SERENA

SERENA®
BUSINESS MASHUPS™

Web Services Developer's Guide

Serena Proprietary and Confidential Information

Trademarks

U.S. Government Rights

Part number: MA-MUAPI-004 Product version: 2009 R1

Publication date: 2009-03-14

Table of Contents

Chapter 1: Preface 9
Guide to Business Mashups Documentation 9
Typographical Conventions 10
Chapter 2: Getting Started 13
About the Business Mashups Web Services API 13
About Web Services 13
SOAP Requests 13
Before You Begin 14
System Requirements 14
Licensing 14
Security 14
Setting Up the Development Environment 14
Generating Web Services Stubs 14
Setting Up Microsoft Visual Studio 15
Upgrading Visual Studio 15
Setting Up Web Service Stub Files 15
Installing the Business Mashups Web Services API 16
Web Service and Script Execution Order 16
Error Message Logs 17
Chapter 3: Web Services API Reference 19
Calls Available 19
CreateAuxItem 23
CreateAuxItemsWithName 25
CreateAuxItemWithName 28
CreateAuxItems 30
CreateFileAttachment 33
CreatePrimaryItem 34
CreatePrimaryItems 37
CreatePrimaryItemsWithName 40
CreatePrimaryItemWithName 42
DeleteAttachment 45
DeleteItem 46
DeleteItems 47

DeleteItemsByQuery 48
DeleteItemsByQueryWithName 49
DeleteMashup 51
Export 52
GenerateUUID 53
GetApplications 54
GetAvailableSubmitTransitions 54
GetAvailableSubmitTransitionsWithName 55
GetAvailableTransitions 57
GetAvailableTransitionsWithStateIDs 58
GetFileAttachment 59
Getitem 60
Getitems 61
GetItemsByQuery 63
GetItemsByQueryWithName 66
GetReports 69
GetSolutions 70
GetSolutionsWithUniqueName 71
GetSubmitProjects 72
GetSubmitProjectsWithName 73
GetTables 74
GetTablesWithName 75
GetUser 76
GetUserPrivileges 77
GetUserWithPreferences 79
GetVersion 80
HasUserPrivilege 80
Import 82
ImportStatus 83
IsUserValid 84
Logout 85
RunReport 86
UpdateFileAttachment 88
UpdateItem 90
UpdateItems 93

UpdateItemsWithName 96
UpdateItemWithName 99
Arguments 101
Attachment-Access-Type 103
Auth 104
FileAttachment 105
FileAttachmentContents 106
FileContents 107
Item 107
ItemLink 108
ItemLink-Type 109
NameValue 110
Note 113
QueryRange 113
ReportAccessLevel 114
ReportCategory 115
ReportsFilter 116
ReportType 118
Set-Value-By 119
Set-Value-Method 120
Table-Type 123
TTItem 123
URLAttachment 132
Value or Values 133
Responses 135
ApplicationData 137
ExtraValue 138
DatePreference 139
Field 140
Field-Type 141
FileAttachment 143
FileAttachmentContents 144
FileContents 144
ImportCurrentOverallStatus 145
ImportCurrentStatus 146

GetReportsResult 146
OrderBy 148
Privilege 149
ProjectData 150
ReportAccessLevel 151
ReportCategory 153
ReportDefinition 153
ReportInfo 155
ReportResult 157
RunReportResult 158
SolutionData 162
Solution-Type 164
SolutionWithUniqueName 165
TableData 166
Table-Type 168
TimePreference 169
Transition 170
Transition-Type 172
TTItem 174
User 179
UserWithPreferences 180
Chapter 4: Tips for Writing Applications 183
Authentication Methods 183
Argument 183
HTTP Basic 183
WS-Security 183
C++ Constants 183
Supported Character Encoding 184
Supported Date/Time Formats 184
Application, Table, and Project IDs 185
Chapter 5: Sample Programs 187
About the Sample Programs 187
SolutionsAndTables 187
CreateItems 187
DeleteItems 188

GetItems 188
UpdateItems 188
Using the Sample Programs 188
Requirements 188
Choosing Authentication 189
Editing the Web Services URL 189
Rebuilding the Executables 189

Chapter 1: Preface

This document describes how to use the Web Services Application Programming Interface (API) of Serena® Business Mashups, a product of Serena Software, Inc. Business Mashups is a Web-architected change request and process management solution that empowers application development teams to improve communication and development processes across the enterprise.

Through the Web services API, you can develop integrations with Business Mashups that create, read, update, and delete items within Business Mashups. The Web services API also lets you list Business Mashups applications and tables.

Audience and Scope

This manual is intended for experienced developers interested in integrating their products with Business Mashups.

Important: Support for development efforts writing Web services is provided by Serena Consulting services. Questions regarding use of Web services operations in orchestration processes as documented are handled by Serena Customer support.

Guide to Business Mashups Documentation

The Business Mashups documentation set includes the following manuals in PDF format. Most documents are installed with Business Mashups; to obtain a document that is not installed with the product or to download the complete documentation set, visit Serena Support.

Manual	Description
Moving to Serena® Business Mashups	Provides migration information for existing TeamTrack customers who are moving to Business Mashups. It explains how to upgrade your existing system, and it explains the expanded Business Mashups paradigm in relation to the TeamTrack paradigm.
Serena® Business Mashups Installation Guide	Provides information on installing Business Mashups, licensing, and creating a database. Database and Web server configuration information is also provided.
Serena® Mashup Composer Guide	Provides details on using Mashup Composer to create the tables, fields, workflows, orchestrations, and other design elements comprised in mashups. Information about saving, versioning, importing, and exporting mashups is also provided. This document is intended for individuals who want to design and maintain mashups.
Serena® Mashup Manager Guide	Provides information on using Mashup Manager to deploy mashups to runtime environments and to promote configured applications from one environment to another.

Manual	Description
Serena® Mashup Administrator Guide	Provides information on configuring and administering the Business Mashups application engine. Instructions for managing projects, user accounts, system settings, and authentication are included.
Serena® Business Mashups User's Guide	Provides information about the Business Mashups Web interface and is intended for end users. Instructions on using the Web interface, including the robust reporting feature in Business Mashups, are included. To ease the process of providing a copy for every user in your system, the *Serena® Business Mashups User's Guide* is provided in PDF and can be accessed from the **Product Information** tab of the **About** page in the Web interface.
SourceBridge User's Guide	Provides information on installing, configuring, and using the SourceBridge extension. The guide is intended for end users who integrate Business Mashups with version control software used within the integrated development environment (IDE) or within the Serena Version Manager Windows or Web clients. *SourceBridge User's Guide* is provided in PDF and can be accessed from the Product Information tab of the About page in the Business Mashups Web interface.
Serena® Business Mashups Web Services Developer's Guide	Provides an overview of all Business Mashups Web services, including descriptions for all calls, arguments, and responses. Installation instructions and information about the sample Web service programs are also provided.
Mashup Script Reference	Provides information on customizing Business Mashups using Mashup Script, a programming language built around VBScript 4.0. This guide is intended for VBScript programmers who want to use Mashup Script to implement custom features in a Business Mashups system.

Typographical Conventions

The following typographical conventions are used in the online manuals and online help. These typographical conventions are used to assist you when using the documentation; they are not meant to contradict or change any standard use of typographical conventions in the various product components or the host operating system .

Table 1.

Convention	Explanation
italics	Introduces new terms that you may not be familiar with and occasionally indicates emphasis.
bold	Emphasizes important information and field names.

Convention	Explanation
UPPERCASE	Indicates keys or key combinations that you can use. For example, press the ENTER key.
`monospace`	Indicates syntax examples, values that you specify, or results that you receive.
monospaced italics	Indicates names that are placeholders for values you specify; for example, *filename*.
`monospace bold`	Indicates the results of an executed command.
vertical rule |	Separates menus and their associated commands. For example, select File | Copy means to select Copy from the File menu. Also, indicates mutually exclusive choices in a command syntax line.

Chapter 2: Getting Started

- About the Business Mashups Web Services API [page 13]
- About Web Services [page 13]

About the Business Mashups Web Services API

With the Business Mashups Web services API (Application Programming Interface), you can access key Business Mashups features from your own applications. This enables you to build your own front-end clients for your users' most common tasks, including:

- Submitting items into projects or auxiliary tables
- Transitioning and updating items
- Deleting items
- Viewing item details
- Listing applications and projects

These items can either be project-based primary items or auxiliary items.

About Web Services

Web services are applications that are accessible using standard Internet protocols and formats such as Extensible Markup Language (XML), Hypertext Transfer Protocol (HTTP), or Simple Object Access Protocol (SOAP). You can implement applications that interact with Web services on any platform in any programming language, as long as the language can create and respond to messages that are sent using SOAP over HTTP.

The Web services API is supported on Windows using the IIS Web server for the GSOAP framework.

Note: To use the Business Mashups Web services from Perl, you need Soap::Lite version 0.69 or later.

SOAP Requests

Serena Business Mashups supports the SOAP protocol for calling Web service operations over HTTP or HTTPS. Web service SOAP messages are essentially specially formatted XML data packages sent between a client and a server. The SOAP protocol is popular since it communicates over HTTP, which typically allows access through company firewalls. Since the data is in XML format, different programming languages on different operating systems can send, receive, and process SOAP messages. For detailed information about the SOAP protocol, visit `http://www.w3.org/` and search for SOAP.

SOAP Web services are described via a WSDL file. A WSDL is simply an XML dataset that defines the calls, arguments, and responses in Web service interactions. A WSDL can be imported into a development environment and integrated with an application using a SOAP toolkit. For more information on setting up your development environment to create applications that can send, receive, and process SOAP messages, see Setting Up the Development Environment [page 14].

The URL for Business Mashups SOAP based Web service calls is:
`http://`*`serverName`*`:`*`aePort`*`/gsoap/gsoap_ssl.dll?aewebservices71`

A detailed list of supported SOAP calls can be found in Calls Available [page 19].

Before You Begin

Before you install and use the Business Mashups Web services API, there are a few things to consider.

System Requirements

The Business Mashups Web services API is supported on Windows with the Microsoft Internet Information Services (IIS) Web server. For the specific versions supported, see the Business Mashups readme.

Licensing

Use of the Web services consumes a license. If a user has already checked out a license, an additional license is not checked out for use of the Web services. If the user logs out but a Web service continues to run for that user, a license is not checked out while the Web service runs.

When administrators or designers log in to Mashup Manager or contact Mashup Manager through Mashup Composer to, say, check out a file, a license is consumed. If that person already has a license checked out for the Application Engine or for Mashup Administrator, Business Mashups uses that license.

Security

The Business Mashups Web services support the use of https to connect from the client. Use of SSL (Secure Sockets Layer) is recommended for any customers connecting to their Web services server in a non-secure environment. Using SSL prevents credentials from being extracted from the messages. IIS should be configured to allow or require SSL to connect to the Web services.

For information on setting up SSL, see the IIS documentation.

Authentication

Business Mashups supports the following authentication types:

- LDAP (Lightweight Directory Access Protocol)
- NT Challenge/Response
- Business Mashups authentication

Setting Up the Development Environment

This section provides important information on:

- Generating code stubs in your integrated development environment (IDE) from the Business Mashups Web services definition (WSDL) file.
- Setting up Microsoft Visual Studio .NET.

Generating Web Services Stubs

The Business Mashups Web services are defined in a WSDL file. You can build an application that interacts with the Business Mashups Web services in any IDE that can generate code stubs from WSDL files. You can create the stub files using the WSDL file at the following URL:

`http://serverName:aePort/gsoap/aewebservices71.wsdl`

Setting Up Microsoft Visual Studio

To create applications that interact with the Business Mashups Web services, you must upgrade Visual Studio and then set up Web service stub files.

Upgrading Visual Studio

To use WS-Security with the Business Mashups Web services, you must install the Microsoft Web Service Enhancements (WSE) product and then enable the WSE for your project.

For information on WS-Security, see Authentication Methods [page 183].

To upgrade Visual Studio .NET 2003 or 2005:

1. Download the appropriate version of WSE from the following location:

   http://msdn.microsoft.com/webservices/webservices/building/wse/default.aspx

2. After installing the WSE, open your solution in Visual Studio .NET.
3. Right-click your project and select **WSE Settings**.
4. Select the **Enable this project for Web Services Enhancements** check box.
5. Click **OK**.

You can now start building applications using the Business Mashups Web services with WS-Security.

Setting Up Web Service Stub Files

To access the Business Mashups Web services from Visual Studio .NET, you must add references to the WSDL file. When you do this, Visual Studio creates stub classes that your application can use to access the Web service methods. You can create these stub classes for Visual Basic or C#.

Once you have created the stub files, you can update them to use the client protocol provided by the WSE.

To set up the Web service stub files:

1. In Visual Studio .NET, create or open a project.
2. Select Project | Add Web Reference.
3. In the dialog box that appears, enter the URL to the Business Mashups Web services in the **URL** field:

   `http://serverName:aePort/gsoap/aewebservices71.wsdl`

4. Enter a name in the **Web reference name** field. This name will be used in your code to refer to the web reference. The sample programs uses `ttweb` as the web reference name.
5. Click **Add Reference**.

Installing the Business Mashups Web Services API

To use the Business Mashups Web services API, you must install Business Mashups version 2009 or later. The Web services API is installed when you perform either a *Complete Installation* or a *Setup for Web Services Only*. By default, the Web services API is installed in the following location:

```
\Business Mashups\Application Engine\webservices
```

On-Demand users can access the aewebservices71 WSDL here:

```
https://serenasupport.serenamashups.com/gsoap/aewebservices71.wsdl
```

For details on the installation process, see the *Serena® Business Mashups Installation Guide*.

Note: The latest Web service calls can be found in the aewebservices71 WSDL. However, all TeamTrack Web services and earlier Business Mashups Web services are still compatible with this release. These prior Web services include ttwebservices and aewebservices70.

Business Mashups provides sample programs written in C# that interact with the Business Mashups Web services API. These samples are discussed in more detail in Chapter 5: Sample Programs [page 187].

Business Mashups also provides an additional WSDL that can be called to read mashup configuration data from an auxiliary table to be used in service orchestrations. The aewebservices71getonly.wsdl can be found in the following location:

```
\Business Mashups\Application Engine\webservices\bin
```

Note: The aewebservices71getonly.wsdl does not expose any calls to alter data, but does include the GetVersion, GetTables, GetItem, GetItems, GetItemsByQuery, and Logout calls. These calls are discussed in more detail in Calls Available [page 19].

Web Service and Script Execution Order

The Application Engine executes Web service functions (see note below), Mashup Script scripts, transition attribute scripts, transition actions and state actions, and mashup events in the following order:

1. Web service function for the pre-transition context
2. Mashup Script for the pre-transition context
3. Transition attribute scripts for the pre-transition context
4. Transition executed by users.
5. Mashup Script for the post-transition context
6. Transition attribute scripts for the post-transition context
7. Web service function for the post-transition context
8. Mashup Script for the post-state context
9. Web Service function for the post-state context

10. Mashup Script for the pre-state context
11. Web Service function for pre-state context
12. Transition actions
13. Mashup events are emitted
14. State actions are performed

Note: An orchestration workflow with a reply is treated the same as a Web service function, in terms of when it's executed.

Error Message Logs

All Web service error messages are recorded in the Event Log on the Business Mashups Web server. You can view the messages in the Event Viewer, along with the Business Mashups Web server messages.

If IIS uses Anonynous Access as its authentication model, you will need to remove the IUSR account from the Guests group and add it to another group (like Users). This will allow the Web services API to write messages to the System Event Log.

Chapter 3: Web Services API Reference

This section provides an overview of all Business Mashups Web services calls, arguments, and responses. For more information on Calling out to Web services, see the *Serena® Mashup Administrator Guide*.

- Calls Available [page 19]
- Arguments [page 101]
- Responses [page 135]

Calls Available

This section describes the Web service API calls that are available in Business Mashups. The calls represent the Web service operations that are invoked from a client and performed on the Application Engine Web Server. The calls receive one or more arguments from the client, perform an operation on the server, and return an XML response to the client when applicable.

The following table lists all supported calls in alphabetical order, followed by a brief description of each operation. Select a call to view detailed information including:

- **Description**

  A brief description of the call.

- **Arguments**

  A table describing the arguments for each call. Both simple and complex types are listed for each argument. For each complex type argument, you can click the argument name for a detailed description. For more information on these types of arguments, see Arguments [page 101].

- **Response**

  A brief description of the call's response. For each complex type response, you can click the response name for a detailed description. For more information on these types of responses, see Responses [page 135].

- **Usage**

  Any notes, additional details, and concerns regarding the call are addressed here.

- **Faults**

  Possible error values are listed here.

- **XML**

  An example of the actual XML payload being sent is displayed here. The XML not only shows the call and its respective elements, you can also see detailed examples of each argument and how to format the expected data.

Calls Available

Call	Description
CreateAuxItem [page 23]	This service creates an auxiliary item in the specified table using the data supplied.
CreateAuxItemsWithName [page 25]	This service creates multiple new auxiliary items using the database name of the table.
CreateAuxItemWithName [page 28]	This service creates a new auxiliary item using the database name of the table.
CreateAuxItems [page 30]	This service creates multiple auxiliary items within the same table.
CreateFileAttachment [page 33]	This service creates a new file attachment associated with an item.
CreatePrimaryItem [page 34]	This service submits an item into the specified project using the data supplied.
CreatePrimaryItemsWithName [page 40]	This service creates multiple primary items using the fully qualified project name.
CreatePrimaryItemWithName [page 42]	This service creates a new primary item using the fully qualified project name.
CreatePrimaryItems [page 37]	This service submits multiple primary items within the same project using the data supplied.
DeleteAttachment [page 45]	This service deletes an existing attachment, which can be a note, item link, URL attachment, or file attachment.
DeleteItem [page 46]	This service uses the delete transition to delete the specified item.
DeleteItems [page 47]	This service uses the delete transition to delete multiple items.
DeleteItemsByQuery [page 48]	This service deletes all the items that match the specified *where* clause.
DeleteItemsByQueryWithName [page 49]	This service deletes all the items that match the specified *where* clause using the database table name.

Call	Description
DeleteMashup [page 51]	This service deletes a specified mashup.
Export [page 52]	This service exports an application as a file attachment in Zip format.
GenerateUUID [page 53]	This service generates a new globally unique UUID and returns it.
GetApplications [page 54]	This service returns the name, description, and UUID of the applications available in an Application Engine database.
GetAvailableSubmitTransitions [page 54]	This service returns all submit transitions for the specified project.
GetAvailableSubmitTransitionsWithName [page 55]	This service returns all submit transitions for the specified project using the fully qualified project name.
GetAvailableTransitions [page 57]	This service returns a list of available transitions for the specified item.
GetAvailableTransitionsWithStateIDs [page 58]	This service returns a list of available transitions for the specified item. The response includes internal IDs for the "to" and "from" states.
GetFileAttachment [page 59]	This service gets an existing file attachment.
Getitem [page 60]	This service returns the item specified, based on user privileges.
Getitems [page 61]	This service returns multiple items, based on user privileges.
GetItemsByQuery [page 63]	This service returns multiple items found using a *where* clause and an *order by* clause to determine the set of items returned.
GetItemsByQueryWithName [page 66]	This service is the same as GetItemsByQuery, but it allows you to specify a table database name instead of the table ID.

Call	Description
GetReports [page 69]	This service returns a list of reports within a specified range, limited by one or more optional filters.
GetSolutions [page 70]	This service returns a list of solutions that can be accessed by the user.
GetSolutionsWithUniqueName [page 71]	This service returns a list of solutions that can be accessed by the user. The response contains the unique name for the solution.
GetSubmitProjects [page 72]	This service returns a list of projects into which the user can submit.
GetSubmitProjectsWithName [page 73]	This service returns a list of projects into which the user can submit, using the database name of the table.
GetTables [page 74]	This service returns a list of tables that can be accessed by the user.
GetTablesWithName [page 75]	This service returns a list of tables that can be accessed by the user given the solution name.
GetUser [page 76]	This services returns basic information about a user, including name, e-mail address, and time zone.
GetUserWithPreferences [page 79]	This services returns basic information about a user, including name, e-mail address, date preference, time preference, and time zone.
GetUserPrivileges [page 77]	This services returns a list of privileges for a specified user.
GetVersion [page 80]	This service returns the Business Mashups version and build number.
HasUserPrivilege [page 80]	This service checks for a specified privilege by name.
Import [page 82]	This service imports an application.
ImportStatus [page 83]	This service returns the status of a specified Import operation.

Call	Description
IsUserValid [page 84]	This service determines whether a specified user is valid.
Logout [page 85]	This service releases any licenses and resources associated with the session.
RunReport [page 86]	This service runs a specified report, given the proper privileges.
UpdateFileAttachment [page 88]	This service updates an existing file attachment for a specified item.
UpdateItem [page 90]	This service updates an item using either a specified transition or the default update transition.
UpdateItems [page 93]	This service updates multiple items.
UpdateItemsWithName [page 96]	This service updates existing items using the specified transition name.
UpdateItemWithName [page 99]	This service updates an existing item using the specified transition name.

CreateAuxItem

Description

This service creates an auxiliary item in the specified table using the data supplied.

Arguments

Argument	Type	Description
auth (optional)	Auth [page 104]	The *Auth* type supplies credentials and optionally, a host name for licensing. The userId and password can be specified with HTTP BASIC or WS-SECURITY instead.
tableID (required)	integer	The table identifier (TS_ID) of the auxiliary table in TS_TABLES where the item will be created.
item (required)	TTItem [page 123]	The *TTItem* type holds the generic data for an item.
responseOptions (optional)	string	If provided, this parameter specifies which parts of an item should be returned in the response.

Response

TTItem is returned. The auxiliary item with updated item data, showing the unique TS_ID of the record and TS_ID of the table to which it was added are displayed. The new UUID for the created item is also returned. For more detail, see TTItem [page 174].

Usage

The CreateAuxItem call provides a method to add a single new record to a given auxiliary table. If the call fails, a new item will not be created. You can add the new record to both custom and system auxiliary tables, given the proper privileges. To create multiple auxiliary items at once, use CreateAuxItems [page 30].

To create notes, item links, and URL attachments on the new auxiliary item, add records to the lists that are defined in TTItem. To create a file attachment, see CreateFileAttachment [page 33].

Tip: You must have the itemID (tableID:internal-item-id) for the auxiliary item prior to attaching a file to it since itemID is required for CreateFileAttachment.

Use the optional responseOptions parameter to limit the data returned for a given item. In responseOptions, you specify the sections of an item that should be returned. The sections that aren't specified are not included in the response. For example, if the items you are creating have a large number of item links, notes, URL and file attachments that don't need to be returned in the response, you can use a comma-separated list in the responseOptions parameter to return only the sections of an item you want. For example:

```
<urn:responseOptions>SECTION:FIXED,SECTION:EXTENDED</urn:responseOptions>
```

This will ensure that only the fixed and extendedFieldList sections are returned. Here are some of the possible sections you can specify:

- SECTION:NONE

  No sections are returned. The genericItem portion of TTItem is the only part returned. The genericItem section is always returned, regardless of the value or values specified in responseOptions.

- SECTION:ALL

  All sections are returned. This is the default value for responseOptions. If no value is specified in responseOptions, ALL is the assumed value.

- SECTION:FIXED

  All parameters in TTItem (from <urn:itemType> to <urn:url>) prior to the extendedFieldList are returned.

- SECTION:EXTENDED

  Returns the entire extendedFieldList section of TTItem.

- SECTION:ATTACHMENTS

  Returns all of the attachment sections of TTItem. You can return the next four sections simply by specifying:

```
<urn:responseOptions>SECTION:ATTACHMENTS</urn:responseOptions>
```

- SECTION:NOTES

  Returns the entire noteList section of TTItem.

- SECTION:ITEMLINKS

  Returns the entire itemLinkList section of TTItem.
- SECTION:URLATTACHMENTS

  Returns the entire urlAttachmentList section of TTItem.
- SECTION:FILEATTACHMENTS

  Returns the entire fileAttachmentList secton of TTItem.

Note: If you specify SECTION:NONE after other sections, those preceding sections will not be returned. For example, SECTION:FIXED,SECTION:NONE,SECTION:EXTENDED will only return the extendedFieldList section.

Faults

- Invalid database pointer.
- The table ID is not valid.
- The user lacks sufficient permission.
- Creating the record fails.
- The submit transition fails.
- Reading the item fails.

XML

The following XML is a snippet of the payload being sent with CreateAuxItem.

```
<urn:CreateAuxItem>
    <urn:auth>
        <urn:userId>admin</urn:userId>
        <urn:password/>
        <urn:hostname>localhost</urn:hostname>
    </urn:auth>
    <urn:tableID>1002</urn:tableID>
    <urn:item>
        <urn:title>Title-Test1-Simple</urn:title>
        <urn:description>A Description</urn:description>
        <urn:activeInactive>true</urn:activeInactive>
    </urn:item>
    <urn:responseOptions>SECTION:FIXED,SECTION:EXTENDED</urn:responseOptions>
</urn:CreateAuxItem>
```

CreateAuxItemsWithName

Description

This service creates multiple new auxiliary items using the database name of the table.

Arguments

Argument	Type	Description
auth (optional)	Auth [page 104]	The *Auth* type supplies credentials and optionally, a host name for licensing. The userId and password can be specified with HTTP BASIC or WS-SECURITY instead.
tableDBName (required)	string	The database name of the table. For example, to create a new item in the Companies table, use TS_COMPANIES.
itemList (required)	TTItem [page 123]	The list of items to be created. The *TTItem* types hold the generic data for each item.
responseOptions (optional)	string	If provided, this parameter specifies which parts of an item should be returned in the response.

Response

TTItem is returned, one for each item created. The auxiliary items with updated item data, showing the unique TS_IDs of each record and TS_IDs of the table to which they were added are displayed. The new UUIDs for the created items are also returned. For more detail, see TTItem [page 174].

Usage

The CreateAuxItemsWithName call provides a method to add multipile new records to a given auxiliary table. If any failures occur, each successive error message is appended to the string that is returned. Multiple error messages are separated by a single newline. Failures do not result in a return before all items have been processed. You can add new records to both custom and system auxiliary tables, given the proper privileges. To create only a single auxiliary item using the database table name, use CreateAuxItemWithName [page 28].

To create notes, item links, and URL attachments on the new auxiliary items, add records to the lists that are defined in TTItem. To create a file attachment, see CreateFileAttachment [page 33].

Tip: You must have the itemID (tableID:internal-item-id) for the auxiliary item prior to attaching a file to it since itemID is required for CreateFileAttachment.

Use the optional responseOptions parameter to limit the data returned for a given item. In responseOptions, you specify the sections of an item that should be returned. The sections that aren't specified are not included in the response. For example, if the items you are creating have a large number of item links, notes, URL and file attachments that don't need to be returned in the response, you can use a comma-separated list in the responseOptions parameter to return only the sections of an item you want. For example:

```
<urn:responseOptions>SECTION:FIXED,SECTION:EXTENDED</urn:responseOptions>
```

This will ensure that only the fixed and extendedFieldList sections are returned. Here are some of the possible sections you can specify:

- SECTION:NONE

  No sections are returned. The genericItem portion of TTItem is the only part returned. The genericItem section is always returned, regardless of the value or values specified in responseOptions.

- SECTION:ALL

  All sections are returned. This is the default value for responseOptions. If no value is specified in responseOptions, ALL is the assumed value.

- SECTION:FIXED

  All parameters in TTItem (from <urn:itemType> to <urn:url>) prior to the extendedFieldList are returned.

- SECTION:EXTENDED

  Returns the entire extendedFieldList section of TTItem.

- SECTION:ATTACHMENTS

  Returns all of the attachment sections of TTItem. You can return the next four sections simply by specifying:

```
<urn:responseOptions>SECTION:ATTACHMENTS</urn:responseOptions>
```

- SECTION:NOTES

  Returns the entire noteList section of TTItem.

- SECTION:ITEMLINKS

  Returns the entire itemLinkList section of TTItem.

- SECTION:URLATTACHMENTS

  Returns the entire urlAttachmentList section of TTItem.

- SECTION:FILEATTACHMENTS

  Returns the entire fileAttachmentList secton of TTItem.

Note: If you specify SECTION:NONE after other sections, those preceding sections will not be returned. For example, SECTION:FIXED,SECTION:NONE,SECTION:EXTENDED will only return the extendedFieldList section.

Faults

- Invalid database pointer.
- The table ID is not valid.
- The user lacks sufficient permission.
- Creating the record fails.
- The submit transition fails.
- Reading the item fails.

XML

The following XML is a snippet of the payload being sent with CreateAuxItemsWithName.

```
<urn:CreateAuxItemsWithName>
    <urn:auth>
        <urn:userId>admin</urn:userId>
        <urn:password/>
        <urn:hostname>localhost</urn:hostname>
    </urn:auth>
    <urn:tableDBName>TTS_ASSETS</urn:tableDBName>
    <urn:itemList>
        <urn:title>Title-Test1-Simple</urn:title>
        <urn:description>A Description</urn:description>
        <urn:activeInactive>true</urn:activeInactive>
    </urn:itemList>
    <urn:itemList>
        <urn:title>Title-Test2-Simple</urn:title>
        <urn:description>A Description</urn:description>
        <urn:activeInactive>true</urn:activeInactive>
    </urn:itemList>
    <urn:responseOptions>SECTION:FIXED,SECTION:EXTENDED</urn:responseOptions>
</urn:CreateAuxItemsWithName>
```

CreateAuxItemWithName

Description

This service creates an new auxiliary item using the database name of the table.

Arguments

Argument	Type	Description
auth (optional)	Auth [page 104]	The *Auth* type supplies credentials and optionally, a host name for licensing. The userId and password can be specified with HTTP BASIC or WS-SECURITY instead.
tableDBName (required)	string	The database name of the table. For example, to create a new item in the Companies table, use TS_COMPANIES.
item (required)	TTItem [page 123]	The *TTItem* type holds the generic data for an item.
responseOptions (optional)	string	If provided, this parameter specifies which parts of an item should be returned in the response.

Response

TTItem is returned. The auxiliary item with updated item data, showing the unique TS_ID of the record and TS_ID of the table to which it was added are displayed. The new UUID for the created item is also returned. For more detail, see TTItem [page 174].

Usage

The CreateAuxItemWithName call provides a method to add a single new record to a given auxiliary table. If the call fails, a new item will not be created. You can add the new

record to both custom and system auxiliary tables, given the proper privileges. To create multiple auxiliary items at once using the database table name, use CreateAuxItemsWithName [page 25].

To create notes, item links, and URL attachments on the new auxiliary item, add records to the lists that are defined in TTItem. To create a file attachment, see CreateFileAttachment [page 33].

Tip: You must have the itemID (tableID:internal-item-id) for the auxiliary item prior to attaching a file to it since itemID is required for CreateFileAttachment.

Use the optional responseOptions parameter to limit the data returned for a given item. In responseOptions, you specify the sections of an item that should be returned. The sections that aren't specified are not included in the response. For example, if the items you are creating have a large number of item links, notes, URL and file attachments that don't need to be returned in the response, you can use a comma-separated list in the responseOptions parameter to return only the sections of an item you want. For example:

```
<urn:responseOptions>SECTION:FIXED,SECTION:EXTENDED</urn:responseOptions>
```

This will ensure that only the fixed and extendedFieldList sections are returned. Here are some of the possible sections you can specify:

- SECTION:NONE

  No sections are returned. The genericItem portion of TTItem is the only part returned. The genericItem section is always returned, regardless of the value or values specified in responseOptions.

- SECTION:ALL

  All sections are returned. This is the default value for responseOptions. If no value is specified in responseOptions, ALL is the assumed value.

- SECTION:FIXED

  All parameters in TTItem (from <urn:itemType> to <urn:url>) prior to the extendedFieldList are returned.

- SECTION:EXTENDED

  Returns the entire extendedFieldList section of TTItem.

- SECTION:ATTACHMENTS

  Returns all of the attachment sections of TTItem. You can return the next four sections simply by specifying:

```
<urn:responseOptions>SECTION:ATTACHMENTS</urn:responseOptions>
```

- SECTION:NOTES

  Returns the entire noteList section of TTItem.

- SECTION:ITEMLINKS

  Returns the entire itemLinkList section of TTItem.

- SECTION:URLATTACHMENTS

  Returns the entire urlAttachmentList section of TTItem.

- SECTION:FILEATTACHMENTS

  Returns the entire fileAttachmentList secton of TTItem.

Note: If you specify SECTION:NONE after other sections, those preceding sections will not be returned. For example, SECTION:FIXED,SECTION:NONE,SECTION:EXTENDED will only return the extendedFieldList section.

Faults

- Invalid database pointer.
- The table ID is not valid.
- The user lacks sufficient permission.
- Creating the record fails.
- The submit transition fails.
- Reading the item fails.

XML

The following XML is a snippet of the payload being sent with CreateAuxItemWithName.

```
<urn:CreateAuxItemWithName>
    <urn:auth>
        <urn:userId>admin</urn:userId>
        <urn:password/>
        <urn:hostname>localhost</urn:hostname>
    </urn:auth>
    <urn:tableDBName>TTS_ASSETS</urn:tableDBName>
    <urn:item>
        <urn:title>Title-Test1-Simple</urn:title>
        <urn:description>A Description</urn:description>
        <urn:activeInactive>true</urn:activeInactive>
    </urn:item>
    <urn:responseOptions>SECTION:FIXED,SECTION:EXTENDED</urn:responseOptions>
</urn:CreateAuxItemWithName>
```

CreateAuxItems

Description

This service creates multiple auxiliary items within the same table.

Arguments

Argument	Type	Description
auth (optional)	Auth [page 104]	The *Auth* type supplies credentials and optionally, a host name for licensing. The userId and password can be specified with HTTP BASIC or WS-SECURITY instead.
tableID (required)	integer	The table identifier (TS_ID) of the auxiliary table in TS_TABLES where the item will be created.

Argument	Type	Description
itemList (required)	TTItem [page 123]	The list of items to be created. The *TTItem* types hold the generic data for each item.
responseOptions (optional)	string	If provided, this parameter specifies which parts of an item should be returned in the response.

Response

TTItem is returned, one for each item created. The auxiliary items with updated item data, showing the unique TS_IDs of each record and TS_IDs of the table to which they were added are displayed. The new UUIDs for the created items are also returned. For more detail, see TTItem [page 174]

Usage

The CreateAuxItems call provides a method to add multipile new records to a given auxiliary table. If any failures occur, each successive error message is appended to the string that is returned. Multiple error messages are separated by a single newline. Failures do not result in a return before all items have been processed. You can add new records to both custom and system auxiliary tables, given the proper privileges. To create only a single auxiliary item, use CreateAuxItem [page 23].

To create notes, item links, and URL attachments on the new auxiliary items, add records to the lists that are defined in TTItem. To create a file attachment, see CreateFileAttachment [page 33].

Tip: You must have the itemID (tableID:internal-item-id) for the auxiliary item prior to attaching a file to it since itemID is required for CreateFileAttachment.

Use the optional responseOptions parameter to limit the data returned for a given item. In responseOptions, you specify the sections of an item that should be returned. The sections that aren't specified are not included in the response. For example, if the items you are creating have a large number of item links, notes, URL and file attachments that don't need to be returned in the response, you can use a comma-separated list in the responseOptions parameter to return only the sections of an item you want. For example:

```
<urn:responseOptions>SECTION:FIXED,SECTION:EXTENDED</urn:responseOptions>
```

This will ensure that only the fixed and extendedFieldList sections are returned. Here are some of the possible sections you can specify:

- SECTION:NONE

  No sections are returned. The genericItem portion of TTItem is the only part returned. The genericItem section is always returned, regardless of the value or values specified in responseOptions.

- SECTION:ALL

  All sections are returned. This is the default value for responseOptions. If no value is specified in responseOptions, ALL is the assumed value.

- SECTION:FIXED

All parameters in TTItem (from <urn:itemType> to <urn:url>) prior to the extendedFieldList are returned.

- SECTION:EXTENDED

  Returns the entire extendedFieldList section of TTItem.

- SECTION:ATTACHMENTS

  Returns all of the attachment sections of TTItem. You can return the next four sections simply by specifying:

```
<urn:responseOptions>SECTION:ATTACHMENTS</urn:responseOptions>
```

- SECTION:NOTES

  Returns the entire noteList section of TTItem.

- SECTION:ITEMLINKS

  Returns the entire itemLinkList section of TTItem.

- SECTION:URLATTACHMENTS

  Returns the entire urlAttachmentList section of TTItem.

- SECTION:FILEATTACHMENTS

  Returns the entire fileAttachmentList secton of TTItem.

Note: If you specify SECTION:NONE after other sections, those preceding sections will not be returned. For example, SECTION:FIXED,SECTION:NONE,SECTION:EXTENDED will only return the extendedFieldList section.

Faults

- Invalid database pointer.
- The table ID is not valid.
- The user lacks sufficient permission.
- Creating the record fails.
- The submit transition fails.
- Reading the item fails.

XML

The following XML is a snippet of the payload being sent with CreateAuxItems.

```
<urn:CreateAuxItems>
    <urn:auth>
        <urn:userId>admin</urn:userId>
        <urn:password/>
        <urn:hostname>localhost</urn:hostname>
    </urn:auth>
    <urn:tableID>1002</urn:tableID>
    <urn:itemList>
        <urn:title>Title-Test1A</urn:title>
        <urn:description>A Description</urn:description>
```

```
            <urn:activeInactive>true</urn:activeInactive>
        </urn:itemList>
        <urn:itemList>
            <urn:title>Title-Test1B</urn:title>
            <urn:description>Another Description</urn:description>
            <urn:activeInactive>true</urn:activeInactive>
        </urn:itemList>
        <urn:responseOptions>SECTION:FIXED,SECTION:EXTENDED</urn:responseOptions>
    </urn:CreateAuxItems>
```

CreateFileAttachment

Description

This service creates a new file attachment associated with an item.

