Die canvas-Uhr

Wir „bauen“ uns eine **Analoguhr** mit HTML **canvas**.

Part I - Create the Canvas

The clock needs an **HTML container**. Create an **HTML canvas**:

HTML code:

```
<!DOCTYPE html>
<html>
  <body>
    <canvas id="canvas" width="400" height="400"
            style="background-color:#333">
    </canvas>
    <script>
      var canvas = document.getElementById("canvas");
      var ctx    = canvas.getContext("2d");
      var radius = canvas.height / 2;

      ctx.translate(radius, radius);
      radius = radius * 0.90
      drawClock();

      function drawClock()
      {
        ctx.arc(0, 0, radius, 0 , 2 * Math.PI);
        ctx.fillStyle = "white";
        ctx.fill();
      }
    </script>
  </body>
</html>
```

Ich erkläre den Code:

Add an HTML **<canvas>** element to your page:

```
<canvas id="canvas" width="400" height="400" style="background-color:#333"></canvas>
```

Create a canvas object (**var canvas**) from the HTML canvas element:

```
var canvas = document.getElementById("canvas");
```

Create a **2d drawing object** (**var ctx**) for the **canvas** object:

```
var ctx = canvas.getContext("2d");
```

Calculate the clock **radius**, using the **height** of the **canvas**:

```
var radius = canvas.height / 2;
```

Using the **canvas height** to calculate the clock **radius**, makes the clock work for all canvas sizes.
Remap the **(0,0)** position (of the drawing object) to the center of the canvas:

```
ctx.translate(radius, radius);
```

Reduce the clock **radius** (to **90%**) to draw the clock well inside the canvas:

```
radius = radius * 0.90;
```

Create a function to draw the clock:

```
function drawClock()
{
  ctx.arc(0, 0, radius, 0 , 2 * Math.PI);
  ctx.fillStyle = "white";
  ctx.fill();
}
```

Part II - Draw a Clock Face

The clock needs a clock face. Create a JavaScript function to draw a clock face:

JavaScript:

```
function drawClock()
{
  drawFace(ctx, radius);
}

function drawFace(ctx, radius)
{
  var grad;

  ctx.beginPath();
  ctx.arc(0, 0, radius, 0, 2 * Math.PI);
  ctx.fillStyle = 'white';
  ctx.fill();

  grad = ctx.createRadialGradient(0, 0 ,radius * 0.95, 0, 0, radius * 1.05);
  grad.addColorStop(0, '#333');
  grad.addColorStop(0.5, 'white');
  grad.addColorStop(1, '#333');
  ctx.strokeStyle = grad;
  ctx.lineWidth   = radius * 0.1;
  ctx.stroke();

  ctx.beginPath();
  ctx.arc(0, 0, radius * 0.1, 0, 2 * Math.PI);
  ctx.fillStyle = '#333';
  ctx.fill();
}
```

Ich erkläre den Code:

Create a **`drawFace()`** function for **drawing the clock face**:

```
function drawClock()
{
  drawFace(ctx, radius);
}

function drawFace(ctx, radius)
{
}
```

Draw the **white circle**:

```
ctx.beginPath();
ctx.arc(0, 0, radius, 0, 2 * Math.PI);
ctx.fillStyle = 'white';
ctx.fill();
```

Create a **`radial gradient`** (**`95%`** and **`105%`** of original clock radius):

```
grad = ctx.createRadialGradient(0, 0, radius * 0.95, 0, 0, radius * 1.05);
```

Create **`3 color stops`**, corresponding with the inner, middle, and outer edge of the arc:

```
grad.addColorStop(0, '#333');
grad.addColorStop(0.5, 'white');
grad.addColorStop(1, '#333');
```

The color stops create a **`3D`** effect.

