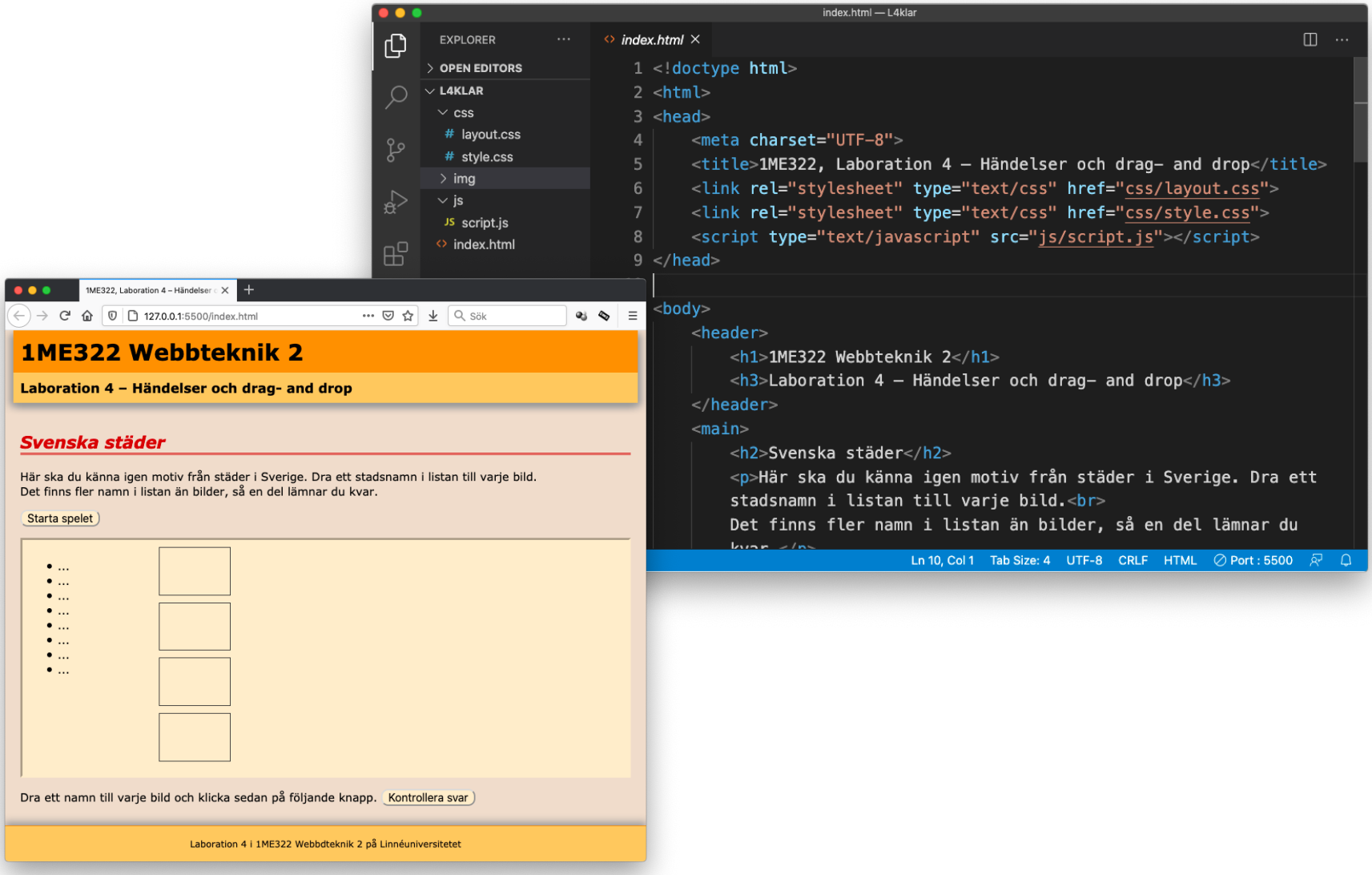
Laboration 4
Händelser och drag and drop

1M322 Webbteknik 2, 7,5hp

Medieteknik

1. Öppna mappen L4

Öppna mappen *L4* i Visual Studio Code (VSC) och öppna filen *index.html* i Live Server.



2. Gränssnitt och filerna

I denna laboration ska du skapa en applikation där det väljs ord och bilder slumpmässigt. I koden och bildfilerna är det svenska städer, men det går att byta ut orden och bilderna till vad som helst. Funktionaliteten kommer ändå bli densamma.

