Umbraco Courier 2.5

Installation and configuration

Per Ploug

11/9/2011

Table of Contents

Introduction

This document describes the available configuration options for Courier 2.5 as well as notes and comments on working with of the build-in item providers

Revision History

- Version 1, 8/11 2011 – Intial Outline and notes – Per Ploug

Installing Courier 2 using a package

1. Open the embrace repository from the developer section in umbraco
2. Browse to the Umbraco PRO category
3. Click Umbraco Courier, and choose install, follow directions on screen
4. At the end of the install, you are prompted for a **location**, enter the domain of the other site you want to use with courier, e.g.: domain.com, www.sample.com or internaldev
5. Courier expects that the user you are deploying content with**, has the same credentials on all locations**, if this is not case, either change credentials or read the chapter on configuring locations in this guide
6. If you have bought a license, copy the .lic file to the website's /bin folder or use the dashboard to configure your license
7. That's it

Updating Courier 2.0 to 2.5

If you already have Courier 2.0 installed, follow these steps to upgrade to 2.5:

1. Backup your `/config/courier.config` file
2. Copy the contents of /bin, /config and /umbraco from the manual update zip, to the root of your site
3. Delete the DLL: `Castle.DynamicProxy2.dll` and it's .pdb file from the /bin folder
4. Update the `/config/courier.config` file with settings from your backed-up one.
    a. Ensure your `<repositories>` node is copied over
    b. You resource filters
    c. And the itemdataresolver settings
    d. In 90% of all installations, a default configuration is used, so `<repositories>` node is the crucial one
5. Your done

Manual Courier installation

If for some reason the package installation fails or due to permissions or other reasons is not an option on your system, we provide a manual installation process.

To manually install, download the manual install package from umbraco.com or one of the hotfix releases from nightly.umbraco.org

The manual process consists of several files:

- **The folders /bin, /config and /umbraco**
  Folders containing the application files for Courier 2
- **/sql folder**
  Folder containing install and uninstall sql files. For install there are variations for each of the 3 supported databases
- **/sql/uninstall folder**
  Contains the single uninstall sql script, which will remove custom courier table as well as the any courier app entries, it will not remove any files

Installing the files

Simply unzip the /bin, /config and /umbraco folders to the root of your website, the archive follows the structure needed to place the files correctly. **Notice**: the archive assumes your umbraco director is located at /umbraco. If not you will need to move those files manually to the right location.

If you have purchased Umbraco Courier, you can download a license file on umbraco.com. This license file must be placed in the websites /bin directory to be registered.

Installing the database

To install the database we need to execute a sql script against the database umbraco is installed on. Courier currently only supports SQL server 2005 and 2008.

- Open Microsoft Sql Server Management Studio and connect to your database.
- Right click your umbraco database and choose "new query"
- Open the /sql/install folder and pick the appropriate sql files. If you use MS Sql, you pick “app.sql” and “create.sql” if not, you pick the files with the correct database name in them.
- Copy the contents of sql files to the query window
- Execute the script
- If any errors are displayed, check your permissions. The install script requires database owner access, as it creates new tables.

Uninstalling the database

In case of error you can use the /sql/uninstall/uninstall.sql file to remove all courier tables. Follow the same procedure as the one describing "installing the database"

Courier 2.5 core configuration

Courier 2.5 comes with sensible defaults, but in special cases, you might need to add or modify core settings.

Core settings are configuration options which covers the entire Courier application, no matter if it runs from umbraco, or a desktop client, all core settings resides inside the `<settings>` xml node in the /config/courier.config file. With the below settings, simply copy the sample xml inside the `<settings>` node and Courier will register it.

