## Full Stack Web Development

KLIANT

# Getting Started with ASP.NET Core

# Course Map: Day 1

1. ASP.NET Core Architecture
2. **Getting Started with ASP.NET Core**
3. Design Patterns, Unit Testing
4. Entity Framework Core

## Course Map: Day 2

5. Introduction to TypeScript

6. Using VS Code for TypeScript

7. Angular 2 Architecture

8. Using Angular CLI for Client Apps

# Agenda

- Installation and Setup
- Getting Started with ASP.NET Core
- Building a RESTful Web API
- Dependency Injection
- Configuration

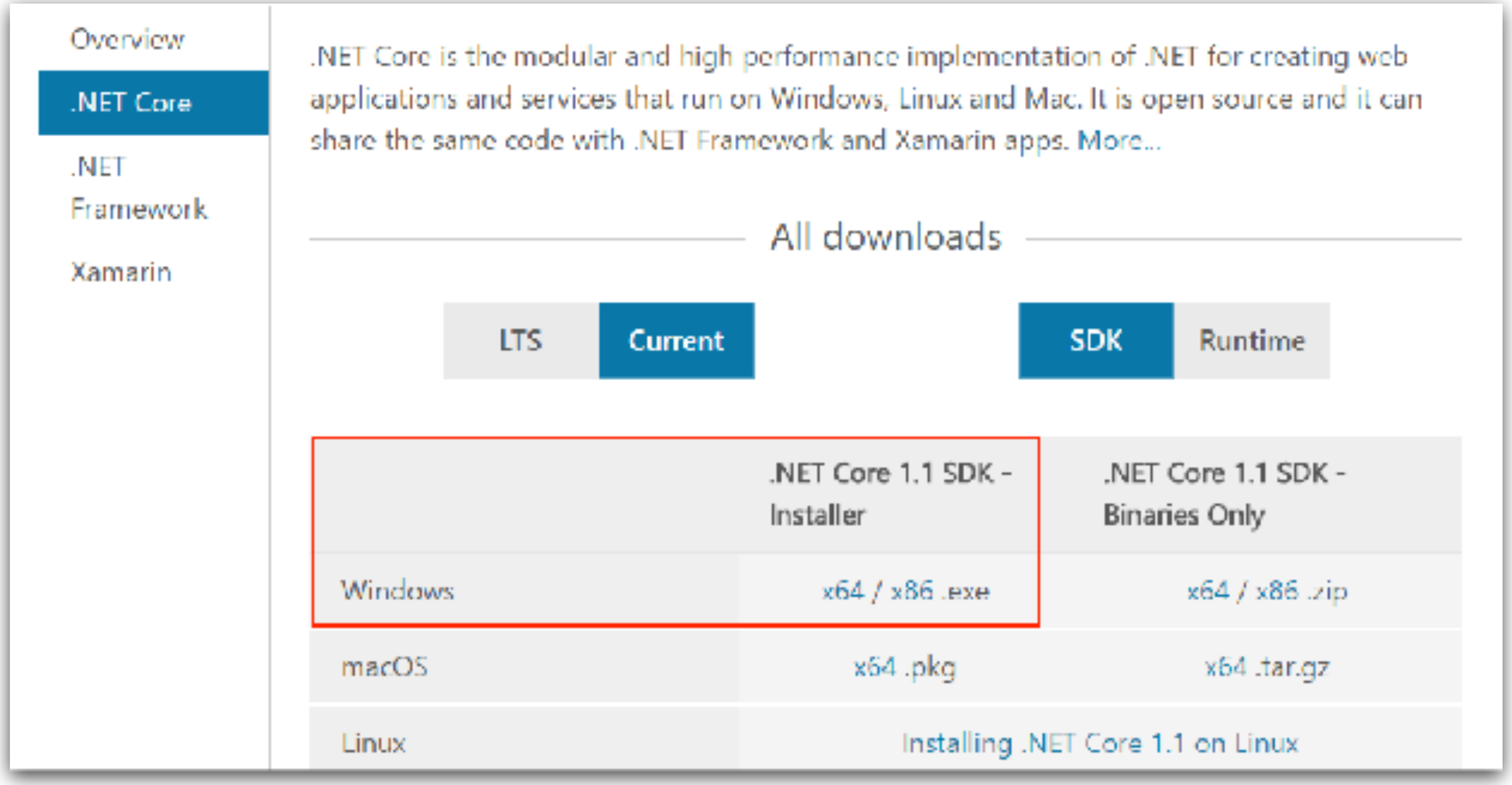
## Get the Bits

github.com / tonysneed /

