OpenAPI uso con Minimal API

Genova – 18/12/2024

Andrea Merlin

Software Architect

@an_merlin - https://amerlin.keantex.com

DotNet Liguria

OpenAPI – API Client generation

OpenAPI – API Client generation

Gestione di API

Creazione di almeno 1 Client API (es. Api Rest/Json)

Lavoro noioso

Introduzione di errori

Spesso include la necessità di integrazione con sistemi di autenticazione

Copia & incolla di Response e Request

Serializzazione e Deserializzazione delle Request e delle Response

Questo tipo di attività è comune a qualsiasi linguaggio di programmazione ed i problemi sono sempre gli stessi

OpenAPI – API Client generation

Gestione di API - Problematiche

Gestione dei client – Stiamo sviluppando API che devono essere usate da diversi client (es. IOS App, Android App, C#....)

API capabilities: un grande quantità di endpoint, con modelli di richiesta e risposta differenti. Se cambia l'api, è necessario aggiornare tutte le chiamate/risposte

Mantenere sincronizzati i cambiamenti

```
public class Order
{
    public int Id { get; set; }
    public int UserId { get; set; }
    public DateTimeOffset OrderDate { get; set; }
}
```

```
interface Order
{
    id: number;
    userId: number;
    orderDate: Date;
}
```

Avere la documentazione aggiornata sarebbe una buona cosa!

Ricordiamo SOAP con il contratto WSDL?

Nella Api risulta nullable, mentre in Typescript NO!

```
public class Order
{
    public int Id { get; set; }
    public int UserId { get; set; }
    public DateTimeOffset OrderDate { get; set; }
    public DateTimeOffset? DeliveryDate { get; set; }
}
```

```
interface Order
{
    id: number;
    userId: number;
    orderDate: Date;
    deliveryDate: Date; //Mismatch!
}
```

OpenAPI – API Client generation

Gestione di API

Sarebbe molto comodo avere un “qualcosa” che ci permetta di generare questo/i client in maniera semplice

E proprio, per dirla tutta, se fosse possibile generare questi client in maniera automatica

L’obiettivo sarebbe quello di poter generare il codice in maniera automatica, ad esempio ad ogni variazione delle api

Tutto questo possibile utilizzando OpenApi

OpenAPI – API Client generation

HATEOAS

DRY: *Don't repeat yourself*

Come possiamo aggiornare tutti i nostri client senza aggiornarli ?

HATEOAS fa parte dello standard REST API, anche se spesso non viene utilizzato

Le informazioni vengono inserite direttamente all’interno delle risposte

> Con HATEOAS, un client interagisce con un'applicazione di rete i cui server applicativi forniscono informazioni in modo dinamico tramite ipermedia. Un client REST ha bisogno di poche o nessuna conoscenza pregressa su come interagire con un'applicazione o un server, oltre a una conoscenza generica dell'ipermedia.
> — Wikipedia -

OpenAPI – API Client generation

HATEOAS

```
{
  "id": 20,
  "userId": 12,
  "text": "just setting up my twttr",
  "favorites": 163601,
  "retweets": 500
}
```

```
tweetCanBeDeleted(): boolean {
    return this.tweet.links.some((x) => x.rel === 'delete');
}
```

```
{
  "id": 20,
  "userId": 12,
  "text": "just setting up my twttr",
  "favorites": 163601,
  "retweets": 500,
  "links": [
    {
      "href": "api/tweets/20",
      "rel": "delete",
      "type": "DELETE"
    },
    {
      "href": "api/tweets/20/retweet",
      "rel": "retweet",
      "type": "POST"
    },
    {
      "href": "api/tweets/20/favorite",
      "rel": "favorite",
      "type": "POST"
    },
  ]
}
```

OpenAPI – API Client generation

HATEOAS

HATEOAS specifica il metodo HTTP e l'URL dell'azione, ma uno schema per la convalida o il corpo della richiesta non fa parte dello standard

Significa che è necessario avere una certa conoscenza delle aspettative del server, vanificando lo scopo originale di HATEOAS

HATEOAS mira a disaccoppiare il client ed il server, ma il client deve comunque conoscere «rel» per poter operare

L’accopiamento non scompare del tutto.

OpenAPI

Cos’è OpenAPI?

