#h8yr

Le Tour de la Version 8

#h8yr

Le Tour de la Version 8

#orangeAllTheThings

Why?

Why?

Everything is Slighly Broken ™

… time to …

Break All The Things Again

When?


DUKE NUKEM
FOREVER

When?

When we have done what we want to do

Full time effort w/ regular alpha asap

7

8

```
d8 = "!f() { \
        git diff --name-only dev-7.6-rc1 dev-7.6-rc2 -- $1 ; \
     }; f"



git d8 src/Umbraco.Core/Logging/
```

```
m8 = "!f() { \
        if $(git dif --binary dev-7.6-rc1 dev-7.6-rc2 -- $1 \
             | grep -q '+++ /dev/null'); \
        then \
            echo "rm $1"; \
            git rm $1; \
        else \
            echo "merge $1"; \
            git dif --binary dev-7.6-rc1 dev-7.6-rc2 -- $1 \
            | git apply -3 --whitespace=nowarn - ; \
        fi ; \
      }; f"
```

```
git m8 src/Umbraco.Core/Logging/Logger.cs
```

```
git d8 src/Umbraco.Web.UI.Client/ | xargs -n 1 git m8
```

Who?

Who?

Shannon - CTO

Stephan - lead dev

D.Team

You

Yes

You

Who?

Where?

Where?

.NET Framework 4.6.1 (4.7?) on Windows

~~.NET Core~~ ~~.NET Standard~~ ~~.NET WhateverComesNext~~

SQL Server (LocalDB, Express...), SqlCE

MySql

C# 7

Latest Everything

What & How?

Legacy

umbraco.businessLogic.Marco, est. 2003

Legacy

What is legacy code?

Code that scares & confuses you

Legacy

Obsolete.kill.kill.kill

Death to Kludges

Legacy

Writing Legacy Code is Easy

Don't

Test

Document

Legacy

Upgrade?

Tests

Tests

Unit Tests vs. Integration Tests

Open Tests

Tests

```
[TestFixture]
[UmbracoTest(Database = UmbracoTestOptions.Database.NewSchemaPerTest)]
public class MyTests : UmbracoTestBase
{
    …
}
```

Application

is it ApplicationEventHandler.Starting or Started or?

Application

Entry point?

Order?

Composition?

```
internal void HandleApplicationStart(object sender, EventArgs evargs)
{
    // ******** THIS IS WHERE EVERYTHING BEGINS ********

    // create the container for the application, and configure.
    // the boot manager is responsible for registrations
    var container = new ServiceContainer();
    container.ConfigureUmbracoCore(); // also sets Current.Container

    // register the essential stuff,
    // ie the global application logger
    // (profiler etc depend on boot manager)
    var logger = GetLogger();
    container.RegisterInstance(logger);
    // now it is ok to use Current.Logger

    ConfigureUnhandledException(logger);
    ConfigureAssemblyResolve(logger);

    // get runtime & boot
    _runtime = GetRuntime();
    _runtime.Boot(container);
}
```

```
/// <summary>
/// Defines the Umbraco runtime.
/// </summary>
public interface IRuntime
{
    /// <summary>
    /// Boots the runtime.
    /// </summary>
    /// <param name="container">The application service container.</param>
    void Boot(ServiceContainer container);

    /// <summary>
    /// Terminates the runtime.
    /// </summary>
    void Terminate();
}

/// <summary>
/// Represents the state of the Umbraco runtime.
/// </summary>
public interface IRuntimeState
{
    …
}
```

Boot

Compose

Acquire MainDom
Determine RuntimeLevel

Boot components

Compose

Compose

Resolution
→ Dependency Injection

LightInject

ApplicationContext.Current.Services
→ Current.Services

Service Locator (anti?!) pattern

Components

IUmbracoComponent

Compose

Initialize

Terminate

```
public class CgComponent : UmbracoComponentBase, IUmbracoUserComponent
{
    public override void Compose(Composition composition)
    {
        base.Compose(composition);

        composition.ContentFinders().Append<MyContentFinder>();
    }

    public void Initialize()
    {
        // ...
    }

    public override void Terminate()
    {
        // ...
    }
}
```

```
public class CgComponent : UmbracoComponentBase, IUmbracoUserComponent
{
    public override void Compose(Composition composition)
    {
        base.Compose(composition);

        composition.ContentFinders().Append<MyContentFinder>();
    }

    public void Initialize(ILogger logger)
    {
        logger.Debug<CgComponent>("hello");
    }

    public override void Terminate()
    {
        // ...
    }
}
```

Application / Components

Components Graph

Requiring / Required Component

Disabled / Enabled Component

RuntimeLevel

Core / User components

Services

and everything underneath

Scopes

Service-level unit-of-work

Database transaction

```
using (var scope = scopeProvider.CreateScope())
{
    …

    scope.Complete();
}
```

Persistence

NPoco

~~DatabaseContext~~ → Scope

~~ReadCommited~~ → RepeatableRead

Locks

```
using (var scope = scopeProvider.CreateScope())
{
    scope.ReadLock(Constants.Locks.ContentTree);

    …
}
```

