Routing

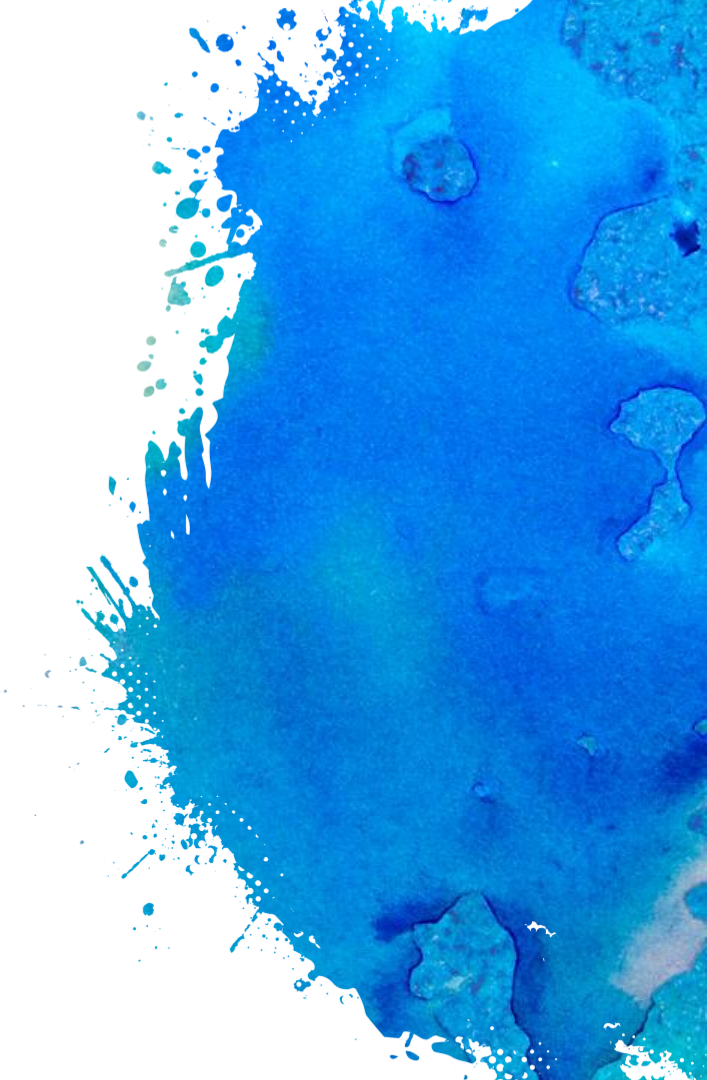
So router....

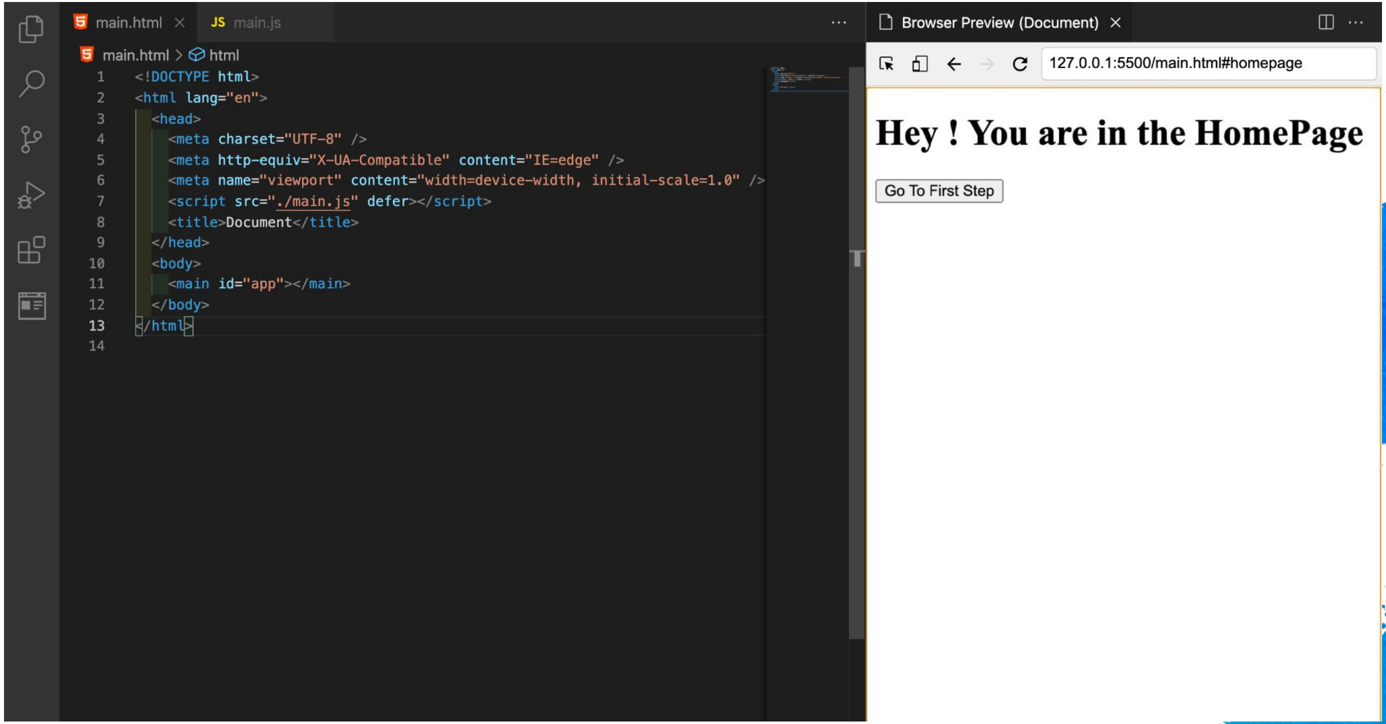
Comes handy being able to modify the URL in SPAs, for client side ... and also to us devS !

