You make **possible**

JavaScript Essentials

Stève Sfartz, stsfartz@cisco.com
API Architect, DevNet

Alexey Borisenko, balexey@cisco.com
Developer Advocate, DevNet

DEVNET-1444

Barcelona | January 27-31, 2020

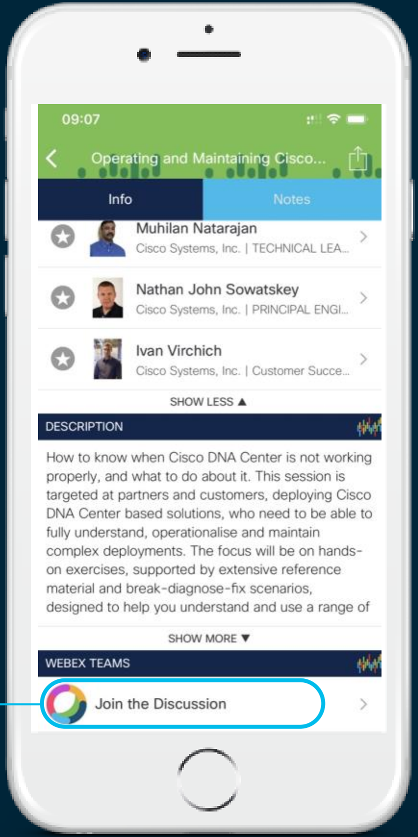
Cisco Webex Teams

Questions?

Use Cisco Webex Teams to chat with the speaker after the session

How

1. Find this session in the Cisco Events Mobile App
2. Click "Join the Discussion"
3. Install Webex Teams or go directly to the team space
4. Enter messages/questions in the team space


09:07
Operating and Maintaining Cisco...
Info | Notes
Muhilan Natarajan
Cisco Systems, Inc. | TECHNICAL LEA...
Nathan John Sowatskey
Cisco Systems, Inc. | PRINCIPAL ENGI...
Ivan Virchich
Cisco Systems, Inc. | Customer Succe...
SHOW LESS ▲
DESCRIPTION
How to know when Cisco DNA Center is not working properly, and what to do about it. This session is targeted at partners and customers, deploying Cisco DNA Center based solutions, who need to be able to fully understand, operationalise and maintain complex deployments. The focus will be on hands-on exercises, supported by extensive reference material and break-diagnose-fix scenarios, designed to help you understand and use a range of
SHOW MORE ▼
WEBEX TEAMS
Join the Discussion

/Cisco/DevNet/SteveSfartz

- API Architect at Cisco DevNet
- Lead for Cisco API Style Guide aiming for simplicity and consistency
- Working to deliver the greatest API Experience for the DevNet community
- Webex Teams & Devices API
- Contributor to DevNet CodeExchange
  - code samples, developer tools, postman collections, awesome-webex, awesome-xapi…

mailto: stsfartz@cisco.com
github: ObjectIsAdvantag
twitter: @SteveSfartz

"vision without execution is hallucination"

Agenda

- About JavaScript
- Server-side JavaScript
- Front-end JavaScript
- Taking JavaScript to the next stage

About JavaScript

About JavaScript

- 1995: created to add dynamic behaviors for Web pages
- Built in 10 days for Netscape Navigator 2.0 release
- Very simple core API
- Lots of flavors (ES5, ES6, coffeescript, flow,typescript)
- Large ecosystem (libraries, npm)
- Javascript is ubiquitous

JavaScript is Ubiquitous

Web Apps

Desktop Apps

Mobile Apps

JavaScript is Ubiquitous

Web Apps

Desktop Apps

Mobile Apps

APIs, Proxys

CLI

JavaScript is Ubiquitous

Web Apps

Desktop Apps

Mobile Apps

Extensibility

APIs, Proxys

CLI

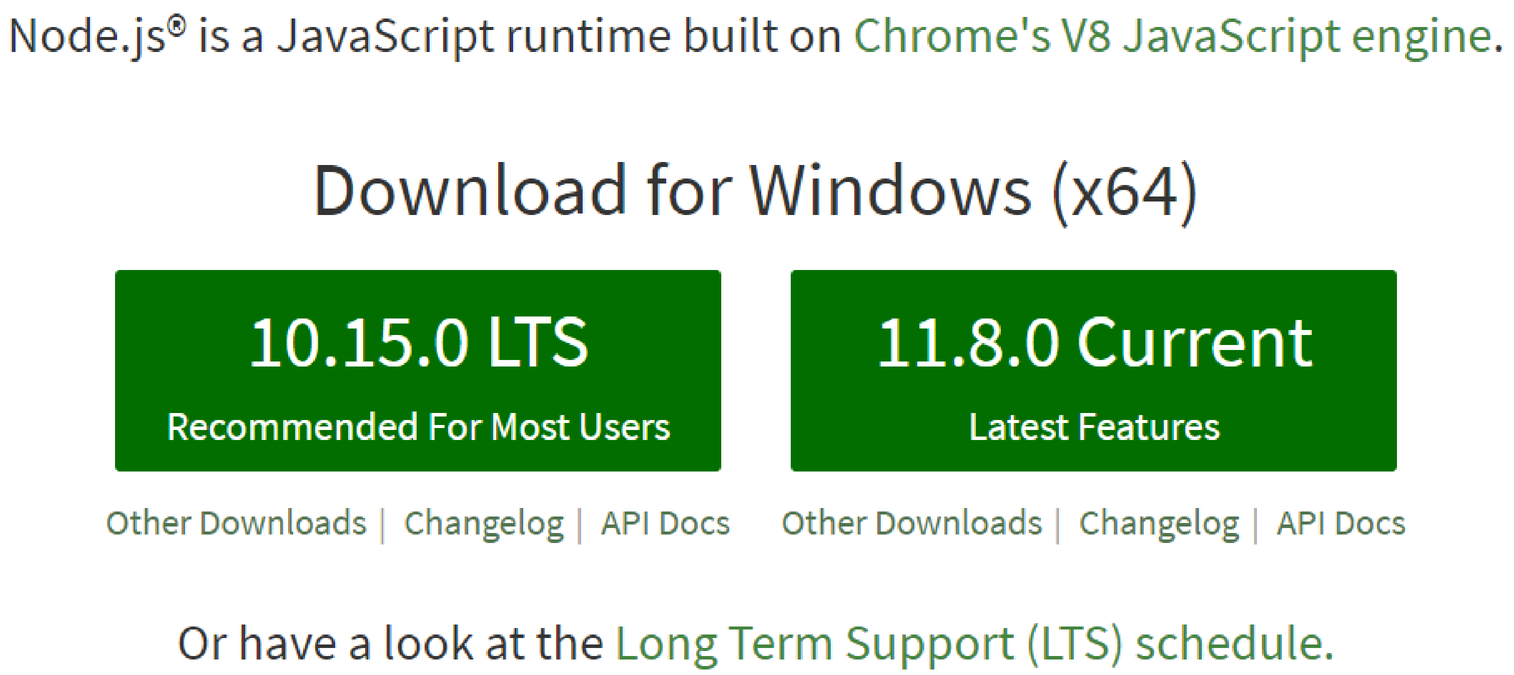
Server-Side Javascript (Node.js)

What is Node.js?

- Node.js® is a JavaScript runtime built on Chrome's V8 JavaScript engine (https://v8.dev)
    - V8 is Google's open source high-performance JavaScript and WebAssembly engine, written in C++.
    - Runs on Windows 7 or later, MacOS 10.12+, and Linux systems
    - Can run standalone, or can be embedded into any C++ application.
    - V8 is used in Chrome and in Node.js.
- Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient.

Node.js runtime

https://nodejs.org/en/



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

Download for Windows (x64)

10.15.0 LTS
Recommended For Most Users

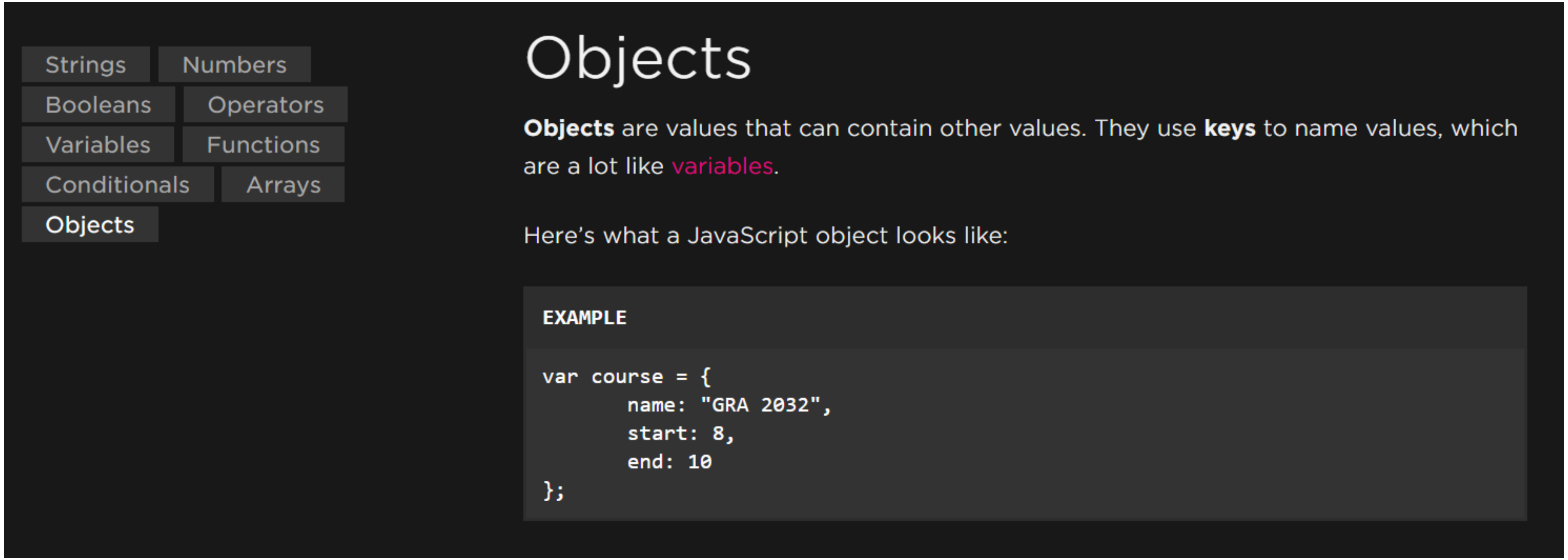
11.8.0 Current
Latest Features

Other Downloads | Changelog | API Docs

Other Downloads | Changelog | API Docs

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

https://www.javascript.com/learn/objects



Objects

Objects are values that can contain other values. They use **keys** to name values, which are a lot like variables.

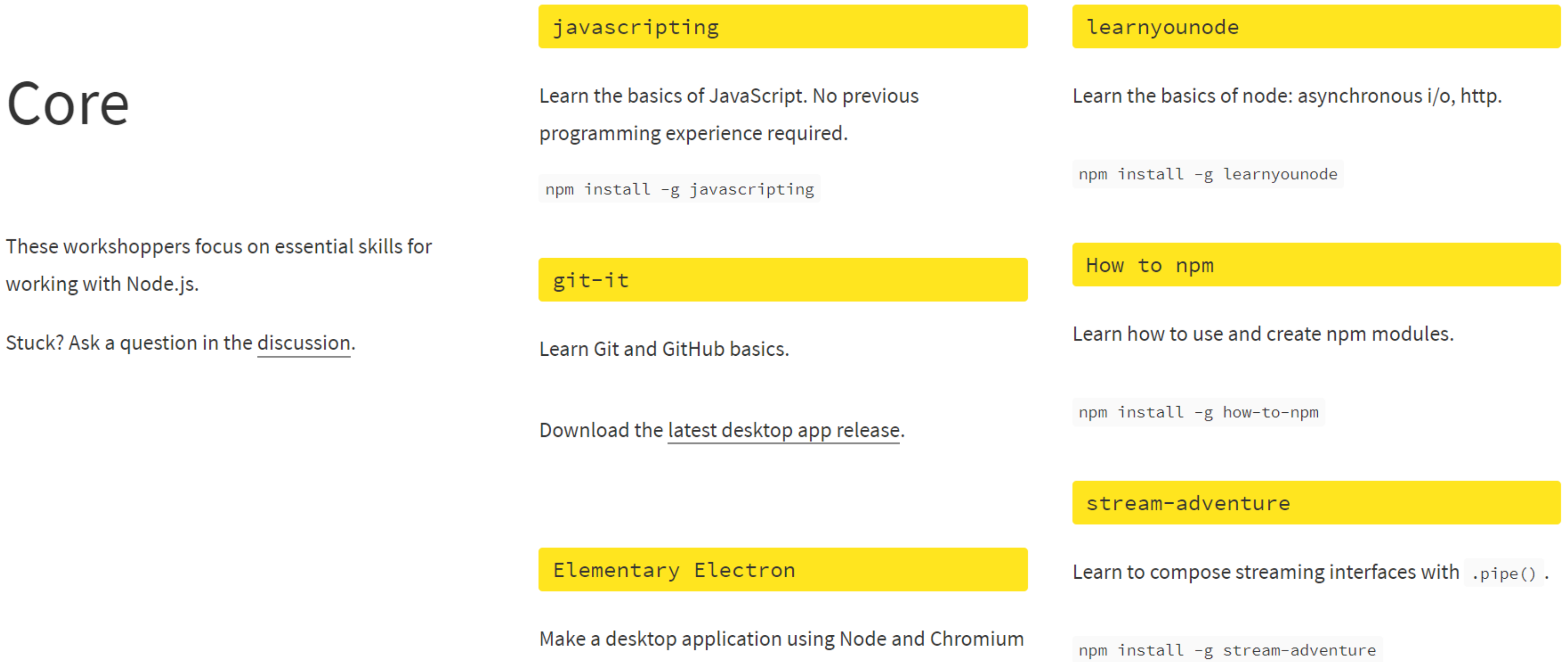
Here's what a JavaScript object looks like:

EXAMPLE

```
var course = {
        name: "GRA 2032",
        start: 8,
        end: 10
};
```

NodeSchool

https://nodeschool.io



Core

These workshoppers focus on essential skills for working with Node.js.

Stuck? Ask a question in the discussion.

javascripting

Learn the basics of JavaScript. No previous programming experience required.

```
npm install -g javascripting
```

git-it

Learn Git and GitHub basics.

Download the latest desktop app release.

Elementary Electron

Make a desktop application using Node and Chromium

learnyounode

Learn the basics of node: asynchronous i/o, http.

```
npm install -g learnyounode
```

How to npm

Learn how to use and create npm modules.

```
npm install -g how-to-npm
```

stream-adventure

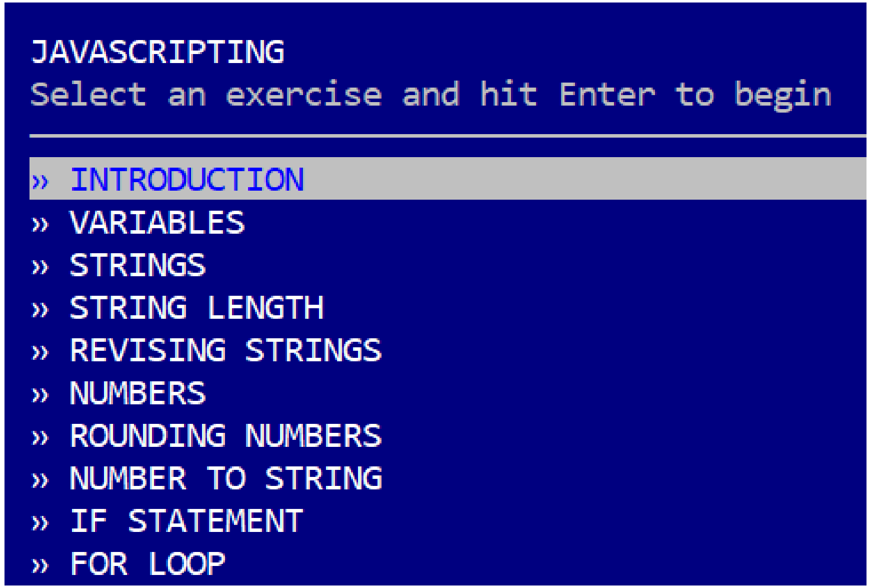
Learn to compose streaming interfaces with `.pipe()`.

```
npm install -g stream-adventure
```

