VERSION 6.0.1

Docent Enterprise

Localization Guide

Docent, Inc.
2444 Charleston Road
Mountain View, CA 94043-1622
650-934-9500
support@docent.com

Designed and written at Docent, Inc., 2444 Charleston Road, Mountain View, CA 94043-1622

Printed in the USA.

Part Number: 06/02

Disclaimer of Warranty

DOCENT, INC. DISCLAIMS, ON BEHALF OF ITSELF AND ITS SUPPLIERS, ALL EXPRESS OR IMPLIED WARRANTIES OR CONDITIONS RELATED TO ANY EVALUATION LICENSE AND DISCLAIMS, ON BEHALF OF ITSELF AND ITS SUPPLIERS, ALL OTHER WARRANTIES AND CONDITIONS, EITHER EXPRESS, IMPLIED OR STATUTORY, INCLUDING, BUT NOT LIMITED TO, IMPLIED WARRANTIES OR CONDITIONS OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND/OR NONINFRINGEMENT. DOCENT, INC. DOES NOT WARRANT THAT THE SOFTWARE WILL MEET CUSTOMER'S NEEDS OR THAT IT WILL OPERATE IN COMBINATION WITH SOFTWARE OWNED BY CUSTOMER.

Limited Liability

IN NO EVENT SHALL DOCENT, INC. OR ITS SUPPLIERS BE LIABLE FOR ANY CONSEQUENTIAL, INCIDENTAL, OR SPECIAL DAMAGES WHATSOEVER (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF PROFITS, BUSINESS INTERRUPTION, LOSS OF INFORMATION, OR OTHER PECUNIARY LOSS) ARISING OUT OF THE USE OF OR INABILITY TO USE THE SOFTWARE, EVEN IF DOCENT, INC. HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. DOCENT, INC.'S AGGREGATE CUMULATIVE LIABILITY, WHETHER IN CONTRACT OR TORT OR OTHERWISE, WILL NOT EXCEED THE AMOUNT OF INTERNAL USE LICENSE FEES PAID TO DOCENT, INC. HEREUNDER.

Notices

All notices permitted or required under this Agreement shall be in writing and shall be by personal delivery, telegram, telex, telecopier, facsimile transmission, or by certified or registered mail, return receipt requested, and shall be deemed given upon personal delivery, five (5) days after deposit in the mail, or upon acknowledgment of receipt of electronic transmission. Notices to DOCENT, INC. shall be sent to the address set forth at the end of this Agreement or such other address as DOCENT, INC. may specify in writing. Notices to Customer shall be sent to the last address of the Customer in DOCENT, INC.'s possession, or such other address as Customer may specify in writing.

General

Nothing herein shall be construed as creating any agency, partnership, or other form of joint enterprise between the parties. This Agreement is governed and interpreted in accordance with the laws of the State of California, excluding its conflict of law rules. The United Nations Convention on Contracts for the International Sale of Goods is expressly disclaimed. If any provision of this Agreement shall be deemed invalid by a court of competent jurisdiction, then such provision shall be deemed modified to the minimum extent necessary to make it enforceable, and the validity of the remaining provisions of this Agreement shall not be affected. This Agreement is the entire and exclusive agreement between DOCENT, INC. and Customer with respect to the Software and supersedes all prior agreements (whether written or oral) and other communications between DOCENT, INC. and Customer with respect to the Software. This Agreement shall not be modified except by a subsequently dated written amendment signed on behalf of Customer and DOCENT, INC. by their duly authorized representatives, and any purchase order or other document purporting to supplement the provisions of this Agreement shall be void.

Docent Incorporated, 2444 Charleston Road, Mountain View, CA 94043-1622

Phone: (650) 934-9500 Fax: (650) 962-9411 Email: support@docent.com

Contents

Preface

Thank you for choosing Docent Enterprise, Docent's suite of optional, separately priced products that help global businesses improve performance by increasing the productivity and effectiveness of individuals and aligning them with the goals and objectives of the corporation.

Docent Enterprise software supports your continuous learning strategy every step of the way from customized curriculum planning, course authoring and delivery, competency assessment, and results tracking to business impact and return on investment analysis.

This chapter discusses the objectives, audience, organization, and conventions used in this guide. It also describes the Docent Enterprise documentation set, as well as how to obtain additional information about Docent products and services.

Objectives

This guide provides instructions for maintaining and customizing a localized Docent Enterprise installation.

Audience

This guide is written for the following types of users:

- System Administrators of multi-lingual sites
- People who customize Docent Enterprise
- Users of Language Packs

Organization

This guide is organized into the following chapters:

Chapter 1, “Overview”—provides an overview of Docent Enterprise localization issues.

Chapter 2, “Localizing Customized App Files”—provides information on the format of resource files and how to create and customize them.

Chapter 3, “Configuring Locales”—provides reference material useful for customizing locale App files and other localization issues.

Chapter 4, “Localizing Entities”—describes the process for localizing entities in Docent Enterprise.

Documentation Conventions

Docent documentation uses the following conventions:

- `Monospaced text`

  Represents examples of text that displays on the screen or in a file. Also represents file names, directories, and command variables.

- `<Bracketed_Monospaced_Text>`

Represents variables that are dependent upon your system configuration. For example, `<Docent_Installation_Directory>` indicates the directory you specified for your Docent Installation.

- **`Boldfaced Monospaced text`**

  Indicates commands and keywords you enter as shown.

Docent Documentation Set

The Docent Enterprise documentation set includes:

- *Product Overview and Installation Guide*

  Provides an overview of all Docent Enterprise products and includes installation and upgrade instructions for the Docent Learning Management Server and the Docent Content Delivery Server.

