BRÜCKENKURS PROGRAMMIEREN - FIONA NÜESCH

REPETITION

ARRAYS UND SCHLEIFEN

frei wählbarer Name — Anzahl Elemente, die die Sammlung beinhaltet (ganzzahlig)

```
int[] arrayName = new int[10];
```

Datentyp — Ein Referenztyp wird über die Anweisung **new** Erzeugt

```
for(int i = 0; i < arrayName.length; i++){
    arrayName[i] = i*2;
}
```

VERZWEIGUNGEN UND BEDINGUNGEN

```
1  int a = 10;
2  int b = 5;
3  int c = 30;
4
5  if( !( a > b && c != a ) || c > a ){
6    println("komplizierte Bedingung erfüllt");
7  }else{
8    println("nicht erfüllt");
9  }
```

METHODEN

```
1
2  void setup(){
3
4    int result = line(2);
5
6    printWords("my", "name", "is", "fiona");
7
8  }
```

Methodenaufruf

```
12
13 int line(int x){
14
15   int y = 2 * x + 1;
16
17   return y;
18 }
19
20
21 void printWords(String word1, String word2, String word3, String word4){
22
23   println( word1 + word2 + word3 +word4 );
24
25 }
26
```

Methodendefinition

Methoden Definition

Rückgabe Typ
(Datentyp des Outputs)

Methodenname

Methodenparameter
(Input mit Typ und Name)

```
int line(int x){

    int y = 2 * x + 1;

    return y;
}
```

Methoden Code Block

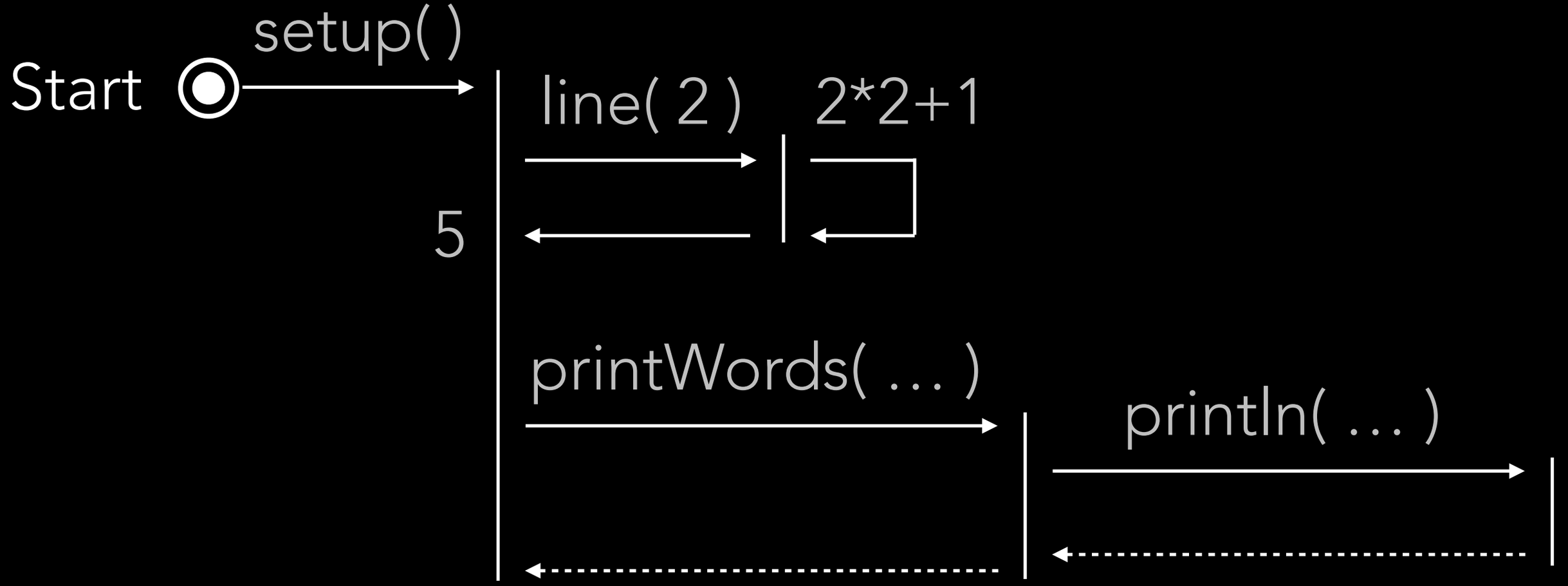
neuer Datentyp: **void** für keinen Rückgabewert (keine return-Anweisung)

return Statement

- Definiert welche Variable zurückgegeben wird.
- Muss dem Typ des Rückgabewerts entsprechen.
- Beendet die Methode.

```
1
2  void setup(){
3
4    int result = line(2);
5
6    printWords("my", "name", "is", "fiona");
7
8  }
```

```
12
13 int line(int x){
14
15   int y = 2 * x + 1;
16
17   return y;
18 }
19
20
21 void printWords(String word1, String word2, String word3, String word4){
22
23   println( word1 + word2 + word3 +word4 );
24
25 }
26
```



```
void setup(){
    int myX = 2;
    int result = line(myX);
    printResult("MY", "RESULT", "IS", result);
}

int line(int x){
    int y = 2*x+0;
    return y;
}

void printResult(String word1, String word2,
String word3, int result){
    println(word1, word2, word3, result);
}
```

Start

```
void setup(){
    int myX = 2;
    int result = line(myX);
    printResult("MY", "RESULT", "IS", result);
}

int line(int x){
    int y = 2*x+0;
    return y;
}

void printResult(String word1, String word2,
String word3, int result){
    println(word1, word2, word3, result);
}
```

```
void setup(){
    int myX = 2;
    int result = line(myX);
    printResult("MY", "RESULT", "IS", result);
}

int line(int x){
    int y = 2*x+0;
    return y;
}

void printResult(String word1, String word2,
String word3, int result){
    println(word1, word2, word3, result);
}
```

```
void setup(){
    int myX = 2;
    int result = line(myX);
    printResult("MY", "RESULT", "IS", result);
}

int line(int x){
    int y = 2*x+0;
    return y;
}

void printResult(String word1, String word2,
String word3, int result){
    println(word1, word2, word3, result);
}
```

```
void setup(){
    int myX = 2;
    int result = line(myX);
    printResult("MY", "RESULT", "IS", result);
}

int line(int x){
    int y = 2*x+0;
    return y;
}

void printResult(String word1, String word2,
String word3, int result){
    println(word1, word2, word3, result);
}
```

```
void setup(){
    int myX = 2;
    int result = line(myX);
    printResult("MY", "RESULT", "IS", result);
}

int line(int x){
    int y = 2*x+0;
    return y;
}

void printResult(String word1, String word2,
String word3, int result){
    println(word1, word2, word3, result);
}
```

```
void setup(){
    int myX = 2;
    int result = line(myX);
    printResult("MY", "RESULT", "IS", result);
}

int line(int x){
    int y = 2*x+0;
    return y;
}

void printResult(String word1, String word2,
String word3, int result){
    println(word1, word2, word3, result);
}
```

```
void setup(){
    int myX = 2;
    int result = line(myX);
    printResult("MY", "RESULT", "IS", result);
}

int line(int x){
    int y = 2*x+0;
    return y;
}

void printResult(String word1, String word2,
String word3, int result){
    println(word1, word2, word3, result);
}
```

