<WA/>
2024

Web Applications

Introduction to the course

Enrico Masala
Antonio Servetti

Goal

- Understanding web architectures
- Understanding and mastering web application design and development
- Gaining in-depth knowledge of the JavaScript language and ecosystem
- Becoming familiar with one of the most popular JavaScript frameworks (React)
- …with special focus on the front-end
- …and especially on SECURITY of web applications

What We Will Learn

JavaScript as a language

- ECMAScript ES6
- Language constructs
- In-depth semantics
- Functional, Asynchronous, Modular, …

The browser ecosystem

- HTML, CSS, page structure
- DOM
- JavaScript in the browser
- Events, Properties, Handlers, APIs

Single Page Applications

- Server-side (bare minimum) with node
- API development
- Backend storage
- Sessions and Authentication

React framework

- Components, Properties, State
- JSX
- Hooks
- Router

Calendar… At a Glance!

1. Intro to JS: basics, objects, functions
2. Intro to JS: async programming, callbacks, DB interaction + Intro to Web
3. Server-side with Express; API design
4. HTML, CSS, Bootstrap
5. JS: modules and other topics, + JS in the browser
6. Intro to React
7. React: props and state
8. React: context, life cycle, forms
9. React router
10. Data fetching and client-server interaction (in React)
11. Authentication

Course Organization

- Classes
  - 3 or 6 h/week
  - Lectures + Exercises (*mixed*)
- Laboratories (room 10i)
  - 1.5 or 1.5+1.5 h/week
  - 2 Lab groups (see later for the split)
  - Starting 2nd week
- **Detailed schedule on the website**
  - Sometimes Thursday free

	MO	TU	WE	TH	FR
08:30					
10:00	🟥 R4 🟩 R4				
11:30	🟥 R4 🟩 R4	🟩 10i			
13:00		🟩 10i		🟥 16	
14:30				🟥 16	
16:00					
17:30					

Classes

- In person, (mostly) in rooms with power outlets at the desks
  - bring your own computer, if possible, to follow the examples/exercises
- Video-recorded and made available soon after each class
  - *not* streamed live
- A few times during the course, we will give you some materials to read/watch before a lecture
  - relatively *short* and published *in advance*

Laboratories

- Starting March 12, 2024
- In rooms with power outlets at the desks
  - **No computers are available in the room, bring your own**

- Text online, some days in advance
- Exercises to be done during Lab hours

- Solution will be posted on GitHub
  - around 1 week after the end of each lab

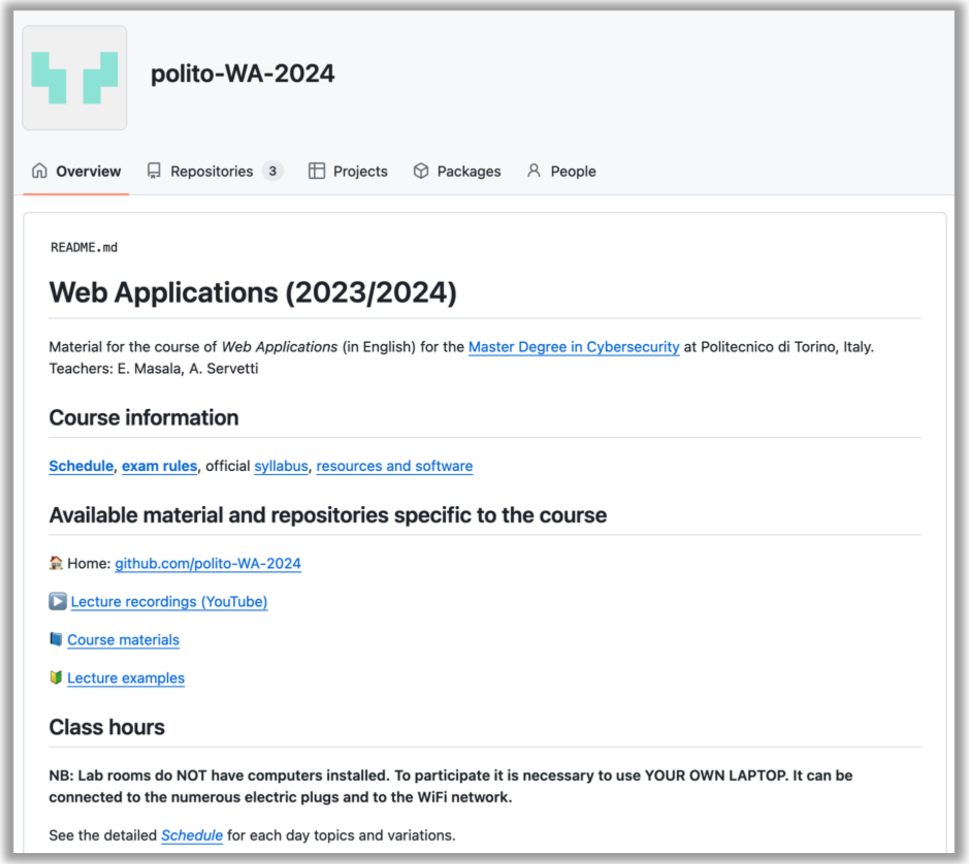
Laboratories

- You will build a simple project during the labs
  - Step by step, following the course topics
- Some labs will last one week, others will span multiple weeks

