Linnéuniversitetet
Institutionen för medieteknik
Interaktiva medier och webbteknologier

http://www.lnu.se/
Växjö
Kandidatprogram 180 hp

Creative Commons (BY-NC-SA)
Henrik Andersen Ph.M.
henrik.andersen@lnu.se

1ME325 Webbteknik 5, 7,5hp

Föreläsning 4b; Objektorienterad programmering med JavaScript (Del 2)

Innehåll

- Paketstrukturer (Namespace)
  - Organisations- och strukturtekniker
- Drag-and-drop
  - HTML 5
  - DOM & Canvas

Syfte

- Introducera paketstrukturer i syfte att skapa strukturerade, skalbars och objektorienterade JavaScript applikationer, tjänster och bibliotek

- Introducera och exemplifiera drag-and-drop-funktionalitet med inbyggda och egentillverkade metoder

Paketstrukturer (Namespace)

- Inledande frågeställningar:
  - *Vad är en paketstruktur?*
  - *Varför används paketstrukturer?*
  - *Vad kan vara ett passande exempel på paketstruktur?*

Paketstrukturer

- Inom programmering kan paketstrukturer beskrivas enligt följande:

  - Teckenuppsättning vars syfte är att fungera som en unik identifierare, för ett enskilt objekt eller klass, alternativt en gruppering av objekt eller klasser

- Paketstrukturer förekommer som en inbygd komponent av flera programmeringsspråk, men inte i JavaScript; *likt annan saknad funktionalitet kan den skapas av utvecklaren*

Paketstrukturer

- Förväntade frågeställningar?
  - *Vad innebär unik? unik i programkoden eller på ”global nivå”?*
  - *Hur förblir teckenutsättningen unik?*

Paketstrukturer

- En paketstruktur består av två komponenter:
  - paketadress (grön); *objekt eller kombination av objekt som fungerar som identifierare*
  - klass (röd); *klass som lagras och förknippas med objektstrukturen. Notera att flera klasser kan finnas under samma paketadress*
- Exempel på paketstruktur:
  - se.lnu.mediatech.webos.Main
  - se.lnu.mediatech.webos.Drag

Paketstrukturer

- Det är vanligt förekommande att omvända domännamn används för att skapa paketstrukturer
- Förväntade frågeställning:
  - *Varför är omvända domännamn passande för paketstrukturer?*