5 ←

result:5

```
void setup(){
    int myX = 2;
    int result = line(myX);
    printResult("MY", "RESULT", "IS", result);
}

int line(int x){
    int y = 2*x+0;
    return y;
}

void printResult(String word1, String word2,
String word3, int result){
    println(word1, word2, word3, result);
}
```

```
void setup(){
    int myX = 2;
    int result = line(myX);
    printResult("MY", "RESULT", "IS", result);
}

int line(int x){
    int y = 2*x+0;
    return y;
}

void printResult(String word1, String word2,
String word3, int result){
    println(word1, word2, word3, result);
}
```

```
void setup(){
    int myX = 2;
    int result = line(myX);
    printResult("MY", "RESULT", "IS", result);
}

int line(int x){
    int y = 2*x+0;
    return y;
}

void printResult(String word1, String word2,
String word3, int result){
    println(word1, word2, word3, result);
}
```

```
void setup(){
    int myX = 2;
    int result = line(myX);
    printResult("MY", "RESULT", "IS", result);
}

int line(int x){
    int y = 2*x+0;
    return y;
}

void printResult(String word1, String word2,
String word3, int result){
    println(word1, word2, word3, result);
}
```

```
void setup(){
    int myX = 2;
    int result = line(myX);
    printResult("MY", "RESULT", "IS", result);
}

int line(int x){
    int y = 2*x+0;
    return y;
}

void printResult(String word1, String word2,
String word3, int result){
    println(word1, word2, word3, result);
}
```

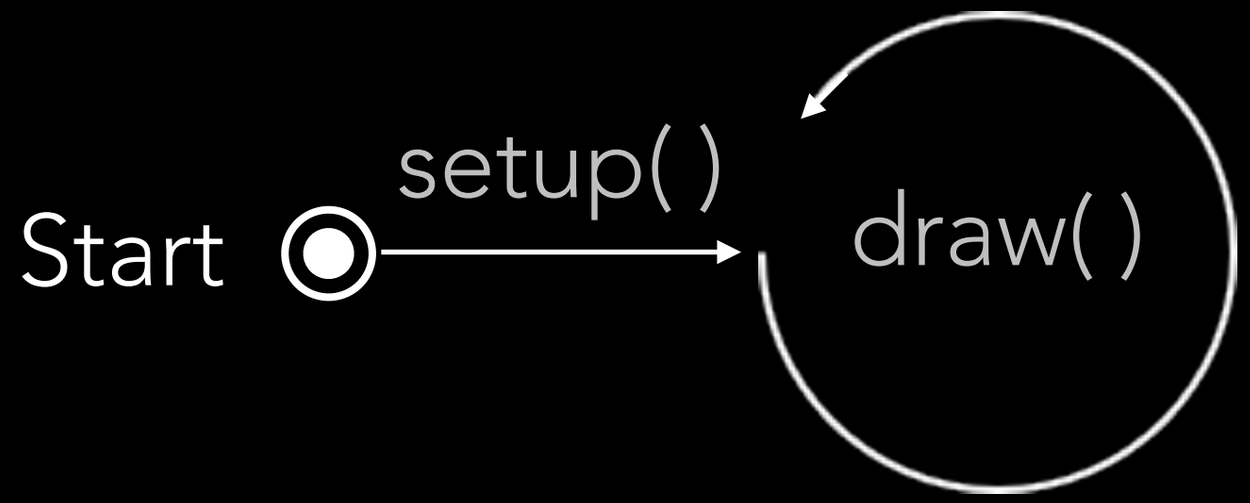
```
void setup(){
    int myX = 2;
    int result = line(myX);
    printResult("MY", "RESULT", "IS", result);
}

int line(int x){
    int y = 2*x+0;
    return y;
}

void printResult(String word1, String word2,
String word3, int result){
    println(word1, word2, word3, result);
}
```

```
void setup(){
    int myX = 2;
    int result = line(myX);
    printResult("MY", "RESULT", "IS", result);
}
```

```
void draw(){
    line(random(1, 10));
}
```

```
int line(int x){
    int y = 2*x+0;
    return y;
}

void printResult(String word1, String word2,
String word3, int result){
    println(word1, word2, word3, result);
}
```

