NEXTpage®

NEXTpage® content@®

ADMINISTRATOR'S GUIDE

Document Information and Copyright Notice

Document Name	Version	Date
NextPage Content@ Administrator's Guide	1.0	September 2002

Copyright Notice

NextPage, Inc.

3125 West Executive Park Way

Lehi, UT 84043

U.S.A.

NextPage Content@ Publisher Administration

The NextPage Content@ Publisher Administrators Guide decribes topics of interest to an adminstrator of a NextPage Content@ Publisher implementation within a corporate IS infrastructure. The topics covered include the following:

- Using the System overview for learning more about how NextPage Triad fits together.
- Administering your system to make use of the MakeGen program so you can publish content to NXT 3.
- Maintaining your system by using such things as a backup and recovery strategy.
- Getting support for problems and troubleshooting.
- Finding documentation that can provide more information about specific features and functions.

System Overview

NextPage Triad exposes data from the Content@ Content Management System through the NXT 3 e-Content Platform to create a complete authoring, management, and online publishing solution for global workforces. In essence, periodic snapshots of Content@ repositories are published to NXT 3 using a scriptable, command line interface. This focused coupling of the two systems at the data level maintains the security, reliability, and scalability of each without sacrificing the interoperability of both.

The NXT 3 e-Content Platform is uniquely qualified to bridge content from multiple sources, including the Content@ Content Management System, into a single Enterprise Content Network (ECN). Unlike UI-level integrations, this ECN combines information at the data level, allowing users to not only see data in a consistent form, but to work with data in a consistent way. Marrying the content-networking strengths of NXT 3 with the content-management features of Content@ creates a best-of-breeds integration called NextPage Triad.

There are three main pieces to this content management and publishing solution. The NextPage Content@ package covers the second and third pieces of the following items:

1. **NextPage NXT 3** links independently-maintained data and documents from across the enterprise into a unified ECN. This ECN acts as a “virtually centralized” collection of all an enterprise’s data, fully indexed and searchable, without physically aggregating the data into a single location. Employees and clients therefore see a unified table of contents with the latest data, taken directly from the source and without replication, while departments maintain local control over their information. This content is organized into what we call a "content collection". This content collection is a single, redistributable file comprising a self-contained copy of the hierarchical data from Content@ or other sources. Content collections can contain any type of document, as well as considerable internal folder structure. Generally, you create a content collection that contains several documents or topics. However, you can also create a single content collection that contains a single published document, especially if that document is very long and contains considerable internal structure.

   NextPage Solo further extends the reach of the ECN by using a client-side web-based application for providing read access to a local copy of ECN data. Essentially, each time Solo connects to the ECN it retrieves a “sync file” containing updates to its content collections. With local copies of the content collections and the latest sync files, Solo can offer many of the NXT 3 e-Content Platform features to users operating away from the network.
2. **XyEnterprise Content@** enforces workflows, maintains version control, and publishes documents in a wide variety of formats. Its choice of Windows or web-based user and management interfaces and tight integration with traditional authoring tools allow users to obtain the proper checks and approvals, track and audit changes over time, and assemble localized XML, HTML, PDF, and Microsoft Office documents.
3. **NextPage Content@ Publisher** includes utilities and samples for mapping between Content@ and NXT 3 data models, as well as components, scripts, and examples for integrating Content@ and NXT 3. These utilities and samples include the following:

Content@ MakeGen Configuration File is an XML file that describes what you want to publish.

Content@ MakeGen utility creates an XML file called a "makefile" that contains the instructions for how to create the content collection. MakeGen uses the provided document source extension (DSE) and the MakeGen configuration file to retrieve the appropriate content from Content@ .

NextPage Builder (***NPBuild*** utility) uses the makefile to create either a new content collection or an update file for an existing collection.

Script samples that you can use to help you automatically make those content collections available through the NXT 3 site (this is known as mounting a content collection).

These items comprise what we call the "build process", where you define what content you want to publish from Content@ and create a content collection. See "Administering Your System" in the *NextPage Content@ Administrator's Guide* for more information about this process.

Thus, you use XyEnterprise Content@ for managing and working with the source documents, you use NextPage Content@ Publisher to take the source documents from XyEnterprise Content@ and publish them to NXT 3 as content collections, and you use NextPage NXT 3 to show documents to users from one location regardless of where the original document is stored.

You typically install NextPage Content@ Publisher on a separate computer from the Content@ servers and the NXT 3 and Solo servers so as to not affect the server performance while creating or updating content. However, it is possible to install NextPage Content@ Publisher on the same computer as NXT 3, for cases where you want to have a testing environment.

Publishing Stages Overview

Assuming both NXT 3 and NextPage Content@ have been previously installed, as documented in the NextPage NXT 3 and NextPage Content@ packages, deploying NextPage Triad consists of three general stages:

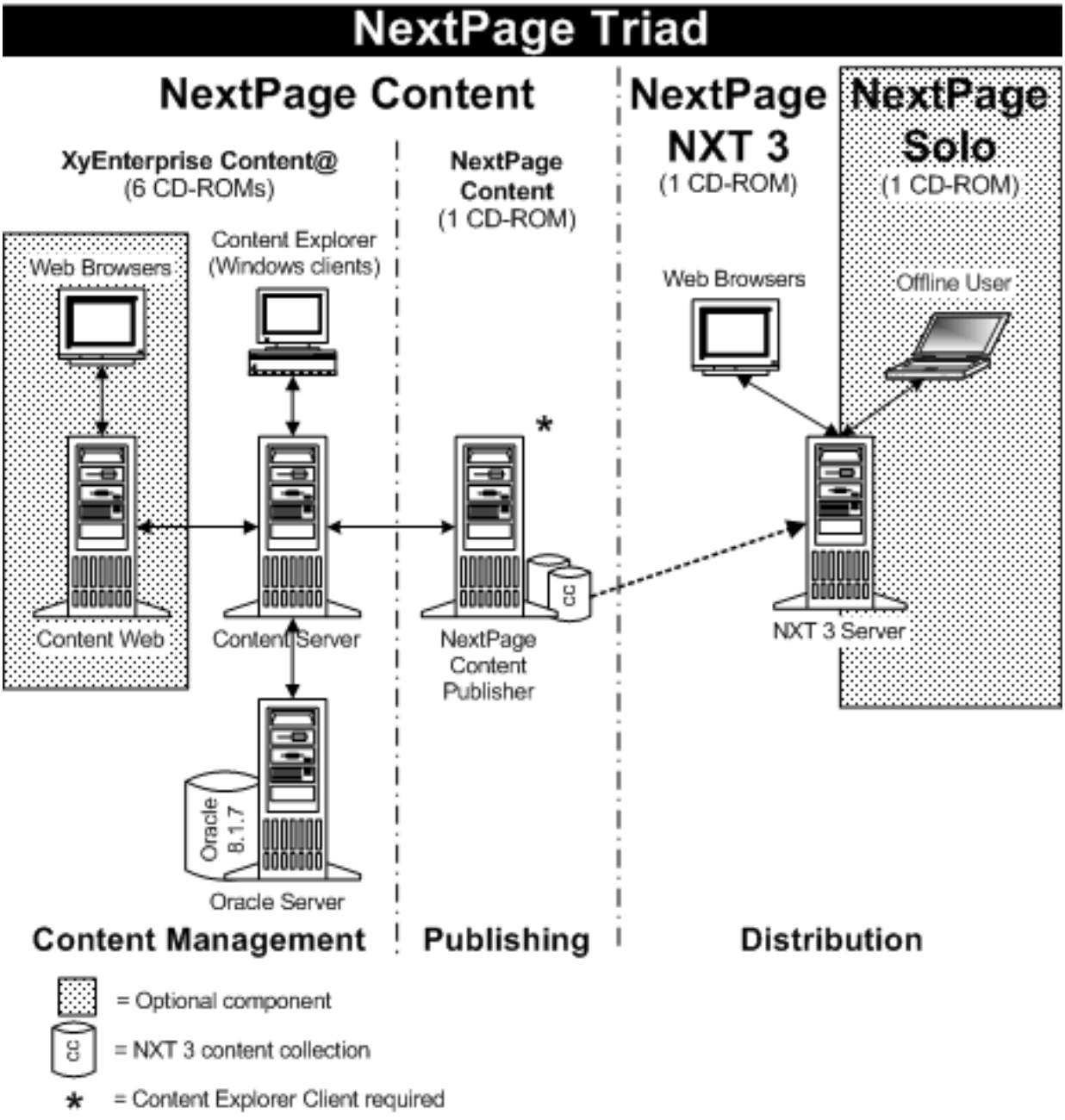
1. Mapping Content@ Data into NXT 3. Internally, Content@ decomposes documents and source data into reusable, interrelated information objects. NXT 3 maintains an internal content hierarchy, described by an XML-based configuration file. Publishing Content@ content to NXT 3 requires creating an XML "makefile" that specifies and maps Content@ objects and properties to NXT 3 data types and metadata. We provide tools and samples in the NextPage Content@ package to aid you in this process. See "Creating a Makefile" in the *NextPage Content@ Administrator's Guide*.
2. Building the Content Collection. You create the content collection using the NXT 3 NPBuild application, which you can run explicitly when you need it or you can create a script that runs the application on a regular basis. NPBuild reads the makefile created in the first stage, connects to a specified Content@ Server, and creates a content collection or an update of an existing content collection of the Content@ repository. If you also use Solo, you need to configure NPBuild to produce new update files containing

the changes from previous versions of the content collection. See "Solo Compatibility Guidelines" in the *NextPage Content@ Administrator's Guide* for more information on the necessary steps.

3. Mounting the Content Collection. Before users can access the newly-created content collection and synchronization files, you must mount the content collection onto an NXT 3 server. You can do this process manually, or create a script to perform it automatically at the end of the build process. See "Mounting a Content Collection" in the *NextPage Content@ Administrator's Guide*.

Environment Overview

NextPage Triad involves several different software packages. The following diagram shows a complete example of what you get with the NextPage Content@ package and how it fits in with the rest of the full solution.



