Provisioner User Guide
S. Hamblett

Table of Contents

1. Introduction

The Provisioner component is intended to allow provisioning of a MODx Revolution installation from an existing, remote MODx Revolution or MODx Evolution installation.

Using this component resources, elements, files and users can be imported from a remote installation to the local installation undergoing provisioning. The component uses graphical interaction to achieve this utilising the same tree view/grid mechanisms employed in the existing Revolution manager. This gives a good level of integration into the manager and leverage's the use of existing MODx component code to the fullest.

This component is not intended as a replacement to the existing Revolution package management system, rather an enhancement of it. The package management system should be used as the main provisioning mechanism of new Revolution installations, this component should be used to 'fill the gaps' as it were after package installation.

No matter how many packages are implemented for Revolution there will always be content that is not packaged, this may be content that the creator doesn't have the necessary skills/time to package or that is just tweaks to existing templates, chunks etc. done locally to please a particular client. This is especially true for existing Evolution installations that don't use packaging as such.

In the past, say in Evolution, these tweaks could be exported to another Evolution installation by a combination of SQL file export/import, zip file creation and ftp upload. This component allows this to be done through a graphical interface by non-technical users.

It should be noted that it does not allow a site to be 'ripped' of its content, the user has to log into the remote site as a manager user that has at least the level of privilege needed to perform the operations needed, i.e. view elements, users etc. Also it operates as a read only viewer, no content in the remote site is changeable using this component.

2. Installation

The provisioner package is installed as any other, download the package using package manager and install, or download the transport file into your packages directory and perform a local package installation. You should be using at least a Revolution 2.0 RC2 installation.

For each remote Evolution site you wish to provision from you need to install the Revolution gateway code so Provisioner can communicate with it. This is a simple matter of uploading an unzipped copy of the file 'revogatway.zip' into the assets folder using Evolution's Manage Files and unzipping it from there. You should now have a revogateway directory containing an 'index.php' file and a 'connectors' directory. That's it, nothing more to do.

On installation the following local entities are created :-

A context named 'provisioner'.

A category named 'Provisioner'.

A usergroup named 'Provisioner'.

A series of system settings under the area and namespace 'Provisioner'.

The purpose of these will be explained in later sections.

After installation please click the 'Home' top menu option, this will re-draw the top menu bar allowing the component menu to now show the Provisioner component. You should also clear your site cache at this point.

The component has 7 tabs, namely Administration, Resources, Elements, Files, Packages , Users and Evolution Site Import.

3. Administration

After starting the Provisioner component log into the remote site on the Administration tab by filling in the account, password and URL details. The URL should be the URL to the connectors directory in the remote installation for Revolution sites, this is usually /connectors under the base install, however Revolution does give you the option to move this at install time, so don't assume this. For Evolution sites it should be the URL of the site itself.

For remote Revolution installations the account chosen must have sufficient privileges to allow your requested operations to be carried out, an administration account usually suffices. i.e. one with at least 'list' permissions. For remote Evolution sites this must be a manager use account.

You must also select the remote site type as being either Revolution or Evolution.

Additionally for Revolution sites you must supply a site identifier for the remote site. This is a unique key generated on installation of the remote Revolution site that ties the AJAX calls made in the front end to the back end connectors for that specific site, i.e. it is an API key. This can be found in the config.inc.php file under /core/config of the remote site. Look near the top of this file for a PHP variable named '$site_id' and copy it exactly, without quotes.

If you do not log in the component will not function. You can log out at any time but you must re-login to continue provisioning. What this actually does is log you into the remote installation as if you had simply invoked its manager page and logged in.

Provisioner stores the currently logged in user along with its other status variables, on

login requests the stored value is compared with the current manager user, if Provisioner is currently logged in and these values are different the log in request is denied. This locks the usage of Provisioner to one user at a a time.

4. Resources

The Resources tab shows a tree view of the resources in the remote installation, on first entry to this tab press the refresh tree icon, an example below.

The tree operates in the same way as the local resources tree, right clicking on a resource will give a menu with an 'import' selection on it. Resources can be imported individually or by parent container. Importation of a container will import all its children and sub-containers fully recursively starting at the import node(*no limit here).

If the remote site is an Evolution installation the tree will appear with only one context. This will be named 'Evolution' and will be the root node of the tree. This has no meaning to the remote Evolution site but is need by the local Revolution installation to correctly draw the tree structure.

Also, an additional import option is given for remote Evolution resources, this being 'Import Convert Tags'. If this is selected on a resource the resource will be 'tag converted' on import i.e. any Evolution tags in the resources content, title or long title fields will be converted to the corresponding Revolution tag syntax.