Det väljs åtta ord och fyra bilder, dvs åtta stadsnamn och bilder från fyra städer.
Användaren ska sedan dra namnen till rätt bilder. Därefter kan man kontrollera hur många rätt man har.

Börja med att studera gränssnittet och filerna

- Öppna *index.htm* i webbläsaren.
  - På webbsidan visas en lista med åtta punkter, där det nu endast visas ... på varje rad.
  - Det finns också fyra bildrutor, för de bilder som ska väljas.
  - Till höger om dessa finns utrymme för de ord som ska dras dit, de korrekta svaren samt ett *img*-element för en förstorad bild.
    - Dessa delar är dock tomma från början, så därför syns inget där.
  - Det finns också två knappar och ett utrymme för meddelande längst ner.
- Öppna *index.htm* i editorn.
  - Identifiera de olika delarna i gränssnittet.
    - En *ul*-lista med *li*-element för orden.
    - *img*-element för bilderna.
    - *p*-element för ord och korrekta svar intill bilderna.
    - Många av dessa element har *id*- eller *class*-attribut, för att kunna referera till dem.
- Öppna *style.css* i editorn
  - Studera CSS-koden och vilka delar i HTML-koden som det refereras till.
- Öppna filen *script.js* i editorn.
  - I *script.js* finns det en del kod inlagd redan. Det är sådant som du redan tränat på i tidigare laborationer, så för att underlätta ditt arbete i denna laboration finns redan de globala variablerna och skal till funktionerna inlagda.
    - Studera den kod som finns och läs de förklarande kommentarerna.

Då programmet är klart, ser du ut så här.

3a. Initiera globala variabler för element i gränssnittet

I *init*-funktionen ska du nu ta fram referenser till olika delar i gränssnittet och spara i globala variabler.

Under kommentaren *"// Referenser till element i gränssnittet"* lägger du in kod för följande:

- Ta fram en referens till *ul*-listan med orden och spara i variabeln *wordListElem*.
  - Med *getElementsByTagName* får man alltid en array, så för att få det första och i detta fall enda *ul*-elementet, måste man indexera med 0.

```
wordListElem = document.getElementById("wordList").getElementsByTagName("ul")[0];
wordElems = document.getElementById("wordList").getElementsByTagName("li");
```

- Ta fram referenser till alla *li*-element i listan. Det blir en array som sparas i variabeln *wordElems*.
  - I detta fall ska du inte indexera.

- Ta fram en array med referenser till *img*-elementen inom *"imgList"*. Spara i variabeln *imgElems*.

- Ta fram referenser till *p*-elementen för användarens svar och *p*-elementen för rätt svar.
  - Använd *getElementsByClassName*.
  - Detta är något nytt som du inte gjort tidigare, så den koden visas i bilden här intill.

```
answerElems = document.getElementsByClassName("userAnswer");
correctElems = document.getElementsByClassName("correctAnswer");
```

- Ta fram en referens till *img*-elementet för den stora bilden med id *"largeImg"* och spara i *largeImgElem*.

3b. Händelsehanterare

Nu ska du lägga på händelsehanterare på knapparna och de små bilderna, dvs du ska koppla dem till funktioner.

Under kommentaren *"// Lägg på händelsehanterare"* skriver du kod för följande:

- Använd funktionen *addEventListener*, för att lägga på händelsehanterare.
  - Se exempel för de båda knappar som redan finns inlagt i koden.
- Gå igenom de små bilderna i *imgElems* i en loop och lägg på följande två händelsehanterare på varje element:
  - *"mouseenter"* och funktionen *showLargeImg*.
  - *"mouseleave"* och funktionen *hideLargeImg*.

Testkod

Nu ska du testa i webbläsaren, men då får du först lägga in tillfälliga meddelanden i de fyra funktionerna.

- I funktionerna *startGame*, *checkAnsers*, *showLargeImg* och *hideLargeImg* lägger du in en programsats, så att du skriver något i meddelandefältet, dvs det du refererar till från *msgElem*.
  - Glöm inte *innerHTML*.
  - Skriv t.ex. *"start"*, *"check"*, *"enter"* respektive *"leave"* i de fyra funktionerna.

