

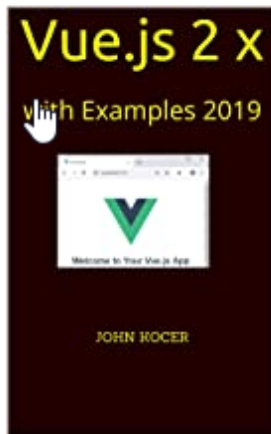
JumpStart ASP.NET Core 3.x and jQuery with Examples

John Kocer

Amazon KDP Publish
Version 2.0

Other books from the Author

Recent Book Reviews

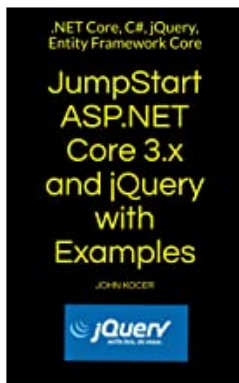


Vue.js 2 x: with Examples 2019 (Part One Book)
by [John Kocer](#)

★★★★★ [v 1](#)

[Kindle](#)

\$9⁹⁹



JumpStart ASP.NET Core 3.x and jQuery with Examples:
jQuery, Entity Framework Core

by [John Kocer](#)

[Kindle](#)

\$0⁰⁰ [kindleunlimited](#)

Free with Kindle Unlimited membership

Or \$9.99 to buy



Angular 9: by Example 2020 (Part One Book 1)
by [John Kocer](#)

[Kindle](#)

\$9⁹⁹



ASP.NET Core 3.x & Angular 9.x : by Example 2020
by [John Kocer](#)

[Kindle](#)

\$9⁹⁹



Pro Angular 7.x and REST APIs By Example 2019: 2019
by [John Kocer](#)

★★★★★ ~ 1

[Kindle](#)

\$9⁹⁹



ASP.NET Web API Core 3.1: Develop Cloud Ready Web
Example 2020

by [John Kocer](#)

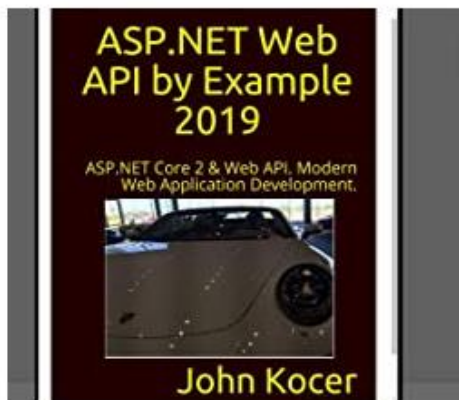
[Kindle](#)

\$9⁹⁹



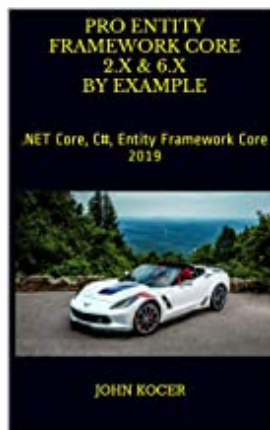
React 16: & REST API with Examples 2019
by [John Kocer](#)

[Kindle](#)
\$9.99



ASP.NET Web API with Examples: ASP.NET Core 2 & Web Application Development.
by [John Kocer](#)

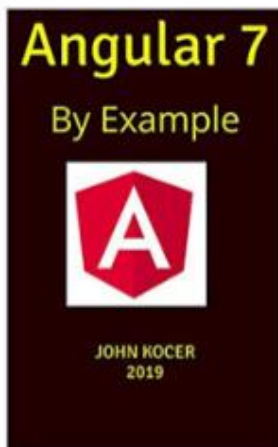
[Kindle](#)
\$9.99



Pro Entity Framework Core 2.x & 6.x By Example: Framework Core 2019
by [John Kocer](#)

★★★★★ ☆ 1

[Kindle](#)
\$9.99



**Angular 7: By Example (Part One Book 1) - John Kocer 2019
Rating**



Read Online

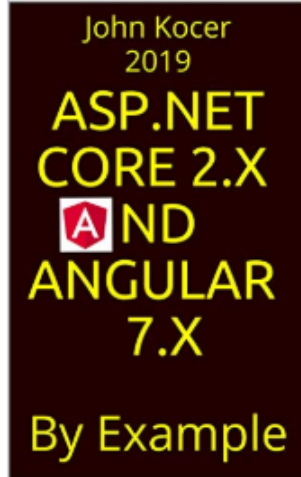


Download

About Angular 7: By Example (Part One Book 1) - John Kocer 2019

Angular 7: By Example (Part One Book 1) by John Kocer 2019 is new release from John Kocer 2019 first published by Amazon KDP Publish; 1 edition (November 22, 2018) that you can read

Feuilleter ↴



ASP.NET Core 2.x nd Angular 7.x: By Example (Part I Book 1) (English Edition) Format Kindle

de John Kocer 2019 (Auteur)

★★★★★ 1
[commentaire client](#)

[> Voir les formats et éditions](#)

Format Kindle
EUR 9,99

Lisez avec notre **Appli gratuite**

ASP.NET Core 2.x and Angular 7.x present a fast jump start for developers who wants to create modern web application with latest web technologies using Angular. We will create Angular SPA with Web APIs (REST) responsive design Simple

★★★★★ **Very practical book !!**

24 décembre 2018

Achat vérifié

The approach of this book is very novative and interesting.

The point is to be focused on the practical aspects.

It's mainly exercices with all we need to know about asp core and angular.

For me it's a good way to adress the main aspects of the laguage without loosing a lot of time on the theorical parts.

So thanks a lot to the author for his approach and I highly recommend this book for all the beginners on angular.



[Look inside ↴](#)

ASP.NET Web API with Examples: ASP.NET Core 2 & Web API. Modern Web Application Development. Kindle Edition

by John Kocer (Author)

★★★★☆ 1 customer review

[See all formats and editions](#)

Kindle
\$9.99

Read with Our **Free App**

ASP.NET Web API with Examples present a fast jump start for developers who wants to create modern web application with latest web technologies, more efficiently than ever, using Visual Studio 2017 ASP.NET Core 2.x, Web APIs and Entity Framework.

Build powerful HTTP services and make the most of the ASP.NET Core Web API platform.

ASP.NET Web API with Examples teaches developers how to add Web APIs (REST) with database

Angular 6.x

By Example

2019



JOHN KOCER

Angular 6.x: By Example 2019 (Part One Book 1)

by [John Kocer](#) | Sold by: Amazon Digital Services LLC | Dec 14, 2018

Kindle Edition

\$9.99



Buy now with 1-Click®

Top 10 Interview Questions for Software Developers: C++, Angular, React, .NET C# SQL, Architecture



C++, Angular, React, .NET C# SQL, Architecture

John Kocer

2019

Top 10 Interview Questions for Software Developers: C++, Angular, React, .NET C# SQL, Architecture

by [John Kocer](#) | Sold by: Amazon Digital Services LLC | Dec 20, 2018

Kindle Edition

\$9.99



Buy now with 1-Click®

John Kocer

Angular 5

with Examples

2018



ASP.NET Core 2.0 and Web API with Examples

John Kocer

2018

Angular 5

ASP.NET Core 2.0 and Web API with Examples

John Kocer

2018

Angular 5

ASP.NET Core 2.0 and Web API with Examples

John Kocer

2018

Angular 5

ASP.NET Core 2.0 and Web API with Examples

John Kocer

2018

Angular 5

ASP.NET Core 2.0 and Web API with Examples

John Kocer

2018

Angular 5

ASP.NET Core 2.0 and Web API with Examples

John Kocer

2018

Angular 5

ASP.NET Core 2.0 and Web API with Examples

John Kocer

2018

Angular 5

ASP.NET Core 2.0 and Web API with Examples

John Kocer

2018

Angular 5

ASP.NET Core 2.0 and Web API with Examples

John Kocer

2018

Angular 5

ASP.NET Core 2.0 and Web API with Examples

John Kocer

2018

Angular 5

ASP.NET Core 2.0 and Web API with Examples

John Kocer

2018

Angular 5

ASP.NET Core 2.0 and Web API with Examples

John Kocer

2018

Angular 5

Angular 5: ASP.NET Core 2.0 and Web API with Examples

by [John Kocer](#) | Sold by: Amazon Digital Services LLC | Mar 2, 2018

Kindle Edition

\$9.99



Buy now with 1-Click®

John Kocer

JumpStart ASP.NET Core 2.0 and jQuery with Examples: .NET Core, C#, jQuery, Entity Framework Core

with Examples

2018



ASP.NET Core 2.0 and jQuery with Examples

John Kocer

2018

ASP.NET Core 2.0 and jQuery with Examples

John Kocer

2018

ASP.NET Core 2.0 and jQuery with Examples

John Kocer

2018

ASP.NET Core 2.0 and jQuery with Examples

John Kocer

2018

ASP.NET Core 2.0 and jQuery with Examples

John Kocer

2018

ASP.NET Core 2.0 and jQuery with Examples

John Kocer

2018

ASP.NET Core 2.0 and jQuery with Examples

John Kocer

2018

ASP.NET Core 2.0 and jQuery with Examples

John Kocer

2018

ASP.NET Core 2.0 and jQuery with Examples

John Kocer

2018

ASP.NET Core 2.0 and jQuery with Examples

John Kocer

2018

ASP.NET Core 2.0 and jQuery with Examples

JumpStart ASP.NET Core 2.0 and jQuery with Examples: .NET Core, C#, jQuery, Entity Framework Core

by [John Kocer](#) | Sold by: Amazon Digital Services LLC | Nov 11, 2017

Kindle Edition

\$9.99



Buy now with 1-Click®

Notice of Rights

All rights reserved. No part of this book may be reproduced or transmitted in any form by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of the publisher. For information on getting permission for reprints and excerpts, contact

SmartIT.Consultation@gmail.com

.

Piracy

Piracy of copyrighted material on the Internet is an ongoing problem across all media. At SmartIT, we take the protection of our copyright and licenses very seriously. If you come across any illegal copies of our works in any form on the Internet, please provide us with the location address or website name immediately so that we can pursue a remedy.

Please contact us at SmartIT.Consultation@gmail.com with a link to the suspected pirated material. We appreciate your help in protecting our authors and our ability to bring you best valuable content.

Note: For those, some life situation who can NOT afford the buy the book we will provide a free legal copy of this book.

