Lecture 15

Advanced UI Implementation

UNIVERSITY OF AUCKLAND

COMPSCI 345 / SOFTENG 350

Dr. Gerald Weber

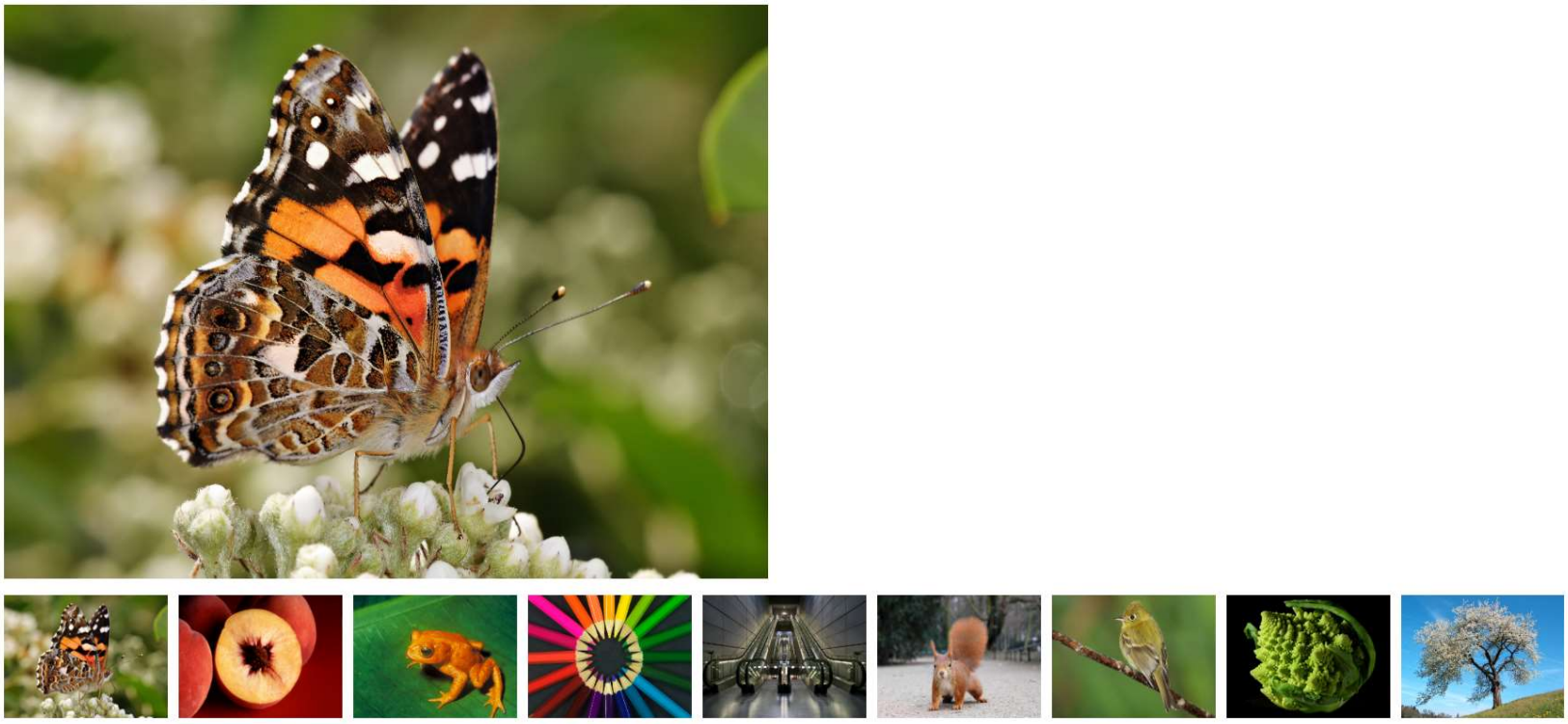
Zoom:

https://auckland.zoom.us/j/92766559788?pwd=a2FzWkJsSFBFRmRzbWNHam1SdUVQQT09

Learning Objectives

- Being able to implement cross-session client-side application storage.
- Understand the single-origin policy of web applications.
- Make application behavior consistent between first-use and ongoing use.

