



2.2 CloudStack User Interface Customization

Revised September 7, 2011



Copyright © 2011 Citrix Systems, Inc. All rights reserved. Specifications are subject to change without notice. The Cloud.com logo, Cloud.com, and CloudStack are trademarks or registered trademarks of Citrix Systems, Inc. All other brands or products are trademarks or registered trademarks of their respective holders.

Contents

- 1 Introduction 4
 - 1.1 License 4
 - 1.2 Support 4
- 2 Customization 5
 - 2.1 Getting Started 5
 - 2.2 Simple Branding..... 6
 - 2.3 Advanced Customizations 6
 - 2.3.1 Changing the API URL..... 7
 - 2.3.2 Localization 8
 - 2.3.3 Changing Session Timeout 8
 - 2.4 Single Sign-on Integration 9
 - 2.4.1 Traditional 9
 - 2.4.2 Shared Key 9
 - 2.5 Cross Site Request Forgery (CSRF)..... 10

1 Introduction

The CloudStack™ User Interface (UI) is a rich AJAX client interface that allows you to manage all aspect of the cloud and is a complete reference implementation of the CloudStack API (for more information on the CloudStack API, see below). The CloudStack UI supports three access roles.

- **Root Admin.** Access to all features of the cloud, including both virtual and physical resource management. You can access the 2.2. API via the link : http://download.cloud.com/releases/2.2/api/TOC_Global_Admin.html
- **Domain-Admin.** Access to only the virtual resources of the clouds that belong to the administrator's domain. You can access the 2.2 API via the link : http://download.cloud.com/releases/2.2/api/TOC_Domain_Admin.html
- **User.** Access to only the features that allow management of the user's virtual instances, storage, and network. You can access the 2.2 API via the link : http://download.cloud.com/releases/2.2/api/TOC_User.html

The following document describes the various methods of user interface customization from simple branding to a complete redesign.

1.1 License

The entire user interface is released under the GNU General Public License v3 or later.

It is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or any later version. This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses>.

1.2 Support

Per our license, you are welcome to modify, add, or reuse any part of the CloudStack UI to suit your needs. However, once modified, Cloud.com™ can no longer support any defects resulting from the customization nor can Cloud.com support any upgrade process to future releases of the CloudStack Management Server.

2 Customization

The CloudStack UI is built entirely on HTML/JSP, CSS, Javascript, and uses jQuery 1.4 as the Javascript Library for all AJAX calls, event handling, and animations. You can find the latest jQuery reference API at <http://api.jquery.com/>. Cloud.com recommends that any changes be made by someone with development experience in the listed above technologies. Cloud.com also recommends using a web development tool such as Firebug for Firefox to help inspect the various elements of the UI for easier modification.

2.1 Getting Started

To get started, log into your CloudStack Management Server and go to the following directory to find all the files and resources that make up the user interface.

`/usr/share/cloud/management/webapps/client/`

The following table will describe all the major file or files located in the directory used to build the user interface.