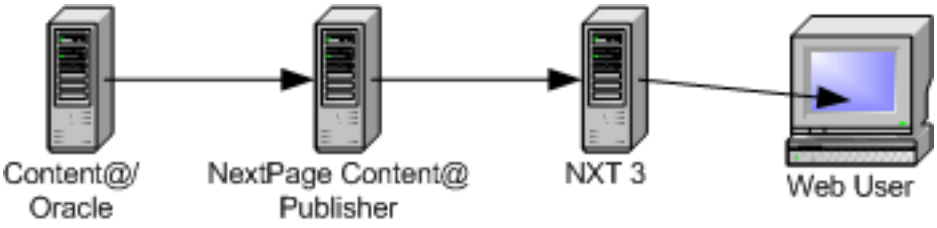
The NextPage Content@ package contains the XyEnterprise Content@ software, which allows you to import and work with content in a structured environment. NextPage Content@ also includes the NextPage Content@ Publisher software that makes it

possible to build content collections that you can then put on an NXT 3 site for online or offline users to access.

There are four main deployment options with NextPage Triad. The person who installed NextPage Content@ will be able to tell you which of the following deployments they used:

Deployment Option 1

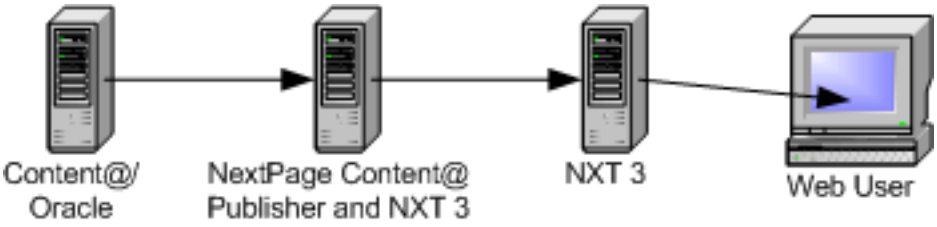
Deployment Option 1 is good for environments that do not need a test environment and that publish to just one production server.



The advantage to this deployment option is that it is a quick and easy process for moving from creation to publication. The disadvantage is that any problems with the content or with the publishing process can be immediately apparent to the users of the NXT 3 site.

Deployment Option 2

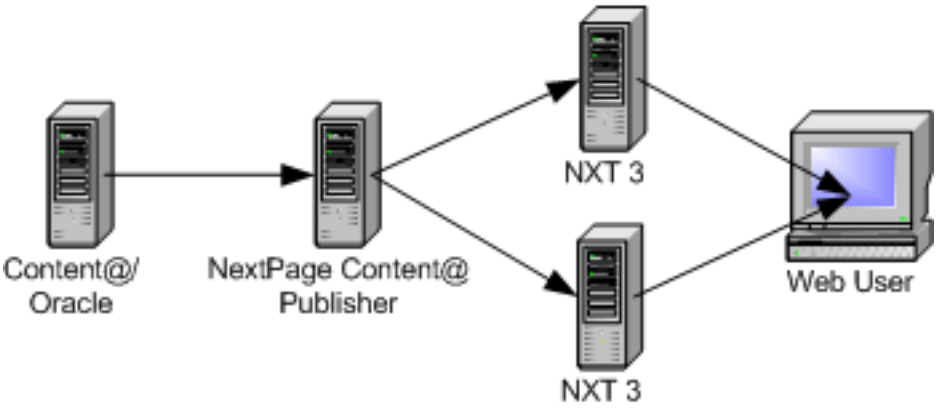
Deployment Option 2 is good for environments that need a test environment and that publish to just one production server.



The advantage to this deployment option is that you can test the publication before sending it out to the live production server for your users. The disadvantage is that departments have to share the same live NXT 3 server.

Deployment Option 3

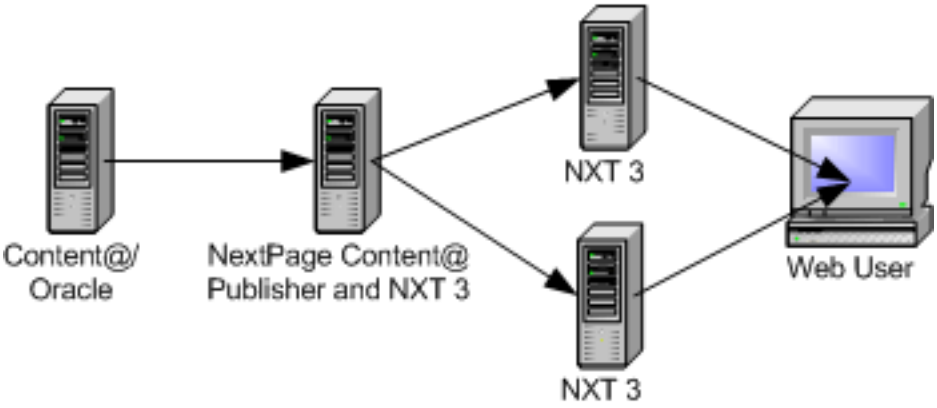
Deployment Option 3 is good for environments where each department hosts their own content on their own NXT 3 server or where the test environment server needs to be accessed by lots of users before going to a live production server.



The advantage to this deployment option is that it allows flexibility to each department about how they host their content. The disadvantage is that it is harder to enforce consistency across the various NXT 3 sites.

Deployment Option 4

Deployment Option 4 is good for environments that need a test environment and that publish different content to multiple production servers.



The advantage to this deployment option is that you can create build scripts that use some, all, or none of the same content and then publish that content to different NXT 3 servers. The disadvantage is that the test environment can become a bit confusing with all the different sites.

Administering Your System

This section describes how to work with NextPage Content@ Publisher to create content collections that you can mount on NXT 3.

Guidelines

Before you start creating content collections, you need to ensure that you understand and follow the guidelines for preserving links and for working with NextPage Solo.

- Link creation guidelines provides information about how to ensure that links you create in your source documents function correctly when you publish the document to NXT 3.
- Solo compatibility guidelines includes information that you need to know if your deployment uses Solo to provide published content to offline users.

Publishing Content to NXT 3

You use the NextPage Content@ Publisher MakeGen utility to help you publish content from Content@ to NXT 3. You need to complete each of the following tasks to publish content from Content@ to NXT 3:

1. Work with the existing configuration file or create a new XML configuration file that is consistent with the Content@ MakeGen DTD. You use the configuration file to describe what you want to publish.
2. Create a makefile by using the MakeGen utility, referencing your configuration file from step one.
3. Build a content collection by using the NPBuild utility and referencing your new makefile from step two. You can choose to create either a content collection or an update file to an existing content collection.
4. Mount the content collection or update file from step three in an NXT 3 site.

Link Creation Guidelines

This document describes how to create links in content managed through Content@ so that the links are valid when you publish that content to an NXT 3 content collection. A link, for the context of this document, is a reference from a document to a resource that is external to that original document. Resources can include things like other documents, images, stylesheets, external scripts, and XML entities. A link reference is a URL.

Ultimately NXT 3 is a browser-based application, and links are resolved by the browser. You may link to any resource that is available by using a URL. However, this document focuses on the class of URLs that represent content contained in NXT 3.

Naming

By default, the name of a Content@ object is carried over as the name of the NXT 3 document, with an extension appended according to the object's content type. Therefore, if an HTML document links to another HTML document (named "contents" in the Content@ database) the link should follow this form:

```
<a href="contents.htm">Table of Contents</a>
```

The Content@ MakeGen configuration file can change the way that NXT 3 document names are generated, but NextPage recommends that the default be used (Content@ object name maps to NXT 3 document name with extension) so that document names are consistent for authors when they create links. See Creating a makefile for more information about how names can be generated, as well as the mappings from content types to file extensions.

Relative Links

Links to resources within NXT 3 are most easily created by using relative URLs, taking into account the document and folder structure. Consider the following hierarchical object structure in Content@:

```
book
 |
 + -- images
 |     |
 |     + -- logo
 |     + -- author_photo
 + -- content
       |
       + -- preface
       + -- chapter01
       + -- chapter02
       + -- chapter03
```

To include a photo of the book's author (author_photo.jpg) in the Preface, the link should look like this:

```
<img src="../images/author_photo.jpg" />
```

Links may also be created relative to the root node of an NXT 3 content collection. This may simplify relative paths in some cases, and is independent of any NXT 3 site structure above the content collection (such as folders or sites).

Links of this variety have this form:

```
<link rel="stylesheet" type="text/css"
 href="<!-- #EXECUTIVE:HOME_PATH -->/styles/main.css">
```

NXT 3 resolves and replaces `<!-- #EXECUTIVE:HOME_PATH -->` with the correct URL to the content collection, independent of the content collection's location within the site, or even within the server that the site is part of. Refer to the *Build Utilities Help* documentation for more information about the #EXECUTIVE:HOME_PATH replacement variable.

ID Links

NXT 3 supports linking between resources by using IDs. This is convenient if you do not want to worry about the name or relative location of a resource. This is especially helpful when creating links between content collections. An NXT 3 ID link has the following form:

```
<a href="<!-- #ID:/#1/#2/#8/#204 -->">Click Here</a>
```

The value after #ID: is the ID of the document in NXT 3. NXT 3 resolves and replaces the ID with the correct URL to the document at run-time. The ID of document is determined by the makefile created with the MakeGen utility, but is typically the ID path from Content@. Two characteristics of good IDs are that they are unique, and that they remain the same over time (they are persistent). While guaranteed to be unique within a database, the Content@ ID path is not guaranteed to be persistent, since the ID path changes if an object is moved inside the database. This can diminish the value of IDs for linking.

