NancyFX

Nancy is a framework for building HTTP based services on .Net and [Mono](http://mono-project.com/). Based on the Ruby library Sinatra

The goal of the framework is to stay out of the way as much as possible.

This means that everything in Nancy is setup to have sensible defaults and conventions, instead of making you jump through hoops and go through configuration hell just to get up and running. With Nancy you can go from zero to website in a matter of minutes. Literally.

Built by the community, as an open-source framework, meaning you get full access to the source code, and is licensed under the [MIT license](http://www.opensource.org/licenses/mit-license.php). You can get Nancy from [Nuget](http://nuget.org/), [our TeamCity server](http://teamcity.codebetter.com/project.html?projectId=project112&tab=projectOverview&guest=true) (for latest builds) or download the source from our [GitHub repositories](http://nancyfx.org/).

One of the core concepts in Nancy is hosts. A host acts as an adaptor for a hosting environment and Nancy, thus enabling Nancy to run on existing technologies such as ASP.NET, WCF and OWIN, or integrated in any given application. Specific host implementations are not shipped with the core Nancy framework. They are shipped separately, as are many other additional functionalities such as forms authentication. It’s like a web framework buffet.

SDHP, just works: pick things up with no nonsense. New modules are autodetected, new dependency, auto injected

Everything can be customized, new ioc container, change how routes are picked, no problem

the amount of “Nancy code” you should need in your application should be minimal. As a testament to this it’s actually possible to fit a functional Nancy application into a single Tweet

Naming should be obvious, required configuration should be minimal

A few conventions, <list> and Nancy is designed to handle DELETE, GET, HEAD, OPTIONS, POST, PUT and PATCH requests

Modules x4

Routing x8

Route Collisions

Route Scoring

Conditional Routing

Route Constraints x3

Actions ConNeg

Actions Response Object

Actions Implicit casts

Actions Views

Super Simple View Engine x3 call out HasCollection

Super Simple View Engine x4

Model Binding x2

Bootstrapper

Auth x3

Pipelines x3

Hosting x4