Paketstrukturer; *vision*

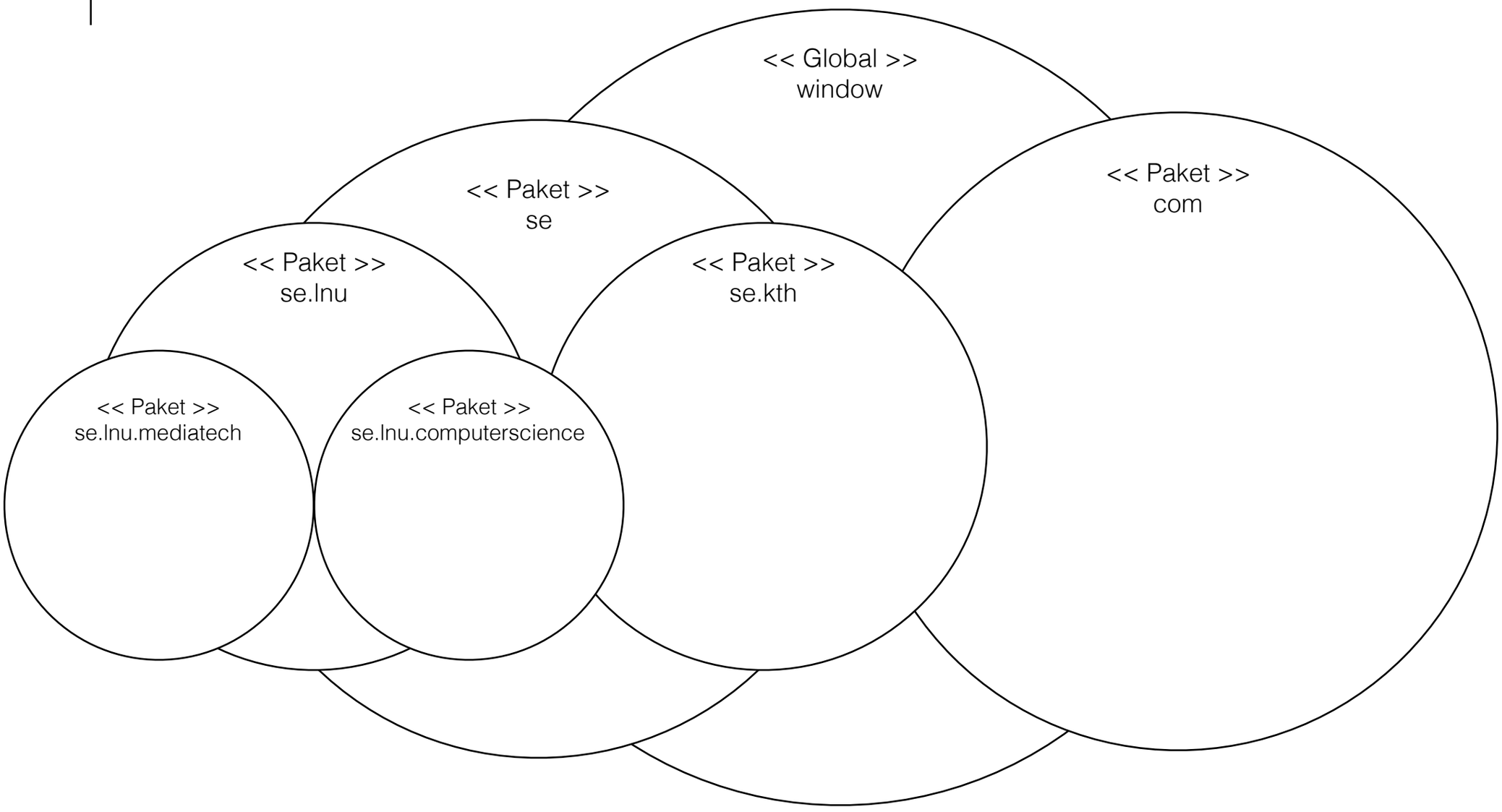

Fig. Illustrerar hur paketstrukturer representerar en serie av omfång dit programkod adresseras i syfte att kapslas in och inte påverka andra eventuella kodbibliotek

Paketstrukturer

- Då JavaScript inte är av strikt karaktär, kan paketstrukturer skapas som en kedja av objektinstanser

- Programkod så som skript, klasser och bibliotek, adresseras till något av paketstrukturens noder, detta i syfte att kapsla in programkoden så att den inte överskrivs av andra potentiella bibliotek eller skriptfiler

Paketstrukturer

- Klasser som ingår i en objektstruktur erhåller en unik identifierare och det kan därför finnas flera klasser med samma namn i ett system

- En identifierare representerar en specifik plats i en objektstruktur och är därmed en adress till vart klassen är sparad

- Identifierare används för att organisera och strukturera programkod så att funktionalitet lagras på beskrivande plats

Paketstrukturer

```
window.se.lnu.drag.Main = function() {
   ...
};

window.se.lnu.load.Main = function() {
   ...
};

var m1 = new se.lnu.drag.Main();
var m2 = new se.lnu.load.Main();
```

Fig. Ovanstående programkod beskriver två Main-klasser som adresserats till olika paketstrukturer

Paketstrukturer

- Förväntade frågeställningar:
  - *Måste man skriva hela paketstrukturen varje gång som en klass skall adresseras eller instansieras?*
  - *Finns det andra fördelar med paketstrukturer förutom att undvika eventuella programkodskrockar (överskrivningar)?*

Paketstrukturer

- Relevant problematik:

  - Sårbar trädstruktur; *rot-paketet eller globalt gemensamma paket löper risk att överskrivas av felaktigt konstruerade bibliotek*

  - Klasser är beroende av paketstrukturen; *paketstrukturen (i sin helhet) måste skapas innan klass-dokument initieras av JavaScript-tolken*

Paketstrukturer; *sårbar trädstruktur*

- Följande programkod skapar en paketstruktur utifrån den omvända domännamnspricipen, utan att ta hänsyn till eventuella tidigare instanser av hela, eller delar av paketstrukturer:

```
window.se = {};
window.se.lnu = {};
window.se.lnu.drag = {};
```

- Samtliga (eventuella) bibliotek under se-paketet överskrivs och på så sätt försvinner från exekveringsmiljön

