Joonistamine

Pilte näitasid veebilehed üsna veebi algusaegadest peale (1990ndatel) Kasutaja soovidest sõltuvate jooniste või animatsioonide loomiseks on aga tulnud mitmesuguseid lisavidinaid kasutada. Varakult tekkisid Java rakendid, siis Flashi abivahendid ning seejärel Silverlighti lisandprogramm.Samuti sai Javaskripti raamistikega nurgataguseid kasutades luua ekraanile mulje joonise tekkimisest - ehkki joone tõmbamiseks näiteks mõnikord tekitati suurel hulgal ruudukujulisi väikesi tekstilõike, värviti lõigukeste taustad ära ning paigutati nõnda, et see kasutajale joonena paistab. Arvestades aga, et veebi loetakse väga mitmesuguste tehniliste võimaluste ja seadistustega masinatega, siis nõnda paljude ja küllalt tugevat riistvara nõudvate tarkvaralahenduste korraga kättesaadavana hoidmine ei kipu õnnestuma. Ikka tuleb teateid, kuidas üks või teine brauser või operatsioonisüsteem jälle mõne lahendusega hakkama ei saa.

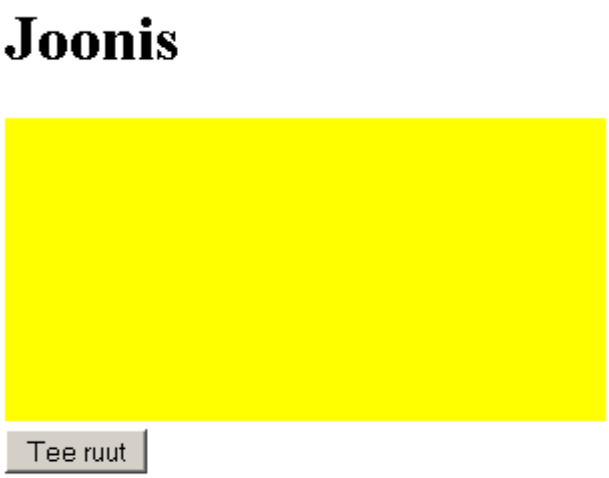
HTML-i viiendat versiooni kavandades otsustati siia vahenditesse sisse panna võimalikult palju hädatarvilikku, mille abil saaks enamiku veebilehtede loomisel tekkivaid soove ära rahuldada. Nõnda on HTML5 igakülgne näitamine seadmele küll veidi keerulisem kui mõne varasema versiooni korral, samas tootjatel on siisk tunduvalt lihtsam teha seadmeid ja tarkvara HTML5 näitamiseks, kui et arvestada laia komplekti mitmesuguste lisandprogrammidega.

Ruut ekraanil

Joonistamiseks ja liigutamiseks lisati HTML5 vahendite hulka element nimega canvas (lõuend). Selle peale on võimalik Javaskripti abil joonistada. Samuti korduva joonistamise/kustutamise abil liikumine tekitada. Lehele saab ta tekitada sarnase käsuga.

```
    <canvas id="tahvel" width="300" height="200"  style="background-
color:yellow"></canvas>
```

Suurus ja taust talle määratud selleks, et lehel selgemalt näha oleks, kus lõuend paistab. Ning id järgi saab hiljem programmikoodiga lõuendi poole pöörduda, et sinna miskit joonistama hakata.



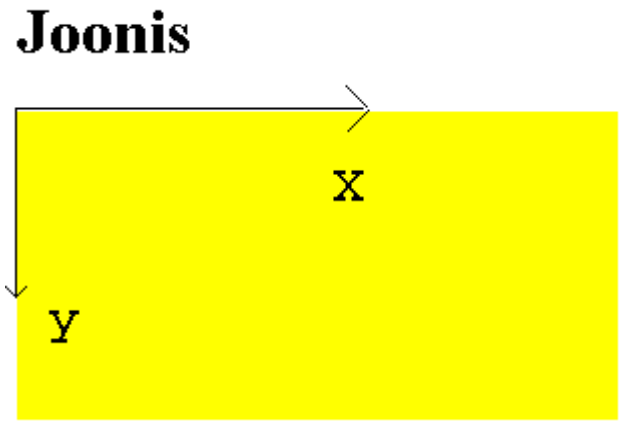
Joonistamise tarbeks siin väike alamprogramm. Tahvlile joonistamise tarbeks tuleb talle kõigepealt ligipääs küsida.

```
var g=document.getElementById("tahvel").getContext("2d");
```

Edasi võib siis juba joonistuskäskude abil pilti ekraanile kuvama asuda.
Lihtsaimaks näiteks siin ristküliku loomine

```
g.fillRect(20, 20, 50, 50); //x, y, laius, kõrgus
```

Arvutil programmi abil joonistades tuleb kõik asjad koordinaatide järgi kätte anda, nii ka ristküliku puhul. Programmeerimisgraafika omapäraks on, et koordinaatide nullpunkt asub vasakul ülanurgas. Sealt liigub x-telg paremale ning y-telg alla.