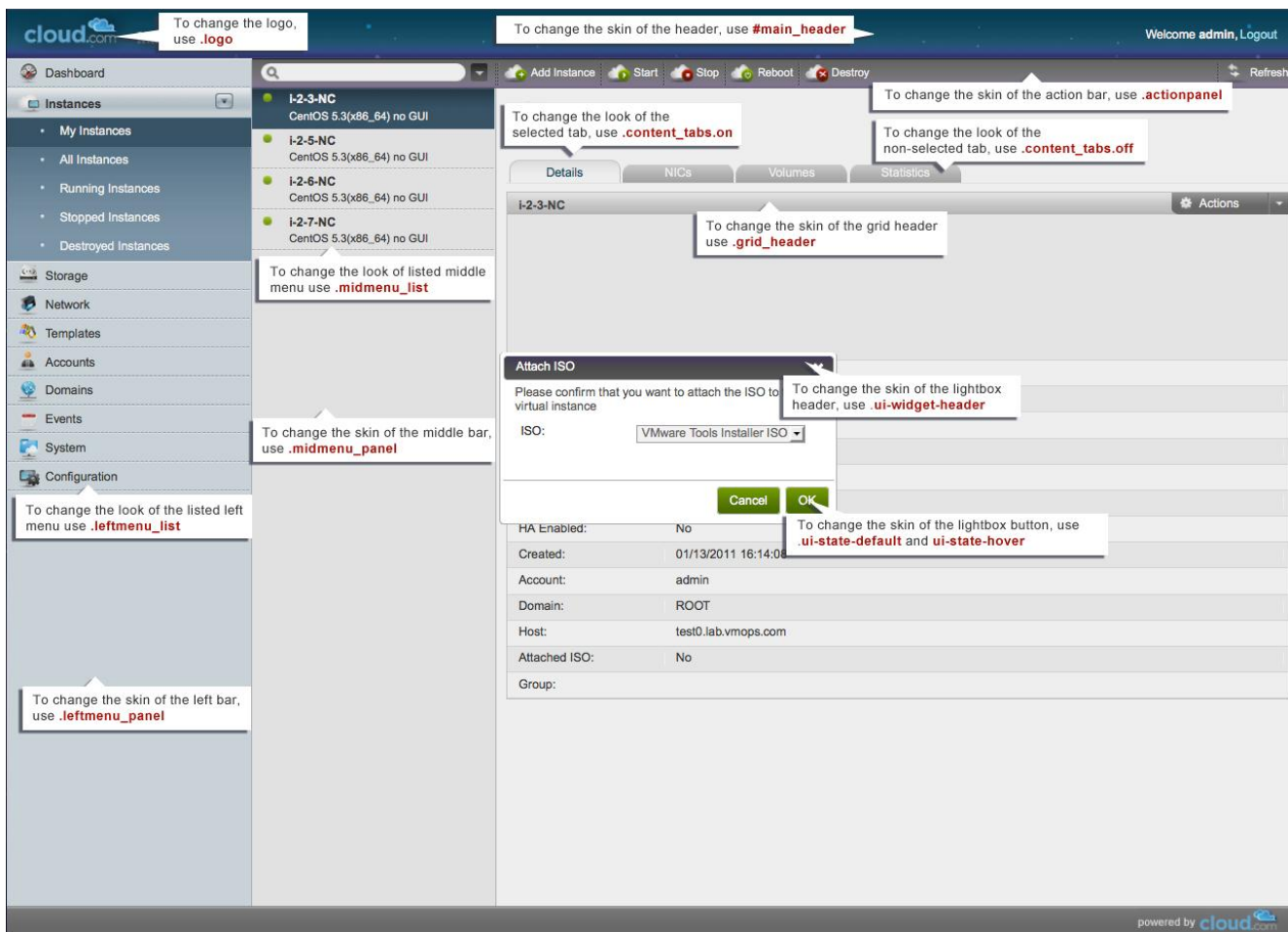
File	Description
index.jsp	The main HTML/JSP page. This is where all CSS and Javascripts are specified.
favicon.ico	Default Cloud.com favicon. Replace this to suit your needs.
/images/*.*	Folder that contains all the images used by the CloudStack UI.
/css/main.css	Main CSS file that contains most of the CSS definitions used by the CloudStack UI.
/css/jquery-ui.custom.css	CSS file used by the jQuery UI library. All default user interface dialog CSS definition are located in this file.
/custom/*.*	Directory that includes all the out-of-box custom HTML, CSS, and Javascript for the default UI.
/jsp/*.jsp	JSP pages that correspond to each major element presented in the user interface.
/script/jquery*.js	Javascript libraries used by the CloudStack UI. You should not have to modify any of these files.
/script/cloud.core.callback.js	Javascript file that you can modify if you wish to integrate the CloudStack UI as a single sign-on solution with your existing website/portal.
/script/cloud.core.js	Javascript file that contains the common functions used by the CloudStack UI.

2.2 CloudStack User Interface Customization

<code>/script/cloud.core.init.js</code>	Javascript file that contains the default initialization logic for the CloudStack UI. This is also the location where you need to specify the default API URL for AJAX calls if you decide to change the default URL.
<code>/script/cloud.core.*.js</code>	Javascript files that correspond to each major element presented in the CloudStack UI.

2.2 Simple Branding

Simple branding is defined as the replacement of the header logo and all major color schemes in the CloudStack UI to match your company's logo and colors. This includes modification of the color of the header, tabs, grid header, and all dialogs. To make these changes, use the reference CSS files found in `/custom/custom*/css`. You can also replace the `favicon.ico` and `/images/cloud_logo.gif` to replace the default Cloud.com images. The diagram below identifies some of the major CSS definitions which are typically modified in simple branding.



