

Caspio Bridge Web Service API

Programmer's Documentation

Caspio Bridge Version 6.5
Document last updated on October 15, 2009

Table of contents

1	What Is In This Document	4
2	Getting Started with Caspio Bridge WS API	4
2.1	Web Service Profiles	4
2.2	MS SOAP Toolkit.....	5
2.2.1	Visual Basic 6.0	5
2.2.2	MS Office 2003 or higher with Office SOAP Toolkit.....	6
2.3	Microsoft .Net Framework	6
2.4	Apache Axis	6
2.5	PHP 5.0.....	7
2.6	SOAP 1.2	7
2.6.1	RPC-style and Document-style methods	7
2.6.2	Complex connectivity sample: Caspio Bridge WS API and Crystal Reports 2008	7
3	Programmer's Reference	9
3.1	General Information.....	9
3.2	Parameters.....	9
3.3	RPC-style Methods.....	12
3.3.1	SelectDataRow	12
3.3.2	SelectDataXML.....	13
3.3.3	InsertData	15
3.3.4	UpdateData.....	16
3.3.5	DeleteData.....	16
3.3.6	CreateEmptyTable.....	17
3.3.7	AddTableField.....	17
3.3.8	DropTableField.....	18
3.3.9	AlterTableField.....	18
3.3.10	GetTableDesignRaw	18
3.3.11	ProximitySearchByCoordinates.....	19
3.3.12	ProximitySearchByCoordinatesRaw.....	20
3.3.13	ProximitySearchByReference	21
3.3.14	ProximitySearchByReferenceRaw	22
3.3.15	GetDataPageAppKey	23
3.3.16	GetDataPageDeploymentStatus.....	23
3.3.17	GetDataPageDeploymentCode	24
3.3.18	DeployDataPage	24
3.3.19	UndeployDataPage.....	24
3.3.20	ListDataPagesRaw.....	25
3.3.21	CreateDataPage.....	25
3.3.22	CreateView	26
3.3.23	ListFiles.....	27
3.3.24	UploadFile.....	27
3.3.25	DownloadFile.....	28
3.3.26	DeleteFile	29
3.4	Document-style Methods	29
3.4.1	SelectData	29
3.4.2	CreateTable.....	30
3.4.3	GetTableDesign.....	31
3.4.4	ListObjects	33
3.4.5	ListDataPages.....	34
3.5	Types	36

3.5.1	Enumerations	36
3.5.2	Structures	38
3.6	Faults	39
4	Code Samples	42
4.1	MS SOAP Toolkit	42
4.1.1	Visual Basic 6.0	42
4.1.2	MS Office with Web Services Toolkit	42
4.1.3	ASP and SOAP toolkit	42
4.2	Microsoft .Net Framework	43
4.3	Apache Axis	43
4.4	PHP 5.0	43
4.4.1	Orders sample	43
4.4.2	AutoComplete sample	44
5	Resources	44

1 What Is In This Document

This document describes Caspio Bridge Web Services Application Programming Interface (WS API). It contains programmer's reference, code examples and guidelines for developers of WS API client software.

Important Note

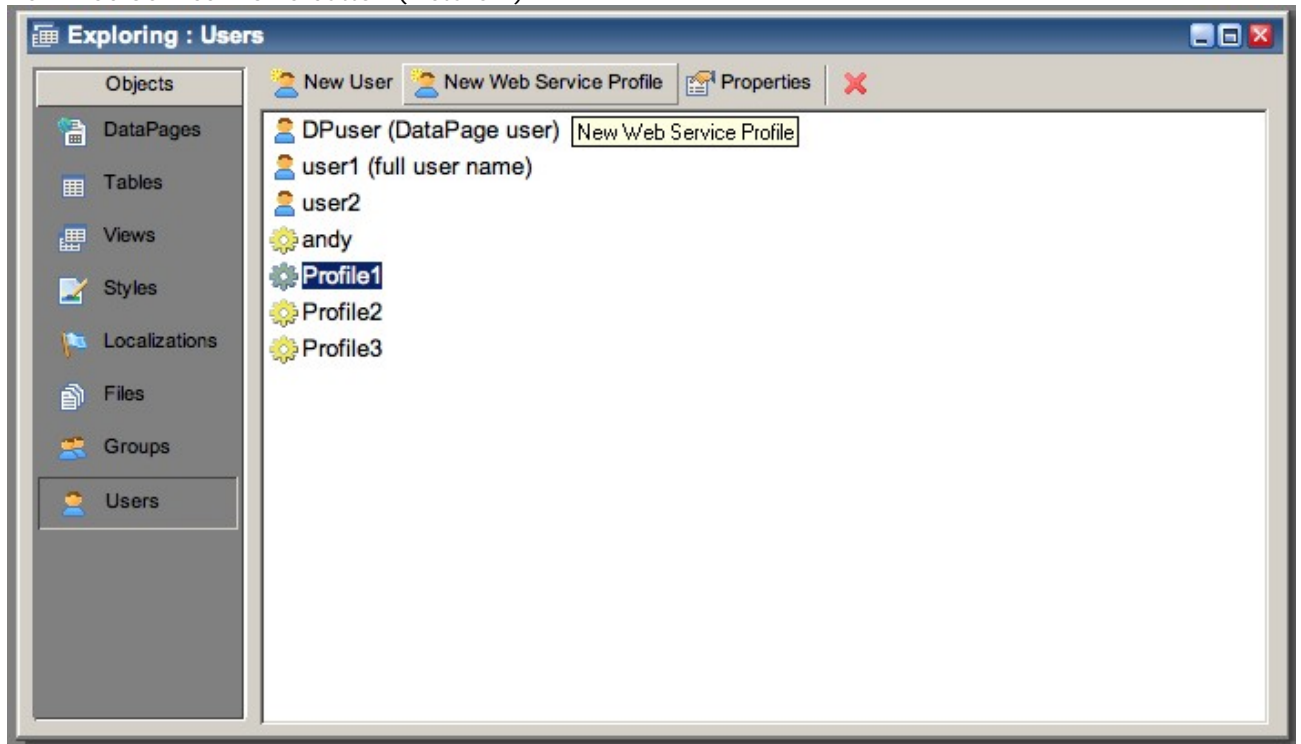
Throughout the document you will see mentioning of `<BridgeSiteHostName>` or `bridge.caspio.net`. `<BridgeSiteHostName>` (or `bridge.caspio.net`) needs to be replaced with the appropriate domain name that hosts your Caspio Bridge application. That domain name could be `bridge.caspio.net`, `b2.caspio.com`, `b3.caspio.com` or other. Information about WSDL URL can be found in Web Service Profile Properties dialog (see Picture 2)

2 Getting Started with Caspio Bridge WS API

This section contains detailed instructions on how to quickly start developing a WS API client. Please read it if you are new with SOAP, WSDL or Web Services in general.

2.1 Web Service Profiles

All WS API methods require Caspio Bridge account has at least one Web Service Profile. Web Service Profile is a special kind of Caspio Bridge user that holds WS API access settings. Like users, Web Service Profiles may be given privileges to create and access tables, views, files and file folders in corresponding Caspio Bridge account. Unlike users, Web Service Profiles cannot log into Caspio Bridge interactively. To create Web Service Profile, open Users tab in Object Explorer window in Bridge user interface and click New Web Service Profile button (Picture 1)



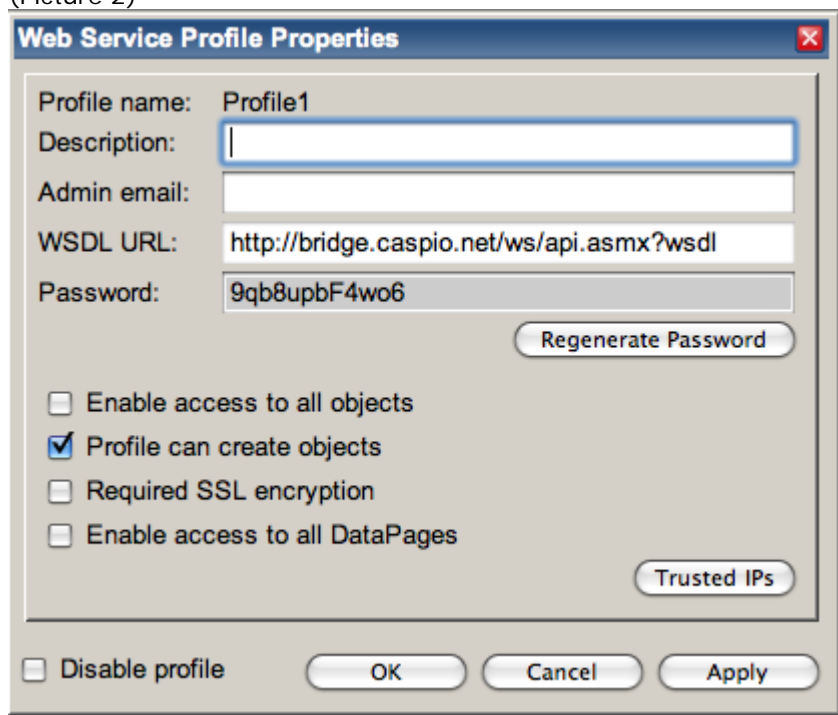
Picture 1

Web Service Profiles have the following properties:

- **Profile name** – Name of the profile, specified once during profile creation.
- **Description** – optional profile description.
- **Admin email** – optional email to be stored together with profile properties.

- **Password** – Profile's password. By default, passwords are generated automatically. Click **Regenerate Password** to create new password. You can type your own password – in this case password has to be a mix of digits, lowercase and uppercase letters, and be at least 10 characters long.
- **Enable access to all objects** – Check this option to make profile a member of Administrators group. This gives the profile all access privileges to all tables, views, files, and file folders in corresponding Caspio Bridge account.
- **Profile can create objects** – Profile with this option turned on can create tables, views, DataPages, and upload files.
- **Required SSL encryption** – Check this option to disable WS API access for this profile using insecure (HTTP) protocol.
- **Enable access to all DataPages** – Allows access to DataPage-related methods of WS API to this profile. Note that in order to create DataPage, a profile should be given **Profile can create objects** privilege. From the other hand, even a profile with **Enable access to all objects** privilege cannot use DataPage-related methods of WS API without this option set.
- **Disable profile** – Check this option to disable Web Service Profile.

All properties except **Name** can be changed at any time using Web Service Profile Properties dialog (Picture 2)



Picture 2

WSDL URL field on **Properties** dialog holds information about WSDL URL for particular account. (See section 4.1 note below)

Additionally, Web Service profile access to Caspio Bridge can be limited based on client's IP. Click **Trusted IPs** button to configure IP-based access restrictions.

2.2 MS SOAP Toolkit

MS SOAP Toolkit 3.0 can be downloaded from:

<http://www.microsoft.com/downloads/details.aspx?familyid=c943c0dd-ceec-4088-9753-86f052ec8450&displaylang=en>.

2.2.1 Visual Basic 6.0

1. Start Visual Basic 6.0 and create a new Standard EXE project.
2. Place a command button (Command1) on Form1 and double-click on Command1

3. Select from menu **Project, References**, select "Microsoft Soap Type Library, v3.0" and click **OK**
4. In Code window under Private Sub Command1_Click() type:

```
Dim aobjWS As New MSSOAPLib30.SoapClient30
aobjWS.MSSoapInit "https://bridge.caspio.net/ws/api.asmx?wsdl"
```

5. From this point you can use aobjWS variable in your code, as described in examples in this document.

2.2.2 MS Office 2003 or higher with Office SOAP Toolkit

1. Ensure that Office SOAP Toolkit is installed (it is listed in Office 2003 setup in Office Shared Features section)
2. Install Office 2003 Web Services toolkit (it can be downloaded from: <http://www.microsoft.com/downloads/details.aspx?FamilyID=fa36018a-e1cf-48a3-9b35-169d819ecf18&DisplayLang=en>.)
3. Start any MS Office application
4. Press Alt+F11 to bring up Visual Basic Editor
5. Select menu **Tools, Web Service reference**.
6. Select **Web service URL** radio button, enter <https://<BridgeSiteHostName>/ws/api.asmx?wsdl> (see Section 4.1 note) and click **Search**
7. In **Search Result** pane check CaspioBridgeAPI and click **Add**
8. Web Services Toolkit will generate a proxy class for you. Note that definitions for custom types will not be generated.