- *Content Authoring Guide*

  Explains how to use Docent Outliner to create Docent online modules. It also explains how to add content, create questions ad scoring, and customize layouts and themes.

- *Content Delivery Administration Guide*

  Explains how to build and publish Docent online modules and CDS reports. It also explains how to work with shared content and deliver modules and CDS reports to remote and mobile students with Docent Mobile.

- *Learning Management Guide*

  Describes the Learning Management Server, Reference Application, App files, and the scripting object model for use in customizing App files.

- *Docent LMS Integrations* (online document)

  Docent works with third-party vendors that supply courses you can integrate with the Learning Management Server. This online document, included on the Docent Enterprise CD, provides details on integrating third-party courses.

- Docent API documents (online documents)

  There are two Docent API documents. The *Docent API Guide* provides an overview of the Docent API including the command line interface, arguments, and C++ reference information. The *Docent Java API Guide* (online document) includes Java reference information for the API.

- Docent Developer Reference Documentation (online documents)

  A set of online reference material. This includes descriptions of the Docent database schema, JavaScript objects, and the JavaScript library.

Accessing Docent Documentation

Docent documentation is available in both hardcopy and online format. Hardcopy documents include the *Product Overview and Installation Guide*, the *Content Delivery Administration Guide*, and the *Learning Management Guide*.

The complete Docent documentation set is provided online. You can:

- Access the documentation set directly on the Docent CD.

  To access copies of Docent documentation, go to the documentation directories on the CD or follow the links to the documentation from the `README.htm` file.

- Install the documentation set with Docent Outliner and access it from the Outliner Help menu.

- Install the Docent documentation set with the Docent Servers. Typically, you can access these manuals by entering the following URL in your Web browser:

  `http://<web_server>/docent/docs`

  where `<web_server>` is the host name of your Web server. See your Docent system administrator for more information.

Providing Feedback

The Docent Technical Publications team is interested in knowing what aspects of our documentation you find helpful or lacking. If you would like to send feedback, use the online form at:

`http://www.docent.com/feedback/documentation.html`

Where to Find Additional Help

In addition to the Docent Enterprise documentation set, Docent offers support documentation, customer assistance, and professional services.

Docent Enterprise CD

The Docent Enterprise CD contains installation files, documentation, and additional resources. It also includes the `READMEFIRST.htm` document which contains the most current information on system requirements, new features, and last minute changes. To order additional copies of the CD, contact your local sales representative or call customer service.

The Docent Enterprise CD also includes a set of sample Docent online modules, CDS reports, CDS shared content, and so on. For more information, see the `READMEFIRST.htm` document.

Docent Language Pack

The Docent Language Pack contains installation files, documentation, and additional resources. It also includes the `README.htm` document which contains the most current information on Language Pack features and last minute changes. To order additional copies of the CD, contact your local sales representative or call customer service.

Docent Online

You will find the latest breaking news, product developments and support information on our Web site, **`www.docent.com`**.

Docent Customer Support

Docent provides domestic and international phone, email, and fax support to registered users. For a complete list of support services and locations, click the Support link on the Docent Web page (**www.docent.com**).

When you contact Docent support, please have information available regarding the setup of your computer (including system configuration, network and server software, and technical specifications) if these are related to your problem.

Docent Extended Services

Docent's experienced professionals are available to provide training and education that will meet your needs, fit your schedule, and bolster your resources. Docent courses are available in online and instructor-led formats. Course developers, learning consultants, and course administrators can attend courses at our offices, or Docent can provide instruction at your site. Courses can also be customized to meet the unique needs of your business.

Docent's Extended Services include:

- Online Courses (Available through Docent University)

  Docent provides an online series of marketing, sales, and basic and advanced technical courses through Docent University.

- Classroom Training (Available through Docent University)

  Choose from a variety of one-day classroom training courses, available at Docent facilities on a regularly scheduled basis. These courses range from basic instruction in the use of the software to advanced topics such as data integration with other applications, custom report generation, and advanced programming. Courses consist of both lecture and hands-on computer lab exercises. Class sizes are kept small to ensure personal attention for every student.

- Customer-Site Training (Available through Docent University)

  To maximize your employee productivity, Docent instructors can provide education on site at your facility.

- Consulting (Available through Docent Professional Services)

  Our experienced team can guide you through assessment, design, development, implementation, integration, maintenance, testing, and beyond. Docent Professional Services will ensure that your Enterprise Learning Solution is customized to meet your company's unique infrastructure, integration needs, and business goals.

For more information about Docent Professional Services call Docent at (650) 934-9500, Monday through Friday, between the hours of 8 a.m. and 5 p.m. (PST), excluding holidays. Alternatively, you can visit our Web site, **`www.docent.com`**.

1

Overview

At the core of the Learning Management Server is a group of proprietary files from Docent called Application (App) files. All application display logic, navigation logic, and business logic is contained in App files. These files can be customized to meet the needs of your users.

This book contains information on how to localize customized content and customize locales for your Docent Enterprise system.

This chapter contains the following sections:

Overview

If you have customized App files in your Docent Enterprise system, text and graphics you have added will not be translated by the installed Language Pack. To enable translations for this customized content, you must update the locale resource file to contain translations for the customized content.

You may also want to update the resource file if you are using a Language Pack of an earlier version than your LMS, since there will be no translations for new strings that have been added to the interface.

You can also make changes to date and time formats, currency formats, and other locale-specific information. These settings can be edited in the locale App file.

To customize your locale and localize customized App files, you must perform the following steps:

- Update the resource file to include the customized text strings. See “Preparing App Files for Localization,” on page 2-2 for more information.
- Provide translations for the new strings in the resource file. The rt2res program described in “Using the rt2res Program,” on page 2-5 provides a framework within which translated strings can be placed.
- Make any necessary changes to the locale App file (such as regional formats for time, date, currency, etc.). See Chapter 3, “Configuring Locales” for detailed information.