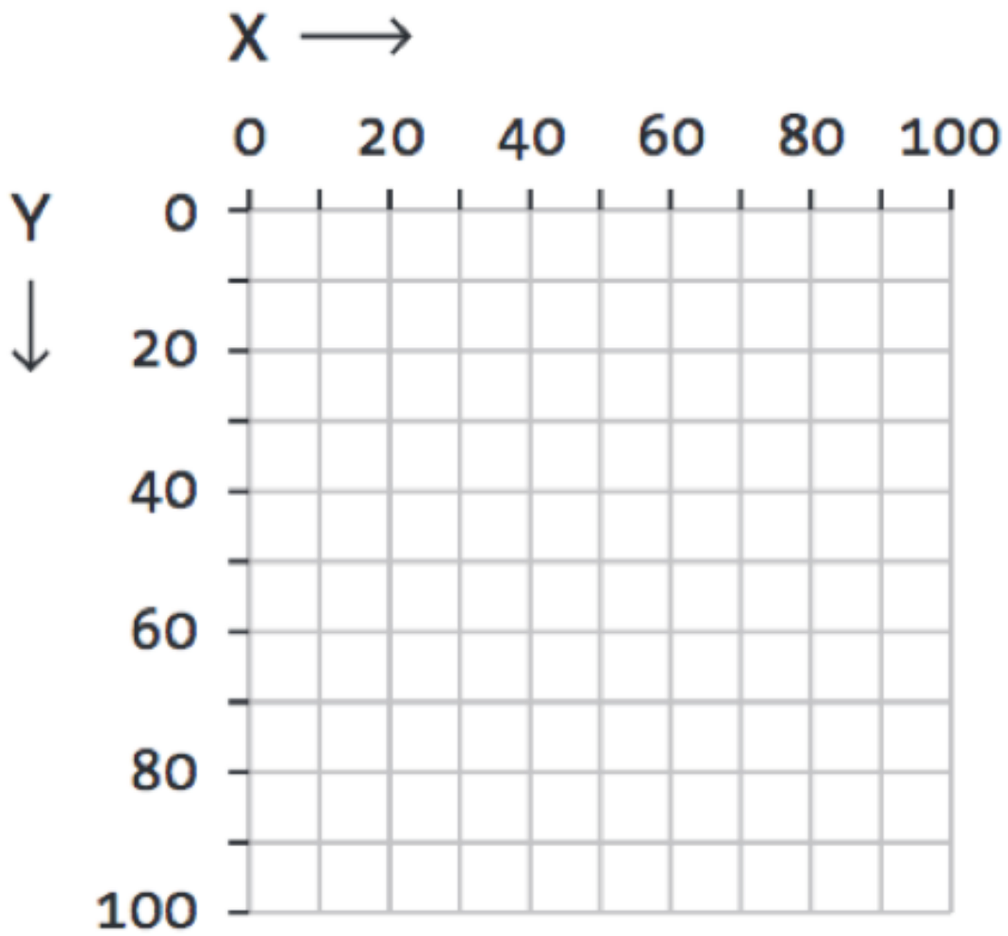
PROCESSING

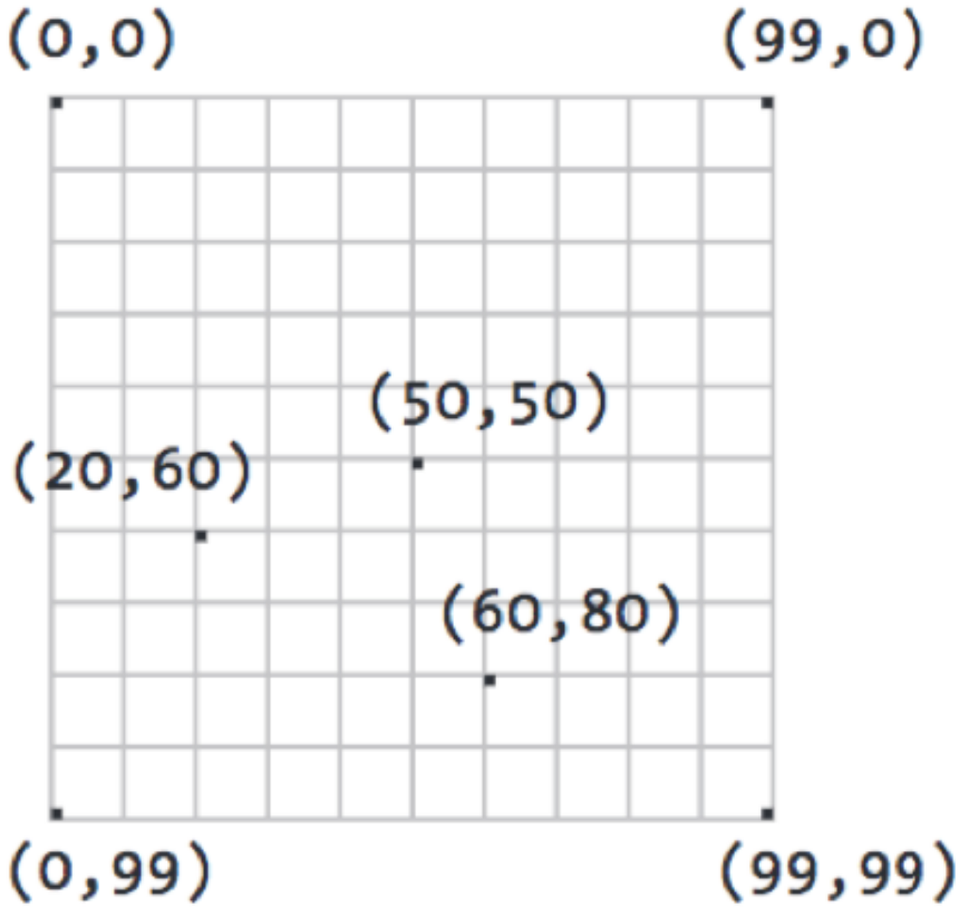


sketch_180613a

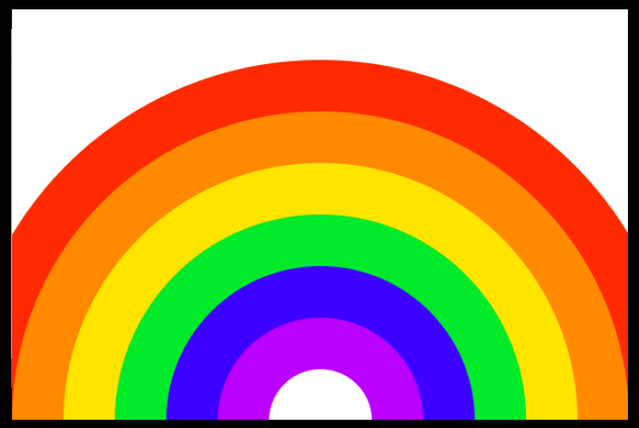
```
1  void setup(){
2    size(300, 300);
3  }
4
5  void draw(){
6    background(0);
7    text("Hello World!", 100, 100);
8    fill(0, 102, 153);
9  }
10
```

Das Koordinatensystem





Grafik



```
1  void setup(){
2    size(600, 400);
3    background(255);
4    noStroke();
5
6    fill(255, 45, 0);
7    ellipse(300, height, 700, 700);
8    fill(255, 137, 7);
9    ellipse(300, height, 600, 600);
10   fill(255, 226, 3);
11   ellipse(300, height, 500, 500);
12   fill(11, 232, 60);
13   ellipse(300, height, 400, 400);
14   fill(62, 12, 255);
15   ellipse(300, height, 300, 300);
16   fill(184, 0, 255);
17   ellipse(300, height, 200, 200);
18   fill(255);
19   ellipse(300, height, 100, 100);
20 }
```

Display Window Dimension 600 x 400 px

Display Window Hintergrundfarbe : weiss

alle folgenden Figuren ohne Outline

fülle folgende Figur mit rot

zeichne einen Kreis
Mittelpunktkoordinaten: (300, height)
und Länge, Breite mit 700 px

