UDC 004.514

SERVICE STACK FOR CREATING WEB APPLICATIONS AND WEB SERVICES

YAUHENI LAPUNOU, YAUHENI SUKHAREU
Polotsk State University, Belarus

Most of the .Net Developers use standard solutions by Microsoft to solve task for the development of web services and web applications, as a rule it's MFC or Asp.Net MVC / Web Apl. However, modern dot Net has a lot of non-standard technologies, just about one of them we discuss in this article.

Service Stack [1] is a complete framework for creating web services and web applications. Creators of it excluded the worst, unnecessary and superfluous in Asp.Net and added a lot of new and useful functionality.

Service Stack appeared in the BBC in 2007 [2]. The company used the standard Enterprise solutions by Microsoft with Waterfall development methodologies. Due to the rapidly changing requirements of the functional BBC was face with a major problem. To make each small change was necessary to extend the cycle by UML to code, which took they until few days, use a variety of workarounds for compatibility formats between the old and new recordings that led to weekly downtime between releases. The team has had the question: "What is the service?" and identified basic principles of it:

- Service isn't necessarily has to use a predefined architecture which always determined before or follow a predetermined format.
- Service shouldn't impose a particular technology or a platform.
- Service shouldn't have pre-determined limit.
- Service has to encapsulate some possibilities for using it again, and it has to organized to be available.
- Access has to organize the most efficient and simple way, with the highest possible re-use and cost-effectively.

A list of these principles is what the definition of service. To this list we can add the fact that good service has to be ready for changes, and it needs to design so that the new changes won't be a problem.

A list of these principles led to the creature of Service Stack which let speeding up the development of several times. Now one half of the team can do more than the whole team did before.

ServiceStack [3] has more advantages than standard dot Net solutions:

- It's easier to use and intuitive.
- It works faster.
- Due to architecture of it, that allows to achieve a high level using of the code, even inexperienced developers.

We have to add in the last paragraph the fact that developers with great experience and high professionalism might to achieve code reuse and with using standard solutions by Microsoft. But it's necessary to understand that it takes the right approach and more time-consuming than using ServiceStack. Due to the flexible architecture and simplicity ServiceStack allows less-skilled developers to achieves better results and to brings a commitment to more effective approaches to the structuring code.

One of decisive arguments in using ServiceStack is that, that the projects can be deployed on Linux, without additional manipulation. This can significantly save on rent servers. You don't have to use the Windows Azure road or contain your Windows Server. Well-server usually approaches for medium-sized projects such as Debian or Ubuntu Server.

ServiceStack Asp.Net wasn't destroyed in developing and root IHttpHandler of it was leave. However, all the rest was put away and re-written with using lightweight alternatives which in some places were adapt specifically to meet the challenges of Web services development. Framework has its own session and authentication providers, a cache manager, the MQ-a broker who works on the basis of Redis. Also within the framework-Redis-own a client has been created that can be used either with or separately from the framework-and in different contexts. The standard IoS (Inversion of Control) containers used a slightly modified version of the popular and lightweight container Func. If desired, can be used instead of any other IoS- container.

The basis for ServiceStack are DTO (Data Transfer Objects, Data Transfer Objects), which are based on flat objects POCO. ISP authentication and sessions, as well as custom services based on the POCO, that allows you to use different backends for different implementations. Default available, for example, to store session typed directly into memory and Redis. Also available is an untyped and session speakers for the fans, but it is, unfortunately, a bit slower.

Consider creating a simple web service to ServiceStack. The first step in the project need to do three basic things:

1. Create a class for the DTO response service.

2. Create a class for the DTO request to the service.

3. Create a class of service, otnasledovav it from the base class, and overriding methods of the same name with the HTTP-methods that you need to implement.

First we need to create a project in Visual Studio or Xamarin Studio. Create an empty ASP.Net-project (Empty Web Application). The solution we call SSExampleProject, and the project - SSExampleAPI.

According to the project in the Solution Explorer (Solution Explorer) do right click, and select "Manage Packages NuGet" (Manage Nuget Packages).

After clicking on the Add button to connect our project downloaded dll ServiceStack'a. If after this step in the project is not Global.asax, you need to add it, and copy standard code ServiceStack site [3] (section Registering your web services and starting your application). There will also be missing web.config file, it must be filled with standard data ServiceStack site [3] (section Configuring Service Stack to run in your application).

After that you need to create ExampleService class, in which will be three classes:

– ExampleRequest - is responsible for the request to the service, and marked Route attribute. What reports for any inquiry that will meet the class. In this case it will be [Route ("/ hello")] and [Route ("/ hello / {Name}")]. Where Name is the string property class.

– ExampleResponse - is responsible for receiving a response from the service. It has one string property Result.

– ExampleService - This service itself is inherited from the base class has a method and Service Any, which takes ExampleRequest and returns the object. The method is a simple code that creates and assigns an instance ExampleResponse Result string query name.

Next, you need to compile and run the project in the browser's address bar to enter after the hostname and port "/ hello / ExapleString". ServiceStack Title query itself determines the type of customer, and in this case will reveal the answer service in a html-wrapper. And in the Result field will be displayed ExampleString line.

Any method name indicates that we have identified the same logic for all HTTP-methods.

Also pay attention to the metadata page that opens instantly when the project starts. It contains the automatically generated documentation to the services. ServiceStack creates different endpoints, and we can work with our service with both RESTful-service or as a SOAP-service.

Also ServiceStack is good because it created the service is very convenient to use Windows Phone, Xamarin.Android or Xamarin.iOS. There are different ways, but the easiest is to select individual files in the classes of requests and responses, and to connect them through the binding of a mobile application project. Also from Nuget set assembly ServiceStack.Client.

REFERENCES

1. Servicestack.net – официальный сайт ServiceStack [Электронный ресурс] / ServiceStack. – Режим доступа: https://servicestack.net/. – Дата доступа: 10.01.2016.
2. slideshare.net [Электронный ресурс] // Гитхаб. – Режим доступа: https://github.com/ServiceStack/ServiceStack/wiki. – Дата доступа: 10.01.2016.
3. GitHub.com [Электронный ресурс] // Сервис презентаций. – Режим доступа: http://www.slideshare.net/newmovie. – Дата доступа: 10.01.2016.