Locales

A locale in Docent Enterprise consists of a language and, if required, a country. Associated with the locale are:

- The language for text display
- Regional time and date formats
- Regional currency and numeric formats

- Regional name formats

Note: While currency and numeric formats are associated with a locale, the currency type is not. You must set the currency type (or types) for your system in the Learning Management Server. For more information on setting the currency type, see the *Learning Management Guide*.

Each user has an associated locale. This locale is selected during the registration process. Once the locale has been chosen, each time the user logs in, text, time and date formats, and other localizable features appear as specified by the user's locale.

Locales are represented within Docent Enterprise by two-letter ISO codes specifying the language and, if applicable, the country associated with the language. The language and country code together define a specific translation. For example, `es` identifies the Spanish translation, and `es_mx` identifies the Mexican Spanish translation.

The country attribute is optional.

For the latest ISO language and country codes, check the following Unicode websites:

```
http://www.unicode.org/unicode/onlinedat/languages.html
http://www.unicode.org/unicode/onlinedat/countries.html
```

Resource Files

To enable rapid translation of the Learning Management Server interface, App files in the Learning Management Server have a special set of HTML tags to indicate blocks of static text that can be replaced with a translated version of the text. The translations for these tagged text strings are contained in a resource file.

When the Learning Management Server converts an App file (`.jsm` file) into a `.jse` file, it uses the locale of the current user to select the corresponding resource file. (For more information on how App files are processed, see the *Learning Management Guide*.)

Based on a user's locale, the server matches text strings marked in the `.jsm` file with the text strings in the matching resource file.The following is a simplified example of the search criteria used for text replacement:

- A text string from the resource file with a matching language code (and country code, if applicable) is used.
- A text string from the resource file with a matching language code, but no country code is used.
- The site default locale is used. The site default locale is set on the "Edit System Setup Parameters" screen in the Learning Management Server.
- The root locale is used (`default.res.xml`).

The root locale is the locale of last resort. The strings in this file are displayed only if there is no more appropriate locale on the system that contains a matching string.

Using <rt> Tags

In each App file, static text is surrounded by the tags, `<rt> </rt>`, where `rt` is an abbreviation for `resource_text`. By placing these tags around a given block of text, the App file author signals the Learning Management Server that this text block should be substituted at runtime with a corresponding value found in the associated resource file.

Whenever you add a static text string to an App file, it is important that you enclose it in `<rt>` tags. When you run the rt2res program to generate or update a resource file, it will not generate an entry for any string that does not have `<rt>` tags. Also, the presence of untagged static text strings in an App file may result in conflicts during the upgrade process.

The `<rt>` tag has two attributes: `"key"` (required) and `"translate"` (optional).

The "`key`" attribute is an alphanumeric identifier used to look up the resource tag referenced in the App file to find the matching string resource in the resource file.

The "translate" attribute defaults to "true" and indicates to a human translator whether or not to translate the string. This allows for strings that must be localized by a method other than translation, such as HTML image tags, which are localized by changing the referenced file name and path.

Resource File Format

Resource files are in XML format and must always use UTF-8 encoding. While the file itself is encoded in UTF-8, the encoding_hint attribute specifies the encoding for the LMS server to use when generating the pages to be displayed. The following is an example of the basic structure of a resource file:

```
<?xml version="1.0" encoding="UTF-8"?>
<resources nextkey="2957">
<encoding_hint>windows-1252</encoding_hint>
<rt key="00004">
<!-- Referred in:  selfreg.jsm -->
<src>First Name</src>
<translation>Prénom</translation>
</rt>
<rt key="00005">
<!-- Referred in:  selfreg.jsm -->
<src>Last Name</src>
<translation>Nom de Famille</translation>
</rt>
</resources>
```

The text located between <src> </src> tags is generated as an exact copy of the text in the App file marked with the <rt> </rt> tags. In the previous example, the App file should have the following text somewhere inside the file:

```
<rt key="00004">First Name</rt>
<rt key="00005">Last Name</rt>
```

Note: Key numbers must be enclosed in double quotes.

Displaying Multiple Languages

In some cases you may need to display more than one language on the same page. For instance, you might have courses created in the Japanese locale that are also available in the English locale. Because the standard English

character encoding cannot display Japanese characters, the course title will not be properly displayed.

The encoding in which a page is displayed is determined by the `encoding_hint` setting. Each locale has a default `encoding_hint` set in its resource file, i.e., `<docent_home>/content/lms/resources/<locale>.res.xml`, where `<locale>` is the name of the locale. For example, the Japanese locale resource file is named `ja.res.xml`. You can customize the default `encoding_hint`. You can also override the default `encoding_hint` by changing the charset attribute of the HTML meta tag in the App file.

Most legacy encodings include lower ASCII characters, so mixing English with any other language often works if you use the other language's legacy encoding. So, for example, to display Japanese and English on the same page, you can use the Japanese encoding SHIFT-JIS.

However, if these same English users will see Russian as well as Japanese, SHIFT-JIS is no longer sufficient. UTF-8 is the only encoding that properly displays all languages. However, if you are using the UTF-8 encoding, you must make sure all of your users have a browser that is compatible with UTF-8. Every browser works differently, but generally there is a font associated with each encoding. Users must configure their browsers to use the correct font when displaying characters in a specific encoding.

For more information on the character encodings supported by Docent Enterprise, see "Supported Encodings," on page 2-4.