It's always a good idea that every element of our project goes in the same line, doesn't it?

Let’s see it in action

Peer Coding , let’s go !
Don't worry about documentation, I got you covered ibn this slides



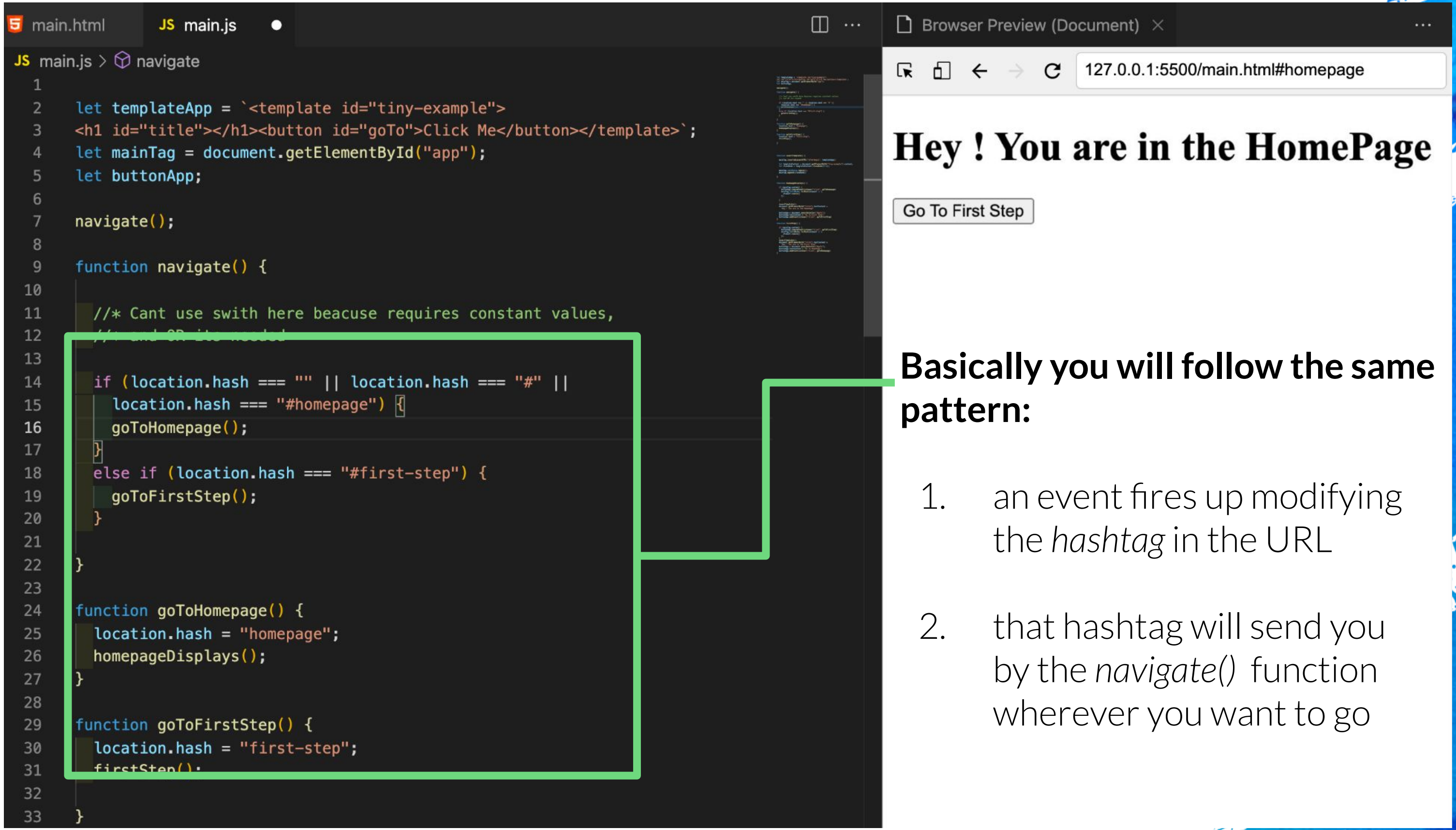


```html
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <script src="./main.js" defer></script>
    <title>Document</title>
  </head>
  <body>
    <main id="app"></main>
  </body>
</html>
```

