# CREAR API MÍNIMA EN .NET 6

## INICIACIÓN Y EJEMPLOS

OpenWebinars

# ÍNDICE

# ÍNDICE

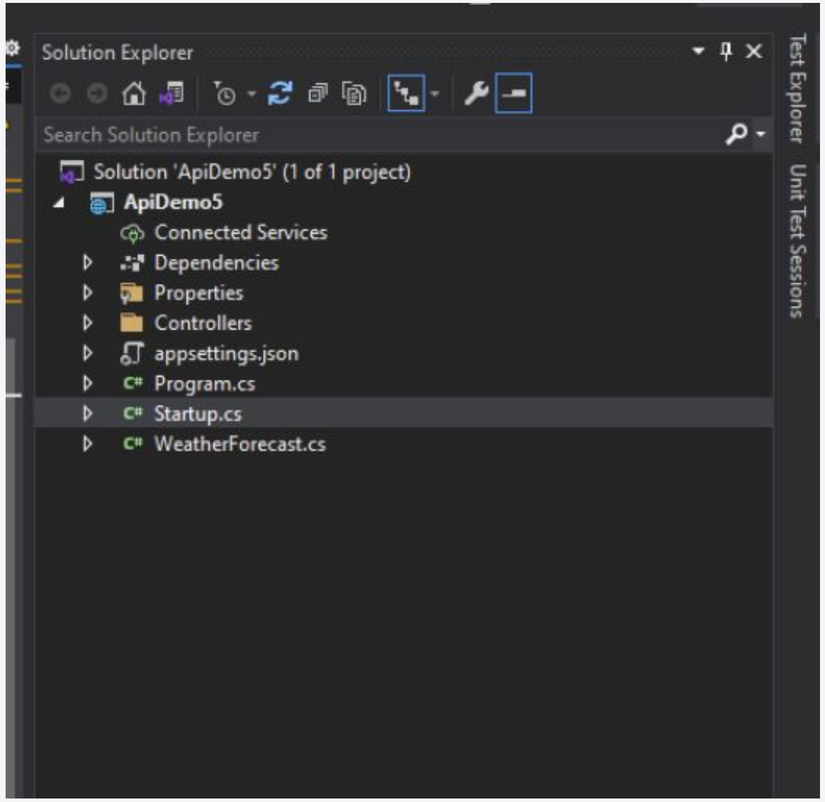
# ¿QUÉ ES UNA API MÍNIMA?

- [Definición de API tradicional] minimizando sus dependencias y su código, reduciéndolo a lo imprescindible.
- Están diseñadas para crear API HTTP con dependencias mínimas.
- Son ideales para microservicios y aplicaciones que desean incluir solo los archivos, las características y las dependencias mínimas en ASP.NET Core.

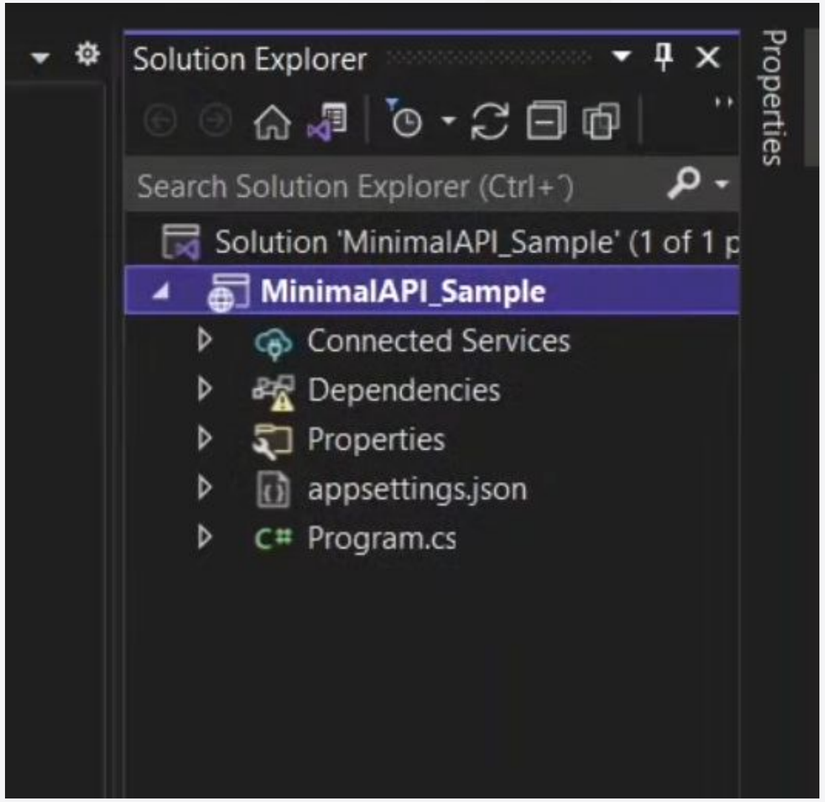
# ¿QUÉ ES UNA API MÍNIMA?

