

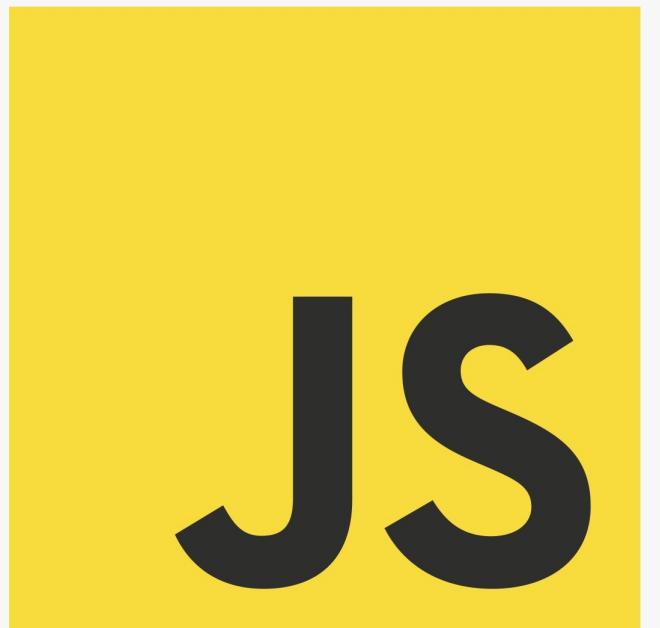
TABLE OF CONTENTS: THEORY LECTURES (CLICK THE TITLES)

- 1 A Brief Introduction to JavaScript →
- 2 JavaScript Versions: ES5, ES6 / ES2015 and ES6+ →
- 3 How our code is executed: JavaScript parsers and engines →
- 4 Execution contexts and the execution stack →
- 5 Execution contexts in detail: creation and execution phases and hoisting →
- 6 Scoping and the scope chain →
- 7 The 'this' keyword →
- 8 The DOM and DOM Manipulation →
- 9 Events and event handling: rolling the dice →
- 10 Everything is an object: Inheritance and the Prototype Chain →
- 11 Closures →
- 12 Event delegation →
- 13 Understanding Asynchronous JavaScript: The Event Loop →
- 14 From Callback Hell to Promises →
- 15 An Overview of Modern JavaScript →

SECTION 1 – INTRODUCTION

WHAT IS JAVASCRIPT?

- JavaScript is a lightweight, cross-platform, object-oriented computer programming language 😅
- JavaScript is one of the three core technologies of web development
- Today, JavaScript can be used in different places:
 - **Client-side: JavaScript was traditionally only used in the browser**
 - Server-side: Thanks to node.js, we can use JavaScript on the server as well
- Javascript is what made modern web development possible:
 - Dynamic effects and interactivity
 - Modern web applications that we can interact with
- **Frameworks/libraries like React and Angular are 100% based on JavaScript: you need to master JavaScript in order to use them!**



02

JavaScript Language Basics

03

How JavaScript Works Behind the Scenes

04

JavaScript in the Browser: DOM Manipulation and Events

05

Advanced JavaScript: Objects and Functions

06

Putting It All Together: The Budget App Project

07

Next Generation JavaScript: Intro to ES6 / ES2015

08

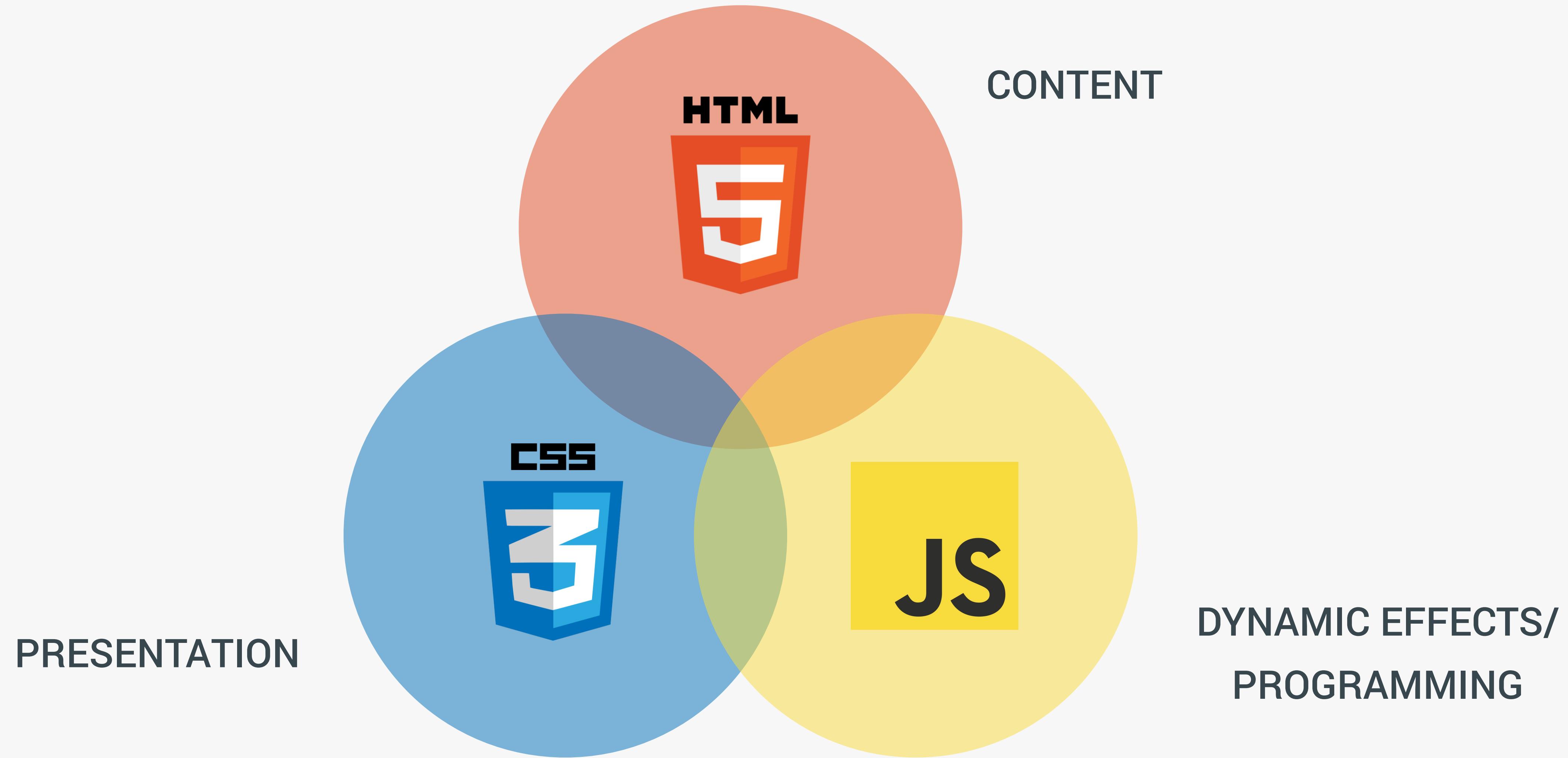
Asynchronous JavaScript: Promises, Async/Await and AJAX

09

Modern JavaScript: Using ES6, NPM, Babel and Webpack

SECTION 2 – JS LANGUAGE BASICS

THE ROLE OF JAVASCRIPT IN WEB DEVELOPMENT



NOUNS, ADJECTIVES AND VERBS



CONTENT

NOUNS

< p > </ p >

means “paragraph”

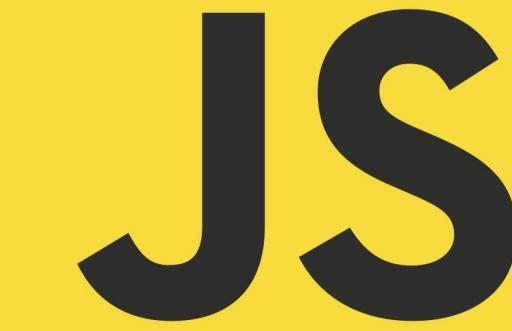


PRESENTATION

ADJECTIVES

p {color: red;}

means “the paragraph
text is red”



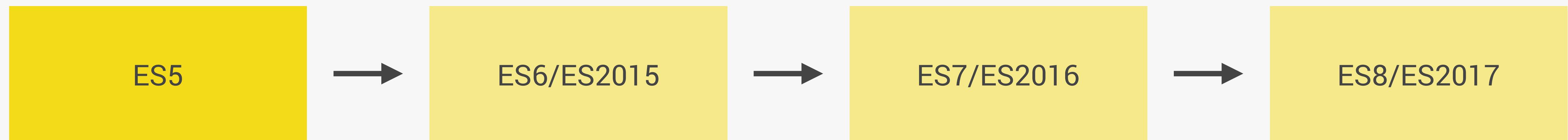
DYNAMIC EFFECTS/
PROGRAMMING

VERBS

p.hide();

means “hide the
paragraph”

JAVASCRIPT VERSIONS... (MORE ABOUT THIS LATER)



PRIMITIVE JAVASCRIPT DATA TYPES

1. **Number:** Floating point numbers, for decimals and integers
2. **String:** Sequence of characters, used for text
3. **Boolean:** Logical data type that can only be true or false
4. **Undefined:** Data type of a variable that does not have a value yet
5. **Null:** Also means 'non-existent'

JavaScript has dynamic typing: data types are
automatically assigned to variables

BASIC BOOLEAN LOGIC: NOT, AND & OR

var A

		AND	TRUE	FALSE
		AND	TRUE	FALSE
var B	TRUE	TRUE	FALSE	
	FALSE	FALSE	FALSE	

- AND (`&&`) => true if **ALL** are true
- OR (`||`) => true if **ONE** is true
- NOT (`!`) => inverts true/false value

var A

		OR	TRUE	FALSE
		OR	TRUE	TRUE
var B	TRUE	TRUE	TRUE	
	FALSE	TRUE	FALSE	

```
var age = 16;
```

```
age >= 20;      // => false  
age < 30;      // => true  
!(age < 30);  // => false
```

```
age >= 20 && age < 30; // =>  
age >= 20 || age < 30; // =>
```

A (VERY) SHORT HISTORY OF JAVASCRIPT

- **1996:** Changed from LiveScript to JavaScript to attract Java developers. **JavaScript has almost nothing to do with Java** 🤪
- **1997:** ES1 (ECMAScript 1) became the first version of the JavaScript language standard:
 - ECMAScript: The language standard;
 - JavaScript: The language in practice.
- **2009:** ES5 (ECMAScript 5) was released with lots of new features.
- **2015:** ES6/ES2015 (ECMAScript 2015) was released: **the biggest update to the language ever!**
- **2015:** Changed to an **annual release cycle** 🙏
- **2016/2017/2018/2019/...:** Release of ES2016/ES2017/ES2018/ES2019/...

JAVASCRIPT TODAY: WHICH VERSION TO USE?



- Fully supported in all browsers;
- Ready to be used today



- Well supported in all **modern** browsers
- No support in older browsers;
- Can use **most features** in production with transpiling and polyfilling (converting to ES5)



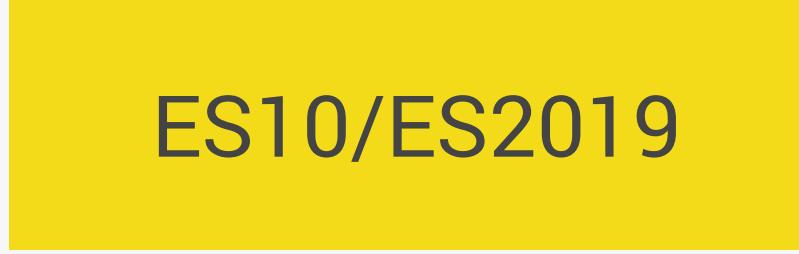
- Well supported in all **modern** browsers
- No support in older browsers;
- Can use **most features** in production with transpiling and polyfilling (converting to ES5)



- Future versions, together called ESNext;
- Some features supported in modern browsers;
- Can already use **some features** in production with transpiling and polyfilling



- Future versions, together called ESNext;
- Some features supported in modern browsers;
- Can already use **some features** in production with transpiling and polyfilling