Dedication

To John and Mike

Course Objectives

After completing this course, you will be able to:

- Create Visual Studio Solution
- Create a new ASP.NET jQuery Web Application
- Write good JavaScript code
- Write functions
- Write good jQuery code
- Debug well JavaScript codes
- Debug well jQuery codes
- Know well How to use Developer Tools
- Know definition of JavaScript
- Know definition of jQuery
- Know definition of TypeScript
- Know definition of Angular
- Know difference between single quotes and double quotes
- Know difference between JavaScript object vs JSON syntax
- Know difference between JSON and XML
- Know the difference between identity and equality operator
- Validate well client side and server site input
- Know jQuery selectors
- Form HTML tags
- Performance improvement
- Bundling and minification
- Optimize page loads with AJAX calls
- Code CRUD operation with ASP.NET MVC Core and jQuery
- Create a new Web API controller
- Write HttpGet API method
- Write HttpPost API method
- Write HttpPut API method
- Write HttpDelete API method
- Consume NuGet packages
- Add dependency Injection
- Write JavaScript classes
- jQuery Ajax HttpGet
- jQuery Ajax HttpPut
- jQuery Ajax HttpPost
- Create Editable HTML table

- jQuery Selectors
- jQuery element visibility toggle
- HTML Tags and jQuery selector usage
- jQuery element CSS attribute toggle
- Form validation with jQuery
- Code JavaScript Components
- Create an AJAX Service Layer
- Debug C# code
- Configure API routes
- Debug API calls
- Test API calls
- Testing HTTP GET APIs
- Testing HTTP POST APIs
- Testing HTTP PUT APIs
- Testing HTTP DELETE APIs
- Testing with Postman
- Testing with Chrome Developers Tool
- Setting of project default IIS values
- Set start controller Route to Home Index method.
- Create a functional jQuery ASP.NET Web API Blog Application
- Pass Data Between Table row and JavaScript.
- Create a jQuery Credit Card Processing application with REST Web APIs.
- Create SPA (Single Page Application) with Responsive Web Design
- Override Bootstrap CSS.
- Creating Generic Repository with Entity Framework.
- Taking advantage of .NET generics
- Creating ASP.NET Core custom API controller CRUD operations with AJAX calls.
- Consuming Entity Framework Core, in-memory database.
- Interface
- Class
- Generic
- async/await
- DataContext
- Overriding onModelCreating
- Overriding SaveChanges
- Overriding Update
- Overriding UpdateAsync
- Overriding SaveChangesAsync
- Creating audit structure
- Dispose
- Creating Abstract class
- Add dependency Injection
- Getting Windows Identity Current User
- Creating a new insert item

- Updating an Item
- Deleting an Item
- Writing HttpPost Create API method.
- Writing HttpPut Update API method.
- Writing HttpDelete Delete API method.
- Use Postman to test REST APIs (GET, PUT, POST, DELETE)
- Lab Exercises to practice learned materials

Contents

Other books from the Author	2
Piracy	9
Dedication	11
Course Objectives	12
Introduction	20
Prerequisites for student	20
<i>Source Code</i>	21
Pre-Requirements	24
Preparing the work environment for Example	24
CH 1 - Creating Your First JavaScript Program in ASP.NET MVC	25
Introduction	25
Creating a New Project	26
Serve static files	33
Summary	38
CH 2- JavaScript Crush Course	39
Introduction	39
Include Script on a HTML Page	39
Reference an External Script on a HTML Page	41
How to add JavaScript file to a project.?	43
Browser Developer Tools	45
What is JavaScript?	46
Functions	47
Parameters and Return Values	48
Working with Objects	48
Creating object	48
JSON and XML similarities	50
JSON and XML differences	50
Which one to choose JSON or XML?	51
The Equality Operator vs. the Identity Operator	52
Listing: The Equality Operator vs. the Identity Operator	52
Converting Numbers to String	53
Converting Strings to Numbers	53

Summary	57
CH 3- jQuery Crash Course.....	Error! Bookmark not defined.
Introduction	Error! Bookmark not defined.
Chapter Objectives	Error! Bookmark not defined.
What is jQuery?.....	Error! Bookmark not defined.
What is Angular?	Error! Bookmark not defined.
What is TypeScript ?	Error! Bookmark not defined.
Where are the css files located in the project?.....	Error! Bookmark not defined.
Resources	Error! Bookmark not defined.
Download jQuery from	Error! Bookmark not defined.
Creating First jQuery Program in ASP.NET MVC.....	Error! Bookmark not defined.
Registering jQuery Library	Error! Bookmark not defined.
jQuery Selectors	Error! Bookmark not defined.
Basic selector and HTML Tags.....	Error! Bookmark not defined.
Element css attribute (Toggling an element background color).....	Error! Bookmark not defined.
Summary	Error! Bookmark not defined.
CHAPTER 4- Form Validation	Error! Bookmark not defined.
Introduction	Error! Bookmark not defined.
Summary	Error! Bookmark not defined.
CHAPTER 5- Remote Validation (Server Site AJAX call Validation)	Error! Bookmark not defined.
How to add Web API to the Project?	Error! Bookmark not defined.
CH 6- Employee Project Setting.....	Error! Bookmark not defined.
Set TodoRepository.....	Error! Bookmark not defined.
Create EmployeeController.....	Error! Bookmark not defined.
CHAPTER 7- ASP.NET Web API.....	Error! Bookmark not defined.
Overview.....	Error! Bookmark not defined.
CH 8 - How to Use Postman with ASP.NET Core Web API Testing.....	Error! Bookmark not defined.
Manuel testing with Postman	Error! Bookmark not defined.
Download Postman	Error! Bookmark not defined.
Run HtmlPageStartUp Project from previous chapter.	Error! Bookmark not defined.
Testing GET with Postman:	Error! Bookmark not defined.
Testing POST with Postman:.....	Error! Bookmark not defined.

Testing PUT with Postman:	Error! Bookmark not defined.
Testing DELETE with Postman:.....	Error! Bookmark not defined.
Summary	Error! Bookmark not defined.
CH 9- AJAX HTTP Calls	Error! Bookmark not defined.
Introduction	Error! Bookmark not defined.
AJAX calls - HttpGet	Error! Bookmark not defined.
AJAX calls – HttpPost	Error! Bookmark not defined.
AJAX calls - HttpPut.....	Error! Bookmark not defined.
AJAX calls - HttpDelete	Error! Bookmark not defined.
Summary	Error! Bookmark not defined.
CH 10- jQuery Debug	Error! Bookmark not defined.
Using Visual Studio Debugger.....	Error! Bookmark not defined.
Developer Tools Debugging	Error! Bookmark not defined.
CH 11- ASP.NET Core And JQuery CRUD Using WEB API	Error! Bookmark not defined.
Creating a HomeController	Error! Bookmark not defined.
Creating updateable HTML table and code jQuery AJAX Http Calls.....	Error! Bookmark not defined.
parseFloat() Function.....	Error! Bookmark not defined.
Debugging an API with a Break point.....	Error! Bookmark not defined.
Summary	Error! Bookmark not defined.
Lab Exercise.....	Error! Bookmark not defined.
CHAPTER 12- jQuery Payment Processing with ASP.NET MVC Core Using WEB API..	Error! Bookmark not defined.
Introduction	Error! Bookmark not defined.
Pre-Requirements	Error! Bookmark not defined.
Creating Payment Blank Solution	Error! Bookmark not defined.
Set PaymentRepository	Error! Bookmark not defined.
Create PaymentController	Error! Bookmark not defined.
Testing GetAllProductList API	Error! Bookmark not defined.
Create a Share directory under wwwroot.....	Error! Bookmark not defined.
Create member.component.html	Error! Bookmark not defined.
Create product.component.html	Error! Bookmark not defined.
Create payment.component.html.....	Error! Bookmark not defined.
Debugging MakaPayment	Error! Bookmark not defined.
Summary	Error! Bookmark not defined.

Lab Exercise	Error! Bookmark not defined.
CH 13 - Getting Started With Entity Framework Core ASP.NET Core	Error! Bookmark not defined.
Pre-Requirements	Error! Bookmark not defined.
SQL Server: How to find all localdb instance names?.....	Error! Bookmark not defined.
Creating solution	Error! Bookmark not defined.
Add Microsoft.EntityFrameworkCore.SqlServer	Error! Bookmark not defined.
Updating Startup.cs file	Error! Bookmark not defined.
Project creation Summary	Error! Bookmark not defined.
Summary	Error! Bookmark not defined.
Entity Framework Core	Error! Bookmark not defined.
Why chose Entity Framework Core?	Error! Bookmark not defined.
Why chose Entity Framework 6.x?	Error! Bookmark not defined.
What's new in EF Core 3.1	Error! Bookmark not defined.
What's new in EF 6.4.....	Error! Bookmark not defined.
CH 14- .NET Entity Framework Core Generic async Operations with Unit of Work Generic Repository	Error! Bookmark not defined.
Introduction	Error! Bookmark not defined.
Change Tracking with inheritance	Error! Bookmark not defined.
Consuming the Generic Repository class.....	Error! Bookmark not defined.
Dependency Injection:.....	Error! Bookmark not defined.
Creating jQuery SPA index page.....	Error! Bookmark not defined.
Chapter Summary.....	Error! Bookmark not defined.
Resources	Error! Bookmark not defined.

JumpStart

ASP.NET Core 3.x and jQuery with Examples

Visual Studio 2019, ASP.NET Core 3.x and jQuery Web Application Development

JumpStart ASP.NET Core 3.x and jQuery present a fast jump start for developers who wants to create modern web application with latest web technologies using Visual Studio 2019, VS Code ASP.NET Core 3.x and jQuery. We will create jQuery SPA with ASP.NET Web API (REST) responsive design Simple Employee CRUD web application and jQuery multi component Product Sale and Payment Processing Web Application. We also learn best tools for verifying and testing Application APIs.

Bonus Chapter: Create a SPA Blog Manager Web Application with jQuery .NET Entity Framework Core Generic async Operations with Generic Repository.

SmartIT- John Kocer

Copyright 2020 (c) SmartIT. All rights reserved.

About the Author

John has 15 years' experience in the information technology field with roles that included IT Consultant, Instructor, Sr. Software Developer, Web Architect, Technical Project Coordinator, Author, Manager and Mentor. He is award-winning mentor and teacher. He also worked at Intel for many years in several locations across the country.

Please email feedback / corrections (technical, grammatical or spelling) to below email.
Contact to Author: SmartIT.Consultation@gmail.com

Introduction

My purpose is present a fast jump start for you with the tools you need to code your first ASP.NET Core 3.0 and jQuery application and be able to search and find more by yourself when you need extra information.

Prerequisites for student

In order to fully utilize this book and meets you goal, nice to have

- Have basic experience creating applications with C#
- Some knowledge about client, server and database web application development.
- Have some working knowledge of HTML
- Some knowledge about Web API REST
- Some knowledge about what is AJAX

How to learn fast from this book (Recommended) or your own learning style

- Download source code of the book's examples. Source code download link is end of the book.
- Have two monitors, if you can, open source code on the one monitor
- If you can with Visual Studio Code divide your screen vertically for html code section and JavaScript section so you can quickly see HTML and JavaScript at the same time.
- I like Developer Tools open on my browser when I am developing, so that I can see any client error or server site errors right away.
- Master debug and tracing code with tips provided with this book.
- When you finish a chapter, there are lab exercise for you to complete and practice your new learned knowledge.
- Finally, make two cheat sheets
 - Your own Cheat sheet (You are creating customized for you) JavaScript cheat sheet.
 - Your own Bug sheet (You are creating customized for you) JavaScript bug sheet. (The goal is the never create the same bug again)
 - Create cheat sheets for other languages (C#, SQL, Web API, jQuery etc.)
 - Kindle reader on Windows is not the best for copying text to a file. So, you can download the source code and look at when you need to copy and paste the source code.
 - **It is very important that you need to set your Kindle View-->Display Options (Ctrl+Shift +D) to adjust for best picture view. The setting below worked for me very well.**

