CITRIX

CloudPlatform

(powered by Apache CloudStack)

User Interface Customization Guide

For CloudPlatform Version 3.0.x

Revised June 12, 2012 10:06 AM

Contents

Introduction

The CloudPlatform™ 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 CloudPlatform API. The CloudPlatform UI supports three access roles.

- **Root Admin**. Access to all features of the cloud, including both virtual and physical resource management. For more information about this API: http://download.cloud.com/releases/3.0/api_3.0/html/TOC_Global_Admin.html
- **Domain-Admin**. Access to only the virtual resources of the clouds that belong to the administrator's domain. For more information: http://download.cloud.com/releases/3.0/api_3.0/html/TOC_Domain_Admin.html
- **User**. Access to only the features that allow management of the user's virtual instances, storage, and network. For more information about this API: http://download.cloud.com/releases/3.0/api_3.0/html/TOC_User.html

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

Support

You are welcome to modify, add, or reuse any part of the CloudPlatform UI to suit your needs. However, once modified, Citrix Systems, Inc. can no longer support any defects resulting from the customization nor can we support any upgrade process to future releases of the CloudPlatform Management Server.

Customization

The CloudPlatform 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/. We recommend that any changes should be made only by someone with development experience in the above-listed technologies. We also recommend using a web development tool, such as Firebug for Firefox, to help inspect the various elements of the UI for easier modification.

Getting Started

To get started, log in to your CloudPlatform 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 describes all the major files in this directory that are used to build the user interface.

File	Description
index.jsp	The main HTML/JSP page. All CSS and Javascript files are specified in this page.
favicon.ico	Default favicon.
/images/*.*	The folder that contains all the images used by the CloudPlatform UI.
/css/main.css	The main CSS file that contains most of the CSS definitions used by the CloudPlatform UI.
/css/ jquery-ui.custom.css	The CSS file used by the jQuery UI library. The default CSS definitions for all the dialogs in the CloudPlatform UI are located in this file.
/custom/*.*	The directory that includes all the out-of-box custom HTML, CSS, and Javascript files for the default UI.
/jsp/*.jsp	The JSP pages that correspond to each major element presented in the CloudPlatform UI.

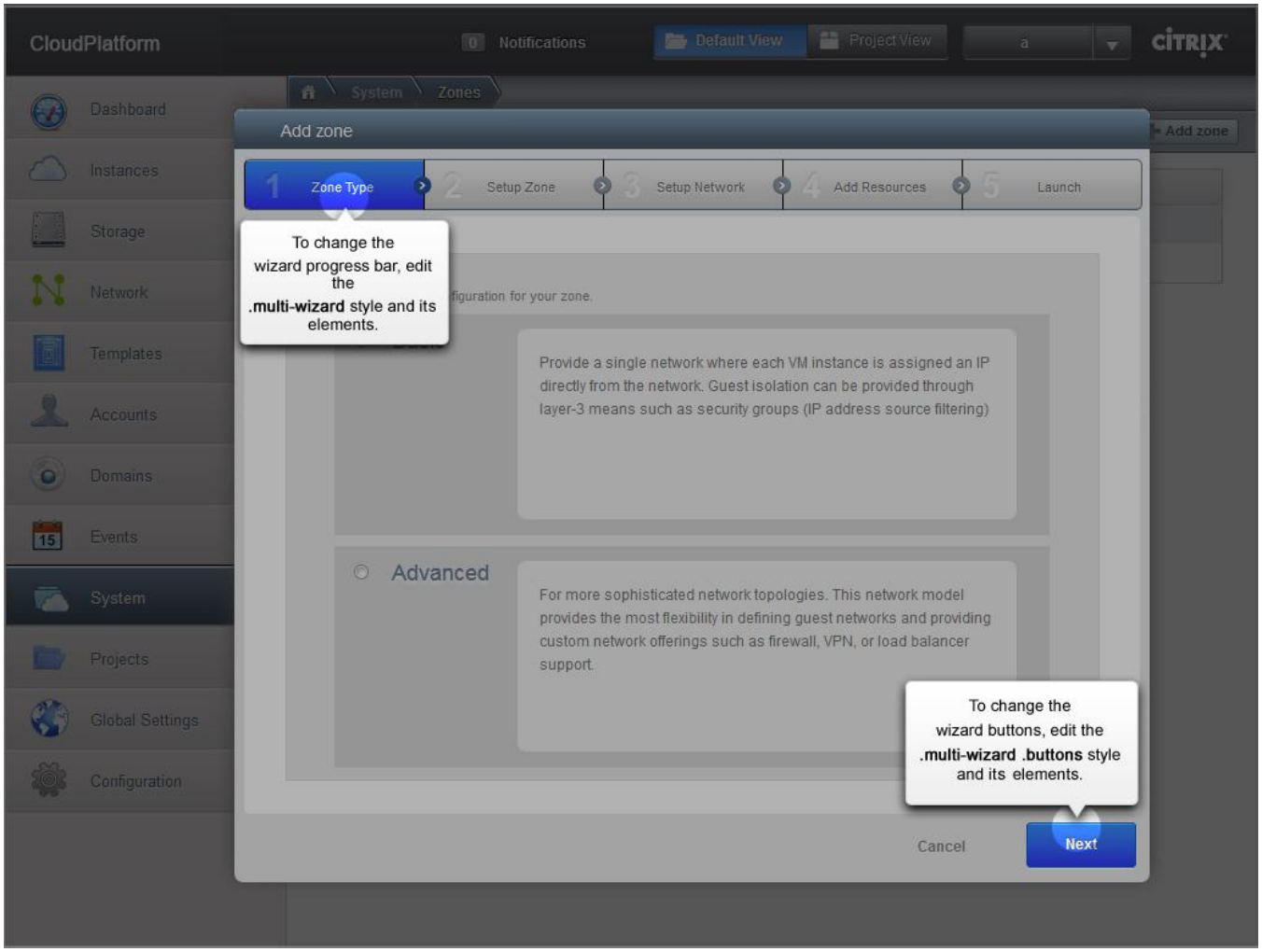
/script/jquery*.js	The Javascript libraries used by the CloudPlatform UI. You need not modify any of these files.
/script/cloud.core.callback.js	The Javascript file that you can modify if you wish to integrate the CloudPlatform UI as a single sign-on solution with your existing website/portal.
/script/cloud.core.js	The Javascript file that contains the common functions used by the CloudPlatform UI.
/script/cloud.core.init.js	The Javascript file that contains the default initialization logic for the CloudPlatform UI. This is the location where you need to specify the default API URL for AJAX calls if you decide to change the default URL.
/script/cloud.core.*.js	The Javascript files that correspond to each major element present in the CloudPlatform UI.

