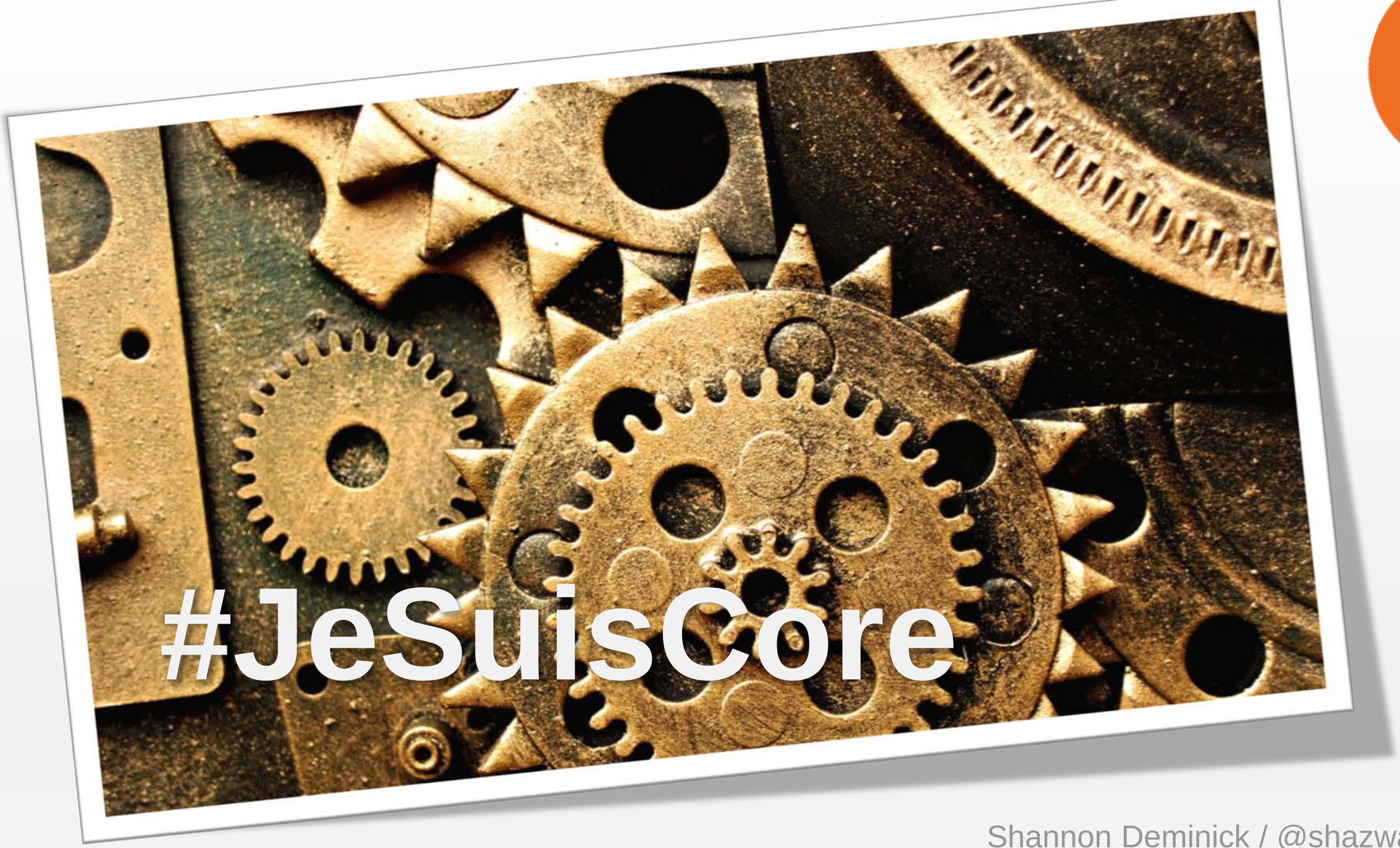


Shannon Deminick / @shazwazza
Stephan Gay / @zpqrtbnk

Demo Roulette®™

Never been done before!

First of its kind!

Your PC ran into a problem and needs to restart. We're just collecting some error info, and then we'll restart for you.

25% complete

For more information about this issue and possible fixes, visit
http://windows.com/stopcode

If you call a support person, give them this info:
Stop code: CRITICAL_PROCESS_DIED

Attempt

Attempt – Using

```
var attempt = service.CreateThing(arguments);
if (attempt.Success)
{
    var thing = attempt.Result;
    Console.WriteLine(thing.Name);
}
else
{
    Console.WriteLine("Could not create a thing.");

    if (attempt.Exception != null)
        Console.WriteLine(attempt.Exception.Message);
}
```

Attempt – Creating

```
public Attempt<Thing> CreateThing(object arguments)
{
    if (!CanCreate(arguments))
        return Attempt<Thing>.Fail();

    try
    {
        return Attempt.Succeed(Create(arguments));
    }
    catch (Exception exception)
    {
        return Attempt<Thing>.Fail(exception);
    }
}
```

Attempt – More Details

```
Attempt<Thing, OperationStatus> attempt = ...

if (attempt)
{
    // ...
}
else
{
    switch (attempt.Status)
    {
        case OperationStatus.FailedNotEnoughDiskSpace:
        // ...
    }
}
```

Attempt – More More Details

```
public struct CreationResult
{
    public string Message { get; set; }

    public Thing Thing { get; set; }
}
```

Attempt – More More Details

```
if (!CanCreate(arguments))
    return Attempt.Fail(new CreationResult { Message = "Invalid arguments." });

try
{
    return Attempt.Succeed(new CreationResult { Thing = Create(arguments) });
}
    catch (Exception exception)
{
    return Attempt.Fail(new CreationResult { Message = "Exception." }, exception);
}
```

Attempt – In Core

```
var contentService = ApplicationContext.Current.Services.ContentService;

var attempt = contentService.Publish(content);

if (attempt == false)
{
    // ...
}
```

