GENERAL ASSEMBLY

JAVASCRIPT DEVELOPMENT

Sasha Vodnik, Instructor

HELLO!

1. Pull changes from the `svodnik/JS-SF-9-resources` repo to your computer
2. Open the `16-deploying > starter-code` folder in your code editor

DEPLOYING YOUR APP

LEARNING OBJECTIVES

At the end of this class, you will be able to

- Understand what hosting is.
- Identify a program's needs in terms of host providers.
- Ensure backward compatibility by using Babel to transpile code.
- Deploy to a web host.

AGENDA

- Update with Firebase
- Delete with Firebase
- Convert code to a module
- Transpile with Babel
- Deploy with Firebase

WEEKLY OVERVIEW

WEEK 9	CRUD & Firebase / Deploying your app
WEEK 10	React / Final project lab
WEEK 11	Final project presentations

EXIT TICKET QUESTIONS

1. Should we keep our Firebase backend code separate from our regular JS when we build our final projects?
2. make up&down vote work
3. Can we link buttons (up down etc) to other websites

CRUD

LAB — IMPLEMENT UPDATE FUNCTIONALITY

KEY OBJECTIVE

- Build the Update functionality of a full-stack app

TYPE OF EXERCISE

- Solo or in pairs

TIMING

10 min

1. Examine the API documentation at
   - https://firebase.google.com/docs/reference/js/firebase.database.Reference#update
   - https://firebase.google.com/docs/reference/js/firebase.database.Reference#set
2. Create a function to make updates to the database
3. Add calls to your new function when data is changed in your app

LAB — IMPLEMENT DELETE FUNCTIONALITY

KEY OBJECTIVE

- Build the Delete functionality of a full-stack app

TYPE OF EXERCISE

- Solo or in pairs

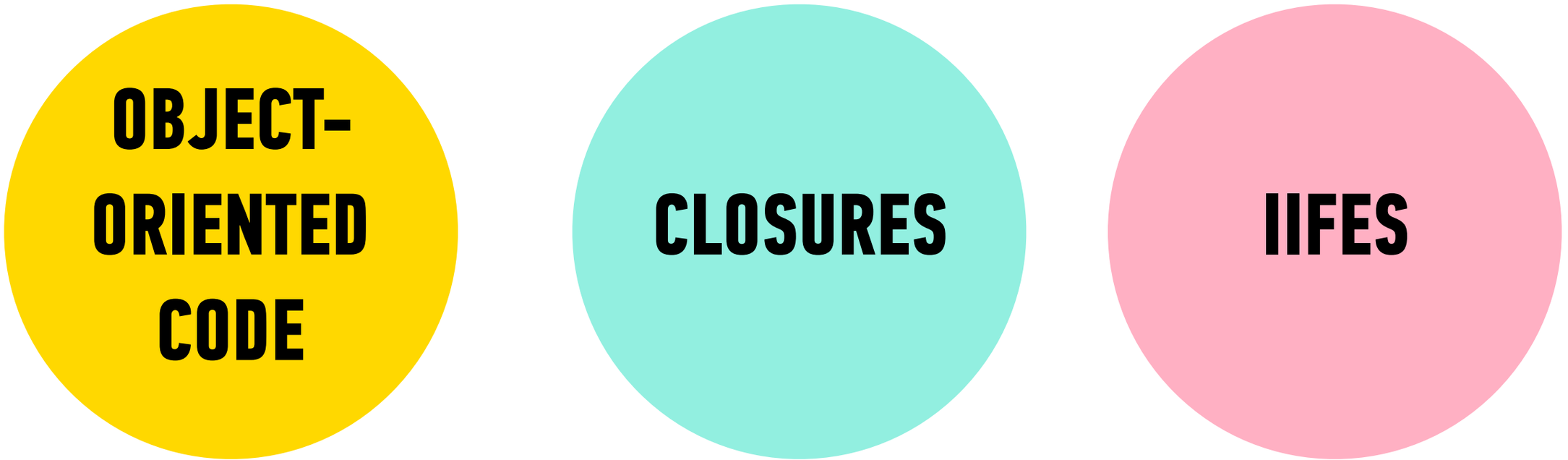
TIMING

5 min

1. Examine the API documentation at https://firebase.google.com/docs/reference/js/firebase.database.Reference#remove
2. Create a function to delete records from the database
3. Add calls to your new function when data is deleted in your app