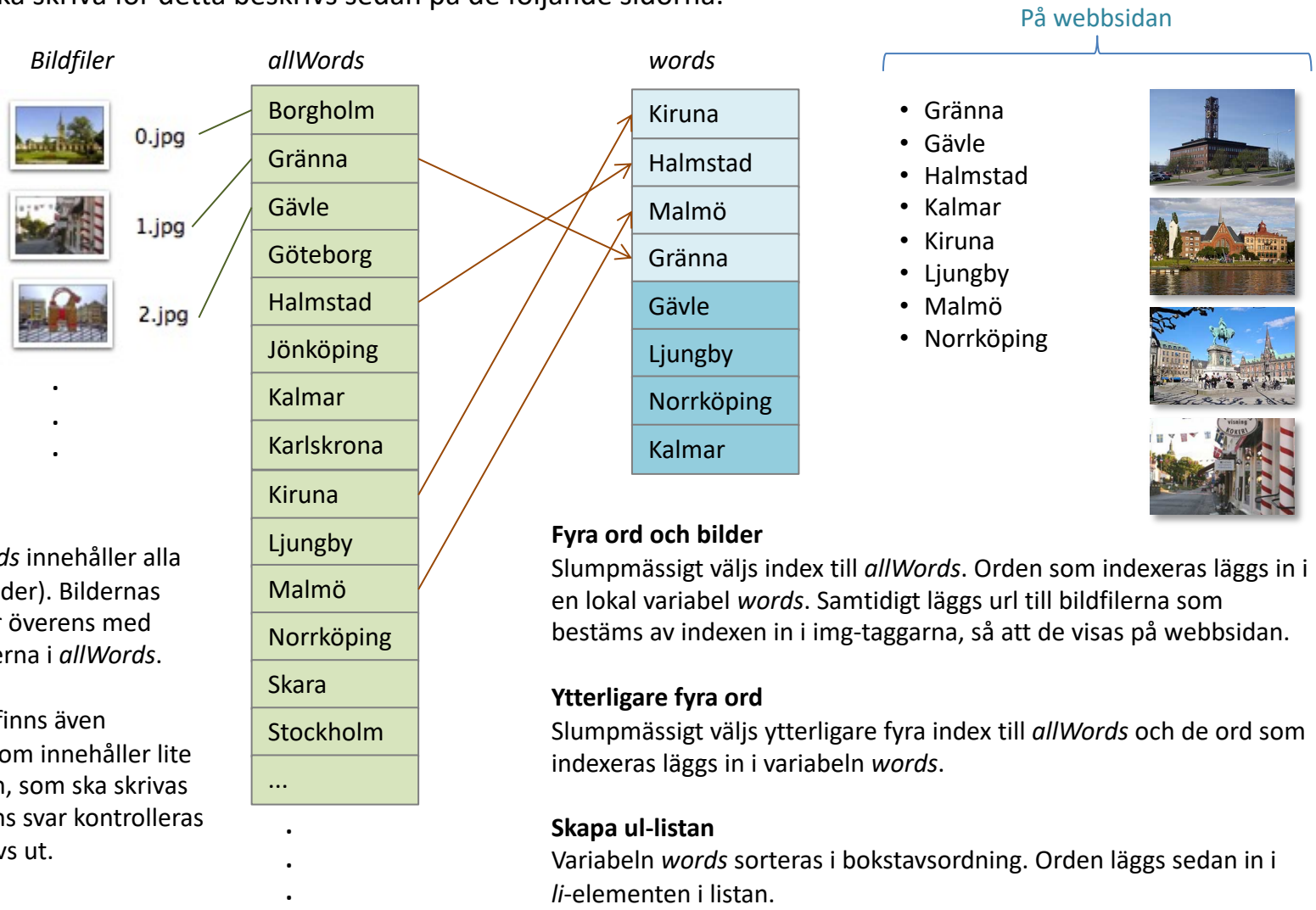
Testa i webbläsaren

- Testa att klicka på knapparna. Längst ner på sidan ska du då få de meddelanden som du skrev i funktionerna.
- Testa också att föra muspekaren över en bild och sedan ut från en bild. För varje bild ska du få meddelandena i funktionerna.

Då du sett att det fungerar, tar du bort meddelandena från funktionerna igen.

4a. Val av ord och bilder

Här beskrivs hur val av ord och bilder ska göras.
Den kod du ska skriva för detta beskrivs sedan på de följande sidorna.



Variabeln *allWords* innehåller alla ord (namn på städer). Bildernas filnamn stämmer överens med indexen för städerna i *allWords*.

I programkoden finns även *allDescriptions*, som innehåller lite extra information, som ska skrivas ut då användarens svar kontrolleras och rätt svar skrivs ut.