Guidelines for Text Strings

The following guidelines explain how to author an App file from which a properly constructed resource file can be generated. If you follow these guidelines, the resource files you create will be easier to localize.

Resource files should contain only translatable text.

Resource files should contain plain text only. They should not contain any SQL, JavaScript, or HTML. Typically, translators are not familiar with programming syntax. Including such syntax risks translators making

inappropriate changes to programming logic. It also precludes (or at least makes extremely difficult) using any kind of automatic translation.

HTML tags are also not translatable text. If certain words in a full phrase or sentence must be enclosed in HTML tags, rather than breaking up the string, use the `messageFormat()` function to insert the static HTML. For more information on this function, see “Using the messageFormat function,” on page 1-11.

If you cannot avoid the use of non-translatable text, set the `translate` attribute for that string to `false` to alert translators not to attempt a translation.

Each translatable string should be a full sentence or phrase.

Do not fragment the strings that will appear in the user interface. This makes them more difficult to translate. Follow this guideline even if it means that some words will appear more than once in the resource file.

Do not concatenate strings.

Do not use programming logic to concatenate static phrases or pluralize words. Instead use programming logic to choose between separately translated phrases or words.

If concatenation is required, use a positioning function to allow reordering. Use the `messageFormat()` function to insert dynamically generated text into a localizable string. The translator will then be able to reposition the dynamic text as appropriate for the target language.

String delimiters are not translatable text.

Quotation marks should go outside the resource tags unless they are intended to be displayed to the user. See the next section for informtion on the types of quotation marks you can use.

Do not include double-quotation marks within text strings.

Double-quotation marks (") must go outside the resource tags. They cannot be included within the text string. You can use single-quotation marks (') within your text strings.

Assets

Assets (gif, jpg, other media files) are also localized using the resource files. All references to asset files should be localizable. Example 4 demonstrates Docent's recommended way of doing this.

Following are some examples of recommended and not recommended styles.

Example 1

Not recommended:

This line in an app file:

```
<b><rt>${(can_see_admin_reports ? "Student" :
"")}Reports</rt></b>
```

Generates this in the resource file:

```
<rt>
<src>${(can_see_admin_reports ? "Student" : "")}Reports</src>
<translation></translation>
</rt>
```

Recommended:

This line in an app file:

```
<b>${(can_see_admin_reports ? "<rt>Student Reports</rt>" :
"<rt>Reports</rt>")}</b>
```

Generates this in the resource file:

```
<rt>
<src>Student Reports</src>
<translation></translation>
</rt>
<rt>
<src>Reports</src>
<translation></translation>
</rt>
```

The text that appears in bold is what you are asking the translator to understand and provide a translation for. The first example is complex and requires that the translator understand the JavaScript conditional operator

and the escape mechanism in order to know what to translate and what to leave alone. In contrast, the translatable text in the recommended example consists of full phrases as they will be seen in the user interface.

Example 2

Not recommended:

App file:

```
<rt>"for dates between " + niceDate(start_date) + " and " +
niceDate(end_date)</rt>
```

Corresponding resource file:

```
<rt>
<src>"for dates between " + niceDate(start_date) + " and " +
niceDate(end_date)</src>
<translation></translation>
</rt>
```

Recommended:

App file:

```
var date_array = [niceDate(start_date), niceDate(end_date)];
var date_text = messageFormat ("<rt>for dates between {0} and
{1}</rt>",
date_array);
```

Corresponding resource file:

```
<rt>
<src>for dates between {0} and {1}</src>
<translation></translation>
</rt>
```

The recommended example shows how to use the JavaScript `messageFormat()` function. Again, the text in bold is what the translator will have to work with. The use of `messageFormat()` allows us to present the translator with a full sentence to translate with no intervening code. The translator can move the bracketed numbers around in the translated text to achieve the proper position in the target language.

Example 3

Recommended:

App file:

```
var bold_array = ["<b>", "</b>"];
var bold_text = messageFormat ("<rt>Here is some {0} emphasized
{1}text</rt>", bold_array);
```

This example shows the recommended way of including HTML tags in a string that must be localizable. A comment in the resource file can explain that the message arguments are HTML tags that should stay on either side of the word or words they are intended to affect.

Example 4

Recommended

App file:

```
var manager_image = "<img " +
messageFormat ('<rt translate="false">src="{0}/manager.jpg"
</rt>',
[UserDomain.imageRoot()]) +
' border="0" alt=' +
'"<rt>Manager</rt>"' +
">";
```

Corresponding resource file:

```
<rt translate=”false”>
<src>src=”{0}/manager.jpg”</src>
<translation>src=”{0}/lang/fr/manager.jpg”</translation>
</rt>
<rt>
<src>Manager</src>
<translation>Responsable</translation>
</rt>
```

This example shows the recommended way of localizing an asset reference. The JavaScript `${manager_image}` can be used in the HTML as the full image tag. This generates two resource strings in the resource file: one for the `alt` tag, and one for the path to the asset. Localizing the path to the asset consists of adding the appropriate language subdirectory.

Note: The translate attribute defaults to `true`. Setting it to `false` alerts the translator that this is not translatable source text. In this example, the source text will have the language subdirectory added.

Using the messageFormat function

The `messageFormat` function allows you to specify the format of a text string in a language-independent way by inserting message arguments from a numbered array into a string. This function appears in `<docent_home>/content/lms/lib/message_format.jsm`.

The `messageFormat` function has the following usage:

```
function messageFormat(msg, args_array)
```

The `msg` parameter is an rt-tagged string. This string is expected to contain a series of numbers enclosed in curly braces. These numbers should match items in the `args_array` parameter. The `args_array` parameter is an array of strings.

