



# Marketo SOAP API Reference

---

For Customer Application Integration  
Version 1.7 Revision 2

Agha Ahsan, Erik Rehn, Bud Smith  
11/07/2011

# Table of Contents

|   |    |
|---|----|
| Overview.....                           | 4  |
| Configuring Your SOAP API Settings..... | 5  |
| Protocol .....                          | 6  |
| Security .....                          | 7  |
| Signature Algorithm.....                | 7  |
| Definitions .....                       | 9  |
| Stream Position .....                   | 9  |
| Success and Result Elements.....        | 9  |
| Lead Key.....                           | 9  |
| Context Header.....                     | 10 |
| Basic Operations.....                   | 11 |
| getLead .....                           | 11 |
| getMultipleLeads.....                   | 12 |
| getLeadActivity .....                   | 13 |
| Lead Synchronization Operations .....   | 15 |
| getLeadChanges.....                     | 15 |
| syncLead .....                          | 16 |
| syncMultipleLeads .....                 | 18 |
| Advanced Operations .....               | 19 |
| getCampaignsForSource.....              | 19 |
| listOperation.....                      | 20 |
| requestCampaign .....                   | 21 |
| Marketo Object Operations .....         | 22 |
| MObjects and Opportunities.....         | 22 |
| Structure of MObjects .....             | 23 |
| External-Keys .....                     | 23 |
| getMObjects .....                       | 23 |
| syncMObjects .....                      | 25 |
| deleteMObjects .....                    | 26 |

|  |    |
|--|----|
| listMObjects.....  | 26 |
| describeMObjects.....  | 27 |
| Error Codes.....   | 29 |
| Appendix A – Data Types.....                                 | 30 |
| Appendix B - Error Responses .....                           | 37 |
| SOAP Faults.....   | 37 |
| Service Exceptions .....                                     | 38 |
| Appendix C - Integrating with Marketo.....                   | 40 |
| Periodic Pull of Updated Leads.....                          | 40 |
| Periodic Pull of Changed Data Value .....                    | 41 |
| Periodic Push of Updated Leads.....                          | 41 |
| Capture New Lead .....                                       | 42 |
| Add Lead to Campaign.....                                    | 42 |
| Trigger Campaign with syncLead .....                         | 43 |
| Trigger Campaign with listOperation .....                    | 44 |
| Push Lead to Sales .....                                     | 45 |
| Appendix D. Marketo SOAP API Java Example.....               | 50 |
| Appendix E. Marketo SOAP API PHP Client .....                | 51 |
| Appendix F. Checking for User Registration Via SOAP API..... | 52 |
| Appendix G. Sample XML .....                                 | 54 |
| Sample XML Request and Response .....                        | 54 |
| MObject XML Schema .....                                     | 56 |
| Sample Objects.....  | 57 |
| Sample Request Fragments.....                                | 58 |
| Metadata .....   | 62 |

## Overview

This is the *Marketo SOAP API Reference*. The Marketo SOAP API gives you access to leads, actions, and more.

SOAP stands for Simple Object Access Protocol. You can use the Marketo SOAP API to access data stored in your instance of Marketo Lead Management, Marketo's main product. You can use the API with a variety of programming and scripting languages, including Java and PHP.

You can view the XML request elements in the WSDL (Web Services Definition Language) for this API, or look at the proxy classes that a SOAP toolkit generates from the WSDL. Use this reference to look at the topic for a given action and see what you need to provide for the SOAP API. You can also see what the XML response looks like.

| API Resources and Links                                 | Description   |
|---|---|
| <a href="#">Marketo SOAP API introduction and links</a> | Link to this document online  |
| <a href="#">Current WSDL (2011-09-24)</a>               | Location of the current WSDL  |
| <a href="#">Java implementation</a>                     | JAR file with Java implementation for API   |
| <a href="#">PHP client</a>                              | PHP file with example client for API  |
| <a href="#">User registration check</a>                 | Example showing a check for user registration via the API                         |
| <a href="#">Configuring Your SOAP API Settings</a>      | How to get the SOAP API endpoint, user ID, and encryption key; also covered below |

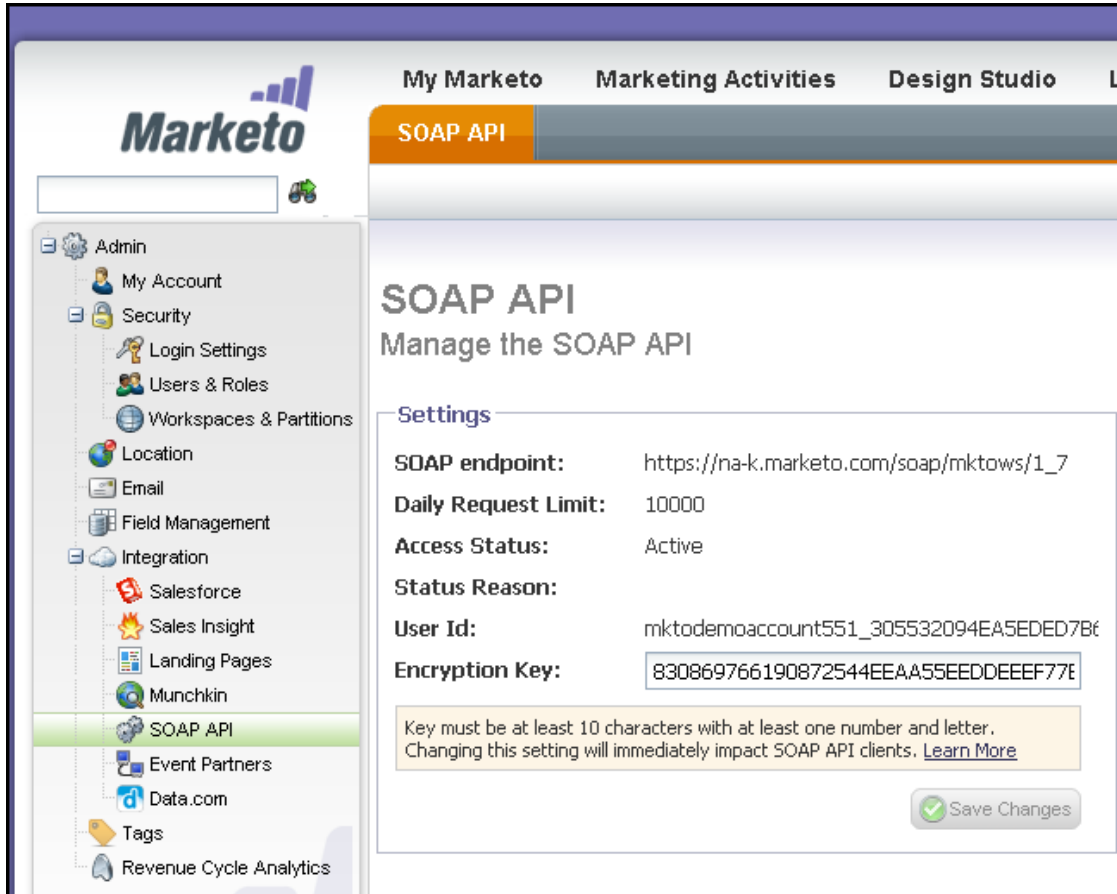
For questions about the SOAP API that are beyond the scope of this document, please contact Marketo support.

**Important:** You must get your SOAP API endpoint, user ID, and encryption key via the SOAP API settings in the Admin section of the application, as described in the next section. This information is also available in the Knowledge Base article, [Configuring the SOAP API](#).

## Configuring Your SOAP API Settings

If you want to use the Marketo SOAP API, follow these steps to get the settings for using it.

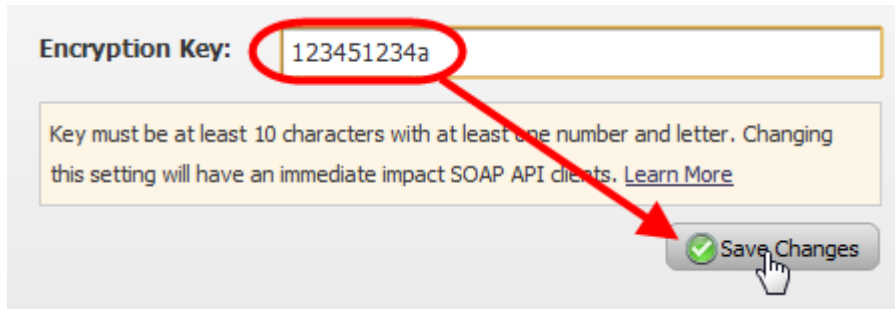
Go to the Admin section of Marketo and open the SOAP API panel. (You must have Administrator access in your Marketo account to access the page.) You'll see the following:



| Field               | Description   |
|---------------------|---|
| SOAP endpoint       | Make your SOAP calls to this URL                              |
| Daily Request Limit | Maximum number of SOAP calls you can make in a 24-hour period |
| Access Status       | <b>Active</b> if everything is ok; otherwise, contact support |
| Status Reason       | Details about your SOAP access status                         |
| User ID             | ID for authenticating your SOAP calls                         |
| Encryption Key      | Key for authenticating your SOAP calls                        |

## Setting your SOAP API encryption key

To use the SOAP API, you must enter an encryption key. This key is used to authenticate you when you make your SOAP calls. Fill out the Encryption Key field, then click **Save Changes**.

A screenshot of a web interface for setting a SOAP API encryption key. It features a text input field labeled "Encryption Key:" containing the value "123451234a". A red circle highlights the input field, and a red arrow points from it to a "Save Changes" button. Below the input field is a yellow warning box with the text: "Key must be at least 10 characters with at least one number and letter. Changing this setting will have an immediate impact SOAP API clients. [Learn More](#)". The "Save Changes" button is a grey button with a green checkmark icon and a mouse cursor hovering over it.

## Protocol

The Marketo API uses the SOAP protocol. SOAP is specifically designed to integrated distributed systems, and has these advantages:

- Well-defined standards
- Precise definition of the service contract
- Multiple options for security
- Well defined error handling
- Strong tools support.

Although the initial API is SOAP based, the infrastructure allows for future implementation of other protocols or models, such as REST.

The Marketo WSDL contains the service and schema definition. Different instances of Marketo software use different SOAP endpoints. Please retrieve your SOAP endpoint from your Marketo instance.

You can setup and manage your API credentials in the **Admin** section of your Marketo account, under the item called **SOAP API**, as shown in the figure above. Send email to [Marketo Support](#) if you require help in configuring your WSDL endpoint.

**IMPORTANT: Use the SOAP endpoint given in the Admin area of your Marketo instance.**

Appendices in this document support use of the API:

- Appendix A describes the XSD data types used in API requests and responses.
- Appendix B describes the error response returned by APIs and includes a set of standardized error codes that can be used to take corrective action.
- Appendix C describes some common use cases for integrating with Marketo and how the Marketo API can be used to fulfill those use cases.
- Appendix D provides an example of using the SOAP API from Java.
- Appendix E shows an example PHP client created using the SOAP API.
- Appendix F provides an example of checking for user registration using the SOAP API.
- Appendix G provides sample XML.

## Security

Marketo API security uses a simple yet highly secure model, based on HMAC-SHA1<sup>1</sup> signatures with messages transmitted over HTTPS. This model has been proven in industry use and it can be used with any integration protocol or model, such as SOAP or REST. A key advantage of this model is that it provides stateless authentication.

HMAC-SHA1 signatures require the following:

- A User ID (also called Access Key) that is transmitted with the service request
- A Signature that is calculated using a shared secret-key and message content and is transmitted with the service request
- A shared secret-key (also called Encryption Key) that is not transmitted with the service request.

Contact Marketo Support to get this information for your account.

## Signature Algorithm

The client will calculate the HMAC-SHA1 signature using the shared secret-key and part of the request message content. The client must include a SOAP header, **AuthenticationHeaderInfo**, to pass authentication information with the SOAP message. The following pseudo code demonstrates the algorithm:

```
// Request timestamp: a timestamp string in W3C WSDL date format
stringToEncrypt = requestTimestamp + clientAccessID;

signatureBytes = Encryter.encrypt('SHA1', secretKey, stringToEncrypt);

signature = toLowerCase( hexEncode(signatureBytes) );

authHeader = "<ns1:AuthenticationHeader>" +
    "<mktowsUserId>" + clientAccessID + "</mktowsUserId>" +
    "<requestSignature>" + signature + "</requestSignature>" +
    "<requestTimestamp>" + requestTimestamp + "</requestTimestamp>" +
    "</ns1:AuthenticationHeader>";
```

## Example Signature

|                   |  |
|-------------------|--|
| Access key        | bigcorp1_461839624B16E06BA2D663                          |
| Secret Key        | 899756834129871744AAEE88DDCC77CDEEDEC1AAAD66             |
| Request timestamp | 2010-04-09T14:04:54-07:00                                |
| Encryption input  | 2010-04-09T14:04:54-07:00bigcorp1_461839624B16E06BA2D663 |
| Signature         | ffbff4d4bef354807481e66dc7540f7890523a87                 |

SHA1 encryption libraries are readily available for a variety of development platforms. The server will perform the same calculation and compare it with the signature contained in the

---

<sup>1</sup> Hashed Message Authentication Code (HMAC), Secure Hash Algorithm 1 (SHA1)

request authentication header. If they are equal, the request is granted. Repeated authentication failures will disable API access and raise an alarm.

#### AuthenticationHeaderInfo

| Element Name     | XSD Type | Notes                |
|------------------|----------|----------------------|
| mktowsUserId     | string   | Client access ID     |
| requestSignature | string   | Calculated signature |
| requestTimestamp | dateTime | Request timestamp    |

Marketo API infrastructure design will permit future enhancements to incorporate other security models, such as WS-Security.



# **Definitions**

## **Stream Position**

Stream position elements contain a position reference for one or more logical streams of time-sequenced data. The position reference may be an approximate external specification like a timestamp, or an exact, but opaque internal specification of position returned by an operation. However, the value should be treated as opaque. We recommend that you not attempt to interpret nor modify it, it but instead treat it as a token.

Stream positions may be defined as a complex, multi-element type, or may be a string. Examples of streams are:

1. Sequence of new leads by creation time.
2. Sequence of activity history records for a single lead.
3. Sequence of all activity log records (possibly filtered by activity type).

The StreamPosition is used to retrieve data in batches, and allows the caller to page through the result. The StreamPosition passed into the API is the value of the StreamPosition returned in the previous response.

## **Success and Result Elements**

Result types are multi-element complex types and occur inside of an API response. Result types are always nested in a (single element) success type wrapper. This extra nesting is needed to support other systems that cannot handle a response with multiple top level elements. One such platform is APEX from Salesforce.com.

## **Lead Key**

A Lead Key is a type and value pair that identifies a lead or a lead owner. This table is a sample of Lead Key types and corresponding values. Refer to the type enumeration in the WSDL for a full list.