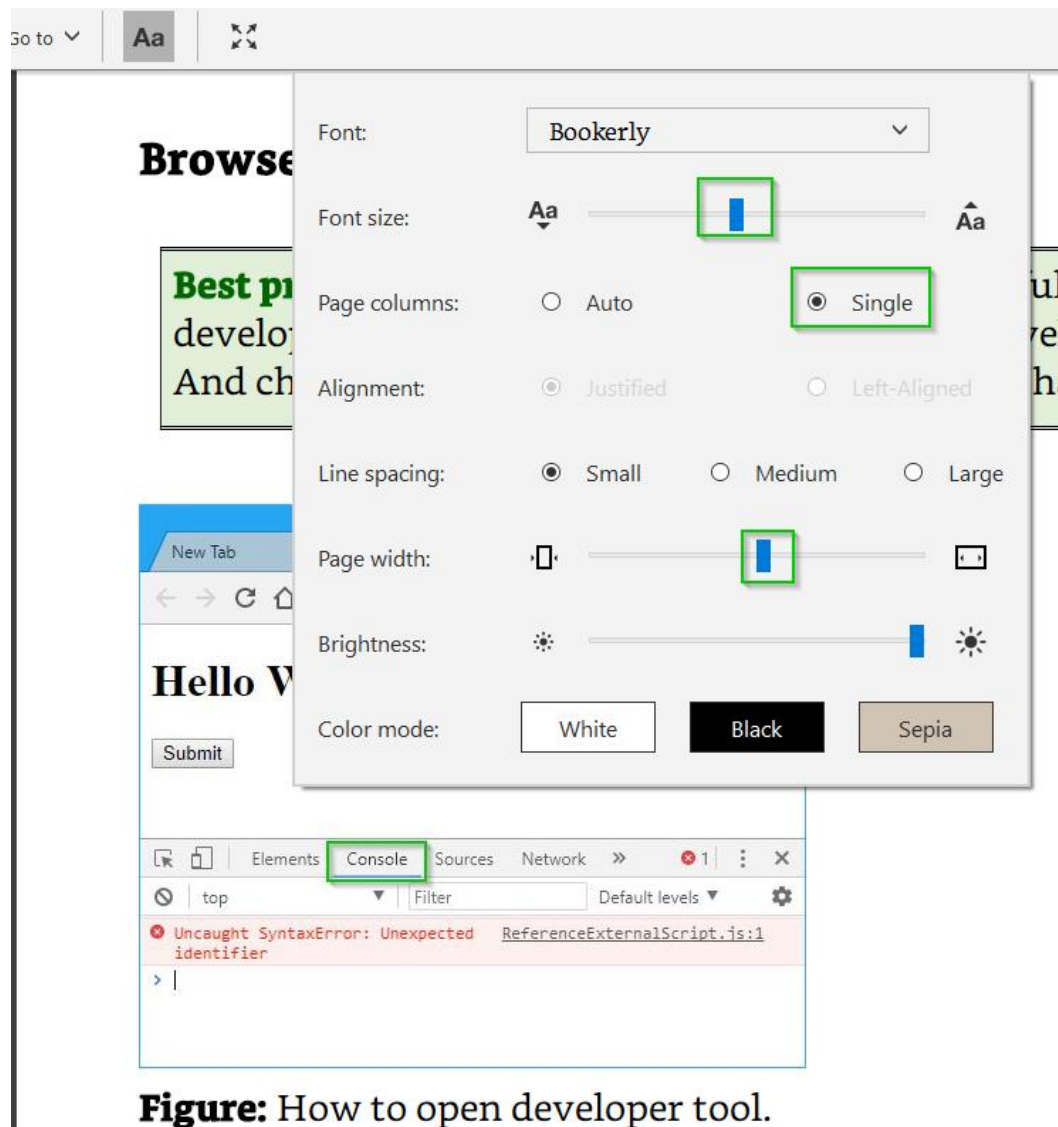


Figure: How to open developer tool.
Figure: Amazon Kindle Optimal Display Setting.

- Don't forget to take a break after every 30 minutes or 45 minutes.

Have your own project ready and Jumpstart your Enterprise project with the knowledge and tools you learned and be able to search and find other missing pieces from developer communities.

Source Code

The source code download link is located at the end of the last chapter resources.

I sincerely hope that you enjoy reading and coding this book and the lab exercises as much as I liked writing and coding it to help you become proficient enough with ASP.NET Core, jQuery and Entity Framework.

I can be get in touch by:

Contact to Author: SmartIT.Consultation@gmail.com

In case of your project needs it, I am also available for consulting, teaching, coding and architecting any size of a projects, all around the World.

Your success is our success. If you like this book, I'd be much grateful, if you took some of your precious time and leaving positive comments, and sending your improvement and critique us in a email below with book and version title.

SmartIT.Consultation@gmail.com

Thanks, a million!

TIP: A good manager praises in public and gives honest feedback and blame in private.

Warning

This book is NOT a jQuery reference book. It is a jQuery and, ASP.NET Core training material.

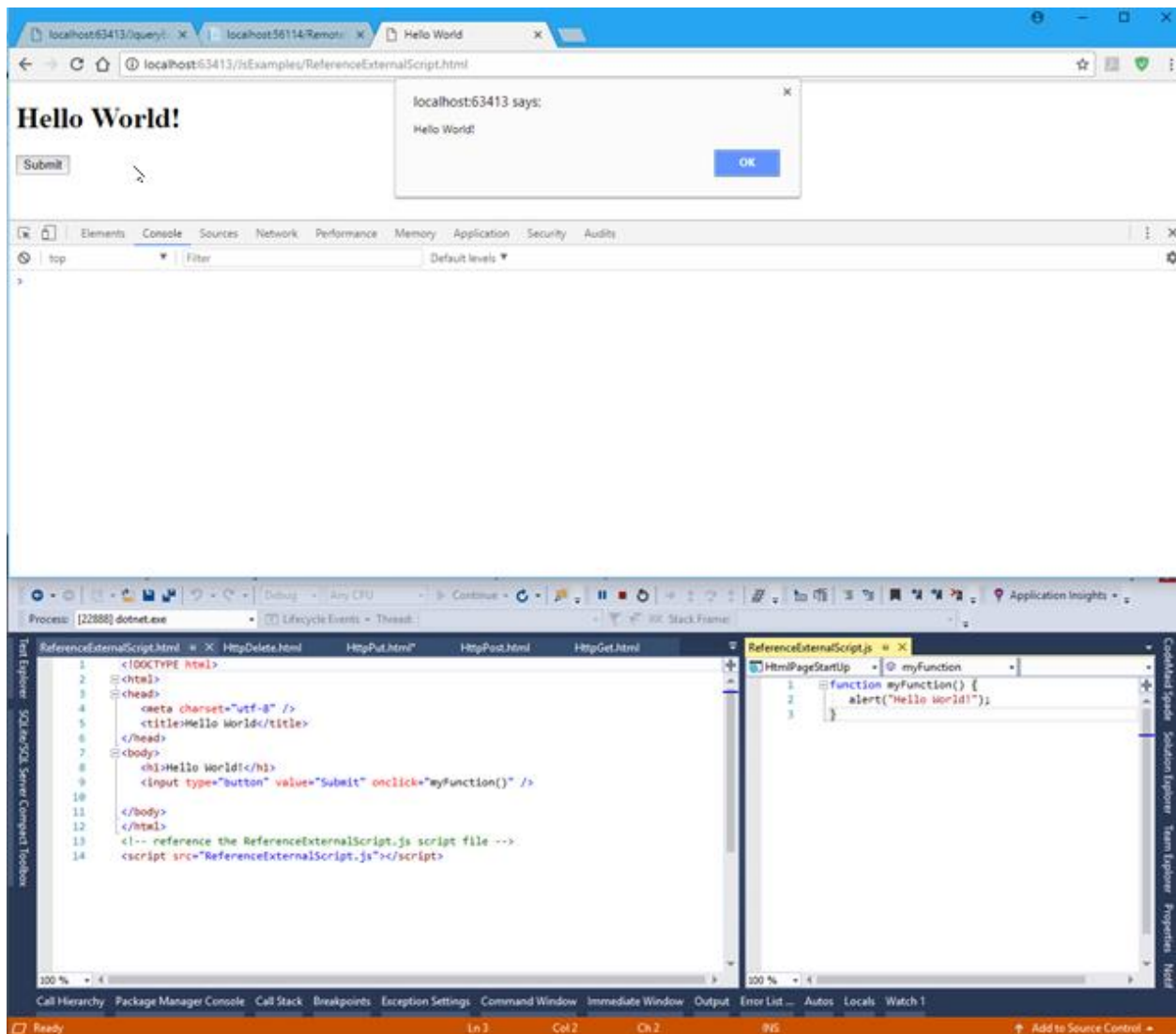


Figure: 4-way monitor setup. UI Page and Console(Debug) windows on the top. HTML code window on the left and JavaScript window on the right. Find your most productive monitors and applications windows setup and use it. .NET Core code will also run on macOS and Linux.

Pre-Requirements

In order to be able to run the examples from the download or build it from scratch, you need to have the following tools:

- Visual Studio 2019 or VS Code latest version
 - Visual Studio download page
<https://www.visualstudio.com/downloads/>
- .NET Core 3.1 or above
- jQuery
 - jQuery installation page
<http://jquery.com/download/>

Preparing the work environment for Example

If you don't already have Visual Studio 2019. Download from <https://www.visualstudio.com/downloads/> and install it.

CH 1 - Creating Your First JavaScript Program in ASP.NET MVC

Introduction

In this session, we will learn:

- Creating a Blank Solution
- Creating an Empty ASP.NET Core Web Application
- Setting a HTML file as startup file.
- `app.UseStaticFiles()`
- `app.UseDefaultFiles();`
- Creating wwwroot directory
- Setting Default controller and view
- `services.AddControllersWithViews();`
- `app.UseRouting();`
- `app.UseEndpoints`
- Entry point of ASP.NET Core
- Startup.cs file

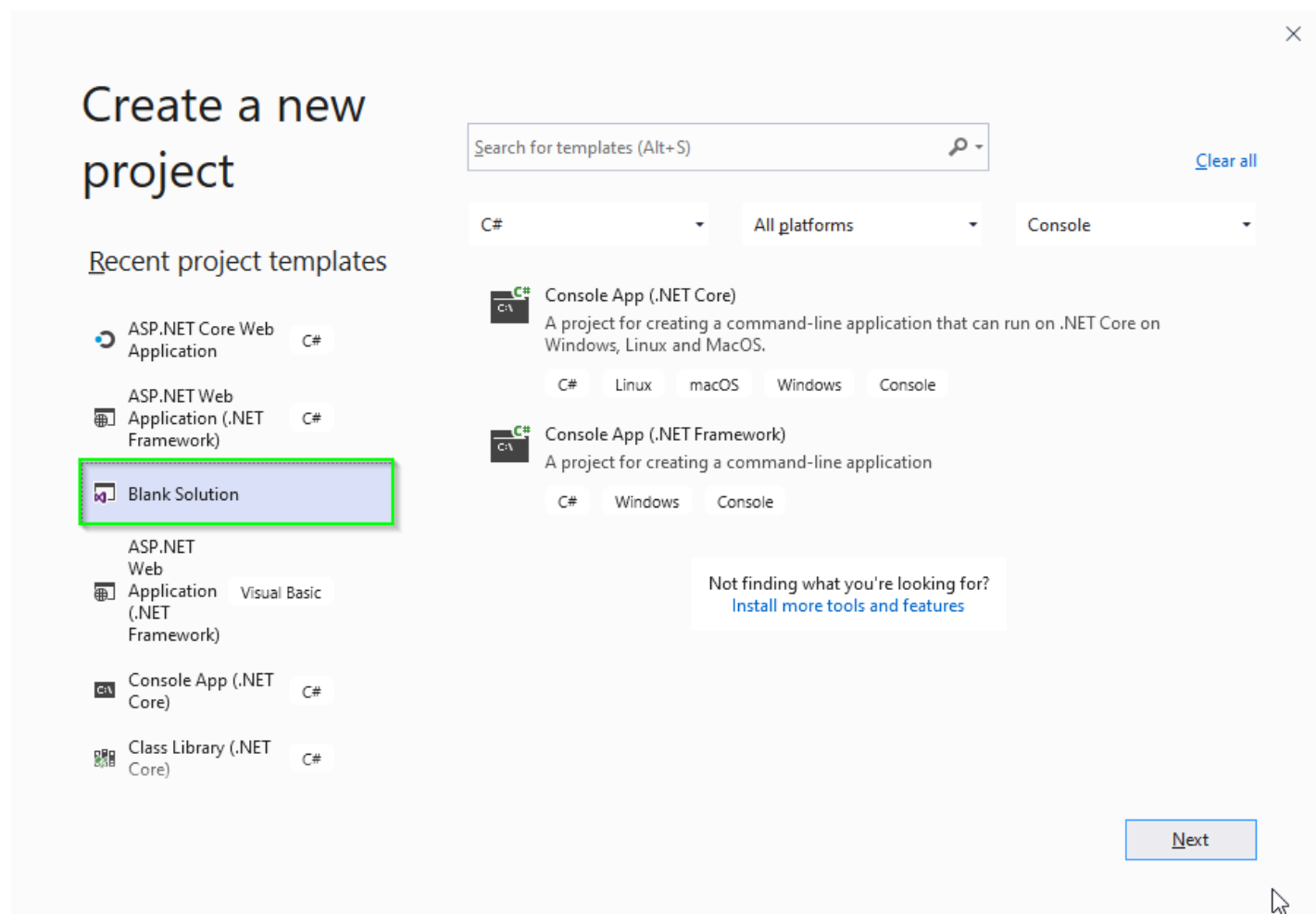
Now we are ready to write our first JavaScript program with MVC Core.