Kliant.AspNetCore-Angular

## Installation and Setup

# Install .NET Core SDK

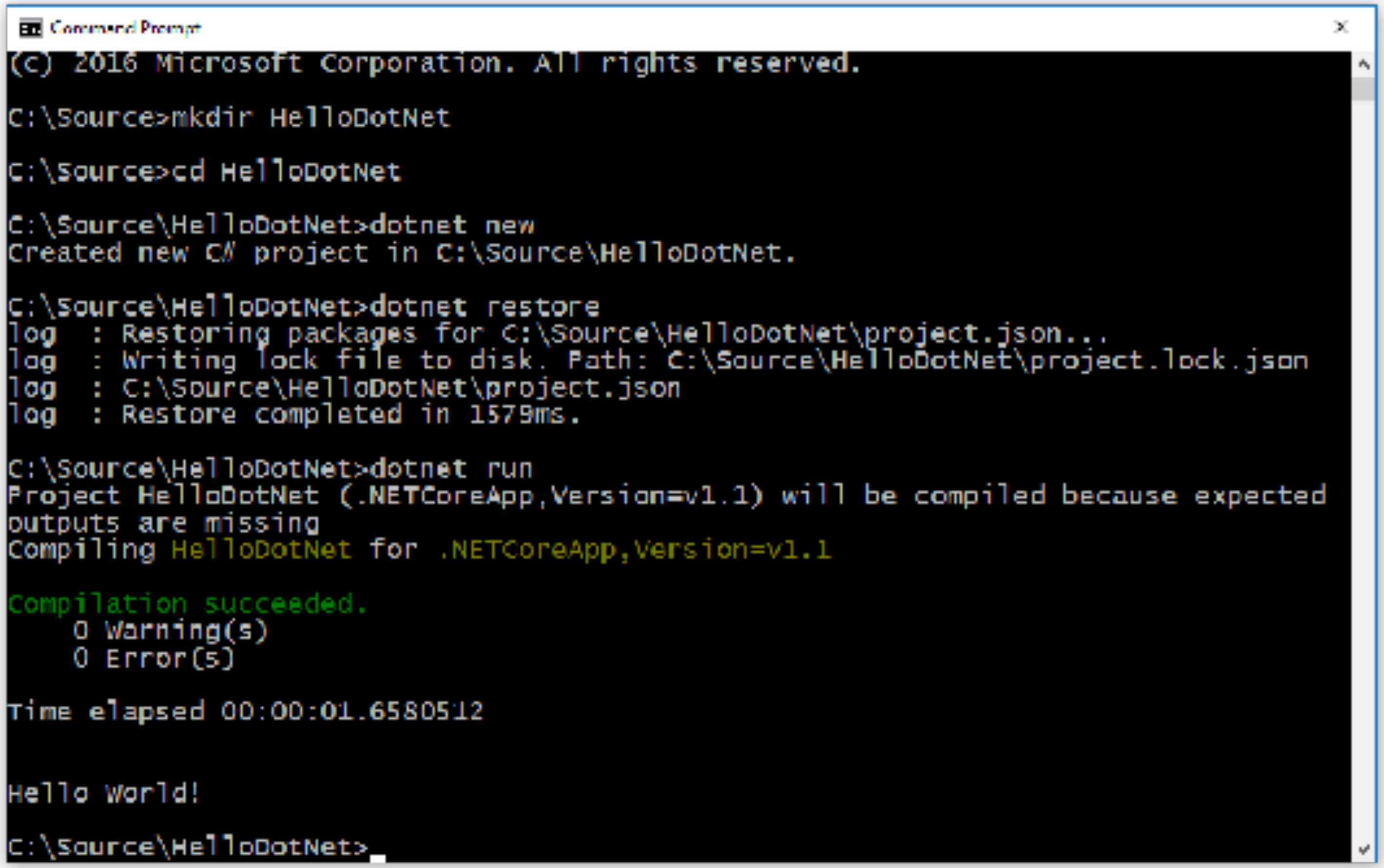


.NET Core is the modular and high performance implementation of .NET for creating web applications and services that run on Windows, Linux and Mac. It is open source and it can share the same code with .NET Framework and Xamarin apps. More...

All downloads

| | .NET Core 1.1 SDK - Installer | .NET Core 1.1 SDK - Binaries Only |
|---|---|---|
| Windows | x64 / x86 .exe | x64 / x86 .zip |
| macOS | x64 .pkg | x64 .tar.gz |
| Linux | Installing .NET Core 1.1 on Linux | |

