Meeldetuletus

Alustuseks mõned lihtsad näited Javaskripti abil töötavatest veebirakendustest. Et kelle jaoks tegemist uue keelega või pole lihtsalt kaua kokku puutunud, siis saaks mõningase tunde kätte.

Kalkulaator

Arvutamise jaoks peab ikka olema koht andmete sisse panekuks ning koht tulemuse nägemiseks. Tekstiväli ning div-kiht sobivad selliseks komplektiks küll. Nupuvajutuse peale saab käima panna funktsiooni. Funktsioonid hea defineerida lehe päiseosas, siis nad hilisema kasutuse tarbeks olemas. Elementide poole pöördumiseks sobib käsklus document.getElementById - mida Javaskripti-põhiste lahenduste puhul tuleb päris sageli välja kutsuda. Mõned raamistikud teevad selle käsu küll lühemaks, praegu kasutame pikka kuju.

Näite juures arvutatakse poes oleva letihinna põhjal selles sisalduv käibemaksu osa.

```
<!DOCTYPE html>
<html>
  <head>
     <title>Käibemaksu arvutus</title>
       <meta charset="UTF-8" />
       <script type="text/javascript">
          function arvuta(){
               var letihind=
                   parseFloat(document.getElementById("letihind").value);
               var km=letihind*20/120;
//             km=Math.round(km*100)/100; //ümardus 2 kohta pärast koma.
               km=km.toFixed(2);
               document.getElementById("vastus").innerHTML=km;
            }
       </script>
  </head>
  <body>
     <h1>Katsetuste leht</h1>
       Palun sisesta letihind:
        <input type="text" id="letihind" />
        <input type="button" value="OK" onclick="arvuta();" />
        <div id="vastus">
          Vastuse koht.
        </div>
  </body>
</html>
```

Ülesandeid

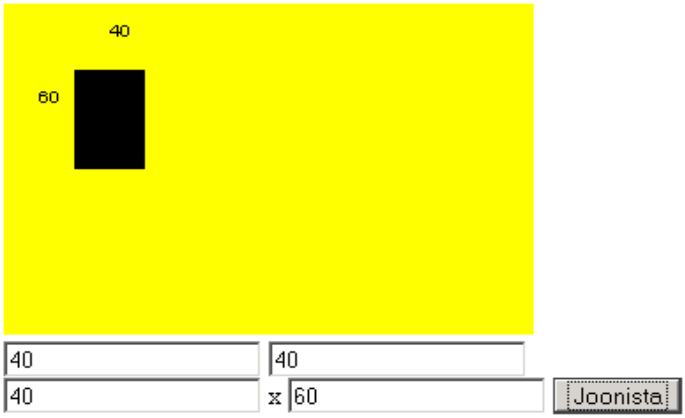
- Pane näide käima
- Koosta kalkulaator valuutavahetusel kättesaadava summa arvutamiseks sisseantud summa, kursi ja komisjonitasu järgi.

Ristkülik ekraanil

HTML5ga koos tuli veebilehele joonistamiseks lõuend ehk canvas. Loojate plaanide järgi saab sellest lähiaastate veebigraafika valitseja. Eks aeg näitab, mis tulevik toob. Aga otseste joonistuskäskudega on küll mugavam jooni ja ringe tekitada, kui selleks kavalalt kombineeritud värvitud taustaga kihte veebilehele võluda. Järgneva joonistuskäsu juures küsitakse lõuend-tahvlilt graafiline kontekst (nagu sulepea) ning tekstikastidest ette antud andmete järgi kuvatakse ekraanile ristkülik.

```
<!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);
         var vasakult=parseInt(document.getElementById("vasakkast").value);
         var ylalt=parseInt(document.getElementById("ylakast").value);
         g.fillStyle="yellow";
         g.fillRect(0, 0, 300, 200);
         g.fillStyle="black";
         g.fillRect(vasakult, ylalt, laius, korgus); //x, y, laius, kõrgus
         g.fillText(korgus, vasakult-20, ylalt+20);
         g.fillText(laius, vasakult+20, ylalt-20);
      }
    </script>
  </head>
  <body>
    <h1>Joonis</h1>
    <canvas id="tahvel" width="300" height="200"
              style="background-color:yellow"></canvas><br />
    <input type="text" id="vasakkast"  value="40" />
      <input type="text" id="ylakast"  value="40" /><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>
```

