Tamino Server Extensions

Tamino allows you to develop, implement, administrate and execute Server Extensions. Tamino Server Extensions can be used to extend the query and mapping Tamino Server functionality by adding user-defined logic. For a description of the functionality available with Tamino Server Extensions, see the Tamino documentation.

In order to extend the Tamino functionality, you install Tamino Server Extension Packages in Tamino databases. These packages contain (among other data) Tamino Server Extension Objects based on COM or on Java. Methods of these objects can then be used to extend the query or mapping functionality of the Tamino Server.

Server Extension Objects based on COM can be implemented in Natural. Please check the Natural Release Notes for information about the Tamino version needed as a prerequisite.

The *Tamino Server Extensions* documentation covers the following topics:

- Overview
- Developing a Tamino Server Extension
- Using Callbacks
- Deploying a Tamino Server Extension
- Installing a Tamino Server Extension
- Tamino Server Extension Example

Overview

This document focuses on the Natural-specific implementation details of Tamino Server Extensions and the Natural tools and techniques used in this process. Essential background information that should help you develop valid Server Extension Function code is contained in the Tamino documentation. You should read the corresponding chapters of the Tamino documentation carefully before starting to develop Tamino Server Extensions.

- To develop Natural-based Server Extensions, you use the Natural Class Builder. A Tamino Server Extension is developed as a NaturalX class that implements interfaces corresponding to a predefined structure. To support the implementation of these interfaces, certain predefined Natural modules are delivered with Natural.
- To deploy a Natural-based Tamino Server Extension in the target environment, you use the usual Natural deployment tools.
- To register your Tamino Server Extension, you use the Natural REGISTER command.

Once you have developed, installed and registered your Natural-based Tamino Server Extension using Natural development tools, you can assign it to a Tamino database and use it in a Tamino schema using the usual Tamino tools. For a detailed description of the usage of these tools, see the Tamino documentation.

- To select methods of your Natural-based Tamino Server Extension into a Server Extension Package and to create a package file, you use the SXS Analyzer.
- To install and administrate Server Extensions in a Tamino database, use the Server Extensions Administration as provided by the Tamino Manager.
- To assign Server Extension functions to a Tamino schema, use the Tamino Schema Editor.
- Server Extensions can be traced using the SXS Trace.

Developing a Tamino Server Extension

The following topics are covered.

- Overview
- Set the Library SYSEXSXS as Steplib
- Create a New Library for Your Project
- Create a NaturalX Class
- Create the Object Data Area
- Edit the Object Data Area
- Link the Connection Interface
- Implement the Method Connect
- Add Server Extension Functions
- Save and Catalog the Class

Overview

The Natural Class Builder is used to develop a Tamino Server Extension. A Natural-based Tamino Server Extension is a NaturalX class that implements interfaces corresponding to a predefined structure. To support the implementation of these interfaces, certain predefined Natural modules are delivered with Natural:

- An Interface Module (Copycode) containing the declaration of the Connection interface defined by Tamino.
- Parameter Data Areas containing parameter definitions for the different types of Server Extension functions.

Set the Library SYSEXSXS as Steplib

When implementing a Tamino Server Extension in Natural, you can use a number of predefined Natural modules contained in the sample library SYSEXSXS. This makes sure that your Tamino Server Extension conforms to the interface defined by Tamino. In order to use these modules in your Tamino Server Extension project, first define the library SYSEXSXS as steplib.

To define the library SYSEXSXS as steplib:

1. Start the Natural Configuration Utility.
2. Select the Natural parameter file you are working with.
3. Choose **Edit > Find** to locate the parameter STEPLIB.
4. Enter SYSEXSXS into the list of steplibs.
5. Save the Natural parameter file.
6. Close the Natural Configuration Utility.
7. Restart Natural Studio.

Create a New Library for Your Project

It is recommended to put all Natural modules for one Tamino Server Extension into one Natural library. Therefore first create a new Natural library for your project.

To create a new library:

1. Select **User Libraries** in the library workspace.
2. Select **New** in the context menu.
3. Choose a name for the library.