Arguments

Argument	Type	Description
auth (optional)	Auth [page 104]	The *Auth* type supplies credentials and optionally, a host name for licensing. The userId and password can be specified with HTTP BASIC or WS-SECURITY instead.
itemID (required)	string	The item identifier. You must specify the item in tableid:internal-item-id format. For example, 1009:50; where 1009 is the TS_ID for the table in TS_TABLES and 50 is the TS_ID for the item in that primary/auxiliary table.
attachmentContents (required)	FileAttachmentContents [page 106]	The file attachment details and content.  **Note:** attachmentContents is of type FileAttachmentContents, but also includes attachment detail found in FileAttachment. The XML example below shows the parameters from both FileAttachment and FileAttachmentContents. See FileAttachment [page 143] for more information.

Response

FileAttachment is returned. The newly added file attachment details are returned (not the content itself). For more detail, see FileAttachment [page 143].

Usage

The CreateFileAttachment call provides a method to add a single attachment to an auxiliary or primary item, given the proper privileges. If the call fails, a file attachment will not be added. To create multiple file attachments for a single item, CreateFileAttachment must be called for each attachment.

Note: The file to be attached is sent as a base64 encoded attachment.

Faults

- Invalid database pointer.
- The item ID is not valid.
- The user lacks sufficient permission.
- Creating the record fails.
- Failed to create the attachment.

XML

The following XML is a snippet of the payload being sent with CreateFileAttachment.

```
<urn:CreateFileAttachment>
    <urn:auth>
        <urn:userId>admin</urn:userId>
        <urn:password></urn:password>
        <urn:hostname>localhost</urn:hostname>
    </urn:auth>
    <urn:itemID>1000:81</urn:itemID>
    <urn:attachmentContents>
        <urn:id>16</urn:id>
        <urn:name>pdf_doc</urn:name>
        <urn:fileName>relnotes.pdf</urn:fileName>
        <urn:showAsImage>false</urn:showAsImage>
        <urn:modificationDateTime></urn:modificationDateTime>
        <urn:url></urn:url>
        <urn:accessType>ATTACHACCESS-UNRESTRICTED</urn:accessType>
        <urn:checksum></urn:checksum>
        <urn:encodedContents>encoded_content_here</urn:encodedContents>
    </urn:attachmentContents>
</urn:CreateFileAttachment>
```

CreatePrimaryItem

Description

This service submits an item into the specified project using the data supplied.

Arguments

Argument	Type	Description
auth (optional)	Auth [page 104]	The *Auth* type supplies credentials and optionally, a host name for licensing. The userId and password can be specified with HTTP BASIC or WS-SECURITY instead.
projectID (required)	integer	The project identifier (TS_ID) of the project in TS_PROJECTS where the item will be created.
item (optional)	TTItem [page 123]	The *TTItem* type holds the generic data for an item.
submitTransID (optional)	integer	If provided, specifies a Submit Transition ID (TS_ID from TS_TRANSITIONS) to use when creating the item.
responseOptions (optional)	string	If provided, this parameter specifies which parts of an item should be returned in the response.

Response

TTItem is returned. The primary item with updated item data, showing the unique TS_ID of the record and TS_ID of the table to which it was added are displayed. The new UUID for the created item is also returned. For more detail, see TTItem [page 174].

Usage

The CreatePrimaryItem call provides a method to add a single new record to a given primary table. If the call fails, a new item will not be created. You can add the new records to both custom and system primary tables, given the proper privileges. To create multiple primary items at once, use CreatePrimaryItems [page 37].

If you specify a specific submit transition for the call to use, that transition needs to be a valid transition for the project you've specified.

Use the optional responseOptions parameter to limit the data returned for a given item. In responseOptions, you specify the sections of an item that should be returned. The sections that aren't specified are not included in the response. For example, if the items you are creating have a large number of item links, notes, URL and file attachments that don't need to be returned in the response, you can use a comma-separated list in the responseOptions parameter to return only the sections of an item you want. For example:

```
<urn:responseOptions>SECTION:FIXED,SECTION:EXTENDED</urn:responseOptions>
```

This will ensure that only the fixed and extendedFieldList sections are returned. Here are some of the possible sections you can specify:

- SECTION:NONE

  No sections are returned. The genericItem portion of TTItem is the only part returned. The genericItem section is always returned, regardless of the value or values specified in responseOptions.

- SECTION:ALL

  All sections are returned. This is the default value for responseOptions. If no value is specified in responseOptions, ALL is the assumed value.

- SECTION:FIXED

  All parameters in TTItem (from <urn:itemType> to <urn:url>) prior to the extendedFieldList are returned.

- SECTION:EXTENDED

  Returns the entire extendedFieldList section of TTItem.

- SECTION:ATTACHMENTS

  Returns all of the attachment sections of TTItem. You can return the next four sections simply by specifying:

```
<urn:responseOptions>SECTION:ATTACHMENTS</urn:responseOptions>
```

- SECTION:NOTES

  Returns the entire noteList section of TTItem.

- SECTION:ITEMLINKS

  Returns the entire itemLinkList section of TTItem.

- SECTION:URLATTACHMENTS

  Returns the entire urlAttachmentList section of TTItem.

- SECTION:FILEATTACHMENTS

  Returns the entire fileAttachmentList secton of TTItem.

Note: If you specify SECTION:NONE after other sections, those preceding sections will not be returned. For example, SECTION:FIXED,SECTION:NONE,SECTION:EXTENDED will only return the extendedFieldList section.

To create notes, item links, and URL attachments on the new primary item, add records to the lists that are defined in TTItem. To create a file attachment, see CreateFileAttachment [page 33].

Faults

- Invalid database pointer.
- The project ID is not valid.
- The user lacks sufficient permission.

- Creating the record fails.
- The submit transition fails.
- Reading the item fails.

XML

The following XML is a snippet of the payload being sent with CreatePrimaryItem.

```
<urn:CreatePrimaryItem>
    <urn:auth>
        <urn:userId>admin</urn:userId>
        <urn:password></urn:password>
        <urn:hostname>localhost</urn:hostname>
    </urn:auth>
    <urn:projectID>29</urn:projectID>
    <urn:item>
        <urn:classification>Change Requests</urn:classification>
        <urn:title>Title-CreatePrimItem-Test1</urn:title>
        <urn:description>A Description</urn:description>
        <urn:activeInactive>true</urn:activeInactive>
    </urn:item>
    <urn:submitTransID>2</urn:submitTransID>
    <urn:responseOptions>SECTION:FIXED,SECTION:EXTENDED</urn:responseOptions>
</urn:CreatePrimaryItem>
```

CreatePrimaryItems

Description

This service submits multiple primary items within the same project using the data supplied.

Arguments

Argument	Type	Description
auth (optional)	Auth [page 104]	The *Auth* type supplies credentials and optionally, a host name for licensing. The userId and password can be specified with HTTP BASIC or WS-SECURITY instead.
projectID (required)	integer	The project identifier (TS_ID) of the project in TS_PROJECTS where the items will be created.
itemList (required)	TTItem [page 123]	The list of items to be created. The *TTItem* types hold the generic data for each item.
submitTransID (optional)	integer	If provided, specifies a Submit Transition ID (TS_ID from TS_TRANSITIONS) to use when creating the items.
responseOptions (optional)	string	If provided, this parameter specifies which parts of an item should be returned in the response.

Response

TTItem is returned, one for each item created. The primary items with updated item data, showing the unique TS_IDs of each record and TS_IDs of the table to which they were added are displayed. The new UUIDs for the created items are also returned. For more detail, see TTItem [page 174].

Usage

The CreatePrimaryItems call provides a method to add multipile new records to a given primary table. If any failures occur, each successive error message is appended to the string that is returned. Multiple error messages are separated by a single newline. Failures do not result in a return before all items have been processed. You can add new records to both custom and system primary tables, given the proper privileges. To create only a single primary item, use CreatePrimaryItem [page 34].

To create notes, item links, and URL attachments on the new primary items, add records to the lists that are defined in TTItem. To create a file attachment, see CreateFileAttachment [page 33].

Tip: You must have the itemID (tableID:internal-item-id) for the primary item prior to attaching a file to it since itemID is required for CreateFileAttachment.

Use the optional responseOptions parameter to limit the data returned for a given item. In responseOptions, you specify the sections of an item that should be returned. The sections that aren't specified are not included in the response. For example, if the items you are creating have a large number of item links, notes, URL and file attachments that don't need to be returned in the response, you can use a comma-separated list in the responseOptions parameter to return only the sections of an item you want. For example:

```
<urn:responseOptions>SECTION:FIXED,SECTION:EXTENDED</urn:responseOptions>
```

This will ensure that only the fixed and extendedFieldList sections are returned. Here are some of the possible sections you can specify:

- SECTION:NONE

  No sections are returned. The genericItem portion of TTItem is the only part returned. The genericItem section is always returned, regardless of the value or values specified in responseOptions.

- SECTION:ALL

  All sections are returned. This is the default value for responseOptions. If no value is specified in responseOptions, ALL is the assumed value.

- SECTION:FIXED

  All parameters in TTItem (from <urn:itemType> to <urn:url>) prior to the extendedFieldList are returned.

- SECTION:EXTENDED

  Returns the entire extendedFieldList section of TTItem.

- SECTION:ATTACHMENTS

  Returns all of the attachment sections of TTItem. You can return the next four sections simply by specifying:

```
<urn:responseOptions>SECTION:ATTACHMENTS</urn:responseOptions>
```

- SECTION:NOTES

  Returns the entire noteList section of TTItem.
- SECTION:ITEMLINKS

  Returns the entire itemLinkList section of TTItem.
- SECTION:URLATTACHMENTS

  Returns the entire urlAttachmentList section of TTItem.
- SECTION:FILEATTACHMENTS

  Returns the entire fileAttachmentList secton of TTItem.

Note: If you specify SECTION:NONE after other sections, those preceding sections will not be returned. For example, SECTION:FIXED,SECTION:NONE,SECTION:EXTENDED will only return the extendedFieldList section.

Faults

- Invalid database pointer.
- The project ID is not valid.
- The user lacks sufficient permission.
- Creating the record fails.
- The submit transition fails.
- Reading the item fails.

XML

The following XML is a snippet of the payload being sent with CreatePrimaryItems.

```
<urn:CreatePrimaryItems>
    <urn:auth>
        <urn:userId>admin</urn:userId>
        <urn:password></urn:password>
        <urn:hostname>localhost</urn:hostname>
    </urn:auth>
    <urn:projectID>29</urn:projectID>
    <urn:itemList>
        <urn:title>Title-Test1A</urn:title>
        <urn:description>A Description</urn:description>
        <urn:activeInactive>true</urn:activeInactive>
    </urn:itemList>
    <urn:itemList>
        <urn:title>Title-Test1B</urn:title>
        <urn:description>A Description</urn:description>
        <urn:activeInactive>true</urn:activeInactive>
    </urn:itemList>
    <urn:submitTransID>2</urn:submitTransID>
    <urn:responseOptions>SECTION:FIXED,SECTION:EXTENDED</urn:responseOptions>
</urn:CreatePrimaryItems>
```

CreatePrimaryItemsWithName

Description

This service creates multiple primary items using the fully qualified project name.

Arguments

Argument	Type	Description
auth (optional)	Auth [page 104]	The *Auth* type supplies credentials and optionally, a host name for licensing. The userId and password can be specified with HTTP BASIC or WS-SECURITY instead.
fullyQualifiedProjectName (required)	string	The fully qualified project name where the item will be created.
itemList (required)	TTItem [page 123]	The list of items to be created. The *TTItem* types hold the generic data for each item.
submitTransName (optional)	string	If provided, specifies a Submit Transition name to use when creating the item.
responseOptions (optional)	string	If provided, this parameter specifies which parts of an item should be returned in the response.

Response

TTItem is returned, one for each item created. The primary items with updated item data, showing unique TS_IDs for each record and TS_IDs of the table to which they were added are displayed. The new UUIDs for the created items are also returned. For more detail, see TTItem [page 174].

Usage

The CreatePrimaryItemsWithName call provides a method to add multipile new records to a given primary table. If any failures occur, each successive error message is appended to the string that is returned. Multiple error messages are separated by a single newline. Failures do not result in a return before all items have been processed. You can add new records to both custom and system primary tables, given the proper privileges. To create only a single primary item using the fully qualified project name, use CreatePrimaryItemWithName [page 42].

To create notes, item links, and URL attachments on the new primary items, add records to the lists that are defined in TTItem. To create a file attachment, see CreateFileAttachment [page 33].

The fullyQualifiedProjectName argument requires that you specify the project name as it exists in the project hierarchy with respect to project inheritance. To submit into the Animation Pro project, you could specify the fully qualified project name starting with the Base Project, followed by any other projects in the inheritance chain that lead to your project where the item is to be submitted. For example:

```
Base Project||Base IDT Project||Software Development||Animation Pro
```

Note: Using the Base Project in the fully qualified name is optional. The only requirement is that you must precede your project name with any projects in the project ancestry that are associated with a workflow *other* than the Base Workflow. In the example above, you could exclude Base Project, but you would need to include the others since they are based off functional workflows and not the Base Workflow.

Use the optional responseOptions parameter to limit the data returned for a given item. In responseOptions, you specify the sections of an item that should be returned. The sections that aren't specified are not included in the response. For example, if the items you are creating have a large number of item links, notes, URL and file attachments that don't need to be returned in the response, you can use a comma-separated list in the responseOptions parameter to return only the sections of an item you want. For example:

```
<urn:responseOptions>SECTION:FIXED,SECTION:EXTENDED</urn:responseOptions>
```

This will ensure that only the fixed and extendedFieldList sections are returned. Here are some of the possible sections you can specify:

- SECTION:NONE

  No sections are returned. The genericItem portion of TTItem is the only part returned. The genericItem section is always returned, regardless of the value or values specified in responseOptions.

- SECTION:ALL

  All sections are returned. This is the default value for responseOptions. If no value is specified in responseOptions, ALL is the assumed value.

- SECTION:FIXED

  All parameters in TTItem (from <urn:itemType> to <urn:url>) prior to the extendedFieldList are returned.

- SECTION:EXTENDED

  Returns the entire extendedFieldList section of TTItem.

- SECTION:ATTACHMENTS

  Returns all of the attachment sections of TTItem. You can return the next four sections simply by specifying:

```
<urn:responseOptions>SECTION:ATTACHMENTS</urn:responseOptions>
```

- SECTION:NOTES

  Returns the entire noteList section of TTItem.

- SECTION:ITEMLINKS

  Returns the entire itemLinkList section of TTItem.

- SECTION:URLATTACHMENTS

  Returns the entire urlAttachmentList section of TTItem.

- SECTION:FILEATTACHMENTS

Returns the entire fileAttachmentList secton of TTItem.

Note: If you specify SECTION:NONE after other sections, those preceding sections will not be returned. For example, SECTION:FIXED,SECTION:NONE,SECTION:EXTENDED will only return the extendedFieldList section.

Faults

- Invalid database pointer.
- The project ID is not valid.
- The user lacks sufficient permission.
- Creating the record fails.
- The submit transition fails.
- Reading the item fails.

XML

The following XML is a snippet of the payload being sent with CreatePrimaryItemsWithName.

```
<urn:CreatePrimaryItemsWithName>
    <urn:auth>
        <urn:userId>admin</urn:userId>
        <urn:password></urn:password>
        <urn:hostname>localhost</urn:hostname>
    </urn:auth>
    <urn:fullyQualifiedProjectName>Your Company Name||Issues||
    →Change Requests</urn:fullyQualifiedProjectName>
    <urn:itemList>
        <urn:classification>Change Requests</urn:classification>
        <urn:title>Title-Test1</urn:title>
        <urn:description>A Description</urn:description>
        <urn:activeInactive>true</urn:activeInactive>
    </urn:itemList>
    <urn:itemList>
        <urn:classification>Change Requests</urn:classification>
        <urn:title>Title-Test2</urn:title>
        <urn:description>A Description</urn:description>
        <urn:activeInactive>true</urn:activeInactive>
    </urn:itemList>
    <urn:submitTransName>Submit</urn:submitTransName>
    <urn:responseOptions>SECTION:FIXED,SECTION:EXTENDED</urn:responseOptions>
</urn:CreatePrimaryItemsWithName>
```

CreatePrimaryItemWithName

Description

This service creates a new primary item using the fully qualified project name.

Arguments

Argument	Type	Description
auth (optional)	Auth [page 104]	The *Auth* type supplies credentials and optionally, a host name for licensing. The userId and password can be specified with HTTP BASIC or WS-SECURITY instead.
fullyQualifiedProjectName (required)	string	The fully qualified project name where the item will be created.
item (required)	TTItem [page 123]	The *TTItem* type holds the generic data for an item.
submitTransName (optional)	string	If provided, specifies a Submit Transition name to use when creating the item.
responseOptions (optional)	string	If provided, this parameter specifies which parts of an item should be returned in the response.

Response

TTItem is returned. The primary item with updated item data, showing the unique TS_ID of the record and TS_ID of the table to which it was added are displayed. The new UUID for the created item is also returned. For more detail, see TTItem [page 174].

Usage

The CreatePrimaryItemWithName call provides a method to add a single new record to a given primary table. If the call fails, a new item will not be created. You can add the new record to both custom and system primary tables, given the proper privileges. To create multiple primary items at once using the fully qualified project name, use CreatePrimaryItemsWithName [page 40].

To create notes, item links, and URL attachments on the new primary item, add records to the lists that are defined in TTItem. To create a file attachment, see CreateFileAttachment [page 33].

The fullyQualifiedProjectName argument requires that you specify the project name as it exists in the project hierarchy with respect to project inheritance. To submit into the Animation Pro project, you could specify the fully qualified project name starting with the Base Project, followed by any other projects in the inheritance chain that lead to your project where the item is to be submitted. For example:

```
Base Project||Base IDT Project||Software Development||Animation Pro
```

Note: Using the Base Project in the fully qualified name is optional. The only requirement is that you must precede your project name with any projects in the project ancestry that are associated with a workflow *other* than the Base Workflow. In the example above, you could exclude Base Project, but you would need to include the others since they are based off functional workflows and not the Base Workflow.

Use the optional responseOptions parameter to limit the data returned for a given item. In responseOptions, you specify the sections of an item that should be returned. The sections that aren't specified are not included in the response. For example, if the items you are creating have a large number of item links, notes, URL and file attachments that don't need to be returned in the response, you can use a comma-separated list in the responseOptions parameter to return only the sections of an item you want. For example:

```
<urn:responseOptions>SECTION:FIXED,SECTION:EXTENDED</urn:responseOptions>
```

This will ensure that only the fixed and extendedFieldList sections are returned. Here are some of the possible sections you can specify:

- SECTION:NONE

  No sections are returned. The genericItem portion of TTItem is the only part returned. The genericItem section is always returned, regardless of the value or values specified in responseOptions.

- SECTION:ALL

  All sections are returned. This is the default value for responseOptions. If no value is specified in responseOptions, ALL is the assumed value.

- SECTION:FIXED

  All parameters in TTItem (from <urn:itemType> to <urn:url>) prior to the extendedFieldList are returned.

- SECTION:EXTENDED

  Returns the entire extendedFieldList section of TTItem.

- SECTION:ATTACHMENTS

  Returns all of the attachment sections of TTItem. You can return the next four sections simply by specifying:

```
<urn:responseOptions>SECTION:ATTACHMENTS</urn:responseOptions>
```

- SECTION:NOTES

  Returns the entire noteList section of TTItem.

- SECTION:ITEMLINKS

  Returns the entire itemLinkList section of TTItem.

- SECTION:URLATTACHMENTS

  Returns the entire urlAttachmentList section of TTItem.

- SECTION:FILEATTACHMENTS

  Returns the entire fileAttachmentList secton of TTItem.

Note: If you specify SECTION:NONE after other sections, those preceding sections will not be returned. For example, SECTION:FIXED,SECTION:NONE,SECTION:EXTENDED will only return the extendedFieldList section.

Faults

- Invalid database pointer.

- The project ID is not valid.
- The user lacks sufficient permission.
- Creating the record fails.
- The submit transition fails.
- Reading the item fails.

XML

The following XML is a snippet of the payload being sent with CreatePrimaryItemWithName.

```
<urn:CreatePrimaryItemWithName>
    <urn:auth>
        <urn:userId>admin</urn:userId>
        <urn:password></urn:password>
        <urn:hostname>localhost</urn:hostname>
    </urn:auth>
    <urn:fullyQualifiedProjectName>Your Company Name||Issues||Change Requests
    </urn:fullyQualifiedProjectName>
    <urn:item>
        <urn:classification>Change Requests</urn:classification>
        <urn:title>Title-Test1</urn:title>
        <urn:description>A Description</urn:description>
        <urn:activeInactive>true</urn:activeInactive>
    </urn:item>
    <urn:submitTransName>Submit</urn:submitTransName>
    <urn:responseOptions>SECTION:FIXED,SECTION:EXTENDED</urn:responseOptions>
</urn:CreatePrimaryItemWithName>
```

DeleteAttachment

Description

This service deletes an existing attachment, which can be a note, item link, URL attachment, or file attachment.

Arguments

Argument	Type	Description
auth (optional)	Auth [page 104]	The *Auth* type supplies credentials and optionally, a host name for licensing. The userId and password can be specified with HTTP BASIC or WS-SECURITY instead.
attachmentID (required)	integer	This is the internal TS_ID of the attachment from the TS_ATTACHMENTS table.

Response

An empty XML response is returned:

```
<ae:DeleteAttachmentResponse></ae:DeleteAttachmentResponse>
```

and the specified note, item link, URL, or file attachment is deleted.

Usage

You can use Getitem [page 60] to find the TS_ID of the attachment to be deleted. If the item has any attachments, they will be listed in noteList, itemLinkList, urlAttachmentList, or fileAttachmentList.

Faults

- Invalid database pointer.
- The attachment ID is not valid.
- Creating the record fails.
- The user lacks sufficient permission.
- Failed to delete the attachment.

XML

The following XML is a snippet of the payload being sent with DeleteAttachment.

```
<urn:DeleteAttachment>
    <urn:auth>
        <urn:userId>admin</urn:userId>
        <urn:password></urn:password>
        <urn:hostname>localhost</urn:hostname>
   </urn:auth>
   <urn:attachmentID>47</urn:attachmentID>
</urn:DeleteAttachment>
```

DeleteItem

Description

This service uses the delete transition to delete the specified item.

Arguments

Argument	Type	Description
auth (optional)	Auth [page 104]	The *Auth* type supplies credentials and optionally, a host name for licensing. The userId and password can be specified with HTTP BASIC or WS-SECURITY instead.
itemID (required)	string	The item identifier. You must specify the item in tableid:internal-item-id format. For example, 1009:50; where 1009 is the TS_ID for the table in TS_TABLES and 50 is the TS_ID for the item in that primary/auxiliary table.

Response

An empty XML response is returned:

```
<ae:DeleteItemResponse></ae:DeleteItemResponse>
```

and the item is deleted using the default delete transition. Failure will not delete item.

Usage

This call permanently deletes the primary or auxiliary item you specify. Any items referring to this item will be reduced or set to (None).

Faults

- Invalid database pointer.
- The item ID is not valid.
- The user lacks sufficient permission.
- The delete transition fails to execute.

XML

The following XML is a snippet of the payload being sent with DeleteItem.

```
<urn:DeleteItem>
    <urn:auth>
        <urn:userId>admin</urn:userId>
        <urn:password></urn:password>
        <urn:hostname>localhost</urn:hostname>
    </urn:auth>
    <urn:sItemID>1006:17</urn:sItemID>
</urn:DeleteItem>
```

DeleteItems

Description

This service uses the delete transition to delete multiple items.

Arguments

Argument	Type	Description
auth (optional)	Auth [page 104]	The *Auth* type supplies credentials and optionally, a host name for licensing. The userId and password can be specified with HTTP BASIC or WS-SECURITY instead.
itemIDList (required)	string	A list of item identifiers. You must specify each item in tableid:internal-item-id format. For example, 1009:50; where 1009 is the TS_ID for the table in TS_TABLES and 50 is the TS_ID for the item in that primary or auxiliary table.

Response

An empty XML response is returned:

```
<ae:DeleteItemsResponse></ae:DeleteItemsResponse>
```

and the items are deleted using the default delete transition. Failure will not delete items.

Usage

This call permanently deletes the primary or auxiliary items you specify. Any items referring to these items will be reduced or set to (None). All items in the list are

processed. If any failures occur, each successive error message is appended to the string that is returned. Multiple error messages are separated by a single newline. Failures do not result in a return before all items have been processed.

Faults

- Invalid database pointer.
- The item ID is not valid.
- The user lacks sufficient permission.
- The delete transition fails to execute.

XML

The following XML is a snippet of the payload being sent with DeleteItems.

```
<urn:DeleteItems>
    <urn:auth>
        <urn:userId>admin</urn:userId>
        <urn:password></urn:password>
        <urn:hostname>localhost</urn:hostname>
    </urn:auth>
    <urn:itemIdList>1006:17</urn:itemIdList>
    <urn:itemIdList>1006:18</urn:itemIdList>
</urn:DeleteItems>
```

DeleteItemsByQuery

Description

This service deletes all the items that match the specified *where* clause.

Arguments

Argument	Type	Description
auth (optional)	Auth [page 104]	The *Auth* type supplies credentials and optionally, a host name for licensing. The userId and password can be specified with HTTP BASIC or WS-SECURITY instead.
tableID (required)	integer	The table identifier (TS_ID) of the table in TS_TABLES where the items will be deleted.
queryWhereClause (required)	string	An SQL statement to find items with TS_ID>0. Sending an empty *where* clause will delete all items.

Response

An empty XML response is returned:

```
<ae:DeleteItemByQueryResponse></ae:DeleteItemsByQueryResponse>
```

and the items are deleted using the default delete transition. Failure will not delete items.

Usage

This call permanently deletes the primary or auxiliary items you specify. Any items referring to these items will be reduced or set to (None). The items that are deleted are determined via the table and *where* clause. Failure stops at the failed item; items before the failure are deleted, and items after the failure are not deleted.

Note: You need only pass the conditions in the *where* clause itself. For example:

```
<urn:queryWhereClause>TS_ISSUEID LIKE '00106' OR TS_ISSUEID LIKE '00029'
 OR TS_ISSUEID LIKE '00105'</urn:queryWhereClause>
```

```
<urn:queryWhereClause>TS_ISSUEID='00038'</urn:queryWhereClause>
```

```
<urn:queryWhereClause>TS_ID='4'</urn:queryWhereClause>
```

To delete all items, send a null query:

```
<urn:queryWhereClause></urn:queryWhereClause>
```

Faults

- Invalid database pointer.
- The table ID is not valid.
- The user lacks sufficient permission.
- The delete transition fails to execute.

XML

The following XML is a snippet of the payload being sent with DeleteItemsByQuery.

```
<urn:DeleteItemsByQuery>
    <urn:auth>
        <urn:userId>admin</urn:userId>
        <urn:password></urn:password>
        <urn:hostname>localhost</urn:hostname>
    </urn:auth>
    <urn:tableID>1006</urn:tableID>
    <urn:queryWhereClause>TS_TITLE LIKE 'Test'</urn:queryWhereClause>
</urn:DeleteItemsByQuery>
```

DeleteItemsByQueryWithName

Description

This service deletes all the items that match the specified *where* clause using the database table name.

Arguments

Argument	Type	Description
auth (optional)	Auth [page 104]	The *Auth* type supplies credentials and optionally, a host name for licensing. The userId and password can be specified with HTTP BASIC or WS-SECURITY instead.
tableDBName (required)	string	The database name of the table. For example, to delete an item in the Incidents table, use TTS_INCIDENTS.
queryWhereClause (required)	string	An SQL statement to find items with TS_ID>0. Sending an empty *where* clause will delete all items.

Response

An empty XML response is returned:

```
<ae:DeleteItemByQueryWithNameResponse></ae:DeleteItemsByQueryWithNameResponse>
```

and the items are deleted using the default delete transition. Failure will not delete items.

Usage

This call permanently deletes the primary or auxiliary items you specify. Any items referring to these items will be reduced or set to (None). The items that are deleted are determined via the table and *where* clause. Failure stops at the failed item; items before the failure are deleted, and items after the failure are not deleted.

Note: You need only pass the conditions in the *where* clause itself. For example:

```
<urn:queryWhereClause>TS_ISSUEID LIKE '00106' OR TS_ISSUEID LIKE '00029'
 OR TS_ISSUEID LIKE '00105'</urn:queryWhereClause>
```

```
<urn:queryWhereClause>TS_ISSUEID='00038'</urn:queryWhereClause>
```

```
<urn:queryWhereClause>TS_ID='4'</urn:queryWhereClause>
```

To delete all items, send a null query:

```
<urn:queryWhereClause></urn:queryWhereClause>
```

Faults

- Invalid database pointer.
- The table ID is not valid.
- The user lacks sufficient permission.
- The delete transition fails to execute.

XML

The following XML is a snippet of the payload being sent with DeleteItemsByQueryWithName.

```
<urn:DeleteItemsByQueryWithName>
    <urn:auth>
        <urn:userId>admin</urn:userId>
        <urn:password></urn:password>
        <urn:hostname>localhost</urn:hostname>
    </urn:auth>
    <urn:tableDBName>TTS_INCIDENTS</urn:tableDBName>
    <urn:queryWhereClause>TS_TITLE LIKE 'Test'</urn:queryWhereClause>
</urn:DeleteItemsByQueryWithName>
```

DeleteMashup

Description

This service deletes a specified mashup and all its data, given the proper privilege.

Arguments

Argument	Type	Description
auth (optional)	Auth [page 104]	The *Auth* type supplies credentials and optionally, a host name for licensing. The userId and password can be specified with HTTP BASIC or WS-SECURITY instead.
sMashupName (required)	string	The name of the mashup to be deleted.

Response

An empty XML response is returned:

```
<ae:DeleteMashupResponse></ae:DeleteMashupResponse>
```

and the mashup is deleted. Failure will not delete mashup.

Usage

This call permanently deletes the mashup you specify.

Note: DeleteMashup not only deletes the specified mashup, but also all the data in that mashup. All tables defined by the mashup and all data in those tables will be permanently deleted. Since the data cannot be recovered, you may want to perform a backup of your database prior to deleting the mashup.

Faults

- Invalid database pointer.
- The mashup name is not valid.
- The user lacks sufficient permission.
- The delete mashup fails to execute.

XML

The following XML is a snippet of the payload being sent with DeleteMashup.

```
<urn:DeleteMashup>
    <urn:auth>
        <urn:userId>admin</urn:userId>
        <urn:password></urn:password>
        <urn:hostname>localhost</urn:hostname>
    </urn:auth>
    <urn:sMashupName>Incident Management</urn:sMashupName>
</urn:DeleteMashup>
```

Export

Description

This service exports an application as a file attachment in Zip format.

Arguments

Argument	Type	Description
auth (optional)	Auth [page 104]	The *Auth* type supplies credentials and optionally, a host name for licensing. The userId and password can be specified with HTTP BASIC or WS-SECURITY instead.
applicationID (required)	string	The application identifier (TS_ID) of the application in TS_APPLICATIONS.
xmlExportOptions (optional)	FileContents [page 107]	Contains the export options for output and filtering.

Response

FileContents is returned. The exported application file is returned via a base 64-encoded attachment. See FileContents [page 144] for more information.

Usage

If the application ID contains alphanumeric characters, it will be interpreted as the UUID for the application. The GetApplications [page 54] call should be used to obtain the desired application ID.

Faults

- Invalid database pointer.
- Creating the attachment fails.
- The user lacks sufficient permission.
- Failed to read the attachment contents.

XML

The following XML is a snippet of the payload being sent with Export.

```
<urn:Export>
    <urn:auth>
        <urn:userId>admin</urn:userId>
        <urn:password></urn:password>
        <urn:hostname>localhost</urn:hostname>
    </urn:auth>
    <urn:applicationID>9</urn:applicationID>
        <urn:xmlExportOptions>
        <urn:checksum></urn:checksum>
        <urn:encodedContents></urn:encodedContents>
    </urn:xmlExportOptions>
</urn:Export>
```

GenerateUUID

Description

This service generates a new globally unique UUID and returns it.

Arguments

Argument	Type	Description
auth (optional)	Auth [page 104]	The *Auth* type supplies credentials and optionally, a host name for licensing. The userId and password can be specified with HTTP BASIC or WS-SECURITY instead.

Response

The new UUID (string) is returned.

```
<ae:GenerateUUIDResponse>
   <ae:return>09d40cc7-c05e-4724-a79f-f0ae4d54b5bf</ae:return>
</ae:GenerateUUIDResponse>
```

Usage

This call currently has no practical usage.

Faults

- The user lacks sufficient permission.

XML

The following XML is a snippet of the payload being sent with GenerateUUID.

```
<urn:GenerateUUID>
    <urn:auth>
        <urn:userId>admin</urn:userId>
        <urn:password></urn:password>
        <urn:hostname>localhost</urn:hostname>
    </urn:auth>
</urn:GenerateUUID>
```

GetApplications

Description

This service returns the name, description, and UUID of the applications available in an Application Engine database.

Arguments

Argument	Type	Description
auth (optional)	Auth [page 104]	The *Auth* type supplies credentials and optionally, a host name for licensing. The userId and password can be specified with HTTP BASIC or WS-SECURITY instead.

Response

ApplicationData (a list of applications) will be returned in the response. Each return element shows the application's TS_ID, UUID, name, description, and application definition UUID. See ApplicationData [page 137] for more information.

Usage

This method is a good starting point if you intend to export applications from Business Mashups. The application ID needed for the Export call is returned in the response. See Export [page 52] for more information.

Faults

- Invalid database pointer.
- No applications are available for the user.

XML

The following XML is a snippet of the payload being sent with GetApplications.

```
<urn:GetApplications>
    <urn:auth>
        <urn:userId>admin</urn:userId>
        <urn:password></urn:password>
        <urn:hostname>localhost</urn:hostname>
    </urn:auth>
</urn:GetApplications>
```

GetAvailableSubmitTransitions

Description

This service returns all submit transitions for the specified project.

Arguments

Argument	Type	Description
auth (optional)	Auth [page 104]	The *Auth* type supplies credentials and optionally, a host name for licensing. The userId and password can be specified with HTTP BASIC or WS-SECURITY instead.

Argument	Type	Description
projectID (required)	integer	The identifier (TS_ID) of the project in the TS_PROJECTS table.
attributename (optional)	string	Returns only transitions that have this transition attribute (see TS_TRANSATTRS for selections). Used only for transitions created via an integration.

Response

Transition or a list of transitions is returned. The list can be empty. See the Transition [page 170] type for more detail.

Usage

The transition or transitions returned is limited by the project you specify.

Faults

- Invalid database pointer.
- The project ID is not valid.
- Reading transition attribute fails.

XML

The following XML is a snippet of the payload being sent with GetAvailableSubmitTransitions.

```
<urn:GetAvailableSubmitTransitions>
    <urn:auth>
        <urn:userId>admin</urn:userId>
        <urn:password></urn:password>
        <urn:hostname>localhost</urn:hostname>
    </urn:auth>
    <urn:projectId>19</urn:projectId>
    <urn:attributeName></urn:attributeName>
</urn:GetAvailableSubmitTransitions>
```

GetAvailableSubmitTransitionsWithName

Description

This service returns all submit transitions for the specified project using the fully qualified project name.

Arguments

Argument	Type	Description
auth (optional)	Auth [page 104]	The *Auth* type supplies credentials and optionally, a host name for licensing. The userId and password can be specified with HTTP BASIC or WS-SECURITY instead.

Argument	Type	Description
fullyQualifiedProjectName (required)	string	The fully qualified project name where the submit transition exists.
attributename (optional)	string	Returns only transitions that have this transition attribute (see TS_TRANSATTRS for selections). Used only for transitions created via an integration.

Response

Transition or a list of transitions is returned. The list can be empty. See the Transition [page 170] type for more detail.

Usage

The transition or transitions returned is limited by the project you specify in the fullyQualifiedProjectName argument.

The fullyQualifiedProjectName argument requires that you specify the project name as it exists in the project hierarchy with respect to project inheritance. To return transitions from the Animation Pro project, you could specify the fully qualified project name starting with the Base Project, followed by any other projects in the inheritance chain that lead to your project where your transition exists. For example:

```
Base Project||Base IDT Project||Software Development||Animation Pro
```

Note: Using the Base Project in the fully qualified name is optional. The only requirement is that you must precede your project name with any projects in the project ancestry that are associated with a workflow *other* than the Base Workflow. In the example above, you could exclude Base Project, but you would need to include the others since they are based off functional workflows and not the Base Workflow.

Faults

- Invalid database pointer.
- The project ID is not valid.
- Reading transition attribute fails.

XML

The following XML is a snippet of the payload being sent with GetAvailableSubmitTransitionsWithName.

```
<urn:GetAvailableSubmitTransitionsWithName>
    <urn:auth>
        <urn:userId>admin</urn:userId>
        <urn:password></urn:password>
        <urn:hostname>localhost</urn:hostname>
    </urn:auth>
    <urn:fullyQualifiedProjectName>Your Company Name||Issues||Change Requests
    </urn:fullyQualifiedProjectName>
    <urn:attributeName></urn:attributeName>
</urn:GetAvailableSubmitTransitionsWithName>
```

GetAvailableTransitions

Description

This service returns a list of available transitions for the specified item. That list can be filtered to include only the transitions that have the specified attribute.

