

Visitor Portal API Guide v1.0.0.0

INTEGRATORS MANUAL

IMPRO TECHNOLOGIES

Table of Contents

| | |
|---------------------------------------|---|
| Document Control..... | 2 |
| Revision Control..... | 2 |
| Visitor Portal API Requirements | 3 |
| API Sample Kit | 3 |
| API Usage Guide..... | 5 |

The information contained in this document is confidential to Impro technologies (Pty) Ltd and/or its suppliers and Agents. Such information is made available subject to the conditions that, 1 it may not be disclosed to third parties other than employees or consultants who need to have access thereto for the purposes of evaluating Impro Technologies (Pty) Ltd.'s response and negotiating any agreement that may arise from it. You may use the information only for such evaluation and negotiation purposes.

Dated: 03/03/2015
UP SLA: V1.0

Document Control

| | |
|-------------------------------------|--|
| Item | Visitor Portal API Guide v1.0.0.0 |
| Impro Technologies Reference Number | VPUM_02_2015 |
| Authors: | Clive van Eeden |
| Security Level: | Confidential – Not for redistribution without consent. |

Revision Control

| Revision | Dated | Revised By | Description of Major Changes |
|----------|------------|-----------------|------------------------------|
| 1.0.0.0 | 2015/03/03 | Clive van Eeden | vPortal API Guide |
| | | | |
| | | | |
| | | | |

The information contained in this document is confidential to Impro technologies (Pty) Ltd and/or its suppliers and Agents. Such information is made available subject to the conditions that, 1 it may not be disclosed to third parties other than employees or consultants who need to have access thereto for the purposes of evaluating Impro Technologies (Pty) Ltd.'s response and negotiating any agreement that may arise from it. You may use the information only for such evaluation and negotiation purposes.

Dated: 03/03/2015
UP SLA: V1.0

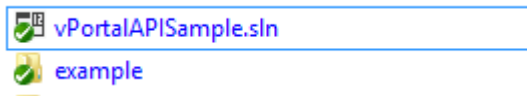
Visitor Portal API Requirements

To communicate with Visitor Portal via the API class library there is a few requirements that you need and should ensure you have before developing/testing/setting up a client:

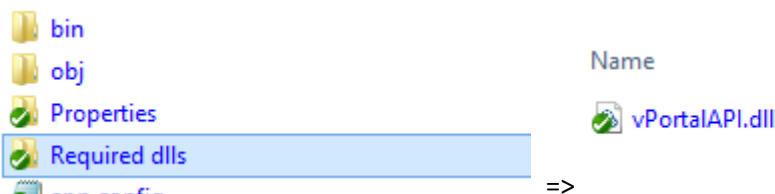
1. You will need a Visitor Portal integration token, username and password per Site/Instance. This would require Impro Technologies to create your client's Site and Login with tokens on the Visitor Portal System before you can integrate them. We will supply you with these details and you in turn will configure them accordingly for your client.
2. To have Visitor Portal communicate with your Access Portal onsite you will need to install the Visitor Portal Client Service, which you can use the same Login and token for connecting.
3. The Visitor Portal Client Service communicates to Access Portal's API, this API requires that you either have an Enterprise License, which has the API License built in or you will need a Pro License with a separate API License.

API Sample Kit

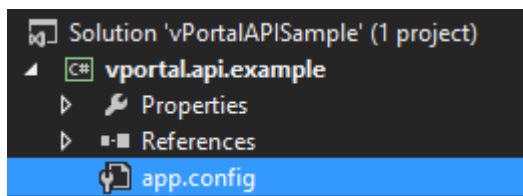
The sample API Kit is a Visual Studio 2013 project (recommend using Community edition) named 'vPortalAPISample.sln', and all code for project resides in the 'example' folder:



The Visitor Portal API class library called 'vPortalAPI.dll' resides in the 'Required dll' folder and must remain there as it is referenced from that folder within the project:



