FREE eBook

LEARNING

nancy

Free unaffiliated eBook created from
Stack Overflow contributors.

#nancy

Table of Contents

About ... 1

Chapter 1: Getting started with nancy ... 2

- Remarks ... 2
- About: ... 2
- Additional Resources: ... 2
- Versions ... 2
- Examples ... 2
  - Create a simple self-hosted Nancy application ... 2
  - Setup Nancyfx with Dotnet core v1.1, Kestrel, and Visual Studio Code on *nix systems ... 3
- Prerequiste steps: ... 3
- Create self hosted NancyFx project: ... 3

Chapter 2: .net core support status ... 7

- Introduction ... 7
- Examples ... 7
  - Status ... 7

Nancy core ... 7

Nancy Boostrappers ... 7

Nancy View Engines ... 7

Nancy Authentication ... 7

Nancy FluentValidation ... 7

- Routing changes ... 7

Credits ... 9

About

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: nancy

It is an unofficial and free nancy ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official nancy.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Chapter 1: Getting started with nancy

Remarks

About:

Nancy is a lightweight framework for building HTTP based services in .Net based off of the Sinatra framework that exists for Ruby. It is designed to allow for handling of several different types of HTTP Requests and provides a simple way to return a response with a small amount of code.

Additional Resources:

- Main website: http://nancyfx.org
- Documentation: https://github.com/NancyFx/Nancy/wiki
- Source Code: https://github.com/NancyFx/Nancy
- Nuget Package: https://www.nuget.org/packages/Nancy/

Versions

Version	Release Date
Nancy 1.4.3	2015-12-21
Nancy 1.4.2	2015-11-23
Nancy 1.4.1	2015-11-05
Nancy 1.4.0	2015-10-29
Nancy 1.3.0	2015-09-25
Nancy 1.2.0	2015-04-17
Nancy 1.1.0	2015-02-18
Nancy 1.0.0	2015-01-23

Examples

Create a simple self-hosted Nancy application

1. Use Nuget to install the Nancy and Nancy.Hosting.Self packages into the project.
2. Instantiate a new NancyHost object and pass in the relevant URL

```
using( var host = new NancyHost( hostConfiguration, new Uri( "http://localhost:1234" ) ) )
{
    host.Start();
    Console.WriteLine( "Running on http://localhost:1234" );
    Console.ReadLine();
}
```

Place this code in your project at the point when you wish to start listening for http traffic.

3. Add a class to your project that inherits from NancyModule and add a constructor method.

```
public class FooModule : NancyModule
{
    public FooModule()
    {
    }
}
```

4. Define routes in the constructor:

```
...
public FooModule()
{
    Get["Bar"] = parameters => {
        return "You have reached the /bar route";
    }
}
```

Setup Nancyfx with Dotnet core v1.1, Kestrel, and Visual Studio Code on *nix systems

Prerequiste steps:

1. Get dotnet core for your platform:

   Dotnet Core

2. Follow instructions and make sure dotnet core is working
3. Get Visual Studio Code for your platform:

   VS Code

4. Launch Visual Studio Code (VS code) and install the C# extension then reload

Create self hosted NancyFx project:

1. Setup a project with a correct project directory structure.

   Open Bash Terminal and type:

```
mkdir nancydotnetcore
cd nancydotnetcore
mkdir src
mkdir test
touch global.json
```

2. Open global.json and enter the following code:

```
{
    "projects":["src", "test"]
}
```

3. In Bash terminal:

```
cd src
mkdir NancyProject1
dotnet new
```

Open folder NancyProject1 in VS code

You will get a warning: "Required assets to build and debug are missing from 'nancyproject1'."

Click "Yes"

Also you will see: There are unresolved dependencies from 'project.json'. Please execute the restore command to continue.

Click "Close" we will get to this soon.

4. Add the dependencies, open "project.json" and overwrite it with the following:

```
{
    "version": "1.0.0-*",
    "buildOptions": {
        "debugType": "portable",
        "emitEntryPoint": true
    },

    "frameworks": {
        "netcoreapp1.1": {
            "dependencies": {
                "Microsoft.AspNetCore.Hosting": "1.1.0",
                "Microsoft.AspNetCore.Server.Kestrel": "1.1.0",
                "Microsoft.AspNetCore.Owin": "1.1.0",
                "Nancy": "2.0.0-barneyrubble",
                "Microsoft.NETCore.App": {
                    "type": "platform",
                    "version": "1.1.0"
                }
            }
        }
    }
}
```

VS code will ask to restore click "Restore"

5. Create folder "Modules" in VSCode project

   In the Modules folder add a file named "IndexModule.cs" then copy and save the following:

```
namespace NancyProject1
{
    using Nancy;
    public class IndexModule : NancyModule
    {
        public IndexModule()
        {
            Get("/", _ => "Hello dotnet core world!");
        }
    }
}
```

6. In the root directory of the project create a file called "Startup.cs" and copy and paste the following:

```
namespace NancyProject1
{
    using Microsoft.AspNetCore.Builder;
    using Nancy.Owin;

    public class Startup
    {
        public void Configure(IApplicationBuilder app)
        {
            app.UseOwin(x => x.UseNancy());
        }
    }
}
```

7. Open file "Program.cs" and overwrite the content with the following and save:

```
namespace NancyProject1
{
    using System.IO;
    using Microsoft.AspNetCore.Builder;
    using Microsoft.AspNetCore.Hosting;

    public class Program
    {
        public static void Main(string[] args)
        {
            var host = new WebHostBuilder()
                .UseContentRoot(Directory.GetCurrentDirectory())
                .UseKestrel()
                .UseStartup<Startup>()
                .Build();

            host.Run();
        }
    }
}
```

8. Done! Now lets run this and see the output.

   Click the debug symbol in VS Code, and Click the run button. It should compile and start the project.

   Open the browser @ http://localhost:5000

9. Pat yourself on the back and enjoy!

Read Getting started with nancy online: https://riptutorial.com/nancy/topic/6049/getting-started-with-nancy

Chapter 2: .net core support status

Introduction

This topic shows the latest status of NancyFx compatibility with .Net Core. Community support is highly welcome to update this.

Examples

Status

> Currently there are numerous incompatibilities with .Net core. This will show status of NancyFx package status. This is grouped by functionality. Currently support is for Nancy 2.* which has not released a stable version yet.

Nancy core

Nancy : 2.0.0-clinteastwood

Nancy Boostrappers

Nancy.Bootstrappers.Ninject - Not Supported

Nancy View Engines

Nancy.ViewEngines.Razor - Not Supported issue #2521

Nancy Authentication

Nancy.Authentication.Stateless : 2.0.0-clinteastwood

Nancy FluentValidation

Note: Must use at least package ""FluentValidation.AspNetCore : 6.4.0-beta9

Nancy.Validation.FluentValidation : 2.0.0-clinteastwood

Routing changes

No Longer supported:

```
Get ["/"] = parameters => {
    return View["index"];
};
```

Changed to:

```
Get("/", parameters => {
    return View["index"];
});
```

Read .net core support status online: https://riptutorial.com/nancy/topic/8591/-net-core-support-status

Credits

S. No	Chapters	Contributors
1	Getting started with nancy	4444, Allistaire Clark, Community, Fritz Seitz, Komeisa
2	.net core support status	Fritz Seitz