For more information about MS SOAP Toolkit usage, see the Microsoft SOAP Toolkit documentation (by default it is installed at C:\Program Files\MSSOAP\Documentation\soap3.chm)

2.3 Microsoft .Net Framework

1. Start a new Visual Studio.Net project.
2. Select menu **Project, Add Web Reference**.
3. In the **URL** field, enter <https://<BridgeSiteHostName>/ws/api.asmx?wsdl>. (See Section 4.1 note) Visual Studio will generate a proxy class that encapsulates all WS API custom types and method calls.

2.4 Apache Axis

1. Create a new Eclipse project
2. Add the following jars to the project
 - **axis.jar**
 - **jaxrpc.jar**
 - **saaj.jar**
 - **commons-logging.jar**
 - **commons-discovery-0.2.jar**
 - **commons-logging-1.0.4.jar**
 - **log4j-1.2.8.jar**
 - **wsdl4j-1.5.1.jar**
 - **mail.jar** (<http://java.sun.com/products/javamail/>)
 - **activation.jar** (<http://java.sun.com/products/javabeans/glasgow/jaf.html>)
 - A JAXP-1.1 compliant XML parser, e.g. Xerces (**xml-apis.jar**, **xercesImpl.jar**) You can find **xml-apis.jar** and **xercesImpl.jar** in Tomcat 5 directory `<TOMCAT_HOME>\common\endorsed\`.
3. Ensure that AXISCLASSPATH includes the jars listed above.

4. Use WSDL2Java tool to generate proxy class which encapsulate all WS API types and method calls. The basic invocation form looks like this:

```
java -cp %AXISCLASSPATH% org.apache.axis.wsdl.WSDL2Java (WSDL-file-URL)
or
java -cp %AXISCLASSPATH% org.apache.axis.wsdl.WSDL2Java
CaspioBridgeAPI.wsdl
```

2.5 PHP 5.0

In order to turn on SOAP and HTTPS support in PHP 5.0 you need to uncomment the following lines in "Windows Extensions" section of php.ini:

```
extension=php_curl.dll
extension=php_openssl.dll
extension=php_soap.dll
```

and restart the web server.

Note that directories with PHP extension must be included in the Path environment variable.

Use the following code to create a new SOAP Client object instance and call the WS API method:

```
$SoapClient = new SoapClient("https://bridge.caspio.net/ws/api.asmx?wsdl");
$SoapClient->functionName(argument1, ..., [argumentN]);
```

2.6 SOAP 1.2

There are several versions of SOAP – a protocol to access and utilize Web services. In order to support wide range of SOAP-aware clients, Caspio Bridge WS API has two main entry points. First one uses mostly RPC-style calls and can be reached at <https://<BridgeSiteHostName>/ws/api.asmx>. The second supports both SOAP 1.1 and SOAP 1.2 versions of protocol and uses Document-style method calls exclusively. It can be reached at <https://<BridgeSiteHostName>/ws/soap12/api.asmx>. It is recommended to use second (SOAP 1.2) version of interface if your SOAP client supports SOAP 1.2 and/or did not support RPC-style calls, and to use first (RPC) version of interface in all other cases. See also note about Caspio Bridge sites in section 4.1

2.6.1 RPC-style and Document-style methods

Everywhere in this document, WS API methods are referred as either RPC-style or Document-style methods. These references make sense only for first (RPC) version of WS API. All methods of second (SOAP 1.2) version of WS API are Document-style. For developer, there is not so much difference between these two styles of SOAP calls – method-call style controls mostly the low-level XML layout of SOAP requests and responses, so this functionality is usually encapsulated in SOAP client software library or proxy class generated by this library. However, some old SOAP client software does not support complex types used in document-style calls, at the same time some new SOAP clients do not support RPC-style calls, considering them deprecated. Moreover, some SOAP clients, for example, PHP 5.2.5 SoapClient class, might have bugs that prevent usage of Document-style methods. For these reasons, methods descriptions in this document always include method call style.

2.6.2 Complex connectivity sample: Caspio Bridge WS API and Crystal Reports 2008

Crystal Reports 2008 is a widely used report generation tool from SAP BusinessObjects (<http://www.sap.com/solutions/sapbusinessobjects/sme/reporting/crystalreports/index.epx>). The following example demonstrates the steps needed to connect Crystal Reports 2008 with Caspio Bridge WS API.

1. In Crystal Reports 2008 main menu, select **File, New, Standard Report**.
2. In **Available Data Sources**, select **Create New Connection, XML and Web Services**.
3. In **XML and Web Services** dialog, choose **Use Web Service Data Source** and click **Next**.
Hint: Crystal Reports 2008 can not use RPC-style method calls, so it is mandatory to use SOAP 1.2 version of WS API in this case. From the other hand, Crystal Reports 2008 cannot use 1.2

version of SOAP protocol, so it is mandatory to use backward-compatible **CaspioBridgeAPI Soap** port of SOAP 1.2 version of WS API (see step 9 below). This port utilizes SOAP version 1.1 protocol, but use Document-style method calls instead of RPC-style.

4. Navigate your browser to <http://<BridgeSiteHostName>/ws/soap12/api.asmx?wsdl> and store response as a WSDL file on local disk.
5. Open this file by text editor and look for the following line

```
<s:complexType name="CaspioBridgeTableRow">
```

This part of WSDL file defines a XML format of the Caspio Bridge WS API **SelectData** method response. Since Caspio Bridge WS API does not know exact table or view a client is going to query, WS API defines the response as

```
<s:sequence>
  <s:any maxOccurs="unbounded" />
</s:sequence>
</s:complexType>
```

This means that CaspioBridgeTableRow may contain any number of any XML nodes that represent table or view fields. However, in order to build a Crystal Reports 2008 report based on some particular data source, caller has to know what the table it is going to query. Suppose it is a **Table1** table with 4 fields: **ID** (Autonumber), **Txt** (ShortText), **DT** (DateTime), and **Bool** (YesNo). Schema definition for this table must be as following:

```
<s:complexType name="CaspioBridgeTableRow">
  <s:sequence>
    <s:element minOccurs="0" maxOccurs="1" name="ID" type="s:int"/>
    <s:element minOccurs="0" maxOccurs="1" name="Txt" type="s:string"/>
    <s:element minOccurs="0" maxOccurs="1" name="DT" type="s:string"/>
    <s:element minOccurs="0" maxOccurs="1" name="Bool" type="s:boolean"/>
  </s:sequence>
</s:complexType>
```

Note the data type mappings.

6. Replace initial **CaspioBridgeTableRow** definition in WSDL file by custom one from above.
7. Save changed file as **WSAPI Table1.wsdl**
8. Back in Crystal Reports dialog, select **Use Local WSDL** radio button, choose **WSAPI Table1.wsdl** file as **Local WSDL file** and click **Next**
9. Select **CaspioBridgeAPI** as a **Service**, **CaspioBridgeAPI Soap** as a **Port**, and **SelectData** as a **Method**. Click **Finish** and Crystal Reports will create a new connection
10. Unfold newly created connection, select **SelectDataResponse/SelectDataResult/Row** and press **>** button.
11. Fill your login credentials, table name and other WS API method arguments. Use **"ID, Txt, DT, Bool"** (without quotes) as **FieldList** value.
12. Click **OK** and finish Report Creation Wizard.
13. Use **FieldExplorer** window to add database fields to the report.

In order to base a report on another table or view, edit original WSDL file and replace common **CaspioBridgeTableRow** definition in the way similar to one described in step 5. Save the result file under different name.

3 Programmer's Reference

This section contains detailed description of all Caspio Bridge WS API methods, types and error codes. Two types of methods are implemented in Web Services API are RPC-style and Document-style methods (except methods called via SOAP 1.2 interface – they are always Document-style). Document-style methods use custom data types and, in fact, are optional.

Note on code examples

Examples in this section are inline demonstration of some common uses of Caspio Bridge WS API. They are written in VB.Net, C#.Net and PHP5. For more comprehensive code examples in different programming languages, see Section 5.

3.1 General Information

URL of Caspio Bridge Web Service: <https://<BridgeSiteHostName>/ws/api.asmx>
 SOAP 1.2 API <https://<BridgeSiteHostName>/ws/soap12/api.asmx>
 WSDL file: <https://<BridgeSiteHostName>/ws/api.asmx?wsdl>
 SOAP 1.2 API <https://<BridgeSiteHostName>/ws/soap12/api.asmx?wsdl>

Important Note

<BridgeSiteHostName> (or bridge.caspio.net) needs to be replaced with the appropriate domain name that hosts your Caspio Bridge application. That domain name could be bridge.caspio.net, b2.caspio.com, b3.caspio.com or other. Information about WSDL URL can be found in Web Service Profile Properties dialog (see Picture 2)

There is a helper web service that allows WS API client to find out a site where particular account is hosted on. It can be reached at the address <http://www.caspio.com/rsa/rsa.asmx>. One of its methods, GetAccountWSDL, returns URL to WSDL file for particular account.

By default, Caspio Bridge WS API can be accessed only via secure (HTTPS) protocol. In order to allow access via unsecured (HTTP) connection for specific Web Service Profile, user must clear "Require SSL encryption" checkbox in Web Service Profile Properties dialog in the Caspio Bridge account. Otherwise, all calls via unsecured connections generate run-time exceptions.

All DataPage-related methods can be used only by Web service profiles with **Enable access to all DataPages** privilege enabled. If this privilege is not enabled for particular Web service Profile, all DataPage-related methods calls will generate run-time exception with code 1023.

In addition, Distance Search-related methods can be called only by Web Service Profiles from the accounts that have Distance Search add-on enabled.

3.2 Parameters

All WS API calls are stateless. Therefore, login credentials are supplied during each WS API call. There are 3 login credentials – **AccountID**, **Profile**, and **Password**.

Complete List of Caspio Bridge WS API Parameters

Parameter	Type	Description
AccountID	string	Caspio Bridge Account ID.
Profile	string	Name of the Web Service Profile in the given account.
Password	string	Password of the Web Service Profile.
ObjectName	string	Name of the Caspio Bridge table or view. The Web Service Profile must have appropriate privilege (Select, Insert, Update, Delete, Alter design) on this object.
IsView	boolean	If <i>true</i> , ObjectName is a view name, if <i>false</i> it is a table name.
FieldList	string	The SQL SELECT clause without the word SELECT. It may include constants, field names, expressions, functions, aliases, aggregates, sub queries, TOP and DISTINCT predicates. If

		empty, "*" (all fields) is used. See also IncludeCalculatedDistance .
Criteria	string	WHERE clause of a SQL statement without word WHERE. May be followed by GROUP BY and HAVING clauses.
OrderBy	string	ORDER BY clause of a SQL statement without words ORDER BY.
FieldDelimiter	string	Specifies the character to be used as a field separator. If empty, comma is used.
StringDelimiter	string	Specifies the character to be used to enclose string data. If empty, single-quote is used.
IsSchema	boolean	If set to TRUE, XML schema will be returned.
IsRaw	boolean	If set to TRUE, XML data will be in RAW mode, set to FALSE for AUTO mode.
IsElements	boolean	If set to TRUE, values are returned as XML elements, if FALSE as attributes.
IsBase64	boolean	If set to TRUE data will be Base64 encoded.
ValueList	string	Comma-separated list of values or expressions. Strings must be enclosed in single quotes. Single quotes inside strings must be represented as 2 single quotes (").
TableName	string	Name of a Caspio Bridge table.
FieldName	string	Name of a Caspio Bridge field.
Type	string	Data type of a field. Value must match a CaspioBridgeColumnType (see Types below).
Unique	boolean	Set to TRUE if new field must be unique. AutoNumber fields must have Unique always set to TRUE; File and LongText fields must have Unique always set to FALSE.
Description	string	Description of a new field.
NewFieldName	string	New name of field or empty string if name should be left unchanged.
NewType	string	New data type of field or empty if type should be left unchanged.
NewUnique	string	New Unique setting for the field, or empty if Unique should be left unchanged. Can be "0", "1", "true" or "false".
NewDescription	string	New description of field or empty if description should be left unchanged.
TableSchema	CaspioBridgeTable	Custom object that describes Caspio Bridge table (see Types below).
Latitude	double	Numeric latitude value used in proximity (distance) search
Longitude	double	Numeric longitude value used in proximity (distance) search
LatitudeField	string	Name of table field or view alias used in proximity (distance) search by coordinates
LongitudeField	string	Name of table field or view alias used in proximity (distance) search by coordinates
ProximityCriteria	string	Criteria used in proximity (distance) search. Must contain comparison operator (usually <, >, <= or >=) and distance value.
Units	int	Specifies the distance measurement unit for proximity (distance) search. 1 – feet, 2 – yards, 3 – statute miles, 4 – nautical miles, 5 – meters, 6 - kilometers
AdditionalCriteria	string	Additional criteria used in proximity (distance) search. Must start with 'AND' or 'OR'
IncludeCalculated Distance	boolean	If set to True, proximity search returns special ' <i>calculated_distance</i> ' field in addition to fields specified in FieldList . If this parameter is set to False and FieldList is empty, '*' (all fields) is assumed as FieldList . If this parameter is set to True and FieldList is empty, only ' <i>calculated_distance</i> ' field is returned.
RefObject	string	Name of the Caspio Bridge table or view used as reference data source in ProximitySearchByReference methods. The