Simple Branding

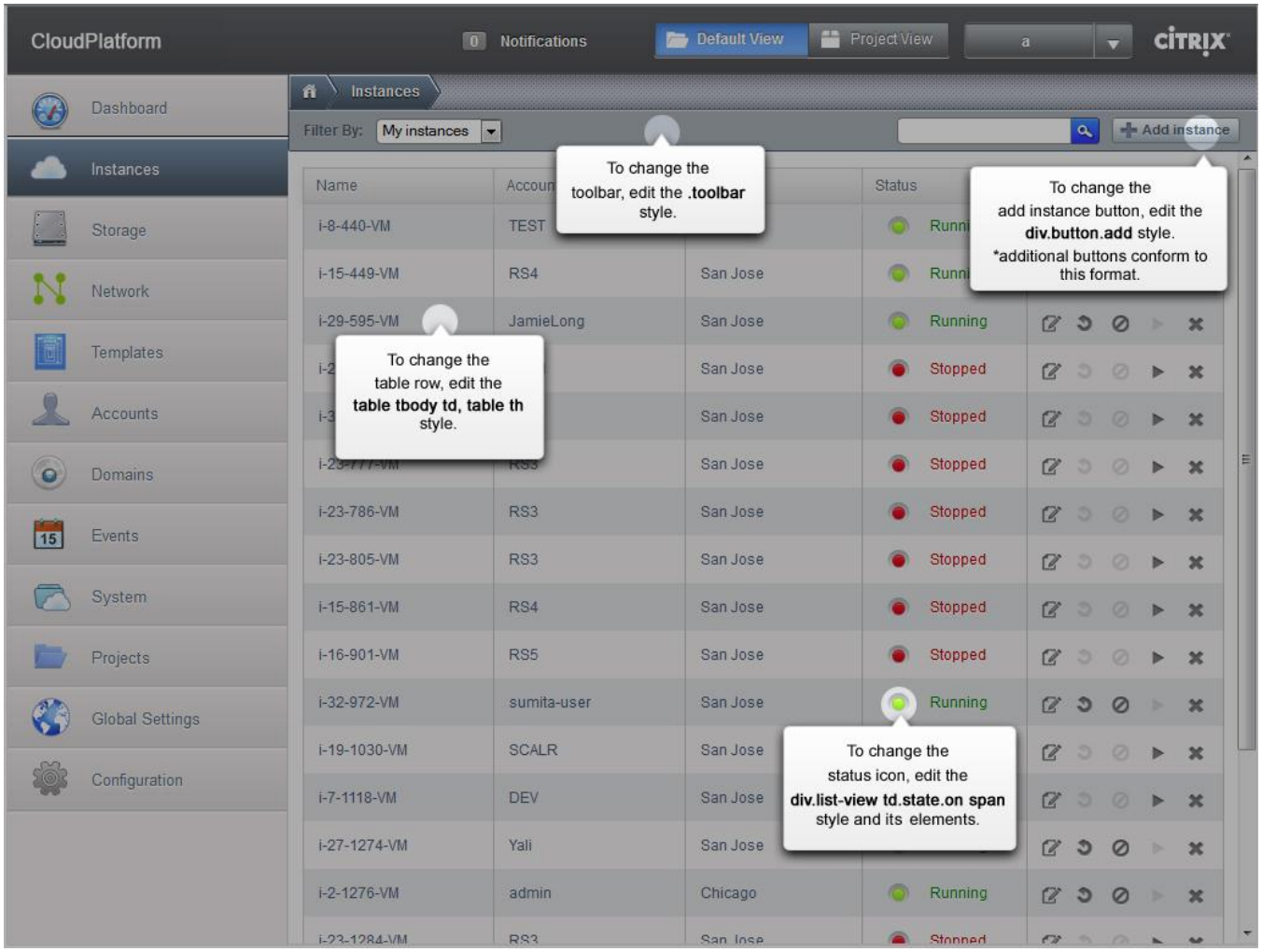
Simple branding is defined as the replacement of the header logo and all major color schemes in the CloudPlatform UI to match your company's logo and colors. This includes modification of the color of the header, tabs, grid header, and all dialogs. To make these changes, use the reference CSS files found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