Fyra ord och bilder
Slumpmässigt väljs index till *allWords*. Orden som indexeras läggs in i en lokal variabel *words*. Samtidigt läggs url till bildfilerna som bestäms av indexen in i img-taggarna, så att de visas på webbsidan.

Ytterligare fyra ord
Slumpmässigt väljs ytterligare fyra index till *allWords* och de ord som indexeras läggs in i variabeln *words*.

Skapa ul-listan
Variabeln *words* sorteras i bokstavsordning. Orden läggs sedan in i *li*-elementen i listan.

Listan med namnen ska alltså visas i bokstavsordning, medan de fyra bilderna ska visas i den ordning de slumpmässigt valdes.

4b. Välja fyra ord och bilder i funktionen startGame

Du ska nu slumpmässigt välja ord (och därmed också bilder) ur variabeln *allWords*. Ett ord får inte väljas mer än en gång, så då ordet är valt, ska det tas bort ur arrayen. Detta blir alltså på samma sätt som att dra kort ur en kortlek i ett exempel i föreläsning F3. Men *allWords* behövs igen oförändrad, då svaren ska kontrolleras och då man startar spelet igen. Så du ska börja med att ta en kopia av *allWords*.

Kod i funktionen *startGame*

- Kopiera *allWords* till *tempList* med koden i bilden här intill.

```
let tempList = allWords.slice(0);
```

  - För en array räcker det inte att endast skriva *tempList = allWords*.
    Då får man endast en referens till arrayen. Om man sedan gör ändringar i *tempList*, gäller det även *allWords*.
    Så för att få en riktig kopia, får man använda *slice*.
  - Med *slice(0)* får du en ny array med alla element från och med position 0, dvs hela arrayen.

Ta fram de fyra första orden, som också blir bilderna

- Inför variabeln *words* som en tom array.

```
let words = [];
```

- Skriv en *for*-loop som genomlöps fyra gånger. I loopen gör du följande:
  - Bestäm ett slumptal som är ett index till *tempList*. Spara slumptalet i variabeln *r*.
  - Det ord i *tempList*, som indexeras av *r*, lägger du in i *words*. Använd *push*.
  - Spara bildens nummer (dvs index till *allWords*) i en ny variabel kallad *ix*.
    - Eftersom du sedan ska ta bort det valda ordet ur *tempList*,
      blir inte index till *tempList* detsamma som index till *allWords*.
      Så du måste nu ta fram index för valt ord i *allWords*. Det gör du genom att med *indexOf* söka efter ordet i *allWords*.

```
let ix = allWords.indexOf(tempList[r]);
```

  - Lägg bilden som bestäms av *ix* i
    img-taggen som bestäms av loopvariabeln.

```
imgElems[i].src = "img/" + ix + ".jpg";
```

  - Ta bort det valda ordet ur *tempList*.
    - Använd *splice*, för att ta bort ordet ur arrayen.

Testa i webbläsaren

- Varje gång du klickar på startknappen ska fyra bilder visas.

4c. Välja fler ord i funktionen startGame

Nu ska du ta fram ytterligare fyra ord till listan.

Välj ytterligare fyra ord

- Lägg in en loop till, där du väljer ytterligare fyra ord ur *tempList* och lägger in dem i *words*.
  - Loopen blir nästan likadan som föregående loop, fast du ska nu inte visa några bilder. Därmed behöver du heller inte ta fram ix.
- Efter loopen lägger du in en programsats, där *words* sorteras.

```
words.sort();
```

Testa i webbläsaren

- Sist i funktionen lägger du in en utskrift av *words* i konsolen, så du kan testa koden.
- Klicka på startknappen, så bör du få fram en utskrift av arrayen i webbkonsolen.
- Då du ser att det fungerar, tar du bort utskriften igen.

```
console.log(words);
```

Skriv ut orden i listan

- Skriv en loop där du går igenom alla element i *wordElems*.
  - Egentligen spelar det ingen roll ifall du använder *wordElems.length* eller *words.length* som gräns i loopen, eftersom båda arrayerna innehåller lika många värden, nämligen åtta referenser till *li*-taggarna respektive åtta ord.
- I loopen ska det element i *wordElems* som indexeras av loopvariabeln tilldelas det ord i *words* som indexeras av loopvariabeln.
  - Glöm inte *innerHTML*.

