 2, 400); ①
  line(mouseX, 0, mouseX, 400);          ②
  line(mouseX * 2, 0, mouseX * 2, 400);    ③
}
```



- ① Linie an der halben `mouseX`-Position
- ② Linie an der `mouseX`-Position
- ③ Linie an der doppelten `mouseX`-Position

## Grundrechenarten in Java

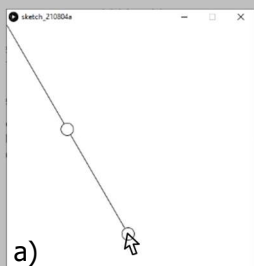
(Operatoren + - \* und /)

Addition  
 $a + b$

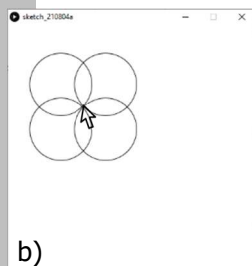
Subtraktion  
 $a - b$

Multiplikation  
 $a * b$

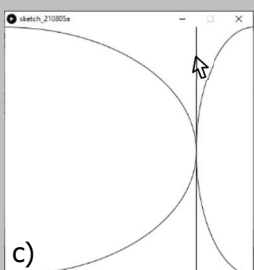
Division  
 $a / b$



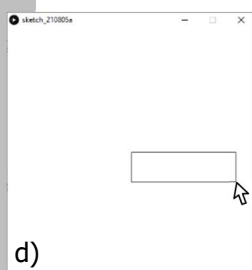
a)



b)



c)



d)

3 Erstelle jeweils einen passenden Sketch.

Verwende die Vorlage Sketch Kap2\_A2.

- a) Eine Linie von der oberen linken Ecke zum Mauszeiger. In der Mitte der Linie und beim Mauszeiger ist ein Kreis.
- b) Die Figur mit den vier Kreisen bewegt sich mit dem Mauszeiger mit.
- c) Die Position des Mauszeigers legt eine senkrechte Linie fest. Die zwei Ellipsen verändern dabei ihre Breite, dass sie bis zur senkrechten Linie reichen.
- d) Das Rechteck hat eine Ecke im Mittelpunkt des Sketches und eine weitere Ecke bewegt sich mit dem Mauszeiger.

Weitere Teilaufgaben findest du bei den Lösungssketchen.

4

Betrachte die Programmcodes und treffe eine Vorhersage, wie sich der Sketch verhalten wird. Überprüfe anschließend deine Vermutung.

Sketch Kap2\_A4a

```
void setup() {
  size(255, 255);
}

void draw() {
  background(mouseX);
}
```

Sketch Kap2\_A4b

```
void setup() {
  size(255, 255);
  fill(0);
}

void draw() {
  circle(mouseX, mouseY, 3);
}
```

Sketch Kap2\_A4c

```
void setup() {
  size(255, 255);
  noStroke();
}

void draw() {
  fill(mouseX, mouseY, 0);
  circle(mouseX, mouseY, 3);
}
```