Lahendusi ja täiendusi

Päris algus

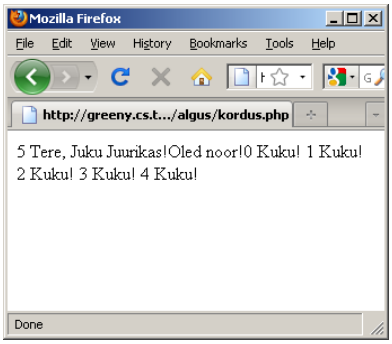
* Käivita konspektis olnud näide kahe arvu liitmise kohta.
* Muuda arve ja tehet, kontrolli tulemusi.

<?=4*6 ?>

Muutuja, valik ja kordus

* Lisa tervitatavale inimesele perekonnanimi eraldi muutujasse, trüki ka selle väärtus.
* Lisa tingimus üle saja-aastaste jaoks teatega "oled väga vana".
* Lisa for-tsükli sisse iga Kuku järjekorranumber.

```
<?= 3+2 ?>

<?php
  $eesnimi="Juku";
  $perekonnanimi="Juurikas";
  echo "Tere, $eesnimi $perekonnanimi!";

  $vanus=5;
  if($vanus<7){
     echo "Oled noor!";
  }

  if($vanus>100){
     echo "Oled hirmus vana!";
  }

  for($i=0; $i<5; $i++){
    echo "$i Kuku! ";
  }
?>
```



PHP HTMLi sees

Kahe sisestatud arvu korrutis

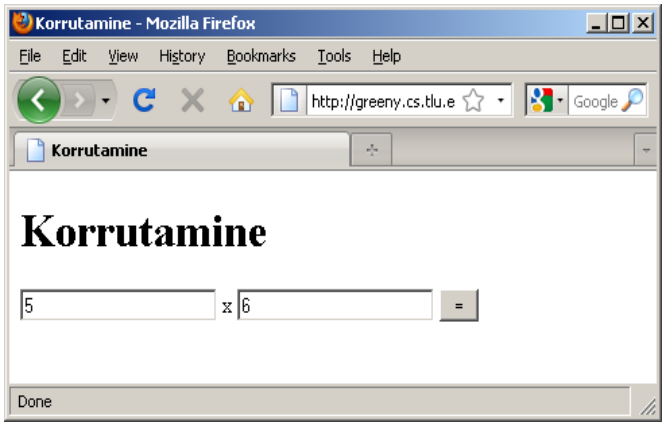
Korrutamise puhul saab PHP ise aru, et pooled võiksid olla arvud. Seega annab soovitud tulemuse ka

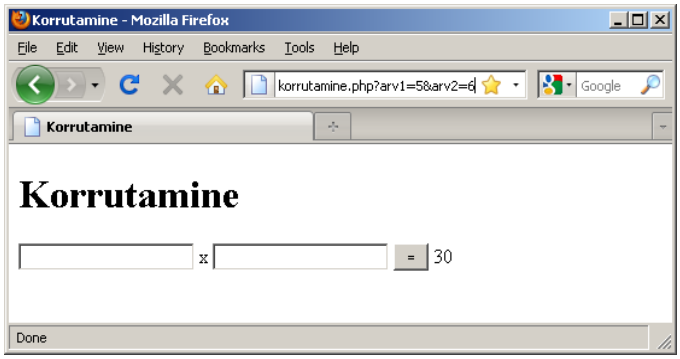
```
echo $_REQUEST["arv1"]*$_REQUEST["arv2"];
```

Kui tahta kindlasti andmetüübiks määrata täisarv, siis aitab funktsioon intval. Ehk siis

```
echo intval($_REQUEST["arv1"])*intval($_REQUEST["arv2"]);
```

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html>
 <head>
     <title>Korrutamine</title>
     <meta http-equiv="Content-Type"
           content="text/html; charset=iso-8859-1" />
 </head>
 <body>
   <h1>Korrutamine</h1>
   <form action="korrutamine.php">
      <input type="text" name="arv1" /> x
      <input type="text" name="arv2" />
      <input type="submit" value="=" />
      <?php
         if(isSet($_REQUEST["arv1"])){
           echo $_REQUEST["arv1"]*$_REQUEST["arv2"];
         }
      ?>
   </form>
 </body>
</html>
```





Korrutustabel

Pealkirjalahtriks tuleb veeru number, sest korrutustabeli puhul saab selle järgi otsida korrutatavale vastust. Sobivaks kujunduseks antakse siin lahtrile klassiks pealkiri, mille kohta siis üleval laadiplokis määratakse rasvane kujundus.

```
echo "<td class='pealkiri'>$veerg</td>";
```

Tavalised lahtrid korrutustabelis saavad oma väärtuseks rea ja veeru numbri korrutise. Tehte puhul tuleb väljatrükitava teksti jutumärgid ära katkestada, avaldis välja arvutada ning punkt-operaatori abil kogu lugu taas üheks tervikuks väljatrükitavaks tekstiks kokku liita.

```
echo "<td>".($rida*$veerg)."</td>";
```

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html>
 <head>
     <title>Korrutustabel</title>
     <meta http-equiv="Content-Type"
            content="text/html; charset=iso-8859-1" />
    <style type="text/css">
       td{
         text-align:right;
       }
       .pealkiri {
         font-weight: bold;
       }
    </style>
 </head>
 <body>
   <h1>Korrutustabel</h1>
   <form action="korrutustabel.php">
      <input type="text" name="ridu" /> rida <br />
      <input type="text" name="veerge" /> veergu  <br />
      <input type="submit" value="Loo" />
      <?php
         if(isSet($_REQUEST["ridu"])){
           $ridu=intval($_REQUEST["ridu"]);
           $veerge=intval($_REQUEST["veerge"]);
           echo "<table>";
           for($rida=0; $rida<=$ridu; $rida++){
             echo "<tr>";
             if($rida==0){ //pealkirjarida
                echo "<td></td>";
                for($veerg=1; $veerg<=$veerge; $veerg++){
                  echo "<td class='pealkiri'>$veerg</td>";
                }
             } else {
               echo "<td class='pealkiri'>$rida</td>";
               for($veerg=1; $veerg<=$veerge; $veerg++){
```

```
                    echo "<td>".($rida*$veerg)."</td>";
                  }
                }
                echo "</tr>";
              }
              echo "</table>";
            }
        ?>
    </form>
  </body>
</html>
```

Lehe koostamine alamosadest

Failitükke ühendav matkalehestik

p2is.php

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html>
 <head>
      <title>Matka leht</title>
      <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
      <style type="text/css">
        @import "kujundus.css";
      </style>
 </head>
 <body>
 <?php require("menyy.php"); ?>
```

menyy.php

```
<div id="menyy">
  <h2>Men&uuml;&uuml;</h2>
  <ul>
     <li><a href="osalejad.php">Osalejad</a></li>
     <li><a href="varustus.php">Varustus</a></li>
     <li><a href="marsruut.php">Marsruut</a></li>
  </ul>
</div>
```

kujundus.css

```
body{
  background-color: #ffeb90;
}


#menyy li {
  display: inline;
  width: 200px;
}

#menyy h2{
  display: none;
}
```

jalus.php

```
  <div id="loputeade">Head matkamist!</div>
  </body>
</html>
```

osalejad.php

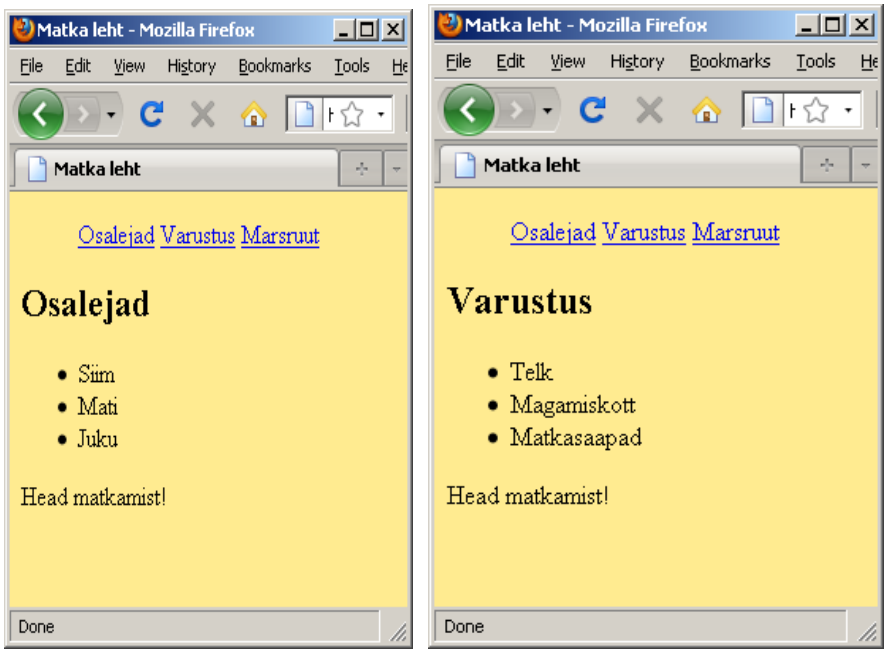
```
<?php require("p2is.php"); ?>
  <div id="sisu">
    <h2>Osalejad</h2>
    <ul>
      <li>Siim</li>
      <li>Mati</li>
      <li>Juku</li>
    </ul>
  </div>
<?php require("jalus.php"); ?>
```

varustus.php

```
<?php require("p2is.php"); ?>
  <div id="sisu">
    <h2>Varustus</h2>
    <ul>
      <li>Telk</li>
      <li>Magamiskott</li>
      <li>Matkasaapad</li>
    </ul>
  </div>
<?php require("jalus.php"); ?>
```

marsruut.php

```
<?php require("p2is.php"); ?>
  <div id="sisu">
    <h2>Marsruut</h2>
    <ul>
      <li>Lihula</li>
      <li>Kirbla</li>
      <li>Virtsu</li>
    </ul>
  </div>
<?php require("jalus.php"); ?>
```



Andmefailid eraldi kataloogis

Laulik

p2is.php

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
 <head>
      <title>Laulik</title>
      <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
      <style type="text/css">
        @import "kujundus.css";
      </style>
 </head>
 <body>
 <?php require("menyy.php"); ?>
```

kujundus.css

```
body{
  background-color: #ffeb90;
}

#menyy{
 float: left;
 padding-right: 30px;
}

#sisu{
 width: 70%;
 float: left;
}

#loputeade{
 clear: left;
}
```

menyy.php

Andmete url-reale sobival kujul edasi andmiseks on loodud funktsioon urlencode - too asendab tühikud plussmärkidega ning muud erisümbolid kuueteistkümnendkoodiga, millele eelneb protsendimärk - nii nagu URLi andmete koostamise eeskiri ette näeb.

```
<div id="menyy">
  <h2>Men&uuml;&uuml;</h2>
  <ul>
   <?php
    $failinimed=scandir("laulud");
    foreach($failinimed as $failinimi){
      $m=explode(".", $failinimi);
      if($m[1]=="txt"){
        $failurl=urlencode($failinimi);
        $failhtml=htmlspecialchars($m[0]);
        echo "<li><a href='lauluvaataja.php?laulufail=$failurl'>$failhtml</a></li>";
      }
    }
   ?>
  </ul>
</div>
```

lauluvaataja.php

Sisseloetavas failinimes asendatakse kindlasti ära kaldkriips ning kaks punkti - vahendid, mille kaudu oleks muidu võimalik asuda veebiserveri kataloogide vahel liikuda ning ka neid andmeid välja õngitseda, mis isegi mitte veebi nähtavates kataloogides ei paikne. Käsu file_get_contents ees on @-märk seetõttu, et siis ei satu veateated olematu faili või muu probleemi tõttu tavakasutajaid häirima ega sissemurdjaid abistama.

```
<?php require("p2is.php"); ?>
<div id="sisu">
 <pre><?php
     if(isSet($_REQUEST["laulufail"])){
        $asendatavad=array("/", "..");
        $fnimi=str_replace($asendatavad, "", $_REQUEST["laulufail"]);
        $sisu=@file_get_contents("laulud/".$fnimi);
        if($sisu){
          echo $sisu;
        } else {
          echo "Andmed puuduvad";
        }
     } else {
       echo "Failinimi puudub";
     }
   ?>
 </pre>
</div>
<?php require("jalus.php"); ?>
```

Laulutekstid eraldi kataloogis nimega laulud.

```
laulud/postipoiss.txt

Postipoiss

Palju aastaid mööda läinud sellest a'ast,
kui veel olemas ei olnud meie maal
ronge, autosid, jalgrattaid
ega tehtud pikki matkaid,
nagu tänapäeval kõikjal näha saad.
```

Uhke postipoiss sõitis kord maanteel,
külast külla, linnast linna viis ta tee.
Kõikjal uudiseid tõi postipoiss kaugelt -
nüüd vaid möödund ajast mälestus on see.
On see, on see ...

Külakõrts ja postijaam seal üheskoos.
Soojas kambris oli tuju ikka hoos.
Siis kui jalgu puhkas hobu,
peeti külmarohust lugu
ja siis sõit läks lahti jälle täies hoos.

Kirja kallimalt ka postipoiss seal tõi.
Vahel suudluse ta selle eest ka sai.
Mitu hõberaha taskus
selle vaeva ära tasus,
mida teekond pikk tal kaasa tuua võis.

Talvel lumi tuiskas, tormas maantee pääl.
Juba kaugelt kostis aisakella hääl.
Soojas kasukas seal sõitis
mõni rikas linnakaupmees.
Uinus magusasti, habe härmas ees.

laulud/viljandi_paadimees.txt

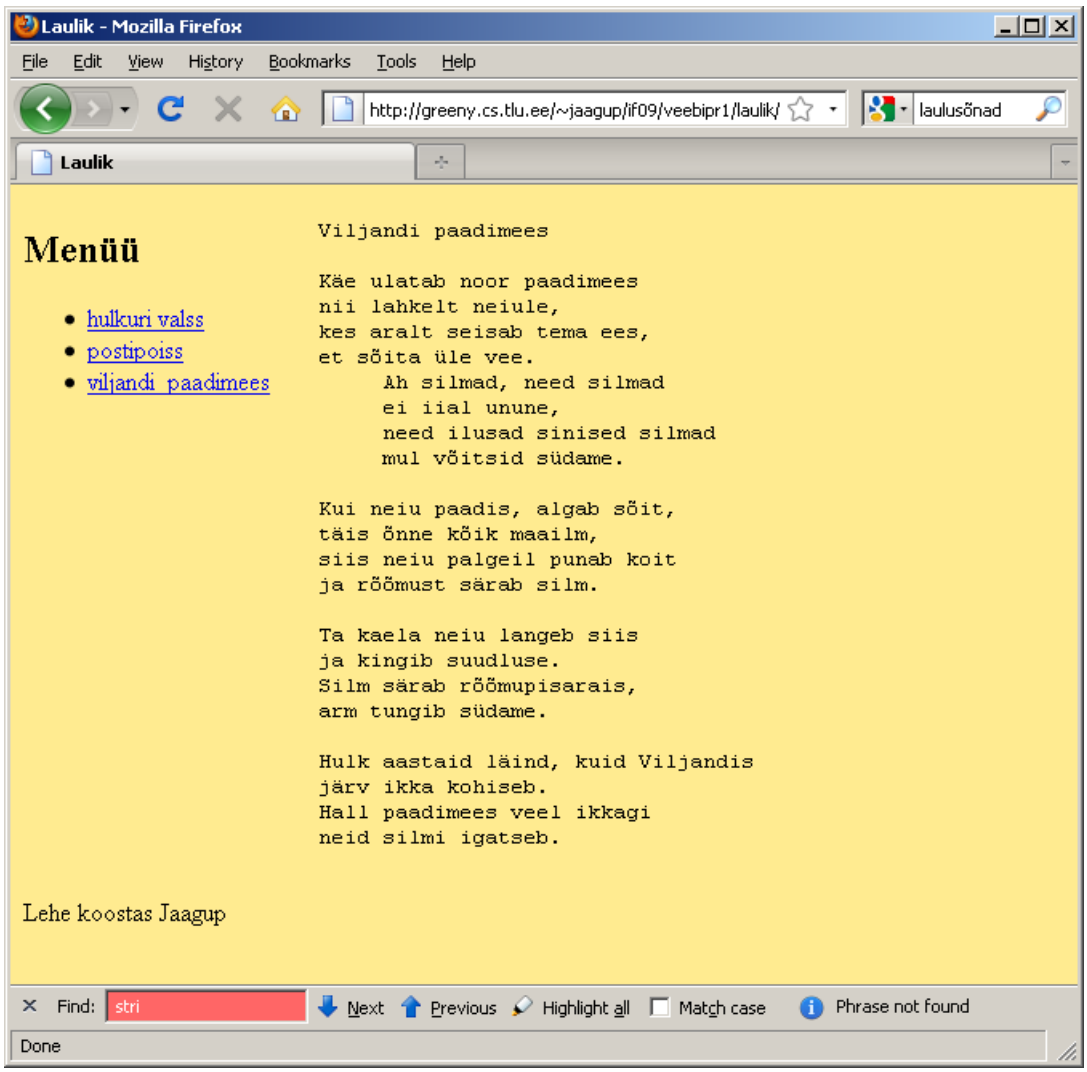
Viljandi paadimees

Käe ulatab noor paadimees
nii lahkelt neiule,
kes aralt seisab tema ees,
et sõita üle vee.
Ah silmad, need silmad
ei iial unune,
need ilusad sinised silmad
mul võitsid südame.

Kui neiu paadis, algab sõit,
täis õnne kõik maailm,
siis neiu palgeil punab koit
ja rõõmust särab silm.

Ta kaela neiu langeb siis
ja kingib suudluse.
Silm särab rõõmupisarais,
arm tungib südame.

Hulk aastaid läind, kuid Viljandis
järv ikka kohiseb.
Hall paadimees veel ikkagi
neid silmi igatseb.

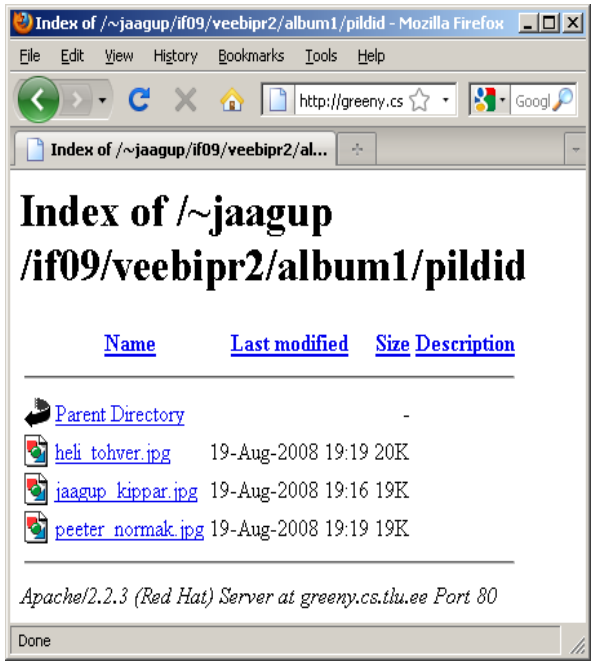


Pildialbum

Menüüsse näidatakse kataloogis "pildid" olevatest failidest vaid need, mis on loetelus lubatud laiendiga. Lehele pannakse pildiviide etteantud failinimega. Erinevalt eelnevast laulikunäitest pole siin muret, et selle koodifaili järgi oleks võimalik veebiserverist andmeid õngitsema hakata. Kuna failinimi läheb ainult HTML-faili sisse, peab brauser hakkama ise andmeid küsima. Ning valest kohast juba veebiserver ise neid kätte ei anna.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
 <head>
      <title>Pildid</title>
      <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
      <style type="text/css">
         #menyy{float: left;}
         #sisu {float: left; padding: 30px;}
      </style>
 </head>
 <body>
 <div id="menyy">
  <h2>Men&uuml;&uuml;</h2>
  <ul>
   <?php
    $failinimed=scandir("pildid");
    $lubatud=array("gif", "jpg", "jpeg", "png");
    foreach($failinimed as $failinimi){
```

```
      $m=explode(".", $failinimi);
      if(in_array($m[1], $lubatud)){
        $pilturl=urlencode($failinimi);
        echo "<li><a href='pildivaataja.php?pildifail=$pilturl'>$failinimi</a></li>";
      }
    }
   ?>
  </ul>
 </div>
 <div id="sisu">
    <?php
      if(isSet($_REQUEST["pildifail"])){
        echo "<img src='pildid/$_REQUEST[pildifail]' alt=''>";
      }
    ?>
 </div>
 </body>
</html>
```



Piltviidetega album

Võrreldes eelnevaga näidatakse failinime kõrval ka selle pilti. Laiskuse ja koodi lihtsuse tõttu ei ole seda pilti eraldi väiksemaks tehtud, vaid lihtsalt HTML-käsuga määratakse menüüpiltidele väiksem suurus. Paljude piltide korral tekitab selline lähenemine jõudlusprobleeme. Et menüüd keritaks kõrval ja ta ei muudaks lehte liiga pikaks, selleks on menüü laadikäsklustesse lisatud

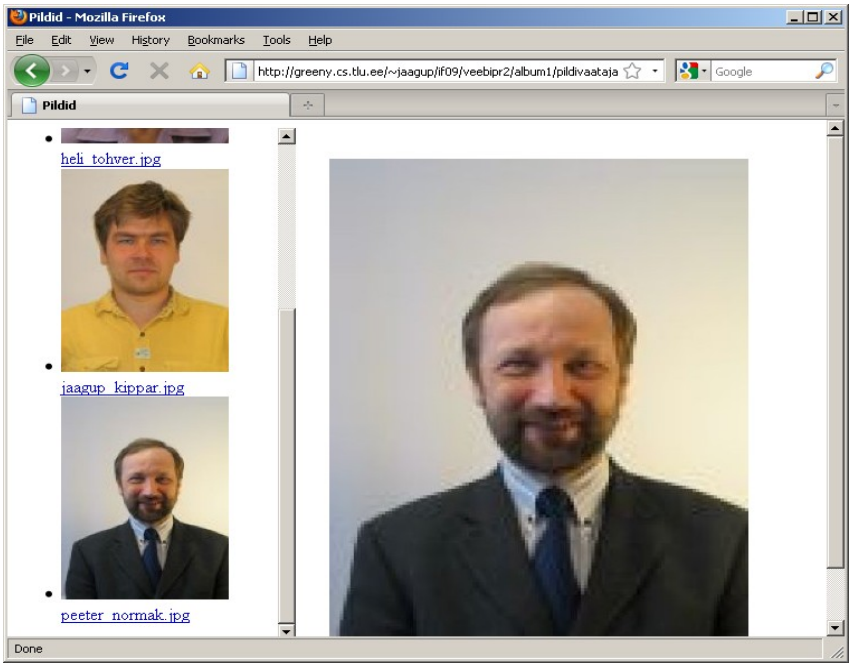
```
          overflow: scroll;
          overflow-x: hidden;
```

Faili nimeks

