Spelprogrammering med JavaScript och HTML5

Vad är vårt syfte?

- Eleverna ska komma igång fort med programmering.
- Det ska vara roligt från början.
- Resultatet ska vara enkelt att visa för kompisar och familj.
- Det ska vara lätt att skriva koden oavsett operativsystem.

Ett enkelt exempel med `library.js`

Så här ser kan koden se ut i utgåva ett av vår bok.

`litetspel.html`

```
<html>
  <head>
    <title>Lek med canvas</title>
    <script src="http://spelprogrammering.nu/library.js"></script>
    <script src="litetspel.js"></script>
  </head>
  <body>
    <canvas id="canvas" width="800" height="400"></canvas>
  </body>
</html>
```

`litetspel.js`

```
function start()
{
  circle(200, 150, 70, "red");
}
```

Problem vi har stött på

- Nybörjare skrev av HTML-koden fel.
- Nybörjare hade problem med att skriva två kodfiler, speciellt när de skulle börja med nästa uppgift och skapa två nya filer.
- Förenklingarna för nybörjare var begränsande för avancerade elever.

Vi hade gjort det hyfsat enkelt för nybörjare att sätta igång och hyfsat enkelt för duktiga elever att fortsätta. Men vi ville att det skulle vara mycket enklare för nybörjare, och samtidigt lättare att bygga vidare för de duktigaste eleverna...

Vår lösning blev en uppdelning i **två bibliotek**. Ett som förenklar så mycket som det bara går och ett som passar in i mer avancerade lösningar.

Det enkla biblioteket - `simple.js`

- Eleverna skapar bara **en** kodfil, t.ex. *mittspel.html*
- De kommer igång direkt med programmeringen, mängden kod som behöver skrivas av utan att förstå är minimerad.
- Koden följer inte nödvändigtvis god sed eller standarder, vi gör allt för att det ska bli så enkelt som möjligt.

Det här är hela HTML-filen som behöver skrivas av:

```
<script src="http://spelprogrammering.nu/simple.js"></script>
<script>

</script>
```

Notera att all vanlig HTML är borta, eleven ska inte behöva ha sett HTML innan. Mellan de två sista script-taggarna skriver man sin JavaScript-kod. Exempelvis:

```
<script src="http://spelprogrammering.nu/simple.js"></script>
<script>
  function start()
  {
    circle(100, 200, 20, 'red');
  }

  function update()
  {
    circle(mouse.x, mouse.y, 20, 'blue');
  }
</script>
```

Detta ger en röd cirkel på x-koordinat 100 och y-koordinat 200, och en blå cirkel som följer muspekaren.

Det enkla biblioteket gör alltså att man nästan helt slipper skriva HTML-kod och man behöver bara skapa en fil. Eleverna kommer igång i princip direkt och får automatiskt tillgång till hela webbläsarens yta.

Vi använder oss av det enkla biblioteket för att lära ut grunderna i programmering, bygger enkla spel och experimenterar med fysik och avancerad matematik. I princip samma saker som redan finns i vår bok, men det blir lättare att komma igång med.

Det avancerade biblioteket: `advanced.js`

- Eleven skriver hela HTML-koden själv, inklusive canvas-element. Antagligen skrivs JavaScript i separata filer.
- Målet är att HTML-koden ska följa standarder och god programmeringssed.
- Eleven kan ha flera canvas-element och experimentera med att använda konceptet för annat än spel. Exempelvis effekter på hemsidor.
- Man kan enkelt använda biblioteket medan man lär sig HTML och CSS.

Exempel där eleven gjort allt i en HTML-fil:

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8" />
    <title>Mitt spel</title>
    <style>
      #one
      {
        border: 1px solid red;
      }
      #two
      {
        border: 1px solid blue;
      }
    </style>
  </head>
  <body>
    <canvas id="one" width="800" height="200"></canvas>
    <canvas id="two" width="800" height="200"></canvas>

    <script src="http://spelprogrammering.nu/advanced.js"></script>
    <script>
      a = new RoboroCanvas('one');
      b = new RoboroCanvas('two');

      a.update = function()
      {
        a.circle(a.mouse.x, a.mouse.y, 10, 'brown');
      };

      b.update = function()
      {
        b.circle(b.mouse.x, b.mouse.y, 10, 'yellow');
      };
    </script>
  </body>
</html>
```

Med det avancerade biblioteket måste eleven själv hantera all HTML och CSS och även skapa canvas-elementen. Som exemplet ovan visar kan man skapa flera canvas-element som blir helt oberoende av varandra.

Eleverna uppmuntras att utnyttja allt som är häftigt med JavaScript, och integrera det med korrekt HTML och CSS.

Vi har delat upp funktionaliteten i `advanced.js` i flera API:er som alla bygger på objektorientering. Själva canvas-ritandet, inklusive musrörelser och pethändelser, går genom RoboroCanvas, tangentbordsinteraktion genom RoboroKeyboard och ljud genom RoboroSound.

Några exempel med `simple.js`

En röd boll