- Two slots, divided by last name:
  - A-L
  - M-Z

Learning Material

- Everything is on GitHub: https://github.com/polito-WA-2024
- Course website
  - Slides
  - Full schedule
  - Links and supplementary material
  - Examples, exercises, labs, exams, …
- Video lectures (screencasts)
  - YouTube - https://www.youtube.com/playlist?list=PLuZyhAOPm9pO_bXADlRo_2m1JNohIW7qA
  - Portale della Didattica (download only)



polito-WA-2024

README.md

Web Applications (2023/2024)

Material for the course of *Web Applications* (in English) for the Master Degree in Cybersecurity at Politecnico di Torino, Italy.
Teachers: E. Masala, A. Servetti

Course information

Schedule, **exam rules**, official syllabus, resources and software

Available material and repositories specific to the course

🏠 Home: github.com/polito-WA-2024

▶️ Lecture recordings (YouTube)

📘 Course materials

🔰 Lecture examples

Class hours

NB: Lab rooms do NOT have computers installed. To participate it is necessary to use YOUR OWN LAPTOP. It can be connected to the numerous electric plugs and to the WiFi network.

See the detailed *Schedule* for each day topics and variations.

Communications

- We will use **Telegram** for the main communications
  - Among students, with teachers, etc.
  - Announcements and official information, and Q&A (using “topics” in Telegram)
- Feel free to contact the teachers for feedback and questions
  - **questions** of general interest (including exam) must be posted in the group, so that everybody can see the answer. NB: Do not exchange suggestions to solve the exam.
- Link to the Telegram group: https://t.me/+fsb5Ie5kZa02NTM0
- Emails can be an **alternative** for slower, more articulated, and private individual communications

About the Exam

1. Project development
   - **Individual,** up to 26 points
   - 20 days of time for development
2. Oral discussion (on the project)
   - individual <u>and mandatory</u>, up to 6 points
   - “live” correction of the submitted project and discussion
   - when: from the official exam day (or later depending on # of submissions)

NB: <u>Similarity checks</u> will be run: excessively similar solutions will be null

Full exam rules in the course website (under "Exams")

Project Development

What

- Develop a web application using
  - React + JavaScript
  - Node + Express
  - SQLite
- According to a functional specification
  - published 20 days before each official exam date

How

- Individually (i.e., not in group)
- Using GitHub Classroom
  - commit + push your project
- Teacher's Evaluation
  - running the application on a clean recent **Linux** distro (with node)
  - examining the code
  - Particular attention to security

Oral Discussion

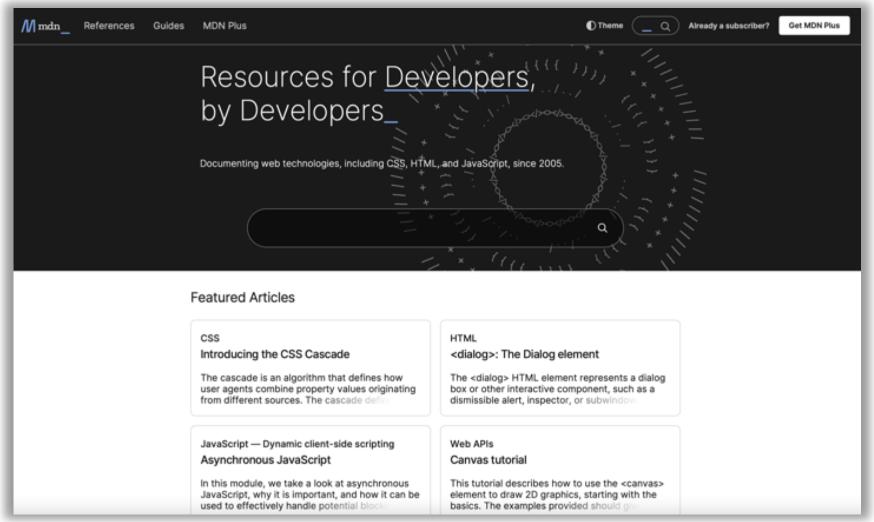
Goals

- To ensure that each student developed the web application by themselves
- To evaluate how much the student can explain the exact behaviour of the code

Evaluation Criteria

- Theoretical and practical knowledge of the project design
- Theoretical and practical knowledge of the project code base
- Readiness and clarity in the replies
- Special focus on security aspects

Resources (fundamentals)



Resources for Developers, by Developers_

Documenting web technologies, including CSS, HTML, and JavaScript, since 2005.