**IMPORTANT:** Not all Lead Key Types are valid for all operations. Refer to each API for acceptable types.

| Lead Key Type | Example<br>Lead Key Value                                   | Notes   |
|---------------|---|---|
| IDNUM         | 64  | The Marketo lead ID.                                    |
| COOKIE        | id:561-HYG-937&token:_mch-marketeto.com-1258067434006-50277 | This is the value generated by the Munchkin JavaScript. |
| EMAIL         | rufus@marketo.com   | An email address can map to more than one lead.         |
| SFDCLEADID    | 0038000000U21EHAAZ  |   |

## Context Header

A Context Header is a special SOAP header that is added to an API to limit the scope of data that will be affected or accessed by an API. The header can specify the workspace that is the target of the operation.

Refer to the Marketo Community Portal Knowledge Base to see how workspaces are used in Marketo. Refer to **MktowsContextHeaderInfo** in **Appendix A** for the structure of this header. Note that Marketo workspaces must be enabled.

**IMPORTANT:** Not all APIs support the Context Header. Currently, only the **syncLead** API supports this header. Refer to the API description to see how it is used by it.

## Basic Operations

These operations are used to retrieve basic information about leads and lead activity in the Marketo System. Several types of keys are supported and are enumerated the WSDL.

### getLead

This function retrieves a single lead record from Marketo, with all field values for the built-in and custom fields for a lead identified by the provided key (LeadKey). Some of the input parameters for retrieving a lead include: Marketo ID, email address, Marketo cookie ID. If the lead exists based on the input parameters, the lead record attributes will be returned in the result.

One thing to note is that for lead attributes that are of data type string and are empty, they will not be returned at all as a part of the response.

#### GetLeadRequest

| Element Name  | XSD Type      | Notes |
|---------------|---------------|-------|
| paramsGetLead | ParamsGetLead |       |

#### ParamsGetLead

| Element Name | XSD Type | Notes  |
|--------------|----------|--|
| leadKey      | LeadKey  | IDNUM, EMAIL, COOKIE, SFDCCONTACTID, SFDCCLEADID |

#### GetLeadResponse

| Element Name | XSD Type       | Notes |
|--------------|----------------|-------|
| success      | SuccessGetLead |       |

#### SuccessGetLead

| Element Name | XSD Type      | Notes |
|--------------|---------------|-------|
| result       | ResultGetLead |       |

#### ResultGetLead

| Element Name   | XSD Type          | Notes  |
|----------------|-------------------|--|
| count          | int               | Number of items in <b>leadRecordList</b> element |
| leadRecordList | ArrayOfLeadRecord |  |

## Errors and warnings

- Error – Key Not Found
- Error – Invalid parameter
- Error – Unexpected error
- Warning – Maximum batch size exceeded [all leads matching email address exceed the maximum batch size for the response; only the first matching leads returned]

## getMultipleLeads

This operation retrieves the field values for all the built-in and custom fields for a group of leads modified or created since a specified time. Some of the input parameters for retrieving multiple leads include: last updated at, list of lead attributes you wish to be returned in the result.

If only a subset of the lead fields are required, then the **includeAttributes** parameter should be used to specify the desired fields. Limiting the lead fields returned can improve the response time of the API.

Each getMultipleLeads function call can only return batches of 1000 leads, maximum. If this call needs to return more than 1000 leads, the result will return a start position, which should be used in subsequent calls to retrieve the next set of leads. Logic should be implemented to check for this value to see if there are more leads to retrieve at this given time.

### GetMultipleLeadsRequest

| Element Name           | XSD Type               | Notes |
|------------------------|------------------------|-------|
| paramsGetMultipleLeads | ParamsGetMultipleLeads |       |

### ParamsGetMultipleLeads

| Element Name      | XSD Type      | Notes   |
|-------------------|---------------|---|
| lastUpdatedAt     | dateTime      | Time lead was last updated or created   |
| streamPosition    | string        | Used for paging. Value is filled in by the API and must be returned on subsequent calls when paging. When present, the <b>lastUpdateAt</b> value is ignored |
| batchSize         | int           | Maximum number of records to be returned. System will limit to 1000 or <b>batchSize</b> , whichever is less   |
| includeAttributes | ArrayOfString | [Since v1.4] Used to limit the lead fields that are returned by the API   |

### GetMultipleLeadsResponse

| Element Name | XSD Type                | Notes |
|--------------|-------------------------|-------|
| Success      | SuccessGetMultipleLeads |       |

### SuccessGetMultipleLeads

| Element Name | XSD Type               | Notes |
|--------------|------------------------|-------|
| Result       | ResultGetMultipleLeads |       |

### ResultGetMultipleLeads

| Element Name     | XSD Type          | Notes  |
|------------------|-------------------|--|
| returnCount      | int               | Number of items in <b>leadRecordList</b> element                   |
| remainingCount   | int               | Number of item not returned  |
| newStartPosition | string            | Filled in by API call. Value should not be modified or interpreted |
| leadRecordList   | ArrayOfLeadRecord |  |

### Errors and warnings

- Error – Invalid parameter
- Error – Unexpected error
- Warning – Maximum batch size exceeded

## getLeadActivity

This function retrieves the history of activity/event records for a single lead identified by the provided key. Some of the input parameters for identifying which lead record to retrieve activity records for include: Marketo ID, email address, cookie ID.

This function also has an input parameter identifying which activity types you wish to be returned in the result. If you want all activity types, a blank value needs to be passed (must have a parameter passed). For more than one activity type, pass in a list of activity types. The complete list of activity types can be found in the API WSDL. Some example activity types are 'VisitWebPage', 'FillOutForm', and 'ClickLink'.

### GetLeadRequestActivity

| Element Name          | XSD Type              | Notes |
|-----------------------|-----------------------|-------|
| paramsGetLeadActivity | ParamsGetLeadActivity |       |

### ParamsGetLeadActivity

| Element Name   | XSD Type           | Notes   |
|----------------|--------------------|---|
| leadKey        | LeadKey            | IDNUM, EMAIL, COOKIE, LEADOWNEREMAIL, SFDACCOUNTID, SFDCONTACTID, SFDLEADID, SFDLEADOWNERID, SFDOPPTYID     |
| activityFilter | ActivityTypeFilter | Activity types can be found in the WSDL.  |
| startPopsition | StreamPosition     |   |
| batchSize      | int                | Maximum number of records to be returned. System will limit to 100 or <b>batchSize</b> , whichever is less. |

### GetLeadActivityResult

| Element Name           | XSD Type               | Notes |
|------------------------|------------------------|-------|
| successGetLeadActivity | SuccessGetLeadActivity |       |

### SuccessGetLeadActivity

| Element Name     | XSD Type         | Notes |
|------------------|------------------|-------|
| leadActivityList | LeadActivityList |       |

### LeadActivityList

| Element Name       | XSD Type              | Notes  |
|--------------------|-----------------------|--|
| returnCount        | int                   | Number of items in <b>activityRecordList</b> element             |
| remainingCount     | int                   | Number of items that can be retrieved with additional API calls. |
| newStartPopsition  | StreamPosition        | Can be passed to subsequent API call to do paging                |
| activityRecordList | ArrayOfActivityRecord |  |

### Errors and warnings

- Error - Key Not Found
- Error – Invalid parameter
- Error – Unexpected error

## Lead Synchronization Operations

These operations allow an external system to poll synchronize lead data between Marketo and another system.

### getLeadChanges

This operation checks for changes that have occurred to leads and lead history in the Marketo system. The changes reported include new leads created, leads updated with the changed fields, and new activity records for leads. The result contains activities that caused the change. This API is similar to the getLeadActivity API, except that it operates on multiple leads instead of a single lead.

This function also has an input parameter identifying which activity types you wish to be returned in the result. If you want all activity types, a blank value needs to be passed (must have a parameter passed). For more than one activity type, pass in a list of activity types. The complete list of activity types can be found in the API WSDL. Some example activity types are: 'VisitWebPage', 'FillOutForm', and 'ClickLink'.

### RequestGetLeadChanges

| Element Name         | XSD Type             | Notes |
|----------------------|----------------------|-------|
| paramsGetLeadChanges | ParamsGetLeadChanges |       |

### ParamsGetLeadChanges

| Element Name   | XSD Type           | Notes   |
|----------------|--------------------|---|
| activityFilter | ActivityTypeFilter | Activity types can be found in the WSDL.  |
| startPopsition | StreamPosition     |   |
| batchSize      | int                | Maximum number of records to be returned. System will limit to 100 or <b>batchSize</b> , whichever is less. |

### ResponseGetLeadChanges

| Element Name         | XSD Type              | Notes |
|----------------------|-----------------------|-------|
| succesGetLeadChanges | successGetLeadChanges |       |

## ResultGetLeadChanges

| Element Name         | XSD Type                | Notes  |
|----------------------|-------------------------|--|
| returnCount          | int                     | Number of items in <b>leadChangeRecordList</b> element |
| remainingCount       | int                     | Number of items not returned                           |
| newStartPosition     | StreamPosition          |  |
| leadChangeRecordList | ArrayOfLeadChangeRecord |  |

## Errors and warnings

- Error - Key Not Found
- Error – Invalid parameter
- Error – Unexpected error

## syncLead

This function requests an insert or update (upsert) operation for a lead record. When updating an existing lead, the lead can be identified one of the follow:

- Marketo ID
- Foreign system ID
- Marketo Cookie
- Email

If any of these attributes are set as a part of the input parameter and if there is a matching lead in Marketo with these values, the existing lead record will be updated instead of created.

Passing the Marketo cookie id as an input parameter helps in associating the new lead with existing anonymous activity records.

Except for **Email**, all of these identifiers are treated as unique keys. The Marketo Id takes precedence over all other keys. If both **ForeignSysPersonId** and the Marketo **Id** are present in the lead record, then the Marketo **Id** will take precedence and the **ForeignSysPersonId** will be updated for that lead. If the only **ForeignSysPersonId** is given, then it will be used as a unique identifier. Optionally, a Context Header can be specified to name the target workspace. Refer to the section titled “Definitions” for an explanation of the Context Header.

When Marketo workspaces are enabled and the header is used, the following rules are applied:

- New leads are created in the primary partition of the named workspace.
- Leads matched by the Marketo Lead ID, a foreign system ID, or a Marketo cookie, must exist in the primary partition of the named workspace, otherwise an error will be returned.
- If an existing lead is matched by email, the named workspace is ignored and the lead is updated in its current partition.



When Marketo workspaces are enabled and the header is NOT used, the following rules are applied:

- New leads are created in the primary partition of the “Default” workspace.
- Existing leads are updated in their current partition.

If Marketo workspaces are NOT enabled, the target workspace MUST be the “Default” workspace. It is not necessary to pass the header.

### SyncLeadRequest

| Element Name   | XSD Type       | Notes |
|----------------|----------------|-------|
| paramsSyncLead | ParamsSyncLead |       |

### ParamsSyncLead

| Element Name  | XSD Type   | Notes   |
|---------------|------------|---|
| leadRecord    | LeadRecord |   |
| returnLead    | boolean    | If set to true, a complete lead record will be returned |
| marketoCookie | string     | A Marketo cookie  |

### SyncLeadResponse

| Element Name    | XSD Type        | Notes |
|-----------------|-----------------|-------|
| successSyncLead | SuccessSyncLead |       |

### SuccessSyncLead

| Element Name | XSD Type       | Notes |
|--------------|----------------|-------|
| result       | ResultSyncLead |       |

### ResultSyncLead

| Element Name | XSD Type       | Notes                       |
|--------------|----------------|-----------------------------|
| leadId       | int            | System ID of affected lead  |
| syncStatus   | LeadSyncStatus | See enumeration in WSDL     |
| leadRecord   | LeadRecord     | Only if request in API call |

### Errors and warnings

- Error – Key Not Found
- Error – Invalid parameter
- Error – Unexpected error

## syncMultipleLeads

This function requests an insert or update (upsert) operation for multiple lead records. If both **ForeignSysPersonId** and the Marketo lead **Id** are present in the lead record, then the Marketo lead **Id** will take precedence and the **ForeignSysPersonId** will be updated. If the only **ForeignSysPersonId** is given, then it will be used as a unique identifier.

You can turn off the dedupe feature with this call. If **dedupEnabled** is set to **true** and no other unique identifier is given (**foreignSysPersonId** or Marketo lead **Id**), then the lead record will be de-duplicated using the email address. Passing in 'false' creates duplicates within Marketo.

### SyncMultipleLeadsRequest

| Element Name            | XSD Type                | Notes |
|-------------------------|-------------------------|-------|
| paramsSyncMultipleLeads | ParamsSyncMultipleLeads |       |

### ParamsSyncMultipleLeads

| Element Name   | XSD Type          | Notes  |
|----------------|-------------------|--|
| leadRecordList | ArrayOfLeadRecord |  |
| dedupEnabled   | boolean           | If set to true, de-duplicate lead record on email address. Default <b>true</b> |

### SyncMultipleLeadsResponse

| Element Name             | XSD Type                 | Notes |
|--------------------------|--------------------------|-------|
| successSyncMultipleLeads | SuccessSyncMultipleLeads |       |

### SuccessSyncMultipleLeads

| Element Name | XSD Type                | Notes |
|--------------|-------------------------|-------|
| result       | ResultSyncMultipleLeads |       |

### ResultSyncMultipleLeads

| Element Name   | XSD Type          | Notes   |
|----------------|-------------------|---|
| syncStatusList | ArrayOfSyncStatus | Contains the operation outcome for each lead. The number of items equals the number of lead records in the request, and is mapped 1-to-1 with the lead record list. |

### Errors and warnings

- Error – Key Not Found
- Error – Invalid parameter
- Error – Unexpected errors

## Advanced Operations

These operations allow access to Marketo marketing and lead nurturing operations.

### getCampaignsForSource

This function returns a list of available Marketo campaigns that can be used as input parameters into the requestCampaign function (below). Campaigns are categorized by the source of the request, which are enumerated in the WSDL.

For additional information on using this API, refer to the use case “Add Lead to Campaign” in Appendix C and to the Marketo Knowledge Base article titled [Request Campaign and Campaign is Requested](#).

#### GetCampaignsForSourceRequest

| Element Name                | XSD Type                    | Notes |
|-----------------------------|-----------------------------|-------|
| paramsGetCampaignsForSource | ParamsGetCampaignsForSource |       |

#### ParamsGetCampagnForSource

| Element Name | XSD Type          | Notes                       |
|--------------|-------------------|-----------------------------|
| source       | ReqCampSourceType | Enumeration defined in WSDL |
| name         | string            | Name of campaign            |
| exactName    | boolean           |                             |

#### GetCampaignsForSourceResponse

| Element Name | XSD Type                    | Notes |
|--------------|-----------------------------|-------|
| success      | ResultGetCampaignsForSource |       |

#### SuccessGetCampaignsForSource

| Element Name | XSD Type                    | Notes |
|--------------|-----------------------------|-------|
| result       | ResultGetCampaignsForSource |       |

#### ResultGetCampaignsForSource

| Element Name       | XSD Type              | Notes  |
|--------------------|-----------------------|--|
| returnCount        | int                   | Number of items in <b>campaignRecordList</b> element |
| campaignRecordList | ArrayOfCampaignRecord |  |

## Errors and warnings

- Error – Invalid parameter
- Error – Unexpected error

## listOperation

This operation is used to perform operations on a static list as a list of leads. The available operations are adding a lead to a static list, removing a lead from a static list, or checking if a lead is a member of a static list. The input parameter for identifying a lead when adding, removing and checking is the Marketo ID.

