This page provides a reference of the **ObjectTypeInfo** properties that you can use to <u>set the type information for custom module classes</u>.

The *ObjectTypeInfo* class has the following types of properties:

- <u>Constructor</u>
- <u>Field names (columns)</u>
- <u>General</u>
- <u>Class relationships and bindings</u>
- <u>Import/Export, Staging, Integration bus</u>
- <u>Continuous integration</u>
- <u>Macros</u>
- <u>Object versioning</u>

> The reference lists the **ObjectTypeInfo** members that are intended for public use. The class also contains other members, which are used internally or handled automatically by the system. We do not recommend working with any of the undocumented members.

Constructor

The following properties are always set through the constructor parameters when creating new *ObjectTypeInfo* instances:

Property	Type	Description
BinaryColumn	string	The name of the class field that stores binary data for objects. Can be *null*.
CodeNameColumn	string	The name of the class field that stores the unique text identifiers of objects. Can be *null* for classes without a dedicated code name column.
DisplayNameColumn	string	The name of the class field that stores the visible names of objects (names used in the administration interface or on the live site). Can be *null*.
GuidColumn	string	The name of the class field that stores the <u>GUID identifiers</u> of objects. Can be *null*.
IDColumn	string	The name of the class field that stores the IDs of objects (i.e. the primary key column). Can be *null* in rare cases (for example binding classes without an identity column).
ObjectClassName	string	**Required** for all classes. The code name assigned to the class in the Kentico administration interface. Identifies the class's definition in the database.
ObjectType	string	**Required** for all classes. Serves as the primary identifier string for the class. Use the object type name to select classes in the API, user interface components (<u>UniGrid</u>, <u>UniSelector</u>), <u>REST calls</u>, etc. The recommended place to define the object type name value is within the code of the *Info* class, in the *OBJECT_TYPE* constant.
ParentIDColumn	string	The name of the class field that stores the IDs of parent objects. Set to *null* for classes without a parent class. See <u>Setting parent-child relationships for classes</u> to learn more.

ParentObjectType	string	The *object type name* of the class's parent (as defined in the type information of the parent class). Set to *null* for classes without a parent class.
ProviderType	[Type](#)	**Required** for all classes. The type of the class's *InfoProvider* class. For example: *typeof(ForumPostInfoProvider)*
SiteIDColumn	string	The name of the class field that stores site IDs for site-related objects. Can be *null*. Only use site ID columns if you do not have a separate [binding class](#) for the site relationship.
TimeStampColumn	string	The name of the class field that stores the last modification date for objects. Can be *null*.

[> Back to list of ObjectTypeInfo property types](#)

Field names (columns)

In addition to the "column" properties used in the *ObjectTypeInfo* constructor, you can set the following optional properties to identify class fields that perform a specific function:

Property	Type	Description
AssemblyName Column	string	The name of the class field that stores the assembly name of a related class in the project's code. Intended for classes whose editing form contains the **Assembly and class selector** form control.
EnabledColumn	string	The name of the class field that indicates whether objects are enabled or disabled. The meaning of the enabled flag depends on the implementation of the class.
ExtensionColumn	string	Intended for classes that store binary data. Sets the name of the class field that stores the extension type of the binary data.
GroupIDColumn	string	The name of the class field that stores group IDs for objects related to specific [community groups](#).
MimeTypeColumn	string	Intended for classes that store binary data. Sets the name of the class field that stores the mime type of the binary data.
ObjectLevelColumn	string	Intended for classes that organize objects in a tree hierarchy. Sets the name of the class field that stores the level of objects in the hierarchy structure.
ObjectPathColumn	string	Intended for classes that organize objects in a tree hierarchy. Sets the name of the class field that stores the path of objects in the hierarchy structure. You can also set the path column separately for paths built out of objects names or object IDs through the **ObjectNamePathColumn** and **ObjectIDPathColumn** properties respectively.
OrderColumn	string	The name of the class field that stores the order of objects (for classes that have a defined order for objects). Allows automatic actions for changing object order on listing pages (i.e. in [UniGrid](#) components).
ResourceIDColumn	string	The name of the class field that stores references (IDs) pointing to a module in Kentico. Intended for classes whose objects have a relationship with a specific module.
SizeColumn	string	Intended for classes that store binary data. Sets the name of the class field that stores the size of the binary data.
VersionGUIDColumn	string	The name of the class field that stores the GUID identifiers for individual [versions of objects](#).