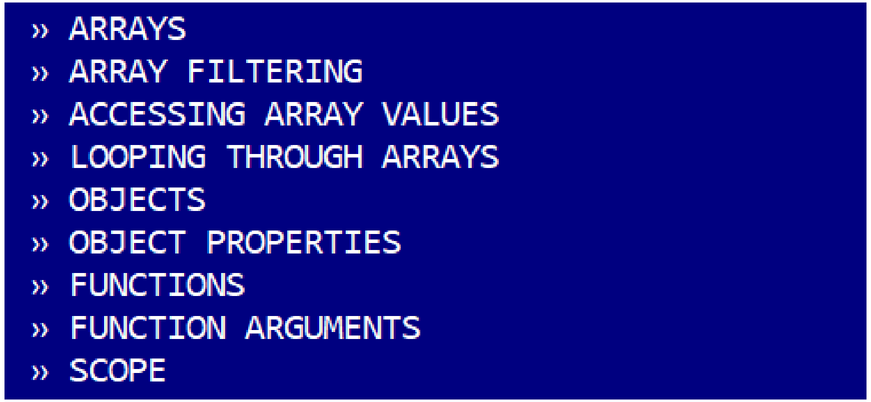
NodeSchool - javascripting

https://github.com/workshopper/javascripting

- Learn the basics of the language
- npm install --global javascripting



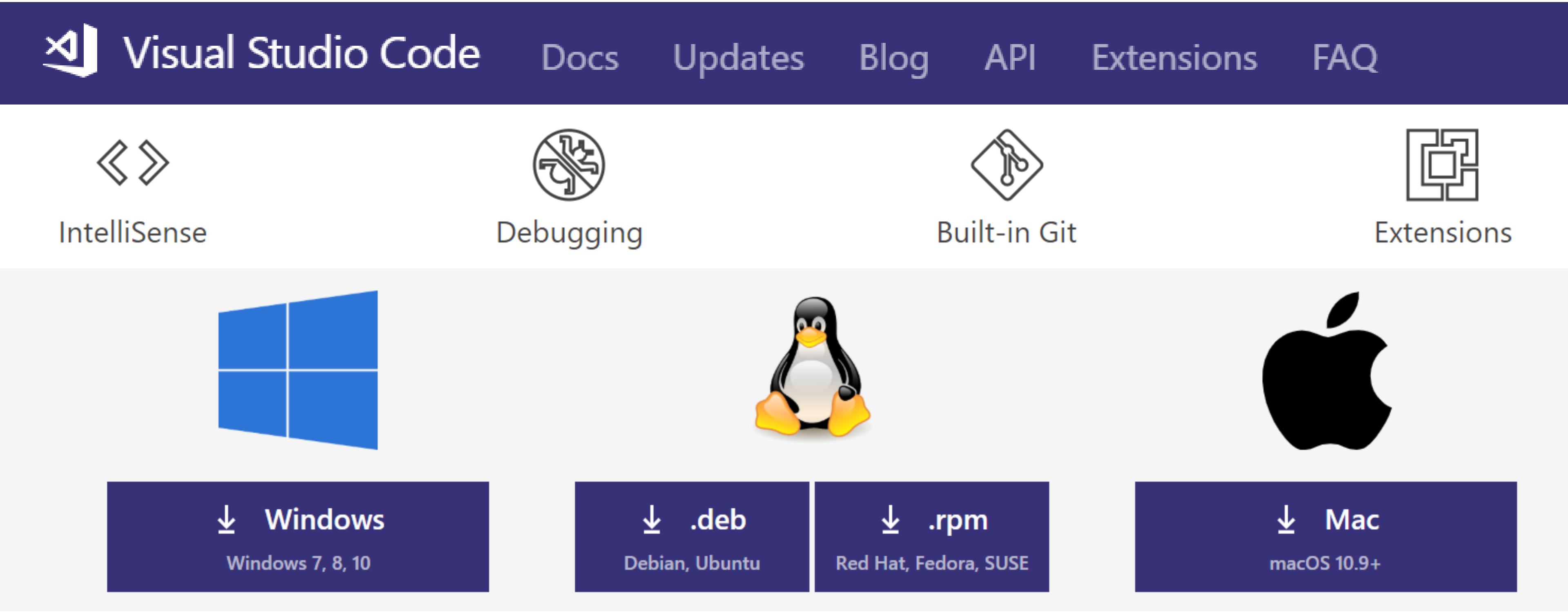
```
JAVASCRIPTING
Select an exercise and hit Enter to begin

» INTRODUCTION
» VARIABLES
» STRINGS
» STRING LENGTH
» REVISING STRINGS
» NUMBERS
» ROUNDING NUMBERS
» NUMBER TO STRING
» IF STATEMENT
» FOR LOOP
```



```
» ARRAYS
» ARRAY FILTERING
» ACCESSING ARRAY VALUES
» LOOPING THROUGH ARRAYS
» OBJECTS
» OBJECT PROPERTIES
» FUNCTIONS
» FUNCTION ARGUMENTS
» SCOPE
```

Building Node.js applications

https://code.visualstudio.com/



Visual Studio Code   Docs   Updates   Blog   API   Extensions   FAQ

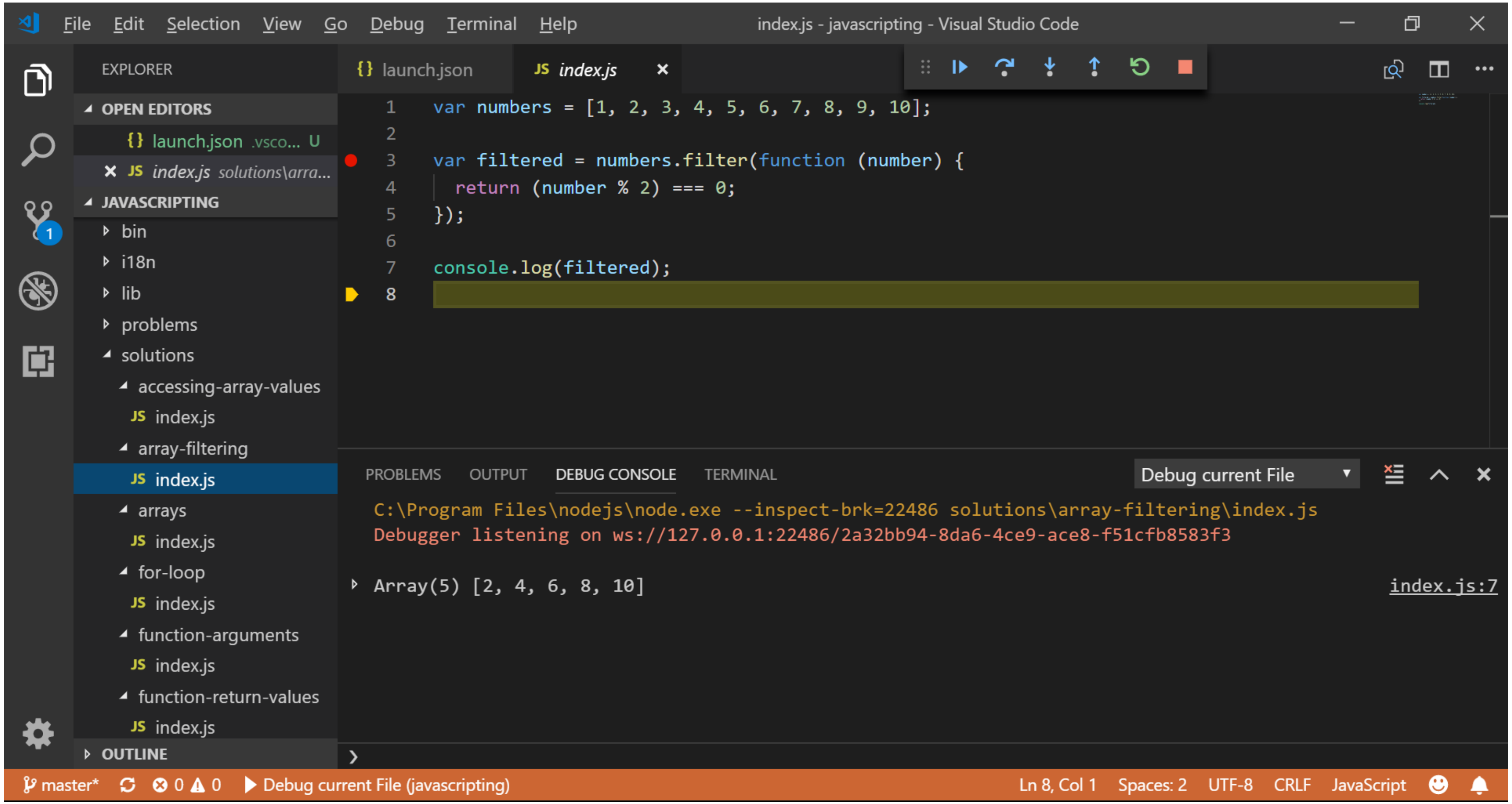
IntelliSense   Debugging   Built-in Git   Extensions

↓ Windows
Windows 7, 8, 10

↓ .deb
Debian, Ubuntu

↓ .rpm
Red Hat, Fedora, SUSE

↓ Mac
macOS 10.9+

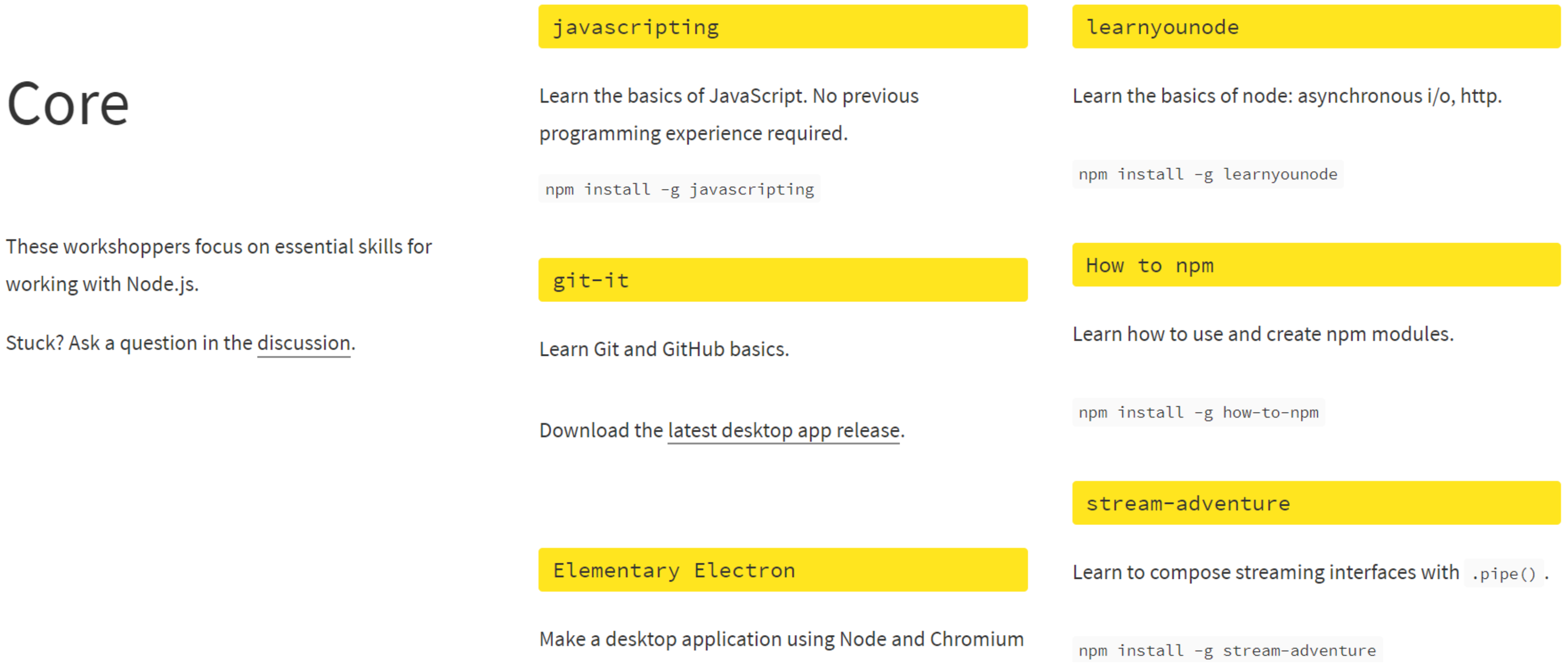


```
var numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10];

var filtered = numbers.filter(function (number) {
  return (number % 2) === 0;
});

console.log(filtered);

```

```
C:\Program Files\nodejs\node.exe --inspect-brk=22486 solutions\array-filtering\index.js
Debugger listening on ws://127.0.0.1:22486/2a32bb94-8da6-4ce9-ace8-f51cfb8583f3

Array(5) [2, 4, 6, 8, 10]                                              index.js:7
```

NodeSchool

https://nodeschool.io



Core

These workshoppers focus on essential skills for working with Node.js.

Stuck? Ask a question in the discussion.

`javascripting`

Learn the basics of JavaScript. No previous programming experience required.

```
npm install -g javascripting
```

`git-it`

Learn Git and GitHub basics.

Download the latest desktop app release.

`Elementary Electron`

Make a desktop application using Node and Chromium

`learnyounode`

Learn the basics of node: asynchronous i/o, http.

```
npm install -g learnyounode
```

`How to npm`

Learn how to use and create npm modules.

```
npm install -g how-to-npm
```

`stream-adventure`

Learn to compose streaming interfaces with `.pipe()`.

```
npm install -g stream-adventure
```

NodeSchool

https://nodeschool.io

`Elementary Electron`

Make a desktop application using Node and Chromium with Electron

```
npm install -g elementary-electron
```

`stream-adventure`

Learn to compose streaming interfaces with `.pipe()`.

```
npm install -g stream-adventure
```

`how-to-markdown`

Learn how to start using Markdown — a lightweight markup language with plain text formatting syntax.

```
npm install -g how-to-markdown
```

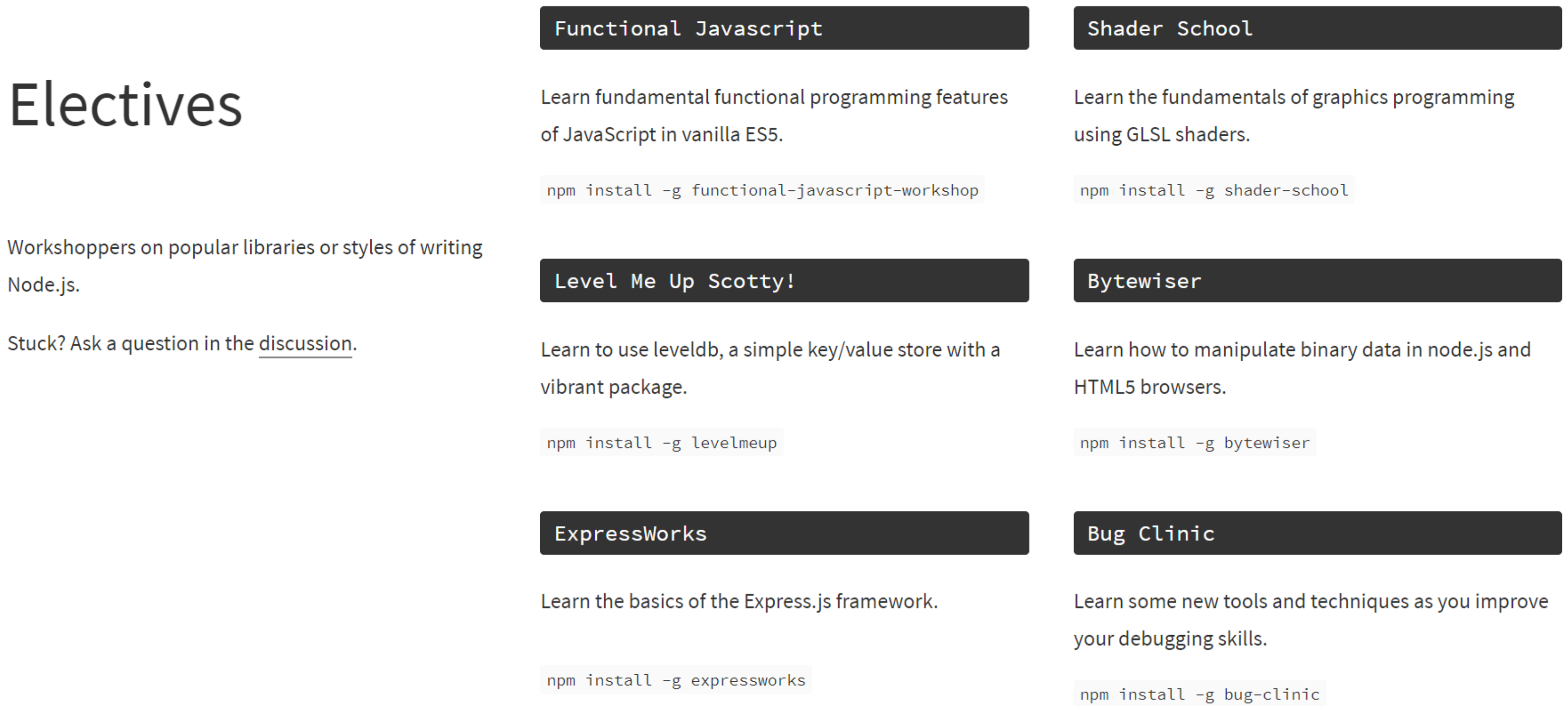
`learnyouhtml`

Learn how to create your first web page.

```
npm install -g learnyouhtml
```

NodeSchool

https://nodeschool.io



Electives

Workshoppers on popular libraries or styles of writing Node.js.

Stuck? Ask a question in the discussion.