Joonis



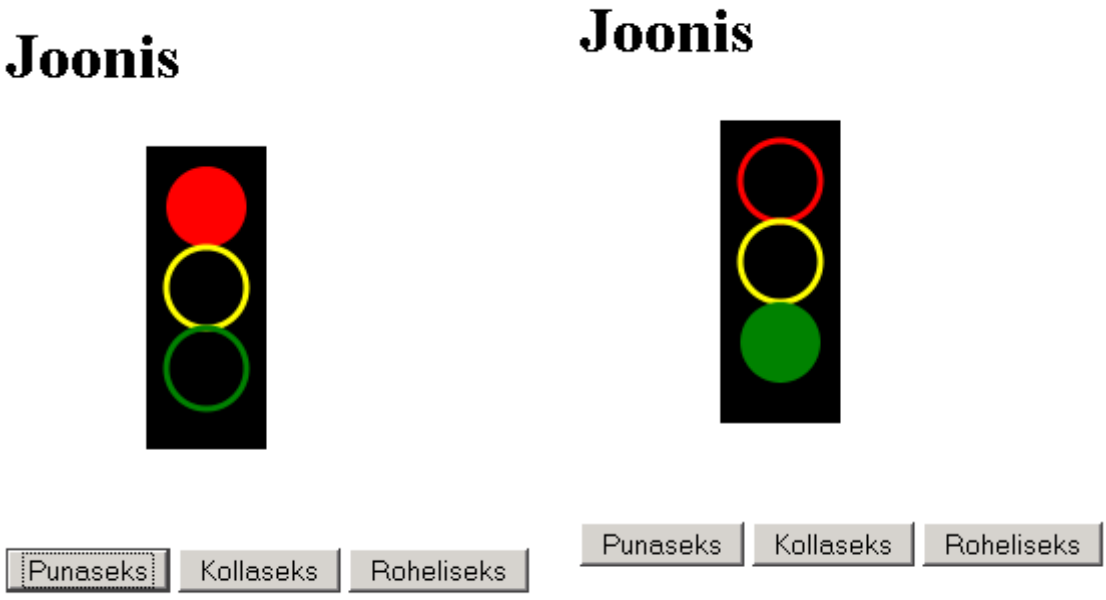
Foor, seisundid

Märgatav osa siinsetest materjalidest tutvustab objektidega majandamist veebilehel. Objekti omapäraks on tema juurde kuuluvad andmed ning samuti käsklused, mis oma tööks andmeid kasutada saavad. Lihtsamatel juhtudel võib sarnase tulemuse saada aga ka tavalisel veebilehel toimetades. Lehtki on suhteliselt eraldiseisev üksus, kus globaalmuutujatena võimalik hoida lehe piires kättesaadavaid andmeid ning neid siis oma käskude juures kasutada. Näitena toodud valgusfoor, kus seisundimuutujas parajasti kirjas põlev värv. Muutuja järgi otstustades joonistatakse sobivasse kohta seda värvi ring. Allpool nuppudega valides saab määrata seisundiks sobiva värvi ning selle peale pannakse pildi uuendamiseks tööle joonistuskäsk.

```
<!DOCTYPE html>
<html>
  <head>
    <title>Joonis</title>
    <script type="text/javascript">
          var seisund="punane";
      function joonista(){
         var g=document.getElementById("tahvel").getContext("2d");
         g.fillStyle="white";
         g.fillRect(0, 0, 300, 200);
         g.fillStyle="black";
         g.fillRect(70, 5, 60, 150);
         g.lineWidth=3;
         g.strokeStyle="red"; g.fillStyle="red";
         g.beginPath();
         g.arc(100, 35, 20, 0, 2*Math.PI, true);
         if(seisund=="punane"){g.fill()} else {g.stroke();}
         g.strokeStyle="yellow"; g.fillStyle="yellow";
         g.beginPath();
         g.arc(100, 75, 20, 0, 2*Math.PI, true);
         if(seisund=="kollane"){g.fill()} else {g.stroke();}
         g.strokeStyle="green"; g.fillStyle="green";
         g.beginPath();
         g.arc(100, 115, 20, 0, 2*Math.PI, true);
         if(seisund=="roheline"){g.fill()} else {g.stroke();}
      }
      function punaseks(){seisund="punane"; joonista();}
```

```
      function kollaseks(){seisund="kollane"; joonista();}
      function roheliseks(){seisund="roheline"; joonista();}
    </script>
  </head>
  <body onload="joonista();">
    <h1>Joonis</h1>
    <canvas id="tahvel" width="300" height="200"
              style="background-color:yellow"></canvas><br />

    <input type="button" value="Punaseks" onclick="punaseks()" />
    <input type="button" value="Kollaseks" onclick="kollaseks()" />
    <input type="button" value="Roheliseks" onclick="roheliseks()" />
  </body>
</html>
```



Ülesandeid

- Pane näide käima
- Koosta taksohinna kalkulaator. Näha on sisseistumistasu ja kilomeetrihind. Väljastatakse sõiduhind vastavalt sisestatud kilomeetrite arvule.
- Lisa kalkulaatorile seisundid öö ja päev. Vastavalt muutub kalkulaatori juures olev pilt. Öösel on hinnad kõrgemad.

Objektid

Tollikalkulaator objektina

Kõigepealt meeldetuletuseks tavaline lihtne kalkulaator.

```
<!doctype html>
<html>
  <head>
    <title>Kalkulaator</title>
        <script>
          function arvuta(){
             var tollidearv=parseFloat(document.getElementById("kast1").value);
                 document.getElementById("vastus").innerHTML=tollidearv*2.54;
          }
        </script>
  </head>
  <body>
     <h1>Arvutamine</h1>
         Tollid:
         <input type="text"   id="kast1" />
         <input type="button" value="Sentimeetriteks" onClick="arvuta();" />
         <div id="vastus"></div>

  </body>
</html>
```

Järgmisena tehakse sisestuskastidest ning arvuta-käsklusest komplekt nimega Arvutaja. Nii on arvuta-käsklusel võimalik this-muutuja kaudu tekstiväljade poole pöörduda ning ei pea keerulisemal lehel enam pikemalt muretsema, et kust andmed kätte saab. Siin näites küll on veel kastide nimed otse koodi sisse kirjutatud, kuid ka selle saab tulevikus paindlikumaks muuta. Lehe sisumärgendi body atribuut onload teatab, et lehe avanemisel tuleb käivitada funktsioon lehealgus(). Selle tulemusena luuakse muutujasse a1 Arvutaja-tüüpi objekt ehk eksemplar. All sentimeetriarvutuse nupu juures sobib tulemuse saamiseks käima panna juba käsklus a1.arvuta(), mis juba teab, millised andmed kust võtta ja kuhu vastus panna.

```
<!doctype html>
<html>
  <head>
    <title>Kalkulaator</title>
        <script>
          var a1;
          function Arvutaja(){
             this.kast=document.getElementById("kast1");
             this.vastusekiht=document.getElementById("vastus");
             this.arvuta=function(){
                    this.vastusekiht.innerHTML=parseFloat(this.kast.value)*2.54;
             }
```

```
          }

          var a1;
          function lehealgus(){
            a1=new Arvutaja();
          }
        </script>
  </head>
  <body onload="lehealgus();">
     <h1>Arvutamine</h1>
         Tollid:
         <input type="text"   id="kast1" />
         <input type="button" value="Sentimeetriteks" onClick="a1.arvuta();" />
         <div id="vastus"></div>

  </body>
</html>
```

Kalkulaator töötab nii nagu ennegi, lihtsalt koodi sees on nüüd koht Arvutaja-objekt oskuste kirjeldamiseks.