Arguments

Argument	Type	Description
auth (optional)	Auth [page 104]	The *Auth* type supplies credentials and optionally, a host name for licensing. The userId and password can be specified with HTTP BASIC or WS-SECURITY instead.
itemID (required)	string	The item identifier. You must specify the item in tableid:internal-item-id format. For example, 1009:50; where 1009 is the TS_ID for the table in TS_TABLES and 50 is the TS_ID for the item in that primary/auxiliary table.
attributeName (optional)	string	Returns only transitions that have this transition attribute (see TS_TRANSATTRS for selections). Used only for transitions created via an integration.

Response

Transition or a list of transitions is returned. The list can be empty. See the Transition [page 170] type for more detail.

Usage

The transition or transitions returned are limited by the item you specify. Only transitions available to the user are returned.

Faults

- Invalid database pointer.
- The item ID is not valid.
- Reading transition attribute fails.

XML

The following XML is a snippet of the payload being sent with GetAvailableTransitions.

```
<urn:GetAvailableTransitions>
    <urn:auth>
        <urn:userId>bill</urn:userId>
        <urn:password></urn:password>
        <urn:hostname>localhost</urn:hostname>
    </urn:auth>
    <urn:itemID>1000:53</urn:itemID>
    <urn:attributeName></urn:attributeName>
</urn:GetAvailableTransitions>
```

GetAvailableTransitionsWithStateIDs

Description

This service returns a list of available transitions for the specified item. That list can be filtered to include only the transitions that have the specified attribute. The response contains internal state IDs in addition to state names.

Arguments

Argument	Type	Description
auth (optional)	Auth [page 104]	The *Auth* type supplies credentials and optionally, a host name for licensing. The userId and password can be specified with HTTP BASIC or WS-SECURITY instead.
itemID (required)	string	The item identifier. You must specify the item in tableid:internal-item-id format. For example, 1009:50; where 1009 is the TS_ID for the table in TS_TABLES and 50 is the TS_ID for the item in that primary/auxiliary table.
attributeName (optional)	string	Returns only transitions that have this transition attribute (see TS_TRANSATTRS for selections). Used only for transitions created via an integration.

Response

Transition or a list of transitions is returned. The list can be empty. See the Transition [page 170] type for more detail.

Usage

The transition or transitions returned are limited by the item you specify. Only transitions available to the user are returned. Use this call instead of GetAvailableTransitions if you want to additionally return the ID or UUID of the "to" and "from" states.

Faults

- Invalid database pointer.
- The item ID is not valid.
- Reading transition attribute fails.

XML

The following XML is a snippet of the payload being sent with GetAvailableTransitionsWithStateID.

```
<urn:GetAvailableTransitionsWithStateID>
    <urn:auth>
        <urn:userId>bill</urn:userId>
        <urn:password></urn:password>
        <urn:hostname>localhost</urn:hostname>
    </urn:auth>
    <urn:itemID>1000:53</urn:itemID>
    <urn:attributeName></urn:attributeName>
</urn:GetAvailableTransitionsWithStateID>
```

GetFileAttachment

Description

This service gets an existing file attachment.

Arguments

Argument	Type	Description
auth (optional)	Auth [page 104]	The *Auth* type supplies credentials and optionally, a host name for licensing. The userId and password can be specified with HTTP BASIC or WS-SECURITY instead.
itemID (required)	string	The item identifier. You must specify the item in tableid:internal-item-id format. For example, 1009:50; where 1009 is the TS_ID for the table in TS_TABLES and 50 is the TS_ID for the item in that primary/auxiliary table
attachmentID (required)	integer	This is the internal TS_ID of the attachment from the TS_ATTACHMENTS table.

Response

FileAttachmentContents is returned. The file attachment is returned as a base64 encoded attachment. See FileAttachmentContents [page 144] for more information.

Usage

You can use the ID returned in the FileAttachmentList parameter of a returned TTItem in the attachmentID argument of GetFileAttachment.

Faults

- Invalid database pointer.
- The item ID is not valid.
- The user lacks sufficient permission.
- Creating the record fails.
- The attachment ID is not valid.
- Failed to read the attachment contents.

XML

The following XML is a snippet of the payload being sent with GetFileAttachment.

```
<urn:GetFileAttachment>
    <urn:auth>
        <urn:userId>admin</urn:userId>
        <urn:password></urn:password>
        <urn:hostname>localhost</urn:hostname>
    </urn:auth>
    <urn:itemID>1000:26</urn:itemID>
    <urn:attachmentID>39</urn:attachmentID>
</urn:GetFileAttachment>
```

Getitem

Description

This service returns the item specified, based on user privileges.

Arguments

Argument	Type	Description
auth (optional)	Auth [page 104]	The *Auth* type supplies credentials and optionally, a host name for licensing. The userId and password can be specified with HTTP BASIC or WS-SECURITY instead.
itemID (required)	integer	The item identifier. You must specify the item in tableid:internal-item-id format. For example, 1009:50; where 1009 is the TS_ID for the table in TS_TABLES and 50 is the TS_ID for the item in that primary/auxiliary table.
responseOptions (optional)	string	If provided, this parameter specifies which parts of an item should be returned in the response.

Response

TTItem is returned. The item data, showing the unique TS_ID of the record and TS_ID of the table to which it belongs are displayed. For more detail, see TTItem [page 174]

Usage

The GetItem call provides a snapshot of an auxiliary or primary item, without invoking any changes against the item. To retrieve multiple items, use Getitems [page 61].

Use the optional responseOptions parameter to limit the data returned for a given item. In responseOptions, you specify the sections of an item that should be returned. The sections that aren't specified are not included in the response. For example, if the items you are getting have a large number of item links, notes, URL and file attachments that don't need to be returned in the response, you can use a comma-separated list in the responseOptions parameter to return only the sections of an item you want. For example:

```
<urn:responseOptions>SECTION:FIXED,SECTION:EXTENDED</urn:responseOptions>
```

This will ensure that only the fixed and extendedFieldList sections are returned. Here are some of the possible sections you can specify:

- SECTION:NONE

  No sections are returned. The genericItem portion of TTItem is the only part returned. The genericItem section is always returned, regardless of the value or values specified in responseOptions.

- SECTION:ALL

  All sections are returned. This is the default value for responseOptions. If no value is specified in responseOptions, ALL is the assumed value.

- SECTION:FIXED

All parameters in TTItem (from <urn:itemType> to <urn:url>) prior to the extendedFieldList are returned.

- SECTION:EXTENDED

  Returns the entire extendedFieldList section of TTItem.

- SECTION:ATTACHMENTS

  Returns all of the attachment sections of TTItem. You can return the next four sections simply by specifying:

```
<urn:responseOptions>SECTION:ATTACHMENTS</urn:responseOptions>
```

- SECTION:NOTES

  Returns the entire noteList section of TTItem.

- SECTION:ITEMLINKS

  Returns the entire itemLinkList section of TTItem.

- SECTION:URLATTACHMENTS

  Returns the entire urlAttachmentList section of TTItem.

- SECTION:FILEATTACHMENTS

  Returns the entire fileAttachmentList secton of TTItem.

Note: If you specify SECTION:NONE after other sections, those preceding sections will not be returned. For example, SECTION:FIXED,SECTION:NONE,SECTION:EXTENDED will only return the extendedFieldList section.

Faults

- Invalid database pointer.
- The item ID is not valid.
- The user lacks sufficient permission.
- Creating the record fails.

XML

The following XML is a snippet of the payload being sent with GetItem.

```
<urn:GetItem>
    <urn:auth>
        <urn:userId>jill</urn:userId>
        <urn:password></urn:password>
        <urn:hostname>localhost</urn:hostname>
    </urn:auth>
    <urn:itemID>1006:17</urn:itemID>
    <urn:responseOptions>SECTION:FIXED,SECTION:EXTENDED</urn:responseOptions>
</urn:GetItem>
```

Getitems

Description

This service returns multiple items, based on user privileges.

Arguments

Argument	Type	Description
auth (optional)	Auth [page 104]	The *Auth* type supplies credentials and optionally, a host name for licensing. The userId and password can be specified with HTTP BASIC or WS-SECURITY instead.
itemIdList (required)	string	A list of item identifiers. You must specify each item in tableid:internal-item-id format. For example, 1009:50; where 1009 is the TS_ID for the table in TS_TABLES and 50 is the TS_ID for the item in that primary or auxiliary table.
responseOptions (optional)	string	If provided, this parameter specifies which parts of an item should be returned in the response.

Response

TTItem is returned, one for each item. The item data, showing the unique TS_IDs of the records and TS_ID of the table to which they belong are displayed. For more detail, see TTItem [page 174]

Usage

The GetItems call provides a snapshot of several auxiliary or primary items, without invoking any changes against the items. To retrieve a single item, use Getitem [page 60].

Use the optional responseOptions parameter to limit the data returned for a given item. In responseOptions, you specify the sections of an item that should be returned. The sections that aren't specified are not included in the response. For example, if the items you are getting have a large number of item links, notes, URL and file attachments that don't need to be returned in the response, you can use a comma-separated list in the responseOptions parameter to return only the sections of an item you want. For example:

```
<urn:responseOptions>SECTION:FIXED,SECTION:EXTENDED</urn:responseOptions>
```

This will ensure that only the fixed and extendedFieldList sections are returned. Here are some of the possible sections you can specify:

- SECTION:NONE

  No sections are returned. The genericItem portion of TTItem is the only part returned. The genericItem section is always returned, regardless of the value or values specified in responseOptions.

- SECTION:ALL

  All sections are returned. This is the default value for responseOptions. If no value is specified in responseOptions, ALL is the assumed value.

- SECTION:FIXED

  All parameters in TTItem (from <urn:itemType> to <urn:url>) prior to the extendedFieldList are returned.

- SECTION:EXTENDED

  Returns the entire extendedFieldList section of TTItem.

- SECTION:ATTACHMENTS

  Returns all of the attachment sections of TTItem. You can return the next four sections simply by specifying:

```
<urn:responseOptions>SECTION:ATTACHMENTS</urn:responseOptions>
```

- SECTION:NOTES

  Returns the entire noteList section of TTItem.

- SECTION:ITEMLINKS

  Returns the entire itemLinkList section of TTItem.

- SECTION:URLATTACHMENTS

  Returns the entire urlAttachmentList section of TTItem.

- SECTION:FILEATTACHMENTS

  Returns the entire fileAttachmentList secton of TTItem.

Note: If you specify SECTION:NONE after other sections, those preceding sections will not be returned. For example, SECTION:FIXED,SECTION:NONE,SECTION:EXTENDED will only return the extendedFieldList section.

Faults

- Invalid database pointer.
- The item ID is not valid.
- The user lacks sufficient permission.
- Creating the record fails.

XML

The following XML is a snippet of the payload being sent with GetItems.

```
<urn:GetItems>
    <urn:auth>
        <urn:userId>admin</urn:userId>
        <urn:password></urn:password>
        <urn:hostname>localhost</urn:hostname>
    </urn:auth>
    <urn:itemIdList>1006:17</urn:itemIdList>
    <urn:itemIdList>1006:18</urn:itemIdList>
    <urn:responseOptions>SECTION:FIXED,SECTION:EXTENDED</urn:responseOptions>
</urn:GetItems>
```

GetItemsByQuery

Description

This service returns multiple items found using a *where* clause and an *order by* clause to determine the set of items returned.

Arguments

Argument	Type	Description
auth (optional)	Auth [page 104]	The *Auth* type supplies credentials and optionally, a host name for licensing. The userId and password can be specified with HTTP BASIC or WS-SECURITY instead.
tableID (required)	integer	The table identifier (TS_ID) of the auxiliary table in TS_TABLES where the item will be created.
queryWhereClause (optional)	string	An SQL statement to find items with TS_ID>0. If not provided, all items in the table are returned.
orderByClause (optional)	string	An SQL statement to order the returned items. Enter a null or empty string for no ordering.
maxReturnSize (optional)	integer	Enter the number of items to return.
responseOptions (optional)	string	If provided, this parameter specifies which parts of an item should be returned in the response.

Response

TTItem is returned, one for each item. The response is a list of items in the specidied table that match the query *where* clause. The list is ordered and limited as specified. For more detailed information, see TTItem [page 174].

Usage

In the maxReturnSize element, set the value to zero to use the system "Listing Report Items" limit. A number greater than zero will limit the return item list to the number you specify. If the number of items that would be returned exceeds the system "Listing Report Items" limit, no items will be returned and an error will be generated.

Note: You need only pass the conditions in the *where* clause itself. For example:

```
<urn:queryWhereClause>TS_ISSUEID LIKE '00106' OR TS_ISSUEID LIKE '00029'
 OR TS_ISSUEID LIKE '00105'</urn:queryWhereClause>

<urn:queryWhereClause>TS_ISSUEID='00038'</urn:queryWhereClause>

<urn:queryWhereClause>TS_ID='4'</urn:queryWhereClause>
```

To return all items, send a null query:

```
<urn:queryWhereClause></urn:queryWhereClause>
```

Use the optional responseOptions parameter to limit the data returned for a given item. In responseOptions, you specify the sections of an item that should be returned. The

sections that aren't specified are not included in the response. For example, if the items you are getting have a large number of item links, notes, URL and file attachments that don't need to be returned in the response, you can use a comma-separated list in the responseOptions parameter to return only the sections of an item you want. For example:

```
<urn:responseOptions>SECTION:FIXED,SECTION:EXTENDED</urn:responseOptions>
```

This will ensure that only the fixed and extendedFieldList sections are returned. Here are some of the possible sections you can specify:

- SECTION:NONE

  No sections are returned. The genericItem portion of TTItem is the only part returned. The genericItem section is always returned, regardless of the value or values specified in responseOptions.

- SECTION:ALL

  All sections are returned. This is the default value for responseOptions. If no value is specified in responseOptions, ALL is the assumed value.

- SECTION:FIXED

  All parameters in TTItem (from <urn:itemType> to <urn:url>) prior to the extendedFieldList are returned.

- SECTION:EXTENDED

  Returns the entire extendedFieldList section of TTItem.

- SECTION:ATTACHMENTS

  Returns all of the attachment sections of TTItem. You can return the next four sections simply by specifying:

```
<urn:responseOptions>SECTION:ATTACHMENTS</urn:responseOptions>
```

- SECTION:NOTES

  Returns the entire noteList section of TTItem.

- SECTION:ITEMLINKS

  Returns the entire itemLinkList section of TTItem.

- SECTION:URLATTACHMENTS

  Returns the entire urlAttachmentList section of TTItem.

- SECTION:FILEATTACHMENTS

  Returns the entire fileAttachmentList secton of TTItem.

Note: If you specify SECTION:NONE after other sections, those preceding sections will not be returned. For example, SECTION:FIXED,SECTION:NONE,SECTION:EXTENDED will only return the extendedFieldList section.

Faults

- Invalid database pointer.
- The table ID is not valid.

- The user lacks sufficient permission.
- Creating the record fails.
- Query exceeds system "Listing Report Items" limit.
- Reading the item fails.

XML

The following XML is a snippet of the payload being sent with GetItemsByQuery.

```
<urn:GetItemsByQuery>
    <urn:auth>
        <urn:userId>admin</urn:userId>
        <urn:password></urn:password>
        <urn:hostname>localhost</urn:hostname>
    </urn:auth>
    <urn:tableID>1006</urn:tableID>
    <urn:queryWhereClause>TS_TITLE LIKE 'New Item'</urn:queryWhereClause>
    <urn:orderByClause>TS_TITLE</urn:orderByClause>
    <urn:maxReturnSize>2</urn:maxReturnSize>
    <urn:responseOptions>SECTION:FIXED,SECTION:EXTENDED</urn:responseOptions>
</urn:GetItemsByQuery>
```

GetItemsByQueryWithName

Description

This service returns multiple items found using a *where* clause and an *order by* clause to determine the set of items returned. You can specify a table database name instead of the table ID with this call.

Arguments

Argument	Type	Description
auth (optional)	Auth [page 104]	The *Auth* type supplies credentials and optionally, a host name for licensing. The userId and password can be specified with HTTP BASIC or WS-SECURITY instead.
tableDBName (required)	integer	The database name of the table. For example, to return an item in the Companies table, use TS_COMPANIES.
queryWhereClause (optional)	string	An SQL statement to find items with TS_ID>0. If not provided, all items in the table are returned.
orderByClause (optional)	string	An SQL statement to order the returned items. Enter a null or empty string for no ordering.
maxReturnSize (optional)	integer	Enter the number of items to return.

Argument	Type	Description
responseOptions (optional)	string	If provided, this parameter specifies which parts of an item should be returned in the response.

Response

TTItem is returned, one for each item. The response is a list of items in the specidied table that match the query *where* clause. The list is ordered and limited as specified. For more detailed information, see TTItem [page 174].

Usage

In the maxReturnSize element, you can set the value to zero to use the system "Listing Report Items" limit. A number greater than zero will limit the return item list to the number you specify. If the number of items that would be returned exceeds the system "Listing Report Items" limit, no items will be returned and an error will be generated.

Note: You need only pass the conditions in the *where* clause itself. For example:

```
<urn:queryWhereClause>TS_ISSUEID LIKE '00106' OR TS_ISSUEID LIKE '00029'
 OR TS_ISSUEID LIKE '00105'</urn:queryWhereClause>

<urn:queryWhereClause>TS_ISSUEID='00038'</urn:queryWhereClause>

<urn:queryWhereClause>TS_ID='4'</urn:queryWhereClause>
```

To return all items, send a null query:

```
<urn:queryWhereClause></urn:queryWhereClause>
```

Use the optional responseOptions parameter to limit the data returned for a given item. In responseOptions, you specify the sections of an item that should be returned. The sections that aren't specified are not included in the response. For example, if the items you are getting have a large number of item links, notes, URL and file attachments that don't need to be returned in the response, you can use a comma-separated list in the responseOptions parameter to return only the sections of an item you want. For example:

```
<urn:responseOptions>SECTION:FIXED,SECTION:EXTENDED</urn:responseOptions>
```

This will ensure that only the fixed and extendedFieldList sections are returned. Here are some of the possible sections you can specify:

- SECTION:NONE

  No sections are returned. The genericItem portion of TTItem is the only part returned. The genericItem section is always returned, regardless of the value or values specified in responseOptions.

- SECTION:ALL

  All sections are returned. This is the default value for responseOptions. If no value is specified in responseOptions, ALL is the assumed value.

- SECTION:FIXED

  All parameters in TTItem (from <urn:itemType> to <urn:url>) prior to the extendedFieldList are returned.

- SECTION:EXTENDED

  Returns the entire extendedFieldList section of TTItem.

- SECTION:ATTACHMENTS

  Returns all of the attachment sections of TTItem. You can return the next four sections simply by specifying:

```
<urn:responseOptions>SECTION:ATTACHMENTS</urn:responseOptions>
```

- SECTION:NOTES

  Returns the entire noteList section of TTItem.

- SECTION:ITEMLINKS

  Returns the entire itemLinkList section of TTItem.

- SECTION:URLATTACHMENTS

  Returns the entire urlAttachmentList section of TTItem.

- SECTION:FILEATTACHMENTS

  Returns the entire fileAttachmentList secton of TTItem.

Note: If you specify SECTION:NONE after other sections, those preceding sections will not be returned. For example, SECTION:FIXED,SECTION:NONE,SECTION:EXTENDED will only return the extendedFieldList section.

Faults

- Invalid database pointer.
- The table ID is not valid.
- The user lacks sufficient permission.
- Creating the record fails.
- Query exceeds system "Listing Report Items" limit.
- Reading the item fails.

XML

The following XML is a snippet of the payload being sent with GetItemsByQueryWithName.

```
<urn:GetItemsByQueryWithName>
    <urn:auth>
        <urn:userId>admin</urn:userId>
        <urn:password></urn:password>
        <urn:hostname>localhost</urn:hostname>
    </urn:auth>
    <urn:tableDBName>TTS_INCIDENTS</urn:tableDBName>
    <urn:queryWhereClause>TS_TITLE LIKE 'New Item'</urn:queryWhereClause>
    <urn:orderByClause>TS_TITLE</urn:orderByClause>
```

```
        <urn:maxReturnSize>2</urn:maxReturnSize>
        <urn:responseOptions>SECTION:FIXED,SECTION:EXTENDED</urn:responseOptions>
    </urn:GetItemsByQueryWithName>
```

GetReports

Description

This service returns a list of reports within a specified range, limited by one or more optional filters.

Arguments

Argument	Type	Description
auth (optional)	Auth [page 104]	The *Auth* type supplies credentials and optionally, a host name for licensing. The userId and password can be specified with HTTP BASIC or WS-SECURITY instead.
queryRange (optional)	QueryRange [page 113]	The *QueryRange* type allows you to specify the number of reports to return. It can be used to limit the number of reports that are returned.
reportsFilter (optional)	ReportsFilter [page 116]	The *ReportsFilter* type allows you to filter the reports that are returned based on optional parameters similar to the **Find Reports** command in the Web interface.

Response

GetReportsResult is returned. A filtered list of reports, within the specified range, is displayed in the response. The list diplays the number of reports returned and a high-level description for each report. For more detail, see GetReportsResult [page 146].

Usage

The GetReports call finds a list of available reports without actually running any of the reports themselves. You can use GetReports to search for all reports that you have privileges to run, modify, and delete within the provided range . You can also use this call to search for auxiliary table reports that you can run, modify, and delete. You can use this call to identify which report you would like to run using the UUID of the report. To run a given report in the returned list, use RunReport [page 86]. You can optionally use the returned report URL to run the report in a Web browser.

Faults

- Invalid database pointer.
- The project ID is not valid.
- The user lacks sufficient permission.
- Returing the report fails.

XML

The following XML is a snippet of the payload being sent with GetReports.

```
<urn:GetReports>
    <urn:auth>
        <urn:userId>admin</urn:userId>
        <urn:password></urn:password>
        <urn:hostname>localhost</urn:hostname>
        <urn:loginAsUserId></urn:loginAsUserId>
    </urn:auth>
    <urn:queryRange>
        <urn:startIndex>1</urn:startIndex>
        <urn:fetchSize>4</urn:fetchSize>
        <urn:totalCount></urn:totalCount>
    </urn:queryRange>
    <urn:reportsFilter>
        <urn:solutionID>1</urn:solutionID>
        <urn:solutionName>BASE_ISSUE_DEFECT_TRACKING</urn:solutionName>
        <urn:projectID>4</urn:projectID>
        <urn:projectName>Base Project||Base IDT Project||Software Development
        →</urn:projectName>
        <urn:authorID>joe</urn:AuthorID>
        <urn:reportType>1</urn:reportType>
        <urn:reportCategory>ALL</urn:reportCategory>
        <urn:reportAccessLevel>USER</urn:reportAccessLevel>
        <urn:reportName>ALL</urn:reportName>
        <urn:includeSubProjects>true</urn:includeSubProjects>
        <urn:createdDateFrom>2007-06-20T15:35:38-07:00</urn:createdDateFrom>
        <urn:createdDateTo>2007-07-20T15:35:38-07:00</urn:createdDateTo>
    </urn:reportsFilter>
</urn:GetReports>
```

GetSolutions

Description

This service returns a list of applications that can be accessed by the user.

Arguments

Argument	Type	Description
auth (optional)	Auth [page 104]	The *Auth* type supplies credentials and optionally, a host name for licensing. The userId and password can be specified with HTTP BASIC or WS-SECURITY instead.

Response

SolutionData (a list of solutions) will be returned in the response. Each return element shows the solution's TS_ID, UUID, name, description, and application definition UUID. See SolutionData [page 162] for more information.

Usage

This call is a good starting point for using the Business Mashups Web services. After calling GetSolutions, you can call GetTables [page 74] to get available tables. You can then use the table identifiers to get items with GetItemsByQuery [page 63], or create auxiliary items with CreateAuxItem [page 23] or CreateAuxItems [page 30]. To create

primary items, use GetSubmitProjects [page 72] followed by CreatePrimaryItem [page 34] or CreatePrimaryItems [page 37].

Note: Primary items require a project identifier, while auxiliary items require a table identifier.

Faults

- Invalid database pointer.
- No solutions are available for the user.

XML

The following XML is a snippet of the payload being sent with GetSolutions.

```
<urn:GetSolutions>
    <urn:auth>
        <urn:userId>admin</urn:userId>
        <urn:password></urn:password>
        <urn:hostname>localhost</urn:hostname>
    </urn:auth>
</urn:GetSolutions>
```

GetSolutionsWithUniqueName

Description

This service returns a list of applications that can be accessed by the user.

Arguments

Argument	Type	Description
auth (optional)	Auth [page 104]	The *Auth* type supplies credentials and optionally, a host name for licensing. The userId and password can be specified with HTTP BASIC or WS-SECURITY instead.

Response

SolutionWithUniqueName (a list of solutions showing the unique name for each solution) will be returned in the response. Each return element shows the solution's TS_ID, UUID, unique name, description, and application definition UUID. See SolutionWithUniqueName [page 165] for more information.

Note: The response for GetSolutionsWithUniqueName returns the unique solution name; the response for GetSolutions does not. Thus, if you want to provide the unique solution name for the GetTablesWithName call, use GetSolutionsWithUniqueName.

Usage

This call is a good starting point for using the Business Mashups Web services. After calling GetSolutionsWithUniqueName, you can call GetTables [page 74] to get available tables. You can then use the table identifiers to get items with GetItemsByQuery [page 63], or create auxiliary items with CreateAuxItem [page 23] or CreateAuxItems [page 30]. To create primary items, use GetSubmitProjects [page 72] followed by CreatePrimaryItem [page 34] or CreatePrimaryItems [page 37].

Note: Primary items require a project identifier, while auxiliary items require a table identifier.

Faults

- Invalid database pointer.
- No solutions are available for the user.

XML

The following XML is a snippet of the payload being sent with GetSolutionsWithUniqueName.

```
<urn:GetSolutionsWithUniqueName>
    <urn:auth>
        <urn:userId>admin</urn:userId>
        <urn:password></urn:password>
        <urn:hostname>localhost</urn:hostname>
    </urn:auth>
</urn:GetSolutionsWithUniqueName>
```

GetSubmitProjects

Description

This service returns a list of projects into which the user can submit. If a table ID is provided, only projects from that table are listed.

Arguments

Argument	Type	Description
auth (optional)	Auth [page 104]	The *Auth* type supplies credentials and optionally, a host name for licensing. The userId and password can be specified with HTTP BASIC or WS-SECURITY instead.
tableID (optional)	integer	The table identifier (TS_ID) of the table in TS_TABLES. Use this argument to limit the projects that are returned.

Response

ProjectData (a list of projects, if multiple are available) are returned in the response. See ProjectData [page 150] for more information.

Usage

The project or projects returned are limited by the table you specify. Only projects available to the user are returned. Use CreatePrimaryItem [page 34] or CreatePrimaryItems [page 37] to submit after the desired project is found.

Faults

- Invalid database pointer.
- The table ID is not valid.
- Error reading transitions table.

XML

The following XML is a snippet of the payload being sent with GetSubmitProjects.

```
<urn:GetSubmitProjects>
    <urn:auth>
        <urn:userId>admin</urn:userId>
        <urn:password></urn:password>
        <urn:hostname>localhost</urn:hostname>
    </urn:auth>
    <urn:tableID>1001</urn:tableID>
</urn:GetSubmitProjects>
```

GetSubmitProjectsWithName

Description

This service returns a list of projects into which the user can submit. If a table name is provided, only projects from that table are listed.

Arguments

Argument	Type	Description
auth (optional)	Auth [page 104]	The *Auth* type supplies credentials and optionally, a host name for licensing. The userId and password can be specified with HTTP BASIC or WS-SECURITY instead.
tableDBName (optional)	string	The database name of the table. For example, to create a new item in the Incidents table, use TTS_INCIDENTS. Use this argument to limit the projects that are returned.

Response

ProjectData (a list of projects, if multiple are available) are returned in the response. See ProjectData [page 150] for more information.

Usage

The project or projects returned are limited by the table you specify. Only projects available to the user are returned. Use CreatePrimaryItem [page 34] or CreatePrimaryItems [page 37] to submit after the desired project is found.

Faults

- Invalid database pointer.
- The table ID is not valid.
- Error reading transitions table.

XML

The following XML is a snippet of the payload being sent with GetSubmitProjectsWithName.

```
<urn:GetSubmitProjectsWithName>
    <urn:auth>
        <urn:userId>admin</urn:userId>
        <urn:password></urn:password>
        <urn:hostname>localhost</urn:hostname>
    </urn:auth>
    <urn:tableDBName>TTS_INCIDENTS</urn:tableDBName>
</urn:GetSubmitProjectsWithName>
```

GetTables

Description

This service returns a list of tables that can be accessed by the user.

Arguments

Argument	Type	Description
auth (optional)	Auth [page 104]	The *Auth* type supplies credentials and optionally, a host name for licensing. The userId and password can be specified with HTTP BASIC or WS-SECURITY instead.
solutionID (optional)	integer	The solution identifier (TS_ID) of the solution in TS_SOLUTIONS.
tableType (optional)	Table-Type [page 123]	The type of table to return. If a table type is specified, only available tables of that type are listed.

Response

TableData is returned. The list of available tables and their fields are returned, optionally filtered by either solution or table type. See TableData [page 166] for more information.

Usage

If no solution ID is supplied, available tables are listed for all solutions. Use GetSolutions [page 70] to return available solution IDs.

Faults

- Invalid database pointer.
- Reading the tables fails.
- GetTables not implemented for table type.

XML

The following XML is a snippet of the payload being sent with GetTables.

```
<urn:GetTables>
    <urn:auth>
        <urn:userId>admin</urn:userId>
        <urn:password></urn:password>
        <urn:hostname>localhost</urn:hostname>
    </urn:auth>
    <urn:solutionID>1</urn:solutionID>
    <urn:tableType>PRIMARY-TABLE</urn:tableType>
</urn:GetTables>
```

GetTablesWithName

Description

This service returns a list of tables that can be accessed by the user given the solution name.

Arguments

Argument	Type	Description
auth (optional)	Auth [page 104]	The *Auth* type supplies credentials and optionally, a host name for licensing. The userId and password can be specified with HTTP BASIC or WS-SECURITY instead.
solutionName (optional)	string	The name of the solution as found in the TS_NAME column of TS_SOLUTIONS.
tableType (optional)	Table-Type [page 123]	The type of table to return. If a table type is specified, only available tables of that type are listed.

Response

TableData is returned. The list of available tables and their fields are returned, optionally filtered by either solution or table type. See TableData [page 166] for more information.

Usage

If no solution name is supplied, available tables are listed for all solutions. Use GetSolutionsWithUniqueName [page 71] to return available solution names.

Faults

- Invalid database pointer.
- Reading the tables fails.
- GetTables not implemented for table type.

XML

The following XML is a snippet of the payload being sent with GetTablesWithName.

```
<urn:GetTablesWithName>
    <urn:auth>
```

```
            <urn:userId>admin</urn:userId>
            <urn:password></urn:password>
            <urn:hostname>localhost</urn:hostname>
        </urn:auth>
        <urn:solutionName>INCIDENT_MANAGEMENT</urn:solutionName>
        <urn:tableType>PRIMARY-TABLE</urn:tableType>
    </urn:GetTablesWithName>
```

GetUser

Description

This services returns basic information about a user, including name, e-mail address, and time zone.

Arguments

Argument	Type	Description
auth (optional)	Auth [page 104]	The *Auth* type supplies credentials and optionally, a host name for licensing. The userId and password can be specified with HTTP BASIC or WS-SECURITY instead.
userID (required)	string	The login ID of the user you wish to describe.

Response

User is returned. Basic information that describes a user account will be returned. For more specific information, see User [page 179].

Usage

The GetUser call is useful when you need data about a user account. The GetUser call retrieves data for a user account as it exists in the TS_USERS table of the database. It does not return data about a contact record from TS_CONTACTS. To retrieve privileges for a given user account, use GetUserPrivileges [page 77]. To determine if a specific user has a given privilege, use HasUserPrivilege [page 80].

Faults

- Invalid database pointer.
- The user ID is not valid.
- The user lacks sufficient permission.

XML

The following XML is a snippet of the payload being sent with GetUser.

```
    <urn:GetUser>
        <urn:auth>
            <urn:userId>admin</urn:userId>
            <urn:password></urn:password>
            <urn:hostname>localhost</urn:hostname>
        </urn:auth>
```

```
    <urn:userId>chris</urn:userId>
</urn:GetUser>
```

GetUserPrivileges

Description

This services returns a list of privileges for a specified user.

Arguments

Argument	Type	Description
auth (optional)	Auth [page 104]	The *Auth* type supplies credentials and optionally, a host name for licensing. The userId and password can be specified with HTTP BASIC or WS-SECURITY instead.
privilegeType (optional)	string	Indicates the privilege is either a normal user privilege or an administrator privilege. The types of privilege for each are listed below.
objectId (optional)	string	The ID of the object you wish to check privileges against. If provided, privilegeType becomes required.
loginId (optional)	string	Login ID of the user to be checked. If this parameter is not provided, the check will be for the user running the Web service.

Response

Privilege is returned. The privileges are returned in the context of either the calling user or the specified user. The privilege name, type, and object UUID (if the privilege applies only to a particular object-e.g., Project or Table) are returned. For more information, see Privilege [page 149].

Usage

The objectId argument can be used to limit the privilege check to a certain database object. For example, if a table privilege is being checked, this will be the table ID of the table. If it is a project privilege being checked, this will be the project ID. For system privileges, this parameter will be empty. If the object ID isn't all numeric digits, the ID will be interpreted as the UUID for the object. Note that if this parameter is provided, the privilegeType is required.

If no privilege type or an invalid privilege type is specified, the call will succeed, but no privilege information will be returned. The tables below contain the possible privilege types.

The available privilege types for a normal user are:

TS_PRIVTYPE_USERSYS
TS_PRIVTYPE_USERPRJ

TS_PRIVTYPE_USERWKF
TS_PRIVTYPE_USERFLD
TS_PRIVTYPE_USERTBL

The available privilege types for an administrator are:

TS_PRIVTYPE_ADMSYS
TS_PRIVTYPE_ADMPRJ
TS_PRIVTYPE_ADMWKF
TS_PRIVTYPE_ADMFLD_PRJ
TS_PRIVTYPE_ADMFLD_WKF
TS_PRIVTYPE_ADMCON
TS_PRIVTYPE_UNKNOWN
TS_PRIVTYPE_SYSMASK
TS_PRIVTYPE_ADMMASK
TS_PRIVTYPE_TBLMASK

Faults

- Invalid database pointer.
- The privilege type name is not valid.
- The login ID is not valid.
- The user lacks sufficient permission.

XML

The following XML is a snippet of the payload being sent with GetUserPrivileges.

```
<urn:GetUserPrivileges>
    <urn:auth>
        <urn:userId>admin</urn:userId>
        <urn:password></urn:password>
        <urn:hostname>localhost</urn:hostname>
    </urn:auth>
    <urn:privilegeType>TS_PRIVTYPE_USERTBL</urn:privilegeType>
    <urn:objectId>1003</urn:objectId>
    <urn:loginId></urn:loginId>
</urn:GetUserPrivileges>
```

Another example:

```
<urn:GetUserPrivileges>
    <urn:auth>
        <urn:userId>admin</urn:userId>
        <urn:password></urn:password>
        <urn:hostname>localhost</urn:hostname>
    </urn:auth>
    <urn:privilegeType>TS_PRIVTYPE_ADMCON</urn:privilegeType>
    <urn:objectId></urn:objectId>
    <urn:loginId>joe</urn:loginId>
</urn:GetUserPrivileges>
```

GetUserWithPreferences

Description

This services returns basic information about a user, including name, e-mail address, date preference, time preference, and time zone.

Arguments

Argument	Type	Description
auth (optional)	Auth [page 104]	The *Auth* type supplies credentials and optionally, a host name for licensing. The userId and password can be specified with HTTP BASIC or WS-SECURITY instead.
userID (required)	string	The login ID of the user you wish to describe.

Response

UserWithPreferences is returned. Basic information that describes a user account will be returned. For more specific information, see UserWithPreferences [page 180].

Usage

The GetUserWithPreferences call is useful when you need data about a user account. The GetUserWithPreferences call retrieves data for a user account as it exists in the TS_USERS table of the database. It does not return data about a contact record from TS_CONTACTS. To retrieve privileges for a given user account, use GetUserPrivileges [page 77]. To determine if a specific user has a given privilege, use HasUserPrivilege [page 80].

Faults

- Invalid database pointer.
- The user ID is not valid.
- The user lacks sufficient permission.

XML

The following XML is a snippet of the payload being sent with GetUserWithPreferences.

```
<urn:GetUserWithPreferences>
    <urn:auth>
        <urn:userId>admin</urn:userId>
```

```
            <urn:password></urn:password>
            <urn:hostname>localhost</urn:hostname>
        </urn:auth>
        <urn:userId>chris</urn:userId>
    </urn:GetUserWithPreferences>
```

GetVersion

Description

This service returns the Business Mashups version and build number.

Arguments

None.

Response

A string is returned, showing the version and build number. For example:

```
<ae:GetVersionResponse>
   <ae:return>Version 7.0.0.02    (Build 501)</ae:return>
</ae:GetVersionResponse>
```

Usage

None.

Faults

None.

