

How to internationalize your website

Experience suggests that an internationalized website takes half as long and costs half as much to localize as a non-internationalized site. As companies offer their products to a growing number of international target markets, the benefits of internationalization increase dramatically. This white paper discusses best practices for creating internationalized websites that can be easily adapted for any target audience without having to create multiple versions of the original site.

Executive Summary

The Web has become a multicultural and multilingual entity where websites must be available in the native languages of all the various target audiences. However, to provide each website in a local language requires more effort than "just translating it". In reality, local markets have specific technical and marketing requirements.

The process of internationalization creates a core website that is not biased towards any particular language or market: all market-specific items are determined outside the core. An internationalized website enables companies to create specific web content for targeted markets faster and at less cost. This white paper profiles techniques for creating internationalized websites. These sites can be easily tailored for any target audience.

Globalization, internationalization, translation and localization

The following four terms characterize different activities that work together to make source content accessible to international target audiences. The terms are sometimes abbreviated to the first and last letters with a number that signifies how many letters there are in between. Globalization (G11N) includes all the company-wide preparations that must be made in order to enter the international marketplace. G11N covers anything that must be done differently in any part of the business to optimize international success.

Internationalization (I18N) is the practice of creating source material that is locale independent. In other words, all language-specific and market-specific content resides outside the core application. In the context of this document, I18N refers to code changes that are made to ensure that a product or website can be localized and that all information is presented in a format to which the end user is accustomed.

Translation is the process of adapting meaning from one language into another. This is not a literal, word-for-word process. Rather, the translator must first understand the meaning communicated by the source language and then author words in the target language that convey the same meaning.

Localization (L10N) is the process of adapting a product or service for a particular country or region. This includes translation, but goes beyond it. For example: generic contact information must be replaced with local contacts for that particular country.

The diagram below may help to illustrate the various definitions.



¹ EMEA PIM submission for 20 languages, one of which is English, English = 5%

Three-tiered Web architecture

As a first step to understanding website design, note that a typical Web page consists of three layers:

1. User Interface (UI)—this is the portion of the program with which the user interacts. The UI is what the user sees and makes the page look good. The UI is the look and feel.
2. Code—this is defined as scripting (either client-side or server-side). Code can make the page dynamic and add interactive elements. The code is the engine.
3. Content—this is the dynamically changing part of the website (the words and graphics on the page). Content can be either static (updated rarely) or dynamic (updated frequently). The content is the information. Content should be updated frequently to keep the site interesting.

Website formats

Websites are published using two main methods: static HTML pages and dynamic HTML pages. These methods are outlined below to illustrate the different approaches to Web design.

Static HTML pages

Basic websites consist of static HTML pages that reside on a Web server and are delivered to a user in response to a browser request. The pages are created in one of two ways:

- author the HTML in a non-WYSIWYG (What You See Is What You Get) text editor, such as Notepad, and construct the page through trial and error
- author the HTML in a WYSIWYG editor, such as Microsoft Front Page or Macromedia Dreamweaver, and rely on the tools to generate the appropriate HTML code

Dynamically generated HTML pages

The majority of corporate websites will dynamically generate an HTML page and pass it through to the browser. The most common way of generating this HTML uses server-side scripting languages to generate specific, dynamic content. The most well-known server and application systems are Microsoft's Active Server Pages, Java 2 Platform Enterprise Edition (J2EE), PHP: Hypertext Preprocessor (PHP), the Apache Group's mod_perl and Apache Server, and Macromedia's Cold Fusion. These systems work interactively with complex databases, such as Oracle and SQL Server as well as enterprise portal solutions, such as IBM's WebSphere or Microsoft's SharePoint, making it possible to build sophisticated e-commerce sites, interactive technical support applications and extensive online catalogs, to name just a few possibilities. This paper will focus primarily on this type of dynamic, database-driven site

Internationalization

This section covers many problems and resolutions for the internationalization

When to internationalize

The best time to internationalize is during core site development before localization has started. The main benefits of this approach are:

- errors are found early in the process
- errors are fixed by people who are familiar with the code
- developers learn how to internationalize a product and thus learn how to avoid internationalization errors in future projects
- the internationalization effort costs less and takes less time
- the localization effort will cost less and take less time

Internationalization issues

Even when a website is destined for release in several languages or markets, it is still easy to overlook potential problems. One language does not necessarily correspond to one market; German, for example, is spoken in Germany, Switzerland and Austria, and there are German speakers in many other countries as well. Language and region (or country) each play a role in determining what content is appropriate for a given user.

There are many different internationalization issues—the number that needs to be considered depends on the format and complexity of the website. This document covers some of the common internationalization issues.

Text embedded in graphics

Where possible, avoid placing text in graphics. Most localizable graphics consist of text on top of some sort of structured background. To localize graphic text, the localizer must access the textual part of the graphic. Localizable graphics should be handed off in a package that supports "layering" so that the text portion of the graphic is on a separate layer and is easily accessible for localization.