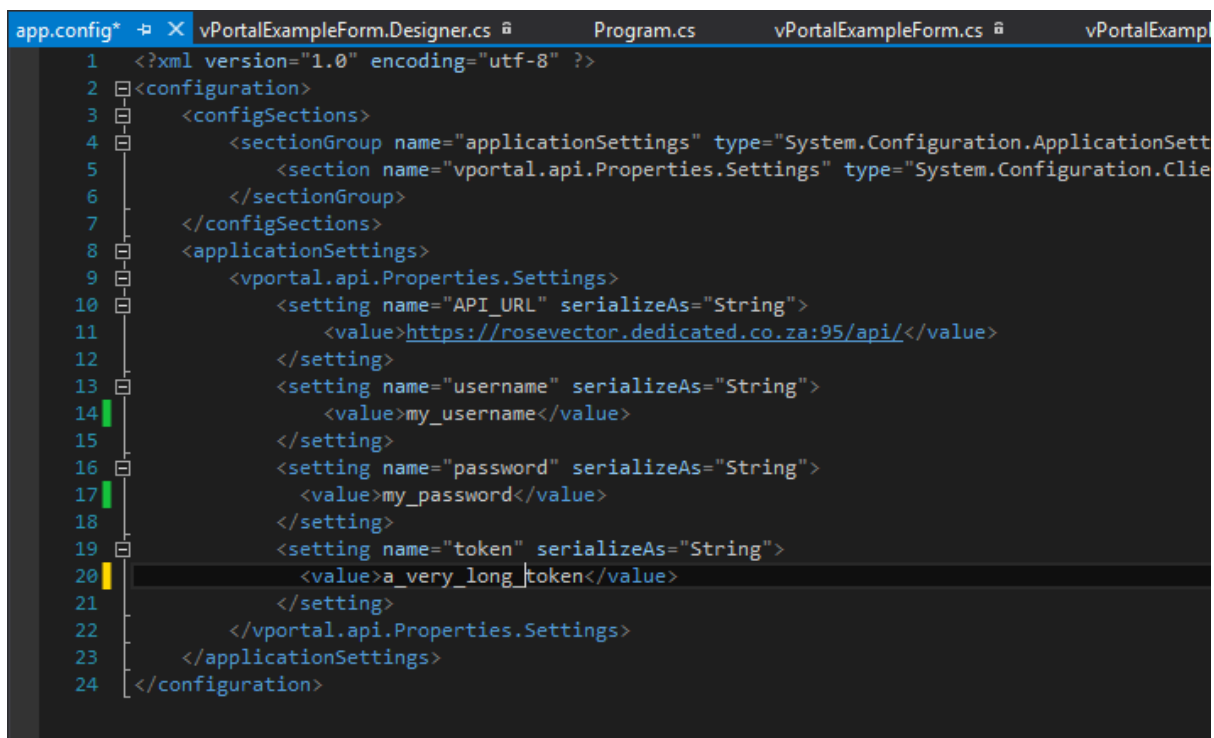
For testing and compiling, input your username, password, token in the app.config file within your project:



The information contained in this document is confidential to Impro technologies (Pty) Ltd and/or its suppliers and Agents. Such information is made available subject to the conditions that, 1 it may not be disclosed to third parties other than employees or consultants who need to have access thereto for the purposes of evaluating Impro Technologies (Pty) Ltd.'s response and negotiating any agreement that may arise from it. You may use the information only for such evaluation and negotiation purposes.