Functional Javascript

Learn fundamental functional programming features of JavaScript in vanilla ES5.

```
npm install -g functional-javascript-workshop
```

Shader School

Learn the fundamentals of graphics programming using GLSL shaders.

```
npm install -g shader-school
```

Level Me Up Scotty!

Learn to use leveldb, a simple key/value store with a vibrant package.

```
npm install -g levelmeup
```

Bytewiser

Learn how to manipulate binary data in node.js and HTML5 browsers.

```
npm install -g bytewiser
```

ExpressWorks

Learn the basics of the Express.js framework.

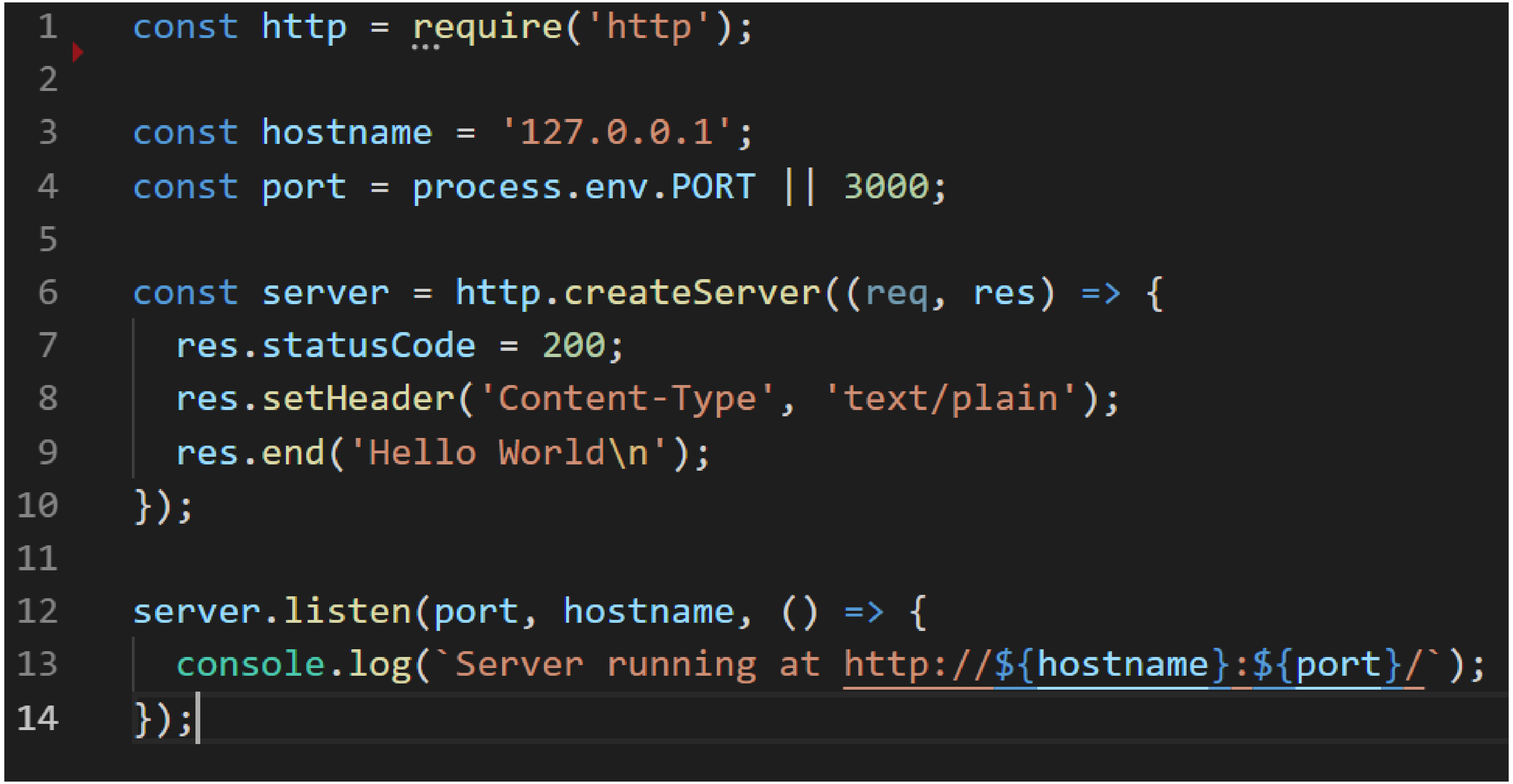
```
npm install -g expressworks
```

Bug Clinic

Learn some new tools and techniques as you improve your debugging skills.

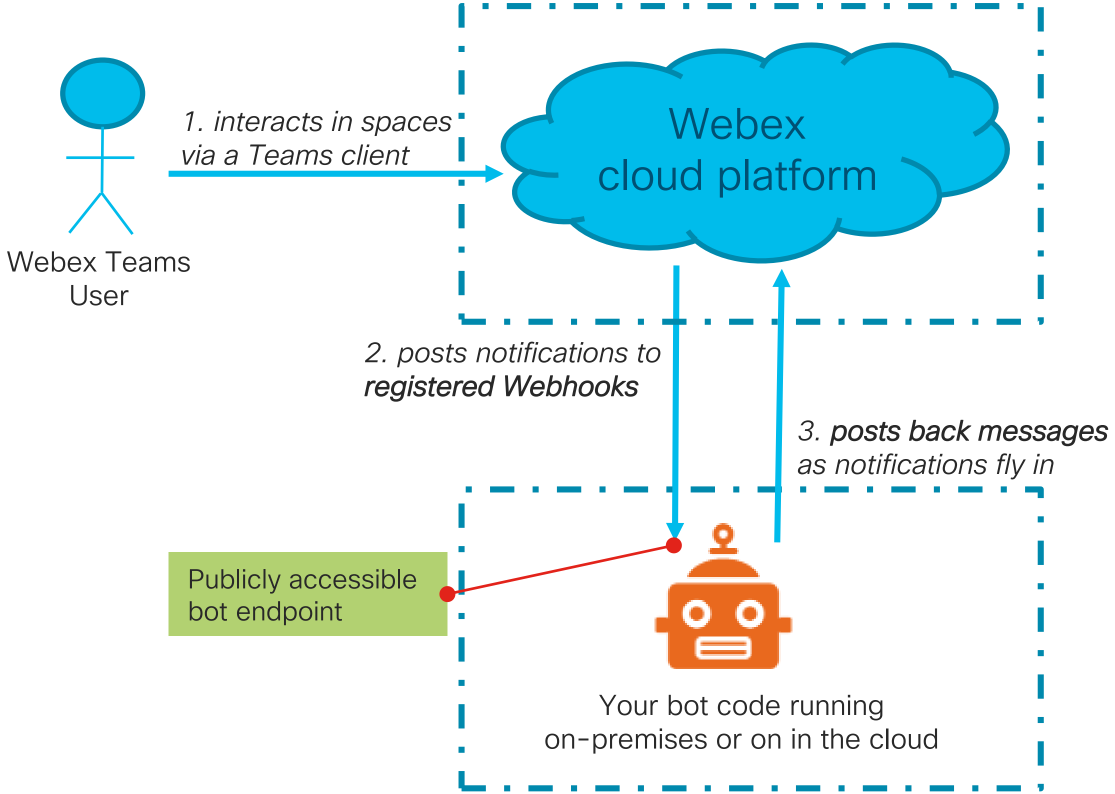
```
npm install -g bug-clinic
```

Simple HTTP Server



```javascript
const http = require('http');

const hostname = '127.0.0.1';
const port = process.env.PORT || 3000;

const server = http.createServer((req, res) => {
  res.statusCode = 200;
  res.setHeader('Content-Type', 'text/plain');
  res.end('Hello World\n');
});

server.listen(port, hostname, () => {
  console.log(`Server running at http://${hostname}:${port}/`);
});
```

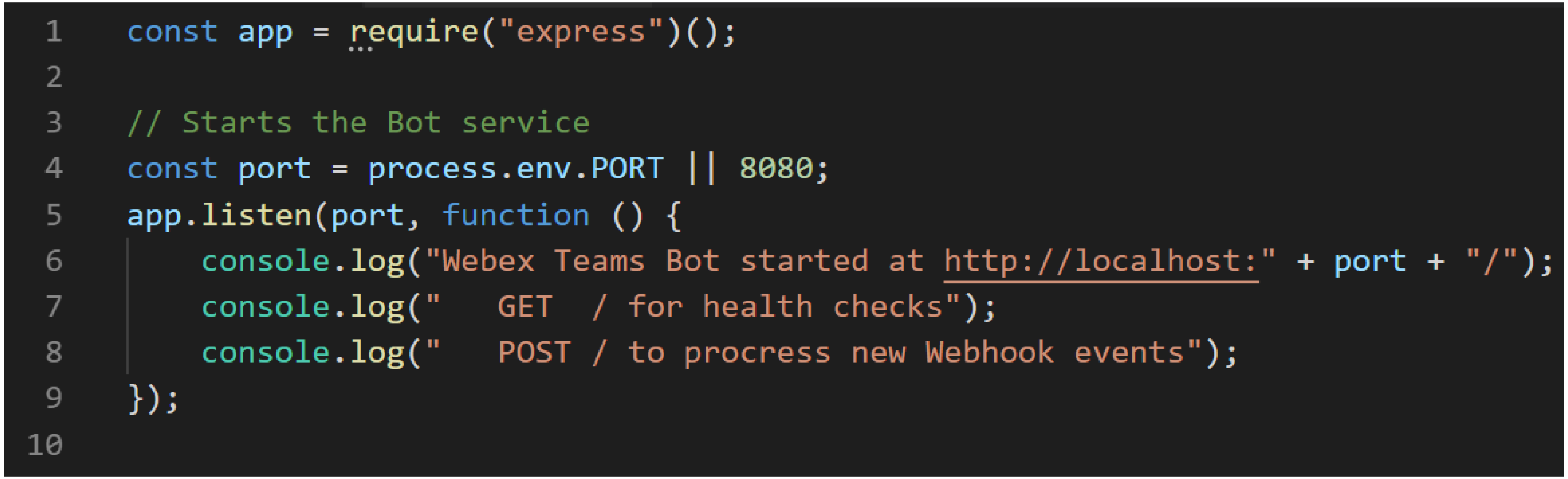
Webex Teams Bot Architecture



Webex Teams User
1. interacts in spaces via a Teams client
Webex cloud platform
2. posts notifications to registered Webhooks
3. posts back messages as notifications fly in
Publicly accessible bot endpoint
Your bot code running on-premises or on in the cloud

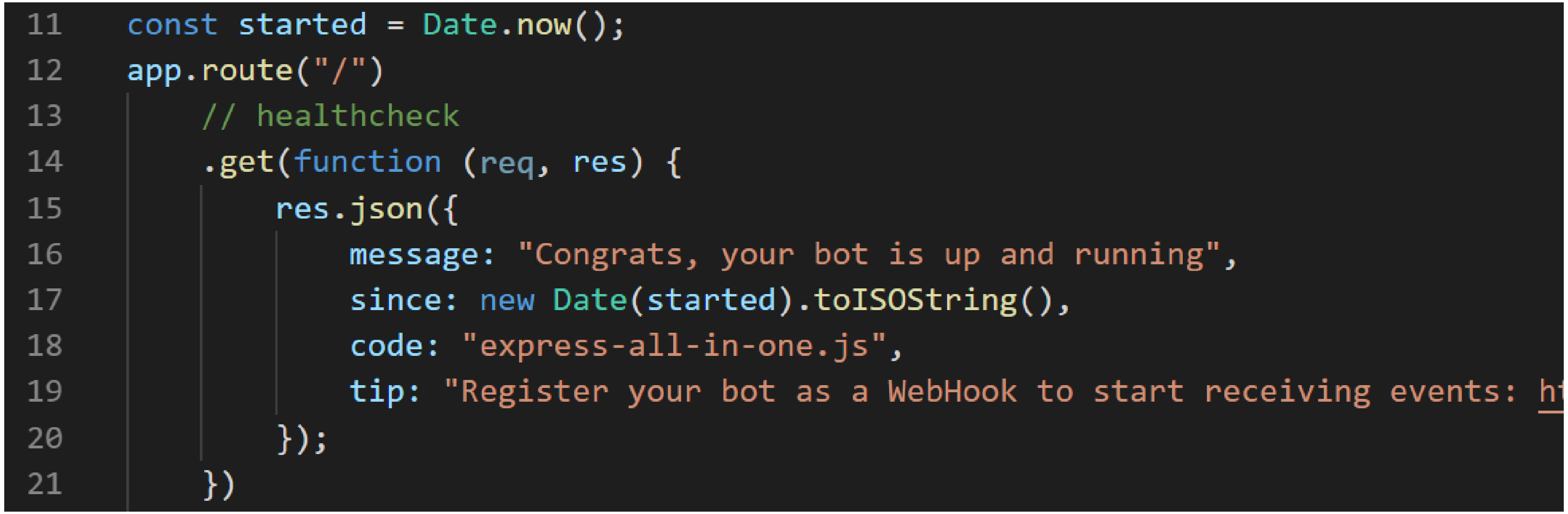
- Register webhook events
  - Messages / created
  - Memberships / created
- As events happen in spaces, receive notifications from Webex
- Security tips
  - Select spaces to fire from via a webhook filter
  - Check on user's email domain in your code
  - Check webhook payload signature via a shared secret

Webex Teams Webhook with Express

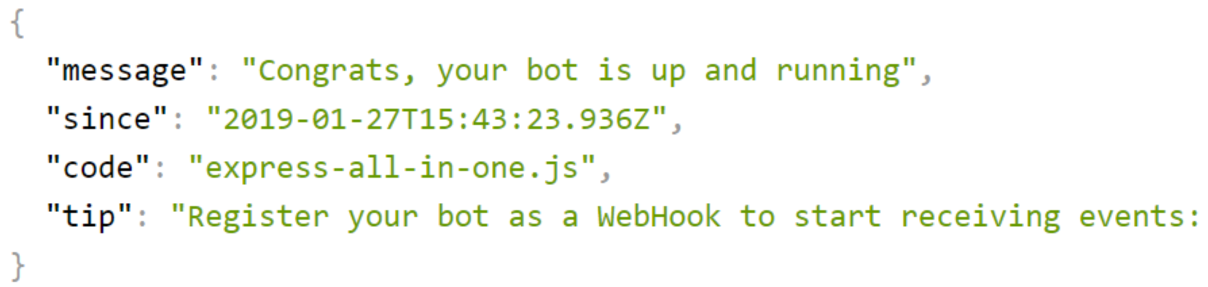


```
const app = require("express")();

// Starts the Bot service
const port = process.env.PORT || 8080;
app.listen(port, function () {
    console.log("Webex Teams Bot started at http://localhost:" + port + "/");
    console.log("   GET  / for health checks");
    console.log("   POST / to procress new Webhook events");
});

```

Webex Teams Webhook with Express

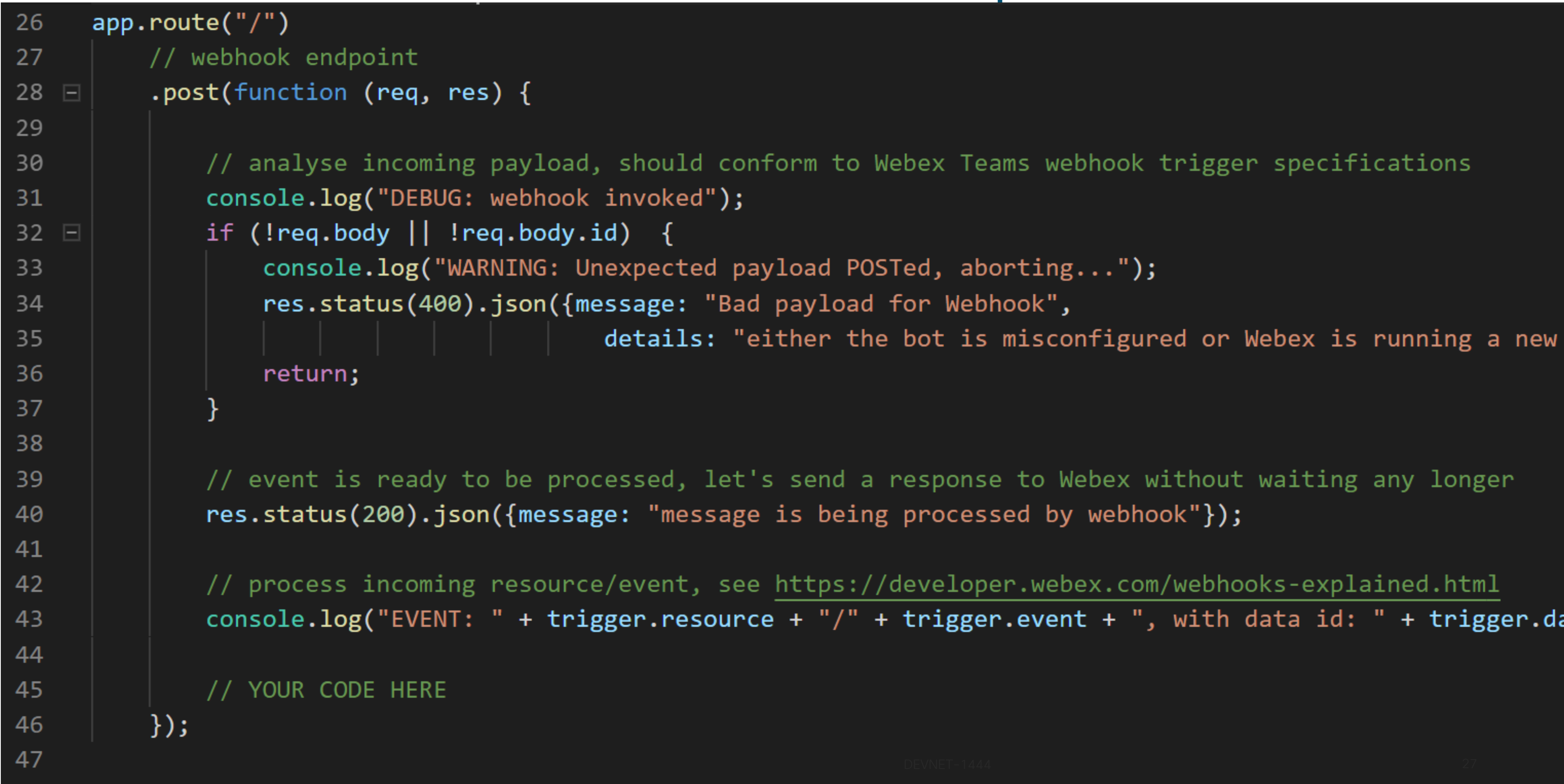


```
const started = Date.now();
app.route("/")
    // healthcheck
    .get(function (req, res) {
        res.json({
            message: "Congrats, your bot is up and running",
            since: new Date(started).toISOString(),
            code: "express-all-in-one.js",
            tip: "Register your bot as a WebHook to start receiving events: ht
        });
    })
```



```
{
  "message": "Congrats, your bot is up and running",
  "since": "2019-01-27T15:43:23.936Z",
  "code": "express-all-in-one.js",
  "tip": "Register your bot as a WebHook to start receiving events:
}
```