CONVERTING CODE TO A MODULE

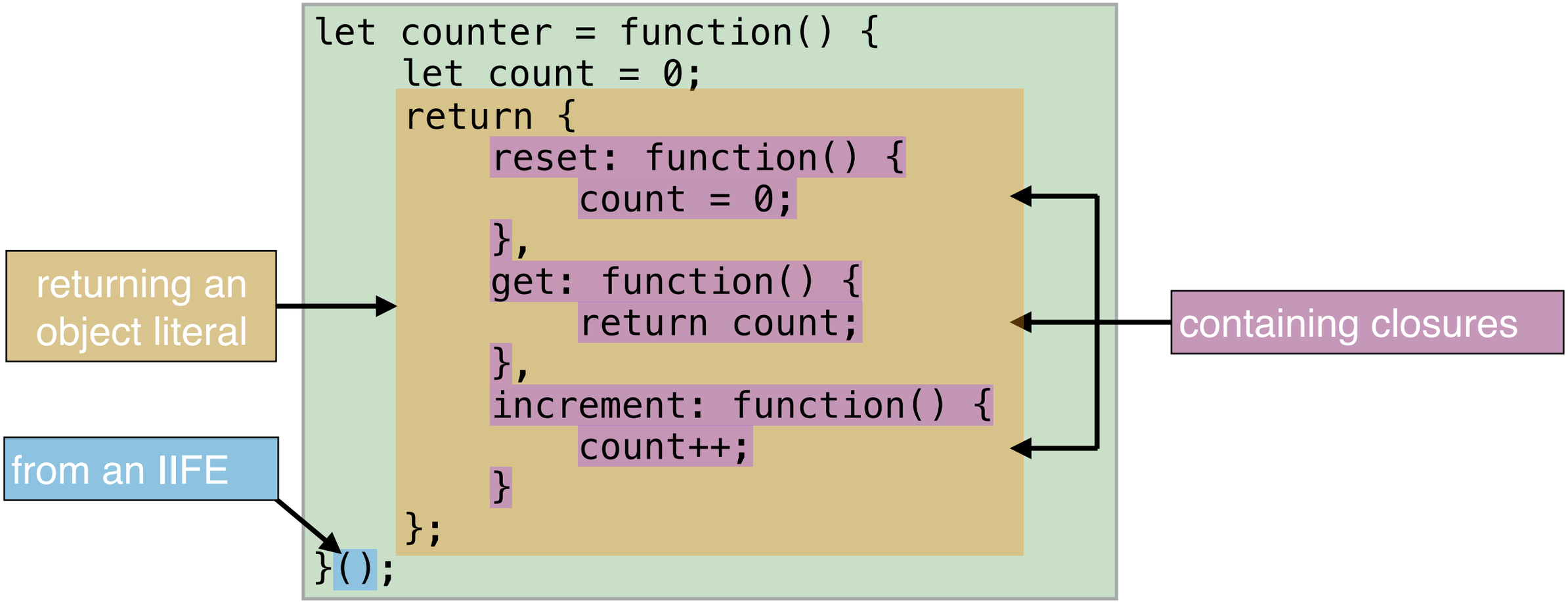
BUILDING BLOCKS OF A MODULE



THE MODULE PATTERN

- Using an IIFE to return an object literal
- The methods of the returned object can access the private properties and methods of the IIFE (closures!), but other code cannot do this
- This means specific parts of the IIFE are not available in the global scope