Testa i webbläsaren

- Då du klickar på startknappen ska du på webbsidan både få en lista med stadsnamn och fyra bilder.

5. Visa/dölj den stora bilden

Nu ska du skriva kod för att visa och dölja den stora bilden, då man för muspekaren över en liten bild.
Denna kod ska ligga i funktionerna *showLargeImg* och *hideLargeImg*.

Du har redan i övning 3b lagt händelserna *mouseenter* och *mouseleave* på bilderna och kopplat det till funktionerna.
Så nu ska du endast skriva koden i funktionerna.

Kod i funktionen *showLargeImg*

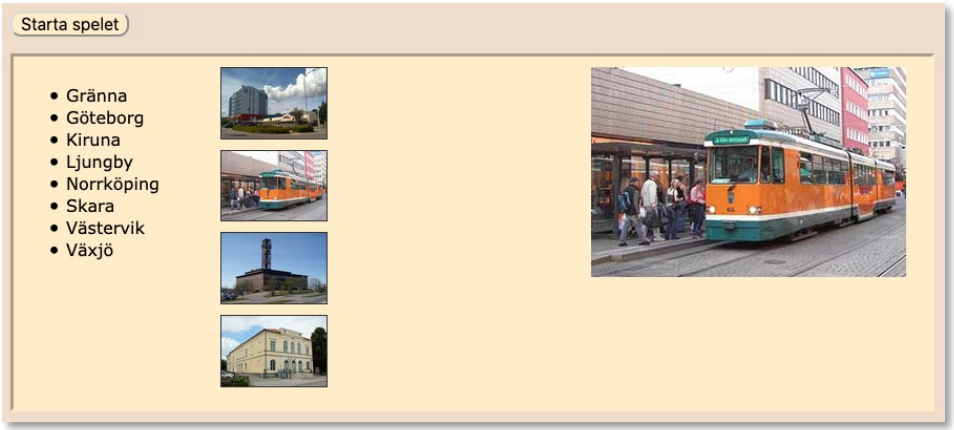
- Då du kommer in i funktionen är *this* en referens till den *img*-tagg som muspekaren är över.
- Avläs *src*-attributet och lägg in det i *src*-attributet för *largeImgElem*.

Kod i funktionen *hideLargeImg*

- I *src*-attributet för *largeImgElem* lägger du in url:en för bilden *empty.png*, så att den bild som visades tas bort igen.

Testa i webbläsaren

- Klicka på startknappen.
- För sedan muspekaren över de små bilderna.
  - Bilden som du pekar på ska då visas som en större bild till höger.



6a. Händelser och funktionalitet för drag-and-drop

startGame
- Elementen för orden ska vara dragbara(både de i listan och de vid bilderna) (draggable=true)
- Händelsehanterare för dragstart och dragend.

Ord i listan kan dras till bilder

Ord vid bilder kan dras till listan och till andra bilder

- Gränna
- Gävle
- Halmstad
-
- Kiruna
- Ljungby
-
- Norrköping


Kalmar


Malmö

dragstart
dragend
(på varje li)

dragstart
dragend
(på varje p efter img, class userAnswer)

dragstartWord
- Lägg på händelsehanterare för dragover och drop på bilderna.
- Lägg på händelsehanterare för dragover och drop på listan.

dragover
drop
(på ul)

dragover
drop
(på varje img)

wordOverImg
- Förhindra "default"-händelse
- drop:
  - Ta bort ordet som dras.
  - Ta fram referens till p-elementet intill img-elementet.
  - Om det redan finns ett ord intill bilden, flyttas det tillbaks till listan.
  - Lägg det ord som drogs i p-elementet.

dragendWord
- Ta bort händelsehanterare för dragover och drop på bilderna.
- Ta bort händelsehanterare för dragover och drop på listan.

wordOverList
- Förhindra "default"-händelse
- drop:
  - Ta bort ordet som dras.
  - Flytta tillbaks ordet till listan.

moveBackToList
- Leta reda på första lediga plats.
- Lägg ordet på denna plats.

Händelsen dragover används inte i denna funktionalitet, men måste ändå finnas med, för att förhindra "default"-händelse för denna operation.

fortsättning på nästa sida ...

6b. Data för drag-and-drop

startGame
- Elementen för orden ska vara dragbara(både de i listan och de vid bilderna) (draggable=true)
- Händelsehanterare för dragstart och dragend.