Xml


<doc1 id="1079" key="1fdc89f1-c51c-4156-ae77-6cc7379d2c3a" parentID="-1" level="1" creatorID="0"
<upload><![CDATA[/media/1002/14265182137_a37d7b2a34_os.jpg]]></upload>
<colorPicker><![CDATA[f3450c]]></colorPicker>
</doc1>
<doc1 id="1080" key="2999cc25-ce51-40be-8576-9db2a1ac42c2" parentID="-1" level="1" creatorID="0"
<upload><![CDATA[/media/1003/14265182137_a37d7b2a34_os.jpg]]></upload>
</doc1>
<cropper id="1083" key="da46fe49-147c-4eb3-bddc-8d0f04538023" parentID="-1" level="1" creatorID="
<image><![CDATA[{
"focalPoint": {
"left": 0.5,
"top": 0.5
},
"src": "/media/1017/dark-forest-road.jpg",
"crops": [
{
"alias": "100x100",
"width": 100,
"height": 100
}
]
]]></image>
</cropper>

Xml – The XmlDocument Way

```
var xml = content.Instance.XmlContent; // is an XmlDocument
var node = xml.GetElementById("1234"); // is an XmlElement
Assert.IsNotNull(node);

var value = node.SelectSingleNode("myProp")?.InnerText;
Assert.AreEqual("some value", value);
```

Xml – The IXPathNavigator Way

```
var nav = UmbracoContext.Current.ContentCache.CreateNavigator();
Assert.IsNotNull(nav);

Assert.IsTrue(nav.MoveToId("1234"));

Assert.IsTrue(nav.MoveToFirstChild());
while (nav.Name != "myProp" && nav.MoveToNext()) { }
Assert.AreEqual("myProp", nav.Name);
Assert.AreEqual("some value", nav.Value);
```

Xml – The IXPathNavigator Way

- IXPathNavigator is all XPath needs to run

- Anything can be navigated, thanks to Core NavigableNavigator and INavigableSource, INavigableContent, INavigableContentType, INavigableFieldType. That’s all. Implement these and be happy.

Xml – Speed?

```
BenchmarkDotNet=v0.9.7.0
OS=Microsoft Windows NT 6.2.9200.0
Processor=Intel(R) Core(TM) i7-3840QM CPU 2.80GHz, ProcessorCount=8
Frequency=2728191 ticks, Resolution=366.5433 ns, Timer=TSC
HostCLR=MS.NET 4.0.30319.42000, Arch=32-bit RELEASE
JitModules=clrjit-v4.6.1055.0

Type=XmlBenchmark  Mode=Throughput  Runtime=Clr
LaunchCount=1

         Method |      Median |     StdDev |  Gen 0 | Gen 1 | Gen 2 | Bytes Allocated/Op |
--------------- |------------ |----------- |------- |------ |------ |------------------- |
    XmlDocument | 930.4868 ns | 29.9177 ns | 761,00 |     - |     - |             290,45 |
 XPathNavigator | 226.0623 ns |  3.9530 ns |  55,18 |     - |     - |              20,89 |
```

NuCache

DEMO

V8 - IoC/Dependency Injection

V8 - IoC/Dependency Injection

- Light Inject is default container which you can use and modify
- All plugin Resolvers now use IoC == DI for all plugins

V8 - IoC/Dependency Injection

```
[PropertyEditor(Constants.PropertyEditors.UploadFieldAlias, "File upload", "fileupload",
    Icon = "icon-download-alt", Group = "media")]
public class FileUploadPropertyEditor : PropertyEditor, IApplicationEventHandler
{
    private readonly MediaFileSystem _mediaFileSystem;
    private readonly IContentSection _contentSettings;
    private readonly ILocalizedTextService _textService;

    public FileUploadPropertyEditor(ILogger logger, MediaFileSystem mediaFileSystem,
                IContentSection contentSettings, ILocalizedTextService textService)
        : base(logger)
    {
        _mediaFileSystem = mediaFileSystem;
        _contentSettings = contentSettings;
        _textService = textService;
    }
```

V8 - IoC/Dependency Injection

```
public class MyBootManager : WebBootManager
{
    public MyBootManager(UmbracoApplicationBase umbracoApplication)
        : base(umbracoApplication)
    {
    }

    //Called to custpmize the IoC container
    protected override void ConfigureApplicationServices(ServiceContainer container)
    {
        base.ConfigureApplicationServices(container);

        container.RegisterSingleton<MyAwesomeService>();
        container.Register<MyStatefulThing>();
        container.Register<MyRequestInstance>(new PerRequestLifeTime());
    }
}
```

V8 - IoC/Dependency Injection

```
public class MyApplication : UmbracoApplication
{
    protected override IBootManager GetBootManager()
    {
        return new MyBootManager(this);
    }
}
```

V8 - IoC/Dependency Injection

```
<%@ Application Inherits="My.Application.MyApplication" Language="C#" %>
```

OAuth & Authentication

OAuth

- Do you have an external Authentication store?
- Want a ‘Single Sign on’ approach between installs?
- Need 2 factor authentication?

OAuth

```
PM> Install-Package UmbracoCms.IdentityExtensions
```

```
PM> Install-Package
UmbracoCms.IdentityExtensions.Google
```

DEMO

Identity Server?

