

C# Extension Methods

Module 5: Extension Method Library (part 2)

Elton Stoneman
geekswithblogs.net/eltonstoneman
elton@sixeyed.com



pluralsight
hardcore developer training

Extension Method Library

- **Part 1 – internals**
 - Core: exceptions, enums
 - Reflection & expressions
 - Entity Framework: easy auditing
- **Part 2 – externals**
 - WCF: safely closing clients
 - WebApi: location headers
 - MVC: HTML helpers

Extension Method Library

- **Part 1 – internals**
 - Core: exceptions, enums
 - Reflection & expressions
 - Entity Framework: easy auditing

- **Part 2 – externals**
 - WCF: safely closing clients
 - WebApi: location headers
 - MVC: HTML helpers



WCF Clients

■ Communication channels

- Closing clients as soon as possible
- ClientBase<T> implements IDisposable

```
using (var client = new Service1Client())
{
    response = client.GetDataUsingDataContract(request);
}
```

- With error handling

```
using (var client = new Service1Client())
{
    try {
        response = client.GetDataUsingDataContract(request);
    }
    catch (Exception ex) {
        Debug.WriteLine(ex.FullMessage());
    }
}
```

Demo 4: WCF Clients

Feature

Ensure WCF
service calls run
and close safely

Task

Extend client to
dispose without
throwing
exceptions

Demo 4: WCF Clients

Demo 4: WCF Clients

■ ICommunicationObjectExtensions

- In custom framework namespace
- Added *DisposeSafely()*

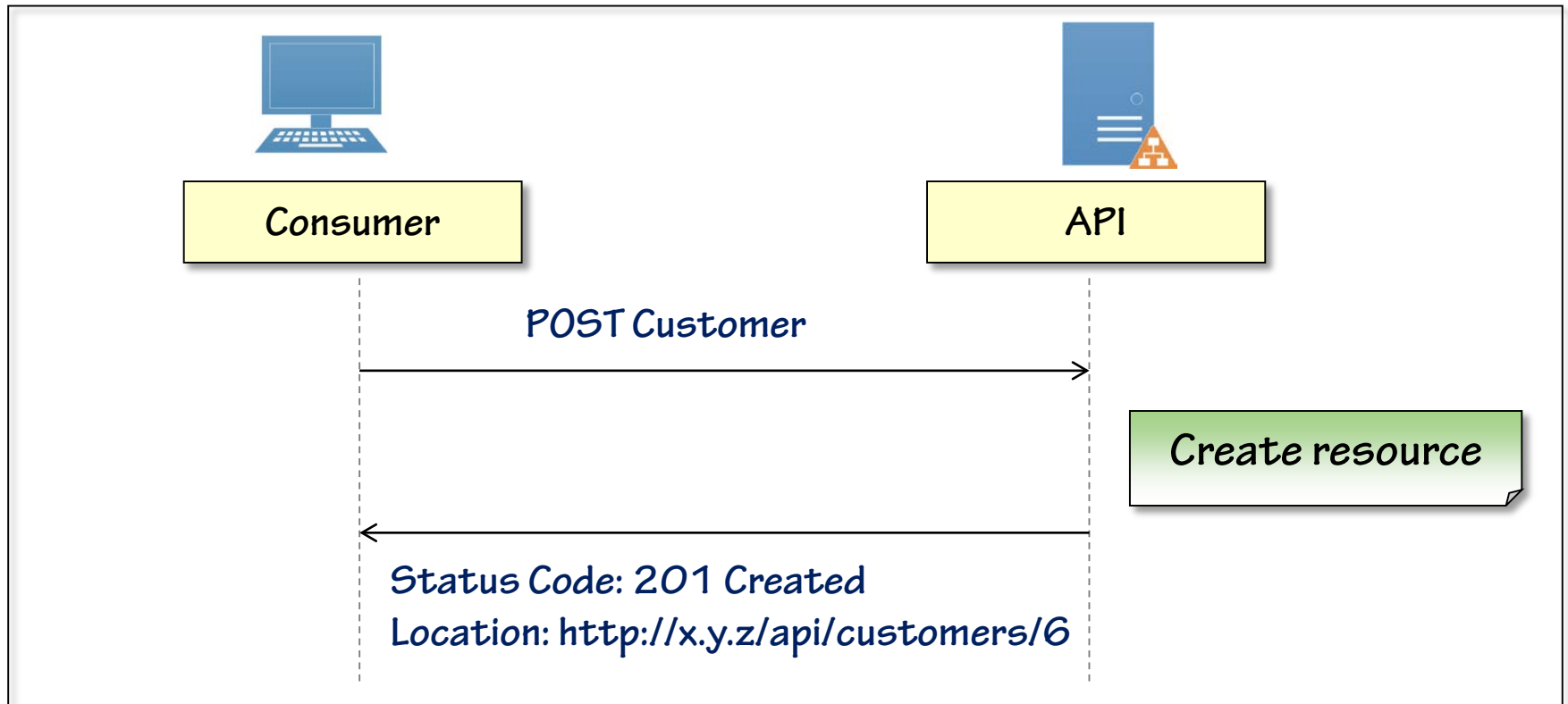
```
public static void DisposeSafely  
    (this ICommunicationObject communicationObject)
```

```
if (communicationObject.State == CommunicationState.Faulted)  
    //etc.  
var disposable = communicationObject as IDisposable;  
if (disposable != null)  
    //etc.
```

```
finally  
{  
    client.DisposeSafely();  
}
```

