ScriptSync User Guide

Table of Contents

Front Matter

What is ScriptSync?

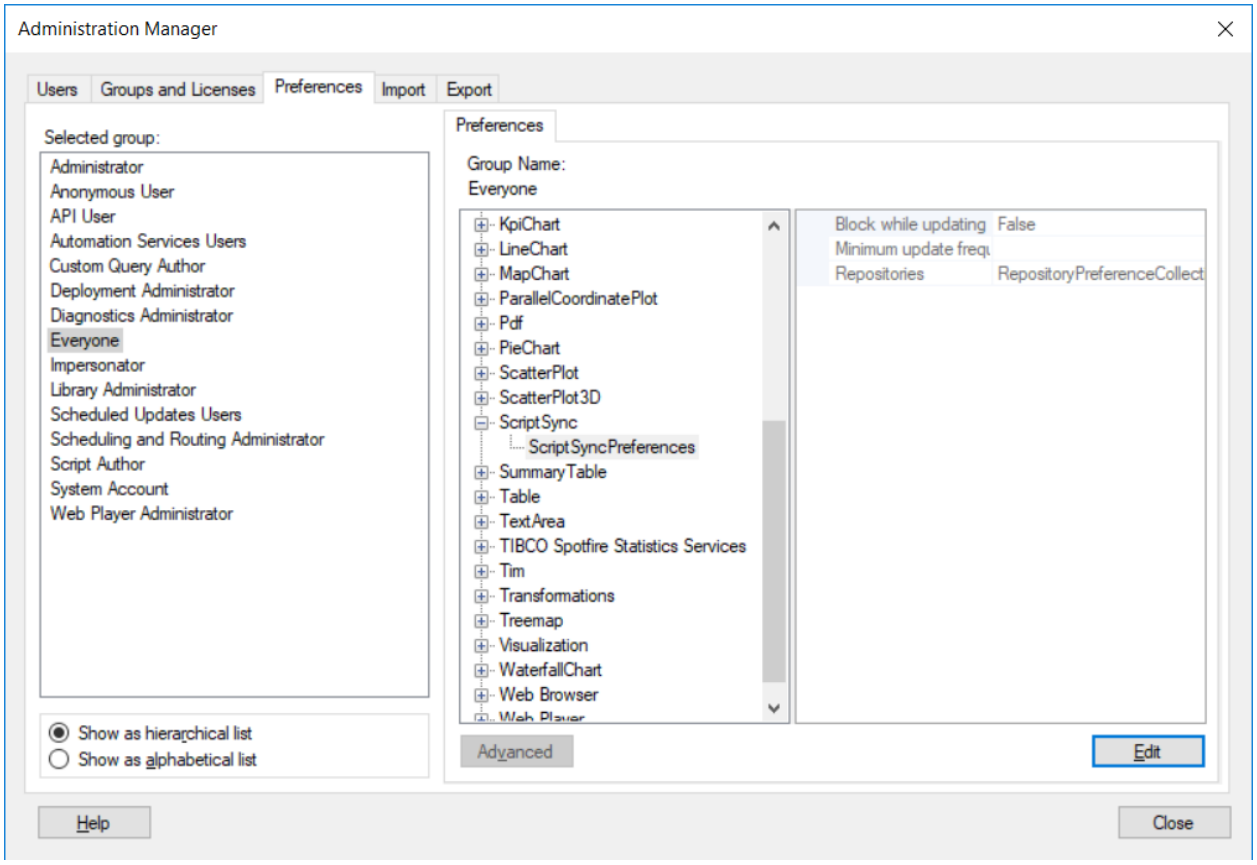
ScriptSync helps administrators and developers manage IronPython scripts, Data Function definitions, and python libraries used in Spotfire. IronPython scripts are usually embedded in documents. Updating scripts requires opening a document, updating a script manually, and saving a new copy of the document to a file or library. ScriptSync automates this process by moving scripts outside of documents, and integrating with Git, a common program used for version control. ScriptSync automatically downloads files under Git version control and makes them available to Spotfire in predetermined locations. Files may then be managed through Git with the assurance that users will automatically receive updated copies when they become available without having to modify Spotfire documents. ScriptSync works with both the Spotfire professional client and the WebPlayer.