https://github.com/Shazwazza/IdentityServerTest

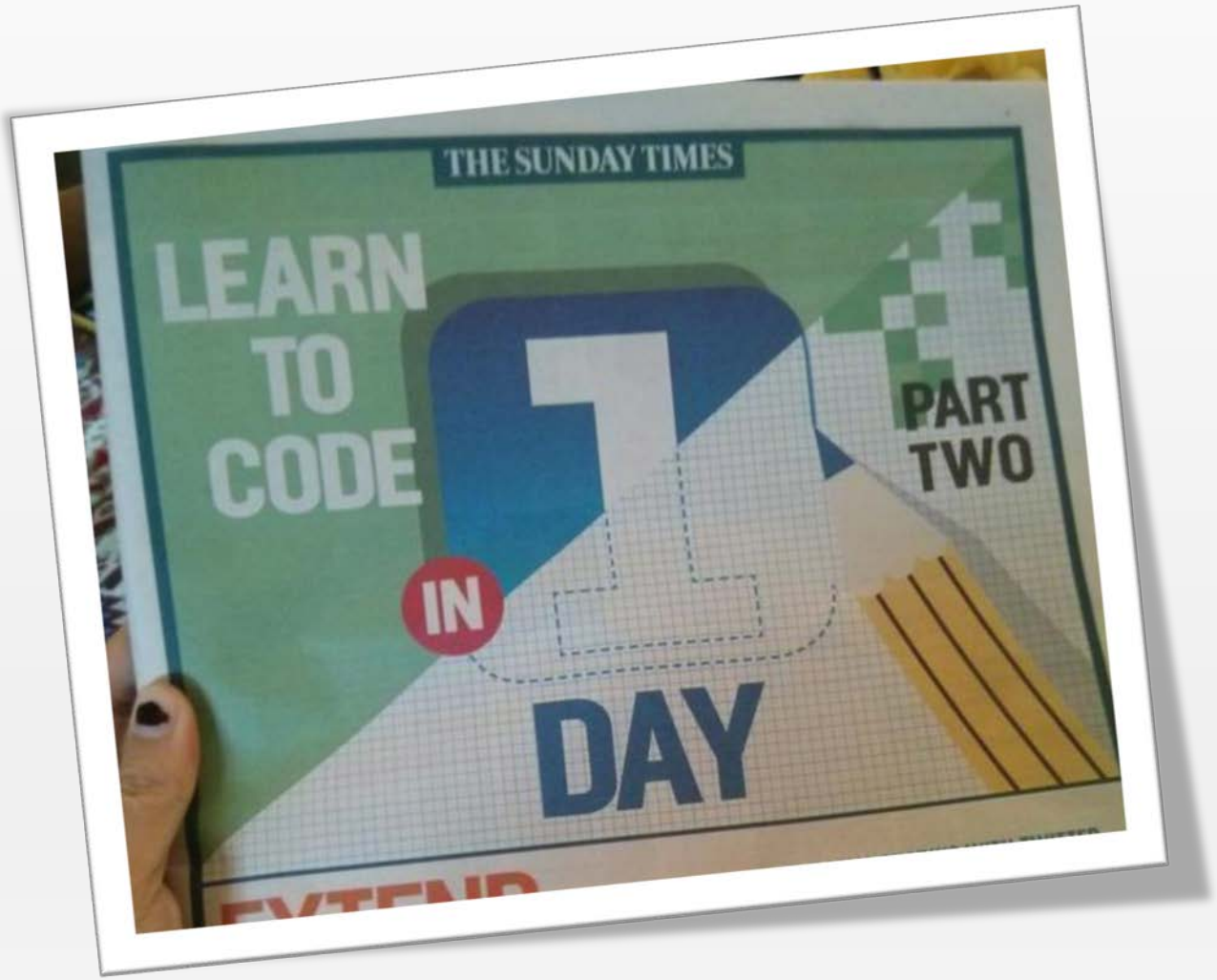
Custom authenticator

- Just want to have custom logic to authenticate a username/password?
- 1. Use: IBackOfficeUserPasswordChecker
- 2. Implement: CheckPasswordAsync
- 3. Register it on startup

It’s documented ☺

https://our.umbraco.org/Documentation/Reference/Security

Updating Content



Updating Content

```
// assuming 'someValue'

var contentService = ApplicationContext.Current.Services.ContentService;
var content = contentService.GetById(1234);
content.SetValue("someProperty", someValue);
var attempt = contentService.Save(content);

// deal with 'attempt'
```

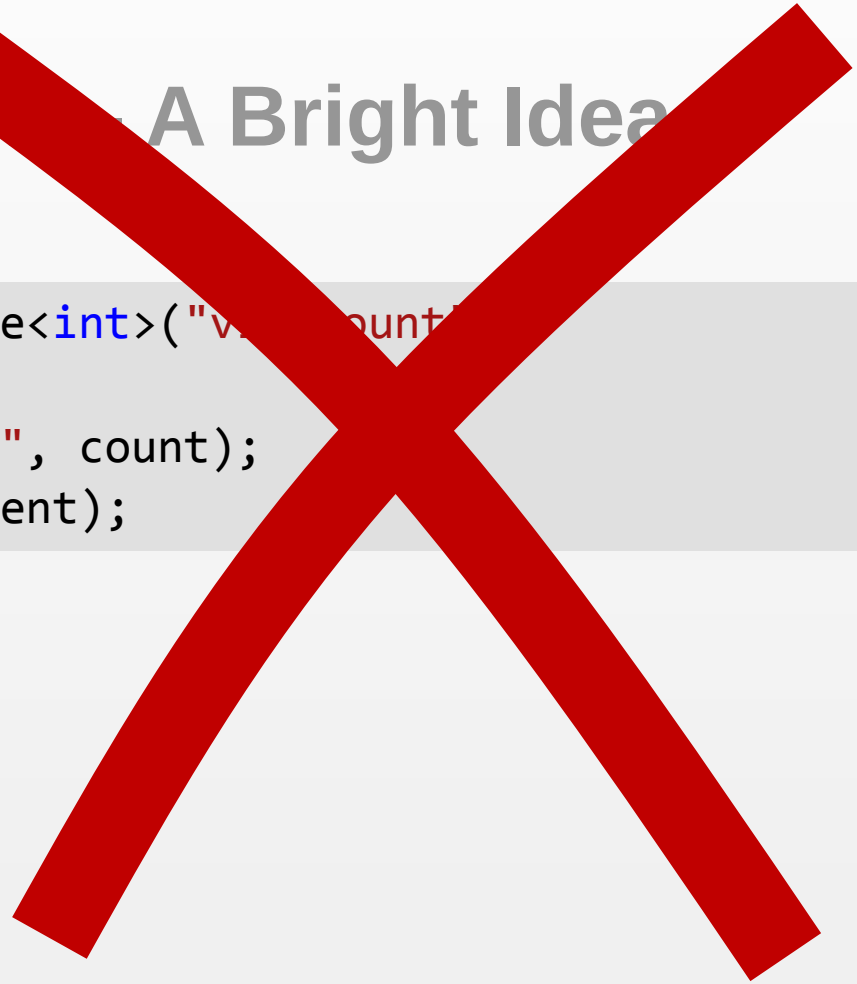
Updating Content – Files

```
Stream stream = // ...

var dataTypeService = ApplicationContext.Current.Services.DataTypeService;
content.SetValue("fileProperty", "filename", stream, dataTypeService);
```