General

Property	Type	Description
AllowDataExport	bool	Indicates whether the system allows users to export the class's data into files of various formats from listing pages. *True* by default for all classes except binding classes. See also: Exporting data from the user interface
IsCategory	bool	Indicates whether the class represents a category for organizing other objects in a tree hierarchy (for example web part categories). The default value is *false*.
LogEvents	bool	Indicates whether the system logs events into the Event log when objects of the class are modified. *False* by default.
ModuleInfo	ModuleInfo	Read only property. Gets the *ModuleInfo* object based on the value of the **ModuleName** property. See also: Initializing modules to run custom code
ModuleName	string	This property is ***always required*** for all type information definitions. Set the value to the ***code name*** of the module under which the given class belongs.
ProviderObject	IInfoProvider	Read only property. Gets an instance of the provider class specified by the **ProviderType** property.
RememberUniGridState	bool	Indicates whether listing pages (UniGrid components) that display the class's objects remember the listing state for individual users (i.e. the filtering options, page number, page size and item order). Individual UniGrid components may override this property.
SupportsCloning	bool	Indicates whether the system allows cloning for the class's objects. *True* by default.
SupportsCloneToOtherSite	bool	Applies to classes with a specified **SiteIDColumn**. Indicates whether objects of the class can be cloned to a different site than the site of the original object. *True* by default.
TypeCondition	TypeCondition	Only affects classes with multiple type information definitions. Sets a condition that identifies which objects of the class belong to the given object type. See: Setting the type information for module classes
TouchCacheDependencies	bool	Indicates whether the system "touches" the corresponding dummy cache keys when an object of the class is modified. This causes the cache to delete all items that depend on the given dummy keys. Set to *true* if you wish to use objects of your custom class as cache dependencies. See also: Setting cache dependencies

Class relationships and bindings

Property	Type	Description

CheckDependencies OnDelete	bool	Applies to classes that are the target of a *Required* type reference from another class. See Adding references between classes for more information. If you set *CheckDependenciesOnDelete* to *true,* the system prevents objects of the class from being deleted if they are the target of a required reference for at least one other object in the system. *False* by default.
Depends On	List<Object Dependency>	Registers fields that store references from the class to other classes (i.e. foreign key fields). Allows the system to maintain referential integrity of relationships when importing or staging objects to other instances, or perform automatic removal of objects when a referenced object is deleted. Assign a *List* collection of **ObjectDependency** objects, each defining one reference. **Note**: Do not include the site or parent reference fields specified in the **SiteIDColumn** and **ParentIDColumn** properties. See Adding references between classes for more information.
Extends	List<ExtraColumn>	Registers external fields that store references from other classes in the system to the given class. Allows the system to maintain referential integrity of relationships when importing or staging objects to other instances, or perform automatic removal of objects when a referenced object is deleted. Assign a *List* collection of **ExtraColumn** objects, each defining one reference that targets the given class. See Adding references between classes for more information.
IsBinding	bool	Indicates whether the class represents a relationship between two or more classes. See Creating custom binding classes for more information.
IsSiteObject	bool	Read only property. Indicates that the class represents site-related objects. *True* if the *SiteIDColumn* or *GroupIDColumn* property is set in the type information, or if the class has a site-related parent class.
NameGlobally Unique	bool	Applies to classes with a specified **SiteIDColumn** that also have the **SupportsGlobalObjects** property set to *true.* If true, the system validates the code names of the class's objects to be unique across both global and site-related objects. If false, global objects can have the same code name as a site-specific object. The default value is *false.*
ParentTypeInfo	Object TypeInfo	Read only property. Returns the *ObjectTypeInfo* instance representing the parent object type.
RegisterAsChildTo ObjectTypes	List<string>	Overrides the default automatic parent-child registration process. Enter a List of object type names that you wish to register as parent types. Only intended for special cases where you need to register multiple parent types (if the parent class has multiple type information definitions).
SiteBinding	string	Read only property. Returns the object type name of the bindings class that stores relationships between the given class and sites (if such a site binding class exists).
SiteBindingObject	BaseInfo	Read only property. Returns the *Info* class (as *BaseInfo)* of the binding class that stores relationships between the given class and sites (if such a site binding class exists).