Feature name	Current browser	Compliers/polyfills															Desktop browsers											Servers/numerics																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																		
Tracer	babel 6	babel 7	es6-shim	es7-shim	es8-shim	es9-shim	es10-shim	es11-shim	es12-shim	es13-shim	es14-shim	es15-shim	es16-shim	es17-shim	es18-shim	es19-shim	es20-shim	es21-shim	es22-shim	es23-shim	es24-shim	es25-shim	es26-shim	es27-shim	es28-shim	es29-shim	es30-shim	es31-shim	es32-shim	es33-shim	es34-shim	es35-shim	es36-shim	es37-shim	es38-shim	es39-shim	es40-shim	es41-shim	es42-shim	es43-shim	es44-shim	es45-shim	es46-shim	es47-shim	es48-shim	es49-shim	es50-shim	es51-shim	es52-shim	es53-shim	es54-shim	es55-shim	es56-shim	es57-shim	es58-shim	es59-shim	es60-shim	es61-shim	es62-shim	es63-shim	es64-shim	es65-shim	es66-shim	es67-shim	es68-shim	es69-shim	es70-shim	es71-shim	es72-shim	es73-shim	es74-shim	es75-shim	es76-shim	es77-shim	es78-shim	es79-shim	es80-shim	es81-shim	es82-shim	es83-shim	es84-shim	es85-shim	es86-shim	es87-shim	es88-shim	es89-shim	es90-shim	es91-shim	es92-shim	es93-shim	es94-shim	es95-shim	es96-shim	es97-shim	es98-shim	es99-shim	es100-shim	es101-shim	es102-shim	es103-shim	es104-shim	es105-shim	es106-shim	es107-shim	es108-shim	es109-shim	es110-shim	es111-shim	es112-shim	es113-shim	es114-shim	es115-shim	es116-shim	es117-shim	es118-shim	es119-shim	es120-shim	es121-shim	es122-shim	es123-shim	es124-shim	es125-shim	es126-shim	es127-shim	es128-shim	es129-shim	es130-shim	es131-shim	es132-shim	es133-shim	es134-shim	es135-shim	es136-shim	es137-shim	es138-shim	es139-shim	es140-shim	es141-shim	es142-shim	es143-shim	es144-shim	es145-shim	es146-shim	es147-shim	es148-shim	es149-shim	es150-shim	es151-shim	es152-shim	es153-shim	es154-shim	es155-shim	es156-shim	es157-shim	es158-shim	es159-shim	es160-shim	es161-shim	es162-shim	es163-shim	es164-shim	es165-shim	es166-shim	es167-shim	es168-shim	es169-shim	es170-shim	es171-shim	es172-shim	es173-shim	es174-shim	es175-shim	es176-shim	es177-shim	es178-shim	es179-shim	es180-shim	es181-shim	es182-shim	es183-shim	es184-shim	es185-shim	es186-shim	es187-shim	es188-shim	es189-shim	es190-shim	es191-shim	es192-shim	es193-shim	es194-shim	es195-shim	es196-shim	es197-shim	es198-shim	es199-shim	es200-shim	es201-shim	es202-shim	es203-shim	es204-shim	es205-shim	es206-shim	es207-shim	es208-shim	es209-shim	es210-shim	es211-shim	es212-shim	es213-shim	es214-shim	es215-shim	es216-shim	es217-shim	es218-shim	es219-shim	es220-shim	es221-shim	es222-shim	es223-shim	es224-shim	es225-shim	es226-shim	es227-shim	es228-shim	es229-shim	es230-shim	es231-shim	es232-shim	es233-shim	es234-shim	es235-shim	es236-shim	es237-shim	es238-shim	es239-shim	es240-shim	es241-shim	es242-shim	es243-shim	es244-shim	es245-shim	es246-shim	es247-shim	es248-shim	es249-shim	es250-shim	es251-shim	es252-shim	es253-shim	es254-shim	es255-shim	es256-shim	es257-shim	es258-shim	es259-shim	es260-shim	es261-shim	es262-shim	es263-shim	es264-shim	es265-shim	es266-shim	es267-shim	es268-shim	es269-shim	es270-shim	es271-shim	es272-shim	es273-shim	es274-shim	es275-shim	es276-shim	es277-shim	es278-shim	es279-shim	es280-shim	es281-shim	es282-shim	es283-shim	es284-shim	es285-shim	es286-shim	es287-shim	es288-shim	es289-shim	es290-shim	es291-shim	es292-shim	es293-shim	es294-shim	es295-shim	es296-shim	es297-shim	es298-shim	es299-shim	es300-shim	es301-shim	es302-shim	es303-shim	es304-shim	es305-shim	es306-shim	es307-shim	es308-shim	es309-shim	es310-shim	es311-shim	es312-shim	es313-shim	es314-shim	es315-shim	es316-shim	es317-shim	es318-shim	es319-shim	es320-shim	es321-shim	es322-shim	es323-shim	es324-shim	es325-shim	es326-shim	es327-shim	es328-shim	es329-shim	es330-shim	es331-shim	es332-shim	es333-shim	es334-shim	es335-shim	es336-shim	es337-shim	es338-shim	es339-shim	es340-shim	es341-shim	es342-shim	es343-shim	es344-shim	es345-shim	es346-shim	es347-shim	es348-shim	es349-shim	es350-shim	es351-shim	es352-shim	es353-shim	es354-shim	es355-shim	es356-shim	es357-shim	es358-shim	es359-shim	es360-shim	es361-shim	es362-shim	es363-shim	es364-shim	es365-shim	es366-shim	es367-shim	es368-shim	es369-shim	es370-shim	es371-shim	es372-shim	es373-shim	es374-shim	es375-shim	es376-shim	es377-shim	es378-shim	es379-shim	es380-shim	es381-shim	es382-shim	es383-shim	es384-shim	es385-shim	es386-shim	es387-shim	es388-shim	es389-shim	es390-shim	es391-shim	es392-shim	es393-shim	es394-shim	es395-shim	es396-shim	es397-shim	es398-shim	es399-shim	es400-shim	es401-shim	es402-shim	es403-shim	es404-shim	es405-shim	es406-shim	es407-shim	es408-shim	es409-shim	es410-shim	es411-shim	es412-shim	es413-shim	es414-shim	es415-shim	es416-shim	es417-shim	es418-shim	es419-shim	es420-shim	es421-shim	es422-shim	es423-shim	es424-shim	es425-shim	es426-shim	es427-shim	es428-shim	es429-shim	es430-shim	es431-shim	es432-shim	es433-shim	es434-shim	es435-shim	es436-shim	es437-shim	es438-shim	es439-shim	es440-shim	es441-shim	es442-shim	es443-shim	es444-shim	es445-shim	es446-shim	es447-shim	es448-shim	es449-shim	es450-shim	es451-shim	es452-shim	es453-shim	es454-shim	es455-shim	es456-shim	es457-shim	es458-shim	es459-shim	es460-shim	es461-shim	es462-shim	es463-shim	es464-shim	es465-shim	es466-shim	es467-shim	es468-shim	es469-shim	es470-shim	es471-shim	es472-shim	es473-shim	es474-shim	es475-shim	es476-shim	es477-shim	es478-shim	es479-shim	es480-shim	es481-shim	es482-shim	es483-shim	es484-shim	es485-shim	es486-shim	es487-shim	es488-shim	es489-shim	es490-shim	es491-shim	es492-shim	es493-shim	es494-shim	es495-shim	es496-shim	es497-shim	es498-shim	es499-shim	es500-shim	es501-shim	es502-shim	es503-shim	es504-shim	es505-shim	es506-shim	es507-shim	es508-shim	es509-shim	es510-shim	es511-shim	es512-shim	es513-shim	es514-shim	es515-shim	es516-shim	es517-shim	es518-shim	es519-shim	es520-shim	es521-shim	es522-shim	es523-shim	es524-shim	es525-shim	es526-shim	es527-shim	es528-shim	es529-shim	es530-shim	es531-shim	es532-shim	es533-shim	es534-shim	es535-shim	es536-shim	es537-shim	es538-shim	es539-shim	es540-shim	es541-shim	es542-shim	es543-shim	es544-shim	es545-shim	es546-shim	es547-shim	es548-shim	es549-shim	es550-shim	es551-shim	es552-shim	es553-shim	es554-shim	es555-shim	es556-shim	es557-shim	es558-shim	es559-shim	es560-shim	es561-shim	es562-shim	es563-shim	es564-shim	es565-shim	es566-shim	es567-shim	es568-shim	es569-shim	es570-shim	es571-shim	es572-shim	es573-shim	es574-shim	es575-shim	es576-shim	es577-shim	es578-shim	es579-shim	es580-shim	es581-shim	es582-shim	es583-shim	es584-shim	es585-shim	es586-shim	es587-shim	es588-shim	es589-shim	es590-shim	es591-shim	es592-shim	es593-shim	es594-shim	es595-shim	es596-shim	es597-shim	es598-shim	es599-shim	es600-shim	es601-shim	es602-shim	es603-shim	es604-shim	es605-shim	es606-shim	es607-shim	es608-shim	es609-shim	es610-shim	es611-shim	es612-shim	es613-shim	es614-shim	es615-shim	es616-shim	es617-shim	es618-shim	es619-shim	es620-shim	es621-shim	es622-shim	es623-shim	es624-shim	es625-shim	es626-shim	es627-shim	es628-shim	es629-shim	es630-shim	es631-shim	es632-shim	es633-shim	es634-shim	es635-shim	es636-shim	es637-shim	es638-shim	es639-shim	es640-shim	es64

WE USE ES5 AND ES6 IN THIS COURSE: WHY?

ES5

- JavaScript fundamentals
- How the language works
- DOM manipulation project
- Advanced language features
- Huge real project

ES6+

- ES6/ES2015 introduction
- Asynchronous JavaScript
- AJAX and API calls
- Modern dev setups (Webpack and Babel)
- Huge real project

- You will have to understand ES5 today and in the future;
- Many tutorials and code you find online today are still in ES5;
- When working on older codebases, these will be written in ES5;
- It's better and easier to learn the fundamentals in ES5, and then update to ES6+.

SECTION 3 – HOW JAVASCRIPT WORKS BEHIND THE SCENES



JONAS.IO
SCHMEDTMANN

THE COMPLETE JAVASCRIPT COURSE

BUILD REAL-WORLD PROJECTS!



@JONASSCHMEDTMAN

```
$classes = array( 'zoom' );  
  
if ( $loop == 0 || $loop % $columns == 0 )  
    $classes[] = 'first';  
  
if ( ( $loop + 1 ) % $columns == 0 )  
    $classes[] = 'last';  
  
$image_link = wp_get_attachment_url( $attachment_id );  
  
// + $image_link )
```

SECTION
HOW JAVASCRIPT WORKS BEHIND THE
SCENES

LECTURE

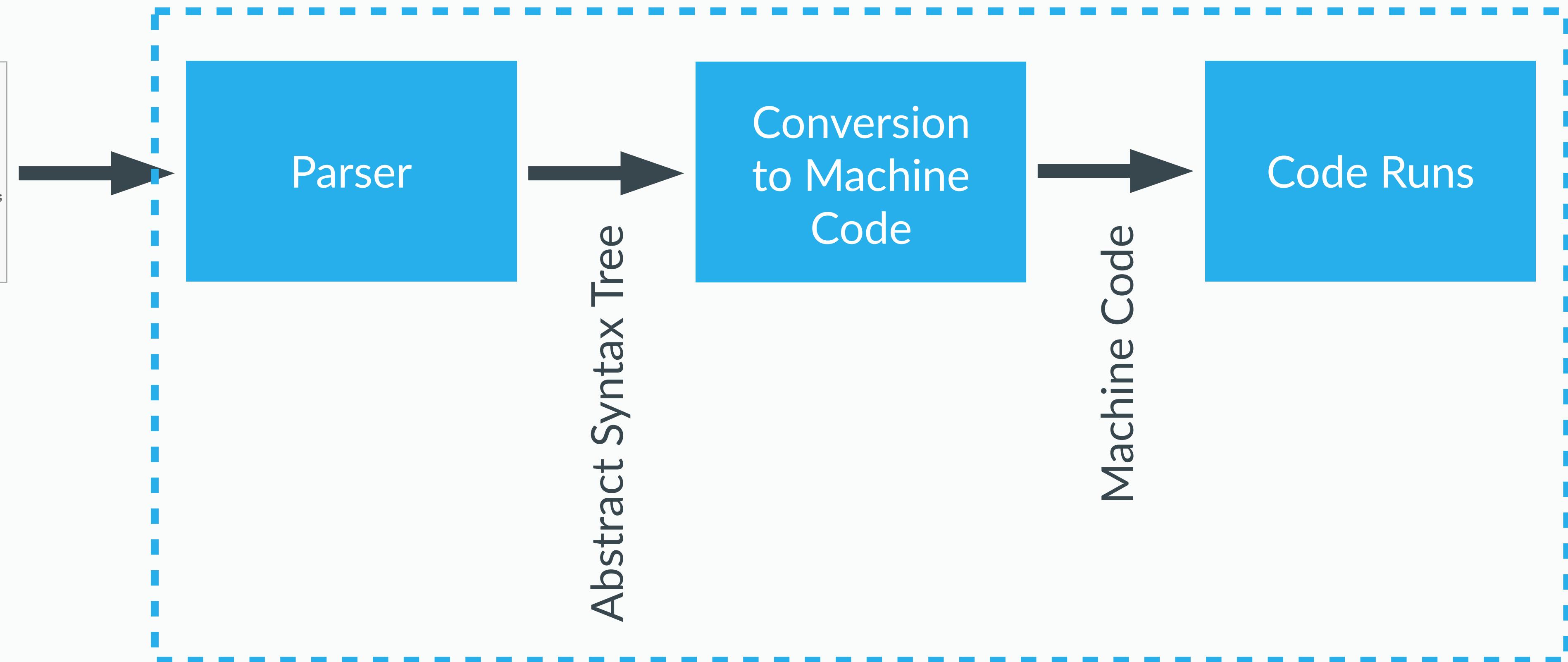
HOW OUR CODE IS EXECUTED:
JAVASCRIPT PARSERS AND ENGINES

WHAT HAPPENS TO OUR CODE?

OUR CODE

```
function calculateAge(yearOfBirth) {  
    return 2016 - yearOfBirth;  
}  
  
var johnsAge = calculateAge(1990);  
  
function yearsUntilRetirement(name, yearOfBirth) {  
    var age = calculateAge(yearOfBirth);  
    var retirement = 65 - age;  
    if (retirement >= 0) {  
        console.log(name + ' retires in ' + retirement + ' years.');//  
    } else {  
        console.log(name + ' is already retired.');//  
    }  
}  
  
yearsUntilRetirement('John', 1990);
```

JAVASCRIPT ENGINE





JONAS.IO
SCHMEDTMANN

THE COMPLETE JAVASCRIPT COURSE

BUILD REAL-WORLD PROJECTS!



@JONASSCHMEDTMAN

SECTION

HOW JAVASCRIPT WORKS BEHIND THE
SCENES

LECTURE

EXECUTION CONTEXTS AND THE
EXECUTION STACK

EXECUTION CONTEXTS

Execution Context

(A box, a container, or a wrapper which stores variables and in which a piece of our code is evaluated and executed)

```
function calculateAge(yearOfBirth) {
  return 2016 - yearOfBirth;
}

var johnsAge = calculateAge(1990);

function yearsUntilRetirement(name, yearOfBirth) {
  var age = calculateAge(yearOfBirth);
  var retirement = 65 - age;
  if (retirement >= 0) {
    console.log(name + ' retires in ' + retirement + ' years.');
  } else {
    console.log(name + ' is already retired.');
  }
}
yearsUntilRetirement('John', 1990);
```

THE DEFAULT

Global Execution Context

- Code that is **not inside any function**
- Associated with the **global object**
- In the browser, that's the **window object**

```
lastName === window.lastName
// true
```

```
var name = 'John'; ←  
  
function first() {  
    var a = 'Hello!';  
    second();  
    var x = a + name;  
}  
  
function second() {  
    var b = 'Hi!';  
    third();  
    var z = b + name;  
}  
  
function third() {  
    var c = 'Hey!';  
    var z = c + name;  
}  
  
first();
```

Execution Context

third()

Execution Context

second()

Execution Context

first()

Global Execution
Context

EXECUTION STACK



JONAS.IO
SCHMEDTMANN

THE COMPLETE JAVASCRIPT COURSE

BUILD REAL-WORLD PROJECTS!