Updating Content: A Bright Idea

```
var count = content.GetValue<int>("viewCount");
count += 1;
content.SetValue("viewCount", count);
service.SaveAndPublish(content);
```



My Own Data


Sales DataBase
TimeKey
CustomerKey
ShipperKey
ProductKey
EmployeeKey
ItemFreight
ItemQuanity
ItemAll
Discount
Action
Title
Address
City
Region
PCode
Country
Phone
Fax

My Own Data – Poco Object

```
[TableName("myProduct")]
[PrimaryKey("id")]
[ExplicitColumns]
internal class ProductDto
{
    [Column("id")]
    [PrimaryKeyColumn]
    public int Id { get; set; }

    [Column("description")]
    [NullSetting(NullSetting = NullSettings.NotNull)]
    public string Description { get; set; }

    [Column("price")]
    [NullSetting(NullSetting = NullSettings.NotNull)]
    [ForeignKey(typeof(NodeDto))]
    public decimal Price { get; set; }
}
```

My Own Data – CRUD

```
public PropertyDataDto CreateProduct(UmbracoDatabase database, string description, decimal price)
{
    var dto = new ProductDto { Description = description, Price = price };
    database.Insert(dto);
    return dto; // has updated Id!
}

public void ChangePrice(UmbracoDatabase database, int productId, decimal price)
{
    var dto = database.Fetch<ProductDto>("SELECT * FROM myProduct WHERE id=@id",
        new { id = productId });
    dto.Price = price;
    database.Update(dto);
}
```

My Own Data – CRUD

```
public void ChangePrice(DatabaseContext dbContext, int productId, decimal price)
{
    var syntax = dbContext.SqlSyntax;

    var sql = new Sql()
        .Select("*")
        .From<ProductDto>(syntax)
        .Where<ProductDto>(x => x.Id == productId);

    var dto = database.Fetch<ProductDto>(sql);
    dto.Price = price;
    database.Update(dto);
}
```

My Own Data – Database

```
var dbContext = ApplicationContext.Current.DatabaseContext;
var database = dbContext.Database;
var syntax = dbContext.SqlSyntax;
```

My Own Data – Setup

```
var logger = ApplicationContext.Current.ProfilingLogger.Logger;

var dbContext = ApplicationContext.Current.DatabaseContext;
var database = dbContext.Database;
var syntax = dbContext.SqlSyntax;

var schemaHelper = new DatabaseSchemaHelper(database, logger, syntax);
schemaHelper.CreateTable<ProductDto>();
```

Profiling

Profiling – Measuring

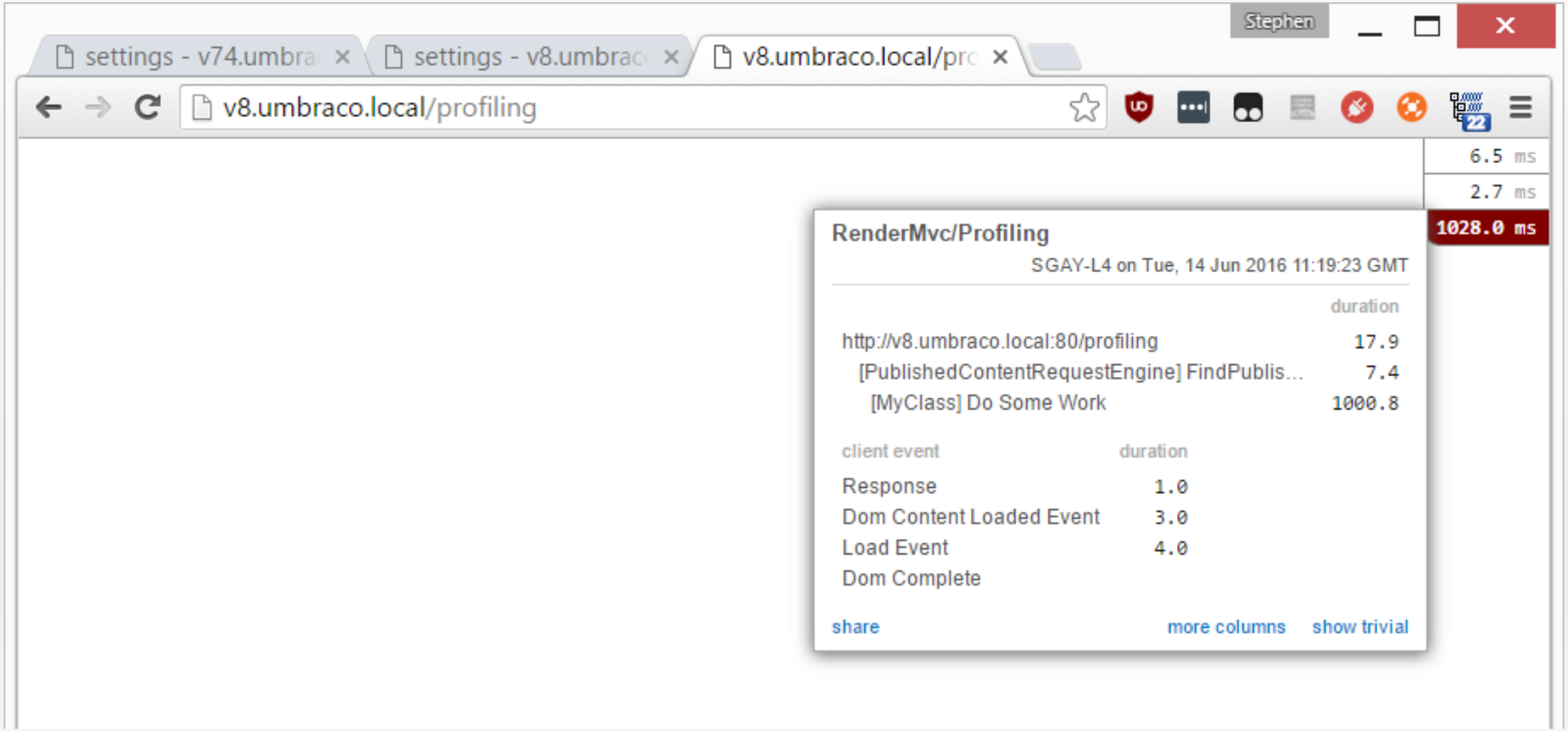
```
var logger = ApplicationContext.Current.ProfilingLogger;

using (logger.DebugDuration<MyClass>("Do Some Work", "Done"))
{
    // do our work...
}
```