Debugging

Settings related to remote connections performed by Courier

Debugmode

Enables logging to the /app_data/courier/log.txt file

Xml	Default value
`<debugMode>true</debugMode>`	false

Map graphs

If enabled, Courier will generate a blumind compatible mindmap after each extraction to map dependencies

Xml	Default value
`<mapGraphs>true</mapGraphs>`	false

Urls and locations

Settings for folders and connections used by Courier

Root folder

The root folder containing all Couriers data. This folder needs changed, if Courier runs outside of the standard umbraco webcontext. (Consult the Courier Contrib Console application for source code http://couriercontrib.codeplex.com)

Xml	Default value

`<paths>`
`  <root>~/path/to/courier</root>`
`</paths>`
	~/app_data/courier

Revisions folder

Specifies the folder within the root folder, which holds each indivual revision folder.

Xml	Default value

`<paths>`
`  <revisions>/folder</revisons>`
`</paths>`
	/revisions

Masterpages folder

The Sql connection to the SQL database Courier should use. Notice this is not nessary as long as the repository pattern is used, as that will then be handled by the Umbraco website data is pulled/pushed from.

Xml	Default value

`<paths>`
`  <masterPages>/folder</masterPages>`
`</paths>`
	~/masterpages (umbraco default)

Database connection

Xml	Default value

`<databaseConnectionString>`
`  DATABASE=yahahdasd;USER ID=etc`
`</databaseConnectionString>`
	The umbraco database connection in the web.config

Use short courier file names

In case of too long paths, shorten file names

Xml	Default value

`<enableShortFileNames>`
`  false`
`</enableShortFileNames>`
	true

Webservices and transfers

Settings related to remote connections performed by Courier, and transfers of resources and item data.

Webservice Url

The url to default Courier Webservice repository .asmx file.

Xml	Default value

`<webservicesPath>`
`  /repository.asmx`
`</webservicesPath>`
	/umbraco/plugins/courier/webservices/repository.asmx

Timeout

Milliseconds before a webclient connection times out.

Xml	Default value
`<timeout>30000</timeout>`	30000

Base64 encoding

Encode all resources and items as bas64 to avoid illegal characters

Xml	Default value

`<disableBase64Encoding>`
`  true`
`</disableBase64Encoding >`
	False

Strip resources from courier files

Strip the raw byte data from .courier files before transfering

Xml	Default value

`<stripResourcesFromCourierFiles>`
`  false`
`</stripResourcesFromCourierFiles>`
	true

Configuring Item Providers

Item providers handles each individual kind of object or data in Umbraco. There are item providers for such things as document types, documents, stylesheets, datatypes, media and so on. Each item provider handles its items in a specific way, and comes with different kinds of configuration options.

Document types

By default the provider adds a dependency only to the default template, and does not transfer document types in the selected types “structure” setting. Also, it filters datatype dependencies by the configured list of ignored datatypes

```
<documentTypeItemProvider>
  <!--
  Include all avaiable templates as dependencies, if false, only the current standard template is included
  -->
  <includeAllTemplates>false</includeAllTemplates>

    <!-- By default we won't add the built-  in datatypes as dependencies,
         if needed, they can be removed from the list below
         Only datatypes which are installed as standard, and does not have any settings are ignored
         to add, find the datatype in the umbracoNode table and copy its uniqueId value to a node below-->
    <ignoredDataTypes>
        <add key="contentPicker">A6857C73-D6E9-480C-B6E6-F15F6AD11125</add>
        <add key="textstring">0CC0EBA1-9960-42C9-BF9B-60E150B429AE</add>
        <add key="textboxmultiple">C6BAC0DD-4AB9-45B1-8E30-E4B619EE5DA3</add>
        <add key="label">F0BC4BFB-B499-40D6-BA86-058885A5178C</add>
        <add key="folderbrowser">FD9F1447-6C61-4A7C-9595-5AA39147D318</add>
        <add key="memberpicker">2B24165F-9782-4AA3-B459-1DE4A4D21F60</add>
        <add key="simpleeditor">1251C96C-185C-4E9B-93F4-B48205573CBD</add>
        <add key="truefalse">92897BC6-A5F3-4FFE-AE27-F2E7E33DDA49</add>
        <add key="contentpicker">A6857C73-D6E9-480C-B6E6-F15F6AD11125</add>
        <add key="datepicker">5046194E-4237-453C-A547-15DB3A07C4E1</add>
        <add key="datepickerWithTime">E4D66C0F-B935-4200-81F0-025F7256B89A</add>
        <add key="numeric">2E6D3631-066E-44B8-AEC4-96F09099B2B5</add>
    </ignoredDataTypes>
</documentTypeItemProvider>
```