Installation

ScriptSync is installed by adding the LeadDiscoveryChemCharts.sdn Spotfire distribution file to a Deployment area on the Spotfire Server. See ‘Deploying Spotfire Packages’ in the Spotfire Deployment and Administration manual for more details on deploying distribution files.

Configuration

The Git repositories that ScriptSync clones and updates locally must be configured through the Spotfire Administration Manager. The Administration Manager can be found in the Spotfire Tools menu.



Selected group:

Administrator
Anonymous User
API User
Automation Services Users
Custom Query Author
Deployment Administrator
Diagnostics Administrator
Everyone
Impersonator
Library Administrator
Scheduled Updates Users
Scheduling and Routing Administrator
Script Author
System Account
Web Player Administrator

(●) Show as hierarchical list
( ) Show as alphabetical list

Preferences

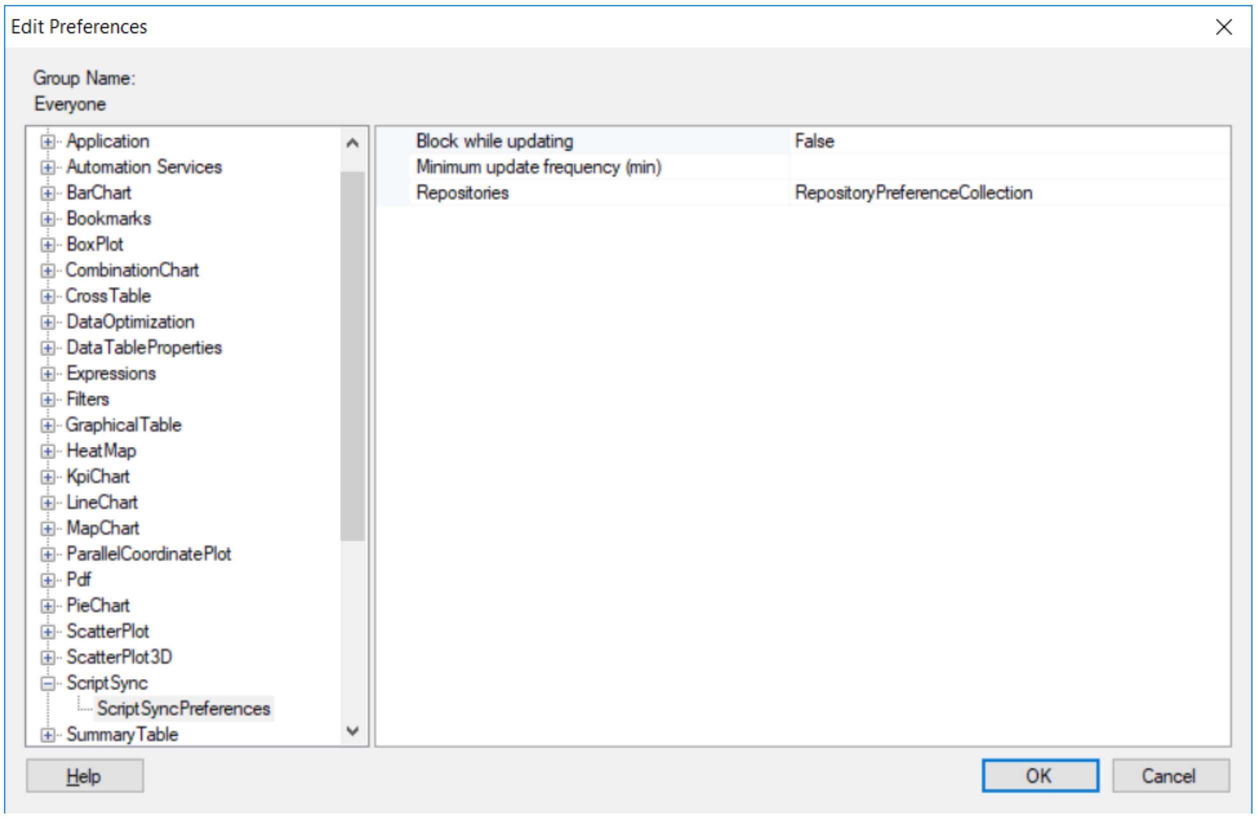