Paketstrukturer; *klasser är beroende av paketstrukturen*

- Följande programkod är felaktig då klassen Main deklareras till en paketstruktur som ännu inte är skapad:

```
window.se.lnu.drag.Main = {
   ...
};

window.se = {};
window.se.lnu = {};
window.se.lnu.drag = {};
```


Paketstrukturer

```
//---------------------------------------------------------------------------
// Package
//---------------------------------------------------------------------------

if (window.se == undefined) window.se = {};
if (window.se.lnu == undefined) window.se.lnu = {};
if (window.se.lnu.drag == undefined) window.se.lnu.drag = {};

//---------------------------------------------------------------------------
// Code
//---------------------------------------------------------------------------

window.se.lnu.drag.Main = {
    ...
};

window.se.lnu.drag.Utils = {
    ...
};
```

Fig. Ovanstående programkod skapar en paketstruktur för ett tilltänkt Drag-bibliotek. Klasserna Main och Utils adresseras senare till paketstrukturen.

Paketstrukturer; *sammanfattning*

- Paketstrukturer är en organisationsprincip där programkod kategoriseras och organiseras enligt beskrivande teckenutsättningar som både refererar och grupperar programkod i beskrivande paket

- JavaScript har inget inbyggt stöd för paketstrukturer men kan skapas på eget initiativ av utvecklare via dynamiska objektstrukturer

- Paketstrukturer används för att kapsla in programkod, färast i syfte att skapa isolerade projekt och bibliotek

Drag-n-Drop

- Ett begrepp inom grafiska användargränssnitt, där användare via inmatningsverktyg, väljer och förflyttar virtuella objekt genom att dra och släppa dem på en ny position eller på ett annat virtuellt objekt

- En väletablerad funktionalitet som exempelvis görs tillgänglig via operativsystems filsystem där användare kan sortera och organisera filer genom att dra och släppa dem på en ny position

Drag-n-Drop; *JavaScript*

- Möjligheten till att skapa drag-and-drop-funktionalitet har funnits sedan tidiga iterationer av webbplattformen; *är en relativt ovanlig funktionalitet på webben*

- Funktionaliteten kan uppnås på flera olika sätt:
  - HTML5; *använder inbyggt stöd i HTML5-standarden*
  - Event DOM; *använder kombination av händelselyssnare och DOM-element*
  - Event Canvas; *använder kombination av händelselyssnare och Canvas-elementet*

Drag-n-Drop; *HTML5*

- Baseras på händelsehantering i kombination med det nya draggable-attributet:

```
<div id="drag-elm" draggable="true"></div>
```

- Ovanstående HTML-kod resulterar i att användaren kan dra ovanstående element utifrån en spökbildsrepresentation; *original-elementet behåller sin ursprungliga position och form*

Drag-n-Drop; *HTML5*

- Händelser som kan kopplas till det element som förflyttas, d.v.s. element med attribut draggable satt till true:

  - ondragstart; *aktiveras då användaren påbörjar drag-and-drop-funktionaliteten*

  - ondrag; *aktiveras varje gång som elementet är under förflyttning*

  - ondragend; *aktiveras då användaren släpper elementet och drag-and-drop-funktionaliteten avslutas*

Drag-n-Drop; *HTML5*

- Händelser som kan kopplas till potentiella ”mål-element”, d.v.s. element dit drag-bara element kan släppas:

  - ondragenter; *aktiveras då det förflyttade elementet förs över en potentiell ”mål-yta”*

  - ondragover; *aktiveras då det förflyttade elementet förts över en potentiell ”mål-yta”*

  - ondragleave; *aktiveras då det förflyttade elementet lämnar den potentiella ”mål-ytan”*

  - ondrop; *aktiveras då det förflyttade elementet släpps på en potentiell ”mål-yta”*

Drag-n-Drop; *HTML5*

- Händelseobjekt (DragEvent) som relaterar till drag-and-drop-funktionalitet, innefattar ett dataöverföringsobjekt

- DataTransfer är ett API som har i syfte att lagra information om element, eller information som relaterar till elementet, under den tid som användaren nyttjar drag-and-drop-funktionaliteten

- Objektet existerar enbart som en egenskap till DragEvent och kan därmed inte instansieras på begäran av utvecklare (saknar konstruktor)

