#GlobalAzure

#GABMUGPeru

#GABMUGPeru

Net 7 minimal api integrando con Angular 15

Anthony Jesus Portilla Cano –MCT

Acerca del Speaker

Ingeniero de sistemas. con 11 años de experiencia profesional.

Actualmente se desempeña como Senior .NET developer anteriormente Senior fullstack .Net para cliente de Costa Rica y EEUU.
Es Microsoft Certified Trainer desde el 2019 además de contar con otras certificaciones Microsoft relacionados a desarrollo y arquitectura.

Linkedin:
https://www.linkedin.com/in/anthonyjesusportillacano/

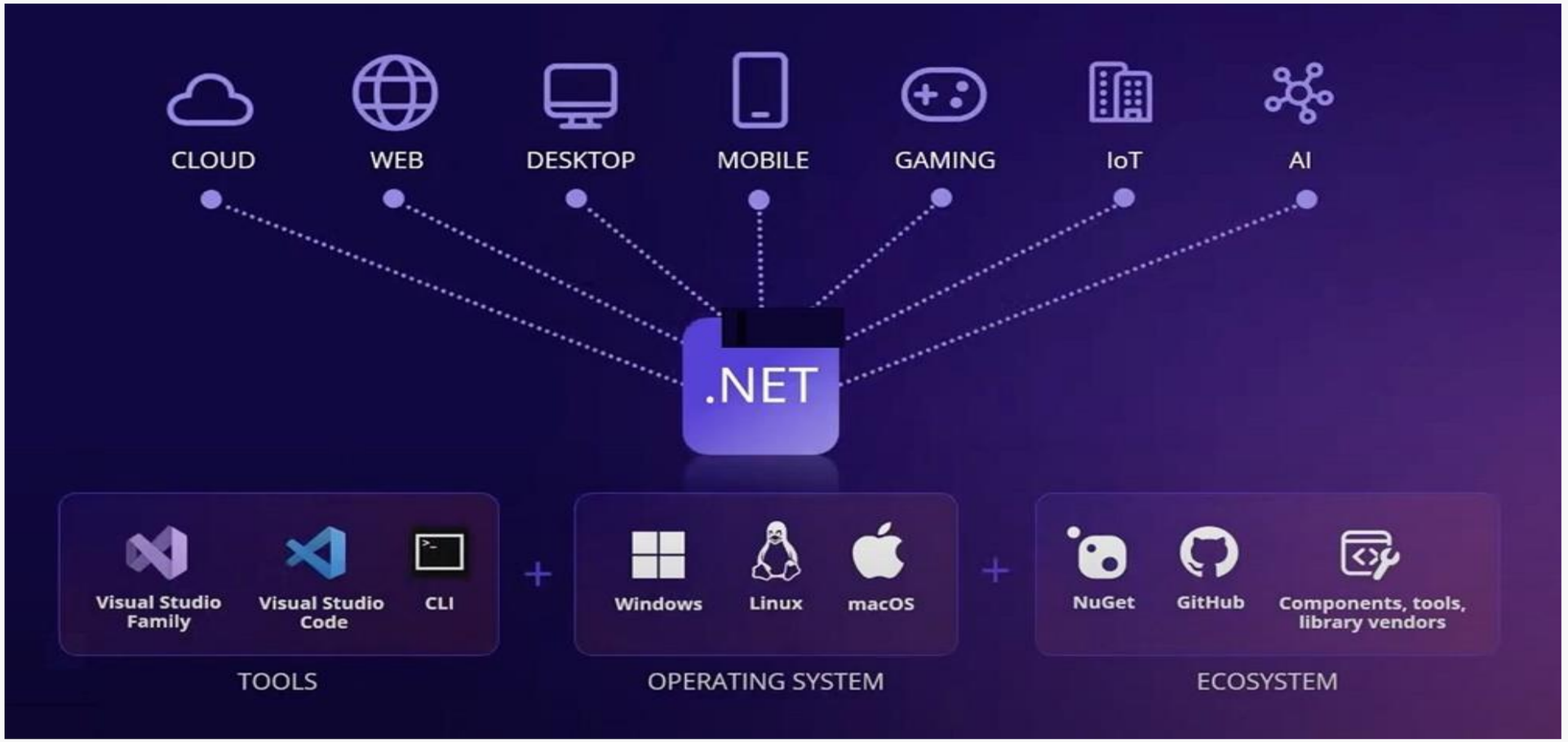
Canal de Youtube: https://www.youtube.com/channel/UC0M-u5Ph4a8WYyoMRym3rUw?view_as=subscriber