Hey ! You are in the HomePage

Go To First Step

Let's use the famous <template> tags ! That are now like family to you, I hope :)



```
let templateApp = `<template id="tiny-example">
<h1 id="title"></h1><button id="goTo">Click Me</button></template>`;
let mainTag = document.getElementById("app");
let buttonApp;

navigate();

function navigate() {

  //* Cant use swith here beacuse requires constant values,
  //* and OR its needed

  if (location.hash === "" || location.hash === "#" ||
    location.hash === "#homepage") {
    goToHomepage();
  }
  else if (location.hash === "#first-step") {
    goToFirstStep();
  }

}

function goToHomepage() {
  location.hash = "homepage";
  homepageDisplays();
}

function goToFirstStep() {
  location.hash = "first-step";
  firstStep();

}
```

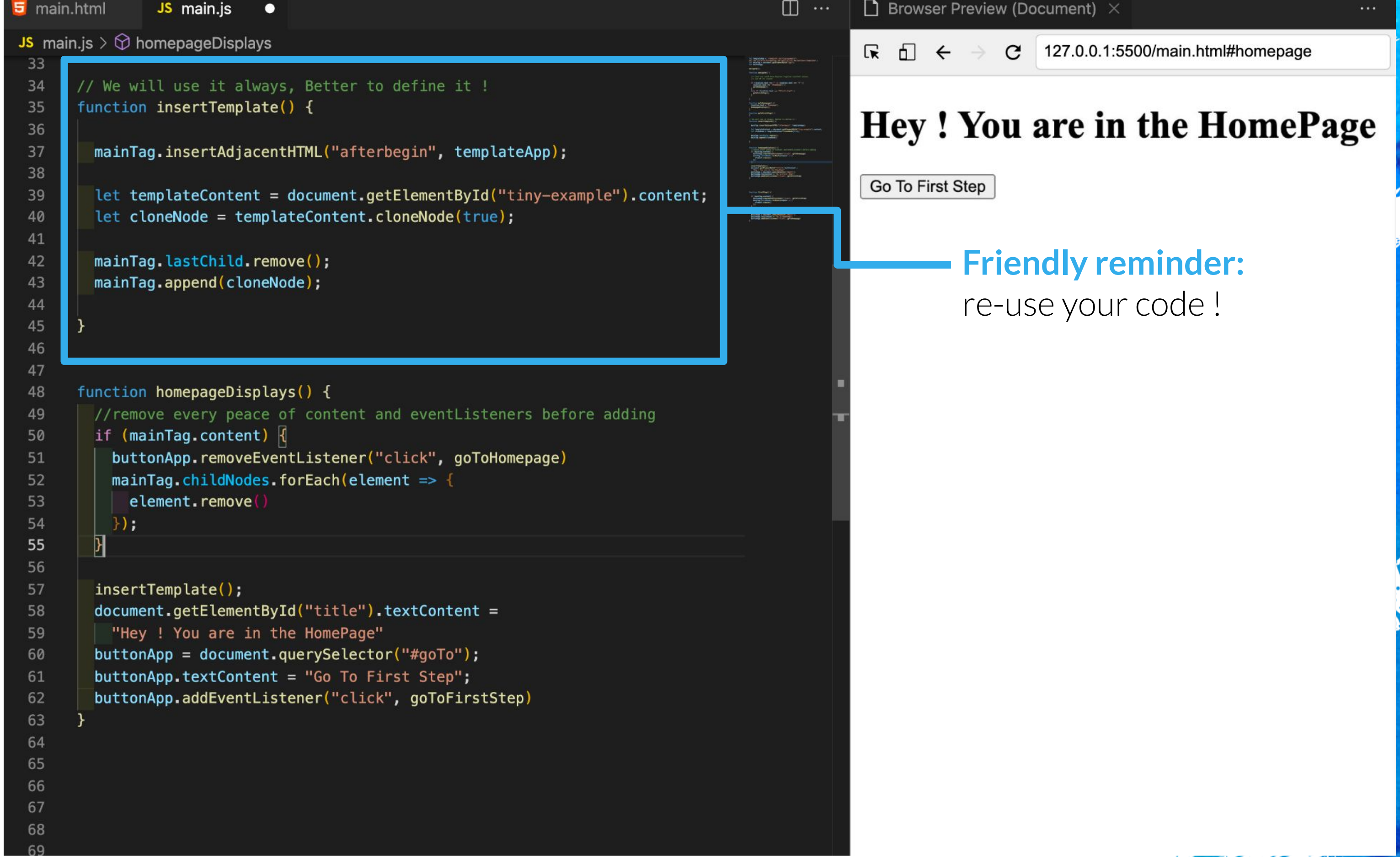
Hey ! You are in the HomePage

Basically you will follow the same pattern:

1. an event fires up modifying the *hashtag* in the URL

2. that hashtag will send you by the *navigate()* function wherever you want to go

Decide which hashtags you want to use and define the functions that the navigate() will use as callback

