

vtlib – vtiger development library
version 1.4

DISCLAIMER:

The vtlib library development is in progress and subject to change.
Source code changes required by vtlib are not yet integrated into vtiger. While we make every effort to make sure modules developed using vtlib will be compatible with future versions of vtiger CRM, some incompatible changes may be required for the next release of vtiger. In which case you will have to re-create your modules with an upgraded version of vtlib for the specific version of vtiger CRM.

Table of Contents

About	3
Version History	4
vtlib Installation	5
Creating a new module with vtlib	6
New Module Development Overview	7
Creating Tab (Menu Item)	8
Creating Block	9
Creating Field	10
Setup Picklist Values	12
Creating Listview Filter	13
Module Settings	14
Enable/Disable Action	14
Sharing Access	15
Custom Fields	16
Final Completed Script	17
Module Templates	21
Testing Module Creation	22
Test Scripts	23
Installing New Module	24
Invoking the Install Script	24
Script Output	25
New Module Tour	26
Default List view	26
Create view	27
Detail view	27
List view	27
Export – Import	28
Sharing Access	28
Custom Field	29
Package APIs	30
Package Export	30
Package Structure	31
Package Import	33
Module Manager	34
Disabling Module	35
Enabling Module	36
Exporting Module	37
Importing Module	39

About

vtlib is a library to ease new module development for vtiger CRM. Developers can use vtlib to develop vtiger CRM modules that add new functionality to vtiger CRM.

vtlib helps developers package the new modules into zip files that other vtiger CRM installations can easily install and use.

In this version, API's are provided to let you create and develop new modules for vtiger CRM more easily. It includes a Module Manager tool for Administrators to install and enable/disable the new modules developed using vtlib.

Version History

1.4	2008-09-29
1.3	2008-09-01
1.2	2008-08-29
1.1	2008-08-27
1.0	2008-08-19

vtlib Installation

vtlib Installation requires the following steps.

1. vtiger 5.0.4 should be installed.
2. Take backup of your vtiger CRM Source directory.
3. Unpack the vtlib-x.y.zip into your vtiger CRM source directory.

NOTE: Source directory in this document refers, vtiger CRM source installation.
If you have used vtiger CRM bundled installation like, .exe or .bin, it will be located in
<vtigercrm_install_directory>\apache\htdocs\vtigerCRM

Creating a new module with vtlib

vtlib simplifies creation of new vtiger CRM modules. Developers can use vtlib to develop vtiger CRM modules that add new functionality to vtiger CRM.

These modules can then be packaged for easy installation by the Module Manager.

To create a new module, use your vtiger CRM installation with the vtlib APIs.

Once the module is developed, you can use the Module Manager export, or the vtlib Package API, to package your module into a zip file. This zip file can be used to install your module on other vtiger CRM instances that need to use the new module

New Module Development Overview

Creating Module can be broken down into two steps:

1. Module UI setup
2. Module directory setup

Module UI setup

Using the API's explained below you will be able to create new modules for vtiger. The following are important steps that should be followed to get basic working module.

- a) Create tab (entry point for module on UI)
- b) Create required module table, module group table.
- c) Create blocks for the module.
- d) Create fields and associate it to blocks.
Set atleast one of the field as entity identifier.
- e) Create custom view with required fields (make sure to create filter name All which is treated as special default filter)

NOTE: Please look at `vtlib.Test.Create.Module1.php` file for a detailed example of the steps explained above.

Module directory setup

Each vtiger CRM module has a module directory associated with the module. `vtlib` directory contains a skeleton `ModuleDir` which can be used as a template for new module that is created. It contains source files that needs to be changed accordingly.

Before you begin, save a copy of `ModuleDir` for additional modules you may create.

1. Rename `ModuleDir` to `<NewModuleName>`
2. Rename `ModuleDir/ModuleFile.php` to `<NewModuleName>.php`
3. Rename `ModuleDir/ModuleFileAjax.php` to `<NewModuleName>Ajax.php`
4. Rename `ModuleDir/ModuleFile.js` to `<NewModuleName>.js`
5. Edit `<NewModuleName>.php`
 - a) Rename Class `ModuleClass` to `<NewModuleName>`
 - b) Update `$table_name` and `$table_index` (Module table name and table index column)
 - c) Update `$groupTable`
 - d) Update `$tab_name`, `$tab_name_index`
 - e) Update `$list_fields`, `$list_fields_name`, `$sortby_fields`
 - f) Update `$detailview_links`
 - g) Update `$default_order_by`, `$default_sort_order`
 - h) Update `$required_fields`
 - i) Update `$customFieldTable`
 - j) Update `$def_detailview_recname` [*Column name whose value will be used to display DetailView text for a record*]
 - k) Rename function `ModuleClass` to function `<NewModuleName>` [This is the Constructor Class]

Creating Tab (Menu Item)

```
include_once('vtlib/Vtiger/Tab.php');  
Vtiger_Tab::create('<NEWMODULE>', '<NEWMODULE_LABEL>', '<MENUNAME>');
```

This API will create new menu item which serves as entry point for the module.

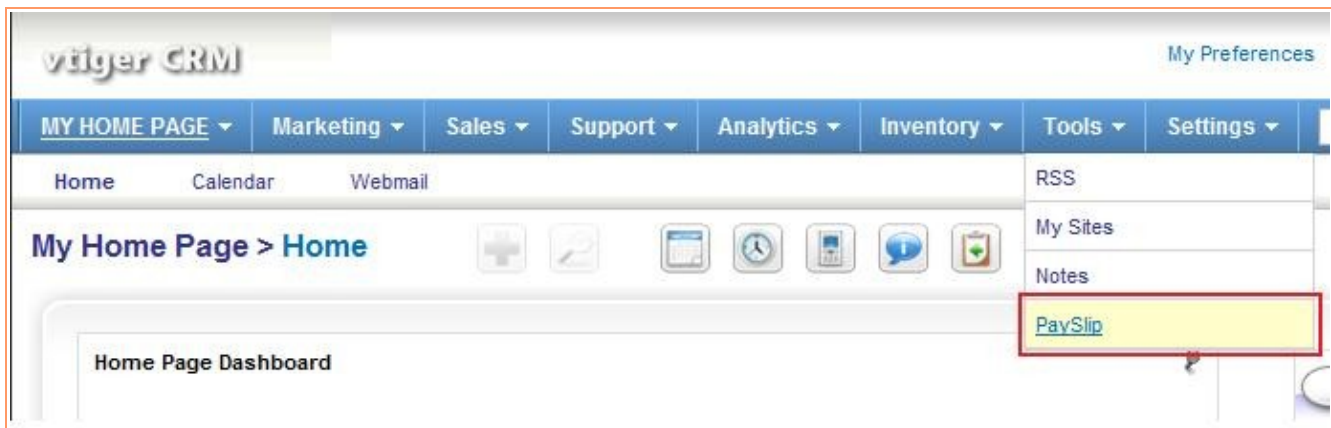
Example:

```
Vtiger_Tab::create('Payslip', 'Payslip', 'Tools');
```



NOTE: You will need to put the <NEWMODULE_LABEL> mapping in the language file (include/language/en_us.lang.php) to see the affect.

```
$app_strings = Array (  
'Payslip' => 'PaySlip',  
...  
);
```



Creating Block

```
include_once('vtlib/vtiger/Block.php');  
vtiger_Block::create('<MODULE>', '<BLOCK_LABEL>');
```

This API will create block for the specified <MODULE>.

