DOKUMENTATION:
PROGRAMMING BASICS PROCESSING

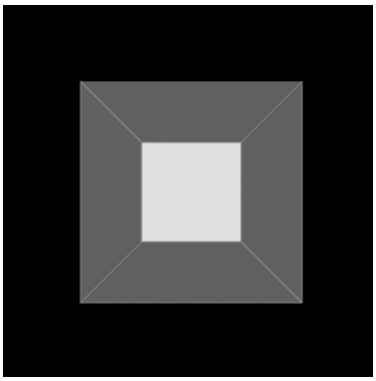
//Nora Gailer
//September 2013

`void MyTreeIs3D ( )`

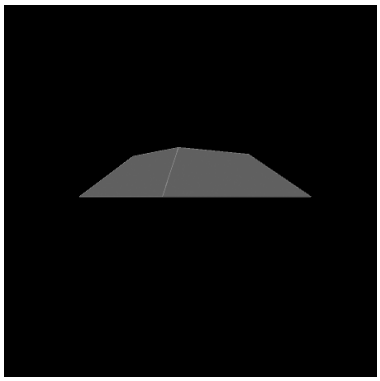
P3D / push-&popMatrix / void Name

{ An dieser Aufgabe hat mir vorallem das Erstellen des 3D-Objektes gefallen. Durch das intensive Auseinandersetzen mit den drei Koordinatenpunkte (x,y,z), wurde mir die Handhabung sicherlich verständlicher. Jedoch wäre es für mich hilfreich gewesen, etwas über die Funktion class() zu wissen. Somit hätte ich den Baum einfacher zusammenbauen und auch einfacher damit herumspielen können, da er als eigenstädiges Objekt funktioniert hätte. Die von mir gewünschte Darstellung des Waldes ist unten zu sehen (Abb. 1).

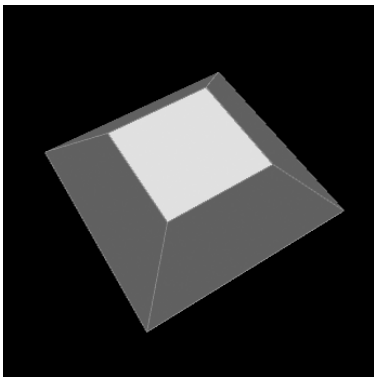
Ich entschied mich für den geometrischen Baum,weil man das vervielfachte Darstellen eines immer gleichen und genauen Objektes einfacher generieren kann. Auch weil es mich fasziniert wie wenig und abstrakte Formen man benutzen kann und trotzdem ist das Gebilde als Baum zu erkennen. }



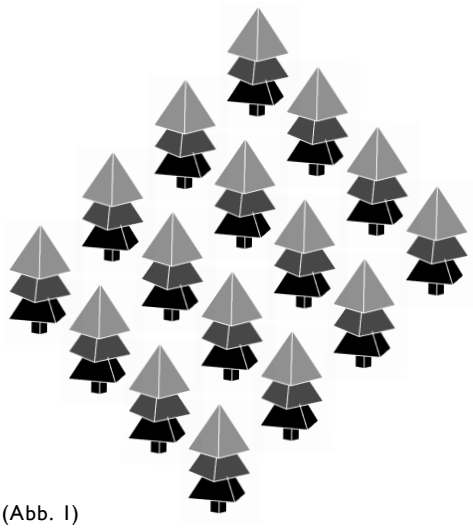
(Abb. 2.1)



(Abb. 2.2)



(Abb. 2.3)



(Abb. 1)

void Turn, Turn, Turn... ( )

atan2 / calcAngle

{ Der Smiley Code und das Spiel vom drehenden Kreis war eine nützliche Übung für mich. In Betrachtung meiner Endaufgabe, konnte ich die gelernten Syntax perfekt wieder anwenden. Auch vermute ich, dass das Spiel mit dem drehenden Kreis Einfluss auf die Idee und Umsetztung meines Games hatte. Es war jedenfalls ein gutes Gefühl, Gelerntes sicher wieder anwenden zu können. }

```
<code>
return -atan2(mouseX - (width / 2),mouseY - (height / 2));
</code>
```

void Planetensystem ( )