Featured Articles

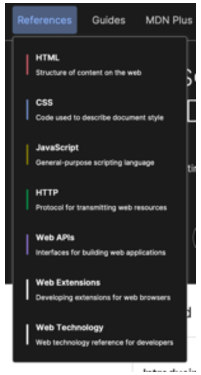
CSS
Introducing the CSS Cascade
The cascade is an algorithm that defines how user agents combine property values originating from different sources. The cascade defin

HTML
<dialog>: The Dialog element
The <dialog> HTML element represents a dialog box or other interactive component, such as a dismissible alert, inspector, or subwindow

JavaScript — Dynamic client-side scripting
Asynchronous JavaScript
In this module, we take a look at asynchronous JavaScript, why it is important, and how it can be used to effectively handle potential block

Web APIs
Canvas tutorial
This tutorial describes how to use the <canvas> element to draw 2D graphics, starting with the basics. The examples provided should giv

Mozilla Developer Network (**MDN**)
https://developer.mozilla.org/



HTML
Structure of content on the web

CSS
Code used to describe document style

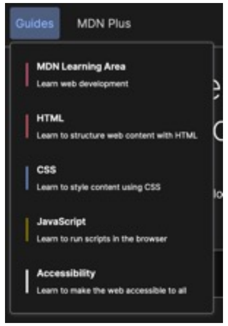
JavaScript
General-purpose scripting language

HTTP
Protocol for transmitting web resources

Web APIs
Interfaces for building web applications

Web Extensions
Developing extensions for web browsers

Web Technology
Web technology reference for developers



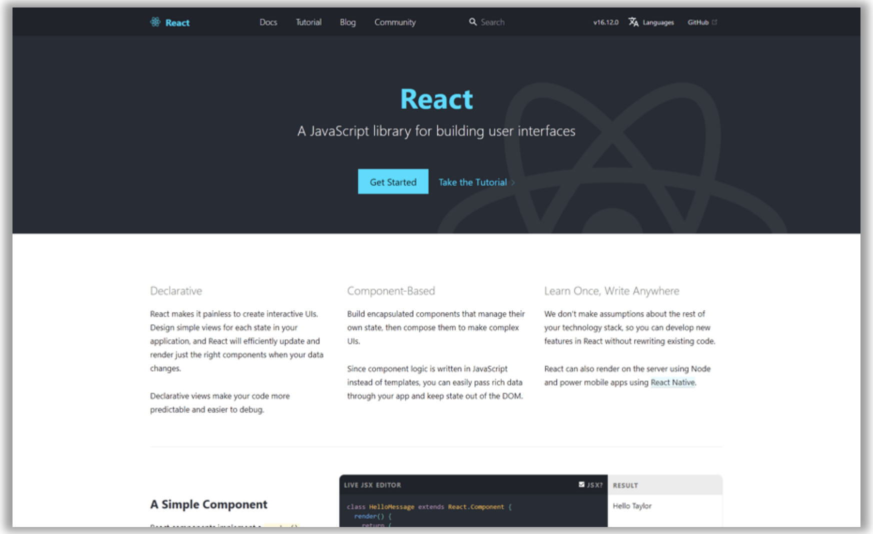
MDN Learning Area
Learn web development

HTML
Learn to structure web content with HTML

CSS
Learn to style content using CSS

JavaScript
Learn to run scripts in the browser

Accessibility
Learn to make the web accessible to all



React

A JavaScript library for building user interfaces

Declarative

React makes it painless to create interactive UIs. Design simple views for each state in your application, and React will efficiently update and render just the right components when your data changes.

Declarative views make your code more predictable and easier to debug.

Component-Based

Build encapsulated components that manage their own state, then compose them to make complex UIs.

Since component logic is written in JavaScript instead of templates, you can easily pass rich data through your app and keep state out of the DOM.

Learn Once, Write Anywhere

We don't make assumptions about the rest of your technology stack, so you can develop new features in React without rewriting existing code.

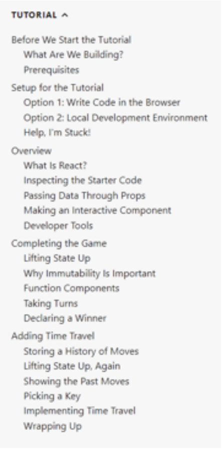
React can also render on the server using Node and power mobile apps using React Native.