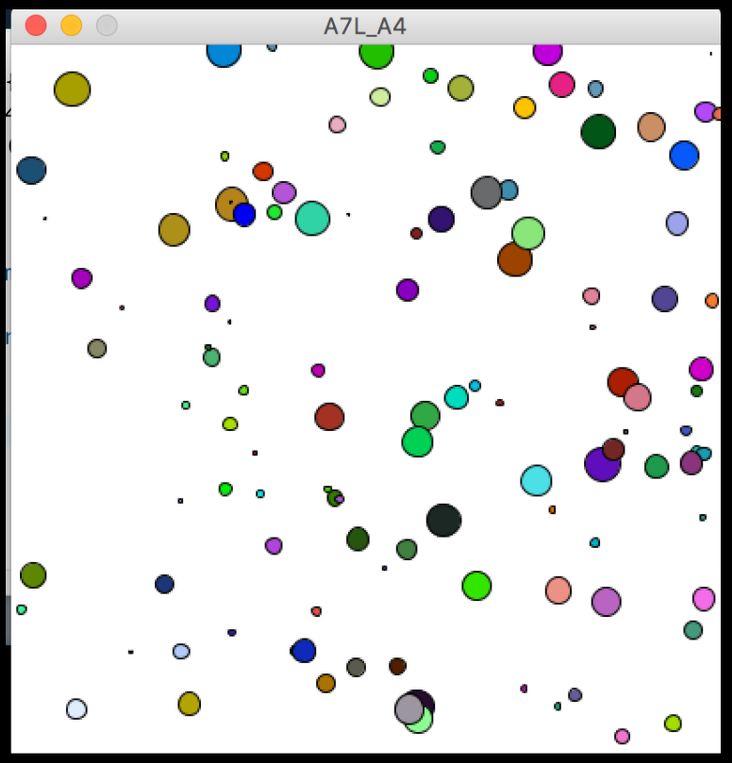
Grafik über die Zeit

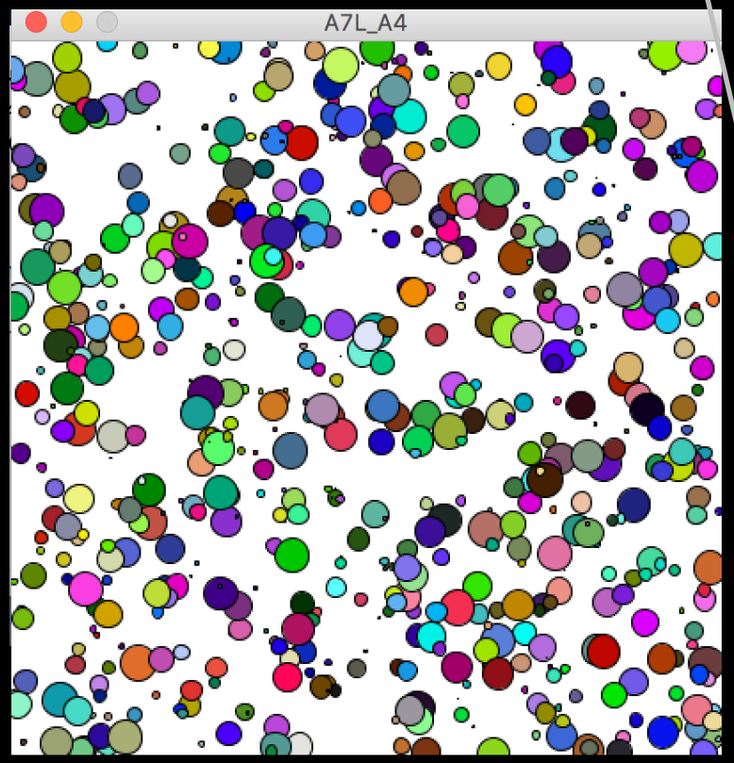
```
1
2  void setup(){
3    size(400, 400);
4    background(255);
5  }
6
7  void draw(){
8    fill(random(255), random(255), random(255));
9    float diam = random(20);
10   ellipse(random(width), random(height), diam, diam);
11 }
```

füllt die folgende Figur für jeden draw Durchgang mit einer zufälligen Farbe

generiert für jeden draw Durchgang eine zufällige Zahl, die als Kreisdurchmesser verwendet wird

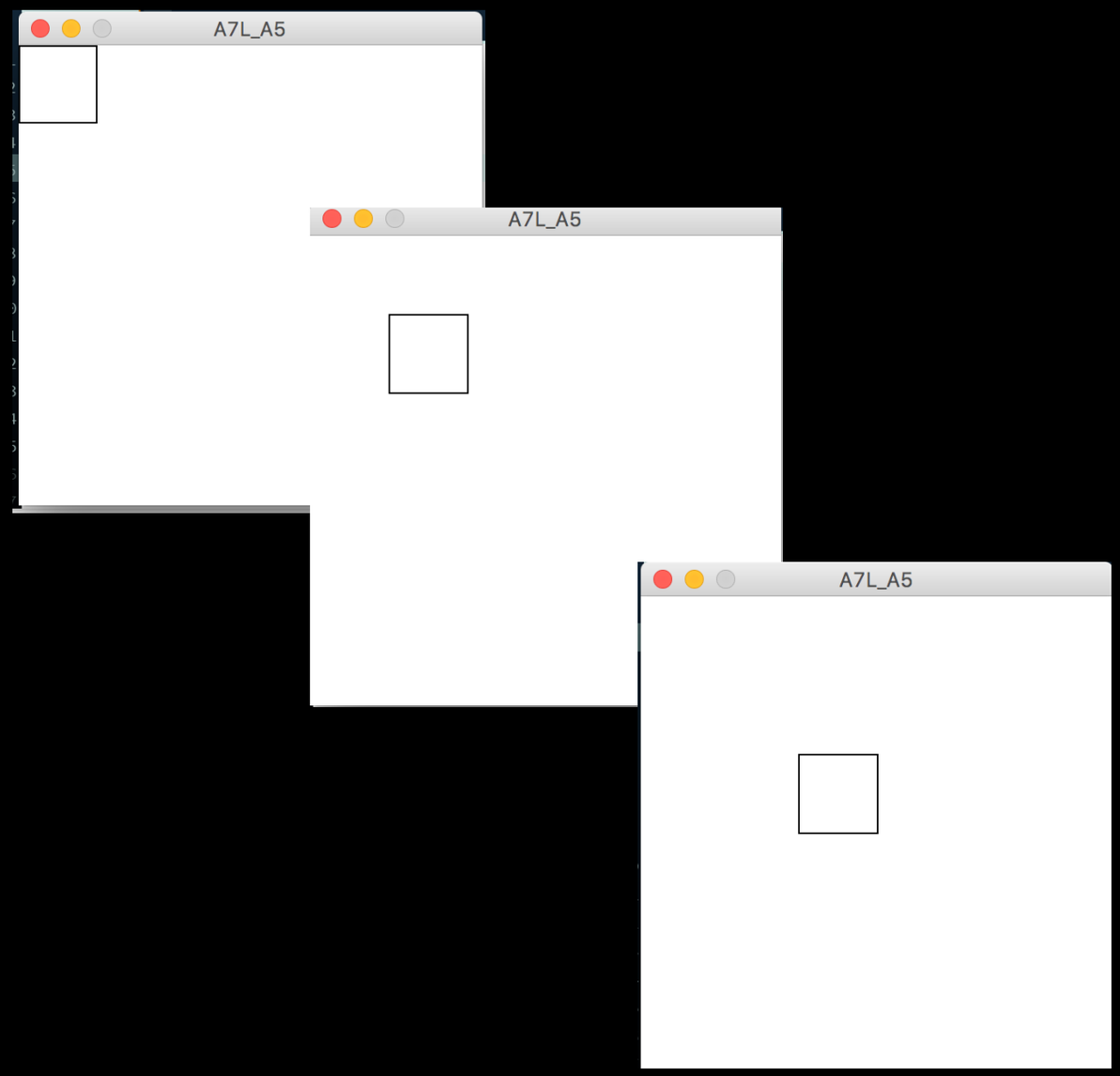
zeichnet für jeden draw Durchgang einen Kreis an zufälliger Stelle