Arvutamine

Tollid: 5 Sentimeetriteks
12.7

Järgmise sammuga muudame arvutaja iseseisvamaks - lisame ka oskuse ennast ekraanile kuvada. Ehk siis väljastada näitamiseks vajaliku HTMLi. Kõigepealt luuakse Arvutaja sisse funktsioonid nimega algus ning arvuta. Loomise lõpul käivitatakse funktsioon algus(). Võtmesõna this alguse juures näitab, et tegemist pole lihtsalt alguse-nimelise funktsiooniga (mis võiks ka kusagil mujal loodud olla), vaid Arvutaja objekti külge kuuluva funktsiooniga algus(). Hilisema pöördumise lihtsustamiseks luuakse Arvutaja külge muutuja this.kiht ning järgmise käsuga määratakse kihi sisse HTML-kood. Samuti lisanduvad Arvutaja külge muutujad kast ja vastusekiht. Arvutamisfunktsiooniks arvuta, mille juures siis tollidest sentimeetrid tehakse.

Lehe avamisele body-elemendi atribuut onload ütleb, et tuleb käivitada funktsioon lehealgus(). Seal antakse väärtus eelnevalt loodud muutujale nimega a1. Väärtuseks saab Arvutaja tüüpi objekt, kellele kihi nimeks antakse ette kiht1. Esialgu kannatab sellisena panna vaid ühe arvutaja lehele, sest muutuja nimi a1 on Arvutaja this.algus-funktsioonis a1.arvuta() juures sisse kirjutatud.

```
<!doctype html>
<html>
  <head>
    <title>Kalkulaator</title>
        <script>
          function Arvutaja(kihinimi){
                 this.algus=function(){
                   this.kiht=document.getElementById(kihinimi);
                   this.kiht.innerHTML=
                    "Tollid: <input type='text'   id='kast1' /> "+
                    "<input type='button' value='Sentimeetriteks' "+
                                         "onClick='a1.arvuta();' /> "+
                    "<div id='vastus'></div>";
                    this.kast=document.getElementById("kast1");
                    this.vastusekiht=document.getElementById("vastus");
```

```
                }
                this.arvuta=function(){
                   this.vastusekiht.innerHTML=parseFloat(this.kast.value)*2.54;
                }
                this.algus();
            }

            var a1;
            function lehealgus(){
              a1=new Arvutaja("kiht1");
            }
          </script>
  </head>
  <body onload="lehealgus();">
     <h1>Arvutamine</h1>
     <div id="kiht1"></div>

  </body>
</html>
```

Arvutamine

Tollid: 5 Sentimeetriteks
12.7

Ülesandeid

- Pane näited käima
- Loo sarnaselt näitele kalkulaator ringi pindala leidmiseks raadiuse järgi.

Mitu kalkulaatorit lehel

Üksiku kalkulaatori saab ka niisama veebilehele kirjutada. Kui neid aga vaja panna hulgem ning mitmesuguste parameetritega, siis on objektide loomisest rohkem kasu. Järgmises näites antakse Arvutajale ette kihi id, kuhu end paigutada ja töökorda sättida. Käsklus

```
window[kihinimi+"_kalkulaator"]=this;
```

loob akna ehk lehe peaobjekti külge muutuja, mille nime sees on kihi nimi ja sõna "kalkulaator", ehk siis esimese Arvutaja loomisel

```
new Arvutaja("kiht1");
```

saab objekti nimeks nõnda kiht1_kalkulaator. Nupu loomisel öeldakse nupule, et tuleb sellenimelise objekti juures välja kutsuda käsklus arvuta()

```
"<input type='button' value='Sentimeetriteks'
     onClick='"+kihinimi+"_kalkulaator.arvuta();' /> "
```

Selle järgi teatakse just õigest kohast sisestatud arv võtta, sest

```
this.kast=document.getElementById(kihinimi+"_kast1");
```

paneb arvutaja külge muutuja nimega kast. Pärast arvutuse juures võetakse sealt

```
this.vastusekiht.innerHTML=parseFloat(this.kast.value)*2.54;
```

```
<!doctype html>
<html>
  <head>
    <title>Kalkulaator</title>
        <script>
          function Arvutaja(kihinimi){
             this.algus=function(){
                this.kiht=document.getElementById(kihinimi);
                window[kihinimi+"_kalkulaator"]=this;
                this.kiht.innerHTML=
                    "Tollid: <input type='text'   id='"+kihinimi+"_kast1' /> "+
                    "<input type='button' value='Sentimeetriteks' "+
                    "onClick='"+kihinimi+"_kalkulaator.arvuta();' /> "+
                    "<div id='"+kihinimi+"_vastus'></div>";
                this.kast=document.getElementById(kihinimi+"_kast1");
                this.vastusekiht=document.getElementById(kihinimi+"_vastus");
             }
             this.arvuta=function(){
                    this.vastusekiht.innerHTML=parseFloat(this.kast.value)*2.54;
             }
             this.algus();
          }


          function lehealgus(){
            new Arvutaja("kiht1");
            new Arvutaja("kiht2");
          }
        </script>
  </head>
  <body onload="lehealgus();">
     <h1>Arvutamine</h1>
         <div id="kiht1"></div>
         <div id="kiht2"></div>
  </body>
</html>
```

Arvutamine

Tollid: 3 Sentimeetriteks
7.62
Tollid: 6 Sentimeetriteks
15.24

Ülesandeid

- Pane näide käima
- Loo sarnaselt lehele ka mitu kalkulaatorit ringi pindala leidmiseks raadiuse järgi