A Simple Component

```
class HelloMessage extends React.Component {
  render() {
```

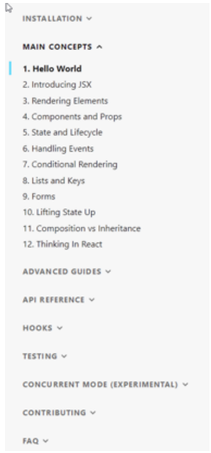
Hello Taylor

React Library
https://reactjs.org/



TUTORIAL ^

- Before We Start the Tutorial
  - What Are We Building?
  - Prerequisites
- Setup for the Tutorial
  - Option 1: Write Code in the Browser
  - Option 2: Local Development Environment
  - Help, I'm Stuck!
- Overview
  - What Is React?
  - Inspecting the Starter Code
  - Passing Data Through Props
  - Making an Interactive Component
  - Developer Tools
- Completing the Game
  - Lifting State Up
  - Why Immutability Is Important
  - Function Components
  - Taking Turns
  - Declaring a Winner
- Adding Time Travel
  - Storing a History of Moves
  - Lifting State Up, Again
  - Showing the Past Moves
  - Picking a Key
  - Implementing Time Travel
  - Wrapping Up



INSTALLATION ⌄

MAIN CONCEPTS ^

1. **Hello World**
2. Introducing JSX
3. Rendering Elements
4. Components and Props
5. State and Lifecycle
6. Handling Events
7. Conditional Rendering
8. Lists and Keys
9. Forms
10. Lifting State Up
11. Composition vs Inheritance
12. Thinking In React

ADVANCED GUIDES ⌄

API REFERENCE ⌄

HOOKS ⌄

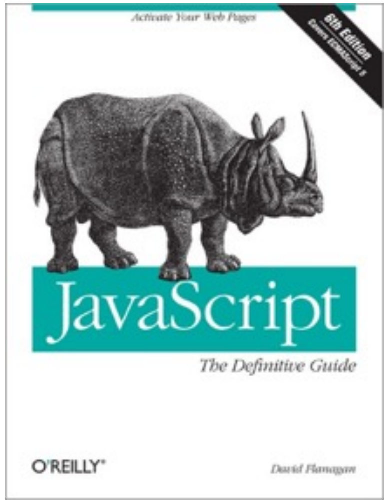
TESTING ⌄

CONCURRENT MODE (EXPERIMENTAL) ⌄

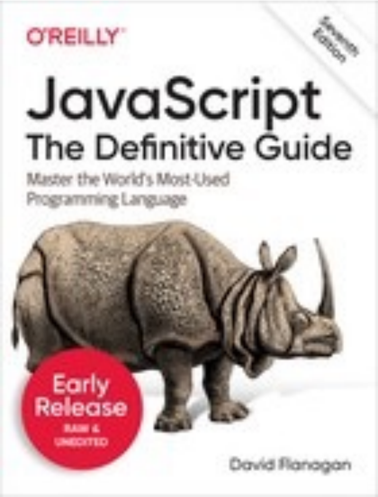
CONTRIBUTING ⌄

FAQ ⌄

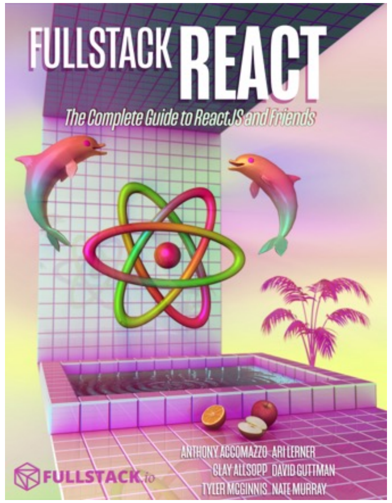
Resources (books)



JavaScript: The Definitive Guide, 6th Edition
By David Flanagan
ISBN 978-0596805524
Release Date: May 2011
(not very updated…)



JavaScript: The Definitive Guide, 7th Edition
By David Flanagan
ISBN 978-1491952023
Release Date: July 2020


FULLSTACK REACT
The Complete Guide to ReactJS and Friends
FULLSTACK.io
ANTHONY ACCOMAZZO · ARI LERNER
CLAY ALLSOPP · DAVID GUTTMAN
TYLER MCGINNIS · NATE MURRAY