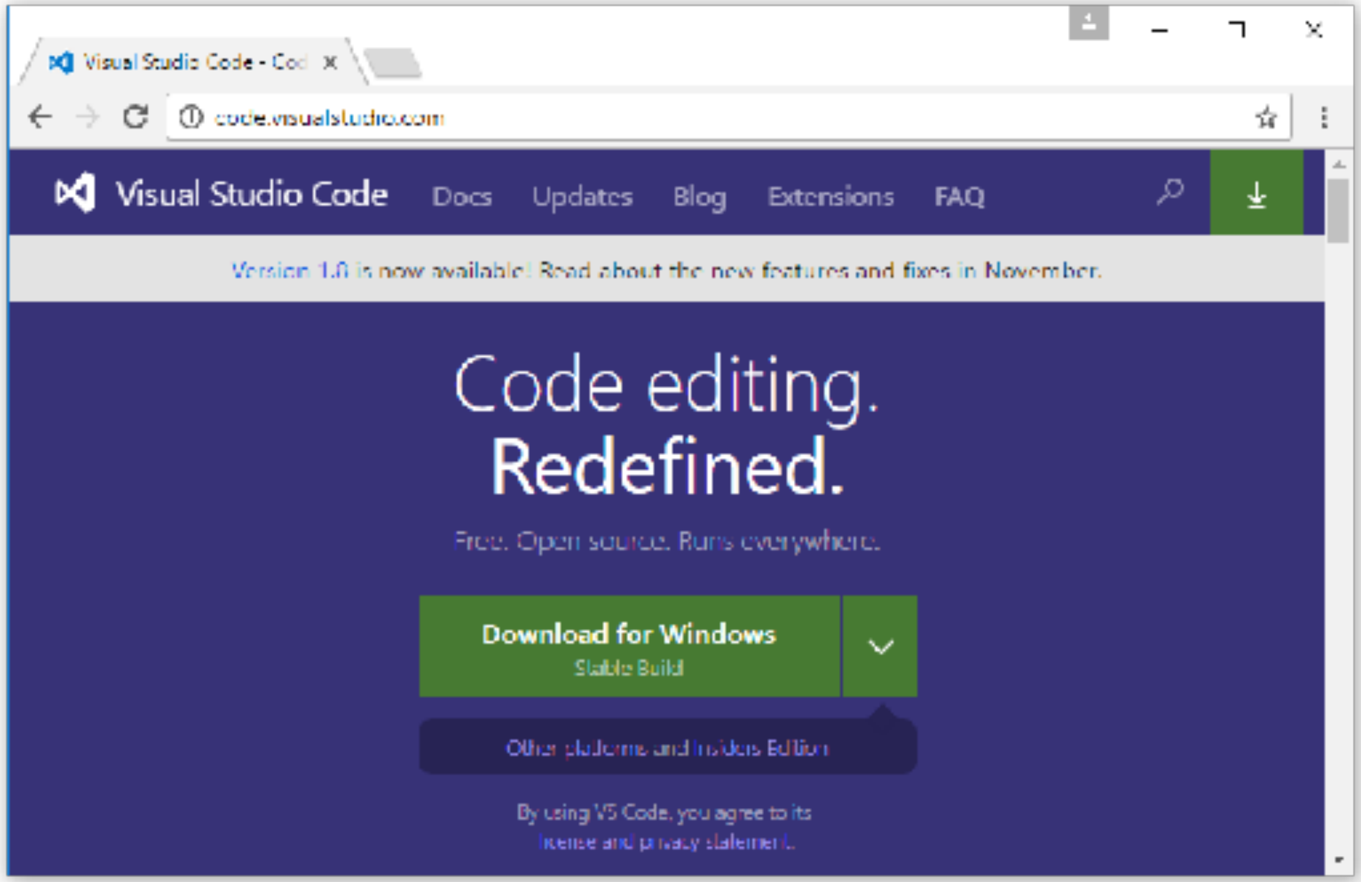
# Your First .NET Core App



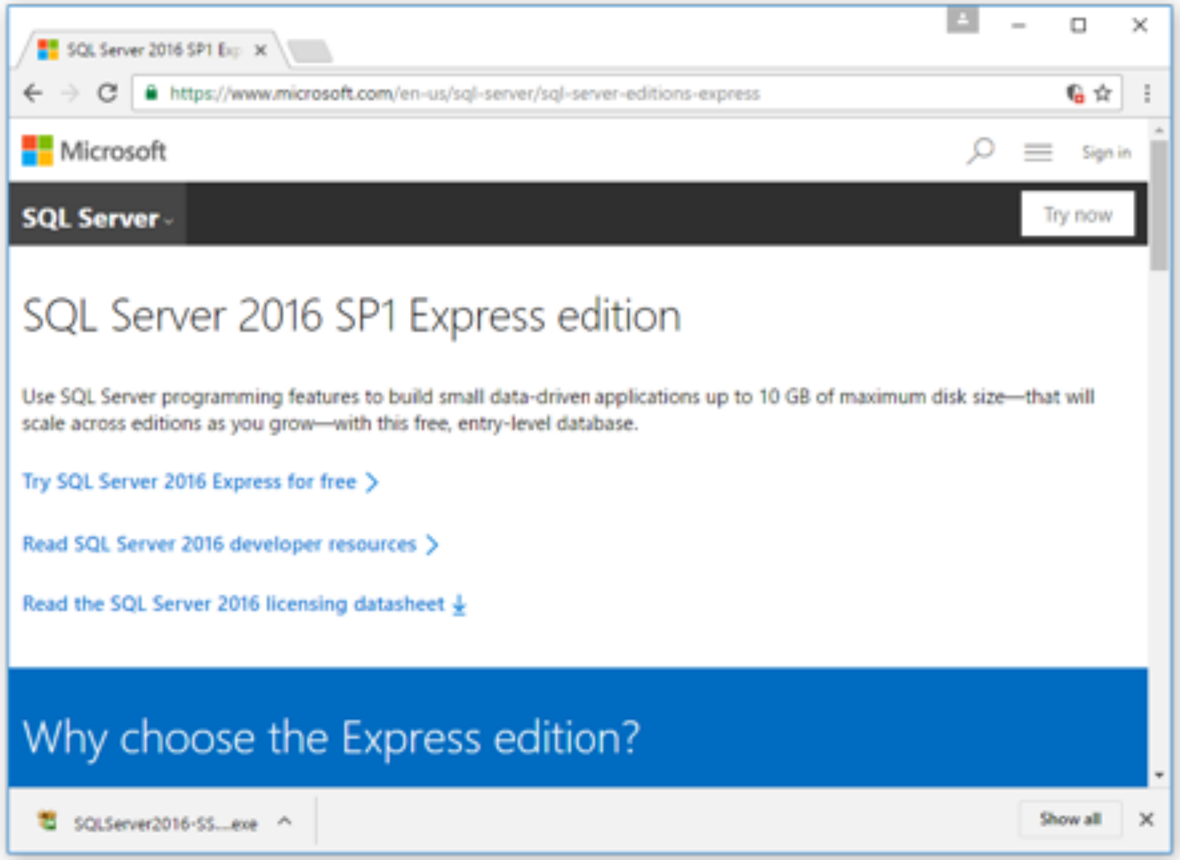
```
(c) 2016 Microsoft Corporation. All rights reserved.

C:\Source>mkdir HelloDotNet

C:\Source>cd HelloDotNet

C:\Source\HelloDotNet>dotnet new
Created new C# project in C:\Source\HelloDotNet.

C:\Source\HelloDotNet>dotnet restore
log  : Restoring packages for C:\Source\HelloDotNet\project.json...
log  : Writing lock file to disk. Path: C:\Source\HelloDotNet\project.lock.json
log  : C:\Source\HelloDotNet\project.json
log  : Restore completed in 1579ms.

C:\Source\HelloDotNet>dotnet run
Project HelloDotNet (.NETCoreApp,Version=v1.1) will be compiled because expected
outputs are missing
Compiling HelloDotNet for .NETCoreApp,Version=v1.1

Compilation succeeded.
    0 Warning(s)
    0 Error(s)

Time elapsed 00:00:01.6580512


Hello World!

C:\Source\HelloDotNet>_
```

- mkdir, cd
- dotnet new
- dotnet restore
- dotnet run

## Visual Studio Code


Visual Studio Code - Coc
code.visualstudio.com
Visual Studio Code
Docs
Updates
Blog
Extensions
FAQ
Version 1.0 is now available! Read about the new features and fixes in November.
Code editing.
Redefined.
Free. Open source. Runs everywhere.
Download for Windows
Stable Build
Other platforms and Insiders Edition
By using VS Code, you agree to its
license and privacy statement.

## SQL Express



SQL Server 2016 SP1 Express edition

Use SQL Server programming features to build small data-driven applications up to 10 GB of maximum disk size—that will scale across editions as you grow—with this free, entry-level database.

