

PROCESSING

EINE EINFÜHRUNG IN DIE INFORMATIK

Created by Michael Kirsch & Beat Rossmly

INHALT

1. Stoff der Vorlesung

1. Vererbung
2. Interfaces

2. Übung

1. Aufgabe 1
2. Aufgabe 2
3. Aufgabe 3

STOFF DER VORLESUNG

VERERBUNG

Wir definieren eine Elterklasse

```
class Elternklasse {  
    // Felder  
    int a,b,c;  
  
    public Elternklasse () {  
        a = 1;  
        b = 2;  
        c = 3;  
    }  
  
    // Methoden  
    void output () {  
        println("a: "+a+", b: "+b+", c: "+c);  
    }  
}
```

VERERBUNG

Wir definieren eine Kindklasse

```
class Kindklasse extends Elternklasse{  
    // Felder  
    // erbt: a,b,c  
  
    public Elternklasse () {  
        super();  
    }  
  
    // Methoden  
    // erbt: output()  
}
```

INTERFACES

Wir definieren ein Interface

```
interface Moveable {  
    void move();  
}
```

INTERFACES

Verschiedene Klassen implementieren das Interface.

```
class Ball implements Moveable {  
    // Felder  
    // Konstruktor  
    // Methoden  
  
    void move () {...}  
}
```

```
class Car implements Moveable {  
    // Felder  
    // Konstruktor  
    // Methoden  
  
    void move () {...}  
}
```

ÜBUNG

AUFGABE 1

```
String testString;
```

```
void setup () {  
    size(600,400);  
    testString = "test 123";  
}
```

```
float berechneMittelwert (float a, float b) {  
    return (a+b)/2;  
}
```

AUFGABE 2

Erzeuge und handle zwei Objekte.

```
class Ball {  
    float dx,dy,x,y,r;  
  
    public Ball (float x, float y, float r) {  
        this.x = x;  
        this.y = y;  
        this.r = r;  
        dx = random(-5,5);  
        dy = random(-5,5);  
    }  
  
    void move () {  
        // ...  
    }  
    void plot () {  
        ellipse(x,y,2*r,2*r);  
    }  
}
```

AUFGABE 2

Erzeuge und behandle zwei Objekte.

```
class Ball { /* ... */ }
Ball b1, b2;

void setup () {
    size(600,400);
    b1 = new Ball (200,200,100);
    b2 = new Ball (200,200,100);
}

void draw () {
    background(0);

    b1.move();
    b1.plot();

    b2.move();
    b2.plot();
}
```

AUFGABE 3

Erzeuge und handle zwei Objekte.

```
class Ball {
    float dx,dy,x,y,r;
    public Ball (float x, float y, float r) {
        // ...
    }

    void move () {
        // ...
    }

    void plot () {
        // ...
    }

    float calcDistance (Ball other) {
        return sqrt(sq(other.x-x)+sq(other.y-y));
    }
}
```

AUFGABE 3

Erzeuge und behandle zwei Objekte.

```
class Ball { /* ... */ }
Ball b1, b2;

void setup () {
    size(800,800);
    b1 = new Ball (200,200,100);
    b2 = new Ball (200,200,100);
}

void draw () {
    background(0);

    // Wir überprüfen ob der berechnete Abstand zwischen den Bällen kleiner als
    // die Summe ihrer Radien ist.
    // falls wahr: fülle Bälle rot
    // sonst: fülle weiß

    b1.move();
    b1.plot();
    b2.move();
    b2.plot();
}
```

AUFGABE 3

Erzeuge und behandle zwei Objekte.

```
class Ball { /* ... */ }
Ball b1, b2;

void setup () {
    size(800,800);
    b1 = new Ball (200,200,100);
    b2 = new Ball (200,200,100);
}

void draw () {
    background(0);

    if (b1.calcDistance(b2)<b1.r+b2.r) {
        fill(255,0,0);
    } else {
        fill(255);
    }

    b1.move();
    b1.plot();
    b2.move();
    b2.plot();
}
```