		Web Service Profile must have Select privilege on this object.
RefsView	boolean	If <i>true</i> , RefObject is a view name, if <i>false</i> it is a table name.
SearchField	string	In ProximitySearchByReference methods, specifies the field in ObjectName table or view used in joining RefObject table or view on RefField field
RefField	string	In ProximitySearchByReference methods, specifies the field in RefObject table or view used in joining ObjectName table or view on SearchField field
RefLatitudeField	string	Name of table field or view alias used in proximity search by reference
RefLongitudeField	string	Name of table field or view alias used in proximity search by reference
Folder	string	Name of the DataPage folder, used together with DataPageName to specify particular DataPage. If empty or a slash (/) character, it represents the root folder
DataPageName	string	Name of the DataPage, used together with Folder to specify particular DataPage
AppKey	string	AppKey of the DataPage, used in most DataPage API methods to specify the DataPage
DeployMethod	string	May be one of the following strings: "IFrame", "Frame", "URL", "Link", "Embedded", "Net", "PHP", "ASP", "ASPX". Last three are for server-side deployment code. Specifies the type of deployment code returned by GetDataPageDeploymentCode method. Values are case-insensitive, may be replaced by first characters from corresponding values (I, F, U, L, E, N, P, A, and X for ASPX). For Mobile Form type of DataPages, only "URL" or "Link" deployment methods can be used.
Https	boolean	If set to True, returned deployment code includes secure https:// protocol prefixes, otherwise it uses http://. For Mobile Forms, value of this parameter is ignored and always assumed False.
HttpsOnly	boolean	If set to True, DataPage is deployed in HTTPS only mode, and cannot be viewed via insecure http:// connection. For Mobile Forms, value of this parameter is ignored and always assumed False.
ExtraInfo	boolean	If set to True, ListDataPagesRaw method returns DataPage Application type and Deployment Status information in addition to DataPage name, AppKey and folder name.
ViewXML	string	XML containing Caspio Bridge view definition. Samples of such XML can be obtained via Export View feature of Caspio Bridge.
DataPageXML	string	XML containing Caspio Bridge DataPage definition. Samples of such XML can be obtained via Export DataPage feature of Caspio Bridge.
FolderID	int	ID of the file folder. Special value "-1" represents root folder. Use ListFiles to get folder ID of a folder with particular name.
IncludeSubfolders	boolean	If set to True, ListFiles will return the whole folder(s) tree content, starting from folder with FolderID as a root. If set to False, ListFiles will return only the content of folder specified by FolderID . If Web Service Profile does not have privileges to access some files and folders, they will not be shown in ListFiles response.
FileID	int	ID of the file to be deleted or downloaded. Use ListFiles to get the file ID of a file with particular name.
FileName	string	Name of the file to be uploaded. Cannot be empty, longer than 255 characters, cannot contain any of the following characters: /+\\<>:"*? See also Overwrite .
FileData	byte array (SOAP's base64binary type)	Content of the file to be uploaded. Cannot be empty.

ContentType	string	Content-type or MIME-type of uploaded file. Caspio Bridge uses this content-type when the file is downloaded, either from file area in user interface, or via DataPages.
Overwrite	boolean	If set to True, UploadFile method overwrites any file in target folder with the same name as specified in FileName with new content. In this case Web Service profile must have Delete/Replace privilege on target file. If set to False, and if the file with the same name as specified in FileName exists in target folder, UploadFile will return error 1037 – “File or folder with such name already exists”. In any case, Web Service Profile must have "Can create objects" privilege. If target folder contains the folder with the same name as specified in FileName , error 1037 is raised regardless the value of Overwrite .

Note on PK_ID.

In addition to the user-defined fields, all Caspio Bridge tables and views have a field called PK_ID. Its data type is long integer in tables and string containing the list of tables' PK_IDs in views. This field is created and maintained automatically by Caspio Bridge. You cannot delete or alter this field or change its value neither via WS API nor via Caspio Bridge interface.

This field does not appear in results returned by **GetTableDesign** and **GetTableDesignRaw** methods. However its values can be obtained via **SelectData**, **SelectDataRaw**, and **SelectDataXML** method.

3.3 RPC-style Methods

These methods use RPC-style web service calls and do not use custom data types. They can be called from virtually any SOAP-enabled client.

3.3.1 SelectDataRaw

Selects data from a Caspio Bridge table or view and returns it in raw format using simple types.

Parameters:

AccountID, Profile, Password, ObjectName, IsView, FieldList, Criteria, OrderBy, FieldDelimiter, StringDelimiter

Return Value: one-dimensional array of strings. Each string contains one result row. Null values are represented by NULL. Fields are separated by **FieldDelimiter**, strings are enclosed in **StringDelimiter**. If query returned no results, an array with one empty string is returned.

Example (VB):

```
' return 10 largest orders from Caspio Bridge table 'orders'
```

```
Dim aRows() As String
```

```
aRows=aobjWS.SelectDataRaw(strAccountID, strProfileName, strPassword, "Orders", false, _
    "TOP 10 OrderID, OrderTotal as Total", _
    "OrderTotal IS NOT NULL", _
    "OrderTotal DESC", _
    "", "")
```

Example (C#):

```
// return 10 largest orders from Caspio Bridge table 'orders'
```

```
string[] aRows;
```

```
aRows=aobjWS.SelectDataRaw(strAccountID, strProfileName, strPassword,
    "Orders", false,
    "TOP 10 OrderID, OrderTotal as Total",
```

```
"OrderTotal IS NOT NULL",
"OrderTotal DESC",
"", "");
```

Example (PHP):

```
// return 10 largest orders from Caspio Bridge table 'orders'

print_r(SoapClient->SelectDataRow($AccountID, $ProfileName, $Password,
"Orders", false, "TOP 10 OrderID, OrderTotal as Total",
"OrderTotal IS NOT NULL",
"OrderTotal DESC",
"", ""));
```

3.3.2 SelectDataXML

Selects data from a Caspio Bridge table or view and returns it as string containing data in SQL Server's XML format.

Parameters:

AccountID, Profile, Password, ObjectName, IsView, FieldList, Criteria, OrderBy, IsSchema, IsRaw, IsElements, IsBase64

Return Value: string containing XML response from the server. In order to be valid, XML is wrapped in root <CaspioBridgeResponse> tag.

Example (with IsRaw=True)

VB:

```
' return largest order from Caspio Bridge table 'orders' using different format modifiers
```

```
Dim strResult As String
```

```
strResult=aobjWS.SelectDataXML(strAccountID, strProfileName, strPassword, _
"Orders", false, _
"TOP 1 OrderID, Str(OrderTotal,6,2) as Total", _
"OrderTotal IS NOT NULL", _
"OrderTotal DESC", _
false, true, false, false)
```

C#:

```
/* return largest order from Caspio Bridge table 'orders' using different format modifiers */
```

```
string strResult;
```

```
strResult=aobjWS.SelectDataXML(strAccountID, strProfileName, strPassword,
"Orders", false,
"TOP 1 OrderID, Str(OrderTotal,6,2) as Total",
"OrderTotal IS NOT NULL",
"OrderTotal DESC",
false, true, false, false);
```

PHP:

```
/* return largest order from Caspio Bridge table 'orders' using different format modifiers */
```

```
$result=SoapClient->SelectDataXML($AccountID, $ProfileName, $Password,
"Orders", false,
"TOP 1 OrderID, Str(OrderTotal,6,2) as Total",
"OrderTotal IS NOT NULL",
"OrderTotal DESC",
false, true, false, false);
```

Results:

```
<CaspioBridgeResponse>
  <row OrderID="1" Total="100.00" />
</CaspioBridgeResponse>
```

Example (with IsRaw=False)

```
VB:
strResult=aobjWS.SelectDataXML(strAccountID, strProfileName, strPassword, _
  "Orders", false, _
    "TOP 1 OrderID, Str(OrderTotal,6,2) as Total", _
  "OrderTotal IS NOT NULL", _
  "OrderTotal DESC", _
  false, false, false, false)
```

```
C#:
strResult=aobjWS.SelectDataXML(strAccountID, strProfileName, strPassword,
  "Orders", false,
    "TOP 1 OrderID, Str(OrderTotal,6,2) as Total",
  "OrderTotal IS NOT NULL",
  "OrderTotal DESC",
  false, false, false, false);
```

```
PHP:
$result=SoapClient->SelectDataXML($AccountID, $ProfileName, $Password,
  "Orders", false,
    "TOP 1 OrderID, Str(OrderTotal,6,2) as Total",
  "OrderTotal IS NOT NULL",
  "OrderTotal DESC",
  false, false, false, false);
```

Results:

```
<CaspioBridgeResponse>
<orders OrderID="1" Total="100.00" />
</CaspioBridgeResponse>
```

Example (with IsElements=True)

```
VB:
strResult=aobjWS.SelectDataXML(strAccountID, strProfileName, strPassword, _
  "Orders", false, _
    "TOP 1 OrderID, Str(OrderTotal,6,2) as Total", _
  "OrderTotal IS NOT NULL", _
  "OrderTotal DESC", _
  false, false, true, false)
```

```
C#:
strResult=aobjWS.SelectDataXML(strAccountID, strProfileName, strPassword,
  "Orders", false,
    "TOP 1 OrderID, Str(OrderTotal,6,2) as Total",
  "OrderTotal IS NOT NULL",
  "OrderTotal DESC",
  false, false, true, false);
```

```
PHP:
$result=SoapClient->SelectDataXML($AccountID, $ProfileName, $Password,
  "Orders", false,
    "TOP 1 OrderID, Str(OrderTotal,6,2) as Total",
  "OrderTotal IS NOT NULL",
  "OrderTotal DESC",
  false, false, true, false);
```

Results:

```
<CaspioBridgeResponse>
<orders>
```

```

<OrderID>1</OrderID>
<Total>100.00</Total>
</orders>
</CaspioBridgeResponse>

```

Example (with IsSchema=True, IsRaw=False, IsElement=True, IsBase64=True)

```

VB:
strResult=aobjWS.SelectDataXML(strAccountID, strProfileName, strPassword, _
  "Orders", false, _
    "TOP 1 OrderID, Str(OrderTotal,6,2) as Total", _
  "OrderTotal IS NOT NULL", _
  "OrderTotal DESC", _
  true, false, true, true)

```

```

C#:
strResult=aobjWS.SelectDataXML(strAccountID, strProfileName, strPassword,
  "Orders", false,
    "TOP 1 OrderID, Str(OrderTotal,6,2) as Total",
  "OrderTotal IS NOT NULL",
  "OrderTotal DESC",
  true, false, true, true);

```

```

PHP:
$result=SoapClient->SelectDataXML($AccountID, $ProfileName, $Password,
  "Orders", false,
    "TOP 1 OrderID, Str(OrderTotal,6,2) as Total",
  "OrderTotal IS NOT NULL",
  "OrderTotal DESC",
  true, false, true, true);

```

Results:

```

<CaspioBridgeResponse>
<Schema name="Schema1" xmlns="urn:schemas-microsoft-com:xml-data"
xmlns:dt="urn:schemas-microsoft-com:datatypes">
  <ElementType name="orders" content="eltOnly" model="closed" order="many">
    <element type="OrderID" />
    <element type="Total" />
  </ElementType>
  <ElementType name="OrderID" content="textOnly" model="closed" dt:type="i4"/>
  <ElementType name="Total" content="textOnly" model="closed" dt:type="string"/>
</Schema>
<orders xmlns="x-schema:#Schema1">
  <OrderID>1</OrderID>
  <Total>100.00</Total>
</orders>
</CaspioBridgeResponse>

```

3.3.3 InsertData

Insert one record in a Caspio Bridge table or view.

