3.– 6. September 2012
in Nürnberg

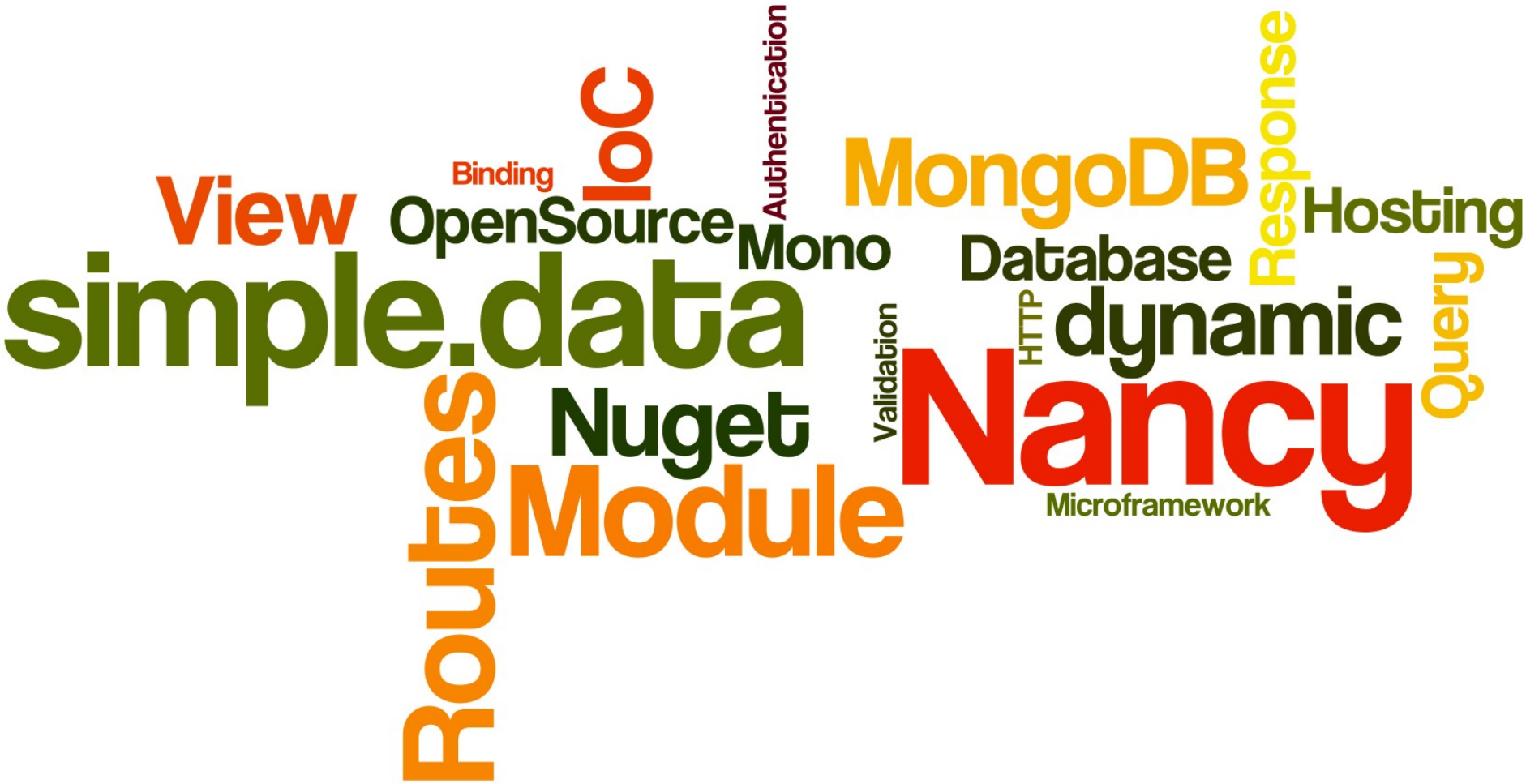
Wissenstransfer
par excellence

EINFACH SIMPEL

Granuläres .NET-Web-Development mit Nancy- und Simple.Data- Frameworks

Timothée Bourguignon

Mathema Software GmbH


Authentication
Response
IoC
Binding
MongoDB
View
OpenSource
Hosting
Mono
Database
simple.data
Query
HTTP
dynamic
Validation
Nancy
Nuget
Routes
Module
Microframework

Why?