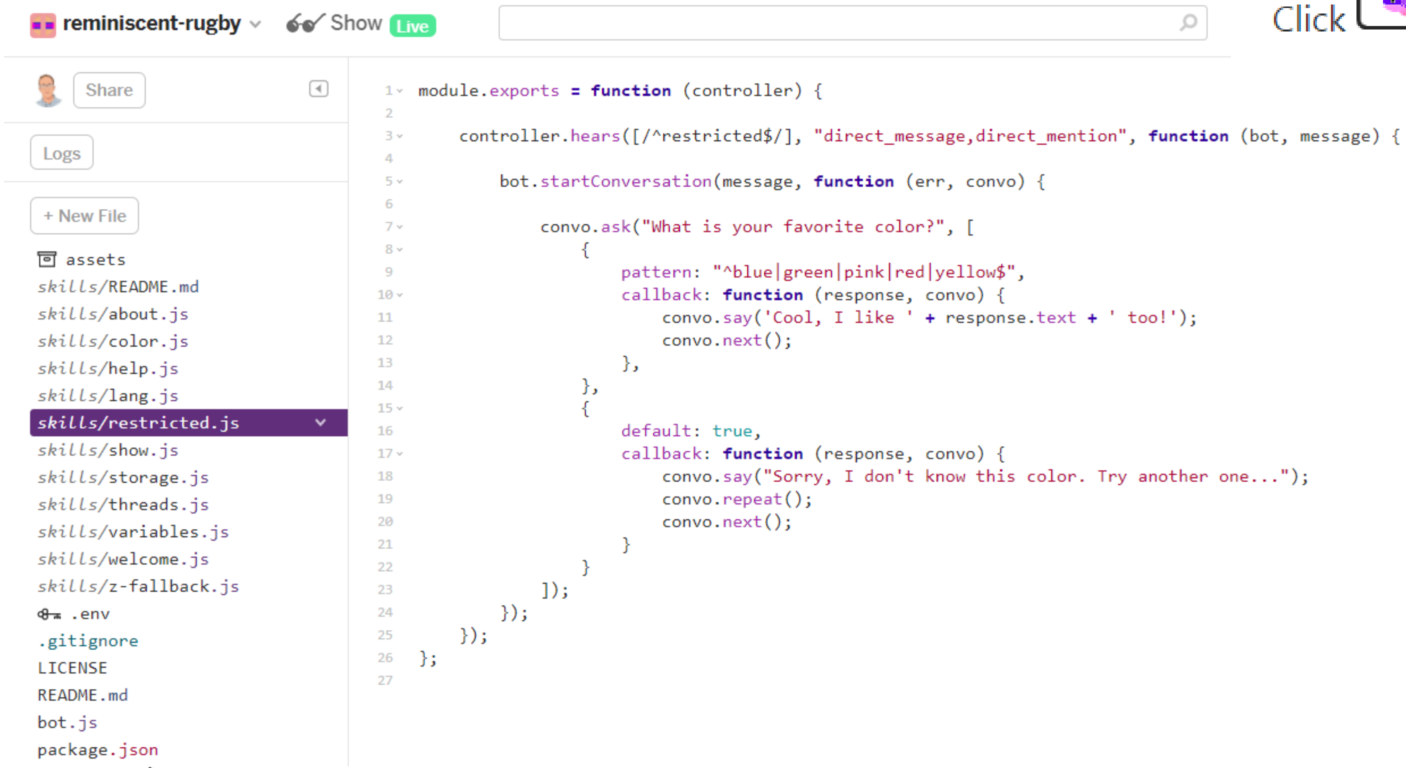
Webex Teams Webhook with Express



```
app.route("/")
    // webhook endpoint
    .post(function (req, res) {

        // analyse incoming payload, should conform to Webex Teams webhook trigger specifications
        console.log("DEBUG: webhook invoked");
        if (!req.body || !req.body.id)  {
            console.log("WARNING: Unexpected payload POSTed, aborting...");
            res.status(400).json({message: "Bad payload for Webhook",
                                    details: "either the bot is misconfigured or Webex is running a new
            return;
        }

        // event is ready to be processed, let's send a response to Webex without waiting any longer
        res.status(200).json({message: "message is being processed by webhook"});

        // process incoming resource/event, see https://developer.webex.com/webhooks-explained.html
        console.log("EVENT: " + trigger.resource + "/" + trigger.event + ", with data id: " + trigger.da

        // YOUR CODE HERE
    });

```

What is takes to code a bot?

https://github.com/CiscoDevNet/botkit-template



```
module.exports = function (controller) {

    controller.hears([/^restricted$/], "direct_message,direct_mention", function (bot, message) {

        bot.startConversation(message, function (err, convo) {

            convo.ask("What is your favorite color?", [
                {
                    pattern: "^blue|green|pink|red|yellow$",
                    callback: function (response, convo) {
                        convo.say('Cool, I like ' + response.text + ' too!');
                        convo.next();
                    },
                },
                {
                    default: true,
                    callback: function (response, convo) {
                        convo.say("Sorry, I don't know this color. Try another one...");
                        convo.repeat();
                        convo.next();
                    }
                }
            ]);
        });
    });
};

```

Quick start on Glitch

Click

Bot skills with Botkit

https://github.com/CiscoDevNet/botkit-template/tree/master/skills

```
module.exports = function (controller) {

    controller.hears([/^color$/], 'direct_message,direct_mention', function (bot, message) {

        bot.startConversation(message, function (err, convo) {
            convo.say('This is a Botkit conversation sample.');

            convo.ask('What is your favorite color?', function (response, convo) {
                convo.say("Cool, I like '" + response.text + "' too!");
                convo.next();
            });
        });
```

CLI Example

https://github.com/ObjectIsAdvantag/webex-guestissuer

```
# Install the CLI
npm install guestissuer -g

# Create a Guest token, and fetch an access token right away (valid for 6 hours)
# Here, the JWT guest token is volatile (neither stored, not returned)
guestissuer quick <userId> <userName> -i <issuerAppId> -s <issuerAppSecret>
```

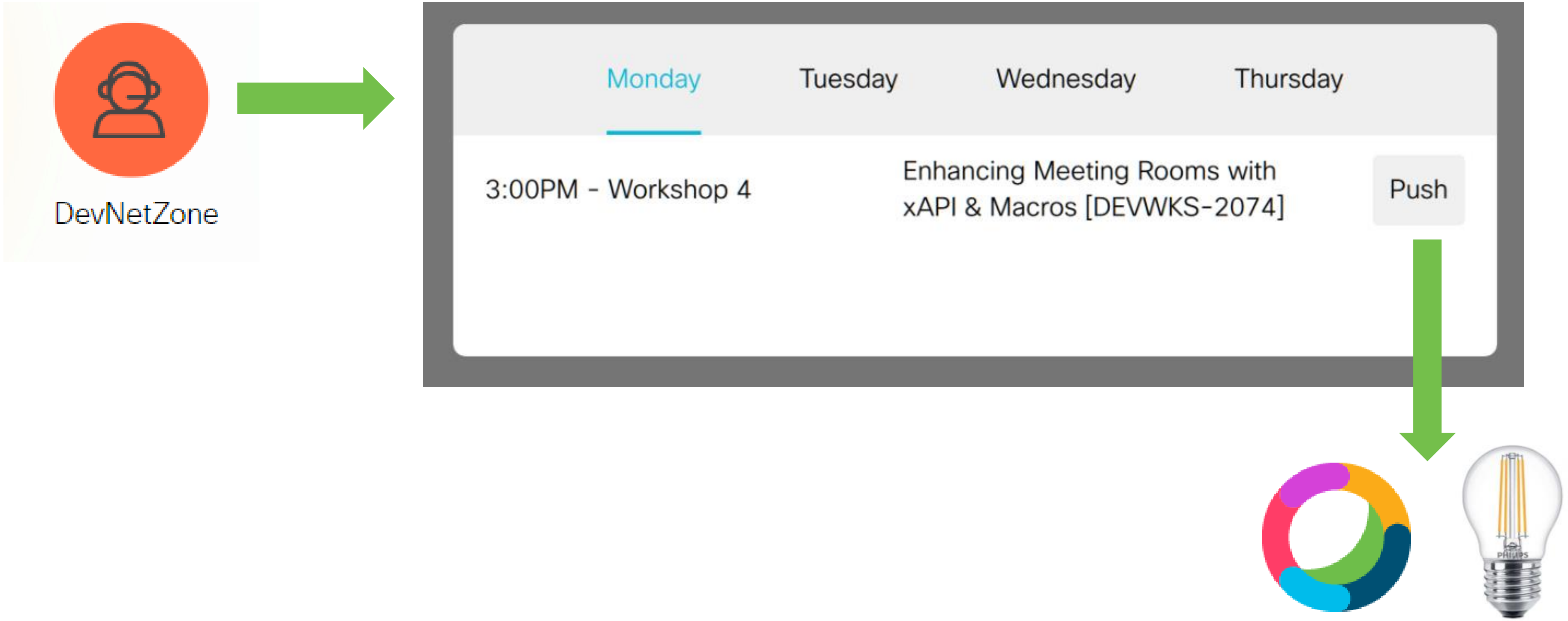
JavaScript is Ubiquitous

Use cases for Server-side JavaScript at Cisco

Server-side

APIs, Proxys	CLI	Extensibility
Webex ChatBots *Webex Emulator*	*Web Guest Issuer CLI*	Webex Macros

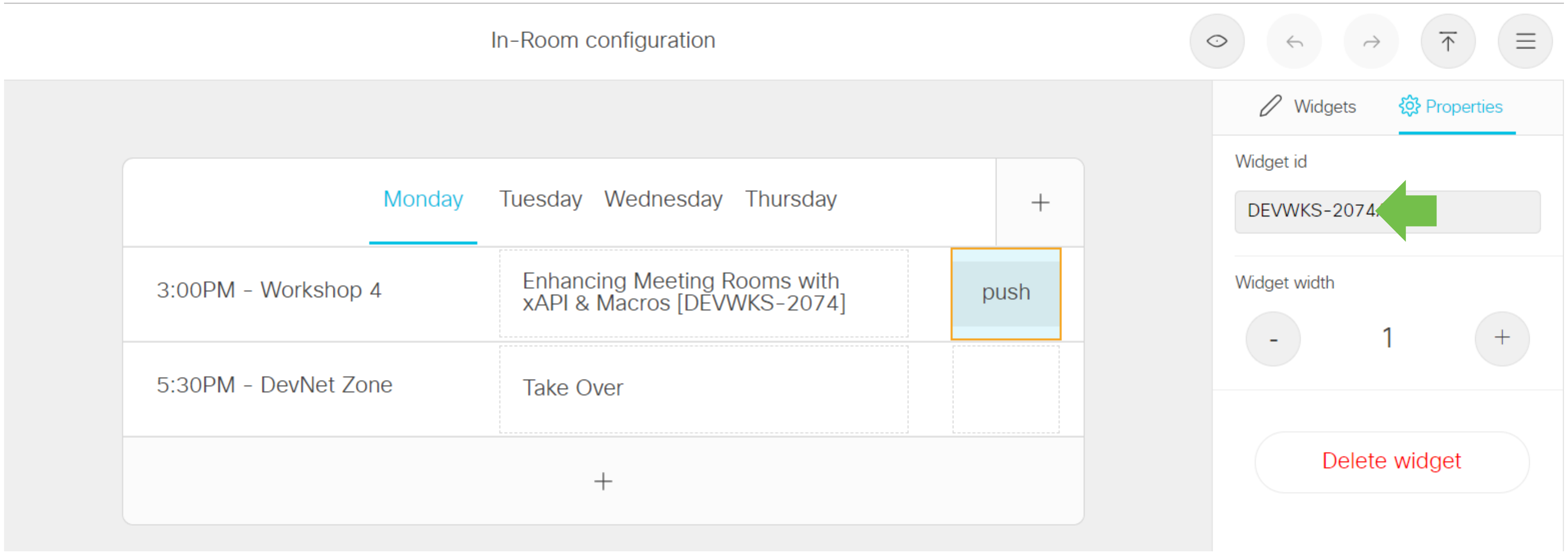
Webex Devices Extensibility



3:00PM - Workshop 4

Enhancing Meeting Rooms with xAPI & Macros [DEVWKS-2074]

Button with widget id: **DEVWKS-2074**



3:00PM - Workshop 4

Enhancing Meeting Rooms with xAPI & Macros [DEVWKS-2074]

