00101000
FINGERS

COÖPERATIE 40FINGERS U.A.
ROERMONDSPLEIN 33 - 6811 JN
ARNHEM - THE NETHERLANDS
T: + 31-26-3653322
INFO@40FINGERS.NET

40FINGERS SEO RE-DIRECT
MANUAL V 20191108

INTRO

The SEO redirect module was first created by us in 2012 when we “converted” a client website from PHP to DNN.
Because the existing website ranked quite well in google we needed a way to add mapping for redirects from the old pages to the new ones.
Over the years we have add more functionality to the module and its interface and we now use it on most of our websites.

When you find typo’s or other issues in this manual, please let us know: https://github.com/40fingers/DNN-SEORedirect

404 and 301

For SEO, it’s useful to redirect renamed or moved pages or files to their new location.
Although DNN has built in functionality for renamed pages, it does not for all other situations.
The SEO redirect allows you to see the incoming 404’s and add a redirect them to a new location if needed.

FUNCTIONALITY

-The module logs what URLs generate a “Page not Found” on the 404 page.
- An admin can see a list of 404s and add redirect (301) rules to another location.
- This is possible for DNN pages and static files.
- The mappings are stored in an XML file in the portal Folder.
This allows you to prepare a redirect list when moving a website from another system to DNN.

A NOTE ON SETTING UP A NEW DNN WEBSITE

This module is helpful when you setup a new website and want to maintain your SEO ranking.
When you do this we advise you to use a subdomain for the new website and not a child portal.
So new.mywebsite.com and not www.mywebsite.com/new as the Portal Alias.
Due to the way DNN handles domain-names, the module can handle relative URL’s but not on child portals (without performance implications). When you use a subdomain you can use relative redirects and they will keep working when you switch to the live DomainName.

HOW IT WORKS

SEO redirect consists of 2 parts.
1. A DNN module you place on your 404 page.
2. A .NET HttpModule for redirects before DNN resolves the URL.

There three types of 404s you can log/redirect.
1. DNN pages
2. Static Content (files, jpg etc.)
4. Non existent URLs

DNN Pages & 404s
To explain the position of this module and some of the design choices we made, we'll first explain when 404 page handling is built into DNN.

DNN has some built in 404 handling, but it has some limitations.
The core functionality is mainly targeted at the situation where the URL of a page changes.
When you change the URL of a DNN page, the old URL remains in the TabURLs table in the database.

This helps for changing URLs, but not for removed pages as in that case the Tab URL is removed.
This also does not cover URLs that never existed of static files.
Moved pages also don't always get the correct redirect.

Also the built in DNN 404 redirect, only works for root pages.
URLs that point to higher levels will be severed as the first page up the tree.
This is needed because in DNN, detail-URLs of modules are handled by the module itself.
The core does not know what detail URLs (could) exist for a module on the page.

Page structure Example:

Home
Products
---- Product 1
---- Product 2
---- Product 3
---- Product 4
Contact

When a user requests the following URLs:
www.website.com/nothing:
The user will see the DNN 404 page.

www.website.com/products/nothing
The user will see the products page and not the DNN 404 page.

This is where the HttpModule is needed.
When you create a redirect for the "nothing" page in the second example, the HTTP module will do the redirect before DNN tries to serve it (as it would show you to the "products" page).

FYI; there is currently no way to force DNN show the 404 page for pages other than the root level.

Static Files
So called static files are files normally not handled by .NET / DNN.
This are files like images, stylesheets, but also URLs that point to extensions not supported extensions like .php.
When you make a request to this type of URL and it does not exist, you get the default IIS 404 page by default.
When you change the default IIS 404 page to the DNN 404 page's URL, you will also see static files in you 404 logging.

INSTALLATION

Take the following steps:

1. Please backup your DNN installation.
2. Install as a normal module, minimum DNN version: DNN 07.02.01

Preparation
Take the following steps:

1. Create your 404 page
2. Make sure the 404 page is a language Neutral page is static files.
3. Put the module on the page, you probably want to hide the page from the menu.
4. Make sure that the page, as well as the SEO Redirect module are visible for all users.
5. Place the "message" content on the 404 page.

The module should be visible on the page for unauthenticated users, but it will not show any content to those users.

This base configuration will allow you to track and redirect all pages that are not a child of an existing root page.