Create a NaturalX Class

A Tamino Server Extension is implemented as a NaturalX Class. Therefore create a new class.

To create a new class:

1. Select your library in the library workspace.
2. Select New Source > Class in the context menu.
3. Choose a name for the class.
4. Select Save in the context menu.
5. Choose a name for the class module.

Create the Object Data Area

An Object Data Area in a NaturalX class is used to contain variables that shall keep their value during the lifetime of an instance of the Tamino Server Extension.

To create an Object Data Area and link it to your class:

1. Select your class in the library workspace.
2. Select New > Object Data Area in the context menu.
3. Choose a name for the Object Data Area.

Edit the Object Data Area

The Object Data Area of a Tamino Server Extension should at least contain an object handle to hold a reference to the Tamino callback object during the lifetime of an instance of your Tamino Server Extension. The Tamino callback object is passed by Tamino to the Server Extension when it loads the Extension. You can use the methods of the callback object to call Tamino functionality from inside your Tamino Server Extension functions.

To add an object handle for the callback object to your Object Data Area:

1. Double-click on the Object Data Area in the library workspace to edit it.
2. In the Data Area Editor select **Insert > Handle**.
3. Choose a name for the object handle (e. g. CALLBACK).
4. Select "Object" as handle type.
5. Click the **Add** button.
6. Click the **Quit** button.
7. If you so wish, add further variables to the Object Data Area as required by your Server Extension.
8. Close the Data Area Editor and save the Object Data Area.

Link the Connection Interface

A Tamino Server Extension must implement the Connection interface ISXSConn. This interface is declared in the interface module ICONN-C in the library SYSEXSXS. To be able to locate the interface module, you must have defined the library SYSEXSXS as steplib. Link this interface module to your class.

To link the interface module to your class:

1. Select your class in the library workspace.
2. Select **Link > Interface Module** in the context menu.
3. Select the interface module ICONN-C in library SYSEXSXS.