Documents

Documents does not include parent or child documents by default, unless it is set during the packaging of that item. However, Courier does link the document other documents found in content pickers or through relations

```
<documentItemProvider>
        <includeParents>false</includeParents>
</documentItemProvider>
```

Media types

Media types are

Media

The media provider does not include children by default, but can be configured to include children (incase a folder is picked from a media picker for example)

```
<mediaItemProvider>
       <!-- Changed to false in 2.1.1 -->
       <includeChildren>false</includeChildren>
</mediaItemProvider>
```

Relations

Relations are picked up automaticly on media and documents, as well as the related items responsible provider. However **RelationTypes are not automaticly added**, you will need to do this manually on each instance.

Tags

Tags are picked up via the tag control, included in the core. It will automaticly move any missing tags and add these to the documents or media nodes.

Configuring item data resolvers

Courier comes with a collection of data resolvers, which are responsible for picking up different types of data and add meaning to this data. An exemple of this is examing content picker data and adding the picked documents as dependencies of the selected document.

The configuration format is the same for all the resolvers, in this case key refers to the name of the datatype (only used for identification in the config file) and the data type GUID which refers to the specific control which the upload control renders.

```
<add key="Upload">5032a6e6-69e3-491d-bb28-cd31cd11086c</add>
```

Files

Resolves all properties with datatypes which contains direct links to files such as the upload control and the ucomponents filepicker

```
<files>
    <!-- add new datatype elements for data types that stores files as a path ex: /meda/223/file.png -->
    <add key="Upload">5032a6e6-69e3-491d-bb28-cd31cd11086c</add>
    <add key="Ucomponents-Filepicker">318a9c2e-3966-4979-8c1d-575c5d5f669b</add>
</files>
```

Content pickers

Resolvers datatypes which lists document ids as either a single ID or a comma separated list of values. Courier supports all uComponents pickers out of the box.

```
<contentPickers>
    <!-- add new datatype elements for data types that stores page ids (ex: "1242" or "1726,2362,2323") --
>
    <add key="contentPicker">158aa029-24ed-4948-939e-c3da209e5fba</add>
    <add key="ultimatePicker">cdbf0b5d-5cb2-445f-bc12-fcaaec07cf2c</add>

    <add key="Ucomponents-XpathCheckboxlist">d2d46927-f4f8-4b1b-add7-661cc09a0539</add>
    <add key="Ucomponents-XpathDropdownlist">57a62843-c488-4c29-8125-52f51873613e</add>
    <add key="Ucomponents-AutoComplete">31aa0d5c-f8e1-4cdc-a66e-c7f8c09498ef</add>
</contentPickers>
```

Media pickers

A media picker acts as a content picker, but refers to media items and uses the media Item provider to load them as dependencies. Courier supports the default media picker and DAMP 2.0 by default.

```
<mediaPickers>
    <!-- add new datatype elements for data types that stores media ids (ex: "1242" or "1726,2362,2323") -
->
    <add key="mediaPicker">EAD69342-F06D-4253-83AC-28000225583B</add>
    <add key="damp2">ef94c406-9e83-4058-a780-0375624ba7ca</add>
</mediaPickers>
```

Keyvalue prevalue editors

Supports all datatypes which uses the keyvalue prevalue editor to save configuration values. This specific setting filters by the full class name of the prevalue editor, not a datatype guid.

```
<keyValuePrevalueEditors>
    <!-- Prevalue editors that store values as a key value pair in the built-
in umbracp prevalue storage, identified by their full class-name -->
    <add key="KeyValuePrevalueEditor">umbraco.editorControls.KeyValuePrevalueEditor</add>
</keyValuePrevalueEditors>
```