Group Name:
Everyone

KpiChart
LineChart
MapChart
ParallelCoordinatePlot
Pdf
PieChart
ScatterPlot
ScatterPlot3D
ScriptSync
    ScriptSyncPreferences
SummaryTable
Table
TextArea
TIBCO Spotfire Statistics Services
Tim
Transformations
Treemap
Visualization
WaterfallChart
Web Browser
Web Player

Block while updating	False
Minimum update frequ	
Repositories	RepositoryPreferenceCollect

The Preferences tab in the Administration Manager

Navigate to the Preference tab of the Administration Manager. Select the group of users for whom ScriptSync will be configured to retrieve Git repositories. Expand the preferences for ScriptSync and click on ScriptSyncPreferences, then click the Edit button in the lower right hand side of the dialog.


Edit Preferences

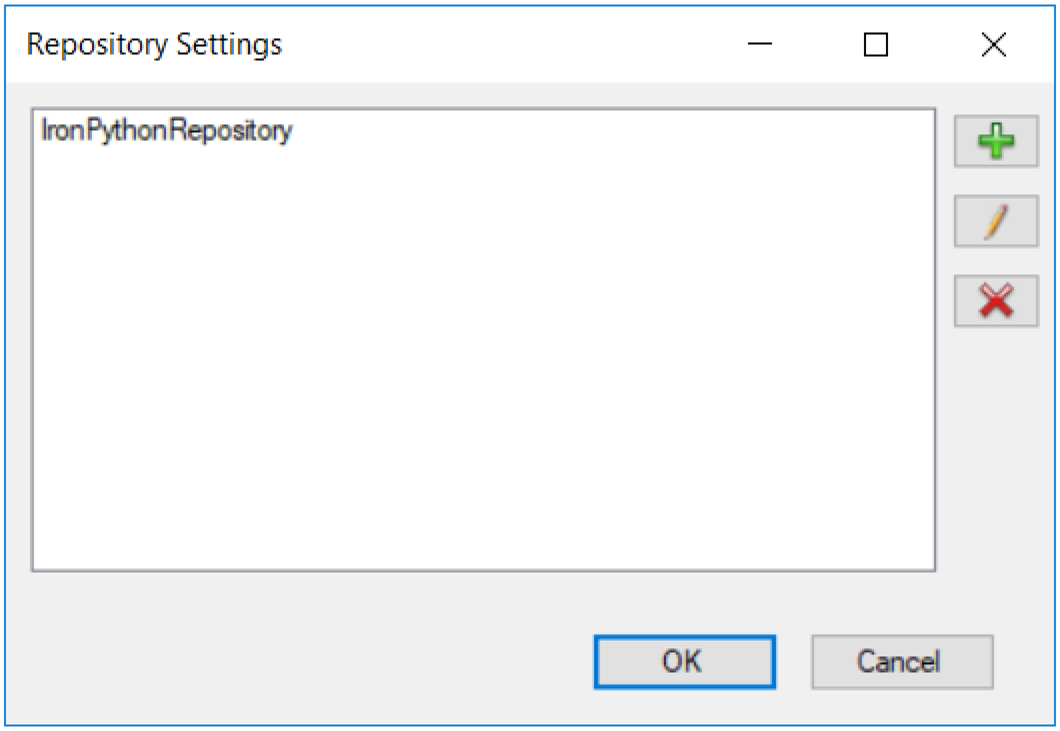
Group Name:
Everyone

Block while updating	False
Minimum update frequency (min)	
Repositories	RepositoryPreferenceCollection