All resource imports are placed in the provisioner context. All imported resources will have the following attributes set as below regardless of their setting in the remote site :-

Not Published

Hidden from Menu

Parent of 0 if a single resource or the resources new parent if a container/children is imported

Context key of provisioner

Other attributes will be as set in the remote site.

This allows resources to be imported without affecting the working of the site, from the provisioner context the resources can be dragged and dropped anywhere in the local installation resource tree as need and can then be edited as needed.

5. Elements

The Elements tab shows a tree view of the elements in the remote installation, on first entry to this tab press the refresh tree icon, an example below.
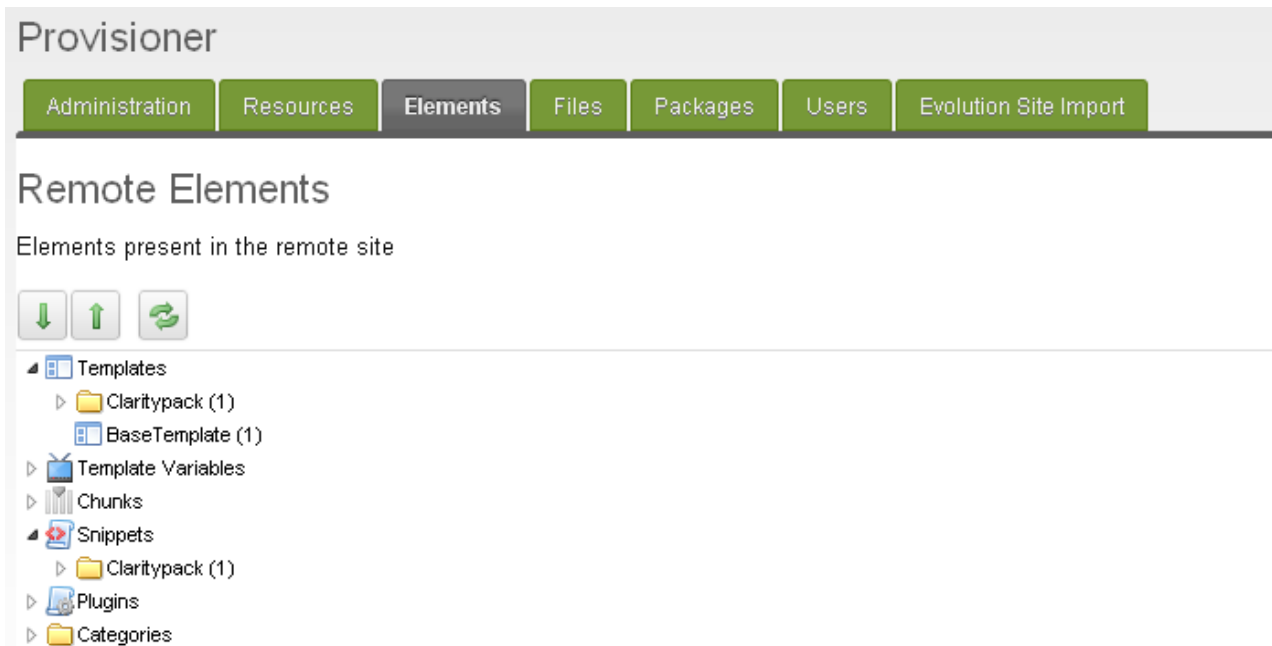
Both elements and categories can be imported. Elements can be imported singly, or by element type and category e.g. all snippets in the 'demo' category.

On import the elements are placed in the element tree under their type and associated with the Provisioner category if imported singly.

If imported by category the category is first checked for existence in the local installation. If it exists the elements are placed in this existing category. If it does not exist it is created and parented to the Provisioner category underneath the element type.

Note that 'merging' goes on here, if during the import an element is found to exist in the local installation already it is NOT re-created, you get no warning of this with elements unlike other imports.

These elements can now be moved/edited as normal.

In a similar manner to resources, chunks and template elements also have an 'Import Convert Tags' option if the remote site is an Evolution installation. This allows syntax for chunk inclusion, place holders etc. to be converted to the corresponding Revolution tag syntax

Categories imported singly are parented to the Provisioner category if they do not already exist.