Agenda

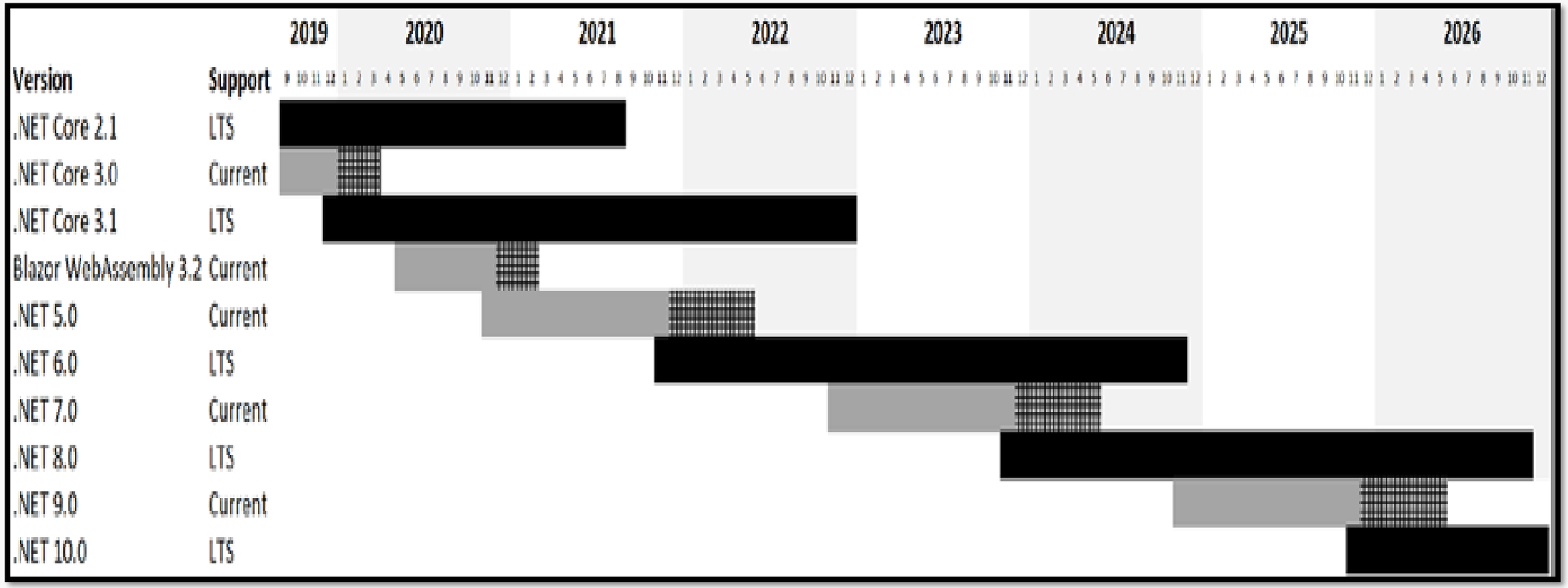
1. **¿Qué es Net 7?**
2. **Minimal APIs vs Controller APIs(Classic)**
3. **Application/WebApplicationBuilder y Routing**
4. **Integración de Routes, Verbs(MapPost, MapPut y MapDelete) y HTTP Status Codes**
5. **¿Que es Angular?.**
6. **Servicios de Azure.**

¿Qué es Net 7?

¿Qué es Net 7?


