Date : 15-6-2020
Morning Session : 9am – 11 PM
By ~ Sundeep Charan Ramkumar Today

Topics: HTML5 API

API: API is service provided by software

Geolocation API
Storage API
Drag & Drop API

1) **Geolocation :** to get the geographical position of a user.

https://www.w3schools.com/html/html5_geolocation.asp

```
function successCallback(position) {
  console.log(position);
}

function errorCallback(err) {
  console.log(err);
}

if (navigator.geolocation) {
  navigator.geolocation.getCurrentPosition(successCallback, errorCallback);
} else {
  console.log("OOPS. My bad :(");
}
```

2) Storage API : two storages of API 1) local (json, string) , 2) session

Both storage objects provide same methods and properties:

- `setItem(key, value)` – store key/value pair.
- `getItem(key)` – get the value by key.
- `removeItem(key)` – remove the key with its value.
- `clear()` – delete everything.

The main features of **localStorage** are:

- Shared between all tabs and windows from the same origin.
- The data does not expire. It remains after the browser restart.

```
function fetchPosts() {
    // Trying to check whether a key called "users" exist.
    var users = localStorage.getItem("users");
    // Checking area
    if (users !== null) {
      console.log("Fetching from cache");
      // Converting string (JSON) to Javascript objects
      var usersArray = JSON.parse(users);
      // return new Promise(function (resolveFunction) {
      //   resolveFunction(usersArray);
      // });

      // Alliter
      return Promise.resolve(usersArray);
    } else {
      console.log("Fetching from server");
      var usersPromise = fetch(corsErrorRemoveURL + "https://reqres.in/api/users")
        .then(function (response) {
          return response.json();
        })
        .then(function (response) {
          return response.data;
        })
```

```
      .then(function (response) {
        console.log(typeof response);
        // Convert into JSON
        var responseJSON = JSON.stringify(response);
        // Storage into LOCAL
        localStorage.setItem("users", responseJSON);
        return response;
      })
      .catch(function (err) {
        console.log(err);
      });
    return usersPromise;
  }
}

fetchPosts()
  .then(function (users) {
    console.log(users);
  })
  .catch(function (err) {
    console.log(err);
  });
```

Ln 2, Col 1

https://javascript.info/localstorage

3) **Drag and Drop :** Drag and drop is a very common feature. It is when you "grab" an object and drag it to a different location **it require 5 event listeners**

https://www.w3schools.com/html/html5_draganddrop.asp

https://developer.mozilla.org/en-US/docs/Web/API/HTML_Drag_and_Drop_API

To make an element draggable, set the **draggable** attribute to

```
<h1>HTML5 APIs</h1>
<div class="container">
  <div style="background: orange;" class="orange" draggable="true"></div>
  <div style="background: red;" class="red" draggable="true"></div>
</div>
```

Event	On Event Handler	Fires when...
`drag`	`ondrag`	...a *dragged item* (element or text selection) is dragged.
`dragend`	`ondragend`	...a drag operation ends (such as releasing a mouse button or hitting the Esc key; see Finishing a Drag.)
`dragenter`	`ondragenter`	...a dragged item enters a valid drop target. (See Specifying Drop Targets.)
`dragexit`	`ondragexit`	...an element is no longer the drag operation's immediate selection target.
`dragleave`	`ondragleave`	...a dragged item leaves a valid drop target.
`dragover`	`ondragover`	...a dragged item is being dragged over a valid drop target, every few hundred milliseconds.
`dragstart`	`ondragstart`	...the user starts dragging an item. (See Starting a Drag Operation.)
`drop`	`ondrop`	...an item is dropped on a valid drop target. (See Performing a Drop.)

```
var draggableDivs = [...document.querySelectorAll('div[draggable="true"]')];
var container = document.querySelector(".container");
var primaryElement = null;
var secondaryElement = null;
container.addEventListener("dragend", function (event) {
  console.log("Drag end");
});
container.addEventListener("dragenter", function (event) {
  event.preventDefault();
  console.log("Drag enter");
});
container.addEventListener("dragexit", function (event) {
  console.log("Drag exit");
});
container.addEventListener("dragleave", function (event) {
  console.log("Drag leave");
});
container.addEventListener("dragover", function (event) {
  event.preventDefault();
});
```

```
container.addEventListener("dragstart", function (event) {
  primaryElement = event.target;
  console.log("Drag start");
});
container.addEventListener("drop", function (event) {
  // Getting the element beneath the draggable element
  secondaryElement = event.target;
  // Swapping the colors
  var tempColor = secondaryElement.style.background;
  secondaryElement.style.background = primaryElement.style.background;
  primaryElement.style.background = tempColor;
});
```

MCQ1:

The permissions for navigation is accepted by default after you accept it for the first time. True or False? 2

- [ ] False
- [x] True 76.92%

MCQ2:

To reveal the location coordinates, which method is used? 2

- [ ] watchCurrentPosition
- [x] getCurrentPosition 63.08%
- [ ] getPosition

MCQ3:

Session storage is used for permanent persistence. True or False? Attempted - 57 (87.69%) EASY

- [ ] True 38.6%
- [x] False 61.4%

MCQ4:

Which drag and drop event is useful to know whether the draggable element is out of the checking boundary? Attempted - 56 (86.15%) EASY

- [ ] dragExit 21.43%
- [ ] dragEnd 26.79%
- [x] dragLeave 53.57%
- [ ] dragStop 8.93%

Website to get fake DATA real quick http://jsonplaceholder.typicode.com/