Example:

```
vtiger_Block::create('Leads', 'LBL_VTLIB_INFORMATION');
```

NOTE: You will need to put the <BLOCK_LABEL> mapping in the module language file (modules/<MODULE>/language/en_us.lang.php) to see the affect.

For the above example, add the mapping in modules/Leads/language/en_us.lang.php

```
$mod_strings = Array (  
'LBL_VTLIB_INFORMATION' => 'vtlib Information',  
...  
);
```

NOTE2: The block will not be displayed until we associate one field to it.

Creating Field

```
include_once('vtlib/Vtiger/Field.php');
$field1 = new Vtiger_Field();

// Populate Field Information
$field1->set('module',      '<MODULE>')
    -> set('columnname',    '<TABLE COLUMN NAME>')
    -> set('tablename',     '<TABLE NAME>')
    -> set('columntype',     '<TABLE COLUMN TYPE>')
    -> set('generatedtype', '<STANDARD (=1) OR CUSTOM (=2) TYPE>')
    -> set('uitype',        '<UI TYPE>')
    -> set('fieldname',     '<FIELD NAME>')
    -> set('fieldlabel',    '<FIELD LABEL ON UI>') // Need language file entry
    -> set('readonly',      '<READONLY (=0) orelse (1)>')
    -> set('presence',      '0')
    -> set('selected',      '0')
    -> set('maximumlength', '<MAXIMUM DATA LENGTH (=100)>')
    -> set('sequence',      null) // will be set dynamically.
    -> set('typeofdata',    '<TYPE OF DATA>')
    -> set('quickcreate',    '<AVAILABLE IN QUICK CREATE FORM(=0) orelse(=1)>')
    -> set('block',         null) // Specifiy blockid here & make blocklabel null
    -> set('blocklabel',    '<BLOCK LABEL to use if block is not set>')
    -> set('displaytype',    '<DISPLAY TYPE>')
    -> set('quickcreatesequence', null) // will be set dynamically
    -> set('info_type',     '<TYPE OF INFORMATION (BAS=Basic,ADV=Advanced)>');

// Create Field
$field1->create();
```

The above API will let you create vtiger field and assoicate it to the module block. You will have to provide all the information described above.

NOTE: During vtiger field creation, the columnname existence is checked in the tablename specified. If it does not exist then the database table is altered and columnname (of columntype) is created.

You can create table before creating field & provide the tablename, columnname to use this table.

```
include_once('vtlib/Vtiger/Utils.php');
Vtiger_Utils::CreateTable('<TABLENAME>', '(<COLUMNNAME COLUMNTYPE>');
```

This API will let us create table with tablename and given column conditions.

Example: To create a new field and associate to custom block in Leads module.

```
include_once('vtlib/Vtiger/Field.php');
$field1 = new Vtiger_Field();
$field1-> set('module',      'Leads')
    -> set('columnname',     'vtlib_leads') // New column
    -> set('tablename',      'vtiger_leaddetails') // Existing table
    -> set('columntype',      'varchar(255)')
    -> set('generatedtype',  '1')
    -> set('uitype',         '2')
    -> set('fieldname',      'vtlibldname')
    -> set('fieldlabel',     'vtlibLeadName')
    // Entry needed in modules/Leads/language/en_us.lang.php
    -> set('readonly',      '1')
```

```
-> set('presence', '0')
-> set('selected', '0')
-> set('maxlength', '100')
-> set('sequence', null)
-> set('typeofdata', 'V~M')
-> set('quickcreate', '1')
-> set('block', null)
-> set('blocklabel', 'LBL_VTLIB_INFORMATION')
-> set('displaytype', '1')
-> set('quickcreatesequence', null)
-> set('info_type', 'BAS');
```

```
$field1->create();
```

After executing this script, make sure to check the mapping in `modules/Leads/language/en_us.lang.php` for fieldlabel (vtlibLeadName) used above.

If you have associated the field with a new table then you will need to update:
`modules/Leads/Leads.php` \$tab_name and \$tab_name_index Arrays

```
var $tab_name = Array( ..., '<NEW_TABLENAME>');
var $tab_name_index = Array( ..., '<NEW_TABLENAME>' => '<ID_COLUMN>');
```

Setup Picklist Values

When creating Picklist Field (uitype **15**, 16, 33, 55, 111), you will have to create a reference which stores the allowed picklist values and link it to the field. This can be achieved by the following API

```
include_once('vtlib/Vtiger/Field.php');  
Vtiger_Field_Instance->setupPicklistValues  
(Array ('value1', 'value2', ...));
```

The tablename related to Vtiger_Field instance should set to module's base table.

Based on fieldname value set for Vtiger_Field instance, the table vtiger_<fieldname> is created (if it does not exists). The values passed as parameter are added to this table and it will be associated with all the existing Roles. This will provide the effect of Role Based Picklist.

Example:

```
$field1 = new Vtiger_Field();  
$field1->setupPicklistValues( Array ('Employee', 'Trainee') );
```

Creating Listview Filter

```
include_once('vtlib/Vtiger/CustomView.php');
Vtiger_CustomView::create('<MODULE>', '<FILTER NAME>'
    [, <setdefault=false>, <setmetrics=false>]);
```

This API will let you create custom view (filter) for the required module. Once the filter is created you need to associate module fields to it. The module field instance need to be populated as explained in previous section.

```
$cv = new Vtiger_CustomView('<MODULE>', '<FILTER NAME>');
$cv->addColumn($Vtiger_Field_Instance);
```

Example:

Create custom view for Leads module and associating the field created in last section.

```
include_once('vtlib/Vtiger/CustomView.php');
include_once('vtlib/Vtiger/Field.php');

Vtiger_CustomView::create('Leads', 'vtlibFilter');

$cv = new Vtiger_CustomView('Leads', 'vtlibFilter');
$field1-> set('module', 'Leads')
    -> set('columnname', 'vtlib_leads')
    -> set('tablename', 'vtiger_leadaddetails')
    -> set('fieldname', 'vtlibldname')
    -> set('fieldlabel', 'vtlibLeadName')
    -> set('typeofdata', 'V~M');

$cv->addColumn($field1);
```

Module Settings

Enable/Disable Action

Import, Export and Merge are treated as Actions for a module. To enable or disable them, you can use the API given below.

```
include_once('vtlib/Vtiger/Module.php');  
Vtiger_Module::disableAction('<MODULE>', '<Action>');  
Vtiger_Module::enableAction('<MODULE>', '<Action>');
```

Example:

To Disable Import and Enable Export for Leads module.

```
Vtiger_Module::disableAction('Leads', 'Import');  
Vtiger_Module::enableAction('Leads', 'Export');
```

NOTE: This will enable or disable the action (tool) for all the existing profiles.

Sharing Access

```
include_once('vtlib/Vtiger/Module.php');  
Vtiger_Module::setDefaultSharingAccess('<MODULE>', '<PERMISSION_TYPE>');
```

Use this API to set default sharing access for the <MODULE>.