The operation can be invoked in a strict or non-strict mode. Strict means that if the operation is not performed, or fails, an exception is returned. Non-strict means that a Boolean status is returned to indicate the outcome of the operation.

### ListOperationRequest

| Element Name        | XSD Type            | Notes |
|---------------------|---------------------|-------|
| paramsListOperation | ParamsListOperation |       |

### ParamsListOperation

| Element Name   | XSD Type          | Notes  |
|----------------|-------------------|--|
| listOperation  | ListOperationType | Enumeration defined in WSDL  |
| listKey        | ListKey           |  |
| listMemberList | ArrayOfLeadKey    | IDNUM, SFDCCONTACTID, SFDCCLEADID  |
| strict         | boolean           | If true, a duplicate add or remove operation will return a failure status for the lead |

### ListOperationResponse

| Element Name | XSD Type             | Notes |
|--------------|----------------------|-------|
| success      | SuccessListOperation |       |

### SuccessListOperation

| Element Name | XSD Type            | Notes |
|--------------|---------------------|-------|
| result       | ResultListOperation |       |

### ResultListOperation

| Element Name | XSD Type          | Notes  |
|--------------|-------------------|--|
| success      | boolean           |  |
| statusList   | ArrayOfLeadStatus | Returned if operation failed on one more leads |

### Errors and warnings

- Error – List not found
- Error – Invalid parameter
- Error – Unexpected error

## requestCampaign

Use this call to add an existing Marketo lead to an existing Marketo campaign. The input parameter for identifying which lead to add is the Marketo ID. The campaign must be configured for invocation through the API.

For additional information, refer to the use case “Add Lead to Campaign” in Appendix C, and to the Marketo Knowledge Base article titled [Request Campaign and Campaign is Requested](#).

### RequestCampaignRequest

| Element Name          | XSD Type              | Notes |
|-----------------------|-----------------------|-------|
| paramsRequestCampaign | ParamsRequestCampaign |       |

### ParamsRequestCampaign

| Element Name | XSD Type          | Notes                            |
|--------------|-------------------|----------------------------------|
| source       | ReqCampSourceType | Enumeration defined in WSDL      |
| campaigned   | int               | Marketo system ID                |
| leadList     | ArrayOfLeadKey    | IDNUM, SFDCCONTACTID, SFDCLEADID |

### RequestCampaignResponse

| Element Name | XSD Type               | Notes |
|--------------|------------------------|-------|
| success      | SuccessRequestCampaign |       |

### SuccessRequestCampaign

| Element Name | XSD Type              | Notes |
|--------------|-----------------------|-------|
| result       | ResultRequestCampaign |       |

### ResultRequestCampaign

| Element Name | XSD Type | Notes |
|--------------|----------|-------|
| success      | boolean  |       |

### Errors and warnings

- Error – Campaign not found
- Error – Lead not found
- Error – Invalid parameter
- Error – Unexpected error

## Marketo Object Operations

MObjects are native objects within Marketo.

MObjects serve as a generic API structure to handle a wide range of API calls. The objects currently supported for at least some calls are: Activity, Opportunity, OpportunityPersonRole, and LeadRecord.

For the initial implementation of MObjects, four calls are supported:

- syncMObjects – Opportunity and OpportunityPersonRole object types only
- getMObjects – Opportunity and OpportunityPersonRole object types only
- deleteMObjects – Opportunity and OpportunityPersonRole object types only
- describeMObject – all four object types.

### **MObjects and Opportunities**

In the case of opportunities, MObjects allow data to be accessed from within Marketo, rather than only from a CRM such as Salesforce.com. MObjects also allow opportunity data from non-Salesforce sources to be stored in Marketo.

Opportunities maintained through the SOAP API behave just like opportunities that come into Marketo through Salesforce. This includes filter and trigger capabilities, as well as capabilities of the Revenue Cycle Analyzer (RCA) product that are driven by opportunity information.

These changes allow Marketo users who are not Salesforce customers to create and maintain opportunity information from other sources. For Marketo users who are Salesforce customers, the new capabilities allow other sources of opportunity information to be added to the opportunities available through Salesforce.

In the future, it's likely that the use of MObject will be expanded to support the use of other data objects within Marketo, with an object key used to identify the object to perform a given call on.

Sample XML and metadata for the [Opportunity](#) and [OpportunityPersonRole](#) objects can be found in Appendix G.

## Structure of MObjects

MObjects consist of:

- A small set of fixed attributes that are common to all MObjects:
  - Required type
  - Optional externalKey
  - Read only id, createdAt, updatedAt
- A list of one or more object specific attributes, as name/value pairs, some of which may be required. For example, name on Opportunity.
- A list of associated object references, as object-name plus
  - Marketo ID or,
  - External-key, as an attribute-name/attribute-value pair.

## External-Keys

External-keys are **custom** fields defined on Marketo objects, such as Activity or Opportunity. The name is the field name and value is the field value, generated in an external system.

**IMPORTANT: External-keys are custom fields. You will need to create a custom field in your Marketo software or in your CRM software to match the external key.**

Marketo does not enforce a unique constraint on these values. It is the responsibility of the API user to ensure that the values are unique.

Should a duplicate occur, Marketo will use the most recently added object, and delete any previously added objects with the same value in the external-key field.

This is similar to the behavior for the Email Address standard field.

**NOTE: There are four supported types of MObjects: LeadRecord, Activity, Opportunity, and OpportunityPersonRole.**

## getMObjects

Retrieves one or more MObjects using a combination of criteria consisting of:

- Zero or one unique ID, either the Marketo ID or external ID
- Zero or more attribute filters as name/value/comparison trios
- Zero or more associated object filters as object name/ID pairs

Returns a list of matching MObjects, all of a single type, up to 100 in a batch, and a stream position token for retrieving successive batches.

Sample SOAP request fragments for retrieving MObjects [by ID](#), [by association](#), and [by criteria](#) can be found in Appendix G.

## Associations

| MObject               | Valid Associations for getMObjects |
|-----------------------|------------------------------------|
| Opportunity           | Company, Lead                      |
| OpportunityPersonRole | Lead, Opportunity                  |

## GetMObjectsRequest

| Element Name      | XSD Type          | Notes |
|-------------------|-------------------|-------|
| paramsGetMObjects | ParamsGetMObjects |       |

## ParamsGetMObjects

| Element Name        | XSD Type               | Notes  |
|---------------------|------------------------|--|
| type                | string                 |  |
| id                  | int                    |  |
| externalKey         | Attrib                 |  |
| mObjCriteriaList    | ArrayOfMObjCriteria    |  |
| mObjAssociationList | ArrayOfMObjAssociation |  |
| streamPosition      | string                 | Used for paging. Value is filled in by API and must be returned on subsequent calls when paging. |

## GetMObjectsResponse

| Element Name       | XSD Type           | Notes |
|--------------------|--------------------|-------|
| successGetMObjects | successGetMObjects |       |

## SuccessGetMObjects

| Element Name | XSD Type          | Notes |
|--------------|-------------------|-------|
| result       | ResultGetMObjects |       |

## ResultGetMObjects

| Element Name      | XSD Type       | Notes  |
|-------------------|----------------|--|
| returnCount       | int            |  |
| remainingCount    | int            | Number of item not returned  |
| newStreamPosition | string         | Filled in by API call. Value should not be modified or interpreted |
| mObjectList       | ArrayOfMObject |  |



## syncMObjects

Accepts an array of MObjects to be created or updated, up to a maximum to 100 per call, and returns the outcome (status) of the operation (CREATED, UPDATED, FAILED, UNCHANGED, SKIPPED) and the Marketo IDs of the MObject(s). The API can be called in one of three operation modes:

- INSERT - only insert new objects, skip existing objects;
- UPDATE - only update existing objects, skip new objects; or
- UPSERT - insert new objects and update existing objects.

In a single API call some updates may succeed and some may fail. An error message will be returned for each failure.

A sample SOAP request fragment for [syncMObjects](#) can be found in Appendix G.

### Associations

| MObject               | Valid Associations for syncMObjects |
|-----------------------|-------------------------------------|
| Opportunity           | Company                             |
| OpportunityPersonRole | Lead, Opportunity                   |

### SyncMObjectsRequest

| Element Name       | XSD Type           | Notes |
|--------------------|--------------------|-------|
| paramsSyncMObjects | paramsSyncMObjects |       |

### ParamsSyncMObjects

| Element Name | XSD Type          | Notes                   |
|--------------|-------------------|-------------------------|
| mObjectList  | ArrayOfMObject    |                         |
| operation    | SyncOperationEnum | See enumeration in WSDL |

### SyncMObjectsResponse

| Element Name        | XSD Type            | Notes |
|---------------------|---------------------|-------|
| successSyncMObjects | successSyncMObjects |       |

### SuccessSyncMObjects

| Element Name | XSD Type           | Notes |
|--------------|--------------------|-------|
| result       | ResultSyncMObjects |       |

## ResultSyncMObjects

| Element Name   | XSD Type          | Notes |
|----------------|-------------------|-------|
| mObjStatusList | ArrayOfMObjStatus |       |

## deleteMObjects

Deletes one or more MObjects and returns the outcome of the operation (DELETED, UNCHANGED, FAILED). Each MObject must be identified by a Marketo ID or and external-key.

A sample SOAP request fragment for [syncMObjects](#) can be found in Appendix G.

### DeleteMObjectsRequest

| Element Name         | XSD Type             | Notes |
|----------------------|----------------------|-------|
| paramsDeleteMObjects | paramsDeleteMObjects |       |

### ParamsDeleteMObjects

| Element Name | XSD Type       | Notes |
|--------------|----------------|-------|
| mObjectList  | ArrayOfMObject |       |

### DeleteMObjectsResponse

| Element Name          | XSD Type              | Notes |
|-----------------------|-----------------------|-------|
| successDeleteMObjects | successDeleteMObjects |       |

### SuccessDeleteMObjects

| Element Name | XSD Type             | Notes |
|--------------|----------------------|-------|
| result       | ResultDeleteMObjects |       |

### ResultDeleteMObjects

| Element Name   | XSD Type          | Notes |
|----------------|-------------------|-------|
| mObjStatusList | ArrayOfMObjStatus |       |

## listMObjects

This function returns the names of Marketo objects that can be used as input into the describeMObjects function (below) for schema discovery operations.

### ListMObjectsRequest

| Element Name       | XSD Type           | Notes |
|--------------------|--------------------|-------|
| paramsListMObjects | ParamsListMObjects |       |

### ParamsListMObjects

| Element Name | XSD Type | Notes           |
|--------------|----------|-----------------|
|              |          | No sub-elements |

### ListMObjectsResponse

| Element Name | XSD Type            | Notes |
|--------------|---------------------|-------|
| success      | SuccessListMObjects |       |

### SuccessListMObjects

| Element Name | XSD Type           | Notes |
|--------------|--------------------|-------|
| result       | ResultListMObjects |       |

### ResultListMObjects

| Element Name | XSD Type | Notes                 |
|--------------|----------|-----------------------|
| objects      | string   | 1 or more in sequence |

## describeMObjects

This operation returns the metadata for an MObject. It takes in the Marketo object as input and returns the field attributes that are associated to that object.

There are three types of MObjects: Standard, Custom or Virtual. Standard and Custom MObjects represent distinct entities, such as Lead or Company. Virtual Objects, such as LeadRecord, comprise fields from one or more objects. Virtual Objects are convenience objects used within the API, but do not exist within the Marketo application.

### DescribeMObjectRequest

| Element Name          | XSD Type              | Notes |
|-----------------------|-----------------------|-------|
| paramsDescribeMObject | ParamsDescribeMObject |       |

### ParamsDescribeMObject

| Element Name | XSD Type | Notes |
|--------------|----------|-------|
| objectName   | string   |       |

### DescribeMObjectResponse

| Element Name | XSD Type               | Notes |
|--------------|------------------------|-------|
| success      | SuccessDescribeMObject |       |

### SuccessListMObjects

| Element Name | XSD Type              | Notes |
|--------------|-----------------------|-------|
| result       | ResultRequestCampaign |       |

### ResultListMObjects

| Element Name | XSD Type        | Notes |
|--------------|-----------------|-------|
| metadata     | MObjectMetadata |       |

### Field Attribute Types

| Type     |
|----------|
| integer  |
| string   |
| dateTime |
| date     |
| boolean  |
| float    |
| email    |
| phone    |
| url      |
| currency |

## Error Codes

These are the error codes that are specific to MObject APIs. The Marketo SOAP API can return other error codes as well.

A sample SOAP [error response](#) can be found in Appendix G.

| Element Name | XSD Type                     | Notes  |
|--------------|------------------------------|--|
| 21100        | No MObject type              | An MObject did not specify the type                                    |
| 21101        | MObject type unknown         | An Object specified an unsupported MObject type                        |
| 21102        | MObject limit exceeded       | Too many MObjects were passed in the API call                          |
| 21103        | MObject association error    | An incorrect MObject association was given                             |
| 21104        | Bad MObject external key     | The external key name does not exist for the MObject                   |
| 21105        | MObject association required | A required MObject association was missing                             |
| 21106        | MObject field error          | An MObject field is not correct or is missing                          |
| 21107        | MObject poor query           | A getMObjects could not be satisfied as it would generate a slow query |
| 21108        | MObject not found            | An MObject could not be found with the given ID or external key        |

## Appendix A – Data Types

This appendix describes the XSD data types used in the API requests and responses.