The screenshot shows the CloudStack user interface with several callouts pointing to specific UI elements and their corresponding CSS classes for customization:

- Header:** To change the logo, use `.logo`; To change the skin of the header, use `#main_header`.
- Navigation:** To change the skin of the left bar, use `.leftmenu_panel`; To change the look of the listed left menu use `.leftmenu_list`.
- Instances List:** To change the look of listed middle menu use `.midmenu_list`; To change the skin of the middle bar, use `.midmenu_panel`.
- Instance Details:** To change the look of the selected tab, use `.content_tabs.on`; To change the skin of the action bar, use `.actionpanel`; To change the look of the non-selected tab, use `.content_tabs.off`; To change the skin of the grid header use `.grid_header`.
- Dialogs:** To change the skin of the lightbox header, use `.ui-widget-header`; To change the skin of the lightbox button, use `ui-state-default` and `ui-state-hover`.

2.3 Advanced Customizations

The following describes various customizations that can be applied to the user interface.

2.3.1 Changing the API URL

The default host URL on a new installation of the CloudStack Management Service is `http://<server>:<port>/client`. Please refer to <http://tomcat.apache.org/tomcat-5.5-doc/index.html> for documentation on how to change default host URL.

An API URL may need to be changed for the following reasons.

- The default configuration of the Tomcat engine has been changed to your desired URL.
- You have a load balancer or proxy server that is fronting a public host URL that is different from what is currently configured as the default API URL.

If at any point the public API URL is different than what is configured by default, you will need to reconfigure your setup by changing the API path and modifying the `cloud.core.init.js` file. Use the following sections to make these changes.

2.3.1.1 To Edit the API Path

By default, the API URL as configured in the user interface is “client/api” which is relative to the default host DNS/IP. If you would like to change API path, use the following steps.

1. Edit the following file.

```
/usr/share/cloud/management/webapps/client/WEB-INF/web.xml
```

2. Within the file, find the following XML tag:

```
<servlet-mapping>
  <servlet-name>apiServlet</servlet-name>
  <url-pattern>/api/*</url-pattern>
</servlet-mapping>
```

3. Change the `<url-pattern>` tag to the desired API URL.

Once you have changed the API path, proceed to the next section for steps to modify the `cloud.core.init.js` file.

2.3.1.2 To Modify the `cloud.core.init.js` File

If the default API URL has changed from “client/api”, use the following steps to modify the `cloud.core.init.js` file.

1. Edit the following file.

```
/usr/share/cloud/management/webapps/client/scripts/cloud.core.init.js
```

2. Find the following jQuery definition in the file:

```
$.ajaxSetup({
  url: "/client/api",
  dataType: "json",
  cache: false,
  error: function(XMLHttpRequest) {
    handleError(XMLHttpRequest);
  },
  beforeSend: function(XMLHttpRequest) {
```

2.2 CloudStack User Interface Customization

```
        if (g_mySession == $.cookie("JSESSIONID")) {  
            return true;  
        } else {  
            $("#dialog_session_expired").dialog("open");  
            return false;  
        }  
    }  
});
```

3. Modify the URL option (this option is highlighted in red in the above example) to your desired URL.

Once this has been modified, all subsequent AJAX calls will be made to the new URL. You may need to refresh the browser to update any cached Javascript files for the new settings to take place.

2.3.2 Localization

The process of localizing the CloudStack User Interface requires a two step process:

1. Adding a new properties file for the language you wish to localize your UI to. The file must be copied over to the following directory in your management server: `/usr/share/cloud/management/webapps/client/WEB-INF/classes/resources`. The default installation of CloudStack contains sample property files for simplified Chinese, Japanese, and Spanish. Simply make a copy of `messages.properties` and rename the new file using the naming convention of `messages_<lang_code>_<country_code>.properties`. Edit the file and translate the English text to your language of choice.

The valid language codes can be found at <http://www.ics.uci.edu/pub/ietf/http/related/iso639.txt>.

The valid country codes can be found at http://www.chemie.fu-berlin.de/diverse/doc/ISO_3166.html.

2. The second thing that is required is to make sure that you set the cookie, "lang" to be your language + country code. You can either modify the current UI to accommodate the new language or just ensure that you set the cookie yourself. To modify the UI, you would need to edit `index.jsp`, find the language drop down menu by searching for "lang_menu", and finally adding a new `` item to match your new language.