The <PERMISSION_TYPE> can be one of the following:

Public_ReadOnly
Public_ReadWrite
Public_ReadWriteDelete
Private

Example:

```
Vtiger_Module::setDefaultSharingAccess('Leads', 'Private');
```

Custom Fields

To enable Custom Fields for the module you will need to follow the steps given below:

1. The module should have the block labeled LBL_CUSTOM_INFORMATION
Look at the section [Creating Block](#)
2. Create table with name vtiger_<modulename>cf (should have primary key column) to save the custom field data.

Example: If your module name is Payslip, then custom field table to be created should be named vtiger_payslipcf with a primary key column integer type in it.

```
include_once('vtlib/vtiger/Module.php');  
vtiger_Utils::CreateTable('vtiger_payslipcf', '(payslipid integer, primary  
key (payslipid))');
```

3. Update \$tab_name and \$tab_name_index Array in the module class with the custom field table information.

```
var $tab_name = Array('vtiger_crmentity', 'vtiger_payslip',  
    'vtiger_payslipcf');  
  
var $tab_name_index = Array(  
    'vtiger_crmentity' => 'crmid',  
    'vtiger_payslip'   => 'payslipid',  
    'vtiger_payslipcf' => 'payslipid');
```

4. Update \$customFieldTable variable entry in the module class with custom field tablename and the primary key column name.

```
var $customFieldTable = Array('vtiger_payslipcf', 'payslipid');
```


Final Completed Script

vtlib.Test.Create.Module1.php

```
<?php
$Vtiger_Utills_Log = true;
include_once('vtlib/Vtiger/Utils.php');
include_once('vtlib/Vtiger/Field.php');
include_once('vtlib/Vtiger/CustomView.php');
include_once('vtlib/Vtiger/Tab.php');
include_once('vtlib/Vtiger/Block.php');

Vtiger_Tab::create('Payslip', 'Payslip', 'Tools');
Vtiger_Block::create('Payslip', 'LBL_PAYSLIP_INFORMATION');
Vtiger_Block::create('Payslip', 'LBL_CUSTOM_INFORMATION');

Vtiger_Utills::CreateTable('vtiger_payslip', '(payslipid integer)');

Vtiger_Utills::CreateTable('vtiger_payslipcf', '(payslipid integer, primary key (payslipid))');

Vtiger_Utills::CreateTable('vtiger_payslipgrouprel',
    '(payslipid integer, groupname varchar(100), primary key(payslipid))');

// Create Name field
$field1 = new Vtiger_Field();
$field1->set('module', 'Payslip')
    ->set('columnname', 'payslipname')
    ->set('tablename', 'vtiger_payslip')
    ->set('columntype', 'varchar(255)')
    ->set('generatedtype', '1')
    ->set('uitype', 2)
    ->set('fieldname', 'payslipname')
    ->set('fieldlabel', 'PayslipName')
    ->set('readonly', '1')
    ->set('presence', '0')
    ->set('selected', '0')
    ->set('maxlength', '100')
    ->set('sequence', null)
    ->set('typeofdata', 'V~M')
    ->set('quickcreate', '1')
    ->set('block', null)
    ->set('blocklabel', 'LBL_PAYSLIP_INFORMATION')
    ->set('displaytype', '1')
    ->set('quickcreatesequence', null)
    ->set('info_type', 'BAS');
$field1->create();

// Module should have atleast one field set as an identifier
$field1->set('entityidfield', 'payslipname')->set('entityidcolumn', 'payslipid');
$field1->setEntityIdentifier();

// Create Payslip Type field
$field2 = new Vtiger_Field();
$field2->set('module', 'Payslip')
    ->set('columnname', 'paysliptype')
    ->set('tablename', 'vtiger_payslip')
    ->set('columntype', 'varchar(255)')
    ->set('generatedtype', '1')
    ->set('uitype', 15)
    ->set('fieldname', 'paysliptype')
    ->set('fieldlabel', 'Payslip Type')
    ->set('readonly', '1')
    ->set('presence', '0')
```

```

-> set('selected', '0')
-> set('maxlength', '100')
-> set('sequence', null)
-> set('typeofdata', 'V~0')
-> set('quickcreate', '1')
-> set('block', null)
-> set('blocklabel', 'LBL_PAYSLIP_INFORMATION')
-> set('displaytype', '1')
-> set('quickcreatesequence', null)
-> set('info_type', 'BAS');
$field2->create();

$field2->setupPicklistValues( Array ('Employee', 'Trainee') );

// Create Date field
$field3 = new Vtiger_Field();
$field3-> set('module', 'Payslip')
-> set('columnname', 'payslipmonth')
-> set('tablename', 'vtiger_payslip')
-> set('columnname', 'date')
-> set('generatedtype', '1')
-> set('uitype', 23)
-> set('fieldname', 'payslipmonth')
-> set('fieldlabel', 'Month')
-> set('readonly', '1')
-> set('presence', '0')
-> set('selected', '0')
-> set('maxlength', '100')
-> set('sequence', null)
-> set('typeofdata', 'D~M')
-> set('quickcreate', '1')
-> set('block', null)
-> set('blocklabel', 'LBL_PAYSLIP_INFORMATION')
-> set('displaytype', '1')
-> set('quickcreatesequence', null)
-> set('info_type', 'BAS');
$field3->create();

// Create Assigned To field
$field4 = new Vtiger_Field();
$field4-> set('module', 'Payslip')
-> set('columnname', 'smownerid')
-> set('tablename', 'vtiger_crmentity')
//-> set('columnname', 'int')
-> set('generatedtype', '1')
-> set('uitype', 53)
-> set('fieldname', 'assigned_user_id')
-> set('fieldlabel', 'Assigned To')
-> set('readonly', '1')
-> set('presence', '0')
-> set('selected', '0')
-> set('maxlength', '100')
-> set('sequence', null)
-> set('typeofdata', 'V~M')
-> set('quickcreate', '1')
-> set('block', null)
-> set('blocklabel', 'LBL_PAYSLIP_INFORMATION')
-> set('displaytype', '1')
-> set('quickcreatesequence', null)
-> set('info_type', 'BAS');
$field4->create();

```

```

// Create Created Time field
$field5 = new Vtiger_Field();
$field5->set('module', 'Payslip')
->set('columnname', 'createdtime')
->set('tablename', 'vtiger_crmentity')
//->set('columnname', 'int')
->set('generatedtype', '1')
->set('uitype', 70)
->set('fieldname', 'createdtime')
->set('fieldlabel', 'Created Time')
->set('readonly', '1')
->set('presence', '0')
->set('selected', '0')
->set('maxlength', '100')
->set('sequence', null)
->set('typeofdata', 'T~0')
->set('quickcreate', '1')
->set('block', null)
->set('blocklabel', 'LBL_PAYSLIP_INFORMATION')
->set('displaytype', '2')
->set('quickcreatesequence', null)
->set('info_type', 'BAS');
$field5->create();

// Create Modified Time field
$field6 = new Vtiger_Field();
$field6->set('module', 'Payslip')
->set('columnname', 'modifiedtime')
->set('tablename', 'vtiger_crmentity')
//->set('columnname', 'int')
->set('generatedtype', '1')
->set('uitype', 70)
->set('fieldname', 'modifiedtime')
->set('fieldlabel', 'Modified Time')
->set('readonly', '1')
->set('presence', '0')
->set('selected', '0')
->set('maxlength', '100')
->set('sequence', null)
->set('typeofdata', 'T~0')
->set('quickcreate', '1')
->set('block', null)
->set('blocklabel', 'LBL_PAYSLIP_INFORMATION')
->set('displaytype', '2')
->set('quickcreatesequence', null)
->set('info_type', 'BAS');
$field6->create();

// Custom View
Vtiger_CustomView::create('Payslip', 'All', true);
$cv = new Vtiger_CustomView('Payslip', 'All');
$cv->addColumn($field1)
->addColumn($field3, 1)
->addColumn($field4, 2)
->addColumn($field6, 3);

Vtiger_CustomView::create('Payslip', 'All2');
$cv = new Vtiger_CustomView('Payslip', 'All2');
$cv->addColumn($field1)
->addColumn($field2, 1)
->addColumn($field5, 3)
->addColumn($field3, 2);