- Simplicity & Readability
- Modularity
- OpenSource

- "Close" to HTTP
- Very explicit routing
- Runs anywhere

- „Super Duper Happy Path“

First date with Nancy

„Lightweight Web Framework for .NET“

nancyfx.org | #nancyfx

Microframework

- Serve up web content

- Lean
- Extensible API
- Simple setup

- “Close to the metal”

„Hello Sinatra“

```
$ gem install sinatra
$ ruby –rubygems app.rb

require 'sinatra'
get '/hi' do
  "Hello World!"
end
```

Sinatra

„Hello Nancy“

```
namespace NancyDemo
{
  public class MainModule : NancyModule
  {
    public MainModule()
    {
      Get["/hi"] = _ => "Hello Nancy!";
    }
  }
}
```

Setting up a Nancy project

- Create an empty web project
- Install Nancy via Nuget:

```
 PM>  Install-package Nancy.Hosting.Aspnet
Or        Install-Package Nancy.Hosting.Owin
Or        Install-Package Nancy.Hosting.Wcf
…
```

  - Adds Nancy assembly references
  - Sets up web.config (if using IIS)

- Create a module and define a route

Architectural Overview

- HTTP handler
- Modules
- Routes
- Response
- Views
- View engines

- Bootstrapping (IoC)
- Hosting
  - ASP.NET
  - WCF
  - Azure
  - Umbraco
  - Owin (~ Ruby's "Rack")
  - Self-hosted

- Authentication
- …

NANCY

Module‘s Anatomy

```
public class MyModule: NancyModule
{
    Public MyModule()
    {
      Get["/"] = _ => { ... };
      Get["/submit"] = _ => { ... };
      Post["/submit"] = _ => { ... };
      Get["/voteup/{id}"] = parameters => { ... };
      Get["/login"] = _ => { ... };
      Get["/logout"] = _ => { ... };
      Post["/login"] = _ => { ... };
    }
}
```

NANCY

Route

- Composed of
  - Method (HTTP Methods: Get, Post, Put, Delete…)
  - Pattern
  - Action (+parameters & result object)
  - Condition (optional)

```
Get["/voteup/{id}"] = x => {
  return View["Voteup", x.id];
};
```

Route pattern

- Literal segments: `"/voteup"`
- Variable segments: `"/voteup/{id}“`
- Literal segments + regular expression back-references: `"/voteup/(?<id>[\d]+)"`

```
Get["/voteup/{id}"] = x => {
  return View["Voteup", x.id];
};
```

NANCY

Route resolver

- Request method & Path matching
- Picks the first route among those which captures the least parameters

Lifecycle

- "Before" pipeline is executed (if present)
- Route's action is invoked
- "After" pipeline is executed (if present)

```
Get["/voteup/{id}"] = x => {
  return View["Voteup", x.id];
};
```

NANCY

Route action, parameters & Response

- Receives a .Net4.0 *"**dynamic**"* object packaging the request parameters

```
x => x.id
```

- Returns an *"**Nancy.Response**"* object

```
Get["/voteup/{id}"] = x => {
  return View["Voteup", x.id];
};
```

Nancy.Response

- *Nancy.Response* implicit casts
  - Int32 ➔ HTTP Status Code (ex: 404, No Found)
  - 'HttpStatusCode' enumerable value
  - String ➔ body of the response
  - 'Action<Stream>'
- Response formatters:
  - As File, Image, Json, Xml & Redirect
- Views

```
Get["/voteup/{id}"] = x => {
  return View["Voteup", x.id];
};
```

NANCY

Serving up views

- Views are discovered in the project
- Supported View Engines
  - SuperSimpleViewEngine (basic HTML & iteration syntax)
  - Razor
  - Spark
  - Django
  - DotLiquid
  - Any other implementation IViewEngine
  - Selected dynamically, based on the view's file extension

NANCY

View model & Model Binding

- Data passed via Query string, captured parameters on routes or body of a request

