

Distinction Import Module *User Guide.*

Distinction *Import Module.*



Licence:

Copyright (c) 2018, Distinction Limited. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Disclaimer:

Distinction Limited provide no guarantees for this software and no dedicated support to its maintenance. The terms of the licence included with the package allow you to take, use and modify the code to fix and change anything as you see fit.

We welcome any bug reports, suggestions for improvements or stories of how you are using the control in your projects through the comments section at devnet.kentico.com, but please remember that we may not be able to respond to any requests as swiftly as we may like.

The module has been tested for some commonly used data classes (e.g. User, SKU, Page Menu item, etc) but in theory it provides the ability to perform updates to virtually any Kentico class type. Please ensure you have checked the documentation and know exactly what you are doing before you try and import into any class type.

Do NOT run on a production environment until you have first tested on a development environment. Distinction will take no responsibility for any errors or data corruption that occur from using this module.

Compatibility:

Only compatible with Kentico v11.

Release notes:

→ v1.0.4:

- Fixed bug with macro input assignment
- Guard against XSS from spreadsheet
- More detailed info in the results window
- Define RunImports permission for access to 'Run Import' tab

→ v1.0.3:

- Fixed bug with static input assignment
- Correctly order the columns in the sample spreadsheet download

→ v1.0.2:

- Added support for importing custom table data

- v1.0.1:
 - Update grid layout for profile columns
 - Use built in controls to handle ProfileColumn ordering
- v1.0.0:
 - Initial package allowing customised imports to Data Classes, TreeNodes and SKUTreeNodes

Known issues:

- **Bug:** When the 'Download Sample file' button is clicked, the Kentico 'loading' spinner displays and does not disappear once the file is downloaded.
Workaround: Clicking anywhere on the screen will cause it to disappear and you can continue as normal.
- If a spreadsheet is incorrectly formatted, it may cause problems during the upload.
 - Check there are no 'phantom' columns in the spreadsheet
 - Check each column is formatted according to the type of data that is contained within (e.g. text, number, etc...)

If in doubt, Download the sample file, and populate with your data.

Current limitations:

- Only compatible with .xlsx files
- Untested for Imports to BizForms
- Designed and tested for Imports in the order of 1000s of objects. Mass imports in the order of 10,000s or more may have implications on site performance.
- Workflow and Version History are not yet supported (module has not been tested as to how it currently performs for objects in a workflow).
- Import will process all configured field mappings and then save to database, with no concept of ordering or dependencies.

Roadmap:

- v1.0.4 - Support for BizForms
- v1.0.5 - Workflows/Version History.

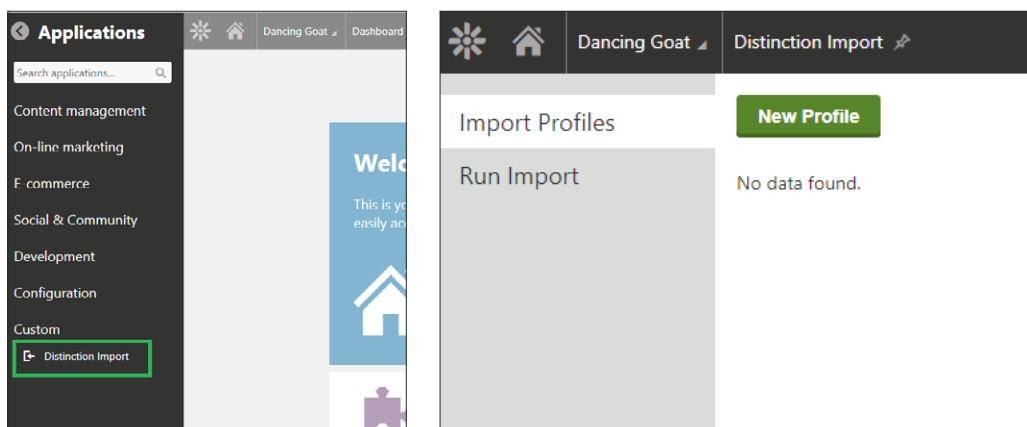
Future versions:

- Provides some logging in the event log, but would be good to have a dedicated log/audit tab within the module itself.
- Ability to store deltas and check if an object has changed before updating.
- Ability to manage dependencies/set order of updates to be processed (e.g. if users are being imported to a role, need to ensure the user insert happens before it tries to set to a role).
- Improved warnings/logging of restrictions/failures – e.g.
 - If not all required fields on an object are being set.
 - If there are restrictions adding page under specified parent node.