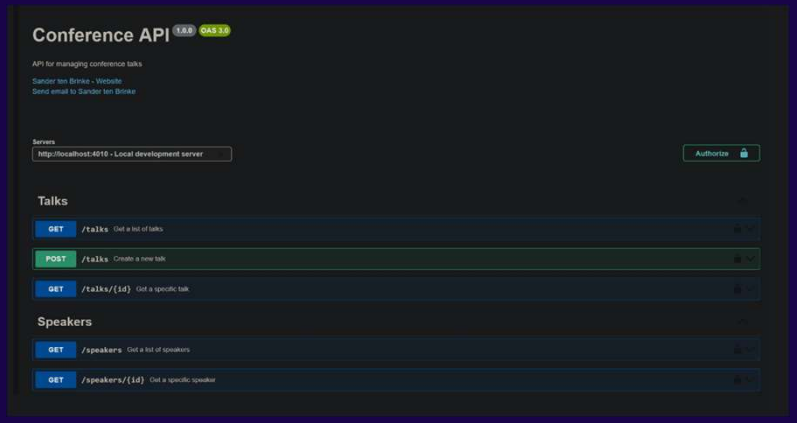
OpenAPI è una specifica per la progettazione, la creazione e la documentazione delle API, realizzata e gestita dall'iniziativa OpenAPI

Per capire come funziona un'API, potrebbe essere necessario entrare nel codice. Tuttavia, questo diventa piuttosto difficile se non conosci il linguaggio di programmazione o il framework pertinente.

Una specifica OpenAPI è un contratto che descrive l'API ed è un ottimo modo per comunicare sull’API

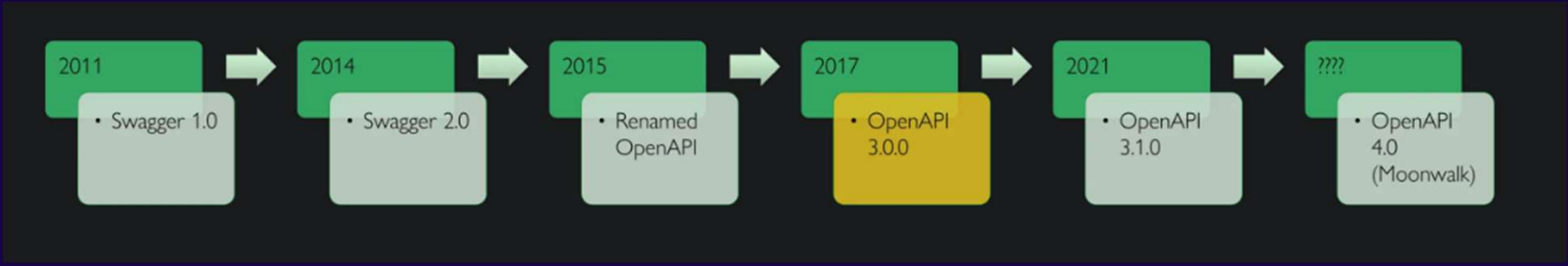
OpenAPI è una specifica, non è legata a un linguaggio di programmazione o framework specifico: può essere utilizzata con qualsiasi linguaggio di programmazione

Esistono numerosi Tools che pemettono di utilizzare le specifiche OpenAPI. Il più noto è sicuramente SwaggerUI.



OpenAPI – API Client generation

OpenAPI - Storia



Spesso nessuno conosce OpenAPI. Ma se si chiede quanti conoscono Swagger tutti lo conoscono! ☺

All'inizio il progetto si chiamava Swagger. Poi è stato rinominato in OpenApi nel 2015

Sono attivi lavori per il rilascio di una nuova versione Sig-MoonWalk (https://github.com/OAI/sig-moonwalk)

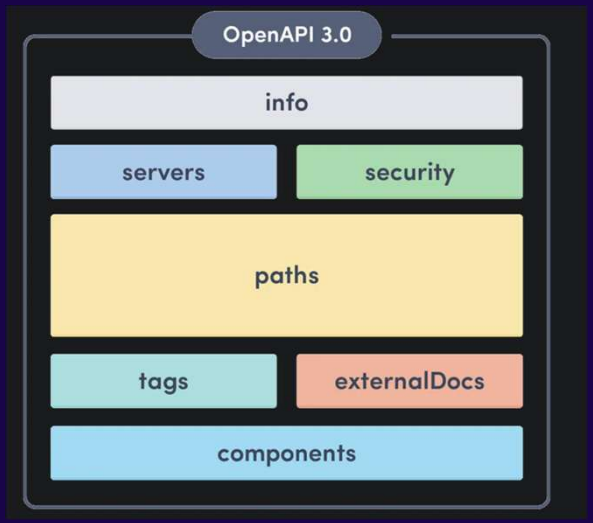
Current Version Version 3.1.0

OpenAPI – API Client generation

OpenAPI - Specifiche

Una specifica può essere scritta in JSON o YAML

Una specifica è composta da:



- Metadati: informazioni generali dell'api (es. Titolo, version, description, server info…
- Paths: tutti gli endpoint dell'API. Informazioni sui modelli delle richieste/risposte, sicurezza, ecc…
- Components: componenti riutilizzabili come I model delle request e delle response, schemi di sicurezza, ecc…
- Security: informazioni sulla sicrurezza dell'API
- Tag: informazioni sui tag dell' API