CLOUD
WEB
DESKTOP
MOBILE
GAMING
IoT
AI
.NET
Visual Studio Family
Visual Studio Code
CLI
Windows
Linux
macOS
NuGet
GitHub
Components, tools, library vendors
TOOLS
OPERATING SYSTEM
ECOSYSTEM

¿Qué es Net 7?


2019
2020
2021
2022
2023
2024
2025
2026
Version
Support
.NET Core 2.1
LTS
.NET Core 3.0
Current
.NET Core 3.1
LTS
Blazor WebAssembly 3.2
Current
.NET 5.0
Current
.NET 6.0
LTS
.NET 7.0
Current
.NET 8.0
LTS
.NET 9.0
Current
.NET 10.0
LTS

#GABMUGPeru

Minimal APIs vs Controller APIs(Classic)

Minimal APIs vs Controller APIs(Classic)

Es una nueva plantilla que utiliza todas las características desde C# 10 y . NET 6 para crear servicios API con mucha menos complejidad, capas, clases que veníamos utilizando con la tradicional plantilla MVC con controladores. Está inspirada en otras tecnologías como node. js.

```
var builder = WebApplication.CreateBuilder(args);
var app = builder.Build();
app.MapGet("/", () => "Hello World");
app.Run();
```

Minimal APIs vs Controller APIs(Classic)

¿Cuál es la diferencia entre este enfoque basado en controlador y la API mínima?

No Startup.cs: cuando crea una API mínima, no necesita un archivo Startup.cs. En su lugar, todas las tareas a las que está acostumbrado se realizan en Program.cs.

Las tareas que solía hacer incluyen configurar rutas y configurar inyecciones de dependencia, seguridad y CORS.

#GABMUGPeru

Application/WebApplicationBuilder y Routing

Application/WebApplicationBuilder y Routing

WebApplication tiene como objetivo facilitar la adición de middleware y puntos finales de forma sencilla.

Por ejemplo, podemos crear una API mínima usando solo el siguiente código

```
WebApplicationBuilder builder = WebApplication.CreateBuilder(args);
WebApplication app = builder.Build();


app.UseHttpsRedirection();
app.UseStaticFiles();
app.MapGet("/", () => "Hello World!");
app.Run();
```

#GABMUGPeru

Integración de Routes, Verbs(MapPost, MapPut y MapDelete)y HTTP Status Codes

Integración de Routes, Verbs(MapPost, MapPut y MapDelete)y HTTP Status Codes

Routing

Lo primero que puede notar es que el patrón para mapear llamadas API se parece mucho a la coincidencia de patrones de los controladores MVC. Esto significa que las API mínimas se parecen mucho a los métodos de controlador. Por ejemplo:

```
app.MapGet("/api/clients", () => new Client()
{
    Id = 1,
    Name = "Client 1"
});
app.MapGet("/api/clients/{id:int}", (int id) => new Client()
{
    Id = id,
    Name = "Client " + id
});
```

Integración de Routes, Verbs(MapPost, MapPut y MapDelete)y HTTP Status Codes

Verbs

Hasta ahora, todo lo que he visto son las API HTTP GET. Hay métodos para los diferentes tipos de verbos. Éstos incluyen:

- MapPost
- MapPut
- MapDelete

Estos métodos funcionan de manera idéntica al método MapGet. Por ejemplo, tome esta llamada para publicar un nuevo cliente

```
app.MapPost("/clients", async (Client model, IJurisRepository repo) =>
{
    // ...
});
```

Integración de Routes, Verbs(MapPost, MapPut y MapDelete) y HTTP Status Codes

Verbs

```
app.MapPut("/clients/{id}", async (int id, ClientModel model,
    IJurisRepository repo) =>
{
    // ...
});
```

```
app.MapDelete("/todoitems/{id}", async (int id, TodoDb db) =>
{
    if (await db.Todos.FindAsync(id) is Todo todo)
    {
        db.Todos.Remove(todo);
        await db.SaveChangesAsync();
        return Results.Ok(todo);
    }

    return Results.NotFound();
});
```