```

```
// Enable Import and Export
Vtiger_Module::disableAction('Payslip','Import');
Vtiger_Module::enableAction('Payslip','Export');

Vtiger_Module::setDefaultSharingAccess('Payslip','Private');

?>
```

Module Templates

Your module specific Smarty template files should be created under
Smarty/templates/modules/<NewModuleName>.

Use vtlib_getModuleTemplate(\$module, \$templateName) API (include/Utils/VtlibUtils.php) as:

```
$smarty->display(vtlib_getModuleTemplate($currentModule, 'MyListview.tpl'));
```

Testing Module Creation

1. After unpacking the vtlib zip file, open <http://localhost/vtigercrm/vtlib.Test.html>
2. Click on [Create Payslip Module](#) to test creation of Payslip Module
3. Click on [Create EmailScanner Module](#) to test creation of EmailScanner Module

NOTE: When you create module as said above, you will see blank links added under Tools menu. Kindly edit include/language/en_us.lang.php and update \$app_strings as shown below:

```
$app_strings = Array (  
'Payslip' => 'PaySlip',  
'EmailScanner' => 'EMAIL SCANNER',  
...  
)
```

After this you will be able to see **Tools » Payslip** and **Tools » EMAIL SCANNER**

Test Scripts

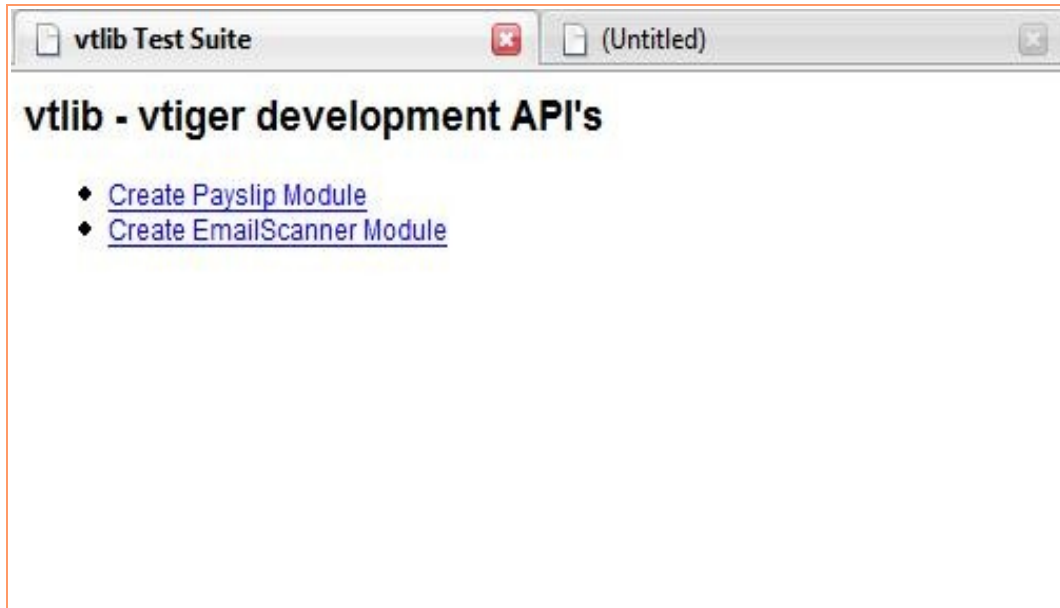
vtlib.Test.html	Provides link to execute the following scripts
vtlib.Test.Create.Menu.Item.php	Example to create menu item
vtlib.Test.Create.Module.Blocks.php	Example to show module creation and block for it.

vtlib.Test.html	Provides link to execute the following scripts
vtlib.Test.Create.Module.Fields.php	Example to show module creation and fields for it.
vtlib.Test.Create.Module1.php	<p>Payslip Module Creation (modules/Payslip dir is already packaged)</p> <p>Make the following entry in include/language/en_us.lang.php after script execution</p> <pre>\$app_strings = Array('Payslip' => 'PaySlip',</pre>
vtlib.Test.Create.Module2.php	<p>EmailScanner Module Creation (modules/EmailScanner dir is already packaged)</p> <p>Make the following entry in include/language/en_us.lang.php after script execution</p> <pre>\$app_strings = Array('EmailScanner' => 'Email Scanner',</pre>

Installing New Module

Invoking the Install Script

vtlib is bundled with install script which allows you to create sample modules. After installation open vtlib.Test.html, like, <http://localhost/vtigercrm/vtlib.Test.html> you will see the following:



Script Output

Given below is the Payslip module (vtlib.Test.Create.Module1.php) script output

```
Creating new item (30) under Tools ... DONE
Creating new menu item Payslip (Payslip) ... DONE
Check Language Mapping entry for Payslip ... TODO
Creating block Payslip (LBL_PAYSLIP_INFORMATION) ... DONE
Check Module Language Mapping entry for LBL_PAYSLIP_INFORMATION ... TODO
Adding PayslipName to Payslip ... DONE
Check Module Language Mapping entry for PayslipName ... TODO
Adding Month to Payslip ... DONE
Check Module Language Mapping entry for Month ... TODO
Adding Assigned To to Payslip ... DONE
Check Module Language Mapping entry for Assigned To ... TODO
Adding Created Time to Payslip ... DONE
Check Module Language Mapping entry for Created Time ... TODO
Adding Modified Time to Payslip ... DONE
Check Module Language Mapping entry for Modified Time ... TODO
Creating CustomView(All) for Payslip ... DONE
Adding PayslipName to Payslip CustomView(38) ... DONE
Adding Month to Payslip CustomView(38) ... DONE
Adding Assigned To to Payslip CustomView(38) ... DONE
Adding Modified Time to Payslip CustomView(38) ... DONE
Creating CustomView(All2) for Payslip ... DONE
Adding PayslipName to Payslip CustomView(39) ... DONE
Adding Assigned To to Payslip CustomView(39) ... DONE
Adding Modified Time to Payslip CustomView(39) ... DONE
Disabling Action Import for module Payslip for Profile 1 ... DONE
Disabling Action Import for module Payslip for Profile 2 ... DONE
Disabling Action Import for module Payslip for Profile 3 ... DONE
Disabling Action Import for module Payslip for Profile 4 ... DONE
Enabling Action Export for module Payslip for Profile 1 ... DONE
Enabling Action Export for module Payslip for Profile 2 ... DONE
Enabling Action Export for module Payslip for Profile 3 ... DONE
Enabling Action Export for module Payslip for Profile 4 ... DONE
```

