## **COURSEWARE**

# **Professional Skills** Agile Fundamentals Jira Git **Databases Introduction** Java Beginner Maven Testing (Foundation) Java Intermediate HTML **CSS** Javascript What is JavaScript Getting started with JS Variables Data types ASI 0 Strict mode Iteration Conditionals with Truthy / Falsey Objects, Arrays + JSON Structuring JS Code Destructuring Scope Functions, function expressions and arrow functions The ECMAScript 6 Specification OOP in JavaScript **Best Practices** Closures Callbacks and Promises Cookies Hoisting Prototypes **Query Parameters**

**Higher Order Functions** 

## **Query Parameters**

#### Contents

- Overview
- <u>Tutorial</u>
- Exercises

#### Overview

Query Parameters refer to URL-based parameters which are appended to the end of a URL address. In JavaScript, these carry significant meaning.

#### **Tutorial**

The URL Search Parameters URL Search Params Java Script interface defines utility methods to work with the query string of a URL:

```
https://test.com/?name=Chris
https://test.com/hello?name=Chris
```

In this case, we have a single query parameter named name, with the value Chris.

You can have multiple parameters like this:

```
https://test.com/hello?name=Chris&age=20
```

The parameters passed as a query string are normally used server-side, to generate a proper response.

To access the value of a query inside the browser, using JavaScript, we have a special API called URLSearchParam, supported by all modern browsers:

```
const params = new URLSearchParams(window.location.search);
```

(note: don't pass the full URL as a parameter to URLSearchParams(), but only the query string part of the URL, which is accessed using window.location.search.)

In case of the above example: window.location.search is equal to the string? name=Chris.

Now that you have the params object you can query it.

Check if a parameter was passed:

```
params.has('test');
```

Get the value of a parameter:

```
params.get('test');
```

Iterate over all the parameters using for...of:

```
const params = new URLSearchParams(window.location.search)
for (const param of params) {
  console.log(param)
}
```

A parameter can have more than one value; in this case, we pass the same parameter name multiple ties like this:

_	-
0	Web Storage
0	DOM Manipulation
0	Handling Events and Timed Events
0	Asynchronous Programming
0	HTTP-Requests
0	XMLHttpRequests
0	Fetch API
Spring Boot	
Selenium	
Sonarqube	
Advanced Testing (Theory)	
Cucumber	
MongoDB	
Express	
NodeJS	
React	
Express-Testing	
Networking	
Security	
Cloud Fundamentals	
AWS Foundations	
AWS Intermediate	
Linux	
DevOps	
Jenkins Introduction	
Jenkins Pipeline	
Markdown	

IDE Cheatsheet

https://test.com/hello?name=Chris&name=Bacon

We have no way to detect if a parameter is passed more than once. If we use params.get('name'), we will only get the first value back.

We can use params.getAll('name') to get back an array with all the values passed.

In addition to has(), get(), and getAll(), the URLSearchParams API offers several other methods that we can use to loop through the parameters:

- forEach() iterates over the parameters
- entries() returns an iterator containing all the parameters key/values
- keys() returns an iterator containing all the parameters keys
- values() returns an iterator containing all the parameters values

Note: The following two lines are equivalent:

```
for (const [key,value] of mySearchParams) {}
for (const [key,value] of mySearchParams.entries()) {}
```

A few other methods that are handy to know are:

- append() to append a new parameter to the object
- delete() to delete an existing parameter
- set() to set the value of a parameter
- sort() to sort parameters by key/value
- toString() to generate a query string from the values

### **Exercises**

There are no exercises for this module.