OpenAPI – API Client generation

OpenAPI - Metadati

```yaml
openapi: 3.0.0
info:
  title: Conference API
  description: API for managing conference talks
  contact:
    name: [redacted]
    url: [redacted]
    email: [redacted]
  # Note: This is the version of your OpenAPI specification, NOT related to API versioning!
  version: 1.0.0
externalDocs:
  url: 'https://example.com/docs'
  description: Find more info here
servers:
  - url: https://api.example.com
    description: Production server
```

OpenAPI – API Client generation

OpenAPI - EndPoint

```yaml
paths:
  /talks:
    get:
      tags:
        - "Talks"
      summary: Get a list of talks
      operationId: Talks_GetTalks
      responses: # Will be handled in the next sections
    post:
      tags:
        - "Talks"
      summary: Create a new talk
      operationId: Talks_CreateTalk
      requestBody: # Will be handled in the next sections
    # Other HTTP methods for /talks can be set up here
  # Other endpoints can be set up here
```

OpenAPI – API Client generation

OpenAPI – EndPoint with parameters

```yaml
paths:
  # Endpoints from the previous example would be here
  /talks/{id}:
    get:
      tags:
        - "Talks"
      summary: Get a specific talk
      operationId: Talks_GetTalk
      parameters:
        - name: id
          in: path
          required: true
          schema:
            type: integer
```

OpenAPI – API Client generation

OpenAPI – Request and response bodies

```yaml
paths:
  /talks:
    post:
      tags:
        - "Talks"
      summary: Create a new talk
      operationId: Talks_CreateTalk
      requestBody:
        required: true
        content:
          application/json:
            schema: # Will be handled next!
              $ref: "#/components/schemas/Talk"
      responses:
        "201":
          description: Created
          content:
            application/json:
              schema: # Will be handled next!
                $ref: "#/components/schemas/Talk"
        "400":
          description: Bad Request
          content:
            application/json:
              schema: # Will be handled next!
                $ref: "#/components/schemas/ProblemDetails"
```

OpenAPI – API Client generation

OpenAPI – Components

```yaml
components: # (Showing only a subset of component types)
  parameters: # ...
  requestBodies: # ... (There's a lot more!)
  schemas:
    Talk:
      type: object
      properties:
        id:
          type: integer
        title:
          type: string
        speaker:
          $ref: "#/components/schemas/Speaker"
        time:
          type: string
          format: date-time
    Speaker:
      type: object
      properties:
        firstname:
          type: string
        lastname:
          type: string
```

OpenAPI – API Client generation

OpenAPI – Security

```yaml
components:
  securitySchemes:
    BasicAuth: # …
    BearerAuth: # …
    ApiKeyAuth: # …
    OpenID: # …
    OAuth2:
      type: oauth2
      flows:
        authorizationCode:
          authorizationUrl: https://example.com/oauth/authorize
          tokenUrl: https://example.com/oauth/token
          scopes:
            read: Grants read access
            write: Grants write access
            admin: Grants access to admin operations
```

Questo schema di sicurezza è solo una definizione. Dobbiamo ancora applicarlo agli endpoint

Applicazione dello schema OAuth2 globalmente:

```yaml
security:
  - OAuth2:
      - read
      - write
  - ApiKeyAuth: []
```

OpenAPI – API Client generation

OpenAPI – Tags

```yaml
tags:
  - name: Talks
    description: Endpoints for managing talks
  - name: Speakers
    description: Endpoints for managing speakers
```

E’ un modo per raggruppare insieme i tuoi endpoint.

È utile per scopi di documentazione, ma anche per generare client API

È possibile definirli per ogni endpoint, come visto negli esempi precedenti, oppure globalmente nel documento

OpenAPI – API Client generation

OpenAPI – Design-First vs Code-First

Due approcci per definire il documento: **Design-First** e **Code-First**

Un approccio design-first significa che prima viene scritto il documento OpenAPI e poi viene implementato nel codice. Questo approccio parte dal presupposto che tutto quello che è nel documento, è fondamentale per lo sviluppo

DESIGN FIRST:

Dobbiamo scrivere sia il documento OpenAPI sia l'implementazione del server, il che significa scrivere la stessa cosa due volte

Poiché stiamo scrivendo la stessa cosa due volte, dobbiamo fare attenzione a implementare correttamente il documento OpenAPI nel nostro codice server. Qualsiasi errore nell'implementazione riduce il valore del documento OpenAPI.

