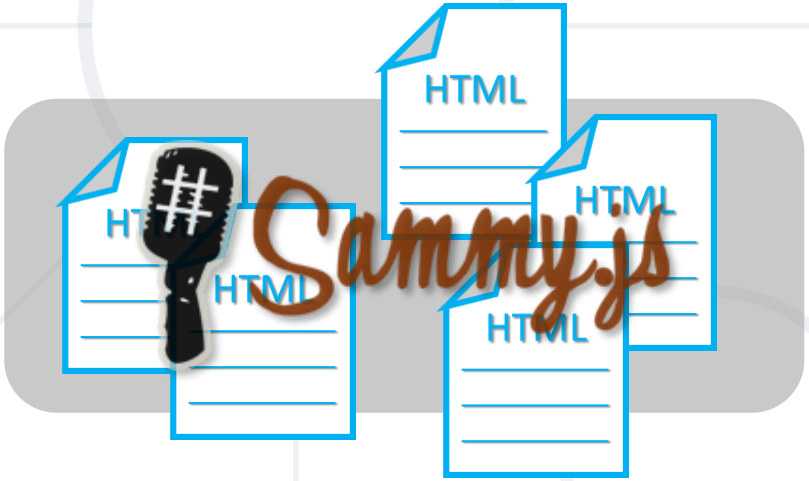
Routing and Patterns

Browser Routing Design Patterns in JS



SoftUni Team

Technical Trainers

Software University

http://softuni.bg

Table of Content

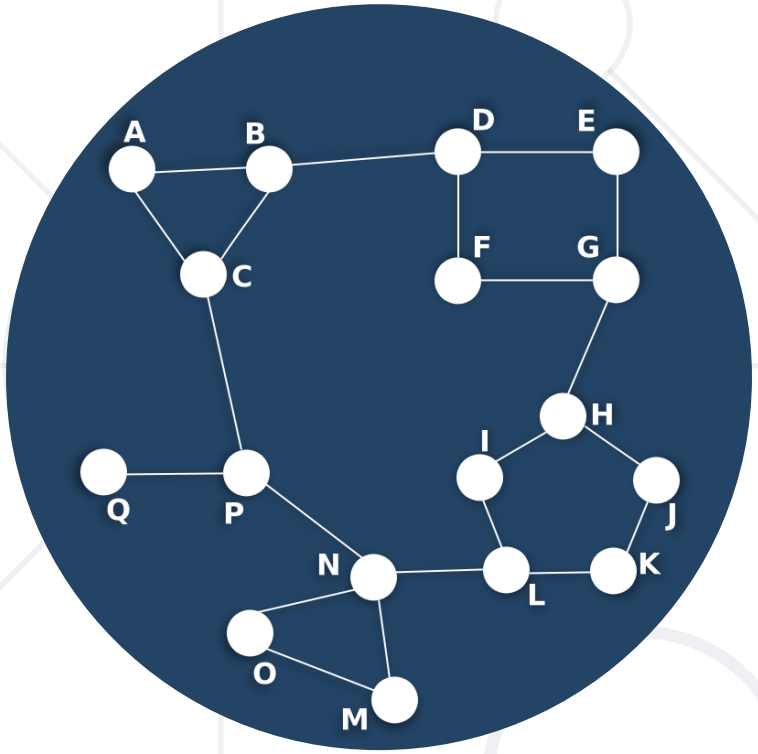
SoftUni Foundation

Have a Question?

SoftUni Foundation

sli.do

#JSCORE



Routing Concepts

Navigation for Single Page Apps

SPA Applications

SoftUni Foundation

- A **single-page application** (SPA) is a website that re-renders its content in response to navigation actions (e.g. clicking a link) **without reloading of the page**.
- Single-page applications can **use state from an external source** (i.e. the URL location) or **track state internally**
  - Internal state SPAs are limited because there is **only one** "entry"
  - With location-based SPAs, the location is **always updating** as you navigate
  - To have location-based SPA we need a special object "Router"