HTTP Location Header

- Response header (RFC 2616)
 - From PUT or POST
 - With 201-Created status code



Demo 5: WebApi POST Responses

Feature

Adopt RFC 2616
and return
Location header
from POSTs

Task

Extend HTTP
response
message to add
location header

Demo 5: WebApi POST Responses

Demo 5: WebApi POST Responses

- **HttpResponseMessageExtensions**

- In System.Net.Http namespace
- Added `AddLocationHeader()`

```
public static void AddLocationHeader  
    (this HttpResponseMessage response,  
     HttpRequestMessage request, int entityId)
```

```
var response = new HttpResponseMessage(HttpStatusCode.Created);  
response.AddLocationHeader(Request, customer.Id);
```

Status Code: 201 Created

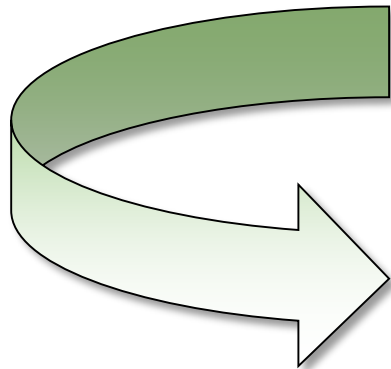
Location: <http://localhost/Sixeyed.ExtensionLibrary.Stubs.Api/api/customers/6>

ASP.NET MVC Html Helper

- Render HTML from .NET objects

- DropDownListFor()
 - IEnumerable<SelectListItem>
- For enums

```
public enum ModuleStatus
{
    [System.ComponentModel.Description("<Please Select>")]
    None = 0,
    Todo = 1,
    [System.ComponentModel.Description("In Progress")]
    InProgress = 2,
    Complete = 3
}
```