Parameters:

AccountID, Profile, Password, ObjectName, IsView, FieldList, ValueList

Return Value: Integer. Number of rows affected.

```

Example (VB):
'insert into table Orders
'OrderDate := current date
'OrderDescription := string "some desc"
'Shipped flag :=0

```

```
aobjWS.InsertData(strAccountID, strProfileName, strPassword, _
  "Orders", false, _
    "OrderDate, OrderDescription, Shipped", _
  "GetDate(),'some desc',0")
```

Example (C#):

```
//insert into table Orders
//OrderDate := current date
//OrderDescription := string "some desc"
//Shipped flag :=0
aobjWS.InsertData(strAccountID, strProfileName, strPassword,
  "Orders", false,
    "OrderDate, OrderDescription, Shipped",
  "GetDate(),'some desc',0");
```

Example (PHP):

```
//insert into table Orders
//OrderDate := current date
//OrderDescription := string "some desc"
//Shipped flag :=0

SoapClient->InsertData($AccountID, $ProfileName, $Password,
  "Orders", false,
    "OrderDate, OrderDescription, Shipped",
  "GetDate(),'some desc',0");
```

3.3.4 UpdateData

Update data in a Caspio Bridge table or view.

Parameters:

AccountID, Profile, Password, ObjectName, IsView, FieldList, ValueList, Criteria

Return Value: Integer. Number of rows affected.

Example (VB):

```
Dim lAffected As Long
lAffected=aobjWS.UpdateData(strAccountID, strProfileName, strPassword, _
  "Orders", false, _
    "OrderDescription, Shipped", _
  "'was not shipped yet',0", _
  "Shipped<>0 and datediff(dy,OrderDate,getdate())<=3")
```

Example (C#):

```
int iAffected;
iAffected=aobjWS.UpdateData(strAccountID, strProfileName, strPassword,
  "Orders", false,
    "OrderDescription, Shipped",
  "'was not shipped yet',0",
  "Shipped<>0 and datediff(dy,OrderDate,getdate())<=3");
```

Example (PHP):

```
$Affected=SoapClient->UpdateData($AccountID, $ProfileName, $Password,
  "Orders", false,
    "OrderDescription, Shipped",
  "'was not shipped yet',0",
  "Shipped<>0 and datediff(dy,OrderDate,getdate())<=3");
```

3.3.5 DeleteData

Delete data from a Caspio Bridge table or view.

Parameters:

AccountID, Profile, Password, ObjectName, IsView, Criteria

Return Value: Integer. Number of rows affected.

Example (VB):

```
Dim lAffected As Long
lAffected=aobjWS.DeleteData(strAccountID, strProfileName, strPassword, _
    "Orders", false, _
    "OrderTotal is null or OrderTotal=0")
```

Example (C#):

```
int iAffected;
iAffected=aobjWS.DeleteData(strAccountID, strProfileName, strPassword,
    "Orders", false,
    "OrderTotal is null or OrderTotal=0");
```

Example (PHP):

```
$Affected=SoapClient->DeleteData($AccountID, $ProfileName, $Password,
    "Orders", false,
    "OrderTotal is null or OrderTotal=0");
```

3.3.6 CreateEmptyTable

Create a Caspio Bridge table without fields.

Parameters:

AccountID, Profile, Password, TableName

Return Value: none.

Example (VB):

```
aobjWS.CreateEmptyTable(strAccountID, strProfileName, strPassword, "Orders")
```

Example (C#):

```
aobjWS.CreateEmptyTable(strAccountID, strProfileName, strPassword, "Orders");
```

Example (PHP):

```
SoapClient->CreateEmptyTable($AccountID, $ProfileName, $Password, "Orders");
```

3.3.7 AddTableField

Add a new field to an existing Caspio Bridge table.

Parameters:

AccountID, Profile, Password, TableName, FieldName, Type, Unique, Description

Return Value: none.

Example (VB):

```
aobjWS.AddTableField(strAccountID, strProfileName, strPassword, "Orders", _
    "OrderID", "AutoNumber", true, "identity field for Orders")
```

Example (C#):

```
aobjWS.AddTableField(strAccountID, strProfileName, strPassword, "Orders",
    "OrderID", "AutoNumber", true, "identity field for Orders");
```

Example (PHP):

```
SoapClient->AddTableField($AccountID, $ProfileName, $Password, "Orders",
    "OrderID", "AutoNumber", true, "identity field for Orders");
```

3.3.8 DropTableField

Delete a field from a Caspio Bridge table.

Parameters:

AccountID, Profile, Password, TableName, FieldName

Return Value: none.

Example (VB):

```
aobjWS.DropTableField(strAccountID, strProfileName, strPassword, "Orders", _  
"OrderID")
```

Example (C#):

```
aobjWS.DropTableField(strAccountID, strProfileName, strPassword, "Orders",  
"OrderID");
```

Example (PHP):

```
SoapClient->DropTableField($AccountID, $ProfileName, $Password, "Orders",  
"OrderID");
```

3.3.9 AlterTableField

Rename, change type or other properties of a field in a Caspio Bridge table.

Parameters:

**AccountID, Profile, Password, TableName, FieldName, NewFieldName, NewType,
NewUnique, NewDescription**

Return Value: none.

Example (VB):

```
aobjWS.AlterTableField(strAccountID, strProfileName, strPassword, "Orders", _  
"OrderID", "", "Number", "false", "")
```

Example (C#):

```
aobjWS.AlterTableField(strAccountID, strProfileName, strPassword, "Orders",  
"OrderID", "", "Number", "false", "");
```

Example (PHP):

```
SoapClient->AlterTableField($AccountID, $ProfileName, $Password, "Orders",  
"OrderID", "", "Number", "false", "");
```

3.3.10 GetTableDesignRaw

Describe an existing Caspio Bridge table using simple types.

Parameters:

AccountID, Profile, Password, TableName

Return Value: Array of string. Each element contains information about one table field in format "FieldName, Type, Unique, Description". If table has no fields, array with one empty string element is returned.

Example (VB):

```
Dim aColumns() As String, tempArray() As String  
aColumns=aobjWS.GetTableDesignRaw(strAccountID, strProfileName, strPassword, "Orders")
```

```
'show the name and type of first column
tempArray=Split(aColumns(0),",")

Msgbox (tempArray(0) & ":" & tempArray(1))
```

Example (C#):

```
string[] aColumns;
string[] tempArray;
aColumns=aobjWS.GetTableDesignRaw(strAccountID, strProfileName, strPassword, "Orders");

//show the name and type of first column
char[] charSeparators = new char[] {' ',' '};
tempArray=aColumns[0].Split(charSeparators);

Msgbox (tempArray[0] + ":" + tempArray[1]);
```

Example (PHP):

```
Print_r(SoapClient->GetTableDesignRaw($AccountID, $ProfileName, $Password, "Orders"));
```

3.3.11 ProximitySearchByCoordinates

Performs proximity search by coordinates and returns string in XML format.

Parameters:

AccountID, Profile, Password, ObjectName, IsView, Latitude, Longitude, LatitudeField, LongitudeField, ProximityCriteria, Units, FieldList, AdditionalCriteria, OrderBy, IncludeCalculatedDistance, IsSchema, IsRaw, IsElements, IsBase64

Return Value: string containing XML response from the server. In order to be valid, XML is wrapped in root *<CaspioBridgeResponse>* tag. See also **SelectDataXML**.

Example (VB):

```
'Formal definition of Proximity Search Measurement Units enumeration
'Mentioned here for sample purpose only
'Simple integer type is actually used in ProximitySearch methods

Public Enum ProximitySearchUnits
    <XmlEnum("Feet")> PS_UNITS_FEET = 1
    <XmlEnum("Yards")> PS_UNITS_YARDS = 2
    <XmlEnum("StatuteMiles")> PS_UNITS_MILES = 3
    <XmlEnum("NauticalMiles")> PS_UNITS_NAUT_MILES = 4
    <XmlEnum("Meters")> PS_UNITS_METERS = 5
    <XmlEnum("Kilometers")> PS_UNITS_KILOMETERS = 6
End Enum

'
' GasStations table has 3 fields: Address, Latitude and Longitude
' The following call returns distances to and addresses of all gas stations
' closer than 100 miles from point with coordinates:latitude=74.5, longitude=-67.6

Dim strResult as String

strResult=aobjWS.ProximitySearchByCoordinates(strAccountID,strProfileName,strPassword, _
    "GasStations", false, 74.5, -67.6, "Latitude", "Longitude", _
    " < 100 ", 3, "Address", " and Address IS NOT NULL", " Address ASC ", _
    True, False, False, True, True)
```

Example (C#):

```
//Formal definition of Proximity Search Measurement Units enumeration
//Mentioned here for sample purpose only
//Simple integer type is actually used in ProximitySearch methods
```

```
Public Enum ProximitySearchUnits
{
    [XmlEnum(Name="Feet")] PS_UNITS_FEET = 1,
    [XmlEnum(Name="Yards")] PS_UNITS_YARDS = 2,
    [XmlEnum(Name="StatuteMiles")] PS_UNITS_MILES = 3,
    [XmlEnum(Name="NauticalMiles")] PS_UNITS_NAUT_MILES = 4,
    [XmlEnum(Name="Meters")] PS_UNITS_METERS = 5,
    [XmlEnum(Name="Kilometers")] PS_UNITS_KILOMETERS = 6
}
```

```
// GasStations table has 3 fields: Address, Latitude and Longitude
// The following call returns distances to and addresses of all gas stations
// closer than 100 miles from point with coordinates:latitude=74.5, longitude=-67.6
```

```
string strResult;
```

```
strResult=aobjWS.ProximitySearchByCoordinates(strAccountID,strProfileName,strPassword,
    "GasStations", false, 74.5, -67.6, "Latitude", "Longitude",
    " < 100 ", 3, "Address", " and Address IS NOT NULL", " Address ASC ",
    True, False, False, True, True);
```

Example (PHP):

```
// GasStations table has 3 fields: Address, Latitude and Longitude
// The following call returns distances to and addresses of all gas stations
// closer than 100 miles from point with coordinates:latitude=74.5, longitude=-67.6
```

```
$Result=SoapClient->aobjWS.ProximitySearchByCoordinates($AccountID, $ProfileName,
$Password, "GasStations", false, 74.5, -67.6, "Latitude", "Longitude",
    " < 100 ", 3, "Address", " and Address IS NOT NULL", " Address ASC ",
    True, False, False, True, True);
```

3.3.12 ProximitySearchByCoordinatesRaw

Performs proximity search by coordinates and returns result in raw format.

Parameters:

AccountID, Profile, Password, ObjectName, IsView, Latitude, Longitude, LatitudeField, LongitudeField, ProximityCriteria, Units, FieldList, AdditionalCriteria, OrderBy, IncludeCalculatedDistance, FieldDelimiter, StringDelimiter

Return Value: See **SelectDataRaw**.

Example (VB):

```
' GasStations table has 3 fields: Address, Latitude and Longitude
' The following call returns distances to and addresses of all gas stations
' closer than 100 miles from point with coordinates:latitude=74.5, longitude=-67.6
```

```
Dim aRows() As String
```

```
aRows=aobjWS.ProximitySearchByCoordinatesRaw( _
    strAccountID,strProfileName,strPassword, _
    "GasStations", false, 74.5, -67.6, "Latitude", "Longitude", _
    " < 100 ", 3, "Address", " and Address IS NOT NULL", " Address ASC ", _
    True, "","")
```

Example (C#):

```
// GasStations table has 3 fields: Address, Latitude and Longitude
// The following call returns distances to and addresses of all gas stations
// closer than 100 miles from point with coordinates:latitude=74.5, longitude=-67.6
```

```
string[] aRows;
```

```
aRows=aobjWS.ProximitySearchByCoordinatesRaw(strAccountID,strProfileName,strPassword,
    "GasStations", false, 74.5, -67.6, "Latitude", "Longitude",
    " < 100 ", 3, "Address", " and Address IS NOT NULL", " Address ASC ",
    True, ",","");
```

Example (PHP):

```
// GasStations table has 3 fields: Address, Latitude and Longitude
// The following call returns distances to and addresses of all gas stations
// closer than 100 miles from point with coordinates:latitude=74.5, longitude=-67.6
```

```
$result=SoapClient->.ProximitySearchByCoordinatesRaw($AccountID, $ProfileName, $Password,
    "GasStations", false, 74.5, -67.6, "Latitude", "Longitude",
    " < 100 ", 3, "Address", " and Address IS NOT NULL", " Address ASC ",
    True, ",","");
```

3.3.13 ProximitySearchByReference

Performs proximity search by reference and returns string in XML format.

Parameters:

AccountID, Profile, Password, ObjectName, IsView, Latitude, Longitude, SearchField, RefObject, RefIsView, RefField, RefLatitudeField, RefLongitudeField, ProximityCriteria, Units, FieldList, AdditionalCriteria, OrderBy, IncludeCalculatedDistance, IsSchema, IsRaw, IsElements, IsBase64

Return Value: string containing XML response from the server. In order to be valid, XML is wrapped in root *<CaspioBridgeResponse>* tag. See also **SelectDataXML**.

