Designtænkningsproces

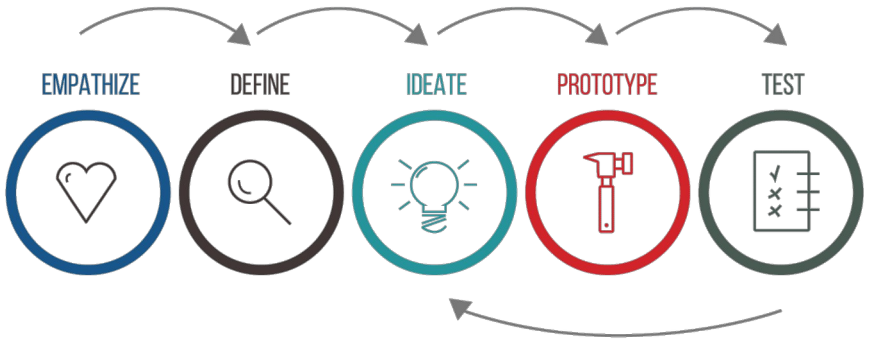
Gestalt lovene kan hjælpe os i appudviklingen med at øge brugervenligheden af vores produkt samt få vores brugere til at fokusere på vores apps indhold.

KISS er en term som er en forkortelse for “Keep it simple stupid” da det er vigtigt at brugeren kan finde ud af at bruge appen, da appudviklerne og brugerne ikke nødvendigvis har den samme forståelse for hvordan appen hænger sammen. Det er også vigtigt ikke at gøre selve appens kode meget kompliceret bare for at få det til at virke, når man kan simplificere det og få det samme resultat. Derfor er det vigtigt at vi designer vores app ud fra KISS, fordi hvis appen er simpel, så er det nemt at forstå. Det ville gøre den mere brugervenligt, så de skal bruge mindre tid til at finde ud af hvordan appen fungerer.

FTF eller First Thing First betyder at det vigtigste ting skal komme først. Vi har gjort nogen overvejelser, som vi valgte at tage med i vores app. Vores app handler om at samle skrald ind fra havet med en båd, og det er det første man ser når man kommer ind i appen

Vi havde en masse idéer, men vi sat os fast på et spil, som går ud på at samle plastik og andet affald op fra havet. Spillet skulle økonomisk støtte en velgørenhed som arbejder med det 14. Verdensmål; livet i havet, ved at indsamle penge fra brugerne ved fx reklamer og indkøb i spillet.

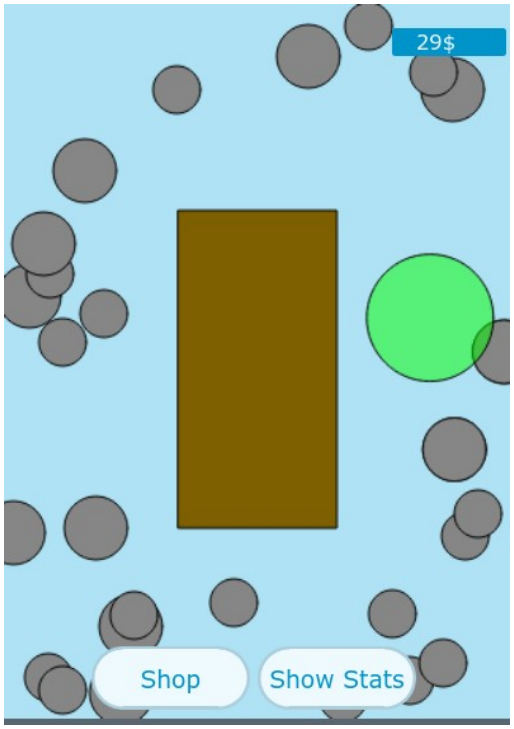
Vi brugte “prototype” fasen i vores designtænkningsproces, fordi at processen af at udvikle en app handler meget om at lave en prototype, få feedback og gå tilbage til designtænkning for at få dækket de nye behov fra den data man fik fra feedbacken. Modellen nedenfor forklarer det med godt.



Vi har lavet en meget tidlig prototype som kan illustrere idéen med vores spil, men mangler en masse features. Vi mangler fx at få billeder ind i stedet for at bruge cirkler og rektangler som pladsholdere, bedre spilmekanikker, reklamesystem, databasesystem osv.

Implementering

Som sagt går vores spil ud på at indsamle skrald op fra havet, som kræver noget kode for at få til at virke.



