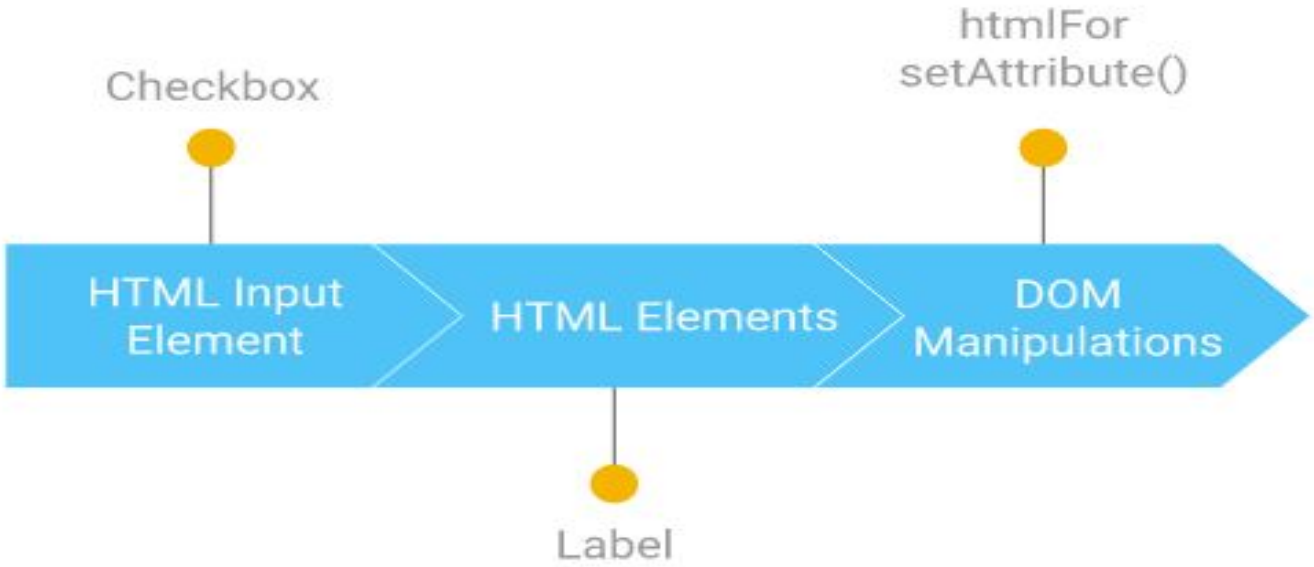
Todoapp

Explanation

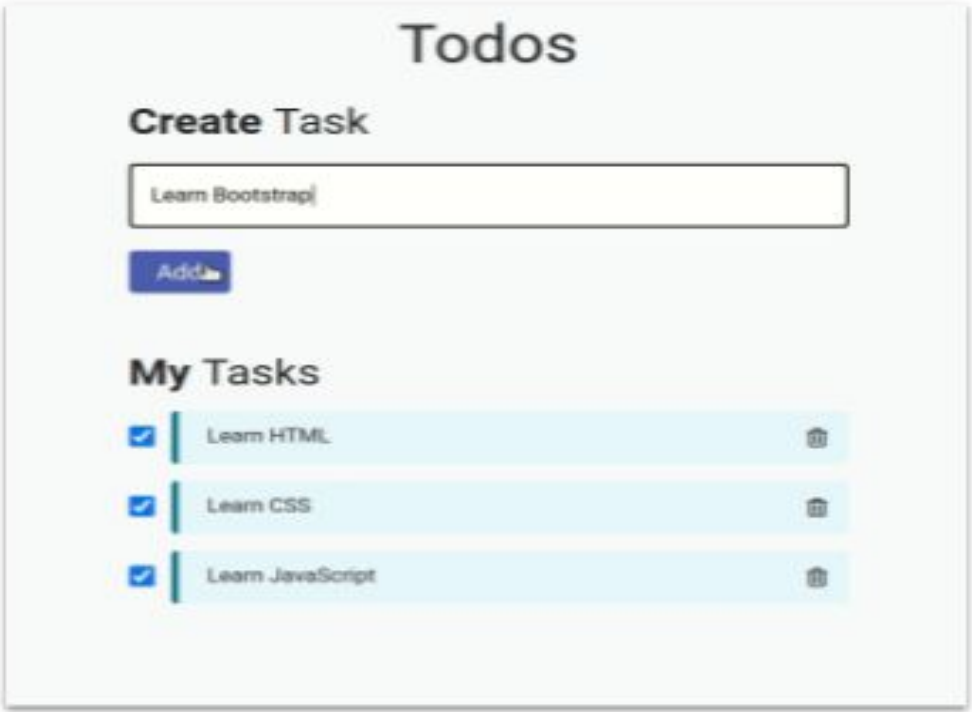
Agenda

Todos Application



Example

Todos Application



How to add

A Checkbox

Statically using HTML?

Checkbox

Label

HTML

```
<body>
  <input type="checkbox" id="myCheckbox" />
  <label for="myCheckbox">Graduated</label>
</body>
```

DOM Manipulations

Creating Checkbox Input Dynamically

HTML

```html
<input type="checkbox" id="myCheckbox" />
<label for="myCheckbox">Graduated</label>
```

JS

```js
let inputElement = document.createElement('input');
inputElement.type = "checkbox";
inputElement.id = "myCheckbox";
document.body.appendChild(inputElement);
```

DOM Manipulations

Creating Checkbox Input Dynamically

HTML

```html
<input type="checkbox" id="myCheckbox" />
<label for="myCheckbox">Graduated</label>
```

JS

```js
let labelElement = document.createElement('label');
labelElement.htmlFor = "myCheckbox";
labelElement.textContent = "Graduated";
document.body.appendChild(labelElement);
```

DOM Manipulations

setAttribute()

To set a **value of an attribute** for a **specified element,** we use setAttribute() method

```
element.setAttribute(attribute, value);
```

If the attribute already exists, the value of the attribute gets **updated**

Powered by NXT WAVE IB HUBS

DOM Manipulations

Creating Checkbox Input Dynamically

HTML

```
<input type="checkbox" id="myCheckbox" />
<label for="myCheckbox">Graduated</label>
```

JS

```
let labelElement = document.createElement('label');
labelElement.setAttribute("for", "myCheckbox");
labelElement.textContent = "Graduated";
document.body.appendChild(labelElement);
```

Powered by NXT WAVE IB HUBS

Prefilled Code

HTML Code

HTML

```
...
<body>
  <div class="todos-bg-container">
    <div class="container">
      ...
    </div>
  </div>
</body>
...
```

Prefilled Code

CSS Code

CSS

```
.todos-bg-container {
  background-color: #f9fbfe;
  height: 100vh;
}
.todos-heading {
  font-family: "Roboto";
  font-size: 46px;
  ...
}
...
```

Example

Todos Application

Steps:

- Create a Single Todo Item
- Create Multiple Todo Items
- Take User Input and Create Todos Dynamically
- Add Delete Todo Item Functionality

Creating Todo Item

Todo List Item

HTML

```
<ul class="todo-items-container" id="todoItemsContainer">
  <li class="todo-item-container d-flex flex-row">
  </li>
</ul>
```

CSS

```
.todo-item-container {
  margin-top: 15px;
}
```

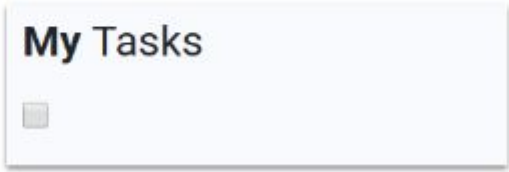
Creating Todo Item

Checkbox Input

HTML

```
<ul class="todo-items-container" id="todoItemsContainer">
  <li class="todo-item-container d-flex flex-row">
    <input type="checkbox" class="checkbox-input" />
  </li>
</ul>
```

```
.checkbox-input {
  width: 20px;
  height: 20px;
  margin-top: 12px;
  margin-right: 12px;
}
```



NXT WAVE

Creating Todo Item

Adding Label Container

HTML

```
...
<input type="checkbox" class="checkbox-input" />
<div class="d-flex flex-row label-container">

</div>
...
```

NXT WAVE

Creating Todo Item

Adding Label Container

HTML

```
.label-container {
  background-color: #e6f6ff;
  width: 100%;
  border-radius: 4px;
  border-style: solid;
  border-width: 5px;
  border-color: #096f92;
  border-right: none;
  border-top: none;
  border-bottom: none;
}
```

Learn HTML

#e6f6ff

#096f92

Powered by NXT WAVE IB HUBS

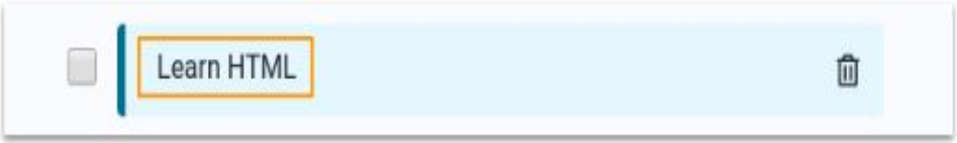
Adding Label Container

Output

My Tasks

Creating Label

Adding for and id attributes



HTML

```
...
<input type="checkbox" id="checkboxInput" class="checkbox-input" />

<div class="d-flex flex-row label-container">
  <label for="checkboxInput" class="checkbox-label">
    Learn HTML
  </label>
</div>
...
```

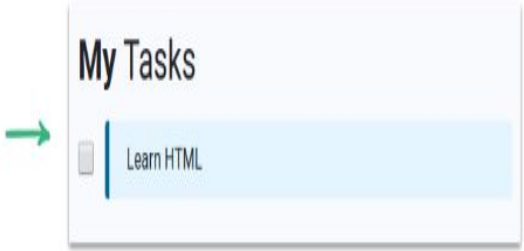
NXT WAVE

Creating Todo Item

Adding Label

CSS

```css
.checkbox-label {
  font-family: "Roboto";
  font-size: 16px;
  font-weight: 400;
  width: 82%;
  margin: 0px;
  padding-top: 10px;
  padding-bottom: 10px;
  padding-left: 20px;
  padding-right: 20px;
  border-radius: 5px;
}
```



Creating Todo Item

Adding Delete Icon

HTML

```html
...
  <div class="d-flex flex-row label-container">
    ...
    <div class="delete-icon-container">
      <i class="far fa-trash-alt delete-icon"></i>
    </div>
  </div>
...
```

Creating Todo Item

Adding Delete Icon

CSS

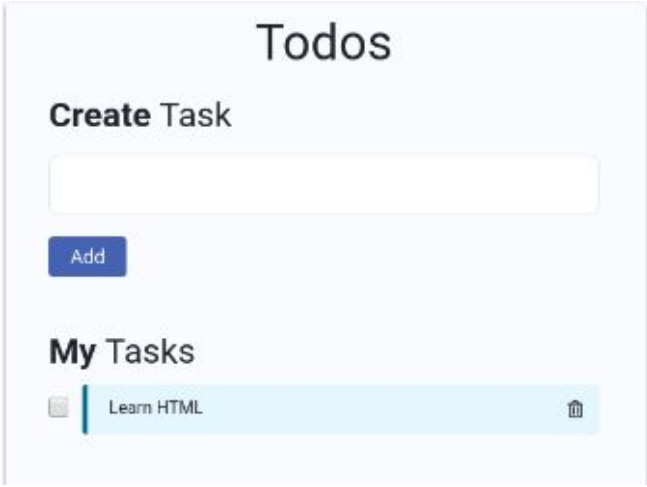
```css
.delete-icon-container {
  text-align: right;
  width: 18%;
}

.delete-icon {
  padding: 15px;
}
```

Creating Todo Item

Todo Item

HTML

```
<ul class="todo-items-container" id="todoItemsContainer">
  <li class="todo-item-container d-flex flex-row">
    <input type="checkbox" id="checkboxInput" class="checkbox-input" />
    <div class="d-flex flex-row label-container">
      <label for="checkboxInput" class="checkbox-label">
        Learn HTML
      </label>
      <div class="delete-icon-container">
        <i class="far fa-trash-alt delete-icon"></i>
      </div>
    </div>
  </li>
</ul>
```



Lecture:2 Let’s create todo dynamically

Creating Todo Item Dynamically

Application Flow

Powered by NXT WAVE iB HUBS

Todos Application

Creating a Todo Element

HTML

```
<ul class="todo-items-container" id="todoItemsContainer">
  <li class="todo-item-container d-flex flex-row">
  </li>
</ul>
```

JS

```
let todoElement = document.createElement("li");
todoElement.classList.add("todo-item-container", "d-flex", "flex-row");
```

NXT WAVE

Todos Application

Creating a Todo Element

HTML

```
<ul class="todo-items-container" id="todoItemsContainer">
  <li class="todo-item-container d-flex flex-row">
  </li>
</ul>
```

JS

```
let todoElement = document.createElement("li");
todoElement.classList.add("todo-item-container", "d-flex", "flex-row");
```

```
todoItemsContainer.appendChild(todoElement);
console.log(todoItemsContainer);
```

NXT WAVE

Todos Application

Creating a Checkbox

HTML

```
<li class="todo-item-container d-flex flex-row">
  <input type="checkbox" id="checkboxInput" class="checkbox-input" />
</li>
```

JS

```
let inputElement = document.createElement("input");
inputElement.type = "checkbox";
inputElement.id = "checkboxInput";
inputElement.classList.add("checkbox-input");

todoElement.appendChild(inputElement);
```

Todos Application

Creating a Label Container

HTML

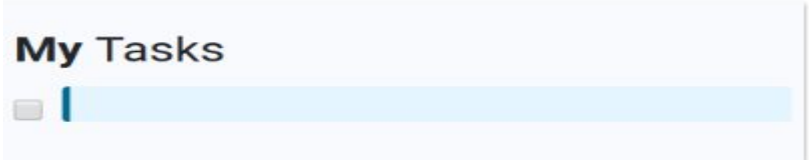
```
...
<input type="checkbox" id="checkboxInput" class="checkbox-input" />
<div class="d-flex flex-row label-container">
</div>
...
```

JS

```
let labelContainer = document.createElement("div");
labelContainer.classList.add("label-container", "d-flex", "flex-row");
todoElement.appendChild(labelContainer);
```

NXT WAV

Creating label container



Todos Application

Creating a Label Element

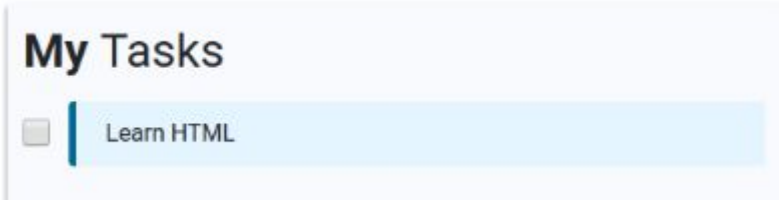
HTML

```html
<div class="d-flex flex-row label-container">
  <label for="checkboxInput" class="checkbox-label">
    Learn HTML
  </label>
</div>
```

JS

```js
let labelElement = document.createElement("label");
labelElement.setAttribute("for", "checkboxInput");
labelElement.classList.add("checkbox-label");
labelElement.textContent = "Learn HTML";

labelContainer.appendChild(labelElement);
```

output:



Todos Application

Creating a Delete Icon Container

HTML

```
<ul class="todo-items-container" id="todoItemsContainer">
  <li class="todo-item-container d-flex flex-row">
    <input type="checkbox" id="checkboxInput" class="checkbox-input" />
    <div class="d-flex flex-row label-container">
      <label for="checkboxInput" class="checkbox-label">
        Learn HTML
      </label>
      <div class="delete-icon-container">
        <i class="far fa-trash-alt delete-icon"></i>
      </div>
    </div>
  </li>
</ul>
```

NXT WAVE

Todos Application

Creating a Delete Icon Container

HTML

```
<div class="d-flex flex-row label-container">
  ...
  <div class="delete-icon-container">
    <i class="far fa-trash-alt delete-icon"></i>
  </div>
</div>
```

JS

```
let deleteIconContainer = document.createElement("div");
deleteIconContainer.classList.add("delete-icon-container");

labelContainer.appendChild(deleteIconContainer);
```

Todos Application

Adding Icon

HTML

```
<ul class="todo-items-container" id="todoItemsContainer">
  <li class="todo-item-container d-flex flex-row">
    <input type="checkbox" id="checkboxInput" class="checkbox-input" />
    <div class="d-flex flex-row label-container">
      <label for="checkboxInput" class="checkbox-label">
        Learn HTML
      </label>
      <div class="delete-icon-container">
        <i class="far fa-trash-alt delete-icon"></i>
      </div>
    </div>
  </li>
</ul>
```

NXT WAVE

Todos Application

Adding Icon

HTML

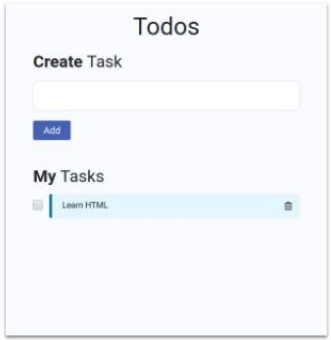
```
<div class="d-flex flex-row label-container">
  ...
  <div class="delete-icon-container">
    <i class="far fa-trash-alt delete-icon"></i>
  </div>
</div>
```

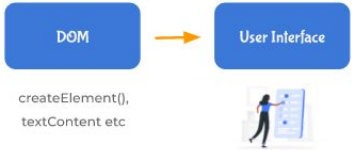
JS

```
let deleteIcon = document.createElement("i");

deleteIcon.classList.add("far", "fa-trash-alt", "delete-icon");

deleteIconContainer.appendChild(deleteIcon);
```

Adding Icon

Output





Creating single todo and multiple todo

Single todo

```
let todoItemsContainer = document.getElementById("todoItemsContainer");

let todo1 = {
  text: "Learn HTML"
}
...
```

Multiple todo

```
...
let todo1 = {
  text: "Learn HTML"
}
let todo2 = {
  text: "Learn CSS"
}
let todo3 = {
  text: "Learn JavaScript"
}
...
```

Creating Multiple Todo Items

Creating List of Todo Objects

JS

```
...
let todoList = [
  {
    text: "Learn HTML"
  },
  {
    text: "Learn CSS"
  },
  {
    text: "Learn JavaScript"
  }
];
...
```

Creating Reusable Function

JS

```
...
function createAndAppendTodo() {
  let todoElement = document.createElement("li");
  ...
  ...
  let deleteIcon = document.createElement("i");
  deleteIcon.classList.add("far", "fa-trash-alt", "delete-icon");
  deleteIconContainer.appendChild(deleteIcon);
}
```

Making Corresponding Changes

JS

```
...
function createAndAppendTodo(todo) {
  ...
  let labelElement = document.createElement("label");
  labelElement.setAttribute("for", "checkboxInput");
  labelElement.classList.add("checkbox-label");
  labelElement.textContent = todo.text;
  labelContainer.appendChild(labelElement);
  ...
}
```

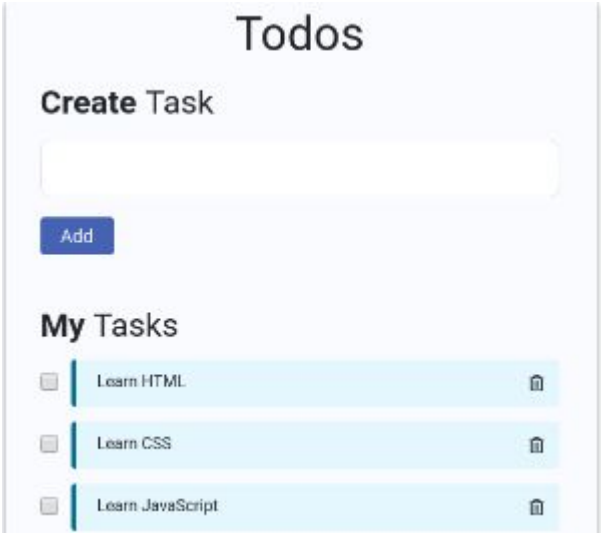
Creating List of Todo Objects

JS

```
...
let todoList = [
  {
    text: "Learn HTML"
  },
  {
    text: "Learn CSS"
  },
  {
    text: "Learn JavaScript"
  }
];
...
```

```
...
createAndAppendTodo(todoList[0]);
createAndAppendTodo(todoList[1]);
createAndAppendTodo(todoList[2]);
```

output:



Can we minimize

Duplication of code?

JS

```
createAndAppendTodo(todoList[0]);
createAndAppendTodo(todoList[1]);
createAndAppendTodo(todoList[2]);
createAndAppendTodo(todoList[3]);
createAndAppendTodo(todoList[4]);
createAndAppendTodo(todoList[5]);
...
```

Loops

Loops allow us to **execute** a block of code **several times**

- **for...of Loop**
- for...in Loop
- for Loop
- while Loop

many more...

Loops

The for...of Loop

Python

Code

```python
my_list = [1, 2, 3, 4];
for each_item in my_list:
    print(each_item)
```

JavaScript

Code

```javascript
let myArray = [1, 2, 3, 4];
for (let eachItem of myArray) {
    console.log(eachItem);
}
```

Creating Multiple Todo Items

Adding Todo Items to Todos List

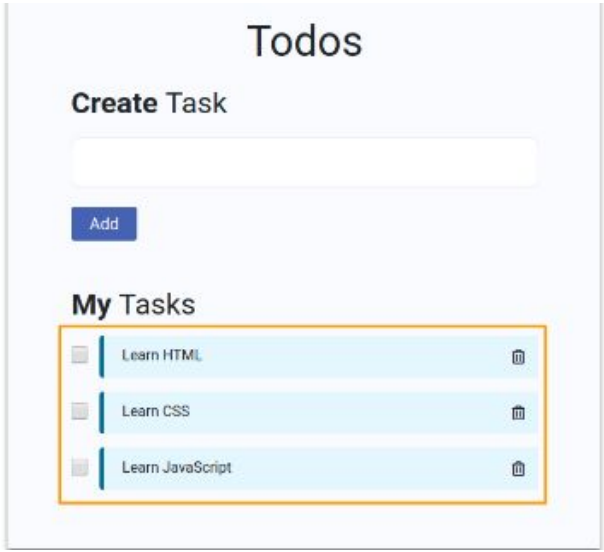
JS

```
[
  {
    text: "Learn HTML"
  },
  {
    text: "Learn CSS"
  },
  {
    text: "Learn JavaScript"
  }
];
```

JS

```
for (let todo of todoList) {
  createAndAppendTodo(todo);
}
```

- Learn HTML
- Learn CSS
- Learn JavaScript



Todos Application

Features

- [x] Creating **Single** Todo Item
- [x] Creating **Multiple** Todo Items
- [ ] Taking User Input and creating Todos **Dynamically**
- [ ] **Checking** a Todo
- [ ] **Deleting** a Todo
- [ ] **Persisting** Todos On Reload using Local Storage

On-Demand Session | Cheat Sheet

1. Most Commonly Made Mistakes

1.1 Most of the JS properties and methods should be in the Camel case.

Most of the JS properties and methods are in the Camel case (the starting letter of each word should be in uppercase except for the first word).

Code	Mistake	Correct Syntax
document.CreateElement()	C in Uppercase	document.createElement()
document.getElementbyId()	b in Lowercase	document.getElementById()
element.textcontent	c in Lowercase	element.textContent
element.classlist.add()	l in Lowercase	element.classList.add()

1.2 The ID should be the same in both the HTML and JS.

1.2.1 Mistake:

HTML

```
1 <h1 id="heading">Shopping List</h1>
```

JAVASCRIPT

```
1 let headingEl = document.getElementById("listHeading");
2 headingEl.textContent = "Items Needed";
```

In the above Code Snippets, the HTML element's text content doesn't change because the ID used in HTML and JS are different.

So, While accessing an HTML element with the ID using JS, **the ID used in the HTML element and the `document.getElementById` method must be the same**.

HTML

```
1 <h1 id="heading">Shopping List</h1>
```

JAVASCRIPT

```
1 let headingEl = document.getElementById("heading");
2 headingEl.textContent = "Items Needed";
```

1.2.2 Mistake:

HTML

```
<h1 id="listHeading ">Shopping List</h1>
```

JAVASCRIPT

```
let headingEl = document.getElementById("listHeading");
headingEl.textContent = "Items Needed";
```

The HTML element's text content doesn't change because there is an extra space at the end of the ID in the HTML code.

So, there shouldn't be any extra spaces in the IDs used in both the HTML and JS.

HTML

```
<h1 id="listHeading">Shopping List</h1>
```

JAVASCRIPT

```
let headingEl = document.getElementById("listHeading");
headingEl.textContent = "Items Needed";
```

1.3. The Function name must be the same in both the Function declaration and the Function call.

1.3.1 Mistake:

JAVASCRIPT

```javascript
function greeting() {
  let message = "Hello Rahul";
  console.log(message);
}

greet();
```

As there is no function called `greet` , we will get an error in the above Code Snippet.

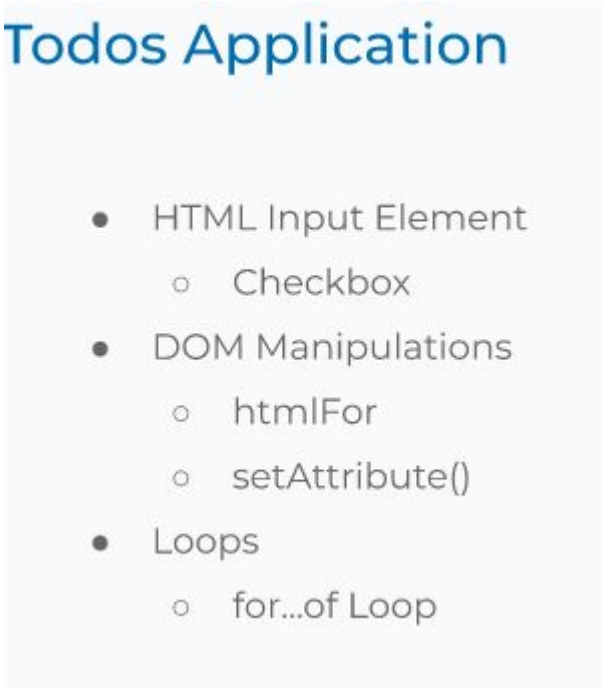
So, while calling a function, you must use the same function name used in the function declaration.

JAVASCRIPT

```javascript
function greeting() {
  let message = "Hello Rahul";
  console.log(message);
}

greeting();
```

Lecture :3 todo

Recap lectures

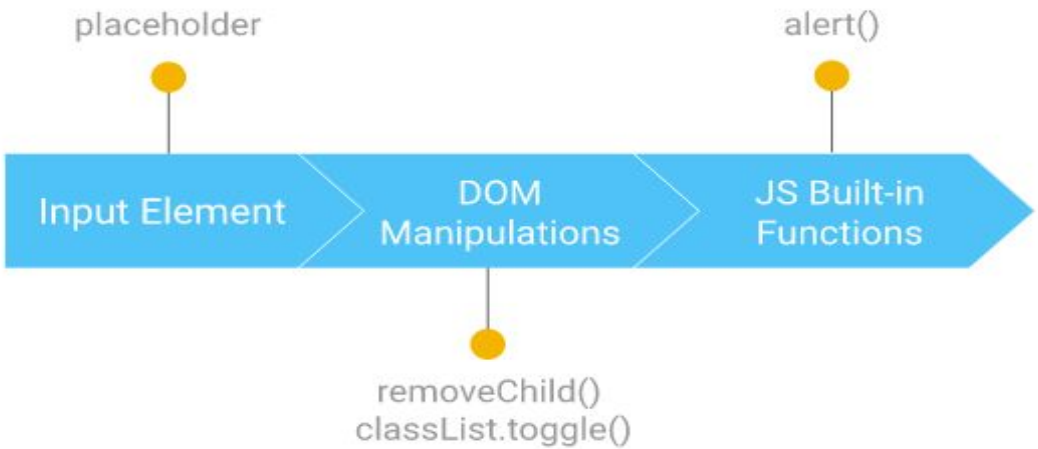


Recap of lectures

Lecture3 :agenda

Agenda

Todos Application



Todos Application

HTML Code

HTML

```
...
<div class="todos-bg-container">
  <div class="container">
    <div class="row">
      ...
    </div>
  </div>
</div>
...
```

Todos Application

CSS Code

CSS

```
.todos-bg-container {
  background-color: #f9fbfe;
  height: 100vh;
}
.todos-heading {
  text-align: center;
  font-family: "Roboto";
  font-size: 46px;
  ...
}
...
```

JS Code

JS

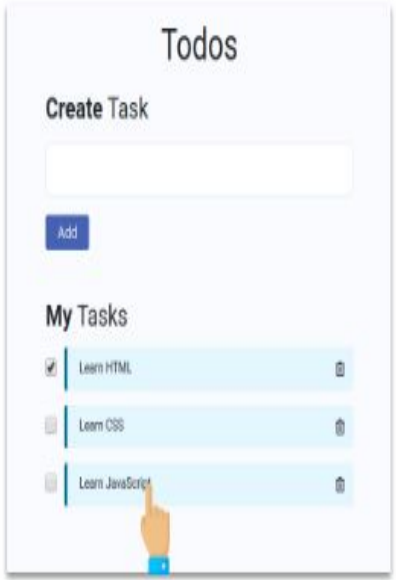
```
let todoItemsContainer = document.getElementById("todoItemsContainer");
let todoList = [
  {
    text: "Learn HTML",
  },
  ...
];
function createAndAppendTodo(todo) {
  let todoElement = document.createElement("li");
  ...
  deleteIconContainer.appendChild(deleteIcon);
}
...
```

Todos Application

Fixing Checkbox Issue

We have to specify a **Unique ID** to each Checkbox.

Provide the same ID to the labels **for** attribute.