Recommended: Download this Chapter's source code and use as reference.

We want an ASP.NET Core project that starts with an HTML file. Most ASP.NET Core Web Projects starts with .cshtml files such as home/index.cshtml. MVC and Razor pages used in ASP.NET needs to be compiled to html, CSS, and JavaScript to run on the browser. HTML, CSS and JavaScript, included jQuery can run on the browser without ASP.NET compiler.

Creating a New Project

Create a new **Blank Solution** and name as `HtmlPageStartUp`.



×

Configure your new project

Blank Solution

Other

Project name

HtmlPageStartUp

Location

D:\aJohn20\jQueryCore3\jQueryCore3Source\01 - First JavaScript

⋮

Solution name ⓘ

HtmlPageStartUp

Back

Create


Figure: Creating blank Solution


Create a new ASP.NET Core Empty Web Application


Add a new project: Select **ASP.NET Web Application** as shown in the screen below.


Add a new project


Recent project templates


 ASP.NET Core Web Application C#

 ASP.NET Web Application (.NET Framework) C#

 ASP.NET Web Application (.NET Framework) Visual Basic

 Console App (.NET Core) C#

 Class Library (.NET Core) C#

 Class Library (.NET Standard) C#

Search for templates (Alt+S)



[Clear all](#)

C#

All platforms

Console



Console App (.NET Core)

A project for creating a command-line application that can run on .NET Core on Windows, Linux and MacOS.

C#

Linux

macOS

Windows

Console



Console App (.NET Framework)

A project for creating a command-line application

C#

Windows

Console

Not finding what you're looking for?

[Install more tools and features](#)

Next

Name the Project as `HtmlPageStartUp`.

×

Configure your new project

ASP.NET Core Web Application C# Linux macOS Windows Cloud Service Web

Project name

HtmlPageStartUp

Location

D:\aJohn20\jQueryCore3\jQueryCore3Source\01 - First JavaScript\HtmlPageStartUp

...

Back Create

Figure: Adding a new project

On the next Screen Select **Empty** project and un-check **Configure for HTTPS** as shown in the below screen capture.

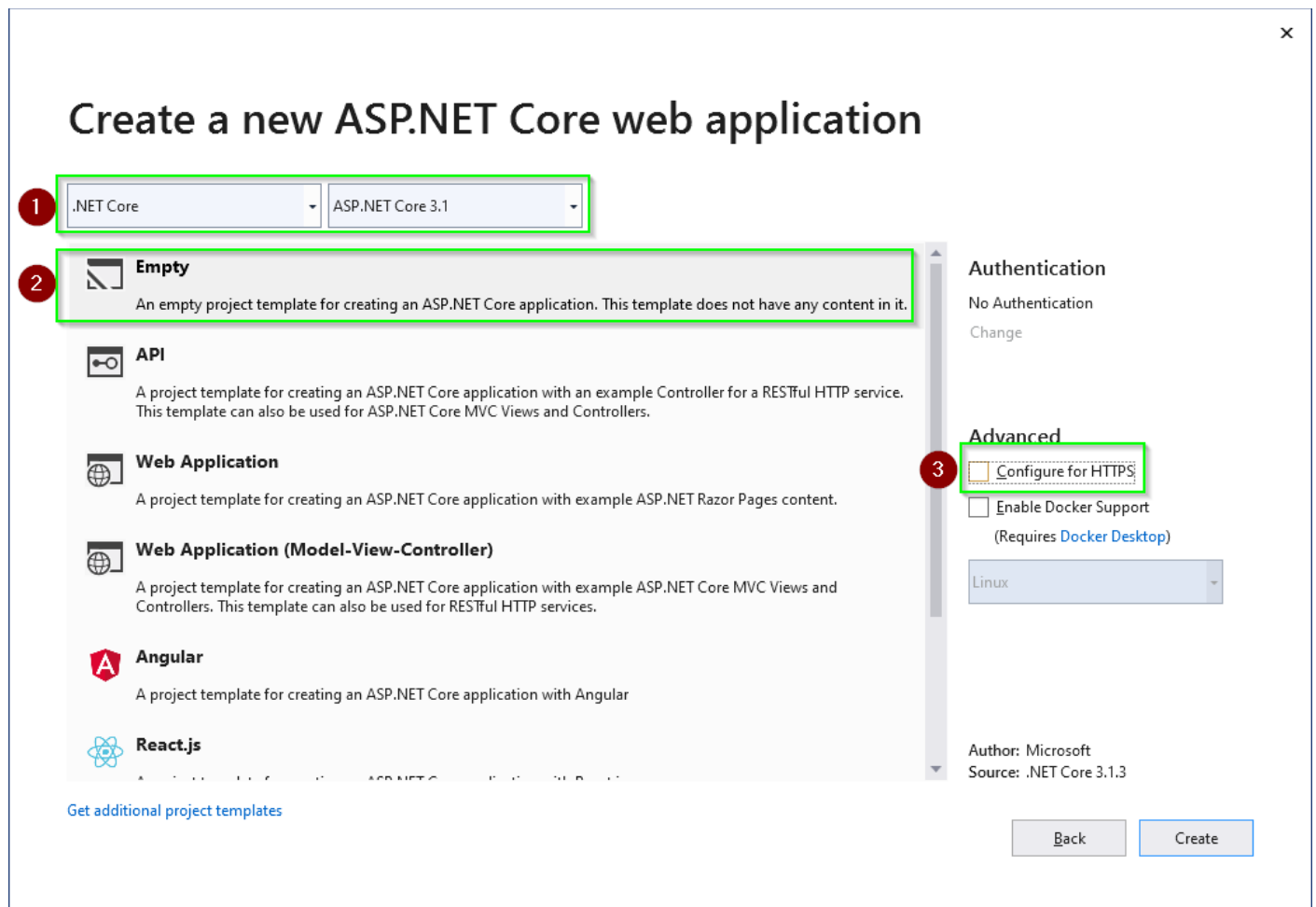


Figure: Creating an empty ASP.NET Core Application.

Compile and run the application.

Here is our first web apps result.

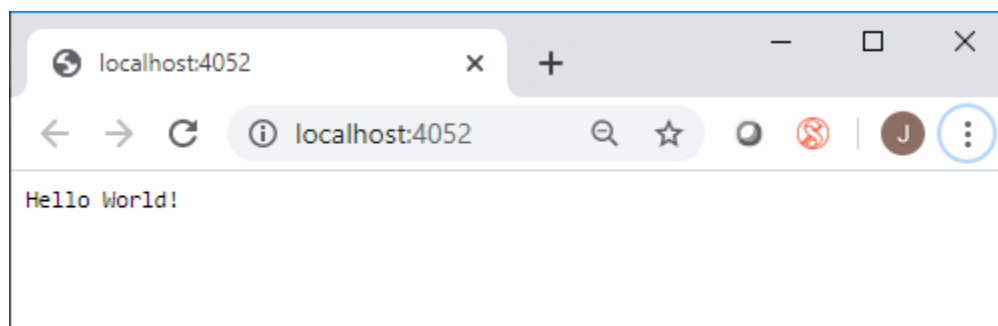
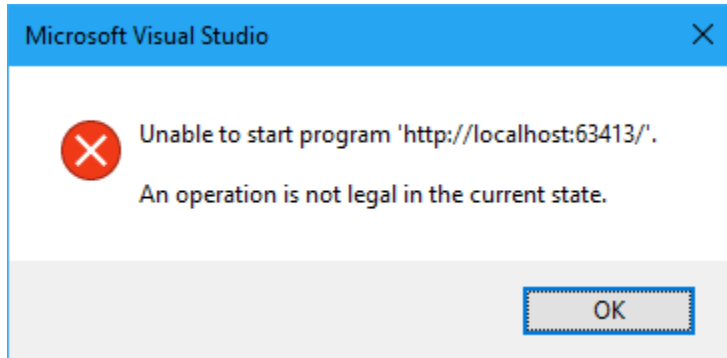


Figure: An empty ASP.NET Core Application HTML page.

If you get this error



Troubleshoot:

Change your Web browser from IIS Express dropdown menu.

Startup.cs File in the project

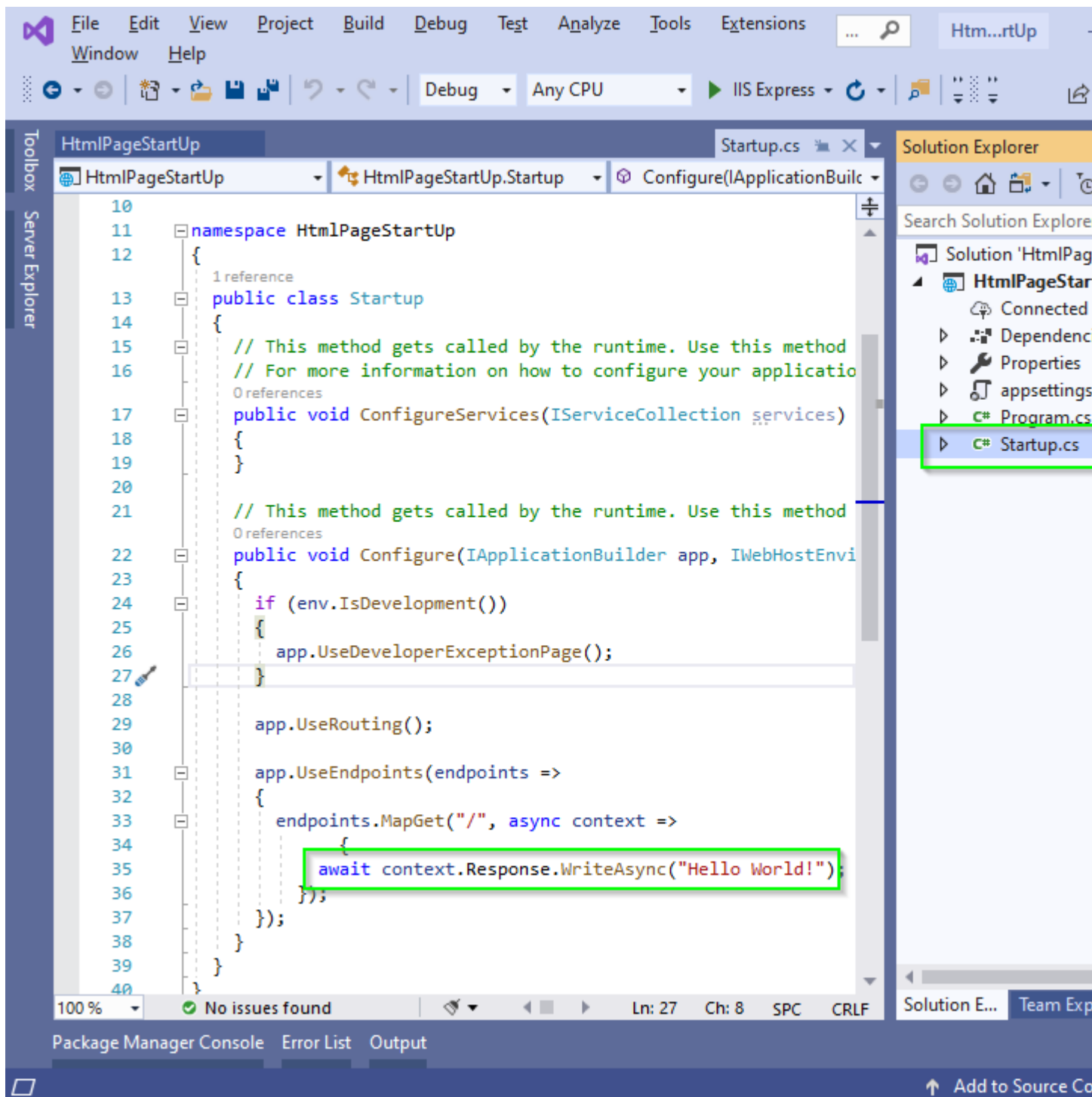


Figure: Startup file in the empty ASP.NET Core Application.