To be able to redirect all possible DNN pages you would have to do that before DNN handles the URL.
The only way to do this is by using the HttpModule, which will fire before DNN tries to resolve the URL.
.(see section "DNN Pages & 404s" above)

Configurating HTTP module
https://weblog.west-wind.com/posts/2012/Oct/25/Caveats-with-the-runAllManagedModulesForAllRequests-in-IIS-78

HTTP module (Configuration IIS7):

Place this as the first element in <modules>

```
<system.webServer>

  <modules>

        <add name="SeoRedirectModule" type="FortyFingers.SeoRedirect.Components.SeoRedirectModule,
40Fingers.DNN.Modules.SeoRedirect" preCondition="managedHandler" />

  </modules>
```

```
</system.webServer>
```

Please note that to redirect the .asp extension (classic asp) you must add this web.config entry or you will get an invalid url error.

Static Files

For performance reasons Static files (.jpg. png. .css etc.) are not handles by .NET / DNN, but by IIS directly
When a 404 occurs for a static file the use gets to see the default IIS 404 page.
When you want the 404s for static files to appear in the loggings you have to set the IIS 404 page to the DNN 404 page in web.config

Adapting the web.config for static file:

```
<system.webServer>

...

    <httpErrors>

      <remove statusCode="404" subStatusCode="-1" />

      <error statusCode="404" prefixLanguageFilePath="" path="/404" responseMode="ExecuteURL" />

    </httpErrors>

</system.webServer>
```

You can also set this in IIS manager, by editing the sites “Error Pages”.
Please note that in DNN9, if your skin causes any 404 errors, it might be that switching to Edit Mode stops working after this change.

There are 2 limitations to this approach:

1. When the site is a multiple languages site:
   The 404 page cannot be a translated page in combination with an “generated” language URL.
   When you want to translate your 404 message there are two options.
   A. Make the 404 page a neutral language page and use a module that supports ML for your message.
   B. Make sure you have a unique portal-alias per language.
2. In Multi-portal environments, all of the portal 404 pages must have the same path / name.

Please note that by default, the DNN 404 page will not return page status to "not found", but to "OK (200)" when you use it for static file this ways. Our module overrules that behavior and sets the page status to 404 not found when a static file is handled.

Side Effects of logging 404s

When you log static 404s you should be aware that this will result in a long list of files.
Nowadays all websites get a lot of traffic by bots/hackers trying to get in.
A lot of this traffic will end up in your listings (like hacks for wordpress, joomla etc.)
This could potentially worry Admins and it would be best not to allow these attempts..
You can avoid this kind of traffic in many ways, but the most reliable we have found is by using URL rewrite rule in IIS.

You can block these requests by using the IIS URL rewriter functionality.
(Please note this is not installed by default, check that before dropping this in your web.config)

Here's an example of the rules we use

```
<rewrite>
        <rules>
                <rule name="CMS" enabled="true" stopProcessing="true">
                  <match url=".*" />
                  <conditions logicalGrouping="MatchAny">
                        <add input="{URL}" pattern=".*joomla.*" />
                        <add input="{URL}" pattern=".*/wp-.*" />
                        <add input="{URL}" pattern=".*filezilla.*" />
                  </conditions>
                  <action type="CustomResponse" statusCode="403" statusReason="Forbidden: Access is denied."
statusDescription="You do not have permission to view this directory or page using the credentials that you supplied."
/>
                </rule>
                <rule name="Extensions" patternSyntax="Wildcard" stopProcessing="true">
                  <match url="*" />
                  <conditions logicalGrouping="MatchAny">
                        <add input="{URL}" pattern="*.php" />
                        <add input="{URL}" pattern="*.ini" />
                        <add input="{URL}" pattern="*.ftpconfig" />
                  </conditions>
                  <action type="CustomResponse" statusCode="403" statusReason="Forbidden: Access is denied."
statusDescription="You do not have permission to view this directory or page using the credentials that you supplied."
/>
```

```
        </rule>
        <rule name="dnn-security" enabled="true" stopProcessing="true">
          <match url=".*" />
          <conditions logicalGrouping="MatchAny">
                <add input="{URL}" pattern="login\.aspx" />
                <add input="{URL}" pattern="register\.aspx" />
          </conditions>
          <action type="CustomResponse" statusCode="403" statusReason="Forbidden: Access is denied."
statusDescription="You do not have permission to view this directory or page using the credentials that you supplied."
/>
</rule>
</rules>
</rewrite>
```

