División de Alta Tecnología

9

ASP.NET Web API

Visual Studio 2017 Web Developer

Objetivos

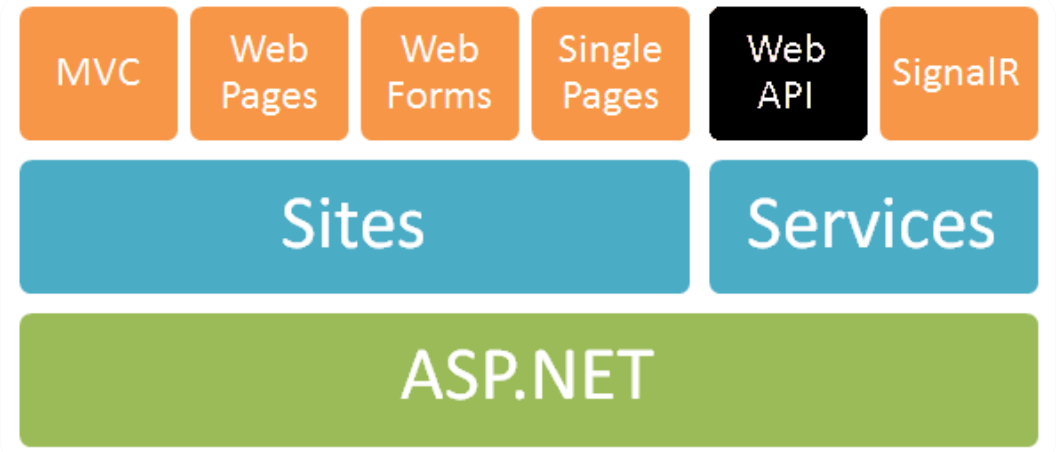
- Comprender el uso de Web Api.
- Implementar aplicaciones web haciendo de uso de Web API.
- Implementar seguridad en aplicaciones web que hacen uso de Web API.
- Realizar el despliegue y hosting de aplicaciones web que usan Web API.

Agenda

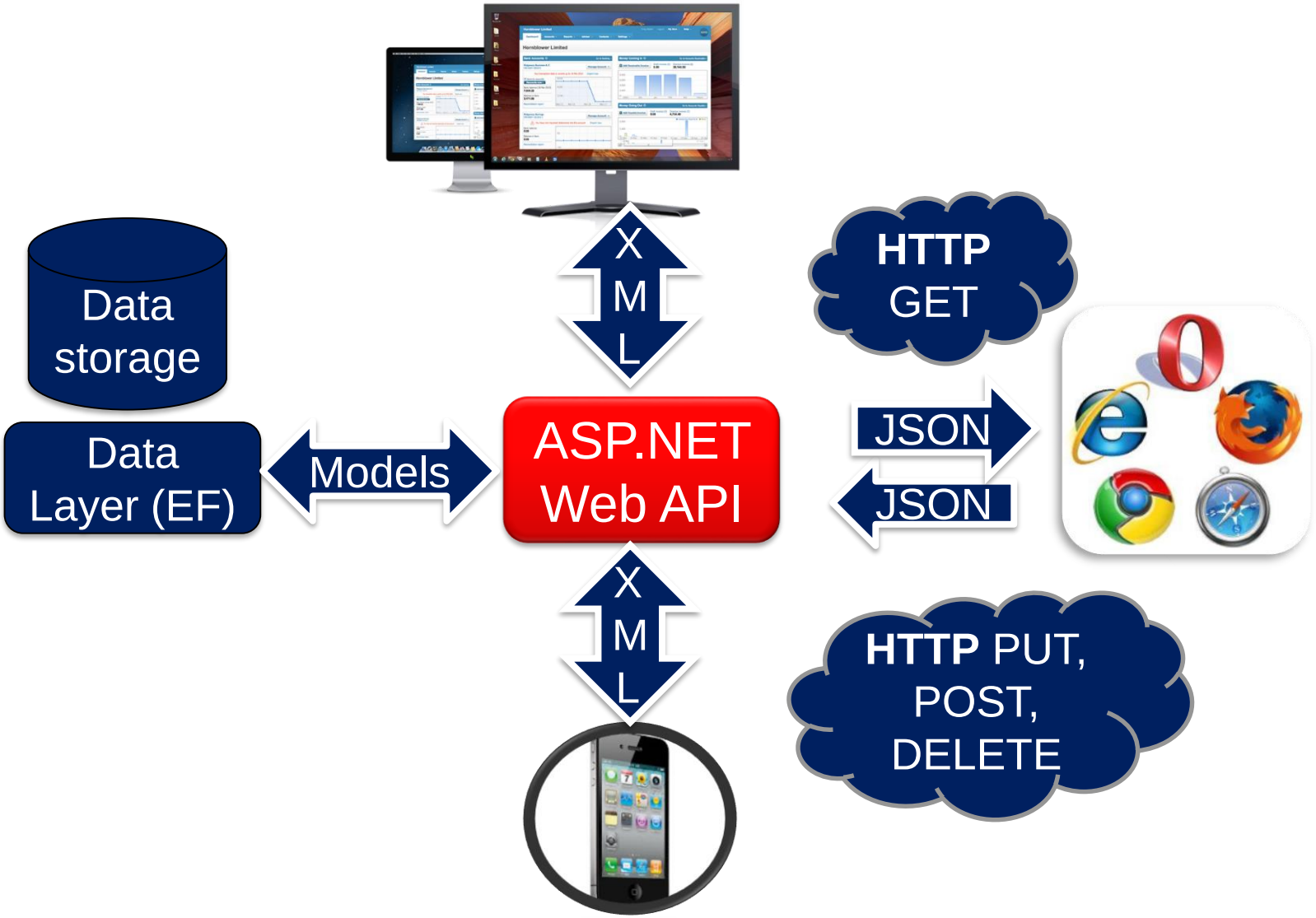
- Introducción de Web Apis.
- Iniciando con la creación de un proyecto.
- Características de una Web Api
- Seguridad en una Web Api.
- Consumiendo una Web Api
- Optimización de una Web Api