- Características:
  - No es necesario el uso de controladores
  - Se crea la configuración y se puede usar en el mismo archivo
  - La definición de los endpoints proviene de las propiedades de la app configurada anteriormente.
  - Puede trabajar con uno o varios puertos
  - Unificación de configuraciones en un solo archivo

# DIFERENCIAS CON API TRADICIONAL




API mínima




API tradicional

**Ejemplos estructuras**

# DIFERENCIAS CON API TRADICIONAL

```
public class Startup
{
    0 references
    public Startup(IConfiguration configuration)
    {
        Configuration = configuration;
    }

    1 reference
    public IConfiguration Configuration { get; }

    // This method gets called by the runtime. Use this method to add services to the container.
    0 references
    public void ConfigureServices(IServiceCollection services)
    {
        services.AddControllers();
        services.AddSwaggerGen(c =>
        {
            c.SwaggerDoc("v1", new OpenApiInfo { Title = "net5", Version = "v1" });
        });
    }

    // This method gets called by the runtime. Use this method to configure the HTTP request pipeline.
    0 references
    public void Configure(IApplicationBuilder app, IWebHostEnvironment env)
    {
        if (env.IsDevelopment())
        {
            app.UseDeveloperExceptionPage();
            app.UseSwagger();
            app.UseSwaggerUI(c => c.SwaggerEndpoint("/swagger/v1/swagger.json", "net5 v1"));
        }
        app.UseHttpsRedirection();
        app.UseRouting();
        app.UseAuthorization();
        app.UseEndpoints(endpoints =>
        {
            endpoints.MapControllers();
        });
    }
}
```

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Hosting;
using Microsoft.Extensions.Configuration;
using Microsoft.Extensions.Hosting;
using Microsoft.Extensions.Logging;

namespace net5
{
    0 references
    public class Program
    {
        0 references
        public static void Main(string[] args)
        {
            CreateHostBuilder(args).Build().Run();
        }

        1 reference
        public static IHostBuilder CreateHostBuilder(string[] args) =>
            Host.CreateDefaultBuilder(args)
                .ConfigureWebHostDefaults(webBuilder =>
                {
                    webBuilder.UseStartup<Startup>();
                });
    }
}
```

**Ejemplos con API tradicional en .NET 5**

# DIFERENCIAS CON API TRADICIONAL

```
using Microsoft.OpenApi.Models;

var builder = WebApplication.CreateBuilder(args);

// Add services to the container.

builder.Services.AddControllers();
builder.Services.AddSwaggerGen(c =>
{
    c.SwaggerDoc("v1", new() { Title = "net6", Version = "v1" });
});

var app = builder.Build();

// Configure the HTTP request pipeline.
if (builder.Environment.IsDevelopment())
{
    app.UseDeveloperExceptionPage();
    app.UseSwagger();
    app.UseSwaggerUI(c => c.SwaggerEndpoint("/swagger/v1/swagger.json", "net6 v1"));
}

app.UseHttpsRedirection();

app.UseAuthorization();

app.MapControllers();

app.Run();
```

Program.cs

**Ejemplo con API mínima en .NET 6**

# DIFERENCIAS CON API TRADICIONAL

Program.cs

```cs
using Microsoft.OpenApi.Models;
using net6; //Nuestro namespace, contiene la clase WeatherForecast

var builder = WebApplication.CreateBuilder(args);
builder.Services.AddEndpointsApiExplorer();
builder.Services.AddSwaggerGen(c=>c.SwaggerDoc("v1", new() { Title = "net6 minimal API", Version = "v1" }));

var app = builder.Build();

if (builder.Environment.IsDevelopment()){
    app.UseDeveloperExceptionPage();
    app.UseSwagger();
    app.UseSwaggerUI(c => c.SwaggerEndpoint("/swagger/v1/swagger.json", "net6 minimal API v1"));
}

string[] Summaries = new[]
{
    "Freezing", "Bracing", "Chilly", "Cool", "Mild", "Warm", "Balmy", "Hot", "Sweltering", "Scorching"
};

app.MapGet("/", ()=>{

    return Enumerable.Range(1, 5).Select(index => new WeatherForecast
    {
        Date = DateTime.Now.AddDays(index),
        TemperatureC = Random.Shared.Next(-20, 55),
        Summary = Summaries[Random.Shared.Next(Summaries.Length)]
    })
    .ToArray();
});

app.UseHttpsRedirection();
app.Run();
```

```cs
using Microsoft.AspNetCore.Mvc;

namespace net6.Controllers;

[ApiController]
[Route("[controller]")]
public class WeatherForecastController : ControllerBase
{
    private static readonly string[] Summaries = new[]
    {
        "Freezing", "Bracing", "Chilly", "Cool", "Mild", "Warm", "Balmy", "Hot", "Sweltering",
    };

    private readonly ILogger<WeatherForecastController> _logger;

    public WeatherForecastController(ILogger<WeatherForecastController> logger)
    {
        _logger = logger;
    }

    [HttpGet]
    public IEnumerable<WeatherForecast> Get()
    {
        return Enumerable.Range(1, 5).Select(index => new WeatherForecast
        {
            Date = DateTime.Now.AddDays(index),
            TemperatureC = Random.Shared.Next(-20, 55),
            Summary = Summaries[Random.Shared.Next(Summaries.Length)]
        })
        .ToArray();
    }
}
```

**Ejemplo con API mínima en .NET 6**