User Guide.

Follow the steps in the documentation to install the Distinction Import module: <https://docs.kentico.com/k11/custom-development/creating-custom-modules/creating-installation-packages-for-modules/installing-modules-from-installation-packages>.

Once installed, the Import module displays under the Custom section in the module/application list.



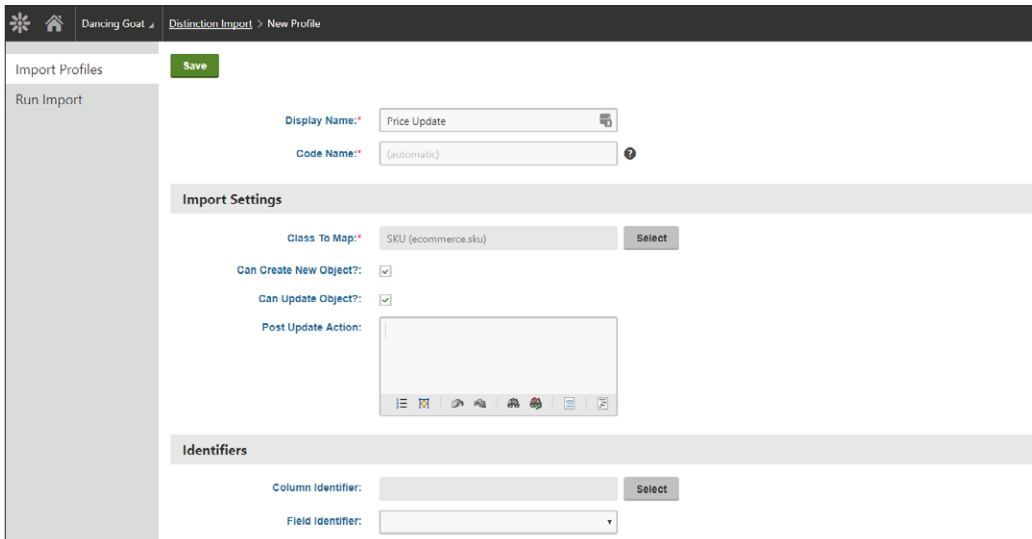
In the module, there are 2 tabs:

- Import Profiles: An 'Import Profile' is a set of rules for how a spreadsheet should be processed – what type of data it is, what to do with each column, etc. This section is where you create and manage all of your profiles.
- Run Import: This is where you upload a spreadsheet and run it based on a specify profile to import the data.

We'll get started by creating our first Import Profile.

Creating an Import Profile.

Step 1: General settings

A screenshot of the 'Distinction Import > New Profile' web interface. The interface has a dark header with navigation icons and the text 'Dancing Goat', 'Distinction Import', and 'New Profile'. On the left is a sidebar with 'Import Profiles' and 'Run Import'. The main area contains a 'Save' button, 'Display Name' (Price Update), 'Code Name' (automatic), an 'Import Settings' section with 'Class To Map' (SKU (ecommerce.sku)), 'Can Create New Object?' (checked), 'Can Update Object?' (checked), and 'Post Update Action' (empty), and an 'Identifiers' section with 'Column Identifier' and 'Field Identifier' dropdowns.