As we can see there is no web page in the solution! The page is rendered in Startup.cs file configure method via `context.Response.WriteAsync("Hello World!");`

Interview Question.

```
What is the entry point of ASP.NET Core Empty Web Application?  
Program.Main: CreateHostBuilder: UseStartup<Startup>()-->Startup.Configure:  
app.UseEndpoints: context.Response.WriteAsync("Hello World!");  
Short answer : Main --> Startup  
Startup is the entry point of the ASP.NET application.  
Angular is framework, jQuery is a JavaScript library. Latest Angular is written with  
TypeScript.
```

Serving static files

Static files are stored within the project's web root directory. The default directory is {content root}/**wwwroot**, but it can be changed via the `UseWebRoot` method.

Static files are accessible via a path relative to the web root. For example, the Web Application project template contains several folders within the `wwwroot` folder:

```
wwwroot  
• css  
• images  
• js
```

Static pages are put in `wwwroot` directory.

Add `app.UseStaticFiles();`

This will enable project to use static css files, images, JavaScript libraries including jQuery library.

Add `app.UseDefaultFiles();`

Configuring Controllers with Views.

Add `services.AddControllersWithViews();` method in the `ConfigureServices`.

Add below code segment:

```
app.UseEndpoints(endpoints =>  
{  
    endpoints.MapControllerRoute(  
        name: "default",  
        pattern: "{controller=Home}/{action=Index}/{id?}");  
});
```

In to end of the `Configure` method.

Using `app.useEndpoint` we configured the Home index as default page.

Here is the updated Startup page.

Listing: Adding `app.UseDefaultFiles();` and `app.UseStaticFiles();` in Startup.cs
HtmlPageStartUp\HtmlPageStartUp\Startup.cs

```
using Microsoft.AspNetCore.Builder;
using Microsoft.AspNetCore.Hosting;
using Microsoft.AspNetCore.Http;
using Microsoft.Extensions.DependencyInjection;
using Microsoft.Extensions.Hosting;

namespace HtmlPageStartUp
{
    public class Startup
    {
        // This method gets called by the runtime. Use this method to add services to
        // the container.
        // For more information on how to configure your application, visit
        // https://go.microsoft.com/fwlink/?LinkID=398940
        public void ConfigureServices(IServiceCollection services)
        {
            services.AddControllersWithViews();
        }

        // This method gets called by the runtime. Use this method to configure the HTTP
        // request pipeline.
        public void Configure(IApplicationBuilder app, IWebHostEnvironment env)
        {
            if (env.IsDevelopment())
            {
                app.UseDeveloperExceptionPage();
            }

            // Look for default files in wwwroot like index.html, default.html etc
            app.UseDefaultFiles();
            // Lets application to use static files such as css, images, js etc
            app.UseStaticFiles();

            app.UseRouting();

            app.UseEndpoints(endpoints =>
            {
                endpoints.MapControllerRoute(
                    name: "default",
                    pattern: "{controller=Home}/{action=Index}/{id?}");
            });
        }
    }
}
```

For this example, we want to use html files(index.html) instead of cshtml (Index.cshtml) files.

Add a new folder called **wwwroot**.

Add an **Index.html** page into wwwroot directory

File location: **CodecampJqCore2\HtmlPageStartUp\wwwroot**

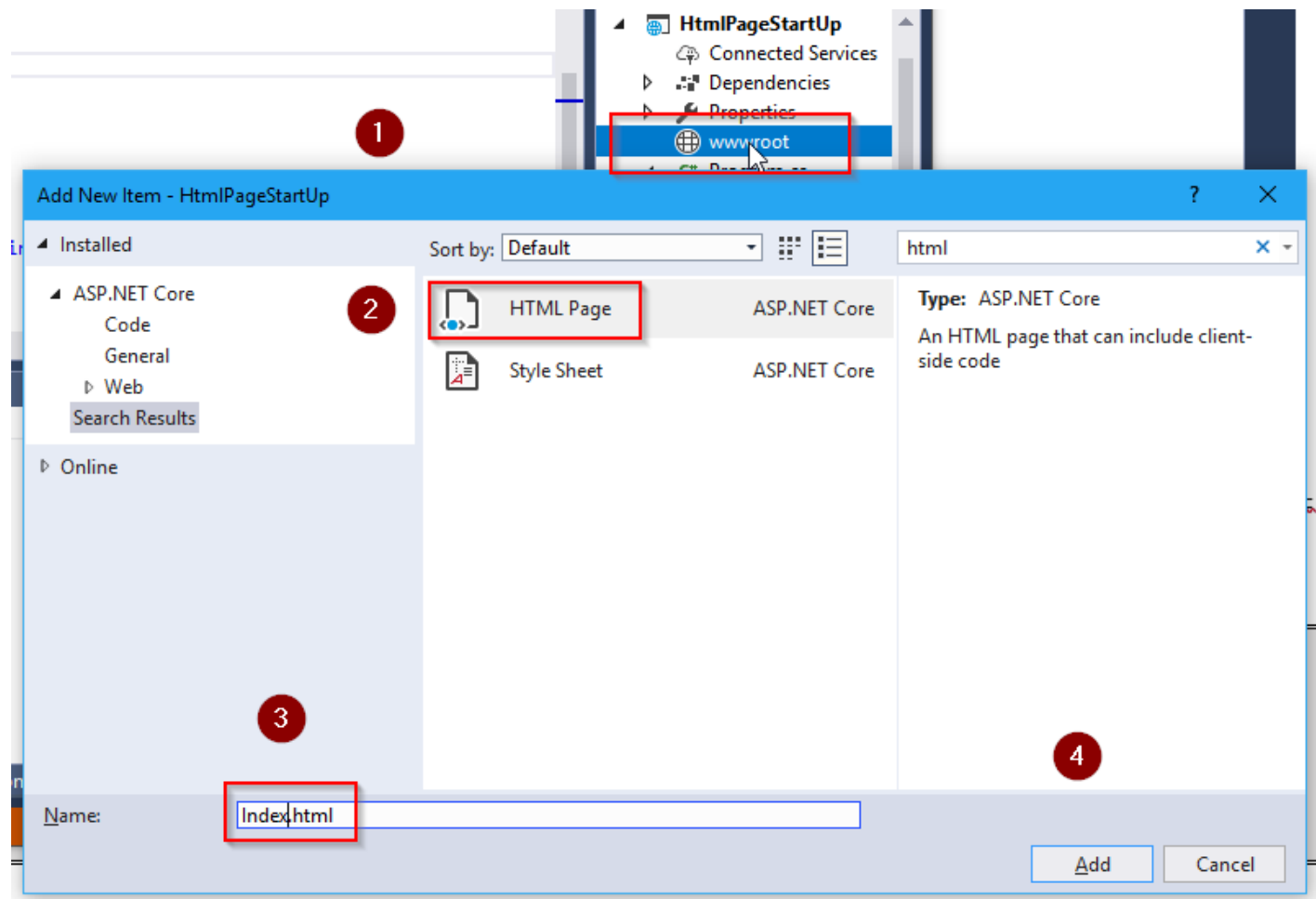


Figure: Adding HTML Page in wwwroot directory.