Suppport sGlobalO bjects	bool	Applies to classes with a specified **SiteIDColumn**. If true, the system allows both site-related objects and global objects whose value in the site ID column is *null.* Must be true if you wish to use export or staging for global objects of the given class.

> Back to list of ObjectTypeInfo property types

Import/Export, Staging, Integration bus

Property	Type	Description
ImportExp ortSettings	ImportExp ortSettings	Defines how the class works with the Export and Import features. Allows you to transfer objects of the given class to other Kentico instances that contain the same custom module. Set through the properties of the *ImportExportSettings* class. See Enabling export and staging for the data of classes for details.
LogIntegra tion	bool	Determines whether the system logs integration bus synchronization tasks for objects of the class. *True* by default.
Synchroniz ationSettin gs	Synchroniz ationSettin gs	Defines how Staging features work with the class's data. Allows you to connect two or more Kentico instances that contain the same custom module, and synchronize changes of the given class's data. Set through the properties of the *SynchronizationSettings* class. See Enabling export and staging for the data of classes for details.

> Back to list of ObjectTypeInfo property types

Continuous integration

Property	Type	Description
ContinuousInte grationSettings	ContinuousInte grationSettings	Determines how the class works with the system's continuous integration solution. Set through the properties of the *ContinuousIntegrationSettings* class. See Enabling continuous integration for custom classes for details.
SerializationSet tings	SerializationSet tings	Determines how the system serializes the class's objects from the database into XML data. The serialization is used by the system's continuous integration solution. Set through the properties of the *SerializationSettings* class. See Enabling continuous integration for custom classes for details.

> Back to list of ObjectTypeInfo property types

Macros

Property	Type	Description

CheckPermissions	bool	Determines whether the system checks read permissions for the given module when accessing objects of the class in macro expressions. The permissions are checked for the user who saved the macro, not the user viewing the result. True by default. See also: Working with macro signatures
ContainsMacros	bool	Indicates whether the class has fields that contain macro expressions in their values. The property determines if objects of the class are processed when resigning macros in the system. True by default. You can set the value to *False* to optimize the performance of the resigning process (during manual resigning and upgrades). Does not affect the resolving of macros.
MacroCollectionName	string	Overrides the name of object collections used for the class's data in macros. If not set, a default name is automatically derived from the class code name.
SensitiveColumns	List<string>	Specifies class fields that the system excludes from the macro engine and data retrieved by the REST service. Allows you to protect fields with sensitive values, for example passwords. Set the sensitive fields by assigning a *List* collection of field names.

> Back to list of ObjectTypeInfo property types

Object versioning

Property	Type	Description
AllowRestore	bool	Determines whether the system stores deleted objects of the class in the recycle bin. The default value is *true,* except for binding classes (i.e. class that have the **IsBinding** property enabled in their type information).
IncludeToVersionParentDataSet	bool	Applies to classes that have a parent class. Determines whether the version data of parent objects includes child objects of the given class. The default value is *true.*
SupportsLocking	bool	Determines whether the class's objects support object locking (check out and check in). The default value is *false.*
SupportsVersioning	bool	Determines whether the class's objects support object versioning. The default value is *false.* **Note**: To allow users to view and manage object versions, the editing interface of the class must provide tabs (a UI element using the *Vertical tabs* page template). The system adds the *Versions* tab automatically once you enable versioning support for the class.

> Back to list of ObjectTypeInfo property types