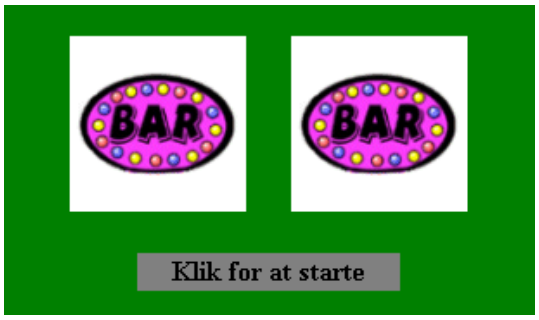
ENARMET TYVEKNÆGT

En enarmet tyveknægt er en spillemaskine, hvor man trækker i et håndtag, og nogle hjul drejer rundt. Normalt er der 3 hjul i en enarmet tyveknægt. Når hjulene efter nogen tid stopper, har man typisk vundet, hvis alle 3 hjul viser det samme symbol. Vi skal lave en hjemmeside med et spil, der fungerer som en enarmet tyveknægt. Vi skriver hjemmesiden i HTML. Layoutet laves med CSS, og selve spillet programmeres i Javascript. Først designer vi en spilleplade, hvor vi i første omgang nøjes med 2 hjul. Spillepladen kunne se ud som vist herunder:



"Bar"-symbolet er som regel det symbol, der udløser højst gevinst, når man har en række med dem.

Grafikfilerne

Vi har 7 grafikfiler med symboler til rådighed: kirsebær, ananas, appelsin, blomme, banan, æble og bar:


kirsebaer.gif


ananas.gif


appelsin.gif


blomme.gif


banan.gif


aeble.gif

bar.gif

Alle grafikbilleder fylder 100 pixels i højden og bredden og indsættes på en hjemmeside med en `<img>`-kode, f.eks. `<img src="kirsebaer.gif">`

HTML-koden

Vi starter med at anbringe 2 bar-symboler på spillepladen. Begge grafikbilleder skal kunne udskiftes med andre grafikbilleder undervejs i spillet. For at javascript-kode skal kunne udskifte billeder, skal vi indføre *id*-variablen med et passende navn i `<img>`-koden, når vi skriver HTML-koden. Bar-symbolet kalder vi `hjul1`, og bar-symbolet til højre kalder vi `hjul2`. Koden for bar-symbolet til venstre skal der derfor skrives sådan:

```
<img src="bar.gif" id="hjul1">
```

Vi anbringer de 2 bar-symboler ved siden af hinanden ved hjælp af en tabel. For at bar-symbolerne ikke skal placeres lige op ad hinanden, anbringer vi en tom celle med en bredde på 20 pixels imellem de 2 bar-symboler. Her er HTML-koden til spillepladen. Stylesheetet anbringes i en fil med navnet *tyv.css*, og javascript-koden anbringes i en fil med navnet *tyv.js* som angivet i koderne `<link>` og `<script>`:

```
<html>

<head>
  <title>Enarmet tyveknægt</title>
  <link rel="stylesheet" type="text/css" href="tyv.css">
  <script type="text/javascript" src="tyv.js"></script>
</head>

<body>
  <br><br><br>
  <table align="center">
    <tr>
      <td width="100" class="hjul">
        <img src="bar.gif" id="hjul1">
      </td>
      <td width="20">
      </td>
      <td width="100" class="hjul">
        <img src="bar.gif" id="hjul2">
      </td>
    </tr>
  </table>
  <br>
  <table align="center">
    <tr>
      <td width="150" class="start" onClick="spil()">
        <b>Klik for at starte</b>
      </td>
    <tr>
  </table>
</body>

</html>
```

Klasserne *hjul* og *start* omtales nærmere i afsnittet om stylesheetet. Det grå felt under spillepladen på side 1 er en tabelcelle med grå baggrund. Hvis man klikker i dette felt (p.g.a. hændelsen "`onClick`"), skal der aktiveres noget javascript-kode i funktionen *spil*.

Stylesheetet

Farverne defineres i et stylesheet med navnet *tyv.css* som angivet i koden `<link>`. Her er stylesheetet:

```
body      {background-color: green;
           color: black;
          }

td.hjul   {background-color: white;
           text-align: center;
          }

td.start  {background-color: gray;
           text-align: center;
          }
```

Tekstfarven bliver sort, og baggrundsfarven bliver grøn. Vi definerer 2 klasser, *hjul* og *start*, for koden `<td>` (tabelcelle). Ved klassen *hjul* bliver baggrundsfarven hvid, og tekst og billeder centreres i cellen. Ved klassen *start* bliver baggrundsfarven grå, og tekst og billeder centreres ligeledes i cellen. For at bruge disse klasser skal vi indsætte `class="hjul"` og `class="start"` i `<td>`-koderne i HTML-dokumentet. Hvis vi ikke bruger klasser, vil alle tabelceller få samme baggrundsfarve.