Fixing Checkbox Issue

Specifying Unique ID

JS

```
let todoList = [
 {
   text: "Learn HTML",
   uniqueNo: 1
 },
 {
   text: "Learn CSS",
   uniqueNo: 2
 },
 {
   text: "Learn JavaScript",
   uniqueNo: 3
 }
];
```

Fixing Checkbox Issue

Adding ID to each Checkbox

JS

```
...
function createAndAppendTodo(todo) {
 let checkboxId = "checkbox" + todo.uniqueNo;
 ...
 inputElement.type = "checkbox";
 inputElement.id = checkboxId;
 ...
 labelElement.setAttribute("for", checkboxId);
 labelElement.classList.add("checkbox-label");
 ...
}
...
```

"checkbox" + 1 → "checkbox1"

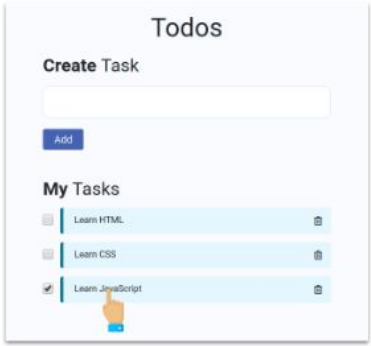
"checkbox" + 2 → "checkbox2"

"checkbox" + 3 → "checkbox3"

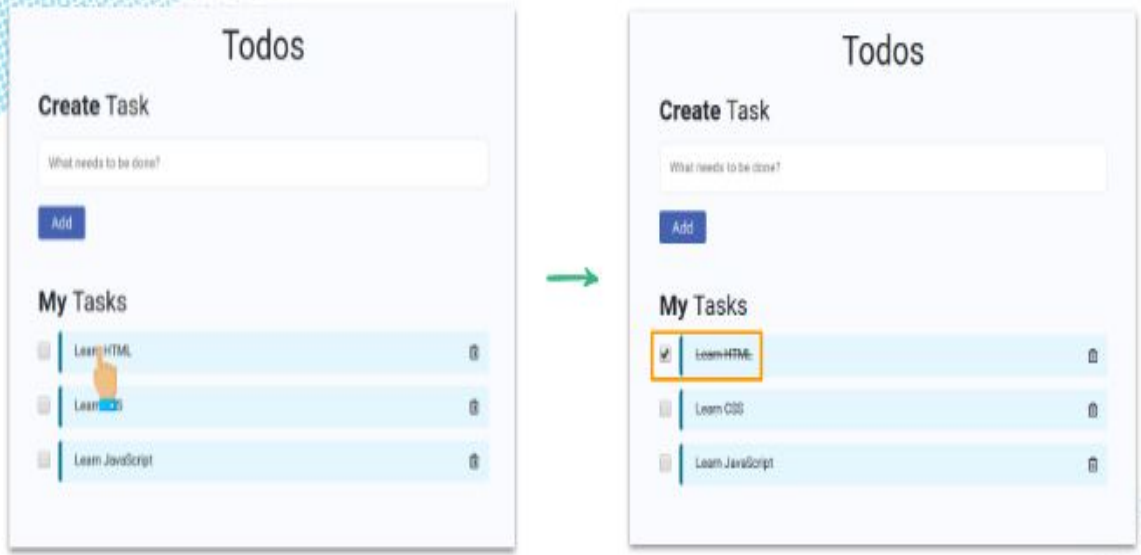
Powered by NXT WAVE | iB HUBS

Fixing Checkbox Issue

Output



How to Strikethrough the Label Text on Checking Checkbox?



Todos Application

Strikethrough the Label Text

Steps:

- Add Required CSS to Strike a Text
- Specify Unique Id to each Label Element
- Add Event Listeners to the Checkboxes
- Change Styles of Label Element based on Checkbox Check

NXT WAVE

Step 1

Adding Required CSS to Strike a Text

CSS

```css
.checked {
  text-decoration: line-through;
}
```

Step 2

Specifying ID to each Label Element

JS

```js
...
function createAndAppendTodo(todo) {
  let checkboxId = "checkbox" + todo.uniqueNo;
  let labelId = "label" + todo.uniqueNo;
  ...
  let labelElement = document.createElement("label");
  labelElement.setAttribute("for", checkboxId);
  labelElement.id = labelId;
}
...
```

```
"label" + 1  ⟶  "label1"
"label" + 2  ⟶  "label2"
"label" + 3  ⟶  "label3"
```

Step 3

Adding Event Listeners to Checkboxes

JS

```js
...
function createAndAppendTodo(todo) {
  let checkboxId = "checkbox" + todo.uniqueNo;
  let labelId = "label" + todo.uniqueNo;
  ...
  let inputElement = document.createElement("input");
  inputElement.type = "checkbox";
  inputElement.id = checkboxId;
  inputElement.onclick = function () {
    onTodoStatusChange(checkboxId, labelId);
  };
  ...
}
```

Step 4 - Change Styles of Label Element based on Checkbox Check

Accessing Label Elements

JS

```
...
function onTodoStatusChange(checkboxId, labelId) {
  let checkboxElement = document.getElementById(checkboxId);
  console.log(checkboxElement.checked);

  let labelElement = document.getElementById(labelId);
}
...
```

Step 4

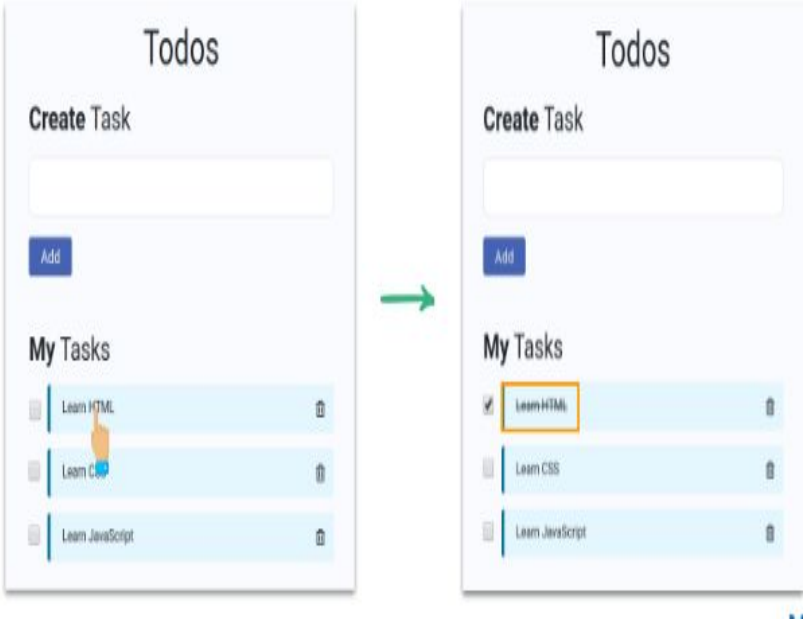
Changing Label Element Styles

JS

```
...
function onTodoStatusChange(checkboxId, labelId) {
  ...
  if (checkboxElement.checked === true) {
    labelElement.classList.add("checked");
  }
  else {
    labelElement.classList.remove("checked");
  }
}
```

Change styles of Label Element based on Checkbox Check

Output



DOM Manipulations

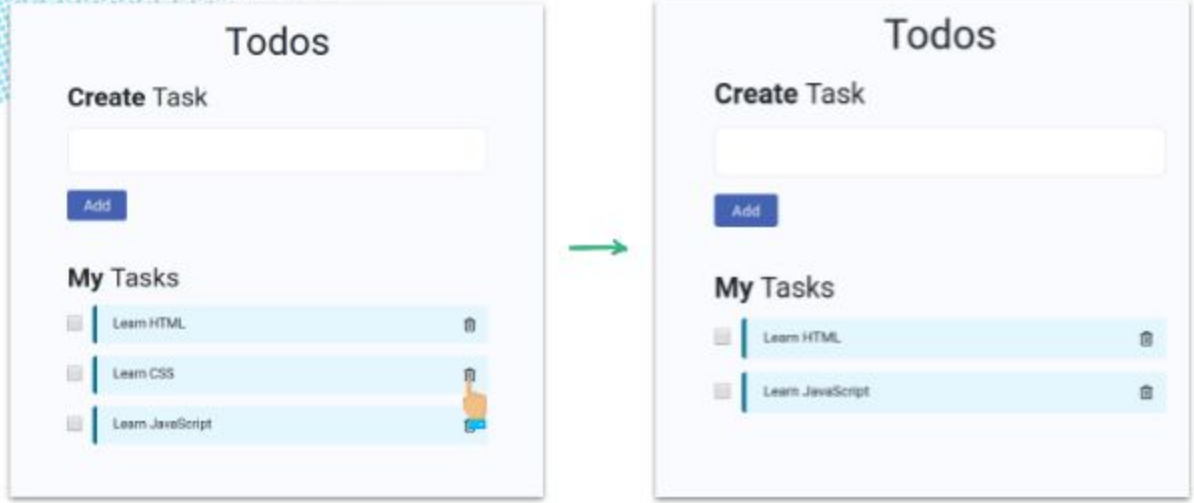
classList.toggle()

Toggle Button

JS

```
...
function onTodoStatusChange(checkboxId, labelId) {
  ...
  labelElement.classList.toggle("checked");
}
```

How to Delete a Todo Item?



Todos Application

Deleting a Todo Item

Steps:

- Specify ID to each Todo Item
- Add Event Listeners to Delete Icon
- Delete Todo Item from the Todo Items Container

Step 1

Specifying ID to each Todo Item

JS

```
...
function createAndAppendTodo(todo) {
  let todoId = "todo" + todo.uniqueNo;
  ...
  let todoElement = document.createElement("li");
  todoElement.classList.add("todo-item-container", "d-flex", "flex-row");
  todoElement.id = todoId;
  todoItemsContainer.appendChild(todoElement);
  ...
}
```

Step 2

Adding Event Listeners to Delete Icon

JS

```
...
let deleteIcon = document.createElement("i");
deleteIcon.classList.add("far", "fa-trash-alt", "delete-icon");
deleteIcon.onclick = function () {
  onDeleteTodo(todoId);
};
deleteIconContainer.appendChild(deleteIcon);
...
```

Step 3

Deleting a Todo Item

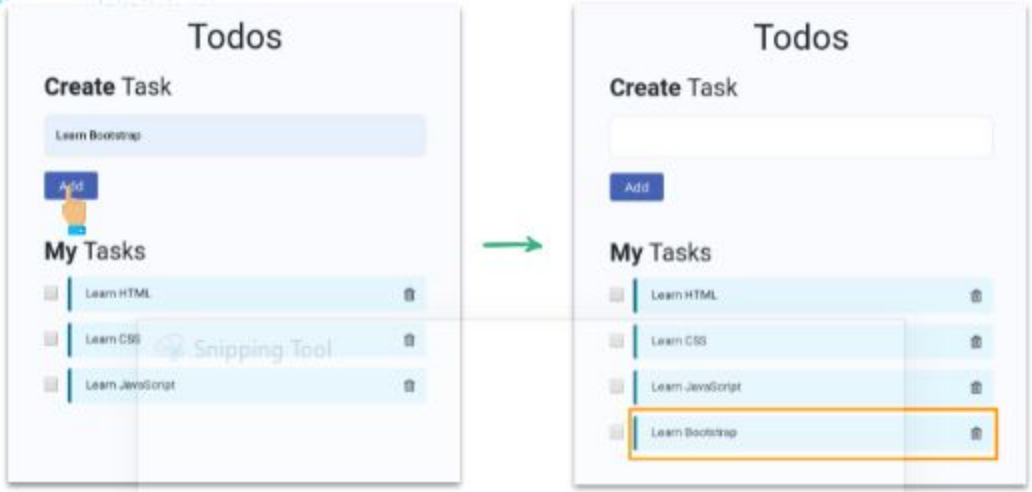
Deleting a Todo Item

Output


```
...
function onDeleteTodo(todoId) {
  let todoElement = document.getElementById(todoId);
  todoItemsContainer.removeChild(todoElement);
}
...
```

Adding a Todo Item

How to Add a new Todo Item on User Input?



Steps:

- Add Event Listener to the Add Button
- Access User Input Value
- Create New Todo Item

Step 1

Adding Event Listeners to the Add Button

HTML

```
<div class="todos-bg-container">
  <div class="container">
    <div class="row">
      ...
      <button class="add-todo-button" id="addTodoButton">Add</button>
      ...
    </div>
  </div>
</div>
```

Step 1

Adding Event Listeners to the Add Button

JS

```
let todoItemsContainer = document.getElementById("todoItemsContainer");
let addTodoButton = document.getElementById("addTodoButton");
...
addTodoButton.onclick = function () {
  onAddTodo();
};
```

Accessing user input:

HTML

```
...
function onAddTodo() {
  let userInputElement = document.getElementById("todoUserInput");
  let userInputValue = userInputElement.value;
}
```

Step 3

Clearing User Input Value

JS

```
...
function onAddTodo() {
  ...
  todosCount = todosCount + 1;
  let newTodo = {
    text: userInputValue,
    uniqueNo: todosCount,
  };
  createAndAppendTodo(newTodo);
  userInputElement.value = "";
}
```

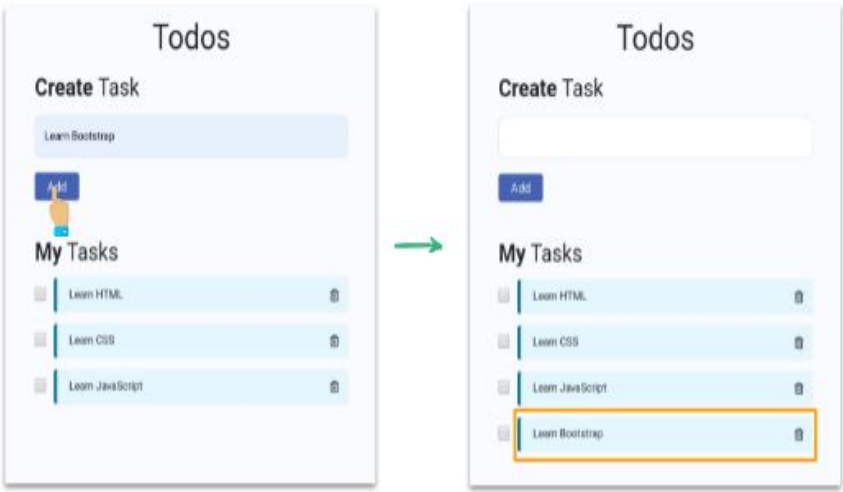
Step 3

Creating a New Todo Item

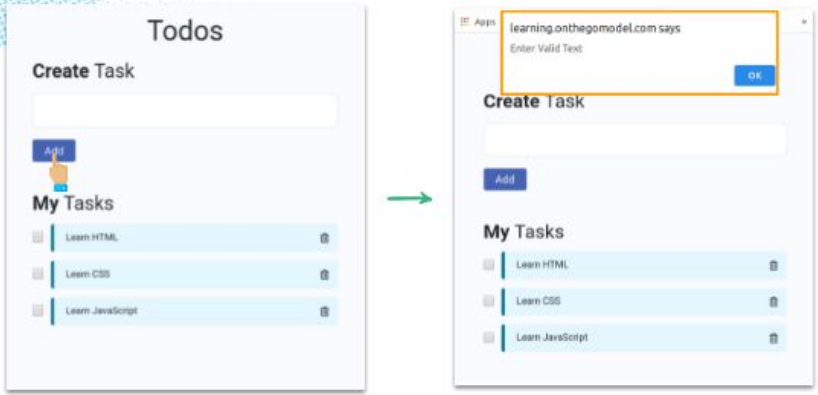
JS

```
...
let todosCount = todoList.length;
function onAddTodo() {
  ...
  todosCount = todosCount + 1;
  let newTodo = {
    text: userInputValue,
    uniqueNo: todosCount
  };
  createAndAppendTodo(newTodo);
}
...
```

Output



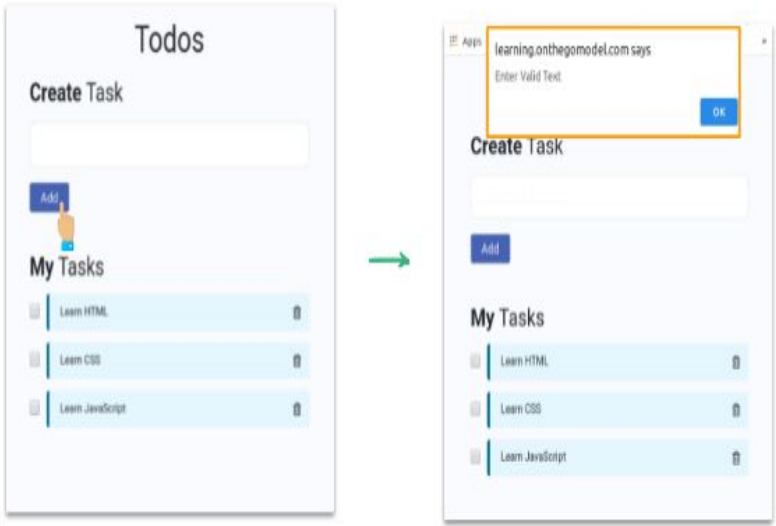
How to show a Warning Message on Invalid User Input?