The following is an example of how the `messageFormat` function is used:

```
messageFormat ("<rt key="12345">The {0} cat jumps over the
{1}.</rt>", ["blue", "dog"]);
```

When a resource file is generated from the above, the result is as follows:

```
<src>The {0} cat jumps over the {1}.</src>
```

Translators can rearrange the position of the numbers as necessary. For example, a Spanish translator might supply the following translation:

```
<translation>El gato {0} salta sobre el {1}</translation>
```

The `messageFormat function` can also be used to insert non-translatable elements into a string, such as HTML formatting elements. For an example of this usage, see “Example 3,” on page 1-10.

2

Localizing Customized App Files

This chapter describes the procedures for localizing customized App files. It contains the following sections:

For more detailed information on App files and how they work, see the *Learning Management Guide*.

Preparing App Files for Localization

If you create your own custom App files or customize your Docent App files, Docent provides the rt2res program to help prepare your App files for localization.

The rt2res program generates an XML resource file for each locale. This program is located in `<docent_home>/bin/rt2res.pl`. It can also be used to update existing resource files.

Note: rt2res does not translate text strings. It creates a framework within which translated strings can be inserted. You must provide the translations for customized strings.

`rt2res.pl` is a Perl script. It can be modified to handle non-standard installations.

Before You Start

To use `rt2res.pl`, be sure you are running Perl 5.6 or later and that the XML::DOM package is installed. If you are unfamiliar with the process of installing Perl modules, Docent recommends that you contact your I.T. department for assistance.

To determine what version of Perl you are running, type `perl -v` at a command prompt.

Note: If you do not have Perl installed, you can download it from `http://www.activestate.com/Products/ActivePerl/`. Extensive documentation and support for installing and configuring Perl is available on this site.

If you are using ActivePerl, the PPM (Perl Package Manager) is the easiest way to install a module. See the following URL for detailed instructions on using the ActivePerl PPM:

`http://aspn.activestate.com/ASPN/Reference/Products/ActivePerl/faq/ActivePerl-faq2.html`

If you are using any other version of Perl, you can download and install XML::DOM using CPAN. In the example below, `<perl_install_dir>`

refers to the full path of the Perl installation on your system, i.e., `/usr/local/perl`.

To install XML::DOM, do the following:

1. Open a DOS command window (Windows) or a terminal window (Solaris).
2. Navigate to your Perl installation's "bin" directory:

   Windows:

   ```
   C:\> cd <perl_install_dir>\bin
   ```

   Solaris:

   ```
   %> cd <perl_install_dir>/bin
   ```

3. Type the following at the command prompt:

   ```
   %> perl -MCPAN -e 'install XML::DOM
   ```

   Comprehensive documentation on installing Perl modules using CPAN is available at the following URL:

   ```
   http://theoryx5.uwinnipeg.ca/CPAN/perl/pod/perlmodinstall.html
   ```

Command Syntax

`rt2res.pl` has the following syntax:

```
rt2res.pl [-g locale,... -e encoding [-d] [-k prefix | -f prefix |
-r]] [-u locale,...] [-v] [-h] <content_dir> [<root_dir>]
```

where `<content_dir>` is the root of the output content directory and `<root_dir>` is the root of the input directory for the `.js` and `.jsm` files. Resource files are created in `<content_dir>/resources`.

`<root_dir>` defaults to `<content_dir>`.

The following table lists the supported command line options:

Table 2-1 Command options for rt2res

Option	Description
-g locale,...	Generates <locale>.res.xml files for each locale in a comma-separated list of locales. Example: -g en,fr,es_mx
-e encoding	Required when generating a resource file. Specifies the string to be used as the encoding hint for the locales for which resource files are generated. Must be one of the supported encodings listed in “Supported Encodings,” on page 2-4.
-d	Generates <locale>.res.xml files as “development language” files in which source and translation strings are the same. Omitting -d yields empty `<translation></translation>` elements.
-k prefix	Creates keys and inserts them into the .jsm file when encountering <rt> tags without ‘key’ attributes. Keys are of the form "XXXYYYYY00001", where XXX is the prefix, YYYYY is a random number, and 00001 is an incrementing number. IMPORTANT: To avoid conflicts during upgrade, do not use the same numeric range as the Docent keys. Always use a prefix to avoid such conflicts.
-u locale,...	Updates existing resource files by adding resources from the App files that are not yet present in the resource file. Does not create keys.
-v	Output progress messages during script execution.
-h	Display a list of the available command options.

Supported Encodings

The following is a list of the character encodings supported by Docent Enterprise:

```
US-ASCII
ISO-8859-1
ISO-8859-2
ISO-8859-3
ISO-8859-4
ISO-8859-5
ISO-8859-6
ISO-8859-7
ISO-8859-8
ISO-8859-9
ISO-8859-10
ISO-8859-13
ISO-8859-14
```

```
ISO-8859-15
CP866
CP874
WINDOWS-1250
WINDOWS-1251
WINDOWS-1252
WINDOWS-1253
WINDOWS-1254
WINDOWS-1255
WINDOWS-1256
WINDOWS-1257
WINDOWS-1258
SHIFT_JIS
EUC-JP
ISO2022-JP
BIG5
GB2312
GBK
HZ-GB-2312
EUC-CN
EUC-TW
ISO2022-CN
ISO2022-CN-EXT
EUC-KR
ISO2022-KR
KOI8-R
KOI8-U
UTF-8
```

Using the rt2res Program

The rt2res program should be used to maintain all resource files. There should be no need to manually edit a resource file except when adding a translation.

Updating an Existing Resource File

During customization, new strings are added to the App files. In order for these strings to be localizable, they must be enclosed in `<rt></rt>` tags and added to the resource files.