```
pildivaataja2.php
```

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
 <head>
      <title>Pildid</title>
      <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
      <style type="text/css">
         #menyy{
           float: left;
           overflow: scroll;
```

```
            overflow-x: hidden;
            height: 500px;
            width: 250px;
          }
          #sisu {float: left; padding: 30px;}
          .menyypilt{width: 150px; border: 0px;}
      </style>
 </head>
 <body>
 <div id="menyy">
  <h2>Men&uuml;&uuml;</h2>
  <ul>
   <?php
    $failinimed=scandir("pildid");
    $lubatud=array("gif", "jpg", "jpeg", "png");
    foreach($failinimed as $failinimi){
      $m=explode(".", $failinimi);
      if(in_array($m[1], $lubatud)){
        $pilturl=urlencode($failinimi);
        echo "<li>
            <a href='pildivaataja2.php?pildifail=$pilturl'><img
               src='pildid/$failinimi' class='menyypilt' alt=''></a>
            <a href='pildivaataja2.php?pildifail=$pilturl'><br />$failinimi</a>
        </li>";
      }
    }
   ?>
  </ul>
 </div>
 <div id="sisu">
    <?php
      if(isSet($_REQUEST["pildifail"])){
        echo "<img src='pildid/$_REQUEST[pildifail]' alt=''>";
      }
    ?>
 </div>
 </body>
</html>
```



Ühe andmetabeliga seotud veebilehestik

Veiste loetelu

```
CREATE TABLE veised(
  id INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
  veisenimi VARCHAR(50),
  mass INTEGER,
  vanus DOUBLE
);


INSERT INTO veised (veisenimi, mass, vanus) VALUES ('Maasu', 400, 5);
INSERT INTO veised (veisenimi, mass, vanus) VALUES ('Tipa',  500, 3.5);
INSERT INTO veised (veisenimi, mass, vanus) VALUES ('Mammu', 300, 6);

mysql> SELECT * FROM veised;
+----+-----------+------+-------+
| id | veisenimi | mass | vanus |
+----+-----------+------+-------+
|  1 | Maasu     |  400 |     5 |
|  2 | Tipa      |  500 |   3.5 |
|  3 | Mammu     |  300 |     6 |
+----+-----------+------+-------+
3 rows in set (0.00 sec)

<?php
  $yhendus=new mysqli("localhost", "jaagup", "xxxxxx", "jaagup");
  $kask=$yhendus->prepare("SELECT id, veisenimi, mass, vanus FROM veised");
  $kask->bind_result($id, $veisenimi, $mass, $vanus);
  $kask->execute();
?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
     <title>Veised lehel</title>
  </head>
  <body>
     <h1>Veiste loetelu</h1>
     <ul>
     <?php
       while($kask->fetch()){
         $veisenimi=htmlspecialchars($veisenimi);
         echo "
           <li>
              $veisenimi - $mass kg, $vanus aastat
           </li>
         ";
       }
     ?>
     </ul>
  </body>
</html>
<?php
  $yhendus->close();
?>
```

Veised tabelina

Suurema massiga veistele lisatakse tabeli lahtrile klass "rammus", laadiga tehakse selle tekst rasvaseks.

```
<?php
  $yhendus=new mysqli("localhost", "jaagup", "xxxxxx", "jaagup");
  $kask=$yhendus->prepare("SELECT id, veisenimi, mass, vanus FROM veised");
  $kask->bind_result($id, $veisenimi, $mass, $vanus);
  $kask->execute();
?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
     <title>Veised lehel</title>
     <style type="text/css">
       .rammus {font-weight: bold;}
     </style>
  </head>
  <body>
     <h1>Veiste tabel</h1>
     <table>
      <tr>
        <th>Veise nimi</th>
        <th>mass (kg)</th>
        <th>vanus (a)</th>
      </tr>
     <?php
       while($kask->fetch()){
         $veisenimi=htmlspecialchars($veisenimi);
         $lisand="";
         if($mass>450){
           $lisand=" class='rammus'";
         }
         echo "
           <tr>
              <td>$veisenimi</td>
              <td $lisand>$mass</td>
              <td>$vanus</td>
           </tr>
         ";
```

```
        }
      ?>
      </table>
  </body>
</html>
<?php
  $yhendus->close();
?>
```

Valitud veise vaatamine

Lehe algul luuakse veisenimedest menüü. Nimele vajutades avatakse sama leht ning saadetakse aadressiribal parameetrina kaasa valitud veise id.

Allpool kontrollitakse, et kas saabus parameeter nimega id. Kui jah, siis püütakse selle id järgi kätte saada ka veise ülejäänud andmed ning neid lehel näidata. Inglisekeelses kirjanduses kiputakse sellise lahenduse kohta ütlema master/detail vaade. Et pikk loetelu(menüü) on master ning valitud kirjet saab detailselt vaadata.

```
<?php
  $yhendus=new mysqli("localhost", "jaagup", "xxxxxx", "jaagup");
?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html>
  <head>
    <title>Veised lehel</title>
    <style type="text/css">
       #menyykiht{
         float: left;
         padding-right: 30px;
       }
       #sisukiht{
         float:left;
       }
       #jalusekiht{
         clear: left;
       }
    </style>
```

```
  </head>
  <body>
    <div id="menyykiht">
        <h2>Veiste loetelu</h2>
        <ul>
          <?php
             $kask=$yhendus->prepare("SELECT id, veisenimi FROM veised");
             $kask->bind_result($id, $veisenimi);
             $kask->execute();
             while($kask->fetch()){
               echo "<li><a href='veisteleht.php?id=$id'>".
                     htmlspecialchars($veisenimi)."</a></li>";
             }
          ?>
        </ul>
    </div>
    <div id="sisukiht">
       <?php
         if(isSet($_REQUEST["id"])){
            $kask=$yhendus->prepare(
              "SELECT id, veisenimi, vanus, mass FROM veised WHERE id=?");
             //Kysim2rgi asemele pannakse aadressiribalt tulnud id,
             //eeldatakse, et ta on tyybist integer (i).
             //(double - d, string - s)
            $kask->bind_param("i", $_REQUEST["id"]);
            $kask->bind_result($id, $veisenimi, $vanus, $mass);
            $kask->execute();
            if($kask->fetch()){
              echo "<h2>".htmlspecialchars($veisenimi)."</h2>";
              echo "$vanus aastat, $mass kilogrammi";
            } else {
              echo "Vigased andmed.";
            }
         } else {
           echo "Tere tulemast avalehele!
                  Vali men&uuml;&uuml;st sobiv veis.";
         }
       ?>
    </div>
    <div id="jalusekiht">
       Lehe tegi Jaagup
    </div>
  </body>
</html>
<?php
  $yhendus->close();
?>
```

Kaks seotud tabelit

Järgnevates näidetes hoitakse andmeid kahes tabelis. Ühes maakonnad maakonnanime ja -keskusega. Teises inimeste andmed. Kusjuures inimeste tabeli tulp maakonna_id näitab maakondade tabeli id peale.

```
maakonnad(id, maakonnanimi, maakonnakeskus)
inimesed (id, eesnimi, perekonnanimi, maakonna_id)
```

Maakondade tabeli loomine:

```
CREATE TABLE maakonnad(
  id INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
  maakonnanimi VARCHAR(30),
  maakonnakeskus VARCHAR(30)
);
```

Andmed sisse:

```
INSERT INTO maakonnad (maakonnanimi, maakonnakeskus) VALUES ('Harjumaa', 'Tallinn');
INSERT INTO maakonnad (maakonnanimi, maakonnakeskus) VALUES ('Raplamaa', 'Rapla');
INSERT INTO maakonnad (maakonnanimi, maakonnakeskus) VALUES ('Saaremaa', 'Kuressaare');
INSERT INTO maakonnad (maakonnanimi, maakonnakeskus) VALUES ('Valgamaa', 'Valga');
```

Kontroll, et jõudsid kohale.

```
mysql> SELECT * FROM maakonnad;
+----+--------------+----------------+
| id | maakonnanimi | maakonnakeskus |
+----+--------------+----------------+
|  1 | Harjumaa     | Tallinn        |
|  2 | Raplamaa     | Rapla          |
|  3 | Saaremaa     | Kuressaare     |
|  4 | Valgamaa     | Valga          |
+----+--------------+----------------+
```

Inimeste tabeli loomine

```
CREATE TABLE inimesed(
```

```
  id INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
  eesnimi VARCHAR(30),
  perekonnanimi VARCHAR(30),
  maakonna_id INT,
  FOREIGN KEY(maakonna_id) REFERENCES maakonnad(id)
);
```

Andmed sisse

```
INSERT INTO inimesed (eesnimi, perekonnanimi, maakonna_id) VALUES ('Juku', 'Tamm', 3);
INSERT INTO inimesed (eesnimi, perekonnanimi, maakonna_id) VALUES ('Kati', 'Tamm', 3);
INSERT INTO inimesed (eesnimi, perekonnanimi, maakonna_id) VALUES ('Mati', 'Kask', 2);
```

Kahte tabelit ühendav päring. SELECT real tulpade nimed, mida näha tahetakse. FROM osas tabelite loetelu, kust andmeid saadakse. Ning WHERE-tingimusega määratakse, milliste tulpade väärtused peavad teisest tabelist rea võtmisel võrdsed olema.

```
SELECT eesnimi, perekonnanimi, maakonnanimi
 FROM inimesed, maakonnad
 WHERE inimesed.maakonna_id=maakonnad.id


+---------+--------------+-------------+
| eesnimi | perekonnanimi | maakonnanimi |
+---------+--------------+-------------+
| Juku    | Tamm          | Saaremaa     |
| Kati    | Tamm          | Saaremaa     |
| Mati    | Kask          | Raplamaa     |
+---------+--------------+-------------+
```

M-iga algavad eesnimed

```
mysql> SELECT * FROM inimesed WHERE eesnimi LIKE 'M%';
+----+---------+--------------+-------------+
| id | eesnimi | perekonnanimi | maakonna_id |
+----+---------+--------------+-------------+
|  3 | Mati    | Kask          |           2 |
+----+---------+--------------+-------------+
```

Kuressaares elavad inimesed. Lisaks tabeleid ühendavale tingimusele võib täiesti julgesti kasutada ka kõiksugu muid tingimusi. Vaid AND vahele üheaegse kehtivuse korral. Või kui lisandub valikuid ORi abil, siis enamasti on mõistlik sulgudega näidata, millised tingimused kokku kuuluvad - tabeleid siduv tingimus peab ikka eraldi jääma, muidu kipub andmete loetelu juures üllatusi tulema.

```
SELECT eesnimi, perekonnanimi, maakonnanimi
 FROM inimesed, maakonnad
 WHERE inimesed.maakonna_id=maakonnad.id
  AND maakonnad.maakonnakeskus='Kuressaare';


+---------+--------------+-------------+
| eesnimi | perekonnanimi | maakonnanimi |
+---------+--------------+-------------+
| Juku    | Tamm          | Saaremaa     |
| Kati    | Tamm          | Saaremaa     |
+---------+--------------+-------------+
```

Sorditud nimed

Andmete kättesaamiseks sobiv funktsioon. Vaikimisi antakse ette tulbanimi ja suund, mille järgi

sortida. Sellisel juhul võib funktsiooni rahumeeli ka ilma parameetriteta välja kutsuda.

Funktsioonist veateate tagastamisel on mitu võimalust. Põhiosas olnud näide tagastas teksti "sobimatu tulp" - see aga ei pruugi kasutajale piisavalt selget tagasisidet anda. Alles siis, kui saabunud andmeid kasutama hakati, anti teada, et tekstist ei saa kirjeid samamoodi välja võtma hakata kui massiivist. Siin tagastatakse esimese kontrolli juures vigase tulbanime puhul lihtsalt tühi massiiv - ehk siis funktsiooni tagastustüüp jääb endisega võrreldes samaks. Sellisel juhul ei tekigi lehele mingit veateadet, lihtsalt näidatavad andmed jäävad tühjaks. Teises kontrollis olev funktsioon die trükib lihtsalt veateate ning lõpetab programmi töö - sellisel juhul pole karta et kasutaja võiks valesid andmeid kahtlustama hakata.

abifunktsioonid.php

```
<?php
  $yhendus=new mysqli("localhost", "jaagup", "xxxxxx", "jaagup");

  function kysiInimesteAndmed($sorttulp="perekonnanimi", $sortsuund="ASC"){
    global $yhendus;
    $lubatudtulbad=array("eesnimi", "perekonnanimi",
                         "maakonnanimi", "maakonnakeskus");
    if(!in_array($sorttulp, $lubatudtulbad)){
      return array();
    }
    $lubatudsuunad=array("ASC", "DESC");
    if(!in_array($sortsuund, $lubatudsuunad)){
      die("Lubamatu suund");
    }
    $kask=$yhendus->prepare("SELECT inimesed.id, eesnimi,
         perekonnanimi, maakonnanimi, maakonnakeskus
       FROM inimesed, maakonnad
       WHERE inimesed.maakonna_id=maakonnad.id
       ORDER BY $sorttulp $sortsuund");
    //echo $yhendus->error;
    $kask->bind_result($id, $eesnimi, $perekonnanimi,
      $maakonnanimi, $maakonnakeskus);
    $kask->execute();
    $hoidla=array();
    while($kask->fetch()){
      $isik=new stdClass();
      $isik->id=$id;
      $isik->eesnimi=htmlspecialchars($eesnimi);
      $isik->perekonnanimi=htmlspecialchars($perekonnanimi);
      $isik->maakonnanimi=htmlspecialchars($maakonnanimi);
      $isik->maakonnakeskus=htmlspecialchars($maakonnakeskus);
      array_push($hoidla, $isik);
    }
    return $hoidla;
  }

  //---------------
  if( array_pop(explode("/", $_SERVER["PHP_SELF"]))=="abifunktsioonid.php"):
?>
<pre>
<?php
  print_r(kysiInimesteAndmed("eesnimi", "DESC"));
?>
</pre>
<?php endif ?>
```

Kõik andmed eesnimede järgi kahanevasse järjekorda sorditud:

```
Array
(
    [0] => stdClass Object
        (
```

```
            [id] => 3
            [eesnimi] => Mati
            [perekonnanimi] => Kask
            [maakonnanimi] => Raplamaa
            [maakonnakeskus] => Rapla
        )

    [1] => stdClass Object
        (
            [id] => 2
            [eesnimi] => Kati
            [perekonnanimi] => Tamm
            [maakonnanimi] => Saaremaa
            [maakonnakeskus] => Kuressaare
        )

    [2] => stdClass Object
        (
            [id] => 1
            [eesnimi] => Juku
            [perekonnanimi] => Tamm
            [maakonnanimi] => Saaremaa
            [maakonnakeskus] => Kuressaare
        )

)
```

Andmete väljastamine veebilehel. Lehe päises otsustatakse valikute abil, milline väljakutsetest käiku läheb. Kui suund on olemas, järelikult on olemas ka tulp - sest siinse veebilehe viidete loomise süsteem lihtsalt töötab nii.

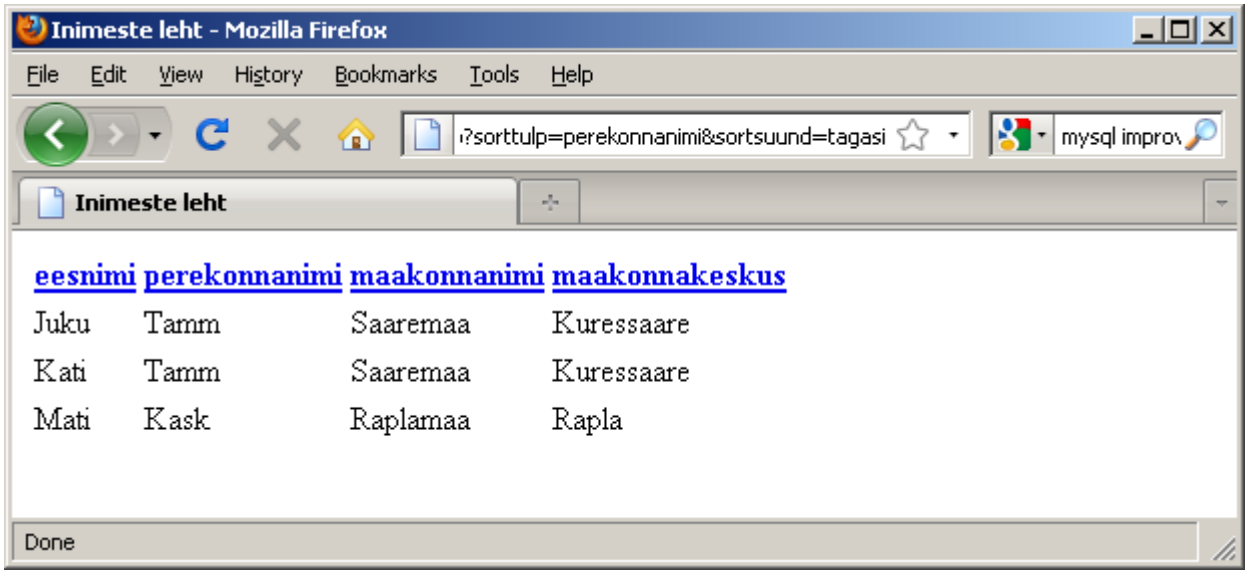
Olemasolevate tulpade loetelu asub massiivis. Tükliga käiakse läbi kõik tulbad ning uuritakse, kas eelnevalt sorditi vastava tulba järgi (muutujas $_REQUEST["sorttulp"]) selle tulba nimi. Kui jah ning samas eelnevalt ei olnud lisandit, mis määraks kahanevas suunas sortimise, siis pannakse vastav lisand külge. Nõnda sorditakse tulba nime vajutamise peale kordamööda kasvavalt ja kahanevalt. Ilma korduse ja paljalt if-ide abil kirjutades oleks selline kontroll päris pikaks läinud.

```
$tulbad=array("eesnimi", "perekonnanimi",
          "maakonnanimi", "maakonnakeskus");
foreach($tulbad as $tulp){
  $lisand="";
  if(isSet($_REQUEST["sorttulp"]) and
      $tulp==$_REQUEST["sorttulp"] and
     !isSet($_REQUEST["sortsuund"])){
      $lisand="&amp;sortsuund=tagasi";
  }
  echo "<th><a href=
   '$_SERVER[PHP_SELF]?sorttulp=$tulp$lisand'>$tulp</a></th>";
}
```

Loetelu fail tervikuna

```
<?php
  require("abifunktsioonid.php");
  if(isSet($_REQUEST["sortsuund"])){
    if(!isSet($_REQUEST["sorttulp"])){
          die("Sorteerimissuund olemas, tulp puudu.");}
    $inimesed=kysiInimesteAndmed($_REQUEST["sorttulp"], "DESC");
  } else if(isSet($_REQUEST["sorttulp"])){
    $inimesed=kysiInimesteAndmed($_REQUEST["sorttulp"]);
  } else {
    $inimesed=kysiInimesteAndmed();
```

```
  }
?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
      <title>Inimeste leht</title>
      <meta http-equiv="Content-Type" content="text/html;charset=utf-8" />
  </head>
  <body>
     <table>
       <tr>
        <?php
          $tulbad=array("eesnimi", "perekonnanimi",
                        "maakonnanimi", "maakonnakeskus");
          foreach($tulbad as $tulp){
            $lisand="";
            if(isSet($_REQUEST["sorttulp"]) and
                $tulp==$_REQUEST["sorttulp"] and
               !isSet($_REQUEST["sortsuund"])){
                $lisand="&amp;sortsuund=tagasi";
            }
            echo "<th><a href=
             '$_SERVER[PHP_SELF]?sorttulp=$tulp$lisand'>$tulp</a></th>";
          }
        ?>
       </tr>
       <?php foreach($inimesed as $isik): ?>
         <tr>
           <td><?=$isik->eesnimi ?></td>
           <td><?=$isik->perekonnanimi ?></td>
           <td><?=$isik->maakonnanimi ?></td>
           <td><?=$isik->maakonnakeskus ?></td>
         </tr>
       <?php endforeach; ?>
     </table>
  </body>
</html>
```



Otsing LIKE ? abil

Niisama vabalt LIKE-avaldise sisse küsimärgiga parameetrit panna MySQL Improved ei kannata. Küll aga saab metamärkidega teksti enne PHP abil valmis ehitada ning siis SQL-lausesse küsimärkide asemele sättida nagu järgnevas lõigus. Ka siin pole muret, et tekstisse sattunud ülakomad ja muud erisümbolid segadust tekitada võiksid. Algses konspektis pruugitud addslashes-käsu asemel soovitatakse kasutada mysql_real_escape_string. Viimane aga nõuab serveril vastava mooduli küljesolekut ning ei pruugi igal pool töötada. Selles mõttes siinne küsimärkidega asendus kõige kindlam.

```
    $kask=$yhendus->prepare("SELECT inimesed.id, eesnimi,
         perekonnanimi, maakonnanimi, maakonnakeskus
       FROM inimesed, maakonnad
       WHERE inimesed.maakonna_id=maakonnad.id
         AND (eesnimi LIKE ? OR perekonnanimi LIKE ?)
       ORDER BY $sorttulp $sortsuund");
    echo $yhendus->error;
    $otsiparam='%'.$otsisona.'%';
    $kask->bind_param("ss", $otsiparam, $otsiparam);
    $kask->bind_result($id, $eesnimi, $perekonnanimi,
      $maakonnanimi, $maakonnakeskus);
```

`abifunktsioonid.php` tervikuna

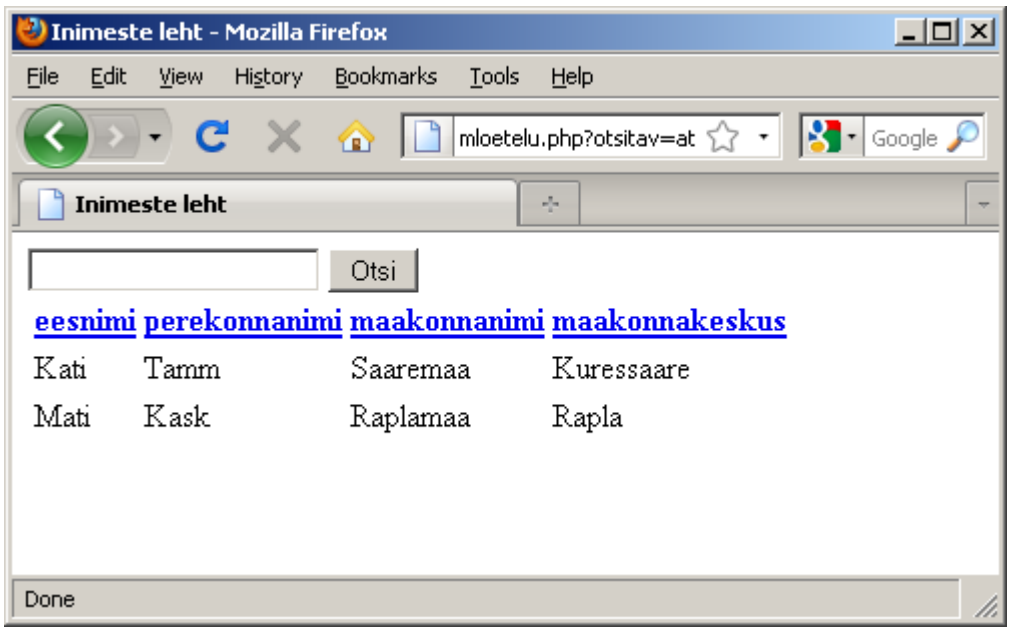
```
<?php
  $yhendus=new mysqli("localhost", "jaagup", "xxxxxx", "jaagup");

  function kysiInimesteAndmed($sorttulp="perekonnanimi",
        $sortsuund="ASC", $otsisona=""){
    global $yhendus;
    $lubatudtulbad=array("eesnimi", "perekonnanimi",
                         "maakonnanimi", "maakonnakeskus");
    if(!in_array($sorttulp, $lubatudtulbad)){
      return array();
    }
    $lubatudsuunad=array("ASC", "DESC");
    if(!in_array($sortsuund, $lubatudsuunad)){
      die("Lubamatu suund");
    }
    $kask=$yhendus->prepare("SELECT inimesed.id, eesnimi,
         perekonnanimi, maakonnanimi, maakonnakeskus
       FROM inimesed, maakonnad
       WHERE inimesed.maakonna_id=maakonnad.id
         AND (eesnimi LIKE ? OR perekonnanimi LIKE ?)
       ORDER BY $sorttulp $sortsuund");
    echo $yhendus->error;
    $otsiparam='%'.$otsisona.'%';
    $kask->bind_param("ss", $otsiparam, $otsiparam);
    $kask->bind_result($id, $eesnimi, $perekonnanimi,
      $maakonnanimi, $maakonnakeskus);
    $kask->execute();
    $hoidla=array();
    while($kask->fetch()){
      $isik=new stdClass();
      $isik->id=$id;
      $isik->eesnimi=htmlspecialchars($eesnimi);
      $isik->perekonnanimi=htmlspecialchars($perekonnanimi);
      $isik->maakonnanimi=htmlspecialchars($maakonnanimi);
      $isik->maakonnakeskus=htmlspecialchars($maakonnakeskus);
      array_push($hoidla, $isik);
    }
    return $hoidla;
  }

  //---------------
  if( array_pop(explode("/", $_SERVER["PHP_SELF"]))=="abifunktsioonid.php"):
?>
<pre>
<?php
  print_r(kysiInimesteAndmed("eesnimi", "DESC", "t"));
```

```
?>
</pre>
<?php endif ?>
```