Profiling – Reporting in UI

```
    ...
    <div class="footer">...</div>
    @Html.RenderProfiler()
</body>
</html>
```

Profiling – Results



Profiling – Results

```
2016-06-14 13:19:23,232 [P16264/D4/T1411] DEBUG MyClass – Do Some Work
2016-06-14 13:19:24,234 [P16264/D4/T1411] DEBUG MyClass - Done (took 1001ms)
```

DEMO

Querying FrontEnd Content

Querying FrontEnd Content

- Navigating the tree, etc
- Avoiding Descendants, creating (and caching) indexes, etc
- XPath vs…
- Recursive querying
- IsComposedOf
- Benchmarking?

Recursion, take 1

```
foreach (var content in Model.AncestorsOrSelf())
{
    if (content.IsComposedOf("globalSharing"))
    {
        twitterAccount = content.GetPropertyValue<string>("twitter");
        if (twitterAccount.IsNullOrWhiteSpace() == false) break;
    }
}
```

Recursion, take 2

```
foreach (var content in Model.AncestorsOrSelf())
{
    var globalSharing = content as IGlobalSharing;
    if (globalSharing != null
        && globalSharing.TwitterAccount.IsNullOrWhiteSpace() == false)
    {
        twitterAccount = globalSharing.TwitterAccount;
        break;
    }
}
```

Recursion, take 3

```
twitterAccount = Model.AncestorsOrSelf()
    .OfType<IGlobalSharing>()
    .Where(x => x.TwitterAccount.IsNullOrWhiteSpace() == false)
    .Select(x => x.TwitterAccount)
    .FirstOrDefault();
```

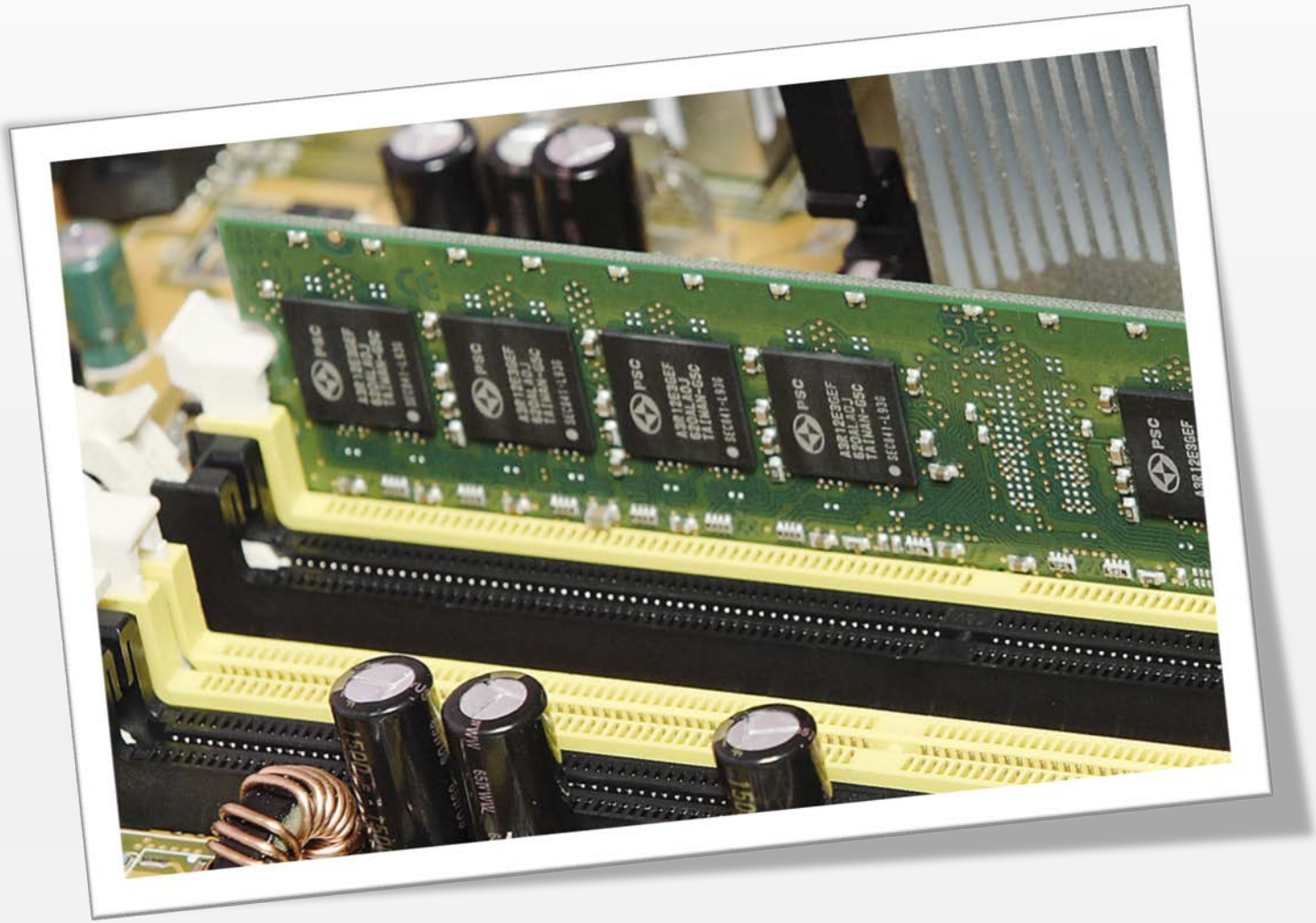
Anti-Recursion

```
var newItems = Model.Descendants()
    .OfType<NewsItem>()
    .OrderBy(x => x.SortOrder)
    .Take(3);
```