Introducción a ASP.NET Web Api

- ASP.NET Web API == plataforma para construer servicios web RESTful
  - Corre sobre el .NET Framework
  - Basada en el funcionamiento de ASP NET



Introducción a ASP.NET Web Api



Caracteristicas

- Facilidad de uso con gran poder.
- Modelo de desarrollo moderno de HTTP.
  - Acceso objetos de HTTP fuertemente tipados.
- Negociación de Contenido
- Model binding y validaciones.
- Enrutamiento con todas las capacidades soportadas por ASP NET MVC.
- Filtros
- Testeabilidad.
- IoC.
- Hosting Flexible (IIS, Azure, self-hosting)
- Soporte e integración con OWIN.

Web API Controllers

- Un controller administra las solicitudes HTTP.
  - Los controladores derivan de la clase **`ApiController`**
  - La Clase ApiController no existe para NetCore
  - **ASP.NET Web API** por defecto asigna las solicitudes **HTTP** a métodos específicos llamados "actions"

Action	HTTP method	Relative URI	Method
Lista de posts	GET	/api/posts	Get()
Post by Id	GET	/api/posts/id	Get(int id)
Crear nuevo Post	POST	/api/posts	Post(PostModel value)
Actualizar un Post	PUT	/api/posts/id	Put(int id, PostModel value)
Eliminar un Post	DELETE	/api/posts/id	Delete(int id)
Post por Categorias	GET	/api/posts?category=news	Get(string category)

Proceso de una solicitud

1. Solicitud web iniciada

```
http://localhost:1337/api/posts
```

2. Encontrando controlador.

```
GET /api/posts HTTP/1.1
Host: localhost:1337
Cache-Control: no-cache
```

3. Controlador responde.

```
HTTP/1.1 200 OK
Content-Length: 11
"some data"
```

```
public class PostsController :
ApiController
{
    public string Get()
    {
        return "Some data";
    }

    public string Edit(Post post)
    {
        ...
    }
}

public class UsersController :
ApiController
{
    ...
}
```

Return Types (2)

- Los Actions devuelven muchos tipos.
- Automaticamente retorna XML o JSON.
  - **T** – Tipo Generico(Puede ser cualquier tipo)

```
public Comment GetCommentById(int id) { ... }
```