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8" />
  <title></title>
</head>
<body>
  <h3>My First HTML Page</h3>
</body>
</html>
```

Listing: Index.html file.

Compile and run the project. We can see that index.html file started as default file. See the figure below.

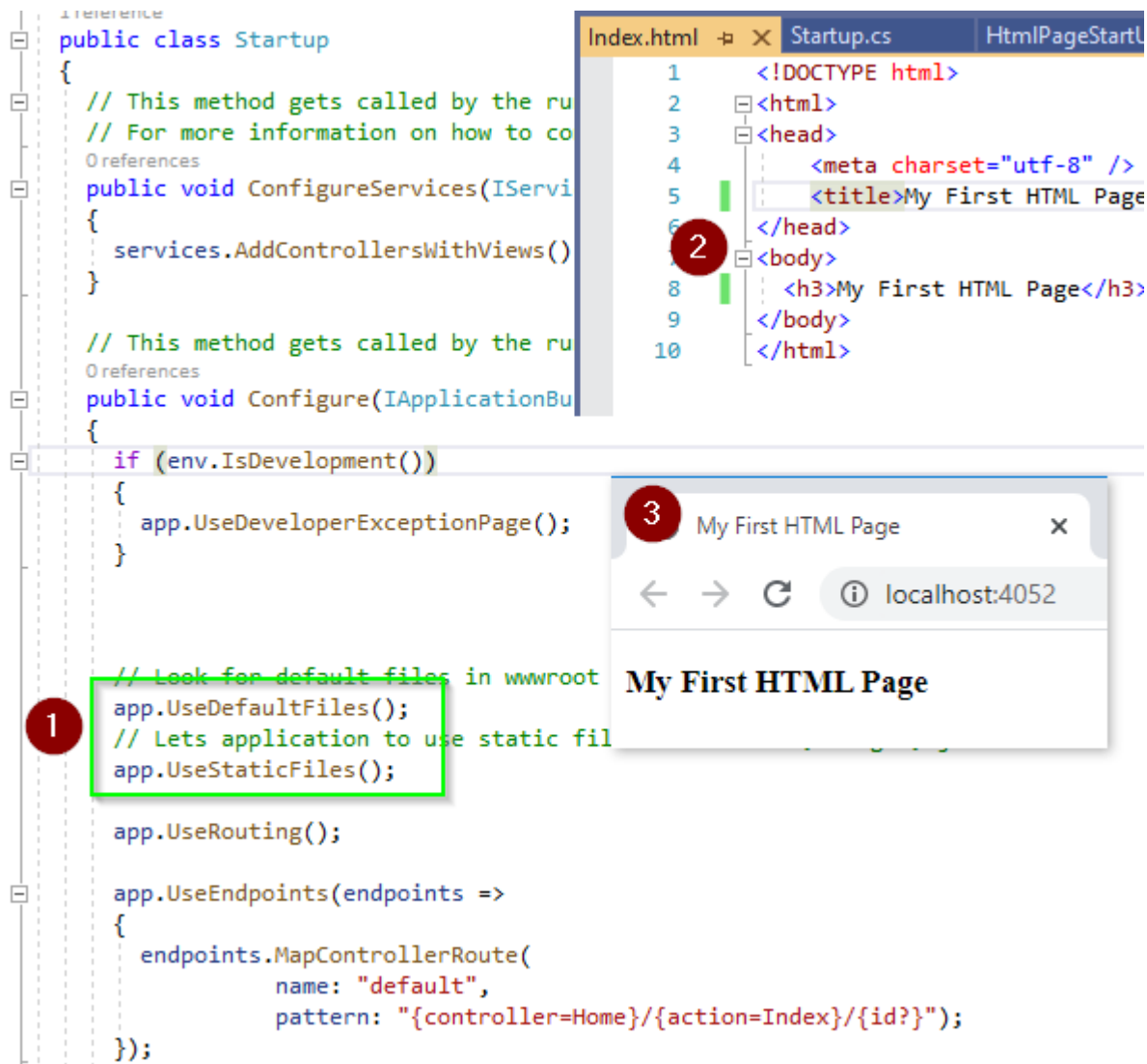


Figure: Setting up default startup file as index.html in `wwwroot` directory and running the application on the first time.

Note: Asp.NET Core Middleware order is important, if we put `app.useRouting()` before the `app.UseStaticFiles()` method, the application will look at the default controller first and our static index.html file may not run.

Add a new folder

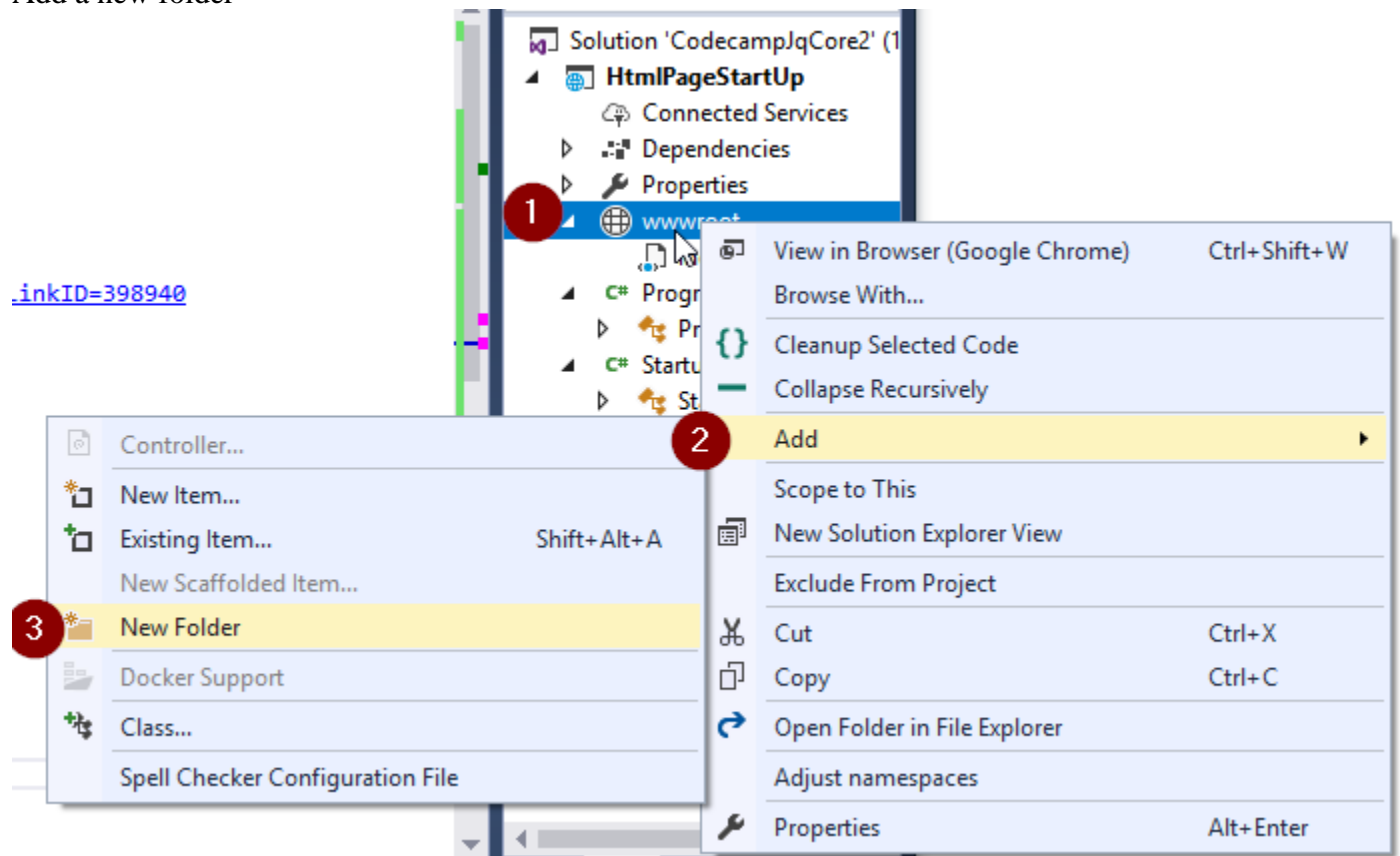


Figure: Adding a new Folder.

Name the new folder as **JsExamples**. We will put all our JavaScript Examples in the **JsExamples** folder.

Folder location: **HtmlPageStartUp\wwwroot\JsExamples**

The next chapter we will continue using this project for JavaScript and jQuery learning and examples.

Summary

In this article, we have learned:

- Creating a Blank Solution
- Creating an Empty ASP.NET Core Web Application
- Setting a HTML file as startup file.
- `app.UseStaticFiles()`
- `app.UseDefaultFiles();`
- Creating wwwroot directory
- Setting Default controller and view
- `services.AddControllersWithViews();`
- `app.UseRouting();`
- `app.UseEndpoints`
- Entry point of ASP.NET Core
- Startup.cs file

CH 2- JavaScript Crush Course

Introduction

In this session, we will learn:

- Include Script on a HTML Page
- Reference an External Script on a HTML Page
- How to add JavaScript file to a project.?
- `document.write()`
- `document.writeln()`
- `console.log()`
- `alert()`
- Developer Tools
- JavaScript definition
- Statements
- Comments
- Functions
- Parameters and Return Values
- Working with Objects
- Json
- XML
- The Equality Operator vs. the Identity Operator
- Converting Numbers to String
- Converting Strings to Numbers
- Using single quote vs double quote

If you want to learn jQuery, then you will need to know JavaScript. However, you don't have to be a JavaScript expert. If you already know JavaScript very well, you can skip this chapter and use as need bases.

When compared to many other programming languages, such as C#, C++ and Java, JavaScript is very easy to pick-up and use it.

Recommended: Download this Chapter's source code and use as reference.

Include Script on a HTML Page

Add an HTML page into `JsExamples` folder and name it `ScriptOnaPage.html`.

Location: HtmlPageStartup\wwwroot\JsExamples

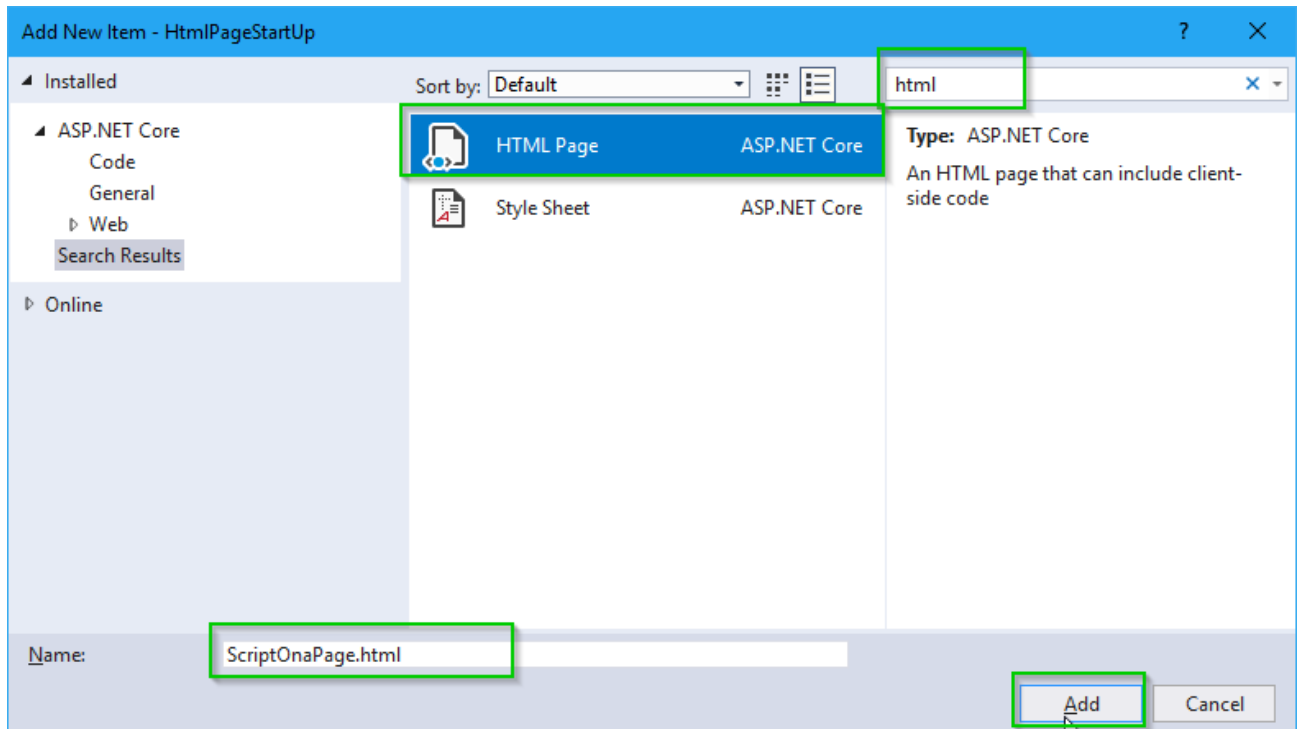


Figure: Adding the **ScriptOnaPage.html**

Somehow, we need to tell the browser that it has to process your JavaScript. To accomplish this, we use the **script** tag.

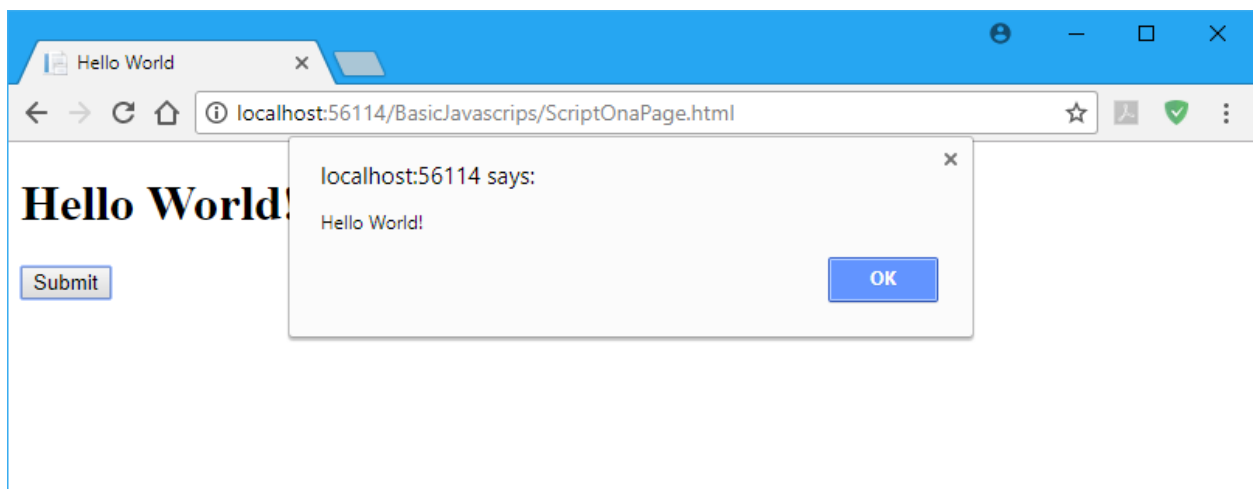


Figure: Running Hello World HTML page.

Listing : The on page Embedded Script in the ScriptOnaPage.html page.
HtmlPageStartUp\HtmlPageStartUp\wwwroot\JsExamples\ScriptOnaPage.html

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8" />
  <title>Hello World!</title>
</head>
<body>
  <h1>Hello World </h1>
  <!--script tag tells browser to register JavaScript blogs-->
  <input type="button" value="Submit" onclick="myFunction()" />
  <script type="text/javascript">
    function myFunction() {
      alert("Hello World!");
    }
  </script>
</body>
</html>
```

TIP: We will put all our JavaScript Examples in the **JsExamples** folder. (**wwwroot/JsExamples**) directory.

Reference an External Script on a HTML Page

We will run the same example with separating JavaScript into an external **ReferenceExternalScript.js** JavaScript file.

Add a new **ReferenceExternalScript.js** JavaScript file in to
HtmlPageStartUp\wwwroot\JsExamples directory
And cut myFunction() from ScriptOnaPage.html to paste in **ReferenceExternalScript.js**

Listing: **ReferenceExternalScript.js** file

```
HtmlPageStartUp\HtmlPageStartUp\wwwroot\JsExamples\ReferenceExternalScript.js
function myFunction() {
  alert("Hello World!");
}
```

Listing: *ReferenceExternalScript.html* file.