What about XPath?

- Can do, on the content cache
- v7: benchmarking required
- v8: slow, compared to Linq

Caching



Caching - DRY

```
CacheHelper cacheHelper = ApplicationContext.Current.ApplicationCache;
```

Caching – Static Cache

```
var staticCache = cacheHelper.StaticCache;
var staticValue = staticCache.GetCacheItem(cacheKey, () =>
{
    return CreateMyPreciousValue(args);
});
```

Caching – Runtime Cache

```
var runtimeCache = cacheHelper.RuntimeCache;
var runtimeValue = runtimeCache.GetCacheItem(cacheKey, () =>
{
    return CreateMyPreciousValue(args);
}, TimeSpan.FromSeconds(10), true);
```

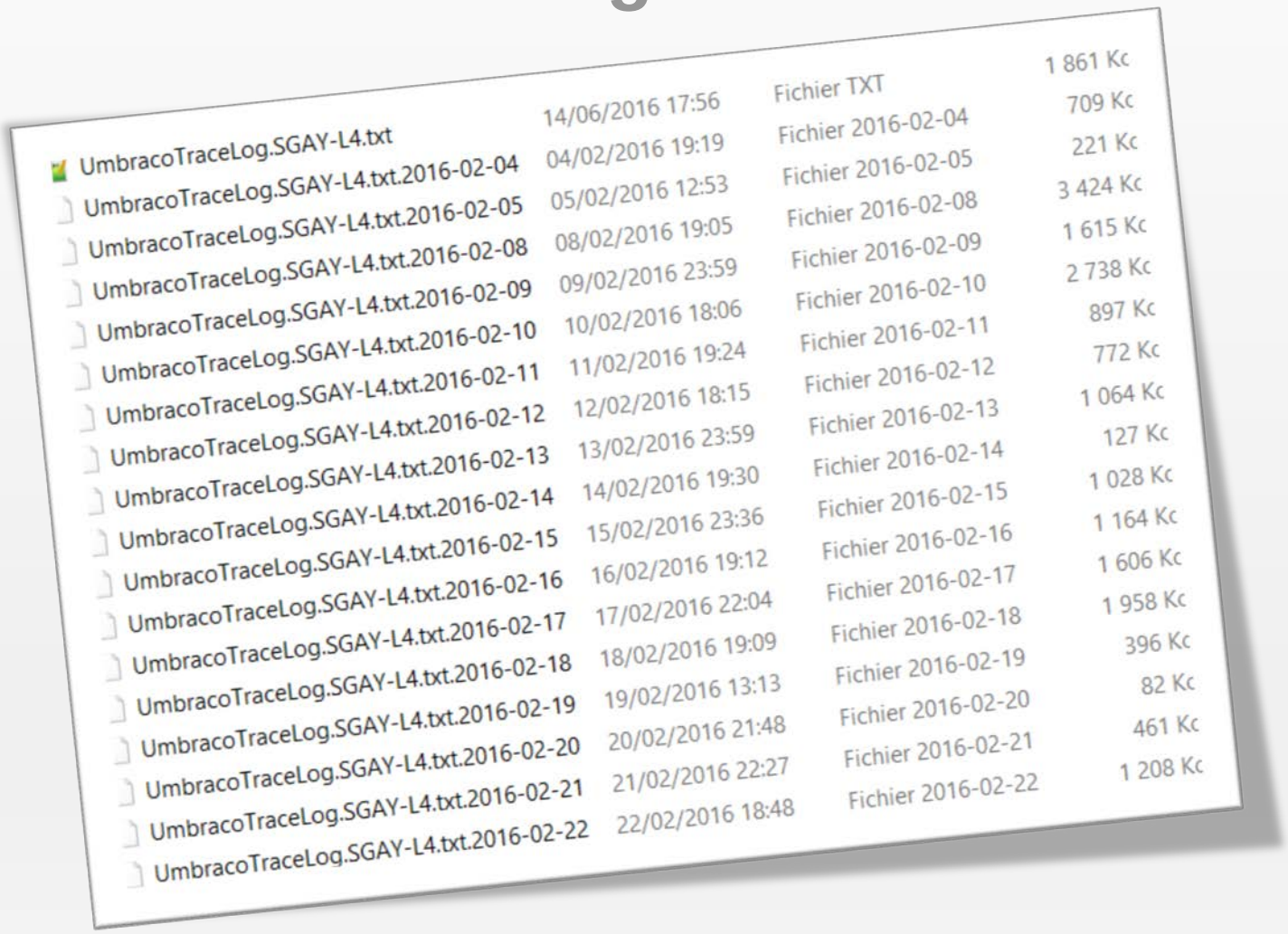
Caching – Request Cache

```
var requestCache = cacheHelper.RequestCache;
var requestValue = requestCache.GetCacheItem(cacheKey, () =>
{
    return CreateMyPreciousValue(args);
});
```