Showing Warning Message

```
JS
...
function onAddTodo() {
  let userInputElement = document.getElementById("todoUserInput");
  let userInputValue = userInputElement.value;
  if (userInputValue === "") {
    alert("Enter Valid Text");
    return;
  }
  ...
}
```


Output



Input Element

Adding Placeholder Text

HTML

```
<body>
  <input type="text" placeholder="Enter your name" />
</body>
```

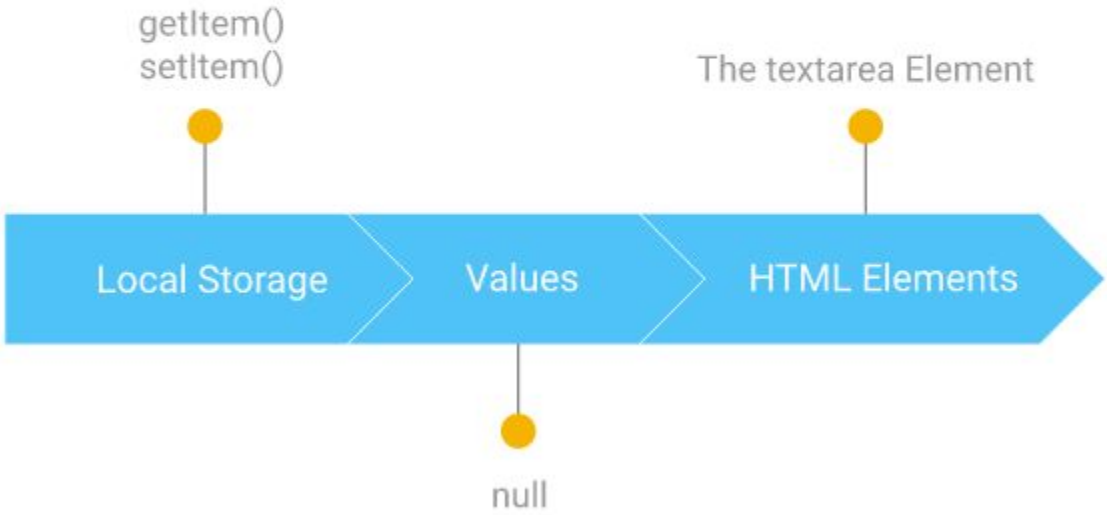
Key Takeaways

- Input Element
  - placeholder
- DOM Manipulations
  - classList.toggle()
  - removeChild()
- JS Built-in Functions
  - alert()

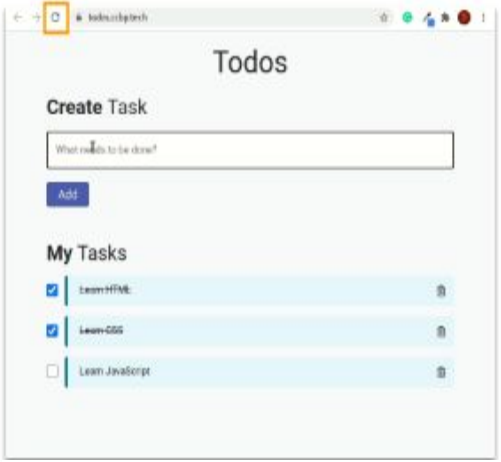
lecture:4

Agenda

Todos Application



What happens to Todo List when we reload the application?



Todos Application

Execution Context

The environment in which JS Code runs is called Execution Context

Execution context contains all the variables, objects, and functions

Variables
Objects
Functions

Example

Counter Application

Execution Context

On Reloading

Striked aswell as deleted an
stored element will be delete

Execution Context is destroyed and recreated whenever we reload an Application.

How to persist todo items on reload?

Persisting Data even on Reload

Storage Mechanisms

Client-Side Data Storage

- Storing Data on the **Client** (user's machine)

Server-Side Data Storage

- Storing Data on the **Server** using some kind of Database

Storage Mechanisms

Client-Side Data Storage Mechanisms

- **Local Storage**
- Session Storage
- Cookies
- IndexedDB

many more...

Local Storage

It allows web applications to store **data locally** within the **User's Browser**

It is a Storage **Object**

Data can be stored in the form of **key-value** pairs

Value provided should always be a **string**

Key	Value
name	Rahul
gender	Male
city	Delhi

Client-Side Data Storage Mechanisms

Local Storage

To access and work with Local Storage, we have below methods:

- setItem()
- getItem()
- clear()
- removeItem()

Storing Data in Local Storage

setItem()

Syntax:

```
localStorage.setItem("Key", "Value");
```

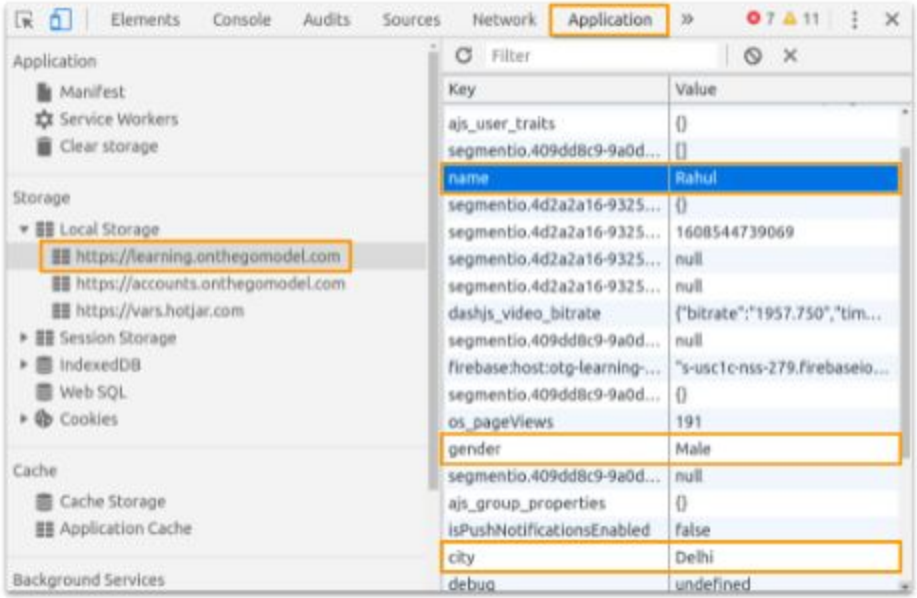
JS

```
localStorage.setItem("name", "Rahul");
localStorage.setItem("gender", "Male");
localStorage.setItem("city", "Delhi");
```

Key	Value
name	Rahul
gender	Male
city	Delhi

Storing Data in Local Storage

setItem()



Getting Data from Local Storage

getItem()

Syntax:

```
localStorage.getItem("Key");
```

JS

```
let name = localStorage.getItem("name");
let gender = localStorage.getItem("gender");
let city = localStorage.getItem("city");

console.log(name);
console.log(gender);
console.log(city);
```

Output

```
Rahul
Male
Delhi
```

Getting Data from Local Storage

getItem()

JS

```
let occupation = localStorage.getItem("occupation");
console.log(occupation);
```

Output

```
null
```

Values

null

We use **null** in situations where we intentionally want a variable to not have a value yet

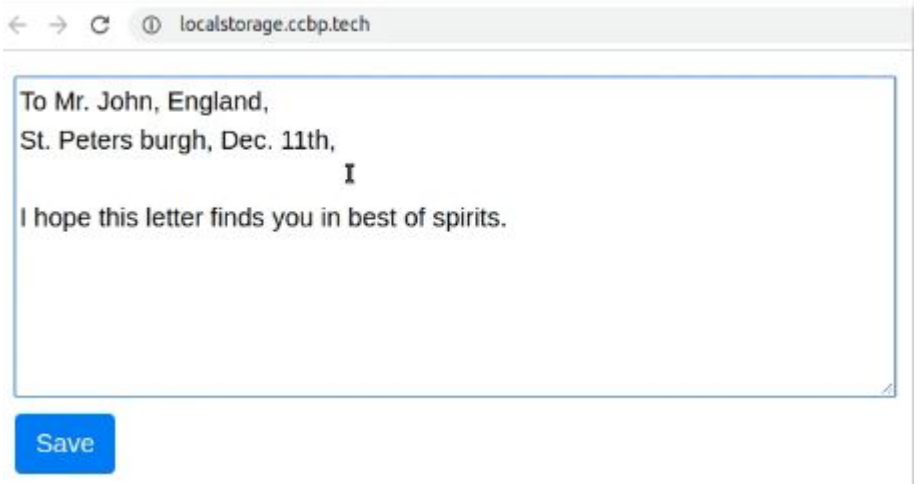
JS

```
let selectedColor = null;
console.log(selectedColor);
console.log(typeof(selectedColor));
```

Output

```
null
object
```

Local storage example:



How can we provide

Multiline Text as input?

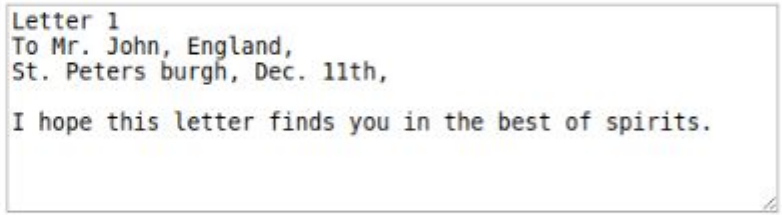
How to provide multiple line as a input?
 Below is the way

HTML Elements

The textarea Element

HTML

```
<textarea rows="8" cols="55">
</textarea>
```



- The rows attribute specifies the number of lines
- The cols attribute specifies the number of characters per line

The textarea Element

HTML

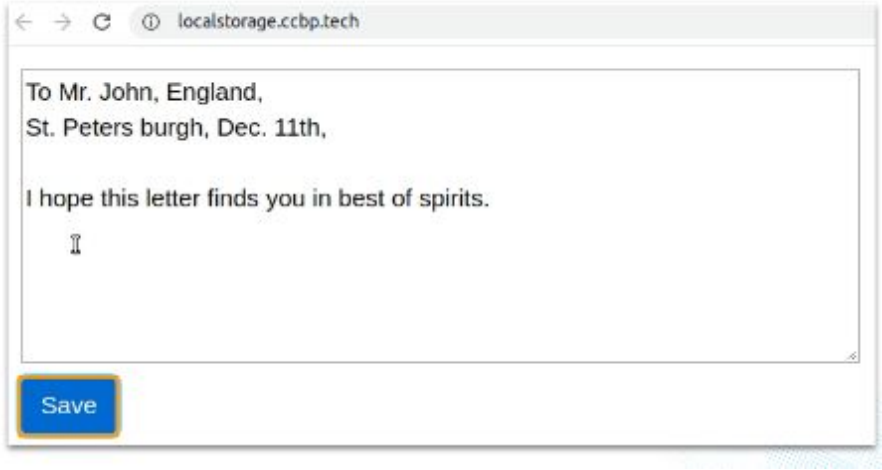
```
<textarea rows="1" cols="5"></textarea>
```

Hello

HTML

```
<textarea rows="3" cols="2"></textarea>
```

He
ll
o



How to store data
In Local Storage
on Button Click?

Storing Data in Local Storage

Adding Button with Event Listener

HTML

```
<textarea rows="8" cols="55"></textarea>
<br />
<button class="btn btn-primary mt-1" id="saveButton">Save</button>
```

JS

```
let saveButton = document.getElementById("saveButton");
saveButton.onclick = function() {


};
```

NXT WAV

Storing Data in Local Storage

Accessing textarea Element value

HTML

```
<textarea rows="8" cols="55" id="message"></textarea>
```

JS

```
let saveButton = document.getElementById("saveButton");
let textAreaElement = document.getElementById("message");
saveButton.onclick = function() {
  let userEnteredText = textAreaElement.value;
};
```

Storing Data in Local Storage

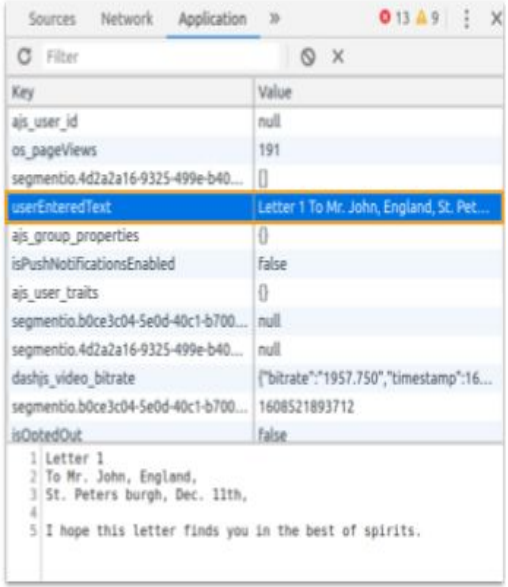
Storing value in Local Storage

JS

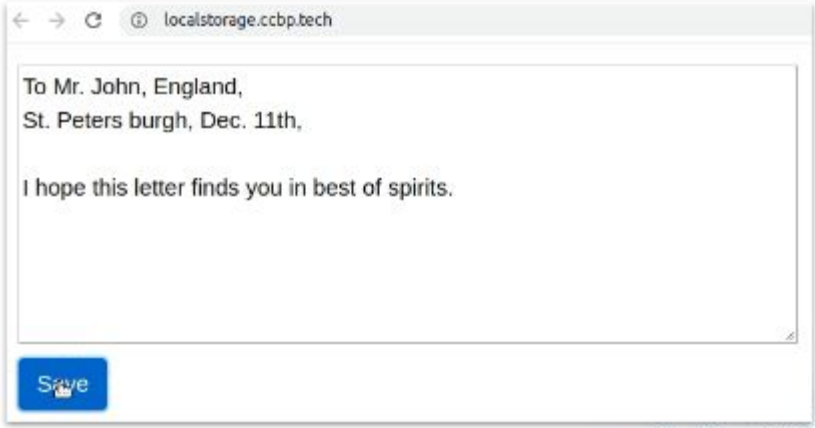
```
...
saveButton.onclick = function() {
  let userEnteredText = textAreaElement.value;
  localStorage.setItem("userEnteredText", userEnteredText);
};
...
```

Storing Data in Local Storage

Output



How to load Text Message automatically on Reload?



