

DOCUMENTATION

Version 1.0 BETA, February 17, 2010

FlexGateQ!

© 2009 FlexDomino.net

PREFACE

We thank you for being a user of FlexGateQ!

This manual is designed to assist Adobe Flex developers in the use of the FlexGateQ! wrapper classes to consume the web services (API) provided by SoapGateQ! for Domino™.

These materials are copyrighted and the intellectual property rights are vested in FlexDomino.net.

Copying through any means is unauthorized without the express written permission of an officer of FlexDomino.net.

Every reasonable attempt has been made to ensure the accuracy of this user manual and that it reflects the operations of the product, however users of FlexGateQ! are responsible for ensuring that the product and its documentation are suitable for the needs of that organization. No warranties in respect of this user manual are being made or can be assumed.

© *FlexDomino.net*, 2010

Lotus, Lotus Notes, Domino and their respective logos as well as the IBM Business Partner logo are all registered trademarks of IBM Corporation.

Flex, Flex Builder, Flash and Flash Player as well as their respective logos are all trademarks of Adobe Corp. All other trademarks are *hereby duly acknowledged as the property of their respective owners.*

INDEX

TABLE OF CONTENTS

Preface.....	2
Index.....	3
Introduction.....	5
Installation and system requirements.....	6
Installation.....	6
API Reference.....	7
Overview.....	7
dominoDBUtilities class.....	8
result:Array [Bindable].....	8
result2d:Array [Bindable].....	8
resultAC:ArrayCollection [Bindable].....	8
xmlhier:HierarchicalData [Bindable].....	8
xmllist:XMLList [Bindable].....	8
xmlDoc:XMLDocument [Bindable].....	9
xml:XML [Bindable].....	9
customlistener:Function.....	9
customfaulthandler:Function.....	9
errortext:String	9
customerrorhandler:Function	10
customfinallyhandler:Function.....	10
mvalsep:String.....	10
serialsep:String.....	10
dominoDBUtilities():Void	11

Documentation for FlexGateQ!

dbcolum():Void	12
dbcolumnx():Void.....	13
dbrowx():Void.....	14
dblookup():Void.....	16
dblookupx():Void.....	17
dbgetformfields():Void.....	19
dbgetfieldtypes():Void.....	20
dbreaddocfields():Void.....	21
dbsavedocfields():Void.....	22
dbrenderdoc():Void.....	23
dbdeletedoc():Void.....	24
dbgetuserinfo():Void.....	25
dbsenddocument():Void.....	26
dbviews():Void.....	27
dbsearch():Void.....	28
dbviewftsearch():Void.....	30
dbftsearch():Void.....	32
dbcallagent():Void.....	34
dominoFormUtilities class.....	35
Appendix.....	36
Notes field and data types.....	36
Data Serialization.....	37

INTRODUCTION

FlexGateQ! Is a set of wrapper classes to consume web services provided by SoapGateQ! for Domino. Whilst SoapGateQ! is a universal gateway to Domino based on a web service API, the FlexGateQ! classes consume these web services and provide Flex/Flash specific functionality.

The collection of SoapGateQ! web service operations comprising a number of native Notes Formula and Lotus Script classes are mirrored in FlexGateQ! with some minor differences in the parameter list of each respective operation.

In addition, FlexGateQ! provides workflow functionality required in Flex/Flash to consume web services. As Flash functions strictly asynchronous, a web service call requires at least two listener (or callback) function that kick in once a web service result is available or failure occurred respectively.

Depending on the return value, some data conversion might be required (for instance the conversion from XML to an array collection) to allow data binding to the Flash applications UI components.

Another issue that arises from the asynchronous model of Flash is the complication that comes with the requirement of calling the same web service operation twice or more for different result sets. One cannot use easily the same listener functions and consequently would need to have a different instance of the web service object with its own listener functions for every required call. The latter can produce quite some messy code.

The FlexGateQ! wrapper classes provide functionality to avoid the above problems. Each instance of the FlexGateQ! class object has its own bindable result properties and listener functions. It also provides properties to assign custom listeners where required.

