Extending the NancyModules

Richard Cirerol
@richardcirerol

pluralsight
hardcore developer training

Extending the NancyModule

Receive Request

Process Request

Return Response

NANCY

Extending the NancyModule

Process Request

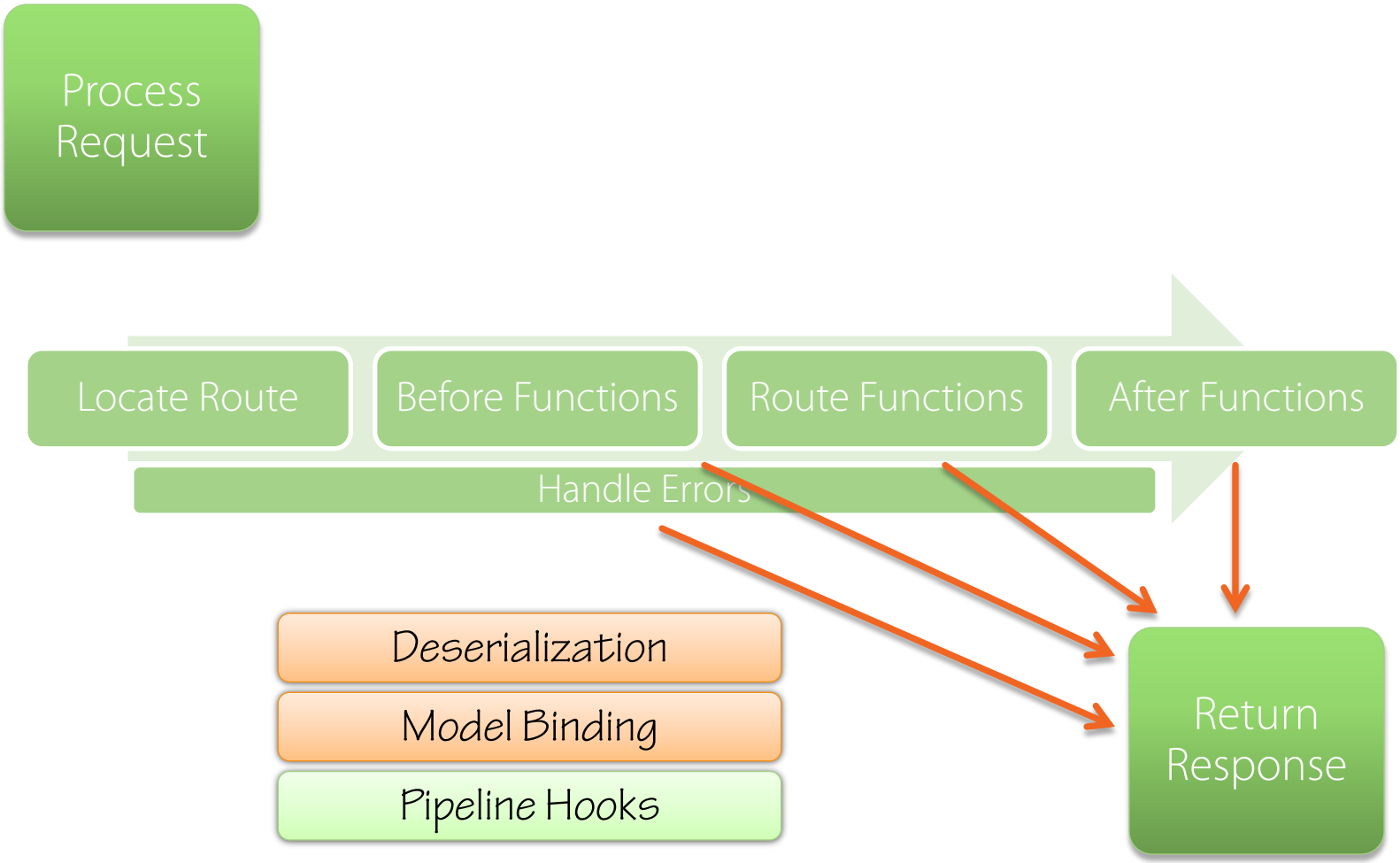
Receive Request

Process Request

Return Response

NANCY

Extending the NancyModule



NANCY

Working with Form data

```
var form = this.Request.Form
```

```
POST /names HTTP/1.1
Host: localhost
Content-Type: application/x-www-url-encoded

name=Getting%20Started%20with%20Nancy&author=Ric
hard%20Cirerol
```

As of version 0.10.0.0, Nancy will only parse form data sent using the application/x-www-url-encoded mime-type.

