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);
      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

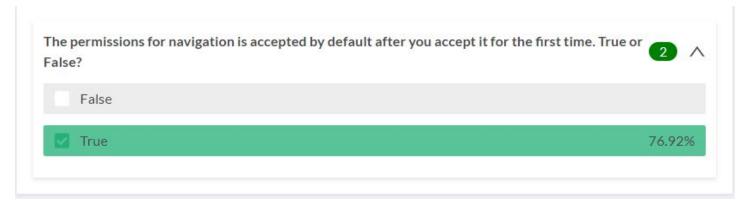
To make an element draggable, set the draggable attribute to

| 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.)                                    |

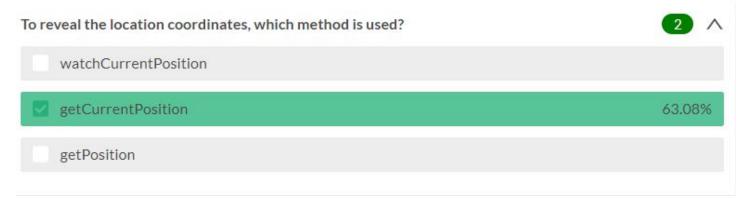
```
var draggableDivs = [...document.querySelectorAll('div[draggable="true"]')];
var container = document.querySelector(".container");
var primaryElement = null;
var secondaryElement = null;
container.addEventListener("dragend", function (event) {
  console.log("Drag end");
});
container.addEventListener("dragenter",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



## MCQ2:



### MCQ3:



### MCQ4:

