Grundlegende Variablentypen und Funktionen in Processing

Variablentypen

In Processing gibt es verschiedene Variablentypen, die verwendet werden, um Daten zu speichern und zu manipulieren. Jeder Typ hat seinen eigenen Verwendungszweck und Speicherbedarf.

Typ	Name	Erklärung
`int`	Ganzzahl	Speichert ganze Zahlen ohne Dezimalstellen.
`float`	Gleitkommazahl	Speichert Zahlen mit Dezimalstellen.
`double`	Doppelte Gleitkommazahl	Ähnlich wie `float`, jedoch mit doppelter Genauigkeit.
`char`	Zeichen	Speichert einzelne Zeichen, z.B. Buchstaben oder Symbole.
`boolean`	Wahrheitswert	Speichert `true` oder `false`.
`String`	Zeichenkette	Speichert Textinformationen, z.B. Wörter oder Sätze.
`color`	Farbe	Speichert Farbwerte, oft in RGB-Format.
`PVector`	Vektor	Speichert einen Vektor mit x-, y- und z-Koordinaten, nützlich für 2D- und 3D-Grafiken.

Variablen deklarieren und initialisieren

Eine Variable wird **deklariert**, indem der Typ und der Name der Variable angegeben werden. Sie kann gleichzeitig **initialisiert** werden, indem ein Anfangswert zugewiesen wird.

```
int alter = 25;
float temperatur = 23.5;
boolean istAktiv = true;
String name = "Max Mustermann";
color hintergrundFarbe = color(255, 0, 0); // Rot
```

Funktionen

Funktionen sind Blöcke von Code, die eine bestimmte Aufgabe ausführen. Sie helfen, den Code zu strukturieren, wiederverwendbar zu machen und die Lesbarkeit zu verbessern.

Aufbau einer Funktion

Eine Funktion besteht aus einem Rückgabetyp, einem Namen und optionalen Parametern. Der Funktionskörper enthält den Code, der ausgeführt wird, wenn die Funktion aufgerufen wird.

```
Rueckgabetyp Funktionsname(Parameterliste) {
    // Funktionskoerper
    // Code, der ausgefuehrt wird
    return Wert; // Falls ein Wert zurueckgegeben wird
}
```

Beispiel einer Funktion

Hier ist ein einfaches Beispiel einer Funktion, die zwei Ganzzahlen addiert und das Ergebnis zurückgibt.

```
int addiere(int a, int b) {
    int summe = a + b;
    return summe;
}
```

Funktionen aufrufen

Eine Funktion wird aufgerufen, indem ihr Name gefolgt von Klammern und den erforderlichen Argumenten verwendet wird.

```
int ergebnis = addiere(5, 3); // ergebnis ist jetzt 8
```

Verschiedene Arten von Funktionen

Art	Name	Erklärung
`void`	Prozedur	Führt Aufgaben aus, gibt jedoch keinen Wert zurück.
`int`, `float`, etc.	Funktion mit Rückgabewert	Führt Aufgaben aus und gibt einen Wert des angegebenen Typs zurück.

Eingebaute Funktionen in Processing

Art	Name	Erklärung
`void setup()`	Setup-Funktion	Wird einmal zu Beginn des Programms ausgeführt, um Einstellungen vorzunehmen.
`void draw()`	Draw-Funktion	Wird kontinuierlich nach `setup()` aufgerufen, um Grafiken zu zeichnen oder Animationen zu erstellen.

Parameter und Argumente

- **Parameter**: Variablen, die in der Funktionsdefinition angegeben sind und als Platzhalter für die Werte dienen, die an die Funktion übergeben werden.
- **Argumente**: Die tatsächlichen Werte, die beim Aufruf der Funktion übergeben werden.

Beispiel mit Parametern und Argumenten

```
float berechneFlaeche(float breite, float hoehe) {
    return breite * hoehe;
}

void setup() {
    float flaeche = berechneFlaeche(5.0, 3.0); // flaeche ist jetzt 15.0
}
```

Lokale und Globale Variablen

- **Lokale Variablen**: Innerhalb einer Funktion definiert und nur innerhalb dieser Funktion sichtbar.
- **Globale Variablen**: Außerhalb aller Funktionen definiert und in allen Funktionen des Programms zugänglich.

Beispiel für lokale und globale Variablen

```
int globalZahl = 10; // Globale Variable

void setup() {
    int lokaleZahl = 5; // Lokale Variable
    println("Globale Zahl: " + globalZahl);
    println("Lokale Zahl: " + lokaleZahl);
}

void draw() {
    println("Globale Zahl in draw: " + globalZahl);
    // println("Lokale Zahl in draw: " + lokaleZahl); // Fehler: lokaleZahl ist
       hier nicht sichtbar
}
```