@JONASSCHMEDTMAN

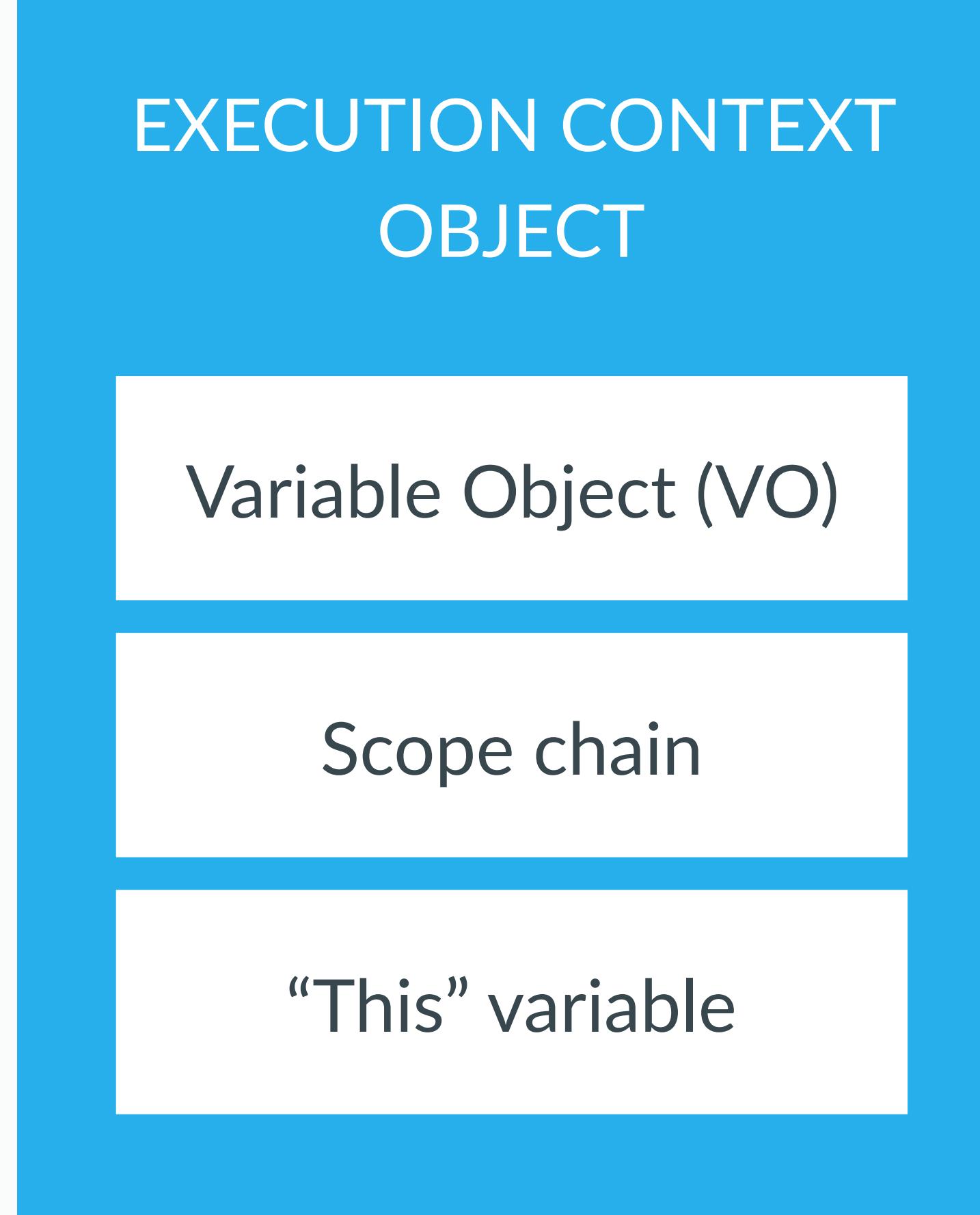
SECTION

HOW JAVASCRIPT WORKS BEHIND THE
SCENES

LECTURE

EXECUTION CONTEXTS IN DETAIL:
CREATION AND EXECUTION PHASES
AND HOISTING

THE EXECUTION CONTEXT IN DETAIL



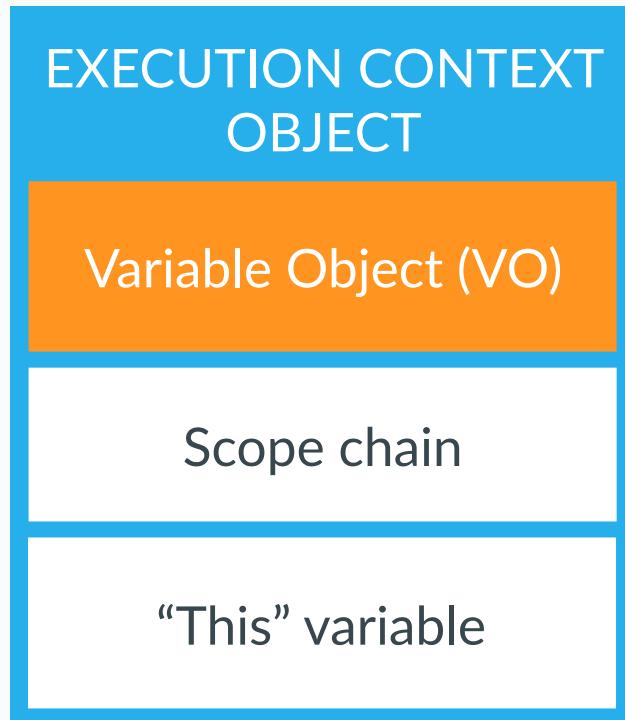
1. Creation phase

- A)** Creation of the Variable Object (VO)
- B)** Creation of the scope chain
- C)** Determine value of the ‘this’ variable

2. Execution phase

The code of the function that generated the current execution context is ran line by line

THE VARIABLE OBJECT



- The argument object is created, containing all the arguments that were passed into the function.
- Code is scanned for **function declarations**: for each function, a property is created in the Variable Object, **pointing to the function**.
- Code is scanned for **variable declarations**: for each variable, a property is created in the Variable Object, and set to **undefined**.

HOISTING



JONAS.IO
SCHMEDTMANN

THE COMPLETE JAVASCRIPT COURSE

BUILD REAL-WORLD PROJECTS!



@JONASSCHMEDTMAN

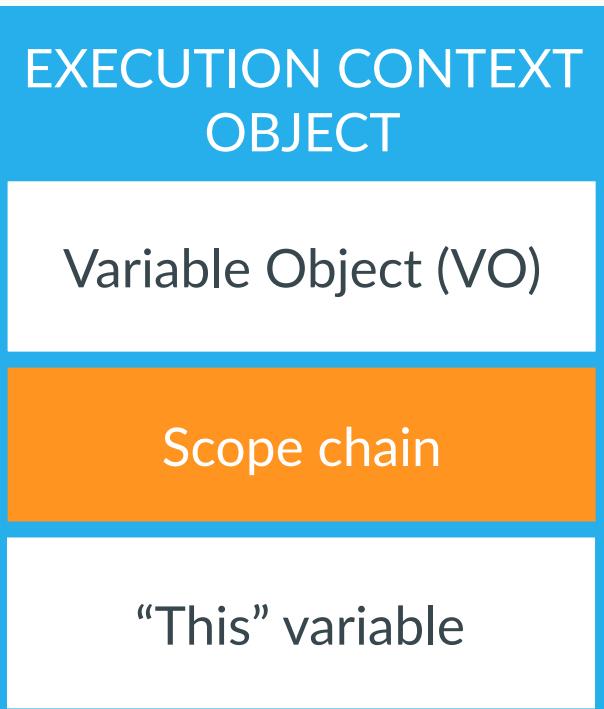
SECTION

HOW JAVASCRIPT WORKS BEHIND THE
SCENES

LECTURE

SCOPING AND THE SCOPE CHAIN

SCOPING IN JAVASCRIPT



- Scoping answers the question “where can we access a certain variable?”
- **Each new function creates a scope:** the space/environment, in which the variables it defines are accessible.
- **Lexical scoping:** a function that is lexically within another function gets access to the scope of the outer function.

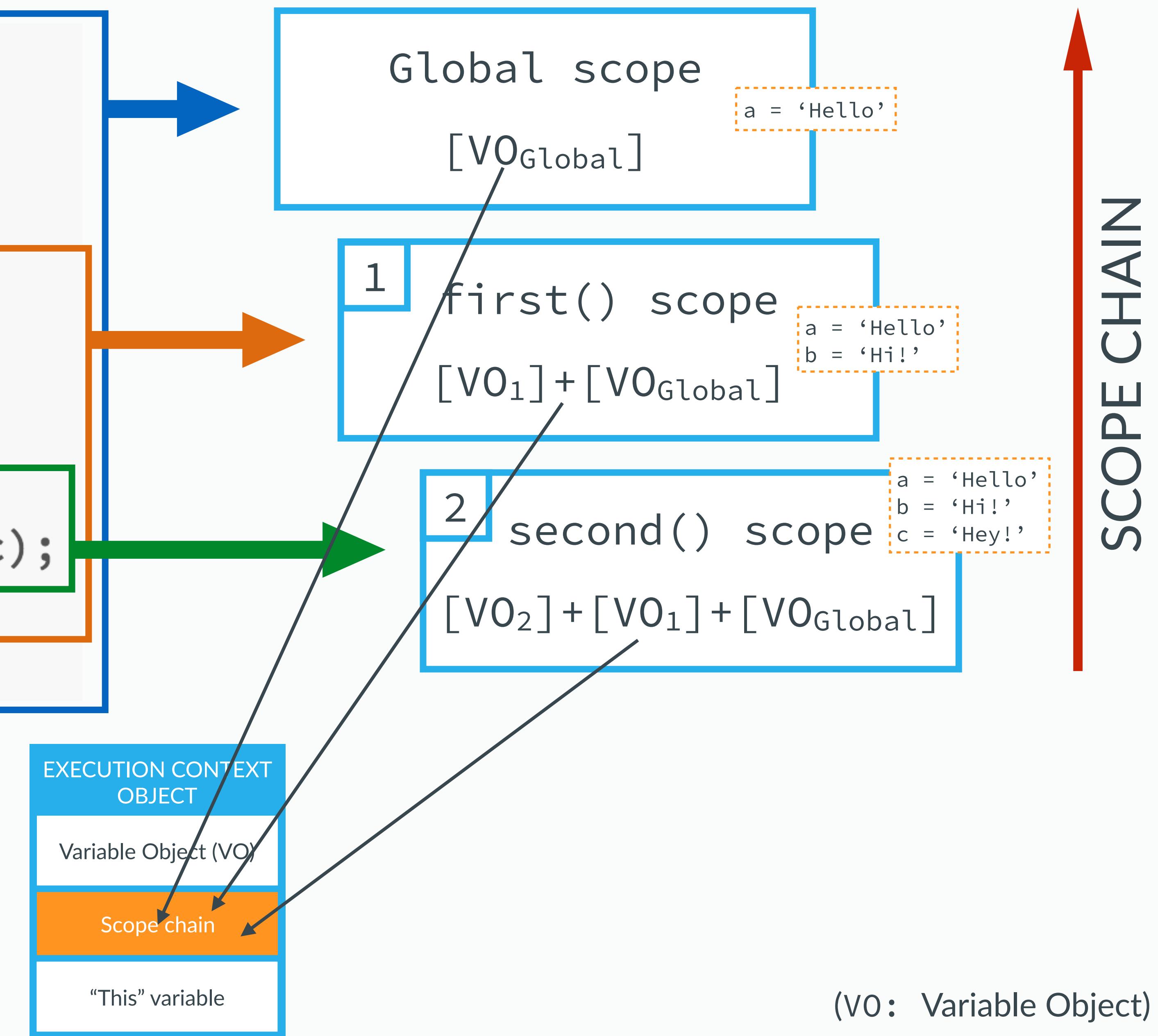
```

var a = 'Hello!';
first();

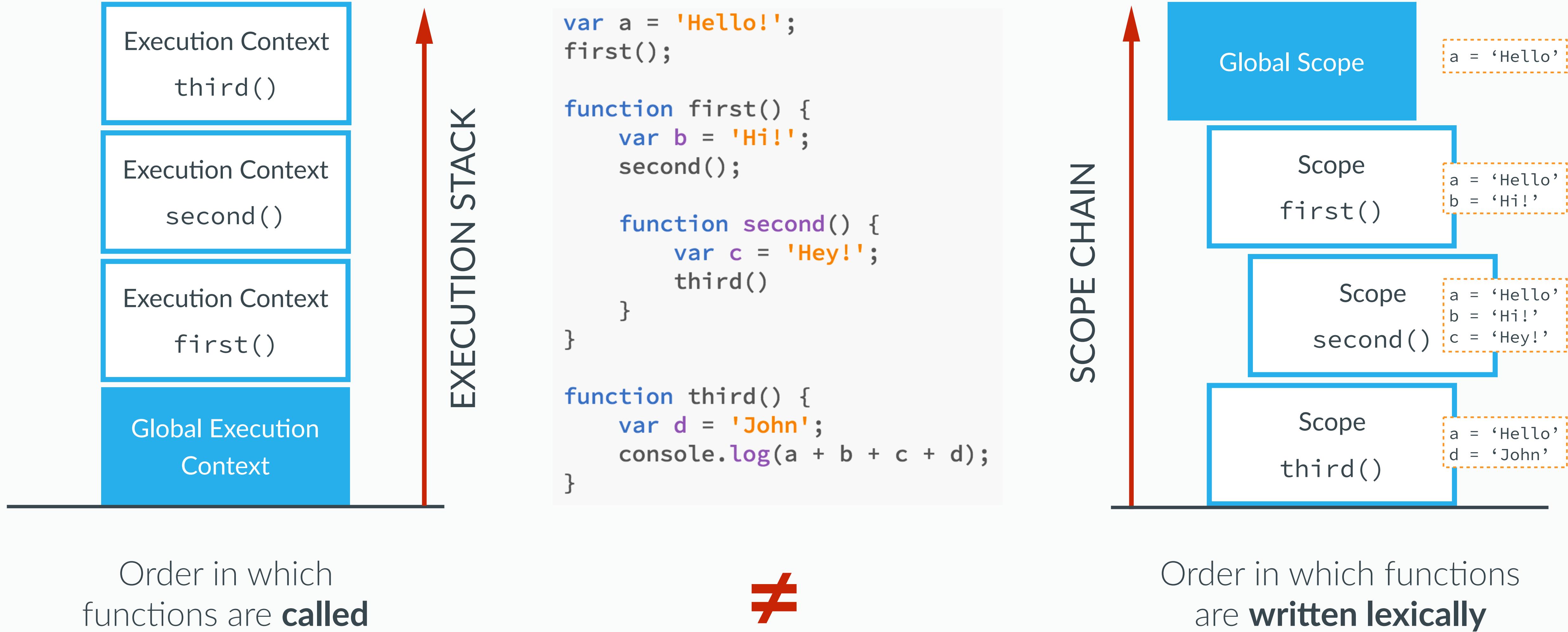
function first() {
  var b = 'Hi!';
  second();

  function second() {
    var c = 'Hey!';
    console.log(a + b + c);
  }
}

