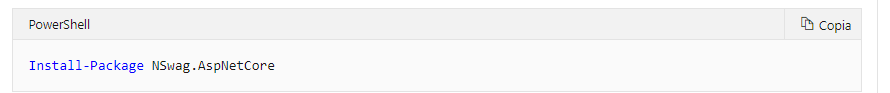
**Swagger implementation con netcore:**

Tutorial online:

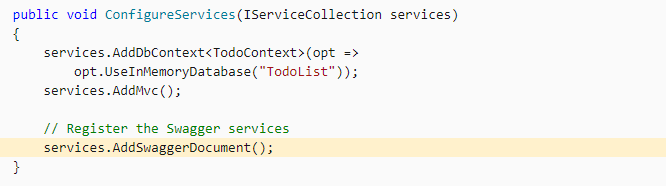
<https://docs.microsoft.com/it-it/aspnet/core/tutorials/getting-started-with-nswag?view=aspnetcore-3.1&tabs=visual-studio>

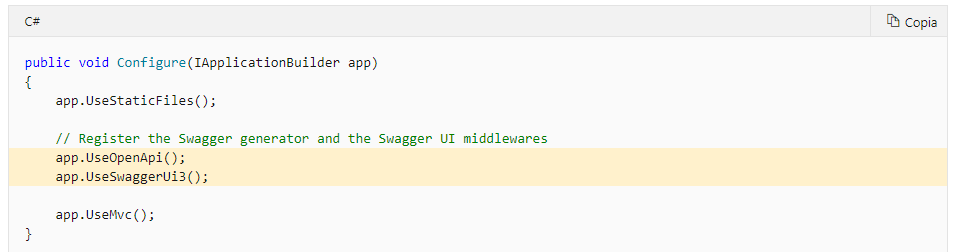
Passaggi:

Installare swagger



Aggiungere il servizio in startup:





Per creare una personalizzazione della documentazione api aggiungere a startup questo:

services.AddSwaggerDocument(config =>

{

config.PostProcess = document =>

{

document.Info.Version = "v1";

document.Info.Title = "ToDo API";

document.Info.Description = "A simple ASP.NET Core web API";

document.Info.TermsOfService = "None";

document.Info.Contact = new NSwag.OpenApiContact

{

Name = "Shayne Boyer",

Email = string.Empty,

Url = "https://twitter.com/spboyer"

};

document.Info.License = new NSwag.OpenApiLicense

{

Name = "Use under LICX",

Url = "https://example.com/license"

};

};

});

per abilitare i commenti aggiungere al file prj la seguente sezione:

<PropertyGroup> <GenerateDocumentationFile>true</GenerateDocumentationFile>

<NoWarn>$(NoWarn);1591</NoWarn>

</PropertyGroup>

Per far gestire dall’ambiente le risposte di ritorno (stati) in base a cio che succede durante l’esecuzione dell’api usare gli attributi ad esempio su un post:

[ProducesResponseType(StatusCodes.Status201Created)] // Created

[ProducesResponseType(StatusCodes.Status400BadRequest)] // BadRequest

Questo aggiunge due stati come argomento di uscita oltre a quello di default 200 aggiunto automaticamente se la classe è decorata con [ApiController]

Per aggiungere commenti personali alle API, ai parametri in ingresso ed ai valori di ritorno commentare direttamente il codice:

