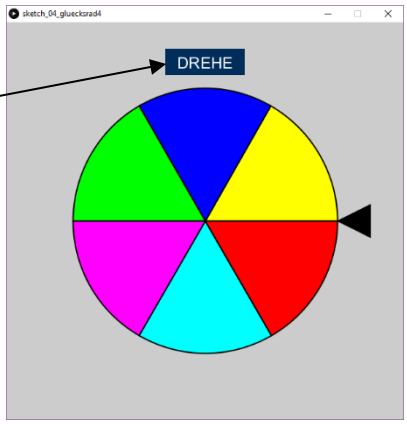
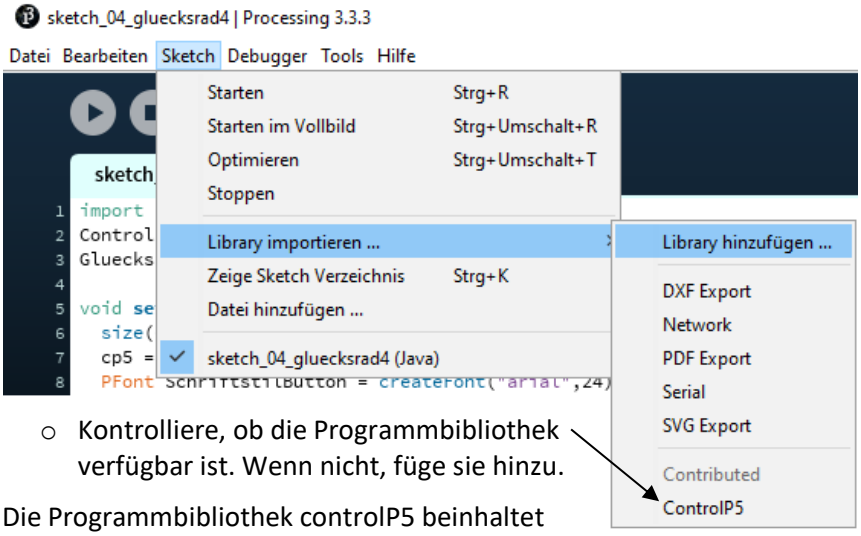
Die Programmbibliothek controlP5 für Processing



- Um das Programm nicht immer beenden und neu starten zu müssen, wenn man das Glücksrad drehen möchte, wird ein Startknopf ergänzt.

Eine Programmbibliothek muss in Processing über das Menü Sketch -> Library importieren ... -> Library hinzufügen ... einmal hinzugefügt werden. Die Bibliothek wird dann im Ordner `\Documents\Processing\libraries` gespeichert:



- Kontrolliere, ob die Programmbibliothek verfügbar ist. Wenn nicht, füge sie hinzu.

Die Programmbibliothek controlP5 beinhaltet u. a. die Klasse `Button`. Ein Button löst durch Mausklick eine Aktion aus. Die Methode `addButton(name)` erzeugt einen Button mit der Beschriftung NAME, durch den die Methode `name()` aufgerufen wird.

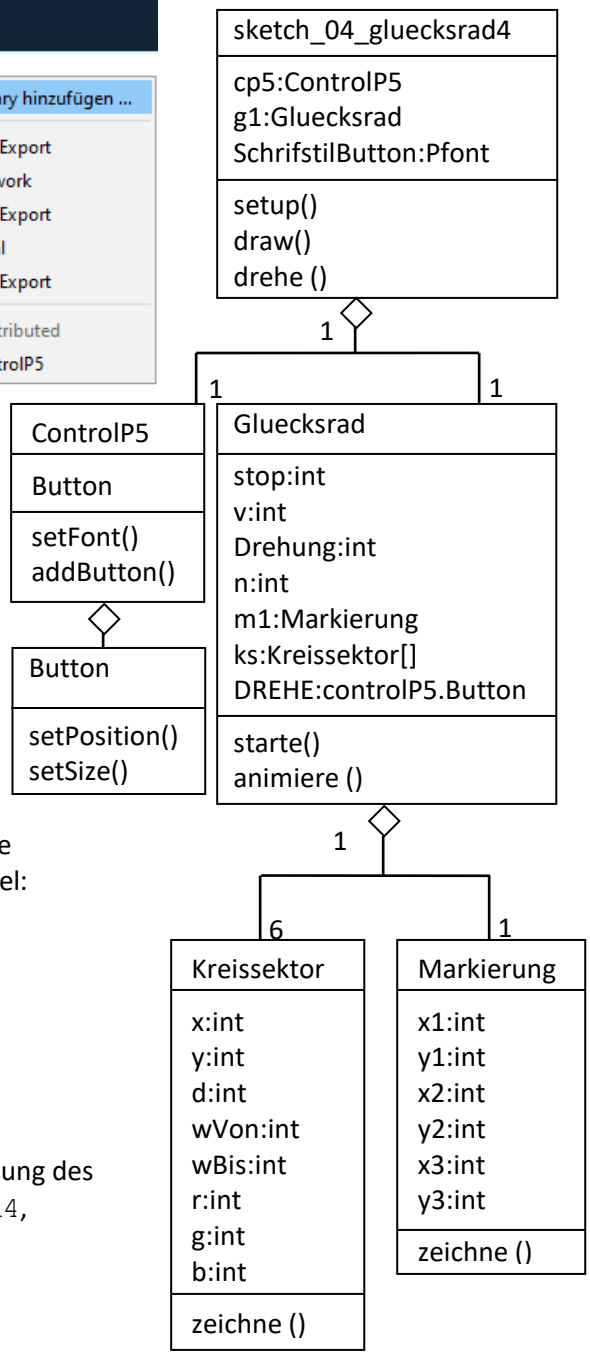
Beispiel: Durch die Anweisungen `ControlP5 cp5;` und `cp5 = new ControlP5(this);` wird das Objekt `cp5` der Klasse `ControlP5` erzeugt.
Der Parameter `this` übergibt der Klasse `ControlP5` eine Referenz auf das aktuelle Objekt.