- Module ➔ View
  - Supports any object Type
  - Uses “dynamic” per default

- Module ⬅ View
  - Model binding for all ways

```
Foo foo = this.Bind();
var foo = this.Bind<Foo>();
this.BindTo(foo);
```

Model Validation

- Similar to MVC

```
using System.ComponentModel.DataAnnotations;


[Required]
public DateTime CreationDate { get; set; }


var result = this.Validate(model);
if (!result.IsValid)
{
  return View["CustomerError", result];
}
```

Demo: hands on Nancy

HackerNews meet Nancy

Simple.Data

A „/“, e.g. an O/RM without O, R or M

Simple.Data

- Inspired by Ruby's DataMapper & ActiveRecord
- Based on the .NET 4.0 "dynamic" keyword

- No SQL injection

- Easy
- Intuitive
- Flexible

- Database agnostic
- Convention over Configuration
- Not an O/RM

Hello Simple.Data

```
public void Greetings()
{
  var db = Simple.Data.Database.Open();

  var hello = db.Hello.FindById(1);

  Console.WriteLine("{0}, {1}!",
    hello.Greeting, hello.Subject);
}
```

Setting up a Simple.Data project

- Install the driver you need via Nuget

```
PM> Install-Package Simple.Data.MongoDB
```

- Create a “dynamic” Database object

Simple CRUD operations

```
public void CRUD()
{
  db.People.FindAllByName("Bob");
  db.People.FindByFirstNameAndLastName("Bob", "X");
  db.Users.All().OrderByJoinDateDescending();
  db.Users.All().OrderByJoinDate().ThenByNickname();

  db.People.Insert(Id: 1, FirstName: "Bob");
  db.People.Insert(new Person(){ Id = 1, Name = "Bob" } );

  db.People.UpdateById(Id: 1, FirstName: "Robert");
  db.People.DeleteById(1);
}
```

Barely less simple operations

- Complex criteria

```
db.Customers.Find(db.Customers.MoneyOwing > 0);
```

- Paging

```
db.Users.All().OrderByNickname().Skip(10).Take(10);
```

- Having

```
db.Posts.All().Having(db.Posts.Comments.CommentId
                                .Count() == 0);
```

- Upsert (Update or Insert)

```
db.Users.Upsert(user);
```

Complex operations

- Eager loading with the “with” operator

```
db.Customers.WithOrders().Get(1);
db.Customers.FindAllByCustomerId(1).WithOrders();
```

- Implicit / Explicit Join, OuterJoin…

Drivers & Supported DBs

- ADO-based access to relational databases:
  - SQL Server 2005 and later (including SQL Azure)
  - SQL Server Compact Edition 4.0
  - Oracle
  - MySQL 4.0 and later
  - SQLite
  - PostgreSQL
  - SQLAnywhere
  - Informix
- MongoDB
- OData

Demo: Nancy, meet Simple.Data

Simple.Data, meet Nancy

Second date with Nancy

In case the SuperDuperHappyPath is not completely Super, Duper or Happy yet…

Nancy's Bootstrapping (IoC)

ses IoC containers to bootstrap the framework

iscover and loady any implementation from INancyBootstrapper

he DefaultNancyBootstrapper uses TinyIOC

ancy also includes boostrappers for

- Ninject
- StructureMap

Authentication

- Nancy.Authentication.Basic
  - Basic HTTP authentication
- Nancy.Authentication.Forms
  - Proper authentication
  - Facebook, oAuth etc. on their way

Authentication

```
public class MyBootstrapper : DefaultNancyBootstrapper
{
  protected override void InitialiseInternal(TinyIoC.TinyIoCContainer
  container)
  {
    base.InitialiseInternal(container);
    FormsAuthentication.Enable(this,
      new FormsAuthenticationConfiguration
      {
        RedirectUrl = "~/login",
        UsernameMapper = container.Resolve<IUsernameMapper>()
      });}}

public class MyModule : NancyModule
{
    public MyModule() : base("/secure")
    {
        this.RequiresAuthentication();
        Get["/"] = _ => "Secure!";
    }
}
```

