Start_Her

Hvad er Processing?

Processing er et programmeringssprog og et udviklingsmiljø, der er designet til at gøre det nemt at lære og skabe alt fra visuelle kunstværker, interaktive grafikker, animationer og programmer.

Det blev skabt af Casey Reas og Ben Fry og er baseret på Java, men det har et enklere sprog, der gør det mere tilgængeligt for begyndere og kunstnere.

Hvad er ChatGPT?

ChatGPT er en avanceret sprogmodel, lavet af firmaet OpenAI, der bruger neurale netværk til at forstå og generere tale og skrift.
Den kan bruges til alt fra naturlig sprogbehandling til dialogsystemer og kreativ skrivning.

Når du bruger ChatGPT, giver du den en prompt, typisk i form af tekst.
Ved hjælp af noget kaldet transformer-baserede neurale netværk er ChatGPT i stand til at "læse" og analysere den tekst du giver den, at forstå konteksten og de sproglige mønstre, og give dig svar på din promt.

Selvom ChatGPT er imponerende, har den nogle begrænsninger.
Den kan indimellem give svar, som kan virke sammenhængende, men ikke altid er korrekte eller relevante. Dens svar kan også variere afhængigt af promptens formulering, hvilket betyder, at du ikke altid får de svar som du regner med.

Start_Her

Introduktion

Kode er alle steder omkring os, i alt fra vores telefoner og skærme til vores elnet og kloaksystemer. Men samtidigt er kode det her mystiske sprog, som kun forstås af computere og programmører.

Måske har du engang set kode, og undret dig over hvad der stod. Måske har du endda forstået noget af sproget, men været forvirret over resten.

Heldigvis findes der masser af muligheder for at gøre det nemmere at forstå og lave kode, og med introduktionen af ChatGPT, er det blevet endnu nemmere at tale til computere i dit almindelige sprog, og få chatrobotten til at oversætte.

Formålet med Workshoppen

Du skal gennem denne workshop, lave en række promts til ChatGPT, der beder den om at lave kode til kodesproget Processing. Denne kode kopierer du så over i Processing programmet, der kan køre din kode, og give dig et lille program.

Hvor avanceret programmet bliver, kommer helt an på din frie fantasi, og din evne til at få ChatGPT til at samarbejde med dig.

For at ændre i din kode, skal du skal bare fortsætte samtalen med ChatGPT, og fortælle den hvordan du gerne vil ændre på programmet. Det kan også være der opstår fejl, som du gerne vil fikse. Beskriv dem så godt du kan til ChatGPT, og læs dens svar igennem inden du kopierer koden ind igen. Efter noget tid, vil det føles som om, at du har din helt egen lille robothjælper, der koder sammen med dig!

Dit_Første_Program

En Processing-sketch består typisk af to funktioner: **setup()** og **draw()**. Her er en kort forklaring af begge funktioner:

1. void setup()

setup()-funktionen kører kun én gang, når din sketch starter.

Det er her, du sætter grundlæggende indstillinger for din skitse, såsom skærmstørrelse, baggrundsfarve og andre indledende konfigurationer.

2. void draw()

draw()-funktionen kører konstant efter **setup**()-funktionen og gentages igen og igen i en løkke, indtil sketchen afsluttes.

Denne funktion bruges normalt til at tegne grafik eller udføre animationer.

I dette eksempel bruger vi **draw**() til at tegne en cirkel med center på musemarkørens position (mouseX og mouseY) og en radius på 50 pixels.

Prøv at taste programmet ind som det står på bagsiden, og tryk på playknappen!

- Gør programmet det du troede det ville gøre?
- Kan du forbedre det ved at ændre i koden?