Local links

Support for the LocalLink:1929 format, the configuration specifies which datatypes CAN contain such links

```
<localLinks>
    <!-- Propertytypes that CAN contain locallinks (like the ones inserted with TinyMCE) -->
    <add key="TinyMCE3">5e9b75ae-face-41c8-b47e-5f4b0fd82f83</add>
    <add key="TextboxMultiple">67db8357-ef57-493e-91ac-936d305e0f2a</add>
    <add key="Textstring">ec15c1e5-9d90-422a-aa52-4f7622c63bea</add>
    <add key="Simple Editor">60b7dabf-99cd-41eb-b8e9-4d2e669bbde9</add>
</localLinks>
```

Macros

Support for resolving macros and their macro parameters inside of datatype values. This configuration specifies which datatypes can contain macro elements

```
<macros>
    <!-- Propertytypes that CAN contain macro mark-up (like the ones inserted with TinyMCE) -->
    <add key="TinyMCE3">5e9b75ae-face-41c8-b47e-5f4b0fd82f83</add>
    <add key="TextboxMultiple">67db8357-ef57-493e-91ac-936d305e0f2a</add>
    <add key="Textstring">ec15c1e5-9d90-422a-aa52-4f7622c63bea</add>
    <add key="Simple Editor">60b7dabf-99cd-41eb-b8e9-4d2e669bbde9</add>
</macros>
```

Configuring macro propertytype resolvers

Courier supports resolving macros parameters, inserted inside of the <umbrac:macro> tags. Courier examines parameters it knows can contain document or media IDs. This configuration specifies which macro property types are able to contain node ids, to either media or documents

```
<macroPropertyTypeResolvers>
    <contentPickers>
        <!-- Macro Property Types, that store Content IDs, to link to media or content -->
        <add key="Media Current">mediaCurrent</add>
        <add key="Content Subs">contentSubs</add>
        <add key="Content Random">contentRandom</add>
        <add key="Content picker">contentPicker</add>
        <add key="Content tree">contentTree</add>
        <add key="Content All">contentAll</add>
    </contentPickers>
</macroPropertyTypeResolvers>
```

If a 3rd party macro propertytype, which could contain a document or media ID is added to your site, you will need to configure the macroPropertyTypeResolvers to detect new ids, which will then be converted to Guids and back again during transfers.

Configuring Resource filtering

Courier comes with the ability to use wildcard filtering against files included in packaged revisions. This means that you can configure Courier to ignore certain files or folders during packaging and therefore avoid adding standard dlls, images and other files that wouldn't need to be transferred at any time.

Use * as a wildcard to do partial matching against certain types or all files in certain folders.

The filter is triggered by all providers, during all packaging operations

```
<resources>
    <!-- files which should not added at any time-->
    <ignore>
        <add>/bin/*.pdb</add>
        <add>/bin/*.xml</add>
        <add>/bin/AjaxControlToolkit.dll</add>
        <add>/bin/Antlr3.Runtime.dll</add>
        <add>/bin/App_Browsers.dll</add>
        <add>/bin/App_global.asax.dll</add>
        <add>/bin/businesslogic.dll</add>
        <add>/bin/Castle.*</add>
        <add>/bin/ClientDependency.Core.dll</add>
        <add>/bin/cms.dll</add>
        <add>/bin/controls.dll</add>
        <add>/bin/CookComputing.XmlRpcV2.dll</add>
        <add>/bin/Examine.dll</add>
        <add>/bin/FluentNHibernate.dll</add>
        <add>/bin/htmlagilitypack.dll</add>
        <add>/bin/ICSharpCode.SharpZipLib.dll</add>
        <add>/bin/interfaces.dll</add>
        <add>/bin/Iron*.dll</add>
        <add>/bin/log4net.dll</add>
        <add>/bin/Lucene.Net.dll</add>
        <add>/bin/Microsoft.*.dll</add>
        <add>/bin/MySql.Data.dll</add>
        <add>/bin/NHibernate.*</add>
        <add>/bin/RazorEngine.*.dll</add>
        <add>/bin/System.Web.*</add>
        <add>/bin/TidyNet.dll</add>
        <add>/bin/Umbraco.Courier.*</add>
        <add>/bin/umbraco.DataLayer.dll</add>
        <add>/bin/UrlRewritingNet.UrlRewriter.dll</add>
        <add>/bin/umbraco.dll</add>
    </ignore>
</resources>
```

Ignoring providers

If there is an issue with a specific provider, no matter what type of provider. You can turn it off by ignoring it.