Editing the ScriptSync Preferences

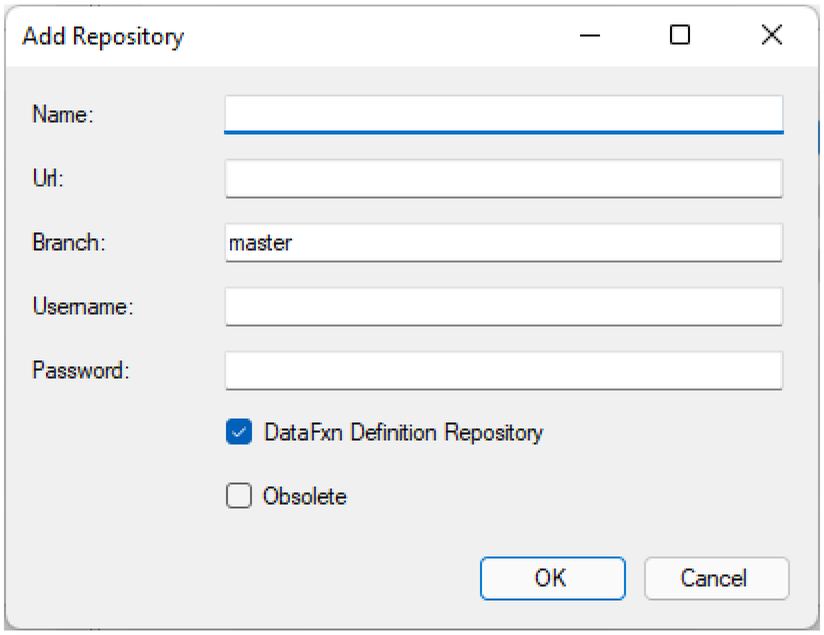
The *Block while updating* preference option controls the behavior during Spotfire startup while ScriptSync is cloning and updating local repositories. If set to False, ScriptSync will clone and update repositories in the background while Spotfire is starting. If set to True, Spotfire will pause until repositories are cloned or updated. Cloning a repository is usually completed quickly enough that repositories will be available by the time a document is opened. With very large repositories, it may be beneficial to pause Spotfire's startup to ensure the repository is available for the first script execution. For most repositories, updates will succeed before Spotfire is available for user interaction and script execution.

Clicking on the *RepositoryPreferenceCollection* preference will open a dialog for configuring connections to repositories to be cloned locally.



The Repository Settings dialog with one configured repository

Repository settings may be added , edited, and deleted in the Repository Settings dialog. Click the add button ➕ to add a new repository, and fill out the appropriate fields.



Name:	
Url:	
Branch:	master
Username:	
Password:	
	[✓] DataFxn Definition Repository
	[ ] Obsolete

