Pivot Collection Tool

Command Line User’s Guide

# Introduction

This document describes how to use the Pivot Collection Tool, and what its distribution package contains. The tool itself is a command-line tool which converts collections from one format to another, and provides optional support for creating or augmenting images associated with the collection. This document describes how to use the tool, what formats are supported, how to create your own templates for image creation, and a brief tutorial on how to perform basic operations.

# Distribution Package

This command-line Pivot Collection Tool is distributed as a single ZIP archive containing the following elements:

* The pauthor.exe executable and related libraries
* A small sample collection based upon an existing, published collection
* This user’s guide

# Command-Line Tool

The command line tool accepts a number of arguments to invoke its various functions and direct its output.

## /source <format> <file>

This mandatory argument specifies the source for the collection to be converted. The format parameter may be one of the following: excel, cxml, or deepzoom. Each of these formats is described in more detail in the Formats section below. The file parameter should specify the file containing the collection data to be converted (e.g., the Excel document or CXML file).

## /target <format> <file>

This mandatory argument specifies where and in what format the converted collection should be written. As with the source argument, the format parameter may be: excel, cxml, or deepzoom, and the file parameter must specify the output file where the collection will be written.

## /html-template <file>

This optional argument specifies a specially formatted HTML file as described in the HTML Template section below. This will cause the HTML template to be rendered with each item’s individual data, and its image replaced (if one was already present) or created (if not). By using the template parameters (as described below), the user may create a unique image for each item based upon the item’s facet values (including its current image).

# Formats

## Excel

The Excel format[[1]](#footnote-1) allows users to specify all aspects of a collection in a series of sheets in an Excel document. The document may optionally provide the following sheets: collection, facet\_categories, and items. The collection sheet describes the attributes of the collection as a whole. The facet\_categories sheet specifies the names and types of the facet categories in the collection. The items sheet contains the data for the individual items in the collection.

The collection sheet is expected to have the columns described in the following table. These labels may be placed in any order along the first row of the sheet with values provided in the second row. If a column is omitted, then that value will be left out of the final collection. If the sheet is omitted, then all the values will be omitted from the final collection.

|  |  |  |
| --- | --- | --- |
| Column | Format | Description |
| name | String | The user-visible name of the collection. |
| icon | Path | The favicon for the collection. Absolute path or path relative to the Excel file. |
| brand\_image | Path | The brand image to be displayed with the collection name. Absolute path or path relative to the Excel file. |
| additional\_search\_text | String | Text to be appended to an item's name to form the Bing search query. Use "\_\_block" to omit the search. |
| copyright\_title | String | The title of the copyright link in the metadata pane. |
| copyright\_url | URI | The URI of the copyright link in the metadata pane. |

The facet\_categories sheet specifies the names and types of the facet categories in the collection. It may have the columns described in the following table in any order. The names are expected to be in the first row with each facet category defined in the subsequent rows. Of all the columns in the facet\_categories sheet, only name and type are mandatory. If they are omitted, then the converter will stop and emit an error message. If any of the is…Visible columns are omitted, then they will be assumed to be true (when the type permits). If the sort\_values column is included, but sort\_name is not, then the sort name will be assumed to be the name of the facet category. If sort\_name is provided, but sort\_values is not, then a basic alphabetic sort will be included for that facet category. The sort\_values column may contain multiple values each on a new line (press Alt-Enter when editing the cell to add a new line). If the facet\_categories sheet is omitted, then facet categories will be inferred from the items themselves. Each named column which appears in the items sheet will be converted into a String facet which has is…Visible set to true for all values

|  |  |  |
| --- | --- | --- |
| Column | Format | Description |
| name | String | The user-visible name of the facet category. |
| type | Path | The type of the facet category (can be: String, LongString, Number, DateTime, or Link) |
| format | Path | A format string compatible with .NET DateTime or Number formatting conventions. |
| is\_filter\_visible | Boolean | Whether this category should be in the filter pane. |
| is\_metadata\_visible | Boolean | Whether this category should be in the metadata pane. |
| is\_wordwheel\_visible | Boolean | Whether a text filter will include this category. |
| sort\_name | String | The name of a custom sort for this category |
| sort\_values | String | An ordered list of values for the custom sort (each on a new line: use Alt-Enter for a new line) |