NOTE: Kindly edit include/language/en_us.lang.php and update \$app_strings as shown below:

```
$app_strings = Array (
'Payslip' => 'PaySlip'
...

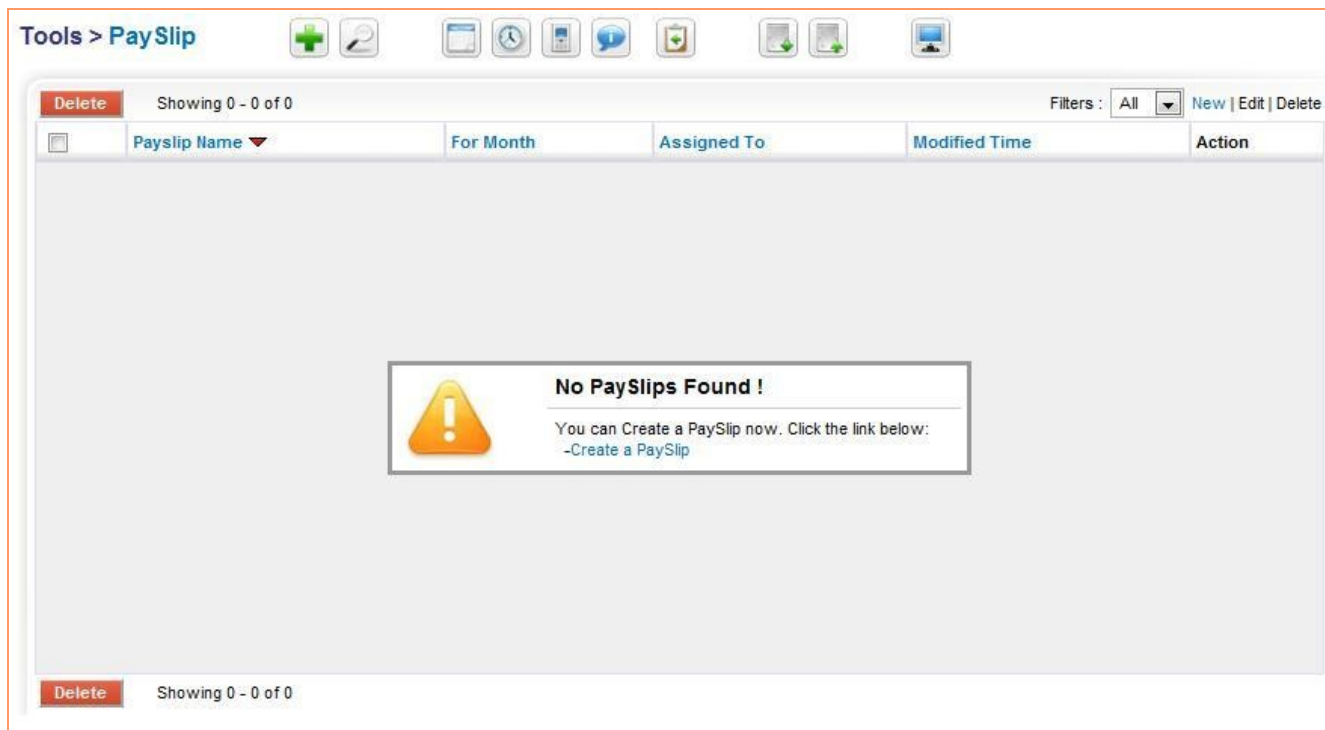
```

New Module Tour

The module link appears under Tools Menu.



Default List view



Create view

Tools > PaySlip

Basic Information

Save Cancel

Payslip Information

*Payslip Name	<input type="text" value="Test Payslip"/>	For Month	<input type="text" value="2008-08-01"/> <small>(yyyy-mm-dd)</small>
Assigned To	<input checked="" type="radio"/> User <input type="radio"/> Group <input type="text" value="admin"/>		

Save Cancel

Detail view

Tools > PaySlip

[132] Test Payslip - PaySlip Information
Updated today (06 Aug 2008)

PaySlip Information

Edit Duplicate Delete

Payslip Information			
Payslip Name	Test Payslip	For Month	2008-08-01
Assigned To User	admin	Created On	2008-08-06 14:54:03
Modified On	2008-08-06 14:54:03		

TAG CLOUD
 Tag it

List view

Tools > PaySlip

Delete Showing 1 - 1 of 1 Filters: All2 New | Edit | Delete

<input type="checkbox"/> Payslip Name	Modified Time	Assigned To	Action
<input type="checkbox"/> Test Payslip	2008-08-06 14:54:03	admin	edit del


Delete Showing 1 - 1 of 1

Export – Import

Tools > PaySlip


Showing 0 - 0 of 0

Filters : All New | Edit | Delete

	PaySlip Name	For Month	Assigned To	Modified Time	Action
<div>No PaySlips Found !<p>You can Create a PaySlip now. Click the link below: -Create a PaySlip</p></div>					

Showing 0 - 0 of 0


Sharing Access

 **Settings > Sharing Access**
Manage module sharing rules & custom sharing rules

1. Organization-level Sharing Rules Recalculate Change Privileges

Potentials	★ Public: Read, Create/Edit, Delete	Users can Read, Create/Edit, Delete other users Potentials
Accounts & Contacts	★ Public: Read, Create/Edit, Delete	Users can Read, Create/Edit, Delete other users Accounts & Contacts
Leads	★ Public: Read, Create/Edit, Delete	Users can Read, Create/Edit, Delete other users Leads
Calendar	★ Private	Users cannot access other users Calendar
Trouble Tickets	★ Public: Read, Create/Edit, Delete	Users can Read, Create/Edit, Delete other users Trouble Tickets
Quotes	★ Public: Read, Create/Edit, Delete	Users can Read, Create/Edit, Delete other users Quotes
Purchase Order	★ Public: Read, Create/Edit, Delete	Users can Read, Create/Edit, Delete other users Purchase Order
Sales Order	★ Public: Read, Create/Edit, Delete	Users can Read, Create/Edit, Delete other users Sales Order
Invoice	★ Public: Read, Create/Edit, Delete	Users can Read, Create/Edit, Delete other users Invoice
Campaigns	★ Public: Read, Create/Edit, Delete	Users can Read, Create/Edit, Delete other users Campaigns
PaySlip	★ Private	Users cannot access other users PaySlip

Custom Field





Settings > Custom Field Settings

Manage your company-wide custom fields.

Custom Fields in "PaySlip" Module

Select Module: PaySlip

New Custom Field

#	Field Label	Field Type	Tools
1	Site	URL	 

[Scroll to Top]

Package APIs

Package Export

You can export module as package using the following API

```
require_once('vtlib/Vtiger/Package.php');  
$package = new Vtiger_Package();  
$package->export('<MODULE NAME>', '<DESTINATION DIR>', '<ZIPFILE NAME>', <DIRECT  
DOWNLOAD>);
```

<MODULE NAME>	Module to be exported
<DESTINATION DIR>	(Optional: Default=test/vtlib) Directory where the zipfile output should be created.
<ZIPFILE NAME>	(Optional: Default=modulename-timestamp.zip) Zipfile name to use for the output file.
<DIRECT DOWNLOAD>	(Optional: Default=false) If true, the zipfile created will be streamed for download and zipfile will be deleted after that.