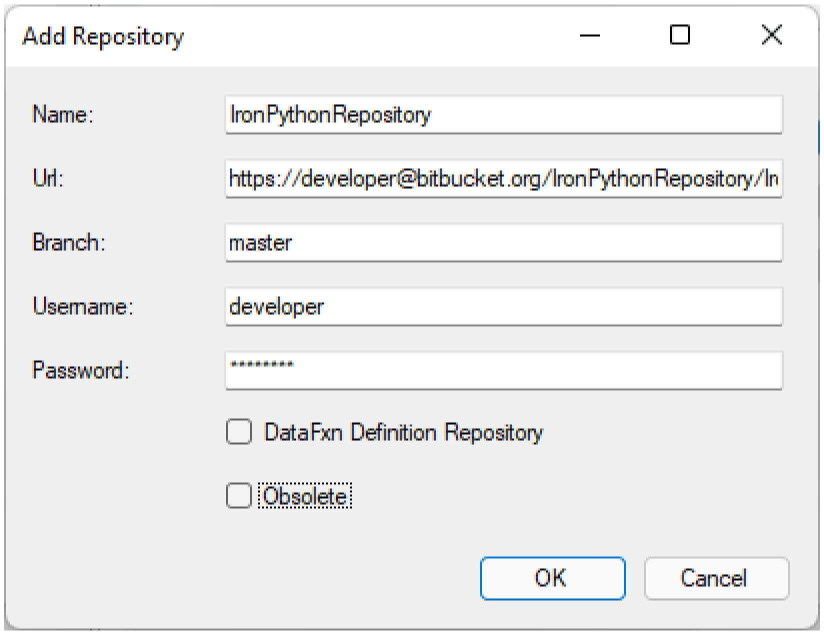
The Add Repository dialog

The *Name* field of the repository is used as the directory name to which the repository will be cloned and as the module name referenced by scripts embedded in Spotfire. The *Url* parameter specifies the repository to be cloned locally to a directory supplied in the *Name* parameter. The Branch parameter is optional. If left empty, the default branch will be cloned. Repositories will be cloned to the branch defined in the Branch parameter, if it is specified. The *Username* and *Password* fields are optional as they aren't required for public repositories. For private repositories, a username and password must be provided. Note that ScriptSync will only clone repositories via https, not via ssh. Cloning via ssh is not supported as it would require private ssh keys in the repository configuration, and this may be a security liability.

The *DataFxn Definition Repository* checkbox should only be checked when configuring repositories containing Data Function definitions. Data Function definitions are downloaded to a different path than python modules.

The *Obsolete* checkbox may be used to remove disused repositories. When the *Obsolete* checkbox is checked, the corresponding user directory will be removed, if present.

Repository configurations may also be edited or deleted by selecting a repository from the list in the Repository Settings dialog, then clicking the edit ✎ button or delete ✖ button, respectively.


Add Repository

Name:	IronPythonRepository
Url:	https://developer@bitbucket.org/IronPythonRepository/In
Branch:	master
Username:	developer
Password:	********

☐ DataFxn Definition Repository

☐ Obsolete

The Edit Repository dialog

ScriptSync updates repositories every time a Spotfire application is opened. In the WebPlayer, an application instance is created every time a user opens a document. High traffic WebPlayer servers don't need to update repositories each time a document is opened. The *Minimum* *update frequency* preference prevents frequent and unnecessary updates. The default Minimum update frequency is one hour. Use the preference to alter the update frequency.

After configuring repositories and the minimum update frequency, exit Spotfire. The repositories will be cloned locally the next time Spotfire is opened.

Local Repository Paths

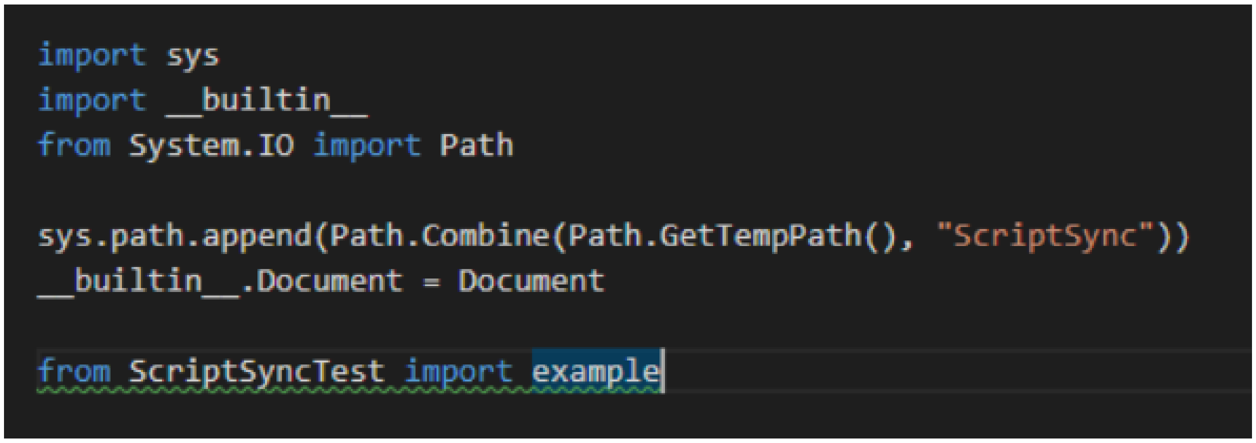
ScriptSync clones repositories locally to the ScriptSync directory in the Windows Temp folder. When using the Spotfire professional client, this will likely be C:\Users\<username>\AppData\Local\Temp\ScriptSync. The ScriptSync folder for the WebPlayer will likely be C:\Windows\Temp\ScriptSync. Both of these directory paths may depend on each machine's configuration. ScriptSync will not work if the user doesn't have write access to their Temp folder.