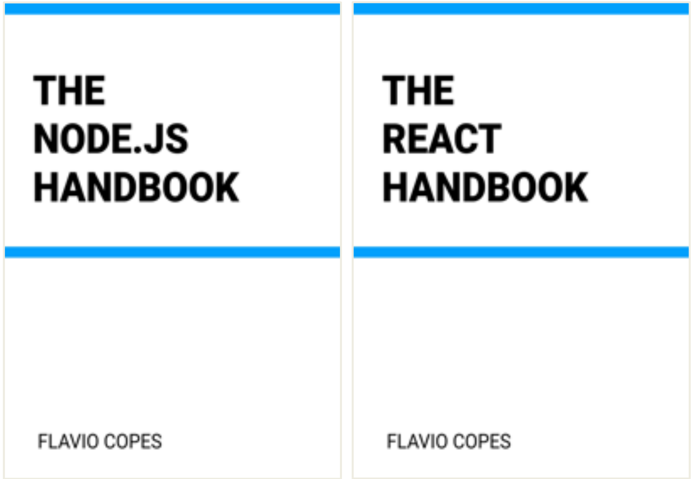
Fullstack React
By Anthony Accomazzo, Nate Murray, Ari Lerner, Clay Allsopp, David Guttman, and Tyler McGinnis
https://www.newline.co/fullstack-react
Release: r40 (January 2020)

…

… and many others

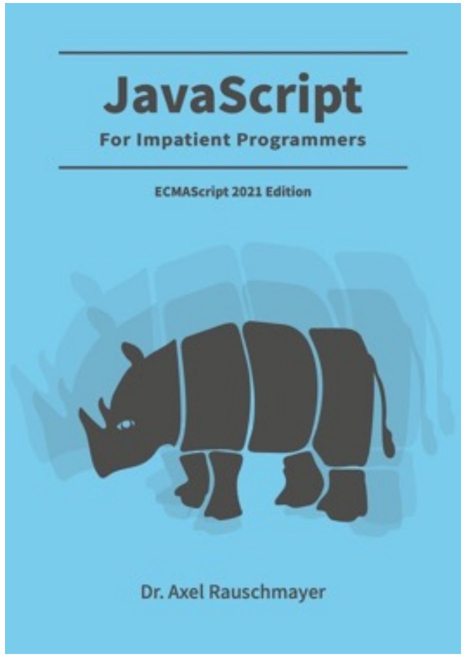
Resources (on-line books)

You Don't Know JS Yet (book series) - 2nd Edition
By Kyle Simpson (@getify)
https://github.com/getify/You-Dont-Know-JS

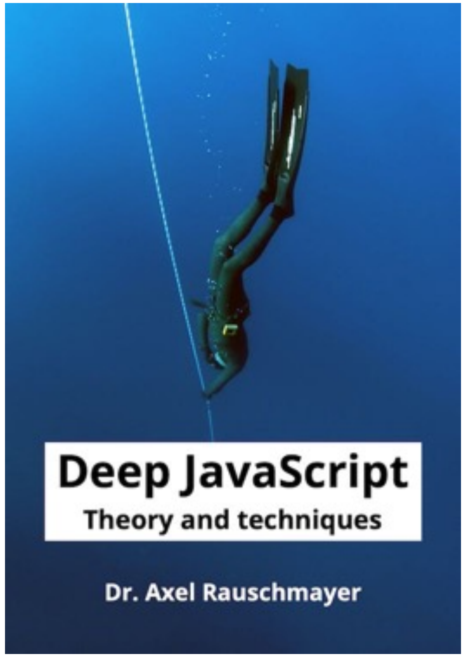


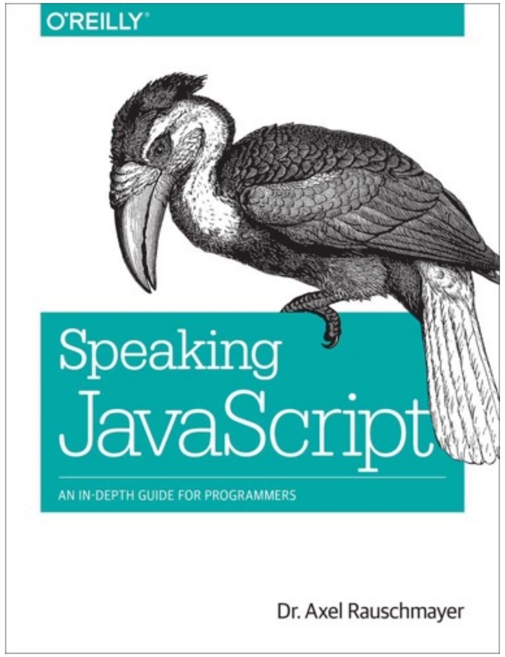
Flavio Copes Handbooks
https://flaviocopes.com/

Resources (on-line books)



+



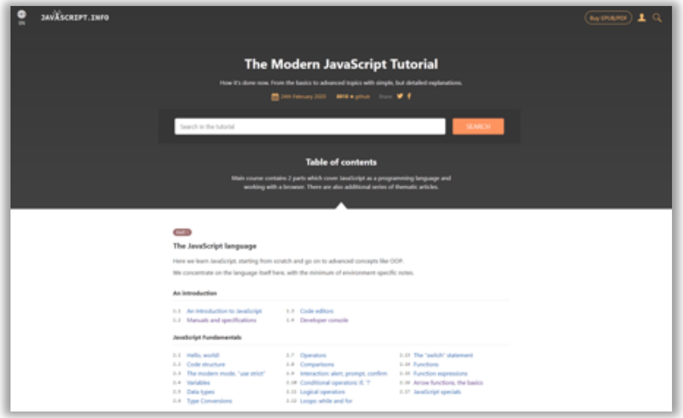


https://exploringjs.com/impatient-js/index.html

https://exploringjs.com/deep-js/index.html

http://speakingjs.com/

More resources…



The Modern JavaScript Tutorial

How it's done now. From the basics to advanced topics with simple, but detailed explanations.