Extending a UI with persistence



Current View is not persisent

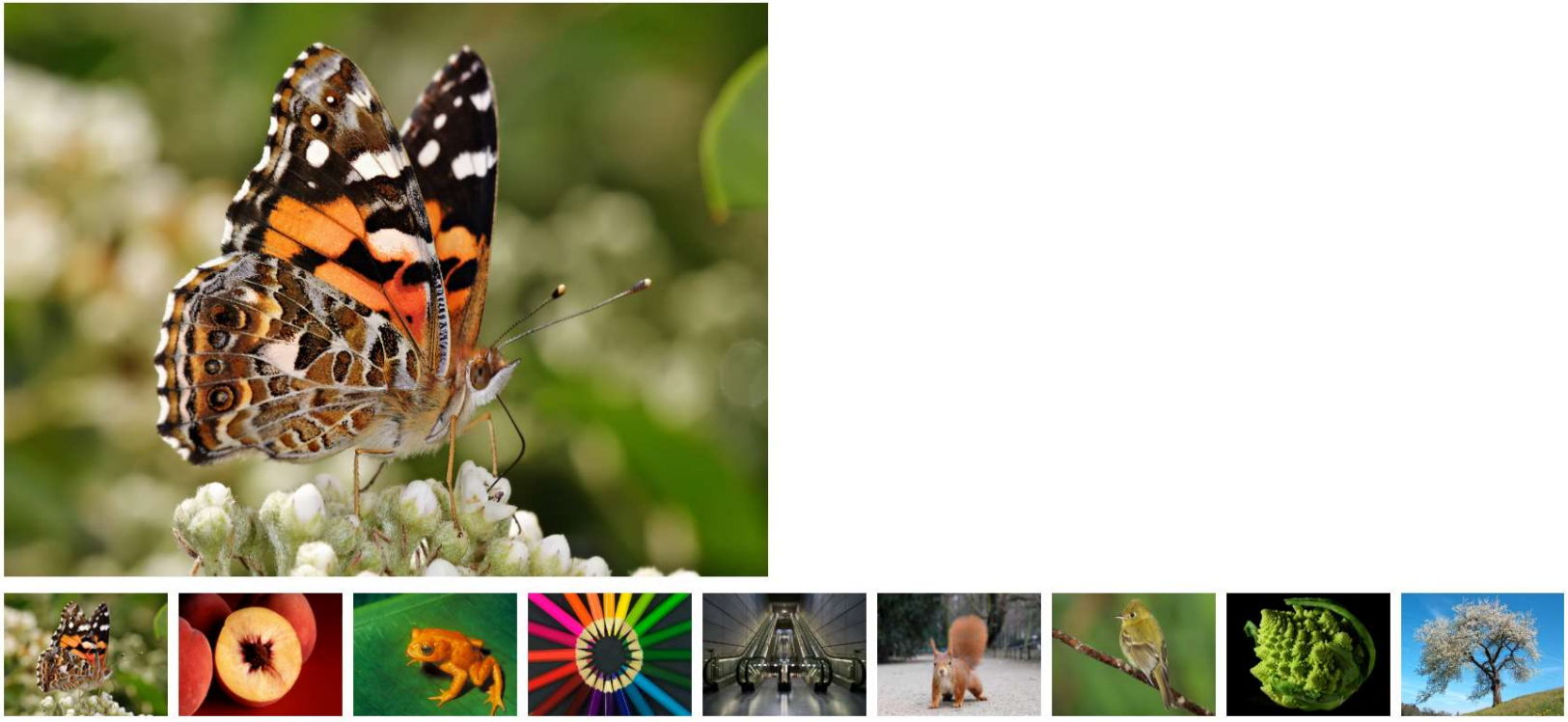








New View resets to original



What if I customize the View ?

Customized order should be persistent

On restart resets to customized order

Customized order restored

Client-Side Storage

- Current standardized client-side storage in the browser a.k.a known as Web storage (DOM storage)
- Object localStorage: a framework-defined javascript object that appears as:
  - same object within browser, (different from data-myTip ~> dataset.myTip)
  - same object across restart of browser
    - for the same origin :
    - For webpages from the same server.
- (There is also Object sessionStorage:
  - Only for one Tab)

Accessing `localStorage`

- Very similar to any javaScript object, but allows only string values.
- Offers object dereferencing and special functions for access:
- Reading values

```
myVar=localStorage.getItem("myField");
```

- For constant strings same as:

```
myVar =localStorage.myField;
```

- Setting values

```
localStorage.setItem("myField", myVar)
```

- For constant strings same as:

```
localStorage.myField=myVar;
```

Using `localStorage`

- What do we have to store in localStorage?
- The current order; represented by order values in thumbnails.
- We make it persistent in the moment we change the order.
- We can use the initial order as our identifiers of thumbnails and as fields in local storage
- Accessing storage:

```
localStorage.setItem(thumb.dataset.ord, y);
x=localStorage.getItem(thumb.dataset.ord);
```

- We also need to store the maximum preference:

```
localStorage.maxPreference;
```

Setting up Session storage

- Detecting state: revisit or first visit

```
const thumbs = document.querySelectorAll('.thumb');
...
// Code to initially store or retrieve data.
  if ( localStorage.maxPreference){
     thumbs.forEach(thumb => {
       thumb.style.order=localStorage.getItem(thumb.dataset.ord);
     })
  } else {
     localStorage.maxPreference=1;
     thumbs.forEach(thumb => {
        localStorage.setItem(thumb.dataset.ord, thumb.dataset.ord);
     })
  }
```