Getting Data from Local Storage

Automatically on Reload

JS

```
let storedUserInputValue = localStorage.getItem("userEnteredText");
if (storedUserInputValue === null) {
  textAreaElement.value = "";
}
else {
  textAreaElement.value = storedUserInputValue;
}
```

Key Takeaways

- Local Storage
  - setItem()
  - getItem()
- Values
  - null
- HTML Elements
  - textArea Element

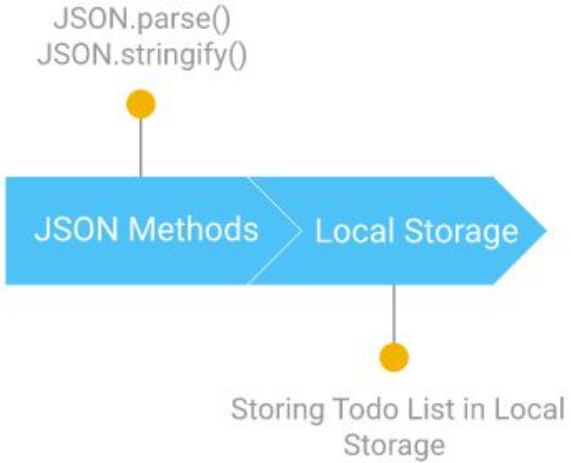
lecture4:todo

Recap

Values

Local Storage can only use **strings** for its keys and value

Trying to store any other type of data (objects) can lead to **unexpected behaviour**



Step:1 question

How can I Store

Other Types of Values

(Objects, Arrays, etc.)

in Local Storage?

step2:anwer

JSON Strings

JSON

Supported Types

Storing Data in Local Storage

JavaScript Object Notation (JSON)

JSON is a data representation format used for:

- Storing Data (Client/Server)
- Exchanging Data between Client and Server

- Number
- String
- Boolean
- Array
- Objects
- Null

JSON

Supported Types

JavaScript	JSON
10	10
"hello"	"hello"
true	true
[1, 2, 3]	[1, 2, 3]
null	null

JS Object vs JSON Object

JS Object

```
{
  name: "Rahul",
  age: 29,
  designation: "Web Developer"
}
```

JSON Object

```
{
  "name": "Rahul",
  "age": 29,
  "designation": "Web Developer"
}
```

JavaScript

JSON Methods

JavaScript provides **JSON methods** to convert Data into **JSON format.**

- JSON.stringify()
- JSON.parse()

JSON Methods

JSON.stringify()

It converts the given value into **JSON String**

JSON.stringify(value)

Value to be converted

JSON Methods

JSON.parse()

It parses a **JSON String** and returns a **JS Object**

JSON.parse(string)

String in JSON format

Example

Storing JS Objects

```
let profile = {
  name: "Rahul",
  age: 29,
  designation: "Web Developer"
};
```

JSON Methods

stringify()

```
let profile = {
  name: "Rahul",
  age: 29,
  designation: "Web Developer"
};
```

```
JSON.stringify(profile);
```

```
'{"name":"Rahul","age":29,"designation":"Web Developer"}'
```

JSON Object Methods

stringify()

JS

```
let stringifiedProfile = JSON.stringify(profile);
console.log(stringifiedProfile);
console.log(typeof(stringifiedProfile));
```

Output

```
{"name":"Rahul","age":29,"designation":"Web Developer"}
string
```

JSON Object Methods

parse()

```
let stringifiedProfile = '{"name":"Rahul","age":29,"designation":"Web Developer"}'
```

```
JSON.parse(stringifiedProfile);
```

```
{
  name: "Rahul",
  age: 29,
  designation: "Web Developer"
}
```

JSON Object Methods

parse()

JS

```
let parsedProfile = JSON.parse(stringifiedProfile);
console.log(parsedProfile);
console.log(typeof(parsedProfile));
```

Output

```
Object {name: "Rahul", age: 29, designation: "Web Developer"}
object
```

NXT WAVE

Local Storage

Storing and Getting Data

JS

```
localStorage.setItem("profileDetails", JSON.stringify(profile));
let stringifiedProfileDetails = localStorage.getItem("profileDetails");
let parsedProfileDetails = JSON.parse(stringifiedProfileDetails);
console.log(parsedProfileDetails);
```

Output

```
Object {name: "Rahul", age: 29, designation: "Web Developer"}
```

Storing Todo List in Local Storage

Todo List

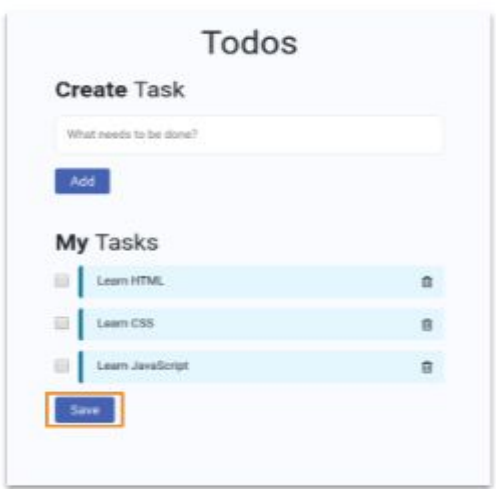
JS

```
let todoList = [
 {
   text: "Learn HTML",
   uniqueNo: 1
 },
 {
   text: "Learn CSS",
   uniqueNo: 2
 },
  ...
];
```

Adding Save Button Statically

HTML

```
<div class="todos-bg-container">
 <div class="container">
   <div class="row">
       ...
       <ul class="todo-items-container" id="todoItemsContainer"></ul>
       <button class="button" id="saveTodoButton">Save</button>
       ...
   </div>
 </div>
</div>
```



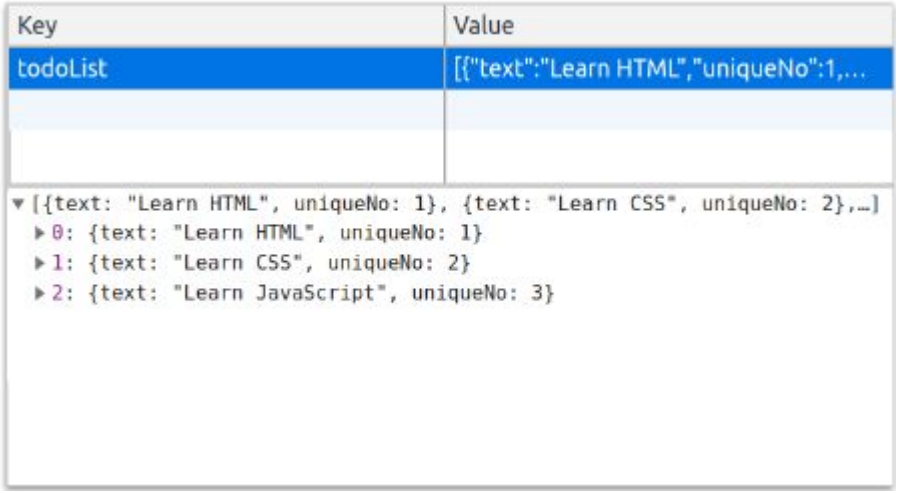
Adding Event Listener Dynamically

JS

```
...
let todoItemsContainer = document.getElementById("todoItemsContainer");
let addTodoButton = document.getElementById("addTodoButton");
let saveTodoButton = document.getElementById("saveTodoButton");
...
saveTodoButton.onclick = function () {
};
...
```

Storing TodoList

```
...
saveButton.onclick = function () {
  localStorage.setItem("todoList", JSON.stringify(todoList));
};
...
```



How to get Todo List from Local Storage?

Getting Todo List from Local Storage

getItem()

JS

```
...
function getTodoListFromLocalStorage() {
  let stringifiedTodoList = localStorage.getItem("todoList");
}
...
```

Getting Todo List from Local Storage

Parsing Stringified TodoList

JS

```
...
function getTodoListFromLocalStorage() {
  let stringifiedTodoList = localStorage.getItem("todoList");
  let parsedTodoList = JSON.parse(stringifiedTodoList);
}
...
```

Getting Todo List from Local Storage

Parsed Todo List

JS

```
...
function getTodoListFromLocalStorage() {
  ...
  if (parsedTodoList === null) {
    return [];
  }
  else {
    return parsedTodoList;
  }
}
...
```

Getting Todo List from Local Storage

Assigning value to Todo List

JS

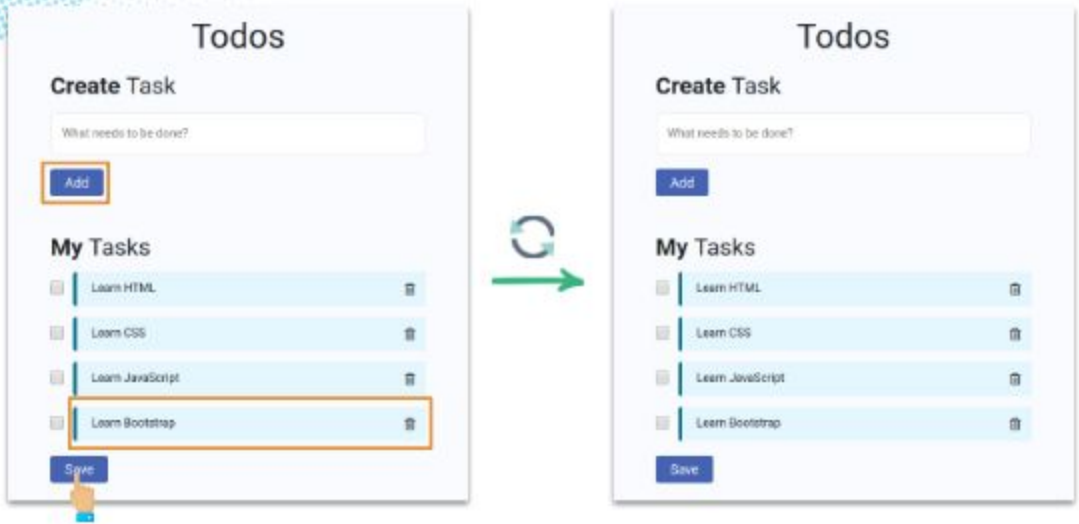
```
...
function getTodoListFromLocalStorage() {
  ...
  if (parsedTodoList === null) {
    return [];
  }
  else {
    return parsedTodoList;
  }
}
...
let todoList = getTodoListFromLocalStorage();
```

Delete existing Todo List

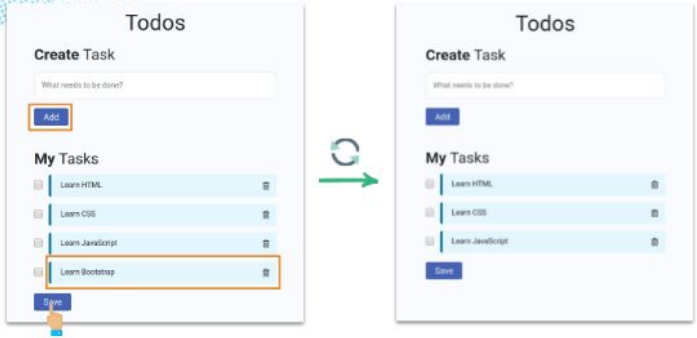
JS

```
let todoList = [
 {
   text: "Learn HTML",
   uniqueNo: 1
 },
 {
   text: "Learn CSS",
   uniqueNo: 2
 },
 ...
];
```

Adding a New Todo Item

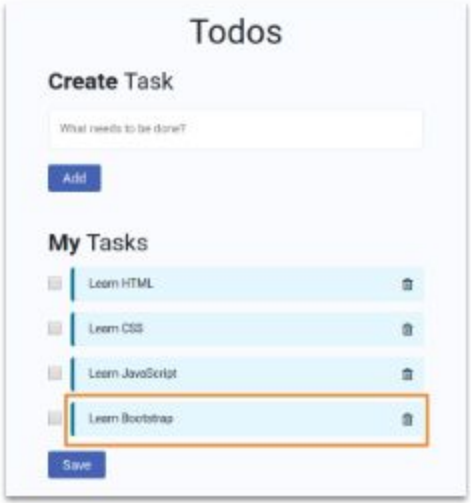


Why is the new Todo Item not persisting on Reload?



On Reload

Local Storage



Key	Value
todoList	[{"text":"Learn HTML","uniqueNo":1,...

```
▼ [{text: "Learn HTML", uniqueNo: 1}, {text: "Learn CSS", uniqueNo: 2},…]
  ▶ 0: {text: "Learn HTML", uniqueNo: 1}
  ▶ 1: {text: "Learn CSS", uniqueNo: 2}
  ▶ 2: {text: "Learn JavaScript", uniqueNo: 3}
```

Persisting New Todo Item on reloading

Adding New Item to the TodoList

JS

```
let todoList = getTodoListFromLocalStorage();
...
function onAddTodo() {
...
 let newTodo = {
   text: userInputValue,
   uniqueNo: todosCount,
 };
 console.log(todoList);
 ...
}
...
```

```
[Object, Object, Object]
1.  ▸0: Object
2.  ▸1: Object
3.  ▸2: Object
```

Persisting New Todo Item on reloading

Adding New Item to the TodoList

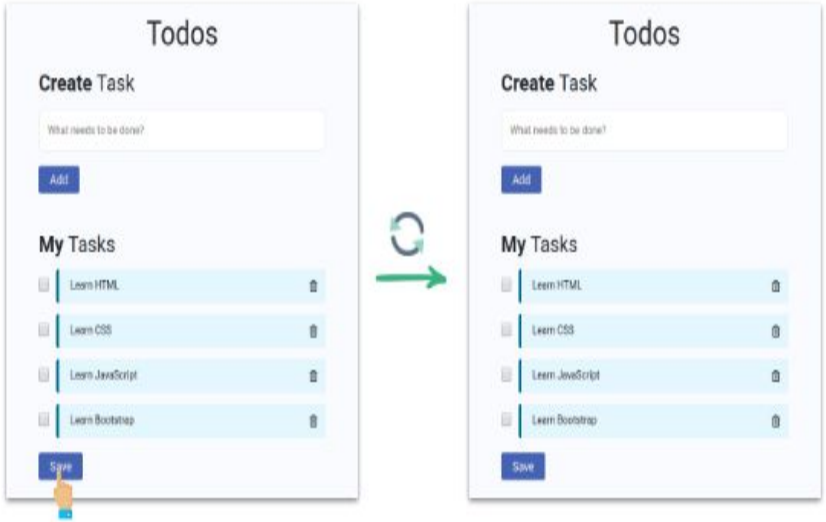
JS

```
let todoList = getTodoListFromLocalStorage();
...
function onAddTodo() {
...
 let newTodo = {
   text: userInputValue,
   uniqueNo: todosCount,
 };
 todoList.push(newTodo);
 console.log(todoList);
 ...
}
...
```

```
[Object, Object, Object, Object]
1.  ▸0: Object
2.  ▸1: Object
3.  ▸2: Object
4.  ▸3: Object
```