Working with Complex data

```
public class Course
{
    public int Id{ get; set; }
    public string Name { get; set; }
    public string Author { get; set; }
}
```

Working with Complex data

```
public class Course
{
    public int Id{ get; set; }
    public string Name { get; set; }
    public string Author { get; set; }
    public string Module[] { get; set; }
}

public class Module
{
    public int Id{ get; set; }
    public string Topic { get; set; }
}
```

NANCY

Working with Complex Types

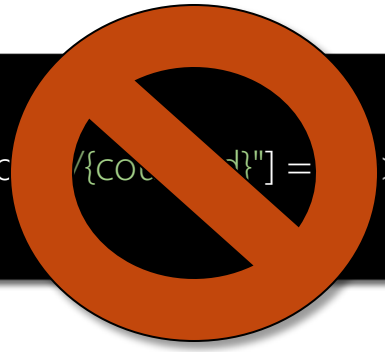
- **Built-In Deserializers**
  - JSON
    - application/json
    - text/json
    - application/vnd…+json
  - XML
    - application/xml
    - text/xml
    - application/vnd…+xml
- **Other**
  - Protocol Buffers
  - CSV
  - Domain-specific

NANCY

NancyModule: Model Binding

```
var model = this.Bind<TModel>() //returns typed object
TModel model = this.Bind()   //returns dynamic object mapped to a type
```

```
Po      /{cou      d}"] =      > {...}
```



NANCY

Working with Complex data

```
Course course = this.Bind();
```

```
var course = this.Bind<Course>();
```

```
POST /courses HTTP/1.1
Host: localhost
Content-Type: application/json

{
  "name"    : "Getting Started with Nancy",
  "author"  : "Richard Cirerol",
  "modules" : [{ "topic" : "Introduction" }]
}
```

Working with Complex data

```
Course course = this.Bind();
```

```
var course = this.Bind<Course>();
```

```
POST /courses HTTP/1.1
Host: localhost
Content-Type: applicati

<course author=
  <name>Gett
  <modul
    <
  <
</co
```

Wait! We need to prepare first!

Working with Complex data

```
[XmlRoot("course")]
public class Course
{
    [XmlAttribute]
    public int Id{ get; set; }
    [XmlAttribute]
    public string Author{ get; set; }
    public string Title { get; set; }
    [XmlArray("modules")]
    public Module Modules[] { get; set; }
}

[XmlRoot("module")]
public class Module
{
    [XmlAttribute]
    public int Id{ get; set; }
    public string Topic { get; set; }
}
```

Working with Complex data

```
Course course= this.Bind();
```

```
var course = this.Bind<Course>();
```

```
POST /couress HTTP/1.1
Host: localhost
Content-Type: xml

<course author='Richard Cirerol'>
  <name>Getting Started with Nancy</name>
  <modules>
    <module>Introduction</module>
  </modules>
</course>
```