Caching – Isolated Cache

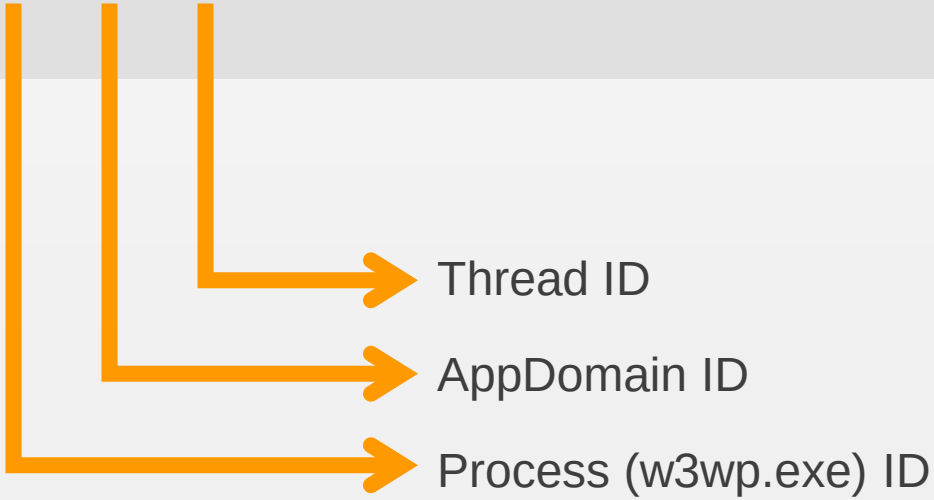
```
var isolatedCache = cache.IsolatedRuntimeCache;
var myThingCacheAttempt = isolatedCache.GetCache<MyThing>();
var myThingCache = myThingCacheAttempt.Result;
var isolatedValue = myThingCache.GetCacheItem(cacheKey, () =>
{
    return CreateMyPreciousValue(args);
}, TimeSpan.FromSeconds(10), true);
```

Better UmbracoTraceLog



UmbracoTraceLog – process, domain & thread

```
2016-06-14 13:19:23,221 [P16264/D4/T1411] DEBUG Umbraco.Web.Routing.PublishedContentRequestEngine
2016-06-14 13:19:23,222 [P16264/D4/T1411] DEBUG Umbraco.Web.Routing.ContentFinderByNiceUrlAndTemplate
2016-06-14 13:19:23,223 [P16264/D4/T1411] DEBUG Umbraco.Web.Routing.ContentFinderByNiceUrl
```



log4net.config – all-or-nothing

```
<root>
  <priority value="Info"/>
  <appender-ref ref="AsynchronousLog4NetAppender" />
</root>
```

log4net.config – fine-grain

```
... DEBUG Umbraco.Web.Routing.PublishedContentRequestEngine - Finder Umbraco.Web.Routing.ContentFind
... DEBUG Umbraco.Web.Routing.ContentFinderByNiceUrlAndTemplate - Valid template: "profiling"
... DEBUG Umbraco.Web.Routing.ContentFinderByNiceUrl - Test route "/"
... DEBUG Umbraco.Web.Routing.ContentFinderByNiceUrl - Got content, id=1228
... DEBUG Umbraco.Web.Routing.PublishedContentRequestEngine - FindPublishedContent: End finders, no d
... DEBUG Umbraco.Web.Routing.PublishedContentRequestEngine - HandlePublishedContent: Begin
... DEBUG Umbraco.Web.Routing.PublishedContentRequestEngine - EnsurePublishedContentAccess: Page is n
... DEBUG Umbraco.Web.Routing.PublishedContentRequestEngine - HandlePublishedContent: End
... DEBUG Umbraco.Web.Routing.PublishedContentRequest - FindTemplate: Has a template already, and no
... DEBUG Umbraco.Web.Routing.PublishedContentRequestEngine - HandleWildcardDomains: Path="-1,1228"
... DEBUG Umbraco.Web.Routing.PublishedContentRequestEngine - HandleWildcardDomains: No match.
... DEBUG Umbraco.Web.UmbracoModule - Response status: Redirect=none, Is404=false, StatusCode=0
```

log4net.config – fine-grain

```
<logger name="Umbraco.Web.Routing.ContentFinderByNiceUrl">
  <level value="WARN" />
</logger>
```

Lang Files


Hola - Hello
Gracias - Thanks
Por Favor - Please
De Nada - You're Welcome

Lang Files – Umbraco Master

- ~/Umbraco/Config/Lang/es.xml

```
<language alias="es" intName="Spanish" localName="español“
    lcid="10" culture="es-ES">

    <area alias="actions">
      <key alias="assignDomain">Administrar hostnames</key>
      <key alias="auditTrail">Auditoría</key>
      <key alias="browse">Nodo de Exploración</key>
      <key alias="changeDocType">Cambiar tipo de documento</key>
      <key alias="copy">Copiar</key>
```

Lang Files – User

- ~/Config/Lang/es.user.xml

```
<language>
  <area alias="actions">
    <key alias="assignDomain">Something Totally Different</key>
  </area>
</language>
```

Lang Files – Plugins

- ~/App_Plugins/{MYPLUGIN}/lang/es.user.xml

```
<language>
  <area alias="actions">
    <key alias="assignDomain">Something Totally Different</key>
  </area>
</language>
```

Concurrency

Concurrency – Database Level Locks

- What’s shared by all instances?