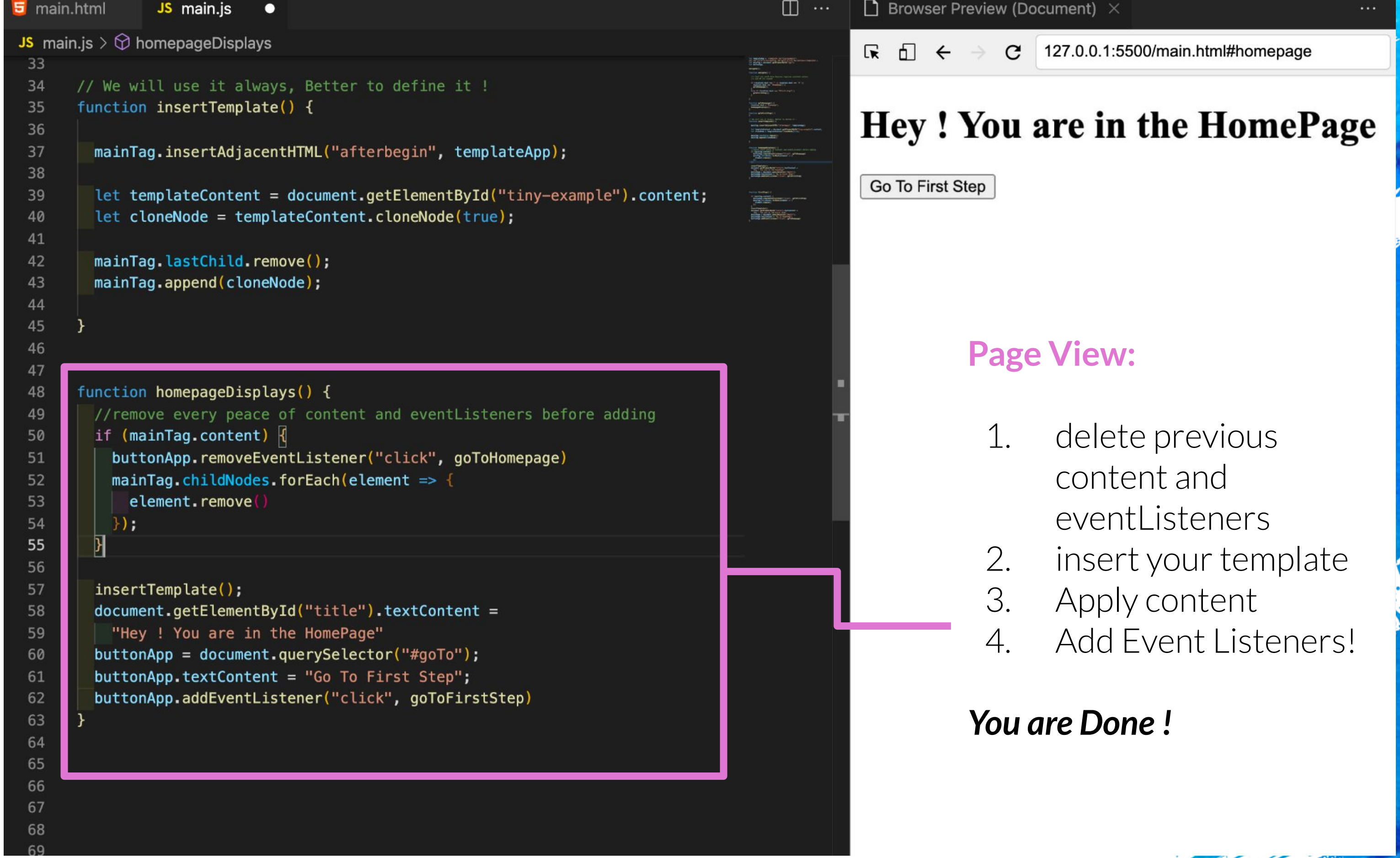

main.html
main.js
main.js > homepageDisplays
Browser Preview (Document)
127.0.0.1:5500/main.html#homepage
Go To First Step

```
// We will use it always, Better to define it !
function insertTemplate() {

  mainTag.insertAdjacentHTML("afterbegin", templateApp);

  let templateContent = document.getElementById("tiny-example").content;
  let cloneNode = templateContent.cloneNode(true);

  mainTag.lastChild.remove();
  mainTag.append(cloneNode);

}


function homepageDisplays() {
  //remove every peace of content and eventListeners before adding
  if (mainTag.content) {
    buttonApp.removeEventListener("click", goToHomepage)
    mainTag.childNodes.forEach(element => {
      element.remove()
    });
  }

  insertTemplate();
  document.getElementById("title").textContent =
    "Hey ! You are in the HomePage"
  buttonApp = document.querySelector("#goTo");
  buttonApp.textContent = "Go To First Step";
  buttonApp.addEventListener("click", goToFirstStep)
}
```

Hey ! You are in the HomePage

Friendly reminder:
re-use your code !

Now with only 2 views one could not mind, but imagine having hundreds ! **Always think big !**


main.html
main.js
main.js > homepageDisplays
Browser Preview (Document)
127.0.0.1:5500/main.html#homepage

```javascript

// We will use it always, Better to define it !
function insertTemplate() {

  mainTag.insertAdjacentHTML("afterbegin", templateApp);

  let templateContent = document.getElementById("tiny-example").content;
  let cloneNode = templateContent.cloneNode(true);

  mainTag.lastChild.remove();
  mainTag.append(cloneNode);

}


function homepageDisplays() {
  //remove every peace of content and eventListeners before adding
  if (mainTag.content) {
    buttonApp.removeEventListener("click", goToHomepage)
    mainTag.childNodes.forEach(element => {
      element.remove()
    });
  }

  insertTemplate();
  document.getElementById("title").textContent =
    "Hey ! You are in the HomePage"
  buttonApp = document.querySelector("#goTo");
  buttonApp.textContent = "Go To First Step";
  buttonApp.addEventListener("click", goToFirstStep)
}
```