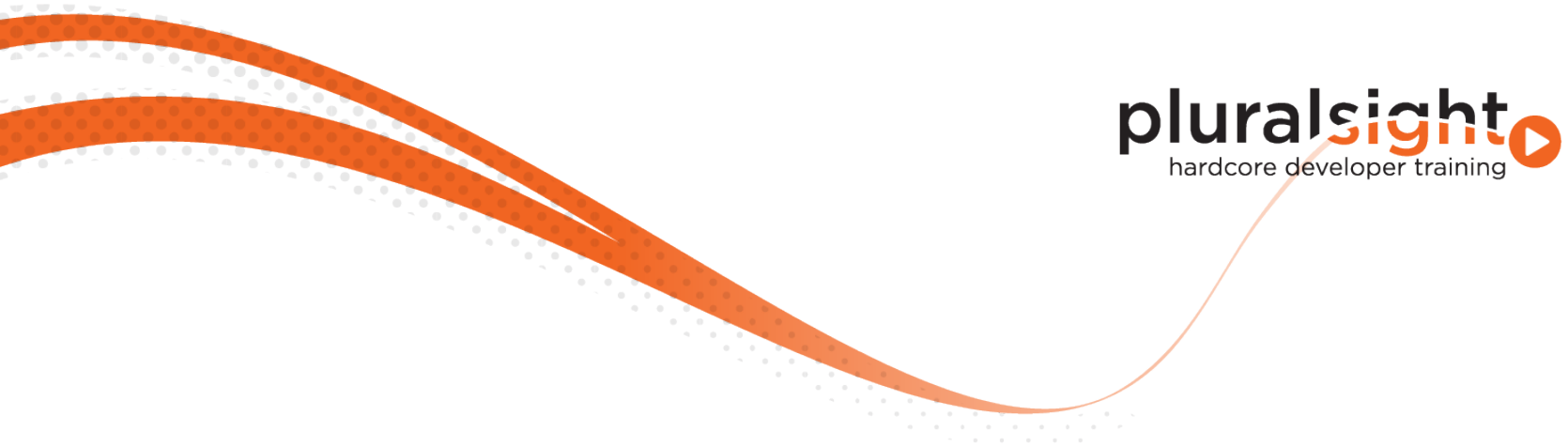
Demo: Working with Complex Types

Requirements:

- **Allow posting via JSON**
  - Update the post route to handle complex model binding
  - Use cUrl to send a POST request with JSON data
- **Allow posting via XML**
  - Annotate the Course class
  - User cUrl to send a POST request with XML data

Pipeline Hooks

Before and After



pluralsight
hardcore developer training

Pipeline Hooks: BeforePipeline

```
Before += (Func<NancyContext, Response> ctx) => {
    // returning non-null response exits pipeline and returns response message
};
```

Adds function to the end of the *BeforePipeline*

Ability to add to the beginning or middle of the *BeforePipeline*

Any non-null response will cause the response to be returned

This construct allows us to:

- Alter the context of the request
- Do some work in the context of the request
- Return an alternate response

Pipeline Hooks: BeforePipeline

```
Before += (Func<NancyContext, Response> ctx) => {
    // returning non-null response exits pipeline and returns response message
};
```

Potential uses:

- Redirect to new URI – 301 Permanent Redirect
- Check the incoming request cache headers – 304 Not Modified
- Module-level authorization - 403 Forbidden
- Ensure valid requests – 400 Bad Request
- Analytics – API Usage
- Adding data to the context

Pipeline Hooks: AfterPipeline

```
After += (Action<NancyContext> ctx) =>{
   // alter the response before returning response message
};
```

Adds function to the end of the *AfterPipeline*

Ability to add to the beginning or middle of the *AfterPipeline*

This construct allows us to:

- Do some work in the context of the request
- Alter the response (add or remove headers)
- Return an alternate response

Pipeline Hooks: AfterPipeline

```
After += (Action<NancyContext> ctx) =>{
   // alter the response before returning response message
};
```

Potential uses:

- Adding generated Response to server Cache
- Create and Add ETag/Last-Modified headers to Response
- Compress Response message
- Analytics – API Usage

Demo: Adding Pipeline Hooks

Requirements:

- **Create a BeforePipeline hook to hide course api requests from web browsers**
- **Create an after hook to log the time taken to process the request**
- **Ensure these hooks only apply to the course requests**

Summary

Binding to message body

- **this.Form**
- **this.Bind()**

Use the BeforePipeline and AfterPipeline

- **Modify the incoming context**
- **Return a modified response**
- **Analytics**