Customizing the Dashboard

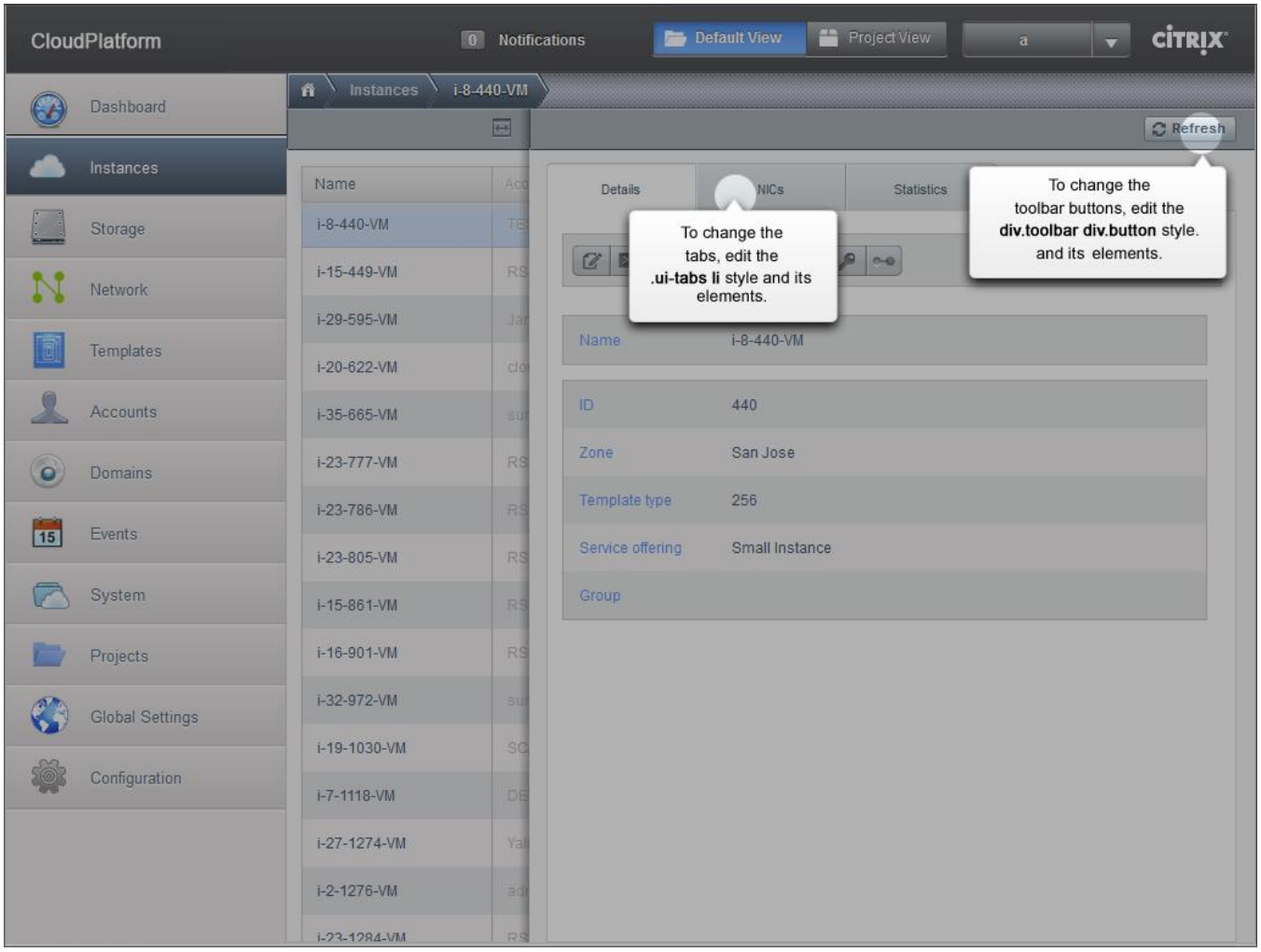
Customizing the Wizard



Customizing the Instances Page



Customizing the Details Page



Advanced Customizations

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

Changing the API URL

The default host URL on a new installation of CloudPlatform is http://<server>:<port>/client. Refer to http://tomcat.apache.org/tomcat-5.5-doc/index.html for information on how to change the default host URL.

You might need to change the API URL 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, and the public host URL 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 need to reconfigure your setup by changing the API path and modifying the cloud.core.init.js file. Use the instructions in the following sections to make these changes.

To Edit the API Path

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

1. Open 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.

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. Open 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(XMLHttpResponse) {
            handleError(XMLHttpResponse);
      },
      beforeSend: function(XMLHttpRequest) {
            if (g_mySession == $.cookie("JSESSIONID")) {
                  return true;
            } else {
                  $("#dialog_session_expired").dialog("open");
                  return false;
            }
      }
});
```

3. Modify the URL option, /client/api, to your desired URL.

   This option is highlighted in red in the above example.

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.

Localization

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

1. Add a new properties file for the language you wish to use.
   a) Copy the property file to the following directory in your management server:

      /usr/share/cloud/management/webapps/client/WEB-INF/classes/resources

   b) Make a copy of messages.properties file and rename the new file by using the naming convention of messages_<lang_code>_<country_code>.properties.

      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
      The default installation of CloudPlatform contains sample property files for simplified Chinese, Japanese, and Spanish.

   c) Edit the file and translate the English text to your language of choice.
2. Set the cookie, “lang” to be your language + country code.
3. 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 need to edit the index.jsp file.
   a) Find the language drop down menu by searching for “lang_menu”
   b) Add a new <li> item to match your new language

For example:

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

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. Open the following file.

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