XML

The following XML shows the payload being sent with GetVersion.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:urn="urn:aewebservices71">
   <soapenv:Header/>
   <soapenv:Body>
      <urn:GetVersion/>
   </soapenv:Body>
</soapenv:Envelope>
```

HasUserPrivilege

Description

This service checks for a specified privilege by name.

Arguments

Argument	Type	Description
auth (optional)	Auth [page 104]	The *Auth* type supplies credentials and optionally, a host name for licensing. The userId and password can be specified with HTTP BASIC or WS-SECURITY instead.

Argument	Type	Description
privilegeName (required)	string	name of the privilege (for example, "TS_USRSYSPRIV_APIACCESS")
objectId (optional)	string	The ID of the object you wish to check privileges against.
loginId (optional)	string	Login ID of the user to be checked. If this parameter is not provided, the check will be for the user running the Web service.

Response

A boolean is returned, showing whether the user has the privilege (true) or not (false). The privilege is checked in the context of either the calling user or the specified user. For example:

```
<ae:HasUserPrivilegeResponse>
   <ae:return>true</ae:return>
</ae:HasUserPrivilegeResponse>
```

Usage

You can call GetUserPrivileges [page 77] for the administrator account or for a user account (assuming it has all privileges) to return the name of every privilege. You can then use these privilege names in the HasUserPrivilege call to determine if other users have the same privilege.

The objecId argument can be used to limit the privilege check to a certain database object. For example, if a table privilege is being checked, this will be the table ID of the table. If it is a project privilege being checked, this will be the project ID. For system privileges, this parameter will be empty. If the object ID isn't all numeric digits, the ID will be interpreted as the UUID for the object.

Faults

- Invalid database pointer.
- The privilege name is not valid.
- The login ID is not valid.
- The user lacks sufficient permission.

XML

The following XML is a snippet of the payload being sent with HasUserPrivilege.

```
<urn:HasUserPrivilege>
    <urn:auth>
        <urn:userId>bill</urn:userId>
        <urn:password></urn:password>
        <urn:hostname>localhost</urn:hostname>
    </urn:auth>
    <urn:privilegeName>TS_ADMCONPRIV_DEPLOY_APPLICATION</urn:privilegeName>
    <urn:objectId></urn:objectId>
```

```
        <urn:loginId></urn:loginId>
    </urn:HasUserPrivilege>
```

Import

Description

This service imports an application.

Arguments

Argument	Type	Description
auth (optional)	Auth [page 104]	The *Auth* type supplies credentials and optionally, a host name for licensing. The userId and password can be specified with HTTP BASIC or WS-SECURITY instead.
xmlInFile (required)	FileContents [page 107]	The Zip file that has the XML payload(s) to be imported. The application file attachment is sent via a base 64-encoded attachment.
adminRepositoryId (required)	string	This argument is currently unused, and should be an empty string.
importResponseEndPoint (optional)	string	The Web service endpoint to call when the import process has completed.
xmlImportOptions (optional)	FileContents [page 107]	XML file containing the import options for behavior and filtering.
validateOnly (optional)	boolean	Performs XML Schema validation and checks UUID references within the import file.

Response

A string is returned, showing an identifier UUID that identifies the import operation. For example:

```
<ae:ImportResponse>
   <ae:return>137e7e12-7c91-449c-a41a-ea28dfbc8390</ae:return>
</ae:ImportResponse>
```

Usage

The xmlInFile argument .zip file can be obtained from the Export [page 52] call or from Mashup Administrator, or it may be a mashup definition being deployed to Business Mashups.

The importResponseEndPoint argument is a Web service endpoint that will be called when the import process has completed. The information sent to this endpoint will include an

import identifier, status (Success, Success with warnings, or Failure), along with an optional list of warning/failure messages.

The validateOnly argument performs XML Schema validation and checks UUID references within the import file. If this argument is set to true, the validation will occur synchronously. If validation is successful, no error will be returned. If unsuccessful, the errors encountered will be returned in a single error.

Faults

- Invalid database pointer.
- The user lacks sufficient permission.
- Creating the attachment fails.
- Failed to read the attachment contents.

XML

The following XML is a snippet of the payload being sent with Import.

```
<urn:Import>
    <urn:auth>
        <urn:userId>admin</urn:userId>
        <urn:password></urn:password>
        <urn:hostname></urn:hostname>
    </urn:auth>
    <urn:xmlInFile>
        <urn:checksum></urn:checksum>
        <urn:encodedContents>encoded_contents_here</urn:encodedContents>
    </urn:xmlInFile>
    <urn:adminRepositoryID></urn:adminRepositoryID>
    <urn:importResponseEndPoint></urn:importResponseEndPoint>
    <urn:xmlImportOptions>
        <urn:checksum></urn:checksum>
        <urn:encodedContents>encoded_contents_here</urn:encodedContents>
    </urn:xmlImportOptions>
    <urn:validateOnly>true</urn:validateOnly>
</urn:Import>
```

ImportStatus

Description

This service returns the status of a specified Import operation.

Arguments

Argument	Type	Description
auth (optional)	Auth [page 104]	The *Auth* type supplies credentials and optionally, a host name for licensing. The userId and password can be specified with HTTP BASIC or WS-SECURITY instead.
importUUID (required)	string	The uuid of the import for which status information is desired

Response

ImportCurrentStatus is returned. The import status (InProgress or Completed), percentage complete, and current step being run are returned. See ImportCurrentStatus [page 146] for more informaiton.

Usage

If no information is found matching the specified import UUID, then an error will be returned. You can use Import [page 82] to return a UUID that can then be entered in the importUUID argument.

Faults

- Invalid database pointer.
- No information for the specified import uuid.
- The user lacks sufficient permission.

XML

The following XML is a snippet of the payload being sent with ImportStatus.

```
<urn:ImportStatus>
    <urn:auth>
        <urn:userId>admin</urn:userId>
        <urn:password></urn:password>
        <urn:hostname></urn:hostname>
    </urn:auth>
    <urn:importUUID>50e7f324-c553-4b13-9b31-85017eff2dbc</urn:importUUID>
</urn:ImportStatus>
```

IsUserValid

Description

This service determines whether a specified user is valid.

Arguments

Argument	Type	Description
auth (optional)	Auth [page 104]	The *Auth* type supplies credentials and optionally, a host name for licensing. The userId and password can be specified with HTTP BASIC or WS-SECURITY instead.
loginID (optional)	string	The login ID of user to be checked.

Response

A boolean is returned, showing whether the user is valid (true) or not (false). The user account is checked in the context of either the calling user or the specified user. For example:

```
<ae:IsUserValidResponse>
   <ae:return>true</ae:return>
</ae:IsUserValidResponse>
```

Usage

IsUserValid will return true if the specified user was found in the database, and is not deleted or disabled. Otherwise false is returned.

Faults

- Invalid database pointer.
- The login ID is not valid.
- The user lacks sufficient permission.

XML

The following XML is a snippet of the payload being sent with IsUserValid.

```
<urn:IsUserValid>
    <urn:auth>
        <urn:userId>admin</urn:userId>
        <urn:password></urn:password>
        <urn:hostname></urn:hostname>
    </urn:auth>
    <urn:loginId>carmen</urn:loginId>
</urn:IsUserValid>
```

Logout

Description

This service releases any licenses and resources associated with the session.

Arguments

Argument	Type	Description
auth (optional)	Auth [page 104]	The *Auth* type supplies credentials and optionally, a host name for licensing. The userId and password can be specified with HTTP BASIC or WS-SECURITY instead.

Response

An empty XML response is returned:

```
<ae:LogoutResponse/>
```

and the session is ended. Failure will keep the session open.

Usage

The Logout call logs out the user from the current active session. There is no effect if previous Web service calls are not made before Logout is called.

Faults

- Authentication error if invalid credentials.

XML

The following XML is a snippet of the payload being sent with Logout.

```
<urn:Logout>
    <urn:auth>
        <urn:userId>admin</urn:userId>
        <urn:password></urn:password>
        <urn:hostname></urn:hostname>
    </urn:auth>
</urn:Logout>
```

RunReport

Description

This service runs a specified report, given the proper privileges.

Arguments

Argument	Type	Description
auth (optional)	Auth [page 104]	The *Auth* type supplies credentials and optionally, a host name for licensing. The userId and password can be specified with HTTP BASIC or WS-SECURITY instead.
queryRange (optional)	QueryRange [page 113]	The *QueryRange* type allows you to specify the number of reports to return. It can be used to limit the number of reports that are returned.
reportUUID (optional)	string	Unique identifier (TS_UUID from TS_REPORTS) for a report. You can use the response from GetReports [page 69] to find this UUID.
reportName (optional)	string	The name of the report to be executed.
reportID (optional)	integer	This is the internal TS_ID of the report from the TS_REPORTS table.
solutionID (optional)	integer	This is the internal TS_ID of the solution from TS_SOLUTIONS that the report is based on.
solutionName (optional)	string	This is the name of the solution that the report is based on (TS_NAME from TS_SOLUTIONS). If solutionID is also specified, then solutionID will take precedence.

Argument	Type	Description
projectID (optional)	integer	This is the internal TS_ID of the project from TS_PROJECTS that the report was created against.
projectName (optional)	string	The fully qualified name of the project.
projectUUID (optional)	string	This is the alternate unique ID of the project that the report was created against.
tableID (optional)	integer	This is the internal TS_ID of the table from TS_TABLES that the report is based on.
tableName (optional)	string	This is the display name (TS_NAME) of the table from TS_TABLES that the report is based on.
reportCategory (optional)	ReportCategory [page 115]	A broader enumeration that limits the response based on the category of report (built-in reports, application reports, reports you authored).
reportAccessLevel (optional)	ReportAccessLevel [page 114]	An enumeration that limits the response based on the report's access level (PRIVATE, GUEST, USER, or MANAGER).

Response

RunReportResult is returned. A high-level description of the report is returned, along with a description of each column in the report. The fields used to order the results are shown as well. In the result parameter, the actual field values can be found. For more detail, see RunReportResult [page 158].

Usage

The RunReport call executes a given report, assuming you have the proper privilege. The GetReports call finds a list of available reports without actually running any of the reports themselves. In order to run one of those returned reports, use RunReport. If no fetchSize is specified in the QueryRange, up to 1000 items can be returned.

In order to run a Built-in report, you must provide either solutionID or solutionName. Since Built-in reports aren't tied to a specific application, you must provide the RunReport call with the solution ID or name (which can be found in the GetSolutionsWithUniqueName call). User-created reports (those stored in the TS_REPORTS table) do not require solutionID or solutionName. To run a user-created report, you simply need to provide the reportUUID (which can be obtained from the GetReports call).

Faults

- Invalid database pointer.
- The user lacks sufficient permission.
- Executing the report fails.

- Could not run built-in report because it needs correct solutionName (or ID) parameter.
- Two or more reports exist with the name <*reportName*>. Please provide a solution ID, solution name, or use additional parameters to identify which report to run.

XML

The following XML is a snippet of the payload being sent with RunReport.

```
<urn:RunReport>
    <urn:auth>
        <urn:userId>admin</urn:userId>
        <urn:password></urn:password>
        <urn:hostname>localhost</urn:hostname>
        <urn:loginAsUserId></urn:loginAsUserId>
    </urn:auth>
    <urn:queryRange>
        <urn:startIndex>1</urn:startIndex>
        <urn:fetchSize>4</urn:fetchSize>
    </urn:queryRange>
         <urn:reportUUID>f4d4421d-043a-4856-89f6-11aa76ffb042</urn:reportUUID>
         <urn:reportName>Appreport1</urn:reportName>
         <urn:reportID>88</urn:reportID>
         <urn:solutionID>14</urn:solutionID>
         <urn:solutionName>REPORTSAPP</urn:solutionName>
         <urn:projectID>25</urn:projectID>
         <urn:projectName>Test Project</urn:projectName>
         <urn:projectUUID>9f7fec5c-aff1-4c6f-be29-8dbe06f02350</urn:projectUUID>
         <urn:tableID>1011</urn:tableID>
         <urn:tableName>Reports Application</urn:tableName>
         <urn:reportCategory>APPLICATION</urn:reportCategory>
         <urn:reportAccessLevel>USER</urn:reportAccessLevel>
</urn:RunReport>
```

UpdateFileAttachment

Description

This service updates an existing file attachment for a specified item.

Arguments

Argument	Type	Description
auth (optional)	Auth [page 104]	The *Auth* type supplies credentials and optionally, a host name for licensing. The userId and password can be specified with HTTP BASIC or WS-SECURITY instead.

Argument	Type	Description
itemID (required)	string	The item identifier. You must specify the item in tableid:internal-item-id format. For example, 1009:50; where 1009 is the TS_ID for the table in TS_TABLES and 50 is the TS_ID for the item in that primary/auxiliary table.
attachmentContents (required)	FileAttachmentContents [page 106]	The file attachment details and content, which are all optional except for the attachment ID.  **Note:** attachmentContents is of type FileAttachmentContents, but also includes attachment detail found in FileAttachment. The XML example below shows the parameters from both FileAttachment and FileAttachmentContents. See FileAttachment [page 143] for more information.

Response

FileAttachment is returned. The newly updated file attachment details are returned (not the content itself). For more detail, see FileAttachment [page 143]

Usage

The UpdateFileAttachment call provides a method to update a single attachment on an auxiliary or primary item, given the proper privileges. If the call fails, the file attachment will not be updated. To update multiple file attachments for a single item, UpdateFileAttachment must be called for each attachment. Any data that is provided is updated as appropriate. Attachment ID and modification time cannot be set.

Faults

- Invalid database pointer.
- The item ID is not valid.
- The user lacks sufficient permission.

XML

The following XML is a snippet of the payload being sent with UpdateFileAttachment.

```
<urn:UpdateFileAttachment>
    <urn:auth>
```

```
            <urn:userId>admin</urn:userId>
            <urn:password></urn:password>
            <urn:hostname>localhost</urn:hostname>
        </urn:auth>
        <urn:itemID>1006:15</urn:itemID>
        <urn:attachmentContents>
            <urn:id>51</urn:id>
            <urn:name>file attach updated MER00023!</urn:name>
            <urn:fileName>wslog.txt</urn:fileName>
            <urn:showAsImage>false</urn:showAsImage>
            <urn:modificationDateTime></urn:modificationDateTime>
            <urn:url></urn:url>
            <urn:accessType>ATTACHACCESS-RESTRICTED</urn:accessType>
            <urn:checksum>15962</urn:checksum>
            <urn:encodedContents></urn:encodedContents>
        </urn:attachmentContents>
    </urn:UpdateFileAttachment>
```

UpdateItem

Description

This service updates an item using either a specified transition or the default update transition.

Note: Only transition types of Regular and Update (the default update) are supported with this service.

Arguments

Argument	Type	Description
auth (optional)	Auth [page 104]	The *Auth* type supplies credentials and optionally, a host name for licensing. The userId and password can be specified with HTTP BASIC or WS-SECURITY instead.
item (required)	TTItem [page 123]	The *TTItem* type holds the generic data for an item.
transitionId (optional)	integer	If provided, specifies a transition ID (TS_ID from TS_TRANSITIONS) to use when updating the item.
responseOptions (optional)	string	If provided, this parameter specifies which parts of an item should be returned in the response.

Response

TTItem is returned. The item with updated item data, showing the unique TS_ID of the record and TS_ID of the table to which it belongs are displayed. The new UUID for the updated item is also returned. For more detail, see TTItem [page 174].

Usage

The UpdateItem call provides a method to update a single record in a primary or auxiliary table. If a transitionId of 0 is specified, the default update transition is used. If the call fails, the item will not be updated. You can update items in both custom and system primary tables, given the proper privileges. To update multiple items at once, use UpdateItems [page 93].

Note: You can alternately specify the display name of an item in the *itemName* element of the item argument when making the UpdateItem call. This allows you to update an item, setting the the itemtype using the display name instead of the prefix and issue ID combination. For example, instead of passing "ENH12345" you can simply pass "Enhancement" instead.

If you specify a specific transition for the call to use, that transition needs to be a valid transition for the item's project.

To create notes, item links, and URL attachments on the item, add records to the lists that are defined in TTItem. To update a file attachment, see UpdateFileAttachment [page 88].

Note: This note applies to all Application Engine Web services prior to aeWebservices71. When updating a Journal field that is set to Append Only, the UpdateItem call will replace existing text with any new text provided in the *value* element. In order to preserve the existing text, you must provide the existing text in the new *value* element and manually append your new content. Use GetItem to retrieve the contents of your existing journal field. You can then paste these contents into the *value* element for the journal field in UpdateItem and add a new entry at the end. You will need to provide a new date/time stamp and use the same line breaks as they appear between journal entries in your GetItem return *value*. Starting with aeWebServices71, you can simply use the APPEND-VALUES option in Set-Value-Method. See Set-Value-Method [page 120] for more information.

Use the optional responseOptions parameter to limit the data returned for a given item. In responseOptions, you specify the sections of an item that should be returned. The sections that aren't specified are not included in the response. For example, if the items you are updating have a large number of item links, notes, URL and file attachments that don't need to be returned in the response, you can use a comma-separated list in the responseOptions parameter to return only the sections of an item you want. For example:

```
<urn:responseOptions>SECTION:FIXED,SECTION:EXTENDED</urn:responseOptions>
```

This will ensure that only the fixed and extendedFieldList sections are returned. Here are some of the possible sections you can specify:

- SECTION:NONE

  No sections are returned. The genericItem portion of TTItem is the only part returned. The genericItem section is always returned, regardless of the value or values specified in responseOptions.

- SECTION:ALL

  All sections are returned. This is the default value for responseOptions. If no value is specified in responseOptions, ALL is the assumed value.

- SECTION:FIXED

  All parameters in TTItem (from <urn:itemType> to <urn:url>) prior to the extendedFieldList are returned.

- SECTION:EXTENDED

  Returns the entire extendedFieldList section of TTItem.

- SECTION:ATTACHMENTS

  Returns all of the attachment sections of TTItem. You can return the next four sections simply by specifying:

```
<urn:responseOptions>SECTION:ATTACHMENTS</urn:responseOptions>
```

- SECTION:NOTES

  Returns the entire noteList section of TTItem.

- SECTION:ITEMLINKS

  Returns the entire itemLinkList section of TTItem.

- SECTION:URLATTACHMENTS

  Returns the entire urlAttachmentList section of TTItem.

- SECTION:FILEATTACHMENTS

  Returns the entire fileAttachmentList secton of TTItem.

Note: If you specify SECTION:NONE after other sections, those preceding sections will not be returned. For example, SECTION:FIXED,SECTION:NONE,SECTION:EXTENDED will only return the extendedFieldList section.

Faults

- Invalid database pointer.
- Invalid item.
- The user lacks sufficient permission.
- Creating the record fails.
- The transaction fails.
- Reading the item fails.
- Invalid Project

XML

The following XML is a snippet of the payload being sent with UpdateItem.

```
<urn:UpdateItem>
    <urn:auth>
        <urn:userId>admin</urn:userId>
        <urn:password></urn:password>
        <urn:hostname>localhost</urn:hostname>
    </urn:auth>
    <urn:item>
        <urn:genericItem>
            <urn:itemID>1000:36</urn:itemID>
            <urn:itemUUID></urn:itemUUID>
            <urn:itemName></urn:itemName>
```

```
            </urn:genericItem>
            <urn:classification></urn:classification>
            <urn:title>TestScript1Title</urn:title>
            <urn:description>A Description</urn:description>
            <urn:activeInactive>true</urn:activeInactive>
        </urn:item>
        <urn:transitionId>0</urn:transitionId>
        <urn:responseOptions>SECTION:FIXED,SECTION:EXTENDED</urn:responseOptions>
    </urn:UpdateItem>
```

UpdateItems

Description

This service updates multiple items.

Note: Only transition types of Regular and Update (the default update) are supported with this service.

Arguments

Argument	Type	Description
auth (optional)	Auth [page 104]	The *Auth* type supplies credentials and optionally, a host name for licensing. The userId and password can be specified with HTTP BASIC or WS-SECURITY instead.
itemList (required)	TTItem [page 123]	The list of items to be updated. The *TTItem* types hold the generic data for each item. You must have the itemID filled in at a minimum for each item in the list.
transitionId (optional)	integer	If provided, specifies a transition ID (TS_ID from TS_TRANSITIONS) to use when updating the items.
responseOptions (optional)	string	If provided, this parameter specifies which parts of an item should be returned in the response.

Response

TTItem is returned, one for each item updated. The primary items with updated item data, showing the unique TS_IDs of each record and TS_IDs of the table to which they were updated are displayed. The new UUIDs for the updated items are also returned. For more detail, see TTItem [page 174].

Usage

The UpdateItems call provides a method to update multipile records in a primary table or auxiliary table. If a transitionId of 0 is specified, the default update transition is used. If any failures occur, each successive error message is appended to the string that is returned. Multiple error messages are separated by a single newline. Failures do not result in a return before all items have been processed. You can update records in both custom and system primary tables, given the proper privileges. To update only a single item, use UpdateItem [page 90].

If you specify a specific transition for the call to use, that transition needs to be a valid transition for the items' project.

To create notes, item links, and URL attachments on the items, add records to the lists that are defined in each TTItem. To update a file attachment, see UpdateFileAttachment [page 88].

Note: This note applies to all Application Engine Web services prior to aeWebservices71. When updating a Journal field that is set to Append Only, the UpdateItem call will replace existing text with any new text provided in the *value* element. In order to preserve the existing text, you must provide the existing text in the new *value* element and manually append your new content. Use GetItem to retrieve the contents of your existing journal field. You can then paste these contents into the *value* element for the journal field in UpdateItem and add a new entry at the end. You will need to provide a new date/time stamp and use the same line breaks as they appear between journal entries in your GetItem return *value*. Starting with aeWebServices71, you can simply use the APPEND-VALUES option in Set-Value-Method. See Set-Value-Method [page 120] for more information.

Use the optional responseOptions parameter to limit the data returned for a given item. In responseOptions, you specify the sections of an item that should be returned. The sections that aren't specified are not included in the response. For example, if the items you are updating have a large number of item links, notes, URL and file attachments that don't need to be returned in the response, you can use a comma-separated list in the responseOptions parameter to return only the sections of an item you want. For example:

```
<urn:responseOptions>SECTION:FIXED,SECTION:EXTENDED</urn:responseOptions>
```

This will ensure that only the fixed and extendedFieldList sections are returned. Here are some of the possible sections you can specify:

- SECTION:NONE

  No sections are returned. The genericItem portion of TTItem is the only part returned. The genericItem section is always returned, regardless of the value or values specified in responseOptions.

- SECTION:ALL

  All sections are returned. This is the default value for responseOptions. If no value is specified in responseOptions, ALL is the assumed value.

- SECTION:FIXED

  All parameters in TTItem (from <urn:itemType> to <urn:url>) prior to the extendedFieldList are returned.

- SECTION:EXTENDED

  Returns the entire extendedFieldList section of TTItem.

- SECTION:ATTACHMENTS

  Returns all of the attachment sections of TTItem. You can return the next four sections simply by specifying:

```
<urn:responseOptions>SECTION:ATTACHMENTS</urn:responseOptions>
```

- SECTION:NOTES

  Returns the entire noteList section of TTItem.
- SECTION:ITEMLINKS

  Returns the entire itemLinkList section of TTItem.
- SECTION:URLATTACHMENTS

  Returns the entire urlAttachmentList section of TTItem.
- SECTION:FILEATTACHMENTS

  Returns the entire fileAttachmentList secton of TTItem.

Note: If you specify SECTION:NONE after other sections, those preceding sections will not be returned. For example, SECTION:FIXED,SECTION:NONE,SECTION:EXTENDED will only return the extendedFieldList section.

Faults

- Invalid database pointer.
- Invalid item.
- The user lacks sufficient permission.
- Creating the record fails.
- The transaction fails.
- Reading the item fails.
- Invalid project.

XML

The following XML is a snippet of the payload being sent with UpdateItems.

```
<urn:UpdateItems>
    <urn:auth>
        <urn:userId>admin</urn:userId>
        <urn:password></urn:password>
        <urn:hostname>localhost</urn:hostname>
    </urn:auth>
    <urn:itemList>
        <urn:genericItem>
            <urn:itemID>1006:17</urn:itemID>
            <urn:itemUUID></urn:itemUUID>
            <urn:itemName></urn:itemName>
        </urn:genericItem>
        <urn:classification></urn:classification>
        <urn:title>TestScript1ATitle</urn:title>
        <urn:description>A Description</urn:description>
        <urn:activeInactive>true</urn:activeInactive>
    </urn:itemList>
    <urn:itemList>
        <urn:genericItem>
            <urn:itemID>1006:18</urn:itemID>
            <urn:itemUUID></urn:itemUUID>
```

```
            <urn:itemName></urn:itemName>
        </urn:genericItem>
        <urn:classification></urn:classification>
        <urn:title>TestScript1BTitle</urn:title>
        <urn:description>A Description</urn:description>
        <urn:activeInactive>true</urn:activeInactive>
    </urn:itemList>
    <urn:transitionId></urn:transitionId>
    <urn:responseOptions>SECTION:FIXED,SECTION:EXTENDED</urn:responseOptions>
</urn:UpdateItems>
```

UpdateItemsWithName

Description

This service updates existing items using the specified transition name.

Note: Only transition types of Regular and Update (the default update) are supported with this service.

Arguments

Argument	Type	Description
auth (optional)	Auth [page 104]	The *Auth* type supplies credentials and optionally, a host name for licensing. The userId and password can be specified with HTTP BASIC or WS-SECURITY instead.
itemList (required)	TTItem [page 123]	The list of items to be updated. The *TTItem* types hold the generic data for each item. You must have the itemID filled in at a minimum for each item in the list.
transitionName	string	The name of the transition to use.
responseOptions (optional)	string	If provided, this parameter specifies which parts of an item should be returned in the response.

Response

TTItem is returned, one for each item updated. The primary items with updated item data, showing the unique TS_IDs of each record and TS_IDs of the table to which they were updated are displayed. The new UUIDs for the updated items are also returned. For more detail, see TTItem [page 174].

Usage

The UpdateItemsWithName call provides a method to update multipile records in a primary table or auxiliary table. If a transitionId of 0 is specified, the default update transition is used. If any failures occur, each successive error message is appended to the string that is returned. Multiple error messages are separated by a single newline. Failures do not result in a return before all items have been processed. You can update records in both custom and system primary tables, given the proper privileges. To update only a single item using a transition name, use UpdateItemWithName [page 99].

If you specify a specific transition for the call to use, that transition needs to be a valid transition for the item's project.

To create notes, item links, and URL attachments on the item, add records to the lists that are defined in TTItem. To update a file attachment, see UpdateFileAttachment [page 88].

Note: This note applies to all Application Engine Web services prior to aeWebservices71. When updating a Journal field that is set to Append Only, the UpdateItem call will replace existing text with any new text provided in the *value* element. In order to preserve the existing text, you must provide the existing text in the new *value* element and manually append your new content. Use GetItem to retrieve the contents of your existing journal field. You can then paste these contents into the *value* element for the journal field in UpdateItem and add a new entry at the end. You will need to provide a new date/time stamp and use the same line breaks as they appear between journal entries in your GetItem return *value*. Starting with aeWebServices71, you can simply use the APPEND-VALUES option in Set-Value-Method. See Set-Value-Method [page 120] for more information.

Use the optional responseOptions parameter to limit the data returned for a given item. In responseOptions, you specify the sections of an item that should be returned. The sections that aren't specified are not included in the response. For example, if the items you are updating have a large number of item links, notes, URL and file attachments that don't need to be returned in the response, you can use a comma-separated list in the responseOptions parameter to return only the sections of an item you want. For example:

```
<urn:responseOptions>SECTION:FIXED,SECTION:EXTENDED</urn:responseOptions>
```

This will ensure that only the fixed and extendedFieldList sections are returned. Here are some of the possible sections you can specify:

- SECTION:NONE

  No sections are returned. The genericItem portion of TTItem is the only part returned. The genericItem section is always returned, regardless of the value or values specified in responseOptions.

- SECTION:ALL

  All sections are returned. This is the default value for responseOptions. If no value is specified in responseOptions, ALL is the assumed value.

- SECTION:FIXED

  All parameters in TTItem (from <urn:itemType> to <urn:url>) prior to the extendedFieldList are returned.

- SECTION:EXTENDED

  Returns the entire extendedFieldList section of TTItem.

- SECTION:ATTACHMENTS

  Returns all of the attachment sections of TTItem. You can return the next four sections simply by specifying:

```
<urn:responseOptions>SECTION:ATTACHMENTS</urn:responseOptions>
```

- SECTION:NOTES

Returns the entire noteList section of TTItem.

- SECTION:ITEMLINKS

  Returns the entire itemLinkList section of TTItem.

- SECTION:URLATTACHMENTS

  Returns the entire urlAttachmentList section of TTItem.

- SECTION:FILEATTACHMENTS

  Returns the entire fileAttachmentList secton of TTItem.

Note: If you specify SECTION:NONE after other sections, those preceding sections will not be returned. For example, SECTION:FIXED,SECTION:NONE,SECTION:EXTENDED will only return the extendedFieldList section.

Faults

- Invalid database pointer.
- Invalid item.
- Creating the record fails.
- Invalid project.
- The user lacks sufficient permission.
- The transaction fails.
- Reading the item fails.
- More than one transition available with the same name.

XML

The following XML is a snippet of the payload being sent with UpdateItemsWithName.

```
<urn:UpdateItemsWithName>
    <urn:auth>
        <urn:userId>admin</urn:userId>
        <urn:password></urn:password>
        <urn:hostname>localhost</urn:hostname>
    </urn:auth>
    <urn:itemList>
        <urn:genericItem>
            <urn:itemID>1000:36</urn:itemID>
            <urn:itemUUID></urn:itemUUID>
            <urn:itemName></urn:itemName>
        </urn:genericItem>
        <urn:classification></urn:classification>
        <urn:title>TestScript1Title</urn:title>
        <urn:description>A Description</urn:description>
        <urn:activeInactive>true</urn:activeInactive>
    </urn:itemList>
    <urn:itemList>
        <urn:genericItem>
            <urn:itemID>1000:37</urn:itemID>
            <urn:itemUUID></urn:itemUUID>
```

```
            <urn:itemName></urn:itemName>
        </urn:genericItem>
        <urn:classification></urn:classification>
        <urn:title>TestScript2Title</urn:title>
        <urn:description>A Description</urn:description>
        <urn:activeInactive>true</urn:activeInactive>
    </urn:itemList>
    <urn:transitionName>Update</urn:transitionName>
    <urn:responseOptions>SECTION:FIXED,SECTION:EXTENDED</urn:responseOptions>
</urn:UpdateItemsWithName>
```

UpdateItemWithName

Description

This service updates an existing item using the specified transition name.

Note: Only transition types of Regular and Update (the default update) are supported with this service.

Arguments

Argument	Type	Description
auth (optional)	Auth [page 104]	The *Auth* type supplies credentials and optionally, a host name for licensing. The userId and password can be specified with HTTP BASIC or WS-SECURITY instead.
item (required)	TTItem [page 123]	The *TTItem* type holds the generic data for an item.
transitionName	string	The name of the transition to use.
responseOptions	string	If provided, this parameter specifies which parts of an item should be returned in the response.

Response

TTItem is returned. The item with updated item data, showing the unique TS_ID of the record and TS_ID of the table to which it belongs are displayed. The new UUID for the updated item is also returned. For more detail, see TTItem [page 174].

Usage

The UpdateItemWithName call provides a method to update a single record in a primary or auxiliary table. If a transitionName of 0 is specified, the default update transition is used. If the call fails, the item will not be updated. You can update items in both custom and system primary tables, given the proper privileges. To update multiple items at once using a transition name, use UpdateItemsWithName [page 96].

If you specify a specific transition for the call to use, that transition needs to be a valid transition for the item's project.

To create notes, item links, and URL attachments on the item, add records to the lists that are defined in TTItem. To update a file attachment, see UpdateFileAttachment [page 88].

Note: This note applies to all Application Engine Web services prior to aeWebservices71. When updating a Journal field that is set to Append Only, the UpdateItem call will replace existing text with any new text provided in the *value* element. In order to preserve the existing text, you must provide the existing text in the new *value* element and manually append your new content. Use GetItem to retrieve the contents of your existing journal field. You can then paste these contents into the *value* element for the journal field in UpdateItem and add a new entry at the end. You will need to provide a new date/time stamp and use the same line breaks as they appear between journal entries in your GetItem return *value*. Starting with aeWebServices71, you can simply use the APPEND-VALUES option in Set-Value-Method. See Set-Value-Method [page 120] for more information.

Use the optional responseOptions parameter to limit the data returned for a given item. In responseOptions, you specify the sections of an item that should be returned. The sections that aren't specified are not included in the response. For example, if the items you are updating have a large number of item links, notes, URL and file attachments that don't need to be returned in the response, you can use a comma-separated list in the responseOptions parameter to return only the sections of an item you want. For example:

```
<urn:responseOptions>SECTION:FIXED,SECTION:EXTENDED</urn:responseOptions>
```

This will ensure that only the fixed and extendedFieldList sections are returned. Here are some of the possible sections you can specify:

- SECTION:NONE

  No sections are returned. The genericItem portion of TTItem is the only part returned. The genericItem section is always returned, regardless of the value or values specified in responseOptions.

- SECTION:ALL

  All sections are returned. This is the default value for responseOptions. If no value is specified in responseOptions, ALL is the assumed value.

- SECTION:FIXED

  All parameters in TTItem (from <urn:itemType> to <urn:url>) prior to the extendedFieldList are returned.

- SECTION:EXTENDED

  Returns the entire extendedFieldList section of TTItem.

- SECTION:ATTACHMENTS

  Returns all of the attachment sections of TTItem. You can return the next four sections simply by specifying:

```
<urn:responseOptions>SECTION:ATTACHMENTS</urn:responseOptions>
```

- SECTION:NOTES

  Returns the entire noteList section of TTItem.

- SECTION:ITEMLINKS

Returns the entire itemLinkList section of TTItem.

- SECTION:URLATTACHMENTS

  Returns the entire urlAttachmentList section of TTItem.

- SECTION:FILEATTACHMENTS

  Returns the entire fileAttachmentList secton of TTItem.

Note: If you specify SECTION:NONE after other sections, those preceding sections will not be returned. For example, SECTION:FIXED,SECTION:NONE,SECTION:EXTENDED will only return the extendedFieldList section.

Faults

- Invalid database pointer.
- Invalid item.
- Creating the record fails.
- Invalid project.
- The user lacks sufficient permission.
- The transaction fails.
- Reading the item fails.
- More than one transition available with the same name.

XML

The following XML is a snippet of the payload being sent with UpdateItemWithName.

```
<urn:UpdateItemWithName>
    <urn:auth>
        <urn:userId>admin</urn:userId>
        <urn:password></urn:password>
        <urn:hostname>localhost</urn:hostname>
    </urn:auth>
    <urn:item>
        <urn:genericItem>
            <urn:itemID>1000:36</urn:itemID>
            <urn:itemUUID></urn:itemUUID>
            <urn:itemName></urn:itemName>
        </urn:genericItem>
        <urn:classification></urn:classification>
        <urn:title>TestScript1Title</urn:title>
        <urn:description>A Description</urn:description>
        <urn:activeInactive>true</urn:activeInactive>
    </urn:item>
    <urn:transitionName>Update</urn:transitionName>
    <urn:responseOptions>SECTION:FIXED,SECTION:EXTENDED</urn:responseOptions>
</urn:UpdateItemWithName>
```

Arguments

This section provides detailed descriptions of Business Mashups Web service arguments. The arguments are the request parameters that can be used by Business Mashups Web

service operations. The arguments listed here contain one or more parameters, which make up the data being sent to the Web service. The parameters listed in each argument are either simple or complex types themselves. If the type is complex, a link to further detail of that type will be provided in the **Type** column.

The following table lists all supported arguments in alphabetical order, followed by a brief description of each type. Select an argument to view detailed information including:

- **Description**

  A brief description of the argument.

- **Parameters**

  A table describing the types for each argument. Both simple and complex types are listed for each argument. For each complex type, you can click the type name for a detailed description.

- **Usage**

  Any notes, addtional details, and concerns regarding the argument are addressed here.

- **XML**

  An example of the actual XML being sent is displayed here. The XML not only shows the argument and its respective elements, you can also see detailed examples of each element and how to format the expected data.

Arguments

Argument	Description
Attachment-Access-Type [page 103]	Indicates the type of attachment on an item.
Auth [page 104]	Supplies credentials and optionally, a host name for licensing.
FileAttachment [page 105]	Holds the details of a file that you upload to the server.
FileAttachmentContents [page 106]	Holds the actual contents of a file attachment.
FileContents [page 107]	Holds the actual contents of a file.
Item [page 107]	Holds the generic data for an item.
ItemLink [page 108]	Holds information about an item link.
ItemLink-Type [page 109]	Indicates the type of item link.

Argument	Description
NameValue [page 110]	Holds a field name with either a single value or a list of values.
Note [page 113]	Holds information about a note.
QueryRange [page 113]	Limits the number of records returned in a query.
ReportAccessLevel [page 114]	Indicates the access level that is assigned to a report.
ReportCategory [page 115]	A filter that describes a grouping of reports.
ReportsFilter [page 116]	Allows you to filter reports.
ReportType [page 118]	Holds information about a type of report.
Set-Value-By [page 119]	Indicates how the value in the NameValue type should be set on an update or create.
Set-Value-Method [page 120]	Indicates the operation that should be performed with the values in the NameValue type.
Table-Type [page 123]	Indicates the type of table.
TTItem [page 123]	Holds all of the Business Mashups field information for an item.
URLAttachment [page 132]	Holds information about a URL attached to an auxiliary or primary item.
Value or Values [page 133]	Holds one or multiple values.

Attachment-Access-Type

Description

Attachment-Access-Type indicates the type of attachment on an item. It is used for note, item link, URL, and file attachments. This type is used to determine whether the attachment has a restriction, is unrestricted, or if neither is set.

Parameters

Name	Type	Description
ATTACHACCESS-DEFAULT	string	Restrict the attachment only as specified by user privileges.
ATTACHACCESS-RESTRICTED	string	Makes the attachment visible only to users who can view the item.
ATTACHACCESS-UNRESTRICTED	string	Makes the attachment visible to all users who can view the item.

Usage

The Unrestricted status makes the file visible to all users who can view the item. You can set a file to have Default status to restrict the file as specified by user privileges. The Unrestricted status is disabled if you do not have privileges to set file attachments as unrestricted for the selected project or auxiliary table.

XML

The following XML shows Attachment-Access-Type as seen in a typical call.