```



EXECUTION STACK VS SCOPE CHAIN





JONAS.IO
SCHMEDTMANN

THE COMPLETE JAVASCRIPT COURSE

BUILD REAL-WORLD PROJECTS!



@JONASSCHMEDTMAN

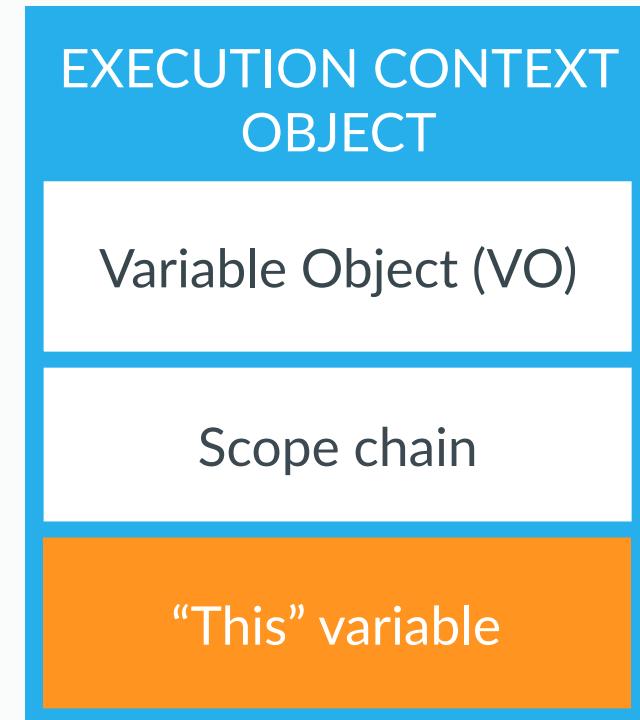
SECTION

HOW JAVASCRIPT WORKS BEHIND THE
SCENES

LECTURE

THE 'THIS' KEYWORD

THE 'THIS' VARIABLE



- **Regular function call:** the `this` keyword points at the global object, (the `window` object, in the browser).
- **Method call:** the `this` variable points to the object that is calling the method.
- *The `this` keyword is not assigned a value until a function where it is defined is actually called.*

SECTION 4 –

JAVASCRIPT IN THE

BROWSER: DOM

MANIPULATION AND

EVENTS



JONAS.IO
SCHMEDTMANN

THE COMPLETE JAVASCRIPT COURSE

BUILD REAL-WORLD PROJECTS!



@JONASSCHMEDTMAN

```
$classes = array( 'zoom' );  
  
if ( $loop == 0 || $loop % $columns == 0 )  
    $classes[] = 'first';  
  
if ( ( $loop + 1 ) % $columns == 0 )  
    $classes[] = 'last';  
  
$image_link = wp_get_attachment_url( $attachment_id );  
  
// + $image_link )
```

SECTION
JAVASCRIPT IN THE BROWSER: DOM
MANIPULATION AND EVENTS

LECTURE

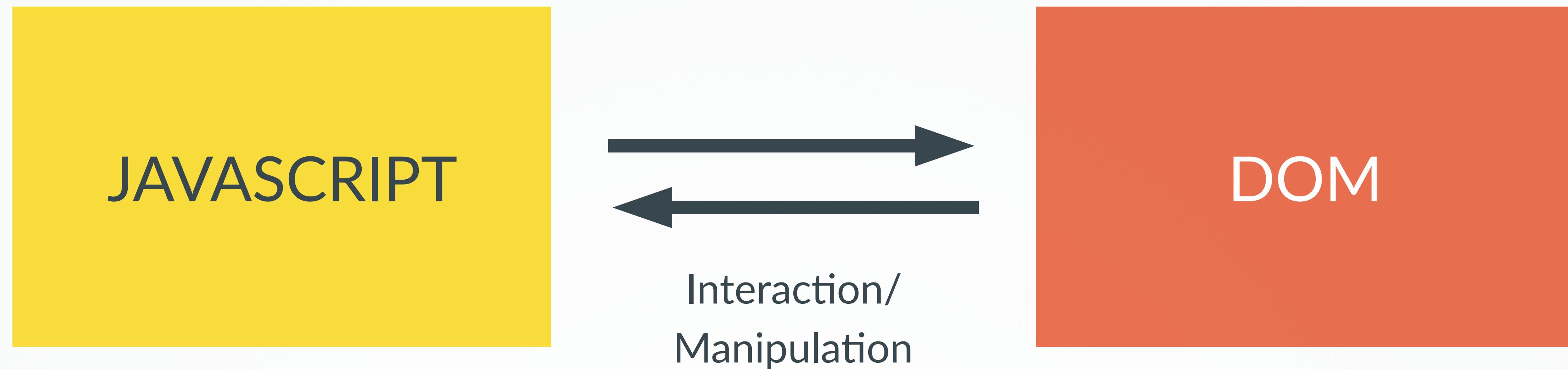
THE DOM AND DOM MANIPULATION

THE DOCUMENT OBJECT MODEL

- **DOM:** Document Object Model;
- Structured representation of an HTML document;
- The DOM is used to connect webpages to scripts like JavaScript;
- For each HTML box, there is an object in the DOM that we can access and interact with.

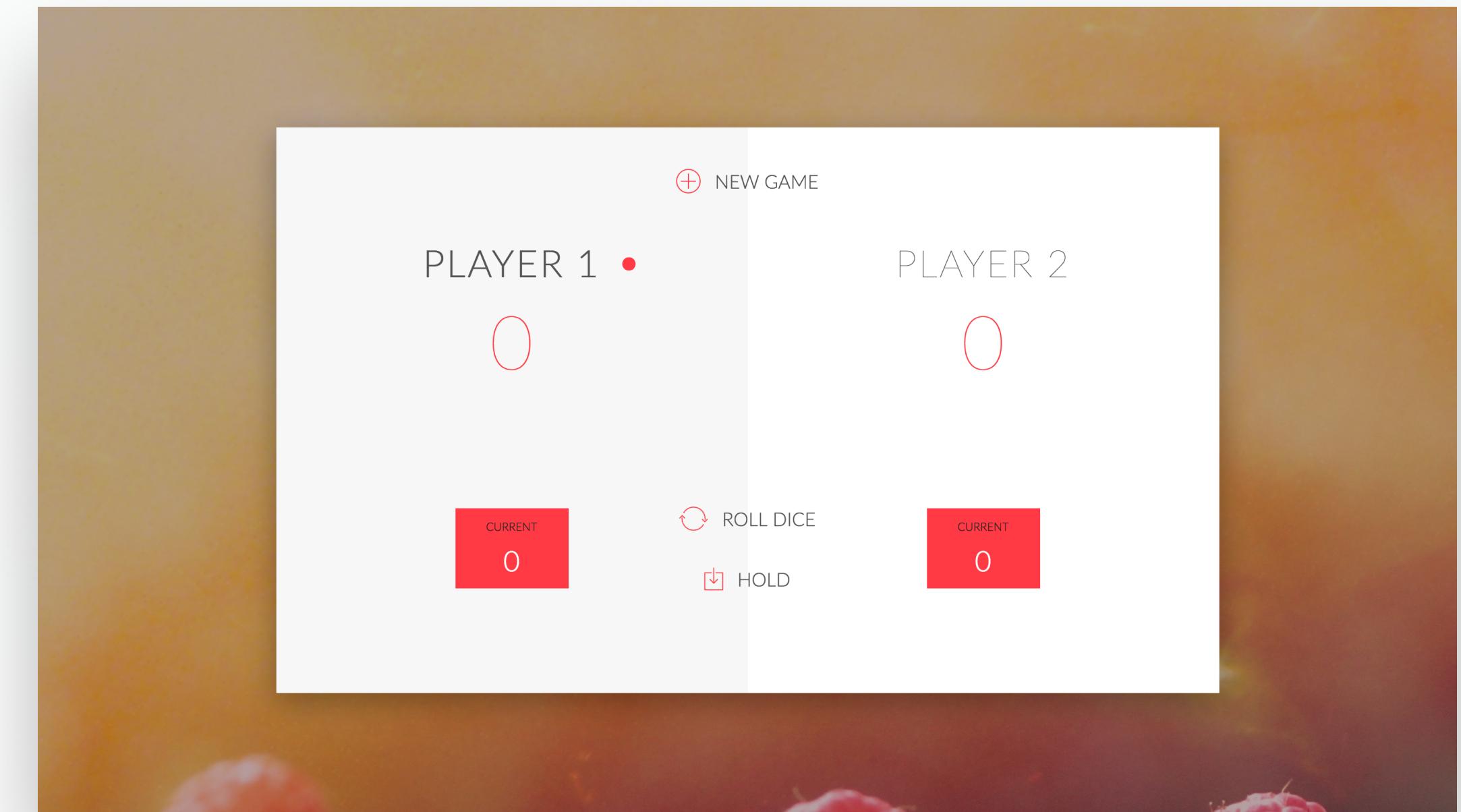
```
<body>
  <section>
    <p>A paragraph with a <a href="#">link</a>.</p>
    <p>Another second paragraph.</p>
  </section>
  <section>
    
  </section>
</body>
```

DOM MANIPULATION



WHAT YOU WILL LEARN IN THIS LECTURE

- How to create our fundamental game variables;
- How to generate a random number;
- How to manipulate the DOM;
- How to read from the DOM;
- How to change CSS styles.





JONAS.IO
SCHMEDTMANN

THE COMPLETE JAVASCRIPT COURSE

BUILD REAL-WORLD PROJECTS!



@JONASSCHMEDTMAN

SECTION

JAVASCRIPT IN THE BROWSER: DOM
MANIPULATION AND EVENTS

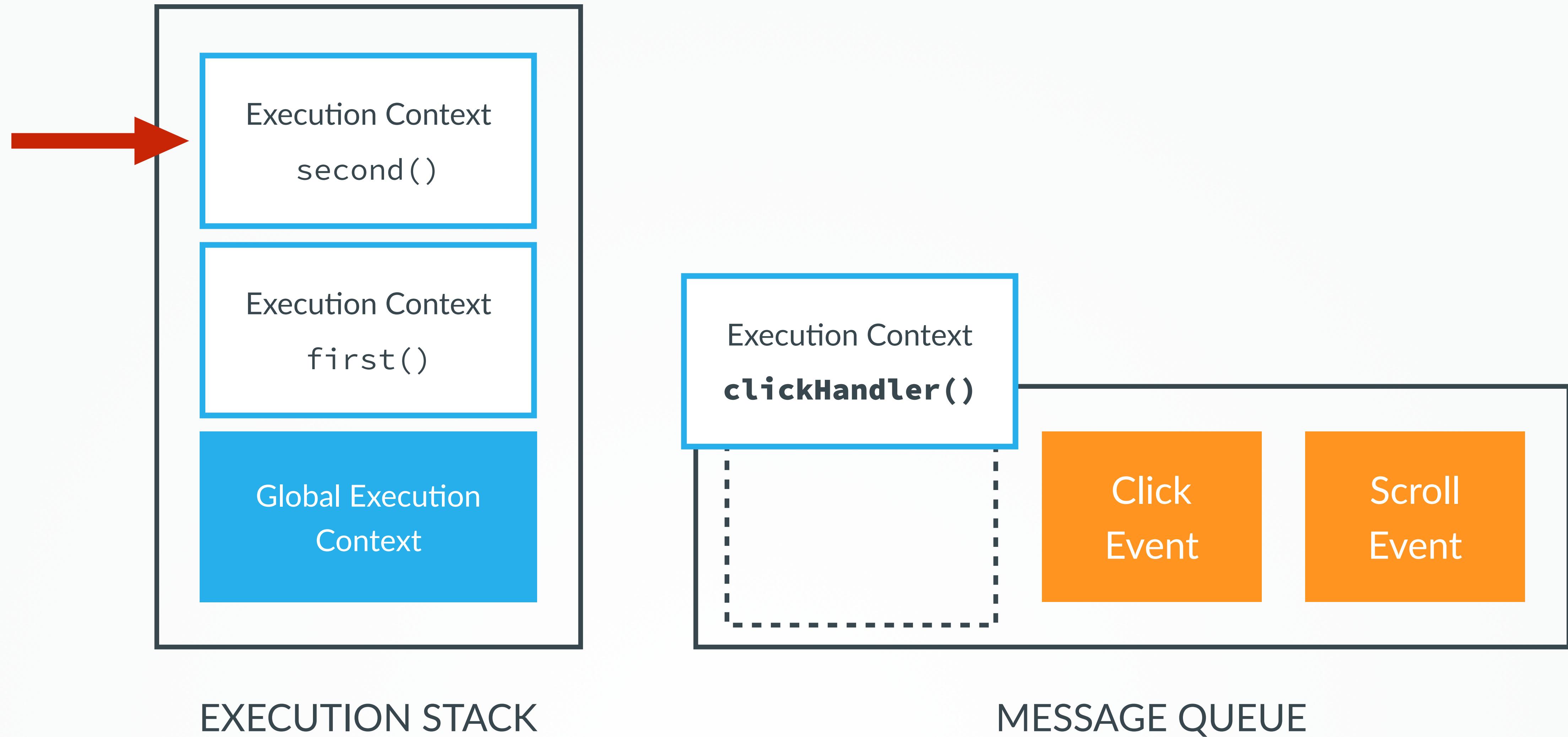
LECTURE

EVENTS AND EVENT HANDLING:
ROLLING THE DICE

WHAT ARE EVENTS?