HtmlPageStartUp\HtmlPageStartUp\wwwroot\JsExamples\ReferenceExternalScript.html

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8" />
  <title>Hello World</title>
</head>
<body>
  <h1>Hello World!</h1>
  <input type="button" value="Submit" onclick="myFunction()" />

</body>
</html>
<!-- reference the ReferenceExternalScript.js script file -->
<script src="ReferenceExternalScript.js"></script>
```

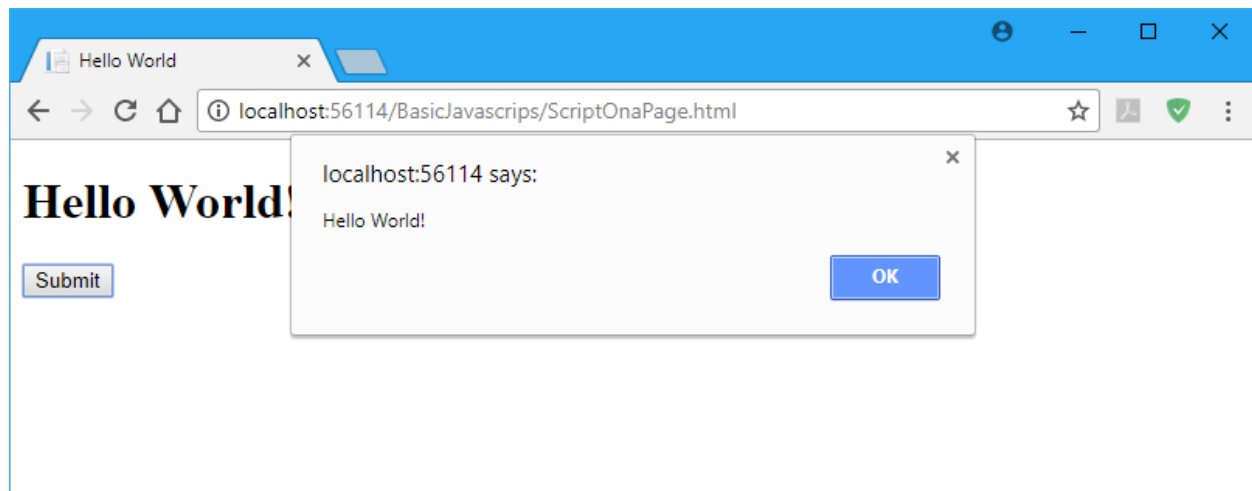


Figure: Running Hello World HTML page.

How to add JavaScript file to a project.?

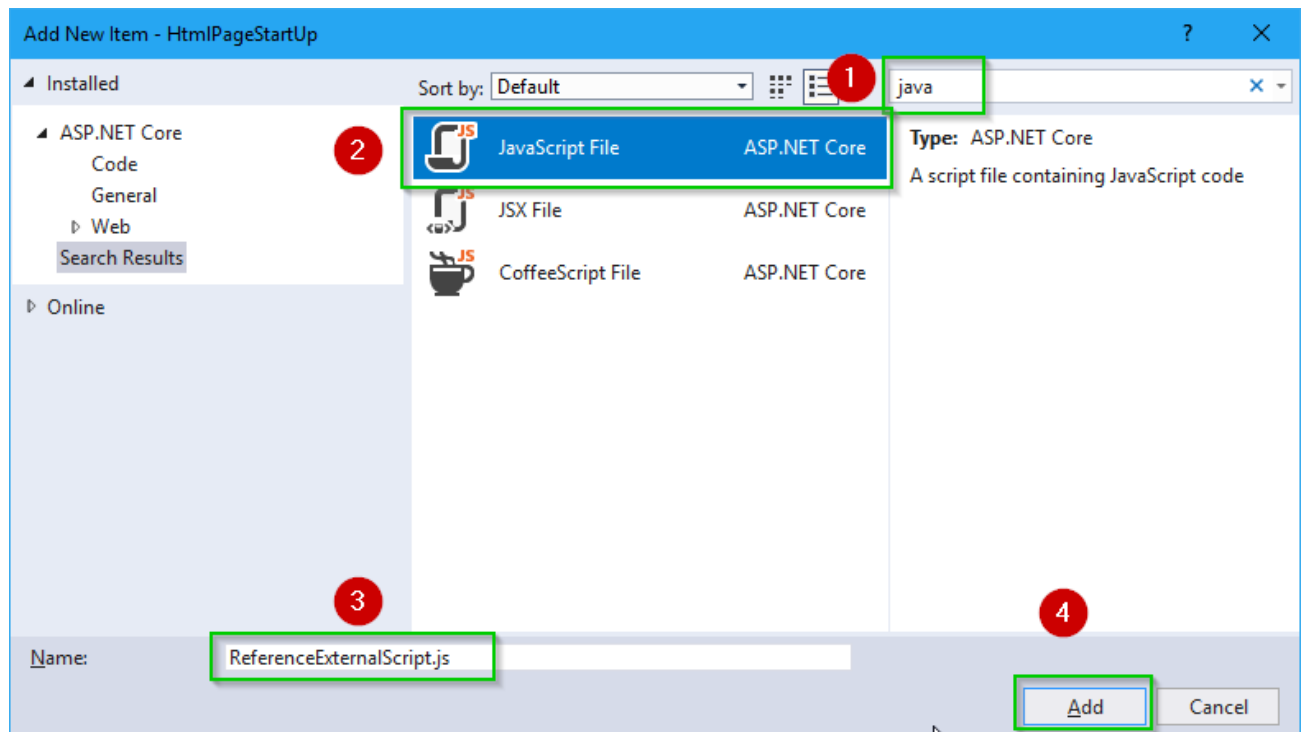


Figure: Adding a JavaScript file to the project.

To add a JavaScript file, right mouse click on the JsExample folder and select Add -->New Item from the menu. Then, select JavaScript File and name it ***ReferenceExternalScript.js***.

Listing: *ReferenceExternalScript.js* file

HtmlPageStartUp\HtmlPageStartUp\wwwroot\JsExamples\ReferenceExternalScript.js

```
function myFunction() {  
    alert("Hello World!");  
}
```

- *document.write()*: JavaScript methods that write to html page.
- *document.writeln()*: JavaScript methods that write to html page.
- *console.log()*: console.log() function writes to the browser console window.
- *alert()*: JavaScript windows alert() function, writes to the popup alert box window.

The below figure (console, alert and document write) shows visually how write to the html, console and alert windows.

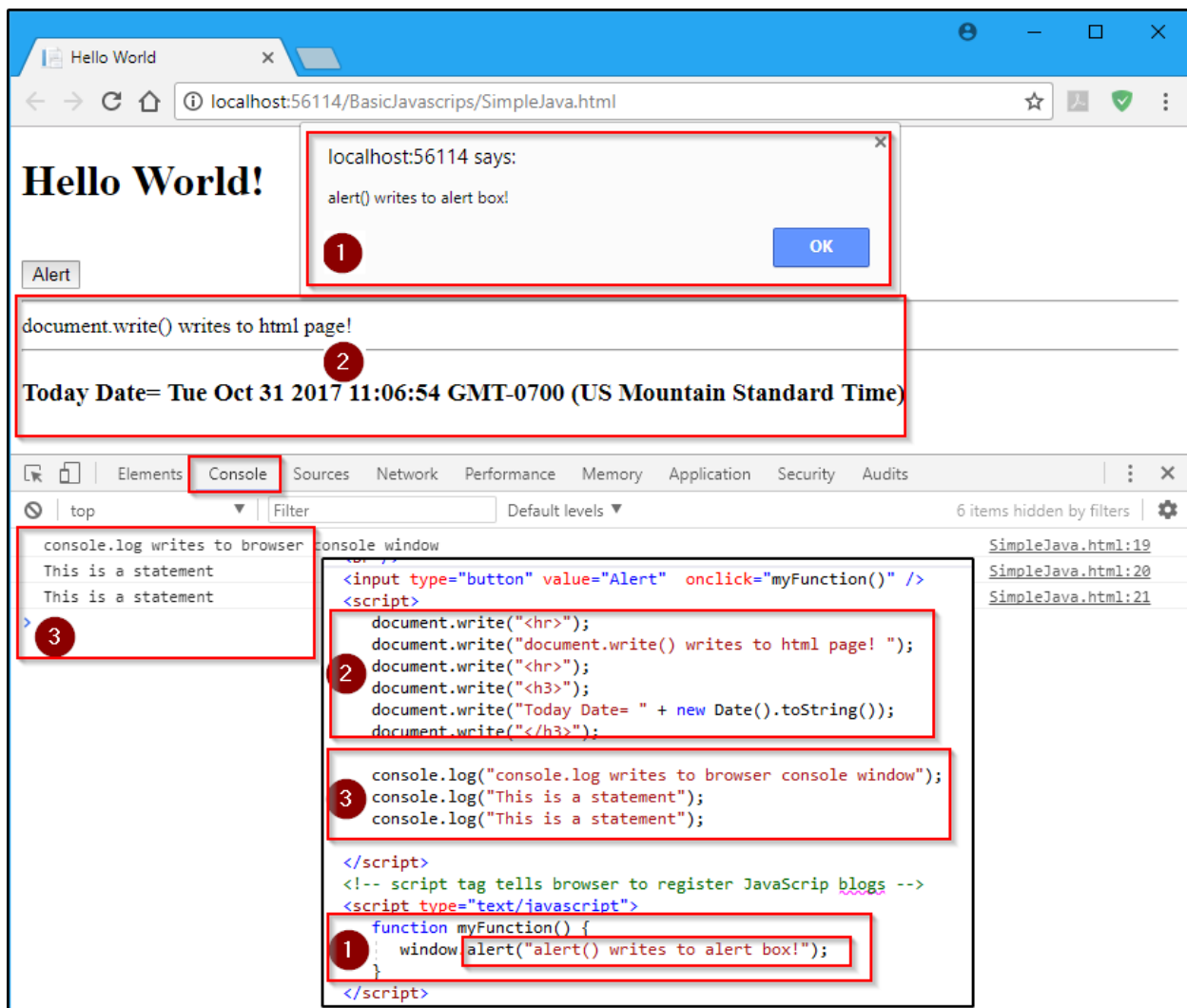


Figure: console, alert and document write. (SimpleJava.html)

Interview Question:

How to write on a html page after page rendered?

One way, we can write to the html page is using javascript `document.write()` or `document.writeln()` method

TIP: All JavaScript Example files for this project will be put in `wwwroot/JsExamples` directory.

Browser Developer Tools

Best practices: Developer Tools can be very helpful when you developing web pages. Pressing F12 brings to developer tools. And choose Console Tab to see if any client error happens.

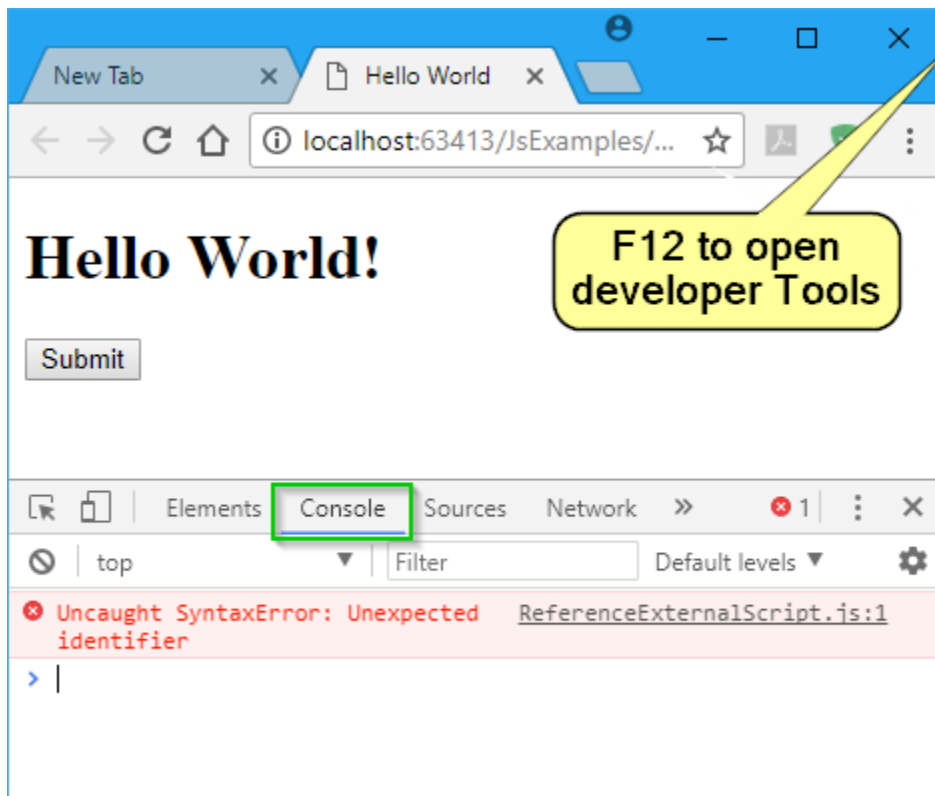


Figure: How to open developer tool.

What is JavaScript?

JavaScript are combination of statements and collections of expressions.

Statements

Statement are usually exist on its own line and, ends with optional semicolon.

