



JavaScript

Web Engineering

Dierk König

Technology Overview

HTML

Validator

CSS

Web MVC

Unit Testing

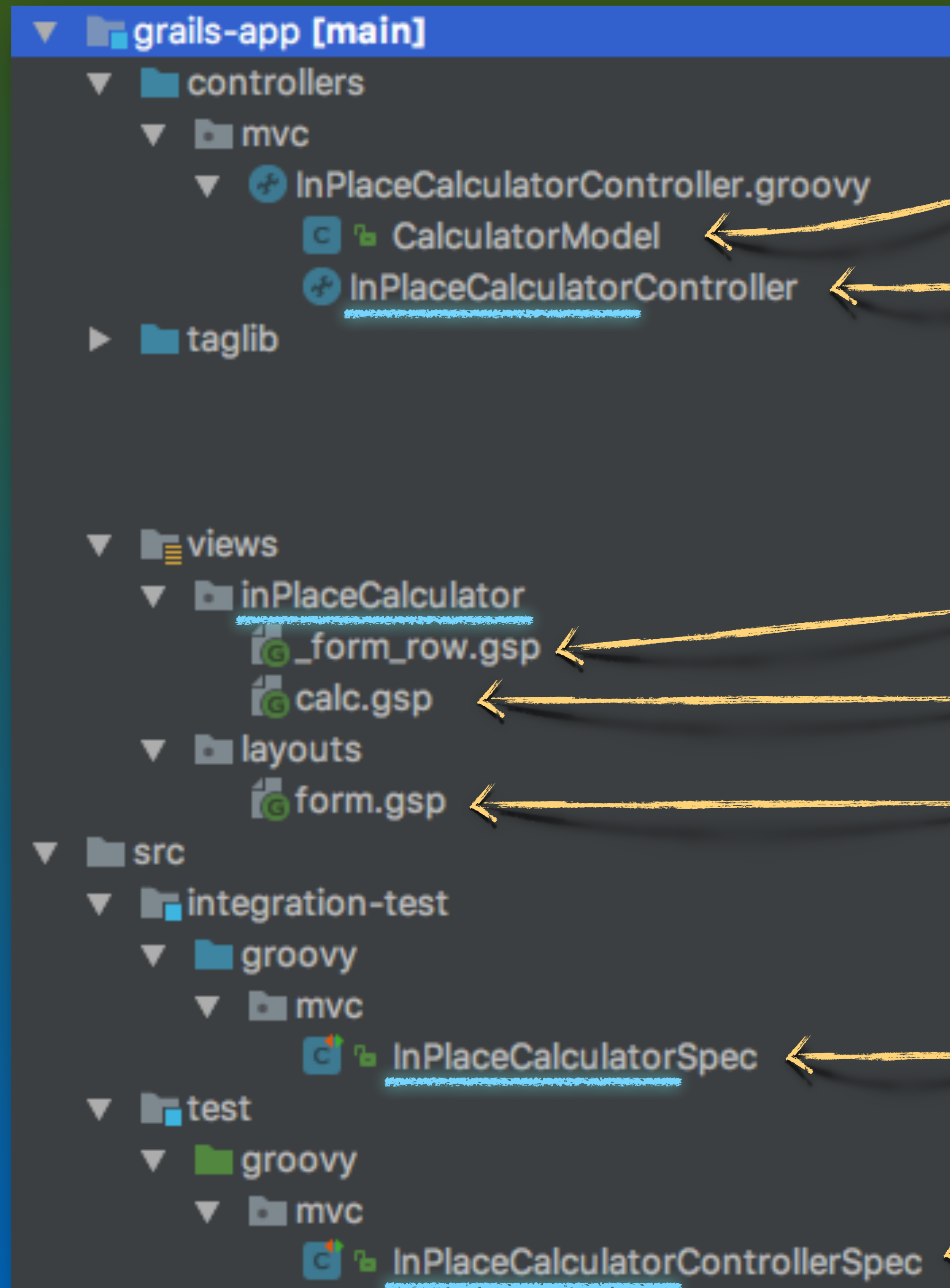
Server Pages

UI Testing

Javascript

UI Testing

Overall Structure



model
controller

template
server page
layout

UI test
unit test

The story so far

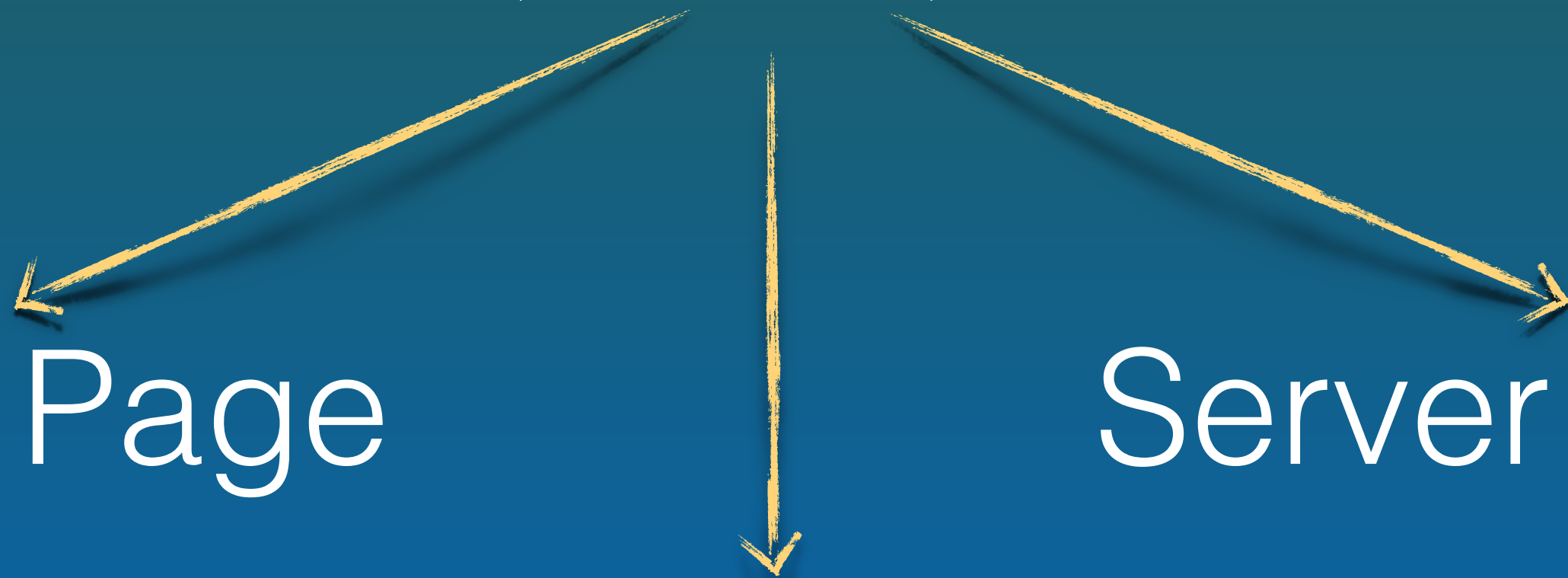
Static Pages - HTML, CSS

MVC - Model, View, Controller

Static Page

Server Page

Dynamic Page



Request - Response

*code as
methods*

```
class MyController  
  def myAction(model)
```

strings

localhost:8080/tempConverter

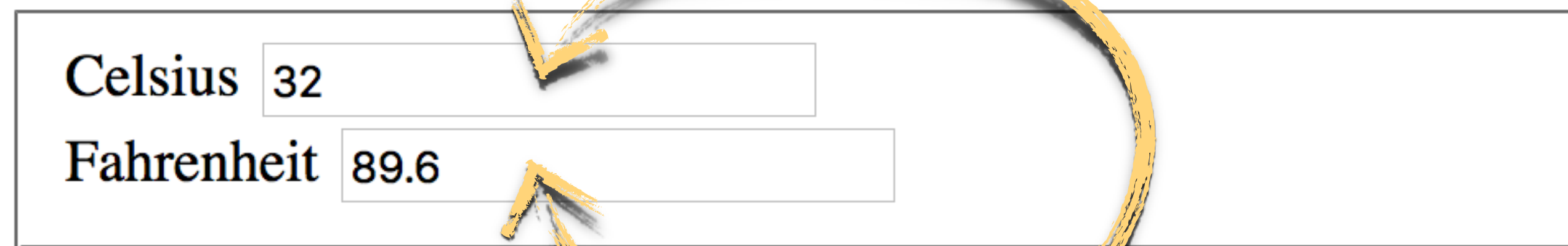
DuckDuckGo Yahoo! Google Maps YouTube Wikipedia News >> +

Temperature Converter

Celsius	<input type="text" value="0.0"/>	32.0
Fahrenheit	<input type="text" value="0.0"/>	-17.8

Direct Manipulation

Temperature Converter with JavaScript



Celsius	<input type="text" value="32"/>
Fahrenheit	<input type="text" value="89.6"/>

code as strings == scripting

JavaScript



Code as string

HTML attribute value

Text content of `<script>` element

External .js file

Event Attributes



onClick,
onMouseOver,
onChange,
onInput,
...

JS Document

```
document.write(html);
```

```
document.getElementById(id);
```

```
document.querySelector(selector);
```

```
...
```

JS Element

element.id

element.value = *newValue*;

element.innerHTML = *newContent*;

element.classList.add(*newStyle*);

...

JS Function Declaration

keyword function name parameter names

function times(x, y) {
 return a * b;
}

value returned

The diagram illustrates the components of a JavaScript function declaration. The code 'function times(x, y) { return a * b; }' is shown. Handwritten yellow arrows point from labels to specific parts of the code: 'keyword' points to 'function', 'function name' points to 'times', 'parameter names' points to '(x, y)', and 'value returned' points to 'a * b'.

