Få tag i element

- Vi har flera metoder för att få tag i referenser till element.
  - `getElementById // Element`
  - `getElementsByClassName // HTMLCollection`
  - `getElementsByName // NodeList`
  - `getElementsByTagName // HTMLCollection`
  - `querySelector // Element`
  - `querySelectorAll // NodeList`
- När vi har en referens till ett element kan vi göra saker med dem.

Navigera i DOM-trädet

- Vi har olika metoder för att få tag i referenser till relaterade element utifrån ett element. Säg att vi har hämtat ett element x med getElementById.
  - `Node.parentNode`
  - `Node.firstChild`
  - `Node.lastChild`
  - `Node.childNodes`
  - `Node.nextSibling`
  - `Node.parentElement`
  - `Node.previousSibling`
  - `Node.removeChild`
- Dessa metoder kan vi använda för att navigera i trädet för att få referenser till de element vi är intresserade av i förhållande till elementet x.

Styra utseende med JS

- Man kan styra enskilda style properties.
  - `document.getElementById("something").style.backgroundColor = "#ccdd33";`
- https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Properties_Reference

CSS vs JS

- Vi kan t ex sätta en styling på ett aktuellt element när det kommer i fokus.
  - ```
    document
      .getElementById("something")
      .addEventListener("focus", function() {
        document
          .getElementById("something")
          .style
          .backgroundColor = "#ccdd33"
        })
    ```
- Det kan vi göra med CSS också.
  - ```
    #something:focus {
      background-color: ”#ccdd33";
    }
    ```
- Med JS är det enklare att styra andra elements utseende.
- https://jsfiddle.net/emmio_micke/r8uz1bdm/

Styra utseende med JS

- Man kan även lägga till eller ta bort klasser.
  - `document.getElementById("div1").classList.add("classToBeAdded");`
  - `document.getElementById("div1").classList.remove("classToBeRemoved");`
- Man kan toggla klasser.
  - `document.getElementById("div1").classList.toggle("classToBeToggled");`
- Och kontrollera om en klass finns.
  - ```
    if (document.getElementById("div1").classList.contains("classToBeChecked"))
    {
      // Do something
    }
    ```
- https://www.kirupa.com/html5/setting_css_styles_using_javascript.htm
    ```

Colgroup

- Colgroup låter oss sätta vissa egenskaper för kolumner i en html-tabell.

- https://developer.mozilla.org/en-US/docs/Web/HTML/Element/colgroup

```
<table>
    <caption>Superheros and sidekicks</caption>
    <colgroup>
        <col>
        <col span="2" class="batman">
        <col span="2" class="flash">
    </colgroup>
    <tr>
        <td> </td>
        <th scope="col">Batman</th>
        <th scope="col">Robin</th>
        <th scope="col">The Flash</th>
        <th scope="col">Kid Flash</th>
    </tr>
    <tr>
        <th scope="row">Skill</th>
        <td>Smarts</td>
```

Output

	Batman	Robin	The Flash	Kid Flash
Skill	Smarts	Dex, acrobat	Super speed	Super speed

Superheros and sidekicks

Styra utseende med JS

- I era Yatzy-spel:
    - Skapa en klass som gör det tydligt att en ”poäng” (t ex raden för fyrtal eller kåk) är otillgänglig.
        - Skapa en knapp som togglar ett par rader som tillgängliga / otillgängliga.
    - Visa vilken spelare som är aktiv.
        - https://jsfiddle.net/emmio_micke/njtdfrkz/

JS DOM

- JS låter oss manipulera DOM:en på olika sätt.
- Hittills har vi t ex ändrat text-innehåll i rubriker, ändrat på styling och läst in input-komponenters värden.
- Vi kan också lägga till, radera och förändra hela element.

Lägga till element

- createElement skapar ett element men gör ingenting med det som påverkar vår DOM och alltså inte vår sida. Det blir bara ett nytt objekt.

- appendChild lägger till ett element sist till barnen till ett annat element.