### ActivityRecord

An occurrence of a lead activity

| Element Name       | XSD Type         | Notes  |
|--------------------|------------------|--|
| id                 | int              | Marketo system ID for an activity              |
| activityDateTime   | dateTime         |  |
| activityType       | string           | See enumeration in WSDL                        |
| mktgAssetName      | string           | Name of a marketing asset in Marketo           |
| activityAttributes | ArrayOfAttribute | List of attributes specific to an ActivityType |
| campaign           | string           | Name of campaign                               |
| foreignSysId       | string           | ID from another system, such as Salesforce.com |
| personName         | string           |  |
| orgName            | string           | Company or organization                        |
| foreignOrgSysId    | string           |  |

### ArrayOfActivityRecord

| Element Name   | XSD Type       | Notes |
|----------------|----------------|-------|
| activityRecord | ActivityRecord |       |

### ActivityTypeFilter

Allows specifying a list of selected activity types to be requested or allows selecting all activity types without enumerating them. Only one of the elements includeTypes or excludeTypes can be specified.

| Element Name | XSD Type            | Notes |
|--------------|---------------------|-------|
| includeTypes | ArrayOfActivityType |       |
| excludeTypes | ArrayOfActivityType |       |

### ArrayOfActivityType

| Element Name   | XSD Type     | Notes                   |
|----------------|--------------|-------------------------|
| activityRecord | ActivityType | See enumeration in WSDL |

### ArrayOfAttrib

| Element Name | XSD Type | Notes |
|--------------|----------|-------|
| attrib       | Attrib   |       |

### Attrib

| Element Name | XSD Type | Notes |
|--------------|----------|-------|
| name         | string   |       |
| value        | string   |       |

### Attribute

A tag/value pair with an optional type. Used in places where a dynamic list of attributes is required.

| Element Name | XSD Type | Notes |
|--------------|----------|-------|
| attrName     | string   |       |
| attrType     | string   |       |
| attrValue    | string   |       |

### ArrayOfAttribute

| Element Name | XSD Type  | Notes |
|--------------|-----------|-------|
| attribute    | Attribute |       |

### CampaignRecord

| Element Name | XSD Type | Notes             |
|--------------|----------|-------------------|
| id           | int      | Marketo system ID |
| name         | string   |                   |
| description  | string   |                   |

### ArrayOfCampaignRecord

| Element Name   | XSD Type       | Notes |
|----------------|----------------|-------|
| campaignRecord | CampaignRecord |       |

### LeadChangeRecord

| Element Name       | XSD Type         | Notes  |
|--------------------|------------------|--|
| id                 | int              | Marketo system ID for an activity              |
| activityDateTime   | dateTime         |  |
| activityType       | string           | See enumeration in WSDL                        |
| mktgAssetName      | string           | Name of a marketing asset in Marketo           |
| activityAttributes | ArrayOfAttribute | List of attributes specific to an ActivityType |
| campaign           | string           | Name of campaign                               |
| mktPersonId        | int              | Marketo system ID                              |

### ArrayOfActivityRecord

| Element Name     | XSD Type         | Notes |
|------------------|------------------|-------|
| leadChangeRecord | LeadChangeRecord |       |

### LeadKey

Contains a reference to a lead by key including which key type is specified and the key value.

| Element Name | XSD Type   | Notes                   |
|--------------|------------|-------------------------|
| keyType      | LeadKeyRef | See enumeration in WSDL |
| keyValue     | string     |                         |

### LeadList

Contains a count and a list of LeadRecords. LeadRecord contains the field names and values for all the externally visible built-in lead fields as well as all custom lead fields.

| Element Name      | XSD Type           | Notes   |
|-------------------|--------------------|---|
| returnCount       | int                | Number of items in <b>leadRecordList</b> element                |
| remainingCount    | int                | Number of items that can be retrieved with additional API calls |
| newStartPopsition | LeadStreamPosition | Can be passed to subsequent API call to do paging               |
| leadRecordList    | ArrayOfLeadRecord  |   |



### LeadRecord

| Element Name       | XSD Type         | Notes  |
|--------------------|------------------|--|
| Id                 | int              | Marketo system ID for a lead   |
| Email              | string           | SMTP email address   |
| ForeignSysPersonId | string           | Unique person identifier generated outside of Marketo                                |
| ForeignSysType     | ForeignSysType   | See enumeration in WSDL. Must be specified when <b>ForeignSysPersonId</b> is present |
| leadAttributeList  | ArrayOfAttribute |  |

### ArrayOfLeadRecord

| Element Name | XSD Type   | Notes |
|--------------|------------|-------|
| leadRecord   | LeadRecord |       |

### LeadStatus

| Element Name | XSD Type | Notes |
|--------------|----------|-------|
| leadKey      | LeadKey  |       |
| status       | boolean  |       |

### ArrayOfLeadStatusRecord

| Element Name     | XSD Type         | Notes |
|------------------|------------------|-------|
| leadStatusRecord | LeadStatusRecord |       |

### MktowsContextHeader

| Element Name    | XSD Type | Notes              |
|-----------------|----------|--------------------|
| targetWorkspace | string   | Name of workspace. |

### MObjAssociation

| Element Name | XSD Type | Notes |
|--------------|----------|-------|
| mObjType     | string   |       |
| id           | int      |       |
| externalKey  | Attrib   |       |

### ArrayOfMObject

| Element Name | XSD Type | Notes |
|--------------|----------|-------|
| mObject      | MObject  |       |

### ArrayOfMObjCriteria

| Element Name | XSD Type     | Notes |
|--------------|--------------|-------|
| MObjCriteria | MObjCriteria |       |

### MObjCriteria

| Element Name | XSD Type       | Notes                   |
|--------------|----------------|-------------------------|
| attrName     | string         |                         |
| comparison   | ComparisonEnum | See enumeration in WSDL |
| attrValue    | anySimpleType  |                         |

### ArrayOfMObjAssociation

| Element Name    | XSD Type        | Notes |
|-----------------|-----------------|-------|
| mObjAssociation | mObjAssociation |       |

### MObject

| Element Name    | XSD Type               | Notes |
|-----------------|------------------------|-------|
| type            | string                 |       |
| id              | int                    |       |
| externalKey     | Attrib                 |       |
| createdAt       | dateTime               |       |
| updatedAt       | dateTime               |       |
| attribList      | ArrayOfAttrib          |       |
| associationList | ArrayOfMObjAssociation |       |

### ArrayOfMObjStatus

| Element Name | XSD Type   | Notes |
|--------------|------------|-------|
| mObjStatus   | MObjStatus |       |

### MObjStatus

| Element Name | XSD Type       | Notes                   |
|--------------|----------------|-------------------------|
| id           | int            |                         |
| externalKey  | Attrib         |                         |
| status       | MObjStatusEnum | See enumeration in WSDL |
| error        | string         |                         |

### MObjFieldMetadata

| Element Name    | XSD Type | Notes   |
|-----------------|----------|---|
| name            | string   | Field name  |
| description     | string   |   |
| displayName     | string   | UI render-able label for field                        |
| sourceObject    | string   | Source MObject of field contained in a Virtual Object |
| dataType        | string   | See list under describeMObject operation              |
| size            | int      |   |
| isReadOnly      | boolean  | Field cannot be written to                            |
| isUpdateBlocked | boolean  | Field cannot be updated                               |
| isName          | boolean  | Field is the readable name of object                  |
| isPrimaryKey    | boolean  | Field is primary key of object                        |
| isCustom        | boolean  |   |
| isDynamic       | boolean  | Field is contained in the dynamicFieldRef element     |
| dynamicFieldRef | string   | Name of field that holds dynamic attributes           |
| updatedAt       | dateTime | Timestamp of last modification                        |

### ArrayOfMObjFieldMetadata

| Element Name | XSD Type          | Notes |
|--------------|-------------------|-------|
| field        | MObjFieldMetadata |       |

### MObjectMetadata

| Element Name | XSD Type                 | Notes                          |
|--------------|--------------------------|--------------------------------|
| name         | string                   |                                |
| description  | string                   |                                |
| isCustom     | boolean                  | A user defined object          |
| isVirtual    | boolean                  | See describeObject API         |
| fieldList    | ArrayOfMObjFieldMetadata |                                |
| updatedAt    | dateTime                 | Timestamp of last modification |

### SyncStatus

Outcome of sync operation for one lead.

| Element Name | XSD Type       | Notes   |
|--------------|----------------|---|
| leadId       | int            | Marketo lead ID. <b>0</b> if new could not be created     |
| syncStatus   | LeadSyncStatus | See enumeration in WSDL                                   |
| error        | string         | Short error description if operation failed for this lead |

### ArrayOfSyncStatus

| Element Name | XSD Type   | Notes |
|--------------|------------|-------|
| syncStatus   | SyncStatus |       |

### StreamPosition

The starting point for retrieving lead activities.

| Element Name    | XSD Type | Notes  |
|-----------------|----------|--|
| latestCreatedAt | dateTime |  |
| oldestCreatedAt | dateTime |  |
| offset          | string   | Filled in by API call. Value should not be modified or interpreted |

### ArrayOfString

| Element Name | XSD Type | Notes |
|--------------|----------|-------|
| stringItem   | string   |       |

## Appendix B - Error Responses

All failed requests will return an HTTP error code of 500, a SOAP Fault message, and a service exception in the fault detail.

### SOAP Faults

SOAP faults that are returned by APIs will include **faultcode**, **faultstring**, and **detail** elements as specified by SOAP.

#### SOAP Fault Code

| Code            | Notes                  |
|-----------------|------------------------|
| SOAP-ENV:Client | Error caused by client |
| SOAP-ENV:Server | Error caused by server |

#### SOAP Fault String

Fault strings are short descriptions of the error.

| String                 | Notes  |
|------------------------|--|
| Internal Error         | Any internal error should be reported to Marketo |
| Authentication failed  |  |
| Request not understood |  |
| Access denied          |  |
| Request limit exceeded |  |
| Input error            |  |
| Lead not found         |  |
| Lead sync failed       |  |
| Request Expired        |  |

#### SOAP Fault Detail

SOAP faults contain a **detail** element that provides additional information about an API failure.

## Service Exceptions

Service exceptions contain an **exception name**, a **message** describing the exception, and an error **code**. Error codes are of two types: **request acceptance error codes** and **request processing error codes**.

| Element          | Notes                         |
|------------------|-------------------------------|
| serviceException | Exception generated on server |

### serviceException

| Element | Notes                             |
|---------|-----------------------------------|
| name    | Name of exception type on server  |
| message | Detailed description of the error |
| code    | Numeric error code (see below)    |

### Request Acceptance Error Codes

| Code  |                        | Notes  |
|-------|------------------------|--|
| 10001 | Internal Error         | Severe system failure                                    |
| 20011 | Internal Error         | API service failure                                      |
| 20012 | Request Not Understood | Unexpected SOAP message                                  |
| 20013 | Access Denied          | Client is blocked from API access                        |
| 20014 | Authentication Failed  | Client did not provide valid credentials                 |
| 20015 | Request Limit Exceeded | Client has exceeded its daily request quota              |
| 20016 | Request Expired        | Request signature is too old                             |
| 20017 | Invalid Request        | Request is missing expected parameter                    |
| 20019 | Unsupported Operation  | Operation invoked is not defined in the Marketo API WSDL |

### Request Processing Error Codes

| Code  |                      | Notes  |
|-------|----------------------|--|
| 20101 | Lead Key Required    | LeadKey is required but was not provided           |
| 20102 | Lead Key Bad         | LeadKey type is not valid                          |
| 20103 | Lead Not Found       | LeadKey value did not match any lead               |
| 20104 | Lead Detail Required | LeadRecord is required but was not provided        |
| 20105 | Lead Attribute Bad   | LeadRecord contains an Attribute with a bad name   |
| 20106 | Lead Sync Failed     | LeadRecord could not be updated or created         |
| 20107 | Activity Key Bad     | LeadActivityFilter contains a bad activity type    |
| 20108 | Lead Owner Not Found | LeadKey specifies a lead owner that does not exist |
| 20109 | Parameter Required   | Parameter value was null or missing                |
| 20110 | Bad Parameter        | Parameter value is bad                             |

|       |                     |  |
|-------|---------------------|--|
| 20111 | List Not Found      | ListKey specifies a list that does not exist             |
| 20113 | Campaign Not Found  | Campaign does not exist                                  |
| 20114 | Bad Parameter       | Parameter value is bad                                   |
| 20122 | Bad Stream Position | Stream position is bad                                   |
| 20123 | Stream at end       | Stream position indicates that no more records available |

## Sample Error Response

```

HTTP/1.1 500 Internal Server Error
Date: Thu, 26 Feb 2009 00:28:44 GMT
Server: Apache/2.2.3 (win32) DAV/2 mod_ssl/2.2.3 OpenSSL/0.9.8d mod_autoindex_color PHP/5.2.0
X-Powered-By: PHP/5.2.0
Set-Cookie: symfony=0d64e9fab0af56fc4439dd7531f6d8ac; path=/
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Content-Length: 546
Connection: close
Content-Type: text/xml; charset=utf-8

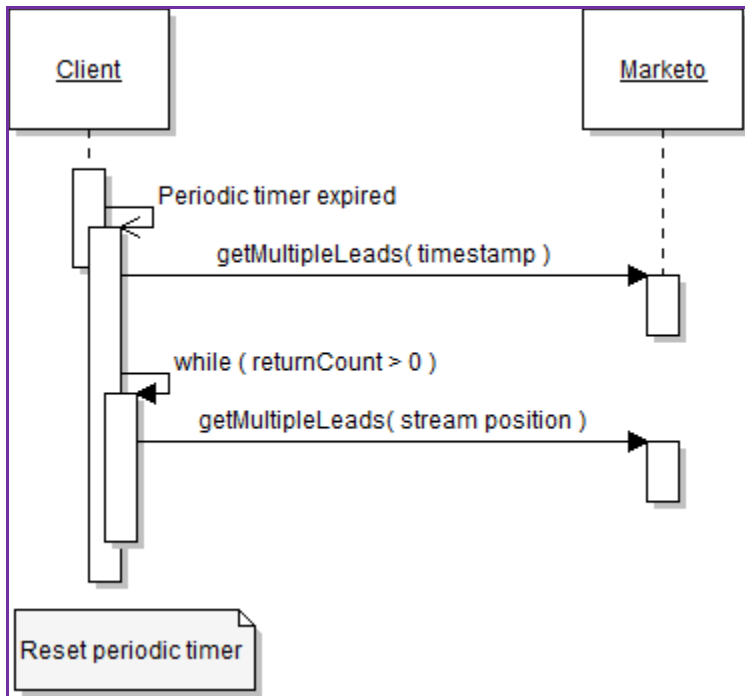
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Body>
    <SOAP-ENV:Fault>
      <faultcode>SOAP-ENV:Client</faultcode>
      <faultstring>Authentication failed</faultstring>
      <detail>
        <ns1:serviceException xmlns:ns1="http://www.marketo.com/mktows/">
          <name>mktServiceException</name>
          <message>Authentication failed</message>
          <code>20014</code>
        </ns1:serviceException>
      </detail>
    </SOAP-ENV:Fault>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

```

## Appendix C - Integrating with Marketo

### Periodic Pull of Updated Leads

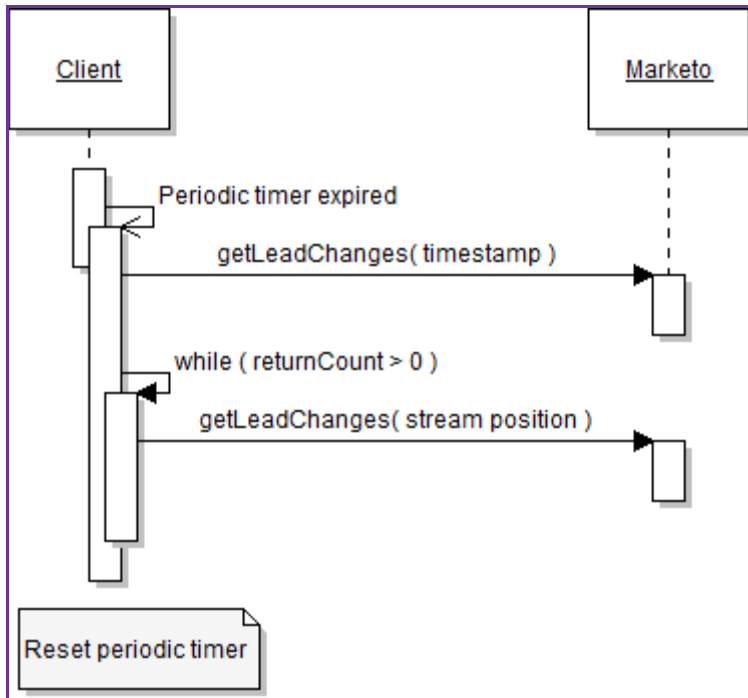
Periodically pull all leads that have been updated in Marketo.