Nõnda kokkupandud koodinäite puhul peaks nupule vajutades ekraanile ruut tekkima.

```
<!DOCTYPE html>
<html>
  <head>
    <title>Joonis</title>
    <script type="text/javascript">
      function joonista(){
         var g=document.getElementById("tahvel").getContext("2d");
         g.fillRect(20, 20, 50, 50); //x, y, laius, kõrgus
      }
    </script>
  </head>
  <body>
    <h1>Joonis</h1>

    <canvas id="tahvel" width="300" height="200"
            style="background-color:yellow"></canvas><br />
    <input type="button" value="Tee ruut" onclick="joonista()" />
  </body>
</html>
```

Joonis

Ülesandeid

* Tutvu ristküliku joonistamise näitega
* Muuda ristküliku suurust ja asukohta
* Tee ekraanile kaks ristkülikut
* Lao ekraanile ristkülikutest koosnev alt laiem torn.

Mõõtude järgi ristkülik

Niisama pildi kuvamiseks pole enamasti vaja programmi kirjutada. Saab valmis pilti näidata või siis seda sobival hetkel vahetada. Kui tahta aga, et kasutaja saaks joonise mõõtmeid muuta, siis võib mõnest koodireast sealjuures kasu tulla. Lihtsamatel juhtudel on küll võimalik sobivad pildid enne valmis teha ning siis vastavalt sisestatud andmetele kasutajale näidata. Kui aga võimalusi palju, siis tuleb ikka vaadata, et mida ja kuidas joonistada.

Andmete sisestamise puhul sobivad samasugused tekstiväljad kui aruvutamise osa juures.

```
    <input type="text" id="laiuskast"  value="40" />
```

```
Nõnda luuakse tekstiväli. Pöördumise jaoks on tal id nimega laiuskast ning
algväärtuseks pannakse sisse talle 40.
```

Kuna arvuti jaoks on tähed ja arvud erinevad, siis tuleb andmetüüp sisselugemisel sobivaks sättida.

```
        var laius=parseInt(document.getElementById("laiuskast").value);
```

tähendab lahtiseletatult, et küsitakse documendi seest laiuskasti-idga element ja sealt tema tekstiline väärtus. Käsklus parseInt muundab teksti arvuks - see tähendab, et 40 pole enam mitte tähed neli ja null, vaid arv nelikümmend, millega soovi korral liita/lahutada saab ning mida võib võtta ja joonise sisendväärtusena. Võrdusmärgi ees olev var laius ütleb, et väljaarvutatud väärtus jäetakse meelde märksõna laius alla ning sealtkaudu on seda võimalik hiljem pruukida. Joonistamise juures lähebki neid kohe tarvis.

```
g.fillRect(30, 30, laius, korgus); //x, y, laius, kõrgus
```

joonistab etteantud laiuse ja kõrgusega ristküliku. Esimesed 30 ja 30 tähistavad vastavalt kujundi kaugust tahvli vasakust ning ülemisest servast.

Allpool on näidatud ka, kuidas teksti ekraanile kuvada.

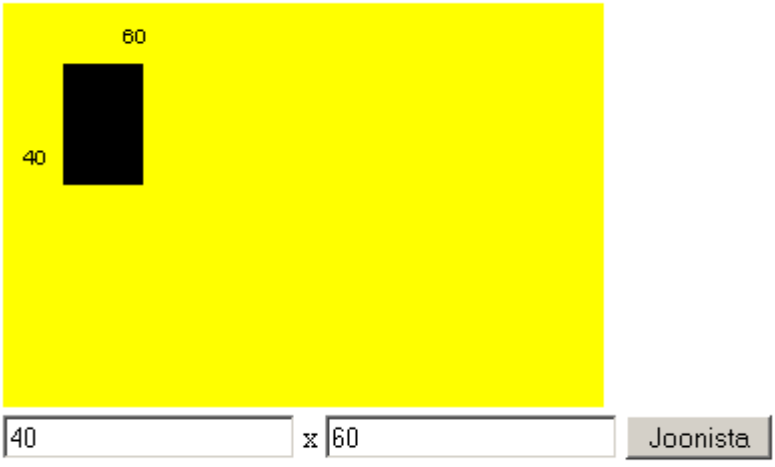
```
g.fillText(korgus, 50, 20);
```

kuvab välja märksõnas ehk muutujas "korgus" oleva väärtuse, paigutades selle alguse tahvli vasakust servast 50 ning ülemisest 20 punkti kaugusele.