BUILDING A MODULE



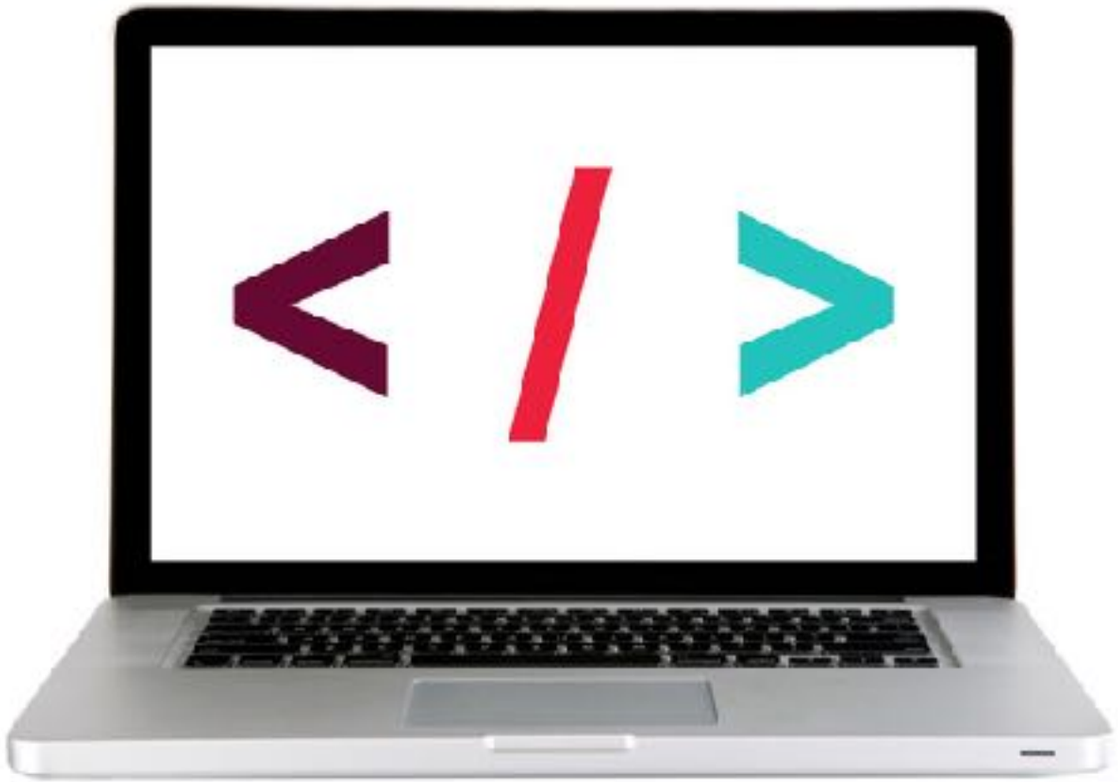
BENEFITS OF THE MODULE PATTERN

- Keeps some functions and variables private
- Avoids polluting the global scope
- Organizes code into objects

CREATING A REVEALING MODULE

- Group variables and functions within an IIFE
- Export an object from the IIFE containing properties and/or methods that are aliases for variables and/or functions within the IIFE
- Change any references to the variables and functions outside of the IIFE to use object notation