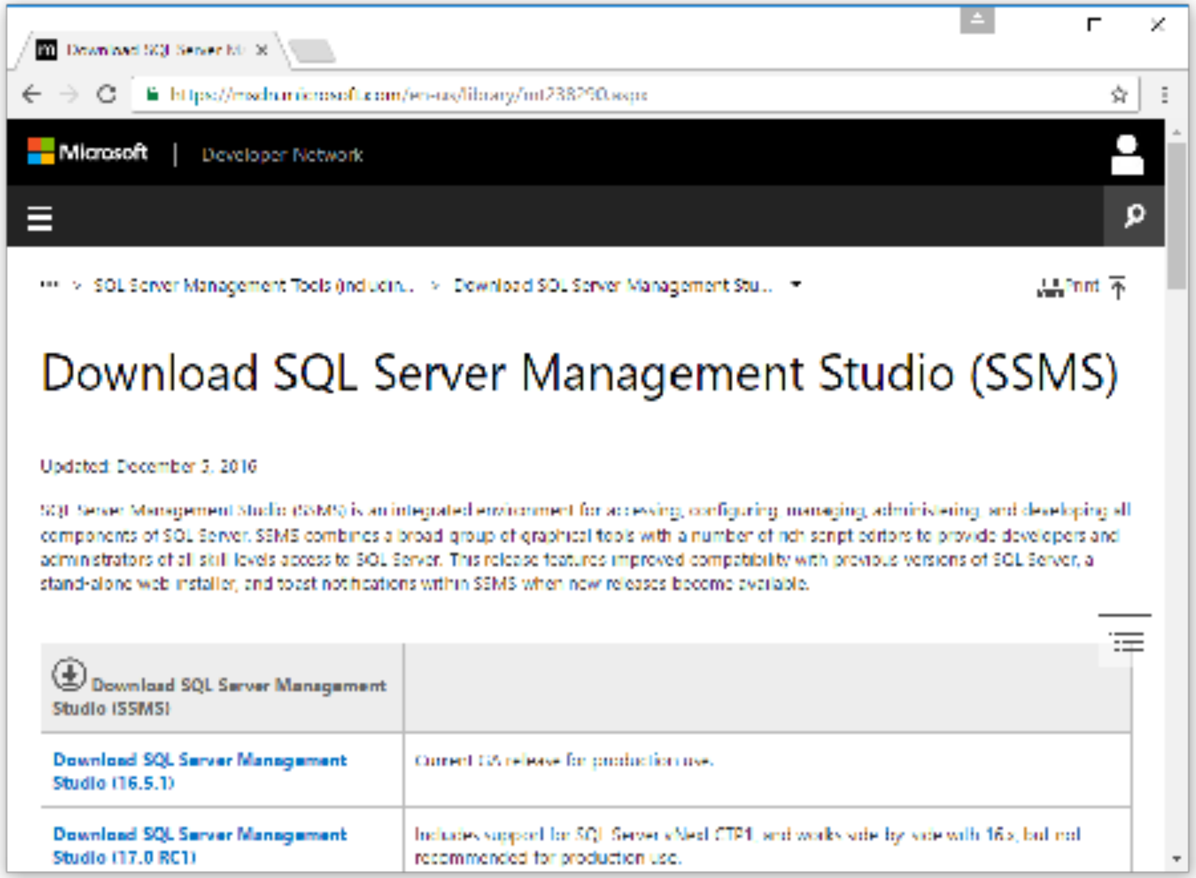
Try SQL Server 2016 Express for free

Read SQL Server 2016 developer resources

Read the SQL Server 2016 licensing datasheet

Why choose the Express edition?

# SQL Express Management Studio


Download SQL Server M
https://msdn.microsoft.com/en-us/library/mt238290.aspx
Microsoft | Developer Network
SQL Server Management Tools (Includin... > Download SQL Server Management Stu...
Print

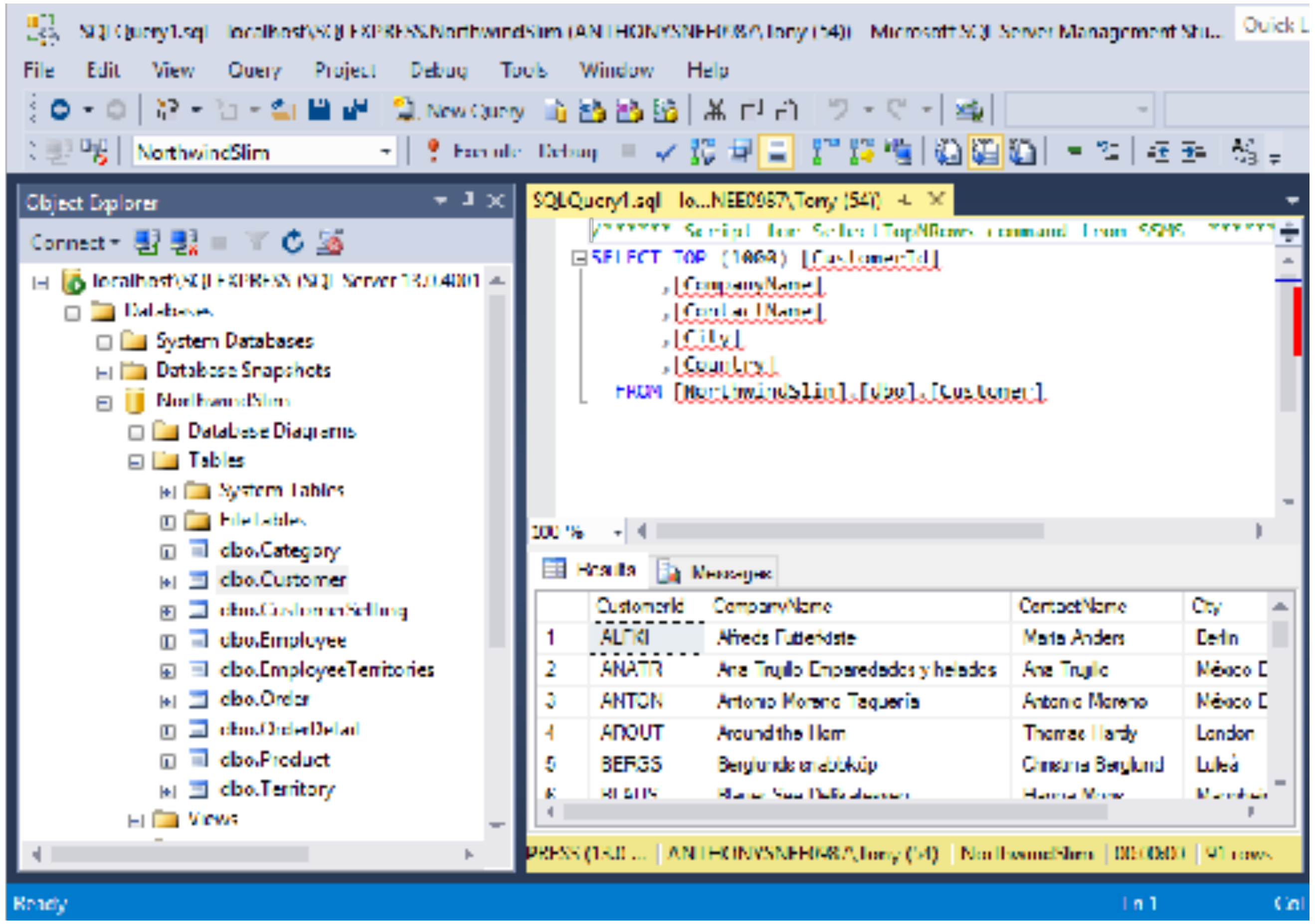
## Download SQL Server Management Studio (SSMS)