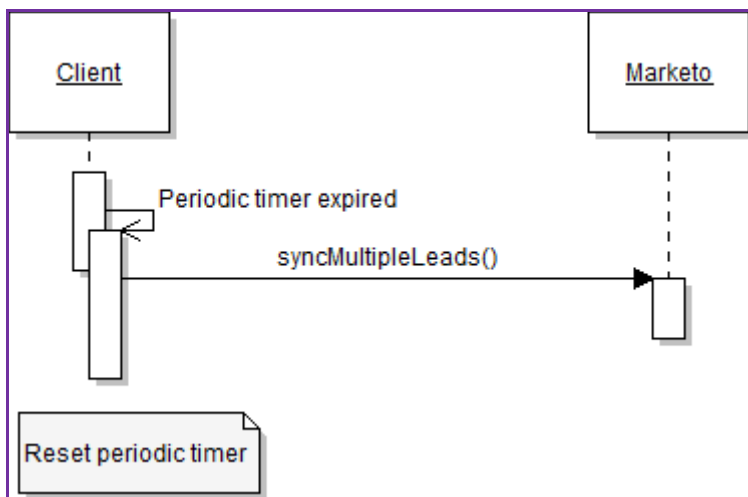
## Periodic Pull of Changed Data Value

Periodically pull any data value that has changed for any lead.



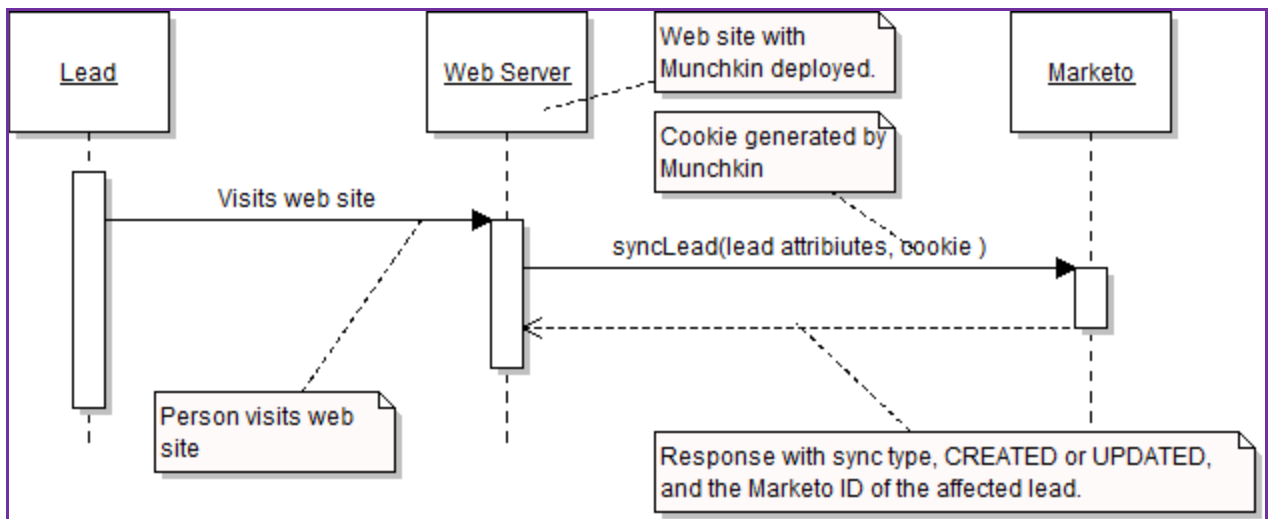
## Periodic Push of Updated Leads

Periodically push a set of updated leads to Marketo.



## Capture New Lead

Capture a new lead and, optionally, associate a Marketo Munchkin cookie.

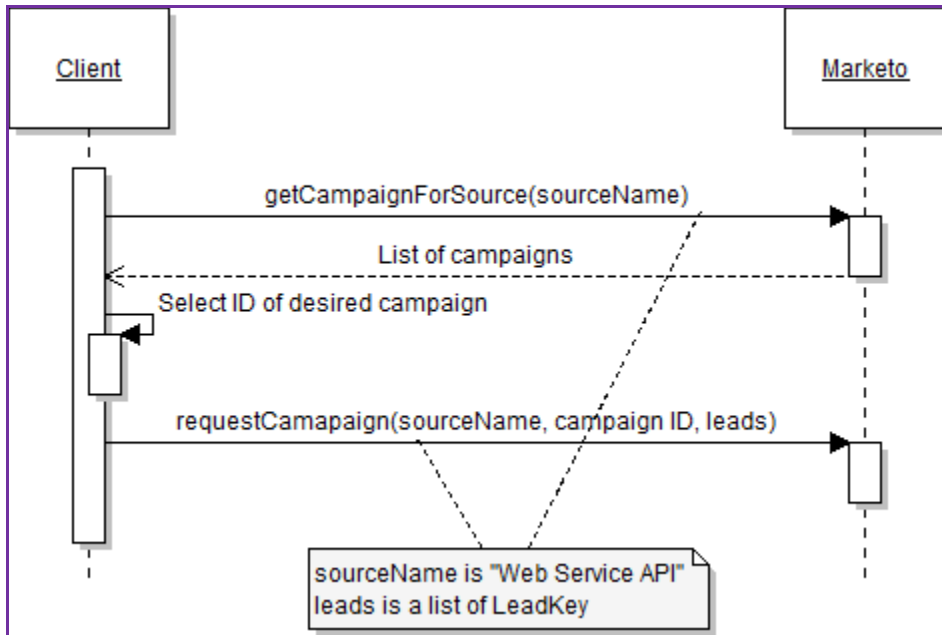


## Add Lead to Campaign

Begin by creating a campaign in Marketo. Adding the **Campaign is Requested** trigger to a campaign and setting the **Source** as **Web Service API** makes the campaign accessible through the **requestCampaign** API. Configure the campaign flow and schedule as desired and activate the campaign.

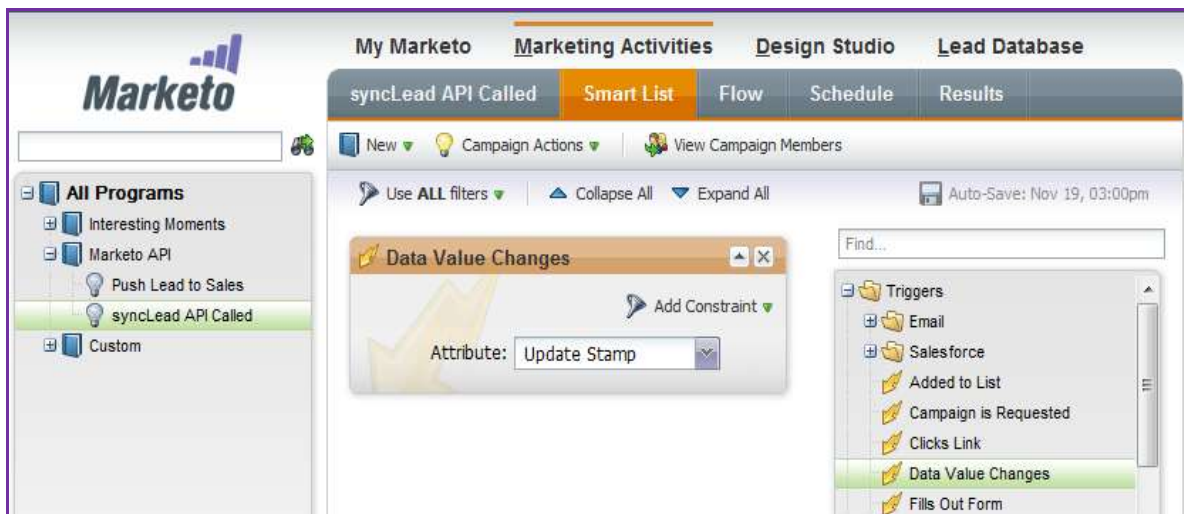


Use the **getCampaignForSource** API to obtain a list of accessible campaigns. Call the **requestCampaign** API with the ID of the desired campaign and a list of leads.



## Trigger Campaign with syncLead

It is often necessary to trigger a campaign when a lead is updated through the syncLead API. This is analogous to triggering a campaign using the **Fills Out Form** trigger when a lead fills out a form. Triggering a campaign from a syncLead API call can be achieved by leveraging the **Data Value Changes** trigger. However, the trigger will not occur if no data value is actually changed. A workaround for this limitation is to add a custom field for the sole purpose of triggering the campaign. The **syncLead** API caller should then update this field with a new value every time, regardless of any other field values. This field can be a numeric type which is updated with the current time, represented as an integer value.

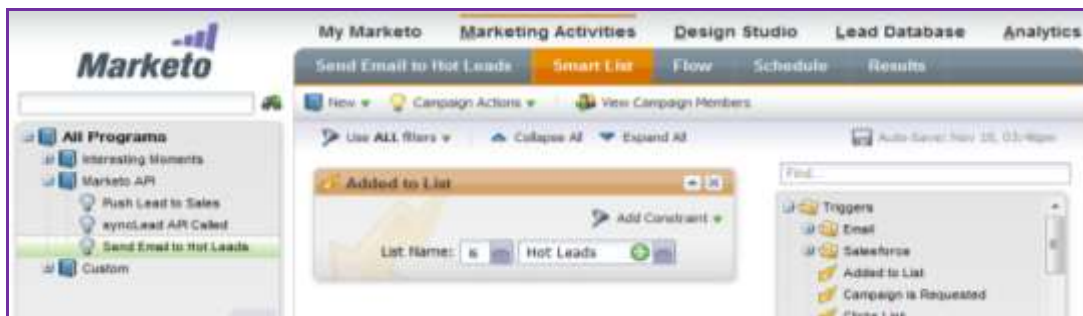


## Trigger Campaign with listOperation

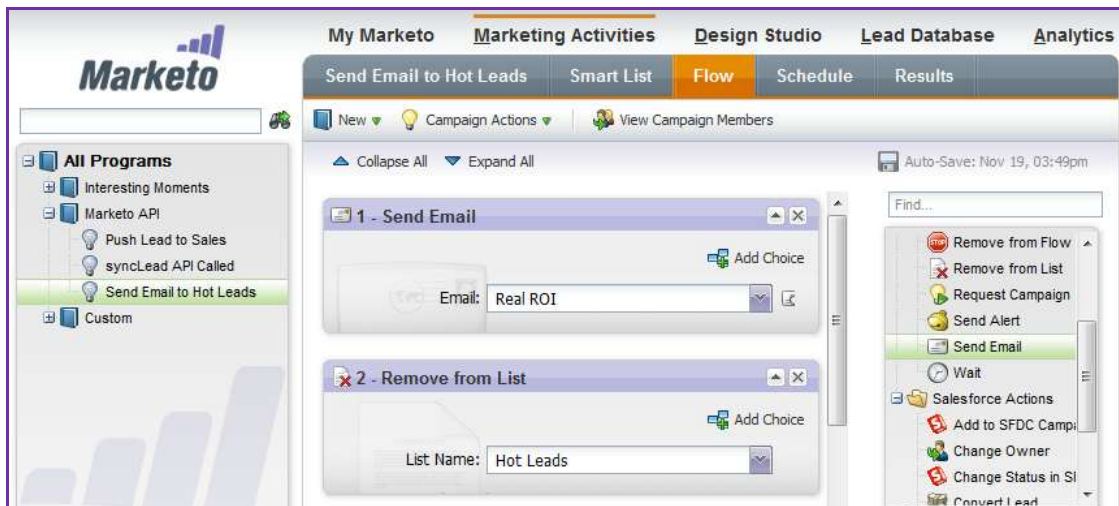
Any campaign that is configured to trigger on **Added to List** or **Removed from List** triggers can be triggered through the **listOperation** API. This simple combination can be used for some sophisticated processing. It may be preferable to create dedicated Static Lists for use with the **listOperation** API.



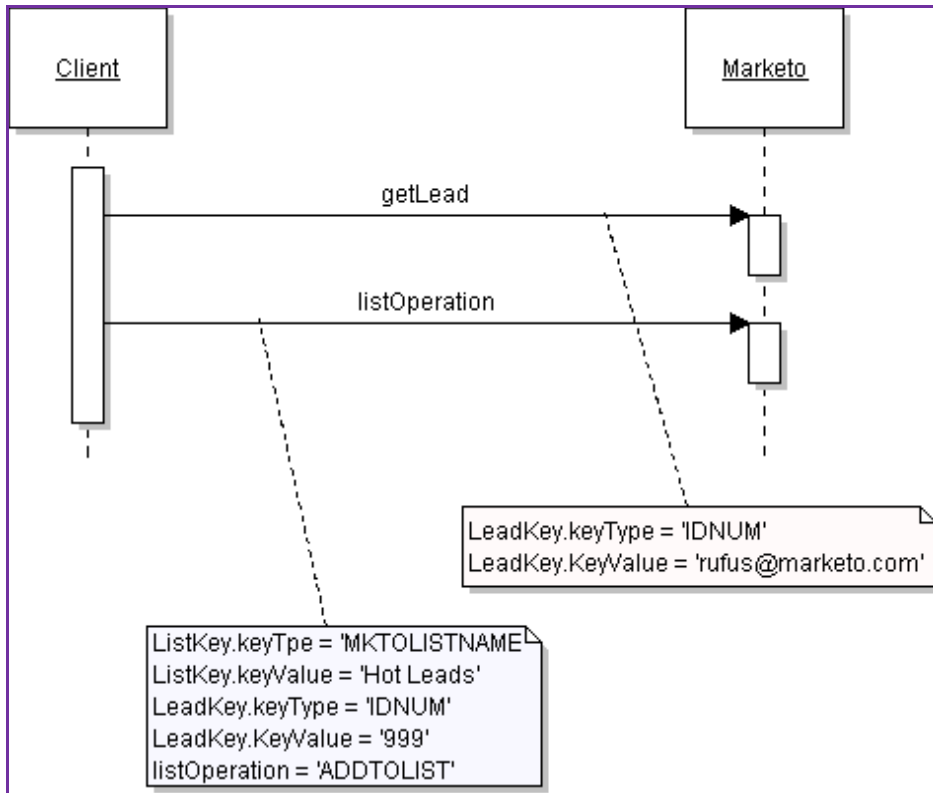
Create a campaign with **Added to List** trigger.



Add desired flow steps to the campaign and activate it. If the final step is **Remove from List** and the campaign schedule is configured with **Run flow every time**, then the campaign can be used repeatedly on the same lead.