```
<urn:attachmentContents>
    <urn:id>16</urn:id>
    <urn:name>pdf_doc</urn:name>
    <urn:fileName>relnotes.pdf</urn:fileName>
    <urn:showAsImage>false</urn:showAsImage>
    <urn:modificationDateTime></urn:modificationDateTime>
    <urn:url></urn:url>
    <urn:accessType>ATTACHACCESS-DEFAULT</urn:accessType>
    <urn:checksum></urn:checksum>
    <urn:encodedContents></urn:encodedContents>
</urn:attachmentContents>
```

Auth

Description

The Auth type supplies credentials and optionally, a host name for licensing. The Auth type parameters are listed below.

Parameters

Name	Type	Description
userID	string	The Business Mashups user Login ID. If you are only specifying the host name, then *userID* is optional
password	string	The password for the user. If you are only specifying the host name, then *password* is optional.

Name	Type	Description
hostname	string	The host name of the client.
loginAsUserId	string	User ID for the Business Mashups login you wish to impersonate. If you are only specifying the host name, then *loginAsUserId* is optional.

Usage

The *Auth* type allows credentials to be provided if not using WS-SECURITY or HTTP BASIC to pass the credentials. The *hostname* element is only needed in case you want to override the client's IP address for licensing purposes, forcing Serena License Manager to use a particular client host. If it's not provided, the code gets the client hostname from the socket.

XML

The following XML shows Auth as seen in a typical call.

```
<urn:auth>
    <urn:userId>admin</urn:userId>
    <urn:password>password</urn:password>
    <urn:hostname>localhost</urn:hostname>
</urn:auth>
```

FileAttachment

Description

The FileAttachment type holds the details of a file that you upload to the server. The FileAttachment type parameters are listed below.

Parameters

Name	Type	Description
id	integer	This is the internal TS_ID of the attachment from the TS_ATTACHMENTS table.
name	string	The name you give the attachment in Business Mashups.
fileName	string	The name of the file as it exists on the file system.
showAsImage	boolean	This flag indicates whether or not graphic attachments are shown as images in the Web interface.
modificationDateTime	dateTime	The date and time when the attachment was last modified. See Supported Date/Time Formats [page 184] for more information.

Name	Type	Description
url	string	The URL for the attachment. The file can be downloaded from this URL.
accessType	Attachment-Access-Type [page 103]	Shows the access type for the attachment. The value is either DEFAULT, RESTRICTED, or UNRESTRICTED.

Usage

FileAttachmentContents is inherited from FileAttachment. FileAttachment is used with FileAttachmentContents to completely describe a file attachment. The file attachment detail consists of an ID, name, and URL. With the URL, client code can download the file directly.

XML

The following XML snippet shows the FileAttachment type in the FileAttachmentList parameter of TTItem.

```
<urn:fileAttachmentList>
    <urn:id>39</urn:id>
    <urn:name>test</urn:name>
    <urn:fileName>wslog.txt</urn:fileName>
    <urn:showAsImage>false</urn:showAsImage>
    <urn:modificationDateTime>2007-06-20T15:35:38-07:00
    →</urn:modificationDateTime>
    <urn:url>http://server:80/tmtrack/tmtrack.dll?AttachmentPage
    →&amp;amp;AttachmentID=39
    </urn:url>
    <urn:accessType>ATTACHACCESS-DEFAULT</urn:accessType>
</urn:fileAttachmentList>
```

FileAttachmentContents

Description

The FileAttachmentContents type holds the actual contents of a file that you upload to the server. The FileAttachmentContents type parameters are listed below.

Parameters

Name	Type	Description
checksum	long	This is a checksum that can be verified to ensure contents of the file are correct.
encodedContents	base64binary	Holds the base64 encoded contents of the file.

Usage

Inherited from FileAttachment. FileAttachmentContents is used with FileAttachment to completely describe a file attachment. For more information, see FileAttachment [page 143].

XML

The following XML is a snippet of the FileAttachmentContents type.

```
<urn:checksum></urn:checksum>
    <urn:encodedContents>encoded_content_here</urn:encodedContents>
```

FileContents

Description

The FileContents type holds the actual contents of a file returned from an Export operation or supplied to an Import operation. The FileContents type parameters are listed below.

Parameters

Name	Type	Description
checksum	long	This is a checksum that can be verified to ensure contents of the file are correct.
encodedContents	base64binary	Holds the base64 encoded contents of the file.

Usage

FileContents holds the actual contents of a file and is used to pass a file to or receive a file from the client. For more information on exporting and importing, see Export [page 52] or Import [page 82].

XML

The following XML is a snippet of the FileContents type as seen in an Export call.

```
<urn:applicationID>9</urn:applicationID>
    <urn:xmlExportOptions>
    <urn:checksum></urn:checksum>
    <urn:encodedContents></urn:encodedContents>
</urn:xmlExportOptions>
```

Item

Description

The Item type holds the generic data for an item. The Item type parameters are listed below.

Parameters

Name	Type	Description
itemID	string	Item identifier which is guaranteed to be unique. The UUID provides an alternate identifier.
itemUUID	string	An alternate unique identifier.
itemName	string	Item name for display purposes.

Usage

The itemID is the identifier that can be used in Web service methods to uniquely identify an item. The itemID is composed of the TS_ID of the table, followed by the TS_ID of the item in that table (for example, 1000:164). The itemName is the display name of the item (for example, ENH10234).

XML

The following XML shows Item as seen in a typical call.

```
<urn:genericItem>
    <urn:itemID>1000:164</urn:itemID>
    <urn:itemUUID>e5546f4e-16c9-4c9e-839c-55d6192939b8</urn:itemUUID>
    <urn:itemName>ENH10234</urn:itemName>
</urn:genericItem>
```

ItemLink

Description

The ItemLink type holds information about an item link, which consists of an item ID and the item link type. The ItemLink type parameters are listed below.

Parameters

Name	Type	Description
id	integer	This is the internal TS_ID of the item link from the TS_ATTACHMENTS table.
itemID	string	The ID of the linked item.
linkType	ItemLink-Type [page 109]	Describes the type of item link. See ItemLink-Type [page 109] for a list of possible values.
modificationDateTime	dateTime	The date and time when the item link was last modified. See Supported Date/Time Formats [page 184] for more information.

Name	Type	Description
accessType	Attachment-Access-Type [page 103]	Shows the access type for the item link. The value is either DEFAULT, RESTRICTED, or UNRESTRICTED.

Usage

The itemLink type allows you to specify an item link to another item. The itemID is the identifier that can be used uniquely identify a linked item. The itemID is composed of the TS_ID of the table, followed by the TS_ID of the item in that table (for example, 1000:164).

XML

The following XML snippet shows the ItemLink type in the ItemLinkList parameter of TTItem.

```
<urn:itemLinkList>
    <urn:id>52</urn:id>
    <urn:itemID>1000:143</urn:itemID>
    <urn:linkType>DEFAULT-ITEM-LINK</urn:linkType>
    <urn:modificationDateTime>2008-03-11T22:17:12-07:00
    →</urn:modificationDateTime>
    <urn:accessType>ATTACHACCESS-DEFAULT</urn:accessType>
</urn:itemLinkList>
```

ItemLink-Type

Description

The ItemLink-Type indicates the type of item link.

Parameters

Name	Type	Description
DEFAULT-ITEM-LINK	string	During creation of a new item link, the DEFAULT_ITEM_LINK is equivalent to TWOWAY_NO_TRIGGERS; otherwise, DEFAULT_ITEM_LINK indicates no type change. Note that returned item links always have DEFAULT_ITEM_LINK as the type.
TWOWAY-NO-TRIGGERS	string	Creates a two-way link between the current item and the selected item without Transition Triggers. This is the default link type.
ONEWAY-NO-TRIGGERS	string	Creates a one-way link from the current item to the selected item without Transition Triggers.

Name	Type	Description
ONEWAY-CURRENT-TRIGGERS-LINKED	string	Creates a one-way link from the current item to the selected item. When the current item transitions, it also triggers the selected item to transition.
TWOWAY-CURRENT-TRIGGERS-LINKED	string	Creates a two-way link between the current item and the selected item. When the current item transitions, it also triggers the selected item to transition.
TWOWAY-LINKED-TRIGGERS-CURRENT	string	Creates a two-way link between the current item and the selected item. When the selected item transitions, it also triggers the current item to transition.
TWOWAY-BOTH-TRIGGERS	string	Creates a two-way link between the current item and the selected item. When either item transitions, it also triggers the linked item to transition.

Usage

Linked items can trigger one another based on certain predefined actions defined in your workflow. You can specify a link type that triggers items, but triggers fire only if they are configured for your workflow.

XML

The following XML shows ItemLink-Type as seen in a typical call.

```
<urn:itemLinkList>
    <urn:id>52</urn:id>
    <urn:itemID>1000:143</urn:itemID>
    <urn:linkType>DEFAULT-ITEM-LINK</urn:linkType>
    <urn:modificationDateTime>2008-03-11T22:17:12-07:00
    →</urn:modificationDateTime>
    <urn:accessType>ATTACHACCESS-DEFAULT</urn:accessType>
</urn:itemLinkList>
```

NameValue

Description

The NameValue type holds a field name with either a single value or a list of values. In the WSDL, the value element is converted to a choice element: value or values. The NameValue parameters are listed below.

Parameters

Name	Type	Description
name	string	The unique field name (database field name) that can be obtained with the **GetTables** call.
id	integer	The unique internal field id (TS_ID from TS_FIELDS) that can be obtained with the **GetTables** call.
uuid	string	Alternate unique field id (TS_UUID from TS_FIELDS) that can be obtained with the **GetTables** function
setValueBy	Set-Value-By [page 119]	Determines how to set the value (which value takes precedence).
setValueMethod	Set-Value-Method [page 120]	Use setValueMethod to append, remove, or replace a value.
value	Value or Values [page 133]	The value or values for the field.

Usage

When secifying a value or values, you can use either the display, internal, or UUID value(s). In addition, you can use the setValueBy element to specify which type of value it is, though it is not required. The setValueBy element is mainly used if you are passing in an empty value. Otherwise, Business Mashups will determine which type of value is set by checking for a non-empty value.

To set an empty value, you must use the SET-VALUE-BY parameter that corresponds to the empty `<value>` parameter. For example, to set an empty value for a field using the internalValue parameter, you could specify:

```
<urn:setValueBy>INTERNAL-VALUE</urn:setValueBy>
    <urn:value>
        <urn:internalValue></urn:internalValue>
```

Alternatively, to set an empty value for a field using the displayValue parameter, you could could specify:

```
<urn:setValueBy>DISPLAY-VALUE</urn:setValueBy>
    <urn:value>
        <urn:displayValue></urn:displayValue>
```

Note: To set an empty value for a single or Multi-Relational field, you must use INTERNAL-VALUE in SetValueBy and empty tags in the internalValue parameter.

XML

The following XML shows the NameValue type as seen in a typical call.

```
<urn:extendedFieldList>
    <urn:name>Product</urn:name>
    <urn:value>
        <urn:displayValue>Animator 2000</urn:displayValue>
    </urn:value>
</urn:extendedFieldList>
<urn:extendedFieldList>
    <urn:name>Company</urn:name>
    <urn:value>
        <urn:displayValue>ACME</urn:displayValue>
    </urn:value>
</urn:extendedFieldList>
<urn:extendedFieldList>
    <urn:name>Level 1 Tech</urn:name>
    <urn:value>
        <urn:displayValue>Chad Support</urn:displayValue>
    </urn:value>
</urn:extendedFieldList>
<urn:extendedFieldList>
    <urn:name>Submit__Close</urn:name>
    <urn:value>
        <urn:displayValue>No</urn:displayValue>
    </urn:value>
</urn:extendedFieldList>
```

The following XML shows the NameValue type with multiple values as seen in a typical call.

```
<urn:extendedFieldList>
   <urn:name>MULTI_USER</urn:name>
   <urn:id>178</urn:id>
   <urn:uuid>f62c6b63-2531-441a-9fff-9cd471bc61ca</urn:uuid>
   <urn:setValueBy>PRECEDENCE-VALUE</urn:setValueBy>
   <urn:setValueMethod>APPEND-VALUES</urn:setValueMethod>
   <urn:values>
       <urn:displayValue>admin</urn:displayValue>
       <urn:internalValue>1</urn:internalValue>
       <urn:uuid>d2d60592-656e-4103-a20d-f12da9305fe4</urn:uuid>
   </urn:values>
   <urn:values>
       <urn:displayValue>bill</urn:displayValue>
       <urn:internalValue>10</urn:internalValue>
       <urn:uuid>7130c9c3-abb6-41f5-bd7a-30c40f47b824</urn:uuid>
   </urn:values>
   <urn:values>
       <urn:displayValue>carmen</urn:displayValue>
       <urn:internalValue>11</urn:internalValue>
       <urn:uuid>9d71b19e-9b72-4731-bec3-3eba938da0de</urn:uuid>
   </urn:values>
</urn:extendedFieldList>
```

Using the example above, if this payload was sent via UpdateItem, the call would append admin, bill, and carmen to the current selections that exist in the "MULTI-USER" field.

Note

Description

The Note type holds information about a note. The Note type parameters are listed below.

Parameters

Name	Type	Description
id	integer	This is the internal TS_ID of the note from the TS_ATTACHMENTS table.
title	string	The title of the note.
note	string	The text of the note.
modificationDateTime	dateTime	The date and time when the note was last modified. See Supported Date/Time Formats [page 184] for more information.
accessType	Attachment-Access-Type [page 103]	Shows the access type for the note. The value is either DEFAULT, RESTRICTED, or UNRESTRICTED.

Usage

The note title is limited to 255 unicode characters. The note body is limited to 65,535 characters.

XML

The following XML snippet shows the Note type in the NoteList parameter of TTItem.

```
<urn:noteList>
    <urn:id>54</urn:id>
    <urn:title>Attention</urn:title>
    <urn:note>This is a note</urn:note>
    <urn:modificationDateTime>2008-03-11T22:17:15-07:00
    →</urn:modificationDateTime>
    <urn:accessType>ATTACHACCESS-DEFAULT</urn:accessType>
</urn:noteList>
```

QueryRange

Description

The QueryRange type allows you to limit the number of records returned in a query. The QueryRange type parameters are listed below.

Parameters

Name	Type	Description
startIndex	integer	This is the first record where the query should start.

Name	Type	Description
fetchSize	integer	The *fetchSize* number is used to limit the number records to return from the entire set of records in the *totalCount*.
totalCount	integer	The number of all the records in the query.

Usage

You can limit the number of reports returned in GetReports [page 69] using QueryRange. For example, if GetReports returns 20 items without a QueryRange, you can set startIndex to 5 and fetchSize to 7 to return reports 5 through 12 in the list of available reports. You can also apply QueryRange when making the RunReport call to limit the number of records returned via the fetchSize.

XML

The following XML snippet shows QueryRange as seen in a typical call.

```
<urn:queryRange>
    <urn:startIndex>5</urn:startIndex>
    <urn:fetchSize>7</urn:fetchSize>
    <urn:totalCount></urn:totalCount>
</urn:queryRange>
```

ReportAccessLevel

Description

ReportAccessLevel is used to describe the access level assigned to a report. The available report access levels are listed below.

Parameters

Name	Type	Description
PRIVATE	string	This access level enables individual users to manage reports they create. Only the user who creates a private report can access, modify, or delete private reports and only if this user is granted "Manage Private Reports" privileges.
GUEST	string	This access level denotes a guest-level report. Users with guest-level report privileges can perform report actions for guest-level reports.
USER	string	This access level denotes a user-level report. Users with user-level report privileges can perform report actions for user-level reports.
MANAGER	string	This access level denotes a manager-level report. Users with manager-level report privileges can perform report actions for manager-level reports.

Usage

ReportAccessLevel is used to limit the reports returned in the reportsFilter argument of the GetReports call. If ReportAccessLevel is not specified, the GetReports response will not be limited by any type of access level. See ReportsFilter [page 116] for further usage.

XML

The following XML snippet shows ReportAccessLevel as seen in a typical call.

```
<urn:reportsFilter>
    <urn:solutionID>1000</urn:solutionID>
    <urn:solutionName>Issues</urn:solutionName>
    <urn:projectID>4</urn:projectID>
    <urn:projectName></urn:projectName>
    <urn:authorID>joe</urn:AuthorID>
    <urn:reportType>LISTING</urn:reportType>
    <urn:reportCategory>QUICKLINKS<urn:reportCategory>
    <urn:reportAccessLevel>USER</urn:reportAccessLevel>
    <urn:reportName></urn:reportName>
    <urn:includeSubProjects>true</urn:includeSubProjects>
    <urn:createdDateFrom>2007-06-20T15:35:38-07:00</urn:createdDateFrom>
    <urn:createdDateTo>2007-07-20T15:35:38-07:00</urn:createdDateTo>
</urn:reportsFilter>
```

ReportCategory

Description

ReportCategory is used to describe a grouping of reports. The available report categories are listed below.

Parameters

Name	Type	Description
ALL	string	This is the default *ReportCategory*. No filter is applied and all reports should be returned.
APPLICATION	string	Return only Application Reports (mashup-specific listing reports designed in Mashup Composer).
BUILTIN	string	Return only built-in reports.
MY	string	Return all reports that display in **My Reports** in the Web interface. Contains all reports that you have authored.
QUICKLINKS	string	Return only reports that are saved as Quick Links.
USERREPORTS	string	Return all reports, with the exception of built-in reports.

Usage

ReportCategory is used to limit the reports returned in the reportsFilter argument of the GetReports call. The default setting is "ALL" which will returns all reports (no filter is applied via this parameter in that case). See ReportsFilter [page 116] for further usage.

XML

The following XML snippet shows ReportCategory as seen in a typical call.

```
<urn:reportsFilter>
    <urn:solutionID>1000</urn:solutionID>
    <urn:solutionName>Issues</urn:solutionName>
    <urn:projectID>4</urn:projectID>
    <urn:projectName></urn:projectName>
    <urn:authorID>joe</urn:AuthorID>
    <urn:reportType>LISTING</urn:reportType>
    <urn:reportCategory>QUICKLINKS<urn:reportCategory>
    <urn:reportAccessLevel>USER</urn:reportAccessLevel>
    <urn:reportName></urn:reportName>
    <urn:includeSubProjects>true</urn:includeSubProjects>
    <urn:createdDateFrom>2007-06-20T15:35:38-07:00</urn:createdDateFrom>
    <urn:createdDateTo>2007-07-20T15:35:38-07:00</urn:createdDateTo>
</urn:reportsFilter>
```

ReportsFilter

Description

The ReportsFilter type allows you to filter the reports returned via the GetReports call. The ReportsFilter type parameters are listed below.

Parameters

Name	Type	Description
solutionID	integer	This is the internal TS_ID of the solution from TS_SOLUTIONS that the report is based on.
solutionName	string	This is the name of the solution that the report is based on (TS_NAME from TS_SOLUTIONS). If solutionID is also specified, then solutionID will take precedence.
projectID	integer	This is the internal TS_ID of the project from TS_PROJECTS that the report was created against.
projectName	string	The fully qualified name of the project.
projectUUID	string	This is the alternate unique ID of the project that the report was created against.

Name	Type	Description
tableID	integer	This is the internal TS_ID of the table from TS_TABLES that the report is based on.
tableName	string	This is the display name of the table that the report is based on (TS_NAME from TS_TABLES).
authorID	string	Login ID of the user who authored the report.
reportType	ReportType [page 118]	An enumeration that indicates the type of report. The default value is 1 for LISTING.
reportCategory	ReportCategory [page 115]	A broader enumeration that limits the response based on the category of report (built-in reports, application reports, reports you authored). The default value is "ALL."
reportAccessLevel	ReportAccessLevel [page 114]	An enumeration that limits the response based on the report's access level (PRIVATE, GUEST, USER, or MANAGER).
reportName	string	The name of the report.
includeSubProjects	boolean	This flag indicates whether or not to include reports that are based on sub-projects of the project specified in *projectID*. The default value is "false."
createdDateFrom	dateTime	Filter reports created after this date.
createdDateTo	dateTime	Filter reports created before this date.

Usage

You can filter the reports returned in GetReports [page 69] using ReportsFilter. For example, you can specify "joe" in the authorID parameter to only return reports that Joe created. Keep in mind that projects do no exist for auxiliary tables so projectID and projectName should be left empty when searching for reports against auxiliary tables. You do not need to specify both solutionID and solutionName. If both are specified, solutionID will be used.

Tip: You can easily find the proper solutionName by looking at the Internal Name of the application in Mashup Composer.

Note: You can treat the reportName as a search parameter and enter only part of the report name. In the example below, "All" is sent in reportName to filter on reports that include the word "All" in the report title. See the XML in RunReportResult [page 158] for an example of the response.

XML

The following XML snippet shows ReportsFilter as seen in a typical call.

```
<urn:reportsFilter>
    <urn:solutionID>1</urn:solutionID>
    <urn:solutionName>BASE_ISSUE_DEFECT_TRACKING</urn:solutionName>
    <urn:projectID>4</urn:projectID>
    <urn:projectName>Base Project||Base IDT Project||Software Development
    →</urn:projectName>
    <urn:projectUUID>57e13982-ea0f-4424-bbbc-377e9d909772</urn:projectUUID>
    <urn:tableID>1000</urn:tableID>
    <urn:tableName>Issues</urn:tableName>
    <urn:authorID>joe</urn:AuthorID>
    <urn:reportType>1</urn:reportType>
    <urn:reportCategory>MY<urn:reportCategory>
    <urn:reportAccessLevel>USER</urn:reportAccessLevel>
    <urn:reportName>All</urn:reportName>
    <urn:includeSubProjects>true</urn:includeSubProjects>
    <urn:createdDateFrom>2007-06-20T15:35:38-07:00</urn:createdDateFrom>
    <urn:createdDateTo>2007-07-20T15:35:38-07:00</urn:createdDateTo>
</urn:reportsFilter>
```

ReportType

Description

ReportType is used to describe the type of report. The available report types are listed below.

Parameters

Name	Type	Description
LISTING	string	Indicates a Listing report type.

Usage

ReportType is used to limit the types of reports returned in the reportsFilter argument of the GetReports call. The default setting is "1" or "LISTING" to indicate a Listing report. See ReportsFilter [page 116] for further usage.

XML

The following XML snippet shows ReportType as seen in a typical call.

```
<urn:reportsFilter>
    <urn:solutionID>1</urn:solutionID>
    <urn:solutionName>BASE_ISSUE_DEFECT_TRACKING</urn:solutionName>
    <urn:projectID>4</urn:projectID>
    <urn:projectName></urn:projectName>
    <urn:authorID>joe</urn:AuthorID>
    <urn:reportType>LISTING</urn:reportType>
    <urn:reportLevel>ALL</urn:reportLevel>
    <urn:reportName></urn:reportName>
    <urn:includeSubProjects>true</urn:includeSubProjects>
```

```
    <urn:createdDateFrom>2007-06-20T15:35:38-07:00</urn:createdDateFrom>
    <urn:createdDateTo>2007-07-20T15:35:38-07:00</urn:createdDateTo>
</urn:reportsFilter>
```

Set-Value-By

Description

Set-Value-By indicates how the value in the NameValue type should be set on an update or create.

Parameters

Name	Type	Description
PRECEDENCE-VALUE	string	If the PRECENDENCE-VALUE is chosen, then the value will be set by looking at the values supplied in order of precedence.
UUID-VALUE	string	Set the value using the UUID value.
INTERNAL-VALUE	string	Set the value using the internal value
DISPLAY-VALUE	string	Set the value using the display value

Usage

If the PRECENDENCE-VALUE is chosen or the Set-Value-By parameter is omitted, then the value will be set by looking at the values supplied in order of precedence. The order is internal, uuid, followed by display. For example, if the internal value is present, it will be used. If the internal value is not present, the uuid value will be used. If neither the internal or uuid values are specified, the display value will be used.

You can use the setValueBy element to specify which type of value it is, though it is not required. The setValueBy element is mainly used if you are passing in an empty value. Otherwise, Business Mashups will determine which type of value is set by checking for a non-empty value.

To set an empty value, you must use the SET-VALUE-BY parameter that corresponds to the empty `<value>` parameter. For example, to set an empty value for a field using the internalValue parameter, you could specify:

```
<urn:setValueBy>INTERNAL-VALUE</urn:setValueBy>
    <urn:value>
        <urn:internalValue></urn:internalValue>
```

Alternatively, to set an empty value for a field using the displayValue parameter, you could could specify:

```
<urn:setValueBy>DISPLAY-VALUE</urn:setValueBy>
    <urn:value>
        <urn:displayValue></urn:displayValue>
```

Note: To set an empty value for a Single or Multi-Relational field, you must use INTERNAL-VALUE in SetValueBy and empty tags in the internalValue parameter.

XML

The following XML shows Set-Value-By as seen in a typical call.

```
<urn:extendedFieldList>
    <urn:name>SINGLE_SELECTION</urn:name>
    <urn:id>182</urn:id>
    <urn:uuid>5dcf0aa8-22d6-4207-a88e-8e989df948df</urn:uuid>
    <urn:setValueBy>PRECEDENCE-VALUE</urn:setValueBy>
    <urn:value>
        <urn:displayValue>yellow</urn:displayValue>
        <urn:internalValue>122</urn:internalValue>
        <urn:uuid>4ad0961d-30dc-4198-8167-5224dcb6c065</urn:uuid>
    </urn:value>
</urn:extendedFieldList>
```

Set-Value-Method

Description

Set-Value-Method indicates how to process values in a text field or any field that contains multiple values such as Multi-Selection, Multi-Relational, Multi-Group, or Multi-User.

Parameters

Name	Type	Description
REPLACE-VALUES	string	If you choose REPLACE-VALUES, Set-Value-Method replaces the current value. Applies to all fields. If a Set-Value-Method is not specified, REPLACE-VALUES is assumed by default.
APPEND-VALUES	string	If you choose APPEND-VALUES, Set-Value-Method appends the value to the previous entry. Applies to Text and all multi-value fields.
REMOVE-VALUES	string	If you choose REMOVE-VALUES, Set-Value-Method removes the value. Only applies to multi-value fields.

Usage

You can use Set-Value-Method to specify how text fields or any field that holds multiple values should process the value or values that are passed in the setValueBy argument. For example, you could use this argument to *replace* one or more values in a Multi-Relational field, *append* new text to a Journal field, or *remove* specified values from a Multi-Selection list. Specifying a Set-Value-Method parameter is useful if you plan to have multiple Web service calls updating the same field on the same item. By using Set-Value-Method, field values can be managed dynamically; there is no need to ensure that the last

update made contains the definitive selections for a given field. Otherwise, if Set-Value-Method is not supplied, the values will be replaced or an empty value will be set, depending on how the value or values are sent.

XML

The following XML snippets show each Set-Value-Method as seen in a typical call, with a brief description of the results.

```
<urn:extendedFieldList>
    <urn:name>MY_TEXT_FIELD</urn:name>
    <urn:setValueBy>INTERNAL-VALUE</urn:setValueBy>
    <urn:setValueMethod>REPLACE-VALUES</urn:setValueMethod>
    <urn:value>
        <urn:displayValue>11</urn:displayValue>
        <urn:internalValue>11</urn:internalValue>
    </urn:value>
</urn:extendedFieldList>
<urn:extendedFieldList>
    <urn:name>MY_TEXT_FIELD</urn:name>
    <urn:setValueBy>INTERNAL-VALUE</urn:setValueBy>
    <urn:setValueMethod>APPEND-VALUES</urn:setValueMethod>
    <urn:value>
        <urn:displayValue>22</urn:displayValue>
        <urn:internalValue>22</urn:internalValue>
    </urn:value>
</urn:extendedFieldList>
```

This call would replace the current text field value with "11" and then append "22" to that value:

```
    11 22
```

```
<urn:extendedFieldList>
    <urn:name>WCRS</urn:name>
    <urn:setValueBy>INTERNAL-VALUE</urn:setValueBy>
    <urn:setValueMethod>REPLACE-VALUES</urn:setValueMethod>
    <urn:values>
        <urn:displayValue>ZMF00026: Release Package - Construction</urn:displayValue>
        <urn:internalValue>1015:10</urn:internalValue>
        <urn:uuid>af93aad0-8245-4e49-ac91-574725f7041f</urn:uuid>
    </urn:values>
    <urn:values>
        <urn:displayValue>ZMF00028: Release Package - Construction</urn:displayValue>
        <urn:internalValue>1015:11</urn:internalValue>
        <urn:uuid>e16d9550-c2bc-41b9-99c6-f716cb5a7022</urn:uuid>
    </urn:values>
</urn:extendedFieldList>
```

This call would replace the current selections in the Multi-Relational WCRS field with two new records (10 and 11) from table 1015.

```
<urn:extendedFieldList>
    <urn:name>WCRS</urn:name>
```

```
    <urn:setValueBy>INTERNAL-VALUE</urn:setValueBy>
    <urn:setValueMethod>APPEND-VALUES</urn:setValueMethod>
    <urn:values>
        <urn:displayValue>ZMF00032: Relase Unit - Construction</urn:displayValue>
        <urn:internalValue>1015:12</urn:internalValue>
        <urn:uuid>15ce4bc2-347e-41bb-acaf-f3dd434a9633</urn:uuid>
    </urn:values>
    <urn:values>
        <urn:displayValue>ZMF00033: Relase Unit - Construction</urn:displayValue>
        <urn:internalValue>1015:13</urn:internalValue>
        <urn:uuid>5e8d1cb0-b177-4aa5-ae8c-bd169e31f6da</urn:uuid>
    </urn:values>
</urn:extendedFieldList>
```

This call would add two additional selections to the Multi-Relational WCRS field. Records 12 and 13 from table 1015 would now be selected in this field along with the previous selections that were made in the field.

```
<urn:extendedFieldList>
    <urn:name>WCRS</urn:name>
    <urn:setValueBy>INTERNAL-VALUE</urn:setValueBy>
    <urn:setValueMethod>REMOVE-VALUES</urn:setValueMethod>
    <urn:values>
        <urn:displayValue>ZMF00032: Relase Unit - Construction</urn:displayValue>
        <urn:internalValue>1015:12</urn:internalValue>
        <urn:uuid>15ce4bc2-347e-41bb-acaf-f3dd434a9633</urn:uuid>
    </urn:values>
    <urn:values>
        <urn:displayValue>ZMF00026: Release Package - Construction</urn:displayValue>
        <urn:internalValue>1015:10</urn:internalValue>
        <urn:uuid>af93aad0-8245-4e49-ac91-574725f7041f</urn:uuid>
    </urn:values>
</urn:extendedFieldList>
```

This call would simply remove records 10 and 12 from the Multi-Relational WCRS field.

```
<urn:extendedFieldList>
    <urn:name>COLORS</urn:name>
    <urn:id>178</urn:id>
    <urn:setValueBy>INTERNAL-VALUE</urn:setValueBy>
    <urn:setValueMethod>REMOVE-VALUES</urn:setValueMethod>
    <urn:values>
        <urn:displayValue>Red</urn:displayValue>
        <urn:internalValue>5</urn:internalValue>
    </urn:values>
    <urn:values>
        <urn:displayValue>Green</urn:displayValue>
        <urn:internalValue>6</urn:internalValue>
    </urn:values>
</urn:extendedFieldList>
```

This call would remove the "Red" and "Green" selections from the Multi-Selection "Colors" field.

Table-Type

Description

Table-Type indicates the type of table. The available table types are listed below.

Parameters

Name	Type	Description
NOT-SPECIFIED	string	Used to not specify a table type.
SYSTEM-TABLE	string	A table type that is inherent to any Business Mashups database.
PRIMARY-TABLE	string	A table that stores information about primary items, which follow an application workflow process.
AUXILIARY-TABLE	string	A table that stores information that may be needed repeatedly. Stores records that do not follow an application workflow process.
SYSTEM-AUXILIARY-TABLE	string	An built-in auxiliary table provided by Business Mashups
ARCHIVE-TABLE	string	A built-in table used to store archived primary or auxiliary records.

Usage

Table-Type can be used to limit the types of tables returned in GetTables [page 74]. It can also be used to identify the type of table returned in the GetTables response.

XML

The following XML shows Table-Type as seen in a typical GetTables call.

```
<urn:GetTables>
    <urn:auth>
        <urn:userId>admin</urn:userId>
        <urn:password></urn:password>
        <urn:hostname>localhost</urn:hostname>
    </urn:auth>
    <urn:solutionID>1</urn:solutionID>
    <urn:tableType>PRIMARY-TABLE</urn:tableType>
</urn:GetTables>
```

TTItem

Description

The TTItem type holds all of the Business Mashups field information for an item along with attached notes and item links. The TTItem type parameters are listed below.

Parameters

Name	Type	Description
genericItem	Item [page 107]	The generic item containing the unique and display identifiers.
itemType	string	Describes items, such as Defects or Change Requests, tracked in a workflow.
classification	string	Otherwise known as project. *Classification* is used to declare the project the item belongs to.
classificationUUID	string	Alternate unique identifier (not the TS_ID) for a project.
title	string	The 80 character fixed length title of the item.
description	string	A description of the item. This is the value given to the system Description field, which is a Text/Memo field.
createdBy	string	The login ID of the user that created the item.
createDate	dateTime	The date the item was created. See Supported Date/Time Formats [page 184] for more information.
modifiedBy	string	The login ID of the user that last modified the item.
modifiedDate	dateTime	The date and time when the item was last modified. See Supported Date/Time Formats [page 184] for more information.
activeInactive	boolean	Indicates whether the item is active (true) or inactive (false). Defaults to true.
state	string	The current state of the item.
owner	string	The login ID of the current owner of the item.
url	string	The exact URL of the item.
extendedFieldList	NameValue [page 110]	Additional fields and values not otherwise specified in *TTItem* are set using the *extendedFieldList*.
noteList	Note [page 113]	The list of notes attached to the item.

Name	Type	Description
itemLinkList	ItemLink [page 108]	The list of items linked to this item.
urlAttachmentList	URLAttachment [page 132]	The list of URL attachments associated with this item.
fileAttachmentList	FileAttachment [page 105]	The list of file attachments associated with this item.

Usage

The TTitem type is essentially used describe the fields of an auxiliary or primary item in Business Mashups. The following TTitem parameters are described in further detail:

- **itemType**

  The itemType parameter is used to describe the item. The available values for itemType are derived from selections in the Item Type field. When you create an Item Type value in Mashup Composer, you create a value and assign it a prefix. In the itemType parameter, you enter only the value, not the prefix. See the XML sample below for an example. Item Type is only applicable to primary table items.

- **classification**

  Known as **project** elsewhere in Business Mashups, the classification parameter is used to assign an item to a project. When creating items using Web services, make sure the proper classification is used with respect to the itemtype, state, and owner. Classification is only applicable to primary table items.

- **state**

  The available states for an item are determined by the project the item is in. If you change the value of the state field for a specific item, you are also moving that item to the specified state in the workflow and it could follow a different process than you intended. If not specified, the initial state is the default. State is only applicable to primary table items.

- **owner**

  The available owner of an item is determined by the state the item is in. If not explicitly set, the value for owner will be derived from the value of the user field designated as the owner of the state. Owner is only applicable to primary table items.

- **url**

  The detailed view of the item can be accessed via this URL. The URL is of the form `"tmtrack.dll?IssuePage&RecordId=10&Template=view&TableId=1002"`, so the `"http://host:port/tmtrack/"` must be pre-pended.

- **extendedFieldList**

  The extendedFieldList is a collection of fields and field values that aren't specifically set elsewhere in TTItem. The XML example below shows each field type in Business Mashups, sent via TTitem in a CreatePrimaryItem call. The example shows how the data for each field type should be entered, using either the display or internal value.

Both the display and internal values are shown in the example, though you can specify either the display, internal, or UUID value instead. Note that Multi-Select and Multi-Relational fields are included as well, with each value being sent in its own element.

Tip: In order to create a new auxiliary or primary item with a file attachment, that attachment must exist in the TS_ATTACHMENTS table of Business Mashups already. Otherwise, you can use CreateFileAttachment after the item has been created to associate it with a new attachment that isn't already in TS_ATTACHMENTS.

XML

The following XML shows TTitem as seen in a typical call. Throughout the extendedFieldList in this example, Business Mashups field types are used in the name element to illustrate how to format the data for that given field type.