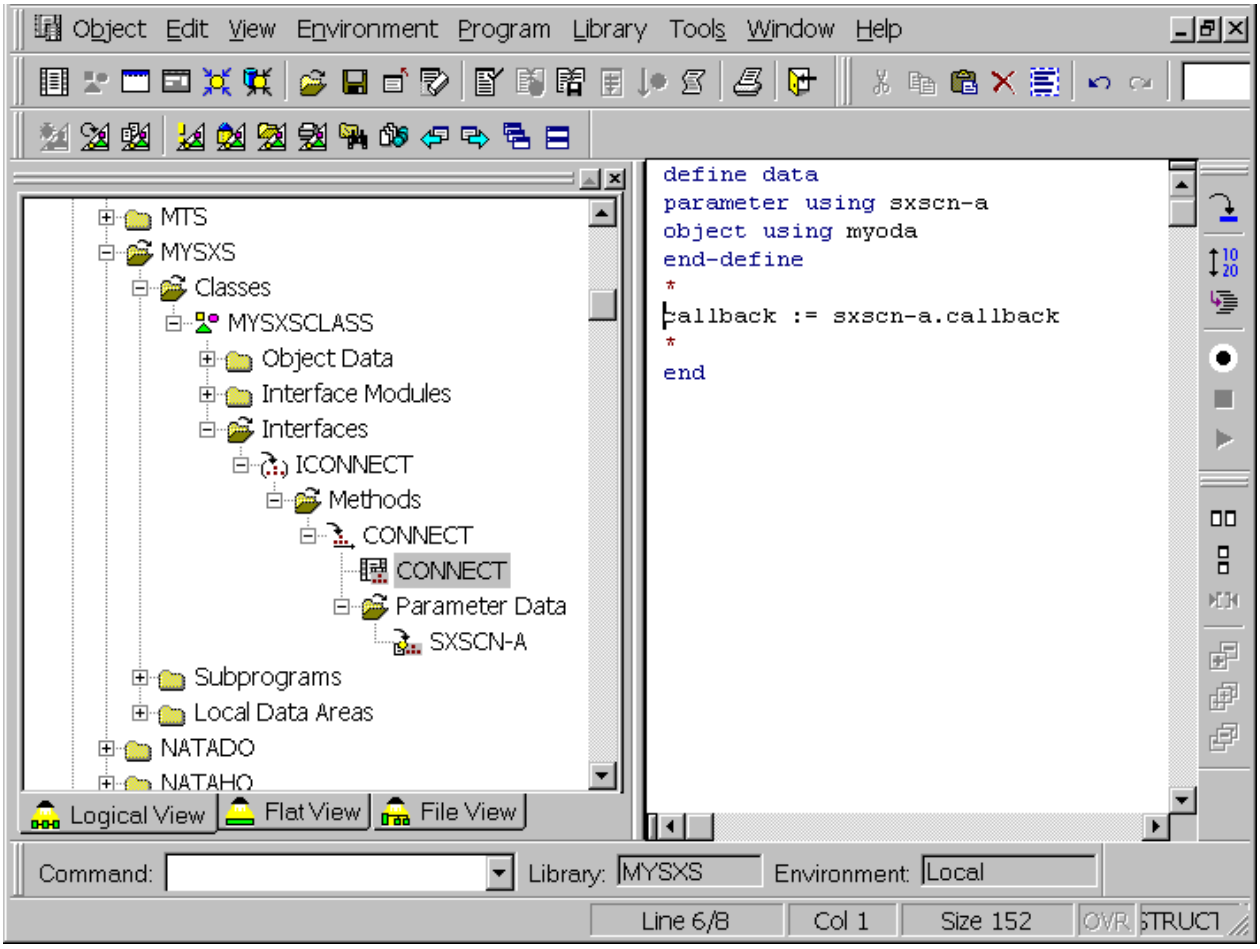
Implement the Method Connect

The method Connect of the ISXSConn should store a handle to the callback object in the Object Data Area. This allows the Server Extension Functions to access Tamino functionality.

To implement the method Connect:

1. Fully expand the branch "Interfaces" of your class in the library workspace.
2. Select the subprogram node "CONNECT". This is the default name for the subprogram that will implement the method Connect.
3. If you so wish, rename the subprogram to a name of your choice by selecting Rename in the context menu.
4. Double-click on the subprogram node to create and edit the method subprogram.

The implementation of the Connect method must include both the Object Data Area of the class and the Parameter Data Area of the method Connect. The body of the method must assign the Callback object handle from the Parameter Data Area to the corresponding object handle defined in the Object Data Area, as shown in the example below.



You can also add further initialization code to the method Connect as required by your Server Extension.

Add Server Extension Functions

You can now start to add your own Server Extension Functions to the class. First you will create a new interface for your class to contain the Server Extension Functions. Then you will add the individual functions to that interface.

To create a new interface:

1. Select the node "Interfaces" of your class in the library workspace.
2. Select New in the context menu.
3. Choose a name for the interface.

To create a new Server Extension Function :

1. Select your interface in the library workspace.
2. Select **New > Method** in the context menu.
3. Choose a name for the method.
4. Select the method
5. Select Link > Parameter Data Area.

Tamino distinguishes different types of Server Extension Functions. Depending on the type of Server Extension Function to be implemented, choose the corresponding Parameter Data Area:

Function	Data
Map In function:	Parameter Data Area SXSMI-A
Map Out function:	Parameter Data Area SXSMO-A
On Delete function:	Parameter Data Area SXSDL-A
Event function:	Parameter Data Area SXSEV-A
Query function:	Parameter Data Area SXSQU-A