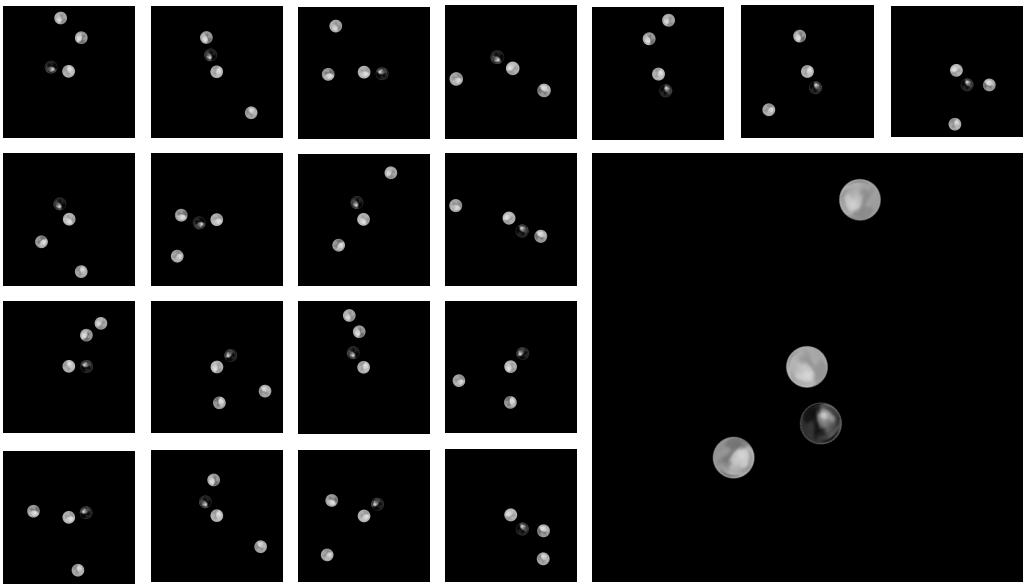
ArrayList

{ Wie nützlich der Befehl Arraylist für das Programmieren mit Processing ist, war mir von Anfang an bewusst. Dennoch brauchte ich einige Zeit, um mich mit dem Aufbau und der Schreibweise vertraut zu machen. Nach dieser Aufgabe, in welcher ich schliesslich ein kleines Planetensystem kreierte (Abb 3), war mir das Prinzip und die Handhabung von Arraylists klar. Dennoch habe ich zu einem späteren Zeitpunkt beim Erstellen der Endaufgabe herausgefunden, dass das Arbeiten mit Arraylist unglaublich komplex sein kann. Schwierig finde ich dabei vor allem das abstrakte Denken mit dem Buchstaben [i] und wie verschachtelt man mit dieser Variabel umgehen kann. }

```
<code>
PImage[] imageList = null;    // variable sauber initialisieren

imageList = new PImage[4];
  imageList[0] = loadImage(„../images/planet4.png“);
  imageList[1] = loadImage(„../images/planet2.png“);
  imageList[2] = loadImage(„../images/planet3.png“);
  imageList[3] = loadImage(„../images/planet1.png“);

for (int i=0 ;i < imageList.length; i++) // Loop für ArrayList
<code>
```

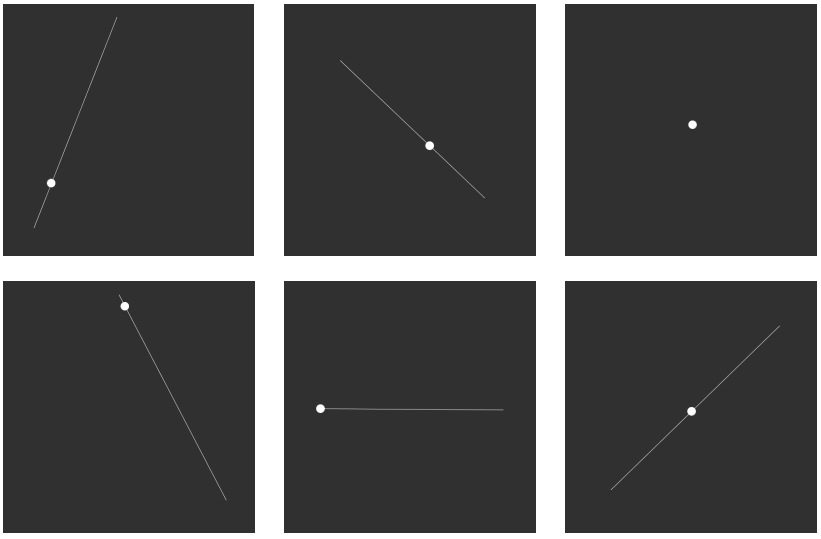
(Abb. 3)

void Get to know PVector( )