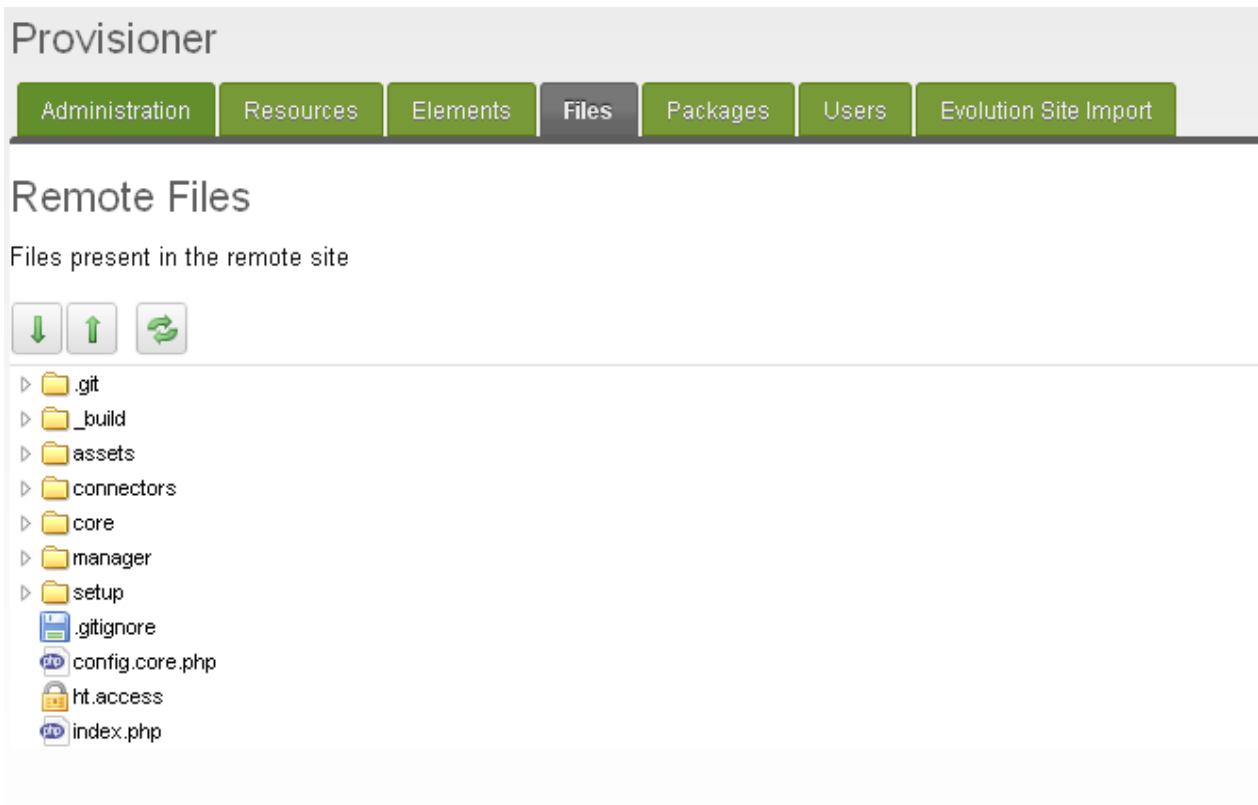All other attributes are as set in the remote site.

Note that you will have to refresh the local element tree to see the changes.

6. Files

The Files tab shows a tree view of the files in the remote installation, on first entry to this tab press the refresh tree icon, an example below.

The root of the remote file tree is dictated by the system settings base_path and rb_base_dir in the remote site. In the above case base_path is the root of the installation and rb_base_dir is empty. Setting rb_base_dir to /assets would result in the assets folder being the root. For remote Evolution sites only rb_base_dir applies.

Files can be imported singly or as a directory. Note that directory imports are NOT recursive, only single files contained in the directory are imported any sub-directories need to be imported separately.

From remote Revolution sites only files to be deemed 'not binary' by Revolution will have the import menu, from remote Evolution sites all file types have the import menu.

All imported file are placed into the directory '/assets/components/provisioner/imports' from here you can use your local file tree view to move the files around.

7. Packages

The packages tab allows the user to see a list of packages present in a remote Revolution site and whether they are installed locally or not, example below.

You will have to refresh the package list to see this. Be especially wary of this if you logout of one site and into another, the list will show the last sites content until refreshed.

The 'Installed Locally' column shows whether the package is installed locally or not.

From a work flow perspective this is where you should start provisioning your site, once you align the packages(the red 'no's go to green 'yes's) the bulk of your provisioning should be done. Note you can't install packages from here you must use the normal package management facilities to do this.

If the remote site is an Evolution installation then, as Evolution has no packages, this grid simply has one entry indicating this.

