# Labb 4 – Interaktivitet II

Multimedia 7.5 hp VT-14

## Introduktion

I denna laboration kommer vi att arbeta mer med javascript för att skapa interaktivitet. Tanken är nu att vi fördjupar oss i alla delarna – javascript, html såväl som css. Men under denna labb arbetar du med fördel med jQuery. När man jobbar med nya tekniker är det ofta troligt att man behöver stanna upp och söka information, så var inte rädd för att aktivt använda internet till att leta information.

## Inlämning

Din inlämning ska bestå av en .zip-fil eller .rar-fil (inga andra komprimeringsformat är tillåtna!) innehållandes följande (med följande struktur):

- labb2_fornamn_efternamn.zip
  - uppgift1 (mapp)
    - buttons.js
    - buttons.css
    - index.html
    - readme.txt (Innehåller namn och versionsnr. på den webbläsare du använt)
  - uppgift2 (mapp)
    - memory.js
    - memory.css
    - index.html
    - readme.txt (Innehåller namn och versionsnr. på den webbläsare du använt)

### Kodstandard

- Ingen Javascript eller CSS ska placeras "inline" i något htmldokument. All javascript-kod skall alltså skrivas i `.js`-filer och css i `.css`-filer.
- All kod ska vara korrekt indenterad! (Läs mer om indentering på htmlhunden om du är osäker)

- Ladda in kända externa bibliotek (i detta fall jQuery) ifrån en CDN (ex: jQuery CDN) istället för att spara biblioteket i ditt projekt.

# Uppgifter

Nedan följer uppgifterna som resulterar i inlämningarna ovan.

## Uppgift 1 Generella event-handlers

Denna första uppgift går mest ut på att förbereda dig inför nästa. Uppgiften går ut på att skriva en generellare `event handler`. En `event handler` är alltså en funktion som körs när ett `event` avfyras.

Uppgiften går ut på att vi ska skapa ett antal knappar med olika texter. När användaren klickar på en av knapparna, ska texten för denna knapp visas upp på en annan plats på sidan, exempelvis i en `<p>`-tagg.

### Förslag på tillvägagångssätt

1. Skapa en HTML, en CSS, en JS-fil, och "koppla ihop" dem.
2. Skapa ett antal (minst tre) knappar (t.ex. med `<a>`-element) med olika text, i HTML:en.
3. Skapa en placeholder för knapp-texten (t.ex. ett `<p>`-element) där texten för den klickade knappen ska visas.
4. Applicera m.h.a. `jQuery` en `event listener` för klick-eventet på alla dessa knappar. Låt alla knapparna använda samma `event handler`
5. Det finns med `jQuery` flera sätt att, i en `event handler`, avgöra vilket element som har avfyrat ett `event`. Sök rätt på något och se till att texten för respektive knapp visas.

### Krav

- Du får i denna uppgift alltså bara använda en enda `event handler`.
- Din uppgift måste vara resonabelt välstilad m.h.a. CSS.

## Uppgift 2 Spelet Memory

Denna uppgift går ut på att skapa en enklare variant av spelet memory. Spelplanen består av ett antal bilder, i par. Spelet går ut på att man ska välja två bilder, om de två bilderna är likadana så får man ta bort dem ifrån spelplanen. När alla bilder är borta ifrån spelplanen är spelet över och man har vunnit!

I vanliga fall spelar man med uppochnedåtvända bilder, för att spelet förstås inte ska vara busenkelt. Detta kommer vi att strunta i när vi bygger vårt eget Memory. Alla brickor i ditt memory-spel ska alltså hela tiden vara "uppåtvända". Man ser alltså alltid alla bilder, och spelet går helt enkelt ut på att klicka på par.

### Krav

- Din spelplan måste vara minst 3x2
- När en korrekt matchning är gjord ska båda brickorna tas bort
- När en inkorrekt matching görs ska ett meddelande visas för användaren
- När spelet är slut ska ett meddelande visas för användaren
- De bilder du använder måste vara av rimlig storlek, både i dimensioner och i "tyngd".
- Din uppgift måste vara resonabelt välstilad m.h.a. CSS.

### Förslag på tillvägagångssätt

1. Skapa en HTML, en CSS, en JS-fil, och "koppla ihop" dem.
2. Skriv ut dina bilder i din HTML.
3. Hitta ett sätt att identifiera par, t.ex. genom `ID`:n, `class`:er, och/eller `src`-attributet.
4. Fundera över vidare steg som behöver tas och bryt ned i delproblem.