2. In the file, add the following XML tag or modify the current one 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
```

Single Sign-on Integration

The user interface is created entirely using the session-based CloudPlatform API. Once a user has successfully logged in, a JSESSIONID cookie is sent back as part of the authorization process. This cookie 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.

These two methods of integrating your portal to CloudPlatform 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 includes a sample AJAX login API call to the CloudPlatform management server. You must make the login API from the CloudPlatform domain; otherwise the browser will reject any cross-browser script calls for security reasons. If your CloudPlatform 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.

Traditional

The traditional way of integrating an existing portal with CloudPlatform 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 is as follows:

- Portal has a link (or iframe) to the CloudPlatform 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 JESSIONID cookie will be automatically set by the browser, and the global variable "g_loginResponse" should be set to the JSON response.

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/html/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&timestamp=YYY&signature=<secure-hash>
```

You must retrieve the single sign-on secret key from the CloudPlatform database under the configuration table for the key "security.singlesignon.key". Copy security.singlesignon.key to the application you wish to integrate CloudPlatform with, and follow the above instructions to sign the login command.

The timestamp parameter is simply the current system time in milliseconds. The fault tolerance value, "security.singlesignon.tolerance.millis", is available in the configuration table. You can change that value to suit your preference. The timestamp passed in as part of the login request needs to be within the CloudPlatform Management Server time plus the fault tolerance time.

Cross Site Request Forgery (CSRF)

The CloudPlatform 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).

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 JESSSIONID and should never be stored in a cookie.

If you plan to implement your own User Interface on top of the CloudPlatform 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.

Contacting Support

The CloudPlatform support team is available to help customers plan and execute their installations. To contact the support team, log in to the support portal at https://na6.salesforce.com/sserv/login.jsp?orgId=00D80000000LWom by using the account credentials you received when you purchased your support contract.