



OpenAPI and Swagger

Servisno-orijentisane arhitekture 2021

Zašto je Swagger potreban?

- Integracija sa REST API-jima je podložna greškama i nekonzistentna
 - Različiti vendori imaju različitu REST semantiku
 - Klijentske biblioteke se mogu razlikovati (vendor, programski jezik,...)
 - Programeri često izbegavaju pisanje dokumentacije
 - Ulazni parametri, moguće vrednosti, modeli i odgovori se često identifikuju metodom pokušaja i pogreški.

Kontinualan razvoj servisa

- Broj servisa koji nude UI i backend opcije neprekidno raste
 - No-Code aplikacije
 - 19000+ public APIs
- Monolith vs Microservices
 - Industrija sve češće koristi manje servise
 - Veća efikasnost, bolja integracija sa kladom

Šta je Swagger?

- Cilj: Definirati standardni, nezavisan od jezika interfejs prema REST API-ju koji omogućava i ljudima i mašinama da identifikuju mogućnosti servisa bez pristupa izvornom kodu ili dokumentaciji.
- Kada je servis propisno definisan pomoću Swagger-a, klijent može interagovati sa servisom sa minimalnom implementacionom logikom.
- Sličan koncept imamo kod interfejsa na nižem programerskom nivou

Šta je Swagger?

- Swagger generiše interaktivnu API konzolu preko koje korisnici mogu lako probati API
- Generiše klijentski SDK kod potreban za implementaciju na različitim platformama
- Swagger fajl je moguće autogenerisati zahvaljujući anotacijama u kodu ili korišćenjem case class definicije

OpenAPI

- Standardizovan način za opis REST API-ja
- Nezavisan od programskog jezika
- Čitljiv od strane programera, korisnika i mašina

Swagger -> OpenAPI

- OpenAPI je bio poznat kao Swagger specifikacija (Smart Bear)
- 2015. formirana je OpenAPI inicijativa
 - Swagger doniran, preimenovan u OpenAPI specifikaciju
 - Linux fondacija, Google, Microsoft, IBM, i td.
- Pojam Swagger sada često podrazumeva okvir za developerske alate vezane za OpenAPI specifikaciju

OpenAPI dokument

- Opisuje dizajn API-ja
- Može biti u jednom ili više fajlova
- Koristi YAML ili JSON
- Može biti validiran, testiran i portovan

OpenAPI dokument - sadržaj

- API endpointi `/users`
- Operacije za svaki endpoint `GET, POST, ...`
- Ulazni parametri `/users?active=true`
- Odgovori i njihov format `200: {"name": "caca"}`
- Autentifikacija `Basic Auth, OAuth2, ...`
- API meta podaci

Swagger alati

- Postoje i open-source i komercijalni alati
 - Swagger Editor – dizajniranje OpenAPI spec fajlova
 - Swagger UI – generator dokumentacije
 - Swagger CodeGen – generisanje kostura server i klijentskih SDK-ova
 - Swagger Inspector – alat za inspekciju i validaciju

Swagger UI

- Mnogi alati poput Swashbuckle i Nswag uključuju embedded verziju
- Može biti hostovan u ASP.NET Core aplikaciji korišćenjem poziva za registraciju middleware-a
- Svaka javna metoda iz kontrolera može biti testirana
- Petstore example

The screenshot displays the Swagger UI for the Petstore API. At the top, it shows the title 'Swagger Petstore' with a version badge '1.0.5'. Below this, the base URL is specified as 'petstore.swagger.io/v2' and the Swagger JSON file is 'https://petstore.swagger.io/v2/swagger.json'. A descriptive paragraph explains that this is a sample server and provides links for more information and a sample API key. There are also links for terms of service, developer contact, and license information. The interface includes an 'Authorize' button and a 'Schemes' dropdown menu currently set to 'HTTPS'. The main section lists the API endpoints under the 'pet' namespace, categorized by HTTP method: POST for image upload and adding a pet, PUT for updating a pet, and GET for finding pets by status, tags, or ID. A 'store' namespace is also visible at the bottom for accessing orders. Each endpoint entry includes the method, the path with parameters, a brief description, and a lock icon indicating authentication requirements.