Testing

- Nuget: `PM> Install-Package Nancy.Testing`
- „Browser“ class

```
var result = browser.Get("/", with => {
  with.HttpRequest(); });
```

- Assert

```
response.Body["#errorBox"]
  .ShouldExistOnce()
  .And.ShouldBeOfClass("floatingError")
  .And.ShouldContain("invalid",
    StringComparison.InvariantCultureIgnoreCase);
```

Nancy.Diagnostic

localhost/_nancy

- Information
- Interactive diagnostic
- Request Tracing
- Settings

NANCY

lancy	Version	v0.11.0.0
	Caches Disabled	true
	Traces Disabled	false
	Case Sensitivity	Insensitive
	Root Path	G:\WORKSPACE\.NetHN\.NetHN\
	Hosting	Aspnet (v0.11.0.0)
	Bootstrapper Container	TinyIoC
	Located Bootstrapper	NetHN.FormsAuthBootstrapper
	Loaded View Engines	SuperSimple
		Razor
Configuration	Route Resolver	Nancy.Routing.DefaultRouteResolver
	Route Pattern Matcher	Nancy.Routing.DefaultRoutePatternMatcher
	Context Factory	Nancy.DefaultNancyContextFactory
	Nancy Engine	Nancy.NancyEngine
	Module Key Generator	Nancy.Bootstrapper.DefaultModuleKeyGenerator
	Route Cache	Nancy.Routing.RouteCache
	Route Cache Provider	Nancy.Routing.DefaultRouteCacheProvider
	View Locator	Nancy.ViewEngines.DefaultViewLocator
	View Factory	Nancy.ViewEngines.DefaultViewFactory
	Nancy Module Builder	Nancy.Routing.DefaultNancyModuleBuilder
	Response Formatter Factory	Nancy.DefaultResponseFormatterFactory
	Model Binder Locator	Nancy.ModelBinding.DefaultModelBinderLocator
	Binder	Nancy.ModelBinding.DefaultBinder
	Binding Defaults	Nancy.ModelBinding.BindingDefaults
	Field Name Converter	Nancy.ModelBinding.DefaultFieldNameConverter
	Model Validator Locator	Nancy.Validation.DefaultValidatorLocator
	View Resolver	Nancy.ViewEngines.DefaultViewResolver
	View Cache	Nancy.ViewEngines.DefaultViewCache
	Render Context Factory	Nancy.ViewEngines.DefaultRenderContextFactory
	View Location Cache	Nancy.ViewEngines.DefaultViewLocationCache
	View Location Provider	Nancy.ViewEngines.FileSystemViewLocationProvider
	Error Handlers	Nancy.ErrorHandling.DefaultErrorHandler
	Csrf Token Validator	Nancy.Security.DefaultCsrfTokenValidator
	Object Serializer	Nancy.DefaultObjectSerializer
	Serializers	Nancy.Responses.DefaultJsonSerializer

My opinion

- Pros
  - It is incredibly simple, readable & flexible
  - It runs everywhere (Self hosted on Mobile?)
  - Great for Webservice or small / fast projects

- Cons
  - Hard to picture a large project with Nancy or Simple.Data
  - Flexibility via „*dynamic*“ can be double edged

  - Nancy & Simple.Data cannot do more or less than the others

  … but they do it in a very elegant and efficient manner

Links & Contacts

- **Andreas Håkansson (NancyFx)**
  - @TheCodeJunkie
  - http://elegantcode.com/

- **Steven Robbins** (NancyFx, TinyIoC)
  - @Grumpydev
  - http://www.grumpydev.com/

- **Mark Rendle** (Simple.Data)
  - @MarkRendle
  - http://blog.markrendle.net/

Vielen Dank!

