Optimizing for Speed: Advanced JS/CSS Management in DNN 6.1

Ian Robinson

10/24/2011

Why We're Here

- Performance is a feature *(an important one)*
- Fast sites lead to satisfied users
- DotNetNuke is largely optimized on the server side, was not so much on the client side

Meeting Goals

- You should understand
  - The client resource management "problem domain"
  - What the ClientDependency Framework is and how to use it
  - The details of the solution in DNN 6.1
  - The development strategies the new API enables

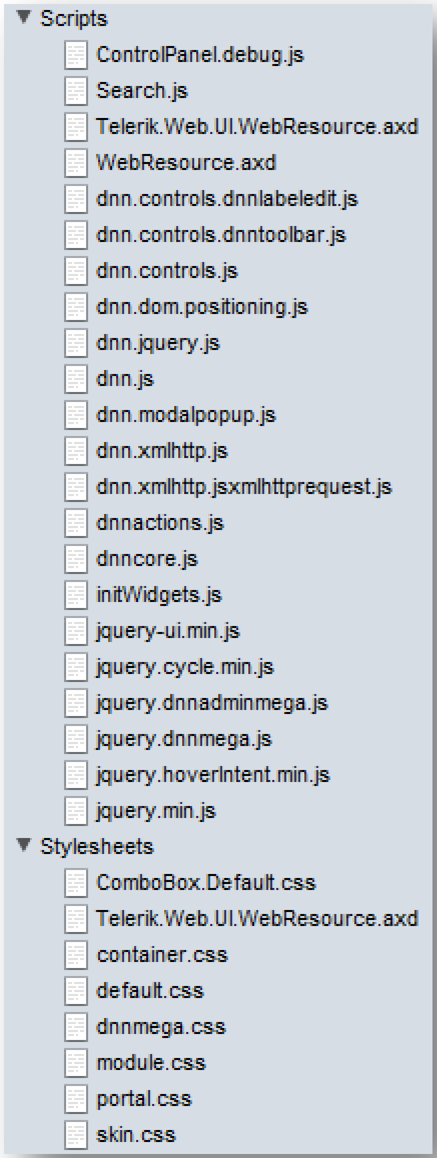
Client Side Performance

***80%** of the end-user response time is **spent on the front-end**. Most of this time is tied up in **downloading all the components** in the page: images, **stylesheets, scripts**, Flash, etc. **Reducing the number of components** in turn reduces the number of HTTP requests required to render the page. **This is the key to faster pages**.*

-Yahoo! Exceptional Performance Team

DotNetNuke 6 – Resources Overview

- Clean install, home page
  - ➢ unauthenticated
    - 6 CSS Files
    - 13 JavaScript Files
  - ➢ Logged in as host
    - 8 CSS Files
    - 22 JavaScript Files



▼ Scripts
- ControlPanel.debug.js
- Search.js
- Telerik.Web.UI.WebResource.axd
- WebResource.axd
- dnn.controls.dnnlabeledit.js
- dnn.controls.dnntoolbar.js
- dnn.controls.js
- dnn.dom.positioning.js
- dnn.jquery.js
- dnn.js
- dnn.modalpopup.js
- dnn.xmlhttp.js
- dnn.xmlhttp.jsxmlhttprequest.js
- dnnactions.js
- dnncore.js
- initWidgets.js
- jquery-ui.min.js
- jquery.cycle.min.js
- jquery.dnnadminmega.js
- jquery.dnnmega.js
- jquery.hoverIntent.min.js
- jquery.min.js

▼ Stylesheets
- ComboBox.Default.css
- Telerik.Web.UI.WebResource.axd
- container.css
- default.css
- dnnmega.css
- module.css
- portal.css
- skin.css

Goals for Improvement

- Reduce the file size of each resource
- Only deliver a resource that is needed
- Combine resources into as few as possible

Client Resource Management: Key Characteristics

- Resource Registration API
  - Request a JS or CSS resource be loaded
- File combination
  - Combine all requests of a given type into one file
- Caching / Persistence
  - Cache the combined file / save it to disk
- Reuse
  - Reuse cached files across pages if appropriate
- Versioning
  - Allow for cache busting based on versioning