Using IronPython Modules

ScriptSync doesn't completely obviate the need for embedded scripts in Spotfire. Embedded scripts will still need to be created in Spotfire documents, and configured to pass appropriate parameters to the scripts. ScriptSync just makes it easy to call external IronPython scripts from repositories configured as modules.

The most straightforward way to use repositories cloned by ScriptSync is as IronPython modules. A Python module is simply a directory that contains a file named __init__.py. Individual scripts may then be imported using a standard import statement "from MODULE import FILE". If functions are defined in the __init__.py file, they may be imported by "import MODULE" then called using MODULE.function(), or imported by "from MODULE import FUNCTION" making the function available without the MODULE prefix. In short, all of the standard Python import and calling conventions may be used.

An example of calling an external script from a module is provided below.



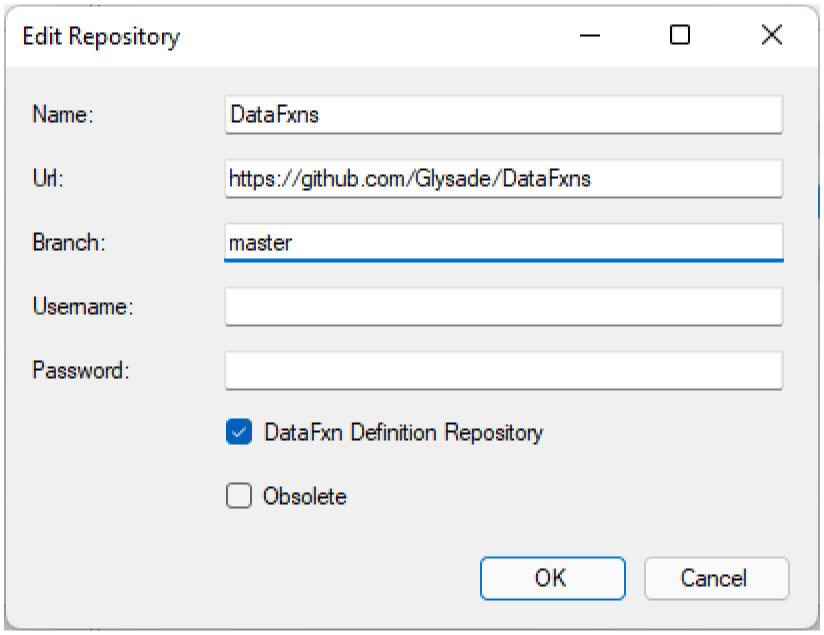
```
import sys
import __builtin__
from System.IO import Path

sys.path.append(Path.Combine(Path.GetTempPath(), "ScriptSync"))
__builtin__.Document = Document

from ScriptSyncTest import example
```

Calling an external IronPython script from an embedded script

IronPython modules may only be imported if the module is defined in the executing script's path. All repositories cloned by ScriptSync may be made available by adding the ScriptSync directory to the path. The line sys.path.append(Path.Combine(Path.GetTempPath(), "ScriptSync")) adds the ScriptSync directory to the search path. Subsequent import statements of ScriptSync repositories configured as modules will succeed after the search path has been modified. In the above example, the statement "from ScriptSyncTest import example" executes an IronPython script example.py located in the ScriptSyncTest module. The example.py script will execute exactly as if it were embedded in Spotfire with the exception that Spotfire always passes a reference to the Document when executing embedded scripts. If module scripts require a reference to the Document, the reference can be passed using the __builtin__ module. By assigning the Document to the __builtin__.Document field, the module script may reference the Document using the same __builtin__.Document field. This may only be necessary when executing module scripts, as opposed to calling methods defined in module scripts.