Kalkulaatorite parameetrid

Eelnevates näidetes vaadatud kalkulaatorid suutsid hakkama saada vaid ühe operatsiooniga. Kujundus ning arvutuskäik oli juba koodi sisse kirjutatud ning valida sai vaid loodud kalkulaatori asukoha ja nende loomise koguse järgi. Arvutaja saab kirjutada ka paindlikumalt - lisaks kihi nimele antakse järgmises näites ette ka sissestuskasti juurde tulev selgitav tekst, vajutusnupule kuvatav tekst ning arvutuskoefitsient kahe suuruse vahel.

```
<!doctype html>
<html>
  <head>
    <title>Kalkulaator</title>
        <script>
          function Arvutaja(kihinimi, kastitekst, nuputekst, koefitsient){
             this.algus=function(){
               this.kiht=document.getElementById(kihinimi);
               window[kihinimi+"_kalkulaator"]=this;
               this.kiht.innerHTML=
                kastitekst+": <input type='text'   id='"+kihinimi+"_kast1' /> "+
                "<input type='button' value='"+nuputekst+"' onClick='"+
                   kihinimi+"_kalkulaator.arvuta();' /> "+
                "<span id='"+kihinimi+"_vastus'></span>";
               this.kast=document.getElementById(kihinimi+"_kast1");
               this.vastusekiht=document.getElementById(kihinimi+"_vastus");
               this.koefitsient=koefitsient;
             }
             this.arvuta=function(){
              this.vastusekiht.innerHTML=
                   parseFloat(this.kast.value)*this.koefitsient;
             }
             this.algus();
          }


          function lehealgus(){
            new Arvutaja("kiht1", "Eurod", "Dollariteks", 1.3);
            new Arvutaja("kiht2", "Tollid", "Sentimeetriteks", 2.54);

          }
        </script>
  </head>
  <body onload="lehealgus();">
     <h1>Arvutamine</h1>
         <div id="kiht1"></div>
         <div id="kiht2"></div>


  </body>
</html>
```

Tulemusena võib edasi luua juba igasugu kalkulaatoreid, mille puhul arvutus piirdub ühe suhtega ehk korrutus/jagamistehtega.

Arvutamine

Eurod: 11 Dollariteks 14.3
Tollid: 5 Sentimeetriteks 12.7

Ülesandeid

- Pane näide käima
- Loo sama näite põhjal kalkulaator käibemaksu arvutamiseks letihinna järgi
- Koosta sarnaselt kihile pandav kalkulaator, kus näidatakse soovitud arv tekstikaste ning nupuvajutuse peale väljastatakse neisse sisestatud arvude aritmeetiline keskmine.

Andmete talletamine objektis

Eelnevate arvutuste juures anti sisestuse põhjal koheselt vastus ning midagi eraldi säilitada pole vaja. Objekti üheks kasulikuks omaduseks on aga tema olek, ehk siis võime väärtusi oma eluea jooksul säilitada. Siin lihtsama näitena arvuline meelespeetav kogus ning nupud selle koguse suurendamiseks ja vähendamiseks. Tulemuse näitamisks funktsioon kuva() ning nuppude külge kinnitatud funktsioonid suuremaks() ja väiksemaks() soovitud muutuse tarbeks.

```
<!doctype html>
<html>
  <head>
    <title>Kalkulaator</title>
        <script>
          function Laohaldus(kihinimi, kogus){
             this.algus=function(){
                this.kiht=document.getElementById(kihinimi);
                this.kogus=kogus;
                window[kihinimi+"_ladu"]=this;
                this.kiht.innerHTML=
                  "<input type='button' value='&lt;' "+
                     "onClick='"+kihinimi+"_ladu.v2iksemaks();' /> "+
                  "<input type='text' id='"+kihinimi+"_vastus' "+
                     "style='width: 50px' disabled />"+
                  "<input type='button' value='&gt;' "+
                  " onClick='"+kihinimi+"_ladu.suuremaks();' /> ";
                this.vastusekiht=document.getElementById(kihinimi+"_vastus");
                this.kuva();
             }
             this.kuva=function(){
                this.vastusekiht.value=this.kogus;
             }
             this.v2iksemaks=function(){
                this.kogus--;
                this.kuva();
             }
             this.suuremaks=function(){
                this.kogus++;
```

```
                this.kuva();
              }
              this.algus();
            }


            function lehealgus(){
              new Laohaldus("kiht1", 100);
              new Laohaldus("kiht2", 50);
            }
          </script>
  </head>
  <body onload="lehealgus();">
     <h1>Arvutamine</h1>
         <div id="kiht1"></div>
         <div id="kiht2"></div>
  </body>
</html>
```

Arvutamine

Ülesandeid

- Pane näide käima
- Lisa objekti kujundusse teine tekstiväli, kus saab määrata, millise koguse võrra olemasolevat väärtust kasvatatakse või kahandatakse
- Tekstivälja asemel saab muudetava suuruse valida rippmenüüst
- Lisa nupp laoseisu nullimiseks