INSTALLATION AND SYSTEM REQUIREMENTS

FlexGateQ! is developed for Flex Builder 3. We will provide FlexGateQ! For Flash Builder 4 once it is released (Gold).

Installation

1. FlexGateQ! does not come with an installer as there are only a few simple steps to “plug and play”
2. Copy the FlexGateQ4Domino.swc file into the *lib* directory of your Flex project.
3. Add following line in the main Action Script or MXML file of your project (or where ever you want to call the web service operations from):

```
import packages.dominoUtilities.dominoDBUtilities;
```

Refer to the script samples and available demos for details on how to use the provided classes.

4. You require of course a Domino server and SoapGateQ! Installed on the Domino server. Please refer to the SoapGateQ! Manual for the respective installation instructions.

API REFERENCE

Overview

Currently FlexGateQ! consists of two wrapper classes:

dominoDBUtilities

This class mirrors mirrors the web service operations comprised in SoapGateQ!

dominoFormUtilities

This class provides workflow functionality for handling Flex/Flash forms in a similar way a Domino developer would work with Notes forms (desktop client). This class integrates with the dominoDBUtilities class and calls its respective web service operations to read, save and send Notes documents.

Under development...

dominoViewUtilities

This class provides workflow functionality for handling Flex/Flash data grids in a similar way a Domino developer would work with Notes views and folders (desktop client). This class integrates with the dominoDBUtilities class and calls its respective web service operations to read Notes views. It also integrates with the dominoFormUtilities class to read, save and send the Notes document listed in the Flex data grid.

Note:

Return value refers to the result returned by the web service operation which is stored in the properties *.result*, *.result2d* and or *.resultAC* of respective the object instance of the dominoDBUtilities class.

The return value of the web service operation call itself is always *Void*.

dominoDBUtilities class

result:Array [Bindable]

The *result* property contains the resulting array from those web service operations that return a single or 2-dimensional array. For the latter the *residx* parameter determines the column used for this single dimensional array property (e.g. *dbcolumn()*, *dblookup()*, *dbcolumnX()*, *dblookupX()*).

result2d:Array [Bindable]

The *result2d* property contains the resulting 2-dimensional array of the respective web service operations returning such arrays (e.g. *dbcolumnX()*, *dblookupX()*).

resultAC:ArrayCollection [Bindable]

The *resultAC* property contains an array collection build from the resulting 2-dimensional array of the respective web service operations returning such arrays (e.g. *dbcolumnX()*, *dblookupX()*).

The *resultAC* property is used for data grid binding, as data grids require array collections for their *dataprovider* property rather than 2-dimensional arrays.

xmlhier:HierarchicalData [Bindable]

For future use.

The *xmlhier* property will contain hierarchical formatted XML coming from not yet implemented web service operations returning data from categorized Notes views. The *xmlhier* property will be used by the not yet implemented *dominoViewUtilities* class.

xmllist:XMLList [Bindable]

For future use.

The *xmldoc* property will contain XML formatted data coming from not yet implemented web service operations. The *xmldoc* property will be used by the not yet implemented *dominoViewUtilities* class.

xml:XMLDocument [Bindable]

For future use.

The *xml* property will contain XML formatted data coming from not yet implemented web service operations. The *xml* property will be used by the not yet implemented *dominoViewUtilities* class.

xml:XML [Bindable]

The *xml* property contain XML formatted data coming from web service operations such as the *dbCallAgent()* method. The *xml* property will be used by the not yet implemented *dominoViewUtilities* class.

customlistener:Function

Custom web service result listener. After instantiating the *dominoDBUtilities* class object, this property can be set before any of the methods calling a web service operation is executed to allow for a customized result handling.

The internal result handler is executed before the *customlistener* function.

customfaulthandler:Function

Custom web service fault listener. After instantiating the *dominoDBUtilities* class object, this property can be set before any of the methods calling a web service operation is executed to allow for customized handling of system faults.