Swagger Petstore 1.0.5
[Base URL: petstore.swagger.io/v2]
<https://petstore.swagger.io/v2/swagger.json>

This is a sample server Petstore server. You can find out more about Swagger at <http://swagger.io> or on [#swagger](irc.freenode.net). For this sample, you can use the api key **special-key** to test the authorization filters.

[Terms of service](#)
[Contact the developer](#)
[Apache 2.0](#)
[Find out more about Swagger](#)

Schemes: **HTTPS** [Authorize](#)

pet Everything about your Pets Find out more: <http://swagger.io>

- POST** `/pet/{petId}/uploadImage` uploads an image
- POST** `/pet` Add a new pet to the store
- PUT** `/pet` Update an existing pet
- GET** `/pet/findByStatus` Finds Pets by status
- GET** `/pet/findByTags` Finds Pets by tags
- GET** `/pet/{petId}` Find pet by ID
- POST** `/pet/{petId}` Updates a pet in the store with form data
- DELETE** `/pet/{petId}` Deletes a pet

store Access to Petstore orders

NSwag i ASP.NET Core

- NSwag nudi mogućnost korišćenja Swagger UI i Swagger generatora za ASP.NET Core aplikacije
- Ima mogućnosti za fleksibilno generisanje koda
- Može se koristiti za generisanje klijenata za third-party API-je

NSwag i ASP.NET Core

- Da bi se generisala Swagger specifikacija za implementirani web API i da bi se servirao Swagger UI za pretraživanje i testiranje web API-ja potrebno je registrovati NSwag middleware
 - Instaliranje odgovarajućeg NuGet paketa
<https://www.nuget.org/packages/NSwag.AspNetCore/>
 - Dodavanje i konfigurisanje Swagger middleware-a

Dodavanje i konfigurisanje Swagger middleware-a - 1

```
public void ConfigureServices(IServiceCollection services)
{
    services.AddDbContext<TodoContext>(opt =>
        opt.UseInMemoryDatabase("TodoList"));
    services.AddMvc();

    // Register the Swagger services
    services.AddSwaggerDocument();
}
```

Dodavanje i konfigurisanje Swagger middleware-a - 2

```
public void Configure(IApplicationBuilder app)
{
    app.UseStaticFiles();

    // Register the Swagger generator and the Swagger UI middlewares
    app.UseOpenApi();
    app.UseSwaggerUi3();

    app.UseMvc();
}
```

Dodavanje i konfigurisanje Swagger middleware-a - 3

- Pokretanje aplikacije
 - <http://localhost:<port>/swagger> - Swagger UI
 - <http://localhost:<port>/swagger/v1/swagger.json> Swagger specifikacija

Generisanje koda

- Postoji više načina za generisanje koda pomoću NSwag-a:
 - [NSwagStudio](#): Windows desktop aplikacija za generisanje klijenskog koda u C# ili TypeScript.
- [NSwag.CodeGeneration.CSharp](#) ili [NSwag.CodeGeneration.TypeScript](#) NuGet paketi za generisanje koda unutar projekta
- NSwag iz komandne linije
- [NSwag.MSBuild](#) NuGet paket
- [Unchase OpenAPI \(Swagger\) Connected Service](#): Servis za Visual Studio za generisanje klijenskog koda u C# ili TypeScript. Generiše i C# kontrolere za OpenAPI servise sa NSwag

Generisanje koda pomoću NSwagStudio

- Instalacija NSwagStudio – [github repozitorijum sa instrukcijama za instalaciju](#)
- Nakon pokretanja programa upisati putanju do swagger.json u textbox sa labelom **Swagger Specification URL**
 - Primer:
<http://localhost:44354/swagger/v1/swagger.json>
- **Create local Copy** generiše JSON reprezentaciju date swagger specifikacije

The screenshot shows the NSwagStudio application window. At the top, the 'Runtime' dropdown is set to 'NetCore22'. Below it, a text box contains the URL 'https://localhost:44354/swagger/v1/swagger.json', and a 'Create local Copy' button is to its right. The 'Swagger Specification URL' label is visible above the text box. Below the text box, the 'Swagger Specification JSON (if specified, the URL is ignored):' label is present. The JSON code is displayed in a text area with line numbers 1 through 13. The code is a Swagger 2.0 specification for a web API.