8. Users

The Users tab shows a grid view of the users in the remote installation, on first entry to this tab press the refresh grid icon, an example below. Be especially wary of this if you logout of one site and into another, the list will show the last sites content until refreshed.

For remote sites that are Evolution installations the ID column has the individual id's annotated with a '_w' if the user is a web user. This allows both manager users and web users to be viewed. In Evolution these are stored in separate tables, thus the id's will be equal in most cases and need to be separated out. An example below.

Right clicking on a user allows the user attributes such as country, email, gender, etc. to be imported.

This saves typing in user details over again if you know a particular remote user will be using your new site.

On import a local user is created in the Provisioner user group. The following attributes are set regardless of their setting in the remote site :-

Role is set to 0

Blocked is set to true

Login Count, Last Login, This Login, Failed Login Count are set to 0

Session Id is cleared.

The password is preserved but you will probably need to set one, I believe XPDO re-encodes this field(as though it had come from the GUI).

9. Evolution Site Import

This tab allows importation of Evolution sites both as a bulk operation and in a 'smart' manner whereby associated data such as categories, templates, resources, TV's resource groups etc. are linked back together in the local site as they were in the remote site. On entry to the tab the following is displayed :-

The above selections allow the user to select which context resources are imported into, whether all imported categories are re-parented to the provisioner category or not and whether the import should include chunks, snippets and plugins.

When the import button is pressed the user is presented with a warning dialog stating that this operation is both destructive and irreversible, the 'yes' button must be pressed here upon which the warning dialog will close and a further press of the import button will actually perform the import.

If the above sequence is not followed no import will occur, so users must actively decide to do the import, it can't be activated accidentally on pressing the import button.

Please be aware that this operation will delete and re-create the following content by default; categories, templates, resources, keywords, meta tags, resource groups and template variables. If selected to do so it will also delete and re-create chunks, snippets, plugins and associated event mappings.

Before discussing potential work flows the exact sequence of operations performed need to be understood, this is detailed below. Please also bear in mind that the import is database based, no remote files are imported in this process, so templates for instance although present in the local site after the import will not work until their associated css files and the like are also imported and made resident locally.

On pressing the import button for the second time the following is performed in order :-

1. All categories from the remote site are retrieved, all local categories are deleted

and then re-created using the remote site list. If the option 'Parent categories' is selected then all the re-created categories are parented to the provisioner category.

2. All templates from the remote site are retrieved, all local templates deleted and then re-created using the remote site list. On creation the templates are linked to the categories created in step 1.

3. All resources from the remote site are retrieved, all local resources deleted and then re-created using the remote site list. The resources are created in the context as set by the 'Context' selection and linked to the templates created in step 2.

4. All keywords, meta tags and document groups from the remote site are retrieved, all local keywords, meta tags and resource groups are deleted and re-created using the remote site list. On creation the keywords and resource groups are linked to their respective resources thus preserving the linkage present in the remote site.

5. All template variables from the remote site are retrieved, all local template variables deleted and then re-created using the remote site list. On creation the template variables are linked to the categories created in step 1.

6. The mappings of template variables to templates, resources and resource groups are retrieved from the remote site. Any existing mappings are deleted and re-created using the remote site list thus preserving the linkage present in the remote site.

7. If the user has selected to include snippets all snippets from the remote site are retrieved, all local snippets deleted and then re-created using the remote site list. On creation the snippets are linked to the categories created in step 1.

8. If the user has selected to include chunks all chunks from the remote site are retrieved, all local chunks deleted and then re-created using the remote site list. On creation the chunks are linked to the categories created in step 1.

9. If the user has selected to include plugins all plugins from the remote site are retrieved, all local plugins deleted and then re-created using the remote site list. On creation the plugins are linked to the categories created in step 1.

10. The mapping of plugins to system events is retrieved from the remote site. Any existing mappings are deleted and re-created using the remote site list thus preserving the linkage present in the remote site. If any system events present in the remote site cannot be mapped to local system events they are ignored.

11. Finally the local cache is cleared and the user is informed of the successful outcome of the import.

Evolution to Revolution tag conversion is implicit in the above operations, unlike import facilities provided on the other tabs where this is optional.

If at any time during the above sequence an error is detected, e.g. failure to save a local object then the sequence is halted and the user informed of the error.

A log file is produced of the import operation in the assets/provisioner/tmp directory named evoimport-<date>.log. If an import fails for any reason this log file should be consulted and should be attached to any issues raised in github. In tandem to this the revogateway code logs import access requests into the event log table of the remote Evolution site. These take the form of informational entries with the source field being

named 'RevoGateway '. These entries should also be consulted along with the local import log to ascertain what both sides of the import operation were doing before the failure occurred.