ID links can also use a combination of ID plus a path relative from the node referenced by ID. This can be useful when the node referenced by ID is not always in a fixed location, but the hierarchy beneath that node is stable. Consider the following Content@ structure. It has three main divisions that are intended to become content collections when published to NXT, and the collections will have the IDs pubs/misc, pubs/book1 and pubs/book2.

```
stable_data
 |
 + -- misc (Collection with ID "pubs/misc")
 |     |
 |     + -- style_sheet
 + -- book1 (Collection with ID "pubs/book1")
 |     |
 |     + -- preface
 |     + -- chapter01
 |     + -- chapter02
 |     + -- chapter03
```

```
 + -- book2 (Collection with ID "pubs/book2")
       |
       + -- preface
       + -- chapter01
       + -- chapter02
       + -- chapter03
       + -- chapter04
```

If "chapter04" (in book2) is an XML document, it can reference the stylesheet from "misc" like this:

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl"
 href="<!-- #ID:pubs/misc?p=style_sheet.xsl -->"?>
<!--  rest of document... -->
```

If the XML for chapter04 contained a cross reference to chapter01 in book1 like this:

```
<xref publication="pubs/book1
document="chapter01">See Book 1 Chapter 1</xref>
```

The XSLT stylesheet could generate HTML output with an ID link by using a template like this:

```
<xsl:template match="xref">
 <a href="<!-- #ID:{@publication}?p={@document} -->">
  <xsl:value-of select="."/>
 </a>
</xsl:template>
```

Refer to the *Build Utilities Help* documentation for more information about ID links.

Solo Compatibility Guidelines

NextPage Solo uses update files to synchronize content for disconnected users. An update file is similar to a content collection, except that it represents only changes (a delta) from an original or master collection. The bulk of the content remains in the master collection, so update files can remain small for efficient downloading over slow network connections. You can also create update files that represent changes to a content collection after a previous update file. Thus, you can have update files for update files.

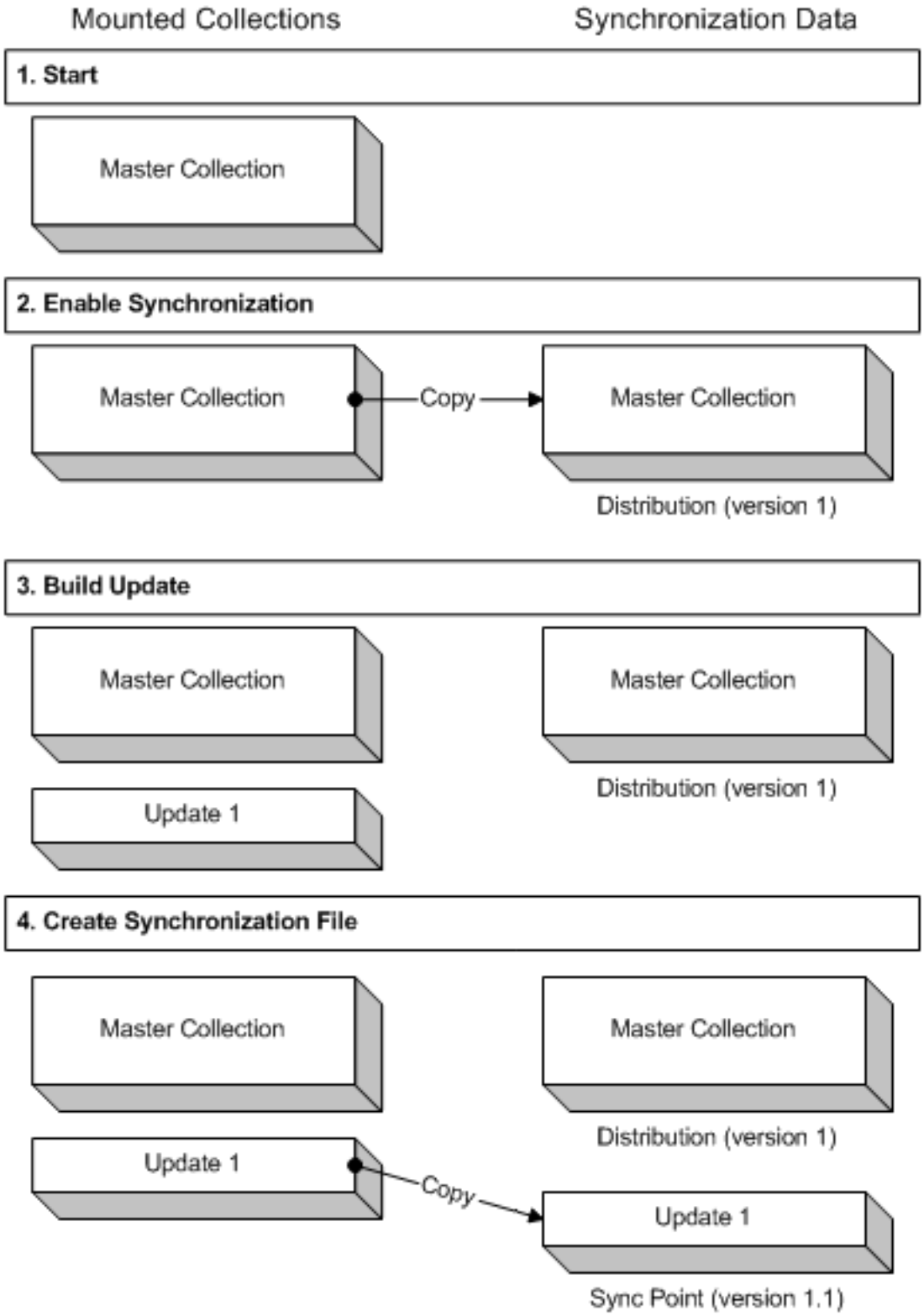
NextPage Solo works with content collections created in two ways: those created for use with the Manage Content feature of NXT 3, and those created with NPBuild. For Manage Content collections, Solo automatically manages the creation of update files. For NPBuild collections, some manual steps are required. This document explains how to create and manage NPBuild update files such that you can synchronize them.

See Also

"Solo Tasks" in *Content Network Manager Help* for NXT 3 for more information about using Solo and views.

Building Masters and Updates

The following diagram shows the process of creating a synchronization point for a content collection created with NPBuild. The content collections and update files that you mount on the live NXT 3 server are shown on the left. The synchronization data that is created by Solo is shown on the right.



1. **Start** - Use NPBuild to create the initial content collection and mount the collection on the server in the view that you enable for synchronization.

2. **Enable Synchronization** - Using the Content Network Manager application, enable the view for offline use. To do this, select the site, and choose Action > Properties.... Then choose the Views tab, select the view to enable, and the Edit button. Then, on the Disconnect tab, choose Setup.... This invokes the Disconnect Setup Wizard. After following the steps of the Wizard, Solo automatically creates the first distribution containing the master content collection.
3. **Build Update** - Use NPBuild to create an update file for new or altered content. To do this, use the -u and -m options for NPBuild. The -u option specifies that you want to create an update file, and -m indicates the collection that acts as the master. Only changes between the master collection and the current source content will be included in the update file. Here's an example of what an NPBuild command line might look like using the -u and -m options:

   ```
   npbuild C:\projects\guide.mak -m T:\collections\guide.nxt -u
   T:\collections\guide001.upd
   ```

   After creating the update file, mount it on the site in place of the master collection. In other words, change the **File name** field in Content Network Manager to point to the update file. You can do this by following these steps:

   Start Content Network Manager.

   Select the content collection that is the master for the update file.

   Choose **Action** and then **Properties** from the menu.

   Change the **File name** field to point to the update file. This ensures that the collection remains available to users of the site without interruption: this is also known as a "hot swap".

   Click **Save Changes**.

   Alternately, you can remove the master content collection, create a new content collection for the update file, and then mount the update file on the site.
4. **Create Synchronization File** - You can either create a synchronization file manually (by choosing Action > Build > Synchronization File... in the Content Network Manager), or schedule synchronization file builds as part of the Disconnect Setup Wizard. In either case, Solo copies the update file and makes it available to disconnected users.

Subsequent Updates

Creating subsequent updates follows the same pattern described in steps 3 and 4 above. Instead of using the original collection as the master, use the most recent update file as the master. Following from the example above, an NPBuild command line would look like this:

```
npbuild C:\projects\guide.mak -m T:\collections\guide001.upd -u
T:\collections\guide002.upd
```

The update file "guide002.upd" then only contains the differences since "guide001.upd" instead of all the cumulative differences since the master was created.

After creating the update file, mount it on the site, making it available to site users as well as readying it for inclusion in the next synchronization file created.

Scripting the Update Process

In addition to using the Content Network Manager for mounting and hot swapping collections and update files, you can also use the COM and Java Site Administrator APIs. These allow you to create scripts or programs that automate the process. For example, using the COM APIs, you can create Window Scripting Host scripts using JavaScript or Visual Basic. Please refer to *Programmers Information* in the NXT 3 documentation for more information.

Working with the MakeGen Configuration File

The first step to getting your content from Content@ to NXT 3 is to create a configuration file (or modify an existing one) that defines what content you want to publish and how you want it to appear in the table of contents of the NXT 3 site.

This topic provides the following information:

- Content@ Data Model Overview
- NXT 3 Data Model Overview
- MakeGen Configuration File Overview
- Configuration File Elements Reference
- Sample Configuration Files
- Configuration File DTD

In order to better understand the configuration file, you need to understand the Content@ and the NXT 3 data models.

Content@ Data Model Overview

Object Type

Every Content@ object is an instance of a single object type. The Content@ application refers to individual pieces of information in the database as objects. These objects are represented as icons in the Content@ window. You use object types to organize the data in your Content@ database.

There are three types of objects in the Content@ database:

- Container objects: These objects contain other objects (children). The children objects can be other containers, data objects, or a combination of data and container objects.
- Data objects: These objects contain data only.
- Hybrid objects: These objects are both container objects and data objects. A hybrid object is an aggregation of all objects that comprise the document, sections, or chapters of your document or manual.

  You can open, reuse, or view a component of the hybrid object, or you may open, reuse, or view all the data in the hybrid object. The hybrid object may also contain links to documents outside the Content@ database.

Content@ objects contain content and metadata. Content@ provides the following methods for representing object metadata:

- User-defined property sheet fields: Property sheet fields are named values. Each object type defines a property sheet containing property sheet fields.
- System property sheet fields: Content@ defines some property sheet fields that end-users cannot modify or delete.
- Fields: Similar to property sheet fields, fields are named values. Content@ offers a set of predefined fields for each object.

- Attributes: Attributes are similar to fields and property sheet fields in that they store named values. However, users with appropriate rights can create attributes at any time. Objects of the same type may have different named attributes.

NXT 3 Data Model Overview

This section provides a brief overview of fundamental NXT 3 objects. Refer to *NXT 3 Quick Start* in the NXT 3 documentation for a more complete explanation.

Content Collection

A content collection is the standard repository for data in NXT 3. A content collection is a collection of documents. Content collections may contain any type of document, as well as considerable internal folder structure. Content collections also contain an index that allows efficient full-text searching over documents in the content collection.

Content collections may also contain metadata for each document. A document's metadata provides information such as the source or categorization of the document.

Document

A document is a data stream that is either authored or dynamically constructed. A document corresponds to a single page viewed in a Web browser.

A document can be of any format, including non-text formats, such as graphics. NXT 3 provides full-text search for documents of the following types:

- HTML
- XML
- Plain text
- Adobe PDF
- Microsoft Word
- Microsoft Excel
- Microsoft Powerpoint
- Corel WordPerfect

Logically, a document represents a collection unit, such as a case, chapter, article, and so forth. At the lowest level, a document is a piece of a larger work to be presented to a Web browser. It could be considered a page, without the usual restrictions of the printed page.

MakeGen Configuration File Overview

You control the generation of the makefile by using the MakeGen configuration file. The configuration file specifies what to publish from Content@ and how to represent this data in NXT 3. The configuration file is expressed in XML. Refer to the end of this section for the complete configuration file DTD.

Using MakeGen, you can accept default behavior and publish with minimal effort or you can add additional definition to your configuration file and publish a specific subset of your Content@ database.

The rule element is significant in the configuration. You specify a single default rule for the configuration. You may also optionally specify rules for specific object types or objects. MakeGen uses the most specific rule available when it decides how or if to publish Content@ nodes.

The rule element defines whether or not a node and its children (and the content of the associated document) are published in NXT 3.

Configuration File Elements Reference

This section describes each of the XML elements in the MakeGen configuration file. You need this information if you want to change the configuration file to publish something other than what the file provides by default.

> **Note:** Before you modify the configuration file, we recommend that you make a backup copy of the original configuration file so you can reuse it as a template again in the future or just to easily publish content with minimal work on your part for any Content@ database.

publication element

The publication element is the root element of the configuration file. The publication element defines what you want to publish and the location of the resulting makefile.

Attribute	Description
version	The version of the configuration file. This currently must be `1.0`.
idpath	The Content@ ID path of the root object to publish. Content@ MakeGen defines a makefile that publishes all of the descendants of this object that satisfy the configuration.
makefile	The complete path of where to create the resulting makefile.
dtd	The uniform resource identifier (URI) of makefile.dtd. The URI refer to a web site URL or to a file in the file system. The URI typically describes the mechanism used to access the resource, the computer where the resource is located, and the file path and name of the resource. For our purposes, you typically need to use a URI that points to a file within the directory where you installed NextPage Content@. By default, the URI should be `file:///C:/Program Files/NextPage/NXT 3/bin/makefile.dtd`.

database element

The database element provides connection information for the Content@ database.

Attribute	Description
host	The computer that hosts the Content@ server that serves the publish data. You can provide the host information as either a machine name or an IP address.
port	The Content@ server port on the host computer. Provide the port using the string representation of the port number.
database	The Content@ database that contains the publish data.
user	The username to use when accessing Content@ server. **Note:** You may want to create a special user in Content@ that you can use with the MakeGen utility.
password	The password to use when accessing Content@ server for the specified user. **Note:** The MakeGen configuration file and the resulting makefile contain the user name and password to the Content@ database in plain text format. If security is an issue, ensure that the directory where these files are kept on the build computer is secured (such as with permissions or encryption).

content-collection element

The content-collection element defines the content collection that the makefile generates.

Attribute	Description
id	The ID of the content collection. This ID must be unique within the NXT 3 site that contains the content collection.
title	The title of the content collection. This is used for the table of contents information.
filename	The complete path to the content collection.

encryption	The encryption of the content collection. Must be one of the following:
	`none`
	`exportable`
	`best`
	The default is `none`.

password	The password, if you choose to password-protect the content collection. Use this when you do not want people to be able to copy the content collection and use it without knowing the password.
stop-words	Flag indicating whether or not to use stop words when indexing the content collection. Specify either `yes` or `no`. The default is `no`.
lang-module	Name of the language module to use to create the content collection. The language module determines how the content collection's text is parsed and indexed. The default is `NextPage US English Server Extension Module Version 2.01`.

indexsheet element

The indexsheet element associates index sheets with the content collection. If you publish metadata with any of your documents, you must include an index sheet with an ID of "metadata".

Attribute	**Description**
id	The unique ID of the index sheet within the configuration file.
src	The complete path to the index sheet.

dse element

NPBuild requires an implementation of the data source extension (DSE) interface to acquire data from Content@. The dse element defines the appropriate implementation of this interface. This element is provided for advanced use of Content@ MakeGen. Usually, you can accept the defaults for this element.

Attribute	Description
id	A mnemonic for the DSE implementation. The default is `Contenta-dse`.
classid	The globally unique identifier associated with the desired implementation of the dse interface. The default is `{4C082E09-D6FB-4be2-B610-69052CD7589F}`.

metadata element

The metadata element defines and names a collection of Content@ properties as NXT 3 metadata.

Attribute	Description
id	The name of this collection of properties. This name must be unique within the configuration file.

metadata-property element

The metadata-property element defines a single property within a metadata definition.

Attribute	Description
name	The name for this property in the published metadata. If you do not specify a publish-name, the property takes the name of the property within Content@.
source	The type of the data in Content@. Specify one of the following: `FLD` for field `PSF` for property sheet field `ATT` for attribute
value	The name of the field, property field or attribute that contains this data.

compound-document-adapter element

The compound-document-adapter element references an adapter that binds together multiple Content@ nodes as a single document in NXT 3. The adapter is a standard Content@ compound document adapter.

Attribute	Description
id	A unique ID for the adapter within the configuration file.
adapter	The name of the configuration adapter.
custom-params	Configuration parameters specific to the adapter.
compound-version	This value can be either `yes` or `no`. When NPBuild produces an incremental content collection, it compares version information for each existing document in the collection against version information in the Content@ database. When an entire Content@ node hierarchy is output as a single NXT 3 document, NPBuild often must examine version information for every node in the Content@ hierarchy. This attribute controls whether NPBuild examines a single node or a node hierarchy for version information. An attribute value of `yes` instructs NPBuild to examine the complete hierarchy. An attribute value of `no` instructs NPBuild to examine a single node.

extension element

Using a property element, you specify the name for each published document. MakeGen attempts to append an extension to the name before publishing the document. The extension is based on the content type of the document. MakeGen uses the value that you specified for content-type when choosing an extension. If you did not specify content-type or if you specified a value from a Content@ object that does not have a value, MakeGen does not append an extension.

Content@ MakeGen has an associated extension for many content types. Use the extension element to override or extend the default extensions.

Attribute	Description
type	The content type for which you want to define a new default extension.
extension	The extension to use for this content type. Provide the extension by itself, without the period.

The following table lists content types with associated default extensions.

Content Type	Extension
application/postscript	ai
application/sgml	sgm
audio/basic	au
video/x-msvideo	avi
image/x-ms-bmp	bmp
application/octet-stream	class
text/css	css
application/msword	doc
application/postscript	eps
application/octet-stream	exe
application/x-fif	fff
image/gif	gif
text/html	htm
image/jpeg	jpg
x-music/x-midi	mid
video/quicktime	mov
video/mpeg-2	mp2v
video/mpeg	mpg
application/pdf	pdf
image/png	png

application/vnd.ms-powerpoint	ppt
application/postscript	ps
video/quicktime	qt
application/rtf	rtf
audio/basic	snd
image/tiff	tif
text/plain	txt
audio/x-wav	wav
image/x-wmf	wmf
application/wordperfect	wp
application/x-WordPerfect-viewer	wpd
application/vnd.ms-excel	xls
text/xml	xml
text/xsl	xsl
application/zip	zip

rule element

The rule element defines the publication rule for an object, all the objects of an object type, or as a default for the entire publication. The rule elements define whether or not publish the object or object type, and whether or not to publish document and metadata for the object or object type.

Attribute	**Description**
publish	Whether or not to publish this object or object type.

publish-children	Whether or not to publish the children of this object or object type. Specify either `yes` or `no`.
publish-content	Whether to publish content for this object or object type. Specify either `yes` or `no`.
indexsheet	The index sheet to use when indexing the publication of this object or object type. The value must match one of the index sheet IDs associated with the content collection. Ignored if publish="no".
metadata	The name of a metadata element defined in this configuration. If this attribute is not present, MakeGen does not publish metadata. Ignored if publish="no".

Note: Apply rules with publish-content="no" to all folder-like objects that may have children, but no content. If you publish an object without content type and without publish-content="no", the object and all of its descendants are written as part of every build, whether the build is incremental or complete.

property element

The property element defines the value for standard NXT 3 node properties. Each node in an NXT 3 content collection has a pre-defined set of properties. You can define NXT 3 properties from fields, property fields, or attributes from the Content@ object or by using string constants.

Attribute	**Description**
name	The name of the NXT 3 property.
source	The source of the data. Must be one of the following: `FLD` for field `PSF` for property sheet field `ATT` for attribute `CON` for constant
value	Either the string constant or the name of the Content@ field, property field, or attribute.

default-source	The source to use if the source, value pair above does not produce a property value. You must define default-value if you define default-source.
default-value	The value to use if the source, value pair above does not produce a property value. You must define default-source if you define default-value.

The following table shows the valid values for the name attribute.

Name	**Type**	**Description**	**Default**
id	string	The ID of the document.	Content@ IDPATH field
name	string	The name of the document. The name must be unique among its sibling nodes.	Content@ NAME field
title	string	The title of the document.	Content@ NAME field
hidden	(yes|no)	A flag indicating whether a document and its children are hidden.	
index	(yes|no)	A flag indicating whether a document is indexed.	
content-type	string	The content type of the document.	Content@ MIME Type property sheet field

encoding	string	The encoding of the document property for this document.	

compression	string	Defines the compression for this object or object type.
		Specify one of the following:
		`none`
		`fast`
		`best`	

Note: Although content-encoding is not required, providing it allows NPBuild to be more efficient as it creates the makefile, which also shortens your build times.

object-type element

The object-type element defines the publication rules associated with a given object type. You use object-type elements to control whether or not to publish all instances of an object type and how to publish instances of an object type.

Attribute	Description
name	The name of the object type.

object element

The object element defines the publication rules associated with a specific object. You use object elements to override object-type elements. For example, use an object element if you want to hide most but not all objects of a given object type in the NXT 3 table of contents that displays for the user.

Attribute	Description
idpath	The idpath of the object that is the subject of the rule.

compound-rule element

The compound-rule element indicates that MakeGen should use a Content@ adapter when publishing the associated object or object type.

Attribute	Description
publish-children	Whether or not to publish the children of this object or object type. Specify either `yes` or `no`.
adapter-id	The ID of a compound-document-adapter defined in this configuration file.
indexsheet	The index sheet to use when indexing the publication of this object or object type. Must be one of the index sheet IDs associated with the content collection.
metadata	The name of a metadata element defined in this configuration. If this attribute is not present, MakeGen does not publish metadata.

Sample Configuration Files

The following sample configuration files help to illustrate how you can create or modify a configuration file to better meet your needs. Refer to the Configuration File Elements Reference for specific information about what belongs in each element and what attributes you can include.

> **Note:** The following samples assume that the contentamakegen.dtd file is in the current working directory (this is probably the C:\Program Files\Nextpage\Content@\bin directory). You may need to change the <!DOCTYPE> directive from the samples to point to a different directory if the contentamakegen.dtd file is not in the current working directory.

Publish document for all descendants of the root object

Each configuration file must specify a content collection and a rule. When a rule appears as a child of the publication element, the rule defines defaults for the entire publication. In this example, the publication default rule accepts all default rule settings. This configuration publishes all descendants of the root node. The publication includes the following at each node:

- An id property obtained from the Content@ IDPATH field
- A name property obtained from the Content@ NAME field
- A title property obtained from the Content@ NAME field
- A content-type property obtained from the Content@ MIME Type property sheet field.

This example also includes a rule for Folder objects. This rule instructs NextPage Content@ not to publish content for Folder objects. It also instructs NextPage Content@ to publish the descendants of Folder objects.