Ord i listan kan dras till bilder

Ord vid bilder kan dras till listan och till andra bilder

dragstart
dragend
(på varje li)

- Gränna
- Gävle
- Halmstad
-
- Kiruna
- Ljungby
-
- Norrköping


Kalmar

dragstart
dragend
(på varje p efter img, class userAnswer)


Malmö

dragover
drop
(på ul)

dragstartWord
- Lägg på händelsehanterare för dragover och drop på bilderna.
- Lägg på händelsehanterare för dragover och drop på listan.
- **Spara referens till element för det ord som dras.**
- **Spara ordet i dataTransfer.**

Global variabel
dragWordElem

dragover
drop
(på varje img)

wordOverImg
- Förhindra "default"-händelse
- drop:
  - Ta bort ordet som dras.
  - Ta fram referens till p-elementet intill img-elementet.
  - Om det redan finns ett ord intill bilden, flyttas det tillbaks till listan.
  - Lägg det ord som drogs i p-elementet.

dragendWord
- Ta bort händelsehanterare för dragover och drop på bilderna.
- Ta bort händelsehanterare för dragover och drop på listan.

wordOverList
- Förhindra "default"-händelse
- drop:
  - Ta bort ordet som dras.
  - Flytta tillbaks ordet till listan.

moveBackToList
- Leta reda på första lediga plats.
- Lägg ordet på denna plats.

6c. Dra ord – händelsehanterare för dragstart och dragend

Nu ska du skriva den kod som behövs för att kunna dra orden som finns i listan. Du börjar i denna övning med koden för att dra och ska i en senare övning komplettera med resterande kod för drop.

Funktionen *startGame*

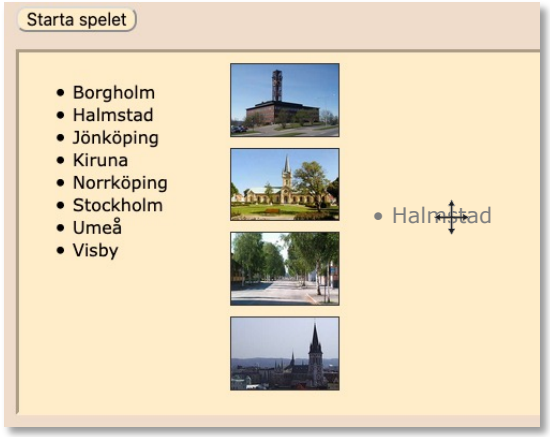
- I den sista loopen i funktionen, där du går igenom *wordElems*, lägger du till två rader för *wordElems[i]* för att lägga på händelsehanterare för:
  - *"dragstart"* och funktionen *dragstartWord*.
  - *"dragend"* och funktionen *dragendWord*.
- Lägg också till en rad, så att elementet blir dragbart.

```
wordElems[i].draggable = true;
```

- Skriv sedan en ny loop, där du går igenom alla *answerElems*.
  - I loopen lägger du på likadana händelsehanterare som i ovanstående loop, fast för *answerElems[i]*.
  - Lägg också till en rad för att göra *answerElems[i]* dragbart.

Testa i webbläsaren

- Klicka på startknappen och prova sedan att dra ett ord i listan.
- Då du släpper det åker det tillbaks till listan,
  eftersom vi inte skrivit resten av koden ännu.



6d. Händelsehanterare för dragover och drop

Nu ska du skriva funktionerna för dragstart och dragend.

Funktionen *dragstartWord*

- Skriv en loop där du går igenom *imgElems*. För varje *imgElems[i]* lägger du på händelsehanterare för:
  - *"dragover"* och funktionen *wordOverImg*.
  - *"drop"* och funktionen *wordOverImg*.
    - Det ska alltså vara samma funktion för båda händelserna.
- Efter loopen lägger du på händelsehanterare på listan, *wordListElem*, för:
  - *"dragover"* och funktionen *wordOverList*.
  - *"drop"* och funktionen *wordOverList*.
- Spara sedan *this* (som är det element som dras) i variabeln *dragWordElem*.
- Sätt sedan *dataTransfer* till att innehålla ordet som dras.

```
e.dataTransfer.setData("text",this.innerHTML);
```

  - Egentligen kan vi ju sedan få fram det ordet genom *dragWordElem*, men för att det ska fungera i alla webbläsare, måste vi lägga in något i *dataTransfer* och då kan vi ju lika gärna lägga ordet där.

Funktionen *dragendWord*

- Kopiera loopen och de två raderna för *wordListElem* från *dragstartWord* och klistra in i *dragendWord*.
- I de fyra rader där du har *addEventListener* ändrar du till *removeEventListener*.