Drag-n-Drop; *HTML5*

- DataTransfer objektet innehåller ett flertal egenskaper och metoder; *följande lista innehåller objektets väsentliga metoder:*

  - setData(key, value); *skriv textbaserad information till objektet i samband med en drag-and-drop-operation. Informationen hanteras som ett nyckel-värde-par.*

  - getData(key); *hämtar befintlig data utifrån nyckelvärde*

  - setDragImage(); *möjlighet till avvikande grafik för spökbild vid drag-and-drop-operation*

- Full dokumentation över DataTransfer: https://developer.mozilla.org/en-US/docs/Web/API/DataTransfer

Drag-n-Drop; *HTML5*

- Funktionell check-lista för HTML5-baserad drag-and-drop-funktionalitet:
  - Applicera attributet draggable på dragbars element
  - Applicera händelselyssnare (dragstart) för att avgöra när drag-and-drop-funktionaliteten påbörjas
  - Applicera händelselyssnare (drop & dragover) för eventuella drop-element
  - Vid dragstart, spara eventuell information i event.dataTransfer
  - vid dragover, förhindra webbläsarens förbestämda beteende
  - vid drop, använd eventuell information i event.dataTransfer för att exempelvis flytta original-elementet till ny position

Drag-n-Drop; *HTML5*

- Exempel på drag-and-drop-funktionalitet som skapats via HTML5-standardens inbyggda stöd:

  - https://cactuar.lnu.se/course/1me325/example/draganddrophtml5/

Drag-n-Drop; *HTML5*

- Förväntad frågeställning:
  - *Finns det några nackdelar med att använda HTML5 varianten av drag-and-drop?*

Drag-n-Drop; *Canvas & DOM*

- Båda tekniker är händelsestyrda och kretsar kring positionering utifrån ett ursprungsobjekt; *document eller canvas*

- Drag-and-drop-funktionaliteten skapas via händelserna: mousedown, mousemove och mouseup