24th February 2020 · github · Share

Search in the tutorial — SEARCH

Table of contents

Main course contains 2 parts which cover JavaScript as a programming language and working with a browser. There are also additional series of thematic articles.

PART 1

The JavaScript language

Here we learn JavaScript, starting from scratch and go on to advanced concepts like OOP.

We concentrate on the language itself here, with the minimum of environment-specific notes.

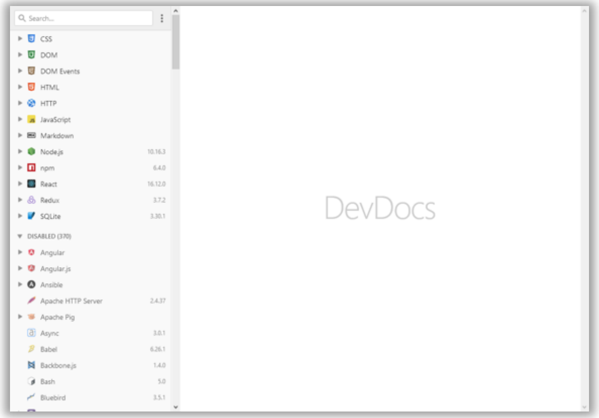
An introduction

1.1 An Introduction to JavaScript
1.2 Manuals and specifications
1.3 Code editors
1.4 Developer console

JavaScript Fundamentals

2.1 Hello, world!
2.2 Code structure
2.3 The modern mode, "use strict"
2.4 Variables
2.5 Data types
2.6 Type Conversions
2.7 Operators
2.8 Comparisons
2.9 Interaction: alert, prompt, confirm
2.10 Conditional operators: if, '?'
2.11 Logical operators
2.12 Loops: while and for
2.13 The "switch" statement
2.14 Functions
2.15 Function expressions
2.16 Arrow functions, the basics
2.17 JavaScript specials

The Modern JavaScript Tutorial
https://javascript.info/

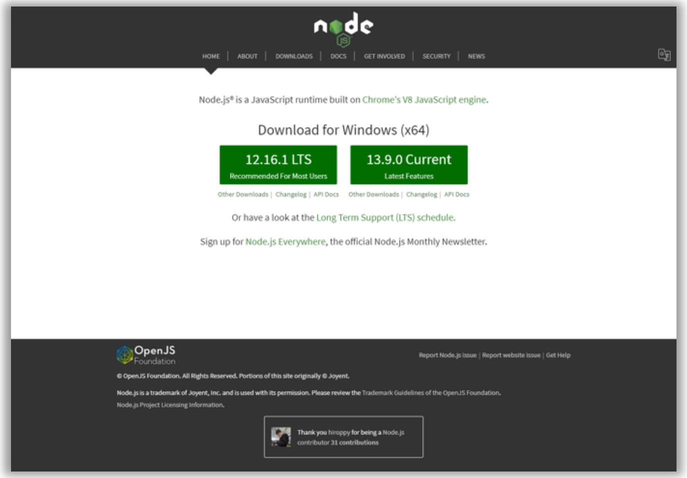


CSS	
DOM	
DOM Events	
HTML	
HTTP	
JavaScript	
Markdown	
Node.js	10.16.3
npm	6.4.0
React	16.12.0
Redux	3.7.2
SQLite	3.30.1
DISABLED (370)	
Angular	
Angular.js	
Ansible	
Apache HTTP Server	2.4.37
Apache Pig	
Async	3.0.1
Babel	6.26.1
Backbone.js	1.4.0
Bash	5.0
Bluebird	3.5.1

DevDocs: API Documentation Browser
https://devdocs.io/

… and many others

Tools


node
HOME | ABOUT | DOWNLOADS | DOCS | GET INVOLVED | SECURITY | NEWS
12.16.1 LTS
Recommended For Most Users
13.9.0 Current
Latest Features
Other Downloads | Changelog | API Docs
Other Downloads | Changelog | API Docs
OpenJS Foundation
Report Node.js issue | Report website issue | Get Help

Node.js® is a JavaScript runtime built on Chrome's V8 JavaScript engine.