Define the gradient as the **`stroke style`** of the drawing object:

```
ctx.strokeStyle = grad;
```

Define the **`line width`** of the drawing object (**`10%`** of radius):

```
ctx.lineWidth = radius * 0.1;
```

Draw the **`circle`**:

```
ctx.stroke();
```

Draw the **clock center**:

```
ctx.beginPath();
ctx.arc(0, 0, radius * 0.1, 0, 2 * Math.PI);
ctx.fillStyle = '#333';
ctx.fill();
```

Part III - Draw Clock Numbers

The clock needs **numbers**. Create a JavaScript **function to draw clock numbers**:

JavaScript:

```
function drawClock()
{
  drawFace(ctx, radius);
  drawNumbers(ctx, radius);
}

function drawNumbers(ctx, radius)
{
  var ang;
  var num;

  ctx.font         = radius * 0.15 + "px arial";
  ctx.textBaseline = "middle";
  ctx.textAlign    = "center";

  for (num = 1;num < 13;num++)
  {
    ang = num * Math.PI / 6;
    ctx.rotate(ang);
    ctx.translate(0, -radius * 0.85);
    ctx.rotate(-ang);
    ctx.fillText(num.toString(), 0, 0);
    ctx.rotate(ang);
    ctx.translate(0, radius * 0.85);
   ctx.rotate(-ang);
  }
}
```

Ich erkläre das Beispiel:

Set the font size (of the drawing object) to 15% of the radius:

```
ctx.font = radius * 0.15 + "px arial";
```

Set the text alignment to the middle and the center of the print position:

```
ctx.textBaseline = "middle";
ctx.textAlign    = "center";
```

Calculate the print position (for 12 numbers) to 85% of the radius, rotated (PI/6) for each number:

```
for (num = 1;num < 13;num++)
{
  ang = num * Math.PI / 6;
  ctx.rotate(ang);
  ctx.translate(0, -radius * 0.85);
  ctx.rotate(-ang);
  ctx.fillText(num.toString(), 0, 0);
  ctx.rotate(ang);
  ctx.translate(0, radius * 0.85);
  ctx.rotate(-ang);
}
```

Part IV - Draw Clock Hands

Die Uhr benötigt Zeiger. Erstelle eine **JavaScript Funktion** zum **Zeichnen von Zeigern**.

JavaScript:

```
function drawClock()
{
  drawFace(ctx, radius);
  drawNumbers(ctx, radius);
  drawTime(ctx, radius);
}

function drawTime(ctx, radius)
{
  var now    = new Date();
  var hour   = now.getHours();
  var minute = now.getMinutes();
  var second = now.getSeconds();

  // hour
  hour = hour % 12;
  hour = (hour   * Math.PI / 6)          +
         (minute * Math.PI / (  6 * 60)) +
         (second * Math.PI / (360 * 60));

  drawHand(ctx, hour, radius*0.5, radius * 0.07);

  // minute
  minute = (minute * Math.PI / 30)       +
           (second * Math.PI / (30 * 60));
  drawHand(ctx, minute, radius * 0.8, radius * 0.07);

  // second
  second = (second*Math.PI / 30);
  drawHand(ctx, second, radius * 0.9, radius * 0.02);
}

function drawHand(ctx, pos, length, width)
{
  ctx.beginPath();
  ctx.lineWidth = width;
  ctx.lineCap = "round";
  ctx.moveTo(0,0);
  ctx.rotate(pos);
  ctx.lineTo(0, -length);
  ctx.stroke();
  ctx.rotate(-pos);
}
```

Ich erkläre das Beispiel:

Use **`Date`** to get **`hour`**, **`minute`**, **`second`**:

```
var now    = new Date();
var hour   = now.getHours();
var minute = now.getMinutes();
var second = now.getSeconds();
```

Calculate the **`angle`** of the **`hour`** hand, and draw it a **`length`** (**`50%`** of **`radius`**), and a **`width`** (**`7%`** of **`radius`**):

```
hour = hour%12;
hour = (hour*Math.PI/6)+(minute*Math.PI/(6*60))+(second*Math.PI/(360*60));
drawHand(ctx, hour, radius*0.5, radius*0.07);
```

Use the same technique for minutes and seconds.
The **`drawHand()`** routine does not need an explanation. It just draws a line with a given length and width.

