11

ASP NET Core

Visual Studio 2017 Web Developer

Objetivos

Al finalizar el capítulo, el alumno logrará:

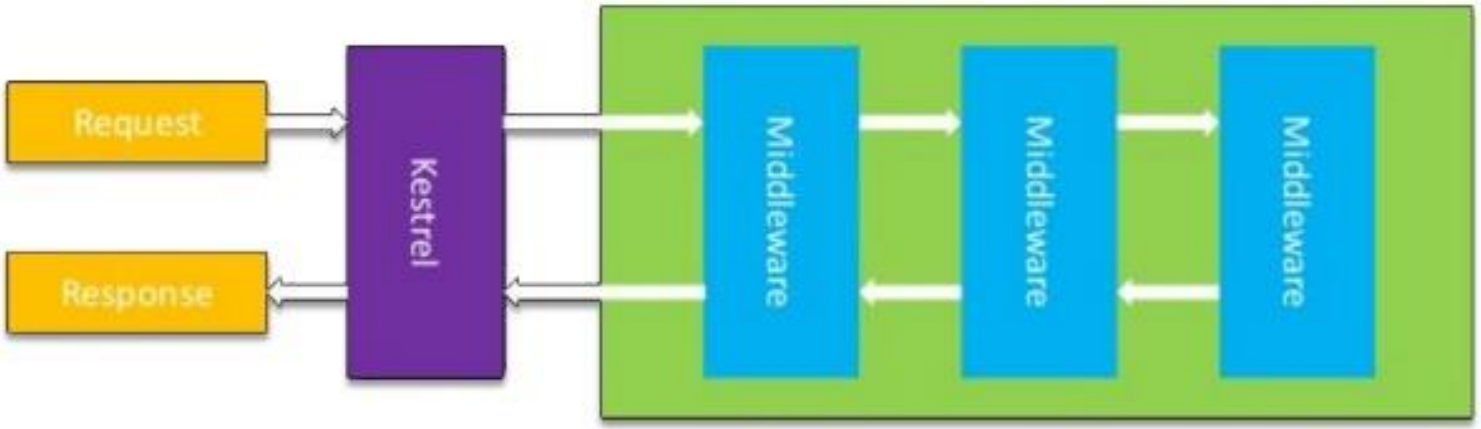
- Crear las aplicaciones con ASP.NET Core 2.0

Agenda

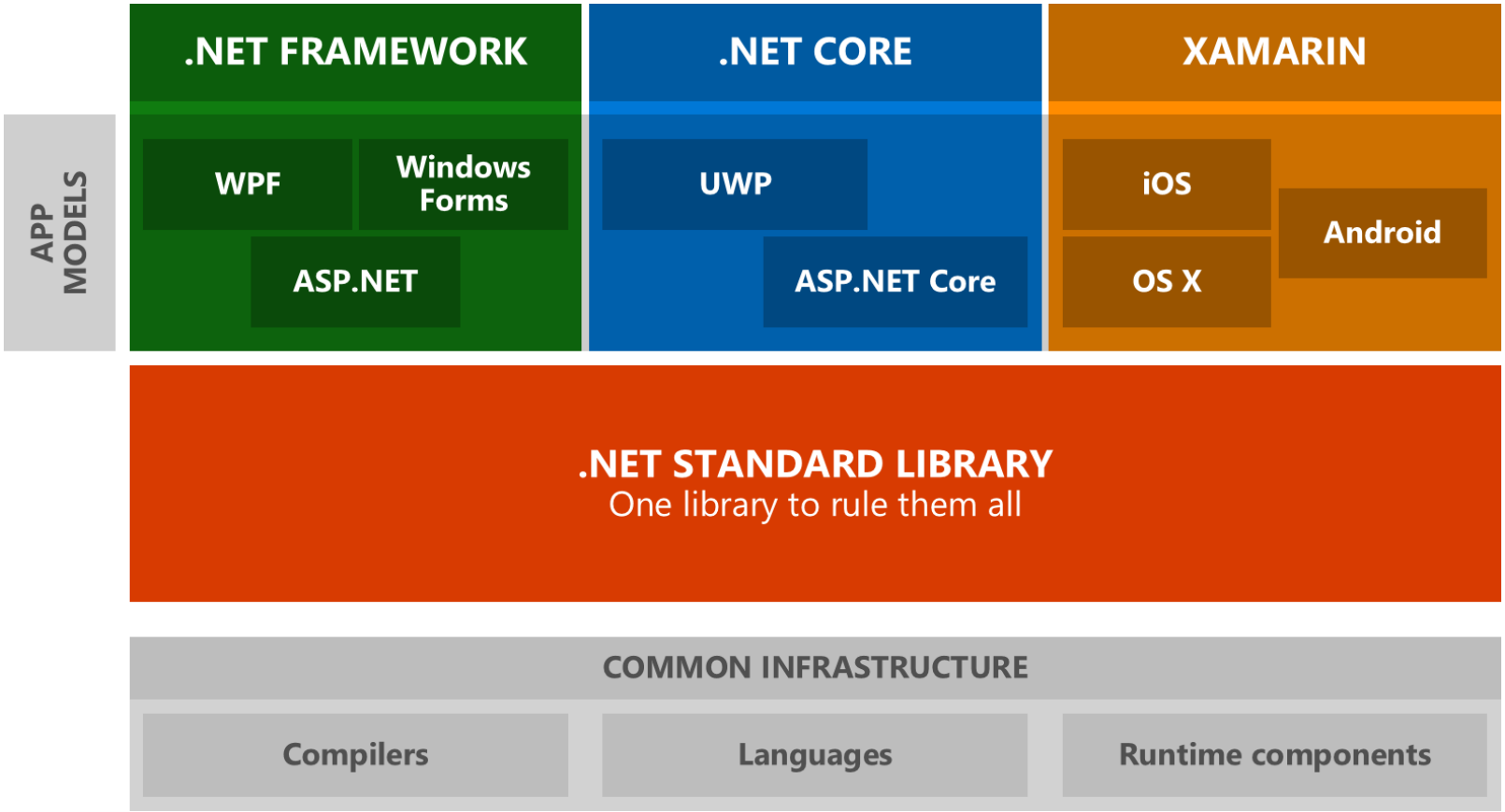
- Introducción a ASP NET Core y Owin.
- Creando una aplicación con ASP NET Core.
- .NET Core vs .NET Standar
- Asp Net Core Mvc
  - Model
  - View
  - Controller
  - Routing
- Asp Net Core Web Api.
  - Routing
- Controller