Finally, the items sheet specifies the data for each individual item. The sheet may have any of the columns described in the table below, as well as a column for each of the facet categories specified in the facet\_categories sheet. Columns for facet categories may have multiple values separated by new lines[[2]](#footnote-2). If a facet\_categories sheet was provided, only the facet category columns and the special columns described below will be used. All other columns will be ignored. If a facet\_categories sheet is not provided, new facet categories will automatically be created for each column in the items sheet. If the items sheet is omitted entirely, then the first sheet in the workbook will be assumed to contain item data, and handled accordingly.

|  |  |  |
| --- | --- | --- |
| Column | Format | Description |
| name | String | The user-visible name of the item |
| image | Path | The image for this item. Absolute path or path relative to the Excel file |
| description | String | A textual description of this item. May be several paragraphs long, but must be plain text. |
| href | URI | The URI launched by the item's "Open" button |

When this format is used as output, the images from the source collection will be copied into a *file*\_images directory (e.g., writing to foobar.xlsx would result in a foobar\_images directory). All of the described sheets will be created, but the values may be left blank if there was no corresponding data in the source collection.

A sample Excel file has been provided in the samples directory of the distribution.

## CSV

The CSV format has allows users to specify a collection completely using CSV formatted text files. If only a single CSV file is provided, then the facet categories will be assumed from the contents of the file in just the same way as if the facet\_categories sheet were omitted from an Excel document. However, the user may provide an optional *file*\_facetcategories.csv and *file*\_collections.csv files to serve the same function as the facet\_categories and collection sheets in the Excel format. The format expected for all three files is the direct analog of that expected in the Excel format.

When this format is used as output, the images from the source collection will be copied into a *file*\_images directory (e.g., writing to foobar.csv would result in a foobar\_images directory). All of the described files will be created, but the values may be left blank if there was no corresponding data in the source collection.

A sample CSV file has been provided in the samples directory of the distribution.

## CXML

The CXML format allows users to specify a collection using the CXML format, but with each item referring to a normal image file instead of DeepZoom artifacts. The CXML file should be fully compliant with the CXML XSD, and the Img tag for each Item tag should specify that item’s image with either an absolute path to the image file, or path relative to the CXML file (URLs are not permitted). Items with no associated image will likewise have no image in the final collection (unless the /html-template argument is also used). If the specified image file cannot be read, then the item will be treated as though no image was specified. In either case, a warning will be written to the console. If two images have the same file name (even if in different directories), then the output behavior is undefined.

When this format is used as output, in addition to the CXML file being generated, the images from the source collection will be copied into a *file*\_images directory (e.g., writing to foobar.cxml would result in a foobar\_images directory).

Please note that this format is not suitable for viewing in Pivot; use DeepZoom for that purpose!

## DeepZoom

The DeepZoom format allows users to specify a CXML file which has associated DeepZoom artifacts (e.g., a DZC file, DZI files, and image pyramids). This behaves identically to the CXML format except that the ImgBase attribute on the Items tag must be defined and refer to the DZC file (either an absolute path, or a path relative to the CXML file), and the Img attributes on the Item tags must refer to the index of that item’s related entry in the DZC (e.g., “#4”).

When this format is used as a target, a directory called *file*\_deepzoom will be created adjacent to the CXML file (e.g., a directory named foo\_deepzoom will be created for a CXML file called foo.cxml). That directory will contain all of the DeepZoom artifacts necessary to display the collection in Pivot.

## HTML Template

Images may be created or augmented by using an HTML template. The template should be a standard HTML file (including CSS and JavaScript), which has a few placeholders to be replaced with facet values for individual items. By default, the template will be rendered at 1200 x 1500 pixels using the IE 7 rendering engine. To change the size of the image, include the following comment in the HTML template to set the size: <!-- size: *width*, *height* -->

The HTML template should contain special placeholders to indicate where data from each item should be placed. The tags can take several forms: Single Value, Indexed Value, and Joined Values. A Single Value tag should look like: {*facet category name*} (e.g., {attendee}). Wherever the tag appears, it will be replaced with the value of the given facet category for that item. If the item has no value for that facet category, the tag will be replaced with an empty string.

If an item has multiple facet values for a given facet category, the template may include a specific value by using an Indexed Value tag: {*facet category name*:*index*} (e.g., {attendees:2}). This will replace the tag with the specified value for the given facet category of each item. If the item does not have a value at the given index (or doesn’t have any value for that facet category), the tag will be replaced with an empty string.

Finally, in order to include all the values for a multi-valued facet, the Joined Value tag may be used: {<facet category name>:join:<delimiter>} (e.g., {attendees:join:, }). This will replace the tag with a string which combines each facet value, separated by the given delimiter string. The delimiter can be any sequence of characters (except the ‘}’ character).