Basic settings:

- **Display Name:** Human readable name of the import profile you're creating.
- **Code Name:** String identifier for the profile (automatically set)

Import settings:

- **Class to Map:** Specify the type of object you are importing the data to. There are 4 categories of class in Kentico:
 - **TreeNode:** Standard Page Types that can be added in the Content Tree.
 - **SKUTreeNode:** As above but are linked to a SKU so represent a product.
 - **Data Class** (e.g. User, stand-alone SKU)
 - **Custom Table:** Tables manually configured by an administrator in the Custom Tables module.

The type of object will affect the fields that are available to map to and where there is an option to set a parent node (see below).

- **Can Create New Object:** Specify if you want to allow inserts into the database.
- **Can Update Object:** Specify if you want to allow existing objects to be updated.
- **Parent node (Only visible for TreeNode classes):** If you have selected a TreeNode or SKUTreeNode class to map, you will need to specify a parent node. All pages will then be created directly under this node.

WARNING: you will need to make sure that scopes and allowed types are correctly configured in the Page Types module to allow for this.

- **Post Update Action:** This is where you can add your own customisations into the import. After each object has been Inserted/Updated, you can specify a macro method to be run.

In most cases, you will need to pass the current object into your macro method. Do this with 'CurrentObject', e.g.:

```
{% MyMacroMethod(CurrentObject) #%}
```

Identifiers:

Please note: When you are first creating an import profile, you will need to stop at this section, save the profile and define the columns (see Step 2) before you can set the identifiers.

The purpose of the identifiers is to determine if an object exists in the database already, and therefore if we need to update it, or create a new object.

For example, if we importing a user with email 'pete@example.com', we want to tell the import to do a database lookup to see if there is an existing user with the email 'pete@example.com'.

- If there is, we want the import to retrieve that user and update the details (providing we have set 'Can Update Object' above).
- If not, we want the import to create a new user with this email address (providing we have set 'Can Create New Object' above).

Setting the identifiers is how we can tell the import which column and field to use for that database lookup.

- **Column Identifier:** Define what column in the spreadsheet contains the unique identifier.
- **Field Identifier:** Define what field on the data type stores the unique identifier.

Sample

Although this section is empty when you first create the Profile, the 'Download Sample file' button will appear when you have defined the columns in the spreadsheet.

This feature allows you to generate a template spreadsheet with the required column headings and sample data. This may be helpful if the data is being populated by someone else, so that they have a clear idea of what is expected.

The screenshot displays the 'Distinction Import' module interface for configuring a 'Price Update' import. The left sidebar shows a navigation menu with 'General', 'Columns', and 'Mappings'. The main content area is titled 'Price Update (ObjectType:distinctionimport_profile)' and includes a 'Save' button. The configuration is divided into several sections: 'General' with 'Display Name' (Price Update) and 'Code Name' (PriceUpdate); 'Import Settings' with 'Class To Map' (SKU (commerce_sku)), 'Can Create New Object?' (checked), 'Can Update Object?' (checked), and a 'Post Update Action' field; 'Identifiers' with 'Column Identifier' (Product Code) and 'Field Identifier' (SKU Number); and a 'Sample' section with a 'Download Sample file' button.

Step 2: Define columns

This is where we define the columns in the spreadsheet that we will be importing. Each column definition consists of:

- **Column Display Name:** The human readable name of the column
- **Column Name:** Auto-generated from the display name with spaces and illegal characters removed. This is what will be used (and expected) in the spreadsheet.
- **Sample Data (optional):** This allows you to give a sample of the type of data you are expecting which will populate the sample file (downloadable from the 'General' tab).

Actions	Display name	Code name	Sample Data
	Product Code	ProductCode	A1234
	New Price	NewPrice	9.99

When the import runs, it will validate the spreadsheet against the define columns and will not proceed if they do not match. The order of the columns is therefore important and can be changed with the up/down arrows in the Column listing.

Step 3: Define mappings

We now need to tell the profile what to do with the data. We therefore define a mapping for every action that we need to be performed during the import. This will either be updating a field on the specified class or calling a macro method.