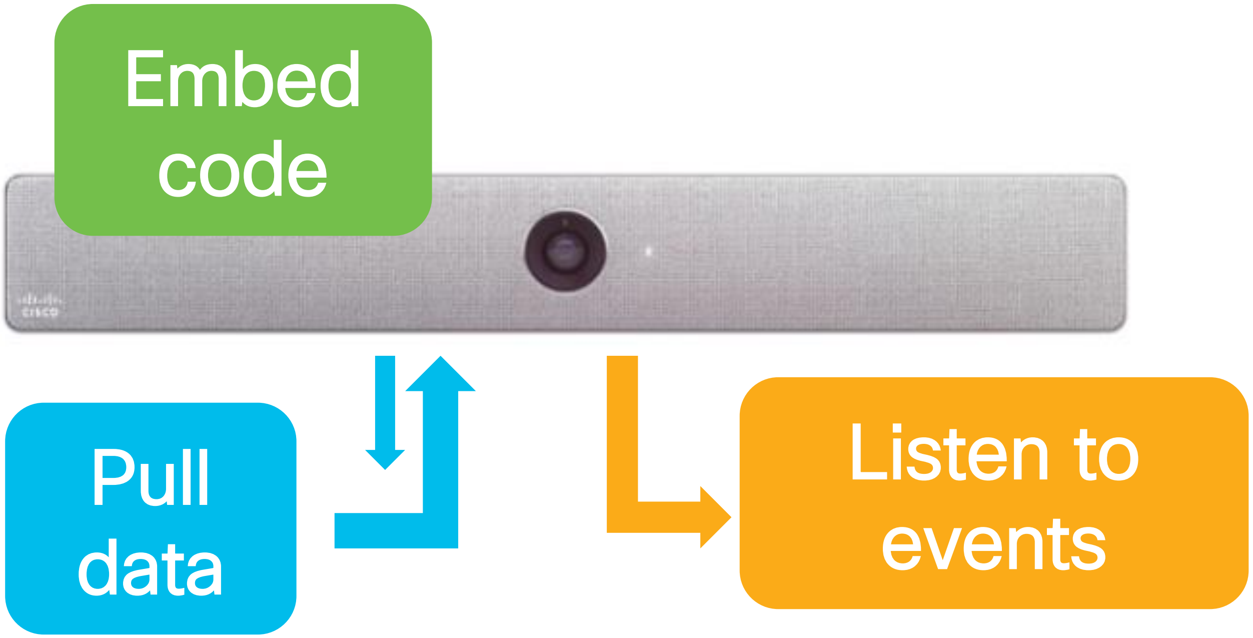
5:30PM - DevNet Zone

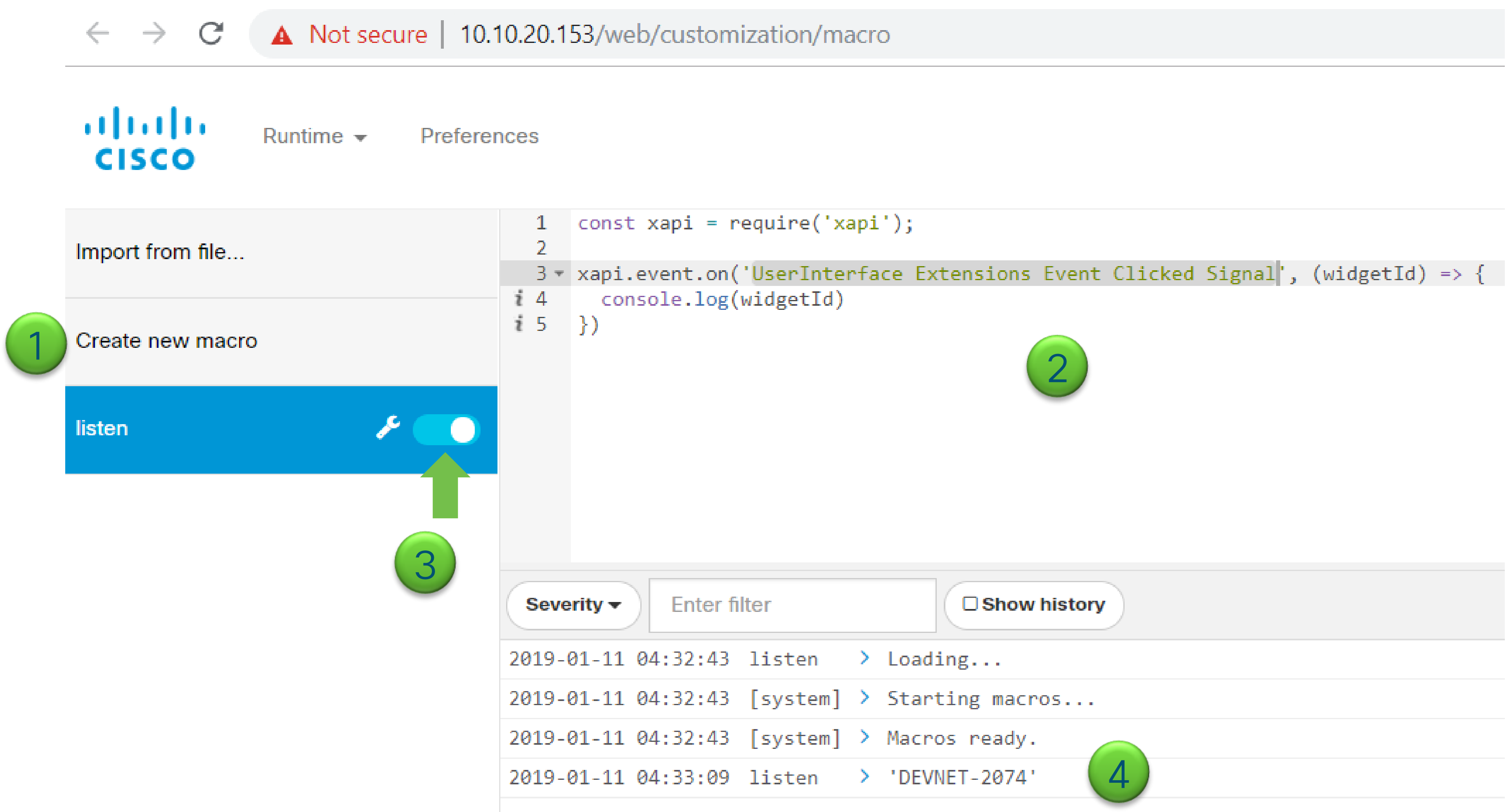
Take Over

Widget id: DEVWKS-2074

Widget width: 1

CE xAPI mapped to a Webex Room Kit





```
const xapi = require('xapi');

xapi.event.on('UserInterface Extensions Event Clicked Signal', (widgetId) => {
  console.log(widgetId)
})
```

```
2019-01-11 04:32:43  listen    > Loading...
2019-01-11 04:32:43  [system]  > Starting macros...
2019-01-11 04:32:43  [system]  > Macros ready.
2019-01-11 04:33:09  listen    > 'DEVNET-2074'
```

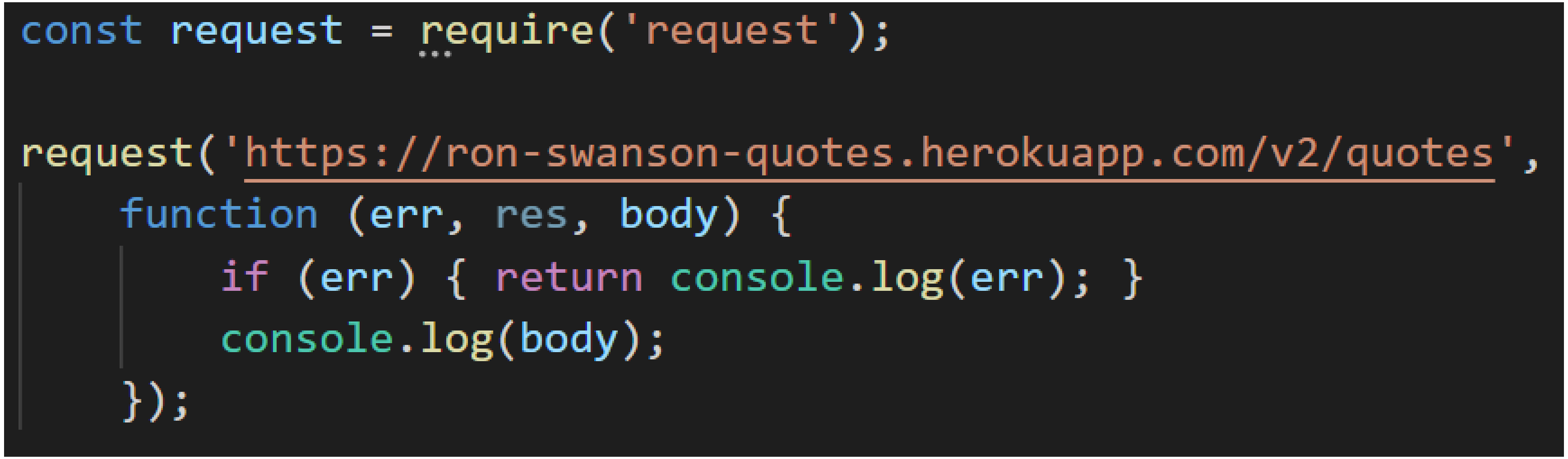
```
xapi.event.on('UserInterface Extensions Event Clicked Signal', (widgetId) => {
    console.log(`new event from widget: ${widgetId}`)

    let markdown = buildMarkdownForSession(widgetId)
    push(markdown)
})


function buildMarkdownForSession(widgetId) {

    let markdown = `no session found for widget identifier: ${widgetId}`
    let session = sessions[widgetId]
    if (session) {
      console.log(`found session with id: ${widgetId}`)
      markdown = `${session.day}, ${session.time}, ${session.location}`
      markdown += `<br/>**\[${session.id}\] - ${session.title}**`
      markdown += `<br/>_${session.description}_`
    }

    return markdown
}
```

Dealing with non-blocking IO

What is Node.js?

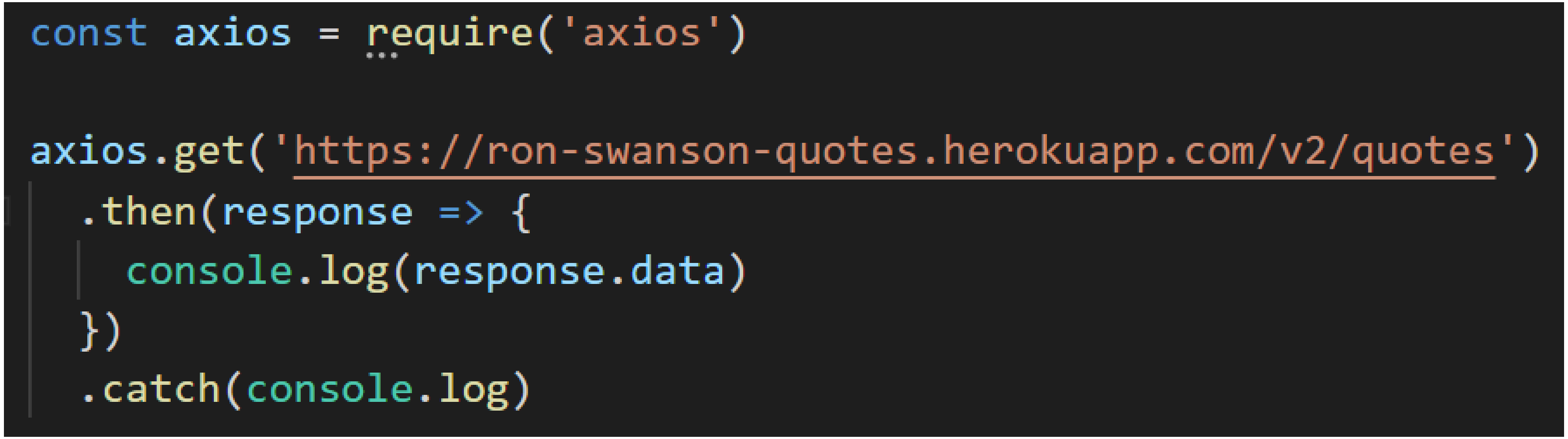
- Node.js® is a JavaScript runtime built on Chrome's V8 JavaScript engine (https://v8.dev)
  - V8 is Google's open source high-performance JavaScript and WebAssembly engine, written in C++.
  - Runs on Windows 7 or later, MacOS 10.12+, and Linux systems
  - Can run standalone, or can be embedded into any C++ application.
  - V8 is used in Chrome and in Node.js.
- **Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient.**

Async with Callbacks (ES5)



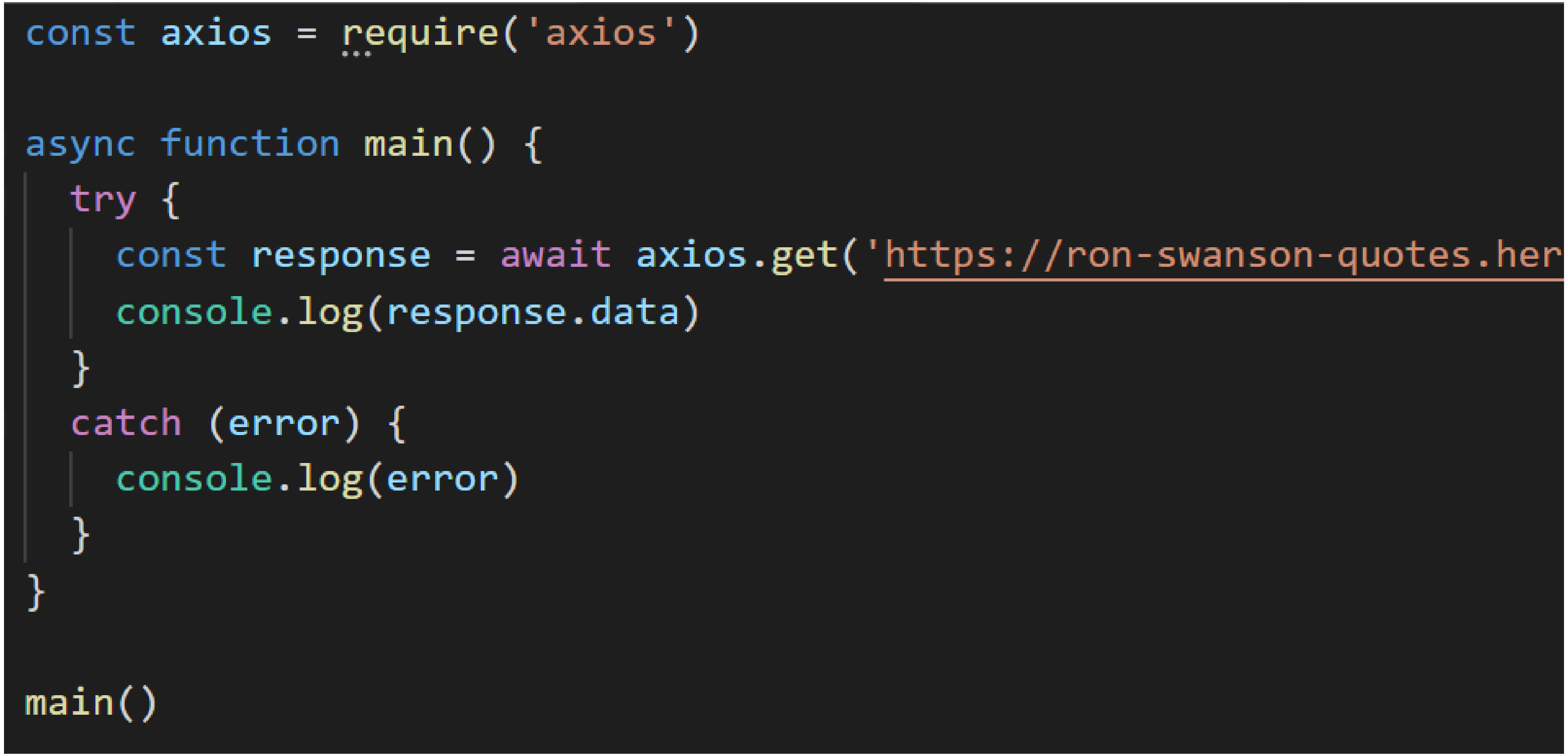
```
const request = require('request');

request('https://ron-swanson-quotes.herokuapp.com/v2/quotes',
    function (err, res, body) {
        if (err) { return console.log(err); }
        console.log(body);
    });
```

Async with Promises (ES6)



```
const axios = require('axios')

axios.get('https://ron-swanson-quotes.herokuapp.com/v2/quotes')
  .then(response => {
    console.log(response.data)
  })
  .catch(console.log)
```

Async with Async (ES8)



```
const axios = require('axios')

async function main() {
  try {
    const response = await axios.get('https://ron-swanson-quotes.her
    console.log(response.data)
  }
  catch (error) {
    console.log(error)
  }
}

main()
```

JavaScript Versions

JavaScript standard and versions

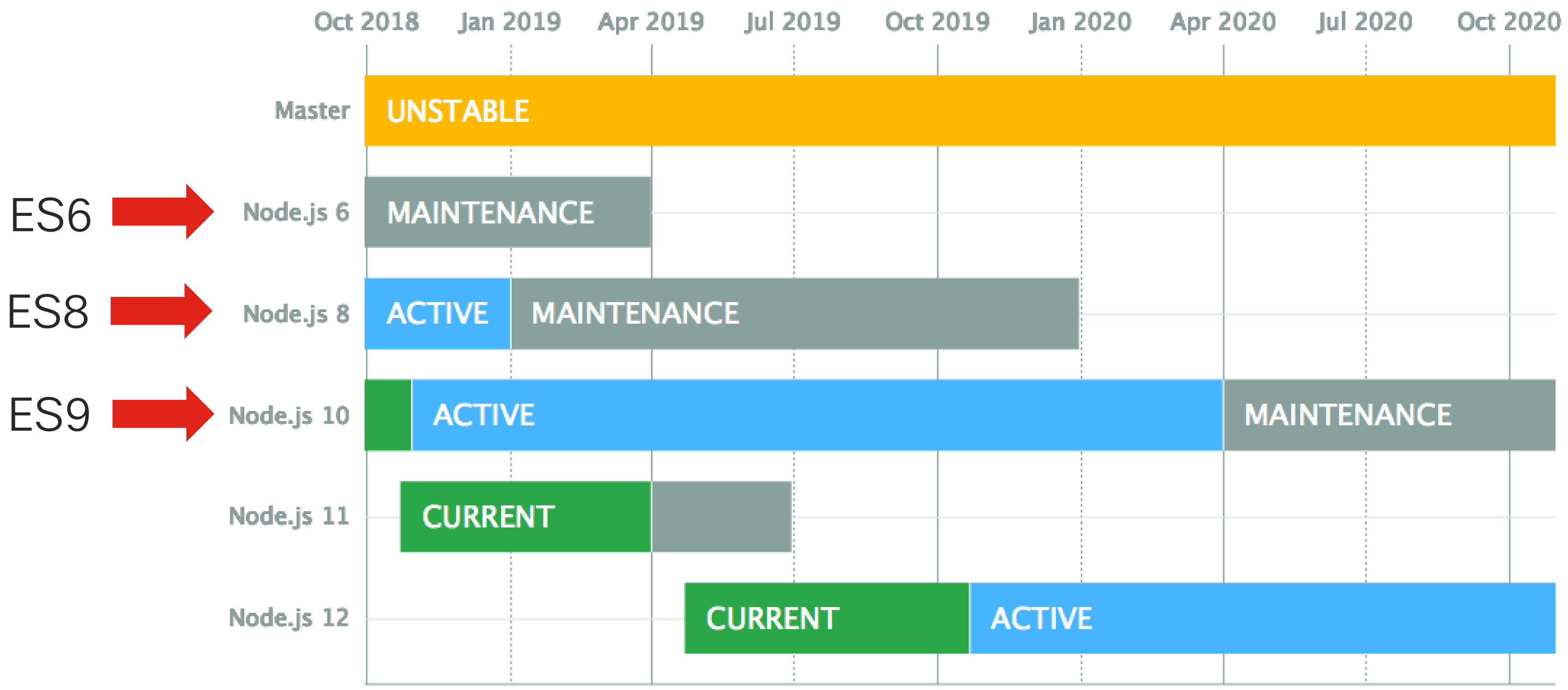
- JavaScript: The commonly used name for implementations of the ECMAScript standard
- ECMAScript: A language standardized by ECMA International and overseen by the TC39 committee.
- ES5 (ECMAScript 5): 5th edition, standardized in December 2009
- ES6 (ECMAScript 6) - ES2015: 6th edition, standardized in June 2015
- ES7+ : yearly releases
  - Standardized or scheduled to be standardized
  - ES7/ES2016 (June 2016), ES8/ES2017 (June 2017), ES9/ES2018 (June 2018)

