# Terminalfour Silktide Integration

[Silktide](https://silktide.com/) provides an integration with Terminalfour to ensure pages that have recently changed are audited by the Silktide system. This document explains how to set up the integration to allow it to connect with the Silktide API.

## How it works

A PHP script placed on the CMS server, is triggered periodically using a CRON job. The script identifies a list of published files that were updated in the last publish and sends them to the Silktide API. The configuration of this script is done via the CMS where a web developer can turn the functionality on/off and determine the defer time for the re-scan.

## Features

* The script will send the information in batches of 1000 URLs by default
* The script will run once per publish time, a check it is in place to ensure that it does not run twice
* The script can configure the following information:
  + the number of seconds to wait before Silktide rescans the page
  + the number of files to send in bulk
  + the delay in seconds between batches
* If a script fails, the script will pick up all the missing URLs and send them to Silktide the next time it runs successfully
* If the local report was not saved for any reason and the script could not get the time it was last run, it will check URLs changed in the previous 2 hours
* The script can be disabled at any time

## Requirements

### Silktide API

* [Release: 14th September 2020](https://monitor.support.silktide.com/guides/change-log/))
* [Documentation](https://monitor.support.silktide.com/guides/cms-integrations/silktide-api-for-cms-integration/) (See Publishing paragraph)

### Terminalfour

* Version: 8.2+

### Server

* PHP 7.2+
* PHP extension: cURL
* Requires time-based job scheduler (CRON for Unix) - this can be set up on one of the web servers but it might be preferable on the CMS server

## 

## Installation Guide

### Before you start

Please ensure that you:

* are a Terminalfour administrator
* have a Silktide API Token as explained in the [Silktide Documentation](https://monitor.support.silktide.com/guides/cms-integrations/silktide-api-for-cms-integration/)
* have downloaded the Terminalfour installation files [Click to download T4 plugin](https://s3.amazonaws.com/assets.silktide.com/plugins/terminal-four-silktide.zip)
* have the [Section Metadata Content Type](https://docs.terminalfour.com/documentation/page-%20management/content-type/section-meta-data-content-type/) or similar system content type installed
* have access to perform SQL queries

### Installing the required files

The implementation requires some files to publish out and to run on the server in order to send the information to Silktide. Therefore, as a first step we are going to add the required files, before explaining how the integration can be configured.

1. Unzip the installation files. The zip contains 2 folders, content-types and media-library. We are going to focus on the media-library folder for now.
2. The media-library folder contains 4 files, 3 of them most likely will have a media type, so first thing to do is to ensure to have a media types available for PHP, JSON and PHAR
3. The media type will require a specific media layout so first go and create it, Go to > Assets > Content types
4. Select Media content type
5. Go to the Content Layouts tab and ensure that path/\* exists or otherwise create it as follow:
   1. Name: path/\*
   2. File Extensions: Default
   3. Syntax type: HTML/XML
   4. Content layout processor: T4 Standard Content
   5. Content Layout Code:
   6. Save it.

<t4 type="content" name="Media" output="file" modifiers="" />

1. After it, we need to create the media types: go to System administration > System settings > Media library
2. Select the Media Types tab and if PHAR Media Type does not exists create one as follows:
   1. Name: PHAR
   2. Permitted Extensions: phar
   3. Maximum file size: 0
   4. Media type options: [ x ] Binary File (select only Binary File)
   5. In Associated content layouts add a new row and select as Media content layout: path/\* and ensure it is selected as default layout (otherwise the script may not work as expected)
   6. Save it
3. After the PHAR do the same thing for PHP
   1. Name: PHP
   2. Permitted Extensions: php
   3. Maximum file size: 0
   4. Media type options: [ x ] Parse for T4 tags (select only parse T4 tags)
   5. In Associated content layouts add a new row and select as Media content layout: path/\* and ensure it is selected as default layout (otherwise the script may not work as expected)
   6. Save it
4. After PHP do the same thing for JSON
   1. Name: JSON
   2. Permitted Extensions: json
   3. Maximum file size: 0
   4. Media type options: [ x ] Parse for T4 tags (select only parse T4 tags)
   5. In Associated content layouts add a new row and select as Media content layout: path/\* and ensure it is selected as default layout (otherwise the script may not work as expected)
   6. Save it
5. After having all media types we can add the Media files inside the Media Library. Go to Content > Media Library
6. Add a new media category or select a pre-existing one as you wish
7. Add the 4 files as media files:
   1. utils.js: We will see this later on it will be used as part of the programmable layout that publish out the correct config information, or if you already have a programmable layout utils file you can merge the code into your own
   2. silktide-integration.phar: is PHP Archive that contains all the logic of the code
   3. run.php the script responsible for sending the information to the Silktide API
   4. config.json the file that helps to configure the script
8. Once the files are in the Media Library ensure to note down the Media ID of each file, we will use them when configuring the integration
9. We need to ensure the files get published otherwise the script will not work, to set this up please go to System administration > Set up sites & channels > Channels
10. Select the channel you’re installing the script on
11. In Publish options enable for the 3 media types just created and select "Always publish"
12. Before saving the note down the Output directory in the Output information section, this will be needed for later on in the installation guide
13. Save Changes

Now you should be able to publish the files.