Updated: December 5, 2016

SQL Server Management Studio (SSMS) is an integrated environment for accessing, configuring, managing, administering, and developing all components of SQL Server. SSMS combines a broad group of graphical tools with a number of rich script editors to provide developers and administrators of all skill levels access to SQL Server. This release features improved compatibility with previous versions of SQL Server, a stand-alone web installer, and toast notifications within SSMS when new releases become available.

| Download SQL Server Management Studio (SSMS) | |
|---|---|
| Download SQL Server Management Studio (16.5.1) | Current GA release for production use. |
| Download SQL Server Management Studio (17.0 RC1) | Includes support for SQL Server vNext CTP1, and works side by side with 16.x, but not recommended for production use. |

# Northwind Slim Sample Database

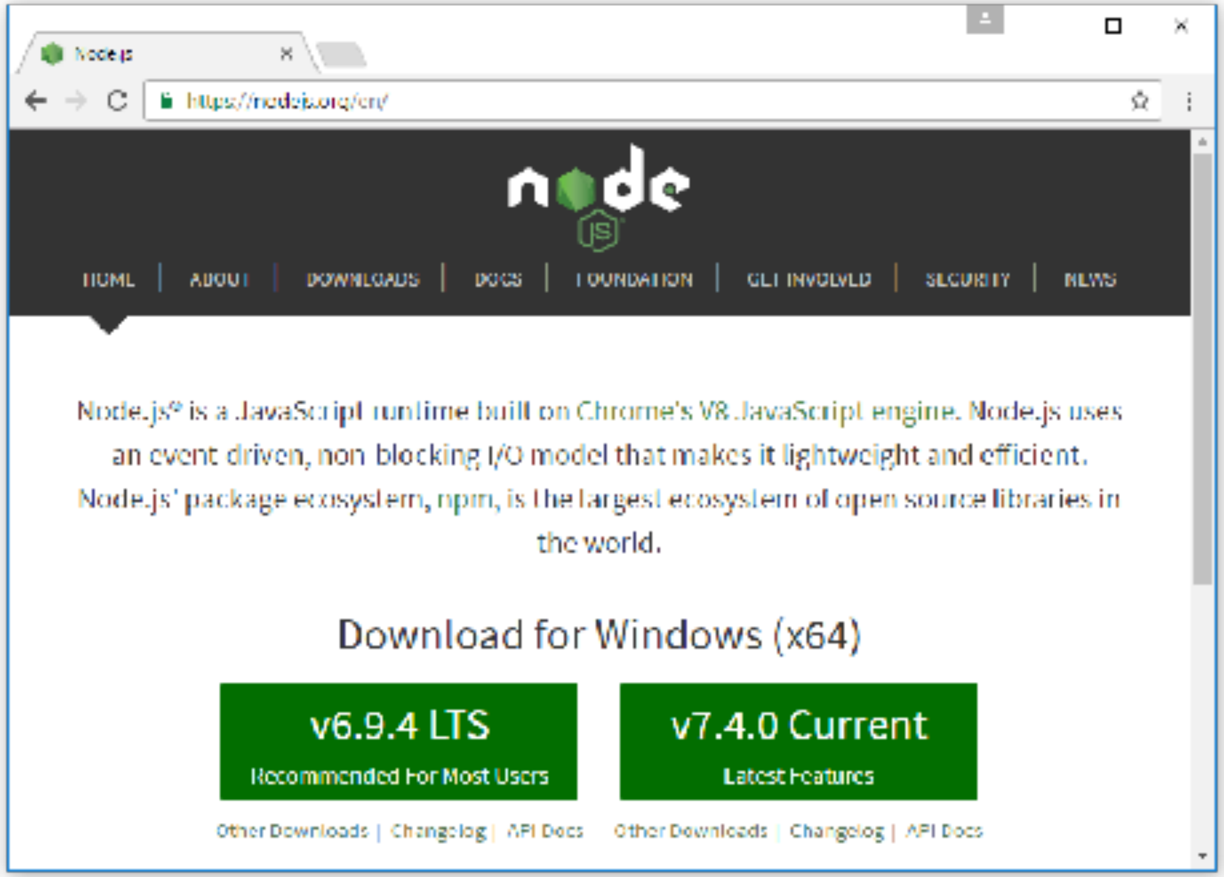


```
/****** Script for SelectTopNRows command from SSMS ******/
SELECT TOP (1000) [CustomerId]
      ,[CompanyName]
      ,[ContactName]
      ,[City]
      ,[Country]
  FROM [NorthwindSlim].[dbo].[Customer]
```

| | CustomerId | CompanyName | ContactName | City |
|---|---|---|---|---|
| 1 | ALFKI | Alfreds Futterkiste | Maria Anders | Berlin |
| 2 | ANATR | Ana Trujillo Emparedados y helados | Ana Trujillo | México D |
| 3 | ANTON | Antonio Moreno Taquería | Antonio Moreno | México D |
| 4 | AROUT | Around the Horn | Thomas Hardy | London |
| 5 | BERGS | Berglunds snabbköp | Christina Berglund | Luleå |

- Download sample db: **bit.ly/northwindslim**
- Create new database: **NorthwindSlim**
- Run script: **northwindslim.sql**

# Node Package Manager


Node.js
https://nodejs.org/en/
HOME | ABOUT | DOWNLOADS | DOCS | FOUNDATION | GET INVOLVED | SECURITY | NEWS

Node.js® is a JavaScript runtime built on Chrome's V8 JavaScript engine. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient. Node.js' package ecosystem, npm, is the largest ecosystem of open source libraries in the world.