- **Events:** Notifications that are sent to notify the code that something happened on the webpage;
- Examples: clicking a button, resizing a window, scrolling down or pressing a key;
- **Event listener:** A function that performs an action based on a certain event. It waits for a specific event to happen.

HOW EVENTS ARE PROCESSED



SECTION 5 – ADVANCED JAVASCRIPT: OBJECTS AND FUNCTIONS



JONAS.IO
SCHMEDTMANN

THE COMPLETE JAVASCRIPT COURSE

BUILD REAL-WORLD PROJECTS!



@JONASSCHMEDTMAN

SECTION

ADVANCED JAVASCRIPT: OBJECTS AND
FUNCTIONS

LECTURE

EVERYTHING IS AN OBJECT:
INHERITANCE AND THE PROTOTYPE
CHAIN

OBJECTS IN JAVASCRIPT

PRIMITIVES

- Numbers
- Strings
- Booleans
- Undefined
- Null

Everything is an object.

(Well, almost everything)

EVERYTHING ELSE ...

- Arrays
- Functions
- Objects
- Dates
- Wrappers for Numbers, Strings, Booleans

... IS AN OBJECT

THE OBJECT ORIENTED PARADIGM

OBJECT-ORIENTED PROGRAMMING

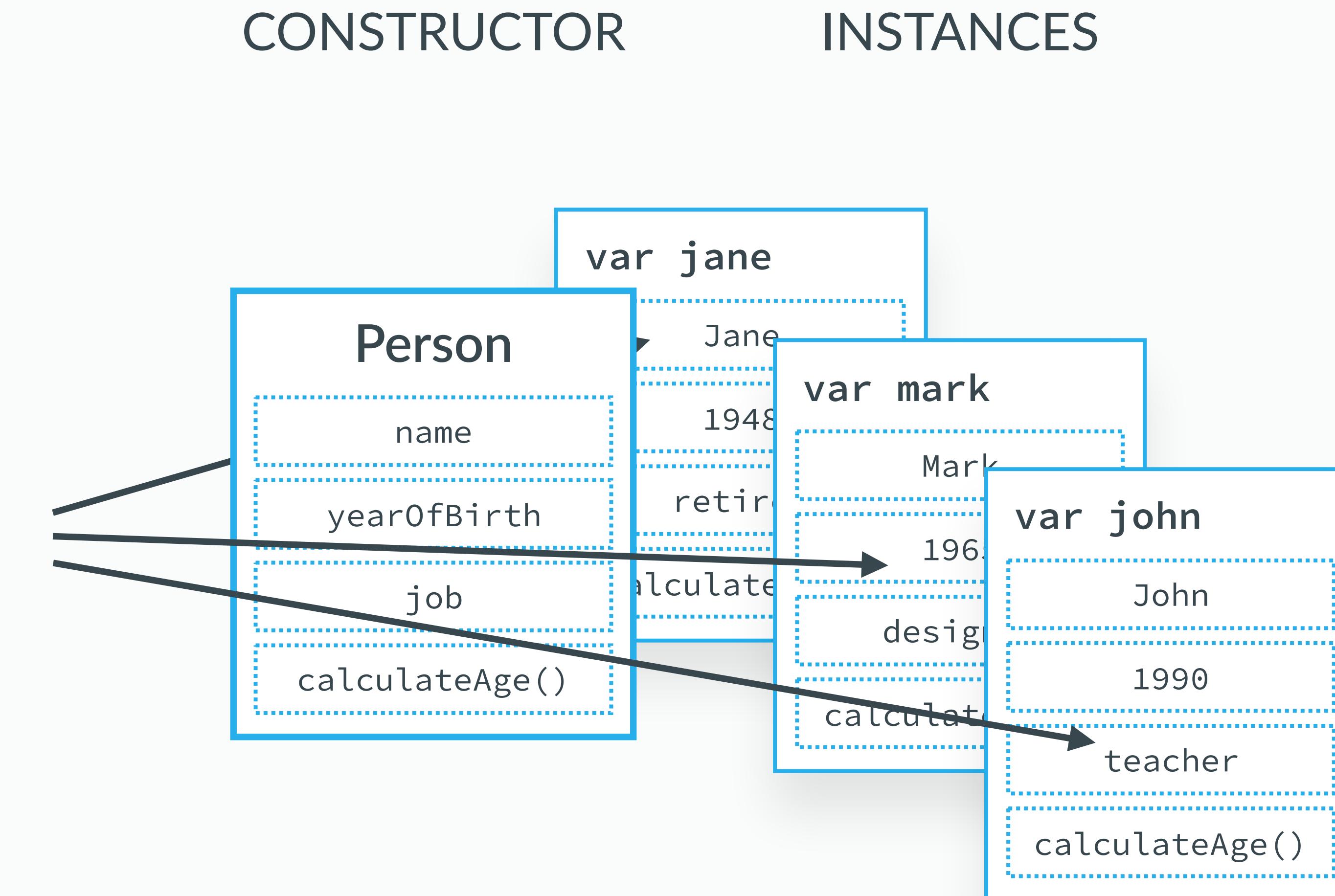
- Objects interacting with one another through methods and properties;
- Used to store data, structure applications into modules and keeping code clean.

```
var john = {  
    name: 'John',  
    yearOfBirth: 1990,  
    isMarried: false  
};
```

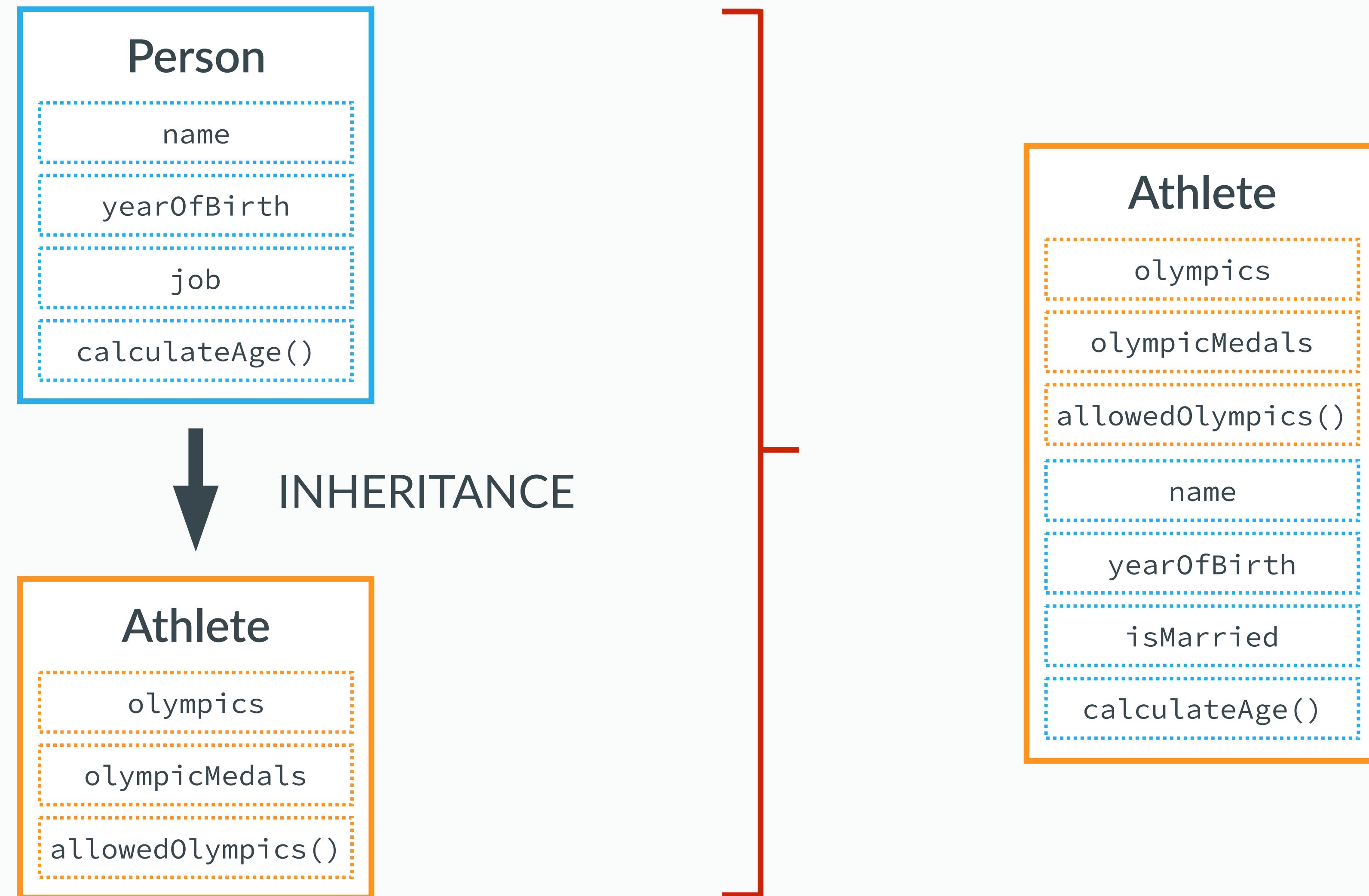
```
var jane = {  
    name: 'Jane',  
    yearOfBirth: 1969,  
    isMarried: true  
};
```

```
var mark = {  
    name: 'Mark',  
    yearOfBirth: 1948,  
    isMarried: true  
};
```

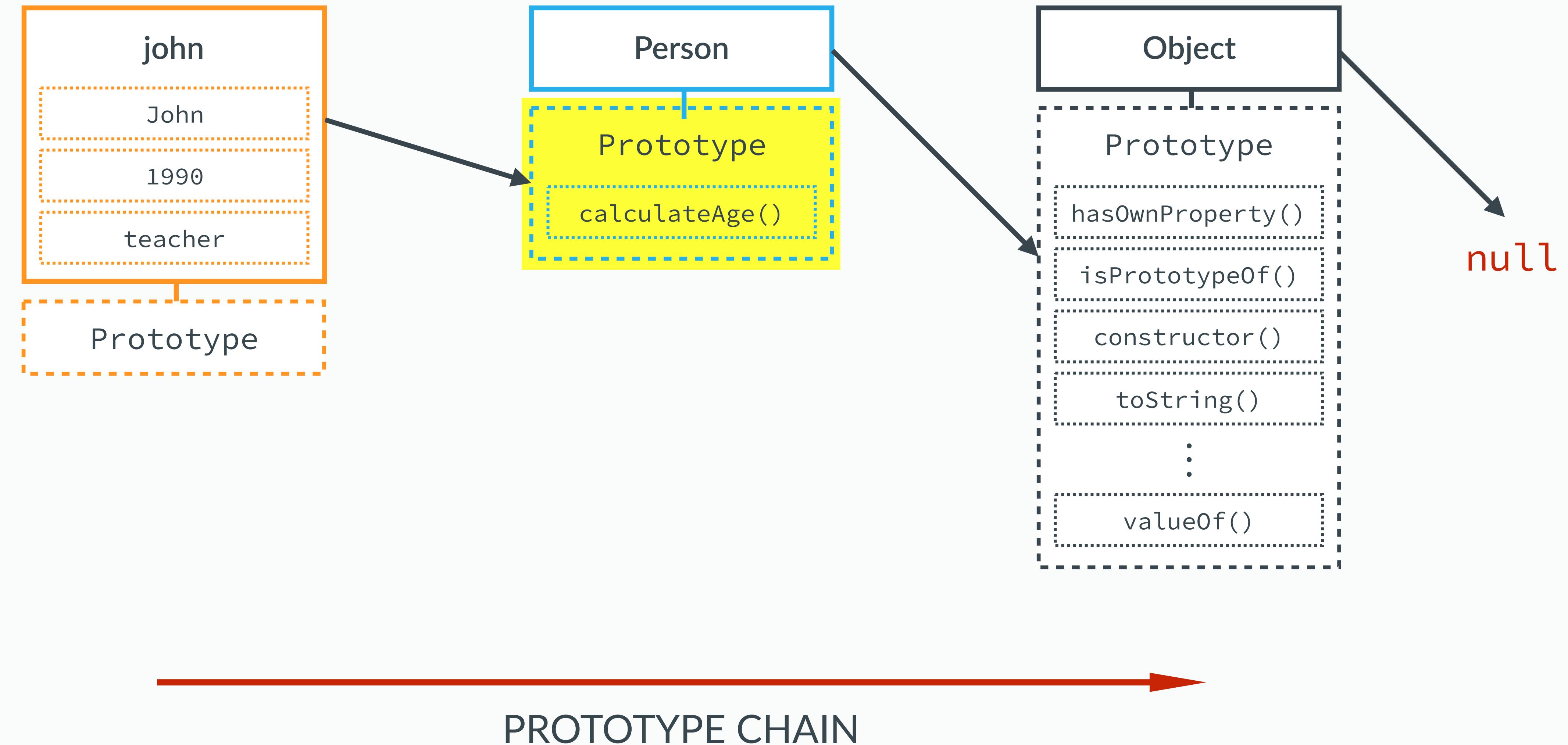
CONSTRUCTORS AND INSTANCES IN JAVASCRIPT



INHERITANCE IN GENERAL



INHERITANCE IN JAVASCRIPT: PROTOTYPES AND PROTOTYPE CHAINS



SUMMARY

- Every JavaScript object has a **prototype property**, which makes inheritance possible in JavaScript;
- The prototype property of an object is where we put methods and properties that we want **other objects to inherit**;
- The Constructor's prototype property is **NOT** the prototype of the Constructor itself, it's the prototype of **ALL** instances that are created through it;
- When a certain method (or property) is called, the search starts in the object itself, and if it cannot be found, the search moves on to the object's prototype. This continues until the method is found: **prototype chain**.