If text-embedded files are necessary, the following guidelines apply:

- provide a well-documented, layered source file with details of the fonts and colors used
- allow room for the text to expand

An alternative to text-embedded graphics is to use the graphic as a background and position the text on top of it. For example, using the tag and forcing the position of the text to be "relative" to the graphic.

Symbols and other design elements

As part of the website design, avoid culture-dependent symbols that might not be clear to an international audience. A classic example would be an American mailbox with a little flag to indicate that there is new mail. This symbol is used on many sites to indicate e-mail but people outside of North America will not necessarily recognize the mailbox. For a website, a better symbol would be an envelope, which is universally understood.

There are also many symbols that may have different meanings in different cultures. If there are any doubts regarding the hidden meaning of some symbols, it is better to use words instead. As a general rule, the following should be avoided in any graphics used:

- hand gestures or body parts
- graphics with multiple meanings, such as using a pillar to indicate a column
- religious or astrological symbols, such as stars, crosses, etc.
- shapes that are tied to culture, such as stop signs, sports equipment, mailboxes, etc.

Locale-specific content

The following list provides some of the items that must be changed during internationalization. Such formats are often hard-coded by developers, but internationalized code will use the system settings for the user's environment to display the data appropriately.

- calendar system: Gregorian (western), Chinese, Japanese, Hebrew, etc.
- date formats: dd/mm/yyyy versus mm/dd/yyyy
- time formats: a.m./p.m. versus 24-hour clock
- currency formats and other financial data: taxes, shipping charges, etc.
- number formats: decimal separator, thousands separator, etc.
- fonts: names, sizes, etc.

For easier localization of fonts, build pages using Cascading Style Sheets (CSS) so that localizers can change the fonts for all pages in one place. This will also reduce the number of tags within the text.

Other issues for consideration, which may not have formatting rules specified by the user's environment include:

- address formats: postal codes, states, number of address lines required, etc.)
- name formats: salutation, order of given name versus surname, titles, degrees, etc.
- telephone number formats: number of digits, country and area codes
- units of measure: Imperial versus Metric
- paper sizes: letter/legal versus A3/A4, etc.
- meaning of colors in different cultures

Concatenated strings

A concatenated string is an error message or other text that is dynamically assembled out of fragments and presented to the user in sentence form. Consider the following ASP code snippet:

```
<%  
if nSource = 0  
sSource = "server"  
else  
sSource = "connection"  
end if  
>  
...  
<P>The <%=sSource%> is currently unavailable</P>
```

In languages where the "server" and "connection" nouns have different genders, it would not be possible to translate the above string because the translation of "The" would depend on the gender of the noun.

The correct solution is to have either multiple error messages that are translated separately, or to find a different way of presenting the same information. For example, to report the number of errors encountered:

```
<P>Process complete, <%=NumErrors%> error(s) encountered</P>
```

Even in English, this is not a good approach. Re-writing the code can make it flexible enough for multiple languages.

```
<P>Process complete. Errors : <%=NumErrors%></P>
```

Sort order

Sort order is not the same for all languages, particularly for languages that do not use the Western alphabet. Even in Swedish, for example, some extended characters (e.g. å) are sorted after the letter z. In many Asian cultures, characters are composed by a prescribed tradition of brushstrokes; characters are sorted by the brush stroke order. Another issue is that after localization, the first letter of the word might change, which will change its position in the sort order list. Thus, it is difficult to sort automatically.

To build an internationalized Web site, it is necessary to either find a way to automatically sort the items, a very difficult task, or ensure that the localizers can change the sort order of the list while they are localizing the code. The best method is to allow the localizers to sort the list.

Text truncation

Truncation occurs when an interface is designed with a particular word or words of the source language in mind. Text usually expands when it is translated. Any words in the target language that are longer are then truncated in the interface. This is a common occurrence on websites where space is at a premium and designers have tried to maximize the available real estate. Designers and developers of the website must anticipate that any or all the words and phrases may expand. The general rule is that, on average, large sections of text will expand by about 30%. Single words and terms, on the other hand, may expand by up to 400%. Having a flexible design is critical to the success of localizing a website.

Truncation can occur in a variety of places, but most noticeably within buttons, graphics and tables.

Character encoding

When localizing a site, one of the first decisions to make is which character encoding to use. The character set encoding tells the browser which characters to display. There are many different character encoding schemes for different writing systems and computer platforms. On the Web, the various schemes are generally divided into two main groups: Unicode (such as UTF-8, an eight-bit representation of Unicode that covers the characters of most of the world's major languages), or native encoding (encoding specific to a language or set of languages). There are costs and benefits to each system available. Regardless of which encoding you choose, perform testing to ensure that the data displays correctly to the end user and that any data sent to and from a form or database maintains data integrity.