When you add a new string to an App file, enclose it in `<rt></rt>` tags. Do not include a "`key`" attribute in the opening `<rt>` tag. Keys are inserted automatically by `rt2res.pl`.

Important: Since the strings in the root locale (default.res.xml) are only displayed if there is no more appropriate translation, translating the original English strings in the root locale (`default.res.xml`) into another language will cause conflicts during upgrade.

a For the root locale:

To add keys and extract the new strings into the resource file, first re-generate the root locale resource file.

If all of the strings in the App file are in English, you can generate the English (`en`) resource file at the same time by replacing "`default`" in the example below with "`default,en`". This eliminates the need to manually enter English translations for each string. If the strings are not in English, you must generate the English resource file separately.

```
perl rt2res.pl -g default -e windows-1252 -d -k XYZ
c:\<docent_home>\content\lms
```

In this example, "`XYZ`" represents the key prefix you want to use and `c:\<docent_home>\content\lms` is the content directory.

Use the `-k` parameter with a prefix of your choice to distinguish your resource strings from those provided by Docent. This is especially important when new versions of the LMS become available, because your strings must have different keys than the Docent-supplied strings.

b For Other Languages:

Next, update resource files for other languages to include the new strings. For example, issue the following command to update the Gaelic resource file:

```
perl rt2res.pl -u ga c:\<docent_home>\content\lms
```

The generated translation elements are empty and ready to send off to a translator.

Creating a New Resource File

To create a resource file for a new locale, use `rt2res.pl -g`. By default this generates a new resource file with empty

<translation></translation> tags. To generate copies of the source strings within the translation elements, add the -d option.

For example, to create a new Gaelic locale, issue the following command:

```
perl rt2res.pl -g ga -e ISO-8859-14 c:\<docent_home>content\lms
```

Where C:\<docent_home>\content\lms is the root of the LMS content directory and ga is the locale name for Gaelic. Gaelic uses the ISO-8859-14, or “Extended Latin 8” encoding.

3

Configuring Locales

This chapter provides information on a variety of configuration options that may be helpful when customizing a locale. It contains the following sections:

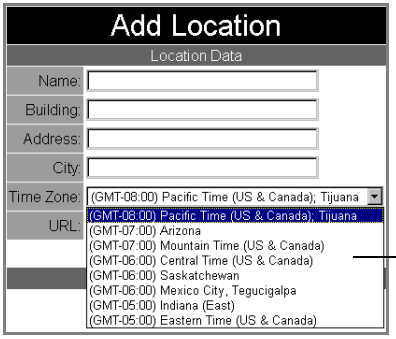
International Time and Date Formats

The Learning Management Server supports multiple time zones and time/date formats. The system determines which time zone to display based on the available learning activity and user information. You can modify the Learning Management Server to display times in a format appropriate for any location. By default, the Learning Management Server includes a set of commonly-used time zones. You can define additional time zones as desired.

- The Location time zone is used to display any times that are associated with a location, such as the location of an instructor-led course. This setting can be modified in the Locations section of the Administration menu.
- The User time zone is used to display times that are not associated with a location. This setting can be modified in the Users section of the Administration menu.
- The Site time zone is used if there is no other time zone associated with a user or location. This setting can be modified in the "Edit System Setup Parameters" section of the Administrator menu or by individual users.
- GMT, or Universal Time, is used for dates and times stored in the database. They are converted to local time for display.
- If you book resources (for example Instructors) for a variety of local and remote course meetings, the Learning Management Server automatically reconciles and flags direct schedule conflicts across time zones. However, the server does not track or allow for potential travel times.
- When LMS Administrators add locations for classroom courses, they must specify a time zone for the location.
- A time zone must be specified for each user.

Setting Time Zones for Locations

When you add a new location to the Learning Management Server, you specify the time zone associated with the location. This enables the server to display the appropriate times for course sessions and meetings at the location. For more information on setting up locations, see “Managing Learning Activities” in the *Learning Management Guide*.



Changing the Time and Date Format

Date and time functions for each locale in the Learning Management Server are stored in a single App file (`<docent_home>/content/lms/lib/lang/<locale>.jsm`, where `<locale>` is the locale name). See Table 3-2 for details.You can change these settings as desired to match your location’s date and time requirements.

To keep a consistent date and time internally, the Learning Management Server uses JavaScript dates, but the `niceDate` and `niceTime` functions take arguments that enable you to specify the desired output format.

Changing the time and date format in <locale>.jsm changes the format in all App files that call that function.

Table 3-2 Date and Time Functions

Function	Description
makeDateFields	Creates a menu to allow users to choose a date in a form
requestDate	Retrieves the date (as a JavaScript date) that was selected in the previous page
makeTimeFields	Creates a menu to allow users to choose the time in a form
requestTime	Retrieves the time (as a JavaScript date) that was selected in the previous page
makeDateWithTime	Takes two JavaScript date objects, uses the date from one and the time from another and returns the resulting JavaScript date object
makeDateTimeFields	Creates a menu to allow users to choose a date and time in a form.
niceDate	Takes the time of a JavaScript date and returns a string in the format specified. Format options include: • DATE_LONG (example = Apr 30, 1997) • DATE_SHORT (format = MM-DD-YY) • DATE_LOG (format = MM-DD-YY HH:MM:SS)
niceTime	Takes the time of a JavaScript date and returns a string in the format specified. Format options include: • TIME_12 (example = 11:11 PM) • TIME_24 (example = 23:11)
niceDuration	Takes two JavaScript dates and returns the duration between them as a string such a “1 hr, 32 min”.
shortDatePair	Takes two JavaScript dates and returns a string. For example, Apr 22 - 30, Apr 22 - May 13, Apr 22 1:00 pm - 3:00 pm

Defining New Time Zones

Time zones for the Learning Management Server are defined in the following file:

<docent_home>/content/lms/lib/zone.jsm

Predefined time zones are listed at the end of the file. You can add additional times zones as desired. Use the format of existing time zones as a model.