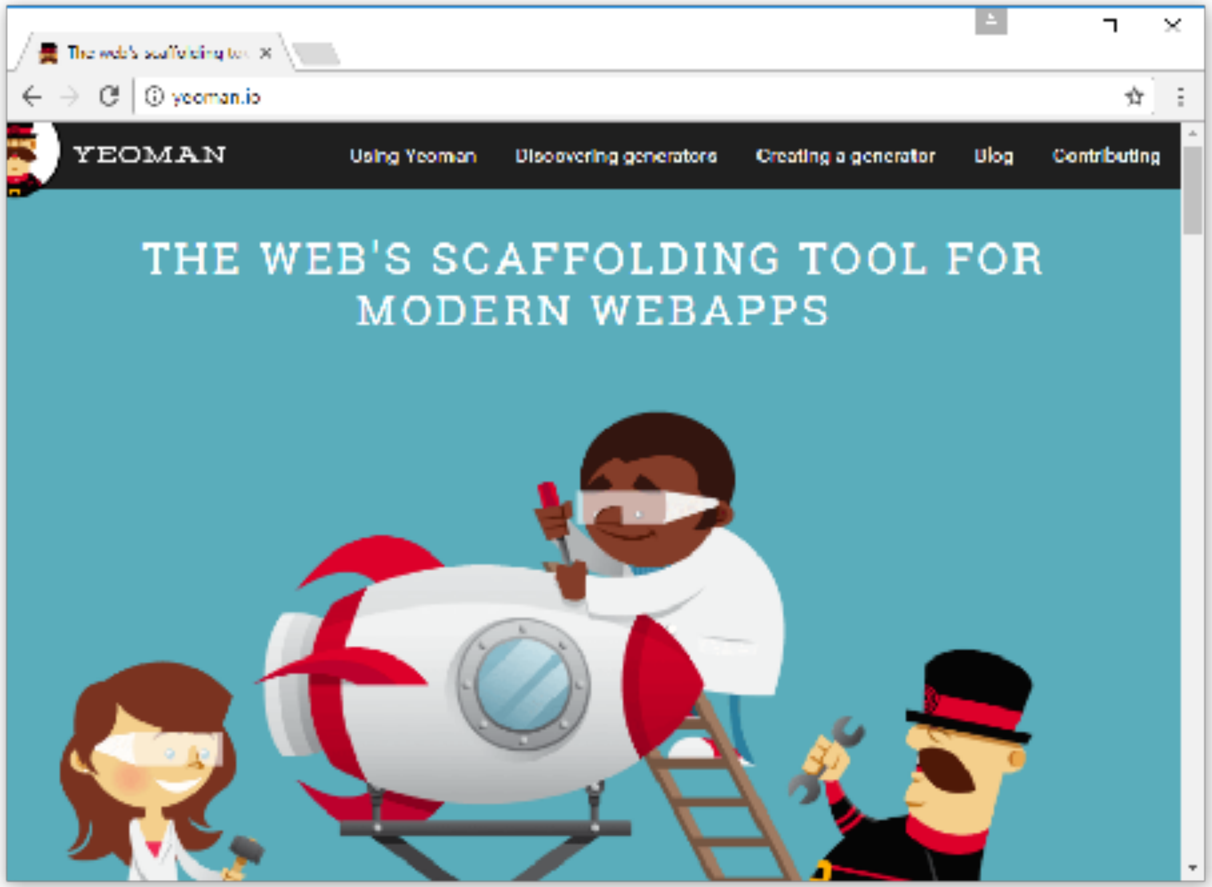
Download for Windows (x64)

v6.9.4 LTS
Recommended For Most Users

v7.4.0 Current
Latest Features

Other Downloads | Changelog | API Docs
Other Downloads | Changelog | API Docs

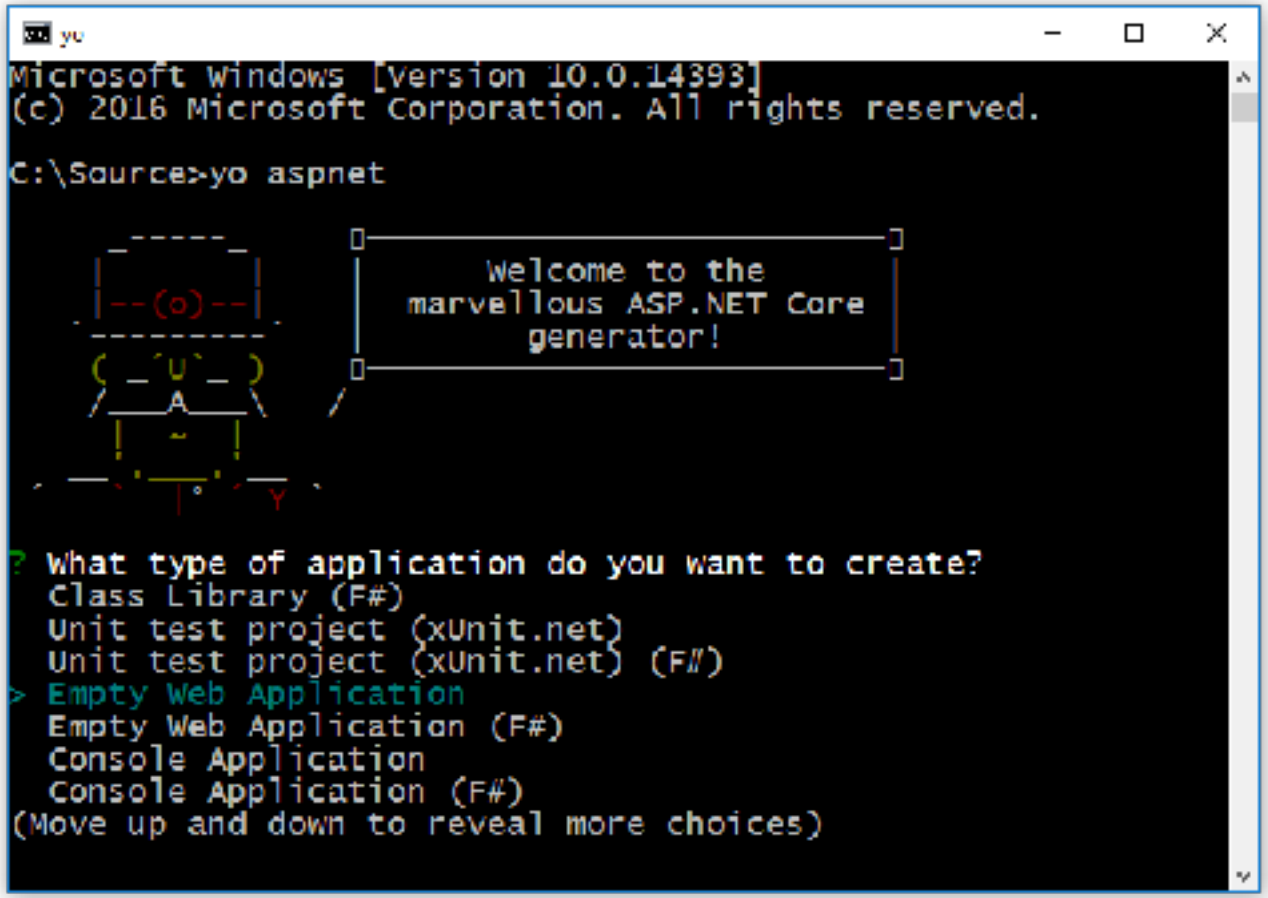
## Yeoman Scaffolding Tool



- npm -i -g yo
- npm -i -g generator-aspnet
- yo aspnet

## Getting Started with ASP.NET Core

# Yeoman AspNet Generator



```
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

C:\Source>yo aspnet

                          Welcome to the
                     marvellous ASP.NET Core
                            generator!

? What type of application do you want to create?
  Class Library (F#)
  Unit test project (xUnit.net)
  Unit test project (xUnit.net) (F#)
> Empty Web Application
  Empty Web Application (F#)
  Console Application
  Console Application (F#)
(Move up and down to reveal more choices)
```

