Plan Plus Online API Implementers Guide using PHP

by Jeremy Zerr <jeremy@zerrtech.com>

Overview

To be able to use the Plan Plus Online API from most common web development platforms requires being able to interact with the API using PHP. This document describes how to use the Plan Plus Online API using PHP and includes code samples and an explanation of how the code works in order to accomplish this. I will show a sample flow where we use an email address to find a contact that already exists in the CRM, and we are able to retrieve information about the contact, and change both custom and non-custom fields for that contact.

Pre-requisites

The pre-requisites listed below are needed to create the code and system to be able to interface with the Plan Plus Online API using PHP.

NuSOAP

In order to allow the translation of a WSDL file into the appropriate PHP class framework to allow interfacing with the API, the open source project NuSOAP was used.
NuSOAP project website:
http://sourceforge.net/projects/nusoap/

For more information on how to use NuSOAP in your project, download the source from the URL above and look at the examples in the "samples/" subdirectory.

Existing Plan Plus Online API documentation

Plan Plus Online supplied documentation at:
http://216.130.241.141/wsdoc/web-services-howto-1.jsp

API Overview

The Plan Plus Online API is defined by WSDL files. The only 3 documented APIs are for the following data structures: Authentication, Organization, Contact

Authentication

WSDL file location: http://planplusonline02.com/cxf/SessionAuthenticatorAPI?wsdl

The purpose of the Authentication API is to provide you the security credentials necessary to use the other APIs that interact with data stored in the Plan Plus Online CRM. There are a couple different authentication methods, but the one that I use in this example is the Company Name, Username, Password method.

```
<?php
// Change companyName, username, and password to match your login
credentials
$companyName = "YCCRM";
$username = "myUsername";
$password = "myPassword";

// For most, these proxy variables don't need to be used
```

```
$proxyhost = '';
$proxyport = '';
$proxyusername = '';
$proxypassword = '';
$timeout = 360;
$response_timeout = 360;

/* Authenticate */
$client = new nusoap_client('http://planplusonline02.com/cxf/
SessionAuthenticatorAPI?wsdl', 'wsdl',
                        $proxyhost, $proxyport, $proxyusername,
$proxypassword, $timeout, $response_timeout);
$err = $client->getError();
if ($err) {
    echo '<h2>Constructor error</h2><pre>' . $err . '</pre>';
}
$authenticateData = array("arg0" => $companyName,"arg1" => $username,"arg2"
=> $password);
$result = $client->call('loginAuthenticate', array("parameters" =>
$authenticateData));

/* Extract the session ID from the authentication response */
$fullSessionString = $result["return"];
$sessId = substr($fullSessionString,10);
echo "session ID: $sessId";
?>
```

Just change the $companyName, $username, $password variables to your actual login parameters to get the code to work. In most cases, a proxy isn't needed, but the proxy variables can be used if needed.

In this code, the result is that we have a session ID returned from the Authentication API that we can use to authenticate all subsequent API calls. This session ID is stored in the $sessId variable that results from the above code being run.

Organization

WSDL file location: http://planplusonline02.com/cxf/OrgAPI?wsdl

The Organization API can be used to get information about Organizations. There is only one brief example with the Organization API, because there wasn't a need to use Organizations with the particular project that required using the API in PHP. The method used is "searchOrganizations".

```
<?php
/* Query by organization */
$organizationName = "REIC";
// session ID $sessId is defined by running Authentication API

$orgclient = new nusoap_client('http://planplusonline02.com/cxf/
OrgAPI?wsdl', 'wsdl',
                        $proxyhost, $proxyport, $proxyusername,
$proxypassword, $timeout, $response_timeout);
$err = $orgclient->getError();
if ($err) {
    echo '<h2>Constructor error</h2><pre>' . $err . '</pre>';
}
$organizationQuery = array("arg0" => $sessId, "arg1" => $organizationName);
$result = $orgclient->call('searchOrganizations', array("parameters" =>
$organizationQuery));

printResult($orgclient,$result);
?>
```

The function printResult is a special print function that will be detailed in the full code that is included at the end of this document.

Contact

WSDL file location: http://planplusonline02.com/cxf/PsnAPI?wsdl

The first example will be how to find a contact using the method "searchContacts". You can use all non-custom fields to search by, and you will get returned all contacts that match the search criteria. For each contact returned, the customerId is the important field, which is the primary key we will use to make other API calls.

```
<?php
/* Query by person */
// email address to search for contact by
$searchCriteria = "joeblow@example.com";
$psnclient = new nusoap_client('http://planplusonline02.com/cxf/
PsnAPI?wsdl', 'wsdl',
                        $proxyhost, $proxyport, $proxyusername,
$proxypassword, $timeout, $response_timeout);
$err = $psnclient->getError();
if ($err) {
    echo '<h2>Constructor error</h2><pre>' . $err . '</pre>';
}
$searchPersonQuery = array("arg0" => $sessId, "arg1" => $searchCriteria);
$result = $psnclient->call('searchContacts', array("parameters" =>
$searchPersonQuery));

if(isset($result["return"]["list"]["customerId"]) > 0) {
  $customerId = $result["return"]["list"]["customerId"];
  $firstName = $result["return"]["list"]["firstName"];
  $lastName = $result["return"]["list"]["lastName"];
  echo "Customer Id is:$customerId";
}
else {
  echo "No customer Id exists";
  exit;
}
```

```
?>
```