Animation

```
1  int x = 0;
2  int y = 0;
3
4  void setup(){
5    size(500, 500);
6  }
7
8  void draw(){
9   background(255);
10  rect(x,y,50,50);
11  x += 5;
12  y += 5;
13 }
```

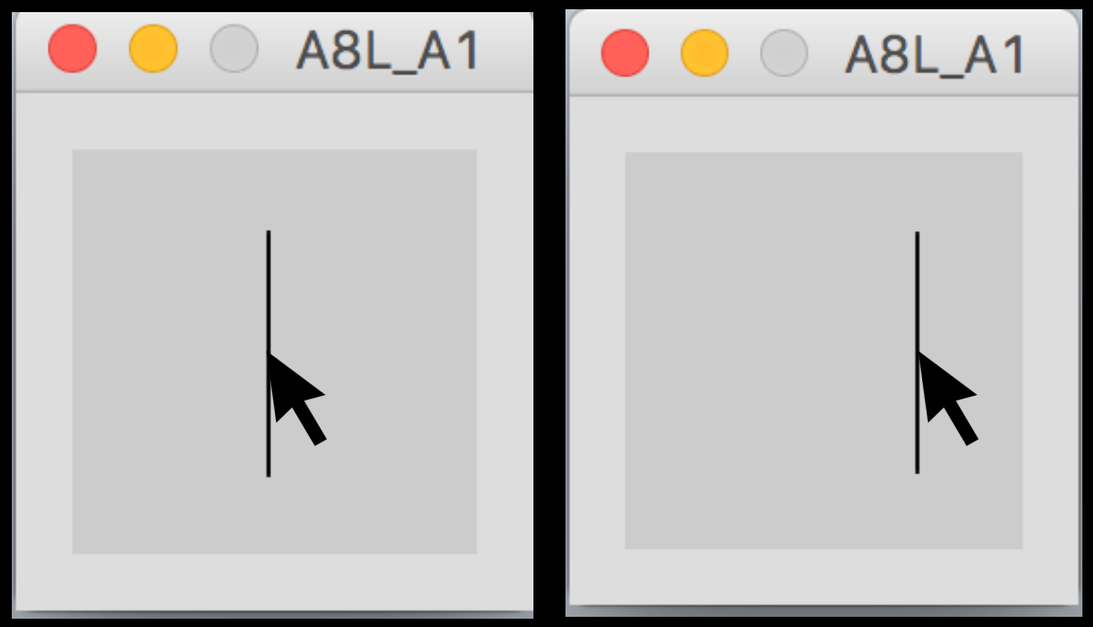


erhöht für jeden draw Durchgang x, y Wert um 5, so dass das Rechteck beim nächsten Durchgang um 5 px nach unten und nach rechts verschoben gezeichnet wird. Da der Hintergrund auch immer neu gezeichnet wird, entsteht eine Animation.

User Input - Maus

mouseX und **mouseY** sind globale Variablen von processing die jeweils die aktuellen Mauskoordinaten enthalten

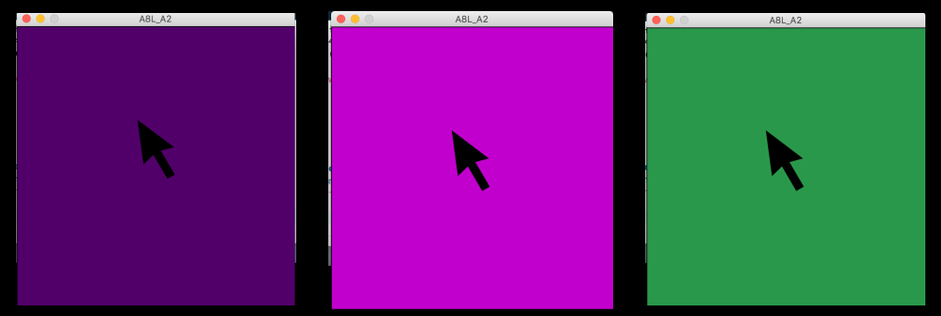
```
void draw() {
  background(204);
  line(mouseX, 20, mouseX, 80);
}
```



void mousePressed wird von processing aufgerufen, wenn der Benutzer mit der Maus klickt.

mousePressed wird nur aufgerufen wenn draw() vorhanden ist. Auch wenn wir da nichts machen

```
void setup(){
  size(400, 400);
  background(0);
  fill(0);
  rect(0,0, width, height);
}

void draw(){
}


void mousePressed(){
  fill(random(255), random(255), random(255));
  rect(0,0,width,height);
}
```



User Input - Keyboard

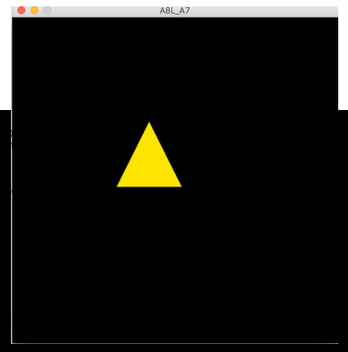
void keyPressed wird von processing aufgerufen, wenn der Benutzer eine Taste klickt.

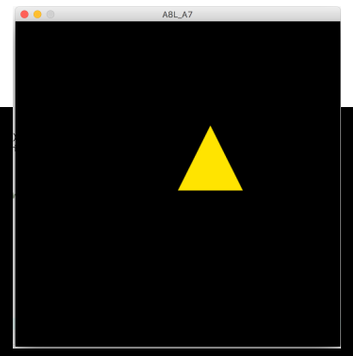
wird nur aufgerufen wenn draw() vorhanden ist.

key und **keyCode** sind globale Variablen von processing die jeweils die aktuelle Tasteneingabe enthalten

```
void setup(){
  background(0);
}

void draw(){
}

void keyPressed(){
  background(0);
  text(key, 50, 50);
}
```


```
int x;
int y;
int size = 50;

void setup(){
  size(500, 500);
  x = width/2;
  y = height/2;

  noStroke();
}

void draw(){
  background(0);
  fill(255, 226, 3);
  triangle( x, y-size, x+size, y+size, x-size, y+size );
}

void keyPressed(){
  if(keyCode == DOWN){
    y += 10;
  }else if(keyCode == UP){
    y -= 10;
  }else if(keyCode == LEFT){
    x -= 10;
  }else if(keyCode == RIGHT){
    x += 10;
  }
}
```