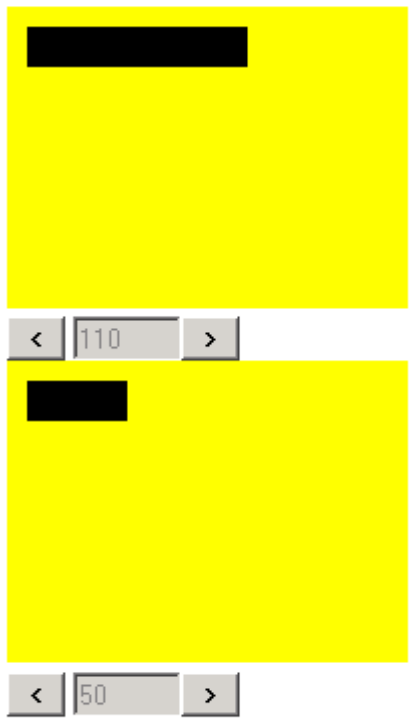
Objekti graafiline väljund

HTML5 juurde kuuluvat Canvast saab sarnaselt sisestuse ja väljundi juures kasutada nagu teisigi veebilehe elemente. Lihtsalt siitkaudu on võimalik väljundit märgatavalt värvilisemaks muuta. Kuvamise funktsioonile lisatakse täiendus: lisaks arvu näitamisele tekstiväljas tehakse lõuendile ka vastava pikkusega kast.

```
<!doctype html>
<html>
  <head>
    <title>Kalkulaator</title>
        <script>
          function Laohaldus(kihinimi, kogus){
             this.algus=function(){
               this.kiht=document.getElementById(kihinimi);
               this.kogus=kogus;
               window[kihinimi+"_ladu"]=this;
               this.kiht.innerHTML=
                 "<canvas id='"+kihinimi+"_joonis' "+
                    "width='200' height='150' "+
```

```
                   "style='background-color: yellow' ></canvas><br />"+
                "<input type='button' value='&lt;' "
                  +" onClick='"+kihinimi+"_ladu.v2iksemaks();' /> "+
                "<input type='text' id='"+kihinimi+"_vastus'"+
                  "style='width: 50px' disabled />"+
                "<input type='button' value='&gt;' "
                    +" onClick='"+kihinimi+"_ladu.suuremaks();' /> ";
                this.vastusekiht=document.getElementById(kihinimi+"_vastus");
                this.joonis=document.getElementById(kihinimi+"_joonis");
                this.kuva();
             }
             this.kuva=function(){
                this.vastusekiht.value=this.kogus;
                var g=this.joonis.getContext("2d");
                g.clearRect(0, 0, 200, 150);
                g.fillRect(10, 10, this.kogus, 20);
             }
             this.v2iksemaks=function(){
                this.kogus--;
                this.kuva();
             }
             this.suuremaks=function(){
                this.kogus++;
                this.kuva();
             }
             this.algus();
          }


          function lehealgus(){
            new Laohaldus("kiht1", 100);
            new Laohaldus("kiht2", 50);
          }
        </script>
  </head>
  <body onload="lehealgus();">
     <h1>Arvutamine</h1>
         <div id="kiht1"></div>
         <div id="kiht2"></div>
  </body>
</html>
```



Ülesandeid

- Pane näide käima
- Lisa tekstiväli näitamaks, mitme ühiku jagu objektis olevat kogust suurendada või vähendada.

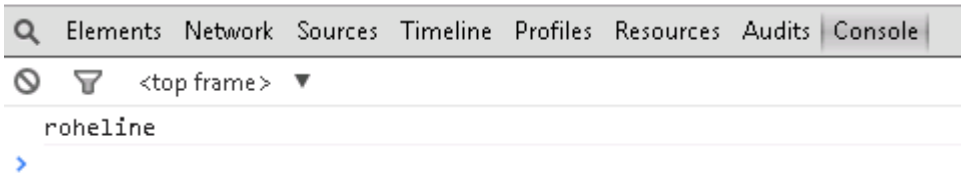
Foor

Objektidega harjumine paistab mõnelegi keeruline olema. Järgnevalt harjumiseks näide, kuidas valgusfoori tööd veebilehel objektina jäljendada. Esmalt lihtsaim variant, kus fooril on üks muutuja ehk väli, mis peab meeles põleva tule värvi. Käsklustega saab seda värvi küsida ning muuta.

```
<!doctype html>
<html>
  <head>
    <title>Foor</title>
    <script>
       function foor(){
          this.tuli="roheline";
          this.kysiTuli=function(){return this.tuli;}
          this.muudaTuli=function(uusTuli){this.tuli=uusTuli;}
       }
       function leheAlgus(){
          var f1=new foor();
          console.log(f1.kysiTuli());
          f1.muudaTuli("kollane");
          document.getElementById("vastus").innerHTML=
            f1.kysiTuli();
       }
    </script>
```

```
  </head>
  <body onload="leheAlgus()">
     <div id="vastus"></div>
  </body>
</html>
```

Käsklus console.log trükib tulemuse testimiseks mõeldud konsooliaknasse (enamasti saab lahti veebilehisejas klahvivajutusega F12)



Lehel kihis näidatud vastus on lihtsalt tekstina seal:

kollane

Mitu foori

Objektitüüp ning objekt ise on eri asjad. Ehk siis ühte tüüpi võib korraga olla mitu objekti. Ning vahel võib mõne tüübi juures töötav objekt ka sootuks puududa. Siin näites siis korraga kaks töötavat foori. Sündides mõlemad rohelis tulega. Edasi muudetakse üks kollaseks ja teine punaseks. Ning taas saab tulemused seest välja küsida:

```
<!doctype html>
<html>
  <head>
    <title>Foor</title>
    <script>
       function foor(){
          this.tuli="roheline";
          this.kysiTuli=function(){return this.tuli;}
          this.muudaTuli=function(uusTuli){this.tuli=uusTuli;}
       }
       function leheAlgus(){
          var f1=new foor();
          var f2=new foor();
          f1.muudaTuli("kollane");
          f2.muudaTuli("punane");
          document.getElementById("vastus").innerHTML=
            "esimene: "+f1.kysiTuli()+
            ", teine: "+f2.kysiTuli();
       }
    </script>
  </head>
  <body onload="leheAlgus()">
     <div id="vastus"></div>
  </body>
</html>
```

esimene: kollane, teine: punane

Ülesandeid

- Pane näited tööle
- Koosta tüüp Lamp, väljaks "seisund" väärtusega true. Koosta käsklus kasSees(), mis siis väljastab seisundi väärtuse, testi.
- Lisa käsklus uusSeisund(asend), mille juures siis vastavalt määratakse seisundi uus väärtus. Testi.
- Lisa käsklus seisundTekstina(), kus siis vastavalt seisundi väärtusena väljastatakse tekstina "põleb" või "ei põle".
- Lisa käsklus vahetaSeisund(), mis siis muutuja "seisund" väärtuse muudab vastupidiseks. Katseta.
- Loo korraga mitu lampi, igaühel oma seisund. Vaheta lampide seisundeid ning veendu, et igaüks näitab õigesti.
- Tee seisundi vahetamiseks ning küsimiseks igale lambile eraldi nupupaar. Katseta ning veendu, et lülitamine töötab õigesti.
- Lisa lambile graafiline liides. Sisselülitatud lambi puhul on näha seest täis ring, väljalülitatud lambi puhul tühi ringjoon.

