Developing Page Templates with MVC or Web Forms

- get visible
- get social
- get engaged
- get in touch
- get converted

the user journey

#epi2012
episerver.com/epi2012
About the presenters

Kristoffer Sjöberg | kris'toefər 'fjo:bɛrdj| ‘Chris’
At EPiServer since 2008

Earlier Netstar employee and have focus on Relate/Community/Mail/...
Now, Configuration Management and occasional last-line support.

Joe Bianco  Solution Architect at EPiServer since 2009

Technical half of the sales team for the Northeast region. Software engineer for 16 years, many of those years as a consultant, focus on .NET since 2001.
Overview of MVC

• "New"

• Model, View, Controller
MVC?

- **Model**
  - Carries data from the Controller to the View

- **View**
  - Screen representation of Model

- **Controller**
  - Heart of the MVC infrastructure
  - Class that implements Action methods that responds to user actions.
Overview of MVC

- Model, View, Controller
- "New"
- Convention over Configuration.
- Doesn’t make use of WebControls, PostBacks or ViewState.
- Views may contain other views, creating a composite view.
- Back to basics
Benefits of using MVC

• Better control of the generated HTML

• Better testability
  • because it helps create a separation of concerns (SoC) design
  • ...and thus, manages complexity

• Code re-use

• ASP.NET MVC framework is a lightweight, highly testable presentation framework that is integrated with existing ASP.NET features. The MVC framework is defined in the System.Web.Mvc assembly.
Drawbacks of using MVC

• Re-implement ViewState-like functionality

• Difficult to create a page which consists of many disjunctive parts that are independently loaded.

• A lot of things happen by convention, so you need to know the conventions before understanding the flow

• An existing site will need to be re-imagined to convert it completely to MVC

However, in EPiServer 7, Web Forms pages and MVC may exist side-by-side.
MVC in ASP.NET

- MVC 3 includes two View Engines
  - **Web Forms** view engine (.aspx / .ascx)
  - **Razor** view engine (.cshtml / .vbhtml)

```html
<%= … %>
```

```html
<div>@Model.Property</div>
```
MVC execution life cycle (generic)
Demo

• Building Page Templates with Web Forms
• Building Page Templates with MVC
CMS registers a few default routes, 

\{\textit{lang}\}/\{\textit{node}\}/\{\textit{partial}\}/\{\textit{action}\} \ is the most important

- \{\textit{lang}\} is optional, allows you to specify the desired language.
- \{\textit{node}\} is ’greedy’, will match as much of your page tree as possible.
- \{\textit{partial}\} only applies if the leaf content node’s type is registered to have an \texttt{EPiServer.Web.IPartialRouter}.
- \{\textit{action}\} is optional, but if specified corresponds to a method name in the \texttt{Controller}.

- \{\textit{lang}\}/\{\textit{node}\}/\{\textit{partial}\}/\{\textit{action}\}

When the leaf \texttt{Content} node has been found, the \texttt{ContentType}’s handler is determined: WebForms \texttt{.aspx}-page, or MVC-style \texttt{Controller}.
MVC helpers in EPiServer CMS 7

- In **WebForms**, you would use the **Property WebControl** to output a property

- In **MVC**, you would type (in a Razor View)

  ```
  @Html.PropertyFor(m => m.MainBody)
  
  - or -

  @Html.BeginEditSection("h3", m => m.Heading)
  @Model.Heading
  @Html.EndEditSection("h3")
  ```
Tagging

- If there are multiple renderers available for a model, you may mark a renderer with a tag, and choose the renderer at the point of output.

```csharp
public class Teaser : BlockData
{
    public virtual string Heading { get; set; }
    public virtual XhtmlString Content { get; set; }
}

[RenderDescriptor(Default = true)]
public partial class DefaultTeaserTemplate : BlockControlBase<Teaser>
{
}

[RenderDescriptor(Tags = new string[] { TagConstants.Sidebar })]
public partial class SidebarTeaserTemplate : BlockControlBase<Teaser>
{
}
Tagging

- To select a renderer based on its tag:

  **WebForms:**
  ```csharp
  <EPiServer:Property PropertyName="MainContentArea"
  RenderSettings="{'Tag':'Sidebar'}" />
  ```

  **MVC:**
  ```csharp
  <%: Html.PropertyFor( m => m.MainContentArea, new { Tag = "Sidebar" })%>
  ViewContext.ViewData["tag"]
  ```
MVC execution life cycle (EPiServer)

Incoming request

IIS

UrlRoutingModule

Matching static file on disk

IIS directly served

RouteTable.Routes

to handler specified by route

EPiServer route handler

MvcRouteHandler

CORE ROUTING
MVC execution life cycle (EPiServer)

EPiServer route handler

MvcRouteHandler

Controller factory

Controller

Instanciate controller based on routing info

Action Method

Filters can kick in at many possible points here

ActionResult

Directly write HTTP response
Additional reading

- [http://www.asp.net/mvc/](http://www.asp.net/mvc/) Microsoft ASP.NET MVC home page

- [NerdDinner (http://nerddinner.codeplex.com/)](http://nerddinner.codeplex.com/) shows the very basics of ASP.NET MVC also the usage of OpenID, Bing Maps, Twitter Integration etc.

- And of course, the EPiServer MVC templates