Drag-n-Drop; *Canvas & DOM*

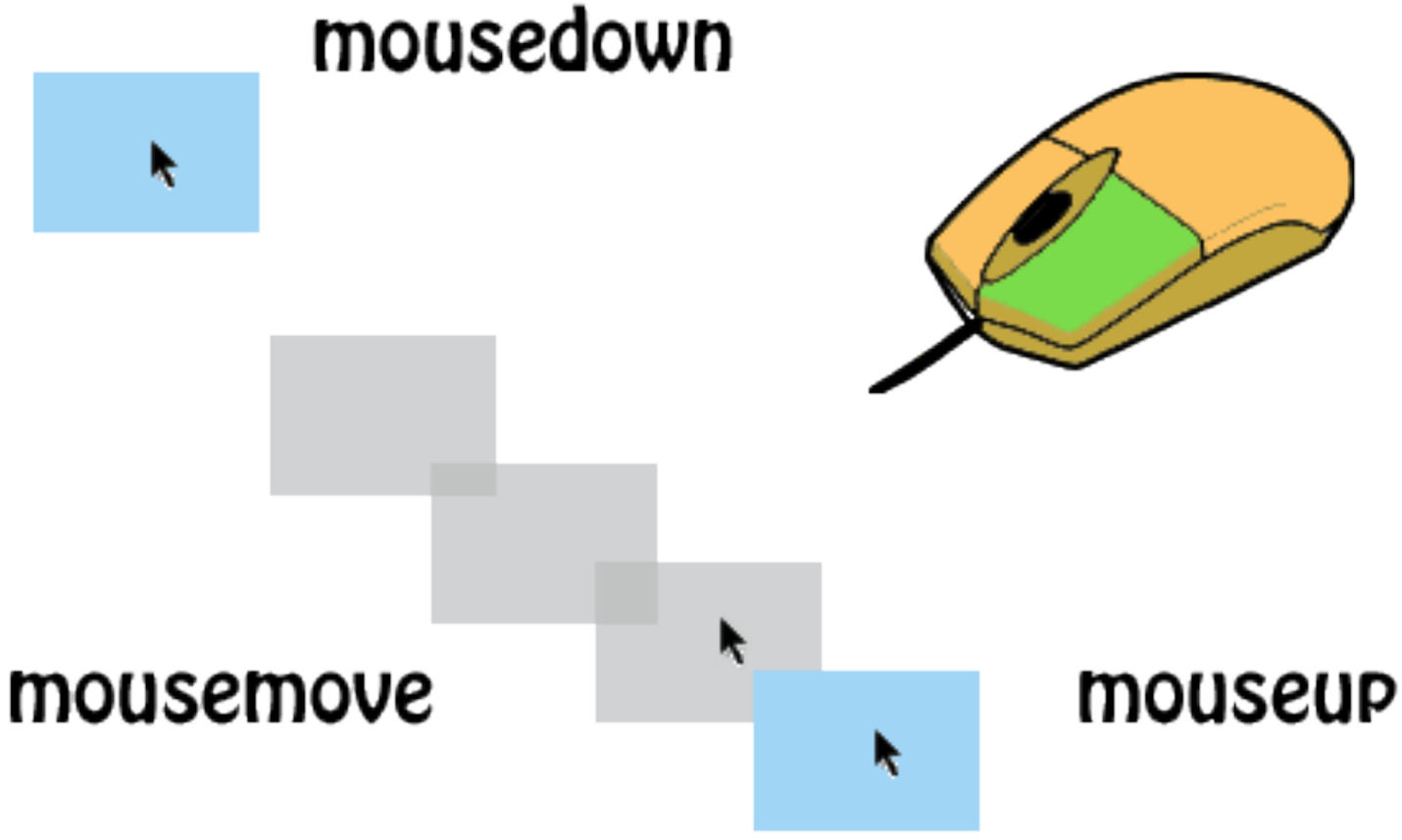

Fig. Illustrerar händelseförloppet för drag-and-drop-funktionalitet.

Drag-n-Drop; *Canvas & DOM*

- Funktionell check-lista för DOM- och Canvas-baserad drag-and-drop-funktionalitet:

  - Applicera händelselyssnare (mousedown) på dragbars element

  - Vid mousedown, applicera händelselyssnare (mousemove) på ursprungsobjekt (document eller canvas). Aktivera även händelselyssnare (mouseup) för att avbryta funktionaliteten

  - Vid mousemove, flytta elementet till muspekarens aktuella position och beräkna eventuell offset mellan mus och elementets koordinater

  - Vid mouseup, avbryt drag-and-drop-processen och ta bort aktuella händelselyssnare (mousemove & mouseup)

Drag-n-Drop; *Canvas & DOM*

- Inledande problematik att behandla:

  - Koordinatsystem och positionering; *vad är en förutsätting för att DOM-element skall kunna placeras på valfri position och hur representeras koordinatsystemet?*

  - Inledande positionering mellan muspekare och element; *skall alltid elementet erhålla exakt samma position som muspekaren?*

  - Aktuell musposition; *vad finns det för möjligheter då muspekarens koordinater måste hämtas?*

Drag-n-Drop; *Canvas & DOM*

- Koordinatsystem och positionering:

  - CSS är en förutsättning, absolut eller relativ positionering måste appliceras på det aktuella elementet

  - CSS-egenskaper som representerar x- och y-koordinater i DOM-strukturen är left (x) och top (y)

Drag-n-Drop; *Canvas & DOM*

- Inledande positionering mellan muspekare och element:

  - Muspekaren är sällan på exakt samma position som elementet; *detta då elementets origo är i dess övre vänstra hörn*

  - Om elementet inte skall ”snappa” (hoppa) till samma position som muspekaren, måste ett offset-värde beräknas