Testa i webbläsaren

- Klicka på startknappen och prova sedan att dra ett ord i listan.
- Det händer inget mer nu, men kontrollera att du inte får något felmeddelande i konsolen, då du drar eller släpper ett ord.

6e. Släpp ord över en bild

Nu ska du lägga till kod, så att man kan släppa ett ord över en bild.

Funktionen *wordOverImg*

- I funktionen finns en parameter kallad *e*. Detta är *Event*-objektet som skickas till funktionen.
- Lägg in en rad för att förhindra webbläsarens "default"-beteende vid drag-and-drop.

```
e.preventDefault();
```

- Lägg sedan in en *if*-sats, där du kontrollerar om händelsen är *"drop"*.
- I *if*-satsen lägger du in följande (som visas i vidstående ruta):
  - Ta bort ordet som dras, genom att lägga in en tom textsträng i elementet.
  - Ta fram en referens till elementet som ligger efter *img*-taggen.
    - Händelsehanteraren för *drop* ligger ju på bilden, så *this* är en referens till *img*-taggen. Men ordet ska skrivas i *p*-elementet som ligger efter denna i HTML-koden.
  - Kontrollera om det redan finns något i *p*-elementet. I så fall anropa *moveBackToList* och skicka med ordet som låg där som parameter.
  - Hämta slutligen det ord som dragits och lägg in det i *p*-elementet.

```
if (e.type == "drop") {
    dragWordElem.innerHTML = "";
    let dropWordElem = this.nextElementSibling;
    if (dropWordElem.innerHTML != "")
        moveBackToList(dropWordElem.innerHTML);
    dropWordElem.innerHTML = e.dataTransfer.getData("text");
}
```

Testa i webbläsaren

- Klicka på start-knappen.
- Dra sedan ord till bilderna.
- Om du drar ett ord till en bild som redan har ett ord, försvinner det ord som fanns där. Det kommer inte tillbaks till listan, eftersom vi ännu inte skrivit någon kod i moveBackToList.

6f. Flytta ord till listan

Nu ska vi skriva koden för att flytta ord till listan.

Funktionen *wordOverList*

- Denna funktion blir ungefär som funktionen *wordOverImg*, fast lite enklare. Vi behöver här inte ta fram någon referens till något *p*-element.
- Det enda som ska göras är att förhindra "default"-beteende. Sedan kollar vi att händelsen är *"drop"* och tar då bort det ord som dragits samt anropar *moveBackToList*.
- Parametern till *moveBackToList* ska vara det ord som dragits, dvs det som finns i *dataTransfer*.

```
e.preventDefault();
if (e.type == "drop") {
    dragWordElem.innerHTML = "";
    moveBackToList(e.dataTransfer.getData("text"));
}
```

Funktionen *moveBackToList*

- Skriv en loop där du går igenom alla *wordElems*.
- I loopen kontrollerar du om ordet i elementet är tomt (dvs "", en tom sträng).
  - I så fall har du hittat en en ledig plats. På denna plats lägger du ordet i parametern *word*.
  - Avbryt sedan loopen med *break*.
    - Vi behöver ju inte leta efter fler lediga platser.

Testa i webbläsaren

- Klicka på start-knappen och prova nu att dra ord till bilderna och även till bilder där du redan lagt ett ord. Prova också att du kan dra orden vid bilderna både till listan och till andra bilder.

7a. Data för att kunna kontrollera svaren

Nu ska du skriva koden för funktionen *checkAnswers*, där du ska kontrollera om användarens svar är korrekta, dvs om rätt stadsnamn är draget till de olika bilderna.

De rätta städerna är ju de som visas i bilderna, så vi ska jämföra en bild med det ord som användaren dragit till den. Då behöver vi "översätta" bilden till namnet på staden. Det vi har är numret i filnamnet som stämmer med index till *allWords*, där vi kan få fram namnet. Så vi skulle kunna med några funktioner för stränghantering ta ut det numret och använda som index. Men vi ska istället göra på ett annat sätt.

Vi har redan indexet, då vi lägger in bilderna i funktionen *startGame*. Där har vi tagit fram det i en variabel kallad *ix*. Vi kan då spara *ix* i *img*-taggarnas *id*-attribut, så kan vi sedan i funktionen *checkAnswers* ta fram *id*-attributet och på så sätt få fram indexet. Vi använder alltså i*mg*-taggarna för att lagra och överföra information från ett skede i programmet till ett annat skede.