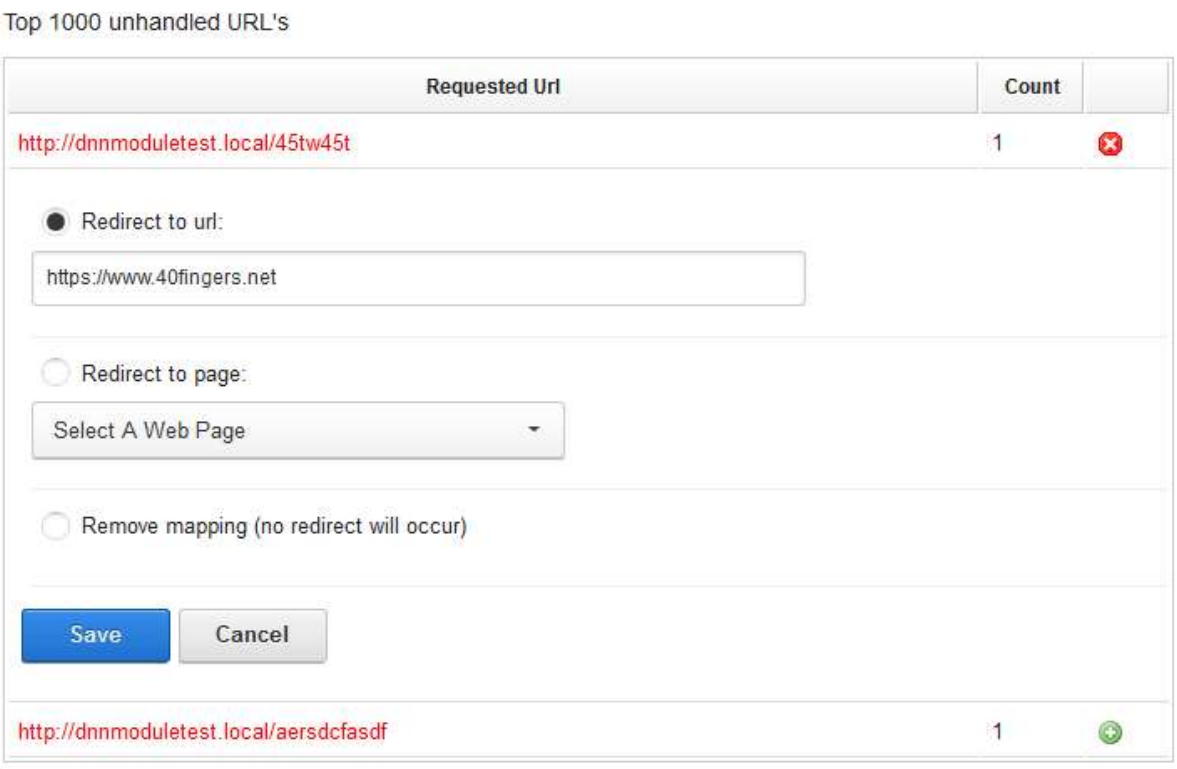
ADDING REDIRECT MAPPINGS

The module offers two option to manage redirect.

A. There's a list of logged URLs on the 404 page for Administrators
   Every URL has a + to allow adding redirects
B. There's a Module action to manage the Redirects.
   Here you can also add and remove Redirects.
   This is also the place to add Regular Expression redirects