Data Functions

Glysade maintains publicly available data function definitions, and Python libraries that are used to enable data functions. Data functions in definition files are accessible to users through the Lead Discovery ChemCharts Data Functions visualization. Organizations have the option to configure the Glysade data function definitions repository, to maintain a private repository of data function definitions, or both as ScriptSync may be configured with multiple repositories in the preferences. The Glysade public data function definitions repository may be configured as follows



Edit Repository

Name:	DataFxns
Url:	https://github.com/Glysade/DataFxns
Branch:	master
Username:	
Password:	

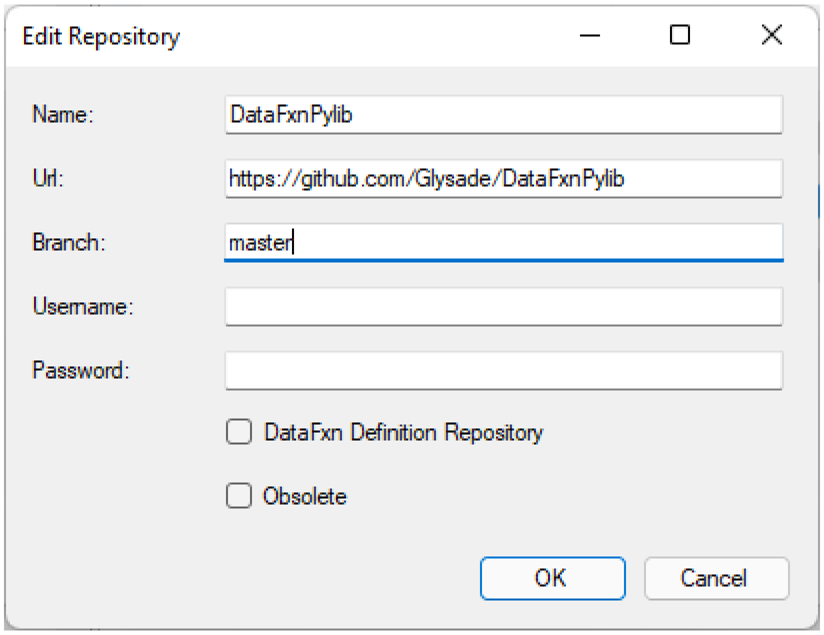
[✓] DataFxn Definition Repository

[ ] Obsolete

Glysade public data function definitions repository configuration

Note that when the *DataFxn Definition Repository* checkbox is checked, the repository will be written to the C:\Users\<username>\AppData\Local\Temp\ChemChartsDataFxns path instead of the …\Temp\ScriptSync directory.

Glysade data functions depend on the Glysade public DataFxnPylib python module repository. The DataFxnPylib repository may be configured in ScriptSync as



Edit Repository

Name:	DataFxnPylib
Url:	https://github.com/Glysade/DataFxnPylib
Branch:	master
Username:	
Password:	

[ ] DataFxn Definition Repository

[ ] Obsolete

Glysade public data function python library repository

Security

Replacing IronPython scripts in Spotfire documents can only be done manually by a user that is a member of a group with permissions to do so. Spotfire can execute any valid IronPython script, and the potential for executing malicious code is a potential security liability. Using ScriptSync to clone repositories is less secure than using embedded scripts given that users will have the ability to edit and execute scripts with the permissions granted to their Windows account. Users will not have the ability to 'push' modified scripts to private repositories unless they have access to the repository. Each organization must determine whether the additional security risk posed by using ScriptSync outweighs the benefits of automatic updates to IronPython scripts.