Otsitavate andmete näitamine

Kui edasiantavaid muutujaid juba rohkem ning nende lisandumise järjekord teadmata, siis on kindlam kõikidele muutujatele panna vaikeväärtused ning need aadressirealt saabuvate andmetega üle kirjutada.

```
<?php
  require("abifunktsioonid.php");
  $sorttulp="eesnimi";
  $sortsuund="ASC";
  $otsisona="";
  if(isSet($_REQUEST["sorttulp"])) {$sorttulp=$_REQUEST["sorttulp"];}
  if(isSet($_REQUEST["sortsuund"])){$sortsuund="DESC";}
  if(isSet($_REQUEST["otsitav"])) {$otsisona=$_REQUEST["otsitav"];}
  $inimesed=kysiInimesteAndmed($sorttulp, $sortsuund, $otsisona);
?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
      <title>Inimeste leht</title>
      <meta http-equiv="Content-Type" content="text/html;charset=utf-8" />
  </head>
  <body>
     <form action="<?=$_REQUEST['PHP_SELF'] ?>">
        <input type="text" name="otsitav" />
        <input type="submit" value="Otsi" />
     </form>
     <table>
       <tr>
        <?php
          $tulbad=array("eesnimi", "perekonnanimi",
               "maakonnanimi", "maakonnakeskus");
          foreach($tulbad as $tulp){
            $lisand="";
            if(isSet($_REQUEST["sorttulp"]) and
               $tulp==$_REQUEST["sorttulp"] and
               !isSet($_REQUEST["sortsuund"])){
                $lisand="&amp;sortsuund=tagasi";
            }
            echo "<th><a href='
             $_SERVER[PHP_SELF]?sorttulp=$tulp$lisand'>$tulp</a></th>";
          }
        ?>
       </tr>
       <?php foreach($inimesed as $isik): ?>
         <tr>
           <td><?=$isik->eesnimi ?></td>
           <td><?=$isik->perekonnanimi ?></td>
           <td><?=$isik->maakonnanimi ?></td>
           <td><?=$isik->maakonnakeskus ?></td>
         </tr>
       <?php endforeach; ?>
     </table>
  </body>
</html>
```



Igasuguse tabeli näitamine

Eelnevas koodis olid tulpade nimed selgelt sisse kirjutatud. Sellisel juhul aga peab nende lisandumisel/muutumisel ka koodi sisse mitmesse kohta parandusi tegema. Järgnev näide võimaldab igasuguse SQL-lausega küsitud andmed tabelina veebilehel välja näidata. Kuna SQL-lause väljund on paratamatult tabel, siis on see võimalik. Käsklused

```
$yhendus->real_query($lause);
$vastus=$yhendus->store_result();
```

kahe peale annavad mälus oleva andmetabeli. Käsu fetch_row abil saab sealt ükshaaval ridu välja võtta ning foreach-tsükli abil kõik väljad tabeli lahtritena ekraanile kuvada.

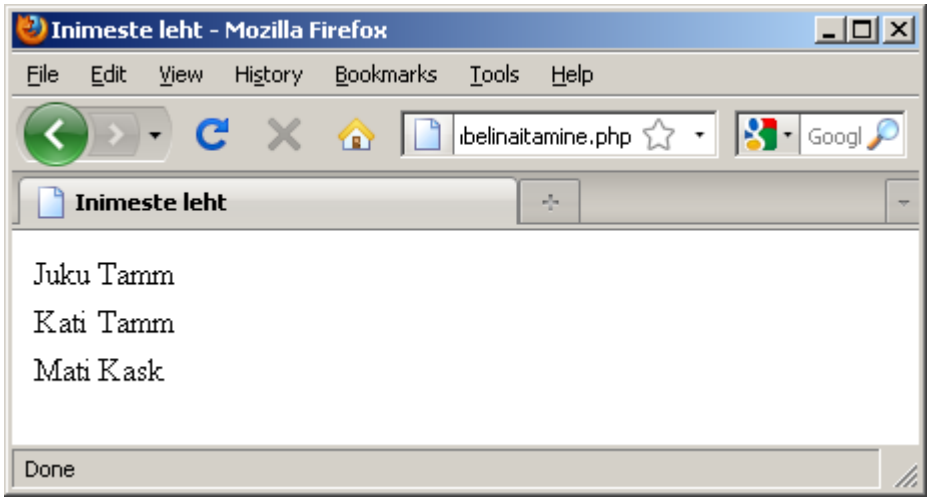
```
while($rida=$vastus->fetch_row()){
 echo "<tr>";
 foreach($rida as $tulbasisu){
   echo "<td>".htmlspecialchars($tulbasisu)."</td>";
 }
 echo "</tr>";
}
```

Fail tervikuna:

`tabelinaitamine.php`

```
<?php
  $yhendus=new mysqli("localhost", "jaagup", "xxxxxx", "jaagup");
  $lause="SELECT eesnimi, perekonnanimi FROM inimesed";
  $yhendus->real_query($lause);
  $vastus=$yhendus->store_result();
?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
```

```
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
      <title>Inimeste leht</title>
      <meta http-equiv="Content-Type"
content="text/html;charset=utf-8" />
  </head>
  <body>
     <table>
       <?php
         while($rida=$vastus->fetch_row()){
          echo "<tr>";
          foreach($rida as $tulbasisu){
            echo "<td>".htmlspecialchars($tulbasisu)."</td>";
          }
          echo "</tr>";
         }
       ?>
     </table>
  </body>
</html>
```



mysqli tulpade uuring

Soovides päringusse ka parameetreid panna, on viisakam prepare-käsuga käsklus ette valmistada ja parameetrid küsimärkide kaudu sisestada. Näite lühiduse huvides pole siin parameetrit lisatud, aga see loodetavasti eelmistest näidetest tuttav.

Kõige keerukam osa näitest tõenäoliselt on väljastusparameetrite tarbeks muutujate ette valmistamine. Üldjuhul bind_param saab omale parameetriks nõnda palju muutujaid, kui palju tulpi SELECT-lauses väljastatakse. Siin me aga ei tea kindlat tulpade arvu, sest me tabel peab hakkama saama igasuguste päringutega. Selleks siis koostatakse muutujate massiiv nõnda, et massiivi iga element saab oma andmed aadressilt, kuhu bind_param-käsu järgi välja õpetatud fetch need paneb. Et tegeldakse mäluaadressiga, seda näitab &-märk omistuse juures. Massiivi indeksiks saab tulba nimi. Massiivi lõppu omistamiseks sobib käskluse $muutujad[]=uusväärtus.

```
        while($tulbaandmed=$metaandmed->fetch_field()){
```

```
        $muutujad[] = &$rida[$tulbaandmed->name];
        }
```

bind_result käsule massiivitäie muutujate ette andmine käib järgnevalt:

```
        call_user_func_array(array($kask, 'bind_result'), $muutujad);
```

Ning kood tervikuna.

```
<?php
  $yhendus=new mysqli("localhost", "if09", "h9ETN", "if09_jaagup1");
  $lause="SELECT eesnimi, perekonnanimi FROM inimesed";
  $kask=$yhendus->prepare($lause);
  $kask->execute();
?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
      <title>Inimeste leht</title>
      <meta http-equiv="Content-Type" content="text/html;charset=utf-8" />
  </head>
  <body>
     <table>
       <?php
         $metaandmed=$kask->result_metadata();
         echo "<tr>";
         while($tulbaandmed=$metaandmed->fetch_field()){
            echo "<th>$tulbaandmed->name</th>";
            $muutujad[] = &$rida[$tulbaandmed->name];
         }
         echo "</tr>";
         call_user_func_array(array($kask, 'bind_result'), $muutujad);

         while($kask->fetch()){
          echo "<tr>";
          foreach($rida as $tulbasisu){
            echo "<td>".htmlspecialchars($tulbasisu)."</td>";
          }
          echo "</tr>";
         }
       ?>
     </table>
  </body>
</html>
```

Inimeste haldus

Seotud tabelitesse andmete lisamisel on programmeerija jaoks hea selge loogika: kõigepealt tuleb andmed panna sinna tabelisse kuhu viidatakse. Ning alles seejärel on põhjust andmeid panna tabelisse kust viidatakse. Ehk siis kõigepealt sisestatakse maakonnad ning alles pärast seda inimesed - juhul kui inimesel peab kirjas olema tema paiknemise maakonna id. Samuti tuli lisada enne kaubagrupid ning alles siis sai lisada kaupu, valides rippmenüüst lisatavale kaubale sobiv grupp.

Kasutajaliidese seisukohalt ei ei pruugi selline lähenemine aga kasutajatele loogiline ega mugav tunduda. Kategooriate või märksõnade puhul tõenäoliselt tahaks inimene valida neid olemasolevate hulgast või lisada uusi. Ning uute lisamiseks ta parema meelega ei lähe neid teisele lehele või mujale kaugele lisama - ta tahab oma toimetused ühe koha peal valmis saada. Javaskriptiga saab muidugi ka mitmesuguseid abivahendeid luua, aga siin näites teeme PHP vahenditega kasutajale ehk selle koha pealt võimalikult mugava rakenduse.

Kõigepealt muudame või lisame eelneva inimeste halduse näitega võrreldes mõned funktsioonid, et neid saaks hiljem kasutajaliidese loomise juures pruukida.

Kasutaja jaoks inimese ja maakonna üheaegsel lisamisel tuleb tehniliselt ikkagi uus maakond kõigepealt tabellisse panna ning küsida selle maakonna id. Seejärel saab lisada inimese andmed tema tabelisse ning määrata talle maakonna_id väärtuseks vastloodud maakonna id. Õnneks saab loodud tabelirea id-väärtuse kätte MySQL Improved Statementi (ehk muutuja $kask) väljalt insert_id. Selle siis nüüd funktsioonid lõpus tagastamegi.

```
function lisaMaakond($maakonnanimi, $maakonnakeskus){
   global $yhendus;
   $kask=$yhendus->prepare(
     "INSERT INTO maakonnad (maakonnanimi, maakonnakeskus)
       VALUES (?, ?)");
   $kask->bind_param("ss", $maakonnanimi, $maakonnakeskus);
   $kask->execute();
   return $kask->insert_id;
}
```

Hoolitsemaks, et sama nimega maakond ei satuks tabelisse mitu korda, on hea panna maakonnanime tulbale unikaalsuse piirang.

```
ALTER TABLE maakonnad ADD UNIQUE(maakonnanimi);
```

Inimese ja maakonna andmete üheaegseks lisamiseks tuleb uus funktsioon, mis saab enesele mõlema tabeli jaoks vajalikud parameetrid. Kõigepaelt uuritakse, kas etteantud maakonnanimega maakond on maakondade tabelis juba olemas. Kui jah, siis lihtsalt küsitakse selle maakonna id ning lisatakse inimese juurde vastava maakonna id. Kui aga pakutud nimega maakonda pole, siis see luuakse eelpooltoodud käsuga, saadakse tagasi loodud maakonna id ning antakse see kaasa inimese andmete lisamisel.

```
function lisaInimeneKoosMaakonnaga($eesnimi, $perekonnanimi,
   $maakonnanimi, $maakonnakeskus){
   global $yhendus;
   $maakonna_id=kysiYksikV22rtusBaasist(
     "SELECT id FROM maakonnad WHERE maakonnanimi=?", $maakonnanimi);
   if(!$maakonna_id){
      $maakonna_id=lisaMaakond($maakonnanimi, $maakonnakeskus);
   }
   lisaInimene($eesnimi, $perekonnanimi, $maakonna_id);
}
```

Olemasoleva maakonna nime küsimiseks ning ka muude võimalike hilisemate toimingute tarbeks lõime abifunktsiooni üksiku väärtuse küsimiseks. Selle abil saab mõneski kohas hea jupi koodi kokku hoida. Ning võimalus päringule ka üks parameeter anda, nagu vahel ette tuleb. Kui päringus kõik olemas ja parameetrit pole vaja, siis seda ei lisata. Maakonnanime järgi id otsimisel on aga igati kasulik, et veebist tulev maakonnanimi õnnestub SQL-lauses küsimärgi kohale paigutada, mitte ei pea muretsema erisümbolite võimaliku mõju üle SQL lauses.

```
function kysiYksikV22rtusBaasist($sqllause, $parameeter=false){
   global $yhendus;
   $kask=$yhendus->prepare($sqllause);
   echo $yhendus->error;
   if($parameeter){
     $kask->bind_param("s", $parameeter);
```

```
    }
    $kask->bind_result($vastus);
    $kask->execute();
    if(!$kask->fetch()){return false;}
    return $vastus;
  }
```

Rippmenüü loomise funktsioonile tuli juurde võimalus elemendi märgendi sisse täiendav lisand kirjutada - näiteks stiilikäskluse või Javaskripti tarbeks nagu siin näites hiljem kasutatakse. Et tegemist valikulise viimase parameetrina, siis võib teda rahumeeli ainult vajaduse korral kasutada ning muul juhul kõrvale jätta.

```
  function looRippMenyy($sqllause, $valikunimi,
                        $valitudid="-1", $ylalisand="")
```

Edasi fail tervikuna:

abifunktsioonid.php

```
<?php
  $yhendus=new mysqli("localhost", "jaagup", "xxxxxx", "jaagup");

  function kysiInimesteAndmed($sorttulp="perekonnanimi", $sortsuund="ASC", $otsisona=""){
    global $yhendus;
    $lubatudtulbad=array("eesnimi", "perekonnanimi", "maakonnanimi", "maakonnakeskus");
    if(!in_array($sorttulp, $lubatudtulbad)){
      return array();
    }
    $lubatudsuunad=array("ASC", "DESC");
    if(!in_array($sortsuund, $lubatudsuunad)){
      die("Lubamatu suund");
    }
    $kask=$yhendus->prepare("SELECT inimesed.id, eesnimi,
         perekonnanimi, maakonnanimi, maakonnakeskus
       FROM inimesed, maakonnad
       WHERE inimesed.maakonna_id=maakonnad.id
         AND (eesnimi LIKE ? OR perekonnanimi LIKE ?)
       ORDER BY $sorttulp $sortsuund");
    echo $yhendus->error;
    $otsiparam='%'.$otsisona.'%';
    $kask->bind_param("ss", $otsiparam, $otsiparam);
    $kask->bind_result($id, $eesnimi, $perekonnanimi,
      $maakonnanimi, $maakonnakeskus);
    $kask->execute();
    $hoidla=array();
    while($kask->fetch()){
      $isik=new stdClass();
      $isik->id=$id;
      $isik->eesnimi=htmlspecialchars($eesnimi);
      $isik->perekonnanimi=htmlspecialchars($perekonnanimi);
      $isik->maakonnanimi=htmlspecialchars($maakonnanimi);
      $isik->maakonnakeskus=htmlspecialchars($maakonnakeskus);
      array_push($hoidla, $isik);
    }
    return $hoidla;
  }

  function kysiInimeseAndmed($id){
    global $yhendus;
    $kask=$yhendus->prepare("SELECT eesnimi, perekonnanimi, maakonna_id
       FROM inimesed WHERE id=?");
    $kask->bind_param("i", $id);
    $kask->bind_result($eesnimi, $perekonnanimi, $maakonna_id);
    $kask->execute();
    echo $yhendus->error;
    if(!$kask->fetch()){return false;}
    $isik=new stdClass();
    $isik->id=$id;
```

```
  $isik->eesnimi=htmlspecialchars($eesnimi);
  $isik->perekonnanimi=htmlspecialchars($perekonnanimi);
  $isik->maakonna_id=$maakonna_id;
  return $isik;
}

function muudaInimeseAndmed($id, $eesnimi, $perekonnanimi, $maakonna_id){
  global $yhendus;
  $kask=$yhendus->prepare("UPDATE inimesed SET eesnimi=?,
    perekonnanimi=?, maakonna_id=? WHERE id=?");
  $kask->bind_param("ssii", $eesnimi, $perekonnanimi, $maakonna_id, $id);
  $kask->execute();
}

function looRippMenyy($sqllause, $valikunimi,
                      $valitudid="-1", $ylalisand=""){
   global $yhendus;
   $kask=$yhendus->prepare($sqllause);
   $kask->bind_result($id, $sisu);
   $kask->execute();
   $tulemus="<select name='$valikunimi' $ylalisand>";
   while($kask->fetch()){
     $valikulisand="";
     if($id==$valitudid){$valikulisand=" selected='selected'";}
     $tulemus.="<option value='$id' $valikulisand>$sisu</option>";
   }
   $tulemus.="</select>";
   return $tulemus;
}

function lisaMaakond($maakonnanimi, $maakonnakeskus){
   global $yhendus;
   $kask=$yhendus->prepare(
     "INSERT INTO maakonnad (maakonnanimi, maakonnakeskus)
       VALUES (?, ?)");
   $kask->bind_param("ss", $maakonnanimi, $maakonnakeskus);
   $kask->execute();
   return $kask->insert_id;
   // ALTER TABLE maakonnad ADD UNIQUE(maakonnanimi);
}

function lisaInimene($eesnimi, $perekonnanimi, $maakonna_id){
  global $yhendus;
  $kask=$yhendus->prepare(
    "INSERT INTO inimesed (eesnimi, perekonnanimi, maakonna_id)
      VALUES (?, ?, ?)");
  $kask->bind_param("ssi", $eesnimi, $perekonnanimi, $maakonna_id);
  $kask->execute();
}

function lisaInimeneKoosMaakonnaga($eesnimi, $perekonnanimi,
   $maakonnanimi, $maakonnakeskus){
   global $yhendus;
   $maakonna_id=kysiYksikV22rtusBaasist(
     "SELECT id FROM maakonnad WHERE maakonnanimi=?", $maakonnanimi);
   if(!$maakonna_id){
      $maakonna_id=lisaMaakond($maakonnanimi, $maakonnakeskus);
   }
   lisaInimene($eesnimi, $perekonnanimi, $maakonna_id);
}

function kustutaInimene($id){
  global $yhendus;
  $kask=$yhendus->prepare("DELETE FROM inimesed WHERE id=?");
  $kask->bind_param("i", $id);
  $kask->execute();
}

function kysiYksikV22rtusBaasist($sqllause, $parameeter=false){
   global $yhendus;
   $kask=$yhendus->prepare($sqllause);
```

```
    echo $yhendus->error;
    if($parameeter){
      $kask->bind_param("s", $parameeter);
    }
    $kask->bind_result($vastus);
    $kask->execute();
    if(!$kask->fetch()){return false;}
    return $vastus;
  }

  //---------------
  if( array_pop(explode("/", $_SERVER["PHP_SELF"]))=="abifunktsioonid.php"):
?>
<pre>
<?php
//  print_r(kysiInimesteAndmed("eesnimi", "DESC", "t"));
//   print lisaMaakond('Hiiumaa', 'K2rdla');
//    lisaInimeneKoosMaakonnaga("Kati", "Kask", "Tartu maakond", "Tartu")
?>
</pre>
<?php endif ?>
```

Pärast vajaminevate funktsioonide loomist saab asuda rahus kasutajaliidest looma. Eraldi tähelepanu väärib rippmenüü loomine. Rippmenüü loomise funktsioon soovib saada SQL-päringu, mille tulemusena küsitakse baasist kahetulbaline tabel. Kui aga olemasolevat maakonda pole loetelus, siis peaks olema mugav moodus uue lisamiseks. Et funktsioonile on praegusel kujul võimalik andmeid ette anda vaid SQL-kujul, siis siitkaudu uue rea lisamegi. Olematu maakonna koodiga 0 teate "Lisa uus". Sellise rea annab `SELECT '0', 'Lisa uus'`

```
mysql> SELECT '0', 'Lisa uus';
+---+----------+
| 0 | Lisa uus |
+---+----------+
| 0 | Lisa uus |
+---+----------+
```

Käsuga UNION ALL saab olemasoleva päringu vastuste lõppu lisada teise päringu vastused.

```
mysql> SELECT id, maakonnanimi FROM maakonnad
    ->  UNION ALL (SELECT '0', 'Lisa uus');
+----+------------------+
| id | maakonnanimi     |
+----+------------------+
| 1  | Harjumaa         |
| 2  | Raplamaa         |
| 3  | Saaremaa         |
| 4  | Valgamaa         |
| 8  | Hiiumaa          |
| 9  | Viljandi maakond |
| 10 | Tartu maakond    |
| 11 | Põlva maakond    |
| 12 | Ida-Virumaa      |
| 0  | Lisa uus         |
+----+------------------+
```

Nõnda saabki kavala lause abil SQLi vastuste tulemusena loodud valikusse andmed sisse anda.

Maakonna sisestamise lahtrid maakonnanimi ja maakonnakeskus said koos nende sisu selgitavate siltidega paigutatud omaette tabelisse id-ga maakonnasisestus. Neid sisestusvälju on põhjust näha vaid juhul, kui kasutaja on valinud uue maakonna lisamise. Tabeli peitmist ja näitamist korraldab Javaskripti funktsioon kontrolliValik, millele antakse kaasa väljakutsuva rippmenüü muutuja (this).

```
          <dt>Maakond:</dt>
          <dd>
            <?php echo looRippmenyy("SELECT id, maakonnanimi FROM maakonnad
```

```
              UNION ALL (SELECT '0', 'Lisa uus')",
                "maakonna_id", "-1", " onchange='kontrolliValik(this)'") ?><br />
          <table id="maakonnasisestus" style="visibility: hidden">
            <tr><td><input type="text" name="maakonnanimi" />
                </td><td><input type="text" name="maakonnakeskus" />
                </td></tr>
            <tr><td>maakonnanimi</td><td>maakonnakeskus</td></tr>
          </table>
        </dd>
```

Ülal skriptiosas oleva funktsiooni abil näidatakse tabelit vaid juhul, kui ees on rippmenüüst viimane valik. Omadus selectedIndex näitab valitud rea järjekorranumbrit, rippmenyy.options.length aga valikuridade arvu. Kuna lugemine algab nullist, siis viimane rida on valitud parajasti siis, kui selectedIndex on elementide arvust ühe võrra väiksem.

```
    <script type="text/javascript">
       function kontrolliValik(rippmenyy){
         var tabel=document.getElementById("maakonnasisestus");
         if(rippmenyy.selectedIndex==rippmenyy.options.length-1){
            tabel.style.visibility="visible";
         } else {
            tabel.style.visibility="hidden";
         }
       }
    </script>
```

Ning koodifail tervikuna.