Käivitav kood tervikuna

```
<!DOCTYPE html>
<html>
  <head>
    <title>Joonis</title>
    <script type="text/javascript">
      function joonista(){
         var g=document.getElementById("tahvel").getContext("2d");
         var laius=parseInt(document.getElementById("laiuskast").value);
         var korgus=parseInt(document.getElementById("korguskast").value);
         g.fillRect(30, 30, laius, korgus); //x, y, laius, kõrgus
         g.fillText(korgus, 60, 20);
         g.fillText(laius, 10, 80);
      }
    </script>
  </head>
  <body>
    <h1>Joonis</h1>
    <canvas id="tahvel" width="300" height="200"
              style="background-color:yellow"></canvas><br />
    <input type="text" id="laiuskast"  value="40" /> x
    <input type="text" id="korguskast" value="60" />
    <input type="button" value="Joonista" onclick="joonista()" />
  </body>
</html>
```

Ning selle järgi valminud joonis.



Ülesandeid

* Tutvu ristküliku mõõtmete määramise näitega.
* Võimalda lisaks mõõtmetele määrata ka ristküliku asukoht ekraanil.
  (Eraldi tekstikastid vasaku ülanurga x- ja y-koordinaadi määramiseks,
  kogu joonistamine jääb ühe funktsiooni sisse.)

* Joonista lehele kaks ristkülikut, kusjuures kummalgi saab määrata vaid laiust.
  Ristkülikud asuvad üksteise all, ning nende vasakud servad kohakuti.

* Joonista lehele kaks ristkülikut, kusjuures kummalgi saab määrata vaid kõrgust. Ristkülikute ülaservad on kohakuti.

* Säti arvutused nõnda, et ristkülikute alaservad on samal kõrgusel,
  pikem ristkülik ulatub ülevalt kõrgemale.

Joon

Joon paistab olema joonistamisele nime andnud. Joone ekraanile kuvamiseks paistab vaja olema veidi rohkem toimetusi kui ristküliku korral, kuid ei midagi võimatut. Joon on tema jaoks joonistustee (Path) osa, kus tuleb üksikuid punkte märkida. Programmikoodiga tuleb käituda, nagu juhitaks pliiatsiotsa. Käsk moveTo ütleb, kuhu koordinaatidele (x ja y) pliiats tõsta, edasine lineTo teatab kuhuni joon tõmmata. Lõpus olev stroke() palub tulemuse ekraanile kuvada.

```
        g.beginPath();
        g.moveTo(30, 40);
        g.lineTo(80, 90);
        g.stroke();
```

Et siin näites tehakse alati sama joon ning kasutaja joone andmeid muuta ei saa, pole talle ka joonistamiseks nuppu vaja. Joonistuskäsk pannakse käima kohe, kui leht on kohale jõudnud ja veebilehitsejas ette kuvatud. Ehk siis lehe sisuosa body sündmuse onload peale.

```
  <body onload="joonista()">
```

Muus osas võib jälle rahulikult jälgida, kuidas kood kokku pandud

```
<!DOCTYPE html>
<html>
  <head>
    <title>Joonis</title>
    <script type="text/javascript">
      function joonista(){
         var g=document.getElementById("tahvel").getContext("2d");
         g.beginPath();
         g.moveTo(30, 40);
         g.lineTo(80, 90);
         g.stroke();
      }
    </script>
  </head>
  <body onload="joonista()">
```

```
    <h1>Joonis</h1>
    <canvas id="tahvel" width="300" height="200"
            style="background-color:yellow"></canvas><br />
  </body>
</html>
```

Edasi koodi põhjal tekkinud joont imetleda ning selle pealt juba edasisi arendusmõtteid mõlgutada.

Joonis

Ülesandeid

* Tutvu joone tõmbamise näitega
* Tõmba joon ülalt alla
* Tõmba joon vasakult paremale
* Tõmba kolm kõrvuti asetsevat joon
* Joonista kuubikust maja, millel oleks joone abil tõmmatud viilkatus peal.