Example:

```
$package = new Vtiger_Package();  
$package->export('Payslip', 'test/vtlib', 'Payslip-Export.zip', true);
```

NOTE: Please make sure test directory under vtigercrm is writeable. A new folder (vtlib) will be created for export purpose.

Package Structure

The exported zipfile (package) has the following structure:

```
manifest.xml
modules/
    ModuleName/
        <Module Related Files>
            language/
                en_us.lang.php
            <Other language Files>
templates/
    <Smarty templates of the Module>
```

manifest.xml has the meta information that will be useful during the import process as shown:

```
<?xml version="1.0"?>
<module>
  <exporttime>YYYY-MM-DD hh:mm:ss</exporttime>
  <name>MODULE NAME</name>
  <label>MODULE LABEL</label>
  <parent>MENU</parent>

  <dependencies>
    <vtiger_version>VTIGER_VERSION_NUMBER</vtiger_version>
  </dependencies>

  <tables>
    <table>
      <name>TABLENAME</name>
      <sql>TABLE SQL</sql>
    </table>
  </tables>

  <blocks>
    <block>
      <label>BLOCK LABEL</label>
      <fields>
        <field>
          <fieldname>payslipname</fieldname>
          <columnname>payslipname</columnname>
          <uitype>UI TYPE</uitype>
          <tablename>TABLE NAME</tablename>
          <generatedtype>GEN TYPE</generatedtype>
          <fieldlabel>FIELD LABEL</fieldlabel>
          <readonly>READONLY</readonly>
          <presence>PRESENCE</presence>
          <selected>SELECTED</selected>
          <maxlength>MAXLEN</maxlength>
          <typeofdata>TYPEOFDATA</typeofdata>
          <quickcreate>QUICKCREATE</quickcreate>
          <displaytype>DISPTYPE</displaytype>
          <info_type>INFOTYPE</info_type>
        </field>
      </fields>
    </block>
  </blocks>

  <customviews>
    <customview>
      <viewname>VIEWNAME</viewname>
      <setdefault>0</setdefault>
      <setmetrics>1</setmetrics>
```

```
<fields>
  <field>
    <fieldname>FIELDNAME</fieldname>
    <columnindex>0</columnindex>
  </field>
</fields>
</customview>
</customviews>
<sharingaccess>
  <default>private</default>
</sharingaccess>

<actions>
  <action>
    <name>Export</name>
    <status>enabled</status>
  </action>
  <action>
    <name>Import</name>
    <status>enabled</status>
  </action>
</actions>
</module>
```


Package Import

You can import module from package using the following API

```
require_once('vtlib/Vtiger/Package.php');
$package = new Vtiger_Package();
$package->import(<MODULE ZIPFILE>, <overwrite>);
```

<MODULE ZIPFILE>	Module zipfile.
<overwrite>	(Optional: Default=false) Overwrite the existing module directory if present

Following API would be useful during the import process:

Validate ZipFile before Import

```
require_once('vtlib/Vtiger/Package.php');
$package = new Vtiger_Package();
$package->checkZip(<MODULE ZIPFILE>);
```

<MODULE ZIPFILE>	Module zipfile.
------------------	-----------------

checkZip returns **true** if package structure in zipfile is as specified in the section **Package Export** above.

Detect Module being Imported

```
require_once('vtlib/Vtiger/Package.php');
$package = new Vtiger_Package();
$package->getModuleNameFromZip(<MODULE ZIPFILE>);
```

<MODULE ZIPFILE>	Module zipfile.
------------------	-----------------

getModuleNameFromZip returns **ModuleName** if checkZip succeeds.

Example:

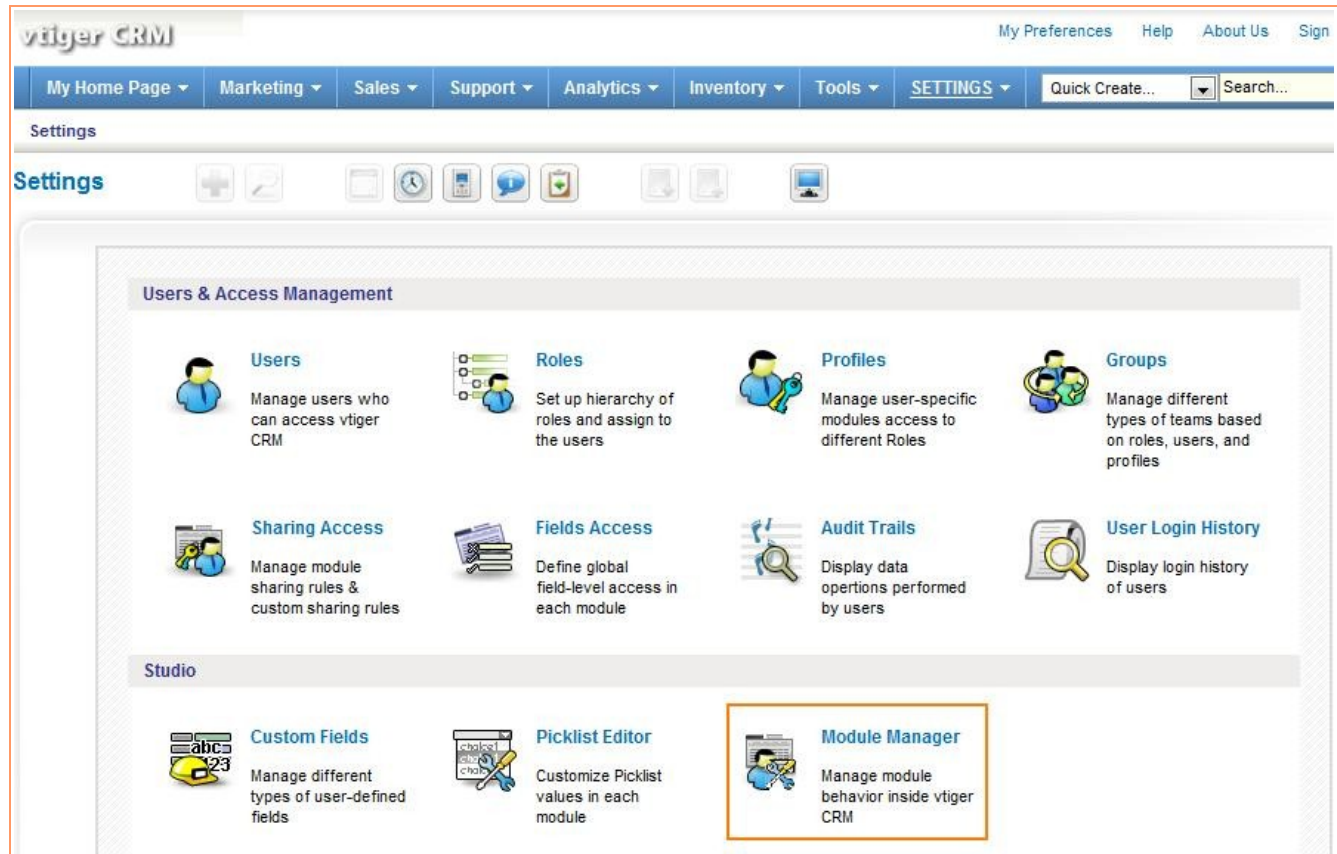
```
require_once('vtlib/Vtiger/Package.php');
require_once('vtlib/Vtiger/Module.php');

$package = new Vtiger_Package();
$module = $package->getModuleNameFromZip('test/vtlib/Payslip.zip');

$module_exists = false;
$module_dir_exists = false;
if($module == null) {
    echo "Module zipfile is not valid!";
} else if(Vtiger_Module::getId($module)) {
    echo "$module already exists!";
    $module_exists = true;
} else if(is_dir("modules/$module")) {
    echo "$module folder exists! Overwrite?";
    $module_dir_exists = true;
}
if($module_exists == false && $module_dir_exists == false) {
    $package->import('test/vtlib/Payslip.zip');
}
```