The internal fault handler is not executed if *customfaulthandler* is set.

errortext:String

If a web service operation throws a known error exception (server side), the operation as such succeeds and the web operations result listener is executed, rather than the fault listener. However, the web service return value will indicate the server side exception and will trigger the execution of the *customfaulthandler* function. It will also set the *errortext* property to the by the web service operation returned error text.

customerrorhandler:Function

Custom error handler (*catch{...}*). After instantiating the *dominoDBUtilities* class object, this property can be set before any of the methods calling a web service operation is executed to allow for customized handling of any code execution exceptions.

The internal error handler is not executed if *customerrorhandler* is set.

customfinallyhandler:Function

Custom finally handler (*finally{...}*). After instantiating the *dominoDBUtilities* class object, this property can be set before any of the methods calling a web service operation is executed to allow for customized handling of any code execution exceptions.

The internal finally handler is not executed if *customfinallyhandler* is set.

mvaluep:String

Separator for first level array (multi-value) serialization.

Default value: "~" (tilde)

Refer to the *Appendix* for more information about data serialization.

serializep:String

Separator for second level array serialization.

Default value: "-"

Refer to the *Appendix* for more information about data serialization.

dominoDBUtilities():Void

Instantiation method for the dominoDBUtilities class.

Parameters:

rURL	String	Root URL to the Domino server on which the SoapGateQ! Database resides
aURL	String	Filepath of the SoapGateQ! database
authent	Boolean	IF true an authentication attempt is done during the object's instantiation
username	String	Username to be authenticated with
password	String	User's password
loginOK	Function	Custom authentication (login) success handler. With the instantiation of the <i>dominoDBUtilities</i> class object, one can provide parameters to attempt a session authentication with the Domino server and to control what happens on successful authentication by providing a custom log-in failure and success callback function.
loginFailure	Function	Custom authentication (login) failure handler. With the instantiation of the <i>dominoDBUtilities</i> class object, one can provide parameters to attempt a session authentication with the Domino server and to control what happens on authentication failure by providing a custom log-in failure callback function.
mainDspObj	DisplayObject	Parent display container in which all dialogs appear

Return value:

Void

dbcolumn():Void

Web service implementation of the Notes Formula function @dbcolumn().

Parameters:

cache	Boolean	if false Domino will update its cache from the data on file
srvname	String	Name of the Domino server on which the source database resides
dbname	String	Pathname of the database (relative to the root data directory of the server)
viewname	String	Name or alias of the view, which data is to be returned
colnumber	Integer	Column number to be returned starting from 1 for the first column in the view

Return value:

Array:Any

The result from the web service operation is stored in the property *result:Array*

dbcolumnx():Void

Web service implementation of the Notes Formula function @dbcolumn() with two extended feature:

- returns one or more columns
- returns columns and or fields

Parameters:

srvname	String	Name of the Domino server on which the source database resides
dbname	String	Pathname of the database (relative to the root data directory of the server)
viewname	String	Name or alias of the view, which data is to be returned
colfields	Array:String	List of column numbers (string, starting from "1") and or fieldnames
addUNID	Boolean	If true an additional column is returned containing the document universal lds
addNoteID	Boolean	If true (and addUNID = False) an additional column is returned containing the document notes lds
residx	Integer	Default column for <i>result</i> property
rowformat	Boolean	If true data is returned by row rather than by column (see <i>Appendix</i>). If the result is used as a data provider for a data grid, this parameter must be true.

Return value:

Array:Any

The result from the web service operation is stored in the property *result2d:Array*, *resultAC:ArrayCollection* and *result:Array* (default column).

dbrowx():Void

Returns column and or field values for a single view row.

Supplemental method to @dbcolumnx(). In principle the method works the same way, however it will return the columns for one Notes document only . The method is used to re-read the values of a single data grid row.

It is called whenever a document representing a particular data grid row is amended using a Flex form (through the dominoFormUtilities class). Consequently we can omit the rowformat parameter as it must be true anyway.

Though this method can obviously be used in a different context it is implemented to be automatically called from the dominoFromUtilities class.

Note: either the addUNID or the addNotesID parameter has to be true and the values have to match the respective parameter setting used for the dbcolumnx() call.