OpenAPI – API Client generation

OpenAPI – Design-First vs Code-First

CODE FIRST:

Viene scritto prima il codice sul server, e solo dopo viene generato il documento

Viene scritto solo il codice, ed il documento viene generato direttamente

Potrebbe essere necessario integrare il codice con informazioni extra, per generare correttamente il codice con OpenAPI

In .NET esiste un ampio ecosistema per generare automaticamente il documento:

Swashbuckle.AspNetCore, Nswag e la libreria Microsoft OpenApi

OpenAPI – API Client generation

OpenAPI – Documentazione

Un Api può essere complessa e difficile da comprendere
Identificare quali endpoint sono disponibili
Quali sono le request/response che si possono utilizzare
Quando usare un endpoint e cosa fa ?

Gli strumenti messi a disposizione da openAPI possono aiutarci in questo

SwaggerUI
Redocly
Scalar (ultimo arrivato)

OpenAPI – API Client generation

OpenAPI – Generazione Server

In un mondo perfetto, si potrebbe scrivere una specifica OpenAPI e generare un server in qualsiasi linguaggio

Cambiando lo stack tecnologico si potrebbe rigenerare il tutto con un linguaggio diverso

Uno strumento è OpenAPI-generator

Non viviamo in un mondo perfetto: l'API non è soltanto un endpoint

La logica non viene comunque generata, quindi è sempre necessario generarla manualmente

Potrebbe essere interessante utilizzare Mock Servers, come ad esempio Prism .

Progetto interessante: Open-Api-Mocker, che consente di eseguire un server mock con dati fake, basato su openAPI

Anche Scalar offre la possibilità di generare un mock server.

OpenAPI – API Client generation

OpenAPI – Generazione di client

Una delle funzionalità più potenti di OpenAPI è la capacità di generare client API.

E’ possibile generare client API in qualsiasi linguaggio

Risparmio in termini di tempo

OpenAPI – API Client generation

OpenAPI – Generazione di client

Si tende a pensare che i client API generati siano "brutti" o non funzionino bene.
Non è del tutto vero: Il problema principale è spesso che il loro documento OpenAPI non è corretto o non contiene informazioni.

"Garbage in, garbage out". Quindi, se scrivete un buon documento OpenAPI, otterrete in cambio un buon client API, e anche il contrario è vero 😉

Ogni framework ha modi diversi per finire con un buon documento OpenAPI. Per .NET

OpenAPI – API Client generation

openapi.tools

OpenAPI – Generazione di client

Una delle funzionalità più potenti di OpenAPI è la capacità di generare client API.

E’ possibile generare client API in qualsiasi linguaggio

Risparmio in termini di tempo

- NSwag può generare client C# e TypeScript. Può anche essere utilizzato nella generazione di client TypeScript per framework/librerie
- Swashbuckle.AspNetCore ha anche un generatore, ma supporta solo C#.
- Kiota generatore di client API di Microsoft per un'ampia varietà di linguaggi. Microsoft Graph SDK il suo client API è generato con Kiota
- OpenAPI Generator è uno strumento molto diffuso in grado di generare client API in un'ampia gamma di linguaggi.

OpenAPI – API Client generation

OpenAPI – NSwag

NSwag è uno strumento open source creato da Rico Suter e altri membri della community.

Ha più di 50 milioni di download

Probabilmente, il miglior strumento per generare client API nell'ecosistema .NET per C# e TypeScript.

Viene persino utilizzato da Visual Studio quando genera client API direttamente dall'IDE

NSwag non genera solo client API. Può anche generare documentazione API che si collega direttamente ai progetti ASP.NET Core

Può essere utilizzato da cli, da GUI (NswagStudio) e integrato direttamente all'interno di MSBuild

E' necessario creare un file .nswag che contiene le impostazioni per il generatore

OpenAPI – API Client generation

OpenAPI – NSwag

Una volta definito il file .nswag con la configurazione è possibile utilizzare nswag (anche dalla GUI) per la generazione del client

```
nswag run conference.nswag
```

Questo codice consente di generare il file .cs contenente il codice del client generato. Nel codice generato verranno create anche tutte le interface (e le loro implementazioni)

OpenAPI – API Client generation

OpenAPI – Kiota

Kiota è uno strumento di generazione client API open source realizzato da Microsoft.

È noto soprattutto per essere utilizzato in Microsoft Graph SDK, piattaforma in cui è possibile accedere a tutto ciò che riguarda i servizi Microsoft utilizzando una singola API.

