Search documentation Search

Managing Custom Data Types

Appian 7.7

Toggle Us

The **Data Types** tab of the Data Management page of the System Administration console allows you to visually design, edit, import, and manage custom data types. A custom data type is a metadata format for information you want to use in process applications. They are stored in the primary data source, when available.

For information on casting one CDT to another, see also: Casting Custom Data

Tu

Data Type Creation

Custom data types can be created in any of the following ways:

- Import an XML Schema Definition Language (.XSD) file.
  - See also: Importing an XSD File and http://www.w3.org/TR/xmlschema11-1/
- Import an XSD defined using a Java object, as part of an Appian Plug-in.
  - See also: Creating a Complex Data Type Using a Java Object
- Build the data type visually, using the Data Type Designer.
  - See also: Data Type Designer
- Import CDTs as part of an application package.
  - See also: Importing an Application
- Import CDTs from a web service.
  - See also: Using the Call Web Service Smart Service

Re

Important Considerations for Using Custom Data Types

- Avoid testing XSD file import on production systems. Every data type contained in an XSD that you import becomes immediately available to a
  process developers (unless you hide the data types).
- Hidden data types appear on the management view, but not on the list of data types available when creating process variables, node inputs, and
  other objects such as data store entities.
- When interacting with web services or databases, work to limit the disk-space consumed by your data. Place the data-collection node in a sub-
  process, filter out any unnecessary data, then delete the sub-process when it is completed.
- When using the `length()` expression function on the field of a process variable that is a complex custom type, the expression results in an erro
  the process variable is null at the time of evaluation. As a best practice, check whether the process variable is not null before attempting to get
  length of one of its fields.
  - See also: length()
- Variables of simple custom types that extend other simple custom types can be saved into each other because they both extend from a simp
  system type (like Text).
  - Variables of complex custom types that extend other complex custom types (at times) may not be successfully used to save data due to the mismatch of fields.
  - At runtime, any fields of a derived type that are not present in the parent type are not mapped.
  - If a parent data type is an abstract complex type with no fields, no data can be saved into it. It is best to avoid abstract types and extending complex types.
- The typename of a custom type is the name as defined in the XSD that was used to create it. When a variable is marked as a multiple, **?list** is appended to the typename to indicate that it holds multiple values. This is different than system data types which have a typename of the form **List of** when they are multiple. Be aware of this when using the `typename()` expression function with custom data types.
  - See also: typename()
- When using multiple application servers and unhiding data types, the data type visibility is only updated in the cache of the application server you
  are logged into. The cache of any other application server is not updated automatically. The change is only applied to the other application serve
  upon the next restart. Work around this by logging into each application server individually after updating the visibility of a custom data type. Th
  updates the show/hide status for the data type on each application server, and avoids having to restart your application servers.

Ot

See Also: Referencing System Data Types versus Custom Data Types

Updating a Custom Data Type

To change a CDT:

- Edit the data type using the Data Type Designer - OR - Delete the existing CDT and import a new version of the CDT using the same name and namespace.
  - See also: Editing Custom Data Types

When a custom data type is referenced throughout your process models, manually updating each reference may involve a time-consuming process. You can automatically update the CDT references in your application in the following manner.

1. Prepare an application for export with all objects that reference the updated CDT.
2. Select the CDT on the Data Management page.
3. Click the **Impact Analysis** button. The Impact Analysis tool displays a report of all application objects that reference an outdated version of the type, and lets you update these by clicking **Update**.
   - **NOTE**: When you update the process models that reference the CDT, a new version is published. Be sure that any draft process model you are currently working on is published before you update it. Otherwise, the most recent changes to your process model will be lost.
   - It is possible to make an incompatible change to a CDT, which is not automatically resolved. Be sure to verify that the new CDT structure i compatible where it is used in your forms, variables, and other references.
   - See also: Data Type Impact Analysis

NOTE: Data store entities must be republished to use a new CDT. If your application server is restarted, and you begin receiving errors when query rul and the Write to Data Store Entities Smart Service nodes execute, it may be because a database schema was modified and its data store entity was no re-verified and republished. You can resolve the errors by republishing any data stores that are mapped to the modified CDT.

- See also: Managing Data Stores and Database Schema Best Practices

Viewing Updated Data Types

- Closing and reopening the Process Modeler allows you to view updated data types in your process models.

Deleting a Custom Data Type

Deleting a data type may have a significant impact on your applications. Rather than disrupt an active process when the data types used by your process variables and other objects (such as data store entities) are deleted, the data type is retained in the system by creating what is essentially a deprecated data type. The application objects that currently use that deprecated type are automatically updated to reference the deprecated type. W call these deprecated types deleted data types, as you can no longer use the type when creating a new object.

In order to delete a custom data type, navigate to the Data Management page, select the checkbox next to the data type, and click **Delete**.

When a custom data type is deleted, the CDT display name changes to append `^1`, `^2`, or `^*a_number*` to the end of the type name. The appended number indicates the version number held by a CDT with the original name (and namespace) when it was deleted from the system.

- Objects that use (reference) a CDT that you delete are automatically updated to reference the deleted data type. They are *not* disabled.
- When a deleted CDT incorporates other CDTs as sub-elements, the referenced CDTs are not altered.

If you create a CDT that incorporates a deleted CDT (such as `dataType^2`) as a sub-element, the deleted CDT is re-created as a CDT that is *not* delete This might happen when importing an application. In such cases, the reference to the deleted CDT is not automatically updated to use the recreated CDT. Instead, the reference to the deleted CDT continues to be used. The only exception to this is the Source Data Type field of service-backed record which automatically update to use the latest recreated CDT when a CDT is reinstated.

NOTE: You **can not** export a deleted CDT.

Hiding a Custom Data Type

Hidden data types do not appear in the Data Type dialog box that displays when you are creating a new node input, node output, process variable, or data store entity.

There are two ways to hide a data type from the data type list:

- Import the data type using the Call Web Service Smart Service. Any data type imported as such defaults to hidden.
  - See also: Call Web Service Smart Service
- Hide the data type manually by selecting its checkbox next to the data type and click **Hide**.

Unhiding (or exposing) a custom data type only exposes it for the application server that you are logged into. Any other application server(s) in the instance update the custom data type visibility when restarted or after the user who exposed the data type logs into Appian through the other application server(s).

Exporting a Custom Data Type

The complex data types you create, or import, can be downloaded in XSD format.

- When designing or viewing a complex data type on the Managing Data page of the System Administration console, the **Download XSD** button displayed.
- If the data type is saved and published, click **Download XSD** to export the data type definition as an XSD file. The **File Downoad** dialog box is displayed.
- Select the download location for the XSD file.