PVecotr / %Modulor

{ Als das Wort Vektoren auftauchte, machte sich mein Hirn auf die weite Suche durch all meine Hinterstübchen. Obwohl schleichend Erinnerungen an den Sekundar-Geometrie-Unterricht zurück kamen, hätte ich mir eine intensivere Auseinandersetzung mit dem Thema gewünscht. Auch weil ich zu einem späteren Zeitpunkt die Relevanz von PVector verstand.
Modulo () sehe ich als nützliche Funktion an, obwohl ich noch nicht ganz verstanden habe, wie es funktioniert. Bei dieser Aufgabe brauchte ich es um der jeweiligen Kondition einen geraden oder ungeraden Zahlenwert beizumessen.
An der Aufgabe fand ich in erster Linie interessant, wie unterschiedlich Lösungsansätze sein können. Was mir bewusst machte, dass es nicht nur einen richtigen Weg gibt. Also kann man auch beim Coding Gestaltung mit einbringen und versuche schöne Lösungswege zu finden. }


```
<code>
click_number += 1;
  if (drawFlag = true
      && click_number % 2 == 1) // falls Klickrate ungerade
<code>
```



(Abb. 4)

void Endaufgaben ( )

class / sin / cos / angleBetween/ fromAngle / degrees / .get/ .mag

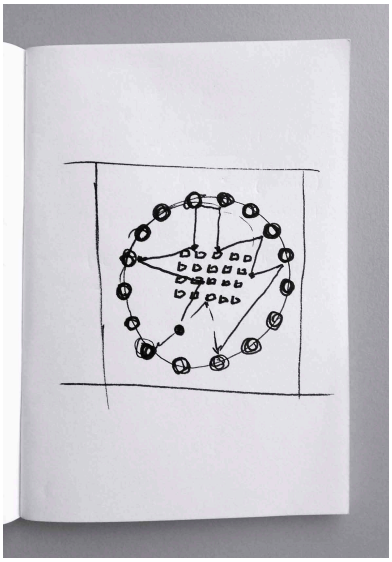
{Das Erstellen einer eigenen Version von BreackOut finde ich eine tolle Idee. Mein Game sollte anstatt ein Rechteck, ein Kreis als Spielfläche haben (Abb. 6.1), auf welchem gleichmässig weitere Kreise verteilt sind ähnlich einer Halskette. Darin befindet sich der Spielball (Abb. 6.6), welcher die Klötzchen in der Mitte abschiessen muss um das Spiel zu gewinnen (Abb. 6.2 - 6.5). Dabei darf das Spielfeld nicht verlassen werden. Dafür ist die mit dem Zeiger zu bewegende Kreiskette zuständig. Je nachdem ob ich den Mauszeiger rechts oder links herum drehe, folgt der Kreis der Bewegung. Wenn der Spielball auf einer der Kreise trifft, prallt er ab. Sonst ist er aus dem Feld und das Spiel ist vorbei. (Abb. 5.1)

Ich wusste, dass ich mir ein hohes Ziel gesteckt habe, ging aber trotzdem mit viel Elan an die Arbeit. Durch diese Aufgabe wurde mir auch richtig bewusst, wie viel Mathematik in Processing steckt. Um meine Ansätze festzuhalten und verstehen zu können, habe ich mir darum viel auf Papier aufskizziert (Abb. 5.2 - 5.5). Meiner Meinung nach ist dies ebenso ein wichtiger Teil vom Arbeitsprozess wie das Verstehen von Funktionen und Strukturen. Am meisten Mühe bereitete mir definitiv die Kollisionsabfrage der Bälle. Wie ich die Abfrage konstruieren sollte und wie ich sie strukturiert aufbauen sollte, damit ich sie auf mehrere Objekte anwenden kann. Erst beim Schreiben dieses Syntax wurde mir die Auswirkung auf den restlichen Code bewusst. Ich musste alle Konditionen neu schreiben.

Die Gestaltung meines Spiels ist sehr simple und einfach. Da mein Spiel nicht wirklich mit einem Thema verbunden ist sondern es sich mehr um die Spielweise und Form dreht, habe ich eine Bildwelt nicht als nötig empfunden. Die Farben habe ich nach persönlichem Geschmack ausgewählt.