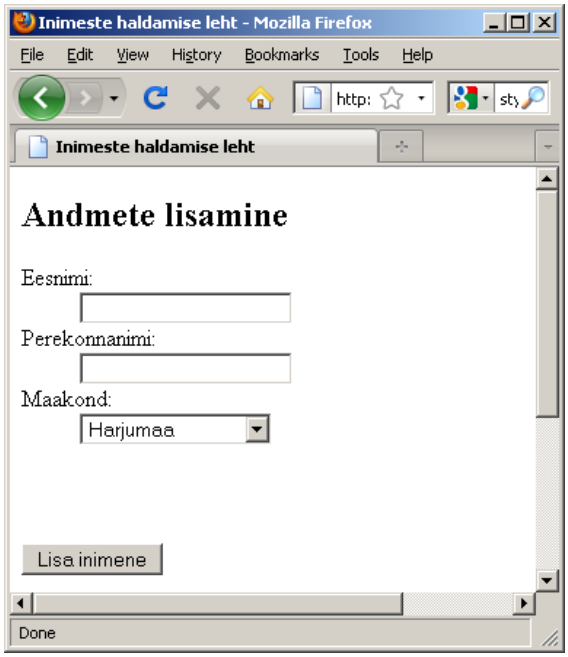
inimhaldus.php

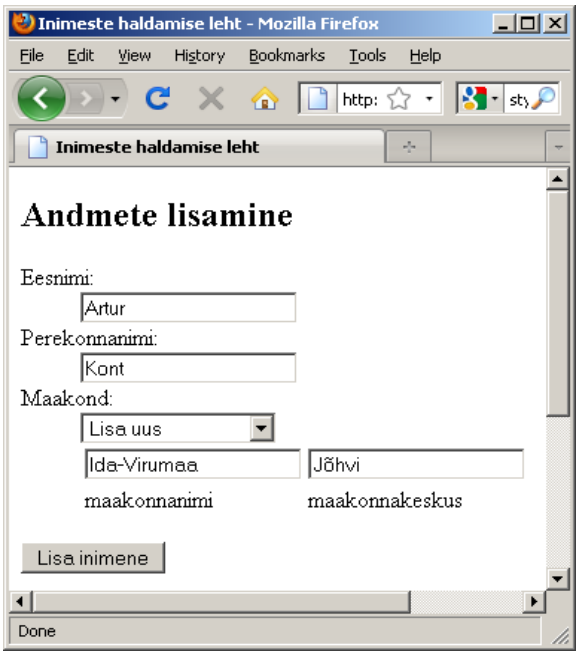
```
<?php
  require("abifunktsioonid.php");
  if(isSet($_REQUEST["inimlisamine"])){
    if($_REQUEST["maakonna_id"]=="0"){
       lisaInimeneKoosMaakonnaga($_REQUEST["eesnimi"], $_REQUEST["perekonnanimi"],
         $_REQUEST["maakonnanimi"], $_REQUEST["maakonnakeskus"]);
    } else {
      lisaInimene($_REQUEST["eesnimi"], $_REQUEST["perekonnanimi"],
        $_REQUEST["maakonna_id"]);
    }
  }
  if(isSet($_REQUEST["muudetava_inimese_id"])){
    muudaInimeseAndmed($_REQUEST["muudetava_inimese_id"],
      $_REQUEST["eesnimi"], $_REQUEST["perekonnanimi"], $_REQUEST["maakonna_id"]);
  }
  if(isSet($_REQUEST["kustutusid"])){
    kustutaInimene($_REQUEST["kustutusid"]);
  }
  $sorttulp="eesnimi";
  $sortsuund="ASC";
  $otsisona="";
  if(isSet($_REQUEST["sorttulp"])) {$sorttulp=$_REQUEST["sorttulp"];}
  if(isSet($_REQUEST["sortsuund"])){$sortsuund="DESC";}
  if(isSet($_REQUEST["otsitav"])) {$otsisona=$_REQUEST["otsitav"];}
  $inimesed=kysiInimesteAndmed($sorttulp, $sortsuund, $otsisona);
?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
      <title>Inimeste haldamise leht</title>
      <meta http-equiv="Content-Type" content="text/html;charset=utf-8" />
      <script type="text/javascript">
         function kontrolliValik(rippmenyy){
           var tabel=document.getElementById("maakonnasisestus");
           if(rippmenyy.selectedIndex==rippmenyy.options.length-1){
```

```
            tabel.style.visibility="visible";
          } else {
            tabel.style.visibility="hidden";
          }
        }
      </script>
  </head>
  <body>
     <form action="<?=$_REQUEST['PHP_SELF'] ?>" method='post'>
      <h2>Andmete lisamine</h2>
        <dl>
          <dt>Eesnimi:</dt>
          <dd><input type="text" name="eesnimi" /></dd>
          <dt>Perekonnanimi:</dt>
          <dd><input type="text" name="perekonnanimi" /></dd>
          <dt>Maakond:</dt>
          <dd>
            <?php echo looRippmenyy("SELECT id, maakonnanimi FROM maakonnad
                UNION ALL (SELECT '0', 'Lisa uus')",
                  "maakonna_id", "-1", " onchange='kontrolliValik(this)'") ?><br />
            <table id="maakonnasisestus" style="visibility: hidden">
              <tr><td><input type="text" name="maakonnanimi" />
                  </td><td><input type="text" name="maakonnakeskus" />
                  </td></tr>
              <tr><td>maakonnanimi</td><td>maakonnakeskus</td></tr>
            </table>
          </dd>
        </dl>
        <input type="submit" name="inimlisamine" value="Lisa inimene" />
     </form>
     <?php
       if(isSet($_REQUEST["muutmisid"])){
         $isik=kysiInimeseAndmed($_REQUEST["muutmisid"]);
         if($isik){
          echo " <h2>Andmete muutmine</h2>
            <form action='$_SERVER[PHP_SELF]' method='post'>
              <input type='hidden' name='muudetava_inimese_id'
                value='$isik->id' />
              <dl>
                <dt>Eesnimi:</dt>
                <dd><input type='text' name='eesnimi' value='$isik->eesnimi' /><dd>
                <dt>Perekonnanimi</dt>
                <dd><input type='text' name='perekonnanimi' value='$isik->perekonnanimi'
/><dd>
                <dt>Maakond</dt>
                <dd>".
                  looRippmenyy("SELECT id, maakonnanimi FROM maakonnad",
                    "maakonna_id", $isik->maakonna_id)
                ."<dd>
              </dl>
              <input type='submit' name='muutmine' value='Muuda andmed' />
            </form>
          ";
         }
       }
     ?>

<br />
     <form action="<?=$_REQUEST['PHP_SELF'] ?>">
        <input type="text" name="otsitav" />
        <input type="submit" value="Otsi" />
     </form>
     <table>
       <tr>
        <th>haldus</th>
        <?php
          $tulbad=array("eesnimi", "perekonnanimi", "maakonnanimi", "maakonnakeskus");
          foreach($tulbad as $tulp){
            $lisand="";
            if(isSet($_REQUEST["sorttulp"]) and $tulp==$_REQUEST["sorttulp"] and
               !isSet($_REQUEST["sortsuund"])){
```

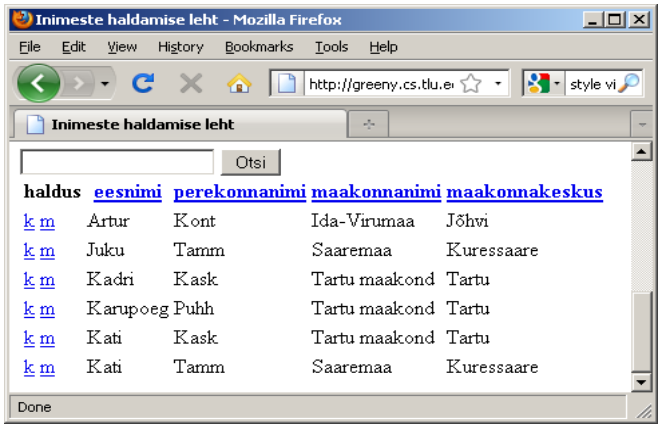
```
                $lisand="&amp;sortsuund=tagasi";
            }
            echo "<th><a href='$_SERVER[PHP_SELF]?sorttulp=$tulp$lisand'>$tulp</a></th>";
          }
        ?>
       </tr>
       <?php
        foreach($inimesed as $isik){
         echo"<tr>
           <td>
             <a href='$_SERVER[PHP_SELF]?kustutusid=$isik->id'>k</a>
             <a href='$_SERVER[PHP_SELF]?muutmisid=$isik->id'>m</a>
           </td>
           <td>$isik->eesnimi</td>
           <td>$isik->perekonnanimi</td>
           <td>$isik->maakonnanimi</td>
           <td>$isik->maakonnakeskus</td>
         </tr>";
        }
      ?>
     </table>
  </body>
</html>
```

Ja mõned vaated valminud rakendusele.





Ja Artur ongi lisatud koos maakonnaga.



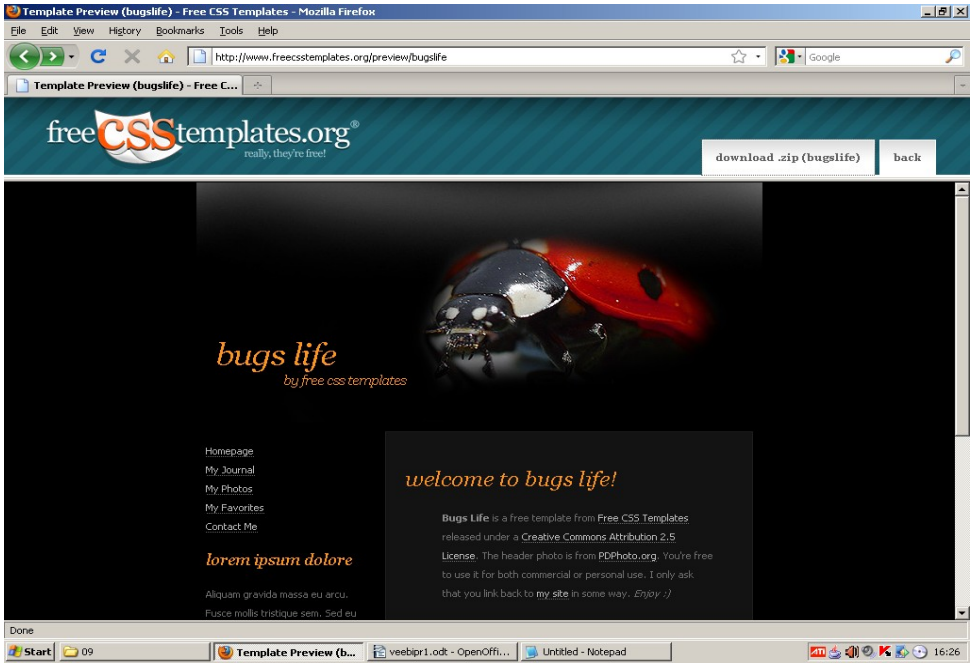
Uudiste leht

Võõra kujunduse kohandamine

Järgnevas näites võetakse aluseks eelnevalt valmis ja vabalt saadaval olev kujundus ning kohandatakse sinna külge rakendus, kus saab uudiseid lisada ja iga uudist eraldi kommenteerida. Selline ühendamine sarnaneb täiesti "tavaliste" rakenduste loomisega, kus kujundaja ülesandeks ongi ilus leht valmis teha ning programmeerija tööks siis sinna sisse õigetele kohtadele vajalikud andmed paigutada.

Mõnigase otsimise peale leidsime, et kujundusaluseks sobiks lepatriinuga tumedal taustal kujundus.

http://www.freecsstemplates.org/preview/bugslife



Veebilehelt alla tõmmatud zip-failist tekkis lahti pakkimisel hulk faile ja katalooge. Lähemal uurimisel selgus siiski, et põhipildi ette kuvamiseks on tarvis üht HTML-faili, tema juurde kuuluvat CSS-i ning üht taustapilti. Muud failid osutusid abiõpetusteks või lihtsalt vanadest versioonidest sisse ununenud piltideks. Nii et kui asuda oma rakendust mõne muu süsteemiga ühendama, on sellega kasulik eelnevalt tutvuda ning vaadata, mida tegelikult vaja läheb.

Nagu pildi pealt näha, on lehe ülaosas pilt koos sinna juurde kuuluva pealkirjatekstiga. Vasakul pool menüü mitmesuguste viidetega ning paremal pool teatekast(id) sisu jaoks. Koodist selgub, et vasaku ploki moodustab kiht id-ga colOne ning parempoolse kiht id-ga colTwo. Ning teine plokk saadakse paremale hõljuma käsuga

```
#colTwo {
        float: right;
        width: 390px;
}
```

Nad mõlemad paiknevad kihis id-ga content, millel on kindel laius määratud. Üheks võimaluseks võõrast koodi "oma kontrolli" alla saada on asuda plokke oma keelde või oma süsteemi järgi ümber nimetama - siis näha, milliste asjadega on eraldi tegeletud. Siin näiteks nimetasime kõigepealt kujundusfaili kujundus.css-iks.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<!--
Design by Free CSS Templates
http://www.freecsstemplates.org
Released for free under a Creative Commons Attribution 2.5 License
-->
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="content-type" content="text/html; charset=utf-8" />
<title>Bugs Life by Free Css Templates</title>
<meta name="keywords" content="" />
<meta name="description" content="" />
<link href="kujundus.css" rel="stylesheet" type="text/css" />
</head>
<body>
<div id="header">
	<h1><a href="#">bugs life </a></h1>
	<h2><a href="http://freecsstemplates.org/">by Free Css Templates </a></h2>
</div>
<div id="content">
	<div id="colOne">
		<div id="menu">
			<ul>
				<li><a href="#" accesskey="1" title="">Homepage</a></li>
				<li><a href="#" accesskey="2" title="">My Journal </a></li>
				<li><a href="#" accesskey="3" title="">My Photos </a></li>
				<li><a href="#" accesskey="4" title="">My Favorites </a></li>
				<li><a href="#" accesskey="5" title="">Contact Me</a></li>
			</ul>
			<h3>Lorem Ipsum Dolore </h3>
			<p>Aliquam gravida massa eu arcu. Fusce mollis tristique sem. Sed eu
eros imperdiet eros interdum blandit. Vivamus sagittis bibendum erat. Curabitur malesuada
turpis nec ante. Suspendisse quis felis.</p>
			<h3>Suspendisse Potenti</h3>
			<p>Sed vestibulum blandit nisl. Quisque elementum convallis purus.
Quisque pellentesque semper massa:</p>
			<ul>
				<li><a href="#">Suspendisse quis gravida</a><a
href="#">&#8230;</a></li>
				<li><a href="#">Vivamus sagittis bibendum</a><a
href="#">&#8230;</a></li>
				<li><a href="#">Nullam et orci in erat&#8230;</a></li>
			</ul>
		</div>
	</div>
	<div id="colTwo">
		<div class="post">
			<h2>Welcome to Bugs life!</h2>
			<p><strong>Bugs Life </strong> is a free template from <a
href="http://freecsstemplates.org/">Free CSS Templates</a> released under a <a
href="http://creativecommons.org/licenses/by/2.5/">Creative Commons Attribution 2.5
License</a>. The header photo  is from <a href="http://pdphoto.org/">PDPhoto.org</a>.
You're free to use it for both commercial or personal use. I only ask that you link back
to <a href="http://freecsstemplates.org/">my site</a> in some way. <em>Enjoy :)</em></p>
		</div>
		<div class="post">
			<h3>Suspendisse Potenti</h3>
			<p>Sed vestibulum blandit nisl. Quisque elementum convallis purus.
Quisque pellentesque semper massa:</p>
			<ul>
				<li><a href="#">Suspendisse quis gravida massa felis.</a></li>
```

```
                        <li><a href="#">Vivamus sagittis bibendum erat.</a></li>
                        <li><a href="#">Nullam et orci in erat viverra
ornare.</a></li>
                        <li><a href="#">Suspendisse quis gravida massa felis.</a></li>
                        <li><a href="#">Curabitur malesuada turpis nec ante.</a></li>
                    </ul>
            </div>
            </div>
</div>
<div id="footer">
      <p>Copyright &copy; 2006 Below the Horizon. Designed by <a
href="http://freecsstemplates.org"><strong>Free CSS Templates</strong></a></p>
</div>
</body>
</html>
```

kujundus.css

```
/*
Design by Free CSS Templates
http://www.freecsstemplates.org
Released for free under a Creative Commons Attribution 2.5 License
*/

body {
      margin: 0;
      padding: 0;
      background: #000000;
      font-family: Tahoma, Arial, Helvetica, sans-serif;
      font-size: 11px;
      color: #707070;
}

h1, h2, h3 {
      text-transform: lowercase;
      font-family: Georgia, "Times New Roman", Times, serif;
      font-weight: normal;
      font-style: italic;
      color: #FF941D;
}

h2 { font-size: 24px; }
h3 { font-size: 18px; }

p, ol, ul, blockquote {
      line-height: 22px;
}

a {
      border-bottom: 1px dotted #707070;
      text-decoration: none;
      color: #BFBFBF;
}

a:hover {
      border: none;
}

strong {
      color: #8D8C8C;
}

/* Posts */

.post {
      margin-bottom: 9px;
      padding: 20px 40px 20px 60px;
      background: #131313;
      border-top: 1px solid #1F1F1F;
```

```
	border-right: 1px solid #1F1F1F;
	border-bottom: 1px solid #1F1F1F;
	border-left: 1px solid #1F1F1F;
}

.post h2, .post h3 {
	margin-left: -40px;
}

/* Header */

#header {
	width: 600px;
	height: 280px;
	margin: 0 auto;
	background: url(lepatriinu.jpg);
}

#header h1 {
	margin: 0;
	padding: 180px 0 0 20px;
	font-size: 36px;
}

#header h2 {
	margin: 0;
	padding: 0 0 0 92px;
	font-size: 16px;
}

#header a {
	border: none;
	letter-spacing: -1px;
	color: #FF941D;
}

/* Content */

#content {
	width: 580px;
	margin: 0 auto;
	padding: 10px;
}

#colOne {
	float: left;
	width: 180px;
}

#colOne ul {
	margin-left: 0;
	padding-left: 0;
	list-style: none;
}

#colTwo {
	float: right;
	width: 390px;
}

/* Footer */

#footer {
	clear: both;
	width: 580px;
	margin: 0 auto;
	padding-top: 20px;
}

#footer p {
	margin: 0;
```

```
	text-align: center;
	font-size: 9px;
}

#footer * {
	color: #333333;
}
```

Koodi puhastus

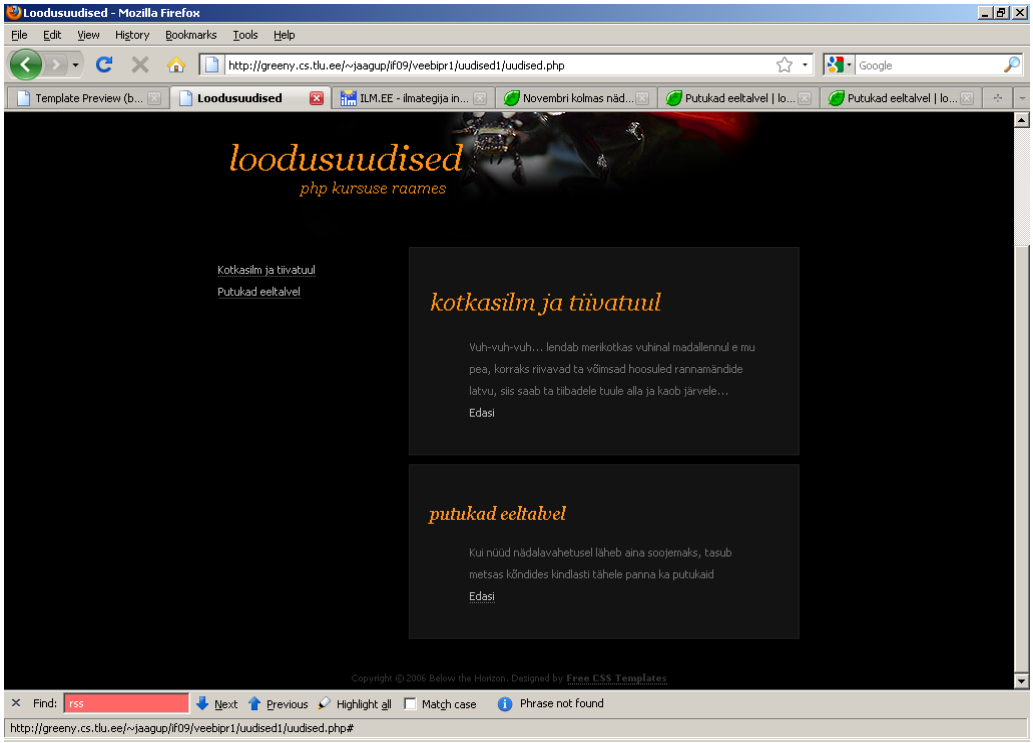
Järgmisena tasub võõrast koodist võimalikult palju lõike eemaldada - jätta alles vaid need, mis on omapoolse rakenduse toimimise juures vajalikud. Pole isegi liigset eemaldamist karta - sest kui ka midagi sai liialt välja võetud, siis saab ju alati algsest originaalist sobiva lõigu uuesti sisse võtta.

Kuna lihtsa uudistelehestiku puhul piisab vasakule vaid uudiseviidetest, siis saigi sealt colOne seest kõik muu peale paari viite välja võetud. Pealkirja sisse omale sobivad teemad. Ning on näha, et stiililehelt määratud text-transform:lowercase teeb kõik sisestatu väiketähtedeks. Teise veergu jääb alles ka vaid üks kast ühe näituudisealusega. Jalus las esiotsa püsib - viisakas on näidata, kust kujundus võetud.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<!--
Design by Free CSS Templates
http://www.freecsstemplates.org
Released for free under a Creative Commons Attribution 2.5 License
-->
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="content-type" content="text/html; charset=utf-8" />
<title>Loodusuudised</title>
<link href="kujundus.css" rel="stylesheet" type="text/css" />
</head>
<body>
<div id="header">
	<h1>Loodusuudised</h1>
	<h2>PHP kursuse raames</h2>
</div>
<div id="content">
	<div id="colOne">
		<div id="menu">
			<ul>
				<li><a href="#" >Kotkasilm ja tiivatuul</a></li>
				<li><a href="#" >Putukad eeltalvel </a></li>
			</ul>
		</div>
	</div>
	<div id="colTwo">
		<div class="post">
			<h2>kotkasilm ja tiivatuul</h2>
			<p>
			Vuh-vuh-vuh... lendab merikotkas vuhinal madallennul e mu pea,
			korraks riivavad ta võimsad hoosuled rannamändide latvu,
			siis saab ta tiibadele tuule alla ja kaob järvele...
			 <br />
			 <a href="#">Edasi</a>
			</p>
		</div>
		<div class="post">
			<h3>Putukad eeltalvel</h3>
			<p>
			Kui nüüd nädalavahetusel läheb aina soojemaks,
			tasub metsas kõndides kindlasti tähele panna ka putukaid
			 <br />
			 <a href="#">Edasi</a>
```

```
                    </p>
                </div>
                </div>
</div>
<div id="footer">
       <p>Copyright &copy; 2006 Below the Horizon. Designed by <a
href="http://freecsstemplates.org"><strong>Free CSS Templates</strong></a></p>
</div>
</body>
</html>
```

Nõnda siis juba tunduvalt lühema ning omapoolsema koodiga leht, mille võib jätta programmipoolset andmete lisamist ootama.