The Big Magic Static L2 Caches

Events

Events

Business vs Technical Events

Business vs Cache Events

Events

Scoped Events

Events

```
// raise immediate event
Published(this, new PublishEventArgs<IContent>(…));


// register event with scope
scope.DispatchEvent(Published, this, new PublishEventArgs<IContent>(…));


// should we go with event classes?
scope.DispatchEvent(new PublishedEvent<IContent>(this, …));
```

Published Content

nesting repeated grids in Archetype rich text

Published Content

Properties → Methods

Rich Content Experience

Built-in PropertySet

New PropertyValue converters

Value Converters

```
public interface IPropertyValueConverter : IDiscoverable
{
    bool IsConverter(PublishedPropertyType propertyType);

    Type GetPropertyValueType(PublishedPropertyType propertyType);

    PropertyCacheLevel GetPropertyCacheLevel(PublishedPropertyType propertyType);

    object ConvertSourceToInter(PublishedPropertyType propertyType,
                                object source, bool preview);

    object ConvertInterToObject(PublishedPropertyType propertyType,
                                PropertyCacheLevel referenceCacheLevel,
                                object inter, bool preview);

    object ConvertInterToXPath(PublishedPropertyType propertyType,
                               PropertyCacheLevel referenceCacheLevel,
                               object inter, bool preview);
}
```

Value Converters

```
public interface IPropertyValueConverter : IDiscoverable
{
    bool IsConverter(PublishedPropertyType propertyType);

    string GetPropertyValueType(PublishedPropertyType propertyType,
                                IDictionary<string, string> contentTypes);

    PropertyCacheLevel GetPropertyCacheLevel(PublishedPropertyType propertyType);

    object ConvertSourceToInter(PublishedPropertyType propertyType,
                                object source, bool preview);

    object ConvertInterToObject(PublishedPropertyType propertyType,
                                PropertyCacheLevel referenceCacheLevel,
                                object inter, bool preview);

    object ConvertInterToXPath(PublishedPropertyType propertyType,
                               PropertyCacheLevel referenceCacheLevel,
                               object inter, bool preview);
}
```

Caches

NuCache

Content types, domains…

Routes cache 

Dynamics

Umbraco.Content(1234)

~~Umbraco.TypedContent(1234)~~

Umbraco.Content(guid)

content.Where(x => x.Visible && x.Price == 3)

~~content.Where("visible and price = 3")~~

Views

the 1001 ways of rendering a property value

Views

Razor

~~WebForms~~

XSLT

Views

UmbracoViewPage<HomePage>

The Model is the Content

~~UmbracoTemplatePage~~

~~RenderModel~~

~~CurrentPage~~

~~dynamic~~

~~Umbraco,Field~~

Views

```
@model.Price

@model.Value("price")
@model.Value<double>("price")

@model.Value("title", recurse: true)
@model.Value(x => x.Title, recurse: true)



@model.Value(x => x.Name, default: "John Doe")
@model.Value(x => x.Name, format: v => v.ToLowerInvariant(), default: "John Doe")
```

Variants

Variants

Variants

A content type can be

Invariant
VaryByCulture
VaryBySegment
VaryByCultureAndSegment

Variants

A proprety type can be (depending on content type)

Invariant

→ Invariant

VaryByCulture

→ Invariant, VaryByCulture

VaryBySegment

→ Invariant, VaryBySegment

VaryByCultureAndSegment

→ Invariant, VaryByCulture, VaryByCultureAndSegment

Variants

Content nodes have culture / segment variant nodes

Content nodes have one url per culture

Variants

A content "item" actually is a content "group"

Master Node

Culture Variant Nodes

Segment Variant Nodes

Variants / Services

```
contentType.VaryBy
propertyType.VaryBy

contentTypeService.SetVaryBy(contentType, VaryBy.Culture);
contentTypeService.SetVaryBy(propertyType, VaryBy.Culture);
```

Variants / Services

```
var content = contentService.Get(id);
var price = content.GetValue("price");
content.SetValue("price", 200.00);
service.Save(content);
```

Variants / Services

```
var group = contentService.GetVariantGroup(id);

var masterContent = group.GetMaster();
var cultureContent = group.GetCulture("fr-FR");

group.DefaultCulture = "fr-FR";
contentService.Save(group);
```

Variants / Front-End

```
var frenchContent = contentCache.Get(1234, "fr-FR");
var mobileFrenchContent = contentCache.Get(1234, "fr-FR", "mobile");

var group = frenchContent.GetVariantGroup();

var defaultCulture = group.DefaultCulture; // CultureInfo
var cultures = group.GetCultures(); // CultureInfo[]
var segments = group.GetSegments("fr-FR"); // string[]
```

Variants / Front-End

```
@model.Value (x => x.Title, lang:"fr-CA", segment:"man-mobile")
@model.GetUrl("fr-CA")

@model.Value (x => x.Title, lang:"fr-CA,*")
```

Variants / Challenges

Implementation

UX

APIs

dev-v8

Thank You!

Stephan
@zpqrtbnk
zpqrtbnk.net