|  |
| --- |
| Microsoft Live Labs |
| PauthorLib Overview |
| A brief orientation to the Pauthor library |

|  |
| --- |
| Andrew Miner  6/15/2010 |

# Purpose

This document provides a high-level overview of the Pauthor library. It consists of a general description of the various namespaces, key classes they contain, and the relationships between them. This document is not an API reference, nor does it give specific guidance on how to use the API to accomplish particular tasks (refer to the other documents included in this distribution for those purposes). This document assumes the reader is already familiar with C# programming concepts, the Pivot application or Silverlight PivotViewer, and the information covered in the Developer Info section of the Pivot website**[[1]](#footnote-1)**.

# Namespaces

The Pauthor library is organized into the following namespaces:

|  |  |
| --- | --- |
| Core | Contains the core classes which define a Pivot collection. |
| Streaming | Contains interfaces which allow you to treat a collection as a stream of items, and implementations to read/write collections to CXML |
| Streaming.OleDb | Contains collection stream implementations which use OLE DB to access data stored in supported formats. |
| Streaming.Filters | Contains interfaces to define filters which alter collection streams as they are read / written, and a few useful implementations |
| Imaging | Contains classes which produce images for items in a Pivot collection. |

# Core Namespace

The Core namespace contains a straight-forward object model of a Pivot collection. This model attempts to reflect the behavior of the various Pivot clients as completely, correctly, and accurately as possible. This means that it contains all the various concepts, properties, and other data available in any of the Pivot viewers, it enforces the rules imposed on collections by those viewers, and (with a few exceptions) makes it impossible to create an object model which is not valid. So much as possible, this model reflects the same concepts, relationships, and terminology as is used in the Pivot Collection XSD and existing Pivot documentation.

## Class Overview

* PivotCollection: an in-memory representation of an entire Pivot collection. Implements ICollectionDefinition, and contains multiple PivotFacetCategory objects and multiple PivotItem objects.
* PivotItem: represents a single item in a Pivot collection. Refers to an ICollectionDefinition to validate individual facet values assigned to it.
* PivotFacetCategory: describes a single aspect of an item in a Pivot collection. Contains a PivotFacetType object and potentially a PivotFacetSortOrder object.
* ICollectionDefinition: summarizes the definition properties of a Pivot collection. It is implemented by PivotCollection in this namespace and IPivotCollectionSource in the Streaming namespace. It is used by PivotItem to verify facet values assigned to it.
* PivotList: a mix of a List and Dictionary. It is used by PivotCollection to contain both PivotFacetCategory and PivotItem objects.

# Streaming Namespace

While the Core namespace assumes that an entire collection will be loaded and manipulated in memory all at once, the Streaming namespace treats collections as streams of items which are each handled one by one. The stream interfaces use many of the classes from the Core namespace, and are convertible between the Streaming and Core models. Using the classes in this namespace makes it possible to write code which can handle collections of any size, as well as read and write them from / to various file formats.

## Class Overview

* IPivotCollectionSource: defines an interface for a readable stream of a Pivot collection. Implements ICollectionDefinition from the Core namespace. All classes which define collection sources or collection source filters implement this interface.
* IPivotCollectionTarget: defines an interface for a writing a Pivot collection stream. All classes which define collection targets or collection target filters implement this interface.
* PivotCollectionBuffer: wraps an in-memory PivotCollection and allows it to serve as either a collection source or collection target
* LocalCxmlCollectionSource: a streaming collection read from a CXML file stored locally on disk
* LocalCxmlCollectionTarget: writes a given collection source to a CXML file on the local file system

# Streaming.OleDb Namespace

This namespace augments the Streaming namespace by adapting OLE DB to the collection stream interfaces. This includes a base class which can be given any OLE DB connection string and queries, as well as specific subclasses which provide implementations for Excel and CSV formats.

## Class Overview

* OleDbCollectionSource/Target: collection source which draws data from an arbitrary OLE DB connection
* ExcelCollectionSource/Target: a subclass of OleDbCollectionSource which specifically handles the Microsoft Excel 2007 format
* CsvCollectionSource/Target: a subclass of OleDbCollectionSource which specifically handles the CSV format

# Streaming.Filters Namespace

This namespace defines source and target filters for collection streams, and provides several helpful implementations of each. Each filter implements one of the collection stream interfaces defined in the Streaming namespace, but delegates nearly all calls to some other instance. The filter will selectively override certain methods to transform the data read from the underlying instance according to the rules for that filter.

Let's consider an example of a source filter which capitalizes the name of each item in a collection. Such a filter would inherit from the PivotCollectionSourceFilter class, and would only need to override the Items get method. When an instance is created, it would be passed the actual collection source (e.g., an instance of LocalCxmlCollectionSource) which contains the data for the collection. Then, in the "get" method, it would read the items from the underlying collection one-by-one. For each item, it would read the name of the item, capitalize it, and then update the item with the new value. It would then "yield return" and proceed to the next item. At no point would the filter handle more than one item at a time. Since a filter's delegate can be any implementation of a collection stream, it is possible for multiple filters to be used in conjunction with one another.

## Class Overview

* PivotCollectionSourceFilter: an abstract base class for classes which wish to alter collection streams. Implements IPivotCollectionSource.
* PivotCollectionTargetFilter: an abstract base class for classes which wish to alter collection streams as they are written into a collection target. Implements IPivotCollectionTarget.
* DeepZoomTargetFilter: alters a collection stream being written through it to include DeepZoom artifacts instead of raw images. Inherits from PivotCollectionTargetFilter.
* UnDeepZoomSourceFilter: alters a collection source by converting DeepZoom images back into normal JPEG images. Inherits from PivotCollectionSourceFilter.

# Imaging Namespace

This namespace contains two stand-alone classes which can create images for items in a collection plus a base class which contains common functionality.

## Class Overview

* ImageCreator: an abstract parent class for objects which deal with creating images
* HtmlImageCreator: creates bitmap images by capturing a rendering of an HTML template
* UnDeepZoomImageCreator: converts a DeepZoom image back into a plain bitmap image

1. http://getpivot.com/developer-info [↑](#footnote-ref-1)