# Introduction

Follow the procedures in this work instruction to add Google Analytics tracking code and feedback forms to your WebHelp systems.

# Download and install Ruby

1. Download the rubyinstaller-1.8.7-p352.exe file from the RubyInstaller site (rubyinstaller.org) to your hard disk and run it to install Ruby on your system.

# Download the script

* Download the file hpp.zip from <location> and unzip it to a folder on your hard disk, for example C:\hpp.

This work instruction assumes that the script is installed in C:\hpp.

Or:

* Clone the MDSOL hpp GitHub repository to your local machine.

The code in the master branch is always the production version of the script and is the same as the version stored in <location>.

# Prepare WebHelp system for processing

1. Follow the standard workflow to prepare your WebHelp system for publication.
2. Make backup copies of your WebHelp systems.

The script requires a previously-unprocessed WebHelp system. If you re-run the script on a WebHelp system, do so on a clean system that you backed up.

# Changing the Windows locale

This procedure applies only if you are processing Japanese files and the names of the HTML files contain Japanese characters. If not, go straight to [Move content files to a content folder](#_Move_content_files_1).

[Try processing Japanese files without performing this step first to see if it works. Unless you have the Enterprise or Ultimate edition of Windows 7, it shouldn’t.]

1. Select **Control Panel > Region and Language** to open the **Region and Language** dialog box.
2. Select the Administrative tab.
3. Click **Change system locale**.
4. Select “Japanese (Japan)” from the list of locales.

# Move content files to a content folder

This procedure applies only if your WebHelp system was created directly from RoboHelp. If your WebHelp was created by single-sourcing from FrameMaker, go to [Configure the script](#_Configure_the_post-processing).

In a WebHelp system created directly from RoboHelp, files containing the documentation content of your help system appear in the root folder with the structural files used to build the system. In a single-sourced WebHelp system, content files appear in a subfolder dedicated to content.

The script assumes that the WebHelp system is structured like a single-sourced system. To accommodate this:

1. Create a Content folder in the root folder of your help system (at the same level as the root .htm file of your WebHelp system).
2. Identify all .htm files containing documentation content and move them to this folder.

**Note:** Do not copy the root file of the help system (the .htm file used to launch the help system) or any other structural .htm file to the Content folder).

1. Configure the script with the location of this folder as described in the [Configure the script](#_Configure_the_post-processing) procedure.
2. After you have run the script as described in [Run the script](#_Run_the_script), move the content files back to the root folder of the help system and delete the Content folder.

# Configure the script

1. If you are using the script for the first time, rename the ruby\hpp-example.yml file to hpp.yml.
2. Update the product key in the hpp.yml file with the name of the product whose help system you are processing. For example, product: Balance.

The file includes a list of all the valid product names. [It doesn’t yet.] Copy the name of the product from this list into the product key.

1. Update the webhelp key with the full path and name of the root file of the WebHelp system.

For example, webhelp: C:/WebHelp/Balance.htm processes the WebHelp system in the C:\WebHelp folder whose root file is Balance.htm.

The root file is the HTML file used to launch the help system.

**Note**: always use forward slashes in paths in the hpp.ini file, even though Windows displays paths with backward slashes.

1. If the folder containing your WebHelp content files does not have the name of the root file without the .htm extension (the default), update the webhelp\_content\_folder key with the fully-qualified path of the content folder.

For example, if your root file is C:/WebHelp/Balance.htm and your content folder is C:/WebHelp/BalanceHelpText, specify the following keys:

webhelp: C:/WebHelp/Balance.htm

webhelp\_content\_folder: C:/WebHelp/BalanceHelpText

You can include <LANG> placeholders in the path. See the next step for an explanation of <LANG> placeholders.

If your WebHelp system was not created by single-sourcing from FrameMaker, you must create the content folder yourself and copy your content files to it as described in [Move content files to a content folder](#_Move_content_files).

If the WebHelp content folder has the default name, specify:

webhelp\_content\_folder: default

1. Update the language key with the standard three-letter code for the language of the WebHelp system you are processing. For example, language: ENG specifies an English WebHelp system.

If you store your WebHelp system in a path that includes a language folder, you can include <LANG> placeholders in the WebHelp path to tell the script to replace <LANG> with the language specified in the language key. For example, the settings

webhelp: C:/WebHelp/<LANG>/Balance.htm

language: ENG

tell the script to process the C:\WebHelp\ENG\Balance.htm WebHelp system.

This feature is mainly useful for processing multiple language versions of a WebHelp system at the same time. To do so, specify multiple languages separated by commas in the language key. For example, the settings

webhelp: C:/WebHelp/<LANG>/Balance.htm

language: ENG,JPN

tell the script to process the C:\WebHelp\JPN\Balance.htm and C:\WebHelp\ENG\Balance.htm WebHelp systems.

**Note**: You must always specify a value for the language key even if you do not use <LANG> in your paths.

# Run the script

1. Launch the Ruby command window by clicking **Ruby1.8.7-p352 > Start Command Prompt with Ruby** on the main Windows menu.
2. Type cd c:\hpp\ruby in the command window to navigate to the folder containing the script.
3. Type ruby hpp.rb to launch the script.

If the script runs without errors, the screen displays the following text:

File: <webhelp-file-name>

Working……Done!

1. If you see errors, verify that you have configured the hpp.yml file correctly, then re-run the script on a clean WebHelp system.
2. If the errors persist:
   1. Re-run the script on a clean WebHelp system with the command ruby hpp.rb > errorlog.txt to create an error log in the c:\hpp\ruby folder.
   2. Email the errorlog.txt file together with a zipped version of the WebHelp system to [help-metrics@mdsol.com](mailto:help-metrics@mdsol.com).