Si può usare con la CLI, si può usare in dotnet , in Docker e altro

E’ disponibile un estensione per Visual studio Code

E’ possibile utilizzare il comando kiota search per cercare in registri pubblici e privati i documenti openAPI

Es. Kiota search github, Kiota download, Kiota generate per la generazione del client

OpenAPI – API Client generation

OpenAPI – Kiota

```
# Kiota is installed as a dotnet tool in this case
dotnet kiota generate --clean-output \
                      --language "csharp" \
                      --openapi "conference-openapi-definition.json" \
                      --output "ApiClients/Kiota" \
                      --namespace-name "ConferenceApp.Clients.Kiota" \
                      --class-name "KiotaConferenceClient"
```

OpenAPI – API Client generation

OpenAPI – OpenAPI Generator

Supporta fino a 70 languaggi di programmazione

Può essere installato tramite npm, brew, scoop, docker e jar

Ad esempio è possibile utilizzarlo all'intern di docker

Viene generato un client API, documentazione dei Pacchetti, istruzioni per l'uso …

```
docker run --rm -v ${PWD}:/local \
          openapitools/openapi-generator-cli:v7.6.0 generate \
           -i /local/conference_api.yml \
           -g csharp \
           -o /local/out/csharp
```

OpenAPI – API Client generation

OpenAPI – Automazione del processo

Per assicurarsi che dotnet nswag sia sempre disponibil all'interno dell'ambiente di sviluppo è possibile installarlo come ambiente locale

https://learn.microsoft.com/en-us/dotnet/core/tools/global-tools?WT.mc_id=DT-MVP-5005050

```
# Ensure you're in the root of your project
# You only need to perform these 2 steps once.
dotnet new tool-manifest
dotnet tool install nswag.console

# Run this (once) to install the tool and make it available in your project
# Don't forget to update your README.MD so others know they need to run this as well if they want
to generate an API client.
dotnet tool restore

# Testing if nswag CLI works
dotnet nswag help
```

OpenAPI – API Client generation

OpenAPI – Automazione del processo

Generazione del client tramite script

File di configurazione nswag

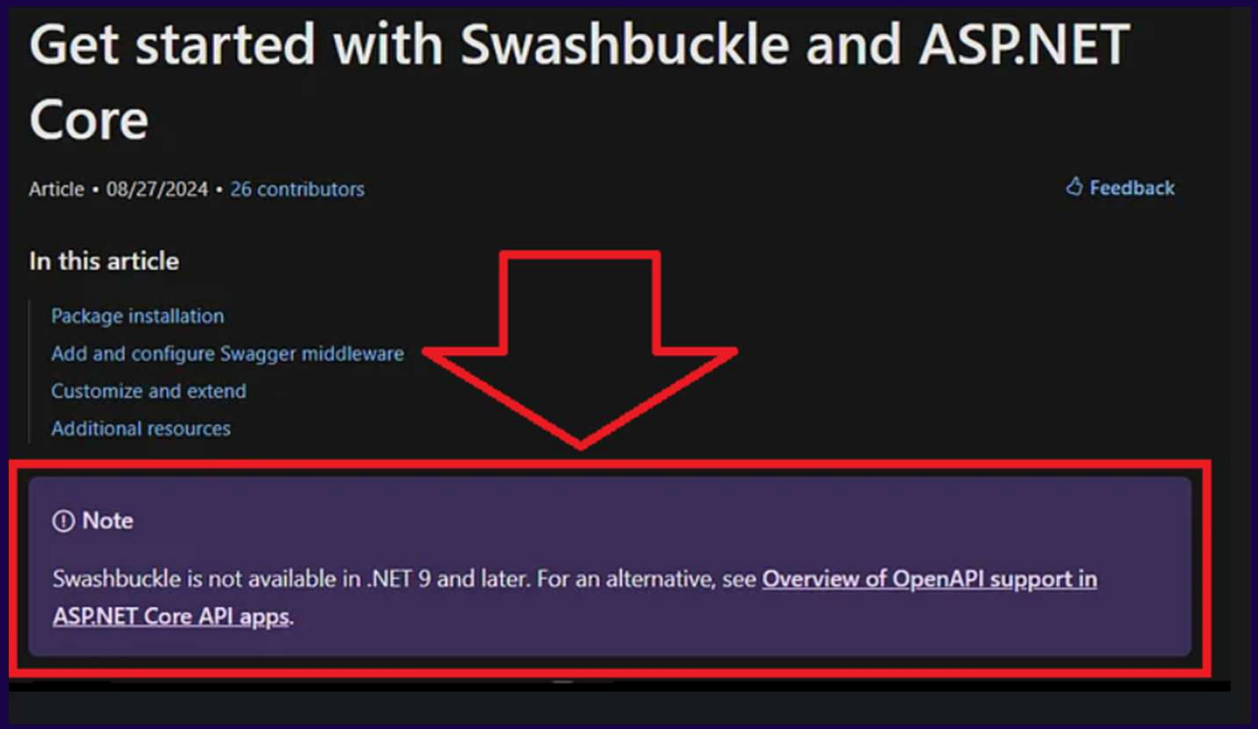
Script in power shell, ma anche script in bash