Example (VB):

```
' GasStations table has 2 fields: Address and ZIP
' ZIPs table has 3 fields: ZIPcode, Latitude and Longitude
' The following call returns distances to and addresses of all gas stations
' closer than 50 kilometers from point with coordinates:latitude=74.5, longitude=-67.6
```

```
Dim strResult as String
```

```
strResult=aobjWS.ProximitySearchByReference(strAccountID,strProfileName,strPassword, _
    "GasStations", false, 74.5, -67.6, "ZIP", "ZIPs", false, "ZIPcode", _
    "Latitude", "Longitude", " < 50 ", 6, "Address", _
    " and Address IS NOT NULL", "", True, False, False, True, True)
```

Example (C#):

```
// GasStations table has 2 fields: Address and ZIP
// ZIPs table has 3 fields: ZIPcode, Latitude and Longitude
// The following call returns distances to and addresses of all gas stations
// closer than 50 kilometers from point with coordinates:latitude=74.5, longitude=-67.6
```

```
string strResult;
```

```
strResult=aobjWS.ProximitySearchByReference(strAccountID,strProfileName,strPassword,
    "GasStations", false, 74.5, -67.6, "ZIP", "ZIPs", false, "ZIPcode",
    "Latitude", "Longitude", " < 50 ", 6, "Address",
```

```
" and Address IS NOT NULL", "", True, False, False, True, True);
```

Example (PHP):

```
// GasStations table has 2 fields: Address and ZIP
// ZIPs table has 3 fields: ZIPcode, Latitude and Longitude
// The following call returns distances to and addresses of all gas stations
// closer than 50 kilometers from point with coordinates:latitude=74.5, longitude=-67.6

$result=SoapClient->ProximitySearchByReference($AccountID,$ProfileName,$Password,
    "GasStations", false, 74.5, -67.6, "ZIP", "ZIPs", false, "ZIPcode",
"Latitude", "Longitude", " < 50 ", 6, "Address",
" and Address IS NOT NULL", "", True, False, False, True, True);
```

3.3.14 ProximitySearchByReferenceRaw

Performs proximity search by reference and returns result in raw format.

Parameters:

AccountID, Profile, Password, ObjectName, IsView, Latitude, Longitude, SearchField, RefObject, RefIsView, RefField, RefLatitudeField, RefLongitudeField, ProximityCriteria, Units, FieldList, AdditionalCriteria, OrderBy, IncludeCalculatedDistance, FieldDelimiter, StringDelimiter

Return Value: See **SelectDataRaw**.

Example (VB):

```
' GasStations table has 2 fields: Address and ZIP
' ZIPs table has 3 fields: ZIPcode, Latitude and Longitude
' The following call returns distances to and addresses of all gas stations
' closer than 50 kilometers from point with coordinates:latitude=74.5, longitude=-67.6

Dim aRows() As String

aRows=aobjWS.ProximitySearchByReferenceRaw(strAccountID,strProfileName,strPassword, _
    "GasStations", false, 74.5, -67.6, "ZIP", "ZIPs", false, "ZIPcode", _
"Latitude", "Longitude", " < 50 ", 6, "Address", _
" and Address IS NOT NULL", "", "", "")
```

Example (C#):

```
// GasStations table has 2 fields: Address and ZIP
// ZIPs table has 3 fields: ZIPcode, Latitude and Longitude
// The following call returns distances to and addresses of all gas stations
// closer than 50 kilometers from point with coordinates:latitude=74.5, longitude=-67.6

string[] aRows;

aRows=aobjWS.ProximitySearchByReferenceRaw(strAccountID,strProfileName,strPassword,
    "GasStations", false, 74.5, -67.6, "ZIP", "ZIPs", false, "ZIPcode",
"Latitude", "Longitude", " < 50 ", 6, "Address",
" and Address IS NOT NULL", "", "", "");
```

Example (PHP):

```
// GasStations table has 2 fields: Address and ZIP
// ZIPs table has 3 fields: ZIPcode, Latitude and Longitude
// The following call returns distances to and addresses of all gas stations
// closer than 50 kilometers from point with coordinates:latitude=74.5, longitude=-67.6
$result=SoapClient->ProximitySearchByReferenceRaw($AccountID, $ProfileName, $Password,
    "GasStations", false, 74.5, -67.6, "ZIP", "ZIPs", false, "ZIPcode",
```

```
"Latitude", "Longitude", " < 50 ", 6, "Address",  
" and Address IS NOT NULL", "", "", "");
```

3.3.15 GetDataPageAppKey

Returns AppKey of DataPage specified by DataPageName and Folder parameters.

Parameters:

AccountID, Profile, Password, Folder, DataPageName

Return Value: string AppKey.

Example (VB):

```
Dim strAppKey As String  
  
strAppKey=aobjWS.GetDataPageAppKey(strAccountID,strProfileName,strPassword, _  
"MyFolder", "MyDataPage")
```

Example (C#):

```
string strAppKey;  
  
strAppKey=aobjWS.GetDataPageAppKey(strAccountID,strProfileName,strPassword,  
"MyFolder", "MyDataPage");
```

Example (PHP):

```
$AppKey=SoapClient->GetDataPageAppKey($AccountID,$ProfileName,$Password,  
"MyFolder", "MyDataPage");
```

3.3.16 GetDataPageDeploymentStatus

Returns flag that indicates deployment status of DataPage specified by AppKey.

Parameters:

AccountID, Profile, Password, AppKey

Return Value: byte. Possible values are 0 (DataPage is not deployed), 1 (DataPage is deployed) or 2 (DataPage is deployed as "HTTPS only")

Example (VB):

```
Dim bDeployStatus As Byte  
  
bDeployStatus=aobjWS.GetDataPageDeploymentStatus(strAccountID, strProfileName, _  
strPassword, strAppKey)
```

Example (C#):

```
byte bDeployStatus;  
  
bDeployStatus=aobjWS.GetDataPageDeploymentStatus(strAccountID, strProfileName,  
strPassword, strAppKey);
```

Example (PHP):

```
$DeployStatus=SoapClient->GetDataPageDeploymentStatus($AccountID, $ProfileName,  
$Password, $AppKey);
```

3.3.17 GetDataPageDeploymentCode

Returns deployment code for DataPage specified by AppKey parameter. Deployment code returned depends on chosen deployment model specified by DeployMethod and Https parameters.

Parameters:

AccountID, Profile, Password, AppKey, DeployMethod, Https

Return Value: HTML string contains DataPage deployment code

Example (VB):

```
Dim strDeployCode As String

strDeployCode=aobjWS.GetDataPageDeploymentCode(strAccountID, strProfileName, _
strPassword, strAppKey, "URL", true)
```

Example (C#):

```
string strDeployCode;

strDeployCode=aobjWS.GetDataPageDeploymentCode(strAccountID, strProfileName,
strPassword, strAppKey, "URL", true);
```

Example (PHP):

```
$DeployCode=SoapClient->GetDataPageDeploymentCode($AccountID, $ProfileName,
$Password, $AppKey, "URL", true);
```

3.3.18 DeployDataPage

Deploys DataPage specified by AppKey parameter.

Parameters:

AccountID, Profile, Password, AppKey, HttpsOnly

Return Value: none

Example (VB):

```
'deploy DataPage
aobjWS.DeployDataPage(strAccountID, strProfileName, strPassword, strAppKey, false)
```

Example (C#):

```
// deploy DataPage
aobjWS.DeployDataPage(strAccountID, strProfileName, strPassword, strAppKey, false);
```

Example (PHP):

```
// deploy DataPage
SoapClinet->DeployDataPage($AccountID, $ProfileName, $Password, $AppKey, false);
```

3.3.19 UndeployDataPage

Un-deploys DataPage specified by AppKey parameter.

Parameters:

AccountID, Profile, Password, AppKey

Return Value: none

Example (VB):

```
'undeploy DataPage  
aobjWS.UndeployDataPage(strAccountID, strProfileName, strPassword, strAppKey)
```

Example (C#):

```
//undeploy DataPage  
aobjWS.UndeployDataPage(strAccountID, strProfileName, strPassword, strAppKey);
```

Example (PHP):

```
//undeploy DataPage  
SoapClient->UndeployDataPage($AccountID, $ProfileName, $Password, $AppKey);
```

3.3.20 ListDataPagesRaw

Returns information about all account DataPages using simple types.

Parameters:

AccountID, Profile, Password, ExtraInfo

Return Value: array of strings. Each string contains comma-separated list of quoted entries in the following order: "DataPage Name","AppKey","Folder Name". Root folder is represented by slash ("/") character. If **ExtraInfo** is set to true, array element strings in addition contain DataPage application type name and deployment status. See description of DataPageAppTypes and DataPageDeploymentStatus enumeration types below, in **Types** section.

If there are no DataPage in account, this method returns one element array with empty string as the only array element.

Example (VB):

```
'list all DataPages  
  
Dim aDPS() as String  
  
aDPS=aobjWS.ListDataPagesRaw(strAccountID, strProfileName, strPassword, false)
```

Example (C#):

```
//list all DataPages  
  
string[] aDPS;  
aDPS=aobjWS.ListDataPagesRaw(strAccountID, strProfileName, strPassword, false);
```

Example (PHP):

```
//list all DataPages  
  
$DPS=SoapClient->ListDataPagesRaw($AccountID, $ProfileName, $Password, false);
```

3.3.21 CreateDataPage

This method was introduced in WS API 6.1

Creates new DataPage in specified folder. Does not replace existing DataPage.

Parameters:

AccountID, Profile, Password, Folder, DataPageXML

Return Value: string. AppKey of newly created DataPage.

Example (VB):

```
Dim strAppKey As String

strAppKey=aobjWS.CreateDataPage(strAccountID, strProfileName, strPassword, "", _
"<Application Type=""WebPage"" AppKey="" "" Name=""SampleDataPage"" & _
" HomePageName=""SampleDataPage""><HTMLWebPage Name=""SampleDataPage"" & _
" PageName=""SampleDataPage""><![CDATA[ Sample HTML here]]></HTMLWebPage></Application>")
```

Example (C#):

```
string strAppKey;

strAppKey=aobjWS.CreateDataPage(strAccountID, strProfileName, strPassword, "",
"<Application Type=\"WebPage\" AppKey=\"\" Name=\"SampleDataPage\"\" +
" HomePageName=\"SampleDataPage\"><HTMLWebPage Name=\"SampleDataPage\"\" +
" PageName=\"SampleDataPage\">"+
"<![CDATA[ Sample HTML here]]></HTMLWebPage></Application>");
```

Example (PHP):

```
$AppKey=SoapClient->CreateDataPage($AccountID, $ProfileName, $Password, "",
"<Application Type=\"WebPage\" AppKey=\"\" Name=\"SampleDataPage\"\" .
" HomePageName=\"SampleDataPage\"><HTMLWebPage Name=\"SampleDataPage\"\" .
" PageName=\"SampleDataPage\">\" .
"<![CDATA[ Sample HTML here]]></HTMLWebPage></Application>");
```

3.3.22 CreateView

This method was introduced in WS API 6.1

Creates new view. Does not replace existing view.

Parameters:

AccountID, Profile, Password, ViewXML

Return Value: none.

Example (VB):

```
aobjWS.CreateView(strAccountID, strProfileName, strPassword, _
"<View Name=""SampleView""><Table Name=""Table1"">" & _
"<Field Name=""f1"" Alias=""Table1_f1""></Table></View>")
```

Example (C#):

```
aobjWS.CreateView(strAccountID, strProfileName, strPassword,
"<View Name=\"SampleView\"><Table Name=\"Table1\">\" +
"<Field Name=\"f1\" Alias=\"Table1_f1\"></Table></View>");
```

Example (PHP):

```
SoapClient->CreateView($AccountID, $ProfileName, $Password,
"<View Name=\"SampleView\"><Table Name=\"Table1\">\" .
"<Field Name=\"f1\" Alias=\"Table1_f1\"></Table></View>");
```

3.3.23 ListFiles

This method was introduced in WS API 6.1

Returns XML string containing information about files and folders in folder specified by FolderID. Result lists only the files, which Web Service Profile has access to.