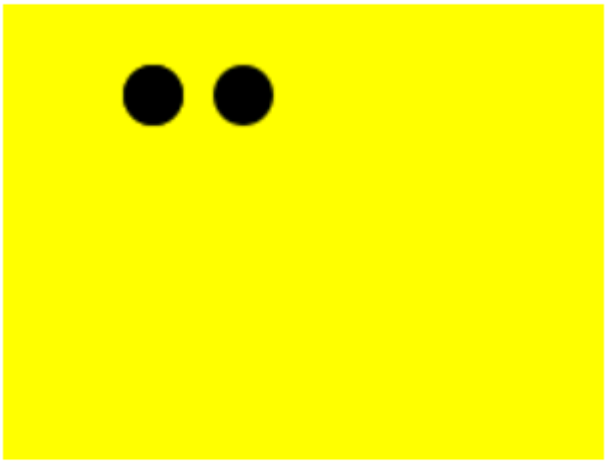
Kujundite lisamine platsile

Objektitüüpe võib lehel olla mitu. Järgmises näiteks on tüüpideks Pall ja Plats. Pall mõistab oma asukoha meeles pidada ning etteantud graafilise konteksti kaudu joonistada. Plats kuvab end kollase ristkülikuna, tema küljes kujundite massiivis on pallid ning ta suudab nad välja joonistada. Algusnäide koostab platsi kahe palliga ning kuvab nad ekraanile.

```
<!doctype html>
<html>
  <head>
    <title>Kujundid</title>
    <script>
       function Pall(x, y, r){
          this.x=x;
          this.y=y;
          this.r=r;
          this.joonista=function(g){
           g.beginPath();
           g.arc(this.x, this.y, this.r, 0, 2*Math.PI, true);
           g.fill();
          }
       }
       function Plats(kihiId){
          window[kihiId+"_joonis"]=this;
          this.alusta=function(){
             var sisu=
               "<canvas id='"+kihiId+"_tahvel' width='200' height='150' "+
                " style='background-color: yellow' ></canvas><br/>";
             document.getElementById(kihiId).innerHTML=sisu;
             this.kujundid=new Array();
             this.tahvel=document.getElementById(kihiId+"_tahvel");
          }
          this.lisaKujund=function(kujund){
            this.kujundid.push(kujund);
            this.joonista();
          }
          this.joonista=function(){
```

```
            var g=this.tahvel.getContext("2d");
            for(var i=0; i<this.kujundid.length; i++){
               this.kujundid[i].joonista(g);
            }
         }
         this.alusta();
      }

      function algus(){
        kiht1_joonis=new Plats("kiht1");
        kiht1_joonis.lisaKujund(new Pall(50, 30, 10));
        kiht1_joonis.lisaKujund(new Pall(80, 30, 10));
      }
    </script>
  </head>
  <body onload="algus();">
     <div id="kiht1"></div>
  </body>
</html>
```



Ülesandeid

- Pane näide käima
- Katseta mitmesuguste suurustega pallidega
- Koosta tsükkel platsile pallide lisamiseks
- Lisa platsile parameetrid loodava platsi suuruse kohta pikslites
- Lisa platsile äärejoon ja määra selle kaugus servast
- Lisa platsile käsklus soovitud arvu pallide tekitamiseks juhuslikesse kohtadesse äärejoone sisse

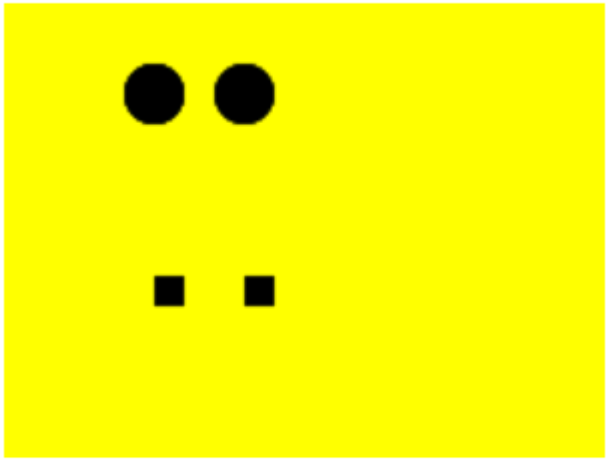
Mitmesugused kujundid

Siin näites lisandub Platsile Palli kõrvale kujundiks Ruut. Et Javaskript massiividesse andmete lisamisel piiranguid ei sea, siis saab siin samasse kujundite massiivi panna läbisegi mõlemaid kujundeid. Ning et neil on mõlemal ühine meetod nimega joonista(), siis saab kujundite massiivi nõnda ka välja kuvada. Selleks kutsutakse Platsi joonista-funktsioonis järgemööda tsükli abil välja

kõikide kujundite joonista-funktsioonid.

```
<!doctype html>
<html>
  <head>
    <title>Kujundid</title>
    <script>
       function Ruut(x, y, a){
          this.x=x;
          this.y=y;
          this.a=a;
          this.joonista=function(g){
             g.fillRect(this.x, this.y, this.a, this.a);
          }
       }
       function Pall(x, y, r){
          this.x=x;
          this.y=y;
          this.r=r;
          this.joonista=function(g){
           g.beginPath();
           g.arc(this.x, this.y, this.r,
                          0, 2*Math.PI, true);
           g.fill();
          }
       }
       function Plats(kihiId){
          window[kihiId+"_joonis"]=this;
          this.alusta=function(){
             var sisu=
               "<canvas id='"+kihiId+"_tahvel' width='200' height='150' "+
                " style='background-color: yellow' ></canvas><br/>";
             document.getElementById(kihiId).innerHTML=sisu;
             this.kujundid=new Array();
             this.tahvel=document.getElementById(kihiId+"_tahvel");
          }
          this.lisaKujund=function(kujund){
            this.kujundid.push(kujund);
            this.joonista();
          }
          this.joonista=function(){
             var g=this.tahvel.getContext("2d");
             for(var i=0; i<this.kujundid.length; i++){
                this.kujundid[i].joonista(g);
             }
          }
          this.alusta();
       }

       function algus(){
         kiht1_joonis=new Plats("kiht1");
         kiht1_joonis.lisaKujund(new Pall(50, 30, 10));
         kiht1_joonis.lisaKujund(new Pall(80, 30, 10));
         kiht1_joonis.lisaKujund(new Ruut(50, 90, 10));
         kiht1_joonis.lisaKujund(new Ruut(80, 90, 10));
       }
    </script>
  </head>
  <body onload="algus();">
     <div id="kiht1"></div>
  </body>
</html>
```