Introducción a ASP NET Core y Owin

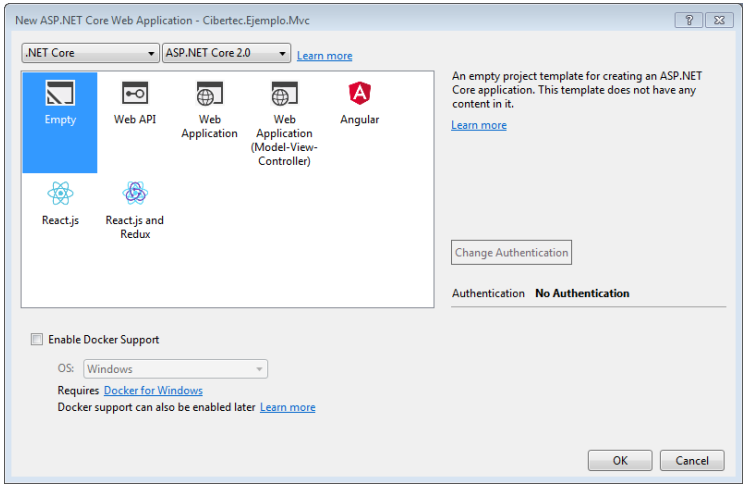
Architecture



Introducción a ASP NET Core

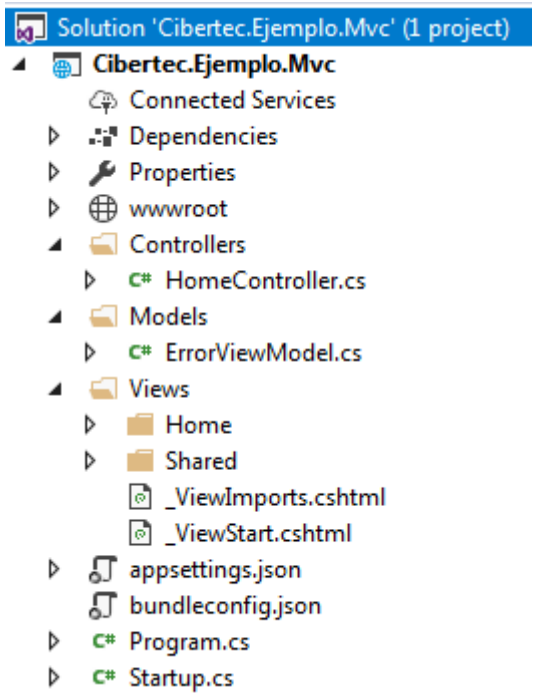


Creando una aplicación con ASP NET Core

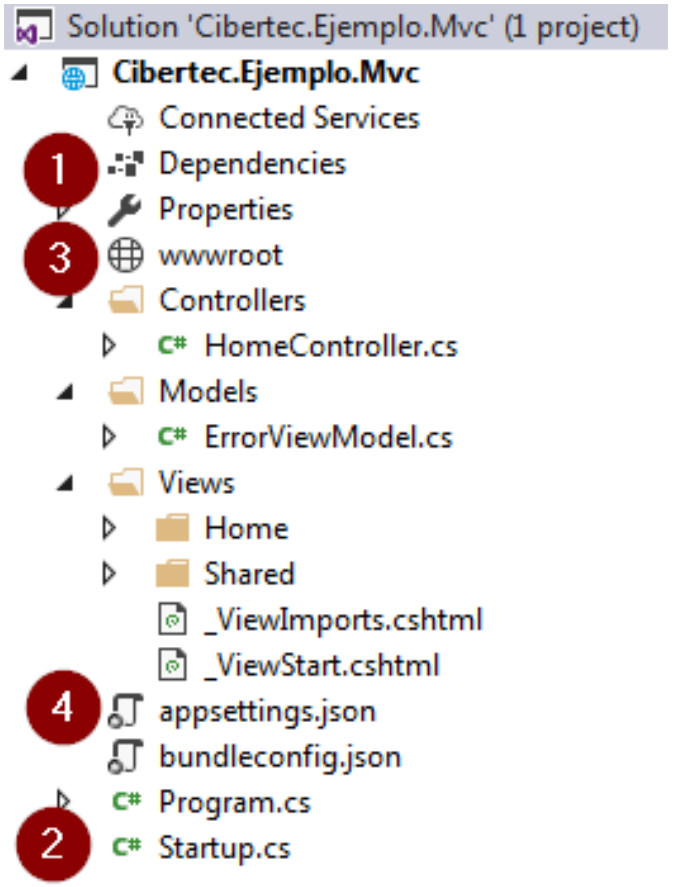


Nuevas plantillas

Nueva estructura de proyecto



Creando una aplicación con ASP NET Core.



1. En lugar de References, tenemos “Dependencies”
2. No existe **Global.asax** y solo tenemos el **Startup.cs**
3. Todo el contenido estático se guarda en “**wwwroot**”
4. En lugar de “**web.config**” tenemos “**appsettings.json**”