Parameters:

srvname	String	Name of the Domino server on which the source database resides
dbname	String	Pathname of the database (relative to the root data directory of the server
viewname	String	Name or alias of the view, which data is to be returned
colfields	Array:String	List of column numbers (string, starting from "1" for the first column in the view) and or fieldnames
addUNID	Boolean	If True an additional column is returned containing the document universal Ids
addNoteID	Boolean	If True (and addUNID = False) an additional column is returned containing the document notes Ids
residx	Integer	Default column for <i>result</i> property
docUNID	String	Universal ID of the document representing the single view row to be returned

Return value:

Array:Any

The result from the web service operation is stored in the property *result2d:Array*, *resultAC:ArrayCollection* and *result:Array* (default column).

dblookup():Void

Web service implementation of the Notes Formula function @dblookup().

Parameters:

cache	Boolean	if false Domino will update its cache from the data on file
srvname	String	Name of the Domino server on which the source database resides
dbname	String	Pathname of the database (relative to the root data directory of the server)
key	String	A value to be matched against the first sorted column in the view
viewname	String	Name or alias of the view, which data is to be returned
colnumber	Integer	Number of a column in the view starting at 1. If a column number is used, the return values are taken from this column in the view
fieldname	String	Name of an item in the documents in the view. If an item name is used, the return values are taken from this item in the documents from which the view data is drawn
failsilent	Boolean	returns "" (null string) instead of an error if the key cannot be found
partialmatch	Boolean	returns a match if the key matches the beginning characters of the column value
returnUNID	Boolean	returns the UNID of the document instead of a field or column value

Return value:

Array:Any

The result from the web service operation is stored in the property *result:Array*

dblookup():Void

Web service implementation of the Notes Formula function @dblookup() with two extended feature:

- returns one or more columns
- returns columns and or fields

Parameters:

cache	Boolean	if false Domino will update its cache from the data on file
srvname	String	Name of the Domino server on which the source database resides
dbname	String	Pathname of the database (relative to the root data directory of the server)
key	String	A value to be matched against the first sorted column in the view
viewname	String	Name or alias of the view, which data is to be returned
colfields	Array:String	List of column numbers (string, starting from "1" for the first column in the view) and or fieldnames
failsilent	Boolean	returns "" (null string) instead of an error if the key cannot be found
partialmatch	Boolean	returns a match if the key matches the beginning characters of the column value
returnUNID	Boolean	returns the UNID of the document in addition to the requested fields and or columns
returnNoteID	Boolean	returns the UNID of the document in addition to the requested fields and or columns (works only if <i>returnUNID</i> = False)
residx	Integer	Default column for <i>result</i> property
rowformat	Boolean	If true data is returned by row rather than by column (see <i>Appendix</i>). If the result is used as a data provider for a data grid, this parameter must be true.

Return value:

Array:Any

The result from the web service operation is stored in the property *result2d:Array*, *resultAC:ArrayCollection* and *result:Array* (default column).

dbgetformfields():Void

Returns the list of fieldnames and -types contained in a specified form.

Parameters:

srvname	String	Name of the Domino server on which the source database resides
dbname	String	Pathname of the database (relative to the root data directory of the server)
form	String	Name or alias of the form, which field list is to be returned
includeLSF	String	If true includes Lotus \$ (system) fields

Return value:

Array:Any

The result from the web service operation is stored in the property *result2d:Array* and *result:Array* (default column = fieldnames).

The returned array is a 2-dimensional array (2 columns, x rows), where column 1 (index 0) contains the field names and column 2 (index 1) the corresponding field or data type.

dbgetfieldtypes():Void

Returns the data types for a given list of fields contained in a given form.

Parameters:

srvname	String	Name of the Domino server on which the source database resides
dbname	String	Pathname of the database (relative to the root data directory of the server)
form	String	Name or alias of the form, which fieldtype list is to be returned
fields	Array:String	List of fieldnames for which the field type is to be returned

Return value:

Array:Integer

The result from the web service operation is stored in the property *result:Array*.

Data types:

See *Appendix*

dbreaddocfields():Void

Returns the values and data types for the list of given fields of a given document.