## Project.json

```
{
 "dependencies": {
   "Microsoft.NETCore.App": {
     "version": "1.1.0",
     "type": "platform"
   },
  "Microsoft.AspNetCore.Server.Kestrel": "1.1.0",
   "Microsoft.Extensions.Logging.Console": "1.1.0",
   "Microsoft.Extensions.Configuration.EnvironmentVariables": "1.1.0"
 }
```

Soon to be replaced by **csproj** file

# Project.json - Debugging

```
"buildOptions": {
  "emitEntryPoint": true,
  "preserveCompilationContext": true,
  "debugType": "portable"
},
```

Required for debugging

## Program.cs

```
public class Program
{
    public static void Main(string[] args)
    {
        var host = new WebHostBuilder()
            .UseKestrel()
            .UseStartup<Startup>()
            .Build();
        host.Run();
    }
}
```

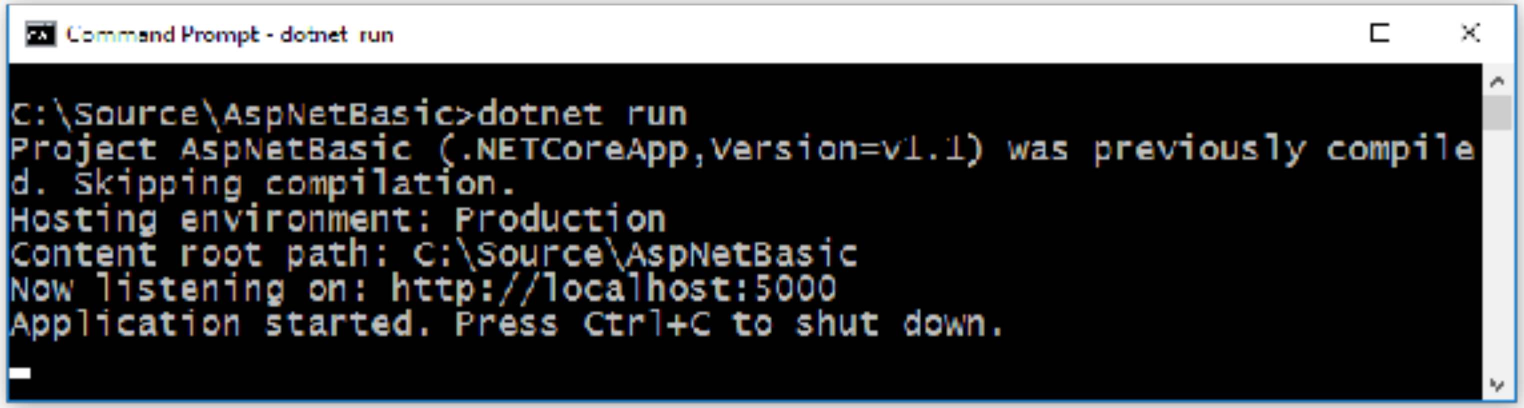
Bootstraps web app

## Startup.cs

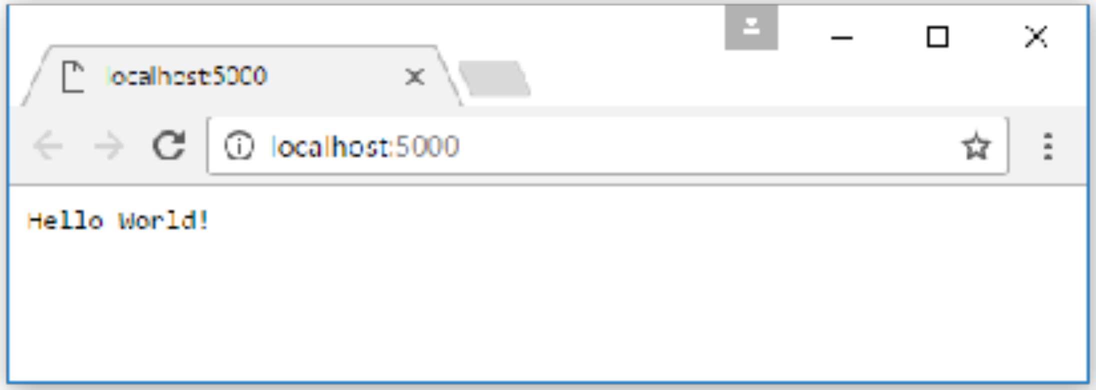
Configures request pipeline

```
public class Startup
{
    public void Configure(IApplicationBuilder app, IHostingEnvironment env)
    {
        app.Run(async (context) =>
        {
            await context.Response.WriteAsync("Hello World!");
        });
    } }
```

# Running the Web App


Command Prompt - dotnet run

```
C:\Source\AspNetBasic>dotnet run
Project AspNetBasic (.NETCoreApp,Version=v1.1) was previously compile
d. Skipping compilation.
Hosting environment: Production
Content root path: C:\Source\AspNetBasic
Now listening on: http://localhost:5000
Application started. Press Ctrl+C to shut down.
```