As can be seen from the above sequence the only viable work flow to adopt is to start with an empty Revolution site then perform the import. This will give a base import of categories, templates, resources and template variables, all linked as they were in the remote site. From here the user can then install any local packages the installation needs such as Wayfinder for instance and then back fill any hand crafted snippets, chunks and plugins using the main Provisioner import facilities on the other tabs, thus eventually building the Revolution site from its Evolution counterpart.

Options are provided to include snippets, chunks, or plugins in the above in any combination. This can be useful for scenarios where the bulk of the snippets are needed and the optimal work flow is to delete the ones that are not, rather than manually add the ones that are. Object names are preserved intact excepting snippet names containing the ':' character, this character is converted into the '-' character. The ':' character is illegal in snippet names in Revolution but not Evolution.

Careful handling is needed here for any of these elements that are part of extras in the remote site that overlap with packages that you need to install into the local site. Utilities like Ditto, Jot, Wayfinder et al come to mind here as these will bring over their own chunks etc. that will be overwritten on local package installation, this can cause confusion if not properly handled and may cause package installations to fail if name clashes occur etc.

Plugins are rarely needed to be brought over in bulk as the vast majority of plugins written for Evolution will not work in Revolution. Extras like TinyMCE and QuickManager come to mind here. These should not be imported using this tool as the import automatically associates the plugin with its events, if one of these events is 'OnManagerPageInit' which is the case for these plugins then severe breakage will occur as the associated files that the plugins call will not be present locally. This can be hard to recover from should it occur.

The Import Time Period selection allows the user to adjust the import time period before either the import finishes or times out. This is a value in seconds and defaults to a value of 120, i.e. 2 minutes.

Most import operations should complete well before this time period is reached however for very large sites this may need adjusting. This value is applied to both the front end(AJAX) timers and the backend end script timing using the PHP function set_time_limit. This function will NOT work if you are running in safe mode, in which case you will be limited to the value set for max_execution_time in your php.ini file.

If you do not get the 'import success....' message box and your 'importing...' box closes taking you back to the import tab the import sequence has timed out, increase this timeout and try again. You can look at the import log to see how far through the import you got before the timeout occurred.

10. Under the Hood

The component itself is just a collection of existing manager UI components such as tree's grids etc. from the existing modext tool kit the Revolution manager is based on, adapted for use as a 3PC. It is in fact a 'mini' manager.

This is made possible because the interface to these components is in the form of JSON data generated by the back end processors on request from AJAX events in the GUI. As long as the interface both front and back is respected any GUI can be placed on the 'front end' of the incoming JSON data. This allows desktop embedded widgets to be created as in the Google desktop say or specially crafted applications for mobile devices to be generated.

Evolution of course has no processors that return JSON data needed for Revolution installs, so to communicate with Evolution sites we have to supply some. This is the revogateway package that needs to be installed in remote Evolution sites you wish to provision from. This package is just a set of connectors and processors that take in the request, get the data from the local evolution database(no xPDO here!) and return it in the JSON format needed by the GUI.

Tag conversion from Evolution sites is accomplished by creating an instance of the modParser095 class and calling its 'translate' method on what you need to convert. Simple really!

11.Security

Before using either a remote Revolution site or Evolution site the user needs to log in to the target site. In both cases this sets up a 'manager' session that is then checked on every access, if the check fails the component simply doesn't work.

For remote Revolution sites access is controlled by permissions granted to the account itself as the processors in these sites check policy on every access. This is very granular allowing the provisioning user to have only the level of access needed for site provisioning.

For remote Evolution sites every access initially goes through a gateway connector that checks for an active manager session, if found a constant is defined that is further checked on every included gateway PHP file, if any find that this constant is not set then they 'die' and return nothing. This is identical to the behaviour of included PHP files in Evolution installations that check for 'IN_MANAGER_MODE', if not found they 'die' and exit.

Enforcing this is mandatory for Revolution sites, as if you are not logged in you have no permissions and hence no access. For Evolution sites it stops the gateway code from being used to 'rip' content out of the site by just issuing a correctly formed HTTP request. If a manager login session is not active, nothing happens.

12.   Further development

With the addition of the Evolution Site Import tab we are approaching GA status now for Provisioner. I expect the functionality of this tab to change over the next one or two beta releases, adding user import for instance may be needed, other than this and bugs aside Provisioner is now nearing completion.

Please feel free to update the forum topic with ideas, suggestions, bug fixes etc. and I'll try and incorporate them. Also, raise any issues you may come across in github.