Location

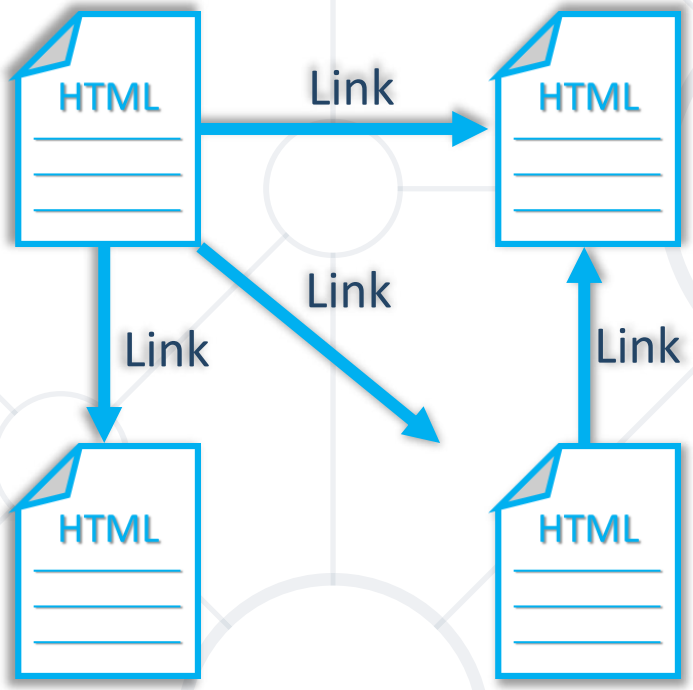
SoftUni Foundation

https://www.example.com/one?key=value#trending

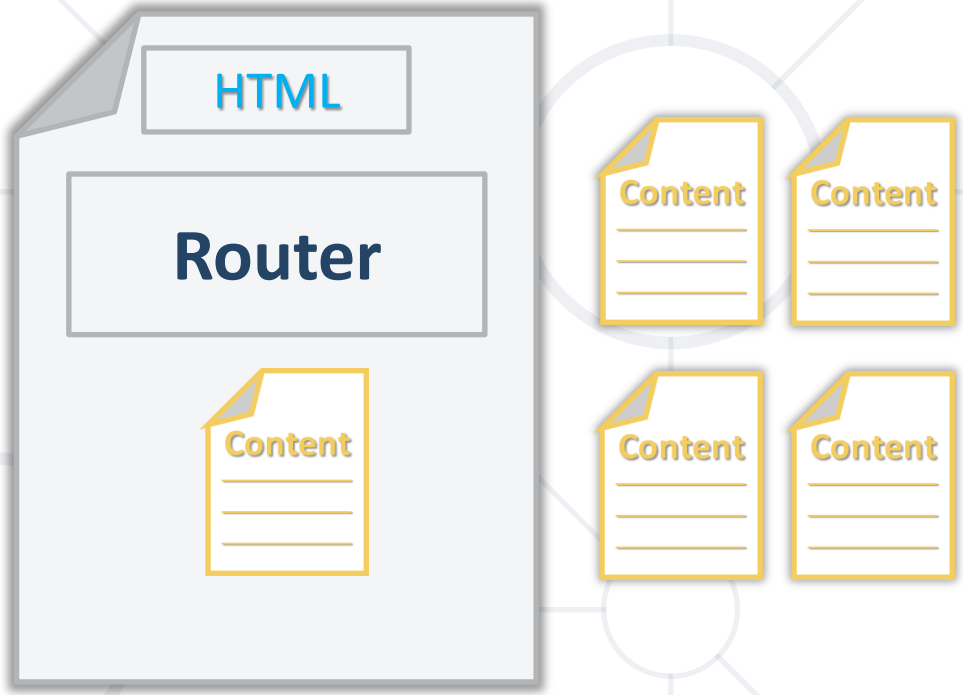
```
location = {
  protocol: "https:",
  hostname: "www.example.com",
  pathname: "/one",
  search: "?key=value",
  hash: "#trending"
}
```

What is Routing?

- Standard Navigation



- Navigation using Routing



- **Routing** allows navigation, **without reloading** the page

How Routers Work

- A **Router** loads the appropriate content when the **location changes**
  - E.g. when the user manually **enters an address**
- Conversely, a change in content is reflected in the address bar
  - E.g. when the user **clicks on a link**
- Benefits:
  - Load all scripts **only once**
  - **Maintain state** across multiple pages
  - Browser **history** can be used
  - Build User Interfaces that **react quickly**

How Routers Work

SoftUni Foundation

- Hash-based routing
- Using the **#hash** part of the URL to simulate different content
- The routing is possible because c hanges in the hash **don't trigger page reload**

```
var url = null;
var getCurrent = function () {
  return window.location.hash;
};
var listen = function () {
  var current = getCurrent();
  if (current !== url) {
    console.log('URL changed to'
      + current);
    url = current;
  }
  setTimeout(listen, 200);
};
Listen();
```

How Routers work

- History API - The DOM **window** object provides access to the browser's history through the **history** object. It exposes useful methods and properties that let you move back and forth through the user's history, as well as manipulate the contents of the history stack
- **HTML5** introduced the **history.pushState()** and **history.replaceState()** methods, which allow you to add and modify **history entries**, respectively
- These methods work in conjunction with the **window.onpopstate** event

The pushState() method

SoftUni Foundation

- **pushState()** takes three parameters: a **state object**, a **title** (which is currently ignored), and (optionally) a **URL**
  - **State** a JavaScript object which is associated with the new history entry
  - **Title** - browsers currently ignore this parameter
    - Passing the **empty string** here should be safe against future changes to the method
  - **URL** - The new history entry's URL is given by this parameter.
    - The browser won't attempt to load this URL after a call to pushState() but it might attempt to load the URL later, for instance after the user restarts the browser.
    - It must be of the same origin as the current URL.

