# **Tutorial: Create a web API with ASP.NET Core**

Here's sample code for an API based on controllers:

* **namespace APIWithControllers;**

public class Program

{

public static void Main(string[] args)

{

var builder = WebApplication.CreateBuilder(args);

builder.Services.AddControllers();

var app = builder.Build();

app.UseHttpsRedirection();

app.MapControllers();

app.Run();

}

}

**using Microsoft.AspNetCore.Mvc;**

namespace APIWithControllers.Controllers;

[ApiController]

[Route("[controller]")]

public class WeatherForecastController : ControllerBase

{

private static readonly string[] Summaries = new[]

{

"Freezing", "Bracing", "Chilly", "Cool", "Mild", "Warm", "Balmy", "Hot", "Sweltering", "Scorching"

};

private readonly ILogger<WeatherForecastController> \_logger;

public WeatherForecastController(ILogger<WeatherForecastController> logger)

{

\_logger = logger;

}

[HttpGet(Name = "GetWeatherForecast")]

public IEnumerable<WeatherForecast> Get()

{

return Enumerable.Range(1, 5).Select(index => new WeatherForecast

{

Date = DateOnly.FromDateTime(DateTime.Now.AddDays(index)),

TemperatureC = Random.Shared.Next(-20, 55),

Summary = Summaries[Random.Shared.Next(Summaries.Length)]

})

.ToArray();

}

}

* **namespace MinimalAPI;**

public class Program

{

public static void Main(string[] args)

{

var builder = WebApplication.CreateBuilder(args);

var app = builder.Build();

app.UseHttpsRedirection();

var summaries = new[]

{

"Freezing", "Bracing", "Chilly", "Cool", "Mild", "Warm", "Balmy", "Hot", "Sweltering", "Scorching"

};

app.MapGet("/weatherforecast", (HttpContext httpContext) =>

{

var forecast = Enumerable.Range(1, 5).Select(index =>

new WeatherForecast

{

Date = DateOnly.FromDateTime(DateTime.Now.AddDays(index)),

TemperatureC = Random.Shared.Next(-20, 55),

Summary = summaries[Random.Shared.Next(summaries.Length)]

})

.ToArray();

return forecast;

});

app.Run();

}

}

* **namespace MinimalAPI;**

public class Program

{

public static void Main(string[] args)

{

var builder = WebApplication.CreateBuilder(args);

var app = builder.Build();

app.UseHttpsRedirection();

var summaries = new[]

{

"Freezing", "Bracing", "Chilly", "Cool", "Mild", "Warm", "Balmy", "Hot", "Sweltering", "Scorching"

};

app.MapGet("/weatherforecast", (HttpContext httpContext) =>

{

var forecast = Enumerable.Range(1, 5).Select(index =>

new WeatherForecast

{

Date = DateOnly.FromDateTime(DateTime.Now.AddDays(index)),

TemperatureC = Random.Shared.Next(-20, 55),

Summary = summaries[Random.Shared.Next(summaries.Length)]

})

.ToArray();

return forecast;

});

app.Run();

}

}

**Both API projects refer to the following class:**

* **namespace APIWithControllers;**

public class WeatherForecast

{

public DateOnly Date { get; set; }

public int TemperatureC { get; set; }

public int TemperatureF => 32 + (int)(TemperatureC / 0.5556);

public string? Summary { get; set; }

}

## **Create a web project**

**Visual Studio**

* From the ****File**** menu, select ****New**** > ****Project****.
* Enter Web API in the search box.
* Select the ****ASP.NET Core Web API**** template and select ****Next****.
* In the ****Configure your new project dialog****, name the project TodoApi and select ****Next****.
* In the ****Additional information**** dialog:
  + Confirm the ****Framework**** is ****.NET 7.0**** (or later).
  + Confirm the checkbox for ****Use controllers(uncheck to use minimal APIs)**** is checked.
  + Select ****Create****.

**Visual Studio Code**

* Open the [integrated terminal](https://code.visualstudio.com/docs/editor/integrated-terminal).
* Change directories (cd) to the folder that will contain the project folder.
* Run the following commands:

**NET CLI**

dotnet new webapi -o TodoApi

cd TodoApi

dotnet add package Microsoft.EntityFrameworkCore.InMemory

code -r ../TodoApi

**These commands:**

* + Create a new web API project and open it in Visual Studio Code.
  + Add a NuGet package that is needed for the next section.

When a dialog box asks if you want to add required assets to the project, select ****Yes.****

****Run without debugging select appropriate NuGet****

### **Test the project**

Press Ctrl+F5 to run without the debugger.

Visual Studio displays the following dialog when a project is not yet configured to use SSL:

Select ****Yes**** if you trust the IIS Express SSL certificate.

[]