{POWER.CODERS}

JavaScript practice & exercises

AGENDA

Today we will look into

- How to use local storage with JavaScript
- Simplifying JavaScript using array functions

`localStorage`

`localStorage` allows JavaScript sites and apps to store and access data right **in the browser** with no expiration date.

The data stored in the browser will persist **even after** the browser window has been closed.

5 METHODS

- `setItem()` : Add key and value to `localStorage`
- `getItem()` : Retrieve a value by the key
- `removeItem()` : Remove an item by key
- `clear()` : Clear all `localStorage`
- `key()` : Passed a number to retrieve nth key

Examples

```
window.localStorage.setItem('name', 'Susanne König');
```

```
window.localStorage.getItem('name');
```

```
window.localStorage.removeItem('name');
```

Work with JSON

`localStorage` can only store **strings**. If you want to store an object or an array, use `JSON.stringify()` and `JSON.parse()`

Work with JSON

`localStorage` can only store **strings**. If you want to store an object or an array, use `JSON.stringify()` and `JSON.parse()`

```
const person = {
    name: "Susanne König",
    location: "Zürich",
}

window.localStorage.setItem('user', JSON.stringify(person));
```

Work with JSON

`localStorage` can only store **strings**. If you want to store an object or an array, use `JSON.stringify()` and `JSON.parse()`

```
const person = {
    name: "Susanne König",
    location: "Zürich",
}

window.localStorage.setItem('user', JSON.stringify(person));
```

```
JSON.parse(window.localStorage.getItem('user'));
```

When to use

`localStorage` is not that very secure, so do not store sensitive data. It is not a substitute for a database.

- Set flag if overlay / popup was shown once and closed
- Store data in form wizard with several steps. Better to use `sessionStorage`

Online resources

- Can I use localStorage
- localStorage vs sessionStorage
- Guide to using localStorage

IMPROVING THE TIP CALCULATOR

TIP CALCULATOR

Change the code of your tip calculator to use the DOM

- add input fields
- add event listeners
- add the result to the DOM

Exercise from 2 weeks ago on Thursday. If you have not done it yet, do that first

Tip calculator

Add a currency converter

Use this free currency API.

- add 2 additional buttons to HTML: convert to USD / convert to EUR
- add event listeners and call the API to convert your CHF values to the chosen currency
- add the result to the DOM

Code refactoring

What is code refactoring?

Rewriting code to improve it and make it cleaner is refactoring.

What is code refactoring?

Rewriting code to improve it and make it cleaner is refactoring.

It is rare to design a program correctly on first go.

What is code refactoring?

Rewriting code to improve it and make it cleaner is refactoring.

It is rare to design a program correctly on first go.

- Requirements can change

What is code refactoring?

Rewriting code to improve it and make it cleaner is refactoring.

It is rare to design a program correctly on first go.

- Requirements can change
- Knowledge can change

Don't repeat yourself

One basic principle of code refactoring.

You can achieve that by

- Using variables to prevent duplication
- Improving event handling
- Improving class handling

Be careful when refactoring

Refactoring code always involves several steps to ensure that the working code does not break accidentally.

BE CAREFUL WHEN REFACTORING

Refactoring code always involves several steps to ensure that the working code does not break accidentally.

Break the change down in to small, independent steps that should each be safe. At each stage you **test the change** and make sure that things still work.

Be careful when refactoring

Refactoring code always involves several steps to ensure that the working code does not break accidentally.

Break the change down in to small, independent steps that should each be safe. At each stage you **test the change** and make sure that things still work.

If they don't work then it's very easy to go back and undo the last small change you made, or review it to see what went wrong.

Simplifying your JS with array functions

array.filter()

```
//the old way
function WhoIsMe(){
  let names = ["Susanne","Mark","Linus","Hussam"];
  for(let name of names){
    if(name==="Susanne")
      return "Susanne";
  }
}
```

array.filter()

```
//the old way
function WhoIsMe(){
  let names = ["Susanne","Mark","Linus","Hussam"];
  for(let name of names){
    if(name==="Susanne")
      return "Susanne";
  }
}
```

```
//the new way
let names = ["Susanne","Mark","Linus","Hussam"];
let me = names.filter(name => {
  return name==="Susanne";
});
```

array.map()

```
//the old way
function WhoIsMe(){
  let names = ["Susanne","Mark","Linus","Hussam"];
  for(let name of names){
    if(name==="Susanne")
      return true;

    return false;
  }
}
```

array.map()

```
//the old way
function WhoIsMe(){
  let names = ["Susanne","Mark","Linus","Hussam"];
  for(let name of names){
    if(name==="Susanne")
      return true;

    return false;
  }
}
```

```
//the new way
let names = ["Susanne","Mark","Linus","Hussam"];
let me = names.map(name => {
  return name==="Susanne";
});
```

array.reduce()

```
//the old way
function WhoIsMe(){
  let names = ["Susanne","Mark","Linus","Hussam"];
  for(let name of names){
    if(name==="Susanne")
      return true;
  }
}
```

array.reduce()

```
//the old way
function WhoIsMe(){
  let names = ["Susanne","Mark","Linus","Hussam"];
  for(let name of names){
    if(name==="Susanne")
      return true;
  }
}
```

```
//the new way
let names = ["Susanne","Mark","Linus","Hussam"];
let me = names.reduce(name => {
  return name==="Susanne";
});
```

Online resources

- filter() method
- map() method
- Use map(), reduce() and filter()
- Array methods explained

Exercises

1. Let's improve our to-do list

Use your own code. In case you cannot find it:

My version on Github

What we will do today

1. store the to-dos locally, so you do not have to start over after refreshing the page
2. use "for of" instead of "for" loop
3. use array methods "sort" and "map" to populate the stored list
4. use array method "reduce" to return the number of to do entries (done and not done)
5. use array method "filter" to remove done to dos automatically from your list

2. RGB COLOR GENERATOR

Change your background generator

Click to see layout

- Use sliders for each single RGB value
- When you change slider position, that value changes and the background does as well
- Output the current RGB value to the DOM

3. Bookmark websites

Add website links and names to a list

Click to see layout

- Add new bookmarks
- Delete bookmarks
- Visit the bookmarked website
- Store the data locally in your browser