The screenshot shows the 'Distinction Import' module interface. The breadcrumb trail is 'Dancing Goat > Distinction Import > Price Update (ObjectType.distinctionimport_mappedcolumn)'. On the left, a sidebar contains 'General', 'Columns', and 'Mappings', with 'Mappings' selected. A green 'Save' button is at the top left of the main area. The configuration fields are: 'Type of Input:' with a dropdown set to 'Data from column cell'; 'Profile Column:' with a text input 'NewPrice' and a 'Select' button; 'Assign To a Field?:' with a checked checkbox; 'Field To Assign:' with a dropdown set to 'SKUPrice'; and 'Use as additional Identifier:' with an unchecked checkbox. A note below the last checkbox states: 'This can only be used for imports to Data Classes'.

- **Type of Input:** Defines what input we want to use for this particular mapping. There are 3 options:
- **Data from column cell:** Will get the cell value for each row in the spreadsheet based on the Profile Column we specify below.
 - **Static Input:** Allows you to set a value that will be the same for every record you are importing (for example, if you are importing to the SKU class and want to ensure that all SKUs are enabled, rather than defining an extra column in the spreadsheet, you can just create a Static Input with an input value of 'true').
 - **Macro Expression:** Allows you to specify a macro expression which will be executed for each record.
- **Profile Column (hidden for 'Static Input' type):** Specifies which Column in the spreadsheet you will get the input value from.
- **Static Input (displays for 'Static Input' type):** Sets the static value to be applied to all records.

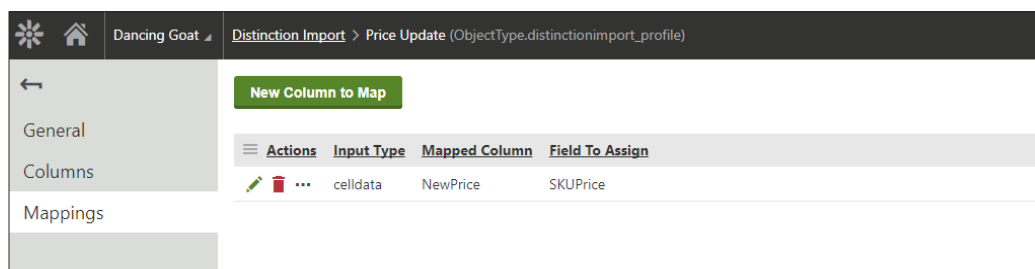
- **Macro Expression (displays for 'Macro Expression' type):** Defines the macro method that will be executed against every record. This allows great flexibility to perform custom actions during the import. The macro expression may return a value which you can then assign to a field (see below), or in some cases it may perform an action without returning a value and so does not require assigning.

You can reference the input value in your expression using 'DataValue', and you can reference the current object using 'CurrentObject', e.g.:

```
{% MyMacroMethod2(DataValue, CurrentObject) #%}
```

- **Assign to a Field:** Specifies if you want to set the input value to a field on the class. The only case where this should be false is if you are using a macro expression which performs a desired action without returning any value.
- **Field to Assign:** Specifies which class field to assign your input value to.
- **Use as additional Identifier:** In Step 1, we defined the Identifiers, which tell the import which column and field to use for trying to find an existing record. In most cases, a class will only require one identifier, however in some cases, a second (or even third) identifier may be needed to find the correct record.

For example: the 'User role' class handles the link between users and the roles they have been assigned to. In order to determine if a record exists we would need to search for a UserID and a RoleID.



Unlike the Column definitions, the order of the mappings is not important.

Running the Import.