Client Dependency Framework

- Open Source Framework
- Microsoft Public License (Ms-PL)
- Originally released Early 2010
- Supports MVC & WebForms
- Used in Umbraco
- **Meets all key characteristics on the previous slide**

JavaScript and CSS Registration

Step 1: Resource Registration

- Script Loader on page
- Register in code

```
var clientDependencyLoader = (ClientDependencyLoader)page.FindControl("Loader");
clientDependencyLoader.RegisterDependency(styleSheet, ClientDependencyType.Css);
```

- Or register in markup

```
<%@ Register Namespace="ClientDependency.Core.Controls" Assembly="ClientDependency.Core" TagPrefix="CD" %>
<CD:JsInclude runat="server" FilePath="~/Resources/Shared/Scripts/jquery/jquery.hoverIntent.min.js" />
<CD:JsInclude runat="server" FilePath="~/Portals/_default/Skins/DarkKnight/jquery.cycle.min.js" />
<CD:CssInclude runat="server" FilePath="/Portals/_default/Skins/DarkKnight/DNNMega/dnnmega.css" />
```

Resource Registration w/ DNN API

- Wrapped script loader control in Default.aspx
- Register in code using DNN API

```
ClientResourceManager.RegisterScript(this.Page, "~/Resources/Shared/Scripts/jquery/jquery.tmpl.js");
```

- Or register in markup using wrapped controls

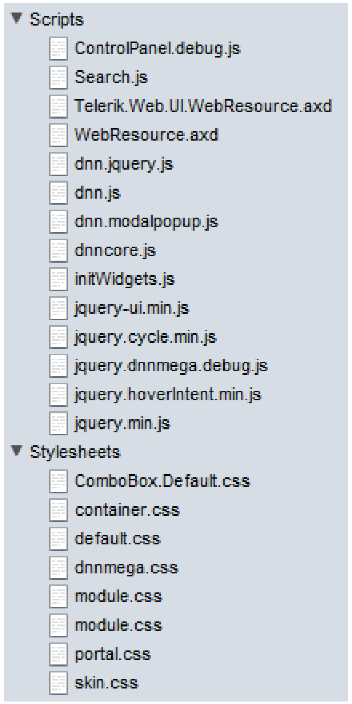
```
<%@ Register TagPrefix="dnn" Namespace="DotNetNuke.Web.Client.ClientResourceManagement" Assembly="DotNetNuke.Web.Client" %>
```

```
<dnn:DnnJsInclude runat="server" FilePath="~/Resources/Shared/Scripts/jquery/jquery.hoverIntent.min.js" />
```

DNN 6.1 w/ Client Dependency

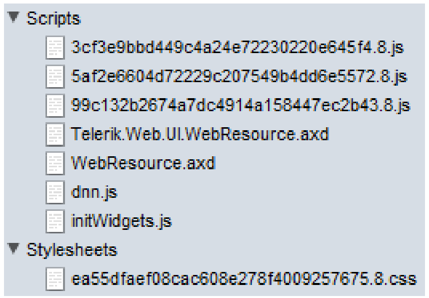
- Home page, clean install
  - ➢ **Unauthenticated**
    - Debug
      - 8 CSS Files
      - 14 JS Files
      - **22 Total**
    - Release
      - 1 CSS Files
      - 7 JS Files
      - **8 Total**
- **14 Fewer Requests**

debug="true"



```
▼ Scripts
    ControlPanel.debug.js
    Search.js
    Telerik.Web.UI.WebResource.axd
    WebResource.axd
    dnn.jquery.js
    dnn.js
    dnn.modalpopup.js
    dnncore.js
    initWidgets.js
    jquery-ui.min.js
    jquery.cycle.min.js
    jquery.dnnmega.debug.js
    jquery.hoverIntent.min.js
    jquery.min.js
▼ Stylesheets
    ComboBox.Default.css
    container.css
    default.css
    dnnmega.css
    module.css
    module.css
    portal.css
    skin.css
```

debug="false"