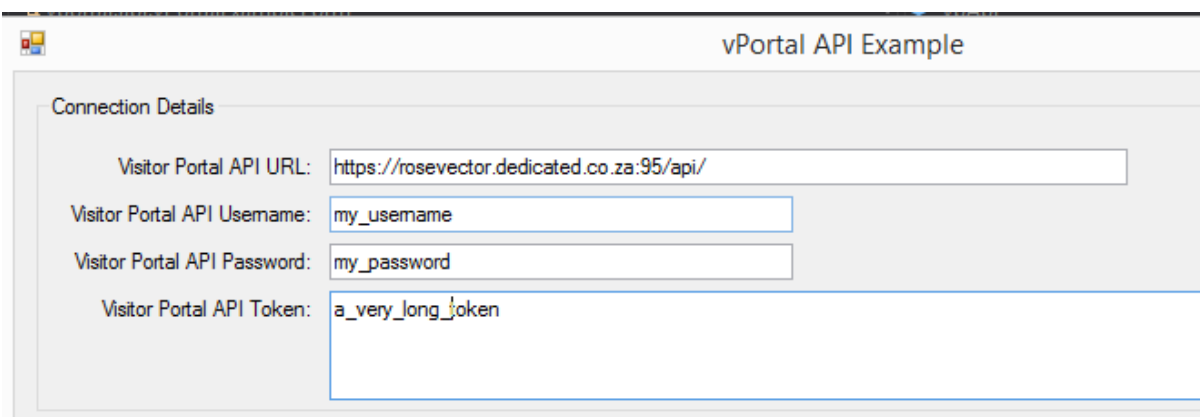
Dated: 03/03/2015
UP SLA: V1.0

The layout looks like so:



```
1 <?xml version="1.0" encoding="utf-8" ?>
2 <configuration>
3   <configSections>
4     <sectionGroup name="applicationSettings" type="System.Configuration.ApplicationSett
5     <section name="vportal.api.Properties.Settings" type="System.Configuration.Clie
6   </sectionGroup>
7 </configSections>
8 <applicationSettings>
9   <vportal.api.Properties.Settings>
10    <setting name="API_URL" serializeAs="String">
11      <value>https://rosevector.dedicated.co.za:95/api/</value>
12    </setting>
13    <setting name="username" serializeAs="String">
14      <value>my_username</value>
15    </setting>
16    <setting name="password" serializeAs="String">
17      <value>my_password</value>
18    </setting>
19    <setting name="token" serializeAs="String">
20      <value>a_very_long_token</value>
21    </setting>
22  </vportal.api.Properties.Settings>
23 </applicationSettings>
24 </configuration>
```

You will see the API_URL value will already be populated correctly for you as it is the current URL address for Visitor Portal. But you will obviously have to change the username, password and token values with the ones provided to you. These values populate the textboxes on form load and reference for the API calls:



vPortal API Example

Connection Details

Visitor Portal API URL:

Visitor Portal API Username:

Visitor Portal API Password:

Visitor Portal API Token:

The information contained in this document is confidential to Impro technologies (Pty) Ltd and/or its suppliers and Agents. Such information is made available subject to the conditions that, 1 it may not be disclosed to third parties other than employees or consultants who need to have access thereto for the purposes of evaluating Impro Technologies (Pty) Ltd.'s response and negotiating any agreement that may arise from it. You may use the information only for such evaluation and negotiation purposes.

Dated: 03/03/2015
UP SLA: V1.0

API Usage Guide

Before making any calls to the API, you need to declare its usage:

```
using System.Configuration;
using vportal.api.vpapi;

namespace vportal.api
{
    public partial class vPortalExampleForm : Form
    {
    }
```

And also declare an instance of it within your class/form:

```
public partial class vPortalExampleForm : Form
{
    //Create an instance of the vPortal API class
    public vPortalAPI vpApi;
}
```

And in your Form_Load event assign your details:

```
private void vPortalExampleForm_Load(object sender, EventArgs e)
{
    apiURLtextBox.Text = vportal.api.Properties.Settings.Default.API_URL.ToString();
    apiUSERtextBox.Text = vportal.api.Properties.Settings.Default.username.ToString();
    apiPSWDtextBox.Text = vportal.api.Properties.Settings.Default.password.ToString();
    apiTOKENtextBox.Text = vportal.api.Properties.Settings.Default.token.ToString();

    vpApi = new vPortalAPI("My App", true, apiURLtextBox.Text, apiUSERtextBox.Text, apiPSWDtextBox.Text, apiTOKENtextBox.Text);
}
```

There are 4 main constructors at this point for retrieving and inserting/updating Master data:

1. vPortalGetAllMasterData:

```
public string vPortalGetAllMasterData(bool NewRecordsOnly);
```

This will return all MASTER and linked MASTER_DETAIL data for the Site that the Login is linked to. There is a Boolean value 'NewRecordsOnly' which, if set to true, will only return new inserted MASTER records that have an MST_FLAG of 0 (meaning they have not been synchronised to Access Portal yet). Whereas if you pass a false here it returns the entire Site list of MASTER records.

Project example for this shown under:

```
private void GetNewMasterRecordsButton_Click(object sender, EventArgs e)
```

2. vPortalGetSpecificMaster:

```
public string vPortalGetSpecificMaster(int CriteriaType, string SearchCriteria);
```

Here you pass some criteria to only return the record/s you require by specifying a criteria type (Search by Firstname, ID Number etc.) and the Search criteria.

There is a predefined list of Criteria types setup in the API, and the sample will show you how to use the values assigned for each type and populate those values to a combobox for selection.

API method for this is:

```
public IList GetSearchCriteriaValues();
```

The usage of this method to populate the combobox is shown in the Form_Load event and must occur after the vpApi assignment:

```
private void vPortalExampleForm_Load(object sender, EventArgs e)
{
    apiURLtextBox.Text = vportal.api.Properties.Settings.Default.API_URL.ToString();
    apiUSERtextBox.Text = vportal.api.Properties.Settings.Default.username.ToString();
    apiPSWDtextBox.Text = vportal.api.Properties.Settings.Default.password.ToString();
    apiTOKENtextBox.Text = vportal.api.Properties.Settings.Default.token.ToString();

    vpApi = new vPortalAPI("My App", true, apiURLtextBox.Text, apiUSERtextBox.Text, apiPSWDtextBox.Text, apiTOKENtextBox.Text);

    //We have an enumerator in place to return Search Criteria values to populate your combobox
    //you may use your own criteria names, but refer to the vPortal API documentation for the correct integer to pass back
    //to the API's constructor. if this is not correct, then no data will be return (or possibly wrong data because the search c
    //Here is an example of populating your combobox with our Search Values:
    this.SearchCriteriaComboBox.DataSource = vpApi.GetSearchCriteriaValues();
    this.SearchCriteriaComboBox.ValueMember = "Key";
    this.SearchCriteriaComboBox.DisplayMember = "Value";
}
```