```
<?xml version='1.0'?>
<!DOCTYPE publication SYSTEM "contentamakegen.dtd">
<publication idpath="/#1/#2/#30" makefile="publishall.xml"
dtd="file:///C:/Program Files/NextPage/NXT 3/bin/makefile.dtd">
 <database host="server" port="6050" database="publish"
  user="sysadmin" password="manager"/>
<content-collection id="contentapublish" title="Published From
Content@" filename="C:\program
files\nextpage\bin\contentapublish.nxt"/>
 <dse/>
 <rule/>
 <!--Define a rule that indicates that Folder objects
  do not have content.-->
 <object-type name="Folder">
  <rule publish-content="no"/>
 </object-type>
</publication>
```

Publish document and metadata for all descendants of the root object

This configuration extends the default rule setting by adding a definition of metadata at each node. From the metadata elements we infer that each object in this Content@ hierarchy contains property sheet values of “dept” and “category. The configuration also includes an indexsheet with id=“metadata” to index the metadata.

```
<?xml version='1.0'?>
<!DOCTYPE publication SYSTEM "contentamakegen.dtd">
<publication idpath="/#1/#2/#30" makefile="publishall.xml"
dtd="file:///C:/Program Files/NextPage/NXT 3/bin/makefile.dtd">
 <database host="server" port="6050" database="publish"
  user="sysadmin" password="manager"/>
 <content-collection id="contentapublish" title="Published From
Content@" filename="C:\program
files\nextpage\bin\contentapublish.nxt">
  <indexsheet id="metadata" src="C:\Program
  Files\nextpage\bin\indexsheets\Metadata.xil"/>
 </content-collection>
 <dse/>
 <rule metadata = "deptandcategory"/>
 <metadata id = "deptandcategory">
  <metadata-property value="dept" source="PSF"
   name="department"/>
  <metadata-property value="category" source="PSF"
   name="category"/>
 </metadata>
</publication>
```

Publish document for all descendants of the root object plus metadata for all instances of a specific object type

In this example, the default publication rule indicates that all descendants of the root object be published with document only. Additionally, this configuration indicates that for all instances of object-type “standard”, both document and metadata should be published and provides the definition of the metadata for the “standard” object-type.

```
<?xml version='1.0'?>
<!DOCTYPE publication SYSTEM "contentamakegen.dtd">
<publication idpath="/#1/#2/#30" makefile="publishall.xml"
dtd="file:///C:/Program Files/NextPage/NXT 3/bin/makefile.dtd">
 <database host="server" port="6050" database="publish"
  user="sysadmin" password="manager"/>
 <content-collection id="contentapublish" title="Published From
Content@" filename="C:\program
files\nextpage\bin\contentapublish.nxt">
  <indexsheet id="metadata" src="C:\Program
  Files\nextpage\bin\indexsheets\Metadata.xil"/>
 </content-collection>
 <dse/>
 <rule/>
 <metadata id="deptandcategory">
  <metadata-property value="dept" source="PSF"
   name="department"/>
  <metadata-property value="category" source="PSF"
   name="category"/>
 </metadata>
 <object-type name="standard">
  <rule metadata="deptandcategory"/>
 </object-type>
</publication>
```

Hide most but not all instances of an object type

The publication-level default for this example is to publish and not hide all descendants of the root object. The example includes a class-object element to override this default and to hide all objects of type “gif”. Finally, the example includes a class object to override the class-object default and to not hide the object with an idpath of "/#1/#2/#30/#87/#123".

```
<?xml version='1.0'?>
<!DOCTYPE publication SYSTEM "contentamakegen.dtd">
<publication idpath="/#1/#2/#30" makefile="publishhidegif.xml"
dtd="file:///C:/Program Files/NextPage/NXT 3/bin/makefile.dtd">
 <database host="server" port="6050" database="publish"
  user="sysadmin" password="manager"/>
 <content-collection id="contentapublish" title="Published From
Content@" filename="C:\program
```

```
files\nextpage\bin\contentapublish.nxt">
  <indexsheet id="metadata" src="C:\Program
  Files\nextpage\bin\indexsheets\Metadata.xil"/>
 </content-collection>
 <dse/>
 <rule metadata="deptandcategory"/>
 <metadata id="deptandcategory">
  <metadata-property value="dept" source="PSF"
   name="department"/>
  <metadata-property value="category" source="PSF"
   name="category"/>
 </metadata>
 <object-type name="gif">
  <rule>
   <property name="hidden" source="CON" value="yes"/>
  </rule>
 </object-type>
 <object idpath="/#1/#2/#30/#87/#123">
  <rule>
   <property name="hidden" source="CON" value="no"/>
  </rule>
 </object>
</publication>
```

Publish each instance of an object type as a compound document

This example demonstrates binding together a compound XML document. The example defines a compound-rule associated with the “CompoundXML” object-type. This rule indicates that each element of this object type should be published with its children as a single document.

```
<?xml version=’1.0’?>
<!DOCTYPE publication SYSTEM "contentamakegen.dtd">
<publication idpath="/#1/#2/#30" makefile="publishcompound.xml"
dtd="file:///C:/Program Files/NextPage/NXT 3/bin/makefile.dtd">
 <database host="server" port="6050" database="publish"
  user="sysadmin" password="manager"/>
 <content-collection id="contentapublish" title="Published From
Content@" filename="C:\program
files\nextpage\bin\contentapublish.nxt">
  <indexsheet id="metadata" src="C:\Program
  Files\nextpage\bin\indexsheets\Metadata.xil"/>
 </content-collection>
 <dse/>
 <rule metadata="deptandcategory"/>
 <metadata id="deptandcategory">
  <metadata-property value="dept" source="PSF"
```

```
   name="department"/>
  <metadata-property value="category" source="PSF"
   name="category"/>
 </metadata>
 <compound-document-adapter id="defaultbind"
  adapter="CustomBindDefault."/>
 <object-type name="CompoundXML">
  <compound-rule adapter-id="defaultbind"/>
 </object-type>
</publication>
```

Configuration File DTD

The following DTD shows the rules for each element and attribute for the configuration file.