LET'S TAKE A CLOSER LOOK



EXERCISE — CONVERT CODE TO A MODULE

KEY OBJECTIVE

- Convert the code for your CRUD app to use the module pattern

TYPE OF EXERCISE

- Solo or in pairs

TIMING

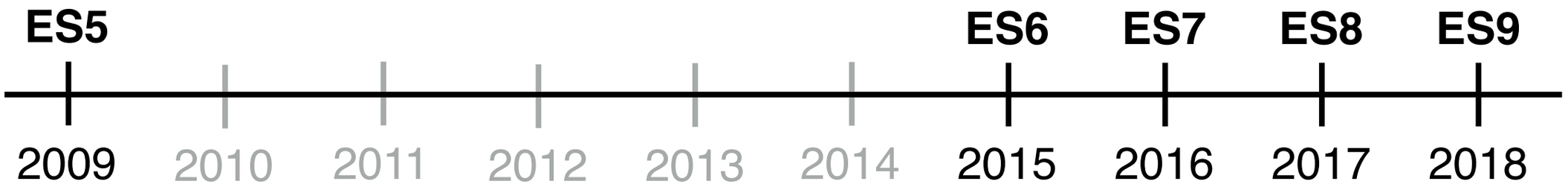
5 min

1. In your completed code from Monday (or the start files for today folder `3-module-exercise`), open `app.js` in your editor
2. Create a new variable called `messageClass`. Its value should be an IIFE that contains the code for the `getPosts()`, `updateMessage()`, and `deleteMessage()` functions, and returns an object containing a method that provides access to the `getPosts()` function.
3. In the `$(document).ready()` code, change the `getPosts()` call to instead call the new method you created.
4. Test your app and make sure all its functionality still works.