Parameters:

srvname	String	Name of the Domino server on which the source database resides
dbname	String	Pathname of the database (relative to the root data directory of the server)
docUNID	String	Universal ID of the document which field values are to be returned
fields	Array:String	List of fields for which values are to be returned

Return value:

Array:String

The result from the web service operation is stored in the property *result2d:Array* and *result:Array* (default column = field values).

All field values are converted to string. This complies with the way the Notes Client UI document work. Whilst edited all fields in a Notes document are edited as text. Default and custom validations control the users input.

Similar to this concept the web service consumer, for instance a Flex form must have input formatters and validations implemented.

Alternatively the consumer must convert the returned string data in accordance with the also provided data type information into whatever is required.

The correlating web service operation *dbSaveDocFields* converts the string data back into the respective data types required in the used Notes form for the document being modified.

See *Appendix* for more information.

dbsavedocfields():Void

Updates field values in a given document or saves values in a new document.

Parameters:

srvname	String	Name of the Domino server on which the source database resides
dbname	String	Pathname of the database (relative to the root data directory of the server)
docUNID	String	Universal ID of the document which field values are to be returned
fields	Array:String	List of fields for which are to be updated
types	Array:Integer	List of corresponding data types. Please refer to the <i>Appendix</i> for the Notes field data types
values	String	List of field values
form	String	Name of form used for new documents
compWF	Boolean	If true runs the NotesDocument.ComputeWithForm() method on the server before the document is saved to file
saveRPC	String	Runs a Notes agent on the server before the document is saved file. This is the equivalent to the WebQuerySave event when submitting a Notes form

Return value:

Boolean

The result from the web service operation is stored in the property `success:Boolean`.

dbrenderdoc():Void

This feature is not yet implemented.

Renders a Notes document into HTML or MIME

Parameters:

srvname	String	Name of the Domino server on which the source database resides
dbname	String	Pathname of the database (relative to the root data directory of the server)
docUNID	String	Universal ID of the document which is to be rendered
noteID	String	Alternatively the NoteID
form	String	Name of form used for rendering the document
noCache	Boolean	True or false. Not yet supported parameter

Return value:

String:HTML/MIME

The result from the web service operation is stored in the property *result:Array* (index 0).

dbdeletedoc():Void

Deletes a document from file.

Parameters:

srvname	String	Name of the Domino server on which the source database resides
dbname	String	Pathname of the database (relative to the root data directory of the server)
docUNID	String	Universal ID of the document which field values are to be returned
force	Boolean	IF true the document will be deleted even if it is currently edited by another user

Return value:

Boolean

The result from the web service operation is stored in the property *success: Boolean*.

dbgetuserinfo():Void

Web service implementation of the Domino class method Notesdirectory.Getmailinfo().

Parameters:

username	String	Name of the registered user
----------	--------	-----------------------------

Return value:

Array:String

Elements returned are as follows:

Mail Server	Home mail server for the specified person
BuildNumber	If getver is true, a string representation of the build number of the specified person's mail server, for example, "303". If getver is false, ""
DominoVersion	If getver is true, a string representation of the Domino version of the specified person's mail server, for example, "Build V80_07042006NP". If getver is false, ""
MailFile	Mail file for the specified person
ShortName	Short form of the specified person's name
MailDomain	Notes Domain of the specified person's mail address
User Name	First entry in the list of user names honored for the specified person
InternetMailAddress	Internet mail address for the specified person
OutOfOffice	Out of Office service type. "1" indicates Agent, "2" indicates Service

dbsenddocument():Void

Web service implementation of the Domino class method
NotesDocument.Send().

Parameters:

srvname	String	Name of the Domino server on which the source database resides
dbname	String	Pathname of the database (relative to the root data directory of the server)
docUNID	String	Universal ID of the document which field values are to be returned
attachForm	Boolean	Attach form used for the send document (as defined in the <i>Form</i> field)
recipients	Array:String	List of recipients in a single string separated by the character passed on in <i>mvaluep</i>

Return value:

Boolean

The result from the web service operation is stored in the property
success:Boolean.

dbviews():Void

Web service implementation of the Domino class property NotesDatabase.Views().

