

Frictionless Development Framework

ME

※ Write software **※**

- Trying to learn and improve
- E-Mail: siliconrob@siliconheaven.net 🏴
- Blog: https://siliconheaven.info/ d□□□
- GitHub: https://github.com/Siliconrob
- I have worked with ServiceStack in production

What is it?

- Written by Demis Bellot dev@servicestack.net
- .NET alternative Web Framework+Ecosystem
- 32,000,000+ downloads
- Commercial license with AGPL FOSS license exception
- Service clients are unrestricted
- All source code is available on GitHub

Support and Resources

- Documentation: It is great
- Customer Forums New questions/comments restricted to registered customers
- User Voice: Feature Requests
- StackOverflow
- Release Notes: Comprehensive and detailed

Key points

- Message based Architecture based on explicit Data Transfer Objects (DTO)
- Runs anywhere
- Works with your choice of language+framework
 - Polyglots welcome!
- Documentation
- Mature and supported: 10+ years of development history
- Growing year over year and Demis is always on top of the hard code paths first for you

Inspiration

- Be productive
- Value composition over inheritance
- Work with Web standards
- Consistency
- Pipelines
- Smalltalk: Message dispatching

Guiding Principles

- Simplicity: 11 + 11 = 2
- Performance as feature 🚀
- Lightweight: code first focus đ□□□d′,□
- Testability: Unit, Integration, Application d□§§
- Message based architecture: Coming back to this later 😂

Current Version

- v5.6 released 8/6/2019
- Prereleases available on MyGet

Release Notes

https://docs.servicestack.net/release-notes-history

Messaging

• Every input request DTO has a corresponding response DTO

Request/Response

- Any DTO object -> serialized to Response ContentType
- HttpResult, HttpError, CompressedResult (IHttpResult) for Customized HTTP response

Messaging Example

Core Components

- ServiceStack.Text: Free from restrictions on use
- ServiceStack.OrmLite: Dapper++
- ServiceStack.Redis
- ServiceStack.Caching
- And many, many more are available

Helpful Tips

- Assemblies are units of deployment NOT logical boundaries
- Keep your model assembly small and dependency free
- Message DTOs should only contain serializable fields NOT behavior
- Extension methods are your friends
- Pipelining behavior in a Fluent functional manner is good
- Remember that HTTP is underlying

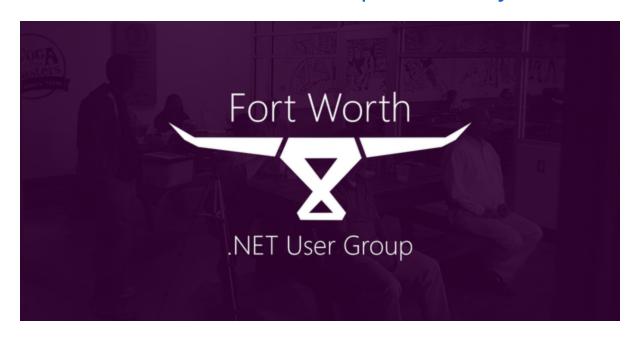
Routing

Explicit

- 1. Any exact Literal Matches are used first
- 2. Exact Verb match is preferred over All Verbs
- 3. The more variables in your route the less weighting it has
- 4. Routes with wildcard variables have the lowest precedence
- 5. When Routes have the same weight, the order is determined by the position of the Action in the service or Order of Registration (FIFO)

Thank you

Fort Worth .NET User Group and TekSystems



Presentation