Invoke the listOperation API.



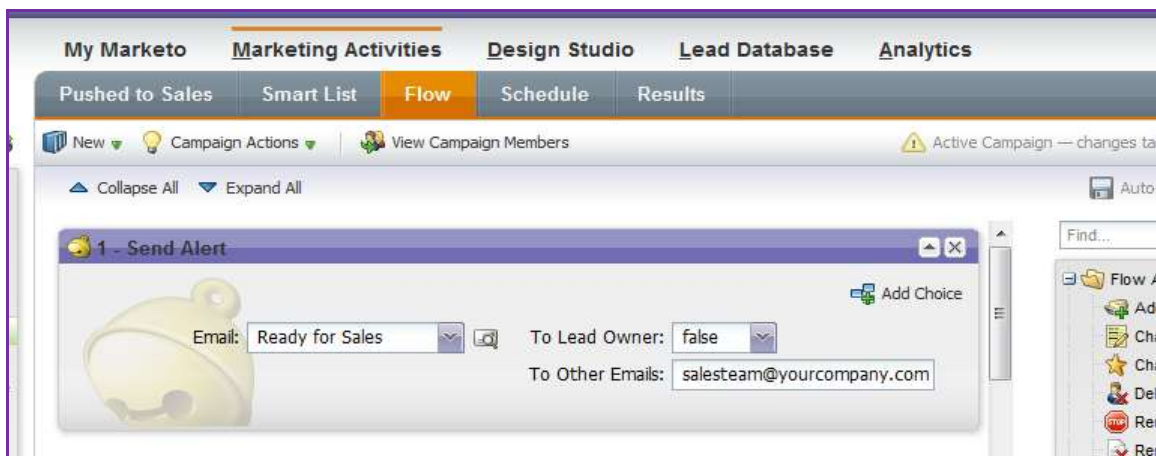
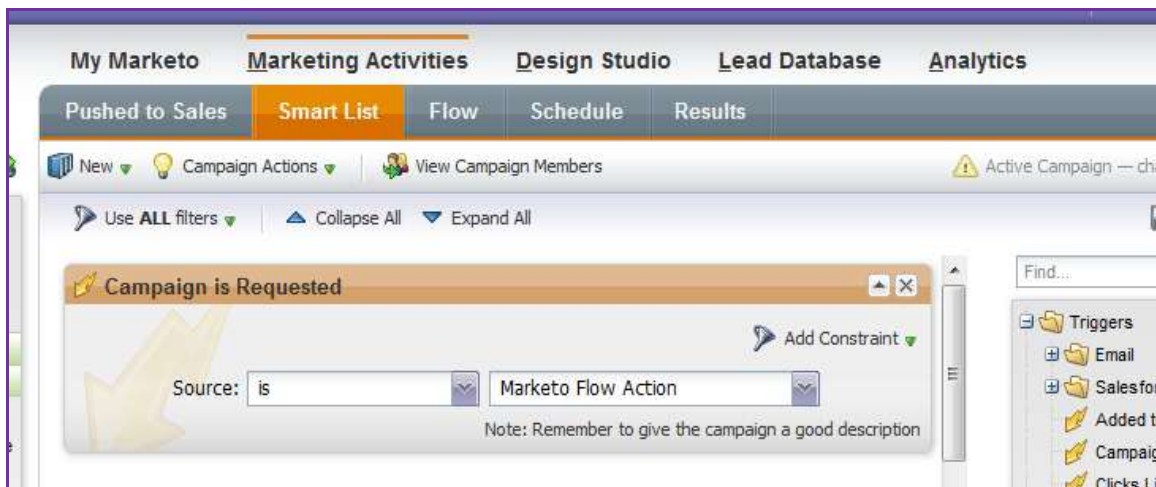
## Push Lead to Sales

Marketo provides a feature rich integration with Salesforce that includes the ability to push nurtured leads to the sales team. However, if Salesforce is not being used – or if, for some reason, the Marketo Salesforce integration is not being used – the alternative described below can be used instead.

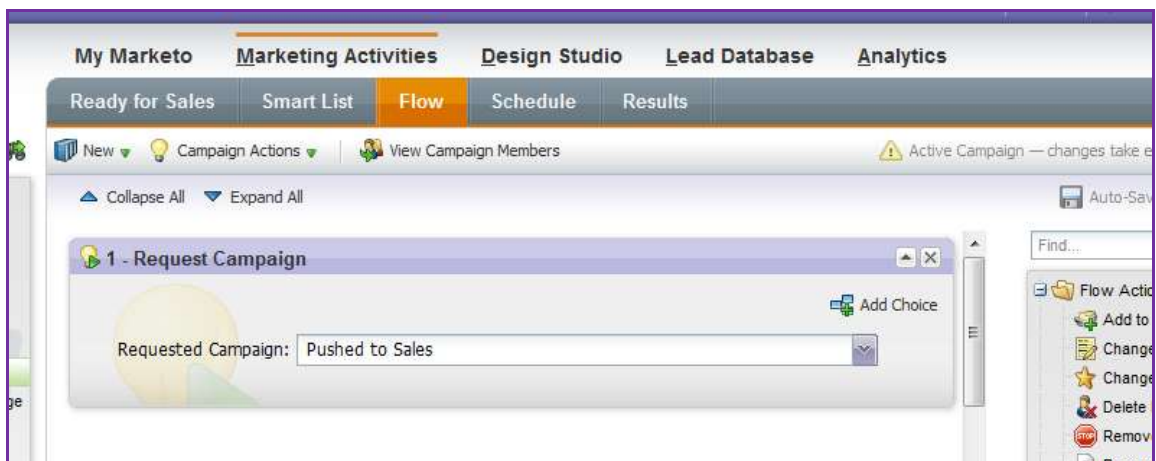
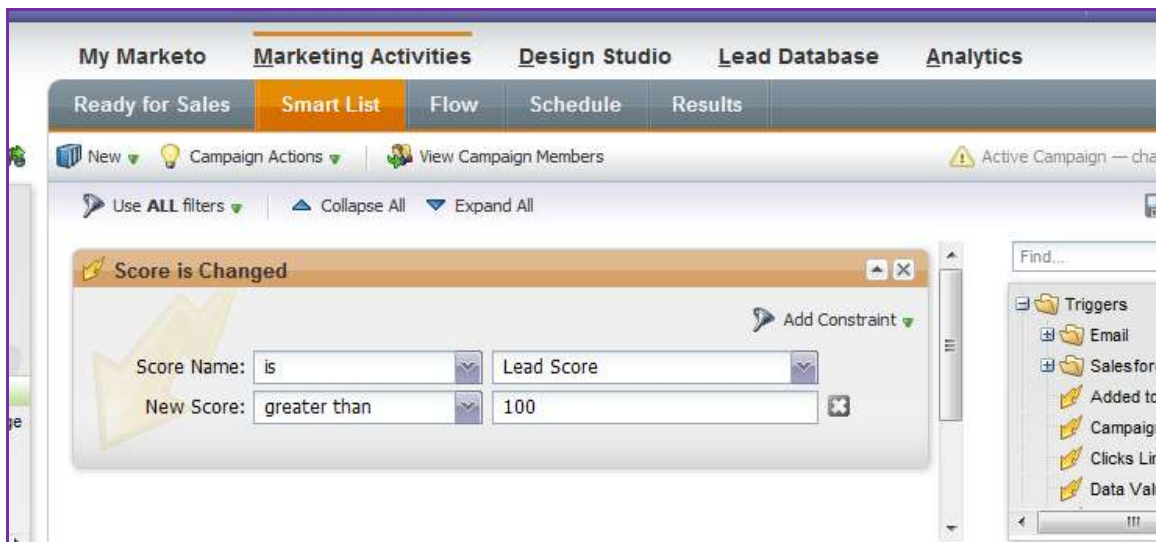
First, create an Alert email, named “Ready for Sales”, with the following content:



Next, create a campaign, named “Push Lead to Sales”, with the following trigger and flow step:



Next, feed the “Push Lead to Sales” campaign from any other campaign that can determine when a lead is ready for sales. As a simple example, you can have another campaign, named “Ready for Sales”, with the following trigger and flow step:



Finally, use the Marketo API **getLeadChanges** to perform a periodic pull of leads that have been added to the “Pushed to Sales” campaign. In the API call, add an activity filter to return only the **Request Campaign** activity type. In the response data, look for records that have the **mktgAssetName** attribute with the value “Pushed to Sales”. Here is a sample request fragment followed by the corresponding response fragment. The response contains the **mktPersonId**, which can be used in a **getLead** API call to pull the lead record.



## Request

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:ns1="http://www.marketo.com/mktows/">
  <SOAP-ENV:Header>
    <ns1:AuthenticationHeader>
      ...
    </ns1:AuthenticationHeader>
  </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    <ns1:paramsGetLeadChanges>
      <startPosition>
        <latestCreatedAt>2010-05-04T00:00:00-07:00</latestCreatedAt>
        <oldestCreatedAt>2010-04-24T00:00:00-07:00</oldestCreatedAt>
        <activityCreatedAt xsi:nil="true"/>
        <offset xsi:nil="true"/>
      </startPosition>
      <activityFilter>
        <includeTypes>
          <activityType>RequestCampaign</activityType>
        </includeTypes>
        <excludeTypes/>
      </activityFilter>
      <batchSize>3</batchSize>
    </ns1:paramsGetLeadChanges>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```



## Response

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:ns1="http://www.marketo.com/mktows/">
  <SOAP-ENV:Body>
    <ns1:successGetLeadChanges>
      <result>
        <returnCount>4</returnCount>
        <remainingCount>0</remainingCount>
        <newStartPosition>
          <latestCreatedAt>2010-05-10T00:00:00-07:00</latestCreatedAt>
          <oldestCreatedAt>2010-05-03T00:00:00-07:00</oldestCreatedAt>
          <activityCreatedAt xsi:nil="true"/>
          <offset>4</offset>
        </newStartPosition>
        <leadChangeRecordList>
          <leadChangeRecord>
            <id>20842</id>
            <activityDateTime>2010-05-07T15:41:32-05:00</activityDateTime>
            <activityType>Request Campaign</activityType>
            <mktgAssetName>Pushed to Sales</mktgAssetName>
            <activityAttributes>
              <attribute>
                <attrName>Campaign ID</attrName>
                <attrType xsi:nil="true"/>
                <attrValue>1011</attrValue>
              </attribute>
              <attribute>
                <attrName>Step ID</attrName>
                <attrType xsi:nil="true"/>
                <attrValue>26</attrValue>
              </attribute>
              <attribute>
                <attrName>Choice Number</attrName>
                <attrType xsi:nil="true"/>
                <attrValue>0</attrValue>
              </attribute>
              <attribute>
                <attrName>Source</attrName>
                <attrType xsi:nil="true"/>
                <attrValue>Marketo Flow Action</attrValue>
              </attribute>
              <attribute>
                <attrName>Lead ID</attrName>
                <attrType xsi:nil="true"/>
                <attrValue>12</attrValue>
              </attribute>
            </activityAttributes>
            <campaign>Ready for Sales</campaign>
            <mktPersonId>12</mktPersonId>
          </leadChangeRecord>
          <leadChangeRecord>
            ...
          </leadChangeRecord>
        </leadChangeRecordList>
      </result>
    </ns1:successGetLeadChanges>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

## **Appendix D. Marketo SOAP API Java Example**

[Here](#) is a Java implementation of Marketo's SOAP API. This is unsupported, but you can use it freely as a starting point for building your own SOAP API client. This information is also available in the Knowledge Base article, [Marketo SOAP API Java example](#).

## **Appendix E. Marketo SOAP API PHP Client**

[Here](#) is a PHP implementation of Marketo's SOAP API. This is unsupported, but you can use it freely as a starting point for building your own SOAP API client. This information is also available in the Knowledge Base article, [Marketo SOAP API PHP Client](#).

## Appendix F. Checking for User Registration via SOAP API

Using the Marketo SOAP API, you can check whether or not your website visitors have filled out a form and present them content accordingly. [Here](#) is a PHP file with an example of this check. This information is also available in the Knowledge Base article, [Checking for User Registration via SOAP API](#).

**Note: the code below is unsupported and is intended to help you develop your own solution.**

In the [resources section](#) of the Marketo website, we use this method for our premium content. For leads who have already registered, their links go directly to the resource. Leads who haven't registered are given links to a landing page. In both cases, the link appears the same:

---



### The New Revenue Engine

★★★★★ | 777 views

This DemandGen Report eBook takes a look at how leading B2B marketers are improving efficiency and effectiveness by building shared sales and marketing machines. Read **The New Revenue Engine** and learn how to:

- ♦ Synergize your marketing and sales efforts
- ♦ Build a combined revenue engine
- ♦ Optimize lead management
- ♦ Create a winning strategy

 [Premium Content - Read More »](#)

---

Behind the scenes, a cookie is used to store whether or not this lead has registered:

- If the user has the premium registration cookie, the premium links are shown.
- If the user does not have that cookie, the lead's record is retrieved via the SOAP API.

Based on the lead's record:

- If the lead has a valid email address, the premium content links are shown and the premium registration cookie is set.
- If not, the premium content links are not shown. Instead, they take leads to a landing page to fill out a form.

First, you'll need to develop a SOAP API connection to Marketo. [This](#) ZIP file has a sample connection PHP file.

You'll need to make the following changes in this file:

```
...
$this->access_key = 'Your API Access Key Goes Here';
$this->secret_key = 'Your Secret API Key Goes Here';

//
// The endpoint is in the "SOAP API Setup" page in the Marketo Admin section
// ex. $soap_end_point = 'https://xx-1.marketo.com/soap/mktows/';
//
$soap_end_point = 'Your SOAP API End Point URL Goes Here';

//
// Errors are sent to this email address. Your web server
// must be configured to send email for this to work correctly.
//
// ex. $this->error_email_address = 'example@example.com';
//
$this->error_email_address = 'Put your debug email address here';
...
```

After creating your SOAP API connection, the next step is to implement the logic that determines which content to show if the user is logged in or not. The `get_premium_url_status()` function returns true if the lead has the premium cookie or if they've filled out a form. Otherwise, it returns false. You can use that result to determine what content the lead should see.

## Appendix G. Sample XML

Below are several XML documents, including the schemas for some objects, and also sample request and response fragments.