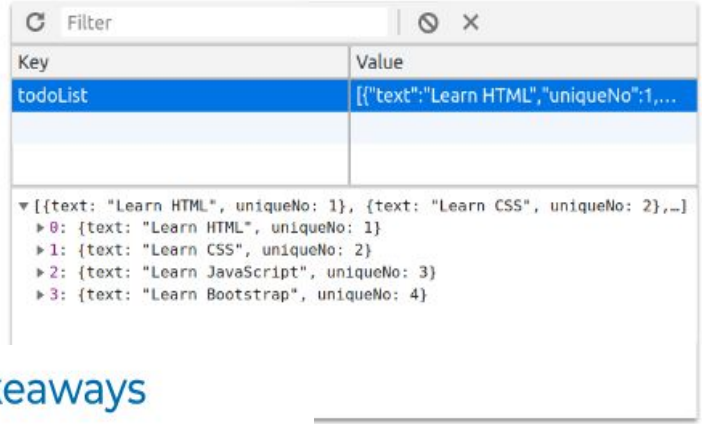
Persisting New Todo Item on reloading

Output



Local Storage

TodoList



Key Takeaways

- JSON Methods
  - JSON.stringify()
  - JSON.parse()
- Local Storage
  - Storing Todo List in Local Storage

Agenda

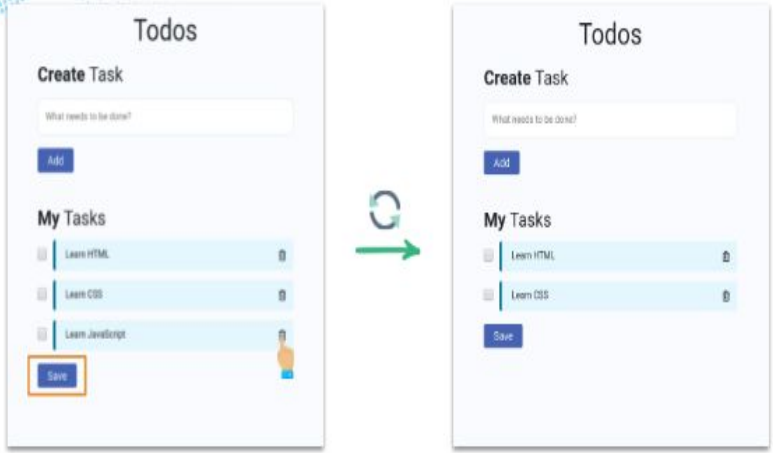
Todos Application

findIndex()
splice()

Array Methods → Local Storage

Deleting a Todo and updating Local Storage

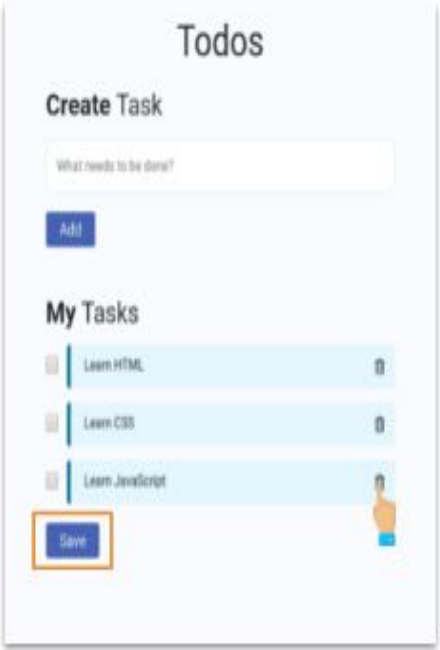
Deleting a Todo Item

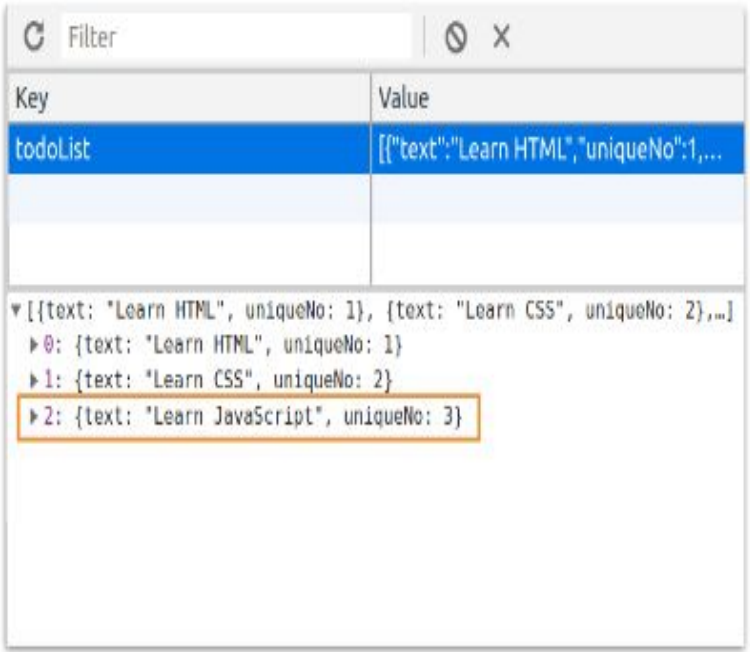


Why did the Deleted Todo item appear again on Reload?

On Reload

Local Storage



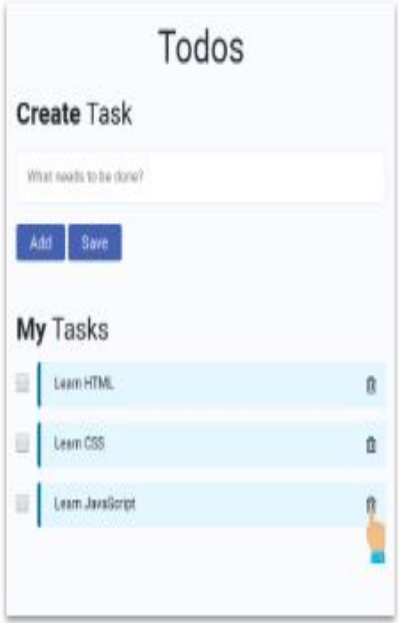


Removing a Deleted Todo from Local Storage

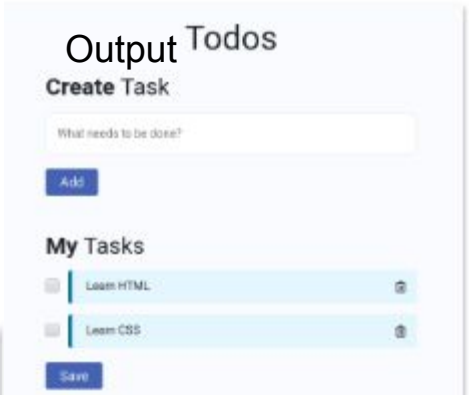
Deleting a Todo Item

JS

```
let todoList = getFromLocalStorage();

...

function onDeleteTodo(todoId) {
  let todoElement = document.getElementById(todoId);
  todoItemsContainer.removeChild(todoElement);
  console.log(todoList);
}

...
```



Output



But data in local storage is not deleting because we are removing from html element not for todolist so lets delete from todolist

```
[Object, Object, Object]
1.  ▶0: Object
2.  ▶1: Object
3.  ▶2: Object
```

How to remove corresponding Todo Object from Todo List?

Ans: The splice() method

Array Methods

Splice()

The splice() method changes the contents of an array.

Using splice() method we can:

- Remove existing Items
- Replace existing Items
- Add new Items

Array Methods

Removing Items - Splice()

Index

arr.splice(Start , Delete Count)

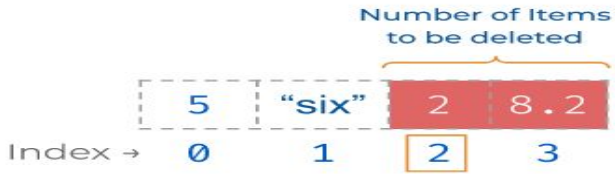
Number of Items to be removed, starting from the given Index

Splice()

Removing existing Items

JS

```
let myArray = [5, "six", 2, 8.2];
myArray.splice(2, 2);
console.log(myArray);
```



JS

```
let myArray = [5, "six", 2, 8.2];
myArray.splice(2, 2);
console.log(myArray);
```

Output

```
[5, "six"]
```

Splice()

Removing existing Items

The splice method returns an **Array** containing the deleted Items

JS

```
let myArray = [5, "six", 2, 8.2];
let deletedItems = myArray.splice(2, 2);
console.log(deletedItems);
```

Output

```
[2, 8.2]
```

JS

```
let myArray = [5, "six", 2, 8.2];
myArray.splice(2, 0);
console.log(myArray);
```

Output

```
[5, "six", 2, 8.2]
```

It did’nt delete any element because it specified zero element to be deleted.

Adding New Items - Splice()

Array Methods

Adding New Items

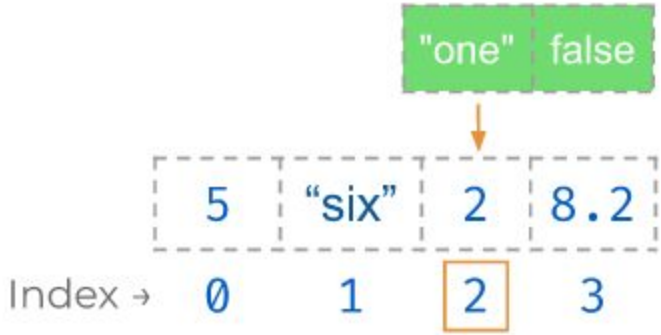
JS

```
let myArray = [5, "six", 2, 8.2];
myArray.splice(2, 0, "one", false);
console.log(myArray);
```

output

Output

```
[5, "six", "one", false, 2, 8.2]
```



Replacing Existing Items - Splice()

Index

Items to be added, starting from the given Index

```
arr.splice(Start , Delete Count, Item1, Item2 ... )
```

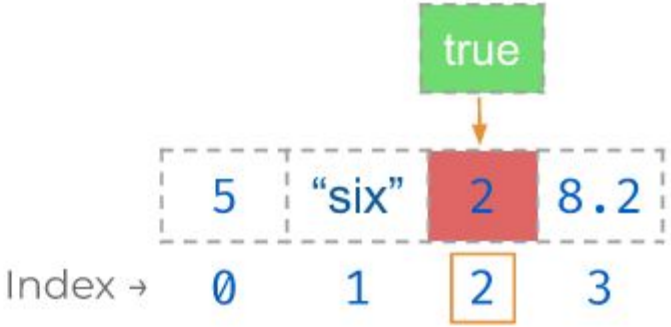
Number of Items to be removed, starting from the given Index

JS

```
let myArray = [5, "six", 2, 8.2];
myArray.splice(2, 1, true);
console.log(myArray);
```

Output

```
[5, "six", true, 8.2]
```



How to Find an index of an Item in Array?

solution:

The findIndex() method

Array Methods

findIndex()

This method returns the **index** of the **first Item** that satisfies the provided **testing function**

If no Item found returns **-1**

	5	12	8	130	44
Index →	0	1	2	**3**	4

findIndex()

```
arr.findIndex(Testing Function)
```

A function to execute on each value in the array

Array

Finding Item Index

```JS
let myArray = [5, 12, 8, 130, 44];

let itemIndex = myArray.findIndex( Testing Function );
```

Testing Function

```JS
let myArray = [5, 12, 8, 130, 44];

let itemIndex = myArray.findIndex(function(eachItem){
  console.log(eachItem);
});
```

Testing Function

Output

```
5
12
8
130
44
```

Finding Item Index

Return true/false

5	12	8	130	44

JS

```
let myArray = [5, 12, 8, 130, 44];
let itemIndex = myArray.findIndex(function(eachItem){
  if (eachItem === 130) {
    return true;
  }
  else {
    return false;
  }
});
console.log(itemIndex);
```

Output

3

Find Index of Item

130

```
function(eachItem){
  if (eachItem === 130) {
    return true;
  }
  else {
    return false;
  }
}
```

Finding Item Index

Example

Find the Customer with ID 102?

```JS
let customersData = [
  {
    name: "Rahul",
    id: 101
  },
  {
    name: "Praveen",
    id: 102
  },
  {
    name: "Vinod",
    id: 174
  }
];
```

```JS
let customersData = [
  {
    name: "Rahul",
    id: 101
  },
  ...
];
let itemIndex = customersData.findIndex(function(eachItem) {


});
console.log(itemIndex);
```

```JS
...
let itemIndex = customersData.findIndex(function(eachItem) {
  if (eachItem.id === 102){
      return true;
  }
  else {
      return false;
  }
});
console.log(itemIndex);
```

Output

1

Let's Remove Todo Object from Todo List?

JS

```
...
function onDeleteTodo(todoId){
  ...
  let deleteElementIndex = todoList.findIndex(function(eachTodo){
    let eachTodoId = "todo" + eachTodo.uniqueNo;
    if (eachTodoId === todoId) {
      return true;
    }
    else {
      return false;
    }
  });
}
```

1

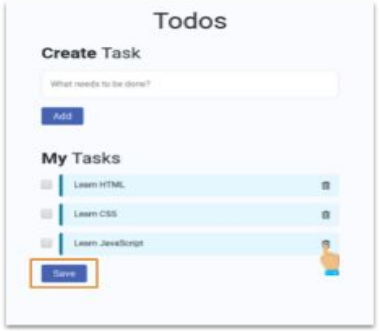
JS

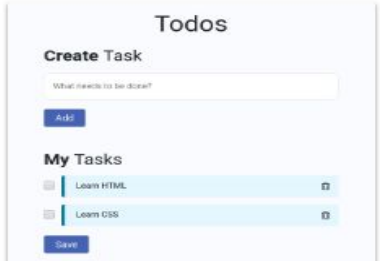
```
...
let todoList = getFromLocalStorage();
...
function onDeleteTodo(todoId){
  ...
  let deleteElementIndex = todoList.findIndex(function(eachTodo){
    let eachTodoId = "todo" + eachTodo.uniqueNo;
    ...
  });
  todoList.splice(deleteElementIndex, 1);
}
...
```

2

Removing a Deleted Todo from Local Storage

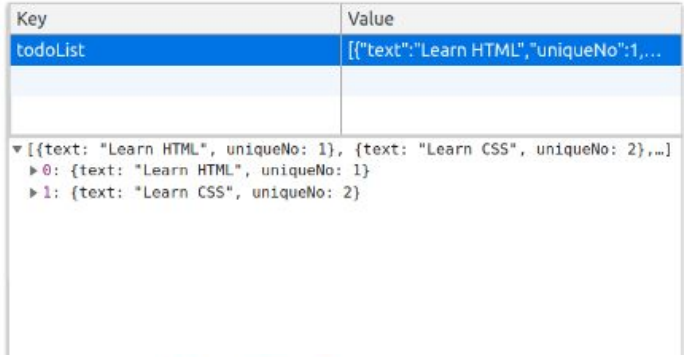
Output





Removing a Deleted Todo from Local Storage

Local Storage



Arrays

Some More Methods

- includes()
- indexOf()
- lastIndexOf()
- shift()
- unShift()
- join()
- sort()

Key Takeaways

- Array Methods
  - splice()
  - findIndex()
- Deleting a Todo and updating Local Storage

cheatsheet

1. Array Methods

Method	Functionality
includes, indexOf, lastIndexOf, find, findIndex()	Finding Elements
push, unshift, splice	Adding Elements
pop, shift, splice	Removing Elements
concat, slice	Combining & Slicing Arrays
join	Joining Array Elements
sort	Sorting Array Elements

1.3 includes()

The `includes()` method returns `true` if the provided item exists in the array. If no item is found, it returns `false` .