The replaceState() method

- **history.replaceState()** operates exactly like history.pushState() except that replaceState() **modifies the current history entry** instead of creating a new one
- It is particularly useful when you want to update the **state object** or **URL** of the current history entry in response to some user action

```
var stateObj = { facNum: "56789123" };
history.pushState(stateObj, "", "student.html");

history.replaceState(stateObj, "", "newStudent.html");
```

The popstate event

SoftUni Foundation

- A **popstate** event is dispatched to the window every time the active history entry changes
- If the history entry being activated was created by a call to **pushState** or affected by a call to **replaceState**, the **popstate** event's **state property** contains a copy of the history entry's state object
- You can read the state of the current history entry without waiting for a **popstate** event using the **history.state property**

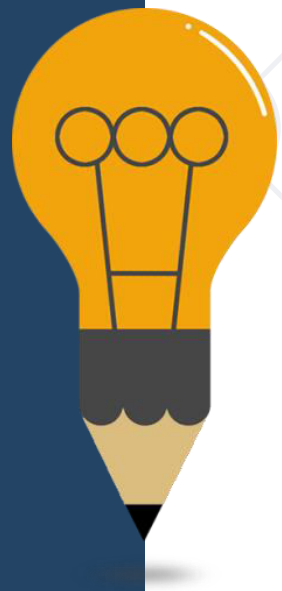
Live Demo

Routing with Sammy.js

Overview and Examples

Sammy.js Overview

- Sammy is a lightweight **routing library**
- Modular design with **plugins** and **adapters**
- **Requires** with jQuery
- Many **additional features**



```
const app = Sammy('#main', function() {
  this.get('#/index.html', () => {
    this.swap('Index');
  })
});
```

Installation

- Download Sammy using WebStorm's terminal

```
npm install --save sammy
```

- Or download from sammyjs.org
- Browser builds will be located in:

```
node_modules/sammy/lib/
```

- Sammy requires jQuery to work!

Note: It's best if your project has a `package.json` file

Application Initialization

SoftUni Foundation

- Create a Sammy instance to initialize your application

Invoke library

Element selector

```
const app = Sammy('#main', function () {
  // Define routes and other logic here
});

$(() => app.run()); // Activate
```

- You may have **multiple apps** running
- Each selector can only hold one app
  - If you refer to it again, it **extends the functionality**

Creating Routes

SoftUni Foundation

- The main **building block** of Sammy is the **route**
  - Defined by method and address (URI)
- Place this block inside a **Sammy initializer**:

Route **Method**

Route **address**

```
this.route('get', '#/about', function() {
  this.swap('<h2>Contact Page</h2>');
});
```

- A note on using **this**: it holds a **reference** to the **router object**, but may not work correctly in an **arrow function**

Route Aliases

- Each method has an alias for shorter code

```
this.get('#/catalog', loadBooks)
```

```
this.post('#/login', userLogin)
```

```
this.put('#/catalog/:bookId', updateBook)
```

```
this.del('#/catalog/:bookId', deleteBook)
```

URL Parameters

- **Parameters** allow for dynamic routes
  - E.g. products in a catalog will load the same page

Parameter **name**

Receive **context**

```
this.get('#/catalog/:productId', (context) => {
  console.log(context.params.productId);
});
```

Access passed in value

- You can get the whole path using **this.path**

Hello Sammy

SoftUni Foundation

index.html

```html
<!DOCTYPE html>
<head>
  <meta charset="UTF-8">
  <title>Hello Sammy</title>
  <!-- Include jQuery and Sammy distributions -->
</head>
<body>
  <header>
    <h1>Hello Sammy</h1>
    <a href="#/index.html">Home</a>
    <a href="#/about">About</a>
    <a href="#/contact">Contact</a>
  </header>
  <div id="main"></div>
</body>
</html>
```

Hello Sammy (2)