JS Lambda Declaration

keyword → `const` *name* → `times` *parameter names* → `(x, y)` `=>` {
 `return a * b;`
}

value returned → `a * b`

JS Lambda Declaration

keyword name parameter names

→ const times = (x, y) =>

a * b;

expression
value returned

JS Lambda event handler

```
evt => handleEvent(evt)
```

Engineering Aspects

Where to put JS code:

in-line only for one-liners

in-page for local functions

.js files for cross-page sharing,
unit testing, linting, tool support, ...

JS Topics not covered

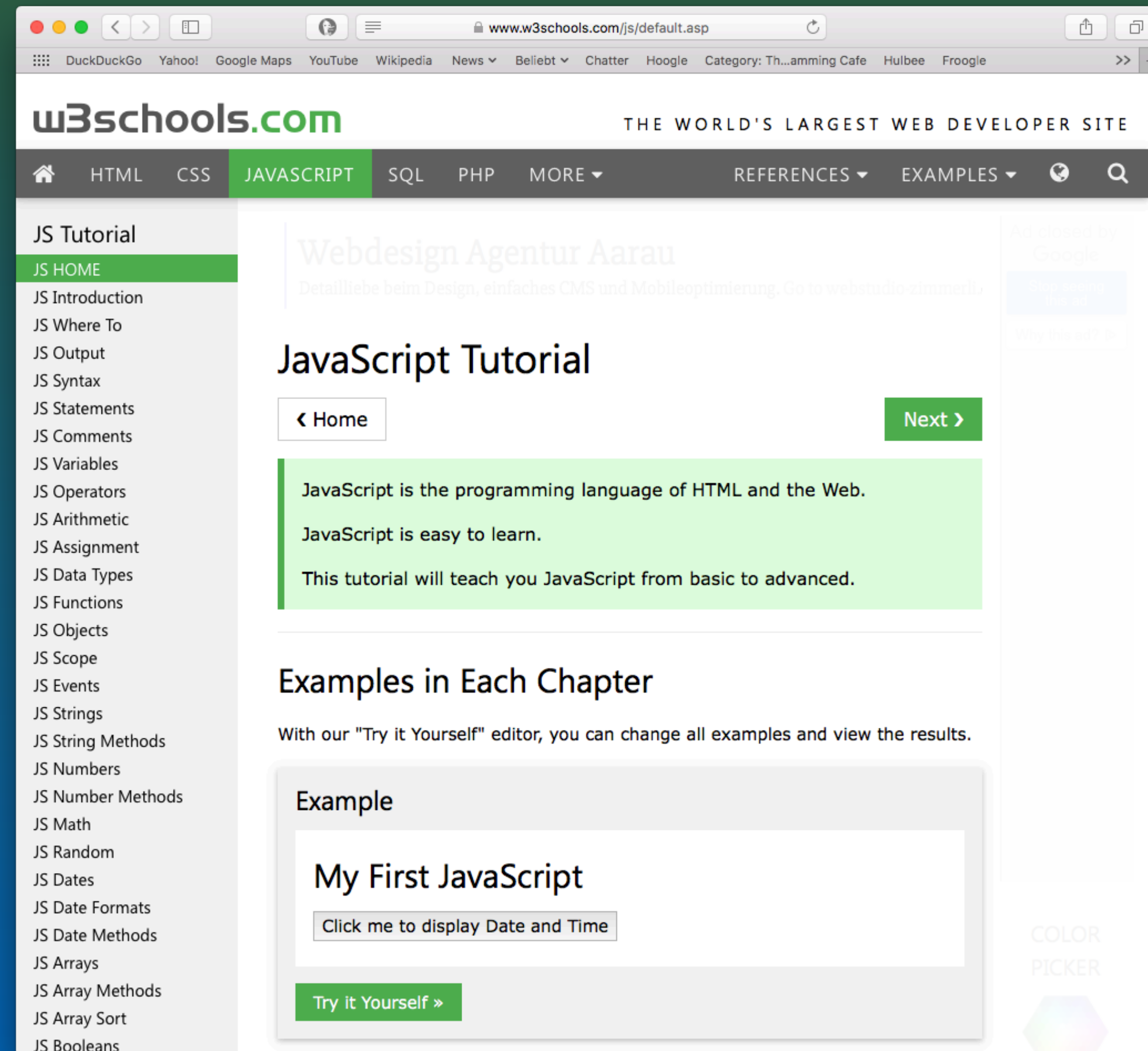
classes, objects, types, literal decl.

scoping, prototype inheritance

function expressions, self-invocation

hoisting, linting, unit testing, ...

www.w3schools.com/js



Mozilla Developer Network

https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/JavaScript_basics