Andmebaasiliides

Veebirakenduse juures tasub järgmisena luua andmetabelid. Kui soovime, et igat uudist saaks eraldi kommenteerida, siis kulub kaks tabelit: üks uudiste tarbeks, teine kommentaaride jaoks. Ning nõnda, et iga kommentaari juures on kirjas kommenteeritava uudise id. Alustuseks aga uudised valmis ja katsetusse.

```
CREATE TABLE uudised(
   id INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
   pealkiri VARCHAR(100),
   sisu TEXT
);

INSERT INTO uudised(pealkiri, sisu)
VALUES('Esimene uudis',
'Esimese uudise tutvustav lause. Pikem jutt edasi');
```

Uudistega seotud toimingud koondame omaette klassi - see aitab rakendust omaette seisvate tükkidena püsida. Klassi sisse käsklus uudiste tutvustuste küsimiseks. Seal antakse välja iga uudise

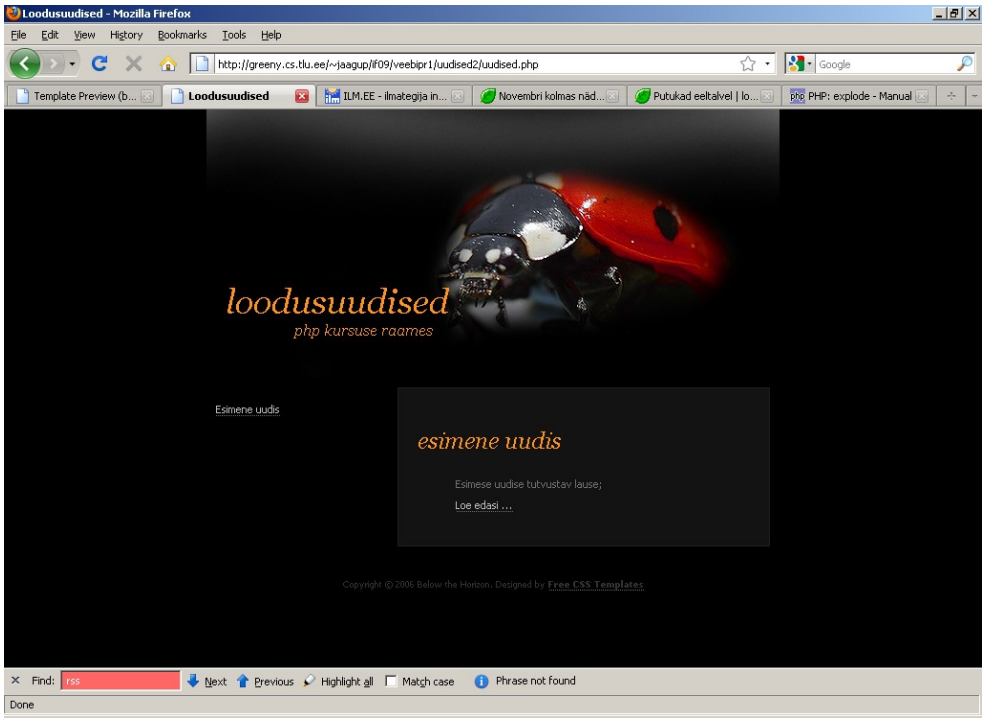
id, pealkiri, sisu ning samuti vaid esimene lause, mida hea avalehele tutvustuseks panna. Praegusel juhul eeldatakse, et lause lõpetab punkt. Ning käsklus explode tükeldab lause punkti koha pealt massiivielementideks ning neist antakse välja omakorda esimene ehk m[0].

```
abifunktsioonid.php
<?php
  $yhendus=new mysqli("localhost", "jaagup", "xxxxxx", "jaagup");

  class Uudised{
    private $ab;
    function __construct($yhendus){
      $this->ab=$yhendus;
    }
    function kysiTutvustused($algus=0, $kogus=3){
       $kask=$this->ab->prepare("SELECT id, pealkiri, sisu FROM uudised");
       $kask->bind_result($id, $pealkiri, $sisu);
       $kask->execute();
       $hoidla=array();
       while($kask->fetch()){
          $uudis=new stdClass();
          $uudis->id=$id;
          $uudis->pealkiri=htmlspecialchars($pealkiri);
          $uudis->sisu=htmlspecialchars($sisu);
          $m=explode(".",$sisu);
          $uudis->algus=htmlspecialchars($m[0]);
          array_push($hoidla, $uudis);
       }
       return $hoidla;
    }
  }

  $uudised=new Uudised($yhendus);
?>
```

Lehel pole enam muud muret, kui et menüüsse tsükliga kirjutada uudiste pealkirjad. Igalt pealkirjalt tuleb viide samale PHP-failile, mille avamisel antakse kaasa uudise id. Samasugune loetelu sai esiotsa pandud ka sisuploki kastidesse. Ainult selle vahega, et seal kuvatakse lisaks pealkirjale veel uudise esimene lause. Hiljem võib lisada piirangu, et mitme (viimase) uudise andmeid kus näidata.

```
<?php
  require("abifunktsioonid.php");
  $loetelu=$uudised->kysiTutvustused();
?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="content-type" content="text/html; charset=utf-8" />
<title>Loodusuudised</title>
<link href="kujundus.css" rel="stylesheet" type="text/css" />
</head>
<body>
<div id="header">
       <h1>Loodusuudised</h1>
       <h2>PHP kursuse raames</h2>
</div>
<div id="content">
       <div id="colOne">
              <div id="menu">
                     <ul>
                            <?php
                              foreach($loetelu as $rida){
                                echo "<li><a href=
        '$_SERVER[PHP_SELF]?uid=$rida->id'>$rida->pealkiri</a></li>";
                              }
                            ?>
                     </ul>
```

```
            </div>
        </div>
        <div id="colTwo">
          <?php
            foreach($loetelu as $rida){
             echo "<div class='post'>
                  <h2>$rida->pealkiri</h2>
                  <p>
                    $rida->algus;
                    <br />
                    <a href='$_SERVER[PHP_SELF]?uid=$rida->id'>Loe edasi ...</a>
                  </p>
                 </div>
              ";
            }
          ?>
              </div>
</div>
<div id="footer">
       <p>Copyright &copy; 2006 Below the Horizon. Designed by <a
href="http://freecsstemplates.org"><strong>Free CSS Templates</strong></a></p>
</div>
</body>
</html>
```



Edasi loome kommentaaride tabeli. Tulp "korras" näitab, et kas seda sobib uudise juures kuvada.

```
CREATE TABLE kommentaarid(
  id INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
  kommenteerija VARCHAR(50),
  kommentaarisisu VARCHAR(50),
  uudise_id INT,
  korras INT DEFAULT 1
);
```

Abifunktsioonidesse tuleb juurde kommentaaride klass. Kusjuures siin tehti nõnda, et kommentaaride klassi eksemplaril on alati enese küljes olemas uudise id, millega vastavad

kommentaarid seotud - nii on pärast vähem andmete liigutamist. Kommentaaride lisamisel saadakse uudise id siis juba objekti muutujast. Kommentaaride küsimisel pole aga vaja midagi ette anda - kui juba vahendusobjekt olemas, siis saab sealt seest ka kommentaarid kätte.

Väljaspoole klassi sai loodud funktsioon tootleSisend. Muidu kipub avatava veebifaili päises tekkivaid programmeerimistoiminguid hulgem tulema ning kujunduse ja koodi erisusest jääb vähe järele. Kui aga toimetused eraldi failis olevasse funktsiooni tuua, siis jääb näidatav fail puhtamaks. Kui on teada kuvatava uudise id, siis luuakse selle tarbeks ka kommentaariobjekt. Ning kui parajasti lisati kommentaar, siis saab loodud objekt ka selle salvestamisega hakkama.

Administraatori jaoks on uudisteklassis juures käsklus uudiste lisamise jaoks.

abifunktsioonid.php

```
<?php
  $yhendus=new mysqli("localhost", "jaagup", "xxxxxx", "jaagup");

  class Uudised{
    private $ab;
    function __construct($yhendus){
      $this->ab=$yhendus;
    }
    function kysiTutvustused($algus=0, $kogus=3){
       $kask=$this->ab->prepare("SELECT id, pealkiri, sisu FROM uudised
         ORDER BY id DESC");
       $kask->bind_result($id, $pealkiri, $sisu);
       $kask->execute();
       $hoidla=array();
       while($kask->fetch()){
          $uudis=new stdClass();
          $uudis->id=$id;
          $uudis->pealkiri=htmlspecialchars($pealkiri);
          $uudis->sisu=htmlspecialchars($sisu);
          $m=explode(".",$sisu);
          $uudis->algus=htmlspecialchars($m[0]);
          array_push($hoidla, $uudis);
       }
       return $hoidla;
    }


    function kysiUudis($id){
      $kask=$this->ab->prepare("SELECT pealkiri, sisu FROM uudised WHERE id=?");
      $kask->bind_param("i", $id);
      $kask->bind_result($pealkiri, $sisu);
      $kask->execute();
      if(!$kask->fetch()){
       die("Vigane uudise ID");
      }
      $uudis=new stdClass();
      $uudis->id=$id;
      $uudis->pealkiri=htmlspecialchars($pealkiri);
      $uudis->sisu=htmlspecialchars($sisu);
      return $uudis;
    }

    function lisaUudis($pealkiri, $sisu){
      $kask=$this->ab->prepare("INSERT INTO uudised (pealkiri, sisu) VALUES (?, ?)");
      $kask->bind_param("ss", $pealkiri, $sisu);
      $kask->execute();
    }
  }


  class Kommentaarid{
    private $ab, $uudise_id;
    function __construct($yhendus, $uid){
```

```
        global $uudised;
        $this->ab=$yhendus;
        $this->uudise_id=$uid;
        $uudised->kysiUudis($uid);
    }

    function lisaKommentaar($kommenteerija, $kommentaarisisu){
        $kask=$this->ab->prepare("INSERT INTO kommentaarid
          (kommenteerija, kommentaarisisu, uudise_id) VALUES (?, ?, ?)");
        $kask->bind_param("ssi", $kommenteerija, $kommentaarisisu, $this->uudise_id);
        $kask->execute();
    }

    function kysiKommentaarid(){
      $kask=$this->ab->prepare("SELECT id, kommenteerija, kommentaarisisu
         FROM kommentaarid WHERE uudise_id=? AND korras=1 ORDER BY id DESC");
      $kask->bind_param("i", $this->uudise_id);
      $kask->bind_result($id, $kommenteerija, $kommentaarisisu);
      $kask->execute();
      $hoidla=array();
      while($kask->fetch()){
        $kommentaar=new stdClass();
        $kommentaar->id=$id;
        $kommentaar->kommenteerija=htmlspecialchars($kommenteerija);
        $kommentaar->kommentaarisisu=htmlspecialchars($kommentaarisisu);
        array_push($hoidla, $kommentaar);
      }
      return $hoidla;
    }
  }

  function tootleSisend(){
    global $kommentaariobj;
    global $yhendus;
    if(isSet($_REQUEST["uid"])){
      $kommentaariobj=new Kommentaarid($yhendus, $_REQUEST["uid"]);
    } else {
      return;
    }
    if(isSet($_REQUEST["kommentaarisisestus"])){
      $kommentaariobj->lisaKommentaar($_REQUEST["kommenteerija"],
$_REQUEST["kommentaarisisu"]);
      header("Location: $_SERVER[PHP_SELF]?uid=$_REQUEST[uid]");
      exit();
    }
  }

  $uudised=new Uudised($yhendus);
?>
```

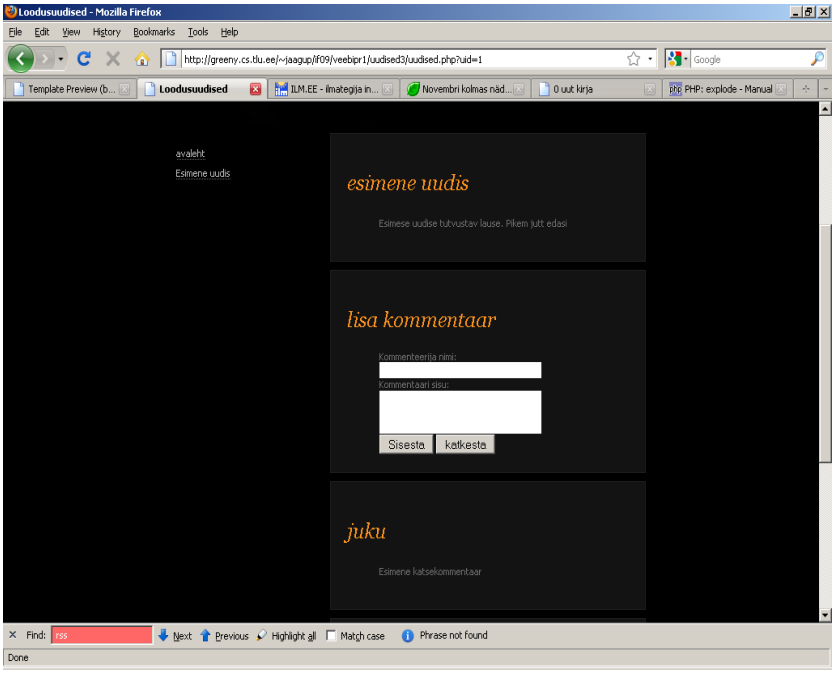
Uudiste lehel näidatakse avamisel vasakmenüüs uudiste pealkirju ning keskel lühitutvustusi. Ühe uudise valimisel avaneb üksikute tutvustuste asemel lehe peal see uudis ning alla lisandub kommenteerimisvorm ja olemasolevad kommentaarid.

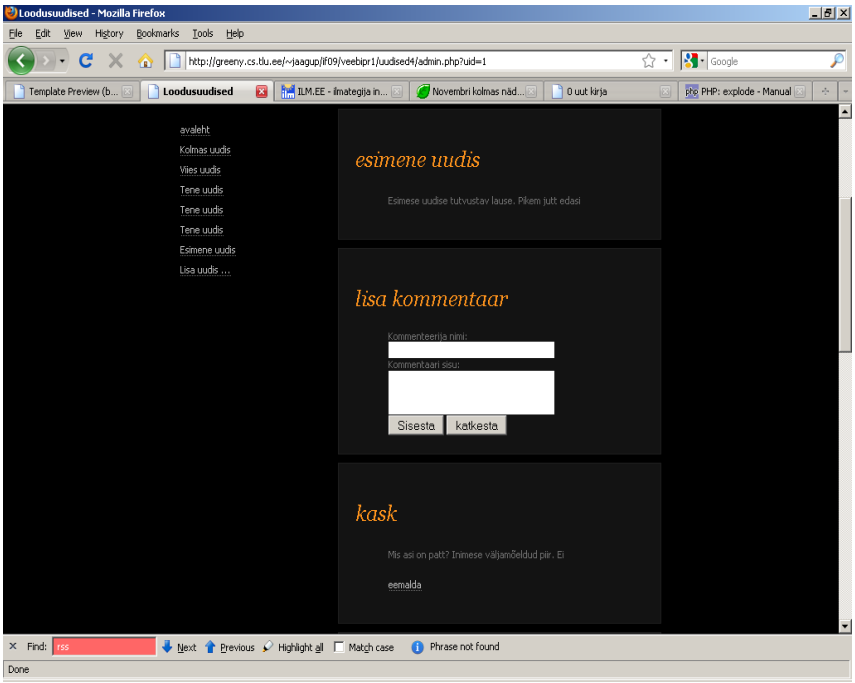
uudised.php

```
<?php
  require("abifunktsioonid.php");
  tootleSisend();
  $loetelu=$uudised->kysiTutvustused();
?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="content-type" content="text/html; charset=utf-8" />
<title>Loodusuudised</title>
<link href="kujundus.css" rel="stylesheet" type="text/css" />
</head>
<body>
```

```
<div id="header">
       <h1>Loodusuudised</h1>
       <h2>PHP kursuse raames</h2>
</div>
<div id="content">
       <div id="colOne">
              <div id="menu">
                     <ul>

                           <?php
                             echo "<li><a href='$_SERVER[PHP_SELF]'>avaleht</a></li>";
                             foreach($loetelu as $rida){
                               echo "<li><a href=
'$_SERVER[PHP_SELF]?uid=$rida->id'>$rida->pealkiri</a></li>";
                             }
                           ?>
                     </ul>
              </div>
       </div>
       <div id="colTwo">
         <?php
          if(!isSet($_REQUEST["uid"])){
           foreach($loetelu as $rida){
            echo "<div class='post'>
                 <h2>$rida->pealkiri</h2>
                 <p>
                   $rida->algus;
                   <br />
                   <a href='$_SERVER[PHP_SELF]?uid=$rida->id'>Loe edasi ...</a>
                 </p>
                </div>
             ";
           }
          } else {
            $rida=$uudised->kysiUudis($_REQUEST["uid"]);
            echo  "
              <div class='post'>
                 <h2>$rida->pealkiri</h2>
                 <p>
                   $rida->sisu
                 </p>
              </div>
              <div class='post'>
                <form action='$_REQUEST[PHP_SELF]' method='post'>
                 <input type='hidden' name='uid' value='$rida->id' />
                 <h2>Lisa kommentaar</h2>
                 Kommenteerija nimi:<br />
                 <input type='text' name='kommenteerija' class='sisestus'/><br />
                 Kommentaari sisu: <br />
                 <textarea name='kommentaarisisu' class='sisestus'></textarea>
                 <br />
                 <input type='submit' name='kommentaarisisestus' value='Sisesta' />
                 <input type='submit' name='kommentaarikatkestus' value='katkesta' />
                </form>
              </div>
            ";
            $kommentaarimassiiv=$kommentaariobj->kysiKommentaarid();
            foreach($kommentaarimassiiv as $kommentaar){
              echo "
                <div class='post'>
                   <h2>$kommentaar->kommenteerija</h2>
                   <p>
                     $kommentaar->kommentaarisisu
                   </p>
                </div>
              ";
            }
          }
          ?>
              </div>
</div>
```

```
<div id="footer">
	<p>Copyright &copy; 2006 Below the Horizon. Designed by <a
href="http://freecsstemplates.org"><strong>Free CSS Templates</strong></a></p>
</div>
</body>
</html>
```





Administraatorileht

Administraatorilehel on vajalik koht uudiste lisamiseks, samuti kommentaaride eemaldamiseks. Kuna abifunktsioonide fail on kõigile kasutada, siis turvakaalutlustel ei pandud vastavate

funktsioonide väljakutseid URLi pealt saabuvate andmete põhjal sinna, vaid tehakse need toimingud administraatorilehe päises. Kui nüüd hiljem administraatorileht kaitsta parooli ja sessioonimuutuja abil, või näiteks paigutada eraldi kausta, kus ligipääsuõigused .htaccess-faili abil piiratud, siis tavakasutaja enam uudiseid lisada või kommentaare sobimatuks muuta ei saa.

Nagu põhjalikumatele andmelehtedele kohane, siin sobimatuid kommentaare ei kustutata, vaid lihtsalt määratakse eraldi tulbas, et nad ei ole "korras". Eelnenud abifunktsioonide faili sai juba lisatud, et näidataks vaid korras kommentaare - niimoodi jääbki veebileht viisakaks, samas on baasist aga võimalik jälitada, et mida siis vahepeal on lehele kirjutada suvatsetud.

admin.php

```
<?php
  require("abifunktsioonid.php");
  tootleSisend();
  if(isSet($_REQUEST["uudiselisamine"])){
    $uudised->lisaUudis($_REQUEST["uudisepealkiri"], $_REQUEST["uudisesisu"]);
    header("Location: $_SERVER[PHP_SELF]");
    exit();
  }
  if(isSet($_REQUEST["pahakommentaar"])){
    $kask=$yhendus->prepare("UPDATE kommentaarid SET korras=0 WHERE id=?");
    $kask->bind_param("i", $_REQUEST["pahakommentaar"]);
    $kask->execute();
  }
  $loetelu=$uudised->kysiTutvustused();
?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="content-type" content="text/html; charset=utf-8" />
<title>Loodusuudised</title>
<link href="kujundus.css" rel="stylesheet" type="text/css" />
</head>
<body>
<div id="header">
       <h1>Loodusuudised</h1>
       <h2>PHP kursuse raames</h2>
</div>
<div id="content">
       <div id="colOne">
              <div id="menu">
                     <ul>

                            <?php
                              echo "<li><a href='$_SERVER[PHP_SELF]'>avaleht</a></li>";
                              foreach($loetelu as $rida){
                                echo "<li><a href=
                         '$_SERVER[PHP_SELF]?uid=$rida->id'>$rida->pealkiri</a></li>";
                              }
                              echo "
                                <li><a href=
                          '$_SERVER[PHP_SELF]?lisauudis=jah'>Lisa uudis ...</a></li>
                                ";
                            ?>
                     </ul>
              </div>
       </div>
       <div id="colTwo">
         <?php
          if(isSet($_REQUEST["lisauudis"])){
            echo "
              <div class='post'>
                <form action='$_SERVER[PHP_SELF]' method='post'>
                   Uudise pealkiri:<br />
                   <input type='text' name='uudisepealkiri' class='sisestus'/><br />
                   Uudise sisu:<br />
```

```
                <textarea name='uudisesisu' class='sisestus' ></textarea><br />
                <input type='submit' name='uudiselisamine' value='Lisa uudis' />
                <input type='submit' name='uudisekatkestus' value='Katkesta' />
              </form>
            </div>
          ";
        }
        if(!isSet($_REQUEST["uid"])){
         foreach($loetelu as $rida){
          echo "<div class='post'>
               <h2>$rida->pealkiri</h2>
               <p>
                 $rida->algus
                 <br />
                 <a href='$_SERVER[PHP_SELF]?uid=$rida->id'>Loe edasi ...</a>
               </p>
              </div>
           ";
         }
        } else {
          $rida=$uudised->kysiUudis($_REQUEST["uid"]);
          echo  "
            <div class='post'>
               <h2>$rida->pealkiri</h2>
               <p>
                 $rida->sisu
               </p>
            </div>
            <div class='post'>
              <form action='$_REQUEST[PHP_SELF]' method='post'>
               <input type='hidden' name='uid' value='$rida->id' />
               <h2>Lisa kommentaar</h2>
               Kommenteerija nimi:<br />
               <input type='text' name='kommenteerija' class='sisestus'/><br />
               Kommentaari sisu: <br />
               <textarea name='kommentaarisisu' class='sisestus'></textarea>
               <br />
               <input type='submit' name='kommentaarisisestus' value='Sisesta' />
               <input type='submit' name='kommentaarikatkestus' value='katkesta' />
              </form>
            </div>
          ";
          $kommentaarimassiiv=$kommentaariobj->kysiKommentaarid();
          foreach($kommentaarimassiiv as $kommentaar){
            echo "
              <div class='post'>
                 <h2>$kommentaar->kommenteerija</h2>
                 <p>
                   $kommentaar->kommentaarisisu
                 </p>
                 <p>
                   <a href=
                    '$_SERVER[PHP_SELF]?pahakommentaar=$kommentaar->id'>eemalda</a>
                 </p>
              </div>
            ";
          }
         }
        ?>
            </div>
</div>
<div id="footer">
      <p>Copyright &copy; 2006 Below the Horizon. Designed by <a
href="http://freecsstemplates.org"><strong>Free CSS Templates</strong></a></p>
</div>
</body>
</html>
```

Smarty

Lihtne tervitus

Lihtne tervitus Smarty abil. Esimese käsuna loetakse sisse Smarty põhiklassi fail (tasub vaadata, millises kataloogis ta asub). Edasi Smarty objekt valmis ning võibki mallilehte kuvada.

```
tervitus1.php
<?php
  require("Smarty-2.6.25/libs/Smarty.class.php");
  $smarty=new Smarty;
  $smarty->display("tervitus1.tpl");
?>
```

Mallilehed asuvad vaikimisi kaustas templates. Tasub jälgida, et sinna kõrvale oleks tehtud ka kaust templates_c ning antud sellele kõik õigused - siis võimalik Smartyl lehti käivituskõlbulikuks kompileerida. Aga esimene malllinäide näeb välja nagu täiesti tavaline HTML-leht.

```
templates/tervitus1.tpl
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html>
 <head>
   <title>Tervitusleht</title>
   <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
 </head>
 <body>
    <h1>Tervitamise leht</h1>
 </body>
</html>
```

Vaatamiseks tuleb avada ikka algne PHP-fail.