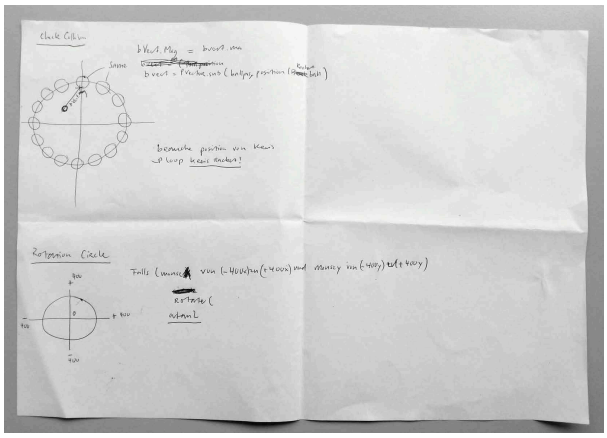
Was mir bei diesem Projekt auffiel und wo ich gerne noch mehr erfahren würde, ist, wie man den Aufbau eines Code bewältigt. Obwohl ich mich in gewissen Punkten an der Vorgabe orientieren konnte, wünschte ich mir eine Erklärung zu Abfolge und Aufbau (Struktogramm) damit ich mich darauf hätte stützen können.

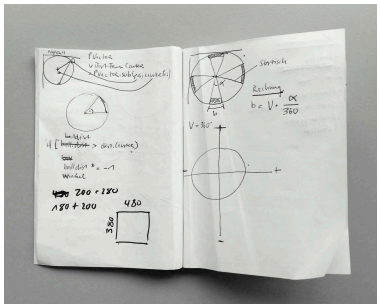
Leider konnte ich die Aufgabe nicht befriedendstellend beenden. Es gibt für mich noch zu viele Fragen in Bereich von Schreibweise und auch wie die verschiedenen Codeschnipsel zueinander stehen. Ich habe auf jedenfall wahnsinnig viel gelernt in den letzten 3. Wochen und möchte meine Kenntnisse im Programmieren von Processing definitiv auch weiterhin vertiefen.}



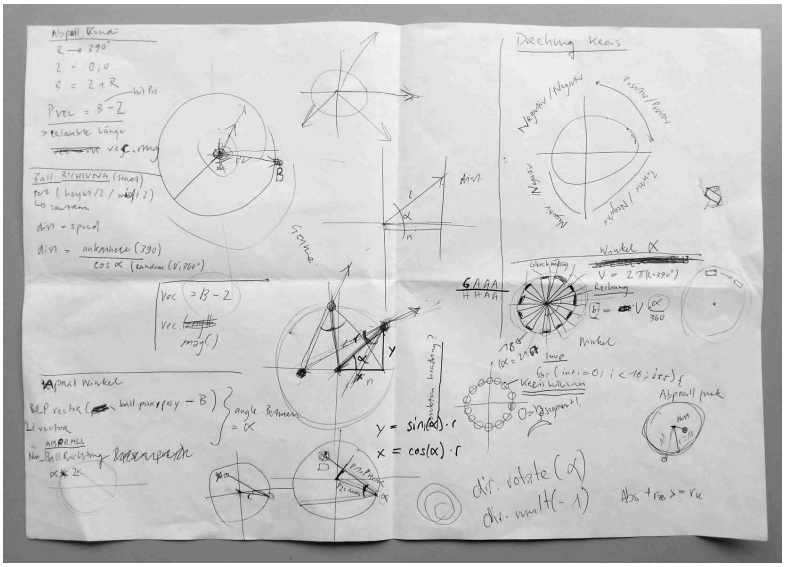
(Abb. 5.1)

(Abb. 5.2)

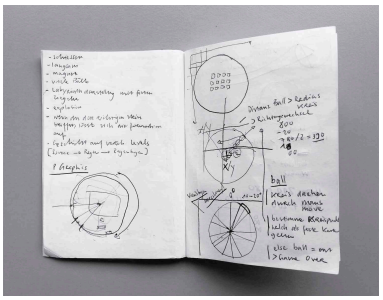




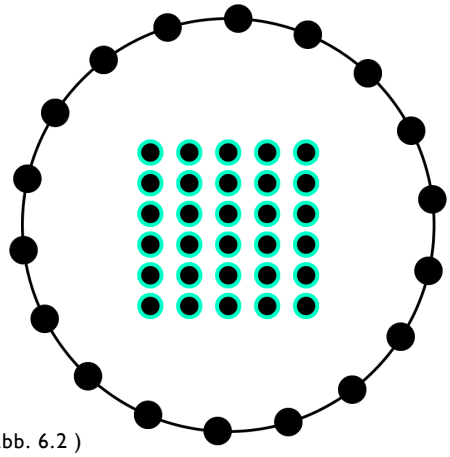
(Abb. 5.3)