Runtime

NetCore22

Specifies the used command line binary; should match the selected assembly type.

Default Variables ('foo=bar,baz=bar'), usage: \$(foo)

Web API or ASP.NET Core via Reflection (deprecated)

JSON Schema .NET Assembly

Swagger Specification ASP.NET Core via API Explorer

Swagger Specification URL:

https://localhost:44354/swagger/v1/swagger.json Create local Copy

Swagger Specification JSON (if specified, the URL is ignored):

```
1 {
2   "x-generator": "NSwag v11.19.2.0 (NJsonSchema v9.10.)",
3   "swagger": "2.0",
4   "info": {
5     "title": "My Title",
6     "version": "1.0.0"
7   },
8   "host": "localhost:44354",
9   "schemes": [
10    "https"
11  ],
12  "consumes": [
13    "application/json"
14  ]
15 }
```

Generisanje koda pomoću NSwagStudio

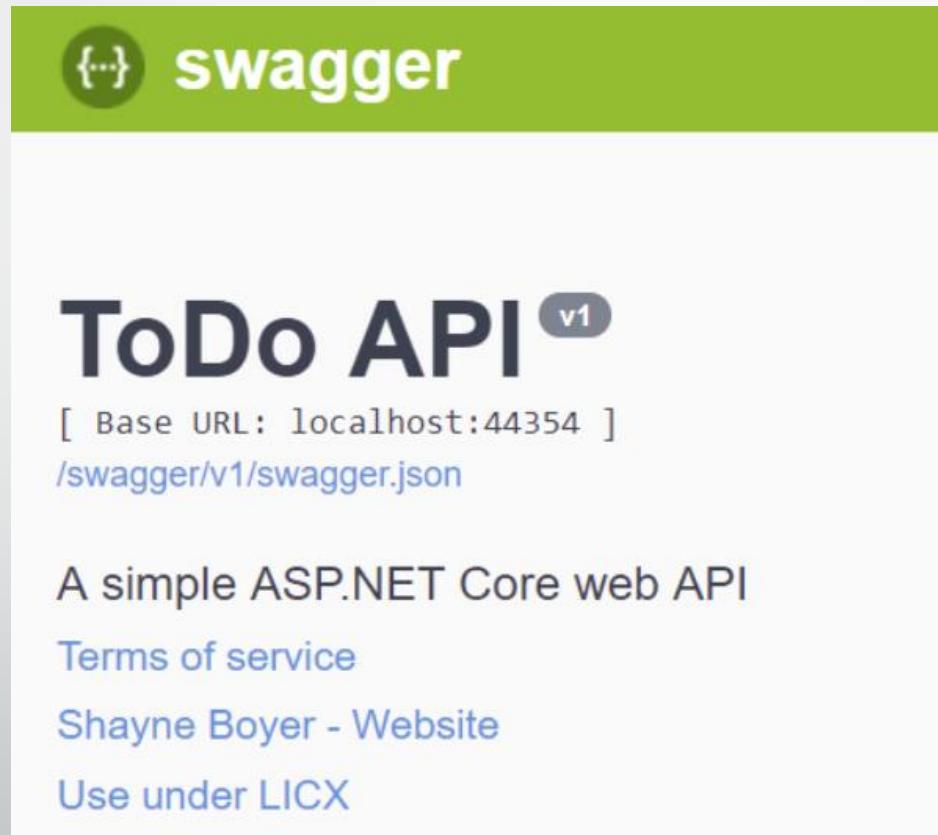
- U **Outputs** delu, čekirati **CSharp Client**.
- **Generate Outputs** generiše kompletan C# kod klijenta za dati projekat. Taj kod je dostupan u **CSharpClient** tabu


Prilagođavanje API dokumentacije

- Swagger omogućava dokumentovanje modela u cilju pojednostavljenja korišćenja webAPI-ja
- U **Startup.ConfigureServices** moguće je proslediti konfiguracijsku akciju metodi **AddSwaggerDocument** radi dodavanja informacija poput imena autora, licence i opisa.

```
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"
        };
    };
});
```

Prilagođavanje API dokumentacije





???