Parameters:

AccountID, Profile, Password, FolderID, IncludeSubfolders

Return Value: string containing XML. Example

```
<CaspioBridgeFileResponse>
  <Folder Name="/" ID="-1">
    <Folder Name="Empty" ID="1000"/>
    <Folder Name="FolderWithFiles" ID="1001">
      <File>
        <ID>2000</ID>
        <Name>SomeFileInSubfolder.txt</Name>
        <ContentType>text/plain</ContentType>
        <Size>1024</Size>
        <DateCreated>1/14/2001 12:00:00 AM</DateCreated>
        <Type>Text document</Type>
      </File>
    </Folder>
  <File>
    <ID>1000</ID>
    <Name>SomeFileInRootFolder.txt</Name>
    <ContentType>text/plain</ContentType>
    <Size>1024</Size>
    <DateCreated>1/14/2001 12:00:00 AM</DateCreated>
    <Type>Text document</Type>
  </File>
</Folder>
</CaspioBridgeFileResponse>
```

Example (VB):

```
Dim strXML as String

'get list of all files and folders
strXML=aobjWS.ListFiles(strAccountID, strProfileName, strPassword, -1, true)
```

Example (C#):

```
string strXML;

// get list of all files and folders
strXML=aobjWS.ListFiles(strAccountID, strProfileName, strPassword, -1, true);
```

Example (PHP):

```
// get list of all files and folders
$xml=SoapClient->ListFiles($AccountID, $ProfileName, $Password, -1, true);
```

3.3.24 UploadFile

This method was introduced in WS API 6.1

Uploads a new file to the file area of Caspio Bridge account, into the folder specified by FolderID. New file is assigned given FileName and ContentType. If file with given name already exists, behavior is specified by Overwrite.

Parameters:

AccountID, Profile, Password, FileName, FileData, ContentType, FolderID, Overwrite

Return Value: none.

Example (VB):

```
Dim fileContent() as Byte
'fill fileContent by file data
'... not shown here
'upload to root folder
aobjWS.UploadFile(strAccountID, strProfileName, strPassword, "file.txt", fileContent, _
"text/plain", -1, false)
```

Example (C#):

```
byte[] fileContent;
//fill fileContent by file data
//... not shown here
//upload to root folder
aobjWS.UploadFile(strAccountID, strProfileName, strPassword, "file.txt", fileContent,
"text/plain", -1, false);
```

Example (PHP):

```
//fill $fileContent by file data
//... not shown here
//upload to root folder
SoapClient->UploadFile($AccountID, $ProfileName, $Password, "file.txt", $fileContent,
"text/plain", -1, false);
```

3.3.25 DownloadFile

This method was introduced in WS API 6.1

Returns a file content.

Parameters:

AccountID, Profile, Password, FileID

Return Value: array of bytes.

Example (VB):

```
Dim aFile() as Byte
aFile=aobjWS.DownloadFile(strAccountID, strProfileName, strPassword, iFileID)
```

Example (C#):

```
byte[] aFile;
aFile=aobjWS.DownloadFile(strAccountID, strProfileName, strPassword, iFileID);
```

Example (PHP):

```
$fileContent=SoapClient->DownloadFile($AccountID, $ProfileName, $Password, $FileID);
```

3.3.26 DeleteFile

This method was introduced in WS API 6.1

Deletes existing file.

Parameters:

AccountID, Profile, Password, FileID

Return Value: none.

Example (VB):

```
aobjWS.DeleteFile(strAccountID, strProfileName, strPassword, iFileID)
```

Example (C#):

```
aobjWS.DeleteFile(strAccountID, strProfileName, strPassword, iFileID);
```

Example (PHP):

```
SoapClient->DeleteFile($AccountID, $ProfileName, $Password, $FileID);
```

3.4 Document-style Methods

These methods use Document-style web service calls and custom data types. They can be called only from SOAP clients that support document-style calls and custom data types.

3.4.1 SelectData

This method was introduced in WS API 6.1

Selects data from a Caspio Bridge table or view and returns it as dataset-like CaspioBridgeDataResponse complex type.

Parameters:

AccountID, Profile, Password, ObjectName, IsView, FieldList, Criteria, OrderBy

Return Value: *CaspioBridgeDataResponse*, defined as array of *CaspioBridgeTableRow*.

Example (VB):

```
' definition of classes used
<System.Xml.Serialization.XmlTypeAttribute( _
Namespace="http://bridge.caspio.net/ws/literalTypes")>
  Public Class CaspioBridgeTableField
    <System.Xml.Serialization.XmlAnyElementAttribute()> _
    public System.Xml.XmlElement() Any;
  End Class

' return 10 largest orders from Caspio Bridge table 'orders'

Dim aRows() As CaspioBridgeTableRow

aRows=aobjWS.SelectDataRow(strAccountID, strProfileName, strPassword, "Orders", false, _
  "TOP 10 OrderID, OrderTotal as Total", _
  "OrderTotal IS NOT NULL", _
  "OrderTotal DESC")
```

Example (C#):

```
// definition of class used
[System.Xml.Serialization.XmlTypeAttribute(Namespace="http://bridge.caspio.net/ws")]
public class CaspioBridgeTableRow {
    [System.Xml.Serialization.XmlAnyElementAttribute()]
    public System.Xml.XmlElement[] Any;
}

// return 10 largest orders from Caspio Bridge table 'orders'

CaspioBridgeTableRow [] aRows;

aRows=aobjWS.SelectData(strAccountID, strProfileName, strPassword,
"Orders", false,
    "TOP 10 OrderID, OrderTotal as Total",
"OrderTotal IS NOT NULL",
"OrderTotal DESC");
```

3.4.2 CreateTable

Creates Caspio Bridge table with multiple fields in one operation.

Parameters:

AccountID, Profile, Password, TableName, TableSchema

Return Value: none.

Example (VB):

```
'definition of CaspioBridgeTable

<System.Xml.Serialization.XmlTypeAttribute( _
    [Namespace]:= "http://bridge.caspio.net/ws/literalTypes")> _
    Public Class CaspioBridgeTable
        <System.Xml.Serialization.XmlArrayItemAttribute("Column", IsNullable:=false)> _
        Public Table() As CaspioBridgeTableColumn
    End Class

'definition of CaspioBridgeColumn

<System.Xml.Serialization.XmlTypeAttribute( _
[Namespace]:= "http://bridge.caspio.net/ws/literalTypes")> _
    Public Class CaspioBridgeTableColumn
        Public Name As String
        Public Type As CaspioBridgeColumnType
        Public Unique As Boolean
        <System.Xml.Serialization.XmlElementAttribute(IsNullable:=true)> _
        Public Description As String
    End Class

'definition of CaspioBridgeColumnType

<System.Xml.Serialization.XmlTypeAttribute( _
[Namespace]:= "http://bridge.caspio.net/ws/literalTypes")> _
    Public Enum CaspioBridgeColumnType
        LongText
        ShortText
        Number
        AutoNumber
        <System.Xml.Serialization.XmlEnumAttribute("Date/Time")> _
        DateTime
        <System.Xml.Serialization.XmlEnumAttribute("Yes/No")> _
```

```

        YesNo
        File
    End Enum

'create table object with one unique String column
Dim cTable As New CaspioBridgeTable
ReDim cTable.Table(0)
cTable.Table(0).Name="NewStringColumn"
cTable.Table(0).Type=CaspioBridgeColumnType.ShortText
cTable.Table(0).Unique=true

'invoke WS API
aobjWS.CreateTable(strAccountID, strProfileName, strPassword, "NewTable", cTable)

```

Example (C#):

```

// definition of CaspioBridgeTable

[System.Xml.Serialization.XmlTypeAttribute(
    Namespace="http://bridge.caspio.net/ws/literalTypes")]
    public class CaspioBridgeTable {
        [System.Xml.Serialization.XmlArrayItemAttribute("Column", IsNullable=false)]
        public CaspioBridgeTableColumn[] Table;
    }
//definition of CaspioBridgeTableColumn

[System.Xml.Serialization.XmlTypeAttribute(
    Namespace="http://bridge.caspio.net/ws/literalTypes")]
    public class CaspioBridgeTableColumn {
        public string Name;
        public CaspioBridgeColumnType Type;
        public bool Unique;
        [System.Xml.Serialization.XmlElementAttribute(IsNullable=true)]
        public string Description;
    }

// definition of CaspioBridgeColumnType

[System.Xml.Serialization.XmlTypeAttribute(
    Namespace="http://bridge.caspio.net/ws/literalTypes")]
    public enum CaspioBridgeColumnType {
        LongText,
        ShortText,
        Number,
        AutoNumber,
        [System.Xml.Serialization.XmlEnumAttribute("Date/Time")]
        DateTime,
        [System.Xml.Serialization.XmlEnumAttribute("Yes/No")]
        YesNo,
        File,
    }

// create table object with one unique String column
CaspioBridgeTable cTable=new CaspioBridgeTable();
cTable.Table[0]=new CaspioBridgeTableColumn();
cTable.Table[0].Name="NewStringColumn";
cTable.Table[0].Type=CaspioBridgeColumnType.ShortText;
cTable.Table[0].Unique=true;

//invoke WS API
aobjWS.CreateTable(strAccountID, strProfileName, strPassword, "NewTable", cTable)

```

3.4.3 GetTableDesign

Describes an existing Caspio Bridge table in XML.

Parameters:

AccountID, Profile, Password, TableName

Return Value: CaspioBridgeTable contains design definition of specified table (see **Types** below).

Example (VB):

```
'definition of CaspioBridgeTable

<System.Xml.Serialization.XmlTypeAttribute( _
    [Namespace]:= "http://bridge.caspio.net/ws/literalTypes")> _
    Public Class CaspioBridgeTable
        <System.Xml.Serialization.XmlArrayItemAttribute("Column", IsNullable:=false)> _
            Public Table() As CaspioBridgeTableColumn
    End Class

'definition of CaspioBridgeColumn

<System.Xml.Serialization.XmlTypeAttribute( _
    [Namespace]:= "http://bridge.caspio.net/ws/literalTypes")> _
    Public Class CaspioBridgeTableColumn
        Public Name As String
        Public Type As CaspioBridgeColumnType
        Public Unique As Boolean
        <System.Xml.Serialization.XmlElementAttribute(IsNullable:=true)> _
            Public Description As String
    End Class

'definition of CaspioBridgeColumnType

<System.Xml.Serialization.XmlTypeAttribute( _
    [Namespace]:= "http://bridge.caspio.net/ws/literalTypes")> _
    Public Enum CaspioBridgeColumnType
        LongText
        ShortText
        Number
        AutoNumber
        <System.Xml.Serialization.XmlEnumAttribute("Date/Time")> _
            DateTime
        <System.Xml.Serialization.XmlEnumAttribute("Yes/No")> _
            YesNo
        File
    End Enum

Dim cTable As CaspioBridgeTable
cTable=aobjWS.GetTableDesign(strAccountID, strProfileName, strPassword, "Orders")

'show the name and description of first column

Msgbox (cTable.Table(0).Name & ":" & cTable.Table(0).Description)
```

Example (C#):

```
// definition of CaspioBridgeTable

[System.Xml.Serialization.XmlTypeAttribute(
    Namespace="http://bridge.caspio.net/ws/literalTypes")]
    public class CaspioBridgeTable {
        [System.Xml.Serialization.XmlArrayItemAttribute("Column", IsNullable=false)]
        public CaspioBridgeTableColumn[] Table;
    }
//definition of CaspioBridgeTableColumn

[System.Xml.Serialization.XmlTypeAttribute(
```

```

Namespace="http://bridge.caspio.net/ws/literalTypes")]
    public class CaspioBridgeTableColumn {
        public string Name;
        public CaspioBridgeColumnType Type;
        public bool Unique;
        [System.Xml.Serialization.XmlElementAttribute(Nullable=true)]
        public string Description;
    }

// definition of CaspioBridgeColumnType

[System.Xml.Serialization.XmlTypeAttribute(
Namespace="http://bridge.caspio.net/ws/literalTypes")]
    public enum CaspioBridgeColumnType {
        LongText,
        ShortText,
        Number,
        AutoNumber,
        [System.Xml.Serialization.XmlEnumAttribute("Date/Time")]
        DateTime,
        [System.Xml.Serialization.XmlEnumAttribute("Yes/No")]
        YesNo,
        File,
    }

CaspioBridgeTable cTable;
cTable=aobjWS.GetTableDesign(strAccountID, strProfileName, strPassword, "Orders");

//show the name and description of first column

Msgbox (cTable.Table[0].Name + ":" + cTable.Table[0].Description);

```

3.4.4 ListObjects

Returns the list of all Caspio Bridge tables and views, accessible by this profile

Parameters:

AccountID, Profile, Password

Return Value: array of CaspioBridgeObject (see **Types** below).