Tim Bourguignon

about.me/timbourguignon

Additional sources

- „Simple.Data, .NET Database access made easier“, Mark Rendle, http://www.slideshare.net/markrendle/simple-data
- Nancy (.NET Micro Web Frameworks), Nicholas Cloud, http://www.nicholascloud.com/2011/05/nancy-net-micro-web-frameworks-p

KayakTest ▸ MainModule ▸ MainModule

```
public MainModule()
{
    Get["/hi"] = _ => "Hello World!";
}
```

Call Stack

Name	Lang
KayakTest.exe!KayakTest.MainModule..ctor.AnonymousMethod__0(dynamic _) Line 9	C#
Nancy.dll!Nancy.Routing.Route.Invoke(Nancy.DynamicDictionary parameters) + 0x34 bytes	
Nancy.dll!Nancy.NancyEngine.ResolveAndInvokeRoute(Nancy.NancyContext context) + 0xc3 bytes	
Nancy.dll!Nancy.NancyEngine.InvokeRequestLifeCycle(Nancy.NancyContext context) + 0x46 bytes	
Nancy.dll!Nancy.NancyEngine.HandleRequest(Nancy.Request request) + 0x96 bytes	
Nancy.dll!Nancy.NancyEngine.HandleRequest.AnonymousMethod__0(object s) + 0x50 bytes	
mscorlib.dll!System.Threading.QueueUserWorkItemCallback.WaitCallback_Context(object state) + 0x2d bytes	
mscorlib.dll!System.Threading.ExecutionContext.Run(System.Threading.ExecutionContext executionContext, System.Threading.ContextCallback callback, object state, bool ignoreSyncCtx) + 0xb0 bytes	
mscorlib.dll!System.Threading.QueueUserWorkItemCallback.System.Threading.IThreadPoolWorkItem.ExecuteWorkItem() + 0x5a bytes	
mscorlib.dll!System.Threading.ThreadPoolWorkQueue.Dispatch() + 0x147 bytes	
mscorlib.dll!System.Threading._ThreadPoolWaitCallback.PerformWaitCallback() + 0x2d bytes	
[Native to Managed Transition]	

MvcTest.Controllers ▸ HomeController ▸ Index

```
public ActionResult Index()
{
    return new ContentResult { Content = "Hello World!" };
}
```