Each type returns an integer value, so before calling the 'vPortalGetSpecificMaster' method, you must get the 'CriteriaType' value from the combobox.

The API method for this is:

```
public int GetSearchCriteriaType(string SearchCriteria);
```

The usage of this method to get the integer value is done like so:

```
int CriteriaType = vpApi.GetSearchCriteriaType(SearchCriteriaComboBox.Text);
```

Project example for this shown under:

```
private void GetSpecificMasterRecordButton_Click(object sender, EventArgs e)
{
    //get specific MASTER record/s from visitor portal based on criteria
}
```

3. vPortalInsertMasterData

```
public string vPortalInsertMasterData(string jsonMasterData);
```

Before calling this method, you need to build up the JSON model data, for this you will call the API's MASTER model class and assign your values to the MASTER class and build it up that way:

```
//Assign your values to API MASTER class Model
MASTER am = new MASTER
{
    MASTER_ID = 0,
    MST_DISPLAYNAME = DisplayNameTextBox.Text,
    MST_TITLE = TitleComboBox.Text,
    MST_FIRSTNAME = FirstNameTextBox.Text,
    MST_MIDDLENAME = MiddleNameTextBox.Text,
    MST_LASTNAME = LastNameTextBox.Text,
    MST_IDNUMBER = IDNumberTextBox.Text,
    MST_GENDER = gender,
    MST_CURRENT = 1,

    //For MASTER_DETAIL data if you need to parse in your code whether you need this or
    not.
    //if there is no data for MASTER_DETAIL you can leave this part out completely.
    MASTER_DETAIL = new List<MASTERDETAIL>()
    {
        //For each item (email, mobile, telephone etc.) add a comma then new MASTERDETAIL()...
        new MASTERDETAIL()
        { MD_VALBINARY = null, MASTER_DETAIL_ID = 0, MD_VALSTRING = EmailTextBox.Text,
          MASTER_DETAIL_TYPE_ID = 24, MASTER_ID = 0 },
        new MASTERDETAIL()
        { MD_VALBINARY = null, MASTER_DETAIL_ID = 0, MD_VALSTRING = CelltextBox.Text,
          MASTER_DETAIL_TYPE_ID = 23, MASTER_ID = 0 }
    }
}
```

It is important to note for the MASTERDETAIL class that you may have more than one item (Mobile number, Email etc.), so for every item you will add a comma at the end and put the next item in. Visitor Portal only functionally uses Email (MASTER_DETAIL_TYPE_ID = 24) and Mobile Number (MASTER_DETAIL_TYPE_ID = 23) for sending communications to Visitors, but any extras you wish to populate will go through to Access Portal. A full list of the types can be found in the PORTAL database in the table MASTER_DETAIL_TYPE. However, custom added types in Portal will not be synced.

Once the MASTER has been built up you will string it all together into JSON format using the BuildMasterInsertUpdate method before calling vPortalInsertMasterData:

```
public string BuildMasterInsertUpdate(object masterRecord);
```

The usage of this method below, assign it to a string:

```
string jsonreturn = vpApi.BuildMasterInsertUpdate(am);
```

The information contained in this document is confidential to Impro technologies (Pty) Ltd and/or its suppliers and Agents. Such information is made available subject to the conditions that, 1 it may not be disclosed to third parties other than employees or consultants who need to have access thereto for the purposes of evaluating Impro Technologies (Pty) Ltd.'s response and negotiating any agreement that may arise from it. You may use the information only for such evaluation and negotiation purposes.

Dated: 03/03/2015
UP SLA: V1.0

4. vPortalUpdateMasterData

```
public string vPortalUpdateMasterData(string jsonMasterData);
```

This is an Update, so it will require you to know the MASTER_ID of the visitor before posting the data. You can use the search Master method found in point 2. And the build-up is done exactly the same the vPortalInsertMasterData method in point 3, with the exception that you will call vPortalUpdateMasterData instead:

```
string result = vpApi.vPortalUpdateMasterData(jsonreturn);
```