

# JS: Partikkel-animasjon

*Skrevet av: Lars Klingenberg*

*Oversatt av: Stein Olav Romslo*

*Kurs: Web*

*Tema: Tekstbasert, Nettside, Animasjon*

*Fag: Matematikk, Programmering*

*Klassetrinn: 5.-7. klasse, 8.-10. klasse, Videregående skole*

## Introduction

I denne oppgåva skal me bruke JavaScript til å få figurar til å bevege seg. Me skal altså lære å animere ved hjelp av JavaScript og noko som heiter `canvas`. Under ser du animasjonen me skal lage.

Oppgåva er den fyrste i ei lita serie av andre `partikkel`-oppgåver, difor er det viktig å forstå det som skjer i denne oppgåva.



I denne oppgåva får du bruk for det du lærte i oppgåva Grunnleggjande JavaScript (`../grunnleggende_js/grunnleggende_js_nn.html`).

## Steg 1: Canvas-elementet

I HTML brukar me `<canvas>` til å teikne figurar ved hjelp av JavaScript. Sjølve `<canvas>`-elementet gjer ikkje så mykje nytte, så difor brukar me JavaScript til å fortelje kva grafikk `<canvas>`-elementet skal innehalde. La oss skrive det som trengst for å

jobbe med canvas :

- ☐ Åpne favoritt-teksteditoren din.
- ☐ Lag ei ny HTML-fil som heiter `partiklar.html`
- ☐ Kopier koden under inn i `partiklar.html` :

```
<html>
<head>
  <meta charset="UTF-8">
  <title>Partikkel-fest</title>
  <style>
    body {
      background-color:#666;
    }

    #canvas {
      background-color:#000;
      margin-left:100px;
    }
  </style>
</head>
<body>
  <canvas id="canvas" width="500" height="500"></canvas>
</body>
</html>
```

## Forklaring: Canvas

- ☐ `<canvas id="canvas" width="500" height="500"></canvas>` er sjølve Canvas -elementet. Den har ei gitt høgde og breidde 500px x 500px . Me skal bruke JavaScript til å lage andre element inne i canvas -elementet.
- ☐ I CSS-en er det lagt til ei grå bakgrunnsfarge til `<body>` og svart bakgrunnsfarge til `<canvas>` .

# Steg 2: Teikne eit objekt

No som me veit korleis `canvas` ser ut er det på tide å prøve det ut:

☐ Set inn `<script>` `</script>` i koden din.

☐ Lag to tomme variablar:

```
var canvas;  
var ctx;
```

☐ No skal me fylle desse variablane når sida vår blir lasta inn. Då brukar me noko som heiter `window.onload`:

```
window.onload = function() {  
    canvas = document.getElementById("canvas");  
    ctx = canvas.getContext("2d");  
}
```

No heldt `canvas`-variabelen på HTML-elementet vårt.

`ctx`-variabelen vil vere det grafiske elementet som blir lagt til `canvas`. Me kan manipulere dette elementet ved hjelp av stil, som me snart skal sjå på.

For å kunne lage grafikk i `canvas` er dei to linjene over nødvendige. No som me har dei på plass kan me starte å teikne!

☐ No skal me lage objekt, så la oss lære litt om kva eit objekt er:

## Forklaring: Objekt

La oss lage eit objekt som skal teiknast. I JavaScript er eit objekt ein variabel som kan halde på fleire verdier eller variablar, som me ofte kallar attributtar. La oss sjå på eit raskt døme med ein bil:

```
var bil = {  
  namn: "Volkswagen",  
  modell: "Golf"  
  antalSeter: 5,  
  farge: "Blå",  
};
```

Det er enkelt å hente ut informasjonen me vil ha frå objektet ved å skrive følgjande:

```
console.log(bil.namn); // Skriv ut namnet på bilen: Volkswagen  
console.log(bil.farge); // Skriv ut farga på bilen: Blå
```

For å endre på ein av attributtane gjer me berre følgjande:

```
bil.farge = "Raud";
```

No blir attributten `farge` endra frå `Blå` til `Raud`.

På denne måten slepp me å lage mange variablar som skal høyre til same element, me brukar heller objekt.

- ☐ Lag eit objekt som heiter `particle` og som inneheldt dei følgjande attributtane: `x`-posisjon, `y`-posisjon, `storleik` og `farge`.
- ☐ Du set sjølv passande verdiar for attributtane. Desse kan det vere lurt å eksperimentere litt med seinare i oppgåva.

Hint

- ☐ Lag ein funksjon som heiter `draw`. Denne skal teikne elementet for oss.
- ☐ I `draw` skal me leggje til kva farge me vil at elementet skal ha. Du bestemmer sjølv kva farge det skal vere:

```
ctx.fillStyle = particle.farge;
```

☐ No skal me teikne eit kvadrat (ein firkant der alle sidene er like lange)

☐ i farga me valte over:

```
ctx.fillRect(particle.x,particle.y,particle.size,particle.size);
```

## Forklaring: ctx.fillRect()

ctx.fillRect() tek inn 4 variablar:

```
ctx.fillRect(x-posisjon, y-posisjon, breidde, høgde);
```

Over brukte me dei attributtane me laga i objektet `particle`.

