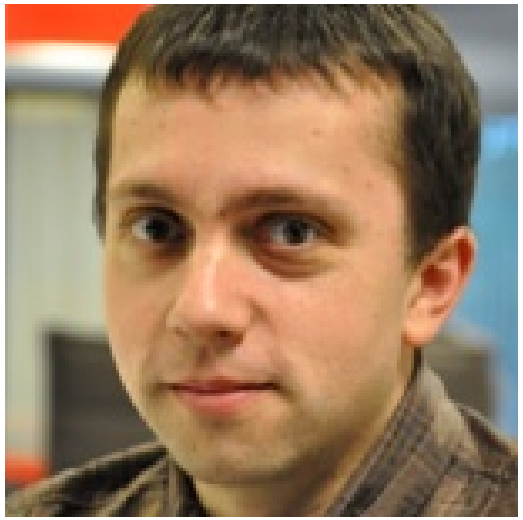
Adding Functionality Using Built-in JavaScript Libraries



Gill Cleeren

@gillcleeren

Outline

Drag and drop API

Web storage options

Geolocation

Target of This Module

We can drag coffee to the shopping basket

We can store the cart locally on the device

We can search for the user's geographical position

Drag and Drop API

Drag and Drop

Commonly used in websites

Previously required Javascript

HTML5 brings Drag and Drop API

events

attributes

dataTransfer

The New Attributes

draggable

```
<img draggable="true">
```

- true makes the element draggable
- auto follows browser default

dropzone

```
<div dropzone="copy"></div>
```

- Not supported in major browsers
- Not needed for drag and drop!

The New Events

- Available events
  - ondragstart
  - ondrag
  - ondragenter
  - ondragover
  - ondragleave
  - ondrop
  - ondragend

```
function handleDragStart(e) {
  this.style.opacity = '0.4';
  ev.dataTransfer.setData("Text", ev.target.id);
}
```

```
function drop(ev) {
    ev.preventDefault();
    var data = ev.dataTransfer.getData("Text");
    ev.target.appendChild(document.getElementById(data));
}
```

The dataTransfer Object

- dataTransfer object is the workhorse of the drag and drop functionality
  - Contains the data being sent during the drag and drop action
- Is set in the dragStart and read in the drop event

```
function dragstart(ev) {
   ev.dataTransfer.effectAllowed='copy';
   ev.dataTransfer.setData("Text", ev.target.getAttribute('id'));
   return true; }
```

```
function drop(ev) {
   var src = ev.dataTransfer.getData("Text");
   ev.target.appendChild(document.getElementById(src));
   return false;
}
```

The dataTransfer Object

- dataTransfer properties
  - dropEffect: used to specify which operation will be performed
    - Value must be value of effectAllowed values: can be all, none, copy, move, link…
  - files
  - types
- dataTransfer methods
  - setData: specifies the data to be dragged
    - format: mimetype
    - data: actual data
  - getData
  - clearData
  - setDragImage
  - addElement

```
var dragIcon = document.createElement('img');
dragIcon.src = 'logo.png';
dragIcon.width = 100;
e.dataTransfer.setDragImage(dragIcon, -10, -10);
```

Demo: Dragging Items into the Cart

Web storage options

Once upon a Time …

- Web apps only had cookies
  - Sent over the line with every request
  - Not secure
  - Limited to 4KB
  - 20 cookies per domain
  - Can be disabled by the user

  → Not a real solution

What We Need in Terms of Storage

More space

Client-side storage

Not deleted on refresh

Not sent over the wire

Web Storage

- Also known as local storage or DOM storage
- Supports persistent storage on the client
  - Better than cookies
  - Has a real API to use
  - Key/value pairs
  - Data is on the device, not transferred with every request
  - Created per domain
  - Supported in all major browsers
  - Limited to 5MB on most browsers

Web Storage

localStorage

- Persistent after browser close
- Spans browser windows and tabs
- Per domain

sessionStorage

- Deleted after browser/tab close
- Not shared between tabs or browser windows

Using Web Storage

```
<script>
// Check browser support
if (typeof(Storage) != "undefined") {
localStorage.setItem("coffee", "Arabica");
document.getElementById("result").innerHTML =
  localStorage.getItem("coffee");
}
else {
    document.getElementById("result").innerHTML =
      "Sorry, Web Storage not supported.";
}
</script>
```