Download for Windows (x64)

Or have a look at the Long Term Support (LTS) schedule.

Sign up for Node.js Everywhere, the official Node.js Monthly Newsletter.

© OpenJS Foundation. All Rights Reserved. Portions of this site originally © Joyent.

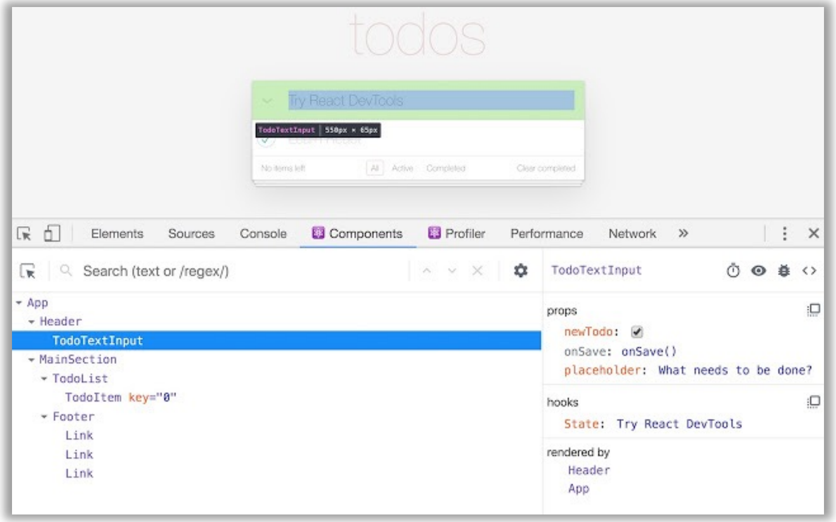
Node.js is a trademark of Joyent, Inc. and is used with its permission. Please review the Trademark Guidelines of the OpenJS Foundation.

Node.js Project Licensing Information.

Thank you hiroppy for being a Node.js contributor 31 contributions

Node.js runtime
Version 20.11 LTS
Always use LTS (long-term support) version
(i.e., even numbers only)
https://nodejs.org/en/

See next slides for installation instructions



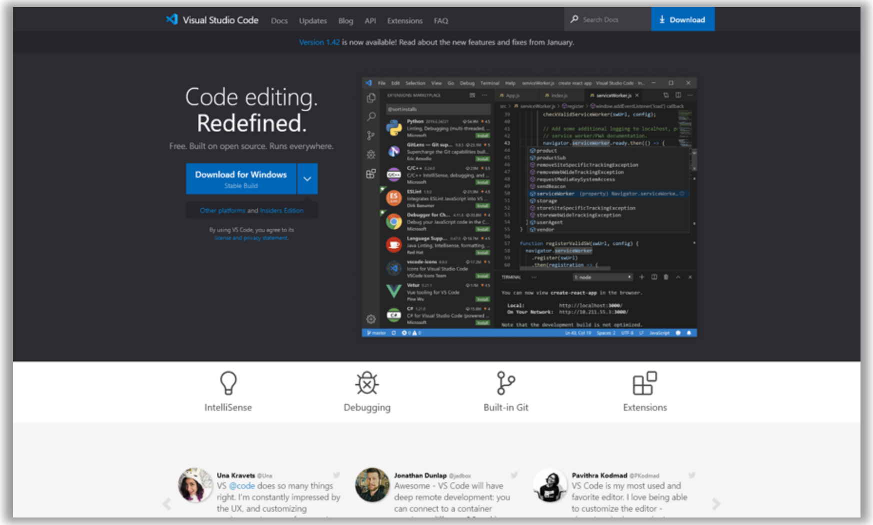
```
▾ App
  ▾ Header
      TodoTextInput
  ▾ MainSection
    ▾ TodoList
        TodoItem key="0"
    ▾ Footer
        Link
        Link
        Link
```

```
TodoTextInput
props
  newTodo: ☑
  onSave: onSave()
  placeholder: What needs to be done?
hooks
  State: Try React DevTools
rendered by
  Header
  App
```

React Developer Tools
Extension for Chrome and Firefox

Safari is NOT recommended in general for debugging

Programming Environment



Version 1.42 is now available! Read about the new features and fixes from January.

Code editing.
Redefined.

Free. Built on open source. Runs everywhere.

By using VS Code, you agree to its license and privacy statement.