Funktionen *startGame*

- I den första loopen (den där bilderna läggs in i *img*-taggarna) lägger du till en rad för att spara variabeln *ix* i *img*-taggens *id*-attribut.

```
imgElems[i].id = ix;
```

7b. Kontrollera svaren

Nu ska funktionen för att kontrollera svaren skrivas. Den anropas, då användaren klickar på knappen "Kontrollera svar". I den funktionen måste vi först kolla att det finns ord dragna till alla bilder. Sedan ska vi också ta bort händelsehanterarna på orden, så att det inte går att fortsätta dra dem. Slutligen ska vi kontrollera svaren och räkna "poäng", 1 poäng per rätt svar.

Funktionen *checkAnswers*

- Lägg in en loop där du går igenom alla *answerElems*, dvs *p*-elementen med de ord som användaren dragit till bilderna. I loopen gör du följande:
  - I en *if*-sats kontrollerar du om elementet är tomt.
  - I så fall skriver du med *alert* "Dra först ord till alla bilder." och sedan avbryter du funktionen med *return*.
- Därefter ska du ta bort händelsehanterarna för drag på alla ord:
  - I en loop går du igenom *wordElems* och sätter *draggable* till *false* och tar bort händelse-hanterarna för *dragstart* och *dragend*.
  - Därefter gör du en likadan loop, fast för *answerElems*.

```
for (let i = 0; i < wordElems.length; i++) {
    wordElems[i].draggable = false;
    wordElems[i].removeEventListener("dragstart",dragstartWord);
    wordElems[i].removeEventListener("dragend",dragendWord);
}
```

- Nu är det dags att kontrollera svaren och räkna "poäng":
  - Inför variabeln *points* och sätt den till 0.
  - Skriv sedan en loop där där du går igenom *answerElems*. I loopen gör du följande:
    - Inför en variabel *ix* och lägg där *id*-attributet i den bild som *imgElems[i]* refererar till.
    - Jämför ordet i *answerElems[i]* med *allWords[ix]* i en *if*-sats och om de är lika räknas *points* upp med 1.
    - I elementet *correctElems[i]* skriver du ut rätt svar. Skriv ut både *allWords[ix]* och *allDescriptions[ix]*.

```
if (answerElems[i].innerHTML == allWords[ix])
    points++;
```

  - Efter loopen skriver du ut antal poäng tillsammans med lämplig text i elementet för meddelanden (*msgElem*).

Testa i webbläsaren

- Dra ord till alla bilder och klicka sedan på knappen för att kontrollera svar.

8. Rensa för nytt spel

Om du klickar på knappen "Starta spelet" igen, så ser du att de gamla orden och svaren ligger kvar intill bilderna. Även meddelandet med antal poäng ligger kvar. Allt detta ska rensas bort, då man startar en ny omgång.

Funktionen *startGame*

- I den sista loopen, där du går igenom *answerElems*, lägger du till två rader för att ta bort både användarens svar och det rätta svaret.

```
answerElems[i].innerHTML = "";
correctElems[i].innerHTML = "";
```

- Efter loopen lägger du in en rad där du tar bort texten i elementet för meddelande.

Testa i webbläsaren

- Kör igenom en omgång och klicka sedan på startknappen igen.

9. Aktivera/inaktivera knaparna

De båda knapparna ska nu aktiveras och inaktiveras i olika skeden i programmet. Detta gör du genom att ändra egenskapen *disabled* mellan *true* och *false*. Du gör alltså på samma sätt som du gjorde i laboration 3.

Funktionen *init*

Sist i funktionen lägger du in kod, så att start-knappen blir aktiv och kontroll-knappen blir inaktiv.

Funktionen *startGame*

Sist i funktionen lägger du in kod, så att start-knappen blir inaktiv och kontroll-knappen blir aktiv.

Funktionen *checkAnswers*

Sist i funktionen lägger du in kod, så att start-knappen blir aktiv och kontroll-knappen blir inaktiv.

Testa i webbläsaren

- Nu är programmet klart, så kör igenom och testa att allt funkar och att du inte får några felmeddelanden i konsolen.

10. Städa koden och lägg till kommentarer

Då du nu är klar med programmet, ser du över det och snyggar eventuellt till indenteringar, m.m.

Kontrollera också att du skrivit en kommentar för varje funktion och varje ny variabel som införs i koden.