```
<script type="text/javascript">
    // This is a single line comment
    function myFunction() {
        window.alert("alert() writes to alert box!");
    }
```

Listing: Example of statements.

Using multiline Comments

Multiline JavaScript comments start with `/*` and end with `*/`.

```
<body>
    <h1>Hello World!</h1>
    <br />
    <input type="button" value="Alert" onclick="myFunction()" />
    <script>
        document.write("Today Date= " + new Date().toString());

    </script>
    <!-- This is a html multiline comments
        script tag tells browser to register JavaScript blogs -->

    <script type="text/javascript">
        /* This is a multiline comment
        starts with slash start
        function myFunction() {
            window.alert("alert() writes to alert box!");
        }

        Multiline comments ends with * slash
        */

        console.log("console.log writes to browser console window");
        console.log("This is a statement");
        console.log("This is a statement");
    </script>
</body>
```

Listing: Using comments

Functions

A function is a block of JavaScript code that is defined once but can be executed, or invoked, any number of times.

To create a function use JavaScript keyword `function` following with a function name and put the function code between a pair of curly braces.

Listing: Simple JavaScript function.

```
function myFunction(name) {  
    alert("Hello " + name);  
}
```

Functions can be defined with a build-in JavaScript function constructor and it is called `Function()`.

Listing: Creating function using the `Function()` Constructor.

```
var myAddFunction = new Function(x, y) {  
    return x + y;  
}  
  
var returnValue = myAddFunction(2, 3);
```

Or we can omit `new` and use lower case function. This is the same with above function.

Listing: Creating function using the `Function()` Constructor.

```
var myAddFunction = function(x, y) {  
    return x + y;  
}  
  
var returnValue = myAddFunction(2, 3);
```

Parameters and Return Values

Listing: Parameters and arguments

```
function add(x, y) { // x and y are parameters
  return x + y;
}

var returnValue = add(2, 3); // 2 and 3 are arguments
```

In the above add function, we return `x + y` value back to the caller.

Parameters: In this function `x` and `y` called the parameter. Parameters allows us the pass values to the function.

Arguments: Arguments are the values that get passed in to the be assigned to these variables.

Interview Question: Are parameters and arguments the same?

They are slightly different things. Argument are get passed into the function when you call the function. Parameters are on the function declaration variables.

Working with Objects

Creating object

There are different ways to create JavaScript objects. Below listing shows two different ways of creating objects. First way creates an empty object and then adds member. Second way is the creating object with literal.

Note: Many beginner developers confuse JavaScript object syntax with JSON syntax.

Listing: Creating objects (Json.html)

HtmlPageStartUp\HtmlPageStartUp\wwwroot\JsExamples\Json.html

```
// employee object
var employee = {};
employee.Name = "Mike Koc";
employee.City = "Los Angeles";
employee.State = "LA";

// using object literals
var myObject = {
    name: "Nancy",
    city: "New York"
};

// Json string
var employee= {
    "firstName": "John",
    "lastName": "Kocer",
    "address": {
        "streetAddress": "One Microsoft Way",
        "city": "Los Angeles",
        "state": "LA",
        "postalCode": 80021
    },
    "phoneNumbers": [
        "212 555-4434",
        "846 555-9967"
    ]
}

return employee;
```

Interview Question:

What are the difference between JavaScript Object literal vs. JSON string?

Let's start with syntax. JavaScript literal uses memberName = and the value in double quote.

JSON string use memberName and values both in *quotes* and key value pair with colon.

JSON is a data interchange format. It's a standard that describes how ordered lists and unordered maps, strings, booleans and numbers can be represented in a string. Just like XML, it is a way to pass structured information between languages. A JavaScript object on the other hand is physical type, just like C# class or interface.

Listing: JSON has no way to represent a function (or dates for that matter)

```
// JSON is just a way to pack an object into a string
// JSON has no way to represent a function (or dates for that matter)
JSON.stringify({
  foo: new Date(),
  blah: function () {
    alert('hello world');
  }
});
// returns the string '{"foo":"2017-11-01T12:01:27.856Z"}'
```

Interview Question:

What are JSON and XML? Which one to choose?

Both JSON and XML can be used to send and receive data from web server.

JSON and XML similarities

- Both Self describing, human readable
- Both hierarchical (values within values)
- Both can be parsed and used many programming languages

JSON and XML differences

- JSON is shorter
- JSON does not use end tag
- JSON is quicker to read and write
- JSON can use arrays
- JSON is more popular usually with mobile web computing
- JSON is faster, and has better performance
- XML has to be parsed with an XML parser. JSON can be parsed by a standard JavaScript function.

Figure: JSON string example

```
// JSON string example
{"employees":[
  { "firstName":"John", "lastName":"Koc" },
  { "firstName":"Jen", "lastName":"Jones" },
  { "firstName":"Mike", "lastName":"Rich" }
]}
```

Figure: XML string example

```
// XML string example
<employees>
  <employee>
    <firstName>John</firstName> <lastName>Koc</lastName>
  </employee>
  <employee>
    <firstName>Jen</firstName> <lastName>Jones</lastName>
  </employee>
  <employee>
    <firstName>Mike</firstName> <lastName>Rich</lastName>
  </employee>
</employees>
```

Which one to choose JSON or XML?

Choose JSON when is possible, because it is light weight and faster.

Best practices: Choose JSON when is possible, because it is light weight and faster.

The Equality Operator vs. the Identity Operator

Listing: The Equality Operator vs. the Identity Operator

```
var numberValue = 7;
var stringValue = "7";
if (numberValue == stringValue) {
    console.log("They are the same");
} else {
    console.log("They are NOT the same");
}
// output: They are the same
```

The output from == script:

They are the same

```
// JSON string example
var numberValue = 7;
var stringValue = "7";
if (numberValue === stringValue) {
    console.log("They are the same");
} else {
    console.log("They are NOT the same");
}
//output: They are NOT the same
```

Figure: Equality vs, identity operator(==, ===)

The output from === script:

They are NOT the same

Interview Question:

Can you tell me what is the difference between equality operator vs. identity operator?
Equality operator == does not check for type for example number= 7 and string = "7" are equal. However, identity operator checks for type and number= 7 and string = "7" are not equal because one is number type other is string type.

Converting Numbers to String

```
// JSON string example  
  
var myData = (7).toString() + String(7);  
    console.log(myData);  
//output: 77
```

The output from [this](#) script:

77

Converting Strings to Numbers

```
var firstString = "7";  
    var secondString = "7";  
    var result = Number(firstString) + Number(secondString);  
    console.log("Result: " + result);  
//output: Result: 14
```

The output from [this](#) script:

Result: 14

Listing: Checking for null or undefined (NullNumberConvert.html)

```
// Checking for null or undefined -----
var customer = {
    name: "John",
    city: null
};
if (!customer.name) {
    console.log("name IS null or undefined");
} else {
    console.log("name is NOT null or undefined");
}
if (!customer.city) {
    console.log("city IS null or undefined");
} else {
    console.log("city is NOT null or undefined");
}
}
//output:
```

Output:

name is NOT null or undefined
city IS null or undefined

What is the difference between using single and double quotes in js. Which should I use for writing web development? (SingleDoubleQuote.html)

HtmlPageStartUp\HtmlPageStartUp\wwwroot\JsExamples\SingleDoubleQuote.html

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8" />
    <!--<script src="lib/jquery/dist/jquery.js"></script>
    <script src="lib/jquery-validation/dist/jquery.validate.js"></script>-->
    <!--<script src="lib/jquery-validation/dist/additional-methods.js"></script>-->
    <title></title>
    <style>
        body {
            font-family: Arial, Sans-serif;
        }

        .error {
            color: red;
            font-family: verdana, Helvetica;
        }
    </style>
</head>
<body>
    Hello HTML Page!
    <hr />
    <div id="uiEmployeeDebug"></div>
    <hr />

    <hr />
    <div id="uiEmployeeTemp"></div>
    <hr />

    <input type="button" value="Submit" onclick="storeEmployeeInDivAsAttribute()" />
```

```

<hr />
<input type="button" value="SubmitRed"
onclick="storeEmployeeInDivAsAtributeDoubleQoute()" />
<script>
    // -----
    var stringValue1 = 'abc';
    var stringValue2 = "abc";
    if (stringValue1 == stringValue2)
        console.log("== They are the same")
    else
        console.log("== They are NOT the same")

    if (stringValue1 === stringValue2)
        console.log("=== They are the same")
    else
        console.log("=== They are NOT the same")
    /// == They are the same
    /// === They are the same

    //document.getElementById("uiEmployeeDebug").html(JSON.stringify(employee));
</script>
</body>
</html>
<script type="text/javascript">
    var employee = {
        name: 'John',
        city: 'Raleigh',
        age: 20
    };

    function storeEmployeeInDivAsAtribute() {
        var employeeStringify =
document.getElementById("uiEmployeeDebug").innerHTML = JSON.stringify(employee);

        var panelDiv = '<div id="uiPanel" data-name=' + employee.name + ' data-
city=' + employee.city + ' data-age=' + employee.age + ' >' + employeeStringify +
'</div>';

        document.getElementById("uiEmployeeTemp").innerHTML = panelDiv;
        console.log(document.getElementById("uiEmployeeTemp").innerHTML);

        //var employeeName =
document.getElementById("uiEmployeeTemp").getAttribute("data-name");
        var employeeName = document.getElementById("uiPanel");
        console.log('employee.name= ' + employeeName.dataset.name);

    }

    function storeEmployeeInDivAsAtributeDoubleQoute() {
        var employeeStringify =
document.getElementById("uiEmployeeDebug").innerHTML = JSON.stringify(employee);

        var panelDiv = "<div id=\"uiPanel\" style=\"color:red\" data-name=" +
employee.name + " data-city=" + employee.city + " data-age=" + employee.age + " >" +
employeeStringify + "</div>";

        document.getElementById("uiEmployeeTemp").innerHTML = panelDiv;
        console.log(document.getElementById("uiEmployeeTemp").innerHTML);

        //var employeeName =
document.getElementById("uiEmployeeTemp").getAttribute("data-name");
        var employeeName = document.getElementById("uiPanel");
        console.log('employee.name= ' + employeeName.dataset.name);

```

```
}  
</script>
```

Listing: Using single quote vs double quote

```
var panelDivS = '<div id="uiPanel" data-name=' + employee.name + ' data-city=' +  
employee.city + ' data-age=' + employee.age + ' >' + employeeStringify + '</div>';  
  
var panelDivD = "<div id=\"uiPanel\" style=\"color:red\" data-name=" +  
employee.name + " data-city=" + employee.city + " data-age=" + employee.age + " >" +  
employeeStringify + "</div>";
```

In JavaScript, there is no significant difference between using single quote or double quote. In practice single quotes are just more convenient and faster to write. Also, keep in mind that when you get to formation JSON data, valid JSON only takes the double quote.

Best practices: Choosing single quote is faster, more readable and convenient. Single quotes are more common.

Summary

In this article, we have learned:

- Include Script on a HTML Page
- Reference an External Script on a HTML Page
- How to add JavaScript file to a project.?
- `document.write()`
- `document.writeln()`
- `console.log()`
- `alert()`
- Developer Tools
- JavaScript definition
- Statements
- Comments
- Functions
- Parameters and Return Values
- Working with Objects
- Json
- XML
- The Equality Operator vs. the Identity Operator
- Converting Numbers to String
- Converting Strings to Numbers
- Using single quote vs double quote