For example, the Pacific time zone entry looks like this:

```
PT : new Zone('<rt key="01928">(GMT-08:00) Pacific Time
(US and Canada); Tijuana</rt>', UTC_ZONES["USPacific"],
true, DST_RULES["usRuleLate"], "<rt key="02995">PST</rt>",
"<rt key="02996">PDT</rt>"),
```

Time zone definitions use the following building blocks:

Table 3-3 Building Blocks for Time Zones

Item	Example	Description
Abbreviation	PT	A unique abbreviation for the time zone
Name	(GMT-8:00) Pacific Time (US & Canada); Tijuana	A long display name for the time zone
UTC Zone	USPacific	The Universal Time offset for the time zone
Daylight Observed	true	A boolean value that indicates whether daylight savings time is observed or not.
Daylight Rule	usRuleLate	The name of the rule used to define when daylight savings time starts and stops. Common DST_RULES are included in the `zone.jsm` file. Use `null` if daylight savings time is not observed.
Standard Time Abbreviation	PST	A short display name to use for times with no daylight rule in effect
Daylight Time Abbreviation	PDT	A short display name to use when daylight rule is in effect

Note: Because the rules for daylight savings time vary across time zones, you might also need to create a new `DST_RULE` for the new time zone. Use the format of existing time zones as a model.

International Currency and Numeric Formats

Currency and numeric formats for the Learning Management Server are stored in a single App file (<docent_home>/content/lms/lib/lang/<locale>.jsm, where <locale> is the locale name). You can change these settings as desired to match your location's requirements.

Only currency formatting is associated with the user's locale. The Learning Management Server supports multiple currencies independent of locale. See the Learning Management Guide for more information on this feature.

To change currency and numeric settings:

1. Use a text editor to open the <locale.jsm> file (<docent_home>/content/lms/lib/lang/<locale>.jsm).
2. Locate the lines that define the currency and numeric formats:

```
new LocaleMonetaryConstants (
'.',                // Decimal Separator
',',                // Grouping Separator
2,                  // Number of digits after decimal separator
3,                  // Number of digits in a group
10,                 // Monetary Input Size
"Front",            // The position of the currency symbol
"USD"               // Default currency
)
```

Change the numeric and currency settings as desired. (See Table 3-4.)

Table 3-4 Numeric and Currency Settings

Setting	Example	Description
Decimal Separator	Period (.)	Symbol used to separate decimal values from whole numbers. For example, 100.4
Grouping Separator	Comma (,)	Symbol used to separate number groupings (thousands, millions, and so on). For example, 1,245,000.
Number of digits after decimal separator	2	The number of digits to display after the decimal separator. For example, 10.03.
Number of digits in a group	3	The number of digits to display after grouping separators. For example, 2,000,000.

Table 3-4 Numeric and Currency Settings

Setting	Example	Description
Monetary Input Size	10	Size of the input field used to specify currency values. This includes decimal and grouping separators but not the currency symbol. For example, $876,543.21
Position of Currency Symbol	Front	The position in which the currency symbol is displayed. For example, $20.00. The other value recognized by the system is “Back”, which displays the currency symbol at the end of the amount.
Default Currency	USD	This property is not currently in use.

3 Save the file.

Name Formats

Some locales, such as `zh_tw` (China), may require specialized name formats. Language packs for these locales contain an extra setting in `<locale>.jsm` to ensure proper name formatting. The `niceName()` function, by default, displays names in the standard format for the locale in use. However, you can change these settings as desired. Because most western locales share the same standard for name formatting, for these locales this function appears in the overall `locale.jsm` file (`<docent_home>/content/lib/locale.jsm`) rather than in the specific `<locale>.jsm`.

The first parameter for the `niceName()` function is an object with at least two properties: `lastname`, `firstname`, and `email` if desired.

The optional second parameter is a format constant, e.g. `niceName.FORMAL`, `niceName.COMMON`, `niceName.EMAIL`, or `niceName.COMMON_EMAIL`. For example, The `niceName.FORMAL` format would display "LastName, FirstName" in western locales. The `niceName.COMMON` format would display "FirstName LastName" in similar locales. In certain Asian locales, the order of name display would be reversed.

Changing Email Language Encoding

Based on the locale of the recipient, email generated by the Learning Management Server is encoded with a particular character set. If desired, you can change the default character set for each locale by editing the `<docent_home>/content/lms/lib/locale_constant.jsm` App file. (For more information on editing App files, see the *Learning Management Guide*.)

A variety of encodings are included in the default version of locale_constant.jsm. You may change the default encoding for a locale by editing the appropriate line in the `MAIL_ENCODINGS` section of `locale_constant.jsm`. This section appears as follows:

```
MAIL_ENCODINGS = {
"root" : "iso-8859-1",
"da" : "iso-8859-1",
"de" : "iso-8859-1",
"el": "iso-8859-7",
"en": "iso-8859-1",
"en_us" : "iso-8859-1",
"en_gb" : "iso-8859-1",
"es" : "iso-8859-1",
"fi" : "iso-8859-1",
"fr" : "iso-8859-1",
"he" : "iso-8859-8",
"it" : "iso-8859-1",
"ja" : "iso-2022-jp",
"ko" : "euc-kr",
"nl" : "iso-8859-1",
"no" : "iso-8859-1",
"pl" : "iso-8859-2",
"pt_br" : "iso-8859-1",
"ru" : "koi8-r",
"sv" : "iso-8859-1",
"zh_cn" : "gb2312",
"zh_tw" : "big5"
};
```

There is a list of all character encodings supported by Docent Enterprise in “Supported Encodings,” on page 2-4.