Ülesandeid

- Pane näide käima
- Paiguta ruute ja palle mitmesugustesse kohtadesse
- Lisa kujundina Puu, mis koosneb võrast ja tüvest. Ette saab anda võra keskkoha, võra raadiuse, tüve paksuse ning pikkuse

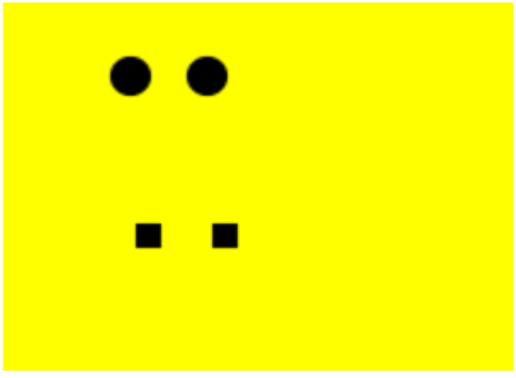
Liikuvad kujundid

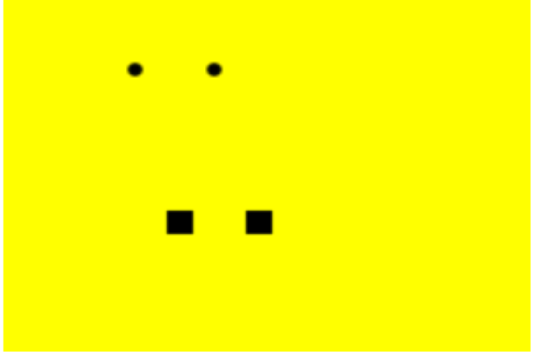
Liikumise jaoks lisati mõlemale objektitüübile funktsioon liigu. Ruut suurendab selle peale oma x-koordinaadi väärtust ühe võrra, Pall aga kahendab raadiust 90% peale. Igatahes mõlemad muutuvad liikumise peale nähtavalt. Platsi liikumisfunktsioon käivitab järgemööda kõigi kujundite liikumisfunktsioonid ning edasi Platsi joonistusfunktsiooni, mis omakorda kõik kujundid välja joonistab.

```
<!doctype html>
<html>
  <head>
    <title>Kujundid</title>
    <script>
       function Ruut(x, y, a){
          this.x=x;
          this.y=y;
          this.a=a;
          this.joonista=function(g){
             g.fillRect(this.x, this.y, this.a, this.a);
          }
          this.liigu=function(){
            this.x++;
          }
       }
       function Pall(x, y, r){
          this.x=x;
          this.y=y;
          this.r=r;
          this.joonista=function(g){
           g.beginPath();
           g.arc(this.x, this.y, this.r, 0, 2*Math.PI, true);
           g.fill();
          }
          this.liigu=function(){
```

```
            this.r*=0.9;
          }
        }
        function Plats(kihiId){
          window[kihiId+"_joonis"]=this;
          this.alusta=function(){
            var sisu=
              "<canvas id='"+kihiId+"_tahvel' width='200' height='150' "+
               " style='background-color: yellow' ></canvas><br/>";
            document.getElementById(kihiId).innerHTML=sisu;
            this.kujundid=new Array();
            this.tahvel=document.getElementById(kihiId+"_tahvel");
            setInterval(kihiId+"_joonis.liigu()", 1000);
          }
          this.lisaKujund=function(kujund){
            this.kujundid.push(kujund);
            this.joonista();
          }
          this.joonista=function(){
            var g=this.tahvel.getContext("2d");
            g.clearRect(0, 0, 200, 150);
            for(var i=0; i<this.kujundid.length; i++){
              this.kujundid[i].joonista(g);
            }
          }
          this.liigu=function(){
            for(var i=0; i<this.kujundid.length; i++){
              this.kujundid[i].liigu();
            }
            this.joonista();
          }
          //Lisage kolmandaks kujundiks kukkuv seest tühi ring.
          this.alusta();
        }

        function algus(){
          kiht1_joonis=new Plats("kiht1");
          kiht1_joonis.lisaKujund(new Pall(50, 30, 10));
          kiht1_joonis.lisaKujund(new Pall(80, 30, 10));
          kiht1_joonis.lisaKujund(new Ruut(50, 90, 10));
          kiht1_joonis.lisaKujund(new Ruut(80, 90, 10));
        }
      </script>
    </head>
    <body onload="algus();">
       <div id="kiht1"></div>
    </body>
</html>
```





Ülesandeid

- Pane näide käima
- Pane ruut liikuma vasakule
- Lisa kujundina alla kukkuv tühi ring