Parameters:

srvname	String	Name of the Domino server on which the source database resides
dbname	String	Pathname of the database (relative to the root data directory of the server)
viewonly	Boolean	If true returns views only
folderonly	Boolean	If true returns folders only
includehidden	Boolean	If true returns also hidden views and folders

Return value:

String:XML

```
<views>
```

```
  <node label="By Severity" data="vwEventsBySeverity1"/>
```

```
  <node label="By Date" data="vwEventsByDate1"/>
```

```
  <node label="By Type" data="vwEventsByType1"/>
```

```
  <node label="By Database" data="vwEventsByDatabase1"/>
```

```
  <node label="By Server" data="vwEventsByServer1"/>
```

```
  <node label="By Assignment" data="vwEventsByAssignment1"/>
```

```
</views>
```

The *label* attribut contains the view name, the *data* attribut contains the alias name of the view or if not alias is available the view name.

dbsearch():Void

Web service implementation of the Domino class method NotesDatabase.Search() with extended features:

- returns one or more columns
- returns columns and or fields
- returns document collection in form of list of UNIDs/NoteIDs

Parameters:

srvname	String	Name of the Domino server on which the source database resides
dbname	String	Pathname of the database (relative to the root data directory of the server)
formula	String	A Notes @function formula that defines the selection criteria
datetime	String	A cutoff date. The method searches only documents created or modified since the cutoff date. Specify "" (null string) to indicate no cutoff date.
maxdocs	Integer	Maximum number of documents returned
colfields	Array:String	List of column numbers (string) and or field names to be returned
returnUNID	Boolean	Returns the UNID of the document in addition to the requested fields and or columns
returnNoteID	Boolean	Returns the UNID of the document in addition to the requested fields and or columns (works only if <i>returnUNID</i> = false)
residx	Integer	Default column for <i>result</i> property
rowformat	Boolean	If true data is returned by row rather than by column (see <i>Return value</i>)

Return value:

Array:Any

The result from the web service operation is stored in the property *result2d:Array*, *resultAC:ArrayCollection* and *result:Array* (default column).

dbviewftsearch():Void

Web service implementation of the Domino class method NotesView.FTSearch() with extended features:

- returns one or more columns
- returns columns and or fields
- returns document collection in form of list of UNIDs/NoteIDs

Parameters:

srvname	String	Name of the Domino server on which the source database resides
dbname	String	Pathname of the database (relative to the root data directory of the server)
viewname	String	Name or alias of the view, which data is to be returned
query	String	The full-text query. Please refer to the Lotus Notes Help for syntax information
maxdocs	Integer	The maximum number of documents you want returned from the query. Set this parameter to 0 to retrieve all documents (up to 5,000 - see the Usage section for more details) that meet the search criteria
colfields	Array:String	List of column numbers (string) and or fieldnames which values are to be returned
returnUNID	Boolean	Returns the UNID of the document in addition to the requested fields and or columns
returnNoteID	Boolean	Returns the UNID of the document in addition to the requested fields and or columns (works only if <i>returnUNID</i> = false)
residx	Integer	Default column for <i>result</i> property
rowformat	Boolean	If true data is returned by row rather than by column (see <i>Return value</i>)

Return value:

Array:Any

The result from the web service operation is stored in the property *result2d:Array*, *resultAC:ArrayCollection* and *result:Array* (default column).

dbftsearch():Void

Web service implementation of the Domino class method NotesDatabase.FTSearch() with extended features:

- returns one or more columns
- returns columns and or fields
- returns document collection in form of list of UNIDs/NoteIDs

Parameters:

srvname	String	Name of the Domino server on which the source database resides
dbname	String	Pathname of the database (relative to the root data directory of the server)
query	String	The full-text query. Please refer to the Lotus Notes Help for syntax information
maxdocs	Integer	The maximum number of documents you want returned from the query. Set this parameter to 0 to retrieve all documents (up to 5,000 - see the Usage section for more details) that meet the search criteria
sortoption	Integer	Optional. Sorting options. Choose one (64) sorts by document date in ascending order. (32) sorts by document date in descending order. (1543) sorts by document creation date in ascending order. (1542) sorts by document creation date in descending order. (8) sorts by relevance score (default)
otheroptions	Integer	Optional. Search options. Combine options by adding (8192) includes Domino databases. (16384) searches for related words. Need not be an exact match.

