
01. Local Storage – (Frontend JS)

What is Local Storage?

- A **browser-based storage** system that lets you **store key-value pairs**.
 - Data is saved even after the page reloads or browser is closed.
-

Key Features:

Feature	Details
Size Limit	~5MB per origin (varies by browser)
Storage Type	Key-Value (both are strings only)
Persistence	Until explicitly cleared by user/code
Scope	Domain-specific (not shared between sites)

Common Methods:

Method	Use
<code>localStorage.setItem(k, v)</code>	Save data
<code>localStorage.getItem(k)</code>	Get data
<code>localStorage.removeItem(k)</code>	Delete a specific key
<code>localStorage.clear()</code>	Remove all data
<code>localStorage.length</code>	Total items stored
<code>localStorage.key(index)</code>	Get key by index

Example:

```
localStorage.setItem("name", "Rahul");      // Save  
  
let name = localStorage.getItem("name");      // Get  
  
localStorage.removeItem("name");              // Delete  
  
localStorage.clear();                        // Clear all
```

Project 01 : Use e.target.value to get live update from input just update local store using localStorage

1.easy way using assignment operator

Html

```
<input type="text" name="name" id="" class="input-field">
<p class="output"></p>
```

JS

```
const input = document.querySelector(".input-field");
const output = document.querySelector(".output");

input.addEventListener('input', (e) => {
  e.preventDefault();

  let userInp = e.target.value
  localStorage.myName=userInp           //set localStorage value
  output.innerText=localStorage.myName   //get localStorage value

});
```

LocalStorage

123456789

123456789

localStorage		
Origin	localStorage	
Key	Value	
http://127.0.0.1:5500	myName	123456789

2.using get and set method of localStorage

```
input.addEventListener('input', (e)=>{
  localStorage.setItem('myName',e.target.value)
  output.innerText=localStorage.getItem('myName')

})
```

[Project link](#) ↗

Project 02 : Use of

```
// localStorage.setItem("name", "Rahul");           // Save  
// let name = localStorage.getItem("name");         // Get  
// localStorage.removeItem("name");                // Delete  
// localStorage.clear();                          // Clear all
```

[Project link](#) 

02: How to store Object in Local Storage

```
HTML:  
<input type="text" name="name" id="" class="input" placeholder="enter myName">  
  <p>my name is : <span class="outputName"></span></p>  
  <p>my age is : <span class="outputAge"></span></p>
```

```
js:  
  
let obj={  
  
  myName:"",  
  myAge: ""  
  
}  
  
input.addEventListener('input', (e)=>{  
  
  obj.myName=e.target.value // पहिलं object cha val update केले  
  localStorage.setItem('myData', obj)  
  
  // आणि नंतर पूर्ण obj पास केला localStorage मध्ये  
  
})  
}
```

आता आपल्याला दोन वेगळे input- name and age घेण्या याएवजी आपण एक object create करून आपण multiple key as input देऊ शकतो .

but problem हा येतो की local storage हे फक्त string घेत so we have to change object to string format

Origin	http://127.0.0.1:5500
Key	Value
myData	[object Object]

- **Problem 01: How to change object to string and reverse using JSON**

```
> JSON.stringify(myData)
< '{"name":"Anrau","age":""}'
> JSON.parse('{"name":"Anrau","age":""}')
< {name: 'Anrau', age: ''}
```

```
let obj={

    myName:"",
    myAge: ""

}

input.addEventListener('input', (e)=>{

    obj.myName=e.target.value
    localStorage.setItem('objData',JSON.stringify(obj))
//लक्ष्य द्या इथे objectdata या localstorage मध्ये आपल्याला कंप्लीट object पास काराईचा
आहे not obj.nameor obj.age
}
})
```

Origin	http://127.0.0.1:5500
Key	Value
objData	{"myName":"lambodar", "myAge":""}
	▼ {myName: "lambodar", myAge: ""} myAge: "" myName: "lambodar"

- **Problem 02:**

- What was the problem?
- After refreshing the page:
- You typed a new age.
- But the name got erased from localStorage.
- Why? Because your code overwrote the entire object in localStorage with the new empty object obj.

```
const obj = JSON.parse(localStorage.getItem('objData')) || {  
    myName: '',  
    myAge: ''  
}
```

हे नक्की का वापरायचं लागतं ते समजून घेऊया...

⌚ तुझी चूक काय होती?

तू सुरुवातीला हे लिहिलं होतंस:

```
let obj = {  
    myName: "",  
    myAge: ""  
};
```

याचा अर्थ:  "प्रत्येकवेळी पेज रीफ्रेश झाल्यावर नवीन रिकामं **object** तयार कर."

म्हणजे, जरी **localStorage** मध्ये आधी माहिती असली तरीही, तू ती वापरत नव्हतास — तू नवीन **obj** बनवत होतास.

त्यामुळे तू फक्त **age** अपडेट केलंस, आणि **name** चं रिकामं **value** पुन्हा **localStorage** मध्ये सेव्ह झालं. म्हणून **name** मजून जात होतं!

✓ मग योग्य मार्ग काय?

```
let obj = JSON.parse(localStorage.getItem('objData')) || {  
    myName: "",  
    myAge: ""  
};
```

याचा अर्थ:

"जर **localStorage** मध्ये आधीची माहिती असेल, तर ती वापर. आणि नसल्यास नवीन रिकामं **object** तयार कर."

हे दोन भाग आहेत:

1. `localStorage.getItem('objData')` → `localStorage` मधून जुनी माहिती घे.
2. `JSON.parse(...)` → ती `string` असते, म्हणून ती परत `object` मध्ये convert कर.
3. `|| { myName: "", myAge: "" }` → जर जुनी माहिती मिळाली नाही (`null` आली), तर नवीन `object` वापर.

💡 सोपं उदाहरण:

समज तुझ्या `localStorage` मध्ये हे आधीपासून आहे:

```
{ myName: "Rahul", myAge: "25" }
```

आता page refresh केल्यावर:

- जर तू नवीन `object` बनवला (`let obj = { myName:"", myAge:"" }`), तर जुना `Rahul` replace होईल.
- पण जर तू `localStorage` मधून जुनी माहिती घेतलीस (`JSON.parse(...)`), तर `Rahul` टिकून राहील!

Here is entire code ↗