Andmete edastamine

Mallide üks põhivõlu on selles, et siin saab PHP muutujad ette valmistada, mallifailidele ette anda ning viimane lihtsalt kuvab neid sobiva koha peal - ilma, et peaks ise kusagilt väärtusi küsima minema. Siin näites anname mallile ette teate väärtuse.

```
tervitus2.php

<?php
  require("Smarty-2.6.25/libs/Smarty.class.php");
  $smarty=new Smarty;
  $smarty->assign("teade", "Tervitame koduseid gripihaigeid.");
  $smarty->display("tervitus2.tpl");
?>
```

Looksulgude vahel $teade kuvab sobivasse kohta teate ekraanile.

```
templates/tervitus2.tpl

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html>
 <head>
   <title>Tervitusleht</title>
   <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
 </head>
 <body>
   <h1>Tervitamise leht</h1>
   <div>
    {$teade}
   </div>
 </body>
</html>
```

Mitme failiga komplekt

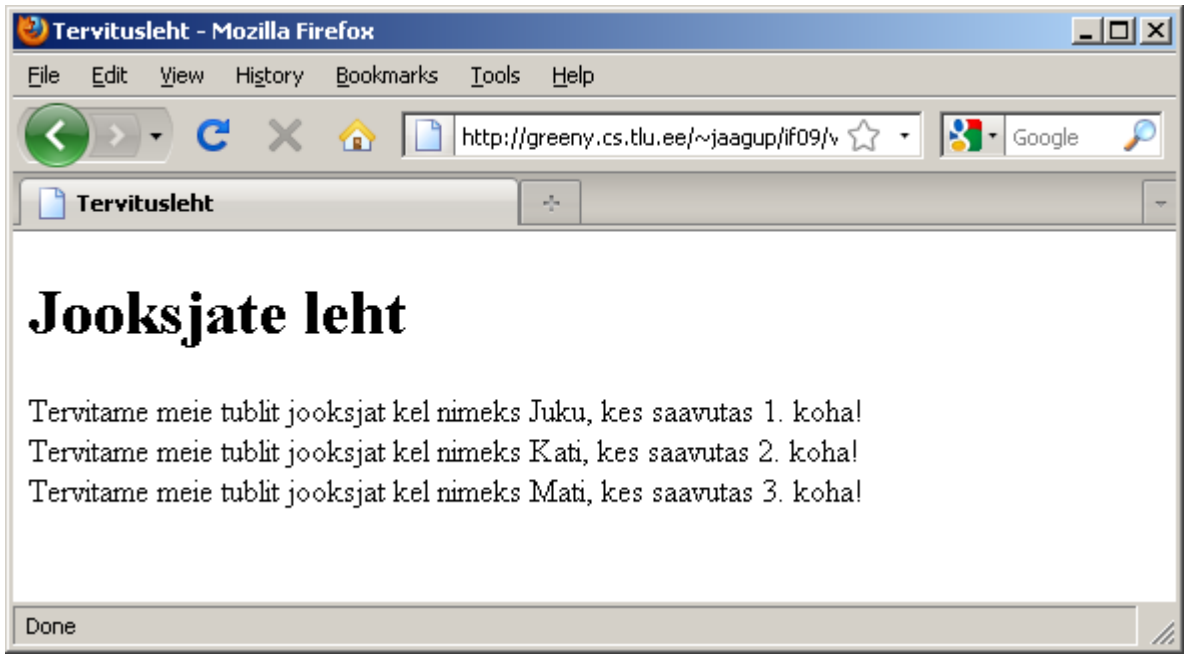
Mallide abil võib lehe jagada julgesti alamosadeks ning siis neid vajadust mööda välja kutsuda. Siin näites on eraldi mallifailiks päis, eraldi ühe jooksja andmete kuvamine ning lõpuks eraldi jalus.

```
templates/jooksjad_p2is.tpl
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html>
 <head>
    <title>Tervitusleht</title>
    <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
```

```
 </head>
 <body>
   <h1>Jooksjate leht</h1>

templates/jooksjad_jooksja.tpl
<div>
   Tervitame meie tublit jooksjat kel nimeks {$eesnimi},
    kes saavutas {$kohanr}. koha!
</div>


templates/jooksjad_jalus.tpl
  </body>
</html>
```

PHP-failis näidatakse kõigepealt päis. Siis igale jooksja andmed kujundatuna jooksja malli järele. Ning lõpuks jalus. Nõnda on kujundajal täiesti vabad käed jooksja andmete ilusaks muutmiseks - ainult muutujate nimed peavad paika jääma.

```
jooksjad.php
<?php
  require("Smarty-2.6.25/libs/Smarty.class.php");
  $smarty=new Smarty;
  $eesnimed=array("Juku", "Kati", "Mati");
  $smarty->display("jooksjad_p2is.tpl");
  $koht=0;
  foreach($eesnimed as $eesnimi){
    $koht++;
    $smarty->assign("eesnimi", $eesnimi);
    $smarty->assign("kohanr", $koht);
    $smarty->display("jooksjad_jooksja.tpl");
  }
  $smarty->display("jooksjad_jalus.tpl");
?>
```



Kahe arvu korrutamine

Lihtne sisestusega veebilehe näide, kus kogu kujundus on mallifailis, arvutus aga PHP-s.

```
templates/korrutamine.tpl

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
```

```
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html>
 <head>
   <title>Korrutusleht</title>
   <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
 </head>
 <body>
    <h1>Korrutamise leht</h1>
    <form action="korrutamine.php">
       <input type="text" name="arv1" /> x
       <input type="text" name="arv2" />
       <input type="submit" name="korrutusnupp" value="=" />
       {$vastusteade}
    </form>
 </body>
</html>
```

korrutamine.php

```
<?php
  require("Smarty-2.6.25/libs/Smarty.class.php");
  $smarty=new Smarty;
  if(isSet($_REQUEST["korrutusnupp"])){
     $vastusteade=$_REQUEST["arv1"]*$_REQUEST["arv2"];
  } else {
     $vastusteade="Palun sisesta andmed!";
  }
  $smarty->assign("vastusteade", $vastusteade);
  $smarty->display("korrutamine.tpl");
?>
```

Mitu mitmele seos

Bussipeatused ja liinid

Näide järgnevate tabelite sidumise põhjal nagu ülal antud ülesandes kirjas.

* Koosta peatuste tabel (id, peatusenimi). Lisa mõned peatused
* Koosta liinide tabel (liini_nr, liininimetus). Lisa mõned liinid
* Koosta peatumisaegade tabel (id, liini_nr, peatuse_id, aeg_algpeatusest). Lisa mõned peatumised
* Koosta reiside tabel(id, liini_nr, algusaeg, suund) (1-edasi, 2-tagasi).
* Koosta leht ühe bussiliini peatumisandmete näitamiseks. Andmed sorditakse peatusesse jõudmise aja järgi.

SQL-laused

Eks andmeid saab talletada mitmeti. Siin on eraldi "põhitabeliteks" peatused ja liinid. Ning peatumisaegade tabel seob need tabelid omavahel kokku, märkides seose juurde, kui palju igasse peatusesse jõudmiseks aega algpeatusest kulub.

```
CREATE TABLE peatused(
  id INT NOT NULL auto_increment PRIMARY KEY,
  peatusenimi VARCHAR(50)
);

INSERT INTO peatused(peatusenimi) VALUES ('Viljandi');
INSERT INTO peatused(peatusenimi) VALUES ('Karksi');
INSERT INTO peatused(peatusenimi) VALUES ('Tartu');
INSERT INTO peatused(peatusenimi) VALUES ('Valga');
INSERT INTO peatused(peatusenimi) VALUES ('Puhja');
INSERT INTO peatused(peatusenimi) VALUES ('Elva');
```

Nagu näha, pole liinide juures automaatselt suurenevat primaarvõtit. Kuna liini number ise peab olema unikaalne, siis võib selle rahumeeli ka ise ja otse INSERT-lausest sisse kirjutada.

```
CREATE TABLE liinid(
   liini_nr INT NOT NULL PRIMARY KEY,
   liini_nimetus VARCHAR(50)
);

INSERT INTO liinid VALUES (10, 'Viljandi-Valga kiir');
INSERT INTO liinid VALUES (11, 'Viljandi-Elva-Tartu');
INSERT INTO liinid VALUES (12, 'Karksi-Tartu');
```

Peatuste ja liinide sidumiseks on vaja teada peatuste id-sid.

```
SELECT * FROM peatused;

+----+-------------+
| id | peatusenimi |
+----+-------------+
|  1 | Viljandi    |
|  2 | Karksi      |
|  3 | Tartu       |
|  4 | Valga       |
|  5 | Puhja       |
|  6 | Elva        |
+----+-------------+

CREATE TABLE peatumisajad(
  id INT NOT NULL auto_increment PRIMARY KEY,
  liini_nr INT,
  peatuse_id INT,
```

```
  aeg_algpeatusest TIME
);

INSERT INTO peatumisajad(liini_nr, peatuse_id, aeg_algpeatusest)
 VALUES(10, 1, '00:00');

INSERT INTO peatumisajad(liini_nr, peatuse_id, aeg_algpeatusest)
 VALUES(10, 2, '00:30');

INSERT INTO peatumisajad(liini_nr, peatuse_id, aeg_algpeatusest)
 VALUES(10, 4, '01:30');
```

Ühe liini andmed olemas. Edasi peab veel koodide järgi vaatama, millist marsruuti liin sõidab.

```
mysql> SELECT * FROM peatumisajad;
+----+----------+------------+------------------+
| id | liini_nr | peatuse_id | aeg_algpeatusest |
+----+----------+------------+------------------+
|  1 |       10 |          1 | 00:00:00         |
|  2 |       10 |          2 | 00:30:00         |
|  3 |       10 |          4 | 01:30:00         |
+----+----------+------------+------------------+

INSERT INTO peatumisajad(liini_nr, peatuse_id, aeg_algpeatusest)
 VALUES(11, 1, '00:00');

INSERT INTO peatumisajad(liini_nr, peatuse_id, aeg_algpeatusest)
 VALUES(11, 5, '00:45');

INSERT INTO peatumisajad(liini_nr, peatuse_id, aeg_algpeatusest)
 VALUES(11, 6, '01:00');

INSERT INTO peatumisajad(liini_nr, peatuse_id, aeg_algpeatusest)
 VALUES(11, 3, '01:35');
```

Kui peatumisaegade tabelis on juba rohkemate liinide andmed, siis tuleb ühe liini andmete kätte saamiseks määrata, millise liini peatusi tahetakse.

```
mysql> SELECT * FROM peatumisajad WHERE liini_nr=11;
+----+----------+------------+------------------+
| id | liini_nr | peatuse_id | aeg_algpeatusest |
+----+----------+------------+------------------+
|  4 |       11 |          1 | 00:00:00         |
|  5 |       11 |          5 | 00:45:00         |
|  6 |       11 |          6 | 01:00:00         |
|  7 |       11 |          3 | 01:35:00         |
+----+----------+------------+------------------+

INSERT INTO peatumisajad(liini_nr, peatuse_id, aeg_algpeatusest)
 VALUES(12, 2, '00:00');

INSERT INTO peatumisajad(liini_nr, peatuse_id, aeg_algpeatusest)
 VALUES(12, 6, '01:15');

INSERT INTO peatumisajad(liini_nr, peatuse_id, aeg_algpeatusest)
 VALUES(12, 3, '01:55');
```

Peatumisaegade tabelis on kirjas vaid ajad algpeatusest. Tahtes teada tegelikke peatusest väljumise aegu, tuleb algsetele aegadele liita juurde algpeatusest väljumise aeg. Õnneks on MySQLil olemas käsklus ADDTIME.

```
SELECT ADDTIME('13:10', aeg_algpeatusest) FROM peatumisajad
  WHERE liini_nr=11
  ORDER BY aeg_algpeatusest;

+-----------------------------------+
| ADDTIME('13:10', aeg_algpeatusest) |
+-----------------------------------+
| 13:10:00                          |
| 13:55:00                          |
| 14:10:00                          |
| 14:45:00                          |
+-----------------------------------+
```

Väljundist on rohkem kasu, kui väljumiskellaaja kõrvale pannakse ka peatuse nimi, mille kohta see käib. Selleks ühendatakse peatumisaegade tabeli kõrvale peatuste tabel ning näidatakse sealt välja vaid rida, mille id vastab peatuseaegade tabelist kuvatava rea peatuse id-ga. Kellaaja tulbale antakse AS käsklusega endise valemi asemel viisakas nimi.

Kuna andmebaasis vaikimisi ei ole ajad järjestatud, siis on kasulik tulemus aegade järgi sortida. Muidu võivad küll peatumisajad õiged olla, aga liini peatusi ei pruugita näidata õiges järjekorras.

```
SELECT peatusenimi,
  ADDTIME('13:10', aeg_algpeatusest) as kellaaeg
  FROM peatumisajad, peatused
  WHERE peatumisajad.peatuse_id=peatused.id
   AND liini_nr=11
  ORDER BY aeg_algpeatusest;


+-------------+----------+
| peatusenimi | kellaaeg |
+-------------+----------+
| Viljandi    | 13:10:00 |
| Puhja       | 13:55:00 |
| Elva        | 14:10:00 |
| Tartu       | 14:45:00 |
+-------------+----------+
```

13.10 oli eelmise näite juures lihtsalt katseks võetud algusaeg. Tegelikkuses on ju sageli samal liinil päevas mitu reisi. Ning bussid sõidavad samal liinil mõlemas suunas. Nii tulebki reiside tabelisse kirja liini nr, väljumisaeg ning suund. Kokkuleppeliselt tähistasime siin algpeatusest lõpppeatuse poole liikuva suuna ühega ning tagasisuuna kahega.

```
CREATE TABLE reisid(
  id INT NOT NULL auto_increment PRIMARY KEY,
  liini_nr INT,
  algusaeg TIME,
  suund INT DEFAULT 1
);
```

Mõned andmed sisse ka.

```
INSERT INTO reisid (liini_nr, algusaeg) VALUES (11, '13:10');
INSERT INTO reisid (liini_nr, algusaeg) VALUES (11, '16:10');

INSERT INTO reisid (liini_nr, algusaeg, suund) VALUES (11, '16:10', 2);

INSERT INTO reisid (liini_nr, algusaeg) VALUES (10, '9:15');
```

Pärisuunas reisi aegadega suaab suhteliselt lihtsalt hakkama - eelmises päringus tuleb 13:10

asendada lihtsalt konkreetse reisi algusajaga ning võibki tulemustega rahul olla.

Tagasireisiga on aga rohkem tegemist. Üldjuhul võiks liinil peatuste vahel mõlemas suunas liikumine sama palju aega võtta. On kindlasti erandeid, aga sel juhul on on tõenäoliselt üheks lahenduseks teha vähemasti andmebaasi tasemel kaks erinevat liini - üks kummagi suuna jaoks. Ning sõita neil liinidel vaid edasisuunas.

Kui aga tegemist tavalise sirge ja sileda ühetaolise tee liiniga, siis tasub kõigepealt leida liini läbimise aeg, ehk siis lõpppeatusesse jõudmise aeg. Ning sealt hakata ükshaaval ülejäänud aegu lahutama. Lõpppeatusest iseenese aja lahutamine annab tulemuseks nulli - aga seda meil ju vaja ongi. Tagasisuunas sõites tulebki buss ju reisi väljumise ajal lõpppeatusest välja. Lõpppeatuse ja temale eelneva peatuse aja vahe näitab, kui palju kulub tolle viimase või siis tagasisuunas esimese peatusevahe läbimiseks. Jällegi on ka seda meil vaja: nii palju tuleb siis tagantotsast väljumise ajale juurde liita, et eelneva peatuse aeg saada.

Kõik algpeatusest kulunud ajad 11nda liini kohta leiab päringuga:

```
SELECT aeg_algpeatusest FROM peatumisajad WHERE liini_nr=11;
```

Suurim neist on lõpppeatuseni jõudmise aeg:

```
SELECT MAX(aeg_algpeatusest) FROM peatumisajad WHERE liini_nr=11;
```

Edasi lahutan sellest ajast iga peatuse kohta tabelis oleva aja. Lahutamiseks ikka käsklus ADDTIME, ainult, et tulbal aeg_algpeatusest on miinusmärk ees.

```
SELECT ADDTIME(
  (SELECT MAX(aeg_algpeatusest) FROM peatumisajad WHERE liini_nr=11),
  -aeg_algpeatusest
 ) as aeg
FROM peatumisajad where liini_nr=11 order by aeg;
```

```
+----------+
| aeg      |
+----------+
| 00:00:00 |
| 00:35:00 |
| 00:50:00 |
| 01:35:00 |
+----------+
```

Tahtes peatuste nimed ka juurde saada, siis tuleb sellele päringule juurde lisada veel peatuste tabel ning sealt id järgi soovitud nimi välja võtta.

```
SELECT peatusenimi, ADDTIME(
  (SELECT MAX(aeg_algpeatusest) FROM peatumisajad WHERE liini_nr=11),
  -aeg_algpeatusest
 ) as aeg
FROM peatumisajad, peatused
WHERE peatumisajad.peatuse_id=peatused.id
 AND liini_nr=11
 ORDER BY aeg;
```

Veebiliides

Edasi paras aeg nende päringute abil veebileht tööle panna. Laiskuse ja lihtsuse tõttu pole siia eraldi faile tekitatud, vaid andmebaasipäringud ja HTMLi kood on kõik ühe lehe peal koos.

Alustuseks liininumbrite loetelu:

```
<?php
   $kask=$yhendus->prepare("SELECT liini_nr, liini_nimetus FROM liinid");
   $kask->bind_result($liini_nr, $liini_nimetus);
   $kask->execute();
   while($kask->fetch()){
      echo "<li><a href='$_SERVER[PHP_SELF]?liini_nr=$liini_nr'>$liini_nr</a>
                $liini_nimetus</li>\n";
   }
?>
```

Edasi iga liini väljumisajad mõlemas suunas. IF-lausega juba SQL-lauses pannakse tulpa "suundtekst" sisse sõidusuunda näitav sõna.

```
$kask=$yhendus->prepare("SELECT id, algusaeg,
  if(suund=1, 'edasi', 'tagasi') as suundtekst
  FROM reisid
  WHERE liini_nr=?");
```

Konkreetse reisi peatusaegade näitamine on sedakorda jaotatud kahe päringu vahel - üks läinuks muidu liialt keerukaks. Kõigepealt saab reisi_id järgi kätte liinu nubri, algusaja ja suuna.

```
$kask=$yhendus->prepare(
        "SELECT liini_nr, algusaeg, suund FROM reisid WHERE id=?");
$kask->bind_param("i", $_REQUEST["reisi_id"]);
$kask->bind_result($liini_nr, $algusaeg, $suund);
$kask->execute();
$kask->fetch();
$kask->close();
```

Vastavalt suunale koostatakse edasine SQL-lause. Pärisuuna puhul piisab peatusest väljumise aja arvutamiseks reisi alguskellaaja ja vastava peatuseni kuluva aja liitmisest.