### Sample XML Request and Response

#### Sample Request XML

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-
ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ns1="http://www.marketo.com/mktows/">
  <SOAP-ENV:Header>
    <ns1:AuthenticationHeader>
      <mktowsUserId>bigcorp1_719871464E02D4291DB270</mktowsUserId>

    <requestSignature>a43b37a6d2a09448b235c98aa49c0af07c99c262</requestSign
ature>
    <requestTimestamp>2011-07-24T19:38:58-07:00</requestTimestamp>
  </ns1:AuthenticationHeader>
</SOAP-ENV:Header>
<SOAP-ENV:Body>
  <ns1:paramsSyncMObjects>
    <mObjectList>
      <mObject>
        <type>Opportunity</type>
        <id/>
        <externalKey/>
        <createdAt/>
        <updatedAt/>
        <attribList>
          <attrib>
            <name>Name</name>
            <value>Knocking Opportunity</value>
          </attrib>
          <attrib>
            <name>Amount</name>
            <value>1234.45</value>
          </attrib>
          <attrib>
            <name>ExternalCreatedDate</name>
            <value>2011-07-22</value>
          </attrib>
          <attrib>
            <name>Fiscal</name>
            <value>2011-4</value>
          </attrib>
          <attrib>
            <name>FiscalYear</name>
            <value>2011</value>
          </attrib>
          <attrib>
```

```

        <name>FiscalQuarter</name>
        <value>4</value>
      </attrib>
    </attribList>
    <associationList/>
  </mObject>
</mObjectList>
<operation>INSERT</operation>
</ns1:paramsSyncMObjects>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>

```

## Sample Response XML

```

<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-
ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ns1="http://www.marketo.com/mktows/">
  <SOAP-ENV:Body>
    <ns1:successSyncMObjects>
      <result>
        <mObjStatusList>
          <mObjStatus>
            <id>116</id>
            <externalKey>
              <name/>
              <value/>
            </externalKey>
            <status>CREATED</status>
            <error/>
          </mObjStatus>
        </mObjStatusList>
      </result>
    </ns1:successSyncMObjects>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

```

## SOAP Fault

```

<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-
ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Body>
    <SOAP-ENV:Fault>
      <faultcode>SOAP-ENV:Client</faultcode>
      <faultstring>21108 - MObject not found</faultstring>
      <detail>
        <ns1:serviceException
xmlns:ns1="http://www.marketo.com/mktows/">
          <name>mktServiceException</name>
          <message>OpportunityPersonRole not found with ID 112
(21108)</message>
          <code>21108</code>
        </ns1:serviceException>
      </detail>
    </SOAP-ENV:Fault>
  </SOAP-ENV:Body>

```

</SOAP-ENV:Envelope>

## MObject XML Schema

```
<xs:complexType name="MObject">
  <xs:sequence>
    <xs:element name="type" type="xs:string" minOccurs="1"
maxOccurs="1" nillable="false"/>
    <xs:element name="id" type="xs:int" minOccurs="0" maxOccurs="1"
nillable="false"/>
    <xs:element name="externalKey" type="tns:Attrib" minOccurs="0"
maxOccurs="1" nillable="false"/>
    <xs:element name="createdAt" type="xs:dateTime" minOccurs="0"
maxOccurs="1" nillable="false"/>
    <xs:element name="updatedAt" type="xs:dateTime" minOccurs="0"
maxOccurs="1" nillable="false"/>
    <xs:element name="attribList" type="tns:ArrayOfAttrib"
minOccurs="0" maxOccurs="1" nillable="true"/>
    <xs:element name="associatonList" type="tns:ArrayOfMObjAssociation"
minOccurs="0" maxOccurs="1" nillable="true"/>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="Attrib">
  <xs:sequence>
    <xs:element name="name" type="xs:string" minOccurs="1"
maxOccurs="1" nillable="false"/>
    <xs:element name="value" type="xs:string" minOccurs="1"
maxOccurs="1" nillable="false"/>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="ArrayOfAttrib">
  <xs:sequence>
    <xs:element name="attrib" type="tns:Attrib" minOccurs="0"
maxOccurs="unbounded" nillable="false"/>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="MObjAssociation">
  <xs:sequence>
    <xs:element name="mObjType" type="xs:string" minOccurs="1"
maxOccurs="1" nillable="false"/>
    <xs:element name="id" type="xs:int" minOccurs="0" maxOccurs="1"
nillable="false"/>
    <xs:element name="externalKey" type="tns:Attrib" minOccurs="0"
maxOccurs="1" nillable="false"/>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="ArrayOfMObjAssociation">
  <xs:sequence>
    <xs:element name="mObjAssociation" type="tns:MObjAssociation"
minOccurs="0" maxOccurs="unbounded" nillable="false"/>
  </xs:sequence>
</xs:complexType>
```



## Sample Objects

### Sample Opportunity

```
<mObject>
  <type>Opportunity</type>
  <id>97</id>
  <externalKey/>
  <createdAt>2011-07-12T10:00:00-7:00</createdAt>
  <updatedAt>2011-07-12T11:00:00-7:00</updatedAt>
  <attribList>
    <attrib>
      <name>Name</name>
      <value>A Golden Opportunity</value>
    </attrib>
    <attrib>
      <name>Amount</name>
      <value>1234.56</value>
    </attrib>
    <attrib>
      <name>ExternalCreatedDate</name>
      <value>2011-07-22</value>
    </attrib>
    <attrib>
      <name>Fiscal</name>
      <value>2011-4</value>
    </attrib>
    <attrib>
      <name>FiscalYear</name>
      <value>2011</value>
    </attrib>
    <attrib>
      <name>FiscalQuarter</name>
      <value>4</value>
    </attrib>
  </attribList>
  <associationList/>
</mObject>
```

### Sample OpportunityPersonRole

**IMPORTANT:** Replace the keys shown as *MyCustomRoleId* and *MyCustomLeadId* with your own custom fields.

```
<MObject>
  <type>OpportunityPersonRole</type>
  <id>123</id>
  <externalKey>
    <name>MyCustomRoleId</name>
    <value>456XYZ-0</value>
  </externalKey>
  <createdAt>2011-07-12T10:00:00-7:00</createdAt>
  <updatedAt>2011-07-12T11:00:00-7:00</updatedAt>
  <attribList>
    <attrib>
```

```

        <name>Role</name>
        <value>Decision Maker</value>
    </attrib>
</attribList>
<associationList>
    <mObjAssociation>
        <mObjType>Opportunity</mObjType>
        <id>97</id>
    </mObjAssociation>
    <mObjAssociation>
        <mObjType>Contact</mObjType>
        <externalKey>
            <name>MyCustomLeadId</name>
            <value>123ABC-0</value>
        </externalKey>
    </mObjAssociation>
</associationList>
</MObject>

```

## Sample Request Fragments

### syncMObjects

```

<SOAP-ENV:Body>
  <ns1:paramsSyncMObjects>
    <mObjectList>
      <mObject>
        <type>OpportunityPersonRole</type>
        <id/>
        <externalKey/>
        <createdAt/>
        <updatedAt/>
        <attribList>
          <attrib>
            <name>IsPrimary</name>
            <value/>
          </attrib>
          <attrib>
            <name>Role</name>
            <value>Pessimist</value>
          </attrib>
        </attribList>
        <associationList>
          <mObjAssociation>
            <mObjType>Opportunity</mObjType>
            <id>143</id>
            <externalKey/>
          </mObjAssociation>
          <mObjAssociation>
            <mObjType>Lead</mObjType>
            <id>1000156</id>
            <externalKey/>
          </mObjAssociation>
        </associationList>
      </mObject>
    </mObjectList>
  </ns1:paramsSyncMObjects>
</SOAP-ENV:Body>

```

```

<mObject>
  <type>OpportunityPersonRole</type>
  <id/>
  <externalKey/>
  <createdAt/>
  <updatedAt/>
  <attribList>
    <attrib>
      <name>IsPrimary</name>
      <value/>
    </attrib>
    <attrib>
      <name>Role</name>
      <value>Doubter</value>
    </attrib>
  </attribList>
  <associationList>
    <mObjAssociation>
      <mObjType>Opportunity</mObjType>
      <id>143</id>
      <externalKey/>
    </mObjAssociation>
    <mObjAssociation>
      <mObjType>Lead</mObjType>
      <id>1000156</id>
      <externalKey/>
    </mObjAssociation>
  </associationList>
</mObject>
<mObject>
  <type>OpportunityPersonRole</type>
  <id/>
  <externalKey/>
  <createdAt/>
  <updatedAt/>
  <attribList>
    <attrib>
      <name>IsPrimary</name>
      <value/>
    </attrib>
    <attrib>
      <name>Role</name>
      <value>Confuser</value>
    </attrib>
  </attribList>
  <associationList>
    <mObjAssociation>
      <mObjType>Opportunity</mObjType>
      <id>143</id>
      <externalKey/>
    </mObjAssociation>
    <mObjAssociation>
      <mObjType>Lead</mObjType>
      <id>1000156</id>
      <externalKey/>
    </mObjAssociation>
  </associationList>

```

```

        </associationList>
      </mObject>
    </mObjectList>
    <operation>INSERT</operation>
  </ns1:paramsSyncMObjects>
</SOAP-ENV:Body>

```

### deleteMObject by ID

```

<SOAP-ENV:Body>
  <ns1:paramsDeleteMObjects>
    <mObjectList>
      <mObject>
        <type>Opportunity</type>
        <id>143</id>
        <externalKey/>
        <createdAt/>
        <updatedAt/>
        <attribList/>
        <associationList/>
      </mObject>
    </mObjectList>
  </ns1:paramsDeleteMObjects>
</SOAP-ENV:Body>

```

### getMObject by ID

```

<SOAP-ENV:Body>
  <ns1:paramsGetMObjects>
    <type>OpportunityPersonRole</type>
    <id/>
    <externalKey/>
    <mObjCriteriaList/>
    <mObjAssociationList>
      <mObjAssociation>
        <mObjType>Opportunity</mObjType>
        <id>143</id>
        <externalKey/>
      </mObjAssociation>
    </mObjAssociationList>
    <streamPosition/>
  </ns1:paramsGetMObjects>
</SOAP-ENV:Body>

```

### getMObject by Association

```

<SOAP-ENV:Body>
  <ns1:paramsGetMObjects>
    <type>OpportunityPersonRole</type>
    <id/>
    <externalKey/>
    <mObjCriteriaList/>
    <mObjAssociationList>
      <mObjAssociation>
        <mObjType>Lead</mObjType>
        <id>1000156</id>
        <externalKey/>
      </mObjAssociation>
    </mObjAssociationList>
  </ns1:paramsGetMObjects>
</SOAP-ENV:Body>

```

```

        </mObjAssociation>
    </mObjAssociationList>
    <streamPosition/>
</ns1:paramsGetMObjects>
</SOAP-ENV:Body>

```

### getMObject by Criteria

```

<SOAP-ENV:Body>
  <ns1:paramsGetMObjects>
    <type>Opportunity</type>
    <id/>
    <externalKey/>
    <mObjCriteriaList>
      <mObjCriteria>
        <attrName>Type</attrName>
        <comparison>EQ</comparison>
        <attrValue>Pipeline</attrValue>
      </mObjCriteria>
      <mObjCriteria>
        <attrName>Stage</attrName>
        <comparison>EQ</comparison>
        <attrValue>Scoping</attrValue>
      </mObjCriteria>
    </mObjCriteriaList>
    <mObjAssociationList/>
    <streamPosition/>
  </ns1:paramsGetMObjects>
</SOAP-ENV:Body>

```

# Metadata

## Opportunity Metadata

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-
ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:ns1="http://www.marketo.com/mktows/">
  <SOAP-ENV:Body>
    <ns1:successDescribeMObject>
      <result>
        <metadata>
          <name>Opportunity</name>
          <description>Marketo Opportunity</description>
          <isCustom>false</isCustom>
          <isVirtual>false</isVirtual>
          <fieldList>
            <field>
              <name>Amount</name>
              <description xsi:nil="true"/>
              <displayName>Amount</displayName>
              <sourceObject>Opportunity</sourceObject>
              <dataType>currency</dataType>
              <size xsi:nil="true"/>
              <isReadonly>false</isReadonly>
              <isUpdateBlocked>false</isUpdateBlocked>
              <isName xsi:nil="true"/>
              <isPrimaryKey>false</isPrimaryKey>
              <isCustom>false</isCustom>
              <isDynamic>false</isDynamic>
              <dynamicFieldRef xsi:nil="true"/>
              <updatedAt>2011-08-24T18:39:06-07:00</updatedAt>
            </field>
            <field>
              <name>CloseDate</name>
              <description xsi:nil="true"/>
              <displayName>Close Date</displayName>
              <sourceObject>Opportunity</sourceObject>
              <dataType>date</dataType>
              <size xsi:nil="true"/>
              <isReadonly>false</isReadonly>
              <isUpdateBlocked>false</isUpdateBlocked>
              <isName xsi:nil="true"/>
              <isPrimaryKey>false</isPrimaryKey>
              <isCustom>false</isCustom>
              <isDynamic>false</isDynamic>
              <dynamicFieldRef xsi:nil="true"/>
              <updatedAt>2011-08-24T18:39:06-07:00</updatedAt>
            </field>
            <field>
              <name>CompanyId</name>
              <description xsi:nil="true"/>
              <displayName>Account</displayName>
```

```

    <sourceObject>Opportunity</sourceObject>
    <dataType>reference</dataType>
    <size xsi:nil="true"/>
    <isReadonly>false</isReadonly>
    <isUpdateBlocked>false</isUpdateBlocked>
    <isName xsi:nil="true"/>
    <isPrimaryKey>false</isPrimaryKey>
    <isCustom>false</isCustom>
    <isDynamic>false</isDynamic>
    <dynamicFieldRef xsi:nil="true"/>
    <updatedAt>2011-09-02T11:55:45-07:00</updatedAt>
  </field>
  <field>
    <name>Description</name>
    <description xsi:nil="true"/>
    <displayName>Description</displayName>
    <sourceObject>Opportunity</sourceObject>
    <dataType>string</dataType>
    <size>2000</size>
    <isReadonly>false</isReadonly>
    <isUpdateBlocked>false</isUpdateBlocked>
    <isName xsi:nil="true"/>
    <isPrimaryKey>false</isPrimaryKey>
    <isCustom>false</isCustom>
    <isDynamic>false</isDynamic>
    <dynamicFieldRef xsi:nil="true"/>
    <updatedAt>2011-08-24T18:39:06-07:00</updatedAt>
  </field>
  <field>
    <name>ExpectedRevenue</name>
    <description xsi:nil="true"/>
    <displayName>Expected Revenue</displayName>
    <sourceObject>Opportunity</sourceObject>
    <dataType>currency</dataType>
    <size xsi:nil="true"/>
    <isReadonly>false</isReadonly>
    <isUpdateBlocked>false</isUpdateBlocked>
    <isName xsi:nil="true"/>
    <isPrimaryKey>false</isPrimaryKey>
    <isCustom>false</isCustom>
    <isDynamic>false</isDynamic>
    <dynamicFieldRef xsi:nil="true"/>
    <updatedAt>2011-08-24T18:39:06-07:00</updatedAt>
  </field>
  <field>
    <name>ExternalCreatedDate</name>
    <description xsi:nil="true"/>
    <displayName>SFDC Created Date</displayName>
    <sourceObject>Opportunity</sourceObject>
    <dataType>datetime</dataType>
    <size xsi:nil="true"/>
    <isReadonly>false</isReadonly>
    <isUpdateBlocked>false</isUpdateBlocked>
    <isName xsi:nil="true"/>
    <isPrimaryKey>false</isPrimaryKey>
    <isCustom>false</isCustom>