- ```
  let p = document.createElement("p");
  document.body.appendChild(p);
  ```
  ```

Lägga till element

- Obs, man kan bara lägga till ett specifikt element en gång. https://jsfiddle.net/emmio_micke/oLrun6ga/

- ```
  let p = document.createElement("p");
  document.body.appendChild(p);
  ```

- *Uppgift:* Skapa en sida med en div med id article.
  `<div id="article"></div>`

  - I denna div ska ni med hjälp av JS lägga till tre nya stycken.
  ```

getAttribute / setAttribute

```
// getAttribute
var div1 = document.getElementById("div1");
var align = div1.getAttribute("align");

alert(align); // shows the value of align for the element
              // with id="div1"

// setAttribute
var b = document.querySelector("button");

b.setAttribute("name", "helloButton");
b.setAttribute("disabled", "");

// https://jsfiddle.net/emmio_micke/dyn35czk/
```

Ta bort element

- remove låter dig ta bort ett element.
- *Uppgift:* I den föregående uppgiften, ta med hjälp av JS bort det sista stycket.

Övningar

- Address Book
  - https://yh.pingpong.se/courseId/11861/content.do?id=5221094
- Todo List
  - https://yh.pingpong.se/courseId/11861/content.do?id=5209549

CSS Preprocessors

T ex SCSS & Less

- Är CSS ett programmeringsspråk?
- Preprocessorer används för att utöka möjligheterna med CSS, t ex:
  - Variabler
  - Nesting
  - Mixins
  - Loopar
  - Arv
- Olika för olika preprocessorer
- Kompileras till css

SCSS
Variabler

```scss
$primary-color: #3bbfce;
$margin: 16px;

.content-navigation {
  border-color: $primary-color;
  color: darken($primary-color, 10%);
}

.border {
  padding: $margin / 2;
  margin: $margin / 2;
  border-color: $primary-color;
}
```

```scss
.content-navigation {
  border-color: #3bbfce;
  color: #2b9eab;
}

.border {
  padding: 8px;
  margin: 8px;
  border-color: #3bbfce;
}
```

SCSS
Nesting

```scss
table.hl {
  margin: 2em 0;
  td.ln {
    text-align: right;
  }
}

li {
  font: {
    family: serif;
    weight: bold;
    size: 1.3em;
  }
}
```

```scss
table.hl {
  margin: 2em 0;
}
table.hl td.ln {
  text-align: right;
}

li {
  font-family: serif;
  font-weight: bold;
  font-size: 1.3em;
}
```

SCSS
Mixins

```scss
@mixin table-base {
  th {
    text-align: center;
    font-weight: bold;
  }
  td, th {
    padding: 2px;
  }
}

#data {
  @include table-base;
}
```

```scss
#data th {
  text-align: center;
  font-weight: bold;
}
#data td, #data th {
  padding: 2px;
}
```

SCSS
Loopar

```scss
$squareCount: 3
@for $i from 1 through $squareCount
  #square-#{$i}
   background-color: red
   width: 50px * $i
   height: 120px / $i
```

```scss
#square-1 {
  background-color: red;
  width: 50px;
  height: 120px;
}

#square-2 {
  background-color: red;
  width: 100px;
  height: 60px;
}

#square-3 {
  background-color: red;
  width: 150px;
  height: 40px;
}
```

SCSS
Arv