Setting up Session storage

- Detecting state: revisit or first visit

```
const thumbs = document.querySelectorAll('.thumb');
...
// Code to initially store or retrieve data.
  if ( localStorage.maxPreference){
     thumbs.forEach(thumb => {
       thumb.style.order=localStorage.getItem(thumb.dataset.ord);
     })
  } else {
     localStorage.maxPreference=1;
     thumbs.forEach(thumb => {
        localStorage.setItem(thumb.dataset.ord, thumb.dataset.ord);
     })
  }
```

Setting up Session storage

- Detecting state: revisit or first visit

```
const thumbs = document.querySelectorAll('.thumb');
...
// Code to initially store or retrieve data.
  if ( localStorage.maxPreference){
     thumbs.forEach(thumb => {
       thumb.style.order=localStorage.getItem(thumb.dataset.ord);
     })
  } else {
     localStorage.maxPreference=1;
     thumbs.forEach(thumb => {
        localStorage.setItem(thumb.dataset.ord, thumb.dataset.ord);
     })
  }
```

Immediate Storage of Favourite order

- Store new **`maxPreference`** and the change for the order of the clicked thumbnail

```
const thumbs = document.querySelectorAll('.thumb’);
...
 function makeMax() {
    localStorage.maxPreference--;
        if (current) {
          current.style.order=localStorage.maxPreference;
          localStorage.setItem(current.dataset.ord,
                       localStorage.maxPreference);
                }
    }
```

Setting of first image shown, 1

```
const thumbs = document.querySelectorAll('.thumb');
...
// Code to initially store or retrieve data.
  if ( localStorage.maxPreference){
     thumbs.forEach(thumb => {
       thumb.style.order=localStorage.getItem(thumb.dataset.ord);
        if (thumb.style.order==localStorage.maxPreference){
            document.getElementById("fullview").src = thumb.src; }
     })
  } else {
     localStorage.maxPreference=1;
     thumbs.forEach(thumb => {
       localStorage.setItem(thumb.dataset.ord, thumb.dataset.ord);
     })
  }
```

Setting of first image shown, 2

```
<img id="fullview"  onclick='makeMax();'
                ondblclick ='myReset();' src="full/blank.jpg">
…
// Code to initially store or retrieve data.
  if ( localStorage.maxPreference){
     thumbs.forEach(thumb => {
       thumb.style.order=localStorage.getItem(thumb.dataset.ord);
        if (thumb.style.order==localStorage.maxPreference){
            document.getElementById("fullview").src = thumb.src; }
     })
  } else {
     localStorage.maxPreference=1;
     thumbs.forEach(thumb => {
        if (thumb.dataset.ord==1){
          document.getElementById("fullview").src = thumb.src;}
        localStorage.setItem(thumb.dataset.ord, thumb.dataset.ord);
     })
  }
```

Setting of first image shown, 2

```
<img id="fullview"  onclick='makeMax();'
                ondblclick ='myReset();' src="full/blank.jpg">
…
// Code to initially store or retrieve data.
  if ( localStorage.maxPreference){
     thumbs.forEach(thumb => {
       thumb.style.order=localStorage.getItem(thumb.dataset.ord);
        if (thumb.style.order==localStorage.maxPreference){
            document.getElementById("fullview").src = thumb.src; }
     })
  } else {
     localStorage.maxPreference=1;
     thumbs.forEach(thumb => {
        if (thumb.dataset.ord==1){
          document.getElementById("fullview").src = thumb.src;}
        localStorage.setItem(thumb.dataset.ord, thumb.dataset.ord);
     })
  }
```

Effort to implement Features

```
const thumbs = document.querySelectorAll('.thumb');

var hasStorage;
if (typeof(Storage) !== "undefined") {
  hasStorage = true;
  // Code to initially store or retrieve data.
  if ( localStorage.maxPreference){
                    thumbs.forEach(thumb => {

                  thumb.style.order=localStorage.getItem(thumb.dataset.ord);
                                    if
(thumb.style.order==localStorage.maxPreference){
                                       document.getElementById("fullview").src =
thumb.src;
                                    }
                    })
  } else {
                    localStorage.maxPreference=1;
                    thumbs.forEach(thumb => {
                                    if (thumb.dataset.ord==1){
                                       document.getElementById("fullview").src =
thumb.src;
                                    }
                                    localStorage.setItem(thumb.dataset.ord,
thumb.dataset.ord);})
  }
} else {
  // Sorry! No Web Storage support..
  hasStorage = false;
}

    var current;
    function changeMainImg(that) {
        document.getElementById("fullview").src=that.src;
                                    current=that;
    }
    function makeMax() {
                      localStorage.maxPreference--;
                                    if (current) {

current.style.order=localStorage.maxPreference;

                  localStorage.setItem(current.dataset.ord, localStorage.maxPreference);
                                    }
    }
```

Implementation effort becomes risk for usability

Lecture notes

- localStorage as an example of how our user interface moves from a web-page to a web application.
- Different functionality has to be treated differently if we move to persistent representation of data.
- Standard solutions can make it easier to offer the same level of comfort to various functionalities in a user interface.
- Simplifying the development of good user interfaces remains an ongoing goal.