Part V - Start the Clock

To start the clock, call the **drawClock** function at intervals:

JavaScript:

```
var canvas = document.getElementById("canvas");
var ctx    = canvas.getContext("2d");
var radius = canvas.height / 2;

ctx.translate(radius, radius);
radius = radius * 0.90

// drawClock();
setInterval(drawClock, 1000);
```

Ich erkläre das Beispiel:

The only thing you have to do (to start the clock) is to call the **drawClock** function at **intervals**.
Substitute:

```
drawClock();
```

With:

```
setInterval(drawClock, 1000);
```

The interval is in milliseconds. drawClock will be called for each 1000 milliseconds.

Der komplette Code:

```
<!DOCTYPE html>
<html>
  <body>
    <canvas id="canvas" width="400" height="400" style="background-color:#333"></canvas>
    <script>
      var canvas = document.getElementById("canvas");
      var ctx    = canvas.getContext("2d");
      var radius = canvas.height / 2;

      ctx.translate(radius, radius);
      radius = radius * 0.90
      setInterval(drawClock, 1000);

      function drawClock()
      {
        drawFace(ctx, radius);
        drawNumbers(ctx, radius);
        drawTime(ctx, radius);
      }

      function drawFace(ctx, radius)
      {
        var grad;

        ctx.beginPath();
        ctx.arc(0, 0, radius, 0, 2*Math.PI);
        ctx.fillStyle = 'white';
        ctx.fill();
        grad = ctx.createRadialGradient(0,0,radius*0.95, 0,0,radius*1.05);
        grad.addColorStop(0, '#333');
        grad.addColorStop(0.5, 'white');
        grad.addColorStop(1, '#333');
        ctx.strokeStyle = grad;
        ctx.lineWidth   = radius*0.1;
        ctx.stroke();
        ctx.beginPath();
        ctx.arc(0, 0, radius*0.1, 0, 2*Math.PI);
        ctx.fillStyle = '#333';
        ctx.fill();
      }

      function drawNumbers(ctx, radius)
      {
        var ang;
        var num;

        ctx.font         = radius * 0.15 + "px arial";
        ctx.textBaseline = "middle";
        ctx.textAlign    = "center";

        for (num = 1;num < 13;num++)
        {
          ang = num * Math.PI / 6;

          ctx.rotate(ang);
          ctx.translate(0, -radius * 0.85);
          ctx.rotate(-ang);
          ctx.fillText(num.toString(), 0, 0);
          ctx.rotate(ang);
          ctx.translate(0, radius * 0.85);
          ctx.rotate(-ang);
        }
      }

      function drawTime(ctx, radius)
      {
        var now    = new Date();
        var hour   = now.getHours();
        var minute = now.getMinutes();
        var second = now.getSeconds();

        // hour
        Hour = hour % 12;
        hour = (hour   * Math.PI / 6)          +
               (minute * Math.PI / (  6 * 60)) +
               (second * Math.PI / (360 * 60));
        drawHand(ctx, hour, radius * 0.5, radius * 0.07);

        // minute
        minute=(minute * Math.PI / 30) + (second*Math.PI / (30 * 60)
        drawHand(ctx, minute, radius * 0.8, radius * 0.07);

        // second
        second=(second*Math.PI / 30);
        drawHand(ctx, second, radius * 0.9, radius * 0.02);
      }

      function drawHand(ctx, pos, length, width)
      {
        ctx.beginPath();
        ctx.lineWidth = width;
        ctx.lineCap   = "round";
        ctx.moveTo(0,0);
        ctx.rotate(pos);
        ctx.lineTo(0, -length);
        ctx.stroke();
        ctx.rotate(-pos);
      }
    </script>
  </body>
</html>
```