```
<script src="http://spelprogrammering.nu/simple.js"></script>
<script>
 function start()
 {
   circle(100, 200, 20, 'red');
 }
</script>
```

Styr färg och storlek med variabler

```
<script src="http://spelprogrammering.nu/simple.js"></script>
<script>
 function start()
 {
   size  = 20;
   color = 'red';
   circle(100, 200, size, color);
 }
</script>
```

Fråga användaren om färg och storlek

```
<script src="http://spelprogrammering.nu/simple.js"></script>
<script>
 function start()
 {
   size  = prompt('What size?');
   color = prompt('What color?');
   circle(100, 200, size, color);
 }
</script>
```

Rita många bollar med en loop

```
<script src="http://spelprogrammering.nu/simple.js"></script>
<script>
 function start()
 {
   size  = prompt('What size?');
   color = prompt('What color?');

   for (i = 0; i < 100; i += 1)
     circle(100 + i, 200 + i, size, color);
 }
</script>
```

Animation

```
<script src="http://spelprogrammering.nu/simple.js"></script>
<script>
  function start()
  {
    x = 100;
  }

  function update()
  {
    clearScreen();
    circle(x, 200, 10, 'red');

    x += 1;
  }
</script>
```

Interaktion med musen, bollen följer musen

```
<script src="http://spelprogrammering.nu/simple.js"></script>
<script>
  function update()
  {
    clearScreen();
    circle(mouse.x, mouse.y, 10, 'red');
  }
</script>
```

Interaktion med tangentbordet, ny boll styrs med pilarna

```
<script src="http://spelprogrammering.nu/simple.js"></script>
<script>
  function start()
  {
    cat = {};
    cat.x = 100;
    cat.y = 100;
    cat.speed = 10;
  }
  function update()
  {
    clearScreen();
    circle(mouse.x, mouse.y, 20, 'red');
    circle(cat.x, cat.y, 20, 'blue');

    if (keyboard.up)
      cat.y -= cat.speed;
    if (keyboard.down)
      cat.y += cat.speed;
```

```
    if (keyboard.left)
      cat.x -= cat.speed;
    if (keyboard.right)
      cat.x += cat.speed;
  }
</script>
```

Avsluta spelet om bollarna kommer nära, lägg till poängräknare, byt ut bollarna

```
<script src="http://spelprogrammering.nu/simple.js"></script>
<script>
  function start()
  {
    cat = {};
    cat.x = 300;
    cat.y = 300;
    cat.speed = 10;

    score = 0;
  }

  function update()
  {
    clearScreen();
    picture(cat.x - 200, cat.y - 150, 'tom.png');
    picture(mouse.x - 50, mouse.y - 50, 'jerry.png');

    text(20, 40, 20, 'Score: '+score);

    if (keyboard.up)
      cat.y -= cat.speed;
    if (keyboard.down)
      cat.y += cat.speed;
    if (keyboard.left)
      cat.x -= cat.speed;
    if (keyboard.right)
      cat.x += cat.speed;

    if (distance(mouse.x, mouse.y, cat.x, cat.y) < 40)
      stopUpdate();

    score++;
  }
</script>
```

Förkorta koden

```
<script src="http://spelprogrammering.nu/simple.js"></script>
<script>
  function start()
  {
    cat = { x: 300, y: 300, speed: 10 };
    score = 0;
  }

  function update()
  {
    clearScreen();
    picture(cat.x  - 200, cat.y  - 150, 'tom.png');
    picture(mouse.x - 50, mouse.y - 50, 'jerry.png');

    text(20, 40, 20, 'Score: ' + score);

    cat.x += (keyboard.right - keyboard.left) * cat.speed;
    cat.y += (keyboard.down  - keyboard.up)   * cat.speed;

    if (distance(mouse.x, mouse.y, cat.x, cat.y) < 40)
      stopUpdate();

    score++;
  }
</script>
```

Vad händer nu

Köp gärna vår bok och ge oss tips och idéer på hur vi kan bli bättre. Vi håller på för fullt med att skriva en ny utgåva där `simple.js` och `advanced.js` används. Denna beräknas vara färdig om några månader.

Vi kommer att hålla `http://www.spelprogrammering.nu/` uppdaterad med mer information.