Una Kravets @Una
VS @code does so many things right. I'm constantly impressed by the UX, and customizing

Jonathan Dunlap @jadbox
Awesome - VS Code will have deep remote development: you can connect to a container

Pavithra Kodmad @PKodmad
VS Code is my most used and favorite editor. I love being able to customize the editor -

Visual Studio Code
https://code.visualstudio.com/

See next slides for installation instructions

Installation Instructions

- Linux (recommended)
  - Node.js: DO NOT install the version coming with the Linux distribution , use:

```
# https://github.com/nodesource/distributions: e.g., for ubuntu
curl -fsSL https://deb.nodesource.com/setup_lts.x | sudo -E bash - && sudo apt-get install -y nodejs
```

  - VSCode: DO NOT install the version coming with the Linux distribution, download the .deb package from https://code.visualstudio.com/ and install it
- MacOS
  - Node.js: Use the LTS package provided by https://nodejs.org/en/
  - VSCode: Use the MacOS package provided by https://code.visualstudio.com/

NB: MacOS is NOT Linux, the final project will be tested on Linux, so beware of upper/lowercase letters in file names (MacOS is not case sensitive, Linux is)

Installation Instructions

- Windows (follow instructions very carefully!)
  - **Use WSL2** and work with all the files, including projects and node modules, always in the filesystem of the Linux subsystem

1. Install WSL2 if not yet done: https://learn.microsoft.com/en-us/windows/wsl/install
2. Then follow Linux instruction to install Node.js from the WSL2 Linux terminal: https://github.com/nodesource/distributions

```
sudo apt install curl  # install curl if not yet present
curl -fsSL https://deb.nodesource.com/setup_lts.x | sudo -E bash - && sudo apt-get install -y nodejs
```

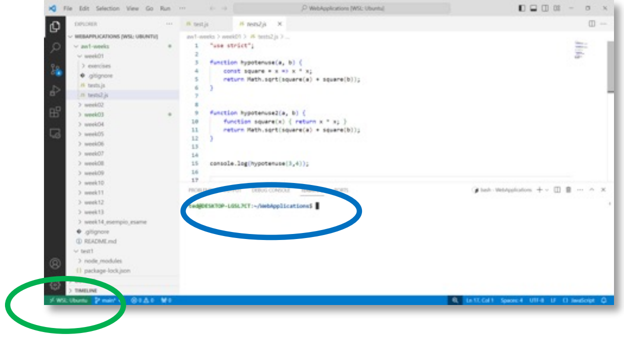
- **DO NOT INSTALL** Node.js directly in Windows with the Windows installation package. Uninstall it if necessary

Installation Instructions

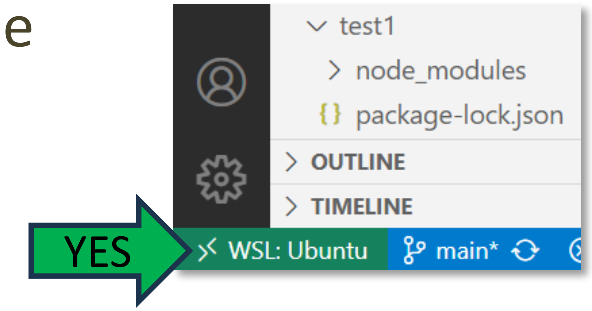
- Windows (VSCode)
  1. Install VSCode in Windows with the Windows installation package (NOT Linux).
  2. Then, open it from the Linux (not Windows) shell terminal by launching WSL and then typing “**code**”.
     It will prompt to install the WSL extension, install it.
  3. Again, remember to always work in the filesystem of the Linux subsystem. If unsure, open VSCode from the Linux terminal with the commands:

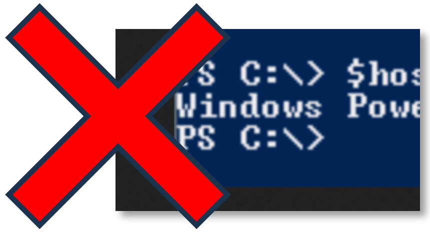
```
cd myFolderName
code .
```

Installation Instructions

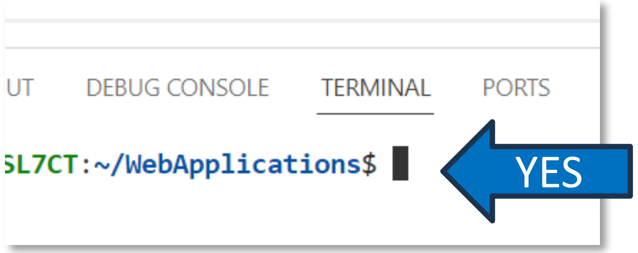


- Windows (working in VSCode)
  - Always install modules (npm command) in Linux when required in lectures and labs!
    (All files are accessible from Windows by entering `\\wsl$` path in Windows file explorer)
  - Always double-check you are working in WSL, by looking at the bottom left corner in VSCode
  - Always check that the terminal is Linux and <u>NOT</u> Windows PowerShell (PS)





```
.$ C:\> $hos
Windows Powe
PS C:\>
```



```
SL7CT:~/WebApplications$ █
```

NB: The exam will be tested under Linux. Beware: <u>Windows is NOT case sensitive</u>

License