Syntax: `arr.includes(item)`

1.4 indexOf()

The `indexOf()` method returns the first index at which a given item can be found in the array. If no item is found, it returns `-1` .

Syntax: `arr.indexOf(item)`

1.5 lastIndexOf()

The `lastIndexOf()` method returns the last index at which a given item can be found in the array. If no item is found, it returns `-1` .

Syntax: `arr.lastIndexOf(item)`

1.6 find()

The `find()` method returns the first item's value that satisfies the provided testing function. If no item is found, it returns `undefined` .

Syntax: `arr.find(Testing Function)`

1.7 unshift()

The `unshift()` method adds one or more items to the beginning of an array and returns the new array length.

Syntax: `arr.unshift(item1,item2, ..., itemN)`

1.8 shift()

The `shift()` method removes the first item from an array and returns that removed item.

Syntax: `arr.shift()`

1.9 concat()

The `concat()` method can be used to merge two or more arrays.

This method does not change the existing arrays but instead returns a new array.

Syntax:

JAVASCRIPT

```
let newArray = arr1.concat(arr2);
```

1.10 slice()

The `slice()` method returns a portion between the specified start index and end index(end index not included) of an array into a new array.

Syntax: `arr.slice(startIndex, endIndex)`

1.11 join()

The `join()` method creates and returns a new string by concatenating all of the items in an array, separated by commas or a specified separator string.

If the array has only one item, then it will be returned without using the specified separator.

Syntax: `arr.join(separator)`

Here `separator` is a string used to separate each item of the array. If omitted, the array items are separated with a comma.

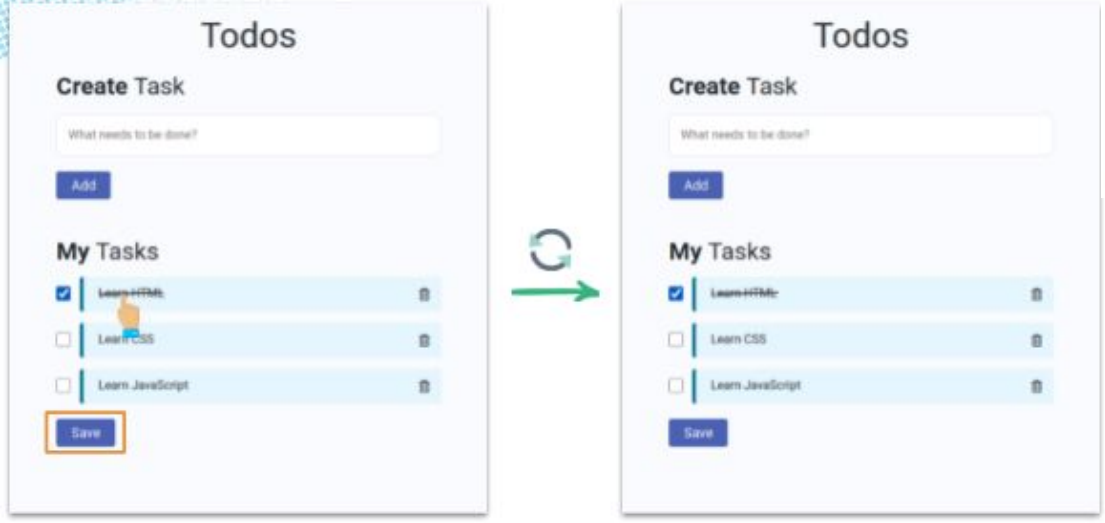
1.12 sort()

The `sort()` method sorts the items of an array and returns the sorted array. The default sort order is ascending.

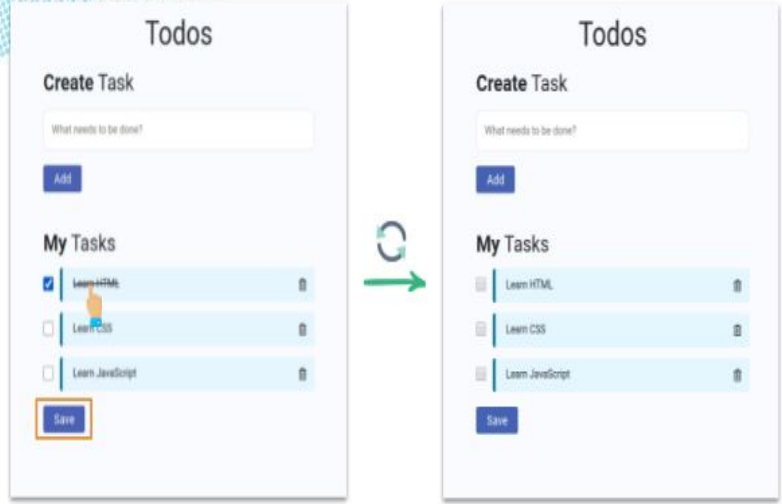
Syntax: `arr.sort()`

Local Storage > Recap

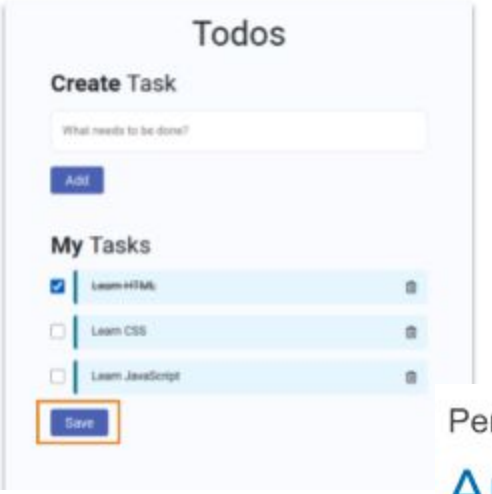
Persisting Todo Checked Status on Reload

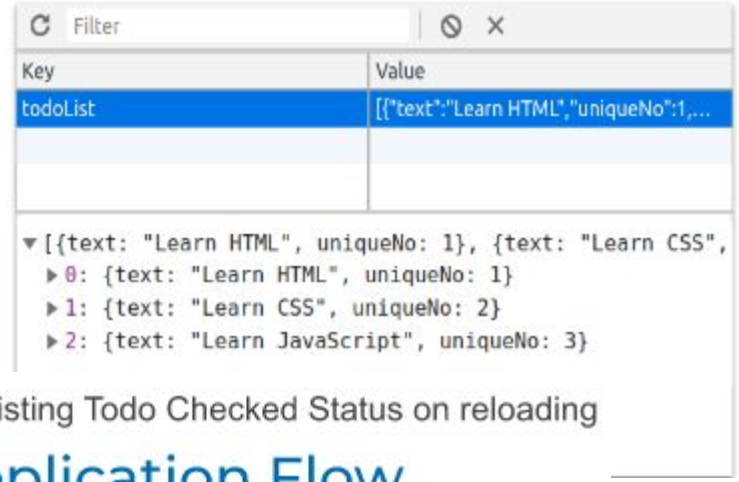


Why the Checked Status does not persist on Reload?



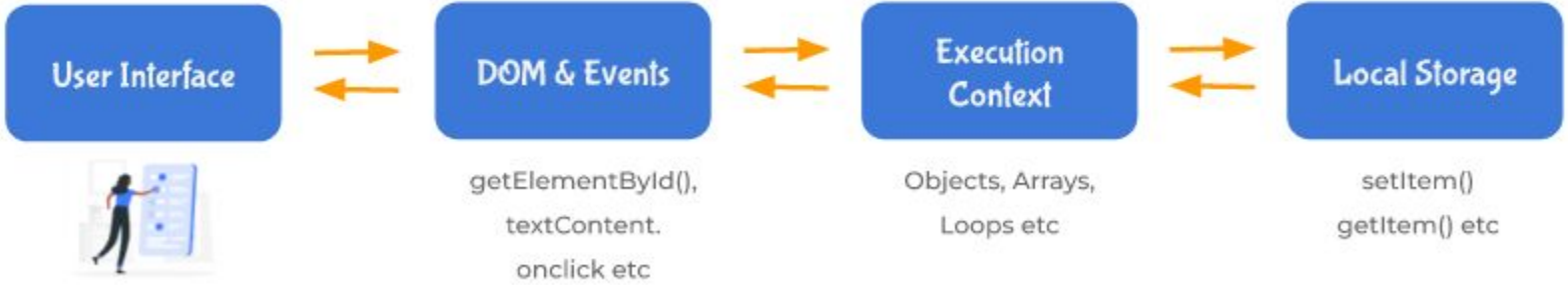
Local Storage





Persisting Todo Checked Status on reloading

Application Flow

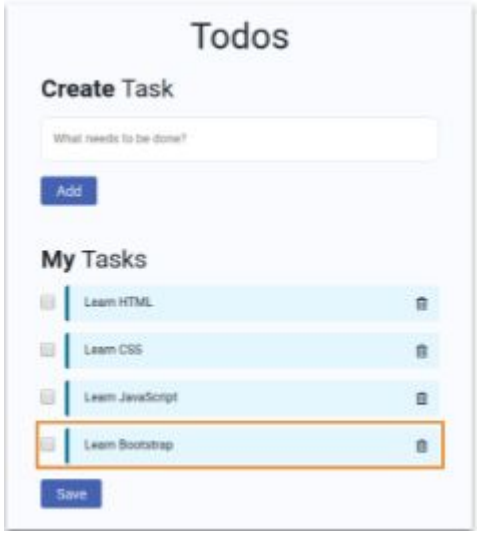


Programmatic Representation

Object

Todo Object should contain all the related information about the Todo Item

- Todo Text
- Unique number
- Checked or not



Persisting Todo Checked Status on reloading

Maintaining Checked Status

Maintain **checked status** in created Todo Items

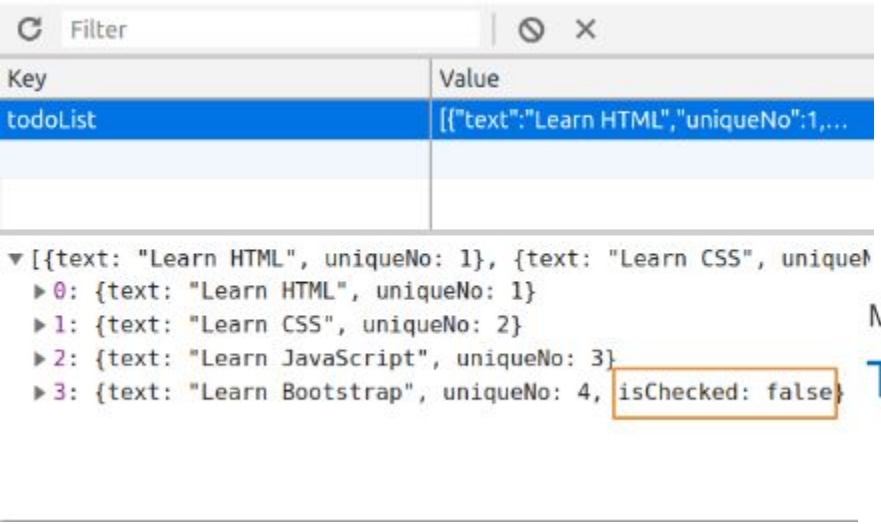
Adding isChecked key to Newly Added Item

JS

```
let todoList = getTodoListFromLocalStorage();
...
function onAddTodo() {
  ...
  let newTodo = {
    text: userInputValue,
    uniqueNo: todosCount,
    isChecked: false
  };
  ...
```

Maintaining Checked Status

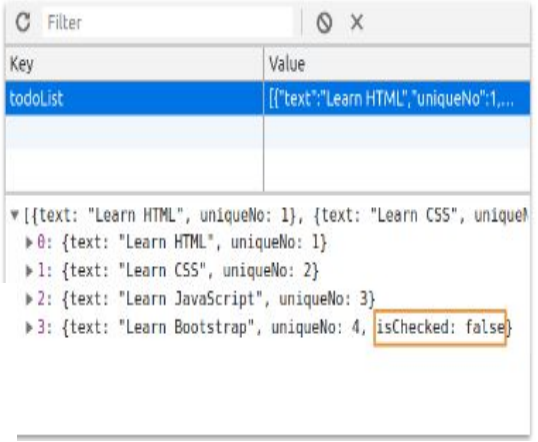
Local Storage



Checked Status in Todo Objects

Some Todo Objects have

Checked Status and some don't.