Add mapping for listed URL

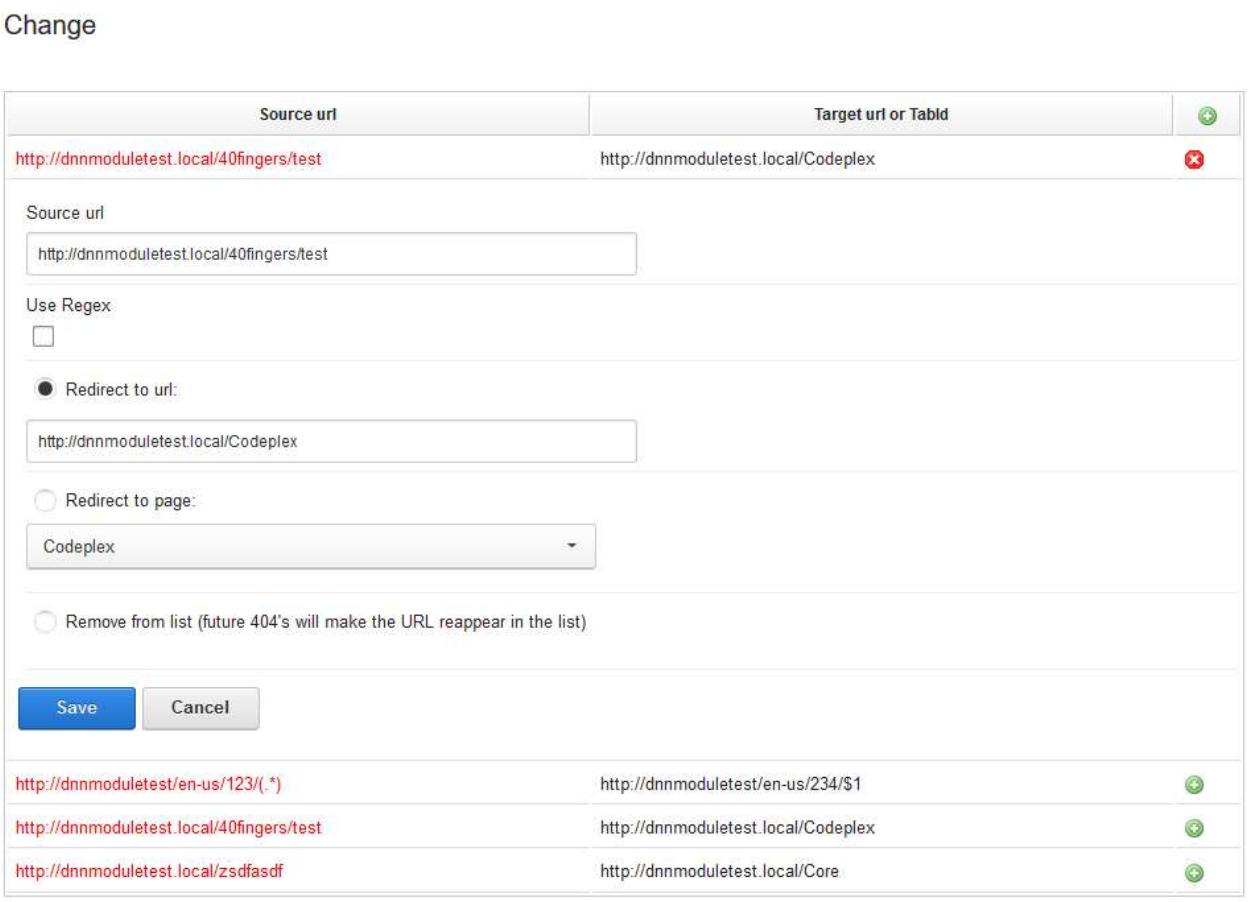
An admin will see the top 404s when logged in.



Here you can directly create a redirect for one of the URLs

Manage Mappings

Manage existing URLs through the Module Action “Edit Mappings”.
Here you can add, change or remove mappings.



DATA STORAGE OF REDIRECT LIST

The module stores above data in the “redirectconfig.xml” file, in the “\Portals\[PortalFolder]\40Fingers\SEORedirect”

folder. You can replace this file with one you generated from other sources.
A backup of the file is made every time you save a new version.

Generating a redirect list:
When you want to generate this list, make sure you also add a unique ID. This is used to identify the redirect internally. The module itself generates a guid when you add an item, but if the id is unique the module should pick them up correctly.

The structure is as follows:

```
<?xml version="1.0"?>

<RedirectConfig xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

  <Mappings>

    <Mapping>

      <Id>a7df76ca-2bd6-4fb4-bda8-e62c949a8ae8</Id>

       <SourceURL><![CDATA[https://www.40fingers.net/Service/Contact]]></SourceURL>

       <TargetURL><![CDATA[https://www.40fingers.net/Contact]]></TargetURL>

       <TargetTabId>685</TargetTabId>

       <UseRegex>false</UseRegex>

    </Mapping>

        <Mapping>

      <Id>690b62c0-9500-4441-96e0-77194780cf60</Id>

       <SourceURL><![CDATA[.*\service]]></SourceURL>

       <TargetURL><![CDATA[https://www.40fingers.net/]]></TargetURL>

       <TargetTabId>120</TargetTabId>

       <UseRegex>true</UseRegex>

    </Mapping>

  </Mappings>
```

</RedirectConfig>

Regex feature

The module also allows for regular expression parsing of URL paths, you can use regex like in the above screenshot (don't forget to check the "use regex" checkbox):

http://www.mydomain.nl/oldpath/(.*) meaning a wildcard to everything after ...path/

http://www.domainnew.nl/newpath/$1

The $1 references the above wildcard (you can use multiple variables)