SoftUni Foundation

app.js

```
const app = Sammy('#main', function () {
    this.get('#/index.html', () => {
        this.swap('<h2>Home Page</h2>');
    });
    this.get('#/about', () => {
        this.swap('<h2>About Page</h2>');
    });
    this.get('#/contact', () => {
        this.swap('<h2>Contact Page</h2>');
    });
});

$(() => {
    app.run();
});
```

Handling Forms

SoftUni Foundation

- **Forms** inside the main element are automatically handled

Route **address**

Route **method**

```
<form action="#/login" method="post">
  User: <input name="user" type="text">
  Pass: <input name="pass" type="password">
  <input type="submit" value="Login">
</form>
```

```
this.post('#/login', (context) => {
  console.log(context.params.user);
  console.log(context.params.pass);
});
```

Names of inputs

Integrating Handlebars

SoftUni Foundation

- Download and include the Handlebars source in your HTML
- Include **sammy.handlebars.js** (found under **lib/plugins**)
- Load the plugin inside a Sammy initializer:

```
this.use('Handlebars', 'hbs');
```

Template file **extension**

- Create a **RenderContext** with **render**, **load** or **partial**

Using Handlebars

SoftUni Foundation

greeting.hbs

```
<h1>{{title}}</h1>
<p>Hello, {{name}}!</p>
```

app.js

```
const app = Sammy('#main', function () {
  this.use('Handlebars', 'hbs');
  this.get('#/hello/:name', function() {
    this.title = 'Hello!'
    this.name = this.params.name;
    this.partial('greeting.hbs');
  });
});
$(() => app.run());
```

Load and swap in the template

Using Handlebars (2)

SoftUni Foundation

- Load a list of **partial templates** (inside a **route** definition):

```
this.loadPartials({
  firstPartial: 'path-to/first.hbs',
  secondPartial: 'path-to/second.hbs',
  thirdPartial: 'path-to/third.hbs'
}).then(function(context) => {
  console.log(context.partials);
  this.partial('pageTemplate.hbs');
});
```

- The **callback** will be executed once all partials are loaded
- Templates are **cached** – there's no need to manually cache them

Additional Features

- Redirect

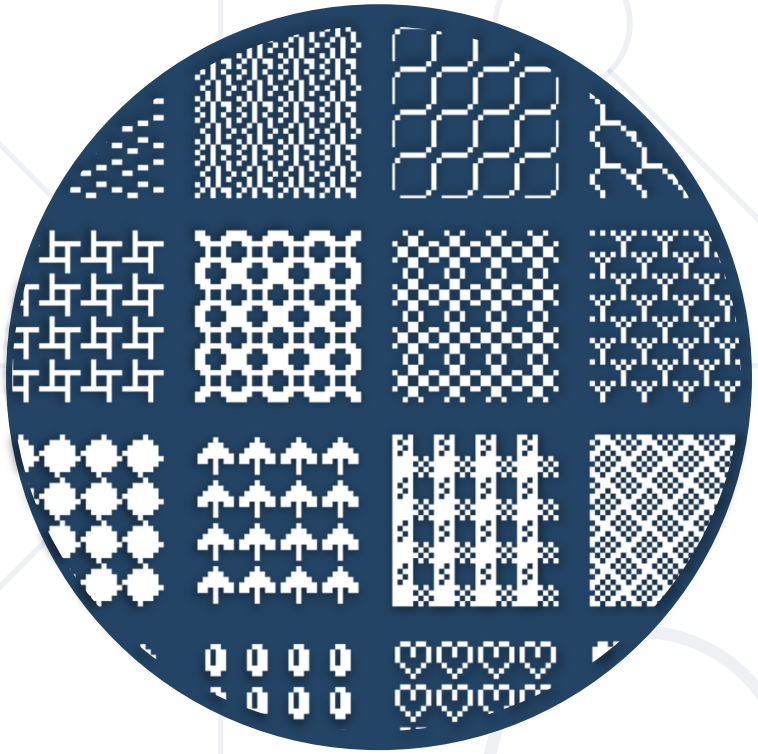
```
this.redirect('#/other/route');
this.redirect('#', 'other', 'route');
```

- Custom events

```
// Register event handler
this.bind('event-name', eventHandlerFunction);

// Raise event
this.trigger('event-name', data);
```

- Useful plugins (found under **lib/plugins**):
  - Storage and Session
  - OAuth2