Dit_Første_Program

```
void setup() {
  size(800, 600); // Sætter skærmstørrelsen til 800x600 pixels
  background(255); // Sætter baggrundsfarven til hvid (RGB-værdi 255, 255, 255)
}
void draw() {
  // Tegn en cirkel med center på musemarkøren og radius 50 pixels
  ellipse(mouseX, mouseY, 50, 50);
}
```

Den_Gode_Samtale_01

Den gode samtale med ChatGPT

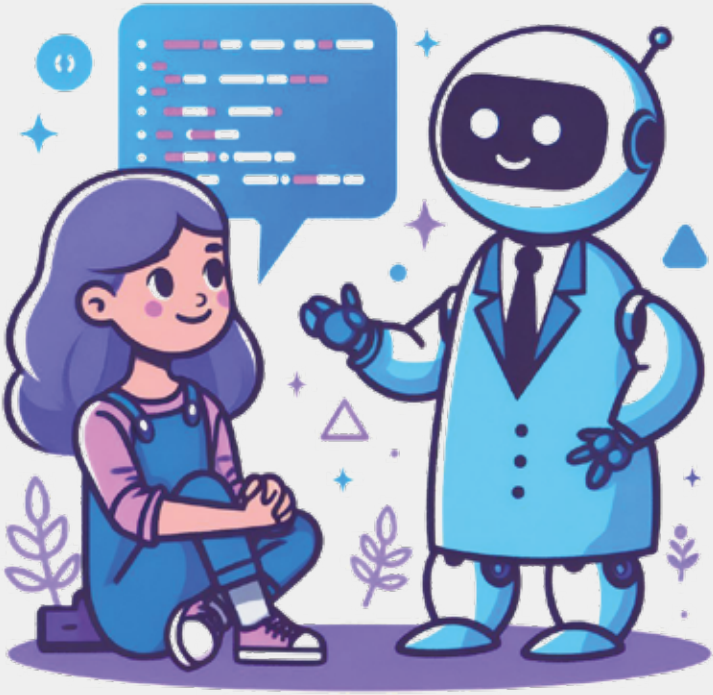
ChatGPT kan være superkraftfuldt værktøj, som kan hjælpe dig med alle mulige slags programmeringsopgaver. Men for at få det bedste ud af ChatGPT, skal du vide, hvordan du snakker med en AI.

Når du skal lære det, kan det hjælpe at se på hvordan du stiller de rigtige spørgsmål, også kendt som det at 'prompte' en AI.

Hvis du ikke er helt præcis med dine spørgsmål, så får du måske ikke den hjælp, du har brug for. Men når du lærer at stille skarpe og klare spørgsmål, åbner du op for en verden af muligheder, hvor ChatGPT kan hjælpe dig med at kode alt fra simple ting til de mere komplicerede projekter.

Det er lidt ligesom at lære et nyt sprog. I starten kan det virke svært, men jo mere du øver dig, desto bedre bliver du. At blive god til at stille de rigtige spørgsmål til ChatGPT kan gøre dine kodningsprojekter sjovere, hurtigere og mere kreative.

Den_Gode_Samtale_02



Billede genereret med DALL·E 3

Promt: Barn får hjælp af en chatrobot til at lave kode.
De taler sammen og er begge glade. 2 tone farver i blå og lilla toner.
Simpel grafik 1990 retrofuturistisk

Den_Gode_Samtale_03

Sådan stiller du de gode spørgsmål:

På dette kort får du nogle gode råd til hvordan du gør din samtale nemmere og mere forståelig for ChatGPT.

Vær specifik:

- **Dårlig forespørgsel:** "Lav en animation af en bold."
  - Bolden bliver placeret i midten af skærmen uden nogen form for bevægelse.
- **Bedre forespørgsel:** "Lav en animation af en bold, der bevæger sig fra venstre til højre på skærmen og reflekterer tilbage, når den rammer kanterne."

Brug trinvise instruktioner:

- **Dårlig forespørgsel:** "Lav en animation af en bold."
- **Bedre forespørgsel:** "Først vil jeg gerne have en funktion, der opretter en bold. Dernæst en funktion, der får den til at bevæge sig."