Example (VB):

```

'definition of CaspioBridgeObject

<System.Xml.Serialization.XmlTypeAttribute( _
[Namespace]:= "http://bridge.caspio.net/ws/literalTypes")> _
    Public Class CaspioBridgeObject
        Public Name As String
        Public Type As ObjectType
        <System.Xml.Serialization.XmlElementAttribute("Privilege")> _
        Public Privilege() As PrivilegeEnumType
    End Class

'definition of CaspioBridge ObjectType
'note that ListObjects never returns objects with types other than Table or View

<System.Xml.Serialization.XmlTypeAttribute( _
[Namespace]:= "http://bridge.caspio.net/ws/literalTypes")> _
    Public Enum ObjectType
        Table
        View
        DataPage
    End Enum

```

```

    End Enum

'definition of CaspioBridge PrivilegesEnum

<System.Xml.Serialization.XmlTypeAttribute( _
[Namespace]:= "http://bridge.caspio.net/ws/literalTypes")> _
    Public Enum PrivilegeEnumType
        Manage
        [Select]
        Insert
        Update
        Delete
        Design
    End Enum

Dim aMyAccessibleObjects() as CaspioBridgeObject
aMyAccessibleObjects= aobjWS.ListObjects(strAccountID, strProfileName, strPassword)

```

Example (C#):

```

//definition of CaspioBridgeObject

[System.Xml.Serialization.XmlTypeAttribute(
Namespace="http://bridge.caspio.net/ws/literalTypes")]
public class CaspioBridgeObject {
    public string Name;
    public ObjectType Type;
    [System.Xml.Serialization.XmlElementAttribute("Privilege")]
    public PrivilegeEnumType[] Privilege;
}

//definition of CaspioBridge ObjectType
//note that ListObjects never returns objects with types other than Table or View

[System.Xml.Serialization.XmlTypeAttribute(
Namespace="http://bridge.caspio.net/ws/literalTypes")]
public enum ObjectType {
    Table,
    View,
    DataPage
}

//definition of CaspioBridge PrivilegesEnum

[System.Xml.Serialization.XmlTypeAttribute(
Namespace="http://bridge.caspio.net/ws/literalTypes")]
public enum PrivilegeEnumType {
    Manage,
    Select,
    Insert,
    Update,
    Delete,
    Design
}

CaspioBridgeObject[] aMyAccessibleObjects;
aMyAccessibleObjects= aobjWS.ListObjects(strAccountID, strProfileName, strPassword);

```

3.4.5 ListDataPages

Returns list of available DataPages.

Parameters:

AccountID, Profile, Password

Return Value: array of CaspioBridgeDataPage objects (see **Types** below).

Example (VB):

```
'definition of CaspioBridgeDataPage

<System.Xml.Serialization.XmlTypeAttribute( _
[Namespace]:= "http://bridge.caspio.net/ws/literalTypes")> _
    Public Class CaspioBridgeDataPage
        Inherits CaspioBridgeObject
        Public AppKey As String
        Public FolderName As String
        Public AppType As DataPageAppTypes
        Public DeploymentStatus As DataPageDeploymentStatus
    End Class

'definition of CaspioBridge ObjectType

<System.Xml.Serialization.XmlTypeAttribute( _
[Namespace]:= "http://bridge.caspio.net/ws/literalTypes")> _
    Public Enum ObjectType
        Table
        View
        DataPage
    End Enum

'definition of DataPageAppTypes

<System.Xml.Serialization.XmlTypeAttribute( _
[Namespace]:= "http://bridge.caspio.net/ws/literalTypes")> _
    Public Enum DataPageAppTypes
        Form
        Table
        Report
        SearchReport
        HTMLPage
        PasswordRecovery
        MobileForm
    End Enum

'definition of DataPageDeploymentStatus

<System.Xml.Serialization.XmlTypeAttribute( _
[Namespace]:= "http://bridge.caspio.net/ws/literalTypes")> _
    Public Enum DataPageDeploymentStatus
        Undeployed
        Deployed
        HTTPS_Only
    End Enum

Dim aMyDataPages() as CaspioBridgeDataPage
aMyDataPages= aobjWS.ListDataPages(strAccountID, strProfileName, strPassword)
```

Example (C#):

```
//definition of CaspioBridgeDataPage

[System.Xml.Serialization.XmlTypeAttribute(
    Namespace="http://bridge.caspio.net/ws/literalTypes")]
```

```

    public class CaspioBridgeDataPage {
        public string Name;
        public ObjectType Type;
        [System.Xml.Serialization.XmlElementAttribute("Privilege")]
        public PrivilegeEnumType[] Privilege;
        public string AppKey;
        public string Folder;
        public DataPageAppTypes AppType;
        public DataPageDeploymentStatus Deployment;
    }

//definition of CaspioBridge ObjectType

[System.Xml.Serialization.XmlTypeAttribute(
    Namespace="http://bridge.caspio.net/ws/literalTypes")]
    public enum ObjectType {
        Table,
        View,
        DataPage
    }

//definition of DataPageAppTypes

[System.Xml.Serialization.XmlTypeAttribute(
    Namespace="http://bridge.caspio.net/ws/literalTypes")]
    public enum DataPageAppTypes {
        Form,
        Table,
        Report,
        SearchReport,
        HTMLPage,
        PasswordRecovery,
        MobileForm,
    }

//definition of DataPageDeploymentStatus

[System.Xml.Serialization.XmlTypeAttribute(
    Namespace="http://bridge.caspio.net/ws/literalTypes")]
    public enum DataPageDeploymentStatus {
        Undeployed,
        Deployed,
        HTTPS_Only
    }

CaspioBridgeDataPage[] aMyDataPages;
aMyDataPages= aobjWS.ListDataPages(strAccountID, strProfileName, strPassword);

```

3.5 Types

This section contains formal XML schemas for custom data types used in Document-style WS API methods. Definitions in this section may be different – generally stricter - from those in the WSDL file.

xsd prefix always refers to XML schema namespace, **wsapi** prefix refers to <http://bridge.caspio.net/ws/literalTypes> namespace.

3.5.1 Enumerations

3.5.1.1 ObjectType

```

<xsd:simpleType name="ObjectType">
<xsd:restriction base="xsd:string">
<xsd:enumeration value="Table" />

```

```

<xsd:enumeration value="View" />
<xsd:enumeration value="DataPage" />
</xsd:restriction>
</xsd:simpleType>

```

3.5.1.2 PrivilegeEnumType

```

<xsd:simpleType name="PrivilegeEnumType">
<xsd:restriction base="xsd:string">
  <xsd:enumeration value="Manage" />
  <xsd:enumeration value="Select" />
  <xsd:enumeration value="Insert" />
  <xsd:enumeration value="Update" />
  <xsd:enumeration value="Delete" />
  <xsd:enumeration value="Design" />
</xsd:restriction>
</xsd:simpleType>

```

3.5.1.3 CaspioBridgeColumnType

```

<xsd:simpleType name="CaspioBridgeColumnType">
<xsd:restriction base="xsd:string">
<xsd:enumeration value="LongText" />
<xsd:enumeration value="ShortText" />
<xsd:enumeration value="Number" />
<xsd:enumeration value="AutoNumber" />
<xsd:enumeration value="Date/Time" />
<xsd:enumeration value="Yes/No" />
<xsd:enumeration value="File" />
</xsd:restriction>
</xsd:simpleType>

```

Note *LongText* fields may contain up to 64000 Unicode characters, *ShortText* fields may contain up to 255 Unicode characters. No truncation is performed.

3.5.1.4 DataPageAppTypes

```

<xsd:simpleType name="DataPageAppTypes">
<xsd:restriction base="xsd:string">
<xsd:enumeration value="Form" />
<xsd:enumeration value="Table" />
<xsd:enumeration value="Report" />
<xsd:enumeration value="SearchReport" />
<xsd:enumeration value="HTMLPage" />
<xsd:enumeration value="PasswordRecovery" />
<xsd:enumeration value="MobileForm" />
</xsd:restriction>
</xsd:simpleType>

```

3.5.1.5 DataPageDeploymentStatus

```

<xsd:simpleType name="DataPageDeploymentStatus">
<xsd:restriction base="xsd:string">
<xsd:enumeration value="Undeployed" />
<xsd:enumeration value="Deployed" />
<xsd:enumeration value="HTTPS_Only" />
</xsd:restriction>
</xsd:simpleType>

```

3.5.2 Structures

3.5.2.1 CaspioBridgeTable

```
<xsd:complexType name="CaspioBridgeTable">
<xsd:sequence>
<xsd:element minOccurs="1" maxOccurs="1" name="Table"
type="wsapi:ArrayOfCaspioBridgeTableColumn" />
</xsd:sequence>
</xsd:complexType>

<xsd:complexType name="ArrayOfCaspioBridgeTableColumn">
<xsd:sequence>
<xsd:element minOccurs="0" maxOccurs="unbounded" name="Column"
type="wsapi:CaspioBridgeTableColumn" />
</xsd:sequence>
</xsd:complexType>
```

3.5.2.2 CaspioBridgeTableColumn

```
<xsd:complexType name="CaspioBridgeTableColumn">
<xsd:sequence>
<xsd:element minOccurs="1" maxOccurs="1" name="Name" type="xsd:string" />
<xsd:element minOccurs="1" maxOccurs="1" name="Type" type="wsapi:CaspioBridgeColumnType"
/>
<xsd:element minOccurs="1" maxOccurs="1" name="Unique" type="xsd:boolean" />
<xsd:element minOccurs="1" maxOccurs="1" name="Description" nillable="true"
type="xsd:string" />
</xsd:sequence>
</xsd:complexType>
```

3.5.2.3 CaspioBridgeObject and CaspioBridgeResponse

```
<xsd:complexType name="CaspioBridgeObject">
<xsd:sequence>
<xsd:element minOccurs="1" maxOccurs="1" name="Name" type="xsd:string" />
<xsd:element minOccurs="1" maxOccurs="1" name="Type" type="wsapi:ObjectType" />
<xsd:element minOccurs="0" maxOccurs="unbounded" name="Privilege"
type="wsapi:PrivilegeEnumType" />
</xsd:sequence>
</xsd:complexType>

<xsd:complexType name="CaspioBridgeResponse">
<xsd:sequence>
<xsd:element minOccurs="0" maxOccurs="unbounded" name="CaspioBridgeObject"
type="wsapi:CaspioBridgeObject" />
</xsd:sequence>
</xsd:complexType>
```

3.5.2.4 CaspioBridgeDataPage and CaspioBridgeDataPageResponse

```
<xsd:complexType name="CaspioBridgeDataPage">
<xsd:complexContent mixed="false">
<xsd:extension base="wsapi:CaspioBridgeObject">
<xsd:sequence>
<xsd:element minOccurs="1" maxOccurs="1" name="AppKey" type="xsd:string" />
<xsd:element minOccurs="1" maxOccurs="1" name="Folder" type="xsd:string" />
<xsd:element minOccurs="1" maxOccurs="1" name="AppType" type="wsapi:DataPageAppTypes" />
<xsd:element minOccurs="1" maxOccurs="1" name="Deployment"
type="wsapi:DataPageDeploymentStatus" />
</xsd:sequence>
</xsd:extension>
```

```

    </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="CaspioBridgeDataPageResponse">
<xsd:sequence>
<xsd:element minOccurs="0" maxOccurs="unbounded" name="CaspioBridgeDataPage"
type="wsapi:CaspioBridgeDataPage" />
    </xsd:sequence>
</xsd:complexType>

```

3.5.2.5 CaspioBridgeDataResponse and CaspioBridgeTableRow

```

<xsd:complexType name="CaspioBridgeDataResponse">
  <xsd:sequence>
    <xsd:element minOccurs="0" maxOccurs="unbounded" name="Row"
      type="wsapi:CaspioBridgeTableRow" />
  </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="CaspioBridgeTableRow">
  <xsd:sequence>
    <xsd:any maxOccurs="unbounded" />
  </xsd:sequence>
</xsd:complexType>

```

3.6 Faults

If Caspio Bridge Web Services is unable to process a request for any reason, it generates a SOAP fault. Caspio Bridge-specific error messages are added to generic SOAP fault message enclosed in figure brackets. Also, Caspio Bridge WS API provides the Detail part of SOAP faults as shown in this example:

```

<Detail>
  <ErrorID>1013</ErrorID> <!--Caspio Bridge WS API error code -->
  <ErrorMessage> Unsecured connection is denied.</ErrorMessage> <!--Description-->
</Detail>

```

ErrorMessage node contains Caspio Bridge-specific error message, it is the same as the part of SOAP fault message enclosed in figure brackets.

The following table contains the full list of all Caspio Bridge specific error codes and messages. Some messages are assigned to more than one error code.