Call Stack

Name	Lang
MvcTest.DLL!MvcTest.Controllers.HomeController.Index() Line 16	C#
[Lightweight Function]	
System.Web.Mvc.dll!System.Web.Mvc.ActionMethodDispatcher.Execute(System.Web.Mvc.ControllerBase controller, object[] parameters) + 0x12 bytes	
System.Web.Mvc.dll!System.Web.Mvc.ReflectedActionDescriptor.Execute(System.Web.Mvc.ControllerContext controllerContext, System.Collections.Generic.IDictionary<string,object> parameters) + 0x	
System.Web.Mvc.dll!System.Web.Mvc.ControllerActionInvoker.InvokeActionMethod(System.Web.Mvc.ControllerContext controllerContext, System.Web.Mvc.ActionDescriptor actionDescriptor, System	
System.Web.Mvc.dll!System.Web.Mvc.ControllerActionInvoker.InvokeActionMethodWithFilters.AnonymousMethod__12() + 0x38 bytes	
System.Web.Mvc.dll!System.Web.Mvc.ControllerActionInvoker.InvokeActionMethodFilter(System.Web.Mvc.IActionFilter filter, System.Web.Mvc.ActionExecutingContext preContext, System.Func<Syste	
System.Web.Mvc.dll!System.Web.Mvc.ControllerActionInvoker.InvokeActionMethodWithFilters.AnonymousMethod__14() + 0x14 bytes	
System.Web.Mvc.dll!System.Web.Mvc.ControllerActionInvoker.InvokeActionMethodWithFilters(System.Web.Mvc.ControllerContext controllerContext, System.Collections.Generic.IList<System.Web.Mv	
System.Web.Mvc.dll!System.Web.Mvc.ControllerActionInvoker.InvokeAction(System.Web.Mvc.ControllerContext controllerContext, string actionName) + 0xed bytes	
System.Web.Mvc.dll!System.Web.Mvc.Controller.ExecuteCore() + 0x75 bytes	
System.Web.Mvc.dll!System.Web.Mvc.ControllerBase.Execute(System.Web.Routing.RequestContext requestContext) + 0x62 bytes	
System.Web.Mvc.dll!System.Web.Mvc.ControllerBase.System.Web.Mvc.IController.Execute(System.Web.Routing.RequestContext requestContext) + 0xb bytes	
System.Web.Mvc.dll!System.Web.Mvc.MvcHandler.BeginProcessRequest.AnonymousMethod__5() + 0x26 bytes	
System.Web.Mvc.dll!System.Web.Mvc.Async.AsyncResultWrapper.MakeVoidDelegate.AnonymousMethod__0() + 0x16 bytes	
System.Web.Mvc.dll!System.Web.Mvc.Async.AsyncResultWrapper.BeginSynchronous<System.Web.Mvc.Async.AsyncVoid>.AnonymousMethod__7(System.IAsyncResult _) + 0xd bytes	
System.Web.Mvc.dll!System.Web.Mvc.Async.AsyncResultWrapper.WrappedAsyncResult<System.Web.Mvc.Async.AsyncVoid>.End() + 0x3f bytes	
System.Web.Mvc.dll!System.Web.Mvc.MvcHandler.EndProcessRequest.AnonymousMethod__d() + 0x33 bytes	
System.Web.Mvc.dll!System.Web.Mvc.SecurityUtil.GetCallInAppTrustThunk.AnonymousMethod__0(System.Action f) + 0x8 bytes	
System.Web.Mvc.dll!System.Web.Mvc.SecurityUtil.ProcessInApplicationTrust(System.Action action) + 0x17 bytes	
System.Web.Mvc.dll!System.Web.Mvc.MvcHandler.EndProcessRequest(System.IAsyncResult asyncResult) + 0x3d bytes	
System.Web.Mvc.dll!System.Web.Mvc.MvcHandler.System.Web.IHttpAsyncHandler.EndProcessRequest(System.IAsyncResult result) + 0xa bytes	
System.Web.dll!System.Web.HttpApplication.CallHandlerExecutionStep.System.Web.HttpApplication.IExecutionStep.Execute() + 0x1b0 bytes	
System.Web.dll!System.Web.HttpApplication.ExecuteStep(System.Web.HttpApplication.IExecutionStep step, ref bool completedSynchronously) + 0xb9 bytes	
System.Web.dll!System.Web.HttpApplication.ApplicationStepManager.ResumeSteps(System.Exception error) + 0x13e bytes	
System.Web.dll!System.Web.HttpApplication.System.Web.IHttpAsyncHandler.BeginProcessRequest(System.Web.HttpContext context, System.AsyncCallback cb, object extraData) + 0xf8 bytes	
System.Web.dll!System.Web.HttpRuntime.ProcessRequestInternal(System.Web.HttpWorkerRequest wr) + 0x1a2 bytes	
System.Web.dll!System.Web.HttpRuntime.ProcessRequestNoDemand(System.Web.HttpWorkerRequest wr) + 0x7d bytes	
System.Web.dll!System.Web.HttpRuntime.ProcessRequest(System.Web.HttpWorkerRequest wr) + 0x47 bytes	
WebDev.WebHost40.dll!Microsoft.VisualStudio.WebHost.Request.Process() + 0x17b bytes	
WebDev.WebHost40.dll!Microsoft.VisualStudio.WebHost.Host.ProcessRequest(Microsoft.VisualStudio.WebHost.Connection conn) + 0x6c bytes	
[Appdomain Transition]	
WebDev.WebHost40.dll!Microsoft.VisualStudio.WebHost.Server.OnSocketAccept(object acceptedSocket) + 0x83 bytes	
mscorlib.dll!System.Threading.QueueUserWorkItemCallback.WaitCallback_Context(object state) + 0x2d bytes	
mscorlib.dll!System.Threading.ExecutionContext.Run(System.Threading.ExecutionContext executionContext, System.Threading.ContextCallback callback, object state, bool ignoreSyncCtx) + 0xb0 bytes	
mscorlib.dll!System.Threading.QueueUserWorkItemCallback.System.Threading.IThreadPoolWorkItem.ExecuteWorkItem() + 0x5a bytes	
mscorlib.dll!System.Threading.ThreadPoolWorkQueue.Dispatch() + 0x147 bytes	
mscorlib.dll!System.Threading._ThreadPoolWaitCallback.PerformWaitCallback() + 0x2d bytes	
[Native to Managed Transition]	