Den_Gode_Samtale_04

Angiv teknologier eller biblioteker:

- Dårlig forespørgsel: "Lav en boldanimation."
  - Generisk kode leveret, muligvis ikke optimeret til et bestemt bibliotek eller sprog.
- Bedre forespørgsel: "Hvordan laver jeg en animeret bold ved hjælp af Processing?"
  - Svar vil sandsynligvis være skræddersyet til Processing's funktioner og biblioteker.

Beskriv det ønskede resultat:

- Dårlig forespørgsel: "Giv mig en boldanimation."
- Bedre forespørgsel: "Jeg vil gerne have en kode, hvor bolden skifter farve, hver gang den rammer en kant."

Spørg efter delresultater:

- Dårlig forespørgsel: "Lav mig en boldanimation."
  - Generisk kode leveret uden klare trin eller uddybning.
- Bedre forespørgsel: "Start med at vise mig, hvordan jeg opretter en bold i midten af skærmen."
  - Når dette trin er godkendt, kan brugeren gå videre med at spørge om bevægelse, farveskift, osv., hvilket giver mere kontrol over udviklingsprocessen.

RGB_Farver

Sådan virker farvekoder:

Når man laver farver i programmering, bruger man ofte RGB til at lave farver ved at kombinere tre grundlæggende farver:

Rød (Red), grøn (Green) og blå (Blue). Hver af disse farver giver man en værdi fra 0 til 255, hvor 0 betyder ingen farve og 255 betyder maksimal farve. Ved at justere værdierne for rød, grøn og blå kan man skabe en bred vifte af farver. For eksempel vil en RGB-farvekode som (255, 0, 0) producere ren rød, mens (0, 255, 0) giver ren grøn, og (0, 0, 255) giver ren blå. Ved at kombinere disse tre farver med forskellige kan man opnå næsten enhver ønsket farve i digitale billeder og grafik.

Eksempler på farvekoder:

- Rød: (255, 0, 0)
- Grøn: (0, 255, 0)
- Blå: (0, 0, 255)
- Gul: (255, 255, 0)
- Lilla: (128, 0, 128)
- Turkis: (0, 255, 255)

På bagsiden kan du selv prøve kræfter med noget kode, hvor du kan skifte farver på formene.

- **Kan du selv finde på nye farver?**
- **Hvordan tror du man laver hvid og sort?**

RGB_Farver

```
void setup() {
  size(400, 400); // Sætter skærmstørrelsen til 800x600 pixels
  background(255); // Sætter baggrundsfarven til hvid
}
void draw() {
  // Tegn et rektangel
  fill(255, 0, 0);
  rect(50, 50, 100, 100);

  // Tegn en ellipse
  fill(0, 255, 0);
  ellipse(200, 200, 100, 100);

  // Tegn en trekant
  fill(0, 0, 255);
  triangle(300, 300, 350, 300, 325, 250);
}
```

Former_og_størrelser

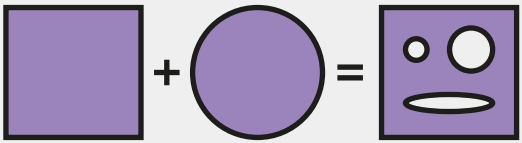
Tænk på former som byggeklodser til at lave tegninger på din computerskærm. Formerne kan være simple, som cirkler og firkanter, eller mere komplekse, som stjerner eller hjerter.

Med en kombination af simple former, kan du lave mange forskellige mønstre og tegninger. For at lave lige den form du har brug for, kan du ændre på en række forskellige elementer, også kaldet parametre

Ændring af Størrelse:

Ændring af Placering:

Kombination af Former:



På bagsiden finder du noget kode med former.