I objektet `particle` har me sett ein `x`- og `y`-posisjon, samt ein storleik som me set på både breidde og høgde for å få eit kvadrat.

☐ Lagre og køyr funksjonen `draw()` når sida lastar.

Forslag til koden så langt:

```
CTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title></title>
  <style>
    body {
      background-color: #666;
    }

    #canvas {
      background-color: #000;
      margin-left: 100px;
    }
  </style>
  <script>

    var canvas;
    var ctx;

    var particle = {
      x: 0,
      y: 0,
      size: 10,
      farge: "red"
    };

    window.onload = function() {
      canvas = document.getElementById("canvas");
      ctx = canvas.getContext("2d");
      draw();
    };

    //Teiknar particle
    function draw() {
      ctx.fillStyle = particle.farge;
      ctx.fillRect(particle.x, particle.y,particle.size,particle.
size);
    };

  </script>

</head>
<body>

<canvas id="canvas" width="500" height="500"></canvas>
```

```
</body>
</html>
```

## Steg 3: Flytt på partikkelen

No som me har fått fram ein raud firkant, som er partikkelen vår, så skal me sjå korleis me får den til å flytte på seg. For å få det til å skje må me leggje til nokre nye attributtar i objektet vårt, og endre desse undervegs i funksjonen vår. For å gjere det må me lære å bruke `setInterval`, men fyrst må me endre på objektet vårt.

☐ I objektet `particle`, legg til attributtane `xSpeed` og `ySpeed`.

☐ Set verdiane `xSpeed` og `ySpeed` til å vere 2 for no.

I `draw` må me endre `particle` sin `x`-posisjon med `xSpeed`, og tilsvarande for `y`-posisjonen. Måten du aukar ein attributt på er slik:

```
objekt.attributt1 = objekt.attributt1 + objekt.attributt2;
```

☐ Legg til det som trengst i `draw` for å få `particle` til å endre `x`- og `y`-posisjonen sin.

Hint

For at me skal få ein animasjon må me køyre `draw` fleire gonger enn berre 1, difor må me bruke `setInterval` for å gjenta `draw`.

☐ Køyr funksjonen `draw` kvart 30. millisekund:

```
setInterval(draw, 30);
```

## Forklaring: `setInterval`

☐ `setInterval` køyrer ein funksjon kvart X. millisekund.

- ☐ Altså tyder `setInterval(draw, 30);` at funksjonen `draw()` kjørest hvert 30. millisekund. NB! 1000 millisekund er eitt sekund.

- ☐ Fjern `draw()`, me treng den ikkje lengre, sidan `setInterval` vil køyre `draw` for oss.
- ☐ Lagre og køyr sida me har laga til no!

Som du ser blir det laga ei lang diagonal stripe. Som du kanskje har skjønnt må me finne ein måte å fjerne den førre boksen me teikna slik at me skapar ein illusjon av at den flyttar på seg, ikkje berre at me teiknar mange etter kvarandre.

- ☐ I starten av `draw` må me bruke `ctx.clearRect(0,0,500,500);` for å fjerne alt som er innanfor det svarte. Altså frå  $(x, y)$ -posisjonen  $(0, 0)$  og heilt til  $(500, 500)$ .
- ☐ Lagre og køyr på nytt!

**Gratulerer! Du har laga din fyrste animasjon i JavaScript!**

## Utfordring

- ☐ Prøv å få partikkelen til å gå rett fram.
- ☐ Få partikkelen til å gå rett ned.
- ☐ Få partikkelen til å gå baklengs.
- ☐ Får du til at partikkelen byttar til ei tilfeldig farge kvar gong den byttar posisjon?



Døme på ferdig kode til oppgåva:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title></title>
  <style>
    body {
      background-color:#666;
    }

    #canvas {
      background-color:#000;
      margin-left:100px;
    }
  </style>
  <script>

    var canvas, ctx;

    var particle = {
      x: 0,
      y: 0,
      xSpeed: 2,
      ySpeed: 2,
      size: 10,
      farge: "red"
    };

    window.onload = function() {
      canvas = document.getElementById("canvas");
      ctx = canvas.getContext("2d");
      setInterval(draw, 30);
    };

    //Teiknar og skyt particle opp
    function draw() {

      ctx.clearRect(0,0,500,500);

      ctx.fillStyle = particle.farge;
      ctx.fillRect(particle.x, particle.y,particle.size,particle.
size);

      particle.x = particle.x + particle.xSpeed;
      particle.y = particle.y + particle.ySpeed;
```

```
}
```

```
</script>
```

```
</head>
```

```
<body>
```

```
<canvas id="canvas" width="500" height="500"></canvas>
```

```
</body>
```

```
</html>
```

Lisens: CC BY-SA 4.0 (<http://creativecommons.org/licenses/by-sa/4.0/deed>)