Javascript-koden

Vi skal programmere et spil, som vælger 2 grafikbilleder. Vi nummererer billederne. I første omgang bruger vi kun symbolerne for kirsebær, ananas og bar. Vi lader tallet 0 svare til kirsebær, tallet 1 svare til ananas og tallet 2 svare til bar. Vi skal altså vælge 2 tilfældige tal blandt tallene 0, 1 og 2. Vi gemmer de 2 tilfældige tal i variablerne *felt1* og *felt2*:

```
felt1 = Math.floor(3*Math.random())
felt2 = Math.floor(3*Math.random())
```

Math.random vælger et tilfældigt tal i intervallet fra 0,000000... til 0,999999... Når man ganger med 3, bliver tallet i intervallet fra 0,000000... til 2.999999... *Math.Floor* fjerner decimalerne, så man får et af tallene 0, 1 eller 2. Tallet i variablen *felt1* skal vælge filnavnet på symbolet i det venstre hjul:

```
if(felt1 == 0) {
  hjul1.src = 'kirsebaer.gif'
}
else if(felt1 == 1) {
  hjul1.src = 'ananas.gif'
}
else {
  hjul1.src = 'bar.gif'
}
```

Vi skal skrive tilsvarende kode for hjulet til højre. For at undgå at skrive den samme kode 2 gange samler vi *if*-sætningen i en funktion, som vi kalder *figur*. Vi skal så bare kalde (dvs. aktivere) funktionen *figur* i alt 2 gange:

```
hjul1.src = figur(felt1)
hjul2.src = figur(felt2)
```

Funktionen *figur* kommer til at se sådan ud:

```
function figur(nr) {
  if(nr == 0) {
    return 'kirsebaer.gif'
  }
  else if(nr == 1) {
    return 'ananas.gif'
  }
  else if(nr == 2) {
    return 'bar.gif'
  }
}
```

Tallet i variablerne *felt1* og *felt2* bliver overført til variablen *nr* i funktionen *figur*. Ud over at vi sparer kode, bliver programmet også mere overskueligt af at blive opdelt i funktioner. Spillet skal give gevinst ved 2 ens symboler eller ved et kirsebær i venstre hjul alene. Her er gevinstlisten:

- 2 kirsebær: 20 kr
- 1 kirsebær i venstre hjul: 10 kr
- 2 ananas: 40 kr
- 2 bar: 100 kr

Vi laver en funktion med navnet *vaerdi*, som skal returnere gevinstens størrelse:

```
function vaerdi(felt1,felt2)
  if(felt1 == 0 && felt2 == 0) {
    return 20
  }
  else if(felt1 == 0) {
    return 10
  }
  else if(felt1 == 1 && felt2 == 1) {
    return 40
  }
  else if(felt1 == 2 && felt2 == 2) {
    return 100
  }
  else {
    return 0
  }
}
```

Vi gemmer resultatet fra funktionen i en variabel med navnet *gevinst*:

```
gevinst = vaerdi(felt1,felt2)
```

Derefter kan vi tælle gevinsterne sammen med en anden variabel, *total*:

```
total = total + gevinst - 10
```

Denne sætning skal læses: Indholdet i variablen *total* bliver lig med den gamle værdi af *total* + værdien af variablen *gevinst* – 10 kr. De 10 kr er prisen for ét spil. Vi skal så sætte *total* til 0, inden spillet starter:

```
total = 0
```

Herunder er hele javascript-koden. Det er funktionen *spil*, som aktiveres, når man klikker på det grå felt på spillepladen på grund af hændelsen *onClick* i HTML-dokumentet.

```
function figur(nr) {
  if(nr == 0) {
    return 'kirsebaer.gif'
  }
  else if(nr == 1) {
    return 'ananas.gif'
  }
  else if(nr == 2) {
    return 'bar.gif'
  }
}

function vaerdi(felt1,felt2) {
  if(felt1 == 0 && felt2 == 0) {
    return 20
  }
  else if(felt1 == 0) {
    return 10
  }
  else if(felt1 == 1 && felt2 == 1) {
    return 40
  }
  else if(felt1 == 2 && felt2 == 2) {
    return 100
  }
  else {
    return 0
  }
}

function spil() {
  ok = true
  total = 0
  while(ok == true) {
    felt1 = Math.floor(3*Math.random())
    felt2 = Math.floor(3*Math.random())
    hjul1.src = figur(felt1)
    hjul2.src = figur(felt2)
    gevinst = vaerdi(felt1,felt2)
    total = total + gevinst - 10
    ok = confirm('Din gevinst er '+gevinst+' kr\nDin saldo er '+total+
                 ' kr\n\nVil du prøve igen?')
  }
}
```

While-løkken sørger for, at spillet kører, så længe variablen *ok* har værdien *true*. Denne variabel får sin værdi fra en *confirm*-boks. Hvis man klikker på "O.k." i *confirm*-boksen, får *ok* værdien *true*. Klikker man på "Annullér", får *ok* værdien *false*.

I teksten til confirm-boksen står der `\n` et par gange. Dette giver et linjeskift i *confirm*-boksen. Her er resultatet af et spil:

Opgaver

1) Udvid spillet, så man bruger alle 7 grafikfiler. Vælg selv passende gevinster.

2) Udvid spillet, så der bliver 3 hjul.