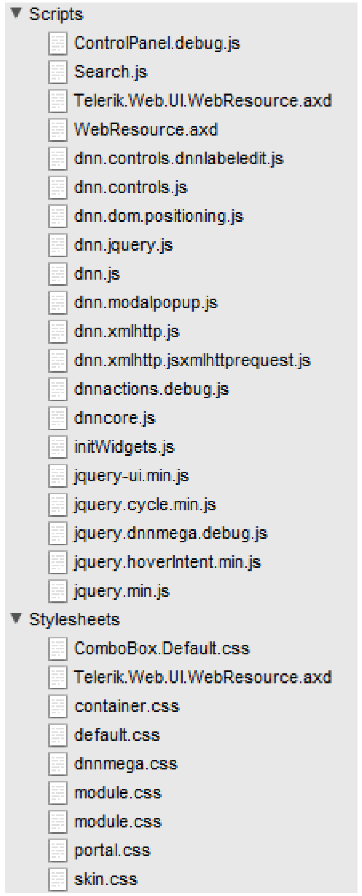


```
▼ Scripts
    3cf3e9bbd449c4a24e72230220e645f4.8.js
    5af2e6604d72229c207549b4dd6e5572.8.js
    99c132b2674a7dc4914a158447ec2b43.8.js
    Telerik.Web.UI.WebResource.axd
    WebResource.axd
    dnn.js
    initWidgets.js
▼ Stylesheets
    ea55dfaef08cac608e278f4009257675.8.css
```

DNN 6.1 w/ Client Dependency

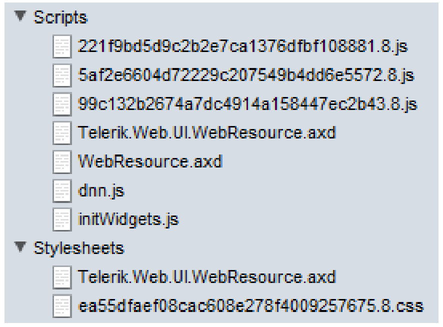
- Home page, clean install
  - ➢ **Logged in as Host**
    - Debug
      - 9 CSS Files
      - 20 JS Files
      - **29 Total**
    - Release
      - 2 CSS Files
      - 7 JS Files
      - **9 Total**
- **20 Fewer Requests**

debug="true"



```
▼ Scripts
    ControlPanel.debug.js
    Search.js
    Telerik.Web.UI.WebResource.axd
    WebResource.axd
    dnn.controls.dnnlabeledit.js
    dnn.controls.js
    dnn.dom.positioning.js
    dnn.jquery.js
    dnn.js
    dnn.modalpopup.js
    dnn.xmlhttp.js
    dnn.xmlhttp.jsxmlhttprequest.js
    dnnactions.debug.js
    dnncore.js
    initWidgets.js
    jquery-ui.min.js
    jquery.cycle.min.js
    jquery.dnnmega.debug.js
    jquery.hoverIntent.min.js
    jquery.min.js
▼ Stylesheets
    ComboBox.Default.css
    Telerik.Web.UI.WebResource.axd
    container.css
    default.css
    dnnmega.css
    module.css
    module.css
    portal.css
    skin.css
```

debug="false"



```
▼ Scripts
    221f9bd5d9c2b2e7ca1376dfbf108881.8.js
    5af2e6604d72229c207549b4dd6e5572.8.js
    99c132b2674a7dc4914a158447ec2b43.8.js
    Telerik.Web.UI.WebResource.axd
    WebResource.axd
    dnn.js
    initWidgets.js
▼ Stylesheets
    Telerik.Web.UI.WebResource.axd
    ea55dfaef08cac608e278f4009257675.8.css
```

Into the Wild

- DNN Core Strength: custom & third party components
- But, usage means resource requests often grow
- Consider these (unauthenticated. As of 8/16/2011)
  - R2integrated.com: 30+ JS files and 5 CSS files
  - DataSprings.com: 18 JS files and 11 CSS files
  - DotNetNuke.com: 16 JS files and 12 CSS files
  - EngageSoftware.com: 23 JS files and 9 CSS files
  - Mybrantford.ca: 17 JS files and 9 CSS files
  - Dreamslider.net: 16 JS files and 9 CSS files

A New Development Approach

Step 2: A New Development Approach

- Freed up to structure as necessary
  - ➢ No longer shove all styles into one module.css file
  - ➢ Can break it out into separate files and request as needed
    - CssInclude(‘base.css’)
    - CssInclude(‘ui-widgets.css’)
    - CssInclude(‘gallery.css’)
  - ➢ Same with JS files