Die Anweisung `cp5.addButton("drehe");` erzeugt den Button mit der Beschriftung `DREHE`.
Dann muss eine gleichnamige Methode erstellt werden, die durch Mausklick auf den Button ausgeführt wird, im Beispiel:

```
public void drehe() {
  g1.starte();
  g1.stop=0;
}
```

Die Beschriftung des Buttons wird mit der Methode `getCaptionLabel().align(ControlP5.CENTER, ControlP5.CENTER)` festgelegt.

- Auf der folgenden Seite befindet sich eine Beschreibung des Programms in der Vorlagedatei `v17_gluecksrad4`, insbesondere zum Erstellen des Buttons (**`I`**).
(*vgl. .\262-materialien\zufall\04-gluecksrad4*)



```
import controlP5.*;
ControlP5 cp5;
Gluecksrad g1;
void setup() {
  size(600,600);
  cp5 = new ControlP5(this);
  PFont SchriftstilButton = createFont("arial",24);
  cp5.setFont(SchriftstilButton);
  g1=new Gluecksrad();
}
void draw(){
  if (g1.stop==0) {
    background(255);
    g1.animiere();
  }
}
public void drehe() {
  g1.starte();
  g1.stop=0;
}
class Gluecksrad {
  int stop;
  int v;
  int Drehung;
  int n;
  Markierung m1;
  Kreissektor[] ks = new Kreissektor[6];
  Gluecksrad() {
    cp5.addButton("drehe")
       .setPosition(240,40)
       .setSize(120,40)
       .getCaptionLabel().align(ControlP5.CENTER, ControlP5.CENTER)
    ;
...
    starte();
    animiere()
  }
  void starte() {
    frameRate(v);
    n=0;
    stop=1;
    Drehung=floor(random(720,1081));
  }
  void animiere() {
    m1.zeichne();
    for (int i = 0; i < 6; i++) {
      ks[i].zeichne();
      ks[i].wVon++;
      ks[i].wBis++;
    }
    n++;
    if (n==Drehung) {
      stop=1;
}  }  }
...
```

Zu Beginn muss controlP5 in das aktuelle Programm importiert und ein Objekt der Klasse `ControlP5` festgelegt werden.

Der Parameter `this` übergibt der Klasse `ControlP5` eine Referenz auf das aktuelle Objekt.

Erstellt ein Objekt für den Schriftstil zur Beschriftung des Buttons.

Das Beenden der Animation mit noLoop() wird nicht mehr möglich sein, weil dann auch controlP5 nicht mehr ausgeführt wird. Deshalb wird das Attribut `stop` eingeführt und zu Beginn mit dem Wert 1 belegt, damit `g1.animiere()` nicht ausgeführt wird.

Durch das Schlüsselwort `public` wird allen anderen Klassen der Zugriff auf die Methode erlaubt. Das gegenteilige Schlüsselwort ist `private`, das den Zugriff auf die Methode von außen verhindert.

Wenn der Wert des Attributs `stop` 0 ist, wird die Animation ausgeführt.

(**I**) Das Erstellen des Buttons besteht aus einer Folge von Methodenaufrufen, die sich auf das Objekt `cp5` beziehen. Diese Folge wird mit ; beendet und nicht eine Sequenz mit { und } angegeben.

In dem darauffolgenen Programmteil sind keine Ergänzungen erforderlich.