Tagasisuuna puhul aga tehakse läbi eelnevalt näidatud keerukam arvutus. Ning et liini numbri parameetrit ei peaks SQL-lausesse mitu korda sisestama, selleks on suurimat aega leidvas alampäringus olev lauseses tabel peatumisajad nimetatud p1-ks, et saaks sealset liininumbrit võrrelda allpool oleva sama tabeli liininumbriga.

```
if($suund==1){
   $ajalause="ADDTIME(?, aeg_algpeatusest)";
} else {
   $suurimaeg="SELECT MAX(aeg_algpeatusest) FROM peatumisajad as p1
         WHERE p1.liini_nr=peatumisajad.liini_nr";
   $ajalause="ADDTIME(?, ADDTIME(($suurimaeg), -aeg_algpeatusest))";
}
```

Aega arvutav lauseosa lisatakse lõpuks sobivale kohale põhilause sisse.

```
$lause="SELECT peatusenimi, $ajalause as aeg
   FROM peatumisajad, peatused
    WHERE peatumisajad.peatuse_id=peatused.id
     AND peatumisajad.liini_nr=?
     ORDER BY aeg";
$kask=$yhendus->prepare($lause);
$kask->bind_param("si", $algusaeg, $liini_nr);
$kask->bind_result($peatusenimi, $aeg);
$kask->execute();
while($kask->fetch()){
  echo "<tr><td>$peatusenimi</td><td>$aeg</td></tr>";
```

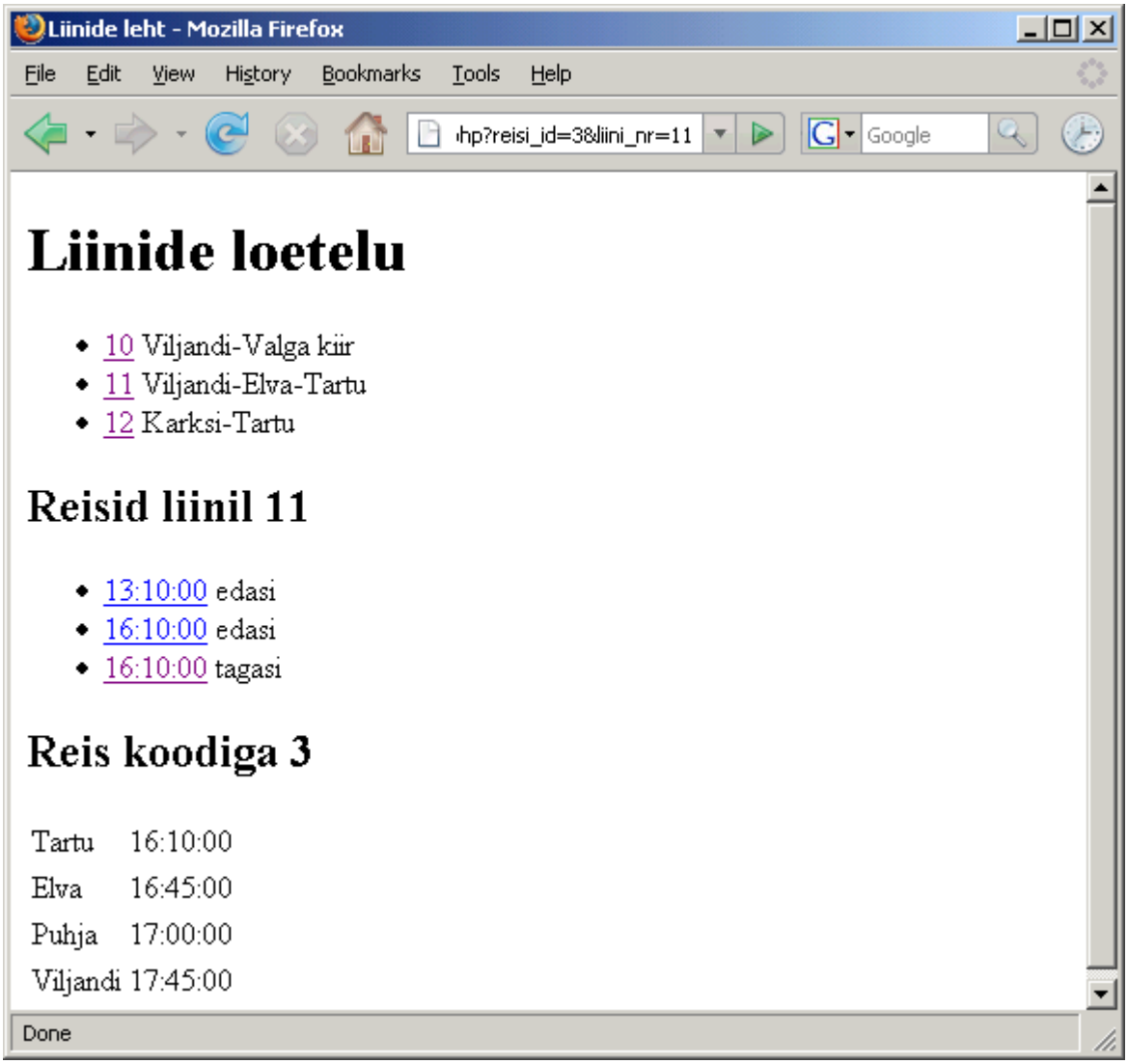
```
          }
```

Ning PHP-fail tervikuna.

vaataja.php

```
<?php
  $yhendus=new mysqli("localhost", "jaagup", "", "jaagup");
?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <title>Liinide leht</title>
    <meta http-equiv="Content-Type" content="text/html;charset=utf-8" />
  </head>
  <body>
     <h1>Liinide loetelu</h1>
     <ul>
       <?php
          $kask=$yhendus->prepare("SELECT liini_nr, liini_nimetus FROM liinid");
          $kask->bind_result($liini_nr, $liini_nimetus);
          $kask->execute();
          while($kask->fetch()){
             echo "<li><a href='$_SERVER[PHP_SELF]?liini_nr=$liini_nr'>$liini_nr</a>
                       $liini_nimetus</li>\n";
          }
       ?>
     </ul>
     <?php if(isSet($_REQUEST["liini_nr"])): ?>
        <h2>Reisid liinil <?=$_REQUEST["liini_nr"] ?></h2>
        <ul>
        <?php
          $kask=$yhendus->prepare("SELECT id, algusaeg,
            if(suund=1, 'edasi', 'tagasi') as suundtekst
            FROM reisid
            WHERE liini_nr=?");
          $kask->bind_param("i", $_REQUEST["liini_nr"]);
          $kask->bind_result($id, $algusaeg, $suundtekst);
          $kask->execute();
          while($kask->fetch()){
             echo "<li><a href=
          '$_SERVER[PHP_SELF]?reisi_id=$id&amp;liini_nr=$_REQUEST[liini_nr]'>$algusaeg</a
>
                       $suundtekst</li>";
          }
        ?>
        </ul>
     <?php endif  ?>
     <?php if(isSet($_REQUEST["reisi_id"])): ?>
       <h2>Reis koodiga <?=$_REQUEST["reisi_id"] ?></h2>
       <table>
       <?php
          $kask=$yhendus->prepare(
                  "SELECT liini_nr, algusaeg, suund FROM reisid WHERE id=?");
          $kask->bind_param("i", $_REQUEST["reisi_id"]);
          $kask->bind_result($liini_nr, $algusaeg, $suund);
          $kask->execute();
          $kask->fetch();
          $kask->close();
          if($suund==1){
             $ajalause="ADDTIME(?, aeg_algpeatusest)";
          } else {
             $suurimaeg="SELECT MAX(aeg_algpeatusest) FROM peatumisajad as p1
                   WHERE p1.liini_nr=peatumisajad.liini_nr";
             $ajalause="ADDTIME(?, ADDTIME(($suurimaeg), -aeg_algpeatusest))";
          }
          $lause="SELECT peatusenimi, $ajalause as aeg
             FROM peatumisajad, peatused
```

```
            WHERE peatumisajad.peatuse_id=peatused.id
             AND peatumisajad.liini_nr=?
             ORDER BY aeg";
          $kask=$yhendus->prepare($lause);
          $kask->bind_param("si", $algusaeg, $liini_nr);
          $kask->bind_result($peatusenimi, $aeg);
          $kask->execute();
          while($kask->fetch()){
            echo "<tr><td>$peatusenimi</td><td>$aeg</td></tr>";
          }
       ?>
       </table>
     <?php endif ?>
  </body>
</html>
```



Viide samale tabelile

Viidete abil seotud võivad olla ka sama tüüpi andmed. Kataloogipuus on ülemkataloog samuti kataloog nagu alamkataloog. Ning kui tegemist inimeste tabeliga, siis iga inimese isa ja ema on

samuti inimesed ning nende andmeid saab inimeste tabelis hoida. Lõputult pole sugupuus andmeid ning kusagilt maalt peavad esivanemate viited tühjaks jääma. Aga päris hulga sugupõlvi kannatab nõnda üles tähendada küll. Selle põhjal ka mõned ülesanded ja lahendused.

* Looge tabel isikud(id, eesnimi, synniaasta, isa_id, ema_id)

```
CREATE TABLE isikud(
  id INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
  eesnimi VARCHAR(50) NOT NULL,
  synniaasta INT NOT NULL,
  isa_id INT,
  ema_id INT
);
```

* Lisage andmed. Kui vanema isikut pole tabelis, siis selle koha peal tühiväärtus NULL

Andmestiku aluseks võtame perekonna, kus on lapsed Juku ja Kati, isa Martin, ema Anne. Ning Martini vanemad on Juhan ja Marta. Anne vanemaid pole tabelisse kantud. Samuti mitte Juhani ja Marta vanemaid.

```
Lapsed:
       Juku 2005              Kati 2003
Vanemad:
       Martin 1970         Anne 1974
Vanavanemad
    Juhan 1935      Marta 1938
```

Kui muud vaikimise väärtust pole määratud, siis tulebki määramata tulpade (isa_id ja ema_id) väärtuseks NULL. Vanimad tegelased tuleb kõigepealt sisse kanda, sest nende IDsid on vaja järgnevate isikute vanemate määramisel.

```
INSERT INTO isikud(eesnimi, synniaasta) VALUES('Juhan', 1935);
INSERT INTO isikud(eesnimi, synniaasta) VALUES('Marta', 1938);
```

Pärast andmete sisestamist saab küsida need välja koos automaatselt lisatud ID-dega.

```
mysql> SELECT * FROM isikud;
+----+---------+------------+--------+--------+
| id | eesnimi | synniaasta | isa_id | ema_id |
+----+---------+------------+--------+--------+
|  1 | Juhan   |       1935 |   NULL |   NULL |
|  2 | Marta   |       1938 |   NULL |   NULL |
+----+---------+------------+--------+--------+

INSERT INTO isikud(eesnimi, synniaasta, isa_id, ema_id)
 VALUES ('Martin', 1970, 1, 2);


INSERT INTO isikud(eesnimi, synniaasta) VALUES ('Anne', 1974);


mysql> SELECT * FROM isikud;
+----+---------+------------+--------+--------+
| id | eesnimi | synniaasta | isa_id | ema_id |
+----+---------+------------+--------+--------+
|  1 | Juhan   |       1935 |   NULL |   NULL |
|  2 | Marta   |       1938 |   NULL |   NULL |
|  3 | Martin  |       1970 |      1 |      2 |
|  4 | Anne    |       1974 |   NULL |   NULL |
+----+---------+------------+--------+--------+
```

```
INSERT INTO isikud(eesnimi, synniaasta, isa_id, ema_id)
 VALUES ('Juku', 2005, 3, 4);

INSERT INTO isikud(eesnimi, synniaasta, isa_id, ema_id)
 VALUES ('Kati', 2003, 3, 4);
```

```
mysql> SELECT * FROM isikud;
+----+---------+------------+--------+--------+
| id | eesnimi | synniaasta | isa_id | ema_id |
+----+---------+------------+--------+--------+
|  1 | Juhan   |       1935 |   NULL |   NULL |
|  2 | Marta   |       1938 |   NULL |   NULL |
|  3 | Martin  |       1970 |      1 |      2 |
|  4 | Anne    |       1974 |   NULL |   NULL |
|  5 | Juku    |       2005 |      3 |      4 |
|  6 | Kati    |       2003 |      3 |      4 |
+----+---------+------------+--------+--------+
```

* Väljastage isikute eesnimed

```
SELECT eesnimi FROM isikud;
```

* Väljastage isikute ees- ja isanimed kui võimalik

Tabelist isikud luuakse päringus kaks eksemplari. Üks nimega uuritavad, teine nimega isad. Ning uuritavate isa_id järgi kõrvutatakse külge isa rida isade tabelist. SELECT-osas määratakse, milliseid tulpi näidataks.

```
SELECT uuritavad.eesnimi, isad.eesnimi as isanimi
  FROM isikud as uuritavad, isikud as isad
   WHERE uuritavad.isa_id=isad.id;
```

```
+---------+---------+
| eesnimi | isanimi |
+---------+---------+
| Martin  | Juhan   |
| Juku    | Martin  |
| Kati    | Martin  |
+---------+---------+
```

* Väljastage inimeste ees-, isa- ja emanimed kui võimalik.
Sarnane eelmise päringuga, ainult, et kolmas isikute tabeli koopia tuli juurde emade nime all.

```
SELECT uuritavad.eesnimi, isad.eesnimi as isanimi, emad.eesnimi as emanimi
  FROM isikud as uuritavad, isikud as isad, isikud as emad
   WHERE uuritavad.isa_id=isad.id AND uuritavad.ema_id=emad.id;
```

* Väljastage inimeste ees-, isa- ja emanimed kui võimalik.

```
SELECT uuritavad.eesnimi, isad.eesnimi AS isanimi
  FROM isikud AS uuritavad LEFT JOIN isikud as isad
   ON uuritavad.isa_id=isad.id;
```

```
+---------+---------+---------+
| eesnimi | isanimi | emanimi |
+---------+---------+---------+
| Martin  | Juhan   | Marta   |
| Juku    | Martin  | Anne    |
| Kati    | Martin  | Anne    |
+---------+---------+---------+
```

* Väljastage kõikide eesnimed, isanimed nendel kel teada (LEFT JOIN)

```
SELECT uuritavad.eesnimi, isad.eesnimi AS isanimi, emad.eesnimi AS emanimi
  FROM isikud AS uuritavad LEFT JOIN isikud AS isad
   ON uuritavad.isa_id=isad.id LEFT JOIN isikud AS emad
   ON uuritavad.ema_id=emad.id;
```

```
+---------+---------+---------+
| eesnimi | isanimi | emanimi |
+---------+---------+---------+
| Juhan   | NULL    | NULL    |
| Marta   | NULL    | NULL    |
| Martin  | Juhan   | Marta   |
| Anne    | NULL    | NULL    |
| Juku    | Martin  | Anne    |
| Kati    | Martin  | Anne    |
+---------+---------+---------+
```

* Väljastage, kui vana olid isa ja ema siis kui trükitav isik sündis.

```
SELECT uuritavad.eesnimi, isad.eesnimi AS isanimi, emad.eesnimi AS emanimi,
    uuritavad.synniaasta-isad.synniaasta as isasyndimisel,
    uuritavad.synniaasta-emad.synniaasta as emasyndimisel
  FROM isikud AS uuritavad LEFT JOIN isikud AS isad
   ON uuritavad.isa_id=isad.id LEFT JOIN isikud AS emad
   ON uuritavad.ema_id=emad.id;
```

```
+---------+---------+---------+---------------+---------------+
| eesnimi | isanimi | emanimi | isasyndimisel | emasyndimisel |
+---------+---------+---------+---------------+---------------+
| Juhan   | NULL    | NULL    |          NULL |          NULL |
| Marta   | NULL    | NULL    |          NULL |          NULL |
| Martin  | Juhan   | Marta   |            35 |            32 |
| Anne    | NULL    | NULL    |          NULL |          NULL |
| Juku    | Martin  | Anne    |            35 |            31 |
| Kati    | Martin  | Anne    |            33 |            29 |
+---------+---------+---------+---------------+---------------+
```

* Väljastage inimesed, kelle isa on vanem kui ema

```
SELECT uuritavad.eesnimi
  FROM isikud as uuritavad, isikud as isad, isikud as emad
  WHERE uuritavad.isa_id=isad.id AND uuritavad.ema_id=emad.id
   AND  isad.synniaasta<emad.synniaasta;
```

```
+---------+
| eesnimi |
+---------+
| Martin  |
| Juku    |
| Kati    |
+---------+
```

* Leia iga inimese laste arv

```
* Leia iga inimese vanima lapse sünniaasta.
```

```
SELECT MAX(synniaasta) FROM isikud;
+-----------------+
| MAX(synniaasta) |
+-----------------+
|            2005 |
```

```
+-----------------+
```

* Leia, mitu inimest on sündinud enne aastat 2000

```
SELECT COUNT(*) FROM isikud WHERE synniaasta<2000;
+----------+
| COUNT(*) |
+----------+
|        4 |
+----------+
```

* Väljasta inimeste nimed sündimise aastakümnete kaupa.

Kõigepealt küsime välja kõik inimesed koos nende sünniaastatega.

```
mysql> SELECT eesnimi, synniaasta FROM isikud;
+---------+------------+
| eesnimi | synniaasta |
+---------+------------+
| Juhan   |       1935 |
| Marta   |       1938 |
| Martin  |       1970 |
| Anne    |       1974 |
| Juku    |       2005 |
| Kati    |       2003 |
+---------+------------+
```

Edasi grupeerime nad sünniaastate kaupa. Kui mõnel aastal sündinuks mitu inimest, siis oleks nad ka selle aasta juures loetelus.

```
SELECT GROUP_CONCAT(eesnimi), synniaasta as synniaeg FROM isikud
  GROUP BY synniaeg
```

```
+-----------------------+----------+
| GROUP_CONCAT(eesnimi) | synniaeg |
+-----------------------+----------+
| Juhan                 |     1935 |
| Marta                 |     1938 |
| Martin                |     1970 |
| Anne                  |     1974 |
| Kati                  |     2003 |
| Juku                  |     2005 |
+-----------------------+----------+
```

Tahtes saada nimesid gruppi kümne aasta kaupa, arvutame sünnikümnendi. Jagame sünniaasta kümnega, võtame täisosa ning korrutame uuesti kümnega. Nii on ka 1938ndal aastal sündinud inimese sünnikümnend selle arvutuse põhjal 1930. Ning edasi saab juba selle tunnuse järgi grupeerida nii, et samal kümnendil sündinud inimesed samasse gruppi jäävad.

```
SELECT GROUP_CONCAT(eesnimi), FLOOR(synniaasta/10)*10 as synniaeg FROM isikud
  GROUP BY synniaeg
```

```
+-----------------------+----------+
| GROUP_CONCAT(eesnimi) | synniaeg |
+-----------------------+----------+
| Juhan,Marta           |     1930 |
| Martin,Anne           |     1970 |
| Juku,Kati             |     2000 |
+-----------------------+----------+
```

* Leia inimese laste arv

Kui tabelis oleks vaid lapsevanema_id, mitte isade või emade oma eraldi, siis käiks päring suhteliselt lihtsamalt. Tuleks vaid kokku lugeda iga andmerea puhul mitme rea pealt talle viidatakse. Kuna siin said tehtud isade ja emade viited eraldi, siis on ka arvutamist rohkem.

```
Alustuseks kõik lapsed koos oma isade nimedega.

SELECT isad.eesnimi, lapsed.eesnimi
 FROM isikud AS isad, isikud AS lapsed
 WHERE lapsed.isa_id=isad.id

+---------+---------+
| eesnimi | eesnimi |
+---------+---------+
| Juhan   | Martin  |
| Martin  | Juku    |
| Martin  | Kati    |
+---------+---------+
```

Iga isa laste arv - kusjuures praegu uuritakse igalt isikult, mitmele inimesele ta isaks on.

```
SELECT isad.eesnimi,
   (SELECT COUNT(*) FROM isikud AS lapsed
     WHERE lapsed.isa_id=isad.id) AS lastearv
 FROM isikud AS isad


+---------+----------+
| eesnimi | lastearv |
+---------+----------+
| Juhan   |        1 |
| Marta   |        0 |
| Martin  |        2 |
| Anne    |        0 |
| Juku    |        0 |
| Kati    |        0 |
+---------+----------+
```

Nüüd näidatakse välja vaid need, kel tõepoolest isana lapsi on. Nagu näha, tuli sama arvutus kahel korral kirja panna. Ühel korral arvutamiseks, teisel korral võrdlemiseks. Vajadusel saab selliseid toiminguid vahe- või abitabelite kaudu optimeerida.

```
SELECT isad.eesnimi,
   (SELECT COUNT(*) FROM isikud AS lapsed
     WHERE lapsed.isa_id=isad.id) AS lastearv
 FROM isikud AS isad
   WHERE (SELECT COUNT(*) FROM isikud AS lapsed
     WHERE lapsed.isa_id=isad.id)>0;


+---------+----------+
| eesnimi | lastearv |
+---------+----------+
| Juhan   |        1 |
| Martin  |        2 |
+---------+----------+
```

Kõik laste ja vanemate vahelised seosed - olgu vanemaks siis isa või ema.

```
SELECT lapsed.eesnimi, vanemad.eesnimi AS vanem
  FROM isikud AS lapsed, isikud AS vanemad
  WHERE lapsed.isa_id=vanemad.id OR
        lapsed.ema_id=vanemad.id;


+---------+--------+
```

```
| eesnimi | vanem  |
+---------+--------+
| Martin  | Juhan  |
| Martin  | Marta  |
| Juku    | Martin |
| Juku    | Anne   |
| Kati    | Martin |
| Kati    | Anne   |
+---------+--------+
```

Sealt edasi saab kätte ka juba iga lapsevanema laste arvu ning laste nimed, mida me algul otsima hakkasime.

```
SELECT vanemad.eesnimi AS vanem,
  COUNT(*) as laste_arv,
  GROUP_CONCAT(lapsed.eesnimi) as laste_nimed
  FROM isikud AS lapsed, isikud AS vanemad
  WHERE lapsed.isa_id=vanemad.id OR
        lapsed.ema_id=vanemad.id
  GROUP BY vanemad.id;
```

```
+--------+-----------+-------------+
| vanem  | laste_arv | laste_nimed |
+--------+-----------+-------------+
| Juhan  |         1 | Martin      |
| Marta  |         1 | Martin      |
| Martin |         2 | Juku,Kati   |
| Anne   |         2 | Juku,Kati   |
+--------+-----------+-------------+
```

Eelnenud päringut saab aga andmete leidmiseks edasi arendada. Nüüd on juures ka roll, et kas tegemisti isa või emaga. Tingimust aitab sõnadesse vormida if-käsklus. Eeldatakse, et kui vastav vanem ei ole lapsele isa, järelikult on ema.

```
SELECT lapsed.eesnimi,
   IF(lapsed.isa_id=vanemad.id, 'isa', 'ema') AS roll
   , vanemad.eesnimi AS vanem
  FROM isikud AS lapsed, isikud AS vanemad
  WHERE lapsed.isa_id=vanemad.id OR
        lapsed.ema_id=vanemad.id;
```

```
+---------+------+--------+
| eesnimi | roll | vanem  |
+---------+------+--------+
| Martin  | isa  | Juhan  |
| Martin  | ema  | Marta  |
| Juku    | isa  | Martin |
| Juku    | ema  | Anne   |
| Kati    | isa  | Martin |
| Kati    | ema  | Anne   |
+---------+------+--------+
```

Iga lapsevanema vanima lapse sünniaasta

```
SELECT eesnimi,
  (SELECT MIN(synniaasta) FROM isikud AS lapsed
   WHERE lapsed.ema_id=vanemad.id OR lapsed.isa_id=vanemad.id)
    AS vanimalapsesynniaasta
 FROM isikud as vanemad;
```

```
+---------+-----------------------+
| eesnimi | vanimalapsesynniaasta |
+---------+-----------------------+
```

```
| Juhan   |                 1970 |
| Marta   |                 1970 |
| Martin  |                 2003 |
| Anne    |                 2003 |
| Juku    |                 NULL |
| Kati    |                 NULL |
+---------+----------------------+
```

Veel lapsevanemaks mitte saanute andmete eemaldamiseks muudeti eelnev päring alampäringuks. Sellisel juhul ei pea vanima lapse sünniaastat võrdlemiseks uuesti välja arvutama, vaid saab tingimuseks kohe kirjutada `WHERE vanimalapsesynniaasta IS NOT NULL.`

```
SELECT eesnimi, vanimalapsesynniaasta FROM
 (SELECT eesnimi,
  (SELECT MIN(synniaasta) FROM isikud AS lapsed
   WHERE lapsed.ema_id=vanemad.id OR lapsed.isa_id=vanemad.id)
    AS vanimalapsesynniaasta
  FROM isikud as vanemad) AS tabel1
  WHERE vanimalapsesynniaasta IS NOT NULL;

+---------+----------------------+
| eesnimi | vanimalapsesynniaasta |
+---------+----------------------+
| Juhan   |                 1970 |
| Marta   |                 1970 |
| Martin  |                 2003 |
| Anne    |                 2003 |
+---------+----------------------+
```

Kasutajad vastavalt huvialadele

* Koosta päring, kus iga huviala juures näidatakse ära sellega seotud kasutajate arv.

```
SELECT huviala_id, COUNT(*) AS kasutajate_arv
  FROM kasutajad_huvialad
  GROUP BY huviala_id;
```

* Koosta päring, kus iga huviala juures näidatakse ära kõik kasutajanimed, kes selle huvialaga seotud on.

```
SELECT huviala_id, GROUP_CONCAT(knimi) AS kasutanimed
  FROM kasutajad_huvialad, kasutajad
  WHERE kasutajad_huvialad.kasutaja_id=kasutajad.id
  GROUP BY huviala_id;
```

Kuna kolm aktiivset kasutajat olid omale kõik neli huviala valinud, siis ongi nad igal pool loetelus.

```
+------------+---------------------+
| huviala_id | kasutanimed         |
+------------+---------------------+
|          1 | zenja,detender,siim |
|          2 | zenja,detender,siim |
|          3 | detender,siim,zenja |
|          4 | zenja,detender,siim |
+------------+---------------------+
```

Tahtes ka huviala nime siia külge saada, on üheks võimaluseks esialgne päring muuta alampäringu abil tabeliks tabel1 ning sellega siduda huvialade tabel nõnda, et tabel1.huviala_id näitab huvialade tabeli id peale. Ning välimise SELECT-lausega siis määratakse, milliseid tulpi tegelikult näha

tahetakse.

```
SELECT huviala, kasutajanimed FROM
(SELECT huviala_id, GROUP_CONCAT(knimi) AS kasutajanimed
  FROM kasutajad_huvialad, kasutajad
  WHERE kasutajad_huvialad.kasutaja_id=kasutajad.id
  GROUP BY huviala_id) as tabel1
  JOIN huvialad ON tabel1.huviala_id=huvialad.id;
```

```
+---------------+---------------------+
| huviala       | kasutajanimed       |
+---------------+---------------------+
| Korvpall      | zenja,detender,siim |
| Jalgpall      | zenja,detender,siim |
| Ujumine       | zenja,detender,siim |
| Maratonijooks | zenja,detender,siim |
+---------------+---------------------+
```

Üleslaadimised

Lihtsaim üleslaadimise näide oli kirjas konspekti põhiosas. Juurde mõningaid vihjeid ja näiteid.

