

TP-03

# JavaScript

(Local storage, Cookie, and session)

<https://javascript.info/>

# TP03 Exercises

## TP02.1: Todo list with LocalStorage

Build a web application allowing user to create do-to list based on following UI designs and functionalities. The data must be stored inside LocalStorage

### To-do List with LocalStorage

Task title

Assingnee

Dateline

Title

John

07/22/2018

Add

Create project	Tola	24/2/2022	
Define project req	Makara	24/3/2022	
Testing on project	Kompheak	24/4/2022	
Deploy project	Minea	24/5/2022	
Hosting	Mesa	31/3/2022	

# TP03 Exercises

## TP02.2: Todo list with SessionStorage

Build a web application allowing user to create do-to list based on following UI designs and functionalities. The data must be stored inside SessionStorage

### To-do List with SessionStorage

Task title

Assingnee






Dateline

Title

John

07/22/2018

Add

Create project	Tola	24/2/2022	
Define project req	Makara	24/3/2022	
Testing on project	Kompheak	24/4/2022	
Deploy project	Minea	24/5/2022	
Hosting	Mesa	31/3/2022	

# TP03 Exercises

## TP02.3: Cookie

Build a web application allowing user to create cookie in a list based on following UI designs and functionalities. The data must be stored inside Cookie

### Cookie Management

Key	Value	Expired	
<input type="text" value="Key"/>	<input type="text" value="value"/>	<input type="text" value="07/22/2018"/>	<input type="button" value="Add"/>
username	tola		<input type="button" value="Set expired"/>
token	j42hk3234khj432khj		<input type="button" value="Set expired"/>
email	test@gic.com		<input type="button" value="Set expired"/>

**Note:** Local cookie is not allowed in Google Chrome, please do your test on Firefox instead

# TP03 Exercises

## TP03.4: Book shop CRUD with Local storage

Build a web application allowing user to create/read/update/delete books. All data must be stored in Local Storage

The screenshot displays a web application interface for a book shop. On the left side, there is a form for adding a new book with the following fields:

- Name:** A text input field containing "c programming".
- Category:** A text input field containing "coding".
- Price:** A text input field containing "1,000 riel".
- Add:** A button to submit the form.

On the right side, there is a grid view of the book collection. The grid contains six book cards, each with a book icon, a title, a price, and a category. Each card also has "Delete" and "Change name" buttons at the top.

Book ID	Name	Price	Category
Book I1	Name: Book I1	Price: 100 riel	Category: English
Book I2	Name: Book I2	Price: 100 riel	Category: English
Book I3	Name: Book I3	Price: 100 riel	Category: English
Book I4	Name: Book I4	Price: 100 riel	Category: English
Book I5	Name: Book I5	Price: 100 riel	Category: English
Book I6	Name: Book I6	Price: 100 riel	Category: English


- ✓ Grid view of book collection
- ✓ Add new book
- ✓ Delete book
- ✓ Update new name

# TP03 Exercises

## TP03.5: Login Form with Cookie

Build a web application that always requires user authentication before getting to the home page.

Given login info:  
Email: gic@gmail.com  
Pwd: Gic123



**Email**

**Password**

Login

### Welcome to Home Page

You will be automatically logged out

Keep refreshing your page

Logout

**Note:** Local cookie is not allowed in Google Chrome, please do your test on Firefox instead

# Local storage, Cookie and Session storage

	Cookies	Local Storage	Session Storage
Capacity	4kb	10mb	5mb
Browsers	HTML4 / HTML 5	HTML 5	HTML 5
Accessible from	Any window	Any window	Same tab
Expires	Manually set	Never	On tab close
Storage Location	Browser and server	Browser only	Browser only
Sent with requests	Yes	No	No

# Local storage & Session storage

## LocalStorage

```
localStorage.setItem('test', 1);

alert( localStorage.getItem('test') ); // 1
```

## Object-like access

```
// set key
localStorage.test = 2;

// get key
alert( localStorage.test ); // 2

// remove key
delete localStorage.test;
```

## Strings only

```
localStorage.user = {name: "John"};
alert(localStorage.user); // [object Object]
```

```
localStorage.user = JSON.stringify({name: "John"});

// sometime later
let user = JSON.parse( localStorage.user );
alert( user.name ); // John
```

## Session Storage

```
sessionStorage.setItem('test', 1);

alert( sessionStorage.getItem('test') ); // after refresh: 1
```



# Cookie

## Reading from document.cookie

```
alert( document.cookie ); // cookie1=value1; cookie2=value2;...
```

## Writing to document.cookie

```
document.cookie = "user=John"; // update only cookie named 'user'  
alert(document.cookie); // show all cookies
```

```
// special characters (spaces), need encoding  
let name = "my name";  
let value = "John Smith"  
  
// encodes the cookie as my%20name=John%20Smith  
document.cookie = encodeURIComponent(name) + '=' + encodeURIComponent(value);  
  
alert(document.cookie); // ...; my%20name=John%20Smith
```

## Cookie options: path=/; domain=site.com; expires; secure; samesite

```
document.cookie = "user=John; path=/; expires=Tue, 19 Jan 2038 03:14:07 GMT"
```

**Note:** Check w3school for supporting functions

# Bonus



## Data-Driven Documents

<https://d3js.org/>

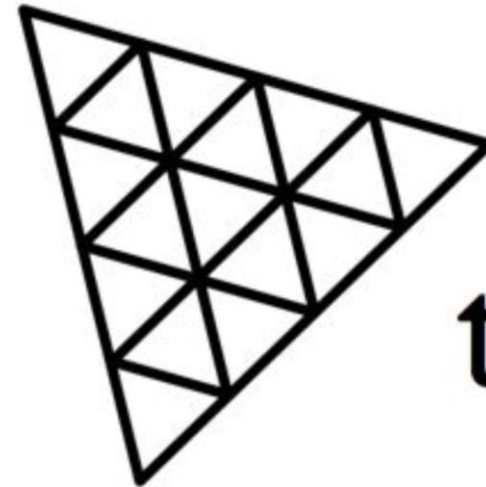
<https://observablehq.com/@d3/gallery>

<https://www.d3indepth.com/>

<https://sbcode.net/threejs/introduction/>

<https://fhtr.org/BasicsOfThreeJS/#27>

<https://threejs.org/>



three.js

Good luck 🍀