This is done by adding its full namespace and class to the configuration, you can ignore any item provider, data resolver, repository provider or any other functionality that's loaded through Couriers provider model.

```
<ignore>
    <!-- Ignore the lucene indexer -->
    <add>Umbraco.Courier.DataResolvers.Events.UpdateLuceneIndexes</add>
    <!-- ignore all ucomponents data resolvers -->
    <add>Umbraco.Courier.uComponents.*</add>
    <!--<add>my.namespace.*</add>-->
</ignore>
```

Configuring Repository Providers

A repository provider encapsulates the logic needed to connect to remote location. API and configuration wise, these are known as Repository, but in the UI they are always refered to the more common term “locations”.

Current available repository providers

- **Courier Webservice**
  Gives you access to another umbraco site running Courier 2. Enables you to remote extract and package contents
- **Network share**
  Enables you to save revisions to a folder on your local machine or on a network share.
- **Subversion** (experimental)
  Can connect to a subversion repository and Pull revision data into your Umbraco instance, however you cannot send changes back to it currently

Network Share

- **Type**: NetworkShareProvider
- **Guid**: e0472598-e73b-11df-9492-0800200c9a66
- **Full name**: Umbraco.Courier.Providers.RepositoryProviders. NetworkShareProvider

The network share repository can transfer items back and forth to a local or network directory where the asp.net application has access. To add a network share repository, add the following to the courier.config under “repositories”

Configuration XML

```
<repository name="Revisions" alias="revisions" type="NetworkShareProvider" visible="true">
    <path>C:\path\to\repository</path>
</repository>
```

Settings

- **Path**: Contains the fully valid path to the directory where revisions should be stored

Courier Webservice

- **Type**: CourierWebserviceRepositoryProvider
- **Guid**: e0472596-e73b-11df-9492-0800200c9a66
- **Full name**: Umbraco.Courier.Providers.RepositoryProviders. CourierWebserviceRepositoryProvider

The courier webservice provider can connect any other website running umbraco, with courier installed as a repository. It is possible to transfer items back and forth using the http protocol. To install, add the following to your courier.config under “repositories”.

Configuration XML

```
<repository name="Live" alias="1" type="CourierWebserviceRepositoryProvider" visible="true">
    <url>http://cws.local</url>
    <user>0</user>

    <login>login</login>
    <password>pass</password>
    <passwordEncoding>Clear|Hashed</passwordEncoding>
</repository>
```

Settings

- **Url**: url to the website where the other instance is accessible
- **User**: The ID of the umbraco user you want to use to authenticate with
- **Login**: (optional) Instead of user ID you can set a specific login name
- **Password**: (optional) Instead of user ID, you can set a specific password
- **PasswordEncoding**: (optional) specify if Courier should keep password clear or Hashed to match your target repository
    - **Note**: Courier alwas encrypts credentials. Encoding is more to do with how Umbraco stores user passwords.

Custom Umbraco Membership Providers

If you use a custom umbraco membership provider, you must always specify the login and password on the repository configuration. And set passwordEncoding to Clear.

Subversion

- **Type**: SubversionRepository
- **Guid**: e0474ca8-e73b-11df-9492-0800200c9a66
- **Full name**: Umbraco.Courier.SubversionRepository. SubversionRepository

Provider can connect to a subversion repository

Configuration XML

```
<repository name="SVN Repo" alias="svnRepo" type="SubversionRepository" visible="true">
    <url>http://cws.local</url>
    <login>login</login>
    <password>pass</password>
</repository>
```

Settings

- **Url**: url to subversion repository
- **Login**: Your subversion username
- **Password**: Your subversion password