```
<urn:CreatePrimaryItem>
    <urn:auth>
        <urn:userId>admin</urn:userId>
        <urn:password>password</urn:password>
        <urn:hostname>localhost</urn:hostname>
        <urn:loginAsUserId></urn:loginAsUserId>
    </urn:auth>
    <urn:projectID>19</urn:projectID>
        <urn:item>
            <urn:genericItem>
                <urn:itemID>1005:6</urn:itemID>
                <urn:itemUUID></urn:itemUUID>
                <urn:itemName>BUG01234</urn:itemName>
            </urn:genericItem>
            <urn:itemType>Bug Report</urn:itemType>
            <urn:classification>MyProject</urn:classification>
            <urn:classificationUUID></urn:classificationUUID>
            <urn:title>My Title</urn:title>
            <urn:description>My description</urn:description>
            <urn:createdBy>admin</urn:createdBy>
            <urn:createDate>2008-05-01T17:40:40-06:00</urn:createDate>
            <urn:modifiedBy>admin</urn:modifiedBy>
            <urn:modifiedDate>2008-05-01T18:30:48-06:00</urn:modifiedDate>
            <urn:activeInactive>true</urn:activeInactive>
            <urn:state>New</urn:state>
            <urn:owner>Bill</urn:owner>
            <urn:url></urn:url>
            <urn:extendedFieldList>
                <urn:name>BINARY_TRINARY</urn:name>
                <urn:id>169</urn:id>
                <urn:uuid></urn:uuid>
                <urn:setValueBy>PRECEDENCE-VALUE</urn:setValueBy>
                <urn:setValueMethod></urn:setValueMethod>
                <urn:value>
                    <urn:displayValue>true</urn:displayValue>
                    <urn:internalValue>0</urn:internalValue>
                    <urn:uuid></urn:uuid>
                </urn:value>
            </urn:extendedFieldList>
```

```
<urn:extendedFieldList>
    <urn:name>DATE_AND_TIME</urn:name>
    <urn:id>170</urn:id>
    <urn:uuid></urn:uuid>
    <urn:setValueBy>PRECEDENCE-VALUE</urn:setValueBy>
    <urn:setValueMethod></urn:setValueMethod>
    <urn:value>
        <urn:displayValue>2008-05-01T06:00:00+00:00
        →</urn:displayValue>
        <urn:internalValue>2008-05-01T06:00:00+00:00
        →</urn:internalValue>
        <urn:uuid/>
    </urn:value>
</urn:extendedFieldList>
<urn:extendedFieldList>
    <urn:name>DATE_ONLY</urn:name>
    <urn:id>171</urn:id>
    <urn:uuid></urn:uuid>
    <urn:setValueBy>PRECEDENCE-VALUE</urn:setValueBy>
    <urn:setValueMethod></urn:setValueMethod>
    <urn:value>
        <urn:displayValue>2008-05-01</urn:displayValue>
        <urn:internalValue>2008-05-01T00:00:00+00:00
        →</urn:internalValue>
        <urn:uuid/>
    </urn:value>
</urn:extendedFieldList>
<urn:extendedFieldList>
    <urn:name>TIME_OF_DAY</urn:name>
    <urn:id>172</urn:id>
    <urn:uuid></urn:uuid>
    <urn:setValueBy>PRECEDENCE-VALUE</urn:setValueBy>
    <urn:setValueMethod></urn:setValueMethod>
    <urn:value>
        <urn:displayValue>17:36:39</urn:displayValue>
        <urn:internalValue>63399</urn:internalValue>
        <urn:uuid/>
    </urn:value>
</urn:extendedFieldList>
<urn:extendedFieldList>
    <urn:name>ELAPSED_TIME</urn:name>
    <urn:id>173</urn:id>
    <urn:uuid></urn:uuid>
    <urn:setValueBy>PRECEDENCE-VALUE</urn:setValueBy>
    <urn:setValueMethod></urn:setValueMethod>
    <urn:value>
        <urn:displayValue>7 17:36:39</urn:displayValue>
        <urn:internalValue>668199</urn:internalValue>
        <urn:uuid/>
    </urn:value>
</urn:extendedFieldList>
<urn:extendedFieldList>
    <urn:name>FOLDER</urn:name>
    <urn:id>174</urn:id>
    <urn:uuid></urn:uuid>
```

```
            <urn:setValueBy>PRECEDENCE-VALUE</urn:setValueBy>
            <urn:setValueMethod></urn:setValueMethod>
            <urn:value>
                <urn:displayValue>InBox</urn:displayValue>
                <urn:internalValue>3</urn:internalValue>
                <urn:uuid></urn:uuid>
            </urn:value>
        </urn:extendedFieldList>
        <urn:extendedFieldList>
            <urn:name>MULTI_GROUP</urn:name>
            <urn:id>175</urn:id>
            <urn:uuid></urn:uuid>
            <urn:setValueBy>PRECEDENCE-VALUE</urn:setValueBy>
            <urn:setValueMethod></urn:setValueMethod>
            <urn:values>
                <urn:displayValue>Administrator</urn:displayValue>
                <urn:internalValue>2</urn:internalValue>
                <urn:uuid></urn:uuid>
            </urn:values>
            <urn:values>
                <urn:displayValue>IDT Engineers</urn:displayValue>
                <urn:internalValue>4</urn:internalValue>
                <urn:uuid></urn:uuid>
            </urn:values>
            <urn:values>
                <urn:displayValue>Everyone</urn:displayValue>
                <urn:internalValue>5</urn:internalValue>
                <urn:uuid></urn:uuid>
            </urn:values>
        </urn:extendedFieldList>
        <urn:extendedFieldList>
            <urn:name>MULTI_RELATIONAL</urn:name>
            <urn:id>176</urn:id>
            <urn:uuid></urn:uuid>
            <urn:setValueBy>PRECEDENCE-VALUE</urn:setValueBy>
            <urn:setValueMethod></urn:setValueMethod>
            <urn:values>
                <urn:displayValue>00016:primaryitem1</urn:displayValue>
                <urn:internalValue>1005:2</urn:internalValue>
                <urn:uuid></urn:uuid>
            </urn:values>
            <urn:values>
                <urn:displayValue>00017:primaryitem2</urn:displayValue>
                <urn:internalValue>1005:5</urn:internalValue>
                <urn:uuid></urn:uuid>
            </urn:values>
        </urn:extendedFieldList>
        <urn:extendedFieldList>
            <urn:name>MULTI_SELECTION</urn:name>
            <urn:id>177</urn:id>
            <urn:uuid></urn:uuid>
            <urn:setValueBy>PRECEDENCE-VALUE</urn:setValueBy>
            <urn:setValueMethod></urn:setValueMethod>
            <urn:values>
                <urn:displayValue>red</urn:displayValue>
```

```
            <urn:internalValue>114</urn:internalValue>
            <urn:uuid></urn:uuid>
        </urn:values>
        <urn:values>
            <urn:displayValue>blue</urn:displayValue>
            <urn:internalValue>115</urn:internalValue>
            <urn:uuid></urn:uuid>
        </urn:values>
        <urn:values>
            <urn:displayValue>green</urn:displayValue>
            <urn:internalValue>116</urn:internalValue>
            <urn:uuid></urn:uuid>
        </urn:values>
    </urn:extendedFieldList>
    <urn:extendedFieldList>
        <urn:name>MULTI_USER</urn:name>
        <urn:id>178</urn:id>
        <urn:uuid></urn:uuid>
        <urn:setValueBy>PRECEDENCE-VALUE</urn:setValueBy>
        <urn:setValueMethod></urn:setValueMethod>
        <urn:values>
            <urn:displayValue>admin</urn:displayValue>
            <urn:internalValue>1</urn:internalValue>
            <urn:uuid></urn:uuid>
        </urn:values>
        <urn:values>
            <urn:displayValue>bill</urn:displayValue>
            <urn:internalValue>10</urn:internalValue>
            <urn:uuid></urn:uuid>
        </urn:values>
        <urn:values>
            <urn:displayValue>carmen</urn:displayValue>
            <urn:internalValue>11</urn:internalValue>
            <urn:uuid></urn:uuid>
        </urn:values>
    </urn:extendedFieldList>
    <urn:extendedFieldList>
        <urn:name>NUMERIC</urn:name>
        <urn:id>179</urn:id>
        <urn:uuid></urn:uuid>
        <urn:setValueBy>PRECEDENCE-VALUE</urn:setValueBy>
        <urn:setValueMethod></urn:setValueMethod>
        <urn:value>
            <urn:displayValue>55</urn:displayValue>
            <urn:internalValue>55</urn:internalValue>
            <urn:uuid/>
        </urn:value>
    </urn:extendedFieldList>
    <urn:extendedFieldList>
        <urn:name>SINGLE_RELATIONAL</urn:name>
        <urn:id>180</urn:id>
        <urn:uuid></urn:uuid>
        <urn:setValueBy>PRECEDENCE-VALUE</urn:setValueBy>
        <urn:setValueMethod></urn:setValueMethod>
        <urn:value>
```

```
            <urn:displayValue>00016:primaryitem1</urn:displayValue>
            <urn:internalValue>1005:2</urn:internalValue>
            <urn:uuid></urn:uuid>
        </urn:value>
    </urn:extendedFieldList>
    <urn:extendedFieldList>
        <urn:name>SINGLE_SELECTION</urn:name>
        <urn:id>182</urn:id>
        <urn:uuid></urn:uuid>
        <urn:setValueBy>PRECEDENCE-VALUE</urn:setValueBy>
        <urn:setValueMethod></urn:setValueMethod>
        <urn:value>
            <urn:displayValue>yellow</urn:displayValue>
            <urn:internalValue>122</urn:internalValue>
            <urn:uuid></urn:uuid>
        </urn:value>
    </urn:extendedFieldList>
    <urn:extendedFieldList>
        <urn:name>NUMERIC_2_FOR_SUM</urn:name>
        <urn:id>183</urn:id>
        <urn:uuid></urn:uuid>
        <urn:setValueBy>PRECEDENCE-VALUE</urn:setValueBy>
        <urn:setValueMethod></urn:setValueMethod>
        <urn:value>
            <urn:displayValue>10</urn:displayValue>
            <urn:internalValue>10</urn:internalValue>
            <urn:uuid/>
        </urn:value>
    </urn:extendedFieldList>
           <urn:extendedFieldList>
        <urn:name>SUMMATION</urn:name>
        <urn:id>184</urn:id>
        <urn:uuid></urn:uuid>
        <urn:setValueBy>PRECEDENCE-VALUE</urn:setValueBy>
        <urn:setValueMethod></urn:setValueMethod>
        <urn:value>
            <urn:displayValue>65</urn:displayValue>
            <urn:internalValue>65</urn:internalValue>
            <urn:uuid/>
        </urn:value>
    </urn:extendedFieldList>
    <urn:extendedFieldList>
        <urn:name>TEXT</urn:name>
        <urn:id>185</urn:id>
        <urn:uuid></urn:uuid>
        <urn:setValueBy>PRECEDENCE-VALUE</urn:setValueBy>
        <urn:setValueMethod></urn:setValueMethod>
        <urn:value>
            <urn:displayValue>some text</urn:displayValue>
            <urn:internalValue>some text</urn:internalValue>
            <urn:uuid/>
        </urn:value>
    </urn:extendedFieldList>
    <urn:extendedFieldList>
        <urn:name>MYUSER</urn:name>
```

```
        <urn:id>186</urn:id>
        <urn:uuid></urn:uuid>
        <urn:setValueBy>PRECEDENCE-VALUE</urn:setValueBy>
        <urn:setValueMethod></urn:setValueMethod>
        <urn:value>
            <urn:displayValue>carmen</urn:displayValue>
            <urn:internalValue>11</urn:internalValue>
            <urn:uuid></urn:uuid>
        </urn:value>
    </urn:extendedFieldList>
    <urn:extendedFieldList>
        <urn:name>CLOSEDATE</urn:name>
        <urn:id>190</urn:id>
        <urn:uuid></urn:uuid>
        <urn:setValueBy>PRECEDENCE-VALUE</urn:setValueBy>
        <urn:setValueMethod></urn:setValueMethod>
        <urn:value>
            <urn:displayValue/>
            <urn:internalValue/>
            <urn:uuid/>
        </urn:value>
    </urn:extendedFieldList>
    <urn:extendedFieldList>
        <urn:name>LASTSTATECHANGEDATE</urn:name>
        <urn:id>191</urn:id>
        <urn:uuid></urn:uuid>
        <urn:setValueBy>PRECEDENCE-VALUE</urn:setValueBy>
        <urn:setValueMethod></urn:setValueMethod>
        <urn:value>
            <urn:displayValue>2008-05-01T23:40:40+00:00
            →</urn:displayValue>
            <urn:internalValue>2008-05-01T23:40:40+00:00
            →</urn:internalValue>
            <urn:uuid/>
        </urn:value>
    </urn:extendedFieldList>
    <urn:extendedFieldList>
        <urn:name>SECONDARYOWNER</urn:name>
        <urn:id>193</urn:id>
        <urn:uuid></urn:uuid>
        <urn:setValueBy>PRECEDENCE-VALUE</urn:setValueBy>
        <urn:setValueMethod></urn:setValueMethod>
    </urn:extendedFieldList>
    <urn:extendedFieldList>
        <urn:name>LASTSTATECHANGER</urn:name>
        <urn:id>195</urn:id>
        <urn:uuid></urn:uuid>
        <urn:setValueBy>PRECEDENCE-VALUE</urn:setValueBy>
        <urn:setValueMethod></urn:setValueMethod>
        <urn:value>
            <urn:displayValue>admin</urn:displayValue>
            <urn:internalValue>1</urn:internalValue>
            <urn:uuid></urn:uuid>
        </urn:value>
    </urn:extendedFieldList>
```

```
            </urn:item>
         <urn:submitTransID>2</urn:submitTransID>
      </urn:CreatePrimaryItem>
```

URLAttachment

Description

The URLAttachment type holds information about a URL attached to an auxiliary or primary item. The URLAttachment type parameters are listed below.

Parameters

Name	Type	Description
id	integer	This is the internal TS_ID of the URL attachment from the TS_ATTACHMENTS table.
name	string	The name you give the URL attachment in Business Mashups.
url	string	The actual URL itself.
showAsImage	boolean	This flag indicates whether or not graphic images in a URL are shown in the Web interface.
modificationDateTime	dateTime	The date and time when the URL attachment was last modified. See Supported Date/Time Formats [page 184] for more information.
accessType	Attachment-Access-Type [page 103]	Shows the access type for the URL attachment. The value is either DEFAULT, RESTRICTED, or UNRESTRICTED.

Usage

You can add a URL to a primary or auxiliary item using URLAttachmentList. URLs can be links to a Web site external to Business Mashups or to a page within Business Mashups.

Note: If you set showAsImage to true, the image appears instead of a hyperlink when viewing the item in the Web interface. If you enter a URL to a graphic file such as `http://www.serena.com/image.gif`, the graphic appears. To show a URL graphically, an image file must be part of the URL. If you enter `www.serena.com` and select to show the URL as an image, an image does not appear because there is not an image file specified in the URL. Typically, this feature works for GIF and JPEG files, but can work in some browsers for PNG and BMP files as well. The file type that you can use for this feature depends on the file types your browser supports.

Tip: Internet Explorer is the only browser that supports URLs to files. Links to files do not work for users accessing the Web interface from other browser types.

XML

The following XML snippet shows the URLAttachment type in the URLAttachmentList parameter of TTItem.

```
<urn:urlAttachmentList>
    <urn:id>38</urn:id>
    <urn:name>test url</urn:name>
    <urn:url>http://www.serena.com/image.gif</urn:url>
    <urn:showAsImage>true</urn:showAsImage>
    <urn:modificationDateTime>2007-06-20T15:35:38-07:00
    →</urn:modificationDateTime>
    <urn:accessType>ATTACHACCESS-DEFAULT</urn:accessType>
</urn:urlAttachmentList>
```

Value or Values

Description

The Value or Values type holds one or multiple values. In the WSDL, it is converted to a choice element.

Parameters

Name	Type	Description
displayValue	string	Holds the display (Web interface) value.
internalValue	string	Holds the internal (database) value.
uuid	string	Holds the UUID value.

Usage

In the value parameter of the extendedFieldList, use either Value or Values, depending on whether or not the field allows multiple selections. Within the value parameter, provide either the display, internal, or uuid form of the value. Which value will ultimately be set is determined by the setValueBy parameter value. See Set-Value-By [page 119] for more information.

To set an empty value, you must use the SET-VALUE-BY parameter that corresponds to the empty <value> parameter. For example, to set an empty value for a field using the internalValue parameter, you could specify:

```
<urn:setValueBy>INTERNAL-VALUE</urn:setValueBy>
    <urn:value>
        <urn:internalValue></urn:internalValue>
```

Alternatively, to set an empty value for a field using the displayValue parameter, you could could specify:

```
<urn:setValueBy>DISPLAY-VALUE</urn:setValueBy>
    <urn:value>
        <urn:displayValue></urn:displayValue>
```

Note: To set an empty value for a single or multi-relational field, you must use INTERNAL-VALUE in SetValueBy and empty tags in the internalValue parameter.

XML

The following XML shows Value as seen in a typical call.

```
<urn:extendedFieldList>
    <urn:name>SINGLE_SELECTION</urn:name>
    <urn:id>182</urn:id>
    <urn:uuid>5dcf0aa8-22d6-4207-a88e-8e989df948df</urn:uuid>
    <urn:setValueBy>PRECEDENCE-VALUE</urn:setValueBy>
    <urn:value>
        <urn:displayValue>yellow</urn:displayValue>
        <urn:internalValue>122</urn:internalValue>
        <urn:uuid>4ad0961d-30dc-4198-8167-5224dcb6c065</urn:uuid>
    </urn:value>
</urn:extendedFieldList>
```

The following XML shows multiple Values as seen in a typical call.

```
<urn:extendedFieldList>
    <urn:name>MULTI_USER</urn:name>
    <urn:id>178</urn:id>
    <urn:uuid>f62c6b63-2531-441a-9fff-9cd471bc61ca</urn:uuid>
    <urn:setValueBy>PRECEDENCE-VALUE</urn:setValueBy>
    <urn:values>
        <urn:displayValue>admin</urn:displayValue>
        <urn:internalValue>1</urn:internalValue>
        <urn:uuid>d2d60592-656e-4103-a20d-f12da9305fe4</urn:uuid>
    </urn:values>
    <urn:values>
        <urn:displayValue>bill</urn:displayValue>
        <urn:internalValue>10</urn:internalValue>
        <urn:uuid>7130c9c3-abb6-41f5-bd7a-30c40f47b824</urn:uuid>
    </urn:values>
    <urn:values>
        <urn:displayValue>carmen</urn:displayValue>
        <urn:internalValue>11</urn:internalValue>
```

```
            <urn:uuid>9d71b19e-9b72-4731-bec3-3eba938da0de</urn:uuid>
        </urn:values>
    </urn:extendedFieldList>
```

Responses

This section provides detailed descriptions of Business Mashups Web service responses. The responses are the data elements that are returned from Business Mashups Web service operations. The responses listed here contain one or more parameters, which make up the data being sent back from the Web service. The parameters listed in each argument are either simple or complex types themselves. If the type is complex, a link to further detail of that type will be provided in the **Type** column.

The following table lists all supported responses in alphabetical order, followed by a brief description of each type. Select a response to view detailed information including:

- **Description**

  A brief description of the response.

- **Parameters**

  A table describing the types for each response. Both simple and complex types are listed for each response. For each complex type, you can click the type name for a detailed description.

- **Usage**

  Any notes, addtional details, and concerns regarding the response are addressed here.

- **XML**

  An example of the actual XML being sent is displayed here. The XML not only shows the response and its respective elements, you can also see detailed examples of each element and how the expected data is formatted.

Responses

Response	Description
ApplicationData [page 137]	Holds the name and other information about an application.
ExtraValue [page 138]	Holds the field type and field value of an item.
DatePreference [page 139]	Indicates a user's preferred date format.
Field [page 140]	Holds the name and other information about a field.
Field-Type [page 141]	Indicates the type of field.
FileAttachment [page 143]	Holds the details of a file that you upload to the server.
FileAttachmentContents [page 144]	Holds the actual contents of a file attachment.

Response	Description
FileContents [page 144]	Holds the actual contents of a file.
ImportCurrentOverallStatus [page 145]	Indicates if an import is in progress or completed.
ImportCurrentStatus [page 146]	Holds the name and other information about an import performed in Business Mashups.
GetReportsResult [page 146]	Holds the number of reports returned, as well as high-level information for each report.
OrderBy [page 148]	Holds the ORDER BY definition for a report.
Privilege [page 149]	Holds the name and other information about a privilege.
ProjectData [page 150]	Holds the name and other information about a project.
ReportAccessLevel [page 151]	Indicates the access level that is assigned to a report.
ReportCategory [page 153]	Indicates the report category. Also known as Privilege Category in the Web interface.
ReportDefinition [page 153]	Holds the columns and any ORDER BY definition for a report.
ReportInfo [page 155]	Holds a high-level description for a report.
ReportResult [page 157]	Holds the actual item data returned in a report.
RunReportResult [page 158]	Holds the results or output of a report.
SolutionData [page 162]	Holds the name and other information about a solution.
Solution-Type [page 164]	Indicates the type of solution.
SolutionWithUniqueName [page 165]	Holds the unique name and other information about a solution.
TableData [page 166]	Holds the name and other information about a table.
Table-Type [page 168]	Indicates the type of table.
TimePreference [page 169]	Indicates a user's preferred time format.

Response	Description
Transition [page 170]	Holds the name and other information about a transition.
Transition-Type [page 172]	Indicates the type of transition.
TTItem [page 174]	Holds all of the Business Mashups field information for an item.
User [page 179]	Holds the name and other information about a user.
UserWithPreferences [page 180]	Holds the name and other information about a user, including additional preferences.

ApplicationData

Description

The ApplicationData type holds the name and other information about an application in Business Mashups. The ApplicationData type parameters are listed below.

Parameters

Name	Type	Description
applicationID	integer	This is the internal TS_ID of the application from the TS_APPLICATIONS table.
applicationUUID	string	This is the alternate unique identifier for the application.
name	string	The name of the application. Derived from TS_NAME column in TS_APPLICATIONS.
description	string	The description of the application. Derived from the TS_DESCRIPTION column in TS_APPLICATIONS.
appDefUUID	string	The unique identifier that applies to the mashup definition in the Mashup Manager repository.

Usage

The ApplicationData type holds the ID, UUID, name and description used to describe an application. You can use GetApplications [page 54] to retrieve the application data shown here.

XML

The following XML snippet shows the ApplicationData type in the return element of the GetApplications response.

```
<ae:GetApplicationsResponse>
 <ae:return>
```

```
  <ae:applicationID>1</ae:applicationID>
  <ae:applicationUUID>4369c5bf-b90e-4f67-83ec-91d4d8ee71c7
  →</ae:applicationUUID>
  <ae:name>Issue Management / Incident Management</ae:name>
  <ae:description>TT Issue Management / TT Incident Management
  →</ae:description>
  <ae:appDefUUID></ae:appDefUUID>
 </ae:return>
 <ae:return>
  <ae:applicationID>2</ae:applicationID>
  <ae:applicationUUID>32eb350c-abd7-4d32-bca8-f92e7092684b
  →</ae:applicationUUID>
  <ae:name>HR Requests / IT Help Desk</ae:name>
  <ae:description></ae:description>
  <ae:appDefUUID>4028ef021460ffb70114ae846b3c003b</ae:appDefUUID>
 </ae:return>
 <ae:return>
  <ae:applicationID>3</ae:applicationID>
  <ae:applicationUUID>global-99febe4e-e8d1-4c9d-983d-437726611e2b
  →</ae:applicationUUID>
  <ae:name>Global Mashup</ae:name>
  <ae:description></ae:description>
  <ae:appDefUUID></ae:appDefUUID>
 </ae:return>
</ae:GetApplicationsResponse>
```

ExtraValue

Description

The ExtraValue type contains the field type and field values for an item returned in a report. The ExtraValue type parameters are listed below.

Parameters

Name	Type	Description
valueType	string	Describes the type of field returned.
value	string	The actual value of the field.

Usage

ExtraValue and itemURL are elements sent back in the ReportResult response to the RunReport call. Note that the field name is not returned, only the type of field and the field value. For more information, see ReportResult [page 157].

XML

The following XML snippet shows ReportResult in the return element of the RunReport response.

```
<ae:fieldValue>
   <ae:valueType>Text</ae:valueType>
   <ae:value>00038</ae:value>
```

```
</ae:fieldValue>
<ae:fieldValue>
   <ae:valueType>Text</ae:valueType>
   <ae:value>Texture image conversion</ae:value>
</ae:fieldValue>
<ae:fieldValue>
   <ae:valueType>State</ae:valueType>
   <ae:value>Assigned to Engineer</ae:value>
</ae:fieldValue>
<ae:fieldValue>
   <ae:valueType>Date/Time</ae:valueType>
   <ae:value>5 0:00:00</ae:value>
</ae:fieldValue>
<ae:fieldValue>
   <ae:valueType>Numeric</ae:valueType>
   <ae:value>2</ae:value>
</ae:fieldValue>
<ae:fieldValue>
   <ae:valueType>Date/Time</ae:valueType>
   <ae:value>07/18/2007</ae:value>
</ae:fieldValue>
```

DatePreference

Description

DatePreference indicates a user's preferred date format. The available options are listed below.

Parameters

Name	Type	Description
DATE-FORMAT-FROM-LOCALE	string	Use the format based on the user's locale.
DATE-FORMAT-MM-DD-YYYY	string	Use a MM-DD-YYYY format for dates.
DATE-FORMAT-DD-MM-YYYY	string	Use a DD-MM-YYYY format for dates.
DATE-FORMAT-DD-MM-YYYY.S	string	Use a DD.MM.YYYY format for dates.
DATE-FORMAT-YYYY-MM-DD	string	Use a YYYY-MM-DD format for dates.

Usage

DatePreference is used to determine how dates display to a user in the Web interface. The various date formats are returned in the datePreference parameter of the GetUserWithPreferences call. See the UserWithPreferences [page 180] response for more information.

XML

The following XML shows DatePreference as seen in the return element of a GetUserWithPreferences response.

```
<ae:GetUserWithPreferencesResponse>
 <ae:return>
  <ae:userId>chris</ae:userId>
  <ae:userName>Chris Tester</ae:userName>
  <ae:id>13</ae:id>
  <ae:uuid>35a48696-4307-4fa9-a429-de35d262d820</ae:uuid>
  <ae:email>ctester@serenateamtrackdb.com</ae:email>
  <ae:emailCC></ae:emailCC>
  <ae:timezone>America/Denver</ae:timezone>
  <ae:offsetFromGMT>-25200000</ae:offsetFromGMT>
  <ae:dstSavings>3600000</ae:dstSavings>
  <ae:datePreference>DATE-FORMAT-MM-DD-YYYY</ae:datePreference>
  <ae:timePreference>TIME-FORMAT-12HOUR</ae:timePreference>
  <ae:namespaceName>00000</ae:namespaceName>
 </ae:return>
</ae:GetUserWithPreferencesResponse>
```

Field

Description

The Field type holds the name and other information about a field in Business Mashups. The field type parameters are listed below.

Parameters

Name	Type	Description
fieldID	integer	This is the internal TS_ID of the field from the TS_FIELDS table.
fieldUUID	string	This is the alternate unique identifier for the field.
name	string	The database name of the field. Derived from the TS_DBNAME column in TS_FIELDS.
displayName	string	The logical name of the field. Derived from the TS_NAME column in TS_FIELDS.
fieldType	Field-Type [page 141]	Describes the type of field.

Usage

The Field type completely describes an available field in Business Mashups. Use GetTables to retrieve a list of fields available for a specified table. The list of fields appears in the fieldList element.

XML

The following XML snippet shows the Field type in the fieldList element of the GetTables response.

```
<ae:GetTablesResponse>
 <ae:return>
  <ae:tableID>37</ae:tableID>
  <ae:tableUUID>TS_COMPANIES</ae:tableUUID>
  <ae:solutionID>0</ae:solutionID>
  <ae:type>SYSTEM-AUXILIARY-TABLE</ae:type>
  <ae:name>TS_COMPANIES</ae:name>
  <ae:displayName>Companies</ae:displayName>
  <ae:description>Aux table of Company Info</ae:description>
  <ae:fieldList>
   <ae:fieldID>13</ae:fieldID>
   <ae:fieldUUID>SYSCODE-37-4</ae:fieldUUID>
   <ae:name>NAME</ae:name>
   <ae:displayName>Company Name</ae:displayName>
   <ae:fieldType>FLDTYPE-TEXT</ae:fieldType>
  </ae:fieldList>
  <ae:fieldList>
   <ae:fieldID>14</ae:fieldID>
   <ae:fieldUUID>SYSCODE-37-300</ae:fieldUUID>
   <ae:name>COMPANYID</ae:name>
   <ae:displayName>Customer ID</ae:displayName>
   <ae:fieldType>FLDTYPE-TEXT</ae:fieldType>
  </ae:fieldList>
  <ae:fieldList>
   <ae:fieldID>15</ae:fieldID>
   <ae:fieldUUID>SYSCODE-37-301</ae:fieldUUID>
   <ae:name>PRIMARYCONTACTID</ae:name>
   <ae:displayName>Primary Contact</ae:displayName>
   <ae:fieldType>FLDTYPE-CONTACT</ae:fieldType>
  </ae:fieldList>
```

Field-Type

Description

Field-Type indicates the type of field. The available field types are listed below.

Parameters

Name	Type	Description
FLDTYPE_UNKNOWN	string	Indicates an unknown field.
FLDTYPE_NUMERIC	string	Indicates a numeric field.
FLDTYPE_TEXT	string	Indicates a text field.
FLDTYPE_DATETIME	string	Indicates a date/time field.
FLDTYPE_SELECTION	string	Indicates a single select field.
FLDTYPE_BINARY	string	Indicates a binary field.

Name	Type	Description
FLDTYPE_STATE	string	Indicates a state field.
FLDTYPE_USER	string	Indicates a user field.
FLDTYPE_PROJECT	string	Indicates a project field.
FLDTYPE_SUMMATION	string	Indicates a summation field.
FLDTYPE_MULTIPLE_SELECTION	string	Indicates a multi-select field.
FLDTYPE_CONTACT	string	Indicates a contact field.
FLDTYPE_INCIDENT	string	Indicates a field unique to Incidents.
FLDTYPE_FOLDER	string	Indicates a folder type field.
FLDTYPE_RELATIONAL	string	Indicates a single relational field.
FLDTYPE_SUBRELATIONAL	string	Indicates a sub-relational field.
FLDTYPE_SYSTEM	string	Indicates a system field.
FLDTYPE_MULTIPLE_RELATIONAL	string	Indicates a multi-relational field.
FLDTYPE_MULTIPLE_GROUP	string	Indicates a multi-group field.
FLDTYPE_MULTIPLE_USERGROUP	string	Indicates a multi-user field.

Usage

Field-Type can be used to identify the type of field returned in the GetTables response.

XML

The following XML shows Field-Type as seen in the fieldType element of a GetTables response.

```
<ae:GetTablesResponse>
 <ae:return>
  <ae:tableID>1003</ae:tableID>
  <ae:tableUUID>b82d2e9f-fea7-4669-84b6-7d5f7d5f50e9</ae:tableUUID>
  <ae:solutionID>7</ae:solutionID>
  <ae:type>PRIMARY-TABLE</ae:type>
  <ae:name>USR_SPRINTS</ae:name>
  <ae:displayName>Sprints</ae:displayName>
  <ae:description/>
  <ae:fieldList>
   <ae:fieldID>121</ae:fieldID>
   <ae:fieldUUID>d18e91a7-ea80-4e55-9838-909a7a1b642e</ae:fieldUUID>
   <ae:name>ISSUEID</ae:name>
```

```
    <ae:displayName>Sprint Id</ae:displayName>
    <ae:fieldType>FLDTYPE-TEXT</ae:fieldType>
   </ae:fieldList>
  </ae:return>
 </ae:GetTablesResponse>
```

FileAttachment

Description

The FileAttachment type holds the details of a file that you upload to the server. The FileAttachment type parameters are listed below.

Parameters

Name	Type	Description
id	integer	This is the internal TS_ID of the attachment from the TS_ATTACHMENTS table.
name	string	The name you give the attachment in Business Mashups.
fileName	string	The name of the file as it exists on the file system.
showAsImage	boolean	This flag indicates whether or not graphic attachments are shown as images in the Web interface.
modificationDateTime	dateTime	The date and time when the attachment was last modified. See Supported Date/Time Formats [page 184] for more information.
url	string	The URL for the attachment. The file can be downloaded from this URL.
accessType	Attachment-Access-Type [page 103]	Shows the access type for the attachment. The value is either DEFAULT, RESTRICTED, or UNRESTRICTED.

Usage

FileAttachmentContents is inherited from FileAttachment. FileAttachment is used with FileAttachmentContents to completely describe a file attachment. The file attachment detail consists of an ID, name, and URL. With the URL, client code can download the file directly.

XML

The following XML snippet shows the FileAttachment type in the return element of the CreateFileAttachment response.

```
<ae:CreateFileAttachmentResponse>
 <ae:return>
  <ae:id>54</ae:id>
  <ae:name>jpg_jpg</ae:name>
  <ae:fileName>Red_moon_desert.jpg</ae:fileName>
  <ae:showAsImage>false</ae:showAsImage>
  <ae:modificationDateTime>2008-03-12T21:45:54-07:00
  →</ae:modificationDateTime>
  <ae:url>http://localhost:80/tmtrack/tmtrack.dll?
  →AttachmentPage&amp;amp;AttachmentID=5</ae:url>
  <ae:accessType>ATTACHACCESS-RESTRICTED</ae:accessType>
 </ae:return>
</ae:CreateFileAttachmentResponse>
```

FileAttachmentContents

Description

The FileAttachmentContents type holds the actual contents of a file that you upload to the server. The FileAttachmentContents type parameters are listed below.

Parameters

Name	Type	Description
checksum	long	This is a checksum that can be verified to ensure contents of the file are correct.
encodedContents	base64binary	Holds the base64 encoded contents of the file.

Usage

Inherited from FileAttachment. FileAttachmentContents is used with FileAttachment to completely describe a file attachment. For more information, see FileAttachment [page 143].

XML

The following XML is a snippet of the FileAttachmentContents type.

```
<ae:checksum></ae:checksum>
 <ae:encodedContents>encoded_content_here</ae:encodedContents>
```

FileContents

Description

The FileContents type holds the actual contents of a file returned from an Export operation or supplied to an Import operation. The FileContents type parameters are listed below.

Parameters

Name	Type	Description
checksum	long	This is a checksum that can be verified to ensure contents of the file are correct.
encodedContents	base64binary	Holds the base64 encoded contents of the file.

Usage

FileContents holds the actual contents of a file and is used to pass a file to or receive a file from the client. For more information on exporting and importing, see Export [page 52] or Import [page 82].

XML

The following XML snippet shows the FileContents type in the return element of the Export response.

```
<ae:ExportResponse>
 <ae:return>
  <ae:checksum>59280</ae:checksum>
  <ae:encodedContents>encoded_content_here</ae:encodedContents>
 </ae:return>
</ae:ExportResponse>
```

ImportCurrentOverallStatus

Description

The ImportCurrentOverallStatus type indicates if an import is in progress or completed. The available ImportCurrentOverallStatus types are listed below.

Parameters

Name	Type	Description
IMPORT-IN-PROGRESS	string	The import is still in progress.
IMPORT-COMPLETED	string	The import is complete.

Usage

ImportCurrentOverallStatus can be used to determine the import status.

XML

The following XML shows ImportCurrentOverallStatus as seen in the return element of a typical ImportStatus response.

```
<ae:ImportStatusResponse>
 <ae:return>
  <ae:importStatus>IMPORT-COMPLETED</ae:importStatus>
  <ae:percentageComplete>100</ae:percentageComplete>
```

```
  <ae:currentStep/>
 </ae:return>
</ae:ImportStatusResponse>
```

ImportCurrentStatus

Description

The ImportCurrentStatus type holds the name and other information about an import performed in Business Mashups. The ImportCurrentStatus type parameters are listed below.

Parameters

Name	Type	Description
importStatus	ImportCurrentOverallStatus [page 145]	Status of import which indicates if the import is still running.
percentageComplete	long	Percentage of the import that is complete.
currentStep	string	A message indicating the current step of the import if it is still in progress.

Usage

The ImportCurrentStatus type provides information about whether or not the import has completed, as well as the percentage complete, and the currently step of the import in process. Use ImportStatus [page 83] to return the status of a given import.

XML

The following XML snippet shows the ImportCurrentStatus type in the return element of the ImportStatus response.

```
<ae:ImportStatusResponse>
 <ae:return>
  <ae:importStatus>IMPORT-COMPLETED</ae:importStatus>
  <ae:percentageComplete>100</ae:percentageComplete>
  <ae:currentStep/>
 </ae:return>
</ae:ImportStatusResponse>
```

GetReportsResult

Description

The GetReportsResult type holds the number of reports returned, as well as high-level information for each report. The GetReportsResult type parameters are listed below.

Parameters

Name	Type	Description
queryRange	QueryRange [page 113]	The *QueryRange* type specifies the number of reports returned.
report	ReportInfo [page 155]	Holds a description of the report.

Usage

The GetReportsResult type is sent back in response to a GetReports call. GetReportsResult summarizes the range of reports returned and provides an overview of each report. The actual contents of a report are obtained in the RunReportResult response. You can use GetReports [page 69] to retrieve the report data shown here.

XML

The following XML snippet shows the GetReportsResult type in the return element of the GetReports response.