Hey ! You are in the HomePage

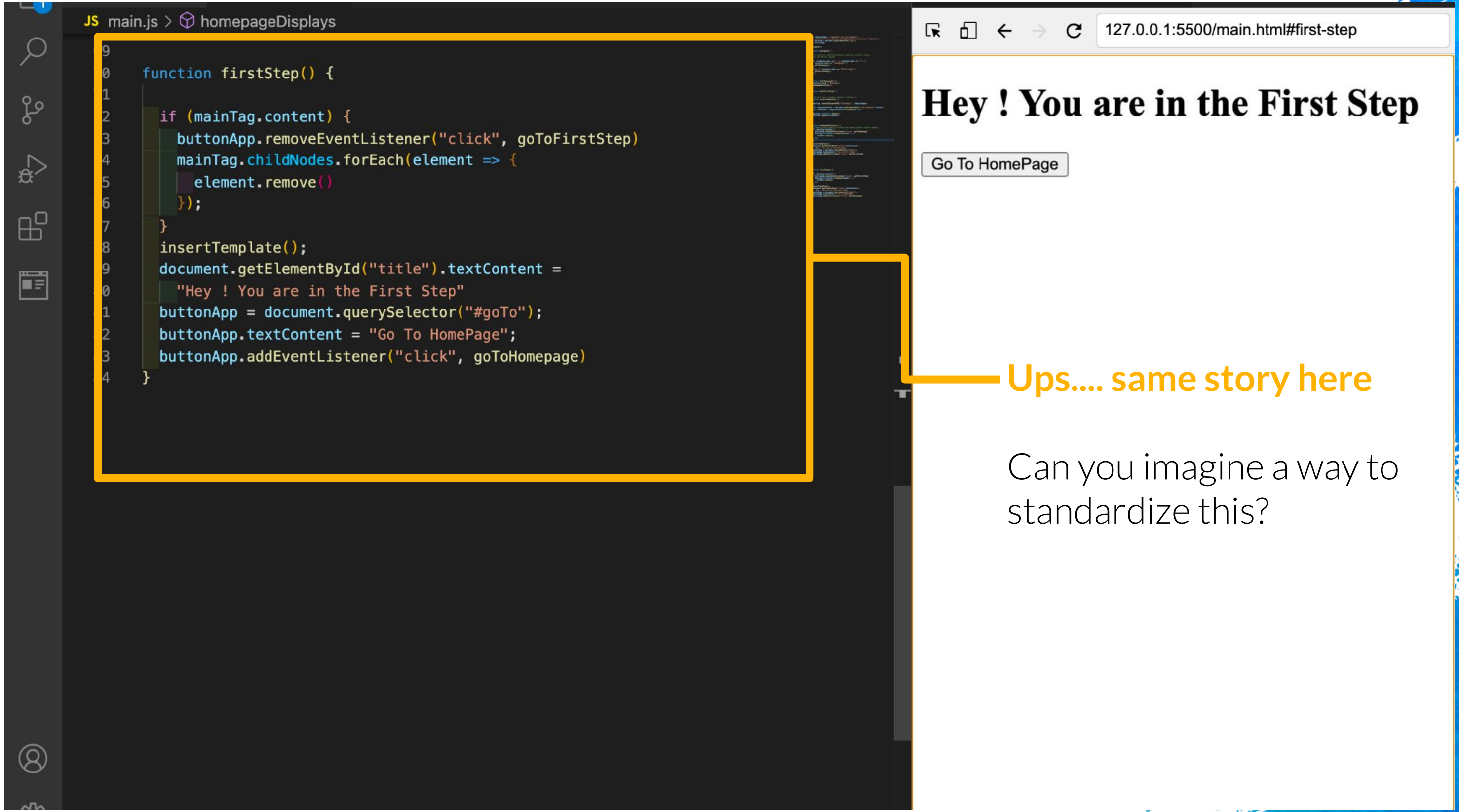
Go To First Step

Page View:

1. delete previous content and eventListeners
2. insert your template
3. Apply content
4. Add Event Listeners!

You are Done !

Decide which hashtags you want to use and define the functions that the navigate() will use as callback



```javascript
function firstStep() {

  if (mainTag.content) {
    buttonApp.removeEventListener("click", goToFirstStep)
    mainTag.childNodes.forEach(element => {
      element.remove()
    });
  }
  insertTemplate();
  document.getElementById("title").textContent =
    "Hey ! You are in the First Step"
  buttonApp = document.querySelector("#goTo");
  buttonApp.textContent = "Go To HomePage";
  buttonApp.addEventListener("click", goToHomepage)
}
```

Hey ! You are in the First Step

Ups.... same story here

Can you imagine a way to standardize this?

Using Objects in this subject could come in handy ... but it doesn't matter in this case study :)

Our STAR is:

- **- hash -**

We will use the string that comes after the # (hashtag) to elaborate the different routes or directions to be followed.

location.hash;

The hash property of the Location interface returns a USVString containing a '#' followed by the fragment identifier of the URL — the ID on the page that the URL is trying to target.

location.hash===”x”

So… our code basically “routes” or send you to the script that its needed depending of the hash value.

Let's say x is “homepage”, so the router will be setted to get the part of the code where the view for the homepage it's being displayed. (hopefully)

Useful Links

Some docu

https://developer.mozilla.org/en-US/docs/Web/API/USVString

https://developer.mozilla.org/en-US/docs/Web/API/Location/hash

Another idea

https://medium.com/@bryanmanuele/how-i-implemented-my-own-spa-routing-system-in-vanilla-js-49942e3c4573

Git-hub

The code for this example is in here, don't go nuts:

https://github.com/Paola3stefania/templates-router-lesson