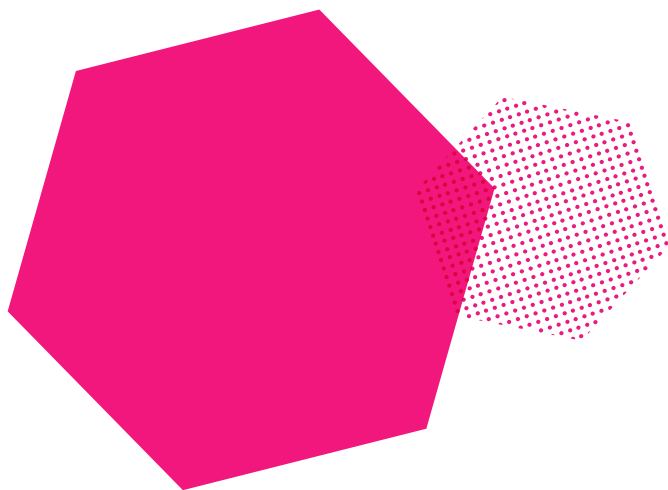
WARNING: It is strongly recommended that you test any import in a development environment before running in production.

Once you have defined your Import Profile and populated the spreadsheet, you are ready to run the import. Go to the 'Run Import' table and select the profile you wish to use, along with the spreadsheet file.

When you hit import, the file will be validated to check that it is in the correct format. If not, a message will be displayed and the import will be cancelled. The details of why the file format was wrong are recorded in the event log.

If the format is correct, the import will run. The progress box will display details about the import, how many rows were processed and any failures that were detected. More in depth information can be viewed in the event log.

The screenshot shows the 'Run Import' interface. At the top, there's a navigation bar with 'Dancing Goat' and 'Distinction Import'. Below this, a sidebar on the left contains 'Import Profiles' and 'Run Import'. The main content area is titled 'Run Import' and includes a note: 'The process will ignore the first line as the 'header''. Below this, there's a 'Select Import Profile:' dropdown menu with 'Price Update' selected. A 'Choose file' button is next to 'No file chosen', and an 'Import now' button is to the right. A 'Progress...' section follows, containing a text box with the message: 'Valid spreadsheet. Starting Import...' and 'Import complete. 1 rows processed. See event log for more details.'

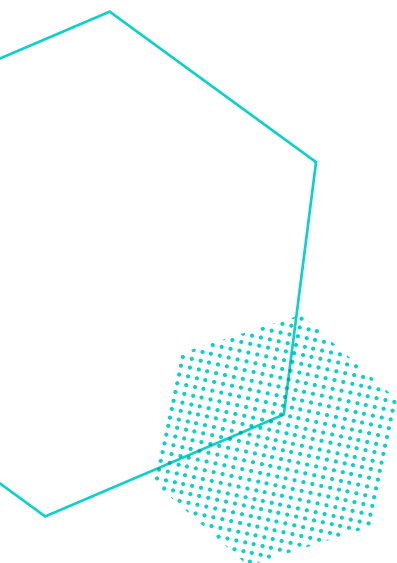
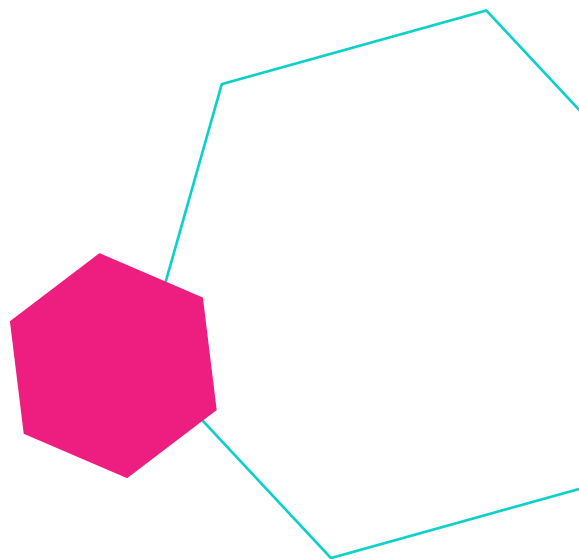


Let's work together

Get in touch

+44 (0) 115 704 3011

work@distinction.co.uk



Nottingham

4-6 Broadway, The Lace Market
Nottingham, NG1 1PS, UK

London

The Stanley Building, 7 Pancras Square
London, N1C 4AG, UK

Boston

470 Atlantic Avenue, 4th Floor
Boston, Massachusetts 02210, US