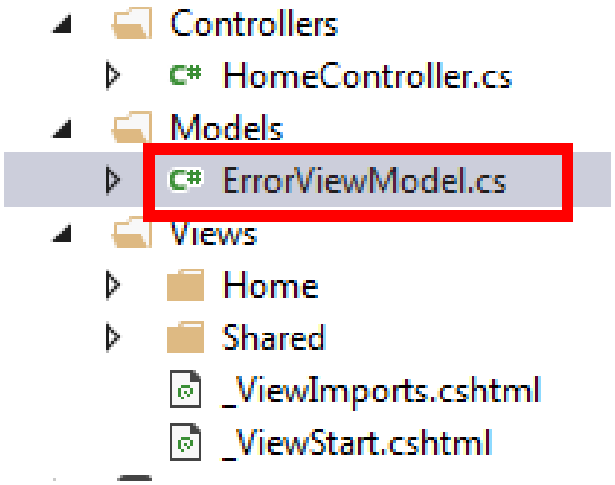
Asp Net Core Mvc (Model)

La misma forma de trabajo de siempre en la creación de modelos.

```
using System;

namespace Cibertec.Ejemplo.Mvc.Models
{
    1 reference
    public class ErrorViewModel
    {
        2 references | 0 exceptions
        public string RequestId { get; set; }

        0 references | 0 exceptions
        public bool ShowRequestId => !string.IsNullOrEmpty(RequestId);
    }
}
```

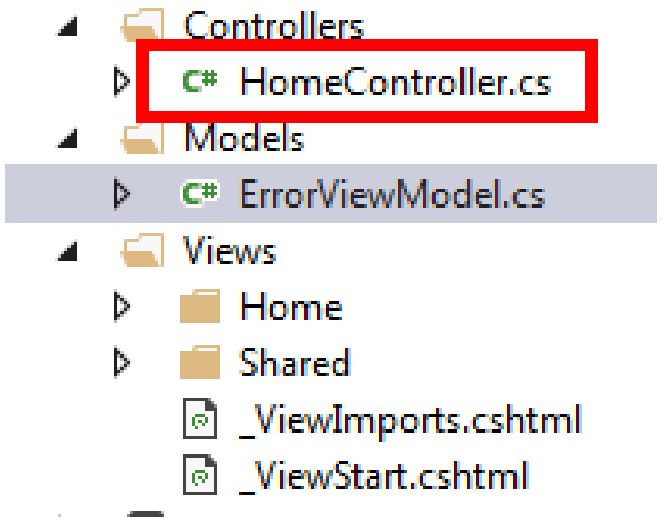


Asp Net Core Mvc (View)