Übung 12 Repetition

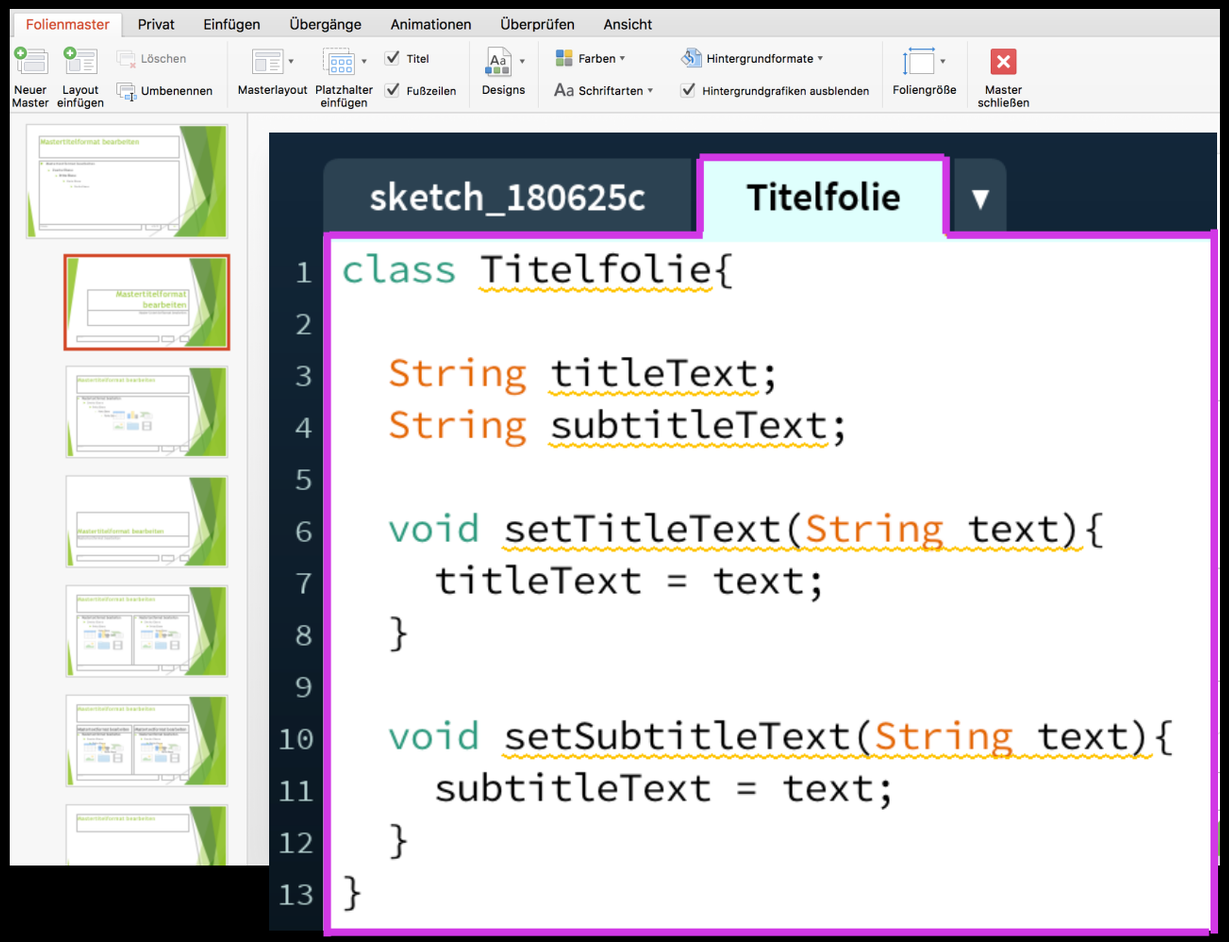
„Repetition.“

–CHRISTIAN BAUER

KLASSEN

ein Template

aus dem wir ein Objekt instanziieren können




```
void setup(){

  Titelfolie folie1 = new Titelfolie();

}
```

Objekt folie1

class KLASSENNAMEN {
 Attribute (class member): in Klasse gültige globale Variablen, speichern Zustand
 Methoden: definieren Funktionalitäten
}

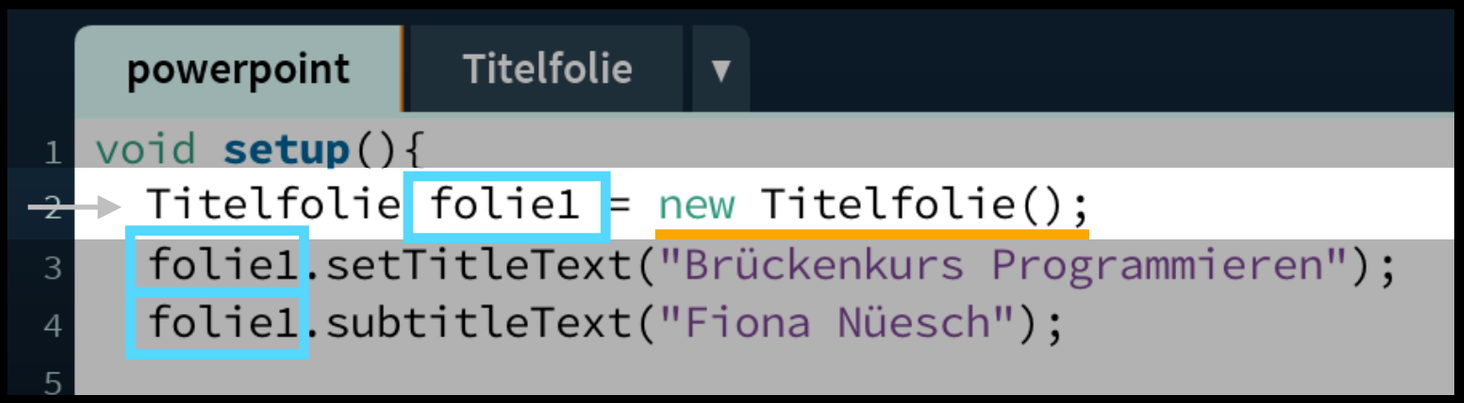
Klassen und Objekte verwenden

sketch_180625c | Titelfolie

```
class Titelfolie{

  String titleText;
  String subtitleText;

  void setTitleText(String text){
    titleText = text;
  }

  void setSubtitleText(String text){
    subtitleText = text;
  }
}
```

```
;
grammieren");
```

new Titelfolie()



neues Objekt vom Typ Titelfolie

new Titelfolie()

powerpoint | Titelfolie

```
1  void setup(){
2    Titelfolie folie1 = new Titelfolie();
3    folie1.setTitleText("Brückenkurs Programmieren");
4    folie1.subtitleText("Fiona Nüesch");
5
```

benutze auf diesem Objekt die Methode setTitleText() um den Zustand zu ändern

setTitleText(„Brück…)

```
3    String titleText;
4    String subtitleText;
5
6    void setTitleText(String text){
7      titleText = text;
8    }
9
10   void setSubtitleText(String text){
11     subtitleText = text;
12   }
```