Documentation for FlexGateQ!

		(4096) includes files that are not Domino databases. (512) uses stem words as the basis of the search. (1024) uses thesaurus synonyms
colfields	Array:String	List of column numbers (string) and or fieldnames which values are to be returned
returnUNID	Boolean	returns the UNID of the document in addition to the requested fields and or columns
returnNotelD	Boolean	returns the UNID of the document in addition to the requested fields and or columns (works only if <i>returnUNID</i> = False)
residx	Integer	Default column for <i>result</i> property
rowformat	Boolean	If true data is returned by row rather than by column (see <i>Return value</i>)

Return value:

Array:Any

The result from the web service operation is stored in the property *result2d:Array*, *resultAC:ArrayCollection* and *result:Array* (default column).

dbcallagent():Void

Web service implementation of the Domino class property NotesAgent.Run().

Parameters:

srvname	String	Name of the Domino server on which the source database resides
dbname	String	Pathname of the database (relative to the root data directory of the server)
agent	String	Name of agent to be executed
paramXML	String:XML	Any XML formatted string which will be attached to an agent context document in a pre-defined field called <i>agentcontextXML</i>
noteID	String	Note ID to an agent context document. If no note ID is provided the web service will create a temporary agent context document

Return value:

String:XML

The return value is either "" (null string) or taken from the agent context document from the pre-defined field *agentcontextXML*. It is stored in the property *xml:XML*.

dominoFormUtilities class

xxx():Void

APPENDIX

Notes field and data types

ATTACHMENT (1084)	means file attachment
AUTHORS (1076)	means authors
DATETIMES (1024)	means date-time value or range of date-time values
EMBEDDEDOBJECT (1090)	means embedded object
HTML (21)	means HTML source text
MIME_PART (25)	means MIME support
NAMES (1074)	means names
NOTELINKS (7)	means link to a database, view, or document
NOTEREFs (4)	means reference to the parent document
NUMBERS (768)	means number or number list
READERS (1075)	means readers
RFC822Text (1282)	means RFC822 Internet mail text
RICHTEXT (1)	means rich text
SIGNATURE (8)	means signature
TEXT (1280)	means text or text list

There are more data types. Please refer to the Lotus Notes Designer documentation.

Data Serialization

At the current stage of development most FlexGateQ! web service operations return a single dimensional array, regardless of whether the operations does collect the requested data in a 2-dimensional array. The Domino 8+ server serializes a 2-dimensional array to a single dimensional array.

As a result an array of values collected by e.g. dbcolumnX for the first 3 columns of a view looking like this...

Column1 Row1	Column2 Row1	Column3 Row1
Column1 Row2	Column2 Row2	Column3 Row2
Column1 Row3	Column2 Row3	Column3 Row3

...is converted to a single dimensional array looking like this...

Column1 Row1
Column1 Row2
Column1 Row3
Column2 Row1
Column2 Row2
Column2 Row3
Column3 Row1
Column3 Row2
Column3 Row3

For some of the offered web service operations the parameter *rowformat* determines the way the data serialization works:

rowformat = True	rowformat = False
------------------	-------------------

Documentation for FlexGateQ!

row1_column1 row1_column2 row1_column3 ... row2_column1 row2_column2 ...	row1_column1 row2_column1 row3_column1 ... row1_column2 row2_column2 ...

Data serialization also effects multi-value column entries or fields. Multi-value column and field values will be converted to type string and each value separated by the character passed on in the *mvalsep* parameter available to all effected web service operations. The data types for single value columns and fields remain intact.

The consumer of the FlexGateQ! web service operation must reverse the serialization and create a 2-dimensional array or array collection from the returned single dimensional array. It also needs to revert back serialized multi-values.

The wrapper classes of FlexGateQ! (for Flex Builder 3) are doing this for all web service operations of FlexGateQ! where the above applies.

Multi-value separator considerations:

The character used in *mvalsep* (default "~" tilde) cannot be part of any of the data columns and or fields received from or send to the server. The same applies for the second (fixed) serialization character "-" used by the system.