ES5 Dec 2009	Object/array methods and extensions, strings, dates, JSON, immutable globals, strict mode
ES6 June 2015	- new syntax for writing complex applications: classes and modules, - Python-style generators and generator expressions, arrow functions, binary data, typed arrays, collections (maps, sets and weak maps), iterators and for/of loops, - Promises, reflection, proxies (metaprogramming for virtual objects)
ES7 June 2016	- exponentiation operator (**) - Array.prototype.includes
ES8 **June 2017**	- Includes await/async, which works using generators and promises.
ES9 June 2018	RegExp enhancements, Promise.prototype finally, await on loops declarations, spread properties

Node.js versions

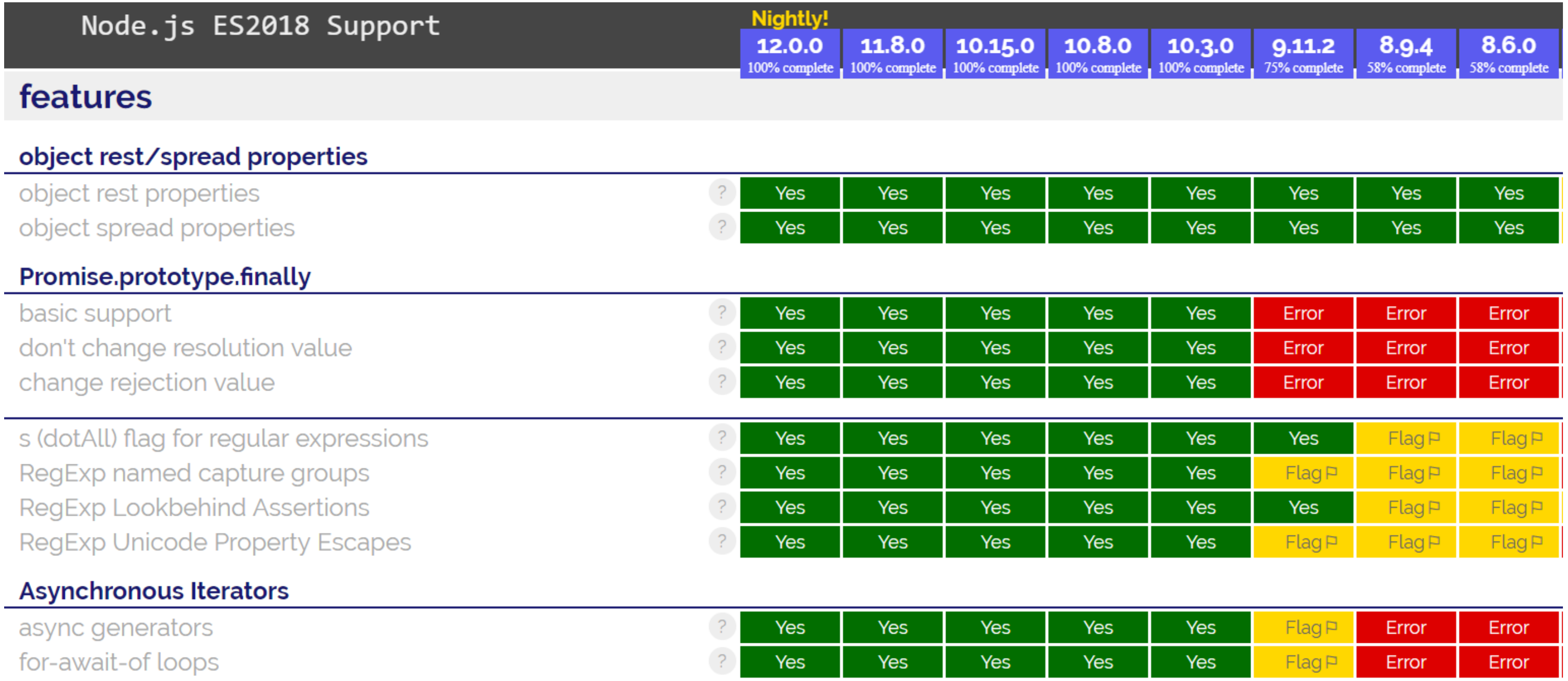
https://github.com/nodejs/Release

Even numbers stand for LTS
Long Term Support versions



Node.js EcmaScript Support

https://node.green



Node.js ES2018 Support		Nightly! 12.0.0 100% complete	11.8.0 100% complete	10.15.0 100% complete	10.8.0 100% complete	10.3.0 100% complete	9.11.2 75% complete	8.9.4 58% complete	8.6.0 58% complete
features									
object rest/spread properties									
object rest properties	?	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
object spread properties	?	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Promise.prototype.finally									
basic support	?	Yes	Yes	Yes	Yes	Yes	Error	Error	Error
don't change resolution value	?	Yes	Yes	Yes	Yes	Yes	Error	Error	Error
change rejection value	?	Yes	Yes	Yes	Yes	Yes	Error	Error	Error
s (dotAll) flag for regular expressions	?	Yes	Yes	Yes	Yes	Yes	Yes	Flag ⚐	Flag ⚐
RegExp named capture groups	?	Yes	Yes	Yes	Yes	Yes	Flag ⚐	Flag ⚐	Flag ⚐
RegExp Lookbehind Assertions	?	Yes	Yes	Yes	Yes	Yes	Yes	Flag ⚐	Flag ⚐
RegExp Unicode Property Escapes	?	Yes	Yes	Yes	Yes	Yes	Flag ⚐	Flag ⚐	Flag ⚐
Asynchronous Iterators									
async generators	?	Yes	Yes	Yes	Yes	Yes	Flag ⚐	Error	Error
for-await-of loops	?	Yes	Yes	Yes	Yes	Yes	Flag ⚐	Error	Error

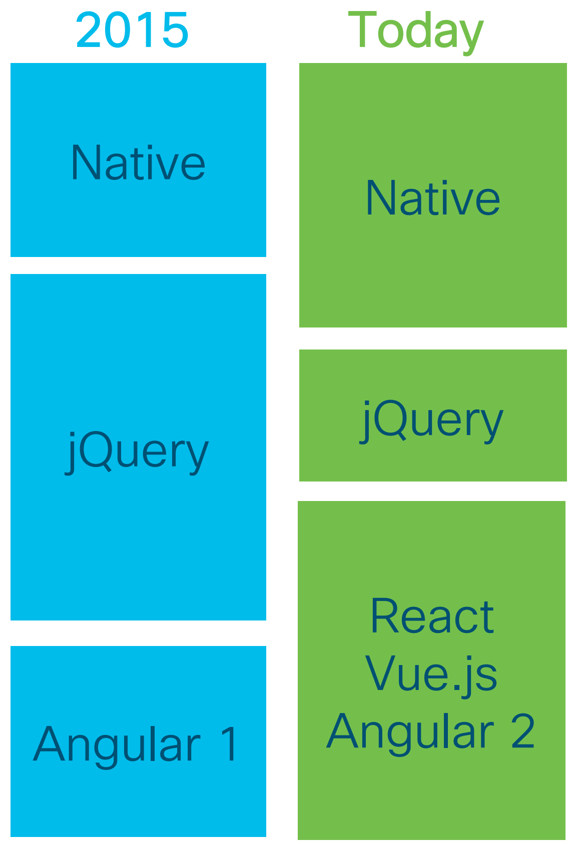
Client-side Javascript

JavaScript in the Browser

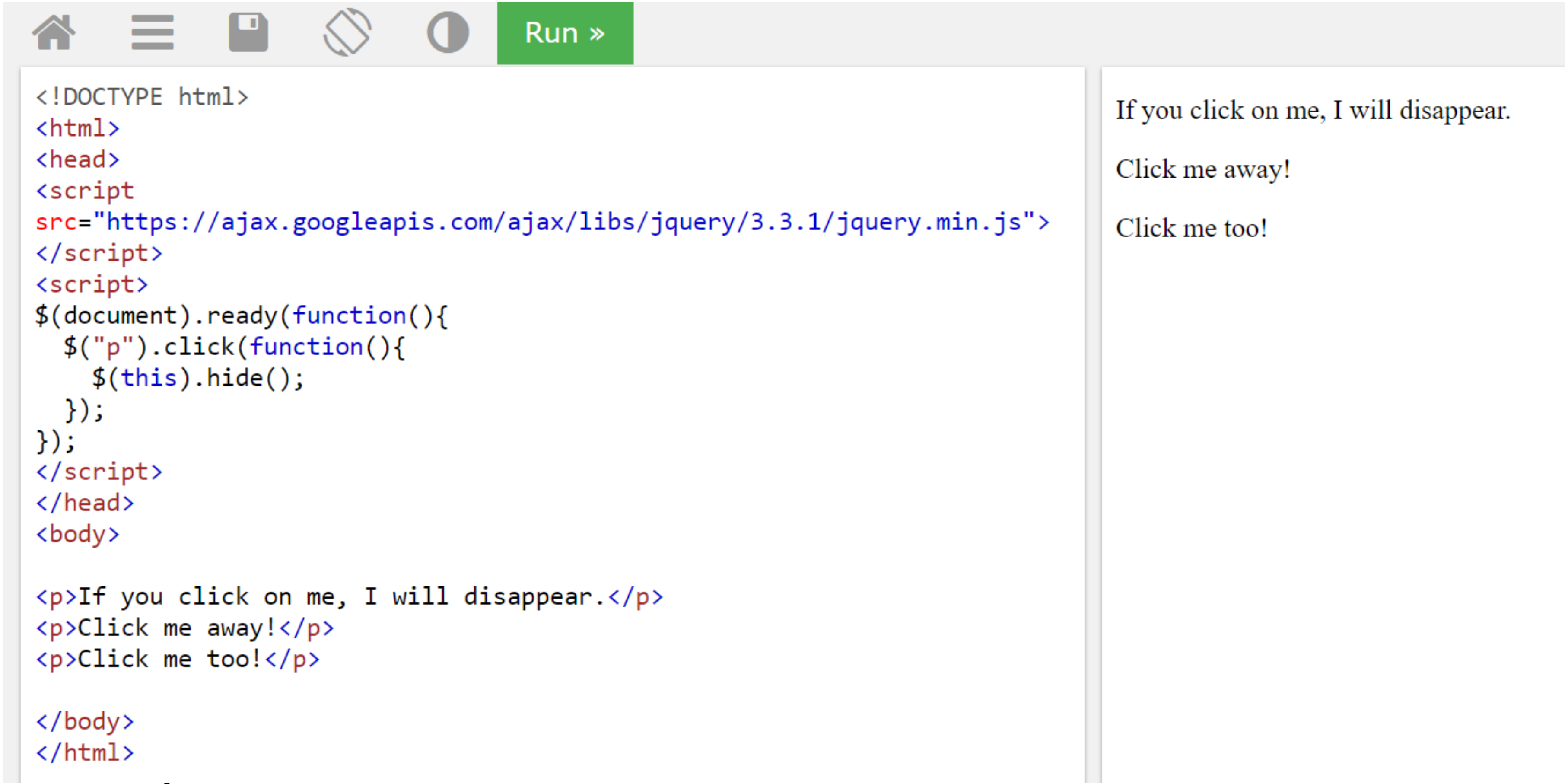
- Simply said, dynamic pages are event listeners and DOM manipulation…ending up with a lot of unmaintainable code.
- Browser compatibility make things even more messy.
- jQuery helps a lot…

```
$(document).ready(function(){
  $("p").click(function(){
    $(this).hide();
  });
```

… unless you're building Single Page Applications.



https://www.w3schools.com/jquery/tryit.asp?filename=tryjquery_hide



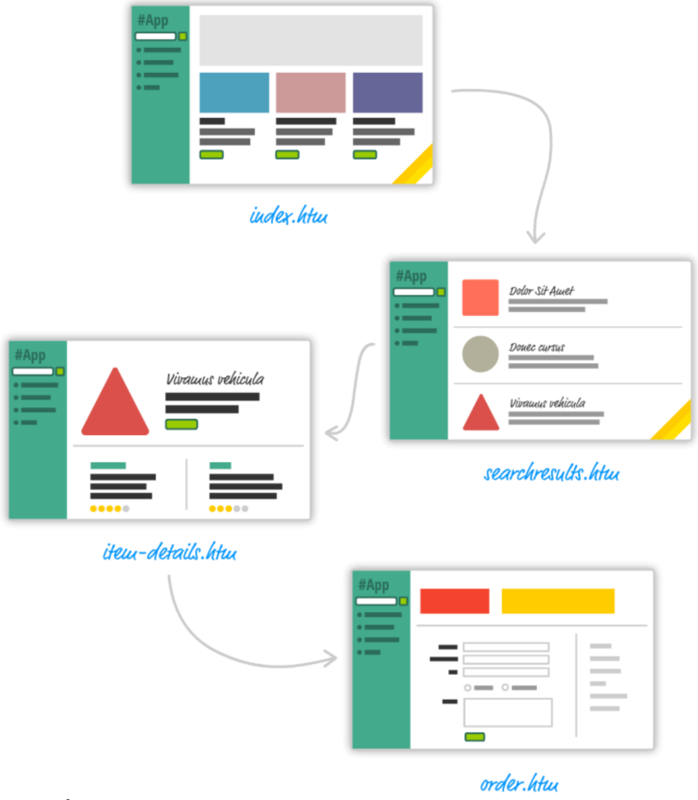
```
<!DOCTYPE html>
<html>
<head>
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js">
</script>
<script>
$(document).ready(function(){
  $("p").click(function(){
    $(this).hide();
  });
});
</script>
</head>
<body>

<p>If you click on me, I will disappear.</p>
<p>Click me away!</p>
<p>Click me too!</p>

</body>
</html>
```

If you click on me, I will disappear.

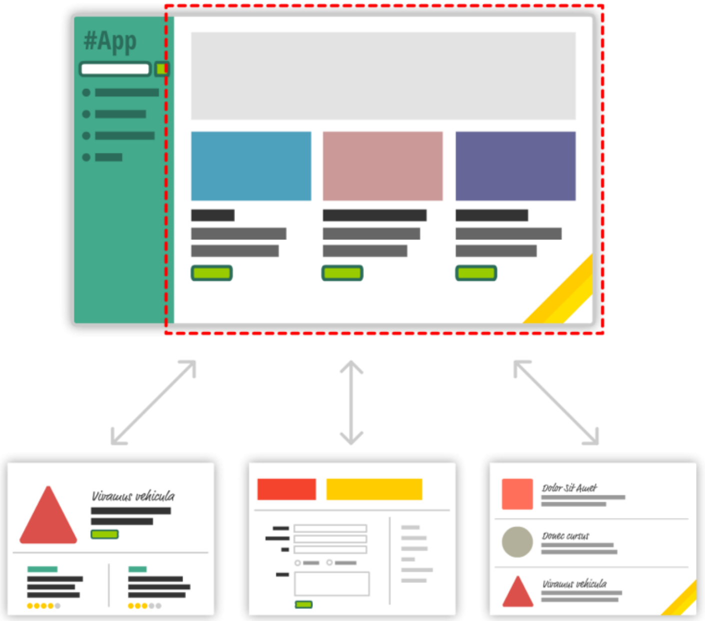
Click me away!

Click me too!