- **Hvad tror du de tegner?**
- **Prøv selv at tegne dit gæt på et stykke papir først. Var det rigtigt?**

Former_og_størrelser

```
void setup(){
  size(400, 400);
}

void draw() {
  background(220);

  // Tegn et rektangel som ansigt
  fill(255); // Hvid farve
  rect(150, 150, 100, 100); // (x, y, bredde, højde)

  // Tegn to øjne som ellipser
  fill(0); // Sort farve
  ellipse(175, 175, 20, 20); // Venstre øje
  ellipse(225, 175, 20, 20); // Højre øje

  // Tegn en mund som en trekant
  fill(255, 0, 0); // Rød farve
  triangle(200, 210, 215, 230, 185, 230); // (x1, y1, x2, y2, x3, y3)
}
```

Koordinatsystemet

Hvordan bruger man koordinatsystemer i processing?:

Forestil dig koordinatsystemet som et stort stykke papir eller en skærm, hvor du kan placere ting ved hjælp af to vigtige oplysninger:
x- og y-koordinater.

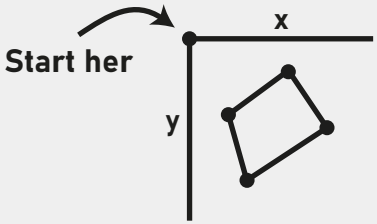
Tænk på x-aksen som den vandrette linje, der går fra venstre mod højre, og y-aksen som den lodrette linje, der går op og ned. Hvert sted på papiret eller skærmen har sine helt egne unikke x- og y-koordinater, ligesom en by på et verdenskort har sine egne bredde- og længdegrader.

Når du lærer at bruge koordinatsystemet, kan du tegne, placere og animere objekter meget præcist, hvilket åbner op for en verden af kreative muligheder i din kode.

VIGTIGT: Koordinatsystemet når man koder tager udgangspunkt i din skærms pixels. Derfor starter koordinatsystemet med første pixel i øverste venstre hjørne af vinduet og slutter i nedereste højre hjørne.

På bagsiden kan du finde noget kode der bruger koordinatsystemet:

- **Hvordan kan du flytte med cirklen?**
- **Kan du sætte andre ting i midten?**



Koordinatsystemet

```
void setup() {
  size(400, 400); // Opret et vindue med en bredde og højde på 400 pixels
  background(220); // Sæt baggrundsfarven til grå
}

void draw() {
  // Tegn en cirkel i midten af skærmen
  float centerX = width / 2; // Beregn x-koordinaten for midten af skærmen
  float centerY = height / 2; // Beregn y-koordinaten for midten af skærmen
  float radius = 50; // Sæt radius for cirklen

  ellipse(centerX, centerY, radius * 2, radius * 2); // Tegn cirklen
}
```

Variabler

Hvad er en variabel?:

I programmering er en variabel som en lille boks eller beholder, hvor du kan gemme information. Forestil dig det som en skattekiste, hvor du kan lægge forskellige ting ind og give dem et navn.

Navnet du giver den kaldes variablens etiket, som du senere kan bruge til at hente eller ændre indholdet i kisten. Lidt ligesom når du sætter etiketter på en bøtte i køleskabet, for at huske hvad for noget mad der ligger i den.

Et eksempel kunne være, hvis du vil have et program der kan huske din alder. Så kan du oprette en variabel med etiketten 'alder' og gemme tallet 15 i den. Senere, når du har brug for at bruge din alder i dit program, kan du bare spørge efter variablen med etiketten 'alder,' og så vil du få tallet 15.

Variabler er meget nyttige, fordi de tillader dig at huske ting i dit program og senere bruge den information.

På bagsiden kan du se et eksempel der bruger variabler til at gemme en persons alder

- **Hvor tit skifter programmet personens alder?**
- **Kan du ændre på hvor hurtigt alderen skifter?**
- **Hvad skal du ændre på, hvis personen starter med at være ældre end 15 år?**

Variabler

```
int alder; // Deklarer en variabel med navnet 'alder' af typen 'int' (heltal)

void setup() {
  size(400, 200);
  alder = 15; // Tildel værdien 15 til variablen 'alder'
}

void draw() {
  background(220);
  textSize(32);
  fill(0);

  // Vis alderen på skærmen
  text("Alder: " + alder, 50, 100);

  // Øg alderen med 1 for hvert frame
  alder = alder + 1;
}
```

If-udsagn

Hvad er et If-udsagn?:

Betingede udsagn (også kaldet 'if-udsagn') er som at have en vej der deler sig i to i din kode.