`animiere()` muss im Konstruktor der Klasse `Gluecksrad` einmal ausgeführt werden, um das Glücksrad zu zeichnen.

Zum Programmstart soll die Animation nicht ausgeführt werden, weshalb der Wert das Attributs `stop` auf 1 gesetzt wird.

Die Animation wird nicht mehr mit `noLoop()` beendet, sondern indem der Wert das Attributs `stop` auf 1 gesetzt wird.

1. Erweitere das Programm `gluecksrad4`:
   - Die Anzahl der Versuche soll mitgezählt werden.
   - Wenn zweimal hintereinander dieselbe Farbe gedreht wird, soll die Textausgabe „Gewonnen!“ eingeblendet werden. (Syntax für `Text` vgl. Lerninhalte 01, S. 6)

   Hinweise:
   - Dafür muss der Button `NEU` ergänzt werden.
     In der zugehörigen Methode werden die Attributwerte auf den Anfangszustand gesetzt.
   - Das Attribut `ButtonDrehe` dient als Zeiger, ob der Button `DREHE` gerade aktiv sein soll:
     Die Methode `drehe()` darf durch Klick auf den Button nur aufgerufen werden (`ButtonDrehe==1`), wenn das Spiel nicht beendet ist.
     Zum Programmstart muss `ButtonDrehe` also auf den Wert 1 gesetzt werden.
   - Die Anzahl durchlaufener Kreissektoren kann mit Hilfe weiterer Attribute in der Methode `zaehleSektoren()` alle 60 Drehungen mitgezählt werden.
     Dadurch werden die Sektoren gegen den Uhrzeigersinn von 0 (gelb) bis 5 (rot) durchnummeriert.
     - Das aktuelle Ergebnis wird in dem Attribut `aktuell` gespeichert.
     - Für das vorangegangene Ergebnis wird das Attribut `zuvor` eingeführt.
     - Das Attribut `zuvor` erhält bei der Konstruktion des Objekts `g1` den Wert 7, der im Programmverlauf nicht vorkommen darf.
     - Nachdem eine Drehung beendet wurde, erhält zuletzt das Attribut `zuvor` den Wert des Attributs `aktuell`.

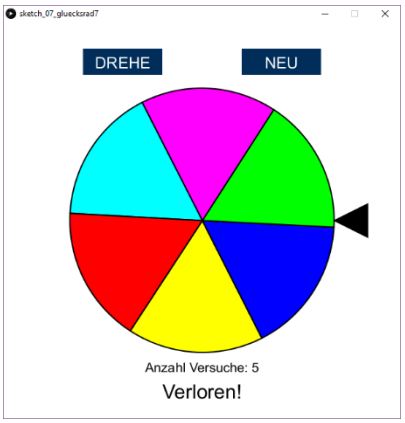
sketch_05_gluecksrad5
cp5:ControlP5 g1:Gluecksrad SchrifstilButton:Pfont
setup() draw() **heu()** drehe ()

g1:Gluecksrad
ButtonDrehe=1 **AnzahlVersuche=0** **aktuell=0** **zuvor=7** stop=1 v=500 Drehung= n=0 **Sektorzaehler=0**; m1 ks **NEU** DREHE
starte() animiere () **zaehleSektoren()** **zeigeSpielstandAn()**

   - Ergänze das Programm `gluecksrad4`.
     Speichere die Datei als `gluecksrad5`.
     (vgl. .\262-materialien\zufall\05-gluecksrad5)

2. Zusatzaufgabe: Nach 5 erfolglosen Versuchen soll die Textausgabe „Verloren!“ eingeblendet und der Button DREHE deaktiviert werden (`gluecksrad6`).
   (vgl. .\262-materialien\zufall\06-gluecksrad6)

   Hinweis: Sollte zufälligerweise die 5. Drehung gewinnen, muss ausgeschlossen werden, dass der Text „Gewonnen!“ mit dem Text „Verloren!“ überschrieben wird. Dafür kann in der zweiten Auswahlstruktur die Anweisung `else if` verwendet werden. Damit wird die zweite `if`-Anweisungen nur ausgeführt, wenn die erste nicht zugetroffen hat.



3. Zusatzaufgabe: Das Glücksrad soll ab einem bestimmten Zeitpunkt, z. B. ab 480 Drehungen vor Schluss, langsam abbremsen (`gluecksrad7`). *(vgl. .\262-materialien\zufall\07-gluecksrad7)*

4. Zu Beginn dieser Lehrplaneinheit hast du den Computer die Zahlen Siebzehn und Vier addieren und multiplizieren lassen. Mittlerweile kannst du in Processing einen Rechner programmieren, der die vier Grundrechenarten mit beliebigen Zahlen beherrscht. Entwickle ein Programm, bei dem zwei Zahlen in Textfelder eingegeben werden können, die dann wahlweise addiert, subtrahiert, multipliziert und dividiert werden können.