To check this manually *run a publish*, since we turned on the option in the channel, a section publish of the Home section is enough. If all steps are working correctly you should see in the Publish Report, visible inside the [Notification Area](https://docs.terminalfour.com/documentation/administration/header--footer-menus/notifications/) or directly on the publish page (Site & Channels > Publish Channels), something similar to:



Please ensure to note down the exact folder where the files are being publish out to ([your-folder-path] in the above code block).

## Configuring the script

If the files are publishing out correctly you’re ready to proceed with setting up the script.

1. First we are going to set up the config.json properly, to do so Go to your Section Metadata content type
2. To [edit your Section Metadata Content type](https://docs.terminalfour.com/documentation/page-management/content-type/section-meta-data-content-type/#toc-edit-a- section-meta-content-type) you must first run an SQL query to temporarily change it from a system content type.
3. Once the Section Meta Data is a normal content type, you need to ensure you have a list called Yes, if you don't create a new one by going to Assets > Lists
   1. Name: Yes
   2. List Items: Yes
      1. Name: Yes
      2. Value: yes
      3. Selected: unchecked
      4. Sublist: None
4. Once you have the list we can use this in a new element we need to add inside the Section Metadata content type, Go to Assets > Content Types and select your Section Metadata content type (usually called Section Meta Description):
   1. Name: "Disable Silktide Integration"
   2. Description: "Check if you want to disable the Silktide Integration (the check should be valid only on the Home)"
   3. Type: Checkbox, List: Yes
   4. Use as filename: false
   5. Required: false
5. Once saved reset it back your Section Metadata content type back to System content type following [the same documentation](https://docs.terminalfour.com/documentation/page-management/content-type/section-meta-data-content-type/#toc-edit-a-section-meta-content-type)
6. Go back to Assets > Content Types and select your Section Metadata content type, this time going to Content layouts
7. Add the following content layout that is responsible to provide the value of the element just added
   1. Name: text/silktide-flag
   2. File Extensions: Default
   3. Syntax type: HTML/XML
   4. Content layout processor: T4 Standard Content
   5. Content Layout Code:
   6. Save it.

<t4 type="content" name="Disable Silktide Integration" output="normal" display\_field="value" />

1. Add another content layout, that will be used to set up the config.json information:
   1. Name: text/silktide-config-json
   2. File Extensions: Default
   3. Syntax type: Javascript
   4. Content layout processor: Javascript Content
   5. Content Layout Code: Copy the content of the file placed inside the installation file zip under /content-types/section-meta- description/text-silktide-config-json.js
   6. Before saving, in the code just copied replace at line 19 <!-- INSERT SILKTIDE TOKEN --> with the **Silktide API Token** (see Before you start paragraph) and <!-- INSERT ID--> with the **utils.js Media ID** just created at point 12.a in the previous paragraph.
   7. Save it.
2. Once we have the content layout for the config.js we need a navigation object that retrieve its content and we can place it in the config.json media file added previously. Therefore, go to Assets > Navigation
3. And create 2 new navigation objects, for both the content layout just created.
   1. Navigation Type: Related Content
   2. Name: Silktide Integration: get Silktide Config JSON for Current Channel
   3. Description: Returns text/silktide-config-json from Section Meta Description
   4. Primary Group: Global
   5. Options: [x] Show pending content in preview and [x] Cache output
   6. Output title: leave blank
   7. Fetch Method: Use Section
   8. Section: Please select the Home section of your channel
   9. Content layout: Use alternate content layout
   10. Alternate content layout: text/silktide-config-json
   11. Before HTML: leave blank
   12. After HTML: leave blank
   13. Save changes
   14. Note down the t4 tag for later
4. Now we need the second navigation object for Disable Silktide Integration with the following attributes:
   1. Navigation Type: Related Content
   2. Name: Silktide Integration: get Flag
   3. Description: Returns text/silktide-flag from Section Meta Description
   4. Primary Group: Global
   5. Options: [x] Show pending content in preview and [x] Cache output
   6. Output title: leave blank
   7. Fetch Method: Use Section
   8. Section: Please select the Home section of your channel
   9. Content layout: Use alternate content layout
   10. Alternate content layout: text/silktide-flag
   11. Before HTML: leave blank
   12. After HTML: leave blank
   13. Save changes
   14. Note down the t4 tag for later
5. Now we can finish the set up by adding our navigation objects in the Media Files, go to Content > Media Library and select config.json, in the editor tab replace all content with the new t4 tag created at point 10 and save
6. Edit run.php, go to the editor tab and for $config['config'] replace <!-- INSERT ID --> with the config.json Media ID and $config['library'] replace <!--INSERT ID --> with silktide-integration.phar Media ID, and $config['disable'] with the navigation t4 tag created at point 11 and save
7. Publish out the results