Traditional Web Apps (server side Web)



SPA (Single Page Apps)



https://www.kirupa.com/react/introducing_react.htm

jQuery

```
<ul class="js-items"></ul>

<script>
  $(function () {
    $.get('https://example.com/items.json')
      .then(function (data) {
        var $itemsUl = $('.js-items');

        if (!data.items.length) {
          var $noItems = $('li');
          $noItems.text('Sorry, there are no items.');
          $itemsUl.append($noItems);
        } else {
          data.items.forEach(function (item) {
            var $newItem = $('li');
            $newItem.text(item);

            if (item.includes('blue')) {
              $newItem.addClass('is-blue');
            }

            $itemsUl.append($newItem);
          });
        }
      });
  });
</script>
```

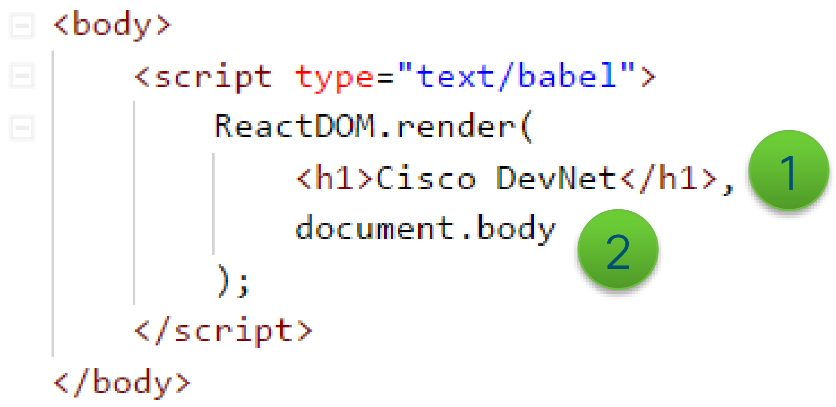
Vue.js

```
<ul class="js-items">
  <li v-if="!items.length">Sorry, there are no items.</li>
  <li v-for="item in items" :class="{ 'is-blue': item.includes('blue') }">
    {{ item }}</li>
</ul>
<script>
  new Vue({
    el: '.js-items',
    data: {
      items: []
    },
    created() {
      fetch('https://example.com/items.json')
        .then((res) => res.json())
        .then((data) => {
          this.items = data.items;
        });
    }
  });
</script>
```

React Basics

- The render method takes two arguments:

1. The HTML-like elements (aka JSX) you wish to output
2. The location in the DOM that React will render the JSX into



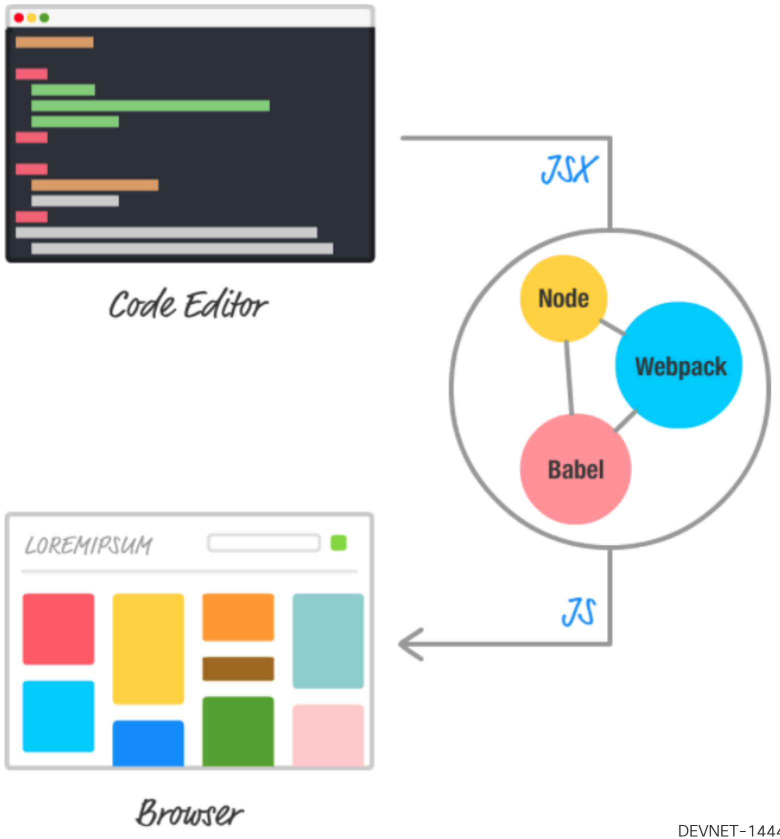
```
<body>
    <script type="text/babel">
        ReactDOM.render(
            <h1>Cisco DevNet</h1>,
            document.body
        );
    </script>
</body>
```

React Components

```
class Card extends React.Component {
  render() {
    var cardStyle = { ... };

    // JSX style
    return (
      <div style={cardStyle}>
      <Square color={this.props.color} />
      <Label color={this.props.color} />
      </div>
    );
  }
}

ReactDOM.render(
  <div>
    <Card color="#FF6663" />
  </div>,
  document.querySelector("#container")
);
```

Building your React Application

https://www.kirupa.com/react/setting_up_react_environment.htm



Javascript versions

- **ECMAScript 5 (ES5)**: Implemented in all modern browsers.
- **ECMAScript 6 (ES6 / ES2015)**: Fairly implemented in modern browsers and IE11+. Secured with Babel transpiling to ES5.
- **ECMAScript 7+ (ES2016+)**: Babel transpiling required.
- canIuse.com
- Learn at Transpile to ES5 via babel, or inject dynamically thru polyfills.js

Babel
https://babeljs.io

- Babel is a JavaScript compiler
  - https://babeljs.io/docs/en/learn
  - https://babeljs.io/learn-es2015/
- Toolchain used to convert ES6+ code into a backwards compatible version of JavaScript in current and older browsers or environments.
  - http://kangax.github.io/compat-table/es6/
- Transform syntax, Polyfill features that are missing in your target environment (through @babel/polyfill), Source code transformations (codemods)

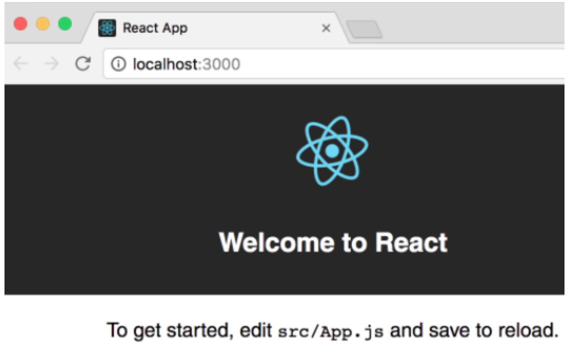
WebPack

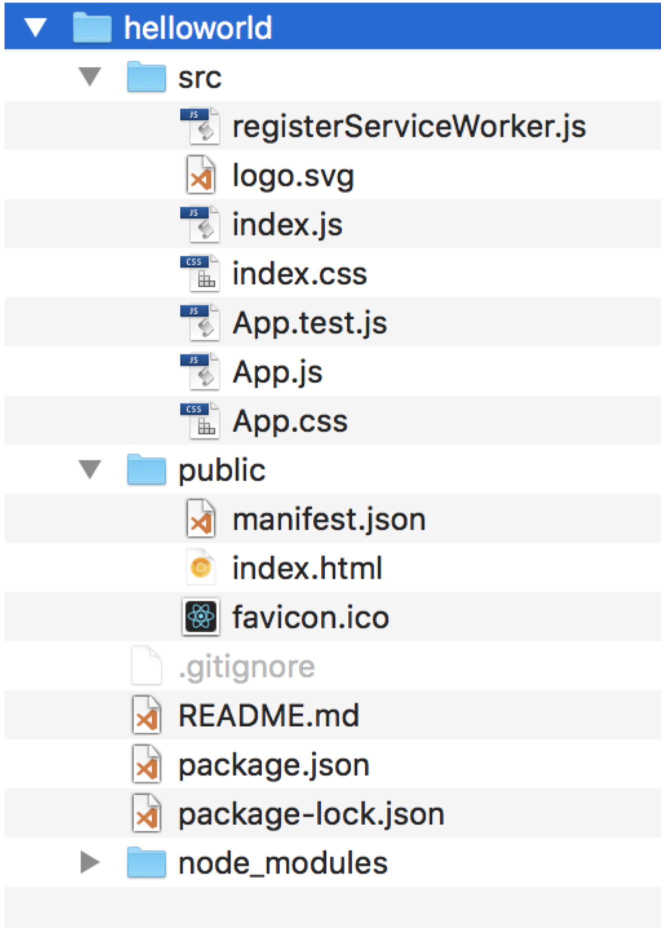
https://github.com/webpack

- Module bundler for Javascript applications
- Takes in various assets (JS, CSS, Fonts, Images, HTML...)
- Transforms, minifies and optimizes to serve one bundle to the browser
- JS library with an extensible architecture (loaders & plugins)
- Builds a dependency graph from webpack.config.js

Utility: create-react-app

```
> npm install -g create-react-app

> create-react-app helloworld

> cd helloworld


# for development

> npm start #


# for packaging
> npm run build
```

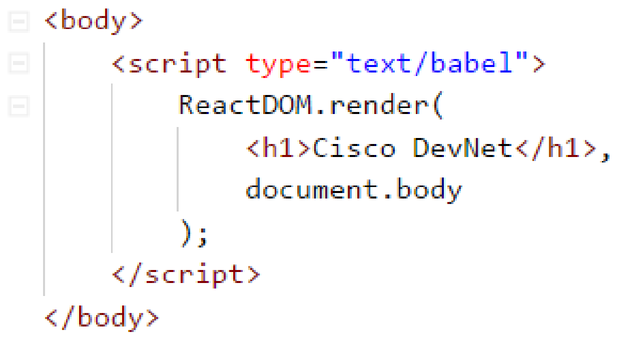

React App
localhost:3000
Welcome to React
To get started, edit `src/App.js` and save to reload.



```
helloworld
  src
    registerServiceWorker.js
    logo.svg
    index.js
    index.css
    App.test.js
    App.js
    App.css
  public
    manifest.json
    index.html
    favicon.ico
  .gitignore
  README.md
  package.json
  package-lock.json
  node_modules
```

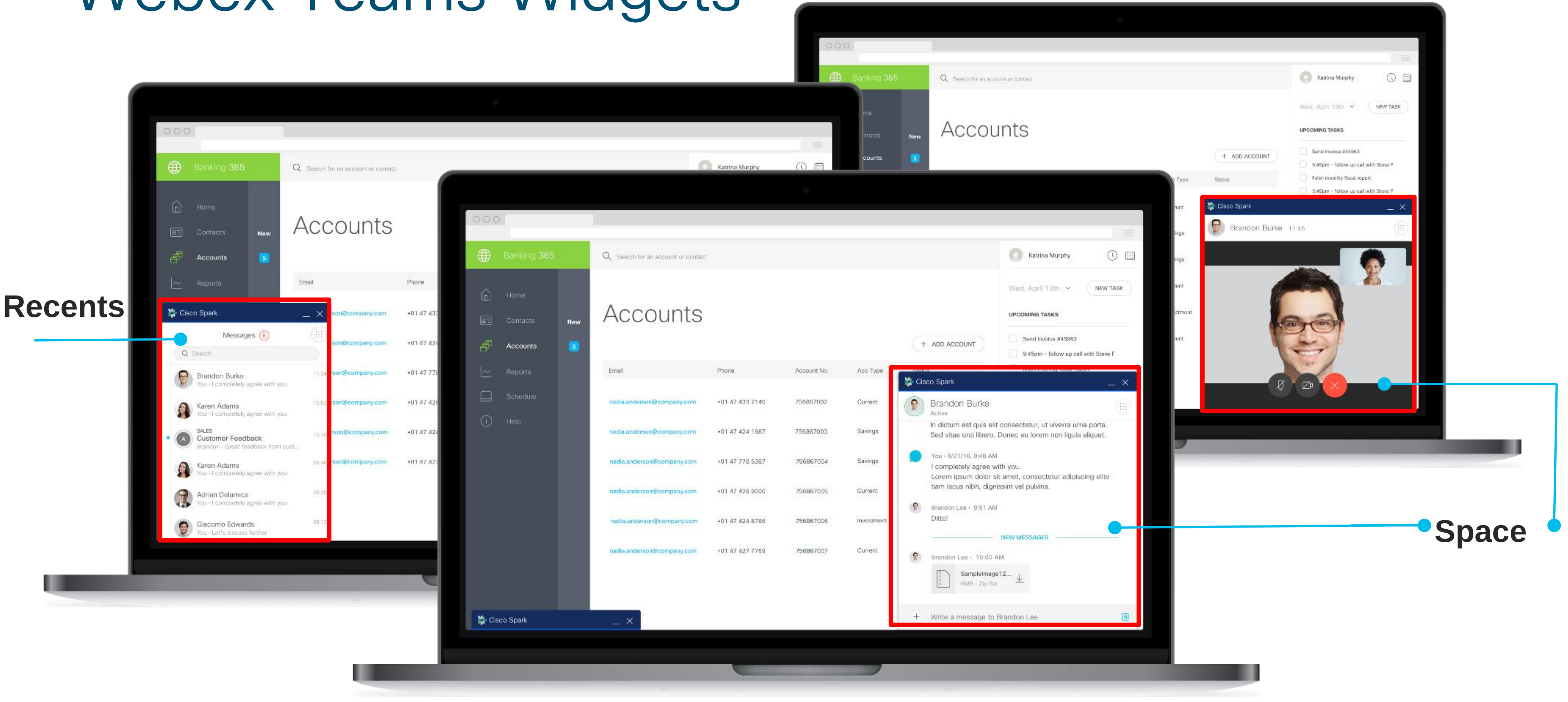
React

- Since it came out in 2013, React has found its way into popular web sites and apps that we use.
- At Cisco: Admin User Interfaces & Dashboards, Webex Teams Desktop clients and developer.cisco.com
- Automatic UI State Management
- Lightning-fast DOM Manipulation
  - In-memory Virtual DOM
- APIs to Create Truly Composable UIs
- Visuals Defined Entirely in JavaScript
  - no framework-specific templating command

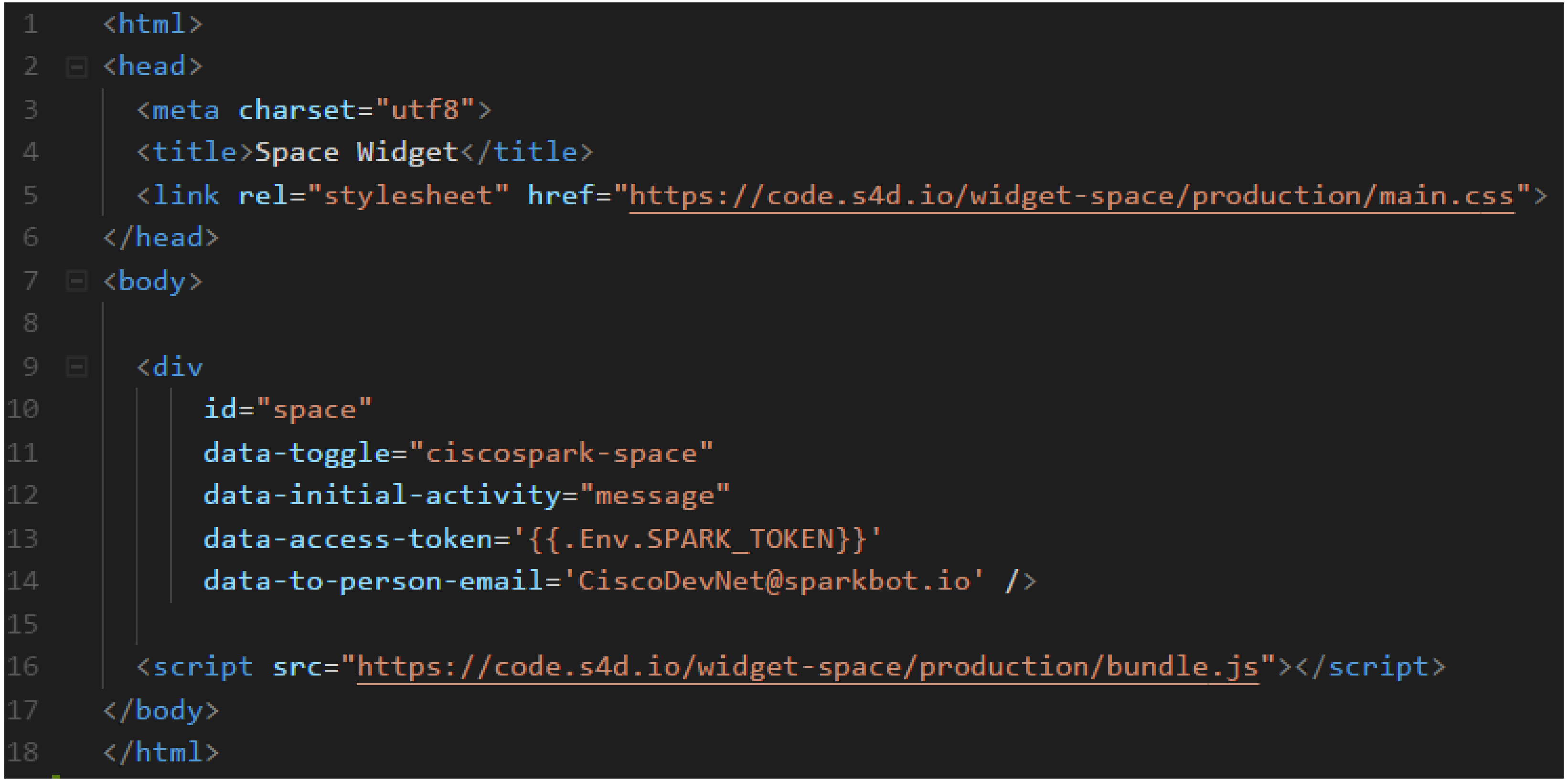


```
<body>
    <script type="text/babel">
        ReactDOM.render(
            <h1>Cisco DevNet</h1>,
            document.body
        );
    </script>
</body>
```