- Lock umbracoLock table rows
- Within RepeatableRead transactions

- Read record = obtains shared (read) lock on that record
- Write record = obtains exclusive (write) lock on that record

Concurrency – Using Database Locks

```
using (var uow = uowProvider.CreateUnitOfWork())
{
    uow.ReadLock(Constants.Locks.ContentTree);

    // nobody can write to the content tree!

    uow.Complete();
}
```

Strings

Strings – The Case of Invariants

```
if (string1.ToUpper() == "CONTENT")
{ ... }
```

```
if (string1.ToUpper() == string2.ToUpper())
{ ... }
```

```
if (string1.Equals(string2, StringComparison.OrdinalIgnoreCase))
{ ... }
```

```
if (string1.InvariantEquals(string2))
{ ... }
```

Strings – Extensions

- InvariantEquals, InvariantStartsWith, InvariantEndsWith
- EncryptWithMachineKey, DecryptWithMachineKey
- EncodeJsString
- EnsureStartsWith, EnsureEndsWith
- ToUrlBase64, FromUrlBase64, ToMd5,ToCSharpString
- ReplaceMany
- *etc*

Strings – Cleaning Strings

```
var filename = name.ToSafeFileName(); // safe on disk and as a url
```

```
var alias = name.ToSafeAlias(); // safe as alias, JavaScript or C# variable...
```

```
var output = input.ToCleanString(cleanStringType);
```

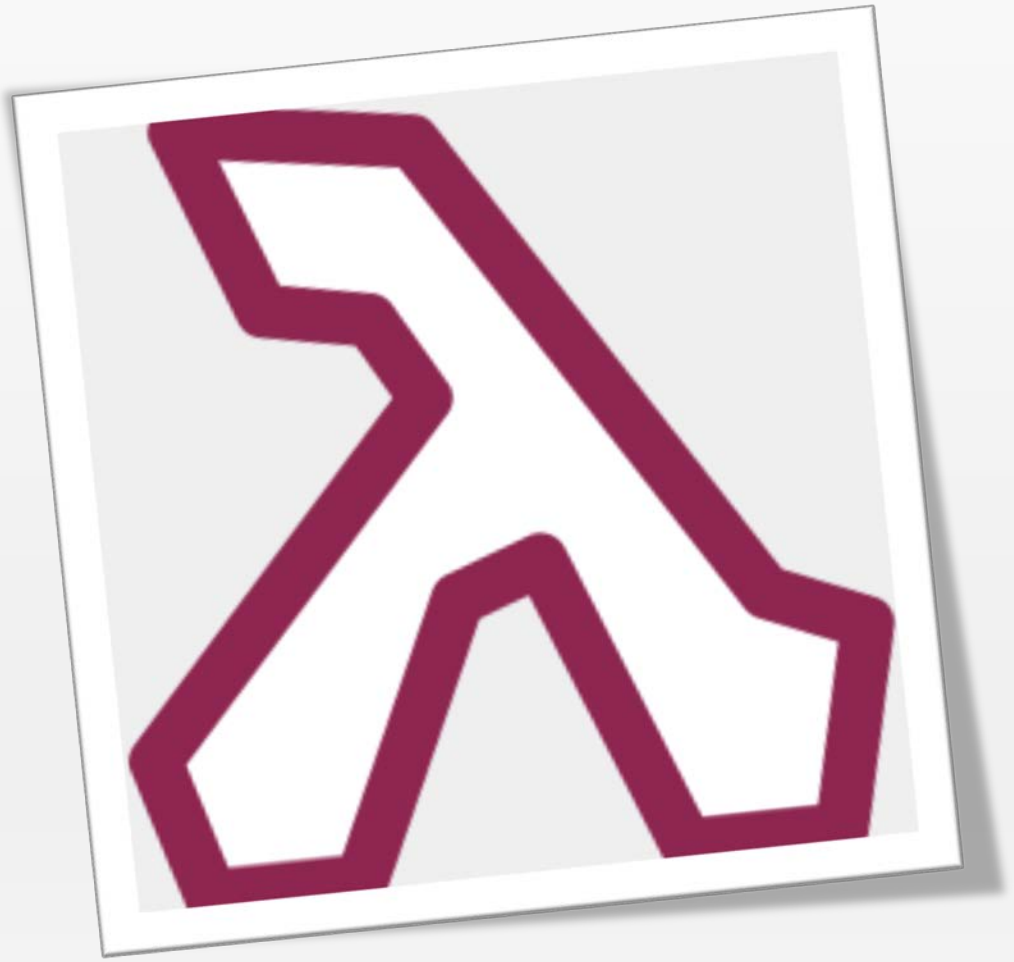
Strings – Clean String Type

- Casing
  - PascalCase | CamelCase | Unchanged | LowerCase | UpperCase
- Encoding
  - Utf8 | Ascii
- Role
  - UrlSegment | Alias | UnderscoreAlias | FileName | ConvertCase

Strings – DefaultShortStringHelper

```
var helper = new DefaultShortStringHelper(umbracoSettings)
    .WithConfig(CleanStringType.Alias, new DefaultShortStringHelper.Config
    {
        BreakTermsOnUpper = true,
        CutAcronymOnNonUpper = true,
        GreedyAcronyms = true,
        IsTerm = (c, leading) => leading
            ? char.IsLetter(c) // only letters
            : (char.IsLetterOrDigit(c) || c == '_'), // letter, digit or underscore
        PostFilter = s => s.Length > 64 ? s.Substring(0, 64) : s, // max 64 chars output
        PreFilter = s => s.Replace("*", "!STAR!"), // pre-process input
        Separator = '-',
        StringType = CleanStringType.Ascii // convert to ASCII
    });
```

LinqPad with Umbraco



LinqPad with Umbraco

- Want to use the Umbraco Core API outside of Umbraco?
- Handy for running scripts
- Not handy for web based operations (i.e. publishing)
  - 7.3+ has ‘Instructions’ table, so web based operations *could* be queued
- IQueryable! – for the future
- https://github.com/Shazwazza/UmbracoLinqPadDriver

LinqPad – Umbraco driver download

Latest release

v1.0.0

21ce1bc

v1.0.0

Shazwazza released this on Mar 27, 2015 · **4 commits** to master since this release

Initial release, see readme for installation details, etc...

You can download the driver (UmbracoLinqPad.lpx) on this page

Downloads

UmbracoLinqPad.lpx 124 KB

This one

Source code (zip)

Source code (tar.gz)

DEMO

- Slides: http://bit.ly/JeSuisCore

- Questions ??

THANKS

Code for this presentation:

github.com/umbraco/CodeGardenRoulette