

GRUNDLAGEN DIGITALER GESTALTUNG 2

2. SEMESTER | SOMMERSEMESTER 2010 | KURS 5

Array Objects

Wenn man viele Objekte einer Klasse arbeitet, wird das Potenzial von Klassen erst wirklich ersichtlich. Bisher haben wir jedes Objekt einzeln benannt, und jedes Objekt einzeln in einer Kommandozeile angesprochen

Damit man das Arbeiten mit Klassen richtig ausschöpfen kann, muß man mit einem „Array“ arbeiten.

„Array“ heißt Datenbereich. Im Grunde genommen funktioniert ein Array wie eine Variable, doch der Unterschied ist, daß ein Array eine beliebige Anzahl von Werten speichern kann.

Im Fall von Objekten heißt es, das man beliebig viele Objekte erstellen kann, die der gleichen Klasse entsprechen.

Bisher schrieben wir, um ein Objekt zu erstellen:

```
Ball b1;
```

Um das Objekt zu konfigurieren heißt es im `setup()`:

```
b1 = new Ball();
```

und um das Objekt dann anzusprechen:

```
b1.meineFunktion();
```

Wenn man nur ein paar Objekte erstellen möchte, macht es Sinn, doch wenn man mit vielen Objekten arbeitet, summieren sich die Kommandozeilen, und es wird sehr umständlich. Dann sollte man mit einem Array arbeiten.

Damit das Programm weiß, dass man mit einem Array arbeiten möchte, setzt man die eckigen Klammern hinter seine Klasse: *Ball[] theBall; // Es wird ein Array der Klasse Ball erstellt, wobei die Objekte theBall heißen und noch mit einer Nummer versehen werden.*

Bei einem Array muß zusätzlich im `setup()` festgelegt werden wie viele Objekte dieser Klasse erstellt werden sollen:

```
theBall = new Ball[100];
```

Dann muß das Objekt konfiguriert werden. Da es sich aber um eine Anzahl von Objekten handelt, wird dies am geschicktesten mit der `for`-structure gelöst:

```
for(int i = 0; i < 100; i++){
    theBall[i] = new Ball(); // Jedes Objekt wird mit einer Nummer versehen
}
```

Jedes einzelne Objekt wird mit einer Nummer versehen, mit der das Objekt dann einzeln angesprochen werden kann. Wenn man alle ansprechen möchte wird wieder die `for`-structure verwendet:

```
for(int i = 0; i < 100; i++){
    theBall[i].display();
}
```

GRUNDLAGEN DIGITALER GESTALTUNG 2

2. SEMESTER | SOMMERSEMESTER 2010 | KURS 5

Array Objects

Und so sieht dann der komplette Code in der Hauptklasse aus:

```
Ball[] theBall;

void setup(){
  smooth();
  size(400,400);

  theBall = new Ball[100];

  for(int i = 0; i<100; i++){
    theBall[i] = new Ball();
  }

}

void draw(){
  background(0);
  for(int i = 0; i<100; i++){
    theBall[i].display();
  }

}
```

Was nun noch fehlt ist die Klasse „Ball“:

```
class Ball{
  float x = width/2;
  float y = height/2;
  float xspeed = random(-8,8);
  float yspeed = random(-8,8);

  void display(){
    x = x + xspeed;
    y = y + yspeed;

    if ((x > width) || (x < 0)) {
      xspeed = xspeed * -1;
    }

    if ((y > height) || (y < 0)) {
      yspeed = yspeed * -1;
    }
    fill(255);
    ellipse(x,y, 10, 10);
  }

}
```

GRUNDLAGEN DIGITALER GESTALTUNG 2

2. SEMESTER | SOMMERSEMESTER 2010 | KURS 5

ArrayList Objects

Ein Array ist statisch, die Anzahl der gespeicherten Werte sind festgelegt.

Im Gegensatz zu einer ArrayList. Eine ArrayList ist ein dynamisch bespielbarer Array, das heißt man kann Objekte hinzufügen und wieder wegnehmen. Interaktiv kann so die Anzahl der Objekte bestimmt werden, die die ArrayList befüllen soll.

Der Syntax heißt bei einem ArrayList dann:

```
ArrayList balls; // Die ArrayList hat den Namen balls
```

Im setup wird dann eine leere ArrayList geschaffen

```
balls = new ArrayList();
```

Um im draw Bereich alle Objekte der ArrayList darzustellen muß wieder mit der for-structure gearbeitet werden

```
for(int i = 0; i < balls.size(); i++){
  Ball ball = (Ball) balls.get(i);
  ball.display();
}
```

Es ist noch nichts zu sehen, weil die ArrayList noch leer ist.

Nun besteht die Möglichkeit die Liste interaktiv, zum Beispiel durch eine Mouseaktion zu bespielen, wenn die Funktion void mousePressed() angewandt wird, wird per Klick ein Objekt der Klasse balls der ArrayList zugefügt:

```
void mousePressed(){
  balls.add(new Ball());
}
```

Eine andere Möglichkeit der Interaktion könnte die Bewegung der Maus sein. Immer wenn sich die Mausposition ändert, soll ein Objekt der ArrayList hinzugefügt werden.

```
void mouseMoved(){
  balls.add(new Ball(mouseX, mouseY));
}
```

Über den Befehl balls.remove(NummerdesObjektes) kann ein Objekt wieder gelöscht werden. Mit balls.remove(0) wird das erste Objekt der Liste gelöscht, mit balls.remove(balls.size()-1) wird das letzte Objekt der Liste gelöscht.

```
if(mousePressed){
  balls.remove(0);
} //das erste Objekt der Liste wird gelöscht, die erste Nummer einer ArrayList ist immer die 0.
```

Oder:

```
if(mousePressed){
  balls.remove(balls.size()-1);
} //das letzte Objekt der Liste wird gelöscht
```

GRUNDLAGEN DIGITALER GESTALTUNG 2

2. SEMESTER | SOMMERSEMESTER 2010 | KURS 5

ArrayList Objects

Über den Befehl `balls.remove(NummerdesObjektes)` kann ein Objekt wieder gelöscht werden. Mit `balls.remove(0)` wird das erste Objekt der Liste gelöscht, mit `balls.remove(balls.size()-1)` wird das letzte Objekt der Liste gelöscht.

```
if(mousePressed){  
  if(balls.size()>0){  
    balls.remove(0);  
  }//das erste Objekt der Liste wird gelöscht, die erste  
  Nummer einer ArrayList ist immer die 0.  
}
```

Oder:

```
if(mousePressed){  
  if(balls.size()>0){  
    balls.remove(balls.size()-1);  
  }//das letzte Objekt der Liste wird gelöscht  
}
```

GRUNDLAGEN DIGITALER GESTALTUNG 2

2. SEMESTER | SOMMERSEMESTER 2010 | KURS 5

.....
Aufgaben bis zum 17.5
.....

Aufgabe 5.1

Überlege dir eine Idee, auf welche Weise sich deine Grafik generieren soll.

Schreibe erst in Pseudo-Code auf ein Stück Papier, was deine Klasse für Variablen benötigt, und was für ein Verhalten es haben soll.