Example:

For example, if you wish to add a French localization, you would make a copy of `messages.properties` and rename it to `messages_fr.properties`. Edit the new file and translate the English text. You would then edit `index.jsp`, and add `<li id="fr"><fmt:message key="label.lang.french"/>` to the dropdown menu.

2.3.3 Changing Session Timeout

The default session timeout for the User Interface is 30 minutes as configured within Tomcat. If you wish to increase this timeout period, use the following steps.

1. Edit the following file.

```
/usr/share/cloud/management/webapps/client/WEB-INF/web.xml
```

2. Within the file, add the following XML tag or modify if it's already there to the desired timeout period in minutes:

```
<session-config>  
    <session-timeout>60</session-timeout>
```



```
</ session-config>
```

3. Reboot the Tomcat container:

```
# service cloud-management restart
```

2.4 Single Sign-on Integration

The user interface is created entirely using the session-based CloudStack API. Once a user account has successfully logged in, a JSESSIONID cookie is sent back as part of the authorization process that can be used until the session has expired on the server. As a result, there are multiple ways that single sign-on can be integrated. Two of these methods are discussed in detail in the following sections. Please feel free to email support@cloud.com if you wish to discuss single sign-on integration that is unique to your environment if the below scenarios do not apply to you.

The two methods of integrating your portal to CloudStack depend on the modification of the “cloud.core.callbacks.js.” This file includes a method, onLogoutCallback(), that can be implemented to redirect the user to your portal if the session times out. The other half of this file include a sample AJAX login API call to the CloudStack management server. You must make the login API from the CloudStack domain or the browser will reject any cross-browser script calls for security reasons. If your CloudStack Management Server and Portal exist within the same domain, you do not have to worry about this. Simply make the login API call from anywhere.

2.4.1 Traditional

The traditional way of integrating an existing portal with CloudStack is to execute the API command “login” on behalf of the user. Using this method, you would need to construct the login command and pass in the required parameters such as the username, account, domain, and password. Upon a successful response, you would only need to ensure that the global variable “g_loginResponse” is set to the JSON response of the login API call. A typical client side single sign-on implementation would look like the following:

- Portal has a link (or iframe) to the CloudStack interface. That link should contain enough information to construct a proper login API call.
- A modified “cloud.core.callbacks.js” intercepts the referred link, constructs the “login” call, and executes it against /client/api URL.
- Upon a successful response, the JSESSIONID cookie will be automatically set by the browser, and the global variable “g_loginResponse” should be set to the JSON response.

2.4.2 Shared Key

The shared key method is very similar to the traditional way except for the additional security by hashing the URL with a shared secret key when making the same login API command. The actual process of signing is very similar to the process described at http://download.cloud.com/releases/2.2/api/global_admin/2.2api_security_details.html under the “API/Secret key security section” with the following exceptions:

- You do not need to pass in the API Key
- The four parameters that must be passed in for the login command are domainId, username, timestamp, and signature.

A sample login request:

<https://<server>:8080/client/api?command=login&username=XXX&domainid=NNN×tamp=YYY&signature=<secure-hash>>

2.2 CloudStack User Interface Customization

To get the single sign-on secret key, you must retrieve it from the CloudStack database under the configuration table for the key “security.singlesignon.key”. Copy this key to the application you wish to integrate CloudStack with and follow the above instructions to sign the login command.

The timestamp parameter is simply the current system time in milliseconds. There is also a fault tolerance configurable value in the configuration table, “security.singlesignon.tolerance.millis” that can be changed to suit your preference. The timestamp passed in as part of the login request needs to be within the CloudStack Management Server time plus the fault tolerance time.

2.5 Cross Site Request Forgery (CSRF)

The CloudStack Management User Interface protects itself from CSRF attacks. Additional information about this can be found at [http://www.owasp.org/index.php/Cross-Site_Request_Forgery_\(CSRF\)](http://www.owasp.org/index.php/Cross-Site_Request_Forgery_(CSRF)).

To protect the User Interface from CSRF attacks, a sessionkey response value is returned upon a successful login attempt. This sessionkey is then passed with all subsequent API command calls. This is different from the JESSIONID and should never be stored in a cookie.

If you plan on implementing your own User Interface on top of the CloudStack Query API, you must ensure the following when using the sessionkey:

- Should ***not*** be stored as a cookie
- Must be returned with every request, for example <http://<server>:8080/client/api?command=XXX&sessionkey=YYY>

If you send any subsequent requests without a valid sessionkey, a 401 Unauthorized HTTP error code will be returned.