setTitleText(„Brück…)

```
powerpoint | Titelfolie
1  void setup(){
2    Titelfolie folie1 = new Titelfolie();
3    folie1.setTitleText("Brückenkurs Programmieren");
4    folie1.subtitleText("Fiona Nüesch");
5
```

subtitleText(„Fiona…“)

benutze auf diesem Objekt die Methode subtitleText() um den Zustand zu ändern

```
3    String titleText;
4    String subtitleText;
5
6    void setTitleText(String text){
7      titleText = text;
8    }
9
10   void setSubtitleText(String text){
11     subtitleText = text;
12   }
```

```
powerpoint | Titelfolie
1  void setup(){
2    Titelfolie folie1 = new Titelfolie();
3    folie1.setTitleText("Brückenkurs Programmieren");
4    folie1.subtitleText("Fiona Nüesch");
5
```

new Titelfolie()

setTitleText(„Brück…)

subtitleText(„Fiona…“)

```
1 void setup(){
2   int i = 1;
3   int j = 2;
4   i = i+j;
5   j = i*2;
6 }
```

powerpoint | Titelfolie ▾

```
1  void setup(){
2    Titelfolie folie1 = new Titelfolie();
3    folie1.setTitleText("Brückenkurs Programmieren");
4    folie1.subtitleText("Fiona Nüesch");
5
6    Titelfolie folie2 = new Titelfolie();
7    folie2.setTitleText("Babedi bubedi");
8    folie2.subtitleText("WM ohne Italien");
9  }
10
```

Objekt folie1

Objekt folie2

KLASSEN KONSTRUKTOREN

- spezielle Methode die verwendet wird um ein **Objekt zu erzeugen/instanziieren**
- wenn wir keinen selber schreiben wird immer der default constructor verwendet
- ist dazu da, den Grundzustand des Objektes zu setzen

Default Constructor

```
Titelfolie title1 = new Titelfolie();
```

Name der Klasse, keine Parameter

```java
class Titelfolie{

  // Klassen Attribute
  String titleText;
  String subtitleText;

  // Konstruktor
  Titelfolie(String title, String subtitle){
    this.titleText = title;
    this.subtitleText = subtitle;
  }

  Titelfolie(String title){
    this.titleText = title;
  }

  // Klassen Funktionalitäten
  void setTitleText(String text){
```

- wird wie eine Methode aber ohne Rückgabe definiert.
- Der Name entspricht immer dem Klassennamen.
- Pro Klasse können mehrere verschiedene definiert werden
- -> verschieden heisst sie müssen sich bei den Parametern unterscheiden

```java
folie1 = new Titelfolie("Babedi Bubedi", "WM ohne Italien");
```

wird wie eine Methode benutzt, nur das die Rückgabe immer ein neues Objekt vom genannten Typ ist.

Mit einer Klasse definieren wir einen neuen Datentyp

```
Titelfolie title1 = new Titelfolie();
```

```
Titelfolie folie1 = new Titelfolie("Babedi Bubedi", "WM ohne Italien"
```

Das heisst wir können einen Array erstellen

```
Titelfolie[] folien = new Titelfolie[10];
```

Datentyp

```
folien[0] = new Titelfolie("Title", "Sub");
```

Konstruktor Aufruf -> neues Objekt

Und generell wie Datentypen benutzen die wir schon kennen

powerpoint_example | Presentation | Titelfolie

```
1  class Presentation{
2    Titelfolie title;
3
4    Presentation(String title, String subtitle){
5      this.title = new Titelfolie(title, subtitle)
6    }
7
8    Presentation(Titelfolie title){
9      this.title = title;
10   }
11
12   Titelfolie getTitle(){
13    return title;
14   }
15 }
```