Web Storage API

setItem	`localStorage.setItem("coffee", "Arabica");`
getItem	`var coffee = localStorage.getItem("coffee");`
removeitem	`localStorage.removeItem("coffee")`
clear	`localStorage.clear()`
length	`var storageSize = localStorage.length;`

Tracking Changes

- Being notified about changes is possible using the storage event
  - Called on setItem, removeItem and clear
  - Only when something did change
- Attributes
  - key
  - oldValue
  - newValue
  - url
  - storageArea

```
window.addEventListener("storage", logMyStorageEvents, false);
```

```
function logMyStorageEvents (e) {
   console.log(e.key);
   console.log(e.oldValue);
   console.log(e.newValue);
}
```

Demo: Saving the Shopping Cart Locally

Geolocation

Geolocation

- Allows browser to retrieve geographic location of the user
  - Latitude
  - Longitude
  - Height
  - Speed
  - …

  → Highly accurate
- Useful for
  - Weather sites
  - Map sites
  - Traffic information

Security Warning

- User gets warning about location tracking
  - Name of tracking site is displayed
  - Can't be bypassed
- User remains in control
  - Can accept or deny access to location
- User can decide if access if for one time or always per site

```
function getPosition() {
    if (navigator.geolocation) {
        navigator.geolocation.getCurrentPosition
          (showPosition, errorCallback);
    }
    else {
        infoParagraph.innerHTML = "Sorry, geolocation not supported";
    }
}
function showPosition(position) {
    infoParagraph.innerHTML="Latitude: " + position.coords.latitude +
    "<br>Longitude: " + position.coords.longitude;
}
function errorCallback(e) {
    infoParagraph.innerHTML = "Something is wrong, we can't find
you";
```

Geolocation API

- watchPosition: used to listen for changes to position of the user

```
var watchId;
watchId = navigator.geolocation.watchPosition(showPosition);
```

- clearWatch: used to stop listening for updates

```
navigator.geolocation.clearWatch(watchId);
```

Important Objects in the Geolocation API

- PositionOptions

```
var options = {
            timeout:5000,
             enableHighAccuracy: true,
             maximumAge:  1000
  };
watchID = navigator.geolocation.watchPosition(
            showPosition,
            errorCallback,
            options);
```

Important Objects in the Geolocation API

- Position:
  - used as container for results of successful location retrieval
  - Exposes Coordinates

```
function showPosition (position) {
     console.log(position.timestamp);
}
```

- Coordinates: contains all retrieved data
  - latitude
  - longitude
  - altitude
  - accuracy
  - speed
  - heading

```
function showPosition(position) {
  var latitude = position.coords.latitude;
  var longitude = position.coords.longitude;
}
```

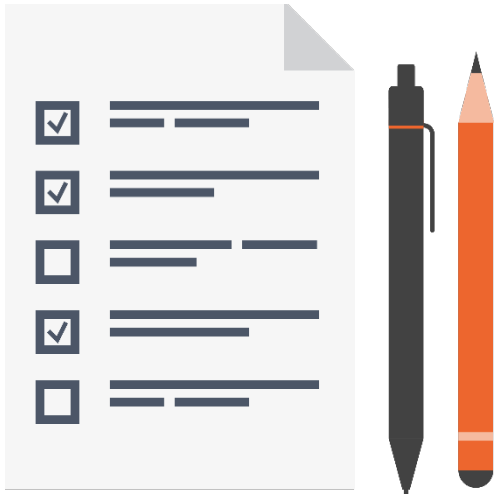
Important Objects in the Geolocation API

- PositionError: on error, the error callback is called
  - Receives a PositionError object

```
function showPositionError(error) {
    switch(error.code) {
        case error.PERMISSION_DENIED:
            infoParagraph.innerHTML = "User denied the geolocation request."
            break;
        case error.POSITION_UNAVAILABLE:
            infoParagraph.innerHTML = "No position information unavailable."
            break;
        case error.TIMEOUT:
            infoParagraph.innerHTML = "A timeout occurred."
            break;
        case error.UNKNOWN_ERROR:
            infoParagraph.innerHTML = "An unknown error occurred."
            break;
    }
}
```

Demo: Adding Support for Geolocation

Summary



Drag and drop enables desktop-like scenarios

Web storage creates more options than cookies did in the past

Geolocation can open up cool options for your sites