JONAS.IO
SCHMEDTMANN

THE COMPLETE JAVASCRIPT COURSE

BUILD REAL-WORLD PROJECTS!



@JONASSCHMEDTMAN

SECTION

ADVANCED JAVASCRIPT: OBJECTS AND
FUNCTIONS

LECTURE

CREATING OBJECTS: FUNCTION
CONSTRUCTORS



JONAS.IO
SCHMEDTMANN

THE COMPLETE JAVASCRIPT COURSE

BUILD REAL-WORLD PROJECTS!



@JONASSCHMEDTMAN

SECTION

ADVANCED JAVASCRIPT: OBJECTS AND
FUNCTIONS

LECTURE

THE PROTOTYPE CHAIN IN THE
CONSOLE



JONAS.IO
SCHMEDTMANN

THE COMPLETE JAVASCRIPT COURSE

BUILD REAL-WORLD PROJECTS!



@JONASSCHMEDTMAN

SECTION

ADVANCED JAVASCRIPT: OBJECTS AND
FUNCTIONS

LECTURE

CREATING OBJECTS: OBJECT.CREATE



JONAS.IO
SCHMEDTMANN

THE COMPLETE JAVASCRIPT COURSE

BUILD REAL-WORLD PROJECTS!



@JONASSCHMEDTMAN

SECTION

ADVANCED JAVASCRIPT: OBJECTS AND
FUNCTIONS

LECTURE

PRIMITIVES VS. OBJECTS



JONAS.IO
SCHMEDTMANN

THE COMPLETE JAVASCRIPT COURSE

BUILD REAL-WORLD PROJECTS!



@JONASSCHMEDTMAN

SECTION

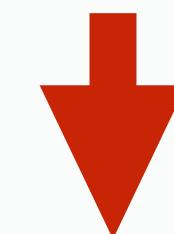
ADVANCED JAVASCRIPT: OBJECTS AND
FUNCTIONS

LECTURE

FIRST CLASS FUNCTIONS: PASSING
FUNCTIONS AS ARGUMENTS

FUNCTIONS ARE ALSO OBJECTS IN JAVASCRIPT

- A function is an instance of the Object type;
- A function behaves like any other object;
- We can store functions in a variable;
- We can pass a function as an argument to another function;
- We can return a function from a function.



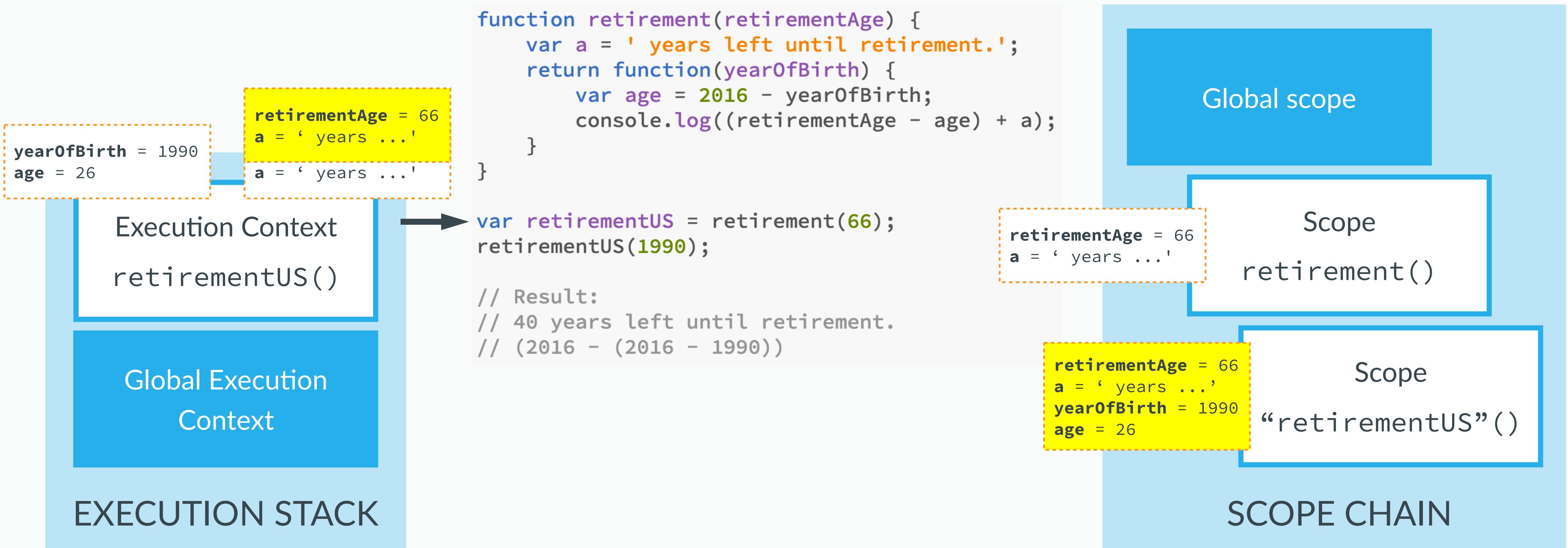
FIRST-CLASS FUNCTIONS

CLOSURES

CLOSURES SUMMARY

An inner function has always access to the variables and parameters of its outer function, even after the outer function has returned.

HOW CLOSURES WORK



CLOSURES

CLOSURES SUMMARY (AGAIN :)

An inner function has always access to the variables and parameters of its outer function, even after the outer function has returned.



JONAS.IO
SCHMEDTMANN

THE COMPLETE JAVASCRIPT COURSE

BUILD REAL-WORLD PROJECTS!



@JONASSCHMEDTMAN

SECTION

ADVANCED JAVASCRIPT: OBJECTS AND
FUNCTIONS

LECTURE

BIND, CALL AND APPLY

NOT SO FAST...

(LET'S THINK TOGETHER)

STRUCTURING OUR CODE WITH MODULES

UI MODULE

DATA MODULE

CONTROLLER MODULE

TO-DO LIST

Add event handler

Get input values

Add the new item to
our data structure

Add the new item to
the UI

Calculate budget

Update the UI



JONAS.IO
SCHMEDTMANN

THE COMPLETE JAVASCRIPT COURSE

BUILD REAL-WORLD PROJECTS!



@JONASSCHMEDTMAN

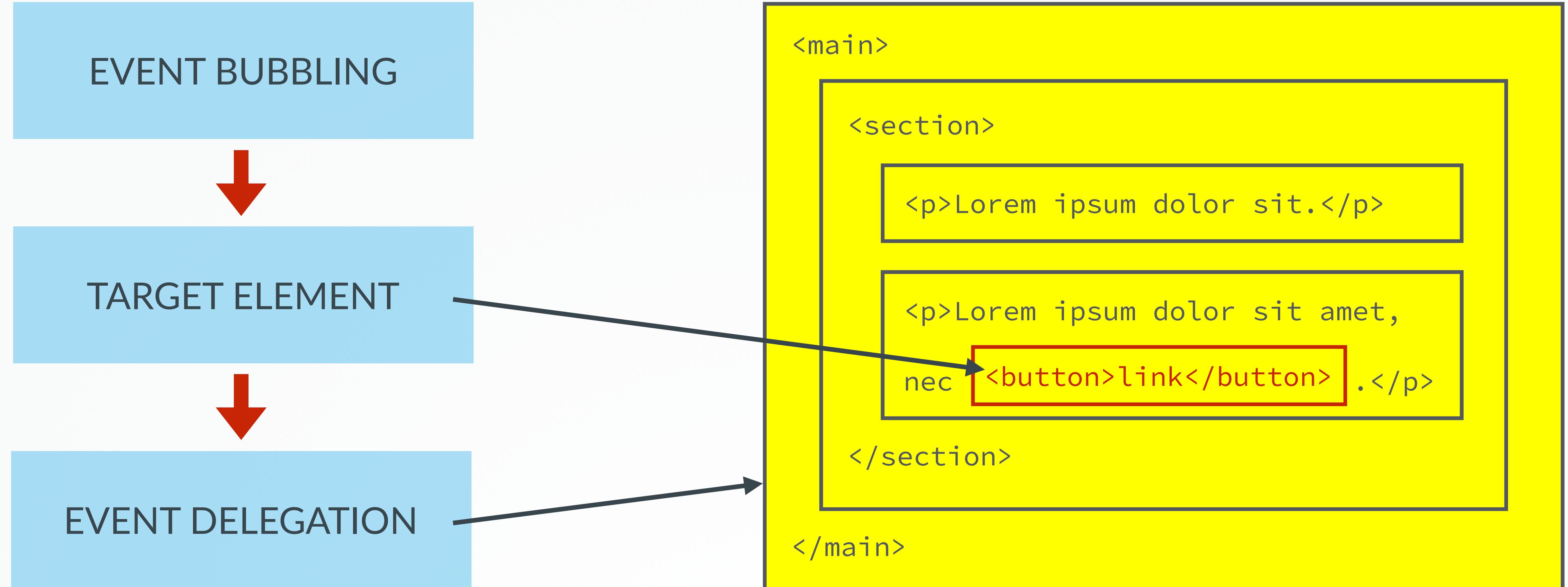
SECTION

PUTTING IT ALL TOGETHER: THE
BUDGET APP PROJECT

LECTURE

IMPLEMENTING THE MODULE PATTERN

EVENT BUBBLING, TARGET ELEMENT AND EVENT DELEGATION



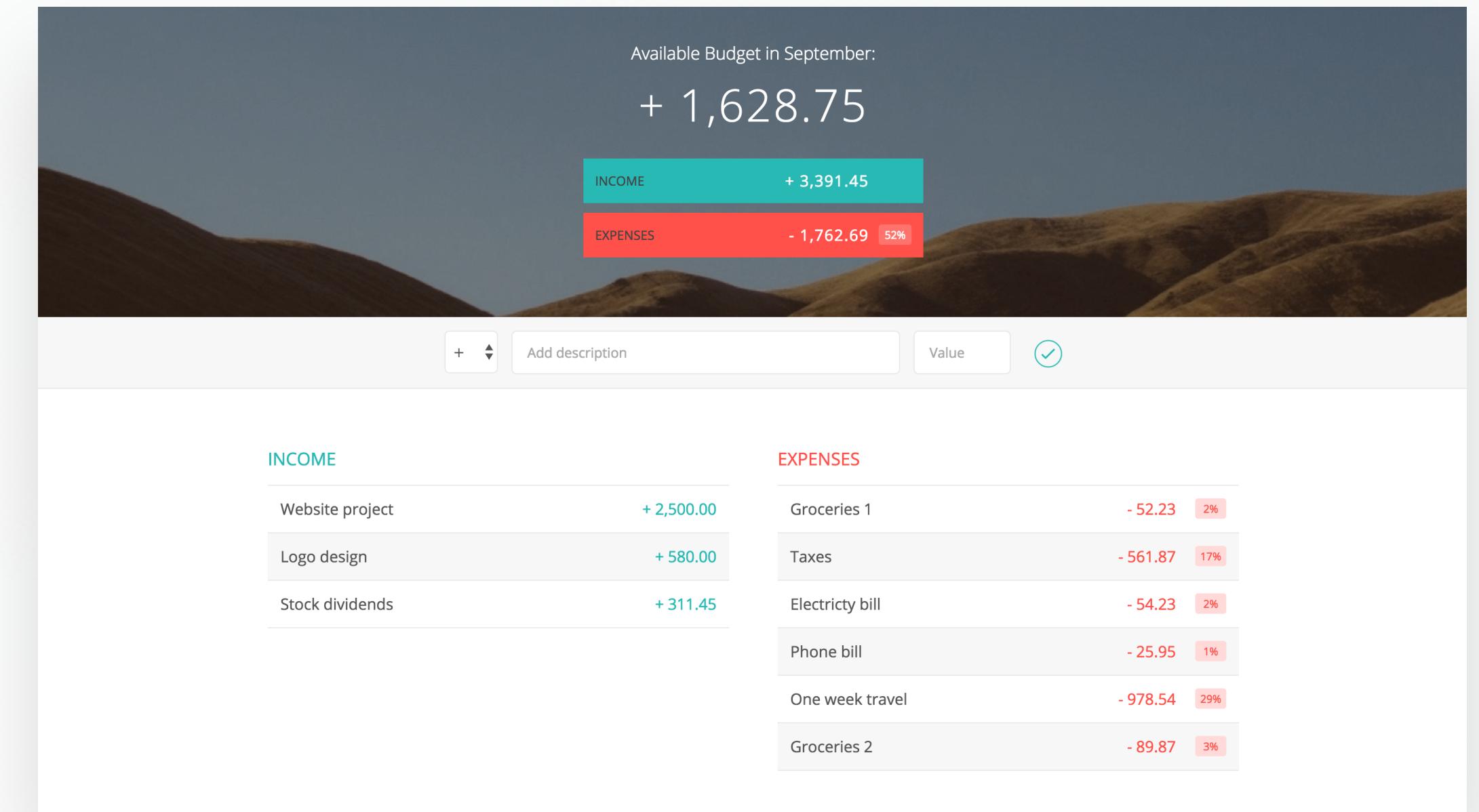
WHEN TO USE EVENT DELEGATION

USE CASES FOR EVENT DELEGATION

1. When we have an element with lots of child elements that we are interested in;
2. When we want an event handler attached to an element that is not yet in the DOM when our page is loaded.