ControlP5
Button Textfield
setFont(Schriftstil:Pfont) addButton(Name:String) addTextfield(Name:String)

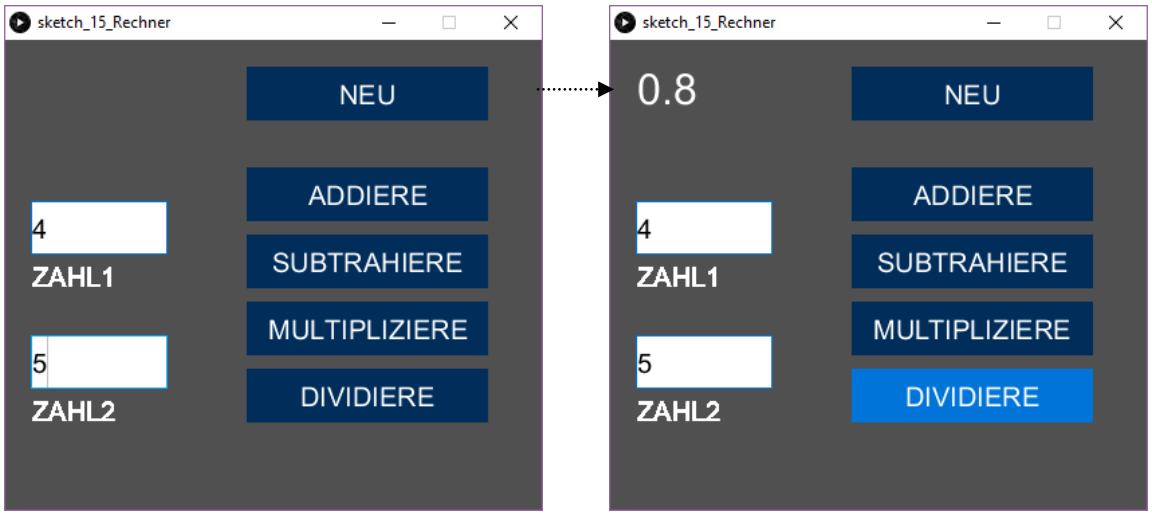
   Hinweise:
   - Der Datentyp für alle Zahlen ist Gleitkomma (`float`).
   - Zur Eingabe der beiden Zahlen benötigst du controlP5-Textfelder. Die wichtigsten Methoden der Klasse `controlP5.Textfield` sind aus dem Klassendiagramm rechts zu ersehen.

Textfield
setPosition(x:int,y:int) setSize(breite:int,höhe:int) setFocus(boolean) setColorBackground (color(r,g,b)) setColor(color(r,g,b)) clear() getText()

   - Der Datentyp `boolean` kann zwei Werte annehmen, nämlich `true` und `false`. Mit der Methode `setFocus(true)` lässt sich festlegen, dass eines der Textfelder zum Programmstart aktiv ist, also die Eingabe in diesem Textfeld stattfindet, ohne
   - Die Farbe der Schrift wird mit `setColor()` festgelegt.
   - Schreibweise:
     - `cp5.get(Textfield.class,"zahl1").clear();` löscht den Inhalt des Textfelds.
     - `tz1=cp5.get(Textfield.class,"zahl1").getText();` liest den Inhalt des Textfelds aus und speichert ihn in dem Attribut `tz1`.
     - `getCaptionLabel().align(ControlP5.CENTER, ControlP5.CENTER)` legt die Beschriftung der Buttons fest.

Button
setPosition(x:int,y:int) setSize(breite:int,höhe:int)

   - Im englischsprachigen Bereich und damit auch in allen Programmiersprachen wird keine Dezimalkomma, sondern Ein Dezimalpunkt für „Kommazahlen“ verwendet, also z. B. nicht 1,5 sonder 1.5.
   - Die Fehlermeldung NaN bedeutet „Not a Number“ – also „keine Zahl“: In ein Textfeld kann jedes beliebige Zeichen eingetragen werden oder es kann auch leer gelassen werden.
     Um zu überprüfen, ob ein Wert keine Zahl ist, kann die folgende Anweisung verwendet werden:
     `if (Float.isNaN(Wert)` (“Wenn der Wert eine Gleitkommazahl ist ...“)
     bzw. `if (!Float.isNaN(Wert))` (“Wenn der Wert **keine** Gleitkommazahl ist ...“)



   - Speichere das Progamm unter der Bezeichnung `Rechner1`.
     Zusatzaufgabe: Versuche im Programm alle möglichen Fehleingaben auszuschließen („abzufangen“).
     (vgl. .\262-materialien\rechnung\15-Rechner1)