Implementation Details

Implementation Details

- Reference Assembly
- Additional web.config section
- Composite files stored in App_Data/ClientDependency
- DNN wrapper API methods
  - RegisterStyleSheet already exists
  - RegisterScript?
  - Wrapper control for user in skins and other controls
- WebUtility and WebControls assemblies need updating
- CDN integration
- Load ordering scheme for both JS & CSS

The New API

- DotNetNuke.Web.Client Assembly
  - ➢ RegisterStyleSheet methods
  - ➢ RegisterScript methods
  - ➢ DnnCssInclude
  - ➢ DnnJsInclude

File Combination

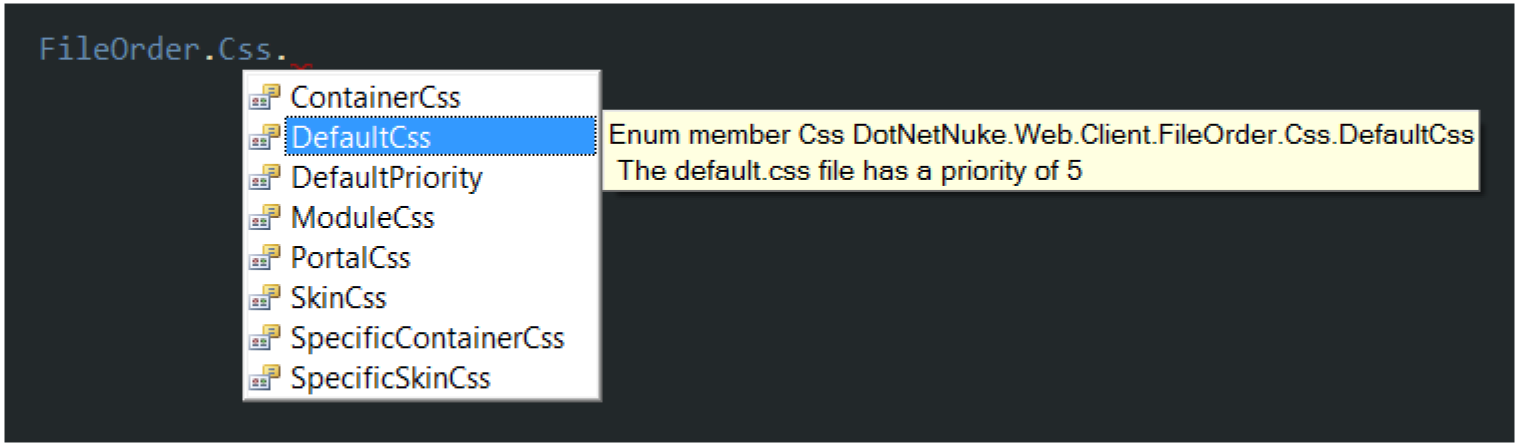
- Duplicates removed based on path/filename
- Combined into one file
- Absolute external URLS (JS & CSS) such as CDN requests are requested separately
- An xml file map is kept on the server
- The dynamic URL is a hash of those file path/names

Location in the Document

- Provider model
- Provider dictates where it is rendered
- Out of the box:
  - LoaderControlProvider
  - PageHeaderProvider
  - LazyLoadProvider
- DNN Provides:
  - DnnBodyRenderProvider
  - DnnFormBottomRenderProvider

File Ordering

- Integer based relative priority
- DotNetNuke core file order enumeration (spaced by 5)



```
FileOrder.Css.
```

ContainerCss
DefaultCss
DefaultPriority
ModuleCss
PortalCss
SkinCss
SpecificContainerCss
SpecificSkinCss

Enum member Css DotNetNuke.Web.Client.FileOrder.Css.DefaultCss
The default.css file has a priority of 5

Caching and Persistence

- ASP.NET Output Caching
  - MSDN: *"On subsequent requests, the page or user control code is not executed; the cached output is used to satisfy the request."*
- Stored on disk for persistence across application restarts
  - Pulled from disk (not rebuilt) and put in cache

VERSIONING

- Integer based version number
- Stored in web.config
- Forces a fresh rebuild of the files
- A variety of ways to increment
  - Install an extension
  - Clear the cache
  - Save Portal.css
  - Perform an upgrade