Integración de Routes, Verbs(MapPost, MapPut y MapDelete) y HTTP Status Codes

Using HTTP Status Codes

Results manejan admiten la mayoría de los códigos de estado que necesitará, como

Results.Ok: 200

Results.Created: 201

Results.BadRequest: 400

Results.Unauthorized: 401

Results.Forbid: 403

Results.NotFound: 404

Etc.

```
app.MapGet("/clients/{id:int}", async (int id, IJurisRepository repo) => {
    try {
        var client = await repo.GetClientAsync(id);
        if (client == null)
        {
            return Results.NotFound();
        }
        return Results.Ok(client);
    }
    catch (Exception ex)
    {
        return Results.BadRequest("Failed");
    }
});
```

Route groups in minimal APIs

Con ASP.NET Core 7, puede aprovechar el nuevo método de extensión MapGroup para organizar grupos de puntos finales que comparten un prefijo común en sus API mínimas.
El método de extensión MapGroup no solo reduce el código repetitivo, sino que también facilita la personalización de grupos completos de puntos finales.
El siguiente fragmento de código muestra cómo se puede usar MapGroup

```
app.MapGroup("/public/authors")
    .MapAuthorsApi()
    .WithTags("Public");
```

El siguiente fragmento de código ilustra el método de extensión MapAuthorsApi.

```
public static class MyRouteBuilder
{
public static RouteGroupBuilder MapAuthorsApi(this RouteGroupBuilder group)
    {
        group.MapGet("/", GetAllAuthors);
        group.MapGet("/{id}", GetAuthor);
        group.MapPost("/", CreateAuthor);
        group.MapPut("/{id}", UpdateAuthor);
        group.MapDelete("/{id}", DeleteAuthor);
        return group;
    }
}
```

¿Que es Angular?

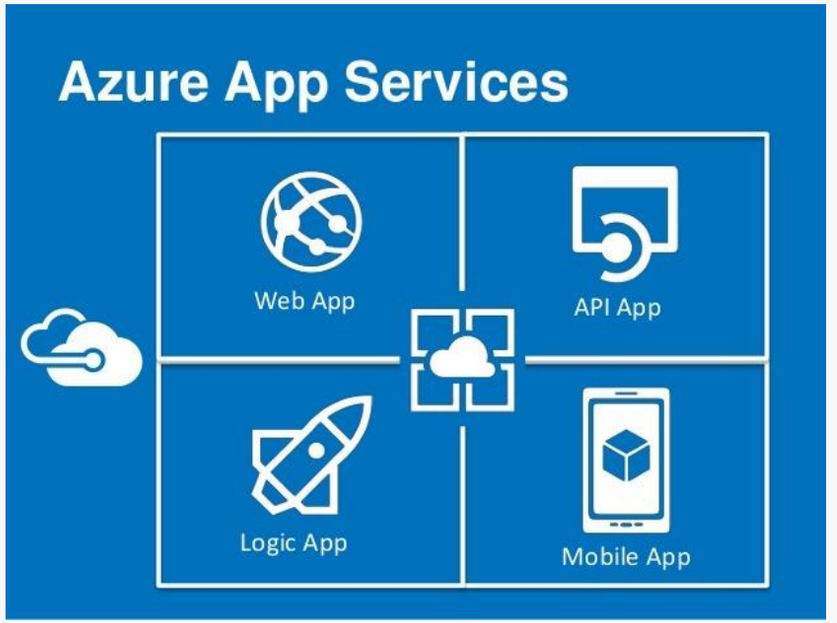
¿Que es Angular?

Angular (comúnmente llamado Angular 2+ o Angular 2) es un framework para aplicaciones web desarrollado en TypeScript, de código abierto, mantenido por Google, que se utiliza para crear y mantener aplicaciones web de una sola página.

#GABMUGPeru

Servicios de Azure

Servicios de Azure



Servicios de Azure

Migrate your PostgreSQL database using export and import

DEMO

Patrocinadores

Microsoft

Colabora

25 años mibanco

baufest

#GLOBALAZURE


• BY COMMUNITY • FOR COMMUNITY •
#GLOBALAZURE
2023