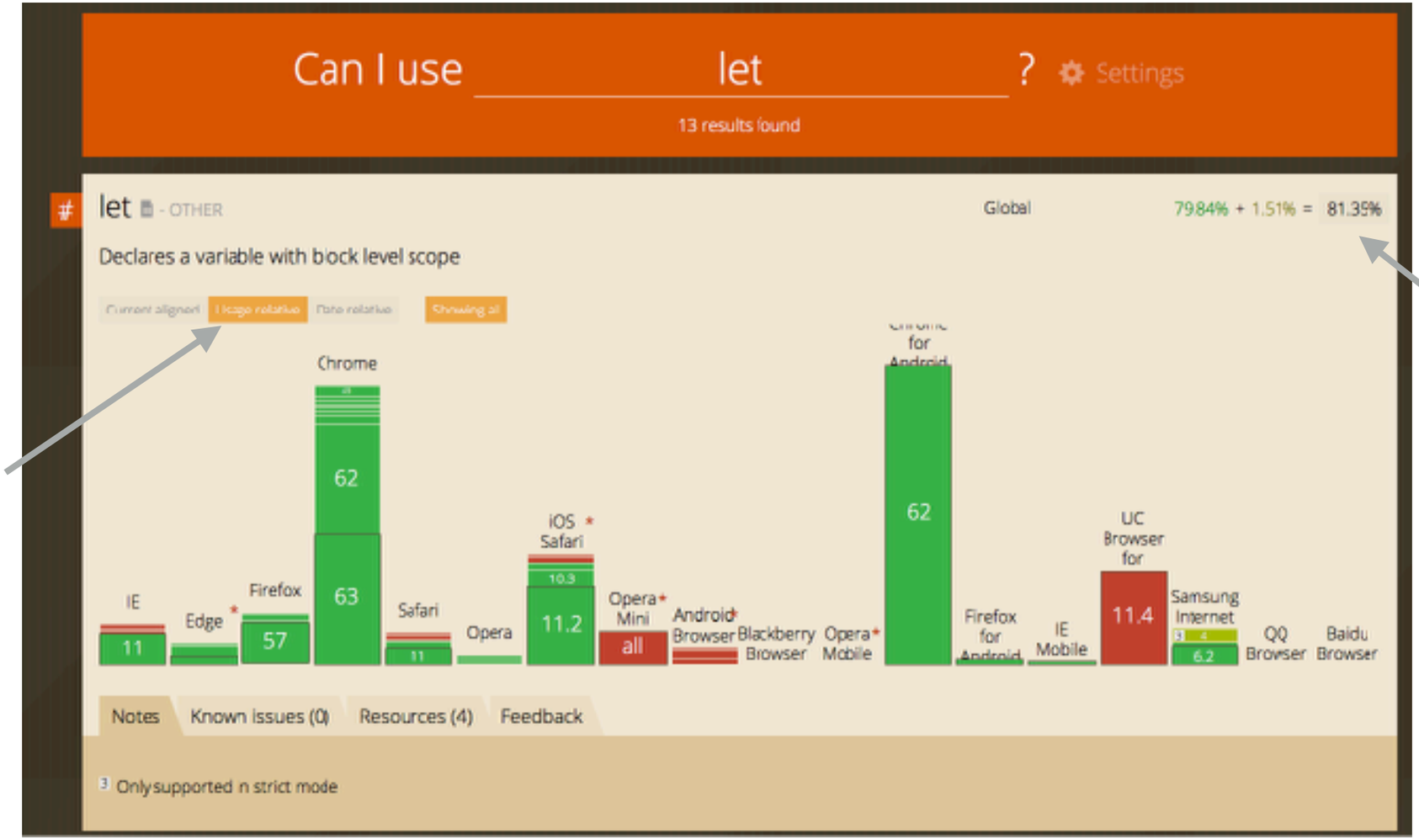
TRANSPILING WITH BABEL

virtually all browsers
in use support ES5

only modern browsers
support ES6+



caniuse.com



"Usage relative" option shows proportional graph

Estimated percent of global browser traffic that can parse this feature

Transpiling involves rewriting code that uses ES6+ features to produce the same result using ES5 code

ES6

```
const taxRate = 0.0875;
let items = [];

let addToCart = () => {
  // do something
}
```

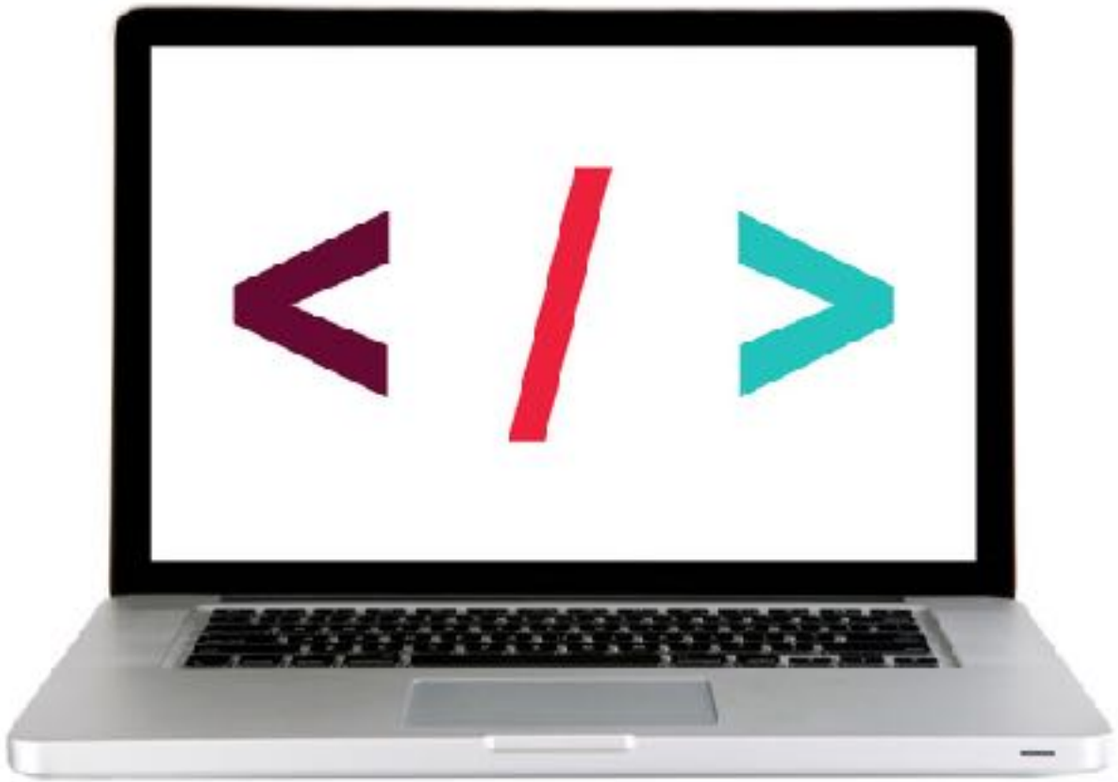
transpiling

ES5

```
var taxRate = 0.0875;
var items = [];

function addToCart() {
  // do something
}
```