Joone värv ja paksus

Lihtsalt tõmmatud joon on ühe punkti laiune ning musta värvi. Arvutisse aga tahame vahel ka mitmekesisemaid pilte. Iga Path-i abil tõmmatud joon on oma pikkuses samasuguste omadustega. Eri joonistusteedeks jagatud jooned aga saavad olla igaüks omamoodi. Joone paksuse määrab tunnus lineWidth, joone värvi strokeStyle. Nii on tehtud all olevas näites.

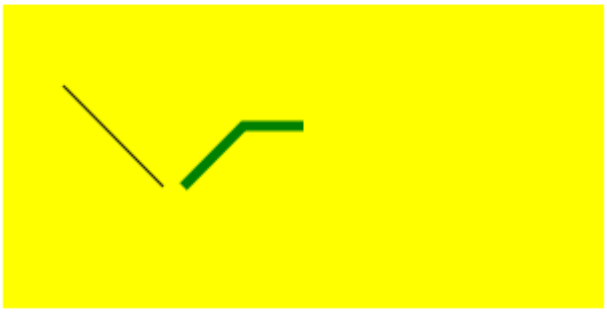
```
<!DOCTYPE html>
<html>
  <head>
    <title>Joonis</title>
    <script type="text/javascript">
      function joonista(){
         var g=document.getElementById("tahvel").getContext("2d");
         g.beginPath();
         g.moveTo(30, 40);
         g.lineTo(80, 90);
         g.stroke();

         g.beginPath();
         g.lineWidth=5;
         g.strokeStyle="green";
```

```
        g.moveTo(90, 90);
        g.lineTo(120, 60);
        g.lineTo(150, 60)
        g.stroke();
      }
    </script>
  </head>
  <body onload="joonista()">
    <h1>Joonis</h1>
    <canvas id="tahvel" width="300" height="200"
              style="background-color:yellow"></canvas><br />
  </body>
</html>
```

Tulemus näha ka joonisel.

Joonis



Ringid

Ümaramate kujundite juures aitavad meid ringid - nii seest täidetud kujul kui ringjoonena. Üllatusena küll Javaskriptil lihtsalt ringi loomise käsklust ei ole. Küll aga on universaalne käsklus kaare loomiseks. Kui aga kaare pikkuseks on täisring ehk 360 kraadi ehk 2Pi radiaani, siis näemegi tulemust ringina. Programmeerimise juures tuleb mitmel puhul vajalikke tulemusi kombineerida kättesaadavate võimaluste abil, nii ka siin.

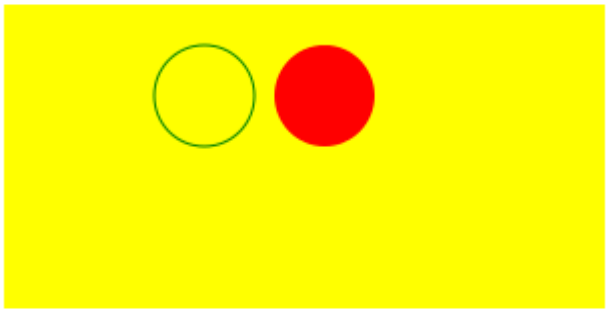
```
<!DOCTYPE html>
<html>
  <head>
    <title>Joonis</title>
    <script type="text/javascript">
      function joonista(){
         var g=document.getElementById("tahvel").getContext("2d");
         g.strokeStyle="green";
         g.beginPath();
         g.arc(100, 45, 25, 0, 2*Math.PI, true);
         g.stroke();

         g.fillStyle="red";
         g.beginPath();
         g.arc(160, 45, 25, 0, 2*Math.PI, true);
         g.fill();

      }
```

```
      window.onload=joonista;
    </script>
  </head>
  <body>
    <h1>Joonis</h1>
    <canvas id="tahvel" width="300" height="200"
              style="background-color:yellow"></canvas><br />
  </body>
</html>
```

Joonis



Nõnda on põhiliste kujundite joonistamise oskus olemas. Edasine sõltub suurelt jaolt juba oma fantaasiast ja ekraanipunktide lugemise jaksust. Nõnda saab mitmesuguseid jooniseid luua, kus kasutajalt tulevad andmed sisse ning nende põhjal tehakse tema soovidele vastav joonis.

Ülesandeid

* Pane ringi joonistamise näide käima
* Joonista valgusfoor
* Kasutaja saab nupule vajutamisega määrata, milline tuli fooris põleb
* Joonista automudel, mille pikkust ja kõrgust saab kasutaja määrata