Eventualmente pubblicare il client generato all'interno di un feed in modo da renderlo accessibile dai componenti del team

```
$ErrorActionPreference = "Stop"

try {
    # Makes the script work when called from any directory
    $scriptPath = Join-Path -Path $PSScriptRoot -ChildPath
"../PATH/TO/NSWAG-CONFIG/conference.nswag"

    Write-Host "Generating Conference API Client with
NSwag..`n"

    & dotnet nswag run $scriptPath

    if ($LASTEXITCODE -ne 0) {
        throw "NSwag code generation failed with exit code
$LASTEXITCODE"
    }

    Write-Host "Conference API client has been generated
successfully."
    Write-Host "The nuget package has NOT been created yet.
This can be done with {YOUR_CI/CD_PIPELINE}."
}
catch {
    Write-Host "An error occurred: $($_.Exception.Message)"
    exit 1
}
```

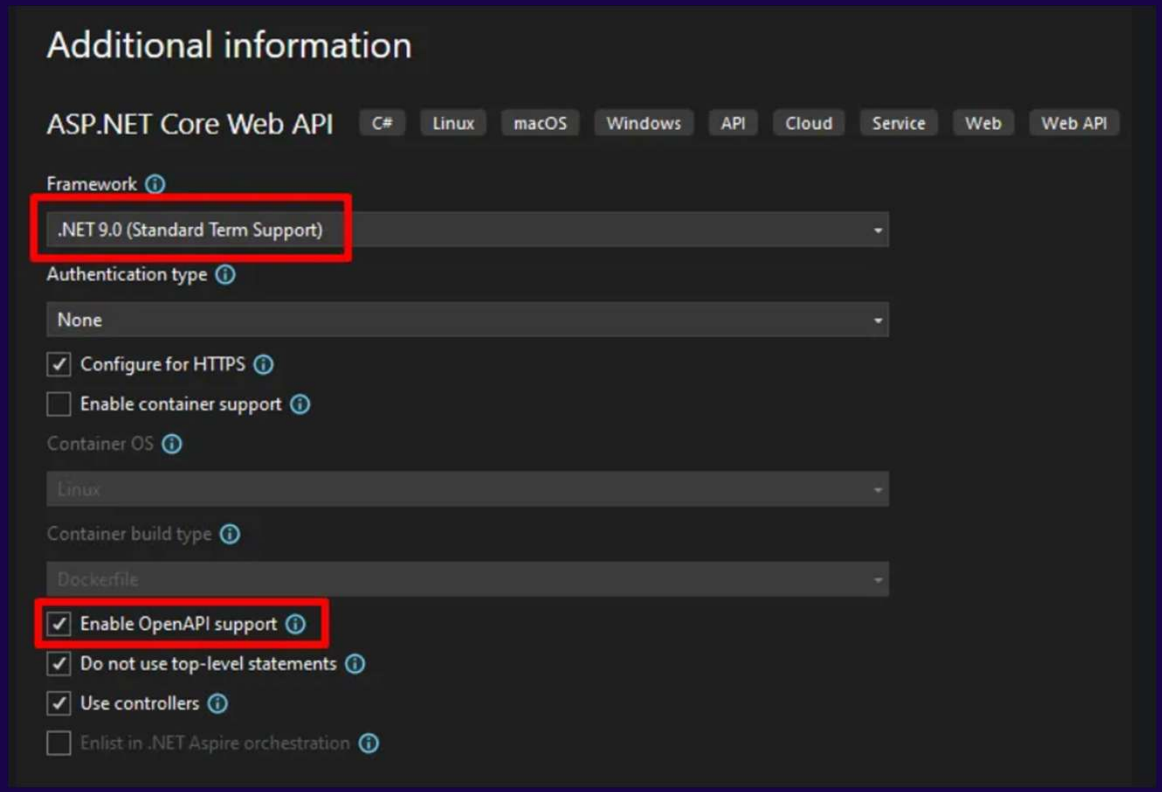
.Net 9 – OpenApi

.Net 9 – OpenApi - Microsoft.AspNetCore.OpenApi



.Net 9 – OpenApi

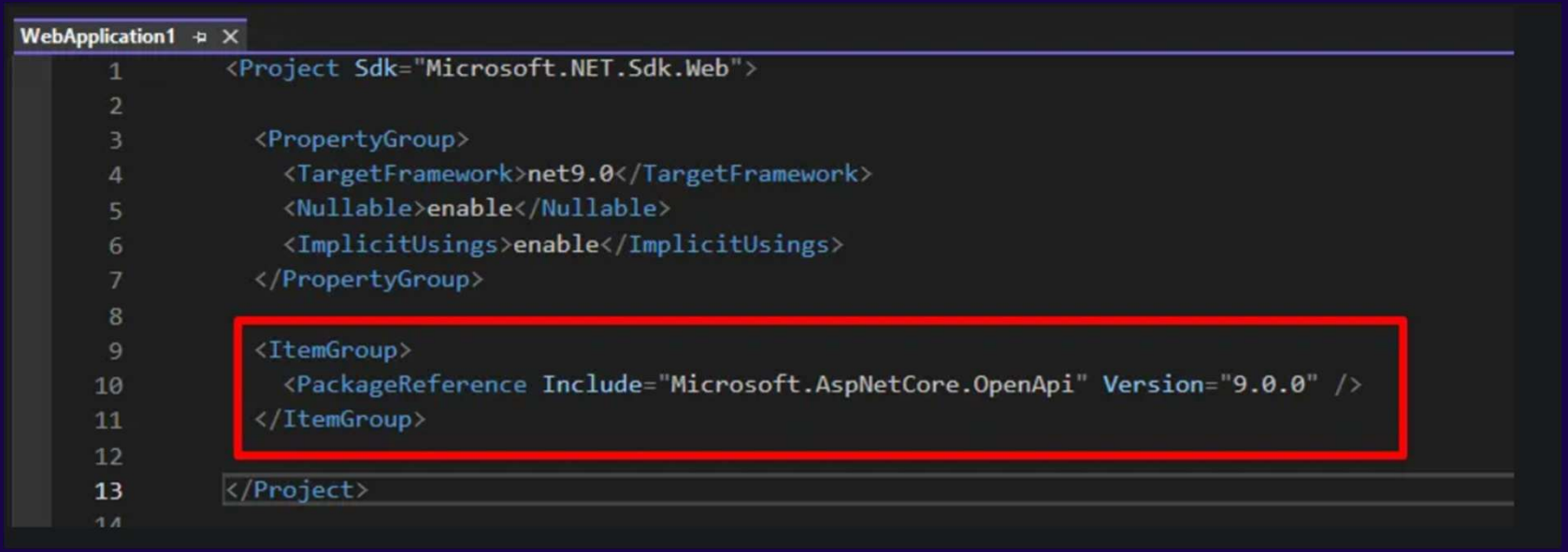
.Net 9 – OpenApi - Microsoft.AspNetCore.OpenApi




```
dotnet add package Microsoft.AspNetCore.OpenApi
```

.Net 9 – OpenApi

.Net 9 – OpenApi - Microsoft.AspNetCore.OpenApi



.Net 9 – OpenApi

.Net 9 – OpenApi - Microsoft.AspNetCore.OpenApi

Discussione 18 Marzo 2024 – Uscita del pacchetto SwashBuckle.AspNetCore da .Net 9 (tanto per intenderci Swagger)

Il pacchetto SwashBuckle.AspNetCore era composto da un componente per la generazione della doc in formato OpeanApi e da Swagger

Swagger escluso dai template di progetto: non era mantenuto da tempo (mai portato a .NET 8)

Microsoft sta rimuovendo dipendenze di librerie di terze parti dai Template di progetto