Maintaining Checked Status

Todo List

- **Remove** existing Todo List from Local Storage
- Create New Todo List

Using Local Storage Methods

Local Storage

To access and work with Local Storage, we have the below methods:

- setItem()
- getItem()
- removeItem()
- clear()

Local Storage

removeItem()

This method **removes** the specified Storage Object Item based on the **key**

```
localStorage.removeItem(key)
```

Name of the key to be removed

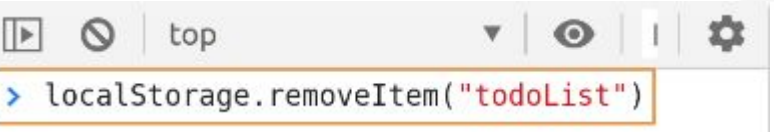
Removing Todo List from Local Storage

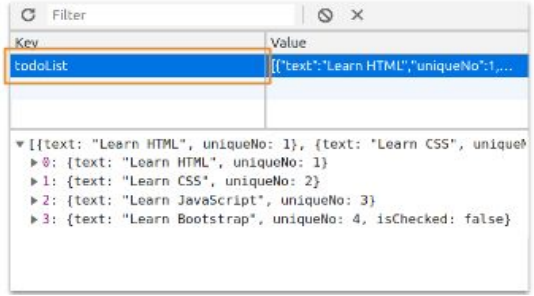
Removing Todo List from Local Storage

Ways to Remove

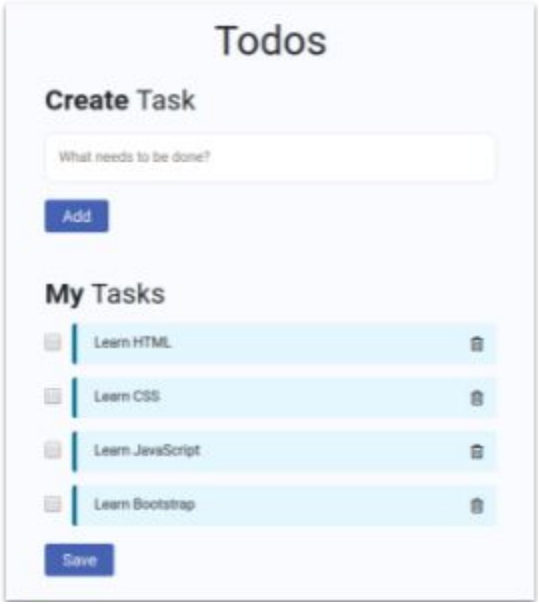
JS

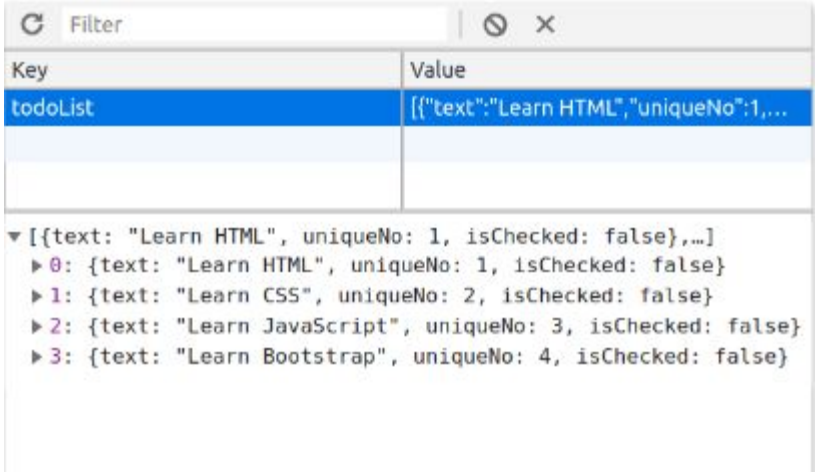
```
...
localStorage.removeItem("todoList");
...
```





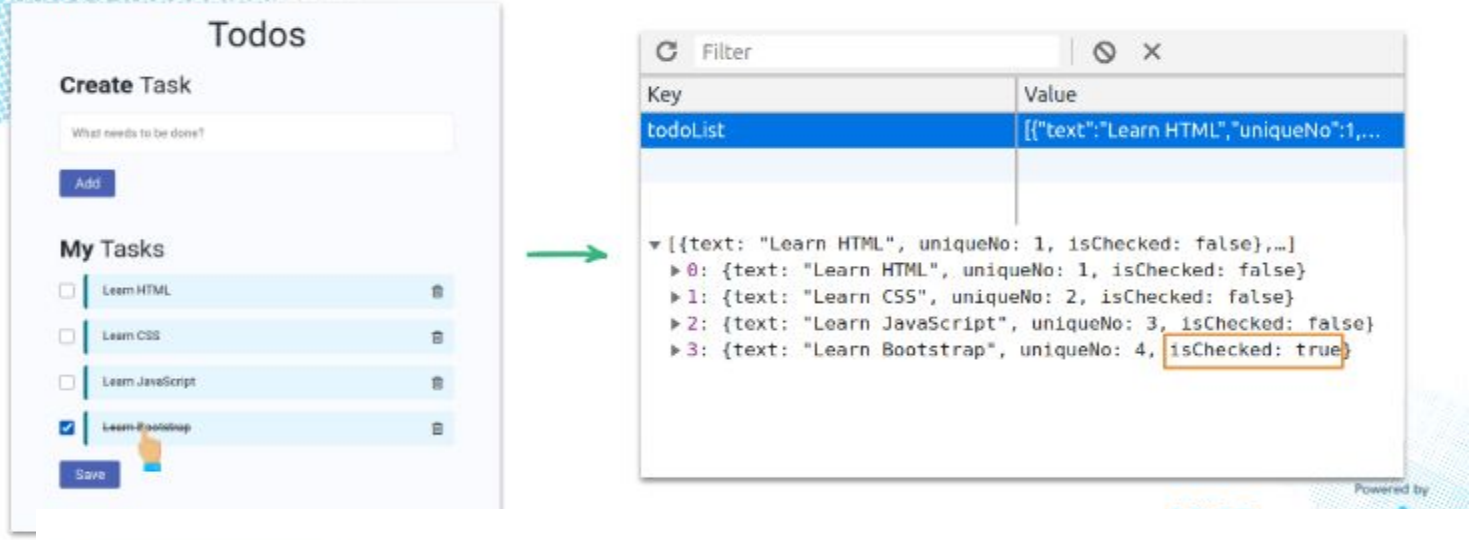
Adding New Todo Items

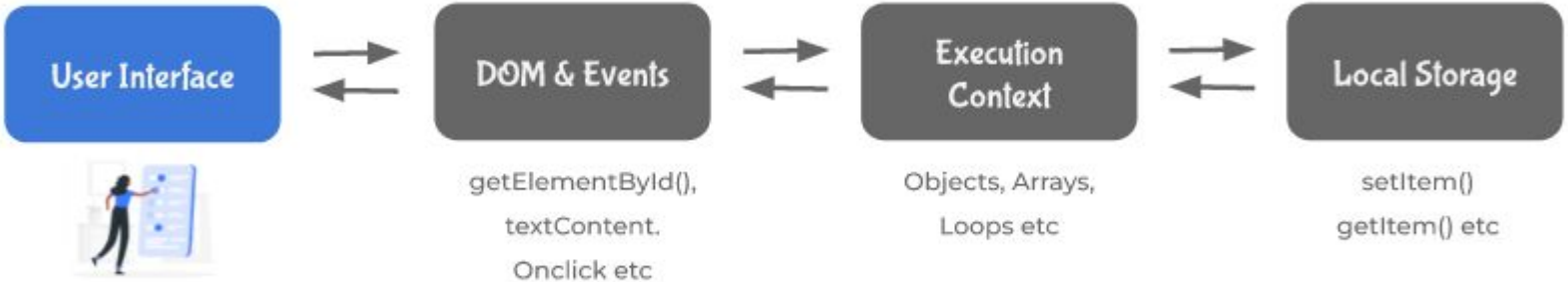




Above all task as added newly by removing old task so ischecked is recorded as false

How to update Todo Checked Status?





Updating Todo Object Checked Status

Finding the Todo Object

```
let todoList = getTodoListFromLocalStorage();
...
function onTodoStatusChange(checkboxId, labelId){
  ...
}
...
```

JS

```
...
function createAndAppendTodo(todo) {
  let todoId = "todo" + todo.uniqueNo;
  ...
  let inputElement = document.createElement("input");
  ...
  inputElement.onclick = function() {
    onTodoStatusChange(checkboxId, labelId, todoId);
  };
  ...
}
...
```

Comparing Todo IDs

JS

```
function onTodoStatusChange(checkboxId, labelId, todoId) {
  ...
  let todoObjectIndex = todoList.findIndex(function(eachTodo) {
    let eachTodoId = "todo" + eachTodo.uniqueNo;
    if (eachTodoId === todoId) {
      return true;
    }
    else {
      return false;
    }
  });
}
```

Accessing the Todo Object

JS

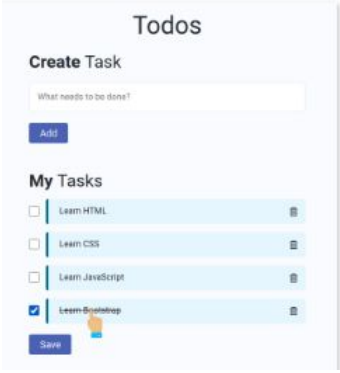
```
let todoList = getTodoListFromLocalStorage();
...
function onTodoStatusChange(checkboxId, labelId, todoId) {
  ...
  let todoObjectIndex = todoList.findIndex(function(eachTodo) {
    let eachTodoId = "todo" + eachTodo.uniqueNo;
    ...
  });
  let todoObject = todoList[todoObjectIndex];
}
...
```

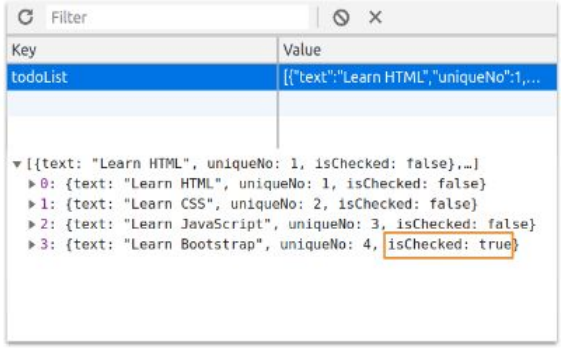
Changing Checked Status

JS

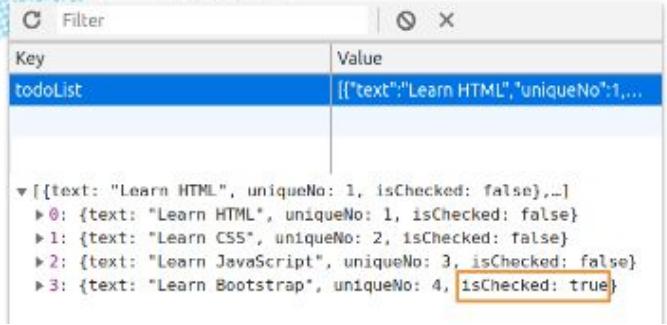
```
...
function onTodoStatusChange(checkboxId, labelId, todoId) {
  ...
  let todoObjectIndex = todoList.findIndex(function(eachTodo) {
    ...
  });
  let todoObject = todoList[todoObjectIndex];
  if(todoObject.isChecked === true){
    todoObject.isChecked = false;
  }
  else {
    todoObject.isChecked = true;
  }
}
```

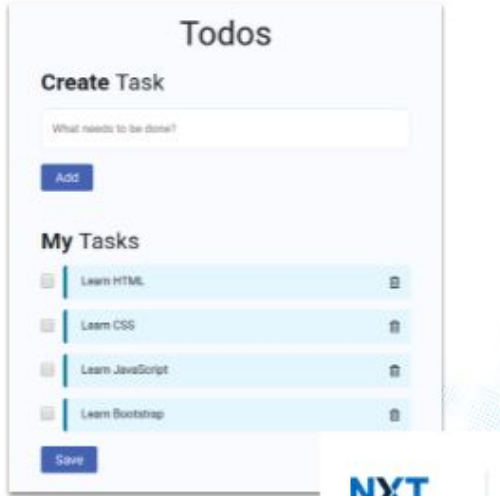
Output





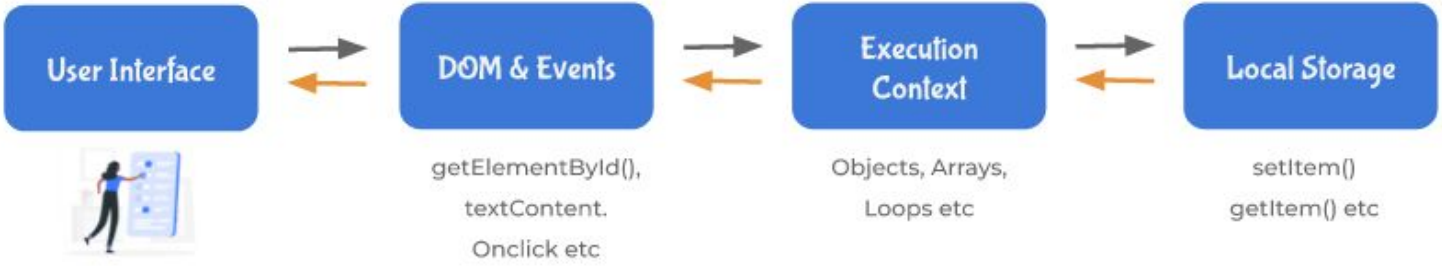
Why is the Todo Item not checked on Reload?





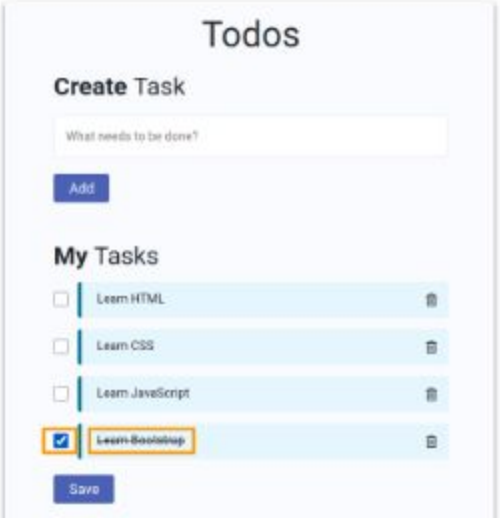
Application Flow

On Create & Append Todos



Reloading

Creating a New Todo Item



While creating a new Todo Item, and isChecked is true

- Checkbox should be in selected state
- Label text should be striked off

Creating a Todo Item

Adding Checked Status

JS

```
...
function createAndAppendTodo(todo){
  ...
  let inputElement = document.createElement("input");
  inputElement.type = "checkbox";
  inputElement.id = checkboxId;
  inputElement.checked = todo.isChecked;
  inputElement.onclick = function () {
  ...
}
...
```

Creating a Todo Item

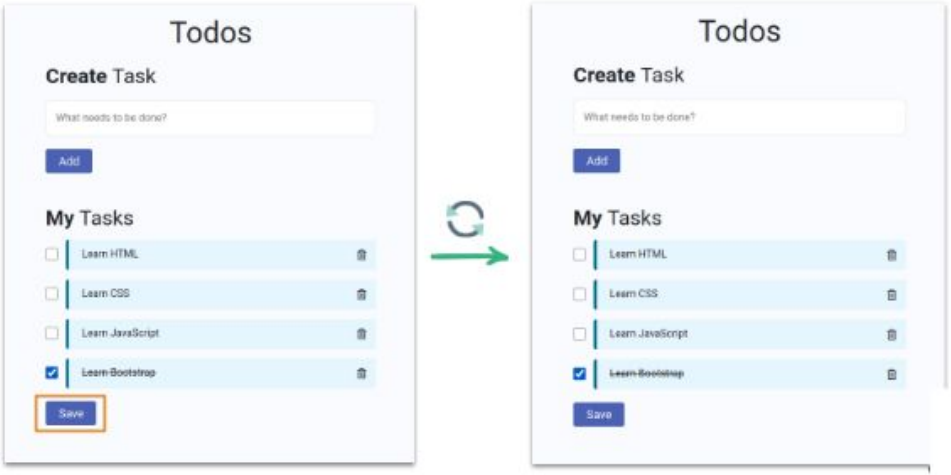
Striking Label Element Text

JS

```
...
function createAndAppendTodo(todo){
  ...
  let labelElement = document.createElement("label");
  ...
  if (todo.isChecked === true) {
    labelElement.classList.add("checked");
  }
  labelContainer.appendChild(labelElement);
  ...
  console.log(todoList);
}
...
```

Persisting Todo Checked Status on Reload

Output



What Have We Done So Far?

Todos Application

Features

- Creating **Single** Todo Item
- Creating **Multiple** Todo Items
- Taking User Input and creating Todos **Dynamically**
- **Checking** a Todo
- **Deleting** a Todo
- **Persisting** Todos On Reload using Local Storage

Loops

Loops allow us to **execute**
a block of code **several times**

- for...of Loop

JSON Methods

JSON.stringify(value)

JSON.parse(string)

Todos Application

Local Storage

- setItem()
- getItem()
- removeItem()