```
.error
  border: 1px #f00
  background: #fdd

.error.intrusion
  font-size: 1.3em
  font-weight: bold

.badError
  @extend .error
  border-width: 3px
```

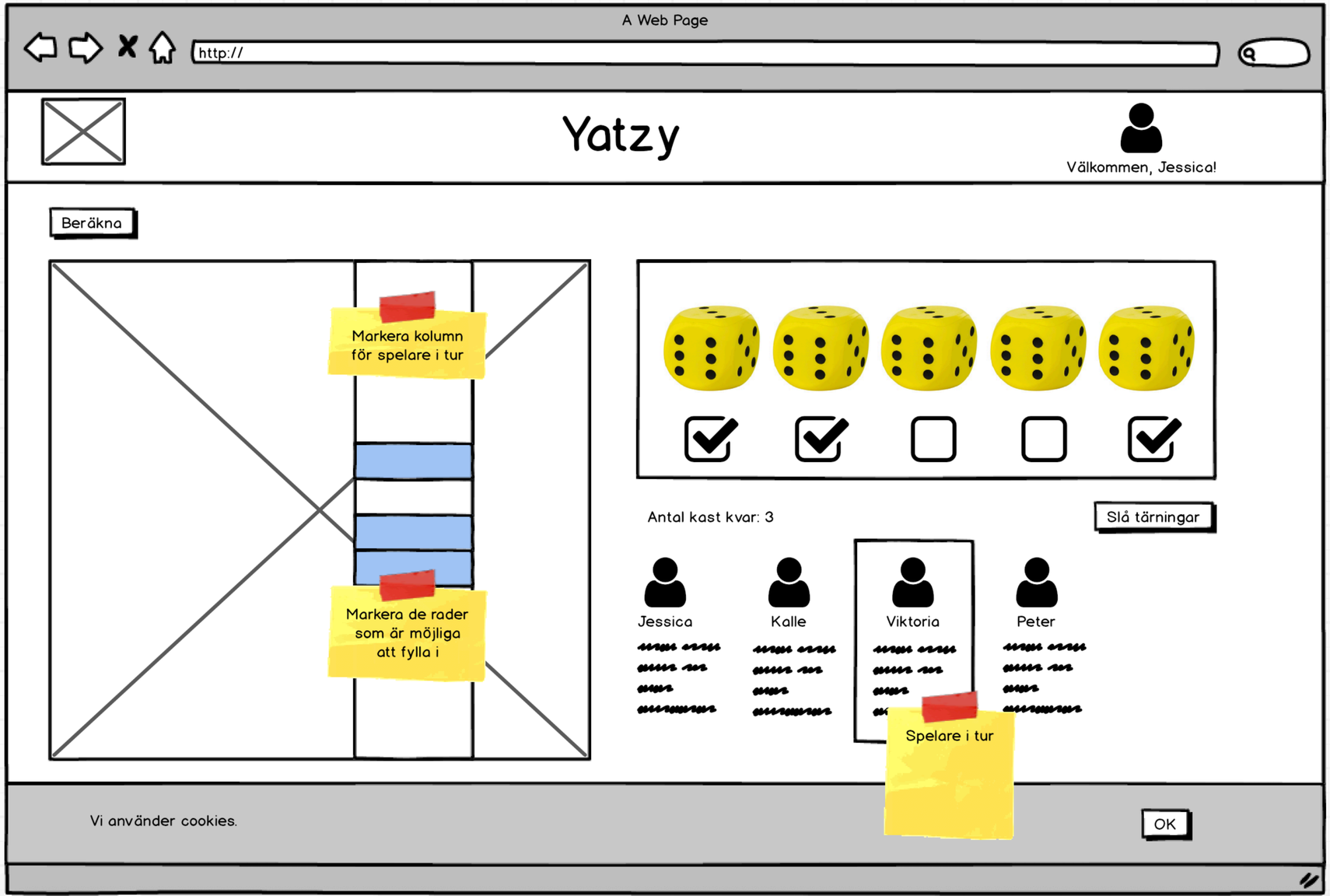
```
.error, .badError {
  border: 1px #f00;
  background: #fdd;
}

.error.intrusion,
.badError.intrusion {
  font-size: 1.3em;
  font-weight: bold;
}

.badError {
  border-width: 3px;
}
```

SCSS

- *Uppgift:* Installera stöd för SCSS i Visual Studio Code.

- https://code.visualstudio.com/docs/languages/css#_transpiling-sass-and-less-into-css

Yatzy - layout



Yatzy

- Gör ett tärningsformulär med fem text-rutor, fem kryssrutor och en knapp.
  - När man klickar på knappen ska alla rutorna få ett nytt tal (simulera tärningskast).
  - *Extra uppgift*: De tärningar som är ”kryssade” ska inte få ett nytt tal.
  - *Extra uppgift:* Håll reda på hur många kast spelaren har kvar. (Tre från början.)
  - *Extra uppgift:* Fundera på hur man skulle kunna göra för att få till bilden nedan, att visa en tärning med rätt antal ögon istället för ett tal i en textruta.
  - Extrauppgift: Gör en funktion som tar en array med fem tal som parameter och returnerar sant om talen innehåller en kåk. (3 av samma + 2 av samma)

```
var slump = Math.random();
console.log(slump);
```