Error ID	Message	Description
1001 11001	Incorrect account name.	Specified Caspio Bridge Account ID does not exist.
1002	Incorrect profile name or password.	Specified Web Service Profile does not exist or incorrect password was passed.
1003	Account is disabled.	Specified Caspio Bridge Account ID is disabled.
1004	Account is expired.	Specified Caspio Bridge Account ID is expired.
1005	Profile is disabled.	Specified Web Service Profile is disabled.
1006	Access to specified object denied.	Specified Web Service Profile has no privilege to access the specified object.
1007	Access from specified IP is denied.	Specified Web Service Profile has IP restrictions and client's IP is not allowed to access WS API via specified profile.
1008	Not enough privilege to perform requested operation.	Specified Web Service Profile does not have enough privileges to perform the requested task.

1010 10003	Table does not exist.	Table specified in TableName or ObjectName parameter does not exist.
1011	View does not exist.	View specified in ObjectName parameter does not exist.
1012	Invalid function argument.	Invalid parameter value was passed.
1013	Unsecured connection is denied.	Unsecured access is disabled for this profile.
1014 10011	Invalid table name "Name".	Table with such name cannot be created. Table names may be up to 32 characters long, may contain letters, digits and underscore, and must start with a letter.
1015	SQL Server error: "Error message"	SQL Server cannot handle the query for the specified reason.
1017	Cannot change value in autonumber field(s): "Name1, Name2"	Values in fields of type Autonumber cannot be changed.
1018	Field "Field Name" must be unique.	Autonumber fields must be unique.
1019	Proximity search feature is not available	Proximity search methods are not available for this account.
1020	Field 'Name' has invalid type for proximity search	Fields used as latitude or longitude fields must be of type 'Number' or 'ShortText' only.
1021	Incorrect Units value	Parameter 'Units' must be an integer from 1 to 6.
1022	Field 'Name' does not exist	Cannot use specified field in proximity search because it does not exist.
1023	DataPage API is not available for this profile.	This Web Service Profile cannot call any DataPage-related methods. Web Service Profile must have Allow DataPage access attribute set to be able to access DataPage-related methods
1024	DataPage 'Name' does not exist.	May occur because of several reasons: <ul style="list-style-type: none"> • DataPage with specified name cannot be found in specified folder, • Specified folder does not exist • DataPage with specified AppKey does not exist ('Name' part is omitted in this case)
1025	Incorrect DeployMethod value	DeployMethod must be one of predefined values. See Parameters section for details.
1026	Mobile forms can not be deployed using requested method.	Mobile forms can be deployed using "URL" or "Link" methods only.
1027	Deployed DataPage limit is reached.	Account's Deployed DataPage limit was reached.
1028	AppKey must be empty.	<i>AppKey</i> attribute in DataPageXML must have empty value.
1029	Invalid DataPage name.	DataPage <i>Name</i> attribute in DataPageXML is incorrect. DataPage name cannot contain any of following characters: +&<>%;\
1030	DataPage folder does not exist.	DataPage folder specified in Folder does not exist.
1031	Invalid view name.	View <i>Name</i> attribute in ViewXML is incorrect. View name may contain only letters, digits and underscore sign. First character of name must be a letter.
1032	Incorrect view XML.	ViewXML is incorrect. See description of CreateView method
1033	File does not exist.	File with ID specified in FileID does not exist.
1034	Folder does not exist.	Folder with ID specified in FolderID does not exist.
1035	File is too large.	Size of FileData is larger than file upload limit set by Caspio Bridge server administrator. File cannot be uploaded.

1036	Invalid file name.	File cannot be created. FileName is either empty or longer than 255 characters or contains one of the following characters: /+\\<>:"*?
1037	File or folder with such name already exists.	Folder specified in FolderID already contains file or folder with the same name as specified in FileName . File cannot be created. If Overwrite was set to true, only folder may cause this error.
1038	Insufficient quota.	Account's quota limit is reached. File cannot be uploaded.
10004	Table already exists.	Table with specified name cannot be created because it already exists.
10005	Field "Name" does not exist.	Cannot drop or alter specified field, because it does not exist.
10006	Field "Name" already exists.	Field with specified name cannot be created or renamed because field with the same name already exists.
10007	Invalid field type "Type".	Specified Type or NewType parameter contains incorrect value.
10010 11010	Permission denied.	See 1008
10012	Invalid field name "Name".	Field with such name cannot be created. Field names may be up to 32 characters long, may contain letters, digits and underscore sign, and must start with letter.
10015	"word" is a reserved word.	Table or field with specified name cannot be created or renamed to the specified name. This name is reserved.
10019	Type conversion is not possible.	Field cannot be converted to the type specified.
10030	Cannot make field "Name" unique.	Specified field cannot be made unique, because it already contains empty or duplicate values or field data type does not support uniqueness.
10500	DataPage already exists.	DataPage with the name specified by DataPageXML already exists in a folder specified by Folder parameter
10510	Field cannot be altered because it is used in authentication.	Field cannot be changed if it is used in user authentication.
10511	Cannot add field because 8060 bytes per row limit is reached.	Each row in Caspio Bridge tables may contain a maximum 8060 bytes of data.
10519	Number of DataPages limit is reached.	DataPage cannot be created because number of DataPages limit is reached.
10525	Field cannot be altered because it is used in a View or DataPage criteria.	Field cannot be changed because it is used in a View or DataPage.
1000 10001	Internal error.	Internal web service error.
11500	Incorrect DataPage XML.	DataPageXML passed to CreateDataPage call is incorrect. If this error is raised during call of other methods, see 10001 error code.
13001	View already exists.	View with the same name as specified by ViewXML already exists.

4 Code Samples

4.1 MS SOAP Toolkit

4.1.1 Visual Basic 6.0

Sample located in the **Samples/MSSOAP/VB** folder. It demonstrates the usage of **GetTableDesignRaw** and **SelectDataRaw** WS API methods. This application retrieves data from your Caspio Bridge table and stores it in a Microsoft Excel worksheet.

To run this sample:

1. Open Microsoft Visual Basic 6.0.
2. Open ImportToExcel.vbp project.
3. Press F5 to run sample.

Note: To run this sample, you should have MS SOAP Toolkit 3.0 and Microsoft Excel 2000 or higher installed.

4.1.2 MS Office with Web Services Toolkit

Sample located in the **Samples/MSSOAP/Office** folder. It demonstrates the usage of proxy classes and **CreateEmptyTable**, **AddTableField**, **InsertData** and **SelectDataXML** WS API methods. This application retrieves data from your Caspio Bridge table as XML, performs XSLT transformation, loads transformed data into MS Excel and utilizes the resulting xls file as a data source for Mail Merge in MS Word.

To run this sample:

1. Open Microsoft Excel.
2. Set Macro Security level to Medium (**Tools** menu, **Options**, **Security** tab, **Macro Security** button).
3. Open MailMerge.xls file, click **Enable Macros**.
4. In order to create a sample table with data, from the menu select **Caspio Bridge API**, **Create Sample Data**.
5. Enter your Web Service Profile credentials and click **Create Table**.
6. From the menu **Caspio Bridge API** select **Run Mail Merge**.
7. Click **Run Mail Merge** button.
8. Microsoft Word will start and **Select Table** dialog will appear.
9. Click OK and Word will create "Form Letters1" document with results of mail merge from the Caspio Bridge table created on step 5.

Note: To run this sample you should have MS SOAP Toolkit 3.0, Windows Scripting Host, Microsoft Excel XP or higher and Microsoft Word XP or higher installed.

4.1.3 ASP and SOAP toolkit

Sample located in **Samples/MSSOAP/ASP** folder. It demonstrates simple auto-complete web form text box populated by data from Caspio Bridge table.

To run this sample:

- Deploy js, htm, asp files to your web server.
- Edit autocomplete.asp and set correct values for the variables *wSDL*, *tableName*, *columnName*, *name*, *profile*, *password*, *where*
 - wSDL – WSDL file of your web-service (e.g. <http://bridge.caspio.net/ws/API.asmx?wSDL>)
 - tableName – name of table that will be used as the data source (e.g. Employees)
 - columnName – column name in data source (e.g. LastName)
 - name – your account name
 - profile - your profile name
 - password - your profile's password
- Use you browser to navigate to index.htm file and type anything in textbox.

Note: This sample requires MSXML 4.0 or higher to be installed on web server.

4.2 Microsoft .Net Framework

Sample located in **Samples/Net** folder. It contains a general example of WS API usage in C# .Net Windows applications and demonstrates the object de-serialization and usage of **SelectDataXML** and **ListObjects** WS API methods. This application connects to your Caspio Bridge table via WS API and displays a popup alert each time new data is added to the table.

To run this sample:

1. Open Microsoft Visual Studio.Net.
2. Open Watchdog.sln project.
3. Press F5 to run sample.
4. Enter your Web Service Profile credentials.
5. Click **Refresh** button.
6. Choose table to watch and click **Start**.

Note: Sample was written for .Net framework version 1.1, but it can be opened by Visual Studio 2005 (or later) and converted to later versions of .Net Framework.

4.3 Apache Axis

Sample located in **Samples/Axis** folder. It contains a general example of WS API usage in Java application and demonstrates the object serialization/deserialization in Java and usage of CreateTable and GetTableDesign WS API methods.

To run this sample:

1. Create **Samples\Axis\CaspioBridge\Lib** subfolder
2. Copy jars mentioned in 3.3 to this folder
3. Start Eclipse 3.1
4. From the menu **File, Import...**, select **Existing project into Workspace**
5. Select **Samples\Axis\CaspioBridge** and press **OK**
6. Go into project **CaspioBridge**
7. Select menu **Run, Run...**
8. Select **Java Application** and press **New**
9. In **Project** text field enter *CaspioBridge*
10. In **Main class** text field enter *Main*
11. Press button **Run**

Note. You have to use Web Service Profile with "Required SSL encryption" option turned off in order to run this sample.

4.4 PHP 5.0

4.4.1 Orders sample

Sample located in **Samples/Php5/Orders** folder. It demonstrates the usage of Caspio Bridge WS API using PHP scripts. Examples include calls of **CreayEmptyTable**, **AddTableField**, **InsertData**, **GetTableDesignRaw** and **SelectDataRaw**.

To run this sample:

1. Navigate to <https://bridge.caspio.net/ws/api.asmx?wsdl> (or other site instead of bridge.caspio.net if your account is hosted on other CaspioBridge site) and save file as "api.wsdl" in the folder where php sample files are located.
2. Open one of the php samples in a browser.

lib.php contains backend implementation of WS API calls.

createOrderTable.php creates the Caspio Bridge table "orders" used in these samples. You can change the default table name by editing the \$tableName constant in lib.php file.

addData.php provides a form which can be used to fill "orders" table with data.

searchData.php demonstrates a form which searches Caspio Bridge table "orders" for a string entered by user. Search is performed in all text fields.

4.4.2 AutoComplete sample

Sample located in **Samples/Php5/Autocomplete** folder. It demonstrates simple auto-complete web form text box, populated by data from Caspio Bridge table.

To run this sample:

- Deploy js, htm, php files and SOAP folder to your web server.
- Edit autocomplete.php and set correct values for variables: \$wsdl, \$tableName, \$columnName, \$name, \$profile, \$password (lines 15-25), where
 - \$wsdl – WSDL file of your web-service (e.g. <http://bridge.caspio.net/ws/API.asmx?wsdl>)
 - \$tableName – name of table which will be the data source (e.g. Employees)
 - \$columnName – column's name in data source (e.g. LastName)
 - \$name – your account's name
 - \$profile - your profile's name
 - \$password - your profile's password
- Use you browser to navigate to index.htm file and type anything in text box.

5 Resources

Caspio Bridge – <http://www.caspio.com>

SOAP Specification - <http://www.w3.org/TR/soap/>

WSDL Specification - <http://www.w3.org/TR/wsdl>

XML Schema specification - <http://www.w3.org/XML/Schema>

About SOAP - <http://www.soapware.org/>

MS SOAP Toolkit -<http://msdn.microsoft.com/en-us/library/aa286526.aspx>

Web Services in Microsoft .Net Framework - <http://msdn.microsoft.com/en-us/library/ms950421.aspx>

Apache Axis - <http://ws.apache.org/soap/>

<http://ws.apache.org/axis/>

SOAP::Lite for PERL - <http://www.soaplite.com/>

SOAP in PHP - <http://php.net/manual/en/book.soap.php>

SOAP in PEAR – <http://pear.php.net/package/SOAP>

SOAP in Python - <http://www.ibm.com/developerworks/webservices/library/ws-pyth5/>

SOAP in Ruby on Rails - <http://stdlib.rubyonrails.org/libdoc/soap/rdoc/index.html>

Existing SOAP libraries - <http://www.soapware.org/directory/4/implementations>