Now we will use the customer ID to get detailed information about the contact, which will allow us to see all of the custom and non-custom fields. We use the method "getContactDetail" for this.

```
<?php
/* get the customer detail */
$getDetailQuery = array("arg0" => $sessId, "arg1" => (int)$customerId);

$result = $psnclient->call('getContactDetail', array("parameters" => $getDetailQuery));

printResult($psnclient,$result);
?>
```

Again, the printResult method is included in the full code at the end of this document.

Now I will show an example of how to change a non-custom field first, and a custom field second. We do both of these by using the "saveContactRecord" method.

First, how to change a non-custom field.

```
<?php
/* change a non-custom field for the contact */
$newURL = "http://www.example4.com";

$changeContact = array("customerId" => (int)$customerId, "firstName" =>
$firstName, "lastName" => $lastName, "url" => $newURL);

$changeDetailQuery = array("arg0" => $sessId, "arg1" => $changeContact);

$result = $psnclient->call('saveContactRecord', array("parameters" =>
$changeDetailQuery));

printResult($psnclient,$result);
?>
```

Now, how to change a custom field. When changing a custom field, there are 2 cases, one is that the custom field has no value yet, and the other is that the custom field already has a value. The data sent to the API needs to be different in each of these cases. Basically, if you do it wrong, you can cause duplicate custom field records within the CRM. The code below handles both cases without any problems.

```
<?php
/* now change the company name of the customer */
$customFieldToChange = "Company Name";
$newCustomFieldData = "ACME Hyper-global";

// search for this field to find the fieldDefinitionId and if data already
```

```
exists
foreach ($result["return"]["customFields"] as $oCustomField) {
  if($oCustomField["fieldDefinition"] == $customFieldToChange) {
    $fieldDefinitionId = $oCustomField["fieldDefinitionId"];
    $cfDataId = $oCustomField["cfDataId"];
    break;
  }
}
if(!isset($fieldDefinitionId)) {
  echo "Could not find custom field $customFieldToChange, exiting...";
  exit;
}

// create the data structure for the upload
$changeCustomFields = array("fieldDefinitionId" => (int)$fieldDefinitionId,
"cfDataValue" => $newCustomFieldData, "olLastUpdateDate" =>
"2010-03-11T23:00:00.000-07:00", "cfDataId" => (int)$cfDataId);

$changeContact = array("customerId" => (int)$customerId, "firstName" =>
$firstName, "lastName" => $lastName, "customFields" =>
$changeCustomFields);

$changeDetailQuery = array("arg0" => $sessId, "arg1" => $changeContact);
$result = $psnclient->call('saveContactRecord', array("parameters" =>
$changeDetailQuery));
?>
```

This starts by using the desired custom field name to change to find the corresponding fieldDefinitionId that is an integer that represents that field, which is really what is used to make the change. The cfDataId field is also pulled to allow changing a custom field that already has a value. If there was no previous value for the field, the cfDataId will be -1, otherwise if it does contain a value, cfDataId will contain a positive integer.

The "olLastUpdateDate" field is required to be something, because if it is left out, or left as -1, the server will not update with the values you are sending. So it is filled in with a dummy time record here to force an update to happen. It is a date that is in the format of the ISO 8601 format: *CCYY-MM-DD*T*hh*:*mm*:*ss*
This can be something like:
1969-07-20T21:28:00-06:00
or
1969-07-21T02:28:00Z

It doesn't appear that the date used is actually significant, just as long as its in an accepted format and is included in the data.