  - **IEnumerable<T>** - foreach-able de un tipo generico.

```
public IEnumerable<Comment> GetPostComments(int id) { ...
}
```

Return Types (2)

- void – retorna un HTTP response 204 (No Content)
- IHttpActionResult – retornan un HTTP abstracto con un status code y la data resultante

```
public IHttpActionResult GetPostComments(int id)
{
    var context = new ForumContext();
    var post = context.Posts.FirstOrDefault(p => p.Id == id);
    if (post == null)
        return this.BadRequest("Invalid post id");

    return this.Ok(post);
}
```

200 OK + serialized data

HTTP Status Codes

- Una buena practica siempre es retornar el status code.
  - Returnar data con un estatus especifico (ejemplo: **`Ok()`**, **`BadRequest()`**, **`NotFound()`**, **`Unauthorized()`**, etc.)

```
var top10Users = context.Users.All()
     .Take(10)
     .Select(u => u.Username);

return this.Ok(top10Users);
```

  - Retornar solo status code

```
return this.StatusCode(HttpStatusCode.Forbidden);
```

Data Source Attributes

- Web API puede especificar de donde vienen los parametros.
  - **[FromUri]** – binds parametros desde la URL

```
http://localhost:1337/api/posts/comments?page=5
```

```
public IHttpActionResult GetComments([FromUri]int page)
{...}
```

  - **[FromBody]** – binds data desde el cuerpo del request.

```
public IHttpActionResult Register(
    [FromBody]RegisterBindingModel user)
{ ... }
```

Lecturas adicionales

Para obtener información adicional, puede consultar los siguientes enlaces:

- Building Your First Web API with MVC 6:
  - https://docs.asp.net/en/latest/tutorials/first-web-api.html

- Web API / WCF / WCF REST / Web Service
  - http://www.dotnet-tricks.com/Tutorial/webapi/JI2X050413-Difference-between-WCF- and-Web-API-and-WCF-REST-and-Web-Service.html

- Status Codes:
  - https://www.w3.org/Protocols/rfc2616/rfc2616-sec10.html
  - https://msdn.microsoft.com/en-us/library/system.net.httpstatuscode%28v=vs.110%29.aspx

Ejercicio Nº 9.1: Configuración de una Web Api con OWIN

Comprende el flujo de OWIN y crea una Web Api desde una aplicación en blanco.

Al finalizar el laboratorio, el alumno logrará:

- Comprender el uso de OWIN y crear una Web Api sin el uso de plantillas.

Ejercicio Nº 9.2: Configurando Inyección de Dependencias

Implementar inyección de dependencias con SimpleInjector.

Al finalizar el laboratorio, el alumno logrará:

- Implementar el inyector de dependencias para facilitar el uso de nuestro patron UnitOfWork.

Ejercicio Nº 9.3: Creando Controladores de una Web Api

Crear controladores con los Actions básicos para administrar una tabla de nuestra base de datos.

Al finalizar el laboratorio, el alumno logrará:

- Implementar controladores con los Actions básicos para administrar una tabla de nuestra base de datos.

Ejercicio Nº 9.4: Autenticación por Bearer Token

Implementa Autenticación por Bearer Token.

Al finalizar el laboratorio, el alumno logrará:

- Configurar autenticación por Bearer Token en una Web Api.

Ejercicio Nº 9.5: Configurando Logs con Log4Net.

Configura herramientas de logging para administrar los errores o mensajes informativos en caso de ser requrerido.

Al finalizar el laboratorio, el alumno logrará:

- Configurar e implementar logging con log4net.

Ejercicio Nº 9.6: Optimizando Web Api con compresión de respuestas

Optimizar el response de una web api con compresión de datos.

Al finalizar el laboratorio, el alumno logrará:

- Implementar, configurar y optimizar el response de una web api.

Resumen

En este capítulo, usted aprendió a:

- Implementar aplicaciones web haciendo de uso de Web API.
- Implementar seguridad en aplicaciones web que hacen uso de Web API.
- Realizar el despliegue de aplicaciones web que usan Web API.

Tarea Nº 9: Implementa seguridad y logs para el módulo Producto.

Implementar el Web API para registrar la venta de productos

Al finalizar el laboratorio, el alumno logrará:

- Aplicar los conceptos sobre Web API