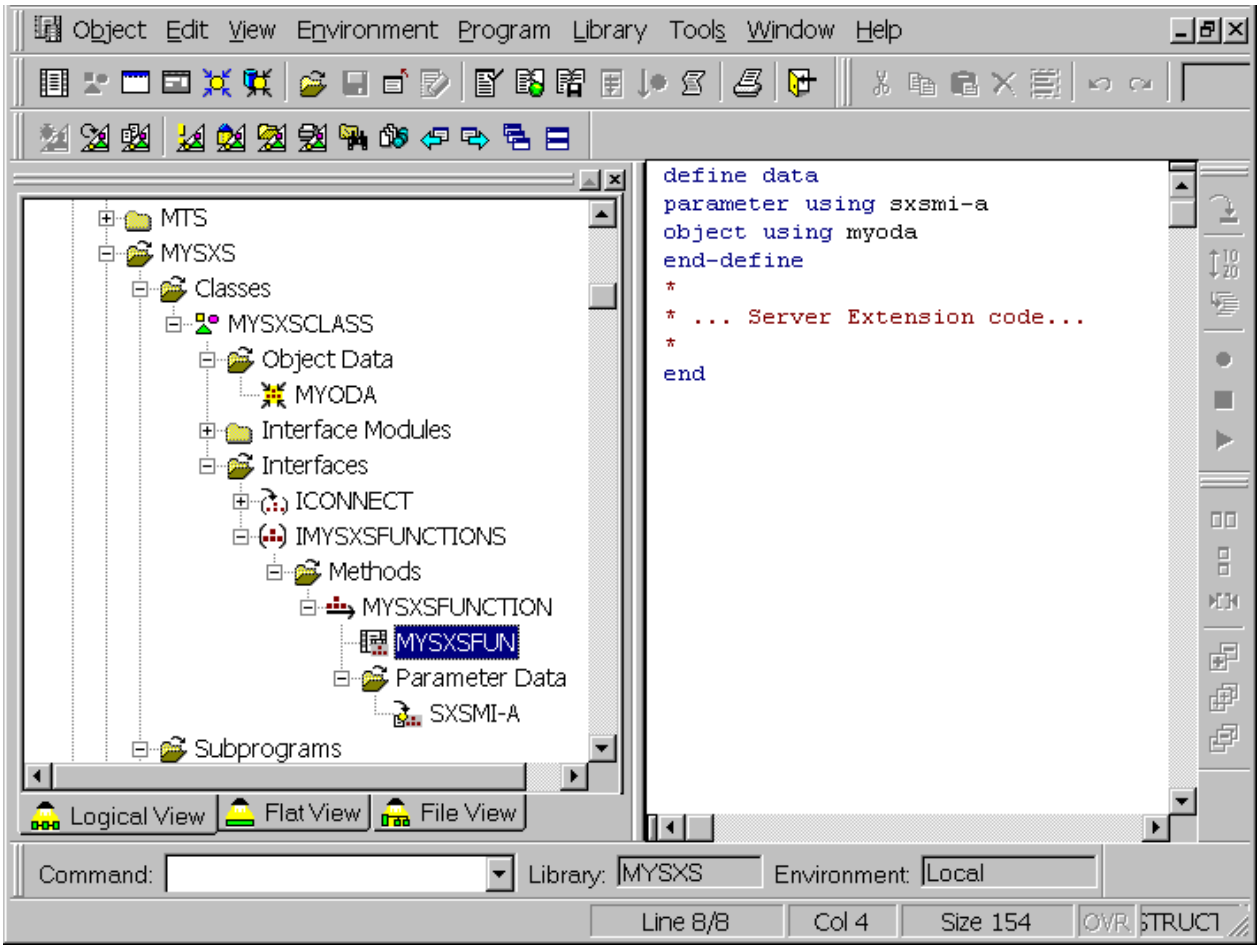
These Parameter Data Areas are contained in the library SYSEXSXS. To be able to locate the Parameter Data Areas, you must have defined the library SYSEXSXS as steplib.

Note:
The Parameter Data Area SXSQU-A is just an example. Query functions can have user-defined parameters. Thus it is not possible to define a common Parameter Data Area for them. If you want to create a query function, please consult the Tamino documentation and check which parameter types are allowed in query functions. Then create your own Parameter Data Area in your project library that matches the needs of your query function.

To implement the Server Extension Function:

1. Select the subprogram node that represents the method implementation.
2. If you so wish, rename the subprogram to a name of your choice by selecting Rename in the context menu.
3. Double-click on the subprogram node to create and edit the method subprogram.

The implementation of the method must include both, the Object Data Area of the class and the Parameter Data Area assigned to the method. The body of the method contains the coding specific to the Server Extension Function.