Konstruktor Aufruf -> ein Objekt wird erstellt

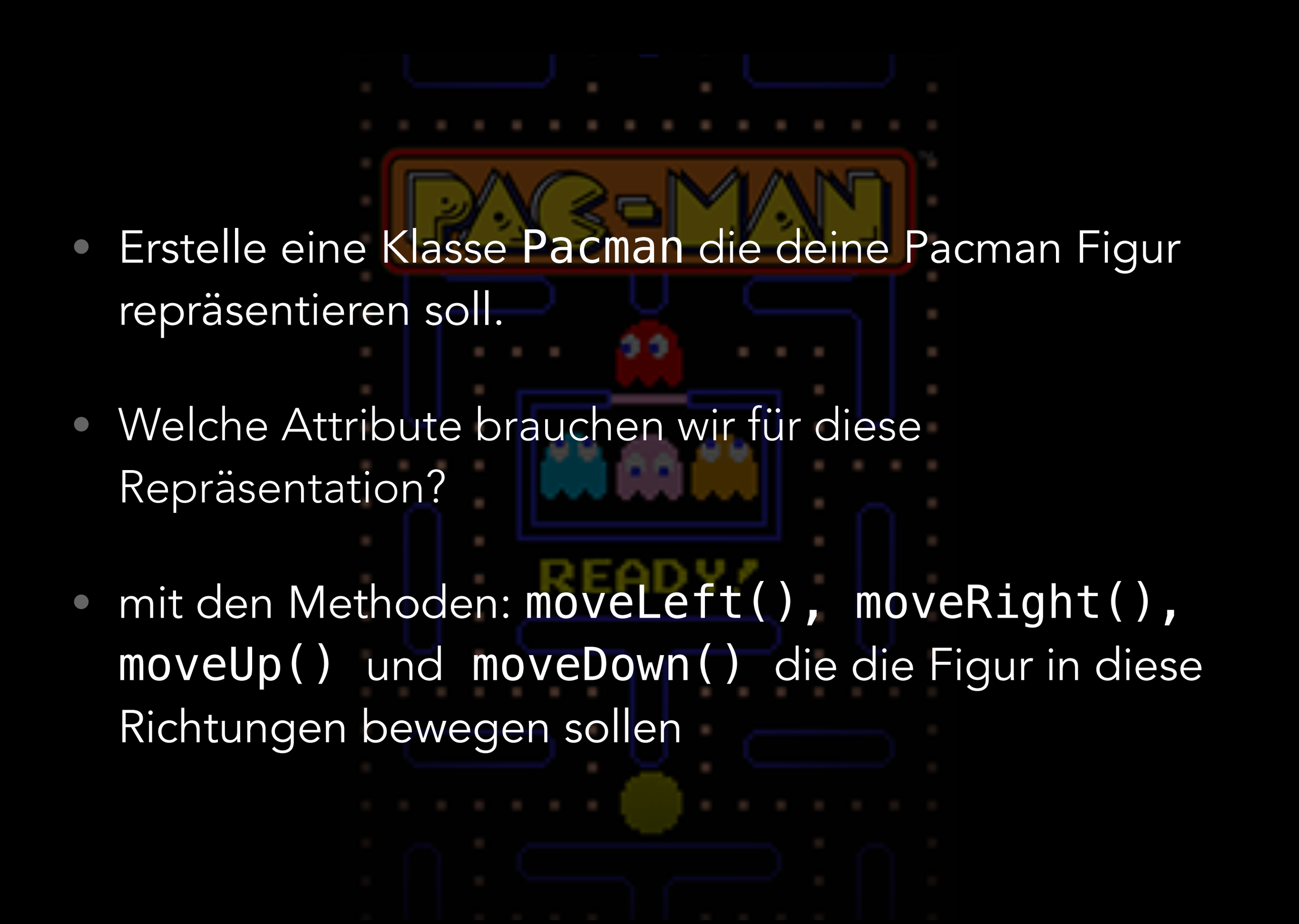
„Oft schiessen trifft das Ziel.“

– SPRICHWORT

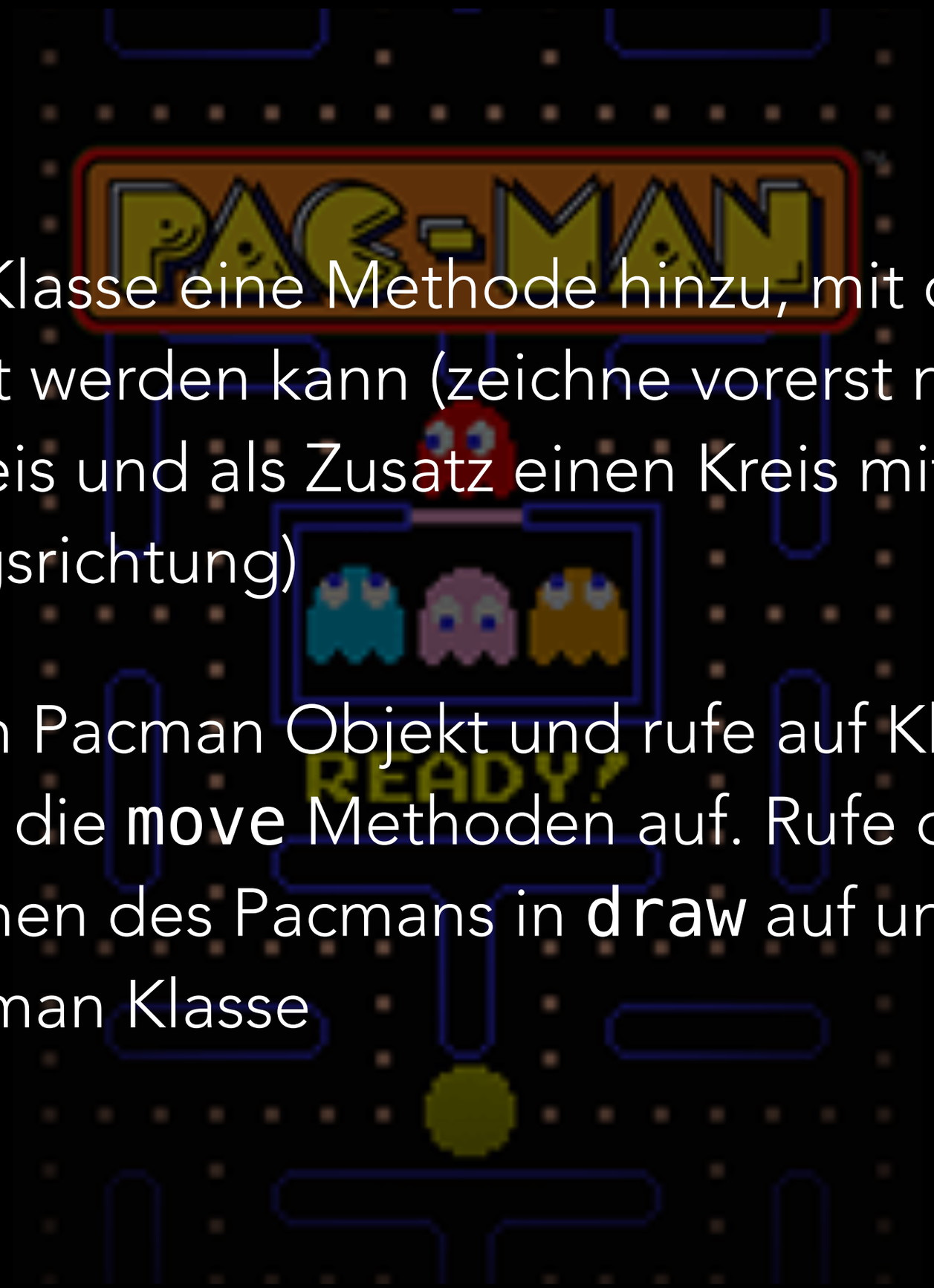
Was wir bis jetzt haben…

- Öffne den Sketch und füge da einen gelben Kreis hinzu, den man über die Tastatur bewegen kann.

- Füge 1 rotes Quadrat hinzu über die man mit dem Kreis nicht drüber fahren kann.



- Erstelle eine Klasse `Pacman` die deine Pacman Figur repräsentieren soll.
- Welche Attribute brauchen wir für diese Repräsentation?
- mit den Methoden: `moveLeft()`, `moveRight()`, `moveUp()` und `moveDown()` die die Figur in diese Richtungen bewegen sollen



- Füge der Klasse eine Methode hinzu, mit der die Figur gezeichnet werden kann (zeichne vorerst nur einen gelben Kreis und als Zusatz einen Kreis mit Öffnung in Bewegungsrichtung)

- Erstelle ein Pacman Objekt und rufe auf Klick der Pfeiltasten die `move` Methoden auf. Rufe die Methode zum zeichnen des Pacmans in `draw` auf und teste so deine Pacman Klasse

- Erstelle eine Klasse `Obstacle` welche die „Wände“ auf dem Spielfeld repräsentieren soll.
- Erstelle einen Konstruktor mit dem man die Position, Länge und Orientierung des Obstacles initialisieren kann. (Die Breite soll immer gleich sein)
- Erstelle eine Methode, die ein Obstacle als Rechteck mit rotem Rand zeichnet.
- Erstelle eine Methode `isTouched(float x, float y)` welche true zurück gibt wenn die x,y Koordinaten das Obstacle berühren oder sogar drüber sind und false wenn nicht

- Erzeuge 10 Obstacle Objekte mit zufälliger Position und Länge

- Prüfe bei jedem Pfeilklick des Benutzers nun ob du die Pacman Figur in diese Richtung bewegen kannst oder ob sie da eine Wand berühren würde. Dann soll sie sich nicht bewegen. (bzw. wieder zurück bewegen) Ergänze dafür die Pacman Klasse mit einer Methode `Point getPosition()` und `float getSize();`
  - Tipp: wenn man nach oben fährt muss geprüft werden ob der obere Rand von Pacman das Rechteck berührt, wenn man nach rechts geht ob der rechte Rand das Rechteck berührt…

- Teste dein Programm & überprüfe deinen Code. Wo könntest du Variablen gebrauchen und wo eine Methode schreiben um Wiederholungen zu vermeiden?