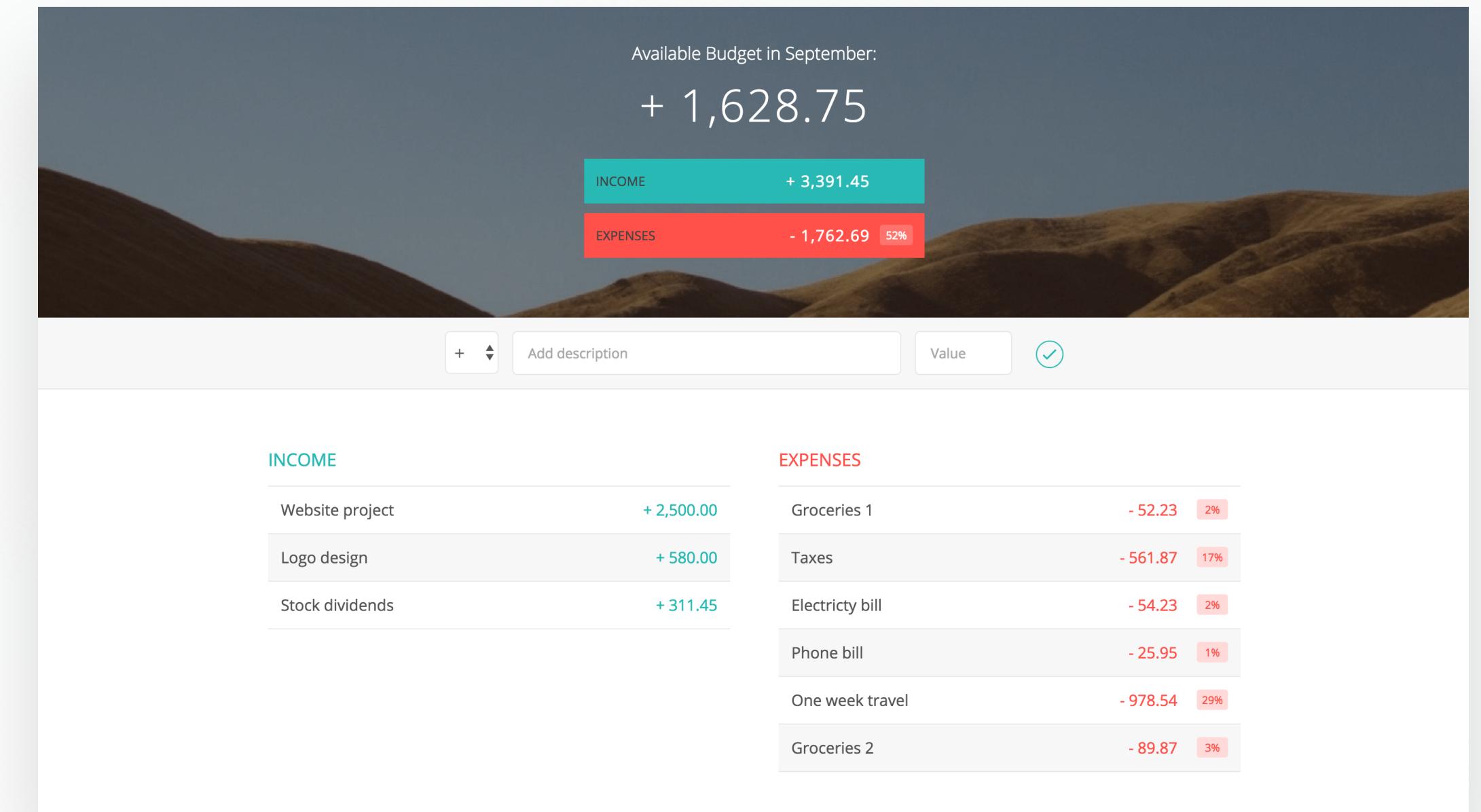
WHAT YOU WILL LEARN IN THIS LECTURE

- How to use event delegation in practice;
- How to use IDs in HTML to connect the UI with the data model;
- How to use the parentNode property for DOM traversing.

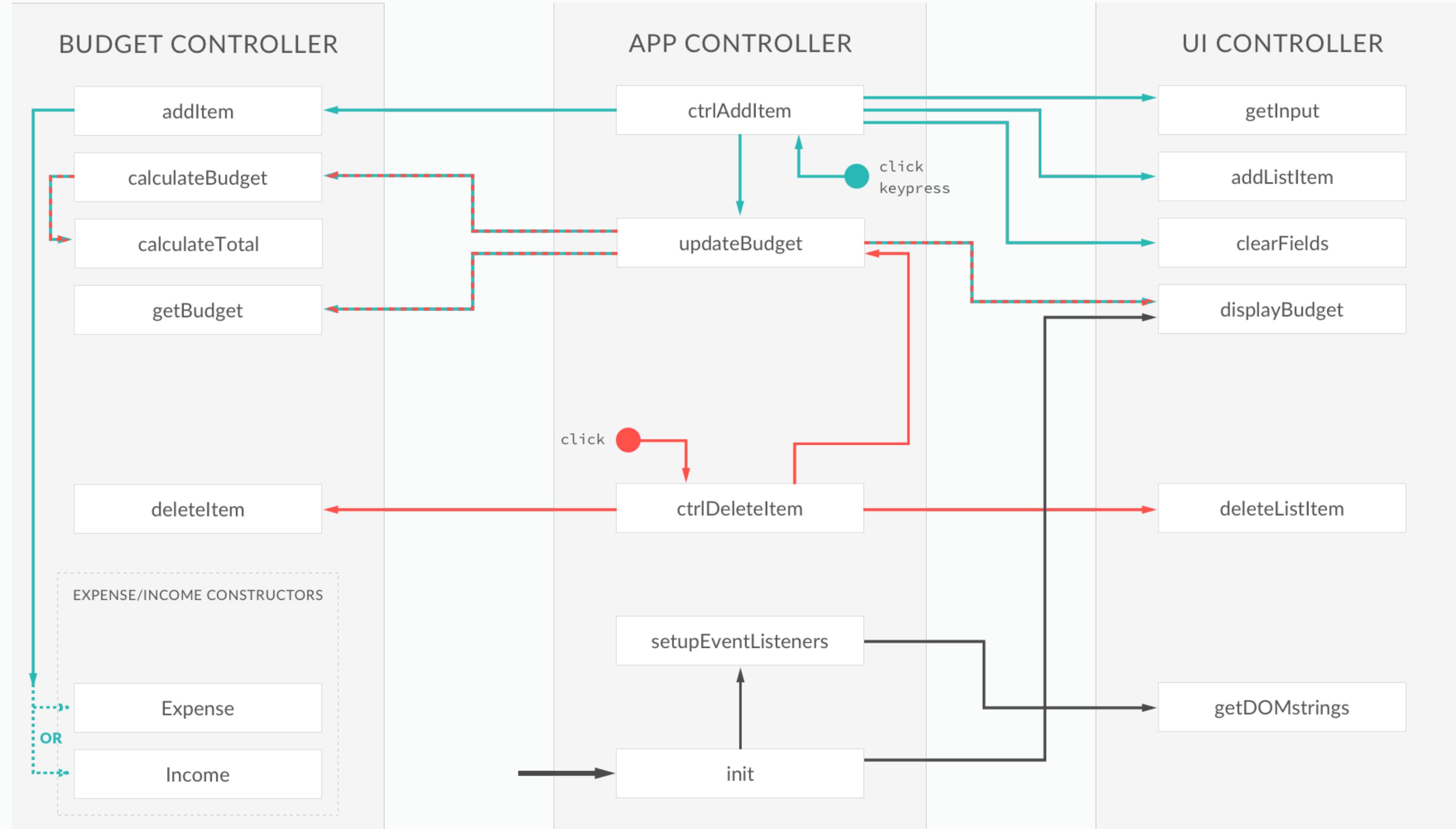


WHAT YOU WILL LEARN IN THIS LECTURE

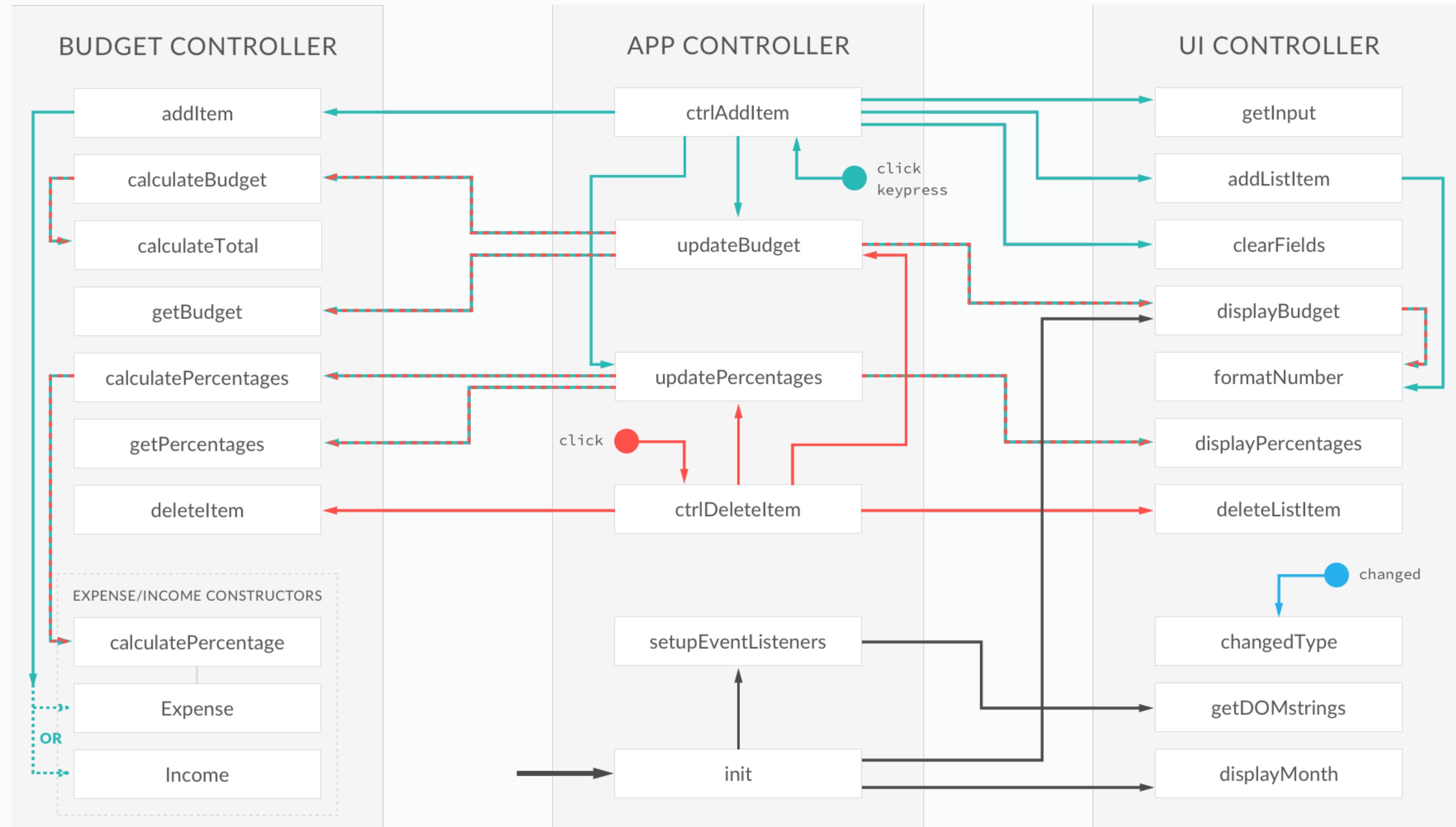
- More DOM manipulation: how to remove an element from the DOM.



AFTER STEP 2... (THAT WE JUST COMPLETED)



OUR FINAL ARCHITECTURE



**SECTION 7 –
GET READY FOR THE
FUTURE:
ECMASCRIPT2015**



JONAS.IO
SCHMEDTMANN

THE COMPLETE JAVASCRIPT COURSE

BUILD REAL-WORLD PROJECTS!



@JONASSCHMEDTMAN

SECTION

NEXT GENERATION JAVASCRIPT: INTRO
TO ES6 / ES2015

LECTURE

SECTION INTRO



JONAS.IO

SCHMEDTMANN

THE COMPLETE JAVASCRIPT COURSE

BUILD REAL-WORLD PROJECTS!