Module Manager

vtlib provides Module Manager configuration tool under Settings. With this you can enable or disable vtiger CRM modules. On disabling the module, it won't be shown on the Menu and access is restricted (including administrator).



Click on the green tick icon.



Enabling Module


Click on the red cross icon.

The screenshot shows the vtiger CRM interface. The top navigation bar includes 'My Home Page', 'Marketing', 'Sales', 'Support', 'Analytics', 'Inventory', 'Tools', 'SETTINGS', and a 'Quick Create...' button. The 'Sales' dropdown menu is open, showing options like 'Accounts', 'Contacts', 'Potentials', 'Quotes', 'Sales Order', 'Invoice', 'Products', 'Price Books', 'Notes', and 'Calendar'. The 'Module Manager' section is visible, showing a list of modules with their status (checked or unchecked) and an 'Import New' button. The 'Leads' module is highlighted with a red box, and its status is shown as unchecked (red cross icon).

Module	Status	Action
Accounts	✓	↑
Contacts	✓	↑
Potentials	✓	↑
Quotes	✓	↑
Sales Order	✓	↑
Invoice	✓	↑
Products	✓	↑
Price Books	✓	↑
Notes	✓	↑
Calendar	✓	↑
Campaigns	✓	↑
Contacts	✓	↑
Dashboard	✓	↑
Emails	✓	↑
Faq	✓	↑
HelpDesk	✓	↑
Home	✓	↑
Invoice	✓	↑
Leads	✗	↑
Notes	✓	↑















Exporting Module

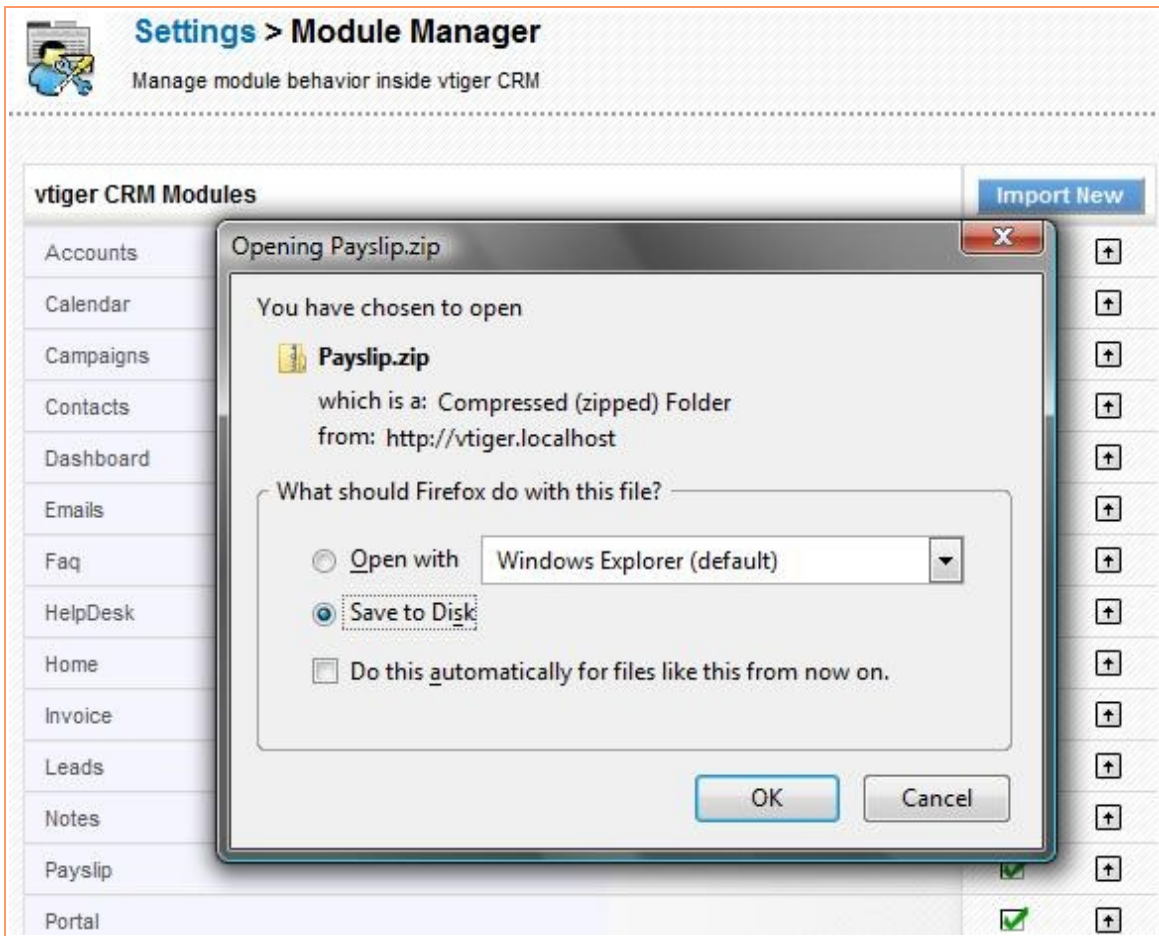
Click on the UP arrow icon in the module manager, which will export the module as a zip file.