Nuova gestione con OpenAPI - Esperienza molto piu ricca e intuitiva oltre Swagger

.Net 9 – OpenApi

.Net 9 – OpenApi - Microsoft.AspNetCore.OpenApi

Supporto Multiprotocol

Il generator analizza la documentazione OpenApi analizzando gli attributi [HttpGet], [ProducesResponseType]

Estensioni OpenAPI per poter interagire ed avere documentazione più completa

AI-Driven Documentation Assistants

Real time Api Metrics

.Net 9 – OpenApi

.Net 9 – OpenApi - Microsoft.AspNetCore.OpenApi

Nuova gestione con OpenAPI - Esperienza molto piu ricca e intuitiva rispetto a Swagger

Support Multiprotocol

Il generator analizza la documentazione OpenApi analizzando gli attributi [HttpGet], [ProducesResponseType]

Estensioni OpenAPI per poter interagire ed avere documentazione più completa

AI-Driven Documentation Assistants

Real time Api Metrics

Native AoT-friendly

Possibilità di personalizzazione con i transformers

.Net 9 – OpenApi

.Net 9 – OpenApi - Microsoft.AspNetCore.OpenApi

OpenAPI in .NET 9

Support generating OpenAPI documents for minimal and controller-based APIs built on top of System.Text.Json for schema generation.

Native AoT-friendly

Built-in OpenAPI document generation is Native AoT-friendly by default and supports native AoT-compatible end-to-ends when used with minimal APIs.

Enhance OpenAPI with metadata

Add OpenAPI metadata (descriptions, tags, validations, etc) with attributes or extension methods (XML doc support in experimental preview)

Customize OpenAPI with transformers

Use Document, Operation, and/or Schema transformers to customize the generated OpenAPI document based on your needs.