LET'S TAKE A CLOSER LOOK



EXERCISE — TRANSPILE CODE USING BABEL

KEY OBJECTIVE

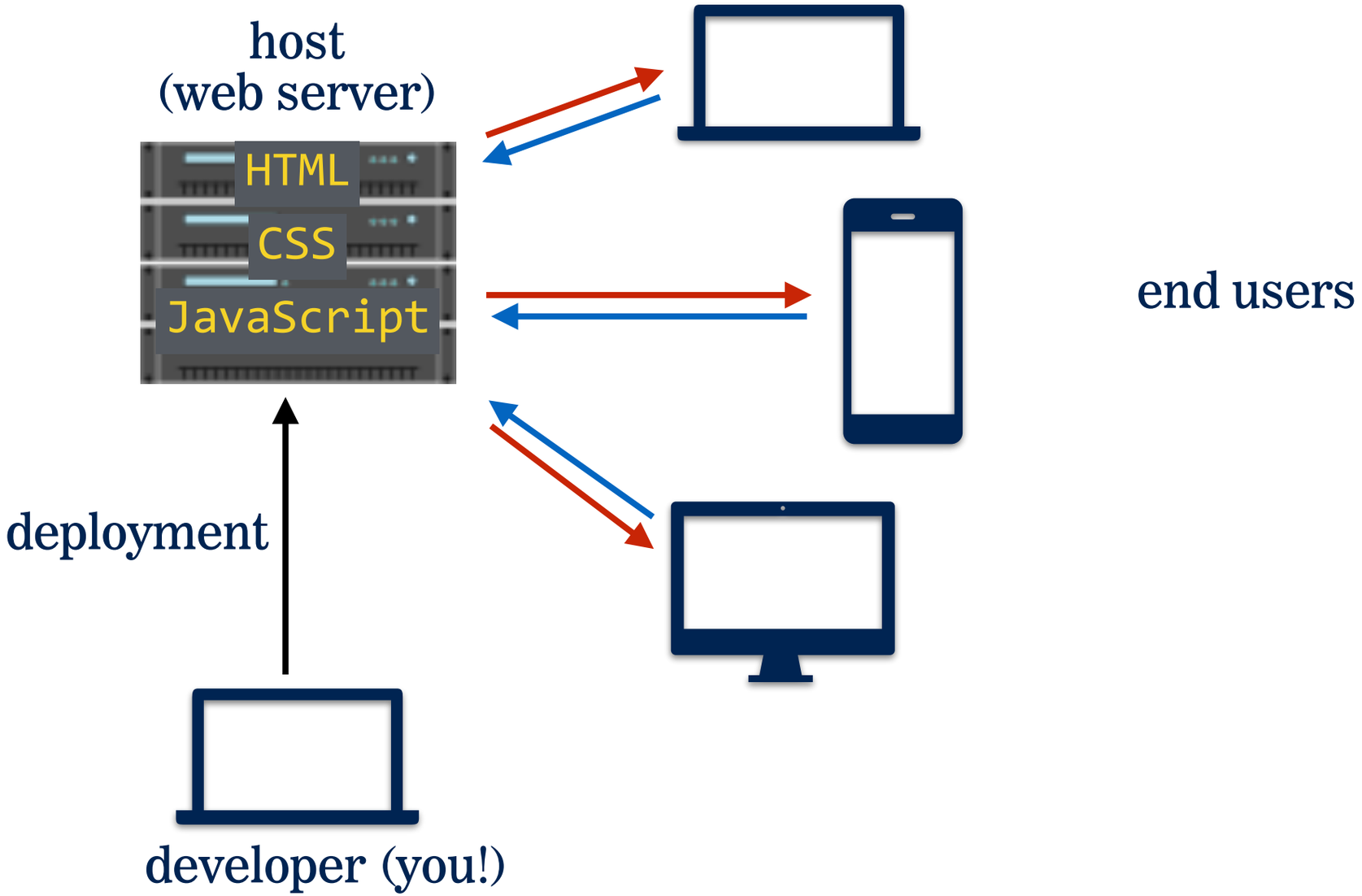
- Ensure backward compatibility by using Babel to transpile code.

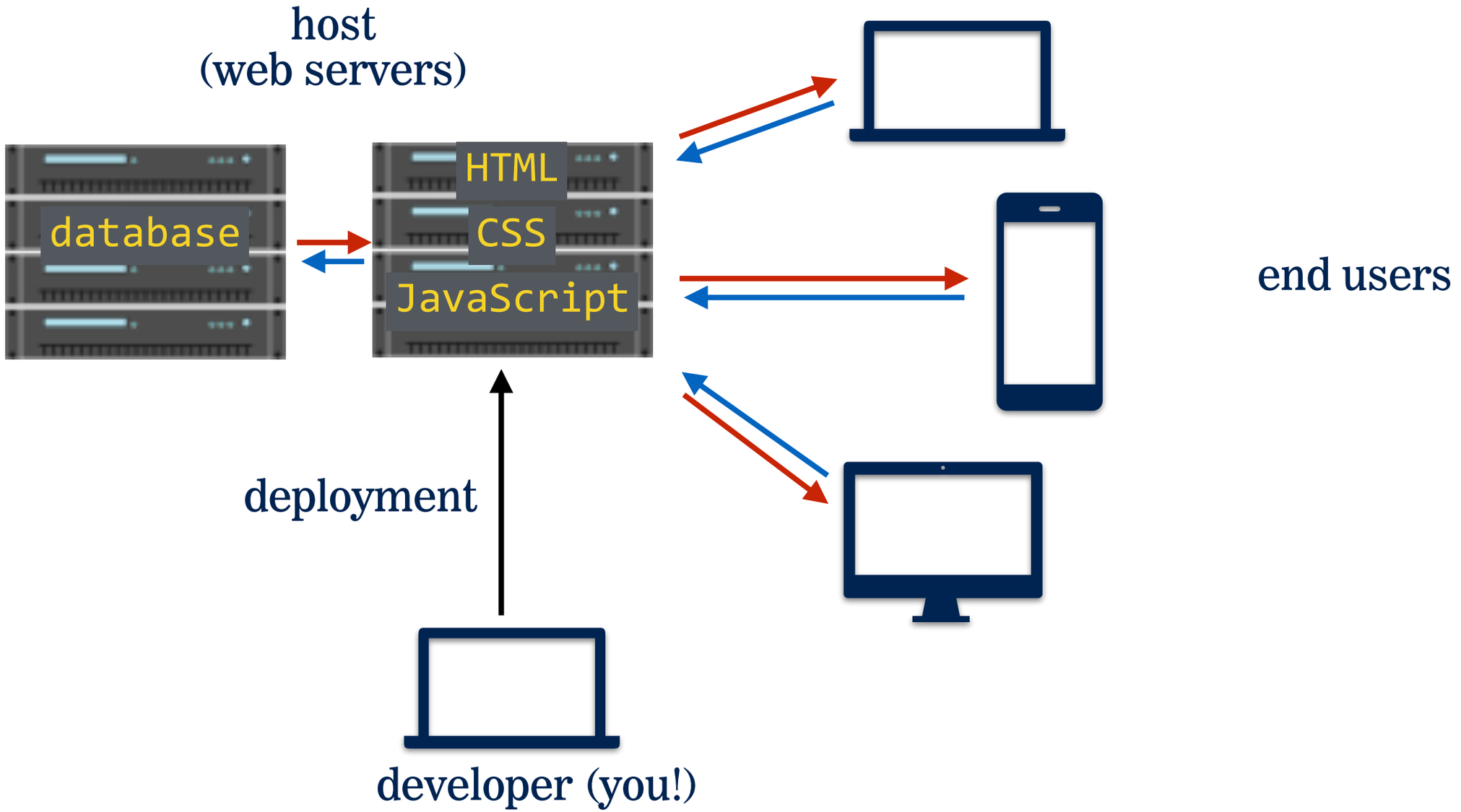
TIMING

5 min

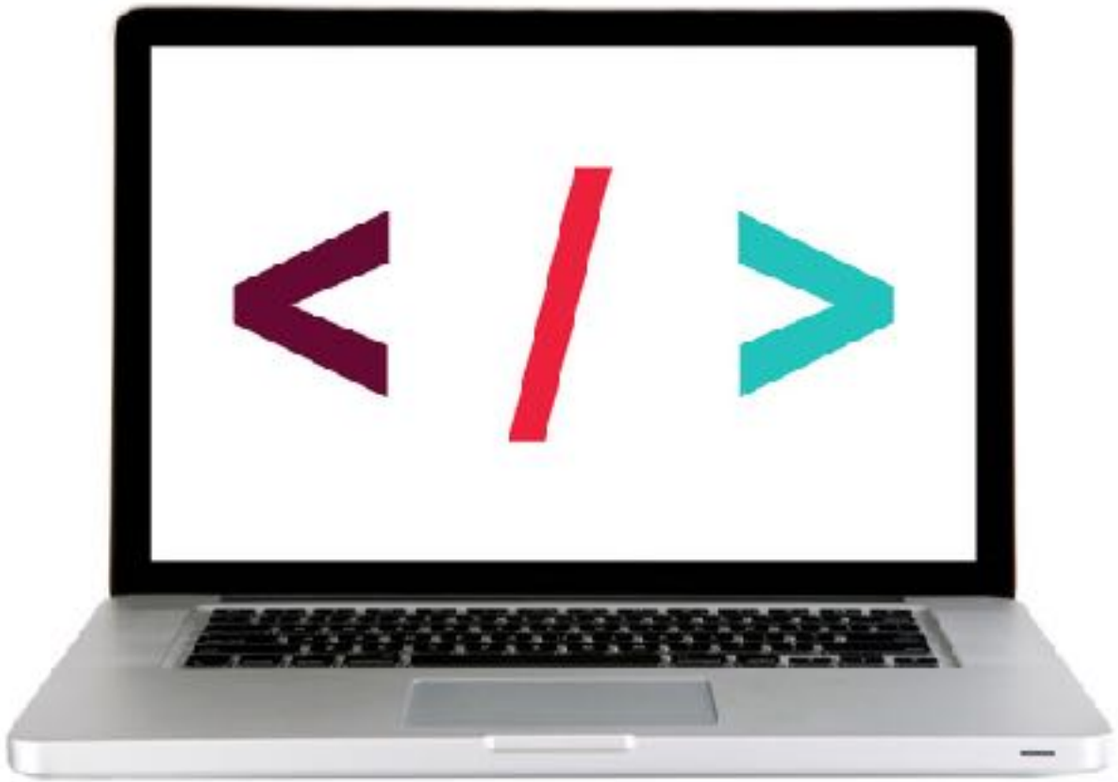
1. Configure Babel for the app you created in class. (If your code isn't quite working, use the code in the `starter-code > 5-transpiling-exercise` folder as a starting point.)
2. Run Babel to create an ES5-compatible version of your code.
3. Open the converted file in your editor and verify the code was transpiled.
4. Test your app in the browser and make sure it still works as it did previously.

DEPLOYMENT


host
(web server)
HTML
CSS
JavaScript
end users
deployment
developer (you!)


host
(web servers)
database
HTML
CSS
JavaScript
end users
deployment
developer (you!)

LET'S TAKE A CLOSER LOOK



EXERCISE — PUSH CHANGES TO FIREBASE

KEY OBJECTIVE

- Deploy to a web host.

TIMING

5 min

1. Make a change to the HTML, CSS, and/or JavaScript for the project you deployed to Firebase.
2. Push your changes to Firebase and verify that your updated code is what you see in your browser at *appname*`.firebaseapp.com`

Exit Tickets!

(Class #16)

LEARNING OBJECTIVES - REVIEW

- Understand what hosting is.
- Identify a program's needs in terms of host providers.
- Ensure backward compatibility by using Babel to transpile code.
- Deploy to a web host.

NEXT CLASS PREVIEW

Intro to React

- Understand the roles of model, view, and controller
- Describe the difference between frameworks and libraries
- Recognize the primary uses of React
- Create a component hierarchy
- Build a React component

Q&A