```

```

    <isDynamic>false</isDynamic>
    <dynamicFieldRef xsi:nil="true"/>
    <updatedAt>2011-08-24T18:39:07-07:00</updatedAt>
  </field>
  <field>
    <name>Fiscal</name>
    <description xsi:nil="true"/>
    <displayName>Fiscal</displayName>
    <sourceObject>Opportunity</sourceObject>
    <dataType>string</dataType>
    <size>255</size>
    <isReadOnly>false</isReadOnly>
    <isUpdateBlocked>false</isUpdateBlocked>
    <isName xsi:nil="true"/>
    <isPrimaryKey>false</isPrimaryKey>
    <isCustom>false</isCustom>
    <isDynamic>false</isDynamic>
    <dynamicFieldRef xsi:nil="true"/>
    <updatedAt>2011-08-24T18:39:06-07:00</updatedAt>
  </field>
  <field>
    <name>FiscalQuarter</name>
    <description xsi:nil="true"/>
    <displayName>Fiscal Quarter</displayName>
    <sourceObject>Opportunity</sourceObject>
    <dataType>integer</dataType>
    <size xsi:nil="true"/>
    <isReadOnly>false</isReadOnly>
    <isUpdateBlocked>false</isUpdateBlocked>
    <isName xsi:nil="true"/>
    <isPrimaryKey>false</isPrimaryKey>
    <isCustom>false</isCustom>
    <isDynamic>false</isDynamic>
    <dynamicFieldRef xsi:nil="true"/>
    <updatedAt>2011-08-24T18:39:06-07:00</updatedAt>
  </field>
  <field>
    <name>FiscalYear</name>
    <description xsi:nil="true"/>
    <displayName>Fiscal Year</displayName>
    <sourceObject>Opportunity</sourceObject>
    <dataType>integer</dataType>
    <size xsi:nil="true"/>
    <isReadOnly>false</isReadOnly>
    <isUpdateBlocked>false</isUpdateBlocked>
    <isName xsi:nil="true"/>
    <isPrimaryKey>false</isPrimaryKey>
    <isCustom>false</isCustom>
    <isDynamic>false</isDynamic>
    <dynamicFieldRef xsi:nil="true"/>
    <updatedAt>2011-08-24T18:39:06-07:00</updatedAt>
  </field>
  <field>

```



```

<name>ForecastCategoryName</name>
<description xsi:nil="true"/>
<displayName>Forecast Category Name</displayName>
<sourceObject>Opportunity</sourceObject>
<dataType>string</dataType>
<size>255</size>
<isReadonly>false</isReadonly>
<isUpdateBlocked>false</isUpdateBlocked>
<isName xsi:nil="true"/>
<isPrimaryKey>false</isPrimaryKey>
<isCustom>false</isCustom>
<isDynamic>false</isDynamic>
<dynamicFieldRef xsi:nil="true"/>
<updatedAt>2011-08-24T18:49:30-07:00</updatedAt>
</field>
<field>
  <name>Id</name>
  <description xsi:nil="true"/>
  <displayName>Id</displayName>
  <sourceObject>Opportunity</sourceObject>
  <dataType>integer</dataType>
  <size xsi:nil="true"/>
  <isReadonly>false</isReadonly>
  <isUpdateBlocked>false</isUpdateBlocked>
  <isName xsi:nil="true"/>
  <isPrimaryKey>false</isPrimaryKey>
  <isCustom>false</isCustom>
  <isDynamic>true</isDynamic>
  <dynamicFieldRef>attributeList</dynamicFieldRef>
  <updatedAt>2011-08-24T18:39:06-07:00</updatedAt>
</field>
<field>
  <name>IsClosed</name>
  <description xsi:nil="true"/>
  <displayName>Is Closed</displayName>
  <sourceObject>Opportunity</sourceObject>
  <dataType>boolean</dataType>
  <size xsi:nil="true"/>
  <isReadonly>false</isReadonly>
  <isUpdateBlocked>false</isUpdateBlocked>
  <isName xsi:nil="true"/>
  <isPrimaryKey>false</isPrimaryKey>
  <isCustom>false</isCustom>
  <isDynamic>false</isDynamic>
  <dynamicFieldRef xsi:nil="true"/>
  <updatedAt>2011-08-24T18:39:07-07:00</updatedAt>
</field>

```

```

<field>
  <name>IsWon</name>
  <description xsi:nil="true"/>
  <displayName>Is Won</displayName>
  <sourceObject>Opportunity</sourceObject>
  <dataType>boolean</dataType>
  <size xsi:nil="true"/>
  <isReadonly>false</isReadonly>
  <isUpdateBlocked>false</isUpdateBlocked>
  <isName xsi:nil="true"/>
  <isPrimaryKey>false</isPrimaryKey>
  <isCustom>false</isCustom>
  <isDynamic>false</isDynamic>
  <dynamicFieldRef xsi:nil="true"/>
  <updatedAt>2011-08-24T18:39:07-07:00</updatedAt>
</field>
<field>
  <name>LastActivityDate</name>
  <description xsi:nil="true"/>
  <displayName>Last Activity Date</displayName>
  <sourceObject>Opportunity</sourceObject>
  <dataType>date</dataType>
  <size xsi:nil="true"/>
  <isReadonly>false</isReadonly>
  <isUpdateBlocked>false</isUpdateBlocked>
  <isName xsi:nil="true"/>
  <isPrimaryKey>false</isPrimaryKey>
  <isCustom>false</isCustom>
  <isDynamic>false</isDynamic>
  <dynamicFieldRef xsi:nil="true"/>
  <updatedAt>2011-08-24T18:39:07-07:00</updatedAt>
</field>
<field>
  <name>LeadSource</name>
  <description xsi:nil="true"/>
  <displayName>Lead Source</displayName>
  <sourceObject>Opportunity</sourceObject>
  <dataType>string</dataType>
  <size>255</size>
  <isReadonly>false</isReadonly>
  <isUpdateBlocked>false</isUpdateBlocked>
  <isName xsi:nil="true"/>
  <isPrimaryKey>false</isPrimaryKey>
  <isCustom>false</isCustom>
  <isDynamic>false</isDynamic>
  <dynamicFieldRef xsi:nil="true"/>
  <updatedAt>2011-08-24T18:49:29-07:00</updatedAt>

```

```

</field>
<field>
  <name>Name</name>
  <description xsi:nil="true"/>
  <displayName>Name</displayName>
  <sourceObject>Opportunity</sourceObject>
  <dataType>string</dataType>
  <size>255</size>
  <isReadonly>false</isReadonly>
  <isUpdateBlocked>false</isUpdateBlocked>
  <isName xsi:nil="true"/>
  <isPrimaryKey>false</isPrimaryKey>
  <isCustom>false</isCustom>
  <isDynamic>false</isDynamic>
  <dynamicFieldRef xsi:nil="true"/>
  <updatedAt>2011-08-24T18:39:07-07:00</updatedAt>
</field>
<field>
  <name>NextStep</name>
  <description xsi:nil="true"/>
  <displayName>Next Step</displayName>
  <sourceObject>Opportunity</sourceObject>
  <dataType>string</dataType>
  <size>255</size>
  <isReadonly>false</isReadonly>
  <isUpdateBlocked>false</isUpdateBlocked>
  <isName xsi:nil="true"/>
  <isPrimaryKey>false</isPrimaryKey>
  <isCustom>false</isCustom>
  <isDynamic>false</isDynamic>
  <dynamicFieldRef xsi:nil="true"/>
  <updatedAt>2011-08-24T18:39:07-07:00</updatedAt>
</field>
<field>
  <name>Probability</name>
  <description xsi:nil="true"/>
  <displayName>Probability</displayName>
  <sourceObject>Opportunity</sourceObject>
  <dataType>integer</dataType>
  <size xsi:nil="true"/>
  <isReadonly>false</isReadonly>
  <isUpdateBlocked>false</isUpdateBlocked>
  <isName xsi:nil="true"/>
  <isPrimaryKey>false</isPrimaryKey>
  <isCustom>false</isCustom>
  <isDynamic>false</isDynamic>
  <dynamicFieldRef xsi:nil="true"/>

```

```

    <updatedAt>2011-08-24T18:39:07-07:00</updatedAt>
  </field>
  <field>
    <name>Quantity</name>
    <description xsi:nil="true"/>
    <displayName>Quantity</displayName>
    <sourceObject>Opportunity</sourceObject>
    <dataType>float</dataType>
    <size xsi:nil="true"/>
    <isReadonly>false</isReadonly>
    <isUpdateBlocked>false</isUpdateBlocked>
    <isName xsi:nil="true"/>
    <isPrimaryKey>false</isPrimaryKey>
    <isCustom>false</isCustom>
    <isDynamic>false</isDynamic>
    <dynamicFieldRef xsi:nil="true"/>
    <updatedAt>2011-08-24T18:39:07-07:00</updatedAt>
  </field>
  <field>
    <name>Stage</name>
    <description xsi:nil="true"/>
    <displayName>Stage</displayName>
    <sourceObject>Opportunity</sourceObject>
    <dataType>string</dataType>
    <size>255</size>
    <isReadonly>false</isReadonly>
    <isUpdateBlocked>false</isUpdateBlocked>
    <isName xsi:nil="true"/>
    <isPrimaryKey>false</isPrimaryKey>
    <isCustom>false</isCustom>
    <isDynamic>false</isDynamic>
    <dynamicFieldRef xsi:nil="true"/>
    <updatedAt>2011-08-24T18:49:29-07:00</updatedAt>
  </field>
  <field>
    <name>Type</name>
    <description xsi:nil="true"/>
    <displayName>Type</displayName>
    <sourceObject>Opportunity</sourceObject>
    <dataType>string</dataType>
    <size>255</size>
    <isReadonly>false</isReadonly>
    <isUpdateBlocked>false</isUpdateBlocked>
    <isName xsi:nil="true"/>
    <isPrimaryKey>false</isPrimaryKey>
    <isCustom>false</isCustom>
    <isDynamic>false</isDynamic>

```

```
        <dynamicFieldRef xsi:nil="true"/>
        <updatedAt>2011-08-24T18:49:29-07:00</updatedAt>
    </field>
</fieldList>
    <updatedAt>2011-09-02T11:55:45-07:00</updatedAt>
</metadata>
</result>
</ns1:successDescribeMObject>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

## OpportunityPersonRole Metadata

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-
ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:ns1="http://www.marketo.com/mktows/">
  <SOAP-ENV:Body>
    <ns1:successDescribeMObject>
      <result>
        <metadata>
          <name>OpportunityPersonRole</name>
          <description>Marketo Opportunity Person Role</description>
          <isCustom>>false</isCustom>
          <isVirtual>>false</isVirtual>
          <fieldList>
            <field>
              <name>ExternalCreatedDate</name>
              <description xsi:nil="true"/>
              <displayName>SFDC Created Date</displayName>
              <sourceObject>OpportunityPersonRole</sourceObject>
              <dataType>datetime</dataType>
              <size xsi:nil="true"/>
              <isReadOnly>>false</isReadOnly>
              <isUpdateBlocked>>false</isUpdateBlocked>
              <isName xsi:nil="true"/>
              <isPrimaryKey>>false</isPrimaryKey>
              <isCustom>>false</isCustom>
              <isDynamic>>false</isDynamic>
              <dynamicFieldRef xsi:nil="true"/>
              <updatedAt>2011-08-24T18:39:07-07:00</updatedAt>
            </field>
            <field>
              <name>Id</name>
              <description xsi:nil="true"/>
              <displayName>Id</displayName>
              <sourceObject>OpportunityPersonRole</sourceObject>
              <dataType>integer</dataType>
              <size xsi:nil="true"/>
              <isReadOnly>>false</isReadOnly>
              <isUpdateBlocked>>false</isUpdateBlocked>
              <isName xsi:nil="true"/>
              <isPrimaryKey>>false</isPrimaryKey>
              <isCustom>>false</isCustom>
              <isDynamic>>true</isDynamic>
              <dynamicFieldRef>attributeList</dynamicFieldRef>
              <updatedAt>2011-08-24T18:39:07-07:00</updatedAt>
            </field>
            <field>
              <name>IsPrimary</name>
              <description xsi:nil="true"/>
              <displayName>Is Primary</displayName>
              <sourceObject>OpportunityPersonRole</sourceObject>
              <dataType>boolean</dataType>
              <size xsi:nil="true"/>
              <isReadOnly>>false</isReadOnly>
            </field>
          </fieldList>
        </metadata>
      </result>
    </ns1:successDescribeMObject>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

```

    <isUpdateBlocked>>false</isUpdateBlocked>
    <isName xsi:nil="true"/>
    <isPrimaryKey>>false</isPrimaryKey>
    <isCustom>>false</isCustom>
    <isDynamic>>false</isDynamic>
    <dynamicFieldRef xsi:nil="true"/>
    <updatedAt>2011-08-24T18:39:07-07:00</updatedAt>
  </field>
  <field>
    <name>OpportunityId</name>
    <description xsi:nil="true"/>
    <displayName>Opportunity Id</displayName>
    <sourceObject>OpportunityPersonRole</sourceObject>
    <dataType>reference</dataType>
    <size xsi:nil="true"/>
    <isReadonly>>false</isReadonly>
    <isUpdateBlocked>>false</isUpdateBlocked>
    <isName xsi:nil="true"/>
    <isPrimaryKey>>false</isPrimaryKey>
    <isCustom>>false</isCustom>
    <isDynamic>>false</isDynamic>
    <dynamicFieldRef xsi:nil="true"/>
    <updatedAt>2011-08-24T18:39:07-07:00</updatedAt>
  </field>
  <field>
    <name>PersonId</name>
    <description xsi:nil="true"/>
    <displayName>Contact Id</displayName>
    <sourceObject>OpportunityPersonRole</sourceObject>
    <dataType>reference</dataType>
    <size xsi:nil="true"/>
    <isReadonly>>false</isReadonly>
    <isUpdateBlocked>>false</isUpdateBlocked>
    <isName xsi:nil="true"/>
    <isPrimaryKey>>false</isPrimaryKey>
    <isCustom>>false</isCustom>
    <isDynamic>>false</isDynamic>
    <dynamicFieldRef xsi:nil="true"/>
    <updatedAt>2011-08-24T18:39:07-07:00</updatedAt>
  </field>
  <field>
    <name>Role</name>
    <description xsi:nil="true"/>
    <displayName>Role</displayName>
    <sourceObject>OpportunityPersonRole</sourceObject>
    <dataType>string</dataType>
    <size>255</size>
    <isReadonly>>false</isReadonly>
    <isUpdateBlocked>>false</isUpdateBlocked>
    <isName xsi:nil="true"/>
    <isPrimaryKey>>false</isPrimaryKey>
    <isCustom>>false</isCustom>
    <isDynamic>>false</isDynamic>
    <dynamicFieldRef xsi:nil="true"/>
    <updatedAt>2011-08-24T18:49:33-07:00</updatedAt>
  </field>

```

```
        </fieldList>
        <updatedAt>2011-08-24T18:49:33-07:00</updatedAt>
    </metadata>
</result>
</ns1:successDescribeMObject>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```