This page describes the data types that you can work with when implementing integration connectors.

Classes

BaseInfo

Represents any siterelated or global data object in Kentico (for example a page template or a poll).

TreeNode

Represents a page in the content tree of a website.

Interfaces

ICMSObject

An interface implemented by both *BaseInfo* and *TreeNode* objects. Allows you to identify Kentico data types.

Enumerations

TaskTypeEnum

Namespace: CMS.DataEngine

Always accompanies processed objects and pages. The value has a slightly different meaning for outgoing and incoming tasks:

- for outgoing tasks – used when creating subscriptions and determines which actions are handled as integration tasks. For example, when you subscribe to *CreateObject*, the integration bus logs synchronization tasks when objects inheriting from *BaseInfo* are created.
- for incoming task – determines what happens to the target object when processing tasks. For example, when you pass a page object and task type *DeleteDocument*, the integration bus deletes the page.

The most common values for **objects** are:

- CreateObject
- UpdateObject
- DeleteObject
- AddToSite (not applicable for global objects)
- RemoveFromSite (not applicable for global objects)
- All – only makes sense for the outbound direction (for subscriptions). Indicates that the integration bus creates tasks for all types of object operations and changes.

The most common values for **pages** are:

- CreateDocument
- UpdateDocument
- DeleteDocument
- PublishDocument
- ArchiveDocument
- All – only makes sense for the outbound direction (for subscriptions). Indicates that the integration bus creates tasks for all types of page operations and changes.

To see the whole list of values, explore the enumeration in Visual Studio.

Note: There is no task type that indicates transitions of pages between workflow steps (except for the *PublishDocument* and *ArchiveDocument* types).

TaskDataTypeEnum

Namespace: CMS.Synchronization

This enumeration determines the types of data which can be synchronized in the synchronization tasks:

- Simple – this option synchronizes only the main object.
- SimpleSnapshot – synchronizes the main object along with the translation information.
- Snapshot – synchronizes the main object with its child objects and bindings.

Translation information enables translation of foreign keys between Kentico and external systems. When using synchronous processing, the required data can be obtained directly within the context of the application.

Possible values and the data which they can synchronize:

Value	Object data	Page data	Translation data (only for asynchronous processing)	Include child objects, bindings, categories, etc.
Simple	✓	✓	✗	✗
SimpleSnapshot	✓	✓	✓	✗
Snapshot	✓	✗	✓	✓

Note: Simple and SimpleSnapshot options do not support synchronous processing.

TaskProcessTypeEnum

Namespace: CMS.Synchronization

Values of this enumeration represent all supported combinations of synchronization modes and data types. When the system searches for connectors with a subscription that matches an operation, and one connector has multiple subscriptions, the system selects the first subscription according to the following priority (highest to lowest):

- SyncSnapshot
- AsyncSnapshot
- AsyncSimpleSnapshot
- AsyncSimple

There is only one synchronous option – SyncSnapshot. This is given by the nature of synchronous processing, where you can always access related objects (parent, children, etc.) using the API.

IntegrationProcessResultEnum

Namespace: CMS.Synchronization

This enumeration stores possible results of outgoing tasks after being processed by external applications. The result determines what happens after the processing of each task. Possible values are:

Value	Meaning	Asynchronous processing result	Synchronous processing result
OK	Processing succeeded	The system deletes the task (or the relation between the task and connector). Processing continues with the next task in the queue.	Processing continues with the next task in the queue.

Error	Critical error occurred	The system logs the error into the synchronization log. Processing stops for the current connector (until the next reprocessing attempt). The task remains in the queue with the failed status. The system attempts to run the task again on subsequent processing requests – reprocessing occurs periodically during a 1 minute interval after the current processing ends, or later when new tasks are logged for the given connector.	The system logs the error into the event log. Task data is lost. Processing stops for all connectors.
ErrorAndSkip	Noncritical error occurred	The system logs the error into the synchronization log. Processing continues with the next task in the queue. The task remains in the queue with the failed status. The system attempts to run the task again on subsequent processing requests – reprocessing occurs periodically during a 1 minute interval after the current processing ends, or later when new tasks are logged for the given connector.	The system logs the error into the event log. Task data is lost. Processing stops for the current connector.
SkipNow	Process the task during next iteration	Processing continues with the next task in the queue. The task remains in the queue and the system attempts to run it again on subsequent processing requests (after new tasks are logged for the given connector).	Task data is lost.

IntegrationProcessTypeEnum

Namespace: CMS.Synchronization

Values of this enumeration are used during the processing of incoming tasks. The value determines whether to process the task immediately and how to proceed when an error occurs.

Value	Meaning
Default	Processes the task immediately. If an error occurs, the processing stops and the type is set to *Error*.
SkipOnce	Does not process the task during the first processing (just sets the type to *Default*, so it is processed during the next processing).
SkipOnError	Processes the task immediately. If an error occurs, the task is skipped and the processing continues.
DeleteOnError	Processes the task immediately. If an error occurs, the task is deleted and the processing continues.
Error	Processing does not continue after encountering a critical error.