Webex Teams Widgets



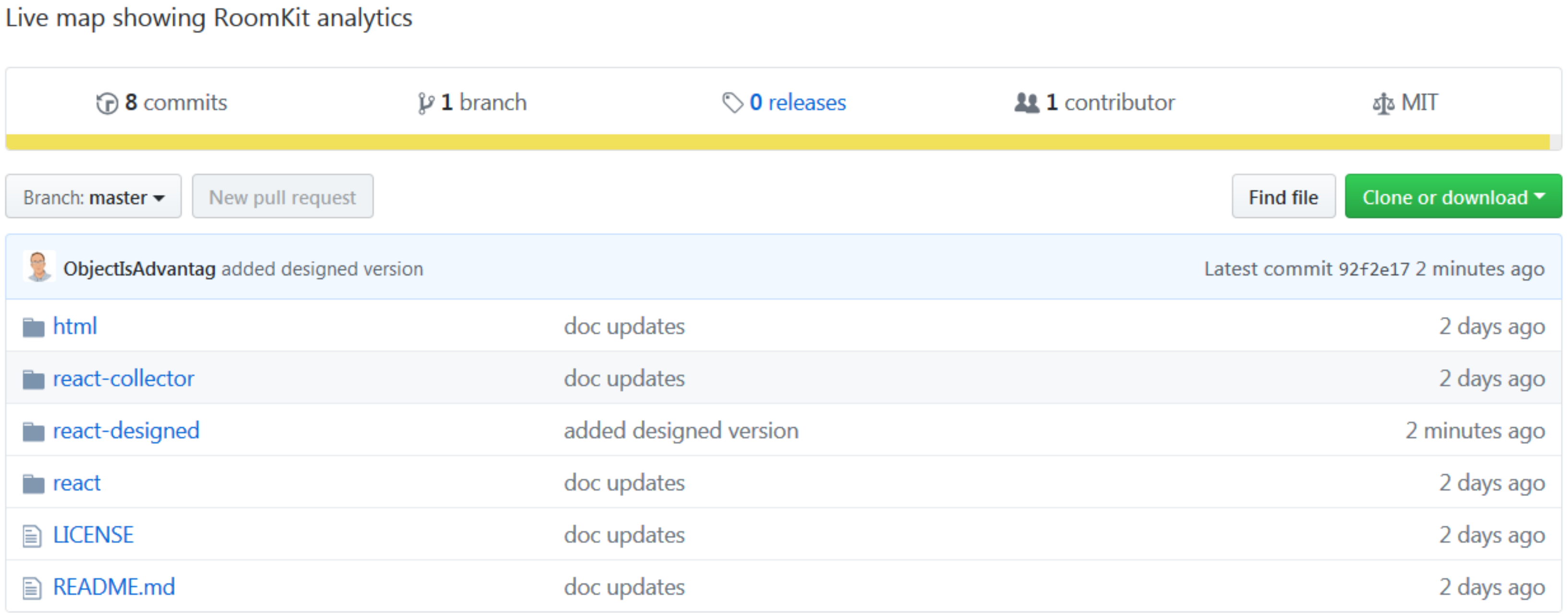
https://github.com/CiscoDevNet/widget-samples



```html
<html>
<head>
    <meta charset="utf8">
    <title>Space Widget</title>
    <link rel="stylesheet" href="https://code.s4d.io/widget-space/production/main.css">
</head>
<body>

    <div
        id="space"
        data-toggle="ciscospark-space"
        data-initial-activity="message"
        data-access-token='{{.Env.SPARK_TOKEN}}'
        data-to-person-email='CiscoDevNet@sparkbot.io' />

    <script src="https://code.s4d.io/widget-space/production/bundle.js"></script>
</body>
</html>
```

React Map Starter Kit

https://github.com/ObjectIsAdvantag/roomkit-react-map



Live map showing RoomKit analytics

8 commits | 1 branch | 0 releases | 1 contributor | MIT

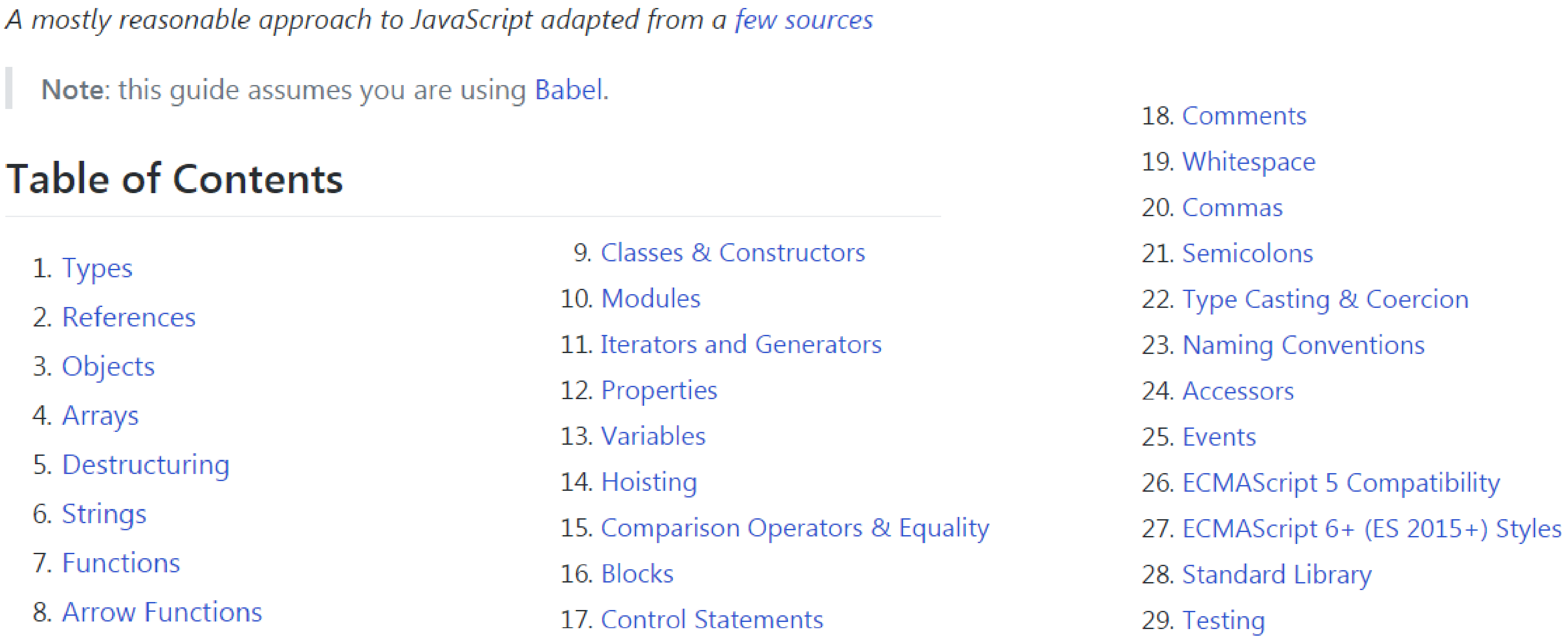
Branch: master | New pull request | Find file | Clone or download

ObjectIsAdvantag added designed version — Latest commit 92f2e17 2 minutes ago

html	doc updates	2 days ago
react-collector	doc updates	2 days ago
react-designed	added designed version	2 minutes ago
react	doc updates	2 days ago
LICENSE	doc updates	2 days ago
README.md	doc updates	2 days ago

Webex Teams JavaScript Styleguide

https://github.com/webex/web-styleguide



A mostly reasonable approach to JavaScript adapted from a few sources

> **Note**: this guide assumes you are using Babel.

Table of Contents

To go further

- Linter
  - Static code analysis tool used in software development for checking if JavaScript source code complies with coding rules.
- TypeScript
  - Optional static type-checking
  - Latest ECMAScript features
  - Compiles to plain JavaScript
  - https://www.typescriptlang.org/
- GraphQL
  - query language for APIs and a runtime for fulfilling those queries

Wrapup

JavaScript is Ubiquitous

Web Apps

Desktop Apps

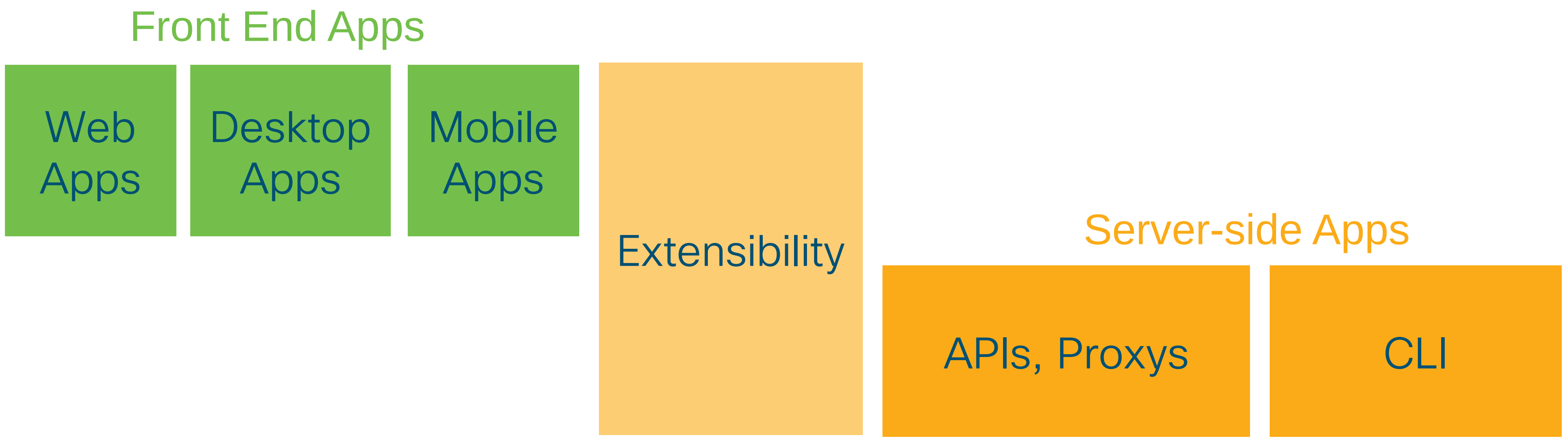
Mobile Apps

Extensibility

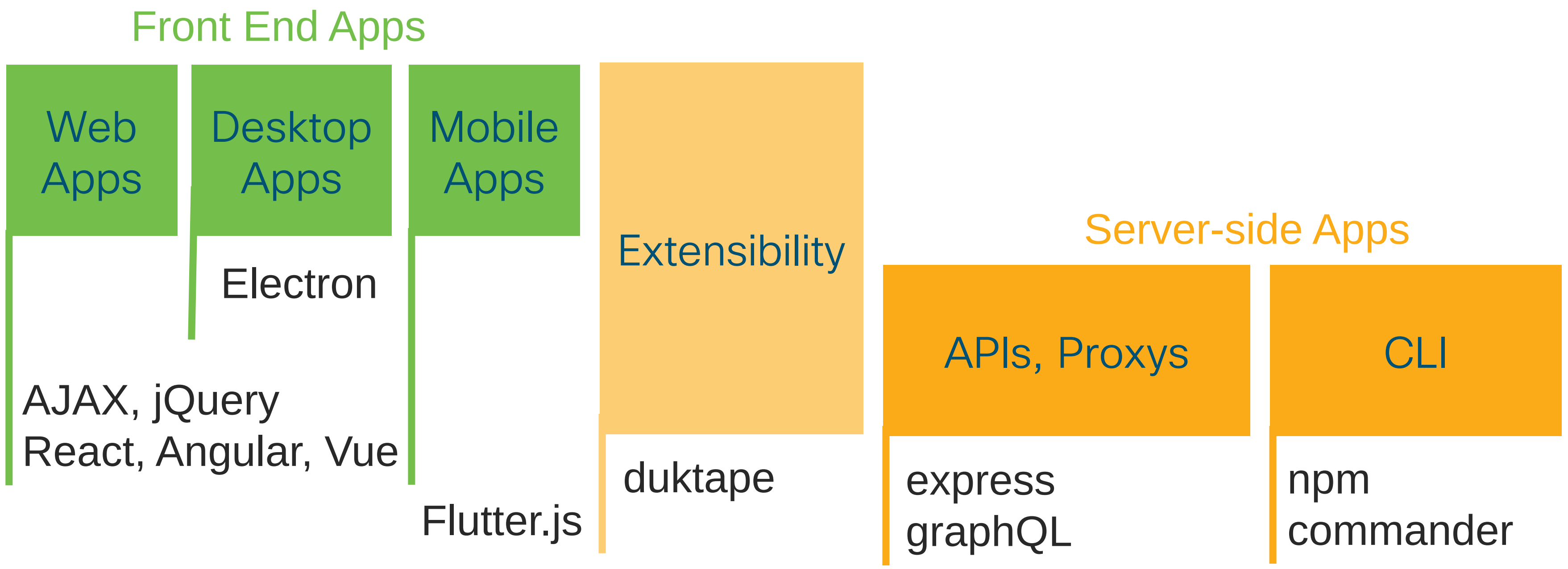
APIs, Proxys

CLI

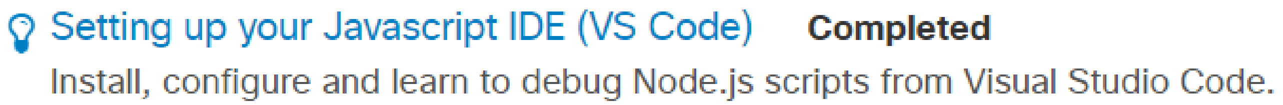
JavaScript is Ubiquitous



JavaScript is Ubiquitous



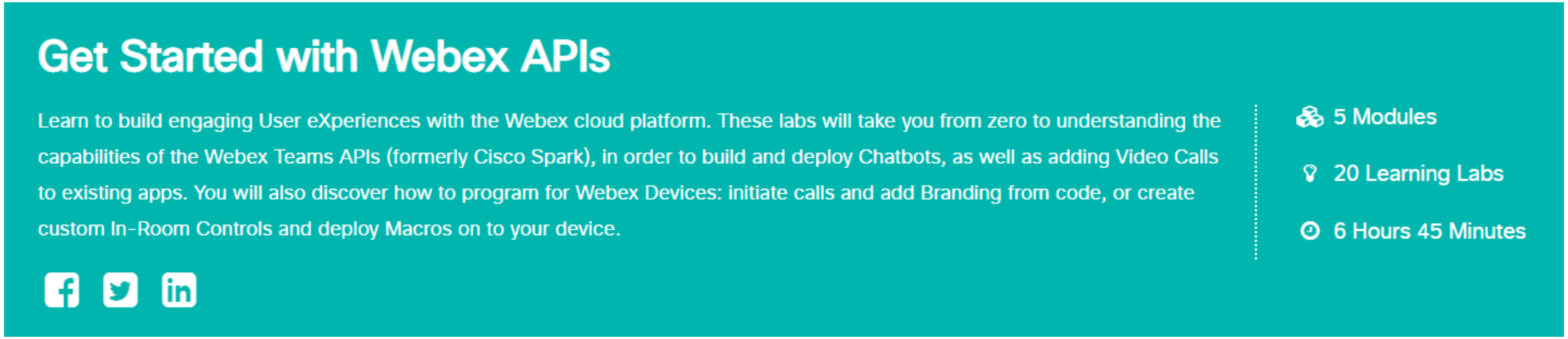
Webex Learning Track



Setting up your Javascript IDE (VS Code) **Completed**

Install, configure and learn to debug Node.js scripts from Visual Studio Code.

https://learninglabs.cisco.com/lab/collab-tools-ide-vscode-sd/step/1



Get Started with Webex APIs

Learn to build engaging User eXperiences with the Webex cloud platform. These labs will take you from zero to understanding the capabilities of the Webex Teams APIs (formerly Cisco Spark), in order to build and deploy Chatbots, as well as adding Video Calls to existing apps. You will also discover how to program for Webex Devices: initiate calls and add Branding from code, or create custom In-Room Controls and deploy Macros on to your device.

5 Modules

20 Learning Labs

6 Hours 45 Minutes

https://learninglabs.cisco.com/tracks/collab-cloud

Resources

- Node.js Coding 101 samples
  - https://github.com/ObjectIsAdvantag/nodejs-coding-101
- awesome-webex
  - https://github.com/CiscoDevNet/awesome-webex
- Webex learning track
  - https://learninglabs.cisco.com/tracks/collab-cloud

Complete your online session survey

- Please complete your session survey after each session. Your feedback is very important.
- Complete a minimum of 4 session surveys and the Overall Conference survey (starting on Thursday) to receive your Cisco Live t-shirt.
- All surveys can be taken in the Cisco Events Mobile App or by logging in to the Content Catalog on ciscolive.com/emea.

Cisco Live sessions will be available for viewing on demand after the event at ciscolive.com.

Continue your education

Demos in the Cisco Showcase

Walk-In Labs

Meet the Engineer 1:1 meetings

Related sessions

Thank you

You make **possible**