Status <Please Select> ▼

Regis

- <Please Select>
- Todo
- In Progress
- Complete

Demo 6: ASP.NET MVC HtmlHelper

Feature

Display fixed
choice lists as
dropdowns in the
Web UI

Task

Extend
HtmlHelper to
build dropdowns
from enums

Demo 6: ASP.NET MVC HtmlHelper

Demo 6: ASP.NET MVC HtmlHelper

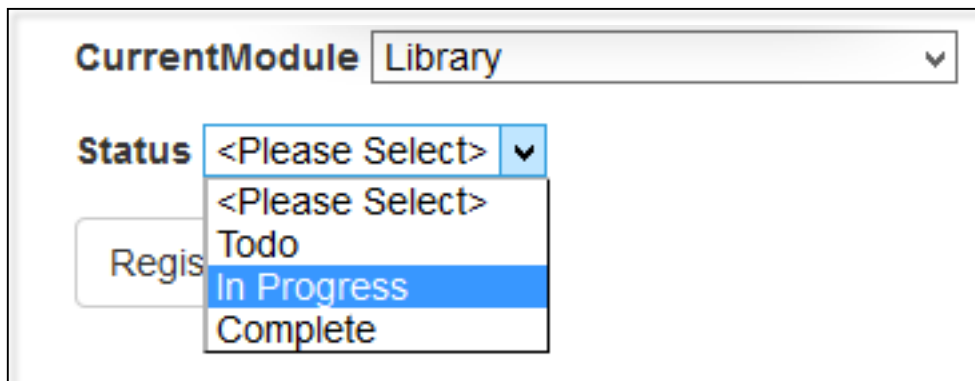
■ HtmlHelperExtensions

- System.Web.Mvc.Html namespace
- Added EnumDropDownListFor()

```
public static MvcHtmlString EnumDropDownListFor<TModel, TEnum>  
(this HtmlHelper<TModel> htmlHelper,  
    Expression<Func<TModel, TEnum>> enumAccessor)
```

```
@Html.LabelFor(model => model.Status)
```

```
@Html.EnumDropDownListFor(model => model.Status)
```



The screenshot shows a web form with two main components. At the top, there is a label 'CurrentModule' followed by a dropdown menu currently displaying 'Library'. Below this, there is a label 'Status' followed by a dropdown menu. The 'Status' dropdown menu is open, showing a list of options: '<Please Select>', '<Please Select>', 'Todo', 'In Progress' (which is highlighted in blue), and 'Complete'. To the left of the 'Status' dropdown, there is a button labeled 'Regis'.

Extension Method Library

- **Core**

- `Exception.FullMessage()`
- `Enum.GetName()` & `Enum.GetDescription()`

- **Reflection**

- `LambdaExpression.ToPropertyInfo()`

- **EntityFramework**

- `DbContext.Save()`

- **WCF**

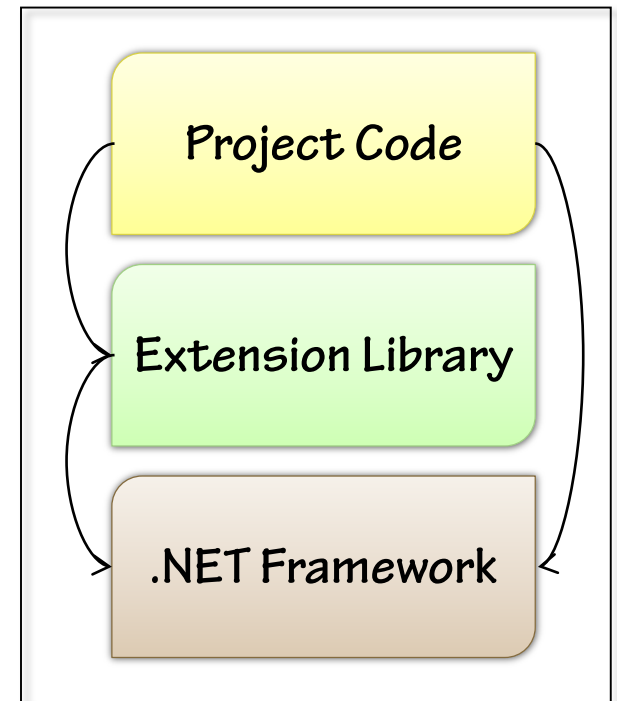
- `CommunicationObject.DisposeSafely()`

- **WebApi**

- `HttpResponseMessage.AddLocationHeader()`

- **ASP.NET MVC**

- `HtmlHelper.EnumDropDownListFor()`



Course Outline

- **Module 1: Introducing Extension Methods** ✓
 - Writing and using
 - What scenarios they enable
- **Modules 2 & 3: Advanced Extension Methods** ✓
 - Restrictions with extension methods
 - How .NET implements extensions
- **Modules 4 & 5: Extension Method Library** ✓
 - Core, Reflection, Entity Framework
 - WCF, WebAPI, ASP.NET MVC
- **Aim of the course**

Learn everything about extension methods in C#

References

- **WCF client Dispose() issues**

- David Barrett describes the problem
 - <http://geekswithblogs.net/DavidBarrett/archive/2007/11/22/117058.aspx>

- **HTTP Location Header**

- Described
 - http://en.wikipedia.org/wiki/HTTP_location
- In RFC2616
 - <http://tools.ietf.org/html/rfc2616>