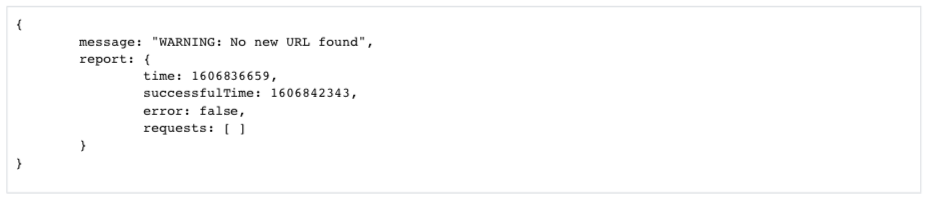
If everything works as expected you should see a live URL [https://www.[yourdomain]/media/[your-folder-path]/run.php](http://www/) a functional PHP script that returns a JSON format and display the report of the transaction made.

After this the only outstanding issue is to ensure to add a schedule task over the CMS server or one of the web servers that points to server location of the above script **[Channel Output Directory]/media/[your-folder-path]/run.php**

After that is done the Integration should be in place and working correctly.

## Read the report

The script added can be triggered manually by going directly to the URL and it will run the script and publish the report that will look like something similar to:



The message visible can vary depending on the work done by the script here to summarise all different statuses.

The format is in JSON

* message: will show the result of action for the script that has just been run
* report: contains details of the latest report
* report.time: timestamp when the latest report run
* report.successfulTime: timestamp of last time where the report was successful
* error: if the script run returned an error
* requests: the requested information sent to the Silktide API

### How to read the message

|  |  |
| --- | --- |
| **Message** | **Description** |
| WARNING: No new URL found | The script run correctly but did not find any new URL(s) to publish. The report shows the current empty report with no request in it. |
| WARNING: Script already executed for the last publish, please publish again to process a new request. | The script was already run for the latest publish, therefore the script does not run twice and a message its shown.  The report shown is the previous successful report. |
| SUCCESS: Sent request to Silktide | The script run correctly and did find URL(s) to send. The report will contain the URL(s) sent to Silktide. |
| ERROR: Issue during HTTP Request to Silktide | Shown when an error has occurred. It is possible that for example the Silktide API Token is incorrect or the script was not able to send to a request to Silktide.  The report shown is the current report which had the error. |

## Disable temporary the script

If for any reason you would like to temporarily disable the script, this can be set within the CMS. Disabling the script will mean both a manual trigger or the scheduled task will not perform any action and no request will be sent to Silktide.

1. Go to Site Structure and edit Home section
2. Under Advanced settings check "Disable Silktide Integration"
3. Publish Section

This will disable the script and not allow it to run until the flag is unchecked again and the home section is republished.

## Change the config file

Within the config file config.json file you can update settings for the integration. To do this:

1. Go to Assets > Content Type
2. Edit the Section Metadata content type
3. Go to content layouts tab
4. Select text/silktide-config-json
5. In content layout code it is possible to change the following variables
   1. silkTideApiURL = '<https://api.silktide.com/cms/update>' (the Silktide API Endpoint)
   2. silkTideApiKey = '<!-- INSERT SILKTIDE TOKEN -->' (the Silktide API Token)
   3. silkTideDeferSec = 1800 (the number of seconds to wait before Silktide will retest the URLs sent, ensure this leaves enough time for your published content to be transferred to your live web server(s). This will vary depending on your T4 set up)
   4. bulkLimit = 1000 (the number of URLs sent in a batch)
   5. delayInSeconds = 0 (the number of seconds to wait before each HTTP Request batches)
   6. basePage = 'index.php' (the basename to remove used as channel default)
6. Save the content layout
7. Run a publish section of your Home section to ensure that the new settings are applied
8. Check this by manually triggering the script in a browser window