@JONASSCHMEDTMAN

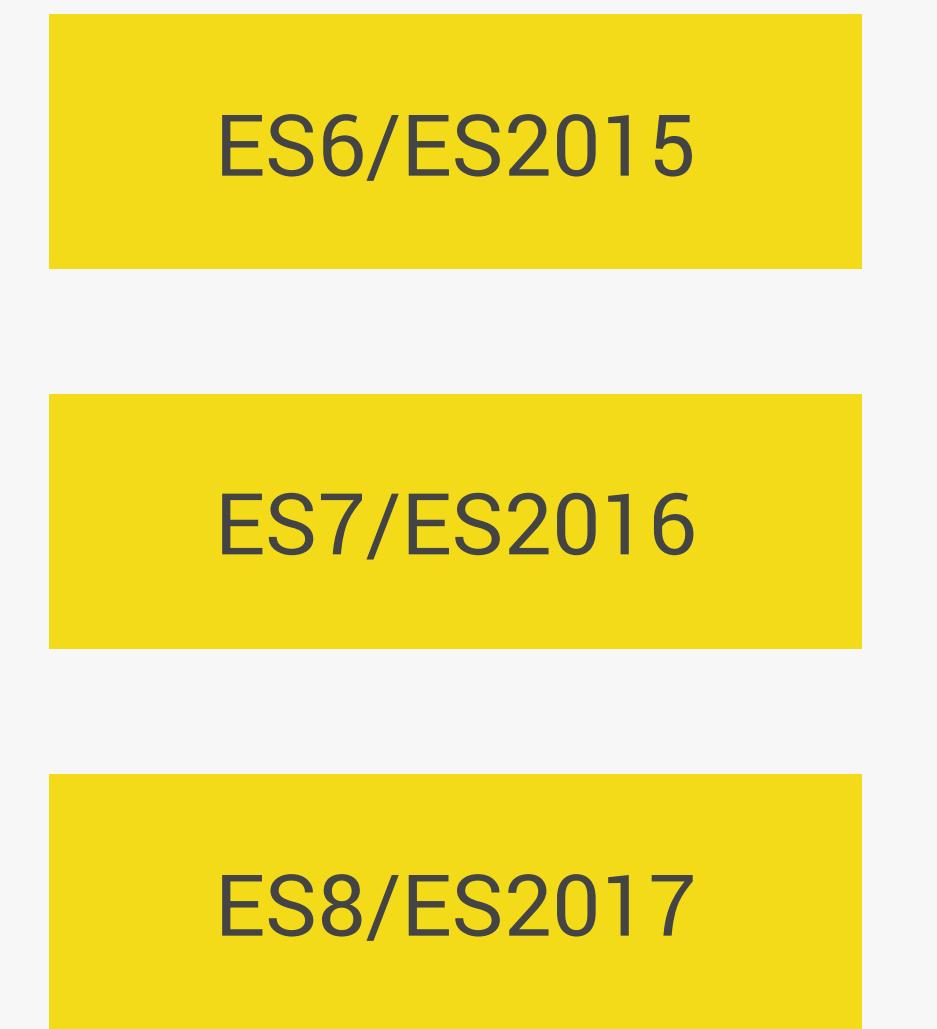
SECTION

NEXT GENERATION JAVASCRIPT: INTRO TO ES6 / ES2015

LECTURE

WHAT'S NEW IN ES6 / ES2015

JAVASCRIPT VERSIONS: QUICK RECAP



- Well supported in all **modern** browsers
- No support in older browsers;
- **Can use most features in production with transpiling and polyfilling (converting to ES5) 😊**

NEW ES6 FEATURES WE'LL COVER IN THIS SECTION

- Variable Declarations with let and const
- Blocks and IIFEs
- Strings
- Arrow Functions
- Destructuring
- Arrays
- The Spread Operator
- Rest and Default Parameters
- Maps
- Classes and subclasses

LATER...

- Promises
- Native modules



JONAS.IO
SCHMEDTMANN

THE COMPLETE JAVASCRIPT COURSE

BUILD REAL-WORLD PROJECTS!



@JONASSCHMEDTMAN

SECTION

NEXT GENERATION JAVASCRIPT: INTRO
TO ES6 / ES2015

LECTURE

VARIABLE DECLARATIONS WITH LET
AND CONST



JONAS.IO
SCHMEDTMANN

THE COMPLETE JAVASCRIPT COURSE

BUILD REAL-WORLD PROJECTS!



@JONASSCHMEDTMAN

SECTION

NEXT GENERATION JAVASCRIPT: INTRO
TO ES6 / ES2015

LECTURE

BLOCKS AND IIFES



JONAS.IO
SCHMEDTMANN

THE COMPLETE JAVASCRIPT COURSE

BUILD REAL-WORLD PROJECTS!



@JONASSCHMEDTMAN

SECTION

NEXT GENERATION JAVASCRIPT: INTRO
TO ES6 / ES2015

LECTURE

STRINGS IN ES6 / ES2015



JONAS.IO
SCHMEDTMANN

THE COMPLETE JAVASCRIPT COURSE

BUILD REAL-WORLD PROJECTS!



@JONASSCHMEDTMAN

SECTION

NEXT GENERATION JAVASCRIPT: INTRO
TO ES6 / ES2015

LECTURE

ARROW FUNCTIONS: BASICS



JONAS.IO
SCHMEDTMANN

THE COMPLETE JAVASCRIPT COURSE

BUILD REAL-WORLD PROJECTS!



@JONASSCHMEDTMAN

SECTION

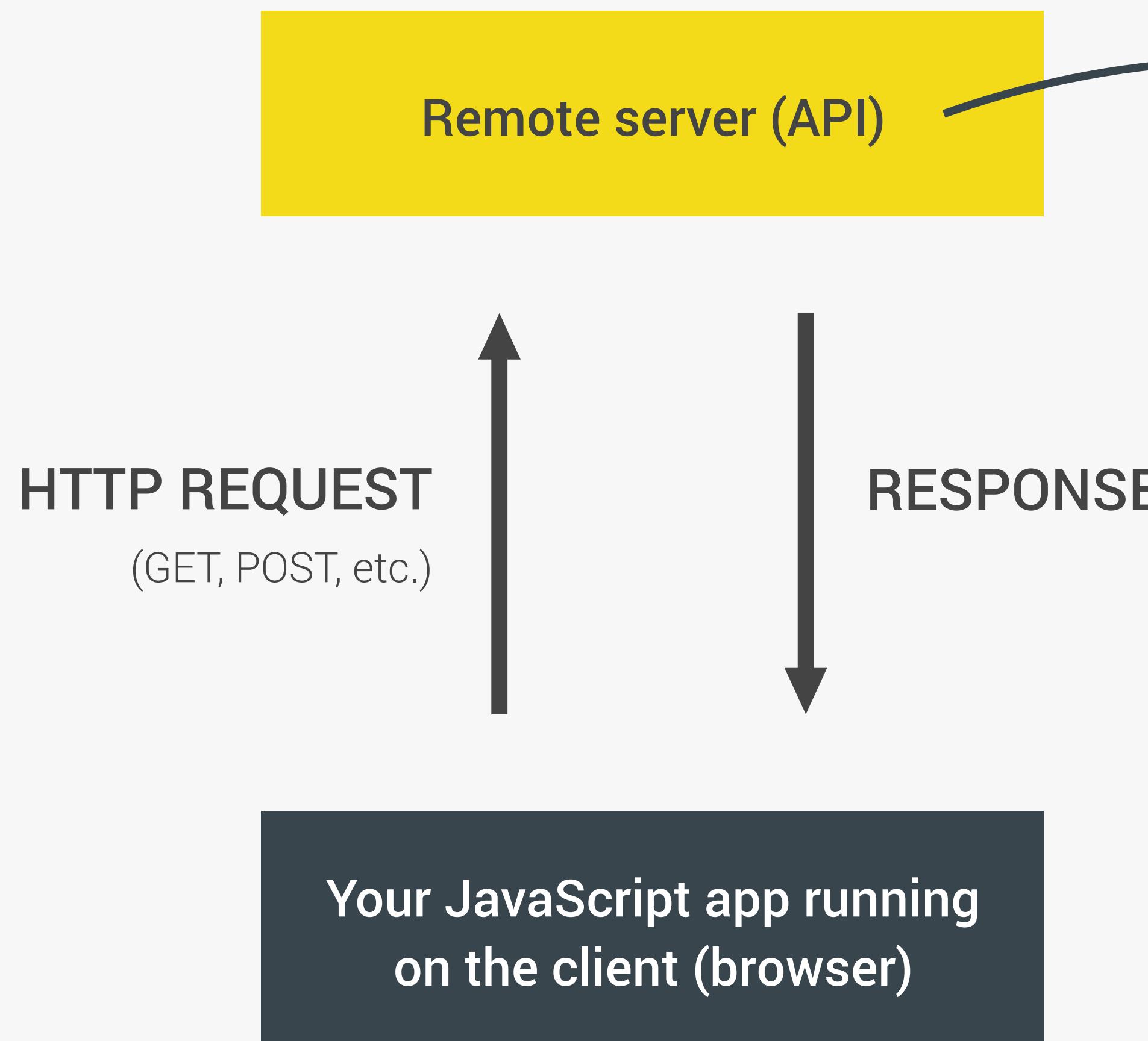
NEXT GENERATION JAVASCRIPT: INTRO
TO ES6 / ES2015

LECTURE

ARROW FUNCTIONS: LEXICAL 'THIS'
KEYWORD

WHAT ARE AJAX AND APIs?

ASYNCHRONOUS JAVASCRIPT AND XML



APPLICATION PROGRAMMING INTERFACE

- Your own API, for data coming from your own server
- 3rd-party APIs:
 - Google Maps
 - Embed Youtube videos
 - Weather data
 - Movies data
 - Send email or SMS
 - Thousands of possibilities...

SECTION 9 – MODERN JAVASCRIPT



THE COMPLETE JAVASCRIPT COURSE

BUILD REAL-WORLD PROJECTS!



@JONASSCHMEDTMAN

```
$classes = array( 'zoom' );  
  
if ( $loop == 0 || $loop % $columns == 0 )  
    $classes[] = 'first';  
  
if ( ( $loop + 1 ) % $columns == 0 )  
    $classes[] = 'last';  
  
$image_link = wp_get_attachment_url( $attachment_id );  
  
$image_link = wp_get_attachment_image( $attachment_id, wp_get_attachment_image_src( $attachment_id, 'single' )[0] );  
  
$class = esc_attr( implode( ' ', $classes ) );  
$str = str_replace( get_the_title( $attachment_id ), $class, $str );
```

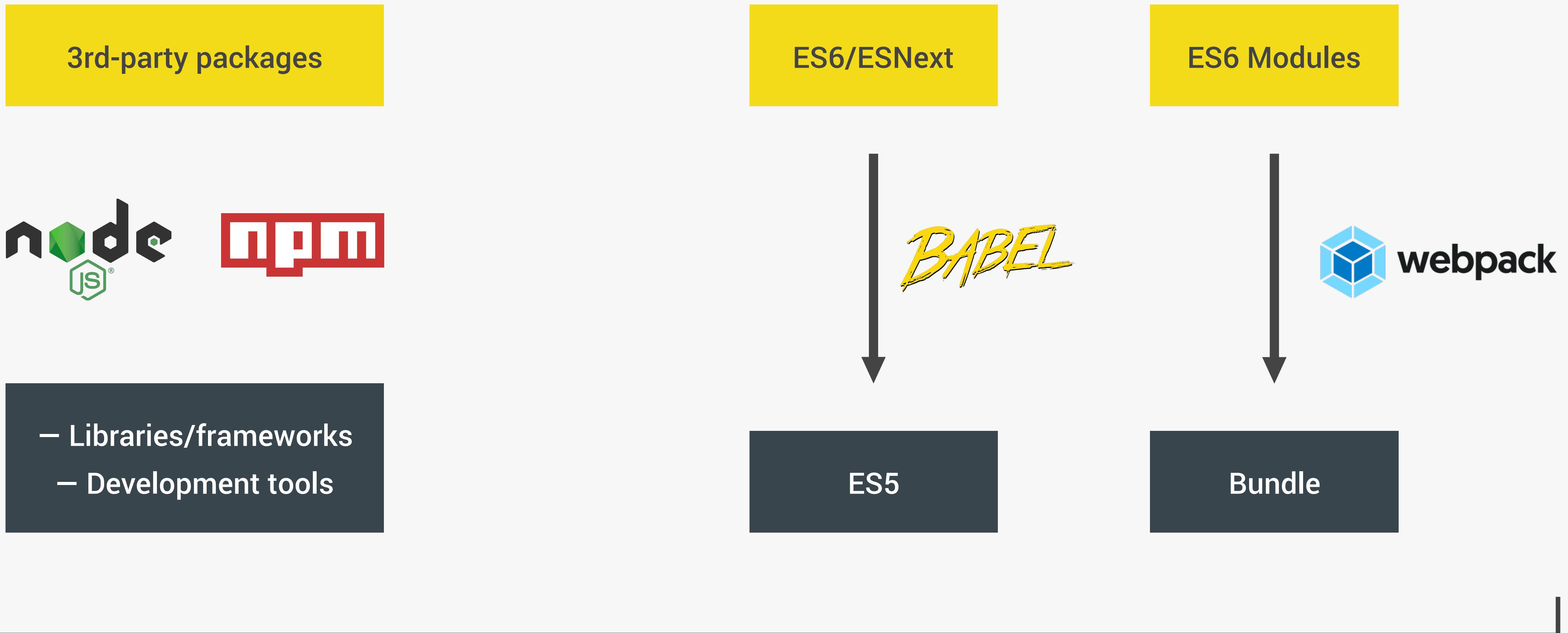
SECTION

MODERN JAVASCRIPT: USING ES6, NPM,
BABEL AND WEBPACK

LECTURE

SECTION INTRO

MODERN JAVASCRIPT: A BRIEF OVERVIEW



Putting it all together with an automated development setup powered by **npm** scripts



JONAS.IO
SCHMEDTMANN

THE COMPLETE JAVASCRIPT COURSE

BUILD REAL-WORLD PROJECTS!



@JONASSCHMEDTMAN

```
$classes = array( 'zoom' );  
  
if ( $loop == 0 || $loop % $columns == 0 )  
    $classes[] = 'first';  
  
if ( ( $loop + 1 ) % $columns == 0 )  
    $classes[] = 'last';  
  
$image_link = wp_get_attachment_url( $attachment_id );  
  
// + $image_link )
```

SECTION
MODERN JAVASCRIPT: USING ES6, NPM,
BABEL AND WEBPACK

LECTURE

A MODERN SETUP: CONFIGURING
WEBPACK

OUR MODEL-VIEW-CONTROLLER ARCHITECTURE (MVC)