Routing with Sammy.js

Overview and Examples

Live Exercise

Summary

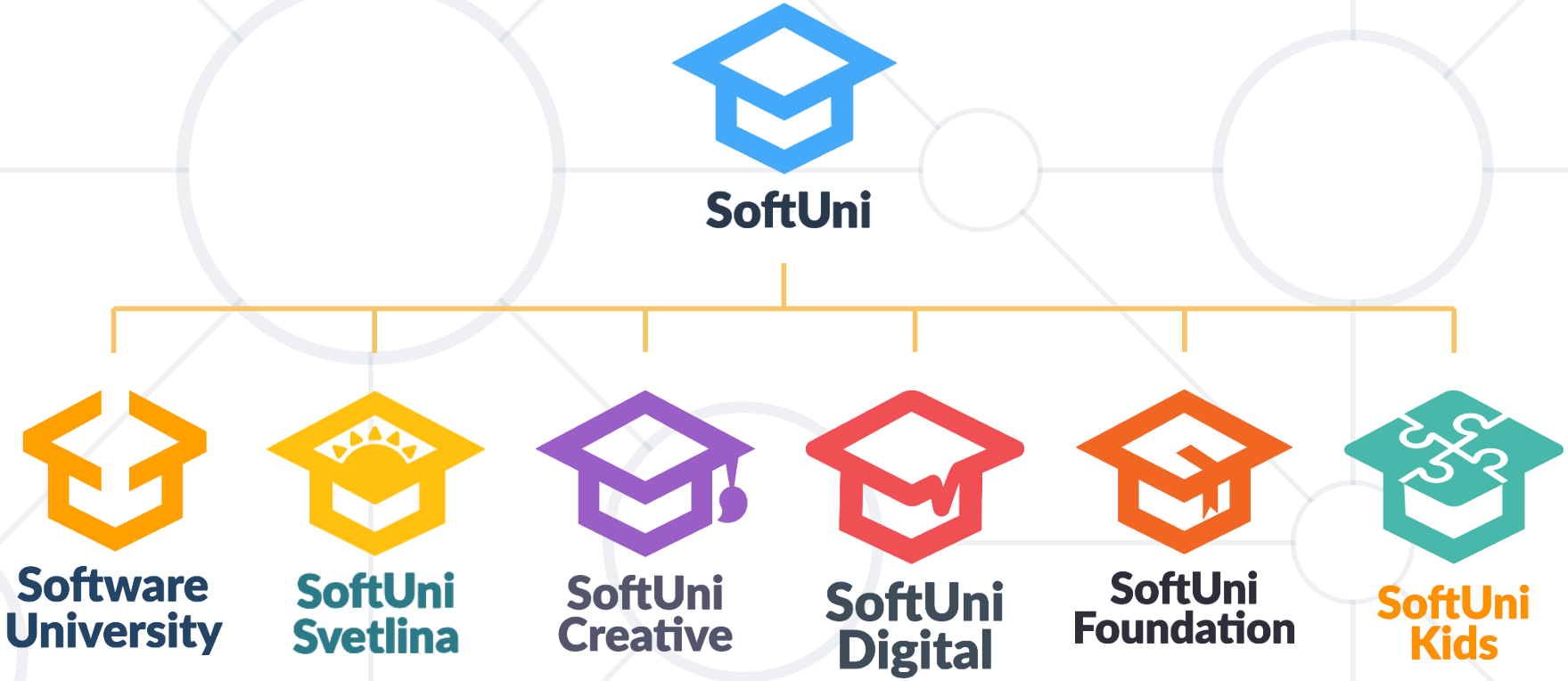
SoftUni Foundation

- Browser **Routing** allows SPAs to use **history**
- **Sammy.js** is a simple routing library

```
const app = Sammy('#main',
function () {
  this.get('index.html', () => {
    this.swap('<h1>Index
Page</h1>');
  })
});
```

- **Modular code** is more maintainable

Questions?



https://softuni.bg/trainings/2082/js-applications-november-2018

SoftUni Diamond Partners

SoftUni Foundation

XSsoftware

SBTech
we know sports

telenor

SoftwareGroup
doing it right

NETPEAK

SmartIT

Postbank
Решения за твоето утре

SUPER HOSTING .BG

INDEAVR
Serving the high achievers

INFRAGISTICS

LIEBHERR

æternity

codexio

SoftUni Organizational Partners

SoftUni Foundation

ИНФОРМАЦИОННО ОБСЛУЖВАНЕ

OneBit SOFTWARE

WORLD OF MYTHS

Lukanet.com

codeXio

Trainings @ Software University (SoftUni)

- Software University – High-Quality Education and Employment Opportunities
  - softuni.bg
- Software University Foundation
  - http://softuni.foundation/
- Software University @ Facebook
  - facebook.com/SoftwareUniversity
- Software University Forums
  - forum.softuni.bg

License

- This course (slides, examples, demos, videos, homework, etc.) is licensed under the "Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International" license