Det første der sker når man starter spillet, er at den går ind i spil-loopet som kører mens spillet bliver spillet. Det første der sker, er at den rydder kanvasset som fjerner det vi tegnede fra sidste frame. Derefter tegner vi alle cirklerne som skal repræsentere affald ved at vi gemmer dem alle sammen i en liste. Når et stykke affald kolliderer med vores kollektions rækkevidde (den grønne cirkel), begynder affaldet at bevæge sig mod båden med en hastighed afhængig af hvor på musen er i forhold til affaldets centrum. Vi definerer hastigheden ved den følgende kode vist under:

```
// Move towards boat
var acceleration = CURcollectionSpeed / (distX + distY);
```

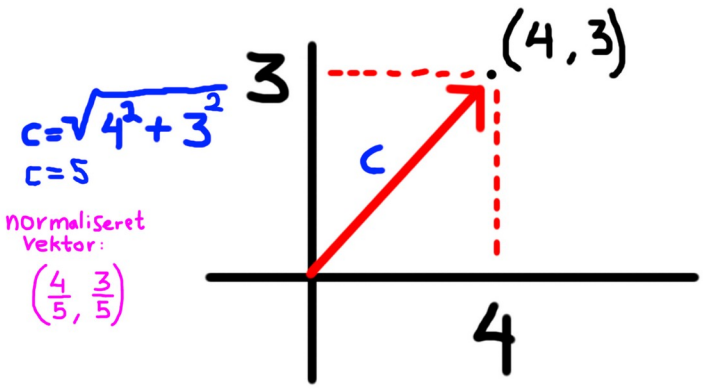
Den giver altså en acceleration afhængig af hvor langt væk affaldet er fra musen. Affaldet skal også vide hvilken en vej den skal gå imod altså hen mod båden. Det første vi gør, er at finde en enhedsvektor som repræsenterer retningen fra affaldet til båden:

```
// Get unit vector
xCom = 150 - x;
yCom = 220 - y;
// Normalize values
normalizeCom(xCom, yCom);
```

“normalizeCom()” er en funktion som vi har implementeret som normaliserer den vektor som vi får fra at trække de to positioner fra hinanden. Funktionen bruger

Pythagoras læresætning til at finde ud af hvor stor en del af de to forskellige komponenter (x og y) opgør af den samlede afstand mellem affaldet og båden:

```
function normalizeCom(x, y) {
  denom = Math.sqrt(Math.pow(x, 2) + Math.pow(y, 2));
  xCom = x/denom;
  yCom = y/denom;
}
```



Som man kan se på billedet ovenfor, beregner vi hvor stor en del den specifikke komponent af vektoren (x el. y) opgør af den samlede afstand.

Testcase

I testen kan man se os, som der tester spillet og dets spilmekanikker. Vi samler blandt andet skrald ind og opgraderer vores rækkevidde for at fange mere skrald.

https://youtu.be/s_gr7OjhHhk

Problemer under udviklingsprocessen

Vi havde problemer med at eksportere vores spil fra Unity, hvilket var vores oprindelige plan. Da vi altså ikke kunne få det til at virke var vi nødt til at lave spillet fra scratch i AppLab. Det gjorde at vi kom bagud i projektet, men det var alligevel ikke helt spildt arbejde. Det tog ikke lang tid at lave spillet forfra i AppLab, da det hele allerede var designet og lavet, bare i Unity, så vi vidste altså allerede hvordan vi ville have det hele til at se ud og fungere. Det endte dog stadig med at blive en meget simplificeret version af det originale spil. Ellers gik processen godt, og vi fik lavet en funktionel prototype af appen, vi havde i tankerne.

*Unity version af spillet**

Da spillet skulle laves i AppLab var det ikke muligt for os at komme rigtige reklamer i spillet. Det betyder selvfølgelig at man ikke kan indsamle rigtige penge til velgørenhed, og på den måde bidrager vores spil ikke til verdensmålet. Det ville vi kunne fikse hvis vi lavede appen i et andet program / engine. Vi bruger altså AppLab til at lave de tidlige prototyper, hvor at hvis det var et projekt som vi skulle lave videre på og blive færdig med, så ville det være mere smart at bruge andre ting som React eller lignende.

AppLab gav os også problemer med performance, fordi at den API de stiller til rådighed, ikke er egnet til at man skal bruge den til at lave et spil. I et spil skal et kanvas nemlig ryddes og tegnes for hver frame og det kan AppLab bare ikke klare. Derudover er spillet meget kedeligt som det er lige nu, da vi som sagt ikke nåede at implementere alle features fra Unity spillet, som vi selv synes er langt sjovere at spille. Ingen af disse ting ville være specielt svære at implementere i AppLab, men det ville bare tage tid, hvilket vi ikke havde nok af, eftersom det viste sig at vi ikke kunne eksportere Unity versionen. Versionen i AppLab skal altså bare demonstrere idéen med vores app og er som sagt en meget tidlig prototype.

Konklusion

Der er stadig ting vi kan gøre for at forbedre performance i AppLab vi kunne fx inddele skærmen i chunks med flere kanvasser. Derefter kunne vi opdatere de kanvasser som der har brug for det og dermed ikke rydde og tegne nyt kanvas hele tiden, men genbruge nogle af de gamle.

Som sagt kunne spillet være sjovere, og generelt mere interessant at se på (lige nu er det bare cirkler og firkanter). Men da appen bare er et proof-of-concept og en meget tidlig prototype, er det godt nok, da vi bare viser ideen.