In addition to the items facets, there are a few additional placeholders related to the item’s standard attributes. These are: {name}, {image}, {href}, and {description}.

|  |  |  |
| --- | --- | --- |
| Tag | Example | Description |
| {name} | {name} | The name of the item |
| {image} | {image} | The absolute path to the item's image |
| {href} | {href} | The URI launched by the item's "Open" button |
| {description} | {description} | The textual description of the item |
| {*facet*} | {attendee} | The value of a specific facet category for this item |
| {*facet*:*index*} | {attendees:2} | A single value of a multi-value facet category |
| {*facet*:join:*delimiter*} | {attendees:join:, } | A string combining all of the values for a facet category using a given delimiter |

# Tutorial

For the sake of this tutorial, let’s assume that you’re a developer who would like to turn some existing data into a Pivot collection. You have some images for your data, but they really don’t stand on their own, so you’d like to use the HTML Template system to create better visuals for your items.

To start, you’ve written some code which creates a CXML file from your data. Each item references a local image file. Now, you’d like to view that collection in Pivot. To convert your collection and images into DeepZoom images, you’d run the following command:

Pauthor.exe /source cxml MyCollection.cxml

/target deepzoom output\MyCollection-dz.cxml

You now open up your collection in Pivot, and see that all your items are there, but they’re just using the original images. You’d like to introduce your HTML template to augment the bare images. Starting with your original collection, you run the following command:

Pauthor.exe /source cxml MyCollection.cxml

/html-template MyTemplate.htm

/target deepzoom output\MyCollection-dz.cxml

Now you have a Pivot collection with your custom images, and as you’re checking it over, you decide you’d like to add a facet which is derived from the values in two other facets. Using Excel would make that really easy, so you decide to convert your collection into an Excel spreadsheet:

Pauthor.exe /source deepzoom output\MyCollection-dz.cxml

/target excel MyCollection.xlsx

You add a new row to the facet\_categories sheet for the new facet category, and then you add a new column to the items sheet which has the formula to calculate the new value. Once everything is finished, you convert the Excel spreadsheet back to a Pivot collection:

Pauthor.exe /source excel MyCollection.xlsx

/target deepzoom output\MyCollection-dz.cxml

You check everything over, and it looks just right. Now your collection is ready to show off!

# Prerequisites

The Pivot Collection Tool comes with most of what it needs, but there are a few prerequisites for using certain formats. In order to use the CSV and Excel formats, you must have Microsoft Office 2007 (or later) or Microsoft Excel 2007 (or later) installed on your computer. In order to view collections in the DeepZoom format, you must have the Microsoft Live Labs Pivot application installed.

# Known Issues

### One of my facet values is getting truncated when converting from Excel!

This happens to long facet values (or facets with many value) when the value of that facet for the first several items in your spreadsheet are shorter than 255 characters. The tool uses OLE DB to read and write information from spreadsheets. When OLE DB reads an Excel document, it attempts to determine whether the field should be represented as a TEXT field (i.e., less than 255 characters) or a MEMO field (i.e., any size) by reading the first few rows of the sheet. If no long values are found, the OLE DB driver assumes the column contains short values and only returns the first 255 characters.

You can work around this problem by re-ordering the rows of your spreadsheet to make sure a long value is included within the first few rows.

### Pivot behaves strangely when viewing my collection created using the “cxml” format!

The “cxml” output format creates a CXML file which references raw JPEG images. While Pivot technically can load such collections, they are much slower and more memory intensive than collections which use DeepZoom artifacts. Occasionally, Pivot will not even be able to load all images from such a collection.

To work around this problem, use the “deepzoom” format for collections you would like to view in Pivot.

### Some of my items are missing their images when using the HTML template!

When generating images using the HTML template system, the tool creates a copy of Internet Explorer 7 to perform the rendering. Sometimes, when using very large images (i.e., larger than 3000 pixels on a side), Internet Explorer will run out of memory and be unable to load the image.

To work around this problem, you can either: pre-shrink your images to roughly the size of your final output, quit all other programs and try again, or run the tool multiple times on subsets of your full collection.

1. Note: This format is different from that supported by the previously released Pivot Excel Plugin. With this format, you can create a Pivot collection from an arbitrary spreadsheet. [↑](#footnote-ref-1)
2. Press Alt-Enter when editing the field in Excel [↑](#footnote-ref-2)