Entire Code Example

Here is a complete listing of the code used, and includes all of the examples above integrated into one single script.

```
<?php
// this should point to wherever you have nusoap installed
```

```
require_once('../lib/nusoap.php');

// Change companyName, username, and password to match your login
credentials
$companyName = "YCCRM";
$username = "myUsername";
$password = "myPassword";

// For most, these proxy variables don't need to be used
$proxyhost = '';
$proxyport = '';
$proxyusername = '';
$proxypassword = '';
$timeout = 360;
$response_timeout = 360;

/* Authenticate */

$client = new nusoap_client('http://planplusonline02.com/cxf/
SessionAuthenticatorAPI?wsdl', 'wsdl',
                        $proxyhost, $proxyport, $proxyusername,
$proxypassword, $timeout, $response_timeout);
$err = $client->getError();
if ($err) {
    echo '<h2>Constructor error</h2><pre>' . $err . '</pre>';
}
$authenticateData = array("arg0" => $companyName,"arg1" => $username,"arg2"
=> $password);
$result = $client->call('loginAuthenticate', array("parameters" =>
$authenticateData));

/* Extract the session ID from the authentication response */
$fullSessionString = $result["return"];
$sessId = substr($fullSessionString,10);
echo "real session: $sessId";

/* Query by organization */
$organizationName = "REIC";
$orgclient = new nusoap_client('http://planplusonline02.com/cxf/
OrgAPI?wsdl', 'wsdl',
                        $proxyhost, $proxyport, $proxyusername,
$proxypassword, $timeout, $response_timeout);
$err = $orgclient->getError();
if ($err) {
    echo '<h2>Constructor error</h2><pre>' . $err . '</pre>';
}
$organizationQuery = array("arg0" => $sessId, "arg1" => $organizationName);
$result = $orgclient->call('searchOrganizations', array("parameters" =>
$organizationQuery));

printResult($orgclient,$result);
```

```
/* Query by person */
// email address to search for contact by
$searchCriteria = "joeblow@example.com";
$psnclient = new nusoap_client('http://planplusonline02.com/cxf/
PsnAPI?wsdl', 'wsdl',
                        $proxyhost, $proxyport, $proxyusername,
$proxypassword, $timeout, $response_timeout);
$err = $psnclient->getError();
if ($err) {
    echo '<h2>Constructor error</h2><pre>' . $err . '</pre>';
}
$searchPersonQuery = array("arg0" => $sessId, "arg1" => $searchCriteria);
$result = $psnclient->call('searchContacts', array("parameters" =>
$searchPersonQuery));

if(isset($result["return"]["list"]["customerId"]) > 0) {
  $customerId = $result["return"]["list"]["customerId"];
  $firstName = $result["return"]["list"]["firstName"];
  $lastName = $result["return"]["list"]["lastName"];
  echo "Customer Id is:$customerId";
}
else {
  echo "No customer Id exists";
  exit;
}

/* get the customer detail */
$getDetailQuery = array("arg0" => $sessId, "arg1" => (int)$customerId);

$result = $psnclient->call('getContactDetail', array("parameters" =>
$getDetailQuery));

printResult($psnclient,$result);

/* change a non-custom field for the contact */
$newURL = "http://www.example4.com";

$changeContact = array("customerId" => (int)$customerId, "firstName" =>
$firstName, "lastName" => $lastName, "url" => $newURL);

$changeDetailQuery = array("arg0" => $sessId, "arg1" => $changeContact);

$result = $psnclient->call('saveContactRecord', array("parameters" =>
$changeDetailQuery));
printResult($psnclient,$result);

/* now change the company name of the customer */
$customFieldToChange = "Company Name";
$newCustomFieldData = "ACME Hyper-global";

// search for this field to find the fieldDefinitionId and if data already
exists
```

```
foreach ($result["return"]["customFields"] as $oCustomField) {
  if($oCustomField["fieldDefinition"] == $customFieldToChange) {
    $fieldDefinitionId = $oCustomField["fieldDefinitionId"];
    $cfDataId = $oCustomField["cfDataId"];
    break;
  }
}
if(!isset($fieldDefinitionId)) {
  echo "Could not find custom field $customFieldToChange, exiting...";
  exit;
}

// create the data structure for the upload
$changeCustomFields = array("fieldDefinitionId" => (int)$fieldDefinitionId,
"cfDataValue" => $newCustomFieldData, "olLastUpdateDate" =>
"2010-03-11T23:00:00.000-07:00", "cfDataId" => (int)$cfDataId);

$changeContact = array("customerId" => (int)$customerId, "firstName" =>
$firstName, "lastName" => $lastName, "customFields" =>
$changeCustomFields);

$changeDetailQuery = array("arg0" => $sessId, "arg1" => $changeContact);
$result = $psnclient->call('saveContactRecord', array("parameters" =>
$changeDetailQuery));

printResult($psnclient,$result);

function printResult(&$client,&$result) {
  if ($client->fault) {
      echo '<h2>Fault</h2><pre>';
      print_r($result);
      echo '</pre>';
  } else {
      // Check for errors
      $err = $client->getError();
      if ($err) {
          // Display the error
          echo '<h2>Error</h2><pre>' . $err . '</pre>';
      } else {
          // Display the result
          echo '<h2>Result</h2><pre>';
          print_r($result);
          echo '</pre>';
      }
  }
}

function printDebug(&$client) {
  echo '<h2>Request</h2><pre>' . htmlspecialchars($client->request,
ENT_QUOTES) . '</pre>';
  echo '<h2>Response</h2><pre>' . htmlspecialchars($client->response,
ENT_QUOTES) . '</pre>';
```

```
  //echo '<h2>Debug</h2><pre>' . htmlspecialchars($client->debug_str,
ENT_QUOTES) . '</pre>';
}
?>
```