Forestil dig, at du spiller et spil, og du vil kontrollere, om din karakter er i live eller død. Du kan bruge et if-udsagn til at sige, 'Hvis karakterens liv er større end 0, så fortsæt spillet; ellers, hvis livet er lig med eller mindre end 0, så er spillet slut.'

Med andre ord giver if-udsagn dig mulighed for at fortælle dit program, hvad det skal gøre, afhængigt af en bestemt situation.

På bagsiden kan du se et ekempel på et if-udsagn, der viser en situation du måske kender til:

- **Hvordan kan du ændre i koden, så du får forskellige beskeder på skærmen?**
- **Kan du finde på andre situationer, hvor de kunne være smart at tjekke om et tal er større eller mindre end et andet?**

If-udsagn

```
int alder = 18; // Laver en variabel 'alder' og tildeler den værdien 18

void setup() {
  size(400, 200);
}

void draw() {
  background(220);
  textSize(32);
  fill(0);

  // Tjek om alderen er over 17
  if (alder > 17) {
    text("Du er myndig!", 50, 100);
  } else {
    text("Du er ikke myndig endnu.", 50, 100);
  }
}
```

Loop (for)

Loop (for) (for-løkke)

En for-løkke som en slags 'gentagelsesmaskine.' Tænk på det som en bager, der bager 10 ens kager på én gang ved hjælp af en stor bageform.

For-løkken gør det samme - den gentager en opgave (eller bager en kage) igen og igen, indtil du fortæller den at stoppe. Opgaven kan ses som ingredienserne til kagen, og for-løkken er bageformen.

Hver gang for-løkken kører, udfører den opgaven med en lille ændring, som at tilføje 1 til en tæller. Ved at tilføje 1, hver gang for-løkken bliver aktiveret, kan vi holde styr på hvor mange gange for-løkken har kørt, og feks begrænse den til kun 'at bage 10 kager'.

Så, ligesom bageren kan bage en række lækre kager ved at gentage processen, kan du bruge en for-løkke til at gentage en opgave mange gange og oprette komplekse mønstre eller lave det samme opgave et bestemt antal gange.

På bagsiden kan du se et eksempel på en for-løkke:

- **Programmet udskriver 10 tal, hvad skal du gøre for at gøre det med 20 eller 100?**
- **Hvad kunne ellers være smart at tælle på den her måde?**

Loop (for)

```
void setup() {
  size(400, 400);
  textSize(32);
  fill(0);
}

void draw() {
  background(220);

  // Brug en for-løkke til at udskrive de første 10 naturlige tal
  for (int i = 1; i <= 10; i++) {
    text("Tal " + i, 50, 50 + i * 30);
  }
}
```

Arrays

Hvad er et Array?:

Et array er som en liste med nummererede kasser, hvor du kan gemme flere elementer af samme type.

Forestil dig en række ens skuffer, hvor hver skuffe har sit eget nummer. Hver skuffe kan indeholde noget forskelligt, som tal, tekst eller endda billeder. Det kan være smart, når du har brug for at organisere og arbejde med mange lignende ting på én gang.

Hvis du for eksempel skulle gemme en liste med 10 navne, så ville du kunne bruge et array til at opbevare dem allesammen på en struktureret måde (f.eks i alfabetisk orden, eller efter hvor gamle personerne på listen er). Når du har gjort det, så er det nemt at få adgang til hvert navn ved at henvise til navnets position i arrayet.

Når man skal genfinde tallet bruger man et index, så hvis man feks skal finde det femte tal i rækken, kan man skrive tal[5]

Man bruger arrays for at gøre det nemmere at håndtere og manipulere store mængder data i programmer.