 **Settings > Module Manager**
Manage module behavior inside vtiger CRM

vtiger CRM Modules

Import New

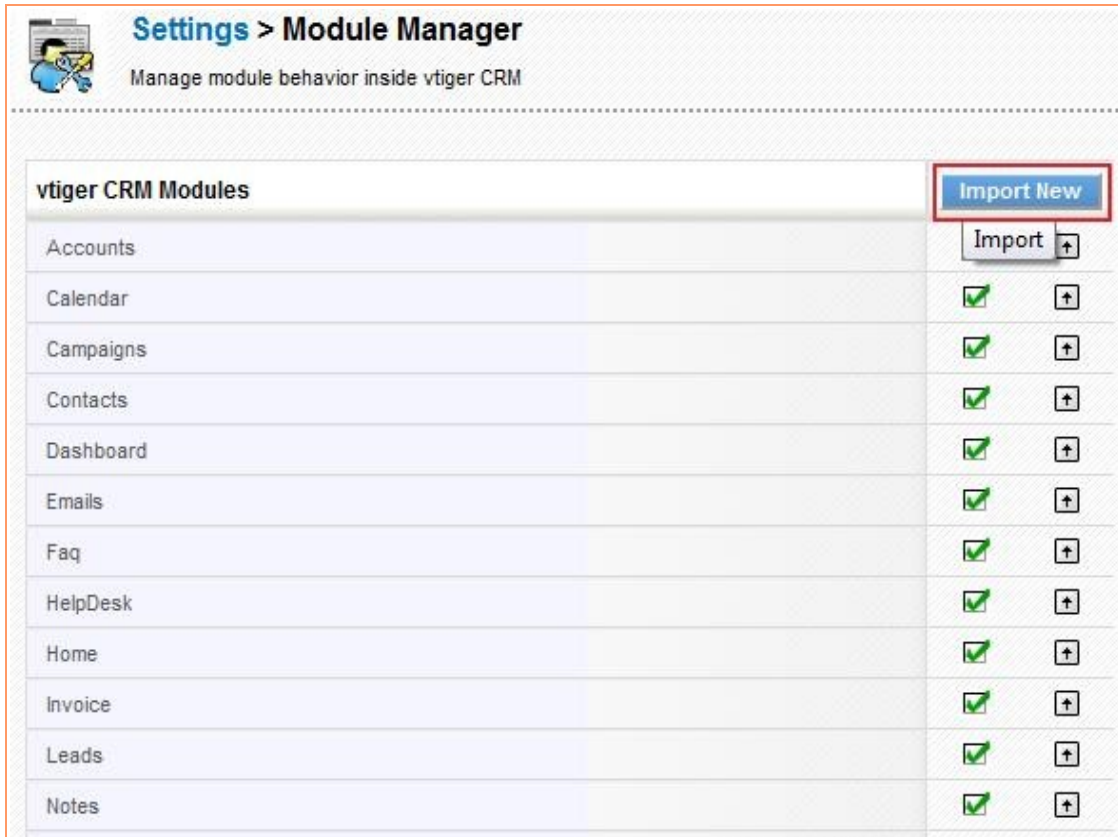
Accounts	<input checked="" type="checkbox"/>	
Calendar	<input checked="" type="checkbox"/>	
Campaigns	<input checked="" type="checkbox"/>	
Contacts	<input checked="" type="checkbox"/>	
Dashboard	<input checked="" type="checkbox"/>	
Emails	<input checked="" type="checkbox"/>	
Faq	<input checked="" type="checkbox"/>	
HelpDesk	<input checked="" type="checkbox"/>	
Home	<input checked="" type="checkbox"/>	
Invoice	<input checked="" type="checkbox"/>	
Leads	<input checked="" type="checkbox"/>	
Notes	<input checked="" type="checkbox"/>	
Payslip	<input checked="" type="checkbox"/>	
Portal	<input checked="" type="checkbox"/>	 Export Payslip



Importing Module

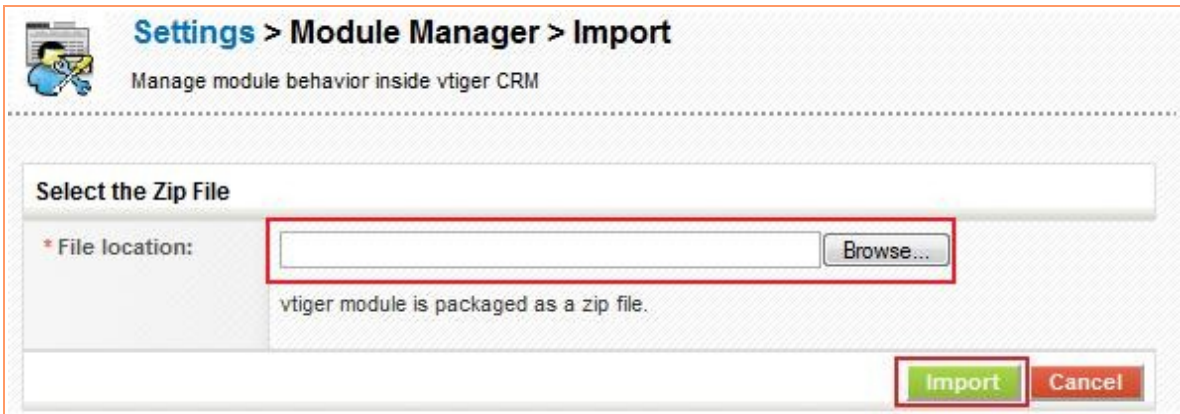
Module manager will let you import new modules. Follow the steps given below:

Click on the Import New button



vtiger CRM Modules		
Accounts		Import +
Calendar	✓	+
Campaigns	✓	+
Contacts	✓	+
Dashboard	✓	+
Emails	✓	+
Faq	✓	+
HelpDesk	✓	+
Home	✓	+
Invoice	✓	+
Leads	✓	+
Notes	✓	+

Select the zip file (module package) that was previously exported or created.



Select the Zip File

* File location: Browse...

vtiger module is packaged as a zip file.

Import Cancel

Verify the import details parsed from zipfile. Click Yes to proceed or No to cancel.



Settings > Module Manager > Import
Manage module behavior inside vtiger CRM

Verify Import Details

Module Name	Payslip
Module Directory	modules/Payslip

Do you want to proceed with the import? **Yes** No

Click on Finish to complete the module import.



Settings > Module Manager > Import
Manage module behavior inside vtiger CRM

Importing Module ...

Creating new item (30) under Tools ... DONE
Synchronizing tab data file ... STARTED
Synchronizing tab data file ... DONE
Setting up sharing access options ... DONE
Creating new menu item Payslip (Payslip) ... DONE
Check Language Mapping entry for Payslip ... TODO
Creating block Payslip (LBL_PAYSLIP_INFORMATION) ... DONE
Check Module Language Mapping entry for LBL_PAYSLIP_INFORMATION ... TODO
Adding PayslipName to Payslip ... DONE
Check Module Language Mapping entry for PayslipName ... TODO
Adding Payslip Type to Payslip ... DONE
Check Module Language Mapping entry for Payslip Type ... TODO
Creating picklist table vtiger_paysliptype ... DONE
Adding Month to Payslip ... DONE
Check Module Language Mapping entry for Month ... TODO
Adding Assigned To to Payslip ... DONE
Check Module Language Mapping entry for Assigned To ... TODO
Adding Created Time to Payslip ... DONE
Check Module Language Mapping entry for Created Time ... TODO
Adding Modified Time to Payslip ... DONE
Check Module Language Mapping entry for Modified Time ... TODO
Creating block Payslip (LBL_CUSTOM_INFORMATION) ... DONE
Check Module Language Mapping entry for LBL_CUSTOM_INFORMATION ... TODO
Creating CustomView(All) for Payslip ... DONE
Adding PayslipName to Payslip CustomView(38) ... DONE
Adding Month to Payslip CustomView(38) ... DONE
Adding Assigned To to Payslip CustomView(38) ... DONE
Adding Created Time to Payslip CustomView(39) ... DONE
Recalculating sharing rules ... STARTED
Recalculating sharing rules ... DONE

Finish

NOTE: If you are trying to import module which already exists or directory which is present in the modules folder you will see the following message.

**Settings > Module Manager > Import**
Manage module behavior inside vtiger CRM

Verify Import Details

Module Name	Payslip Exists
Module Directory	modules/Payslip Exists

Cancel