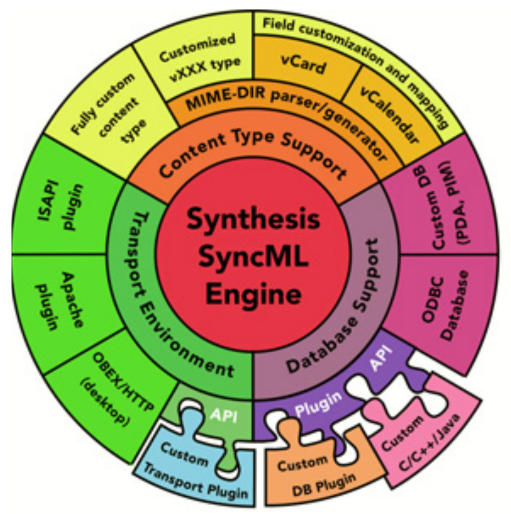
Script calling sequence for Server Sync Session



Session level
Sequence is run once per sync session)

Datastore level
Each datastore has its own set of these scripts, and if multiple datastores are involved in a sync session, these sequences are run separately for each datastore

Datatype level
Each datatype has its own set of these scripts. Usually, only one datatype is involved per datastore, but it is possibe that sending and receiving type is different.

Session context
This is the most "global" context for a session. Variables defined here are accessible from all other contexts using SESSIONVAR() and SETSESSIONVAR() functions.

Login Context
This context exists only during login. If you want to set session-wide parameters during login, use SETSESSIONVAR() to access session context's variables.

Datastore context
This context is only for a few scripts that operate on the fairly abstract level of SyncML "datastores", and are not involved in actual access to user data.

Database context
This context is for scripts which are related to the actual database interfacing of a datastore. They can access database related functions as the SQLxxxx() in SQL/ODBC datastores.

Datatype context
This context is for scripts that operate on the level of SyncML datatypes, such as vCard or RFC822 email. They operate on internal data fields (as defined in <fieldlist>), but cannot access the database. Primary use of datatype context scripts is for implementing special behaviour (such as merging, comparing, filtering) for data items of a certain type.

Session context	Login Context	Datastore context	Database context	Datatype context	
Initialisation of sync session					
sessioninitscript					first script executed. Define session-global variables here.
	logininitscript				called before accessing database for login. Can be used to implement fully custom login check
	logincheckscript				called to do extra checks to accept/deny a login and/or to store user-specific options into session variables
	loginfinishscript				has final say about allowing login. Can e.g. be used to log or count login failures.
		alertscript			called when alert command is received. Can be used to allow or deny certain sync modes, or force sync mode depending on server side user settings.
			adminreadyscript		called when administrative data has been read (e.g. date of last successful sync)
customgetputscript					can be used to issue custom get/put commands to a client.
rulescript					If a remote rule (device dependency rule) matches the current device, its rulescript is called. Usually, the rulescript sets session global flags and parameters using SETSESSIONVAR().
				initscript	called whenever a datatype is used by a datastore.
		datastoreinitscript			called before user data is accessed for the first time. Can be used to modify filters that define the sync set
				filterinitscript	called to check if a filter script must be run for every item to be sent to the client. In addition, it can request that **all** items (not only the modified/added ones) must be filtered to determine the sync set.
			optionfilterscript		called for SQL databases to calculate a WHERE clause for filtering while fetching data, thus enormously increasing performance when syncing small subsets of large data sets.
			initscript		called before reading or writing first user data item in the database. For <array> maps, this is called once per array before the first array element is read or written.
Per user data item in server's sync set					
			afterreadscript		called after reading an item from the database. Used to implement custom field conversions. When using <array> maps, these may execute separate initscript/afterreadscript/finishscript
				filterscript	called to perform custom filtering. Note that if filterinitscript returns FALSE, this scipt is not called
Per user data item received from client:					
				incomingscript	called after parsing data received from client. This script can be used to check for and eventually repair invalid data (like events too far in the past or future, missing mandatory fields)
				processitemscript	called before processing an incoming data command (like add, update, delete). Can be used to implement special behaviour like ignoring or rejecting items based on content. This is heavily used e.g. for non-symmetric datastores like email in the sample config
				comparescript	implements custom comparison between two data items - this is needed in slow sync or conflict case only
				mergescript	implements custom merging of data between two items (usually to keep som data from an item that looses a conflict into the item that wins the conflict)
			localidscript		called to obtain the identifier (like a table primary key) for adding a new data item into the database
			beforewritescript		called before writing an item to the database. Used to implement custom field conversions.
			afterwritescript		called after writing an item to the database. Can be used to store related detail data. When using <array> maps, these may execute separate initscript/beforewritescript/afterwritescript and finishscript
		receiveditemstatusscript			this is called to check and eventually modify the status code sent to the client. It can decide to abort the sync on
receiveditemstatusscript					certain error conditions or suppress or modify error codes. Script exists twice - on datastore and session levels.
Per item sent to client					
				outgoingscript	called before generating formatted data (e.g. a vCard item) to be sent to client. Can be used to normalize data, provide defaults or add data based on calculation (like a FN property in vCard).
Per status received for item from client					
		sentitemstatusscript			this is called to check and eventually modify the status code received from the client. It can decide to irgnore or
sentitemstatusscript					modify some errors reported by the client. Script exists twice - on datastore and session levels.
Finalisation of sync					
			finalisationscript		This script is called once for every item that was inserted or updated in the database, if the <fieldmap> contains at least one <map> with mode "x". This script is useful to establish inter-item database relations after all items are already synchronized.
			finishscript		called after all user data accesses (but not necessarily admin accesses) are done.
			syncendscript		called after all other database accesses are done - for successful as well as failed syncs. Can be used to write extra data to the database, for example summary data for the user.
		datastorefinishscript			last chance to do something related to this datastore sync. For example, use SHELLEXECUTE() to call an external program to finalize the sync.
customendputscript					can be used to issue a custom put after all datastores are finished (e.g. proprietary summary/status information for the client)
sessionfinishscript					last chance to do something related to this sync session.
Called when get, put or result command with unrecognized targetURI is received					
customputresulthandlerscript					called when a SyncML PUT, RESULT or GET command with unrecognized targetURI is received - can be used
customgethandlerscript					to handle proprietary extensions.