```
<ae:GetReportsResponse>
   <ae:return>
      <ae:queryRange>
         <ae:startIndex>2</ae:startIndex>
         <ae:fetchSize>3</ae:fetchSize>
         <ae:totalCount>8</ae:totalCount>
      </ae:queryRange>
      <ae:report>
         <ae:reportID>3</ae:reportID>
         <ae:reportUUID>0d853d5d-6795-4e35-9d97-76d195fb3d32
         →</ae:reportUUID>
         <ae:reportName>All Active Issues I Submitted</ae:reportName>
         <ae:reportURL>tmtrack.dll?ReportPage&amp;Template=reports%
         →2Flistframe&amp;ReportId=3</ae:reportURL>
         <ae:solutionID>1</ae:solutionID>
         <ae:solutionName>BASE_ISSUE_DEFECT_TRACKING</ae:solutionName>
         <ae:tableID>1000</ae:tableID>
         <ae:projectID>4</ae:projectID>
         <ae:createdBy>Joe Manager</ae:createdBy>
         <ae:createDate>2007-06-20T21:33:16+03:00</ae:createDate>
         <ae:modifiedBy/>
         <ae:modifiedDate>1969-12-31T23:59:59Z</ae:modifiedDate>
         <ae:execDate>2007-07-21T02:36:39+03:00</ae:execDate>
      </ae:report>
      <ae:report>
         <ae:reportID>9</ae:reportID>
         <ae:reportUUID>894cbfa0-15d3-47db-99ce-cc4247b25bd2
         →</ae:reportUUID>
         <ae:reportName>All Active Software Issues</ae:reportName>
         <ae:reportURL>tmtrack.dll?ReportPage&amp;Template=reports%
         →2Flistframe&amp;ReportId=9</ae:reportURL>
         <ae:solutionID>1</ae:solutionID>
         <ae:solutionName>BASE_ISSUE_DEFECT_TRACKING</ae:solutionName>
```

```
            <ae:tableID>1000</ae:tableID>
            <ae:projectID>4</ae:projectID>
            <ae:createdBy>Joe Manager</ae:createdBy>
            <ae:createDate>2007-06-21T01:43:47+03:00</ae:createDate>
            <ae:modifiedBy/>
            <ae:modifiedDate>1969-12-31T23:59:59Z</ae:modifiedDate>
            <ae:execDate>2007-10-25T22:30:55+03:00</ae:execDate>
         </ae:report>
         <ae:report>
            <ae:reportID>10</ae:reportID>
            <ae:reportUUID>bc23638c-6643-41cc-bc38-e9b435017a33
            →</ae:reportUUID>
            <ae:reportName>All Software Issues I Own</ae:reportName>
            <ae:reportURL>tmtrack.dll?ReportPage&amp;Template=reports%
            →2Flistframe&amp;ReportId=10</ae:reportURL>
            <ae:solutionID>1</ae:solutionID>
            <ae:solutionName>BASE_ISSUE_DEFECT_TRACKING</ae:solutionName>
            <ae:tableID>1000</ae:tableID>
            <ae:projectID>4</ae:projectID>
            <ae:createdBy>Joe Manager</ae:createdBy>
            <ae:createDate>2007-06-21T01:46:05+03:00</ae:createDate>
            <ae:modifiedBy>Joe Manager</ae:modifiedBy>
            <ae:modifiedDate>2007-07-09T06:50:39+03:00</ae:modifiedDate>
            <ae:execDate>2007-10-25T23:30:09+03:00</ae:execDate>
         </ae:report>
      </ae:return>
   </ae:GetReportsResponse>
```

OrderBy

Description

The OrderBy type contains the ORDER BY definition for a report, if ORDER BY is specified for the report. The ReportDefinition type parameters are listed below.

Parameters

Name	Type	Description
firstFieldName	string	The first field used to order report results.
secondFieldName	string	The second field used to order report results.

Usage

The OrderBy type describes which field or fields are used to order the results returned in a report. OrderBy may have an empty return value even though certain fields are desginated in the report definition to sort the results.

XML

The following XML snippet shows the OrderBy type in the reportDefinition element of the RunReport response.

```
<ae:reportDefinition>
   <ae:orderBy/>
```

Privilege

Description

The Privilege type holds the name and other information about a privilege in Business Mashups. The Privilege type parameters are listed below.

Parameters

Name	Type	Description
name	string	The name of the privilege.
objectUUID	string	The UUID of the object to which the privilege applies. May be set to null if not scoped to an object. May return the ID and not UUID in some cases.
type	string	The type of privilege. See below.

Usage

The Privilege type completely describes an available privilege in Business Mashups. Use GetUserPrivileges [page 77] to retrieve a list of privileges available for a specified user.

The available privilege types for a normal user are:

TS_PRIVTYPE_USERSYS
TS_PRIVTYPE_USERPRJ
TS_PRIVTYPE_USERWKF
TS_PRIVTYPE_USERFLD
TS_PRIVTYPE_USERTBL

The available privilege types for an administrator are:

TS_PRIVTYPE_ADMSYS
TS_PRIVTYPE_ADMPRJ
TS_PRIVTYPE_ADMWKF
TS_PRIVTYPE_ADMFLD_PRJ
TS_PRIVTYPE_ADMFLD_WKF

TS_PRIVTYPE_ADMCON
TS_PRIVTYPE_UNKNOWN
TS_PRIVTYPE_SYSMASK
TS_PRIVTYPE_ADMMASK
TS_PRIVTYPE_TBLMASK

XML

The following XML snippet shows the Privilege type in the return element of the GetUserPrivileges response.

```
<ae:GetUserPrivilegesResponse>
 <ae:return>
  <ae:name>TS_ADMPRJPRIV_ADDPROJECT</ae:name>
  <ae:objectUUID>ROOTPROJECT</ae:objectUUID>
  <ae:type>TS_PRIVTYPE_ADMPRJ</ae:type>
 </ae:return>
 <ae:return>
  <ae:name>TS_ADMPRJPRIV_EDITPROJECT</ae:name>
  <ae:objectUUID>ROOTPROJECT</ae:objectUUID>
  <ae:type>TS_PRIVTYPE_ADMPRJ</ae:type>
 </ae:return>
 <ae:return>
  <ae:name>TS_ADMPRJPRIV_DELETEPROJECT</ae:name>
  <ae:objectUUID>ROOTPROJECT</ae:objectUUID>
  <ae:type>TS_PRIVTYPE_ADMPRJ</ae:type>
 </ae:return>
 <ae:return>
  <ae:name>TS_ADMPRJPRIV_ASSIGNPROJECTPRIVS</ae:name>
  <ae:objectUUID>ROOTPROJECT</ae:objectUUID>
  <ae:type>TS_PRIVTYPE_ADMPRJ</ae:type>
 </ae:return>
 <ae:return>
  <ae:name>TS_ADMPRJPRIV_ADDPROJECT</ae:name>
  <ae:objectUUID>1bb8e27a-3156-49e6-8257-f7379e6aa498</ae:objectUUID>
  <ae:type>TS_PRIVTYPE_ADMPRJ</ae:type>
 </ae:return>
</ae:GetUserPrivilegesResponse>
```

ProjectData

Description

The ProjectData type holds the name and other information about a project in Business Mashups. The ProjectData type parameters are listed below.

Parameters

Name	Type	Description
projectID	integer	This is the internal TS_ID of the projects from the TS_PROJECTS table.
projectUUID	string	This is the alternate unique identifier for the project.
name	string	The name of the project. Derived from TS_NAME column in TS_PROJECTS.
fullyQualifiedName	string	The fully qualified name of the project.
description	string	The description of the project. Derived from the TS_DESCRIPTION column in TS_PROJECTS.

Usage

The ProjectData type holds the ID, UUID, name and description used to describe a project. You can use GetSubmitProjects [page 72] to retrieve the project data shown here.

For more information on the fullyQualifiedName, see the fullyQualifiedProjectName argument in the **Usage** section of CreatePrimaryItemWithName [page 42].

XML

The following XML snippet shows the ProjectData type in the return element of the GetSubmitProjects response.

```
<ae:GetSubmitProjectsResponse>
 <ae:return>
  <ae:projectID>11</ae:projectID>
  <ae:projectUUID>a4fa9b92-136d-4cc0-b9dd-53a84229defd</ae:projectUUID>
  <ae:name>Technical Support</ae:name>
  <ae:fullyQualifiedName>Customer Support||Technical Support
  →</ae:fullyQualifiedName>
  <ae:description></ae:description>
 </ae:return>
 <ae:return>
  <ae:projectID>35</ae:projectID>
  <ae:projectUUID>2f1c91ea-956b-4402-88e2-d1e102c2bc88</ae:projectUUID>
  <ae:name>Product Support</ae:name>
  <ae:fullyQualifiedName>Customer Support||Product Support
  →</ae:fullyQualifiedName>
  <ae:description></ae:description>
 </ae:return>
</ae:GetSubmitProjectsResponse>
```

ReportAccessLevel

Description

ReportAccessLevel is used to describe the access level assigned to a report. The available report access levels are listed below.

Parameters

Name	Type	Description
PRIVATE	string	This access level enables individual users to manage reports they create. Only the user who creates a private report can access, modify, or delete private reports and only if this user is granted "Manage Private Reports" privileges.
GUEST	string	This access level denotes a guest-level report. Users with guest-level report privileges can perform report actions for guest-level reports.
USER	string	This access level denotes a user-level report. Users with user-level report privileges can perform report actions for user-level reports.
MANAGER	string	This access level denotes a manager-level report. Users with manager-level report privileges can perform report actions for manager-level reports.

Usage

ReportAccessLevel is always returned in the ReportInfo parameter of the GetReportsResult response. See ReportInfo [page 155] for further detail.

XML

The following XML snippet shows ReportAccessLevel as seen in a typical response.

```
<ae:reportInfo>
 <ae:reportID>88</ae:reportID>
 <ae:reportUUID>9cc7c684-7c62-402b-b249-a0a620fed283</ae:reportUUID>
 <ae:reportName>Appreport1</ae:reportName>
 <ae:reportURL>tmtrack.dll?ReportPage&amp;Template=reports
 →%2Flistframe&amp;ReportId=88</ae:reportURL>
 <ae:reportType>LISTING</ae:reportType>
 <ae:reportCategory>APPLICATION</ae:reportCategory>
 <ae:reportAccessLevel>USER</ae:reportAccessLevel>
 <ae:solutionID>14</ae:solutionID>
 <ae:solutionName>REPORTSAPP</ae:solutionName>
 <ae:tableID>1011</ae:tableID>
 <ae:projectID>25</ae:projectID>
 <ae:projectUUID>9f7fec5c-aff1-4c6f-be29-8dbe06f02350</ae:projectUUID>
 <ae:createdBy>Newton Engineer</ae:createdBy>
 <ae:createDate>2009-02-18T12:50:38+02:00</ae:createDate>
 <ae:modifiedBy>Joe Manager</ae:modifiedBy>
 <ae:modifiedDate>1969-12-31T23:59:59Z</ae:modifiedDate>
 <ae:execDate>2009-02-18T13:20:45+02:00</ae:execDate>
 <ae:isQueryAtRuntime>false</ae:isQueryAtRuntime>
</ae:reportInfo>
```

ReportCategory

Description

ReportCategory is used to describe a grouping of reports. The available report categories are listed below.

Parameters

Name	Type	Description
APPLICATION	string	Indicates the report is an Application Report: a mashup-specific listing report designed in Mashup Composer.
BUILTIN	string	Indicates the report is a built-in report.
USERREPORTS	string	Indicates the report is a user-created report.

Usage

ReportCategory is used to describe reports returned in the ReportInfo parameter of the GetReportsResult response. See ReportInfo [page 155] for further detail.

XML

The following XML snippet shows ReportCategory as seen in a typical response.

```
<ae:reportInfo>
 <ae:reportID>88</ae:reportID>
 <ae:reportUUID>9cc7c684-7c62-402b-b249-a0a620fed283</ae:reportUUID>
 <ae:reportName>Appreport1</ae:reportName>
 <ae:reportURL>tmtrack.dll?ReportPage&amp;Template=reports
 →%2Flistframe&amp;ReportId=88</ae:reportURL>
 <ae:reportType>LISTING</ae:reportType>
 <ae:reportCategory>APPLICATION</ae:reportCategory>
 <ae:reportAccessLevel>USER</ae:reportAccessLevel>
 <ae:solutionID>14</ae:solutionID>
 <ae:solutionName>REPORTSAPP</ae:solutionName>
 <ae:tableID>1011</ae:tableID>
 <ae:projectID>25</ae:projectID>
 <ae:projectUUID>9f7fec5c-aff1-4c6f-be29-8dbe06f02350</ae:projectUUID>
 <ae:createdBy>Newton Engineer</ae:createdBy>
 <ae:createDate>2009-02-18T12:50:38+02:00</ae:createDate>
 <ae:modifiedBy>Joe Manager</ae:modifiedBy>
 <ae:modifiedDate>1969-12-31T23:59:59Z</ae:modifiedDate>
 <ae:execDate>2009-02-18T13:20:45+02:00</ae:execDate>
 <ae:isQueryAtRuntime>false</ae:isQueryAtRuntime>
</ae:reportInfo>
```

ReportDefinition

Description

The ReportDefinition type contains columns and the order by definition for a report. The ReportDefinition type parameters are listed below.

Parameters

Name	Type	Description
orderBy	OrderBy [page 148]	Indicates which fields are used to organize the report results.
columns	Field [page 140]	Holds a description the fields used in the report.

Usage

The ReportDefinition describes the basic structure of a report. The ReportDefinition type contains a description of each of the fields used in the report. If the report uses any of the columns to order the results, the OrderBy parameter will indicate which column(s) are used.

XML

The following XML snippet shows the ReportDefintion type in the return element of the RunReport response.

```
<ae:reportDefinition>
   <ae:orderBy/>
   <ae:columns>
      <ae:fieldID>91</ae:fieldID>
      <ae:fieldUUID>47a65f2b-8203-4c63-a757-dfd35ba6fb5c</ae:fieldUUID>
      <ae:name>Backlog Id</ae:name>
      <ae:displayName>Backlog Id</ae:displayName>
      <ae:fieldType>FLDTYPE-TEXT</ae:fieldType>
   </ae:columns>
   <ae:columns>
      <ae:fieldID>92</ae:fieldID>
      <ae:fieldUUID>dbe69e7f-024d-4f99-a71d-e2f8a6f8bfe2</ae:fieldUUID>
      <ae:name>Title</ae:name>
      <ae:displayName>Title</ae:displayName>
      <ae:fieldType>FLDTYPE-TEXT</ae:fieldType>
   </ae:columns>
   <ae:columns>
      <ae:fieldID>107</ae:fieldID>
      <ae:fieldUUID>d582fd4b-a4da-477f-8560-3f467430e438</ae:fieldUUID>
      <ae:name>Associated Sprint</ae:name>
      <ae:displayName>Associated Sprint</ae:displayName>
      <ae:fieldType>FLDTYPE-RELATIONAL</ae:fieldType>
   </ae:columns>
   <ae:columns>
      <ae:fieldID>101</ae:fieldID>
      <ae:fieldUUID>51d3c6cc-28a3-4732-a6be-ed56a96c7575</ae:fieldUUID>
      <ae:name>Est. Time to Complete</ae:name>
      <ae:displayName>Est. Time to Complete</ae:displayName>
      <ae:fieldType>FLDTYPE-DATETIME</ae:fieldType>
   </ae:columns>
   <ae:columns>
      <ae:fieldID>102</ae:fieldID>
      <ae:fieldUUID>82f272fe-25c7-4984-b637-d5c757b93a74</ae:fieldUUID>
      <ae:name>Actual Time to Complete</ae:name>
```

```
        <ae:displayName>Actual Time to Complete</ae:displayName>
        <ae:fieldType>FLDTYPE-DATETIME</ae:fieldType>
    </ae:columns>
</ae:reportDefinition>
```

ReportInfo

Description

The ReportInfo type contains a high-level description for a report. The ReportInfo type parameters are listed below.

Parameters

Name	Type	Description
reportID	integer	This is the internal TS_ID of the report from the TS_REPORTS table.
reportUUID	string	Alternate unique identifier (TS_UUID from TS_REPORTS) for a report.
reportName	string	This is the name of the report (TS_NAME from TS_REPORTS).
reportURL	string	This is the URL for the report.
reportType	ReportType [page 118]	An enumeration that indicates the type of report.
reportCategory	ReportCategory [page 153]	A broader enumeration that indicates the category of report (built-in reports, application reports, reports you authored).
reportAccessLevel	ReportAccessLevel [page 151]	An enumeration that indicates the report's access level (PRIVATE, GUEST, USER, or MANAGER).
solutionID	integer	This is the internal TS_ID of the solution from TS_SOLUTIONS that the report is based on.
solutionName	string	This is the name of the solution that the report is based on (TS_NAME from TS_SOLUTIONS). If solutionID is also specified, then solutionID will take precedence.
tableID	integer	This is the internal TS_ID of the table from TS_TABLES that the report is based on.

Name	Type	Description
projectID	integer	This is the internal TS_ID of the project from TS_PROJECTS that the report was created against.
projectName	string	The fully qualified name of the project.
projectUUID	string	This is the alternate unique ID of the project that the report was created against.
createdBy	string	The name of the user who authored the report.
createDate	dateTime	The date the report was created.
modifiedBy	string	The name of the user that last modified the report.
modifiedDate	dateTime	The date the report was last modified.
execDate	dateTime	The date the report was last executed.
isQueryAtRuntime	boolean	This flag is used to indicate whether or not the report is a Query at Runtime report.

Usage

The ReportInfo type holds data used to describe a report. You can use GetReports [page 69] to retrieve the report data shown here.

XML

The following XML snippet shows the ReportInfo type in the return element of the GetReports response.

```
<ae:reportInfo>
 <ae:reportID>88</ae:reportID>
 <ae:reportUUID>9cc7c684-7c62-402b-b249-a0a620fed283</ae:reportUUID>
 <ae:reportName>Appreport1</ae:reportName>
 <ae:reportURL>tmtrack.dll?ReportPage&amp;Template=reports
 →%2Flistframe&amp;ReportId=88</ae:reportURL>
 <ae:reportType>LISTING</ae:reportType>
 <ae:reportCategory>APPLICATION</ae:reportCategory>
 <ae:reportAccessLevel>USER</ae:reportAccessLevel>
 <ae:solutionID>14</ae:solutionID>
 <ae:solutionName>REPORTSAPP</ae:solutionName>
 <ae:tableID>1011</ae:tableID>
 <ae:projectID>25</ae:projectID>
 <ae:projectUUID>9f7fec5c-aff1-4c6f-be29-8dbe06f02350</ae:projectUUID>
 <ae:createdBy>Newton Engineer</ae:createdBy>
 <ae:createDate>2009-02-18T12:50:38+02:00</ae:createDate>
 <ae:modifiedBy>Joe Manager</ae:modifiedBy>
```

```
 <ae:modifiedDate>1969-12-31T23:59:59Z</ae:modifiedDate>
 <ae:execDate>2009-02-18T13:20:45+02:00</ae:execDate>
 <ae:isQueryAtRuntime>false</ae:isQueryAtRuntime>
</ae:reportInfo>
```

ReportResult

Description

The ReportResult type contains the actual item data returned in a report. The ReportResult type parameters are listed below.

Parameters

Name	Type	Description
itemURL	string	This is the URL for the item returned in the report results.
fieldValue	ExtraValue [page 138]	Holds the field type and field value for each field returned in the report results.

Usage

The ReportResult is sent back in response to RunReport [page 86]. The ReportResult type contains the URL for each item returned by the report. You can prepend `http://serverName/tmtrack/` to the itemURL contents and access the item from the Web interface. For every column defined in the report, the field type and field value is returned as well in the ExtraValue type.

XML

The following XML snippet shows ReportResult in the return element of the RunReport response.

```
<ae:result>
   <ae:itemURL>tmtrack.dll?IssuePage&amp;RecordId=6&amp;
   →Template=view&amp;TableId=1000</ae:itemURL>
   <ae:fieldValue>
      <ae:valueType>Text</ae:valueType>
      <ae:value>00038</ae:value>
   </ae:fieldValue>
   <ae:fieldValue>
      <ae:valueType>Text</ae:valueType>
      <ae:value>Texture image conversion</ae:value>
   </ae:fieldValue>
   <ae:fieldValue>
      <ae:valueType>State</ae:valueType>
      <ae:value>Assigned to Engineer</ae:value>
   </ae:fieldValue>
   <ae:fieldValue>
      <ae:valueType>Date/Time</ae:valueType>
      <ae:value>5 0:00:00</ae:value>
   </ae:fieldValue>
```

```
      <ae:fieldValue>
         <ae:valueType>Numeric</ae:valueType>
         <ae:value>2</ae:value>
      </ae:fieldValue>
      <ae:fieldValue>
         <ae:valueType>Date/Time</ae:valueType>
         <ae:value>07/18/2007</ae:value>
      </ae:fieldValue>
   </ae:result>
   <ae:result>
      <ae:itemURL>tmtrack.dll?IssuePage&amp;RecordId=7&amp;
      →Template=view&amp;TableId=1000</ae:itemURL>
      <ae:fieldValue>
         <ae:valueType>Text</ae:valueType>
         <ae:value>00039</ae:value>
      </ae:fieldValue>
      <ae:fieldValue>
         <ae:valueType>Text</ae:valueType>
         <ae:value>Upgrade from v3.0 does not work correctly.</ae:value>
      </ae:fieldValue>
      <ae:fieldValue>
         <ae:valueType>State</ae:valueType>
         <ae:value>Assigned to Engineer</ae:value>
      </ae:fieldValue>
      <ae:fieldValue>
         <ae:valueType>Date/Time</ae:valueType>
         <ae:value>1:00:00</ae:value>
      </ae:fieldValue>
      <ae:fieldValue>
         <ae:valueType>Numeric</ae:valueType>
         <ae:value>3</ae:value>
      </ae:fieldValue>
      <ae:fieldValue>
         <ae:valueType>Date/Time</ae:valueType>
         <ae:value>07/08/2007</ae:value>
      </ae:fieldValue>
   </ae:result>
```

RunReportResult

Description

The RunReportResult type contains the results or output of a report. The RunReportResult type parameters are listed below.

Parameters

Name	Type	Description
queryRange	QueryRange [page 113]	The *QueryRange* type specifies the number of reports returned.
reportInfo	ReportInfo [page 155]	Holds a description of the report.

Name	Type	Description
reportDefinition	ReportDefinition [page 153]	Holds the columns in a report and indicates which columns are used to order the results.
result	ReportResult [page 157]	Holds the actual data in the columns.

Usage

The RunReportResult is sent back in response to RunReport [page 86]. The RunReportResult type contains the actual data you would see in response to executing a report in the Web interface.

XML

The following XML snippet shows the ReportInfo, ReportDefintion, and ReportResult in the return element of the RunReport response.

```
<ae:RunReportResponse>
   <ae:return>
      <ae:queryRange>
         <ae:startIndex>2</ae:startIndex>
         <ae:fetchSize>4</ae:fetchSize>
         <ae:totalCount>9</ae:totalCount>
      </ae:queryRange>
      <ae:reportInfo>
       <ae:reportID>88</ae:reportID>
       <ae:reportUUID>9cc7c684-7c62-402b-b249-a0a620fed283</ae:reportUUID>
       <ae:reportName>Appreport1</ae:reportName>
       <ae:reportURL>tmtrack.dll?ReportPage&amp;Template=reports
       →%2Flistframe&amp;ReportId=88</ae:reportURL>
       <ae:reportType>LISTING</ae:reportType>
       <ae:reportCategory>APPLICATION</ae:reportCategory>
       <ae:reportAccessLevel>USER</ae:reportAccessLevel>
       <ae:solutionID>14</ae:solutionID>
       <ae:solutionName>REPORTSAPP</ae:solutionName>
       <ae:tableID>1011</ae:tableID>
       <ae:projectID>25</ae:projectID>
       <ae:projectUUID>9f7fec5c-aff1-4c6f-be29-8dbe06f02350</ae:projectUUID>
       <ae:createdBy>Newton Engineer</ae:createdBy>
       <ae:createDate>2009-02-18T12:50:38+02:00</ae:createDate>
       <ae:modifiedBy>Joe Manager</ae:modifiedBy>
       <ae:modifiedDate>1969-12-31T23:59:59Z</ae:modifiedDate>
       <ae:execDate>2009-02-18T13:20:45+02:00</ae:execDate>
       <ae:isQueryAtRuntime>false</ae:isQueryAtRuntime>
      </ae:reportInfo>
      <ae:reportDefinition>
         <ae:orderBy/>
         <ae:columns>
            <ae:fieldID>67</ae:fieldID>
            <ae:fieldUUID>6654a10a-8f73-4e2f-9035-24322a81f61d
            →</ae:fieldUUID>
            <ae:name>Item Id</ae:name>
```

```
            <ae:displayName>Item Id</ae:displayName>
            <ae:fieldType>FLDTYPE-TEXT</ae:fieldType>
         </ae:columns>
         <ae:columns>
            <ae:fieldID>34</ae:fieldID>
            <ae:fieldUUID>54abccf1-f929-4539-bb0c-2150d6f01f15
            →</ae:fieldUUID>
            <ae:name>Title</ae:name>
            <ae:displayName>Title</ae:displayName>
            <ae:fieldType>FLDTYPE-TEXT</ae:fieldType>
         </ae:columns>
         <ae:columns>
            <ae:fieldID>59</ae:fieldID>
            <ae:fieldUUID>a78600c8-e07a-4a58-8b3b-91c0dc8cd432
            →</ae:fieldUUID>
            <ae:name>State</ae:name>
            <ae:displayName>State</ae:displayName>
            <ae:fieldType>FLDTYPE-STATE</ae:fieldType>
         </ae:columns>
         <ae:columns>
            <ae:fieldID>43</ae:fieldID>
            <ae:fieldUUID>5722fe4f-2352-430d-b636-2a8d27f15e54
            →</ae:fieldUUID>
            <ae:name>Est Time to Fix</ae:name>
            <ae:displayName>Est Time to Fix</ae:displayName>
            <ae:fieldType>FLDTYPE-DATETIME</ae:fieldType>
         </ae:columns>
         <ae:columns>
            <ae:fieldID>46</ae:fieldID>
            <ae:fieldUUID>9014b6ac-8a07-49ef-8e3e-48b6919fef54
            →</ae:fieldUUID>
            <ae:name>Priority</ae:name>
            <ae:displayName>Priority</ae:displayName>
            <ae:fieldType>FLDTYPE-NUMERIC</ae:fieldType>
         </ae:columns>
         <ae:columns>
            <ae:fieldID>68</ae:fieldID>
            <ae:fieldUUID>56cf7f79-9eee-416a-ab9d-da6cb4990324
            →</ae:fieldUUID>
            <ae:name>Est Date to Fix</ae:name>
            <ae:displayName>Est Date to Fix</ae:displayName>
            <ae:fieldType>FLDTYPE-DATETIME</ae:fieldType>
         </ae:columns>
      </ae:reportDefinition>
      <ae:result>
         <ae:itemURL>tmtrack.dll?IssuePage&amp;amp;RecordId=6&amp;amp;
         →Template=view&amp;amp;TableId=1000</ae:itemURL>
         <ae:fieldValue>
            <ae:valueType>Text</ae:valueType>
            <ae:value>00038</ae:value>
         </ae:fieldValue>
         <ae:fieldValue>
            <ae:valueType>Text</ae:valueType>
            <ae:value>Texture image conversion</ae:value>
         </ae:fieldValue>
```

```
      <ae:fieldValue>
         <ae:valueType>State</ae:valueType>
         <ae:value>Assigned to Engineer</ae:value>
      </ae:fieldValue>
      <ae:fieldValue>
         <ae:valueType>Date/Time</ae:valueType>
         <ae:value>5 0:00:00</ae:value>
      </ae:fieldValue>
      <ae:fieldValue>
         <ae:valueType>Numeric</ae:valueType>
         <ae:value>2</ae:value>
      </ae:fieldValue>
      <ae:fieldValue>
         <ae:valueType>Date/Time</ae:valueType>
         <ae:value>07/18/2007</ae:value>
      </ae:fieldValue>
   </ae:result>
   <ae:result>
      <ae:itemURL>tmtrack.dll?IssuePage&amp;amp;RecordId=7&amp;amp;
      →Template=view&amp;amp;TableId=1000</ae:itemURL>
      <ae:fieldValue>
         <ae:valueType>Text</ae:valueType>
         <ae:value>00039</ae:value>
      </ae:fieldValue>
      <ae:fieldValue>
         <ae:valueType>Text</ae:valueType>
         <ae:value>Upgrade v3.0 not working correctly.</ae:value>
      </ae:fieldValue>
      <ae:fieldValue>
         <ae:valueType>State</ae:valueType>
         <ae:value>Assigned to Engineer</ae:value>
      </ae:fieldValue>
      <ae:fieldValue>
         <ae:valueType>Date/Time</ae:valueType>
         <ae:value>1:00:00</ae:value>
      </ae:fieldValue>
      <ae:fieldValue>
         <ae:valueType>Numeric</ae:valueType>
         <ae:value>3</ae:value>
      </ae:fieldValue>
      <ae:fieldValue>
         <ae:valueType>Date/Time</ae:valueType>
         <ae:value>07/08/2007</ae:value>
      </ae:fieldValue>
   </ae:result>
   <ae:result>
      <ae:itemURL>tmtrack.dll?IssuePage&amp;amp;RecordId=11&amp;amp;
      →Template=view&amp;amp;TableId=1000</ae:itemURL>
      <ae:fieldValue>
         <ae:valueType>Text</ae:valueType>
         <ae:value>00044</ae:value>
      </ae:fieldValue>
      <ae:fieldValue>
         <ae:valueType>Text</ae:valueType>
         <ae:value>Color Palette is not saving changes</ae:value>
```

```
            </ae:fieldValue>
            <ae:fieldValue>
               <ae:valueType>State</ae:valueType>
               <ae:value>Assigned to Engineer</ae:value>
            </ae:fieldValue>
            <ae:fieldValue>
               <ae:valueType>Date/Time</ae:valueType>
               <ae:value>1 0:00:00</ae:value>
            </ae:fieldValue>
            <ae:fieldValue>
               <ae:valueType>Numeric</ae:valueType>
               <ae:value>3</ae:value>
            </ae:fieldValue>
            <ae:fieldValue>
               <ae:valueType>Date/Time</ae:valueType>
               <ae:value>07/17/2007</ae:value>
            </ae:fieldValue>
         </ae:result>
         <ae:result>
            <ae:itemURL>tmtrack.dll?IssuePage&amp;RecordId=20&amp;
            →Template=view&amp;TableId=1000</ae:itemURL>
            <ae:fieldValue>
               <ae:valueType>Text</ae:valueType>
               <ae:value>00063</ae:value>
            </ae:fieldValue>
            <ae:fieldValue>
               <ae:valueType>Text</ae:valueType>
               <ae:value>Help file for images is missing</ae:value>
            </ae:fieldValue>
            <ae:fieldValue>
               <ae:valueType>State</ae:valueType>
               <ae:value>Assigned to Engineer</ae:value>
            </ae:fieldValue>
            <ae:fieldValue>
               <ae:valueType>Date/Time</ae:valueType>
               <ae:value>4:00:00</ae:value>
            </ae:fieldValue>
            <ae:fieldValue>
               <ae:valueType>Numeric</ae:valueType>
               <ae:value>4</ae:value>
            </ae:fieldValue>
            <ae:fieldValue>
               <ae:valueType>Date/Time</ae:valueType>
               <ae:value>07/08/2007</ae:value>
            </ae:fieldValue>
         </ae:result>
      </ae:return>
   </ae:RunReportResponse>
```

SolutionData

Description

The SolutionData type holds the name and other information about an solution in Business Mashups. The SolutionData type parameters are listed below.

Parameters

Name	Type	Description
solutionID	integer	This is the internal TS_ID of the solution from the TS_SOLUTIONS table.
solutionUUID	string	This is the alternate unique identifier for the solution.
name	string	The name of the solution. Derived from TS_NAME column in TS_SOLUTIONS.
displayName	string	The display name of the solution.
type	Solution-Type [page 164]	The type of solution. The possible values are: TEAMTRACK_SOLUTION, USER_SOLUTION, and THIRD_PARTYSOLUTION.
prefix	string	The prefix of the solution. Derived from the TS_PREFIX column in TS_SOLUTIONS.
description	string	The description of the solution. Derived from the TS_DESCRIPTION column in TS_SOLUTIONS.

Usage

The SolutionData type holds the ID, UUID, name and description used to describe a solution. You can use GetSolutions [page 70] to retrieve the solution data shown here.

XML

The following XML snippet shows the SolutionData type in the return element of the GetSolutions response.

```
<ae:GetSolutionsResponse>
 <ae:return>
  <ae:solutionID>1</ae:solutionID>
  <ae:solutionUUID>29feba75-af30-4718-bcdc-3c9f287746c8</ae:solutionUUID>
  <ae:name>Issue Management</ae:name>
  <ae:displayName>Issue Management</ae:displayName>
  <ae:type>TEAMTRACK-SOLUTION</ae:type>
  <ae:prefix>TTT</ae:prefix>
  <ae:description>TeamTrack Issue Management</ae:description>
 </ae:return>
 <ae:return>
  <ae:solutionID>2</ae:solutionID>
  <ae:solutionUUID>19c38428-7489-4a3d-811b-25e3c33fbc64</ae:solutionUUID>
  <ae:name>Incident Mgmt</ae:name>
  <ae:displayName>Incident Management</ae:displayName>
  <ae:type>TEAMTRACK-SOLUTION</ae:type>
  <ae:prefix>TTS</ae:prefix>
  <ae:description>TeamTrack Incident Management</ae:description>
```

```
 </ae:return>
 <ae:return>
  <ae:solutionID>3</ae:solutionID>
  <ae:solutionUUID>07a570f3-be5b-4ce0-bbc3-cf3b65e2a577</ae:solutionUUID>
  <ae:name>HR Requests</ae:name>
  <ae:displayName>HR Requests</ae:displayName>
  <ae:type>USER-SOLUTION</ae:type>
  <ae:prefix>USR</ae:prefix>
  <ae:description></ae:description>
 </ae:return>
 <ae:return>
  <ae:solutionID>4</ae:solutionID>
  <ae:solutionUUID>8eb5e91f-2451-4f1d-a77c-70321dfcfac3</ae:solutionUUID>
  <ae:name>IT Help Desk</ae:name>
  <ae:displayName>IT Help Desk</ae:displayName>
  <ae:type>USER-SOLUTION</ae:type>
  <ae:prefix>USR</ae:prefix>
  <ae:description></ae:description>
 </ae:return>
</ae:GetSolutionsResponse>
```

Solution-Type

Description

Solution-Type indicates the type of solution.

Parameters

Name	Type	Description
TEAMTRACK-SOLUTION	string	Indicates that the solution was created by Serena.
USER-SOLUTION	string	Indicates that the solution was created by a user.
THIRDPARTY-SOLUTION	string	Indicates that the solution was created by a third party.

Usage

The Solution-Type helps identify the type of solution returned in the SolutionData response. For more information, see SolutionData [page 162].

XML

The following XML shows Solution-Type as seen in a typical response.

```
<ae:GetSolutionsResponse>
 <ae:return>
  <ae:solutionID>1</ae:solutionID>
  <ae:solutionUUID>29feba75-af30-4718-bcdc-3c9f287746c8</ae:solutionUUID>
  <ae:name>Issue Management</ae:name>
  <ae:displayName>Issue Management</ae:displayName>
  <ae:type>TEAMTRACK-SOLUTION</ae:type>
```

```
 <ae:prefix>TTT</ae:prefix>
 <ae:description>TeamTrack Issue Management</ae:description>
</ae:return>
```

SolutionWithUniqueName

Description

The SolutionWithUniqueName type holds the name and other information about an solution in Business Mashups. The SolutionWithUniqueName type parameters are listed below.

Parameters

Name	Type	Description
solutionID	integer	This is the internal TS_ID of the solution from the TS_SOLUTIONS table.
solutionUUID	string	This is the alternate unique identifier for the solution.
uniqueName	string	The unique name of the solution. Derived from TS_NAME column in TS_SOLUTIONS.
name	string	The tab name for the solution. Derived from TS_TABNAME in TS_SOLUTIONS.
displayName	string	The display name of the solution. Derived from TS_DISPLAYNAME in TS_SOLUTIONS.
type	Solution-Type [page 164]	The type of solution. The possible values are: TEAMTRACK_SOLUTION, USER_SOLUTION, and THIRD_PARTYSOLUTION.
prefix	string	The prefix of the solution. Derived from the TS_PREFIX column in TS_SOLUTIONS.
description	string	The description of the solution. Derived from the TS_DESCRIPTION column in TS_SOLUTIONS.

Usage

The SolutionWithUniqueName type holds the ID, UUID, unique solution name and description used to describe a solution. You can use GetSolutionsWithUniqueName [page 71] to retrieve the solution data shown here.

XML

The following XML snippet shows the SolutionWithUniqueName type in the return element of the GetSolutionsWithUniqueName response.

