Variablen überwachen – Debugging

Der Debugger in Processing

Vor dem Start eines Programms kann man den Debugger einschalten:

Es öffnet sich beim Einschalten ein neues Fenster, in dem alle Variablen (die eigenen und die von Processing vorgegebenen) überwacht werden können. Noch ist das Fenster leer.

Um die Veränderung der Werte unserer Variablen beobachten zu können, müssen wir sogenannte Haltepunkte setzen und das Programm dann schrittweise durchlaufen.

Einen Haltepunkt setzt man entweder über Debugger – Breakpoints umschalten (Strg+B) oder direkt über einen Klick auf die Zeilennummer. Diese wird dann durch eine Raute ersetzt.

```
5  void setup(){
6    size(600,400);
7    setZufallsArray(10);
◆    veraenderung();
9  }
```

Man kann beliebig viele Haltepunkte setzen.

Gestartet wird das Debuggen mit der gewohnten Startschaltfläche. Direkt daneben ist die Schaltfläche für zeilenweises Durchlaufen des Programms. Darauf folgt die Schaltfläche, die den Sprung zum nächsten Haltepunkt auslöst. Die letzte Schaltfläche beendet das Debuggen.

Beispiel 1:

Programmcode:

```
1  int a = 10;
2  String name = "Hans";
3  float[] zufaelle;
4
5  void setup(){
6    size(600,400);
7    setZufallsArray(10);
◆    veraenderung();
9  }
10
11 void setZufallsArray(int anzahl){
12   zufaelle = new float[anzahl];
13   for (int i=0; i<anzahl; i++){
14     zufaelle[i]=random(100);
15   }
16 }
17
18 void veraenderung(){
19   a = a + 10;
20   name = "Hallo "+name;
21 }
```

Haltepunkt in der Zählschleife, schon zweimal durchlaufen:

Variables

Name	Value
anzahl	10
i	2
a	10
name	Hans
zufaelle	float[10]
[0]	99.36788
[1]	1.2837172
[2]	0.0
[3]	0.0
[4]	0.0
[5]	0.0
[6]	0.0
[7]	0.0
[8]	0.0
[9]	0.0
Processing	

Haltepunkt beim Aufruf der Methode `veraenderung()`, nach einem weiteren Schritt:

Variables

Name	Value
a	20
name	Hallo Hans
zufaelle	float[10]
[0]	55.755745
[1]	30.926615
[2]	74.54737
[3]	13.114464
[4]	42.40731
[5]	58.34924
[6]	80.59637
[7]	27.273481
[8]	59.76641
[9]	21.62329
Processing	

Beispiel 2[1]:

Programmcode:

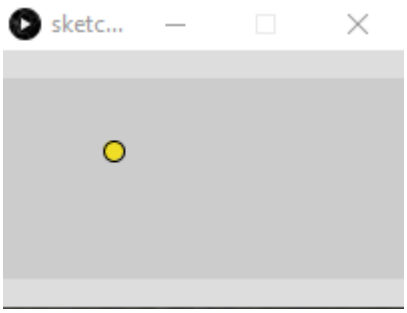
```
1  void setup() {
2    size(200, 100);
3  }
4
5  void draw() {
6  }
7
8  void mousePressed() {
9    fill(int(random(0, 255)), int(random(0, 255)), int(random(0, 255)));
◆    ellipse(mouseX, mouseY, 10, 10);
11 }
```

Nach dem Starten des Debuggers, wartet das Programm auf einen Mausklick auf dem Fenster. Die Koordinaten der Mausposition werden dann im Überwachungsfenster angezeigt.

Geht man schrittweise weiter, sieht man den ersten Kreis auf dem Startfenster.

Variables

Name	Value
Processing	
width	200
height	100
mouseX	55
mouseY	36
pmouseX	55
pmouseY	36
key	□
keyCode	0
keyPressed	false
focused	true
frameRate	60.00653
frameCount	814



Klickt man erneut, sieht man die neuen Koordinaten und nach dem Weitergehen auch den dazugehörigen Kreis.

1 Die Beispiele finden Sie unter 01_hintegrund/05_debugging_processing_beispiele