Functions

Functions (Funktionen) sind Unterprogramme, die bestimmte, isolierte Aufgaben übernehmen können. Wir kennen bisher die Systemfunktionen

```
void setup() // Initialisierung
void draw()  // wiederholte Ausführung
```

Neu dazu kommt eine weitere Systemfunktion

```
void mousePressed() // wird einmal ausgeführt, wenn die Maus gedrückt wurde
```

Wir haben auch schon verschiedene Funktionen im Programmcode verwendet, ohne uns um ihren detaillierten Aufbau kümmern zu müssen.

Syntax

Der Aufbau einer einfachen Funktion ist

```
void meineFunktion( Parameters ) {

...
}
```

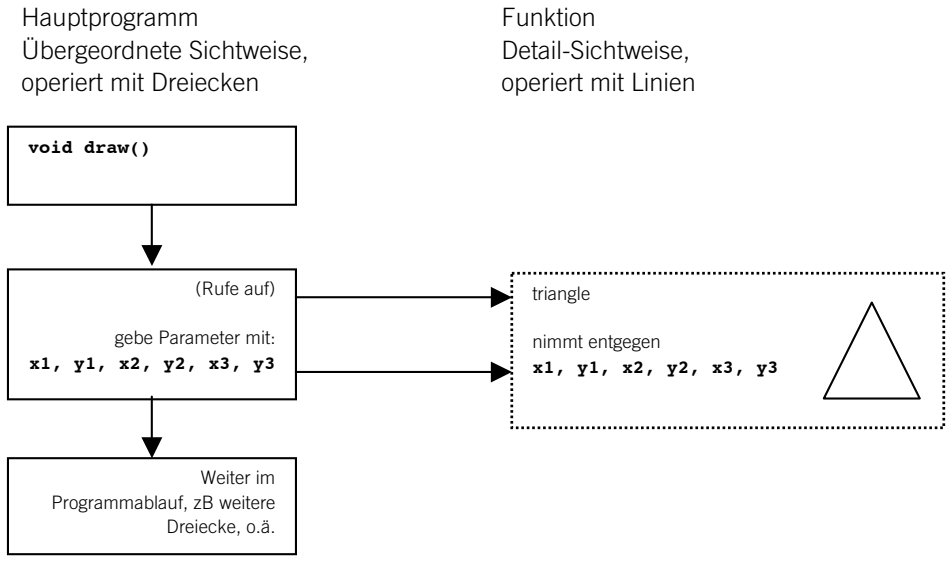
`void` bedeutet:	Funktion ohne Rückgabewert
`meineFunktion`:	Name der Funktion, frei wählbar
`Parameters`:	eine Liste von Variablen, die als Parameter übergeben wird.
`{`	die geschweiften Klammern bezeichnen Anfang und Ende des Funktionskörpers – also des Programmcodes der Funktion

Flussdiagramm:

Ein Beispiel für eine Funktion ist die Systemfunktion

```
triangle(int x1, int y1, int x2, int y2, int x3, int y3).
```

Anstatt im Programm drei Linien zeichnen zu müssen, übergeben wir diese Aufgabe an die Funktion ab.



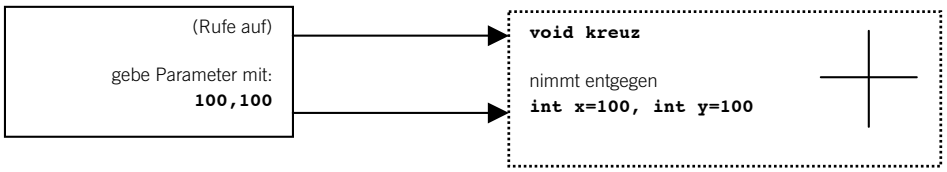
Beispiel

Eine Funktion die ein Kreuz am Punkt x,y zeichnet

```
void kreuz(int x, int y) { // Parameters müssen als Variablen deklariert werden: int x etc.
    line(x-10, y, x+10, y);
    line(x, y-10, x, y+10);
}
```

Damit kann an jedem beliebigen Ort x,y ein Kreuz gezeichnet werden. Der Aufruf aus der draw-Funktion geschieht beispielsweise so:

```
void draw() {
  kreuz(100,100);
}
```



Nimm und lies

Functions

Reas/Fry, "Processing", S. 181, Functions