- Close the Program Editor and save the subprogram.

To add further Server Extension functions, repeat "To create a new Server Extension Function".

Save and Catalog the Class

Finally save the class and recatalog the whole project library.

1. Select your class in the library workspace.
2. Select **Save** in the context menu.

3. Select your library in the library workspace
4. Select **Cat All** in the context menu.

Using Callbacks

Tamino callbacks are interfaces of the Tamino Server that can be used in a Server Extension Function. They enable access to both the various databases that can be administrated by the Tamino Server and system information available in the running Tamino Server. To use a Callback function from within a Natural-based Tamino Server Extension, do the following:

- Consult the Tamino documentation to find out the parameters of the callback function you wish to use.
- In the Object Data Area of the NaturalX class that implements your Tamino Server Extension you have defined an object handle as a reference to the callback object. Send a corresponding method call to this object handle.

Deploying a Tamino Server Extension

Having developed the NaturalX class that implements your Tamino Server Extension, deploy it into the target environment with the usual Natural deployment tools, for instance the Object Handler. A Tamino Server Extension must be installed on the same machine where the Tamino server is running.

Register the class in the target environment under an arbitrary COMSERVERID. If necessary, see the NaturalX documentation on COMSERVERIDs and the different options of the REGISTER command.

Installing a Tamino Server Extension

Installing a Natural-based Tamino Server Extension is the same procedure as installing any COM-based Tamino Server Extension:

1. Use the SXS Analyzer to create a Server Extension Package. In the SXS Analyzer, select as the file to analyze the type library of your NaturalX class. As with any NaturalX class, the type library is located in the directory *<natdir>\<natvers>\Natural\Etc\<comserverid>\<classname>\<version>*, where *<natdir>* is the Natural installation directory, *<natvers>* the installed Natural version, *<comserverid>* the COMSERVERID under which the class was registered, *<classname>* the class name and *<version>* the class version (currently always "v1"). Proceed as usual with the SXS Analyzer to create a Server Extension Package.
2. Install the Server Extension Package into a Tamino database using the Server Extensions Administration as provided in the Tamino Manager.
3. Afterwards you can use the Tamino Server Extension as usual, for instance in XQuery functions or to map XML sub-documents in the Tamino Schema Editor.

Tamino Server Extension Example

The Natural sample library SYSEXSXS contains a simple Natural-based Tamino Server Extension as a programming example. The sample works on the Employees schema and maps parts of the schema, the salary data, on Server Extension Functions. These functions work as follows:

- Whenever an Employee element is inserted into the schema, the salary data of this Employee is passed to the Map In function SXSStoreToFile. This function creates a file in the TEMP directory and stores the data into the file.
- Whenever an Employee element is read from the schema, the salary data of this Employee is requested from the Map Out function SXSRetrieveFromFile. This function opens the corresponding file in the TEMP directory and reads the data from the file.
- Whenever an Employee element is deleted from the schema, the On Delete function SXSDeleteFile is called. This function deletes the corresponding file in the TEMP directory.

The sample Employees schema and some sample data are contained in the RES subdirectory of the sample library SYSEXSXS.

The sample can be driven comfortably using the Tamino Demo application contained in the sample library SYSEXINS. To run the sample Tamino Server Extension, proceed as follows:

1. Install the sample Tamino Server Extension in a Tamino database as described above in "Deploying a Tamino Server Extension" and "Installing a Tamino Server Extension".
2. Start the dialog MENU in the library SYSEXINS.
3. Set the Tamino URL to your Tamino database.
4. Enter "NATSXSDemoData" as collection name.
5. Execute "Define Tamino Schema" and select the schema "EmployeeSXSSchema.tsd" from the RES subdirectory of the sample library SYSEXSXS.
6. Execute "Load Tamino Data" and select the data file "EmployeeSXSData.xml" from the RES subdirectory of the sample library SYSEXSXS.
7. Execute the sample queries and update records. Note that part of the Employee data (the salary data) is now handled by the Tamino Server Extension.
8. Delete the Employee data.
9. Delete the Employee schema.