El cambio mas significativo en comparación con Asp Net Mvc, viene a ser en la vista, ya que cambio Razor

```
<div class="navbar-collapse collapse">
    <ul class="nav navbar-nav">
        <li><a asp-area="" asp-controller="Home" asp-action="Index">Home</a></li>
        <li><a asp-area="" asp-controller="Home" asp-action="About">About</a></li>
        <li><a asp-area="" asp-controller="Home" asp-action="Contact">Contact</a></li>
    </ul>
</div>
```

Asp Net Core Mvc (Controller)



En los controladores en lugar de tener un ActionResult (Una clase), tenemos un IActionResult (una Interface)

```
0 references | 0 requests | 0 exceptions
public IActionResult Index()
{
    return View();
}
```

Asp Net Core Mvc (Routing)

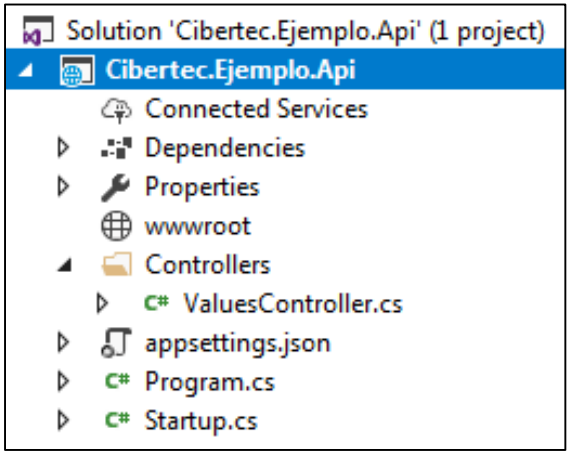
Soporta enrutamiento por configuración

```
app.UseMvc(routes =>
{
    routes.MapRoute(
        name: "default",
        template: "{controller=Home}/{action=Index}/{id?}");
});
```

Y también enrutamiento por notación

```
[Route("Home")]
- references
public class HomeController
{
```

Asp Net Core Web Api



La estructura de la versión Web Api es similar al de una aplicación MVC, la única diferencia es que como porte del response los controladores devuelven el HttpCodeStatus y la data correspondiente.

```
[HttpGet]
[Route("list/{page}/{rows}")]
0 references | Cesar Velarde, 21 hours ago | 1 author, 1 change | 0 requests | 0 exceptions
public IActionResult GetList(int page, int rows)
{
    var startRecord = ((page - 1) * rows) + 1;
    var endRecord = page * rows;
    return Ok(_unit.Customers.PagedList(startRecord, endRecord));
}
```

Ejercicio Nº 11.1: Crear una aplicación Web Api con ASP NET Core

Crear una aplicación Web Api con Asp Net Core.

Al finalizar el laboratorio, el alumno logrará:

- Crear una aplicación Web Api con ASP NET Core.

Ejercicio Nº 11.2: Exponiendo datos con una Web Api de ASP NET Core

Exponer datos con una Web Api hecha con Asp Net Core.

Al finalizar el laboratorio, el alumno logrará:

- Exponer datos haciendo uso de una Web Api hecha con Asp Net Core.

Ejercicio Nº 11.3: Implementando autenticación por Token con ASP NET Core

Implementar autenticación por Token.

Al finalizar el laboratorio, el alumno logrará:

- Implementar autenticación por Token en una Web Api hecha con Asp Net Core.

Ejercicio Nº 11.4: Agregando soporte a CORS

Implementar soporte CORS a nuestra Web Api.

Al finalizar el laboratorio, el alumno logrará:

- Implementar soporte CORS en una Web Api hecha con Asp Net Core.

Ejercicio Nº 11.5: Publicación de una Web Api de ASP NET Core en IIS y acceder desde nuestra aplicación React

Publicar una aplicación Web Api hecha con Asp Net Core.

Al finalizar el laboratorio, el alumno logrará:

- Publicar y consumir una Web Api de ASP NET Core.

Tarea Nº 11: Investiga al respecto de las nuevas características entre Asp .Net Core y Asp .Net Mvc

Conocer cuales son la diferencias entre Asp .Net Core y Asp .Net Mvc.