```
<!--
 DTD for ContentaMakeGen
 Version 1.0
 Copyright (c) NextPage, Inc. 2002
-->

<!ENTITY % default-compression "'none'">
<!ENTITY % default-indexsheet "#IMPLIED">

<!ELEMENT publication (database, content-collection,
  dse,(rule|conpound-rule),metadata*,
  compound-document-adapter*, extension*,
  (object-type|object)*)>
<!ATTLIST publication
 version   CDATA #FIXED "1.0"
 idpath   CDATA #REQUIRED
 makefile  CDATA #REQUIRED
 dtd    CDATA #REQUIRED>

<!ELEMENT database EMPTY>
<!ATTLIST database
 host    CDATA #REQUIRED
 port    CDATA #REQUIRED
 database  CDATA #REQUIRED
 user    CDATA #REQUIRED
 password  CDATA #REQUIRED>

<!ELEMENT content-collection (indexsheet*)>
<!ATTLIST content-collection
 id     CDATA  #REQUIRED
 title   CDATA  #REQUIRED
```

```
 filename  CDATA  #REQUIRED
 encryption (none|exportable|best) "none"
 password  CDATA  #IMPLIED
 stop-words (yes|no) "no"
 lang-module CDATA  "NextPage US English
   Server Extension Module. Version 2.01">

<!ELEMENT indexsheet EMPTY>
<!ATTLIST indexsheet
 id     ID   #REQUIRED
 src    CDATA  #REQUIRED>

<!ELEMENT dse EMPTY>
<!ATTLIST dse
 id     CDATA  "contenta-dse"
 classid   CDATA  "{4C082E09-D6FB-4be2-B610-69052CD7589F}">

<!ELEMENT metadata (metadata-property+)>
<!ATTLIST metadata
 id     ID   #REQUIRED>

<!ELEMENT metadata-property EMPTY>
<!ATTLIST metadata-property
 name    CDATA #REQUIRED
 source   (FLD|PSF|ATT) #REQUIRED
 value   CDATA #IMPLIED>

<!ELEMENT compound-document-adapter EMPTY>
<!ATTLIST compound-document-adapter
 id     ID   #REQUIRED
 adapter   CDATA #REQUIRED
 custom-params CDATA #IMPLIED
 compound-version (yes|no) "yes">

<!ELEMENT extension EMPTY>
<!ATTLIST extension
 type    CDATA #REQUIRED
 extension  CDATA #REQUIRED>

<!ELEMENT rule (property*)>
<!ATTLIST rule
 publish   (yes|no) "yes"
 publish-children (yes|no) "yes"
 publish-content (yes|no) "yes"
 indexsheet IDREF #IMPLIED
```

```
 metadata  IDREF #IMPLIED>

<!ELEMENT property EMPTY>
<!ATTLIST property
 name    (id|name|title|hidden|index|content-type|
encoding|compression) #REQUIRED
 source   (FLD|PSF|ATT|CON) #REQUIRED
 value   CDATA #REQUIRED
 default-source (FLD|PSF|ATT|CON) #IMPLIED
 default-value CDATA #IMPLIED>

<!ELEMENT object-type (rule|compound-rule)>
<!ATTLIST object-type
 name    CDATA #REQUIRED>

<!ELEMENT object (rule|compound-rule)>
<!ATTLIST object
 idpath   CDATA #REQUIRED>

<!ELEMENT compound-rule (property*)>
<!ATTLIST compound-rule
 publish-children (yes|no) "no"
 adapter-id IDREF #REQUIRED
 indexsheet IDREF #IMPLIED
 metadata  IDREF #IMPLIED
```

Creating a Makefile

Content@ MakeGen is a Java program. You use a Windows batch file to establish the environment required by the Java Runtime Environment (JRE) in order to simplify execution. To run the MakeGen utility, you invoke the batch file and pass in the complete path to the configuration file. For example,