Drag-n-Drop; *Canvas & DOM*

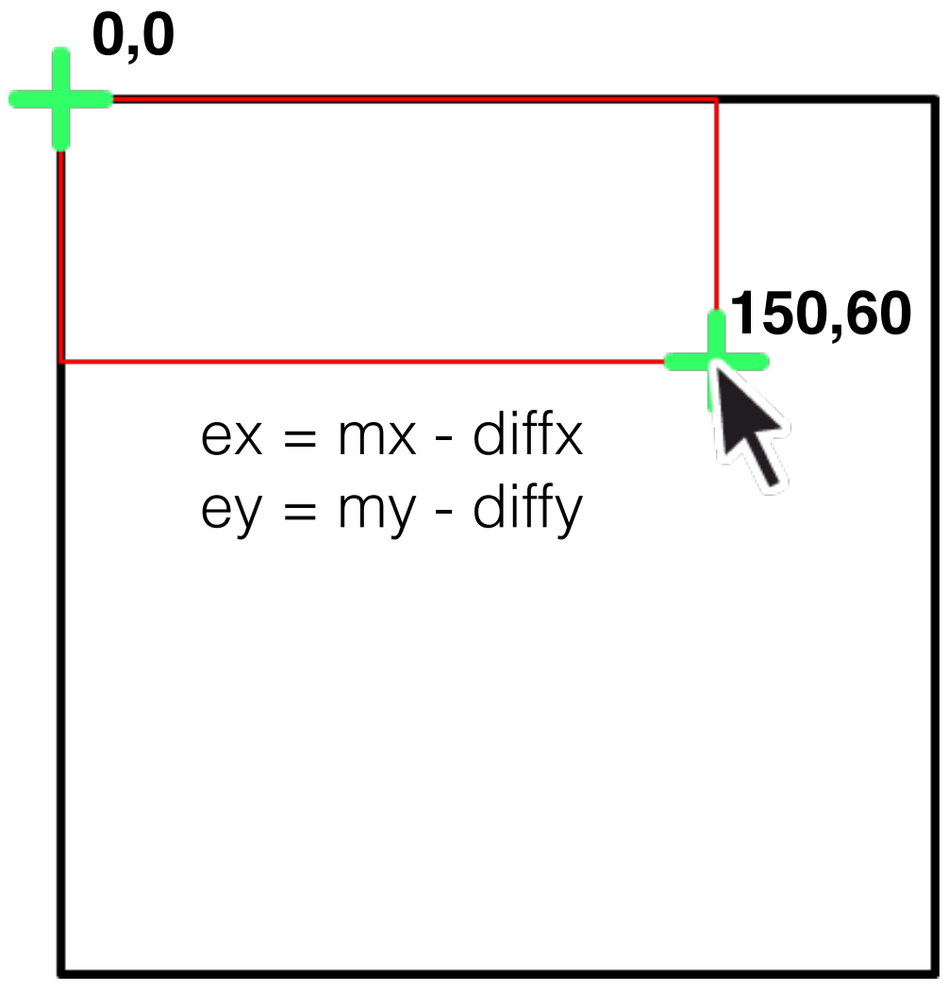

Fig. Illustrerar offset mellan element och muspekare. Används för att undvika ”snapping”

Drag-n-Drop; *Canvas & DOM*

- Aktuell musposition:

  - Musens koordinater kan enbart hämtas i samband med mushändelser (MouseEvent); *detta innebär att koordinaterna inte kan hämtas för varje renderad bildruta (frame)*

  - Händelsen mousemove är den bästa möjligheten att hämta ut en aktuell position, detta görs enbart då muspekaren förflyttar sig; *elementet eftersträvar senast kända position*

Drag-n-Drop; *exempel*

- Länk till JavaScript-bibliotek för drag-and-drop-funktionalitet (JavaScript & DOM):

  - https://cactuar.lnu.se/course/1ME325/example/dragndrop/se/lnu/drag/build/index.html

Drag-n-Drop; *exempel*

- Förväntade frågeställningar:
  - *Varför skapa funktionaliteten som ett bibliotek?*
  - *Resulterar inte biblioteket i mer arbete och programkod?*
  - *Är bibliotek att föredra?*

Drag-n-Drop; *programkodsreflektion*

```
window.onload = function(event) {
    var windows = document.getElementsByClassName("window");
    var drag = new DragnDrop();
    for (var i = 0; i < windows.length; i++) {
        drag.add(windows[i], windows[i].getElementsByClassName("menu")[0]);
    }
};
```

Fig. Ovanstående programkod ligger till grund för föregående applikation. Övrig programkod tillhör biblioteket och inte applikationskoden

Drag-n-Drop; *begränsningar*

- Biblioteket saknar stöd för drop-target; *det finns ingen inbyggd funktionalitet för att kontrollera om användaren släpper ett element på ett annat*

- *Detta kan vara en tänkbar patch för den sömnlöse studenten*

Sammanfattning

- Paketstrukturer utgör grunden för att kapa organiserade kodbaser i syfte att bilda isolerade JavaScript-bibliotek

- Drag-and-drop-funktionalitet kan uppnås i webbläsare via en kombination av JavaScript och CSS, alternativt med den nya HTML5-teknologin

Frågor