```
<ae:GetSolutionsWithUniqueNameResponse>
 <ae:return>
  <ae:solutionID>1</ae:solutionID>
  <ae:solutionUUID>29feba75-af30-4718-bcdc-3c9f287746c8</ae:solutionUUID>
  <ae:uniqueName>BASE_ISSUE_DEFECT_TRACKING</ae:uniqueName>
  <ae:name>IDT</ae:name>
  <ae:displayName>Base Issue Defect Tracking</ae:displayName>
  <ae:type>TEAMTRACK-SOLUTION</ae:type>
  <ae:prefix>TTT</ae:prefix>
  <ae:description>TeamTrack Issue Management</ae:description>
 </ae:return>
 <ae:return>
  <ae:solutionID>2</ae:solutionID>
  <ae:solutionUUID>19c38428-7489-4a3d-811b-25e3c33fbc64</ae:solutionUUID>
  <ae:uniqueName>INCIDENT_MANAGEMENT</ae:uniqueName>
  <ae:name>Incident Mgmt</ae:name>
  <ae:displayName>Incident Management</ae:displayName>
  <ae:type>TEAMTRACK-SOLUTION</ae:type>
  <ae:prefix>TTS</ae:prefix>
  <ae:description>TeamTrack Incident Management</ae:description>
 </ae:return>
 <ae:return>
  <ae:solutionID>3</ae:solutionID>
  <ae:solutionUUID>07a570f3-be5b-4ce0-bbc3-cf3b65e2a577</ae:solutionUUID>
  <ae:uniqueName>HR_REQUESTS</ae:uniqueName>
  <ae:name>HR Requests</ae:name>
  <ae:displayName>HR Requests</ae:displayName>
  <ae:type>USER-SOLUTION</ae:type>
  <ae:prefix>USR</ae:prefix>
  <ae:description></ae:description>
 </ae:return>
 <ae:return>
  <ae:solutionID>4</ae:solutionID>
  <ae:solutionUUID>8eb5e91f-2451-4f1d-a77c-70321dfcfac3</ae:solutionUUID>
  <ae:uniqueName>IT_HELP_DESK<ae:uniqueName>
  <ae:name>IT Help Desk</ae:name>
  <ae:displayName>IT Help Desk</ae:displayName>
  <ae:type>USER-SOLUTION</ae:type>
  <ae:prefix>USR</ae:prefix>
  <ae:description></ae:description>
 </ae:return>
</ae:GetSolutionsResponse>
```

TableData

Description

The TableData type holds the name and other information about a table in Business Mashups. The TableData type parameters are listed below.

Parameters

Name	Type	Description
tableID	integer	This is the internal TS_ID of the table from the TS_TABLES table.
tableUUID	string	This is the alternate unique identifier for the table.
solutionID	integer	This is the TS_ID of the solution from the TS_SOLUTIONS table. Once a table is tied to a workflow, a solution ID is created.
type	Table-Type [page 168]	The type of table returned.
name	string	The database name of the table. Derived from TS_DBNAME column in TS_TABLES.
displayName	string	The logical name of the table. Derived from TS_NAME column in TS_TABLES.
description	string	The description of the table. Derived from the TS_DESCRIPTION column in TS_TABLES.
fieldList	Field [page 140]	List of fields defined for the table.

Usage

The TableData type holds the ID, UUID, name and description used to describe a table. You can use GetTables [page 74] to retrieve the table data shown here.

XML

The following XML snippet shows the TableData type in the return element of the GetTables response.

```
<ae:GetTablesResponse>
   <ae:return>
      <ae:tableID>1003</ae:tableID>
      <ae:tableUUID>b82d2e9f-fea7-4669-84b6-7d5f7d5f50e9</ae:tableUUID>
      <ae:solutionID>7</ae:solutionID>
      <ae:type>PRIMARY-TABLE</ae:type>
      <ae:name>USR_SPRINTS</ae:name>
      <ae:displayName>Sprints</ae:displayName>
```

```
            <ae:description/>
            <ae:fieldList>
               <ae:fieldID>121</ae:fieldID>
               <ae:fieldUUID>d18e91a7-ea80-4e55-9838-909a7a1b642e</ae:fieldUUID>
               <ae:name>ISSUEID</ae:name>
               <ae:displayName>Sprint Id</ae:displayName>
               <ae:fieldType>FLDTYPE-TEXT</ae:fieldType>
            </ae:fieldList>
         </ae:return>
      </ae:GetTablesResponse>
```

Table-Type

Description

Table-Type indicates the type of table. The available table types are listed below.

Parameters

Name	Type	Description
NOT-SPECIFIED	string	Used to not specify a table type.
SYSTEM-TABLE	string	A table type that is inherent to any Business Mashups database.
PRIMARY-TABLE	string	A table that stores information about primary items, which follow an application workflow process.
AUXILIARY-TABLE	string	A table that stores information that may be needed repeatedly. Stores records that do not follow an application workflow process.
SYSTEM-AUXILIARY-TABLE	string	An built-in auxiliary table provided by Business Mashups
ARCHIVE-TABLE	string	A built-in table used to store archived primary or auxiliary records.

Usage

Table-Type can be used to limit the types of tables returned in GetTables [page 74]. It can also be used to identify the type of table returned in the GetTables response.

XML

The following XML shows Table-Type as seen in the type element of a GetTables response.

```
<ae:GetTablesResponse>
 <ae:return>
  <ae:tableID>1003</ae:tableID>
  <ae:tableUUID>b82d2e9f-fea7-4669-84b6-7d5f7d5f50e9</ae:tableUUID>
```

```
  <ae:solutionID>7</ae:solutionID>
  <ae:type>PRIMARY-TABLE</ae:type>
  <ae:name>USR_SPRINTS</ae:name>
  <ae:displayName>Sprints</ae:displayName>
  <ae:description/>
  <ae:fieldList>
   <ae:fieldID>121</ae:fieldID>
   <ae:fieldUUID>d18e91a7-ea80-4e55-9838-909a7a1b642e</ae:fieldUUID>
   <ae:name>ISSUEID</ae:name>
   <ae:displayName>Sprint Id</ae:displayName>
   <ae:fieldType>FLDTYPE-TEXT</ae:fieldType>
  </ae:fieldList>
 </ae:return>
</ae:GetTablesResponse>
```

TimePreference

Description

TimePreference indicates a user's preferred time format. The available options are listed below.

Parameters

Name	Type	Description
TIME-FORMAT-12HOUR	string	Use a 12 hour clock.
TIME-FORMAT-24HOUR	string	Use a 24 hour clock
TIME-FORMAT-USE-GMT-OFFSET	string	Use a GMT offset.
TIME-FORMAT-HONOR-DAYLIGHT	string	Honor daylight savings.

Usage

TimePreference is used to determine how time is displayed to a user in the Web interface. The various time formats are returned in the timePreference parameter of the GetUserWithPreferences call. See the UserWithPreferences [page 180] response for more information.

XML

The following XML shows TimePreference as seen in the return element of a GetUserWithPreferences response.

```
<ae:GetUserWithPreferencesResponse>
 <ae:return>
  <ae:userId>chris</ae:userId>
  <ae:userName>Chris Tester</ae:userName>
  <ae:id>13</ae:id>
  <ae:uuid>35a48696-4307-4fa9-a429-de35d262d820</ae:uuid>
  <ae:email>ctester@serenateamtrackdb.com</ae:email>
  <ae:emailCC></ae:emailCC>
```

```
    <ae:timezone>America/Denver</ae:timezone>
    <ae:offsetFromGMT>-25200000</ae:offsetFromGMT>
    <ae:dstSavings>3600000</ae:dstSavings>
    <ae:datePreference>DATE-FORMAT-MM-DD-YYYY</ae:datePreference>
    <ae:timePreference>TIME-FORMAT-12HOUR</ae:timePreference>
    <ae:namespaceName>00000</ae:namespaceName>
   </ae:return>
  </ae:GetUserWithPreferencesResponse>
```

Transition

Description

The Transition type holds the name and other information about a transition in Business Mashups. The Transition type parameters are listed below.

Parameters

Name	Type	Description
transitionID	integer	This is the internal TS_ID of the transition from the TS_TRANSITIONS table.
transitionUUID	string	This is the alternate unique identifier for the transition.
name	string	The name of the transition. Derived from the TS_NAME column in TS_TRANSITIONS.
fromState	string	The name of the state from which the transition begins.
fromStateID	integer	The ID of the state from which the transition begins.
fromStateUUID	string	The UUID of the state from which the transition begins.
toState	string	The name of the state to which the transition moves or ends.
toStateID	integer	The ID of the state to which the transition moves or ends.
toStateUUID	string	The UUID of the state to which the transition moves or ends.

Name	Type	Description
type	Transition-Type [page 172]	The type of transition.
fullyQualifiedPostIssueProjectName	string	For a TRANSITION_POST type of transition, this indicates the fully qualified project name into which the item is to be submitted.
transitionAttributes	string	The list of transition attributes associated with this transition.

Usage

The Transition type completely describes an available transition or transitions on an item in Business Mashups. Use GetAvailableTransitionsWithStateIDs [page 58] or GetAvailableTransitions [page 57] to retrieve a list of transitions available for a specified item.

Note: The fromStateID, fromStateUUID, toStateID, and toStateUUID parameters are only returned with the GetAvailableTransitionsWithStateID call.

For more information on the fullyQualifiedPostIssueProjectName, see the fullyQualifiedProjectName argument in the **Usage** section of the CreatePrimaryItemWithName [page 42] call.

The transitionAttributes argument is optional and is only used as another means to describe the transition. A transition attribute is typically associated with unique enabler license - integration transitions.

XML

The following XML snippet shows the Transition type in the return element of the GetAvailableTransitions response.

```
<ae:GetAvailableTransitionsResponse>
 <ae:return>
  <ae:transitionID>1</ae:transitionID>
  <ae:transitionUUID>update</ae:transitionUUID>
  <ae:name>Update</ae:name>
  <ae:fromState>Any</ae:fromState>
  <ae:fromStateID>1</ae:fromStateID>
  <ae:fromStateUUID>aa6e6c71-c7be-4ea0-aa18-34ecf9220327</ae:fromStateUUID>
  <ae:toState>Same</ae:toState>
  <ae:toStateID>2</ae:toStateID>
  <ae:toStateUUID>7718fe04-7dc2-4fdc-b01c-4bea5ff96253</ae:toStateUUID>
  <ae:type>TRANSITION-UPDATE</ae:type>
 </ae:return>
 <ae:return>
  <ae:transitionID>2</ae:transitionID>
  <ae:transitionUUID>delete</ae:transitionUUID>
```

```
  <ae:name>Delete</ae:name>
  <ae:fromState>New</ae:fromState>
  <ae:fromStateID>23</ae:fromStateID>
  <ae:fromStateUUID>ef4c6e4b-eacd-49e3-8fd9-79b555c8796a</ae:fromStateUUID>
  <ae:toState>Deleted</ae:toState>
  <ae:toStateID>3</ae:toStateID>
  <ae:toStateUUID>b67c66d1-a5a3-49e6-b6ae-941eb6d315e5</ae:toStateUUID>
  <ae:type>TRANSITION-DELETE</ae:type>
 </ae:return>
 <ae:return>
  <ae:transitionID>62</ae:transitionID>
  <ae:transitionUUID>e8050ea6-198f-4944-90f7-d95867956d43
  →</ae:transitionUUID>
  <ae:name>Review</ae:name>
  <ae:fromState>State</ae:fromState>
  <ae:fromStateID>6</ae:fromStateID>
  <ae:fromStateUUID>e6ed1a6d-f50a-44ee-98cf-a5e7471479cd</ae:fromStateUUID>
  <ae:toState>In Review</ae:toState>
  <ae:toStateID>22</ae:toStateID>
  <ae:toStateUUID>d3fc2861-15a2-461b-b7ba-72b4a4d79fb2</ae:toStateUUID>
  <ae:type>TRANSITION-REGULAR</ae:type>
 </ae:return>
</ae:GetAvailableTransitionsResponse>
```

Transition-Type

Description

Transition-Type indicates the type of transition. The available transition types are listed below.

Parameters

Name	Type	Description
TRANSITION-REGULAR	string	Denotes a Regular transition.
TRANSITION-COPY	string	Denotes a Copy transition.
TRANSITION-POST	string	Denotes a Post transition that submits a record into an application table based on the transition of a primary item.
TRANSITION-SUBMITPROBLEM	string	Denotes a Publish transition that creates a Knowledge Base problem or resolution.
TRANSITION-MOBILE	string	Denotes a transition that is available to Mobile Connect users.
TRANSITION-SUBTASK	string	Denotes a transition that creates a Subtask.

Name	Type	Description
TRANSITION-UPDATE	string	Denotes an Update transition.
TRANSITION-DELETE	string	Denotes a Delete transition.
TRANSITION-EXTERNALPOST	string	Denotes a transition that will submit a record into an external database by sending an email message to perform a special kind of e-mail submission.

Usage

Transition-Type is used to describe the types of transition returned in the Transition response returned from the various GetTransition calls. For more information see GetAvailableTransitions [page 57].

XML

The following XML shows Transition-Type as seen in the type element of a GetAvailableTransitionsResponse.

```
<ae:GetAvailableTransitionsResponse>
 <ae:return>
  <ae:transitionID>1</ae:transitionID>
  <ae:transitionUUID>update</ae:transitionUUID>
  <ae:name>Update</ae:name>
  <ae:fromState>Any</ae:fromState>
  <ae:fromStateID>1</ae:fromStateID>
  <ae:fromStateUUID>aa6e6c71-c7be-4ea0-aa18-34ecf9220327</ae:fromStateUUID>
  <ae:toState>Same</ae:toState>
  <ae:toStateID>2</ae:toStateID>
  <ae:toStateUUID>7718fe04-7dc2-4fdc-b01c-4bea5ff96253</ae:toStateUUID>
  <ae:type>TRANSITION-UPDATE</ae:type>
 </ae:return>
 <ae:return>
  <ae:transitionID>2</ae:transitionID>
  <ae:transitionUUID>delete</ae:transitionUUID>
  <ae:name>Delete</ae:name>
  <ae:fromState>New</ae:fromState>
  <ae:fromStateID>23</ae:fromStateID>
  <ae:fromStateUUID>ef4c6e4b-eacd-49e3-8fd9-79b555c8796a</ae:fromStateUUID>
  <ae:toState>Deleted</ae:toState>
  <ae:toStateID>3</ae:toStateID>
  <ae:toStateUUID>b67c66d1-a5a3-49e6-b6ae-941eb6d315e5</ae:toStateUUID>
  <ae:type>TRANSITION-DELETE</ae:type>
 </ae:return>
 <ae:return>
  <ae:transitionID>62</ae:transitionID>
  <ae:transitionUUID>e8050ea6-198f-4944-90f7-d95867956d43
  →</ae:transitionUUID>
  <ae:name>Review</ae:name>
  <ae:fromState>State</ae:fromState>
```

```
    <ae:fromStateID>6</ae:fromStateID>
    <ae:fromStateUUID>e6ed1a6d-f50a-44ee-98cf-a5e7471479cd</ae:fromStateUUID>
    <ae:toState>In Review</ae:toState>
    <ae:toStateID>22</ae:toStateID>
    <ae:toStateUUID>d3fc2861-15a2-461b-b7ba-72b4a4d79fb2</ae:toStateUUID>
    <ae:type>TRANSITION-REGULAR</ae:type>
   </ae:return>
  </ae:GetAvailableTransitionsResponse>
```

TTItem

Description

The TTItem type holds all of the Business Mashups field information for an item along with attached notes and item links. The TTItem type parameters are listed below.

Parameters

Name	Type	Description
genericItem	Item [page 107]	The generic item containing the unique and display identifiers.
itemType	string	Describes items, such as defects or change requests, tracked in a workflow. ItemType is only applicable to primary table items.
classification	string	Otherwise known as project. *Classification* is used to declare the project the item belongs to. *Classification* is only applicable to primary table items.
classificationUUID	string	Alternate unique identifier (not the TS_ID) for a project.
title	string	The 80 character fixed length title of the item.
description	string	A description of the item. This is the value given to the system Description field, which is a Text/Memo field.
createdBy	string	The login ID of the user that created the item.
createDate	dateTime	The date the item was created. See Supported Date/Time Formats [page 184] for more information.
modifiedBy	string	The login ID of the user that last modified the item.

Name	Type	Description
modifiedDate	dateTime	The date and time when the item was last modified. See Supported Date/Time Formats [page 184] for more information.
activeInactive	boolean	Indicates whether the item is active (true) or inactive (false). Defaults to true.
state	string	The current state of the item. State is only applicable to primary table items.
owner	string	The login ID of the current owner of the item. Owner is only applicable to primary table items.
url	string	The exact URL of the item. The detailed view of the item can be accessed via this URL.
extendedFieldList	NameValue [page 110]	Additional fields and values not otherwise specified in *TTItem* are set using the *extendedFieldList*.
noteList	Note [page 113]	The list of notes attached to the item.
itemLinkList	ItemLink [page 108]	The list of items linked to this item.
urlAttachmentList	URLAttachment [page 132]	The list of URL attachments associated with this item.
fileAttachmentList	FileAttachment [page 105]	The list of file attachments associated with this item.

Usage

The TTitem type is essentially used describe the fields of an auxiliary or primary item in Business Mashups.

- **url**

  The detailed view of the item can be accessed via this URL. The URL is of the form `"tmtrack.dll?IssuePage&RecordId=10&Template=view&TableId=1002"`, so the `"http://host:port/tmtrack/"` must be pre-pended.

- **extendedFieldList**

  The extendedFieldList is a collection of fields and field values that aren't specifically set elsewhere in TTItem.

XML

The following XML snippet shows the TTitem type in the return element of the UpdateItem response.

```
<ae:UpdateItemResponse>
 <ae:return>
  <ae:genericItem>
   <ae:itemID>1000:145</ae:itemID>
   <ae:itemUUID>70fe1c9c-1b4c-4226-a8a3-200b6553fbff</ae:itemUUID>
   <ae:itemName>BUG00110</ae:itemName>
  </ae:genericItem>
  <ae:itemType>Bug Report</ae:itemType>
  <ae:classification>Animation Pro</ae:classification>
  <ae:classificationUUID>d0fc3213-bc68-4ecd-a599-f790d78ffe54
  →</ae:classificationUUID>
  <ae:title>Title-CreatePrimItem-Test3</ae:title>
  <ae:description>A Description</ae:description>
  <ae:createdBy>admin</ae:createdBy>
  <ae:createDate>2008-03-06T22:58:29-08:00</ae:createDate>
  <ae:modifiedBy>admin</ae:modifiedBy>
  <ae:modifiedDate>2008-03-06T22:58:29-08:00</ae:modifiedDate>
  <ae:activeInactive>true</ae:activeInactive>
  <ae:state>New</ae:state>
  <ae:owner>joe</ae:owner>
  <ae:url>http://server:80/tmtrack/tmtrack.dll?IssuePage&amp;
  →RecordId=145&amp;Template=view&amp;TableId=1000</ae:url>
  <ae:extendedFieldList>
   <ae:name>CLOSEDATE</ae:name>
   <ae:id>39</ae:id>
   <ae:uuid>73fbd37a-3b8d-484f-86de-0bfdcc428f2c</ae:uuid>
   <ae:setValueBy>PRECEDENCE-VALUE</ae:setValueBy>
   <ae:setValueMethod></ae:setValueMethod>
   <ae:value>
    <ae:displayValue></ae:displayValue>
    <ae:internalValue></ae:internalValue>
    <ae:uuid></ae:uuid>
   </ae:value>
  </ae:extendedFieldList>
  <ae:extendedFieldList>
   <ae:name>CONTACT</ae:name>
    <ae:id>228</ae:id>
    <ae:uuid>a86e1220-a065-4d59-9251-f7e8ee042c2f</ae:uuid>
    <ae:value>
     <ae:displayValue/>
     <ae:internalValue>38:0</ae:internalValue>
     <ae:uuid/>
    </ae:value>
  </ae:extendedFieldList>
  <ae:extendedFieldList>
   <ae:name>CONTACT:PHONE_NUMBER</ae:name>
    <ae:id>233</ae:id>
    <ae:uuid>5534f14e-65d8-4a36-a2b1-0584d6e0ae7f</ae:uuid>
    <ae:value>
     <ae:displayValue>800-555-5555</ae:displayValue>
     <ae:internalValue/>800-555-5555</ae:internalValue>
     <ae:uuid/>
    </ae:value>
  </ae:extendedFieldList>
  <ae:extendedFieldList>
```

```
 <ae:name>LASTSTATECHANGEDATE</ae:name>
 <ae:id>48</ae:id>
 <ae:uuid>e6657e37-953a-42f9-b6cb-0a2b09fcd996</ae:uuid>
 <ae:setValueBy>PRECEDENCE-VALUE</ae:setValueBy>
 <ae:setValueMethod></ae:setValueMethod>
 <ae:value>
  <ae:displayValue>2008-03-07T06:58:29+00:00</ae:displayValue>
  <ae:internalValue>2008-03-07T06:58:29+00:00</ae:internalValue>
  <ae:uuid></ae:uuid>
 </ae:value>
</ae:extendedFieldList>
<ae:extendedFieldList>
 <ae:name>LASTSTATECHANGER</ae:name>
 <ae:id>49</ae:id>
 <ae:uuid>3f7c2f6d-e9e7-44ab-a990-0116f8f365cc</ae:uuid>
 <ae:setValueBy>PRECEDENCE-VALUE</ae:setValueBy>
 <ae:setValueMethod></ae:setValueMethod>
 <ae:value>
  <ae:displayValue>admin</ae:displayValue>
  <ae:internalValue>11</ae:internalValue>
  <ae:uuid>0bfd6a6d-0084-41c7-b3e8-b694a6ee5e99</ae:uuid>
 </ae:value>
</ae:extendedFieldList>
<ae:extendedFieldList>
 <ae:name>PLANNED_COMPLETION_DATE</ae:name>
 <ae:id>148</ae:id>
 <ae:uuid>6b0ac9e7-0dd2-4ed6-b0a1-45f3db4e3ecf</ae:uuid>
 <ae:setValueBy>PRECEDENCE-VALUE</ae:setValueBy>
 <ae:setValueMethod></ae:setValueMethod>
 <ae:value>
  <ae:displayValue>1970-08-20</ae:displayValue>
  <ae:internalValue>1970-08-20T00:00:00+00:00</ae:internalValue>
  <ae:uuid></ae:uuid>
 </ae:value>
</ae:extendedFieldList>
<ae:extendedFieldList>
 <ae:name>ADDITIONAL_NOTES</ae:name>
 <ae:id>150</ae:id>
 <ae:uuid>2ab77b9b-2c29-41b5-a2dc-f4809d7ac4a6</ae:uuid>
 <ae:setValueBy>PRECEDENCE-VALUE</ae:setValueBy>
 <ae:setValueMethod></ae:setValueMethod>
 <ae:value>
  <ae:displayValue>additional note txt</ae:displayValue>
  <ae:internalValue>additional note txt</ae:internalValue>
  <ae:uuid></ae:uuid>
 </ae:value>
</ae:extendedFieldList>
<ae:extendedFieldList>
 <ae:name>FUNCTIONAL_AREA</ae:name>
 <ae:id>151</ae:id>
 <ae:uuid>cbc434e0-556b-4fa4-a903-c76bc11233c9</ae:uuid>
 <ae:setValueBy>PRECEDENCE-VALUE</ae:setValueBy>
 <ae:setValueMethod></ae:setValueMethod>
 <ae:value>
  <ae:displayValue>User Interface</ae:displayValue>
```

```
    <ae:internalValue>75</ae:internalValue>
    <ae:uuid>fddc634f-9e60-4a3b-87ee-c304c02b0543</ae:uuid>
   </ae:value>
  </ae:extendedFieldList>
  <ae:extendedFieldList>
   <ae:name>SECONDARYOWNER</ae:name>
   <ae:id>268</ae:id>
   <ae:uuid>33142ec1-9fbe-4ea8-97bf-4c4fceb04a6a</ae:uuid>
   <ae:setValueBy>PRECEDENCE-VALUE</ae:setValueBy>
   <ae:setValueMethod></ae:setValueMethod>
  </ae:extendedFieldList>
  <ae:extendedFieldList>
   <ae:name>REVIEWERS</ae:name>
   <ae:id>269</ae:id>
   <ae:uuid>cb32b40b-549b-4d55-9d2a-723a975af18e</ae:uuid>
   <ae:setValueBy>PRECEDENCE-VALUE</ae:setValueBy>
   <ae:setValueMethod></ae:setValueMethod>
  </ae:extendedFieldList>
  <ae:extendedFieldList>
   <ae:name>PRODUCT</ae:name>
   <ae:id>274</ae:id>
   <ae:uuid>7a68c9e6-4892-45e4-ab84-52f78275a082</ae:uuid>
   <ae:setValueBy>PRECEDENCE-VALUE</ae:setValueBy>
   <ae:setValueMethod></ae:setValueMethod>
   <ae:value>
    <ae:displayValue></ae:displayValue>
    <ae:internalValue>1004:0</ae:internalValue>
    <ae:uuid></ae:uuid>
   </ae:value>
  </ae:extendedFieldList>
  <ae:extendedFieldList>
   <ae:name>REQUIRED_COMPLETION_DATE</ae:name>
   <ae:id>279</ae:id>
   <ae:uuid>6c3f701a-11cf-4210-945d-a517a3fa085b</ae:uuid>
   <ae:setValueBy>PRECEDENCE-VALUE</ae:setValueBy>
   <ae:setValueMethod></ae:setValueMethod>
   <ae:value>
    <ae:displayValue>1970-08-20</ae:displayValue>
    <ae:internalValue>1970-08-20T00:00:00+00:00</ae:internalValue>
    <ae:uuid></ae:uuid>
   </ae:value>
  </ae:extendedFieldList>
  <ae:extendedFieldList>
   <ae:name>TIMEOFDAY</ae:name>
   <ae:id>295</ae:id>
   <ae:uuid>068afcbd-cbdc-48f8-823b-2b96ff6d3e22</ae:uuid>
   <ae:setValueBy>PRECEDENCE-VALUE</ae:setValueBy>
   <ae:setValueMethod></ae:setValueMethod>
   <ae:value>
    <ae:displayValue>00:11:13</ae:displayValue>
    <ae:internalValue>673</ae:internalValue>
    <ae:uuid></ae:uuid>
   </ae:value>
  </ae:extendedFieldList>
```

```
 </ae:return>
</ae:UpdateItemResponse>
```

User

Description

The User type holds the name and other information about a user in Business Mashups. The User type parameters are listed below.

Parameters

Name	Type	Description
userId	string	The login ID of the user.
userName	string	The display name of the user.
id	integer	This is the internal TS_ID of the user from the TS_USERS table.
uuid	string	An alternate unique identifier for the user record.
email	string	The user's primary email address.
emailCC	string	The user's CC email addresses.
timezone	string	The user's time zone.
offsetFromGMT	integer	This is the time zone's raw GMT offset.
dstSavings	integer	The amount of time in milliseconds to be added to local standard time to get local wall clock time.
namespaceName	string	The user's namespace name.

Usage

The User type provides a user account in Business Mashups. Use GetUser [page 76] to retrieve a list of attributes for a specified user.

The timezone parameter is a programmatic ID; for example, "America/Los_Angeles". This ID is used to call up a specific real-world time zone. It corresponds to the IDs defined in the standard Olson data used by UNIX systems, and has the format continent/city or ocean/city.

The offsetFromGMT parameter is the time zone's raw GMT offset (i.e., the number of milliseconds to add to GMT to get local time, before taking Daylight Saving Time into account). If DST is in effect for a given date, use the dstSavings value to adjust this offset.

The dstSavings parameter is the amount of time in milliseconds to be added to local standard time to get local wall clock time. If Daylight Saving Time is not observed in this

user's timezone, this value will be 0. This value should be used only to adjust a date/time that is within the DST observation period.

The namespaceName parameter returns the name of the namespace that was generated while provisioning the customer environment. If the user does not belong to a namespace, then the default namespace name ("00000") is returned.

XML

The following XML snippet shows the User type in the return element of the GetUser response.

```
<ae:GetUserResponse>
 <ae:return>
  <ae:userId>chris</ae:userId>
  <ae:userName>Chris Tester</ae:userName>
  <ae:id>13</ae:id>
  <ae:uuid>35a48696-4307-4fa9-a429-de35d262d820</ae:uuid>
  <ae:email>ctester@serenateamtrackdb.com</ae:email>
  <ae:emailCC></ae:emailCC>
  <ae:timezone>America/Denver</ae:timezone>
  <ae:offsetFromGMT>-25200000</ae:offsetFromGMT>
  <ae:dstSavings>3600000</ae:dstSavings>
  <ae:namespaceName>00000</ae:namespaceName>
 </ae:return>
</ae:GetUserResponse>
```

UserWithPreferences

Description

The UserWithPreferences type holds the name and other information about a user in Business Mashups. The UserWithPreferences type parameters are listed below.

Parameters

Name	Type	Description
userId	string	The login ID of the user.
userName	string	The display name of the user.
id	integer	This is the internal TS_ID of the user from the TS_USERS table.
uuid	string	An alternate unique identifier for the user record.
email	string	The user's primary email address.
emailCC	string	The user's CC email addresses.
timezone	string	The user's time zone.

Name	Type	Description
offsetFromGMT	integer	This is the time zone's raw GMT offset.
dstSavings	integer	The amount of time in milliseconds to be added to local standard time to get local wall clock time.
datePreference	DatePreference [page 139]	This is the user's preferred date format.
timePreference	TimePreference [page 169]	This is the user's preferred time format.
namespaceName	string	The user's namespace name.

Usage

The UserWithPreferences type provides a user account in Business Mashups. Use GetUserWithPreferences [page 79] to retrieve a list of attributes for a specified user.

The timezone parameter is a programmatic ID; for example, "America/Los_Angeles". This ID is used to call up a specific real-world time zone. It corresponds to the IDs defined in the standard Olson data used by UNIX systems, and has the format continent/city or ocean/city.

The offsetFromGMT parameter is the time zone's raw GMT offset (i.e., the number of milliseconds to add to GMT to get local time, before taking Daylight Saving Time into account). If DST is in effect for a given date, use the dstSavings value to adjust this offset.

The dstSavings parameter is the amount of time in milliseconds to be added to local standard time to get local wall clock time. If Daylight Saving Time is not observed in this user's timezone, this value will be 0. This value should be used only to adjust a date/time that is within the DST observation period.

The namespaceName parameter returns the name of the namespace that was generated while provisioning the customer environment. If the user does not belong to a namespace, then the default namespace name ("00000") is returned.

XML

The following XML snippet shows the UserWithPreferences type in the return element of the GetUserWithPreferences response.

```
<ae:GetUserWithPreferencesResponse>
 <ae:return>
  <ae:userId>chris</ae:userId>
  <ae:userName>Chris Tester</ae:userName>
  <ae:id>13</ae:id>
  <ae:uuid>35a48696-4307-4fa9-a429-de35d262d820</ae:uuid>
  <ae:email>ctester@serenateamtrackdb.com</ae:email>
  <ae:emailCC></ae:emailCC>
  <ae:timezone>America/Denver</ae:timezone>
  <ae:offsetFromGMT>-25200000</ae:offsetFromGMT>
```

```
      <ae:dstSavings>3600000</ae:dstSavings>
      <ae:datePreference>DATE-FORMAT-MM-DD-YYYY</ae:datePreference>
      <ae:timePreference>TIME-FORMAT-12HOUR</ae:timePreference>
      <ae:namespaceName>00000</ae:namespaceName>
     </ae:return>
    </ae:GetUserWithPreferencesResponse>
```

Chapter 4: Tips for Writing Applications

This section provides guidelines for getting started writing applications that interact with the Business Mashups Web services.

- Authentication Methods [page 183]
- C++ Constants [page 183]
- Supported Character Encoding [page 184]
- Supported Date/Time Formats [page 184]
- Application, Table, and Project IDs [page 185]

Authentication Methods

The Business Mashups Web services API supports the following authentication methods. Choose the method most appropriate for your environment. Use of SSL is recommended with any of these authentication methods.

Argument

By default, the Business Mashups Web services use the argument method of authentication. Authentication occurs each time a method is called. The Auth argument passes the Business Mashups user ID and password in plain text. You can also use this argument to specify the host name for licensing purposes, instead of using the client's IP address.

HTTP Basic

HTTP Basic authentication is defined in the HTTP header. The Business Mashups user ID and password are passed with Base64 encoding.

To define HTTP Basic authentication in Visual Studio .NET, add an authorization header to the request. Typically you do this by overriding the GetWebRequest method. In the sample programs, you can use the `-basic` command-line argument to do this.

WS-Security

WS-Security (Web Services Security) authentication creates a security token in the SOAP header. The Business Mashups credentials are passed as a Username token and a Base64-encoded password.

To define WS-Security in Visual Studio .NET, add a Username token to the SOAP header, which contains the plain text user ID and the Base64-encoded password. You must also have Microsoft's WSE installed.

C++ Constants

For reference, Business Mashups provides constants in C++ format. You can find these constants in the `TSDef.h` file, located in the Business Mashups API package (`tsapi.zip`.

For information on the Business Mashups database, see the `schema.doc` file, also located in the API package.

Supported Character Encoding

The Business Mashups Web services use UTF-8 encoding. Single-byte characters are automatically supported with UTF-8. To enable support for multi-byte characters, you must set the expected encoding to UTF-8 on the client side. In Visual Studio 2005 C#, you can do this by overriding the GetReaderFromMessage method and setting the reader's encoding to UTF-8.

Supported Date/Time Formats

The Business Mashups date/time field values are strings and must be in XML date/time format. The value is expected to be in coordinated universal time (UTC). Below are four examples of date/time strings that you can send to the Business Mashups Web services:

#	Example	Explanation
1	2006-04-28	No offset from 00:00:00, April 28, 2006 *— or—* 2006-04-28T00:00:00+00:00
2	2006-04-28T13:20-6:00	6-hour offset from 1:20:00 p.m., April 28, 2006 *— or—* 2006-04-28T19:20:00+00:00
3	2006-04-28T19:20:30+0:00	No offset from 7:20:30 p.m., April 28, 2006
4	2006-04-28T13:20:30.45-6:00	6-hour offset from 1:20:30 p.m., April 28, 2006 *— or—* 2006-04-28T19:20:30+00:00

If you use the format shown in example 1, the Business Mashups Web services return *Date Only* field values in the same format. If you use the formats shown in examples 2, 3, or 4, the Business Mashups Web services return *Date Time* field values in the same format as example 3.

Note: Business Mashups does not currently store the fractions of a second when the format in example four is used.

The Business Mashups Web services API has two special system date/time fields, `item.createDate` and `item.modifiedDate`, which have values in seconds since January 1, 1970 (the modified Julian date). These fields are automatically populated by Business Mashups and it is recommended that you do not modify them. SOAP will convert these values to its date/time format for transport in XML.

You can explicitly set these values as shown in the following C# example code:

```
string MyString = "Jan 1, 2006 12:12:12 am";
```

```
DateTime MyDateTime = DateTime.Parse(MyString);

web.TTItem item = new web.TTItem();

item.createDate = MyDateTime;

item.modifiedDate = MyDateTime;
```

Application, Table, and Project IDs

The GetSolutions, GetTables, and GetSubmitProjects methods are a useful starting point to retrieve IDs of applications, tables, and projects. These IDs are required when you use certain methods, such as CreatePrimaryItem and DeleteItem.

Chapter 5: Sample Programs

- About the Sample Programs [page 187]
- Using the Sample Programs [page 188]

About the Sample Programs

Business Mashups provides five sample program source files that you can use with the Business Mashups sample database. Use these programs to see functional examples of C# source code for applications that interact with Business Mashups Web services. You can find these samples in your installation under `Application Engine\webservices\samples`.

The samples include:

- SolutionsAndTables [page 187]
- CreateItems [page 187]
- DeleteItems [page 188]
- GetItems [page 188]
- UpdateItems [page 188]

Important: Support for development efforts writing Web services is provided by Serena Consulting services. Questions regarding use of Web services operations in orchestration processes as documented are handled by Serena Customer support.

SolutionsAndTables

Demonstrates returning a list of applications, tables, and projects.

This program:

1. Returns the applications for which the user has permissions.
2. Returns the tables and fields for application 1.
3. Returns all auxiliary tables and fields.
4. Returns all primary tables and fields.
5. Returns the tables and fields for which the user has permissions.
6. Returns the projects for which the user has submit permissions for table 1000.
7. Returns the projects for which the user has submit permissions.

CreateItems

Demonstrates submitting items into a specified Business Mashups project, and creating auxiliary items within a specified table.

This program:

1. Creates three items in auxiliary table 1004.
2. Creates three items in the Image Builder project.

DeleteItems

Demonstrates deleting specified items.

This program:

1. Creates an item in the Image Builder project and then deletes the item.
2. Creates two items in the Image Builder project and then deletes the items.
3. Creates two items in the Image Builder project and then deletes the item using a query string.

GetItems

Demonstrates returning items based on user privileges.

This program:

- Returns all items from table 1000.

UpdateItems

Demonstrates updating item fields and updating items using a specified transition.

This program:

1. Creates an item in the Image Builder project.
2. Updates the title and description of the item.
3. Updates the item using the transition ID 0.
4. Creates two more items in the Image Builder project.
5. Updates the two items.

Using the Sample Programs

The following sections discuss the requirements for using the sample programs and what you should modify in the programs:

- Requirements [page 188]
- Choosing Authentication [page 189]
- Editing the Web Services URL [page 189]
- Rebuilding the Executables [page 189]

Requirements

You must have the following installed:

- Business Mashups 2009 R1 or later, connected to the sample database
- One of the following:
  - Microsoft Visual Studio .NET 2003
  - Microsoft Visual Studio .NET 2005
- (optional) For use with WS-Security authentication:
  - WSE 2.0 for .NET 2003

- WSE 3.0 or later for .NET 2005

  Download from http://msdn.microsoft.com/webservices/webservices/building/wse/default.aspx

Choosing Authentication

By default, the samples use the argument method for authentication. You can change the authentication method to HTTP Basic by using the `-basic` command-line argument.

For a description of the authentication methods, see Authentication Methods [page 183].

Editing the Web Services URL

By default, the Business Mashups Web services URL points to a server called `localhost:80`. To point to your actual server, locate the following files in each sample program directory:

- `Sample*.cs`
- `Reference.cs`
- `Reference.map`
- `aewebservices71.wsdl`

and change the server name and port number in this line:

```
ttserv.Url ="http://serverName:aePort/gsoap/gsoap_ssl.dll?aewebservices71";
```

Rebuilding the Executables

The sample programs include pre-built executables. If you have changed the Web services URL or the authentication method, rebuild the solution and use the new executables.

You can find the executables in the `bin\release` folder.