localhost:5000

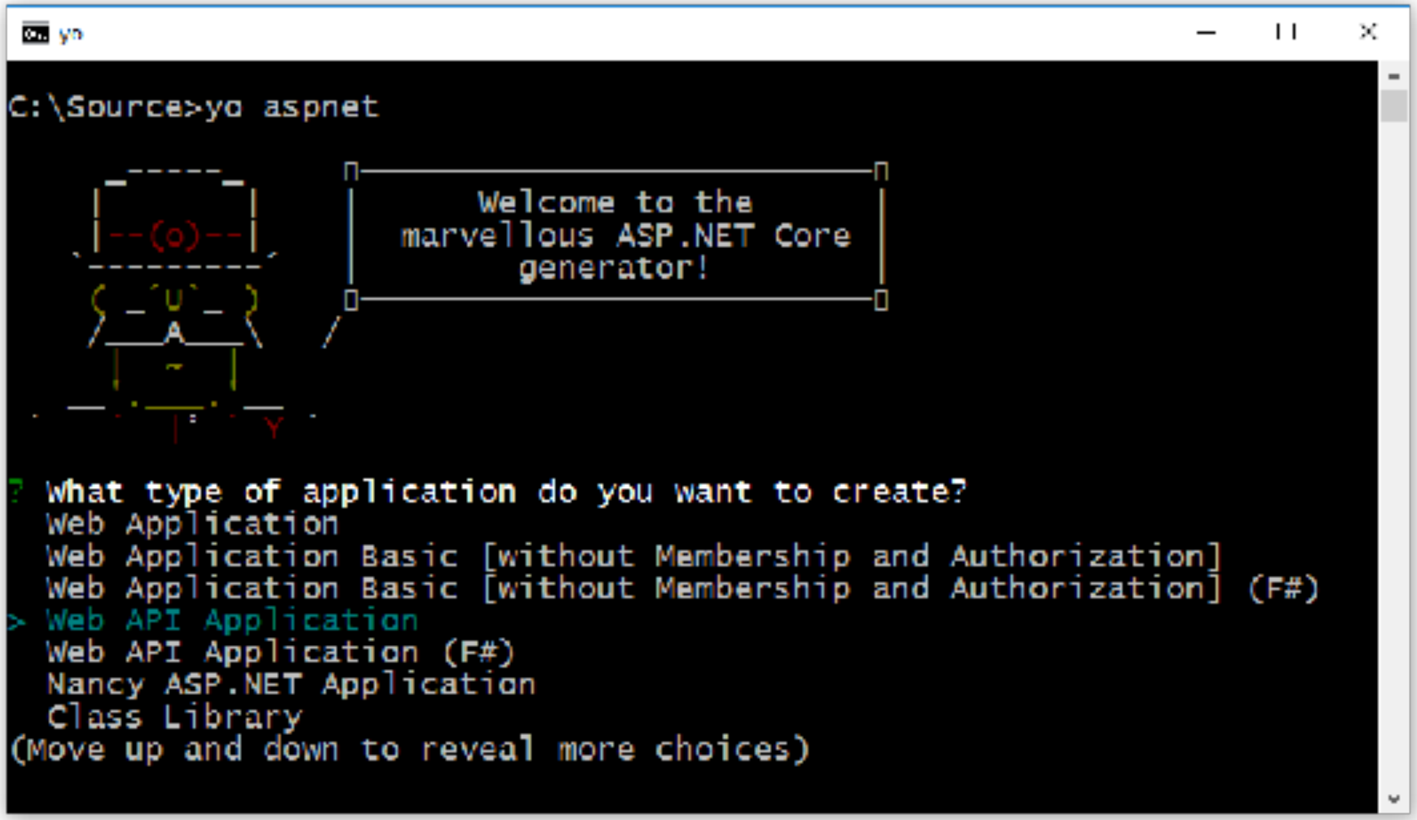
localhost:5000

```
Hello World!
```

## Demo: ASP.NET Core Basic App

## Building a RESTful Web API

# Scaffold Web API Application



```
C:\Source>yo aspnet

                        Welcome to the
                  marvellous ASP.NET Core
                        generator!

? What type of application do you want to create?
  Web Application
  Web Application Basic [without Membership and Authorization]
  Web Application Basic [without Membership and Authorization] (F#)
> Web API Application
  Web API Application (F#)
  Nancy ASP.NET Application
  Class Library
(Move up and down to reveal more choices)
```

## Web API Startup: Services

Configures services

```
public class Startup
{
    public void ConfigureServices(IServiceCollection services)
    {
        // Add MVC to DI container.
        services.AddMvc();
    }
}
```

# Web API Startup: Pipeline

Configures request pipeline

```
public class Startup
{
   public void Configure(IApplicationBuilder app)
    {
        // Configure MVC in request pipeline.
        app.UseMvc();
    }
}
```

## Controller: Get

```
[Route("api/[controller]")]
public class ValuesController : Controller
{
    [HttpGet]  // GET api/values
    public IActionResult Get()
    {
        return Ok(new string[] { "value1", "value2" });
    }
```

Use attributes to control routes

## Controller: Post

```
[HttpPost]  // POST api/values
public IActionResult Post([FromBody]string value)
{
    // TODO: Insert new value
    return CreatedAtAction("Get", new { id = 1 }, newValue);
}
```

FromBody is required

Returns status 201 Created.
Sets response Location header.

## Controller: Put

```
[HttpPut("{id}")]  // PUT api/values/5
public IActionResult Put(int id, [FromBody]string value)
{
    // TODO: Insert new value
    return Ok(newValue);
}
```

## Controller: Delete

```
[HttpDelete("{id}")]  // DELETE api/values/5
public IActionResult Delete(int id)
{
    // TODO: Insert new value
    return NoContent();
}
```

Returns status 204 No Content

## Dependency Injection System

- Unified dependency injection
  - Register services in Startup.ConfigureServices
  - Specify lifetime
    - Singleton, transient, scoped to request
  - Services available throughout entire web stack
    - Middleware, filters, controllers, model binding, etc
    - Can **replace** default DI container

## Dependency Injection: Config Service

```
public class Startup
{
    public void ConfigureServices(IServiceCollection services)
    {
        // Add values store to DI container.
        services.AddTransient<ValuesStore>();
    }
}
```

## Dependency Injection: Use Service

```
public class ValuesController : Controller {
    ValuesStore _store;

    // Inject dependency
    public ValuesController(ValuesStore store) {
        _store = store;
    }
```

## Configuration System

- New configuration framework
  - Intended to *replace* web.config
  - Supports **multiple sources**
    - Json, Xml, Ini files; command-line args; environment variables
  - Supports **complex structures**
    - Beyond simple key-value pairs

## Configuration: Program.Main

```
public static void Main(string[] args)  {
    // Build configuration
    var config = new ConfigurationBuilder()
        .AddCommandLine(args)
        .AddEnvironmentVariables(prefix: "ASPNETCORE_")
        .Build();
    var host = new WebHostBuilder()
        .UseConfiguration(config)
```

# Configuration: Startup

Set config in Startup ctor

```
public class Startup {
    public Startup(IHostingEnvironment env) {
        var builder = new ConfigurationBuilder()
            .AddJsonFile($"appsettings.{env.EnvironmentName}.json")
            .AddEnvironmentVariables();
        Configuration = builder.Build();
    }
public IConfigurationRoot Configuration { get; }
```

## Questions?