```
ContentaMakeGen "C:\Program Files\NXT\bin\publishstaging.xml"
```

If your path contains any spaces, you must use quotation marks at the start and end of the path information.

> **Note:** The MakeGen configuration file and the resulting makefile contain the user name and password to the Content@ database in plain text format. If security is an issue, ensure that the directory where these files are kept on the build computer is secured (such as with permissions or encryption).

You may also include the following command line options.

Option	Description
-logfile “logfilename”	By default, Content@ MakeGen sends all diagnostic output to the console. Use this option to write diagnostic output to a named logfile.
-noconsole	Use this option if you do not want to see diagnostic output on your console window.
-noinfo	Use this option to remove information level diagnostic output.

ContentaMakeGen returns a numeric return code. A value of zero indicates success. A non-zero value indicates failure. ContentaMakeGen logs a fatal error message every time it fails and returns a non-zero value.

Troubleshooting

MakeGen produces a makefile that later acts as input for the NPBuild utility when you create a content collection. For large data sources, the process of generating a makefile and building from that makefile may take many minutes or even hours.

MakeGen fails and does not generate a makefile if it cannot generate the makefile specified in the configuration file or if there is any ambiguity in the configuration file. MakeGen also logs a fatal error whenever it terminates prematurely and does not generate a makefile.

We designed the MakeGen utility to look for ambiguities in its configuration file when it first starts so that it fails at the beginning of the process rather than at the end after a lot of processing time.

The following are some situations that cause MakeGen to fail:

- Calling Content@ MakeGen with a configuration file that does not satisfy the DTD.
- Specifying Content@ connection information that does not result in a Content@ connection.
- Specifying a field name, attribute name, or property name in a rule and applying this rule to a Content@ object that does have the matching field, property, or attribute.
- Specifying rules that result in more than one document attribute with the same name.
- Specifying the identifier of a non-existent compound document adapter.
- Specifying the identifier of a non-existent metadata definition.
- Specifying an invalid root node.
- Specifying a makefile where Content@ MakeGen cannot write.
- Specifying more than one object rule with the same idpath.
- Specifying more than one object-type rule with the same name.
- Specifying a port that has no whole-number representation.
- Specifying metadata and not specifying an index sheet named “metadata”.

Diagnostic Output

Content@ MakeGen groups diagnostic output into the following levels:

Level	**Description**
Fatal	A fatal error during processing. Content@ MakeGen terminates when it encounters a fatal error. If no fatal errors occur, Content@ MakeGen runs to completion and generates a makefile.
Error	An error during processing. The error is not fatal, but does affect the content of the resulting makefile.
Warn	A problem occurred during processing that is not fatal, but does affect the content of the resulting makefile.
Info	A message about the status or progress of the makefile generation.

By default Content@ MakeGen prints Fatal, Error, Warn and Info messages to the console window. You can use command-line options to modify these defaults.

Building a Content Collection

You use the NPBuild utility provided with NextPage Content@ Publisher to create a content collection (or an update file to an existing content collection) using the makefile you generated with the MakeGen utility.

To run the NPBuild utility, do the following:

1. Open a command prompt. If the NPBuild utility is not in your path, you need to get to the directory location where NPBuild is installed (generally, C:\Program Files\NextPage\Builder 3).
2. Run NPBuild, passing it the location and name of the makefile, as in the following example:

   ```
   npbuild "C:\Program Files\NXT\bin\publish.mak"
   ```
3. Check to ensure that the content collection was created.

   Note: A line in the makefile determines where NPBuild puts the resulting content collection, so you may have to check that makefile to determine where the content collection was saved. Search for ".nxt" to find the file name and the directory location.

You may also include the following command line options:

Option	Description
/A	Forces NPBuild to rebuild and reindex the entire collection.
/L<log file path>	Specifies where log file information is written. You can specify either a full path or a relative path. If the path contains spaces, use quotation marks around the log file path information.
/U	Creates an update file instead of updating the master content collection. See the Solo Compatibility Guidelines for more information about how to use this option.

You can also create a Windows batch file that runs NPBuild. This batch file can get called from a general batch file that also calls the MakeGen utility.

See *Build Utilities Help* for more information about using NPBuild to create a content collection using a makefile.

Mounting a Content Collection

In order for your users to see the content of the content collection, you need to put it on your NXT 3 site (known as "mounting" the content collection).

The steps you need to take depend on if this content collection is completely new, if it is a revision of an existing content collection, or if it is an update file based on an existing collection.

- If this is a new content collection, you need to add the content collection to your NXT 3 site.
- If this is simply a revised version of an existing content collection, you need to overwrite the old content collection with the new one.
- If this is an update file for an existing content collection, you need to change the **File name** field for the master content collection to point to the update file.

For each of these scenarios you can use one of two tools:

- NXT 3 Content Network Manager program to mount the appropriate content collection. This provides you with a graphical interface for working with your site and content collections. See *Build Utilities Help* and *Content Network Manager Help* in the NXT 3 documentation for more information about how to put a content collection on an NXT 3 site.
- COM and Java Site Administrator APIs. These allow you to create scripts or programs that automate the process. For example, using the COM APIs, you can create Window Scripting Host scripts using JavaScript or Visual Basic. Please refer to *Programmers Information* in the NXT 3 documentation for more information.

Maintaining Your System

This section contains the following topics:

- Backing Up and Restoring content collections
- Reinstalling NextPage Content@ Publisher
- Monitoring Performance

Backing Up and Restoring

Backing Up

NextPage recommends that you add a line to the script file when you build and mount a content collection that copies the content collection to your backup location as well as the NXT 3 server directory location.

Alternately, you can manually copy (or set up your backup program to do so) all the NXT files in your NXT 3 server to a backup location.

Restoring

To restore a content collection, perform the following steps:

1. Start Content Network Manager (part of NXT 3).
2. Click on the collection you want to restore the backup of.
3. Click the **Online/Offline** button on the toolbar to take the content collection offline so users cannot access it.
4. Save the changes to the server by clicking the **Save Changes** button on the toolbar.
5. Copy the backup copy of the content collection over the top of the existing content collection.
6. Click the **Online/Offline** button on the toolbar to put the content collection back online so users can access it.
7. Save the changes to the server by clicking the **Save Changes** button on the toolbar.

Reinstalling

To reinstall NextPage Content@ Publisher, perform the following steps:

1. Run the NextPage Content@ Publisher install.
2. When prompted, select the option to reinstall NextPage Content@ Publisher.

Monitoring Performance

The goal of monitoring key aspects of the operating system, web server, and application is to identify bottlenecks. This can be useful to determine the load caused when you have NextPage Content@ Publisher and NXT 3 on the same server.

You can perform these general kinds of system checks:

- Check CPU utilization in user and kernel space for the total system and on each CPU. See CPU Statistics
- Confirm that there is no paging or swapping. See Memory Statistics
- Check that network latencies between machines are acceptable. See Network Statistics
- Identify disks with poor response times or long queues. See Disk Statistics
- Identify server latency problems. See Web Server Statistics and Application Statistics

CPU Statistics

CPU utilization is the most important operating system statistic in analyzing performance. You should measure CPU utilization for the entire system and for each individual CPU on multi-processor environments. Utilization on a per CPU basis can help detect single-threading and scalability issues.

Most operating systems report CPU usage as time spent in user mode and time spent in kernel mode. These additional statistics allow a better analysis of what the CPU is actually executing.

For NextPage Content@ Publisher server system, where there is generally only one application running, the server runs CSpace activity in user mode. Activities required to service CSpace requests (such as committing, I/O, memory management, and process/thread creation and tear down) run in kernel mode. In a system where all CPU is fully utilized, a healthy NextPage Content@ Publisher system runs between 65% and 95% in user mode.

CPU monitoring tools for Windows NT/2000

System\% User Time (PerfMon)

This counter corresponds to the percentage of time the processor is spending executing user processes such as NextPage Content@ Publisher Server.

System\% Privileged Time (PerfMon)

This counter corresponds to the percentage of time the processor is spending executing Windows NT 4.0 or Microsoft Windows® 2000 kernel commands. If this counter is consistently high when the Physical Disk counters is high, consider a faster or more efficient disk subsystem.

> **Note**: Different disk controllers and drivers use different amounts of kernel processing time. Efficient controllers and drivers use less privileged time, leaving more processing time available for user applications, increasing overall throughput.

System\Context Switches/sec (PerfMon)

This counter indicates the rate at which processors on the computer are switched from one thread to another. High values for this counter indicate contention problems over common resources. The server is wasting a large number of CPU cycles to manage context switching between threads. These CPU cycles are nonproductive because they are not directly involved in serving Web requests. Consider adjusting the `AspThreadGateEnabled` and `AspProcessorThreadMax` IIS metabase settings to see if context switches can be minimized.

System\ Processor Queue Length (PerfMon)

This counter corresponds to the number of threads waiting for processor time. A processor bottleneck develops when threads of a process require more processor cycles than are available. If more than a few processes are trying to utilize the processor's

time, you might need to install a faster processor or an additional processor if you are using a multiprocessor system.

Memory Statistics

You can use virtual memory statistics mainly as a check to validate that there is very little paging or swapping activity on the system. System performance degrades rapidly and unpredictably when paging or swapping occurs.

You can use individual process memory statistics to help detect memory leaks due to a programming failure to deallocate memory taken from the process heap. Use these statistics to validate that memory usage does not increase after the system has reached a steady state after startup. This problem is particularly acute on multi-threaded applications on middle tier machines where session state may persist across user interactions, and on completion state information that is not fully deallocated.

Memory monitoring tools for Windows NT/2000

Memory\% Committed Bytes In Use (PerfMon)

This counter indicates the percentage of memory that has been committed to run applications. For most Web applications, committed bytes increase as the test ramps up, but should reach a stable state where the value remains fairly constant.

Memory\Page Faults/sec (PerfMon)

This counter indicates the rate at which page faults are occurring. A page fault occurs when a process requests a page of memory and the system cannot find it at the requested location. If this number is high (compared to the situation when no test load is applied), it might be because too much memory is dedicated to the application and not enough to the system. You may want to add more memory to your server. It might also indicate design problems in your application, such as unnecessary creation and destruction of objects within a script.

Disk Statistics

NextPage Content@ Publisher CSpaces reside on multiple disks so the performance of the I/O subsystem is very important to the performance of the application. Most operating systems provide extensive statistics on disk performance.

The most important disk statistics are the current response time and the length of the disk queues. These statistics indicate whether the disk is performing optimally or it is being overworked. If a disk shows response times over 20 milliseconds, then it is performing badly or is overworked and is a potential bottleneck. Also, if disk queues start to exceed two, then the disk is a potential bottleneck of the system.

Disk monitoring tools for Windows NT/2000

Physical Disk\% Disk Time (PerfMon)

This counter indicates the percentage of time the disk spends servicing read or write requests. To monitor this counter, you must activate it by typing `diskperf -y` at the command prompt. If this value is high while other resource usages (such as CPU or network) are low, high disk activity is a likely performance bottleneck. Adding more memory to the server may improve the performance.

Physical Disk\Avg. Disk Queue Length (PerfMon)

The Current Disk Queue Length counter indicates how many system requests are waiting for disk access. The number of waiting I/O requests should be sustained at no more than 1.5 to 2 times the number of spindles within the physical disk. Most disks have one spindle, although redundant array of inexpensive disks (RAID) devices usually have more. A hardware RAID device appears as one physical disk in System Monitor; RAID devices created through software appear as multiple instances.

Network Statistics

You can use network statistics in much the same way as disk statistics to determine if a network or network interface is overloaded or not performing optimally. In today's networked applications, network latency can be a large portion of the actual user response time. For this reason, these statistics are a crucial debugging tool.

Network monitoring tools for Windows NT/2000

Network Interface\Bytes Total/sec (PerfMon)

This counter indicates the rate at which bytes are sent and received through the network interface. If this number is approaching the total bandwidth of the network interface card (NIC), the network interface may have become the bottleneck. Consider using an NIC with greater bandwidth capacity or using multiple NICs for the same network segment.

% Network Utilization (Network Monitor)

This shows the percentage of network bandwidth in use on this network segment. The % Network Utilization bar graph displays the percentage of your network's available resources that are being used. This bar has a maximum value of 100 percent. A black line appears on the bar to represent the maximum value reached in any capture session.

Web Server Statistics

Web Server statistics are another good indication of overall performance.

Windows NT/2000 IIS Web Server Monitoring Tools

Total Method Requests/Sec

This counter indicates the rate at which the Web server is receiving requests using the GET and POST method.

Bytes Total/Sec

This counter indicates the total rate of bytes transferred to and from the IIS Web server. Bytes Total/sec helps to determine if the network is getting too close to its bandwidth threshold because of the large data throughput.

Active Server Pages\Request Execution Time

This counter indicates the time (in milliseconds) spent executing the most recent request on the Web server. The sum of this counter and Request Wait Time represents the latency (response time) of the application from the server perspective.

Active Server Pages\Request Wait Time

This counter indicates the number of milliseconds that the most recent request spent waiting in the ASP queue. The goal is to tune your application or system to keep this number at zero.

> **Note** These two latency-related counters apply only to ASP applications. If you need to measure latency in a non-ASP application, you will need to rely on other mechanisms such as time stamping your application.

Application Statistics

Sessions Statistics

You can use Current Sessions and Max Sessions Reached to determine if a particular server is serving too many concurrent users. Serving more than the planned number of concurrent users causes the response time for all users to go beyond acceptable standards.

Running Since

You can use Running Since to determine the up time of the server. When intentional or planned restarts are taken into account Running Since is a measure of the reliability of the server. Unplanned restarts are usually the result of a critical error in some component of the application. NextPage Content@ Publisher traps these errors and automatically restarts the server. Trapped errors are logged in the NPLog.txt file.

Getting Support

NextPage is committed to providing technical support for your NextPage Content@ Publisher deployment. This section describes topics that are relevant to obtaining support from NextPage.

- Maintenance and Support Programs
- Reporting Product Defects and Enhancements
- New Support Ticket Checklist
- Support Ticket Severity Levels

Maintenance and Support Programs

NextPage offers two types of annual fee-based Maintenance and Support Programs. Both the Basic Service Program and the Premier Service Program are available for all currently shipping NextPage products.

Basic Service Program

The Basic Service Program, which is suitable for most businesses, features unlimited phone and email support during business hours and a software subscription service.

The Basic Service Program offers the following entitlements:

- **Software Subscription:** You receive maintenance release updates and upgrades of NextPage products covered in your purchase contract when and if available.
- **Tiered Technical Support:** You receive unlimited, toll free telephone and electronic mail (email) defect correction support from NextPage Support Services engineers during normal business hours.
- **Remote Diagnosis/Access:** When requested and required, the Support Services staff will dial into your site to diagnose and solve reported NextPage product defects.
- **Knowledgebase Access:** For each NextPage product, NextPage Support Services maintains a knowledgebase of known problems and their solutions. You receive free, unlimited access to this knowledgebase as well as access to technical tips and hints, tutorials, and technical notes.
- **Two (2) Registered Customer Contacts:** You are allowed two (2) registered customer contacts.
- **Pricing and Availability:** Pricing for the Basic Service Program is product dependent, with the prices published along with the appropriate product descriptions.

Premier Service Program

The Premier Service Program is best for businesses deploying mission critical web-based applications. The Premier Service Program features all the Basic Service Program benefits, plus unlimited phone and e-mail support 24 hours a day, 365 days a year with a one hour response time for critical calls.

The Premier Service Program offers the following entitlements:

- **Software Subscription:** Same as the Basic Service Program.
- **Tiered Technical Support:** Your registered customer contacts are allowed to receive unlimited, toll free telephone and electronic mail (e-mail) support 24 hours a day, 365 days a year for Category 1 calls from the NextPage Support Services engineers. Outside of normal business hours (Monday - Thursday, 5:00 AM - 8:00 PM Mountain Time; Friday, 5:00 AM - 5:00 PM Mountain Time), a NextPage Senior Support Services Engineer will respond to critical problems within one hour of call initiation. NextPage Support Services will respond to all problems logged by electronic mail within four (4) hours.

- **Remote Diagnosis/Access:** Same as the Basic Under the Premier Service Program but this entitlement is available 24 hours a day, 365 days a year (availability outside normal business hours is limited to critical defects).
- **Knowledgebase Access:** Same as the Basic Service Program.
- **Educational Discounts:** All registered customer contacts receive a 15% discount on all NextPage training courses.
- **Three (3) Registered Customer Contacts:** You are allowed three (3) registered customer contacts.
- **Pricing and Availability:** Pricing for Premier Service Program is product dependent, with the prices published along with the appropriate product descriptions.

Maintenance and Support Contract

You must have a Maintenance & Support (M & S) contract to receive technical support from NextPage Support Services on NextPage products. You can obtain an M & S contract by contacting your local NextPage partner or NextPage Sales at 1.800.NEXTPAGE (800.639.8724) or 801.768.7600 or by email at `sales@nextpage.com`.

Each Maintenance & Support contract includes a Support ID Number that you must include on all correspondence to NextPage Support Services.

Reporting Product Defects and Enhancements

There are a number of ways you can report a product defect or request an enhancement.

Email

The email address for NextPage Support Services is `support@nextpage.com`

Telephone

You can contact NextPage Support Services as follows.

Monday through Thursday	5:00 AM to 8:00 PM Mountain Standard Time (MST)(GMT-07:00)
Friday	5:00 AM to 5:00 PM
Inside the U.S.	800.NEXTPAGE (800.639.8724)
In Maine, Alaska, Hawaii and Canada	800.543.6546
Outside the U.S.	800.NEXTPAGE, 801.768.7500, or 801.768.7600

FTP Site

The Support Services FTP Site address is ftp://ftp.nextpage.com.

You can access the Support Services FTP site through an FTP client, such as WSFTP or FTPExplorer, or through your browser (must be Internet Explorer 5 or later and have "Enable folder view for FTP sites" checked under Tools > Internet Options > Advanced).

> **Note:** Before you upload a file to the FTP site, you should phone or email Support Services to let them know you are uploading a file.

Login

To log in to the Support Services FTP Site, use the following:

User Name: `anonymous`

Password: no password

Folders

- `Incoming` - Use this folder to upload files to the FTP site. With anonymous access, you can open and upload files to this folder, but you cannot see the contents of the folder.
- `Public` - Use this folder to download files from NextPage Support Services. With anonymous access, you can download from this folder, but you cannot write to this folder.
- `Private` - Contents of this folder are hidden to any anonymous users.

- `Folio` - Contains Folio product downloads.

NextPage-Tech List Server

NextPage-Tech is a list server that notifies subscribers of important technical issues and announcements related to NextPage products. NextPage-Tech is available to all NextPage customers and partners, and is directed at a technical audience. Emails are generated when a new product is released, a TechNote is released, or when a significant technical issue is reported.

Subscribe to NextPage-Tech

To subscribe to NextPage-Tech, send an email to:

```
ls@nextpage.com
```

In the body (not subject) of the email, enter the following, replacing John Doe with your name:

```
Subscribe NextPageTech John Doe
```

Unsubscribe from NextPage-Tech

To unsubscribe from NextPage-tech, send an email to:

```
ls@nextpage.com
```

In the body (not subject) of the email, enter the following:

```
SignOff NextPageTech
```

WebTicket

NextPage Support Services WebTicket is a web interface to the Support Services Tickets database. It allows you to log in and add new tickets, view your open or closed tickets, and even add comments to your existing tickets.

To access the Support Services WebTicket, go to http://support.nextpage.com and click on Support Services WebTicket in the table of contents. Or, go straight to the following URL: http://support.nextpage.com/nxt/gateway.dll/WebTicket?f=templates$fn=default.htm

User Name and Password

To log in to WebTicket, you must have a User Name and Password. Your User Name and Password can only be obtained from NextPage Support Services.

To obtain a User Name and Password, please use the WebTicket User Name and Password Request Form. You will need to provide the following information on the form:

- Name
- Company Name
- Support ID Number
- Email Address
- Desired Password

After completing the form, you will receive an email reply confirming that your User Name and Password have been activated.

If you have forgotten your User Name or Password, or if you are having trouble logging in, please use the WebTicket User Name and Password Request Form, or send an email to `support@nextpage.com` outlining the problem you are having.

Please include as much of the information listed above as you can. You will receive an email reply containing your User Name and Password and any specific instructions for logging in.

Logging In to WebTicket

To log in to WebTicket, simply enter your **User Name** and **Password**, and click the **Log On** button. If you do not have a User Name and Password, see the section above entitled "User Name and Password".

> **Note:** Your browser must have cookies enabled in order to use NextPage Support Services WebTicket. If you are using Internet Explorer 6.0, you must "Accept All Cookies". To do this, go to **Tools**, **Internet Options**, **Privacy**, and select **Accept All Cookies**.

Viewing NextPage Support Tickets

Once you have logged in, WebTicket opens the Support Tickets List and displays a list of all your open tickets. The list includes the Ticket ID, Received Date/Time, Product, Assigned To, and Status.

Above the list, there are three tabs: Open Tickets, Closed Tickets, and All Tickets. Use these tabs to display the desired list of tickets.

The Support Tickets List shows only the tickets that are assigned to the contact you are logged in as, and does NOT show all tickets for the entire company. In other words, you are logging in as a contact, not a company, and the lists are generated accordingly.

Viewing Ticket Details

To read the details of a particular ticket, click the Ticket ID in the Ticket ID column. The Ticket Detail window will be opened, displaying all the details for the ticket.

The Ticket Detail window is divided into five sections - Customer Information (top), Ticket Information (middle), Problem Description, Problem Resolution, and Input Additional Comments (described below).

When you are finished viewing the ticket, click the Tickets (Search) button at the top to return to the Open Tickets list.

Adding Additional Comments to a Ticket

You can add additional comments to your tickets from the Support Services WebTicket. The page for adding additional comments to a ticket is located on the Ticket Detail screen. At the bottom of the Ticket Detail screen, you will see a text area entitled "Input additional comments on this ticket". You can use this field to add notes and comments to the ticket or respond to questions sent to you by a Support Services Engineer.

Type all the necessary information in the text box, then hit the "Save Additional Comments" button to submit your comments. All additions are amended to the bottom of the "Problem Description" field.

You can add comments to any ticket, regardless of the whether the ticket is open or closed. When a new comment is added, the status of the ticket is changed to "Open" so the Support Services Engineer will be aware of the change.

Creating New Support Tickets

To add a new Support Ticket, click the "Create" button at the top of the WebTicket screen.

The Add Ticket Window contains a form designed to help you give us the necessary information for working on your ticket. Most of the fields are not required, but fill in all the information that is applicable. The more information you provide, the faster Support Services will be able to find a solution.

Once you have entered all the necessary information, click the "Save Ticket" button. The ticket will be added and assigned a Ticket ID. The Ticket Details window will be opened showing you the details of the new ticket, including the assigned Ticket ID. Please use the Ticket ID with all correspondence to NextPage Support Services.

Searching for Support Tickets

To search for a Ticket by Ticket ID, open the Support Tickets List by clicking the "Tickets" (Search) button. At the top of the Support Tickets List, there is a search field. Type in the Ticket ID and click the magnifying glass button to search. The ticket will be displayed in the Support Tickets List. Click the Ticket ID to view the ticket details.

New Support Ticket Checklist

When opening a new ticket with NextPage Support Services, it is important that you supply as much relevant information as possible. This helps avoid the need to send email back and forth requesting additional information, and helps us respond to your needs faster. To better help us serve you, especially when opening a ticket via email, please provide all relevant information from the following list:

1. Contact Information.

   Company Name

   Contact Name

   Support ID Number
2. NextPage product name and version.
3. Operating system, including Service Patch numbers.
4. Machine specifications (CPU, RAM, HDD free space, connection speed, etc).
5. Web server name and version (IIS, Netscape, etc), if using LivePublish or NXT 3 Server.
6. Browser(s), including version, build number, and encryption level.
7. Duplication steps to duplicate the problem (be as descriptive as possible).
8. Relevant files, if needed for duplication or testing. Email files to `support@nextpage.com`, or upload to the Support Services FTP Site (ftp://ftp.nextpage.com). Only include files you have modified that you feel might be causing the problem. If the source files cannot be sent, describe the file content as best you can, including file type, size, folder configuration, and the build process used to create the content collection. Some of the files that may be needed include:

   nextpage.sdf

   NXT 3\bin*.ini

   NXT 3\templates\enu*.htm

   Source files and MAK files
9. Error messages displayed or logged as a result of the problem.
10. Number of concurrent users at the time of the problem.
11. The server's and/or client's processor and memory usage (pegged, flat, etc).
12. Other software installed on the system (SQL Server, Tivoli, SMS, etc).
13. Any other details that may be important to the situation.

Support Ticket Severity Levels

NextPage Support Services uses the following definitions for categorizing the severity of product defects.

Severity Level 1	A down situation, whereby a customer is unable to do production work, and a work-around is either not available or is unacceptable to the customer. The product may: • corrupt or permanently destroy data • repeatedly fail catastrophically • require repeated reboots of the system
Severity Level 2	A major function / product is unusable and no work-around is available, but the customer is able to do some production work. The product may: • be usable but incomplete (one or more documented commands/functions are inoperable/missing) • fail catastrophically • require rebooting of the system • suffer sufficient degraded performance (throughput/response) such that there is a severe impact on use

Severity Level 3	There is a loss of a function or resource that does not seriously affect the customer's operations or schedules. Any problem, which was originally reported as Category 1 or Category 2, but has been temporarily solved with a Work-Around, shall be reduced to Category 3. This category includes problems associated with the installation of NextPage Product(s).
Severity Level 4	All other problems with NextPage Product(s) other than those falling within the categories above. This category includes errors in Product Documentation and instances when the product does not operate strictly according to specifications.

Finding Documentation

NextPage Content@ Publisher exposes data from the Content@ Content Management System through the NXT 3 e-Content Platform and allows publishing departments to easily author, manage and assemble mission critical information. With NextPage Content@ Publisher, you can speed your content creation process while increasing information accuracy, quality and consistency and reducing editing and maintenance costs.

Use the NextPage Content@ Publisher documentation when you need help with the part of the product that bridges XyEnterprise's Content@ and NextPage's NXT 3. This documentation is all contained in the "docs" folder (and the "print" subfolder for the PDF versions) of the "Content@" directory where you installed NextPage Content@. Typically, you install this in the "C:\Program Files\NextPage\Content@" directory.

To find out what documentation you can use when working with NextPage Content@ Publisher, click on your **Start** button, choose **Programs**, choose **NextPage**, choose **NextPage Content@ Publisher**, and then choose **NextPage Content@ Publisher Documentation**.

Documentation Updates

Subsequent to release, you can obtain the latest NXT 3 and NextPage Content@ Publisher documentation online at http://docs.nextpage.com.