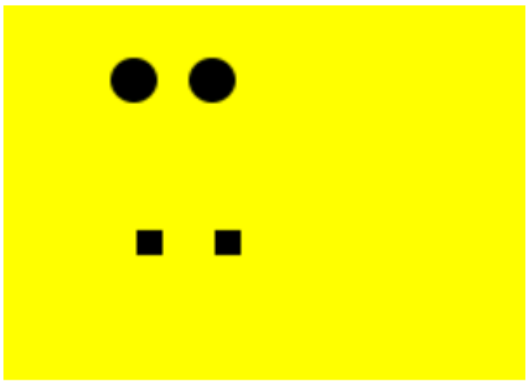
Hiirele reageeerivad liikuvad kujundid

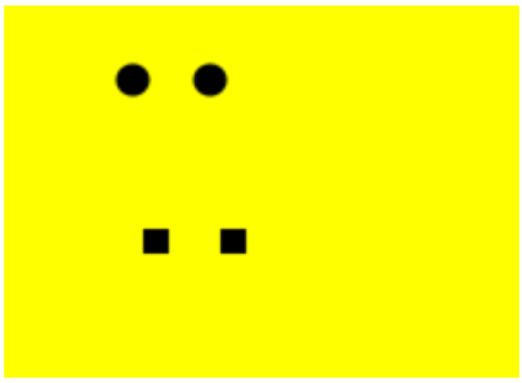
Kui kujundid jälgivad ühist käskude struktuuri, siis saab sellise "hulgikaubanduse" abil neile mitmesuguseid oskusi külge pookida. Siin näites kipuvad ruudud aegamisi paremale ekraanilt välja nihkuma ning ringid väga väikeseks minema. Näitena lisati aga oskused, kus saab hiire abil ruudu jälle lähemale kutsuda ning ringi suuremaks tagasi muuta. Ruut kontrollib, et kui hiire asukoha x-koordinaat on ruudust vasakul, siis leitakse x-i suunaline kaugus hiirest ning peegeldatakse ruut sama kaugele vasakule. Palli puhul aga antakse hiirega palli tabamise puhul lihtsalt taas raadiuseks 20 pikslit. Et hiire jälgimine töötaks, selleks taas püüab plats kõigepealt ise hiiresündmuse kinni.

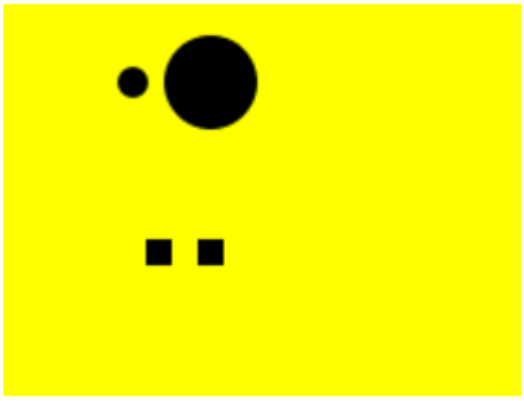
```
<!doctype html>
<html>
  <head>
    <title>Kujundid</title>
    <script>
       function Ruut(x, y, a){
          this.x=x;
          this.y=y;
          this.a=a;
          this.joonista=function(g){
             g.fillRect(this.x, this.y, this.a, this.a);
          }
          this.liigu=function(){
            this.x++;
          }
          this.hiirAlla=function(hx, hy){
             if(hx<this.x){
               var vahe=this.x-hx;
               this.x=hx-vahe;
             }
          }
       }
       function Pall(x, y, r){
          this.x=x;
          this.y=y;
          this.r=r;
          this.joonista=function(g){
           g.beginPath();
           g.arc(this.x, this.y, this.r,
                         0, 2*Math.PI, true);
           g.fill();
          }
          this.liigu=function(){
            this.r*=0.9;
          }
          this.hiirAlla=function(hx, hy){
             var dx=hx-this.x;
             var dy=hy-this.y;
             var kaugus=Math.sqrt(dx*dx+dy*dy);
             if(kaugus<this.r){this.r=20;}
          }
```

```
      }
      function Plats(kihiId){
         window[kihiId+"_joonis"]=this;
         this.alusta=function(){
            var sisu=
              "<canvas id='"+kihiId+"_tahvel' width='200' height='150' "+
               " style='background-color: yellow' "+
               " onmousedown='"+kihiId+
                           "_joonis.hiirAlla(event)' ></canvas><br/>";
            document.getElementById(kihiId).innerHTML=sisu;
            this.kujundid=new Array();
            this.tahvel=document.getElementById(kihiId+"_tahvel");
            setInterval(kihiId+"_joonis.liigu()", 1000);
         }
         this.lisaKujund=function(kujund){
           this.kujundid.push(kujund);
           this.joonista();
         }
         this.joonista=function(){
            var g=this.tahvel.getContext("2d");
            g.clearRect(0, 0, 200, 150);
            for(var i=0; i<this.kujundid.length; i++){
               this.kujundid[i].joonista(g);
            }
         }
         this.liigu=function(){
            for(var i=0; i<this.kujundid.length; i++){
               this.kujundid[i].liigu();
            }
            this.joonista();
         }
         this.hiirAlla=function(e){
            var tahvlikoht=this.tahvel.getBoundingClientRect();
            var hx=e.clientX-tahvlikoht.left;
            var hy=e.clientY-tahvlikoht.top;
            for(var i=0; i<this.kujundid.length; i++){
               this.kujundid[i].hiirAlla(hx, hy);
            }
         }
         this.alusta();
      }

      function algus(){
        kiht1_joonis=new Plats("kiht1");
        kiht1_joonis.lisaKujund(new Pall(50, 30, 10));
        kiht1_joonis.lisaKujund(new Pall(80, 30, 10));
        kiht1_joonis.lisaKujund(new Ruut(50, 90, 10));
        kiht1_joonis.lisaKujund(new Ruut(80, 90, 10));
      }
    </script>
  </head>
  <body onload="algus();">
     <div id="kiht1"></div>
  </body>
</html>
```







Ülesandeid

- Pane näide käima
- Lisa kujundiks alla kukkuv seest tühi ring. Ringi tabamisel hakkab ta taas tahvli ülaservast kukkuma
- Lisa kujunditele funktsioon kasPihtas(hiireX, hiireY), mis vastab true või false vastavalt sellele, kas etteantud hiire koordinaadid asuvad kujundi seesl. Tee Platsi hiirAlla ümber nõnda, et iga kujundi juures kontrollitakse kõigepealt, et kas hiirega saadi kujundile pihta. Edasi alles hiirega tabamuse puhul kutsutakse välja vastava kujundi kasPihtas. See võimaldab kujundi hiirAlla funktsiooni juurest tabamuse kontrolli välja võtta.