Üleslaetavad failid ei pruugi sugugi olla vaid pildifailid. Kõiksugu nimedega faile ei või me vabalt veebikataloogi paigutada. Küll aga me võime inimesele salvestamise ajal soovitud failinime pakkuda. Järgnevas näites pannakse ükskõik milline üleslaetava faili sisu kausta nimega pildid, faili sisse andmed.dat. Ja faili nimega andmed.ini kirjutatakse selle faili algne nimi. Sellisel juhul on kasutajapoolsel faili küsimisel võimalik vana nimi säilitada.

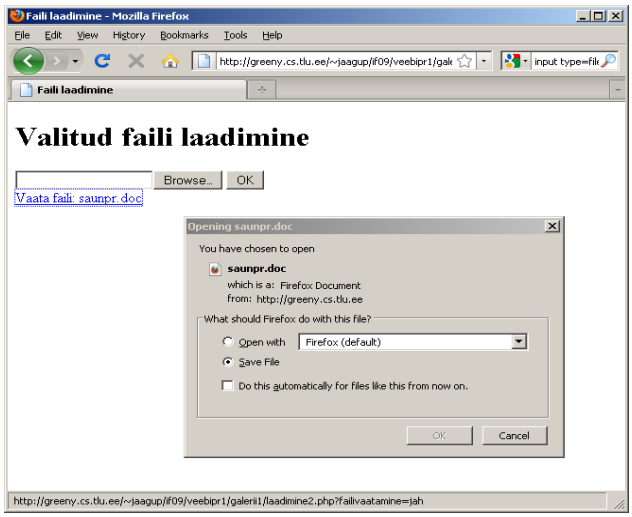
Failinime pakkumiseks on loodud päis kujul:

Content-disposition: attachment; filename=teade.rtf

kus siis teade.rtf-i asemel on pakutav nimi, mis praegusel juhul loetakse failist andmed.ini. Tingimusega kontroll, et kui on lehele seatud parameeter pildivaatamine, siis antakse tagasi eelnevalt salvestatud fail koos oma nimega.

```
<?php
  if(isSet($_FILES["valitudfail"])){
    $f=fopen("pildid/andmed.ini", "w");
    fputs($f, $_FILES["valitudfail"]["name"]);
    fclose($f);
    move_uploaded_file($_FILES["valitudfail"]["tmp_name"],
      "pildid/andmed.dat");

  }
  if(isSet($_REQUEST["failivaatamine"])){
    header("Content-disposition: attachment; filename=".
      file_get_contents("pildid/andmed.ini"));
    echo file_get_contents("pildid/andmed.dat");
    exit();
  }
?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <title>Faili laadimine</title>
    <meta http-equiv="Content-Type" content="text/html;charset=utf-8" />
  </head>
  <body>
```

```
    <h1>Valitud faili laadimine</h1>
    <form action='<?=$_SERVER["PHP_SELF"] ?>'
         method="post" enctype="multipart/form-data">
       <input type="file" name="valitudfail" />
       <input type="submit" value="OK" />
    </form>
    <?php
      if(file_exists("pildid/andmed.ini")){
        echo "<a href='$_SERVER[PHP_SELF]?failivaatamine=jah'>Vaata
          faili: ".file_get_contents("pildid/andmed.ini")."</a>";
      }
    ?>
  </body>
</html>
```



Vana nimega veebikataloogis

Päisega failinime pakkumisel on puuduseks, et veebiserver ei lisa failitüüpi (nt. image/gif) ja brauseril võib olla raskusi õige tüübi aimamisel. Selle saab küll ise ka eraldi päisega kaasa kirjutada (Content-type: image/gif), kuid ise oma koodis kõikvõimalikke tüüpe meeles pidada on tülikavõitu. Lihtsam on lasta veebiserveril vastavat tuttavat tööd teha.

Turvakaalutlustel on lubatavad laiendid piiratud. Kui laetaval failil on vastav laiend, siis ta paigutatakse kataloogi. Kui aga mitte, siis ei panda teda sinna. Edasi võib juba rahus veebikataloogis olevat faili vaadata.

```
<?php
  if(isSet($_FILES["valitudfail"])){
    $lubatud=array("gif", "jpg", "doc", "odt");
    $m=explode(".", $_FILES["valitudfail"]["name"]);
    $laiend=array_pop($m); //massiivist viimane
    if(in_array($laiend, $lubatud)){
     $f=fopen("pildid/andmed4.ini", "w");
     fputs($f, $_FILES["valitudfail"]["name"]);
     fclose($f);
     move_uploaded_file($_FILES["valitudfail"]["tmp_name"],
      "pildid/{$_FILES[valitudfail][name]}");
    } else {
       die("lubamatu laiend");
    }
  }
?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
```

```
  <head>
    <title>Faili laadimine</title>
    <meta http-equiv="Content-Type" content="text/html;charset=utf-8" />
  </head>
  <body>
     <h1>Valitud faili laadimine</h1>
     <form action='<?=$_SERVER["PHP_SELF"] ?>'
          method="post" enctype="multipart/form-data">
        <input type="file" name="valitudfail" />
        <input type="submit" value="OK" />
     </form>
     <?php
       if(file_exists("pildid/andmed4.ini")){
         echo "<a href='pildid/".
           file_get_contents("pildid/andmed4.ini")."'>Vaata
           faili: ".file_get_contents("pildid/andmed4.ini")."</a>";
       }
     ?>
  </body>
</html>
```

Pildialbum

* Koosta pildigalerii. Failid laetakse üles eraldi kataloogi ning sealt näidatakse veebilehele. Failid nummerdatakse 1.png, 2.png vastavalt kirje id-le andmebaasitabelis. Piltide juurde kuulub pealkiri.
* Salvestatakse pildi üleslaadimise ajal olnud laiend (gif, jpg/jpeg või png). Näitamise ajal on pildil õige laiend küljes, failinimeks on ikka piltide tabeli vastava pildi id-number.
* Pilte näidatakse lehel viie kaupa, nuppudega viited edasi ja tagasi.

Pildiandmete tabelis hoidmiseks tuleb siis kõigepealt vastav tabel teha. Iga pildi kohta id-number, pildi pealkiri ja laiend. Pildi vana, üleslaadimisel olnud nime ei salvestata.

```
CREATE TABLE pildid(
  id INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
  pealkiri VARCHAR(255),
  laiend VARCHAR(4)
);
```

Haldamise lehel saab valida faili. Kui vajutati lisamisnupule ja faili andmed jõudsid kohale, siis kõigepealt lisatakse rida andmebaasi – kasutaja pandud pildi pealkiri ning laiend. Loodud uue kirje id saab küsida muutujast $yhendus->insert_id. See number pannakse nüüd ka faili nimeks. Laiend jäetakse nii nagu on. Üles lubatakse laadida vaid gif, jpg ja png-lõppudega faile.

```
piltidehaldus.php
<?php
  $yhendus=new mysqli("localhost", "jaagup", "xxxxxx", "jaagup");
  if(isSet($_REQUEST["lisamine"])){
    $teatesisu="";
    if(!isSet($_FILES["pildifail"]["name"]) or
        strlen($_FILES["pildifail"]["name"])==0){$teatesisu="Probleem failiga";}
    else{
      $failiandmed=pathinfo($_FILES["pildifail"]["name"]);
      $laiend=$failiandmed["extension"];
      $lubatud=array("gif", "jpg", "png");
      if(!in_array($laiend, $lubatud)){
        $teatesisu="Lubamatu laiend";
      } else {
       $kask=$yhendus->prepare("INSERT INTO pildid (pealkiri, laiend)
          VALUES (?, ?)");
       $kask->bind_param("ss", $_REQUEST["pealkiri"],$laiend);
       $kask->execute();
       if(move_uploaded_file($_FILES["pildifail"]["tmp_name"],
         "pildid/$yhendus->insert_id.$laiend")){
```

```
          $teatesisu="fail lisatud";
        } else {
          $teatesisu="Kopeerimisprobleem";
        }
       }
     }
  }
?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <title>Piltide haldus</title>
    <meta http-equiv="Content-Type" content="text/html;charset=utf-8" />
  </head>
  <body>
     <?=$teatesisu ?>
     <h2>Pildi lisamine</h2>
     <form action="<?=$_SERVER['PHP_SELF'] ?>" method="post"
        enctype="multipart/form-data">
         <input type="file" name="pildifail"><br />
         pealkiri: <input type="text" name="pealkiri" />
         <input type="submit" name="lisamine" value="Lisa pilt" />
     </form>
  </body>
</html>
```

Piltide näitamiseks piisab andmebaasipäringust ning tsüklist, mis loob viited juba kataloogis leiduvatele piltidele.

```
       while($kask->fetch()){
        echo "<h3>$pealkiri</h3>
          <a href='pildid/$id.$laiend'><img src=
   'pildid/$id.$laiend' style='height: 200px;border: 0px' alt='' /></a>\n";
       }
```

Lehekülgede kaupa näitamiseks hoitakse eraldi muutujas $hulk kirjas piltide arvu näidataval lehel. Muutuja $algus näitab, mitmendast lehest alates pilte näitama hakatakse. Päring

```
  $kask=$yhendus->prepare("SELECT id, pealkiri, laiend FROM pildid
     ORDER BY id LIMIT $algus, $hulk");
```

annab välja pildid loendist alates elemendist $algus ning nii mitu tükki, kui palju näitab muutuja $hulk. MySQLi sõna LIMIT lihtsalt on selline õnnelik abiline.

Leheküljenumbrite kuvamiseks sobib järgmine tsükkel:

```
       for($koht=0; $koht<$kogus; $koht+=$hulk){
         if($koht==$algus){echo ($koht/$hulk+1)." ";}
         else {echo "<a href='$_SERVER[PHP_SELF]?algus=$koht'>".($koht/$hulk+1)."</a> ";}
       }
```

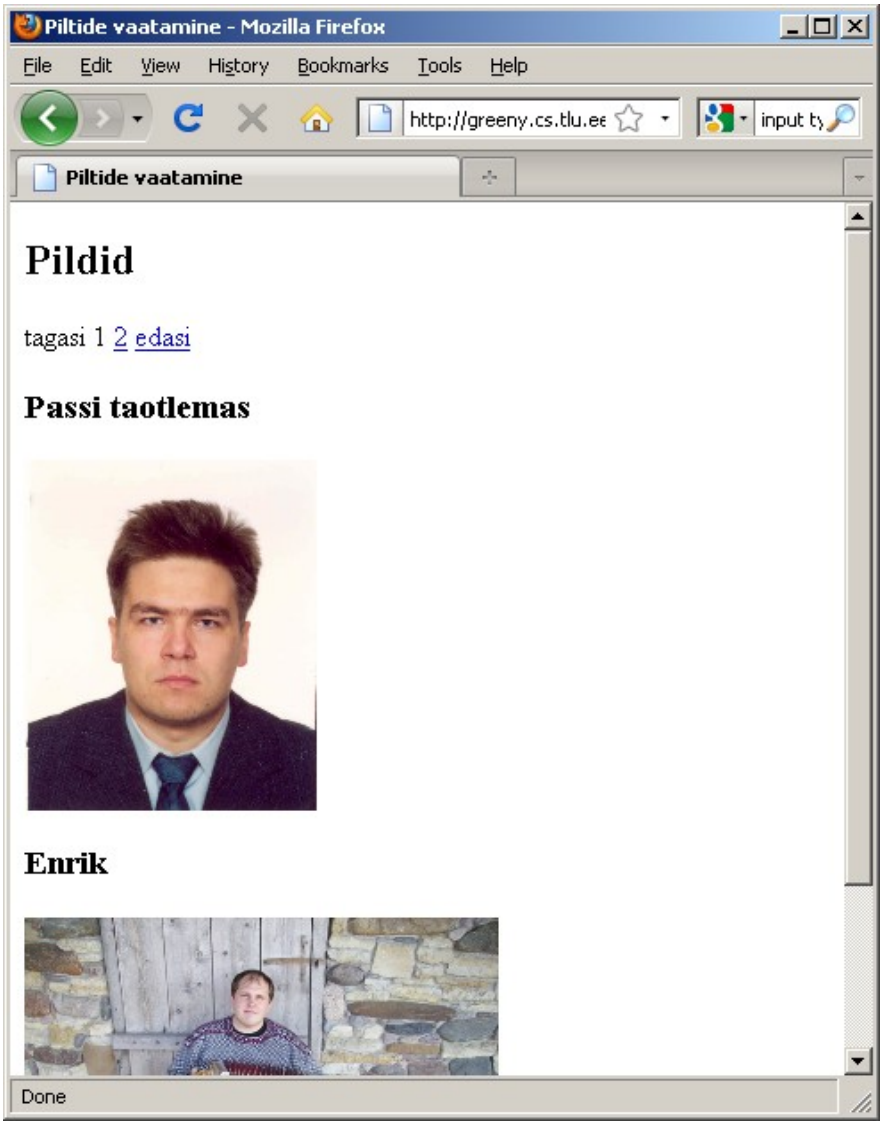
Arvutus $koht/$hulk näitab, et mitmenda leheküljega on tegemist – kui $koht on lehte alustava pildi järjekorranumber ning $hulk piltide arv lehel. Kuna selline loendamine algab nullist, siis tavainimesele tavalisemaks loendamiseks lisatakse leheküljenumbrile juurde 1. Kui on tegemist parasjagu ees oleva lehega, ehk siis $koht on võrdne lehe esimese pildi järjekorranumbrit näitava muutujaga $algus, sellisel juhul arvu viiteks ei muudeta – samale lehele pole ju vaja ehk uuesti minna. Muul juhul saab viitele vajutades sobiva alguspildi ette keerata nagu selle järjekorranumber viite parameetrina kaasa läheb.

Edasi ja tagasi viited arvutavad vastavalt leheküljе jagu pildinumbreid sobivas suunas. Kui juba alustatakse pildist 0 ehk päris algusest, siis pole põhjust tagasiviidet aktiivseks teha. Kui aga

tagantpoolt pole midagi vaadata (ehk $algus+$hulk on jõudnud piltide koguseni), siis jäetakse viide ”edasi” viitest ilma.

piltidevaatamine.php

```
<?php
  $yhendus=new mysqli("localhost", "jaagup", "xxxxxx", "jaagup");
  $kask=$yhendus->prepare("SELECT COUNT(*) FROM pildid");
  $kask->bind_result($kogus);
  $kask->execute();
  $kask->fetch();
  $kask->close();
  $algus=0;
  if(isSet($_REQUEST["algus"])){
    $algus=intval($_REQUEST["algus"]);
  }
  $hulk=2;
  $kask=$yhendus->prepare("SELECT id, pealkiri, laiend FROM pildid
     ORDER BY id LIMIT $algus, $hulk");
  $kask->bind_result($id, $pealkiri, $laiend);
  $kask->execute();
?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <title>Piltide vaatamine</title>
    <meta http-equiv="Content-Type" content="text/html;charset=utf-8" />
  </head>
  <body>
     <h2>Pildid</h2>
     <?php
       if($algus!=0){
          echo "<a href='$_SERVER[PHP_SELF]?algus=".($algus-$hulk)."'>tagasi</a> ";
       }  else {
          echo "tagasi ";
       }
       for($koht=0; $koht<$kogus; $koht+=$hulk){
         if($koht==$algus){echo ($koht/$hulk+1)." ";}
         else {echo "<a href='$_SERVER[PHP_SELF]?algus=$koht'>".($koht/$hulk+1)."</a> ";}
       }
       if($algus+$hulk<$kogus){
          echo "<a href='$_SERVER[PHP_SELF]?algus=".($algus+$hulk)."'>edasi</a> ";
       } else {
          echo "edasi ";
       }
       echo "<br />";
       while($kask->fetch()){
        echo "<h3>$pealkiri</h3>
          <a href='pildid/$id.$laiend'><img src='pildid/$id.$laiend' style='height:
200px;border: 0px' alt='' /></a>\n";
       }
     ?>
  </body>
</html>
```



Lisamine ja muutmine

Kui toimetusi rohkem, siis on nad kasulik eraldi faili paigutada, siis jääb näidatav fail lihtsamaks ja puhtamaks. Näites siis lisamine, küsimine, muutmine ka kustutamine eraldi funktsioonidena. Samuti eraldi funktsioon veebilehele saabuva sisendi töötlemiseks, kus analüüsitakse saabuvaid parameetreid ning otsustatakse, et kas ja milline toimetus tuleb käivitada.

`abifunktsioonid.php`

```
<?php
  $yhendus=new mysqli("localhost", "if09", "h9ETN", "if09_jaagup1");

  function lisaPilt(){
    global $yhendus;
    $teatesisu="";
    if((!isSet($_FILES["pildifail"]["name"])) or
strlen($_FILES["pildifail"]["name"])==0){$teatesisu="Fail puudu";}
    else{
      $failiandmed=pathinfo($_FILES["pildifail"]["name"]);
      $laiend=$failiandmed["extension"];
      $lubatudlaiendid=array("gif", "jpg", "jpeg", "png");
```

```
    if(!in_array($laiend, $lubatudlaiendid)){
      return "Lubamatu laiend";
    }
    $kask=$yhendus->prepare("INSERT INTO pildid (pealkiri, laiend)
       VALUES (?, ?)");
    $kask->bind_param("ss", $_REQUEST["pealkiri"],$laiend);
    $kask->execute();
    if(move_uploaded_file($_FILES["pildifail"]["tmp_name"],
       "pildid/$yhendus->insert_id.$laiend")){
      $teatesisu="fail lisatud";
    } else {
      $teatesisu="Kopeerimisprobleem";
    }
  }
  return $teatesisu;
}

function kysiPildid(){
  global $yhendus;
  $kask=$yhendus->prepare("SELECT id, pealkiri, laiend FROM pildid");
  $kask->bind_result($id, $pealkiri, $laiend);
  $kask->execute();
  $hoidla=array();
  while($kask->fetch()){
    $pilt=new stdClass();
    $pilt->id=$id;
    $pilt->pealkiri=$pealkiri;
    $pilt->laiend=$laiend;
    array_push($hoidla, $pilt);
  }
  return $hoidla;
}

function kysiPilt($id){
  global $yhendus;
  $kask=$yhendus->prepare("SELECT pealkiri, laiend FROM pildid WHERE id=?");
  $kask->bind_param("i", $id);
  $kask->bind_result($pealkiri, $laiend);
  $kask->execute();
  $kask->fetch();
  $pilt=new stdClass();
  $pilt->id=$id;
  $pilt->pealkiri=$pealkiri;
  $pilt->laiend=$laiend;
  $kask->close();
  return $pilt;
}

function muudaPilt($id, $pealkiri, $failimas){
   global $yhendus;
   $pilt=kysiPilt($id);
   if(strlen($failimas["name"])>0){
     unlink("pildid/$pilt->id.$pilt->laiend");
     $failiandmed=pathinfo($failimas["name"]);
     $laiend=$failiandmed["extension"];
     move_uploaded_file($failimas["tmp_name"], "pildid/$id.$laiend");
     $kask=$yhendus->prepare("UPDATE pildid SET pealkiri=?, laiend=? WHERE id=?");
     $kask->bind_param("ssi", $pealkiri, $laiend, $id);
   } else {
     $kask=$yhendus->prepare("UPDATE pildid SET pealkiri=? WHERE id=?");
     $kask->bind_param("si", $pealkiri, $id);
   }
   $kask->execute();
}

function kustutaPilt($id){
  global $yhendus;
  $pilt=kysiPilt($id);
  @unlink("pildid/$pilt->id.$pilt->laiend");
  $kask=$yhendus->prepare("DELETE FROM pildid WHERE id=?");
  $kask->bind_param("i", $id);
```

```
    $kask->execute();
  }

  function tootleSisend(){
    if(isSet($_REQUEST["lisamine"])){
      return lisaPilt();
    }
    if(isSet($_REQUEST["muutmisid"])){
      muudaPilt($_REQUEST["muutmisid"], $_REQUEST["pealkiri"], $_FILES["pildiandmed"]);
    }
    if(isSet($_REQUEST["kustutusid"])){
      kustutaPilt($_REQUEST["kustutusid"]);
    }
    return "";
  }

?>
```

Põhifaili juures saab sel juhul mõnekümne koodireaga piiluda. Abifunktsioonide failist loetakse sisse vajalikud käsklused. Palutakse töödelda sisend ning küsitakse näidatavate piltide loetelu. Edasi vaja nad ainult lehele mugavalt paigutada ning muutmise/kustutamise tarbeks viited külge paigutada.

piltidehaldus2.php

```
<?php
  require("abifunktsioonid.php");
  $teatesisu=tootleSisend();
  $pildid=kysiPildid();
?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <title>Piltide haldus</title>
    <meta http-equiv="Content-Type" content="text/html;charset=utf-8" />
  </head>
  <body>
     <?=$teatesisu ?>
     <?php
       if(isSet($_REQUEST["mid"])){
          $pilt=kysiPilt($_REQUEST["mid"]);
          echo "<h2>Pildi muutmine</h2>
           <form action='$_SERVER[PHP_SELF]' method='post' enctype='multipart/form-data'>
            <input type='hidden' name='muutmisid' value='$_REQUEST[mid]' />
            Pealkiri: <input type='text' name='pealkiri' value='$pilt->pealkiri' /><br />
            <img src='pildid/$pilt->id.$pilt->laiend' alt='' /><br />
            <input type='file' name='pildiandmed' />
            <input type='submit' value='muuda' />
           </form>
          ";

       }
     ?>
     <h2>Pildi lisamine</h2>
     <form action="<?=$_SERVER['PHP_SELF'] ?>" method="post"
        enctype="multipart/form-data">
         <input type="file" name="pildifail"><br />
         pealkiri: <input type="text" name="pealkiri" />
         <input type="submit" name="lisamine" value="Lisa pilt" />
     </form>
     <?php
       foreach($pildid as $pilt){
         echo "<div>
           <h3>$pilt->pealkiri</h3>
           <a href='$_SERVER[PHP_SELF]?kustutusid=$pilt->id'>k</a>
           <a href='$_SERVER[PHP_SELF]?mid=$pilt->id'>m</a>
           <img src='pildid/$pilt->id.$pilt->laiend' alt='' />
         </div>";
       }
```

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

Kui kood töös, siis tuleb vaid sobivad pildid galeriisse üles seada.