Localized Image Files

Predefined graphics files for the Fade theme are available for several languages. These files enable you to include language-specific graphics in the Learning Management Server (for example, buttons on the LMS home page.)

These files are located in:

```
<webroot>/docent/lms/lang/<locale>
```

where <webroot> is the document root directory of your Web server and where <locale> is the two letter county code for a language plus the optional two letter country code, as explained earlier. For example, on the Netscape Web server, you could find the French version of these files in:

```
/Netscape/suitespot/docs/docent/lms/lang/fr
```

The Mexican Spanish version of the files would be located in:

```
/Netscape/suitespot/docs/docent/lms/lang/es_mx
```

Image references are marked in the resource file with the attribute "translate=false". This is so that translators know not to translate this reference, but rather to localize it by adding the correct path to the localized image.

Localized WorldPay Screens

If you have installed Language Packs for any of the languages supported by WorldPay, Docent Enterprise passes the locale parameter to WorldPay via a servlet, and WorldPay screens automatically appear in the appropriate language for the selected locale.

WorldPay supports the following languages:

- English
- Japanese
- Dutch
- Spanish

- French
- Swedish
- German

This servlet defines locales in the same way as Docent Enterprise, using two-letter ISO codes.

Localizing Custom Content for WorldPay

If you need WorldPay content displayed in a language other than those listed above, you must localize both the Docent servlet and the WorldPay server. Contact WorldPay for information on how to localize the WorldPay server.

The translatable strings for the Docent servlet are contained in the `docent_messages.properties` file. This file is in UTF-8 encoding. It specifies both the character encoding and the translations, using the following syntax:

```
ENCODING = <character_encoding>
<key> = <translation>
```

Important: Do not translate any of the text before the "=" in these entries. These key names are required for the operation of the servlet and must not be modified.

To localize the Docent servlet, do the following:

1. On the Docent Enterprise CD, navigate to the /connectors/worldpay directory.
2. Open `docentworldpay.jar` and extract the `docent_messages.properties` file.
3. Open `docent_messages.properties` in a text editor that supports UTF-8 encoding.
4. Modify the character encoding, if necessary. For example, in the case of the Italian locale, the character encoding is the same as for the default or English locale:

```
ENCODING = iso-8859-1
```

5 Translate the strings following the "=" into the desired language. Do not translate or modify the key text. For example, to translate the text for the NO_PURCHASE_URL key into Italian, you would edit the line to read as follows:

```
NO_PURCHASE_URL = Non può ottenere il difetto PurchaseURL: dovete
specificarli
```

6 When you have entered the desired translations, save the file with the name docent_messages_<locale>.properties, where <locale> is the new locale code. For example, the Italian version of the file would be named docent_messages_it.properties.

This file must be saved in a directory that is in the classpath for the servlet, such as the config directory, in which the select.properties file is located.

4

Localizing Entities

In Docent Enterprise, the term *entity* refers to any user-definable component such as the name or description of a learning activity, resource, or role. These entities are stored in the database. Many entities can be localized.

This chapter describes the procedures for entering translations for entities.

About Translation Tables

Many tables in the Docent Enterprise schema contain localizable entities. For each of these tables, the schema includes a translation table which contains the localized strings. Some localizable entities, such as roles and permissions, are included in the Docent Language Packs.

Localizable Entities

The following is a list of the entity types that can be localized:

Entity Type	Translatable Attributes
Activities	Name, Description
Activity categories	Name, Description
Activity profiles	Name, Description
Activity reports	Title, Code, Description
Activity domains	Name, Description
Catalog sections	Name, Code, Description
Certificates	Name, Code, Description
CDS sites	Server Name, Region Served
Domains	Name, Description
Email configurations	Default
Job Categories	Code, Title, Description
Sessions	Code, Notes
Learning Activities	Title, Code, Description
Locations	Name, Building Address, City
Permissions	Description
Resources	Name, Description
Resource Types	Name, Description
Resource Attributes	Name
Roles	Name, Description
Score maps	Name, Description, Representation
Competencies	Name, Description
To Do Items	Short Description, Long Description

You can enter information in certain fields in the Learning Management Server in any language. For instance, you may wish to provide translated learning activity titles and descriptions for users in certain locales.

Before you enter a translation for an entity (such as a resource name), that entity must exist in the system already. If the resource has not yet been created, the first locale under which you enter the resource name becomes the default for that entity, and is displayed in that language in all locales for which a translation has not been entered. If you later want to change the default language in which that entity is displayed, you must set your locale to the site default locale and modify it on the appropriate screen.

However, if the resource already exists, you may change your locale to any locale and enter a new name for that resource in the appropriate language. The translated name you enter is automatically saved for that locale.

Note: Since navigating the LMS interface in an unfamiliar language may be confusing, it is best to have someone fluent in each language perform manual translations.

For example, to create a new resource for which the default name is “Desk”, you would do the following:

1. Set your locale to the site default locale.
2. Create a new resource.
3. Enter “Desk” as the name, and “Wood, four legs” as the description.

   This resource now appears in all locales with the name and description you just entered.

   Now add translations for the name and description of this resource. To enter a French translation, do the following:

1. Set your locale to `fr`.
2. Locate the “Desk” resource.
3. Click Edit.
4. Edit the “Name” field to read “Bureau” and the “Description” to read “Bois, quatre jambes.”

This resource now appears in the French locale as “Bureau.” It still appears in all other locales as “Desk”.

Index