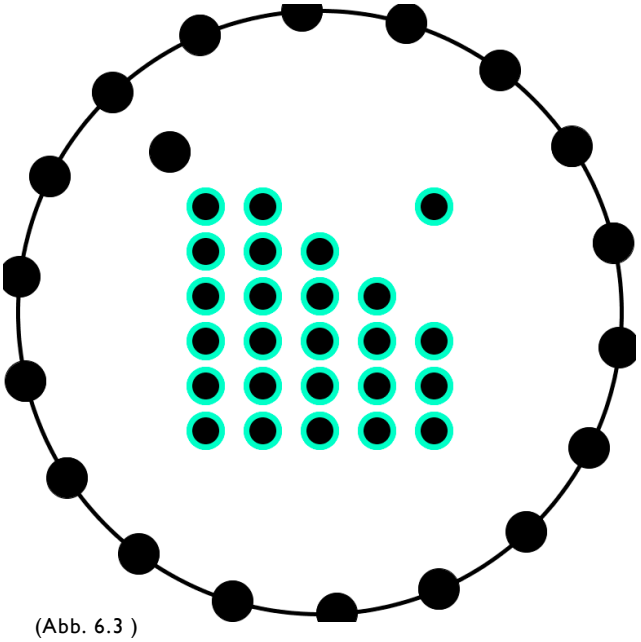
(Abb. 5.5)



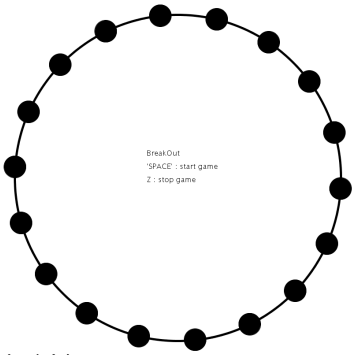
(Abb. 5.4)



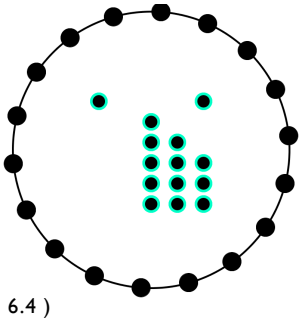
(Abb. 6.2)



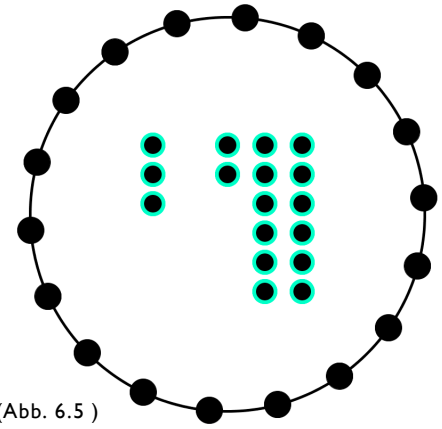
(Abb. 6.3)



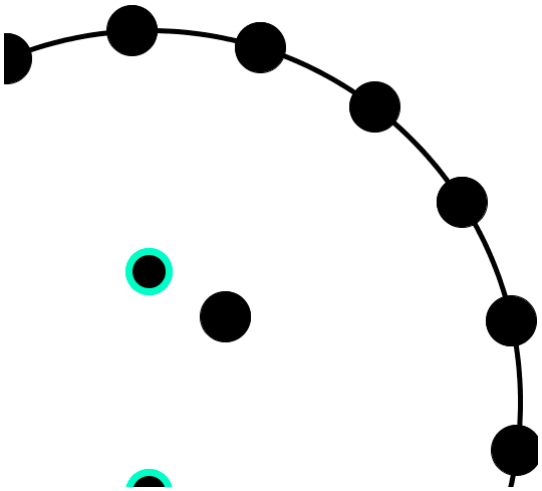
(Abb. 6.1)



(Abb. 6.4)



(Abb. 6.5)



(Abb. 6.6)