På bagsiden kan du se et ekempel på hvordan man kan bruge et array:

- **Ekemplet viser alle tallene på skærmen. Hvad ville du skulle skrive, for at få vist et bestemt tal?**
- **Kan du tænke på andre situationer, hvor det kunne være smart at kunne genfinde et tal i en række af andre tal?**
- **Hvis du skulle gemme andre ting end tal, hvordan tænker du så at man kunne bruge et array?**

Arrays

```
int[ ] tal = {5, 10, 15, 20, 25}; // Opret et array med tal

void setup() {
  size(400, 200);
  textSize(32);
  fill(0);
}

void draw() {
  background(220);

  // Vis alle tal i arrayet på skærmen
  for (int i = 0; i < tal.length; i++) {
    text("Tal " + i + ": " + tal[i], 50, 50 + i * 30);
  }
}
```

Funktioner

Hvordan kan man bruge funktioner?

En funktion er som en opskrift eller en lille, selvstændig opgave, som din computer kan udføre.

Du kan oprette dine egne funktioner til at udføre specifikke opgaver, og når du vil udføre den opgave, skriver du bare funktionen, i stedet for at skrive hele koden igen. Det gør dit program mere organiseret, nemmere at forstå og lettere at ændre i, da du kun behøver at skifte opskriften ét sted, hvis du vil ændre dens adfærd overalt, hvor den bruges.

På bagsiden kan du finde et eksempel på en simpel funktion:

- **Hvad skal du ændre på, hvis du gerne vil have funktionen til at sige noget andet?**
- **Kan du finde på andre ting, der kunne være smart at kunne gentage på en nem måde?**

Funktioner

```
void setup() {
  size(400, 200);
  textSize(32);
  fill(0);

  // Kald funktionen 'hilsen'
  hilsen();
}

void draw() {
  // Intet behøver at gøres i 'draw' i dette eksempel
}

// Definition af en funktion kaldet 'hilsen'
void hilsen() {
  background(220);
  text("Hej verden!", 50, 100);
}
```

Mus- og tastatur

Hvordan kan musen og tastaturet bruges til at styre dit program?

Når dit program kører, kan det være at du gerne vil styre det, uden at skulle ændre i koden hele tiden. Der kan det være smart med mus- og tastaturinteraktioner, så du kan kommunikere med programmet mens det er i gang.

Musen kan bruges til at se hvor du klikker, og hvordan du bevæger den, hvilket giver dig mulighed for at påvirke objekter på skærmen.
For eksempel kan du bruge musen til at tegne, trække objekter rundt eller klikke på knapper i et spil.

Tastaturet giver dig mulighed for at indtaste tekst eller kommandoer, som din computer kan forstå. For eksempel kan du bruge tastaturet til at skrive beskeder, flytte en spilfigur eller udføre handlinger i et tekstbaseret program.

På bagsiden kan du se nogle eksempler på hvordan man kan bruge musen og tastaturet:

- **Hvordan ville du skrive koden, hvis du skulle bevæge cirklen i andre retninger end op med tastaturet?**
- **Kan du lave det sådan at der er flere ting på skærmen, og nogle af dem kan styres med musen, mens andre styres med tastaturet?**
- **Kan du finde på andre måder at bruge musen på?**

Mus- og tastatur

```
float cirkelX = 200; // X-koordinat for cirkel
float cirkelY = 200; // Y-koordinat for cirkel

void setup() {
  size(400, 400);
  background(220);
}

void draw() {
  // Tegn en cirkel, der kan flyttes med musen
  ellipse(cirkelX, cirkelY, 50, 50);

  // Hvis tasten 'W' trykkes, flyt cirklen opad
  if (keyPressed && key == 'w') {
    cirkelY -= 5;
  }
}

 void mousePressed() {
  // Flyt cirklen til musens position, når musen klikkes
  cirkelX = mouseX;
  cirkelY = mouseY;
}
```