.Net 9 – OpenApi

.Net 9 – Generazione della documentazione dell'endpoint

```
public class ProductController : ControllerBase
{
    [HttpGet("{id}")]
    [ProducesResponseType(typeof(Product), StatusCodes.Status200OK)]
    [ProducesResponseType(StatusCodes.Status404NotFound)]
    public IActionResult GetProduct(int id)
    {
        var product = productService.GetProduct(id);
        return product != null ? Ok(product) : NotFound();
    }
}
```

Il generatore analizzerà gli attributi, come [ProductResponseType] per produrre automaticamente la documentazione

.Net 9 – OpenApi

.Net 9 – Estensioni OpenApi

```
[HttpPost]
[Route("add-product" )]
[OpenApiCustomTag("API e-commerce")]
[OpenApiResponseExample(typeof(ProductResponse), typeof(ProductResponseExampleProvider))]
public async Task<IActionResult> AddProduct (ProductRequest request)
{
    // Logica dell'endpoint
}
```

Vengono utilizzati i tag OpenApiResponseExample e OpenApiCustomTag che in Swagger dovrebbero essere inseriti all’interno del file swagger.json, oppure utilizzare delle annotazioni separate

.Net 9 – OpenApi

.Net Core 9 – Documentazione Multi protocollo

```
[HttpGet("products")]
[GrpcRoute("grpc/ProductService/GetProducts")]
[OpenApiOperation("List all products", Description = "Gets a list of all available products.")]
public async Task<IActionResult> GetProducts()
{
    // Retrieve and return products
}
```

Viene utilizzato [GrpcRoute] che in .NET 9 integra gRPC . Swagger non supporta gRPC senza particolari configurazioni o tool esterni

.Net 9 – OpenApi

.Net Core 9 – AI-Driven Documentation Assistants

```
{
    "assistantConfig": {
        "enableAI": true,
        "suggestions": [
            {
                "context": "GetProduct",
                "example": "GET /api/products/{id} - Retrieves product by ID"
            }
        ]
    }
}
```

Si può effettuare l’embed realtime per gli esempi delle singole API

.Net 9 – OpenApi

.Net Core 9 – Real-Time Api Metrics

```
{
    "metrics": {
        "endpoint": "/api/products",
        "averageResponseTime": "120ms",
        "errorRate": "2%",
        "lastError": "Product not found"
    }
}
```

Permette di ottenere una visione immediata delle prestazioni degli endpoint e degli errori
Maggiore velocità nella diagnostica dei problemi

Swagger richiede integrazioni e strumenti esterni per fare questo tipo di analisi.

.Net 9 – OpenApi

.Net Core 9 – Scalar

```
dotnet add package Scalar.AspNetCore
```

```
app.MapScalarApiReference();
```

/scalar/v1

```
app.MapScalarApiReference(o =>
    o.WithTheme(ScalarTheme.None)
    .WithEndpointPrefix("none/{documentName}")
);
```

```
app.MapPost("/weatherforecast", (WeatherForecast forecast) =>
{
    return forecast;
});

app.MapPut("/pets/{id}", (string id) =>
{
    return id;
});
app.MapDelete("/pets/{id}", (string id) =>
{
    return id;
});
```

```
app.MapScalarApiReference(o => o
    .WithTheme(ScalarTheme.Mars)
    .WithModels(false)
    .WithSidebar(false)
    .WithEndpointPrefix("special/{documentName}")
);
```

.Net 9 – OpenApi

.Net Core 9 – Recap

```
dotnet add package Microsoft.AspNetCore.OpenApi
```

```
builder.Services.AddOpenApi();
app.MapOpenApi();
```

```
builder.Services.AddOpenApi(options =>
{
    options.AddDocumentTransformer((document, context, cancellationToken) =>
    {
        document.Info.Contact = new OpenApiContact
        {
            Name = "ByteHide Support",
            Email = "support@bytehide.com"
        };
        return Task.CompletedTask;
    });
});
```

```
app.MapGet("/hello/{name}", (string name) => $"Hello, {name}!")
    .WithSummary("Get a personalized greeting")
    .WithDescription("This endpoint returns a personalized greeting based on the provided name.")
    .WithTag("Greetings");
```

.Net 9 – OpenApi

.Net Core 9 – Recap

```
builder.Services.AddOpenApi(options ⇒
{
    options.AddDocumentTransformer((document, context, cancellationToken) ⇒
    {
        document.Info.Contact = new OpenApiContact
        {
            Name = "ByteHide Support",
            Email = "support@bytehide.com"
        };
        return Task.CompletedTask;
    });
});
```

```
Microsoft.Extensions.ApiDescription.Server
```

```
<PropertyGroup>
  <OpenApiDocumentsDirectory>./</OpenApiDocumentsDirectory>
</PropertyGroup>
```

GRAZIE

Compilate il feedback

Grazie!!!