Official Name	Common Name	Languages Covered
ISO 8859-1	Latin-1	Western European
ISO 8859-2	Latin-2	Eastern European
ISO 8859-3	Latin-3	Southeastern European
ISO 8859-4	Latin-4	English and Cyrillic
ISO 8859-5	Cyrillic	English and Arabic
ISO 8859-6	Arabic	English and Greek
ISO 8859-7	Greek	English and Hebrew
ISO 8859-8	Hebrew	Western European and Turkish
ISO 8859-9	Latin-5	Scandinavian
ISO 8859-10	Latin-6	Japanese
JIS X0208		Japanese
KS C5601		Korean
GB 2312	Chinese	Simplified Chinese (PRC)
BIG5		Traditional Chinese (Taiwan)
Unicode		Universal

Table 1 Some common character sets

Dynamic data

Website components could include databases, form elements, COM objects, JavaScript™, DHTML, ActiveX® controls, Java etc. Each component of the website should be considered carefully and the following questions addressed:

- what am I putting in? (for example, JavaScript)
- what am I getting out? (for example, from a database)
- is what I put in what I get out? (for example, form elements)

The answers to these questions may not be the same for different elements on the site. If all strings for several language versions are in a single database, it may be necessary to use Unicode or UTF-8. If the HTML has been encoded natively, how will the data be transformed so that the user can read it on the website?

Identifying visitors: global gateways

If the website is to receive visitors from many geographical areas around the world and it is required to provide a personalized service to everyone, it may be necessary to employ some server-side scripting techniques to achieve this. ASP, JSP and XSP can all provide this level of functionality. A good way to do this is to present first-time users of the site with a global gateway page. After they have selected their country and language preference, place a cookie containing that information on their computers and direct them to the localized pages. The next time they access the site, read the cookie and direct them straight to the localized material without requiring them to go through the gateway again.

Identifying and tracking internationalization problems

As websites become more complex and interact with other applications, such as Content Management Systems and databases, they can benefit from the techniques developed for software internationalization. It is very helpful to create a bug database and a bug tracking/resolution system to register, track and resolve internationalization bugs.

Specifications

Sometimes just reading a functionality or technical specification from the point of view of an international user can reveal a number of issues. For example, take a technical specification which states that a website will display the date and time it was last updated. This data is collected from the server date and time. However, some countries are on the other side of the International Dateline, so any date displayed may make the site appear at least a day older than it really is. A better solution is to calculate the date based on the user's time, so that the content always appears fresh.

Test Cases

One of the best ways to analyze whether a website is internationalized is to write test cases. Obviously, this is best when there are dedicated international testers. Writing test cases allows testers to focus on smaller and smaller units of the project. When test cases are written for each component, testing can be modularized. The test cases can be used again at later stages of the project to test the files once they have been localized, for example.

Localization of Content

This section covers some general problems and resolutions for the localization of website content.

For more information on localizing content, please refer to SDL's White Paper:

- Creating, Managing and maintaining a global website

Content style

The target audience for a website wants to get in and out quickly to access the information they need... Web users process pages mostly by skimming and scanning. Put the important information first. Modularize the information and 'chunk' it to help users navigate your content quickly and get to the specific information they seek. Keep the site flat: no deeper than two levels, if possible. Pay attention to the mood of the content: avoid humor, as it rarely translates well from language to language or culture to culture. Language that seems merely informal to one person might be considered rude by people in other parts of the world. To be concise, just stick to the facts.

Here is a list of things to avoid when developing Web content:

- words with multiple meanings
- abbreviations
- mnemonics
- acronyms
- telegraphic style
- slang or jargon
- gender
- creation of new words
- shortened plurals or word combinations
- anything that portrays a way of life or culture specific to one country

Content location

Any text displayed to the end-user should be centralized to make the content easier to localize. Content Management Systems (CMS) centralize content in a database and provide powerful tools to index, manage and distribute content. Single-sourcing strategies using XML and XSLT style sheets, for example, allow companies to re-use content in different delivery formats, such as print, Web and online. These systems can save you time and money throughout the entire content lifecycle.

Static content

Static content can be held in the website pages (HTML, ASP etc.) as this will not change at regular intervals. Localization of the static content would typically be done using a translation tool, such as SDLX, that can easily handle Web pages.

Dynamic content

Dynamic content is best held in a database for ease of maintenance. Localization of the dynamic content can take a variety of forms, depending on the tools and applications used to manage it. The most efficient way would be a defined process for identifying updated content and automatically routing it through a pre-defined translation workflow.

For more information on automating the localization of dynamic content, please refer to SDL's White Paper: Building a Better Globalization Management System Business Case at <http://www.sdl.com> under Articles & Links, White Papers/Articles.

Code content

If there is localizable text within the code tier of the site, ensure that it is commented as much as possible. Localizers are generally hired for their translation skills, not for their programming skills. The easier it is for the localizer to identify the localizable text, the smoother the project will be. If the website contains scripting that needs to be localized, hire a localizer with sufficient experience who will not inadvertently alter the script of the website.

An alternative to comments is to enclose any translatable text inside specific XML tags. It then becomes a relatively simple task to write a script that pulls the translatable text out of the code and puts it in a file that can be given to the localizer. Once the localized text is returned from translation, a reverse script can place the translated text back into the file.

Web resources - the REZ file

A more robust solution for localizable text in code would be to externalize the text out of the code into a resource file. Below is a possible approach to this.

Overview

The basic process places translatable text in a text file that is included in the server-side scripting file with a statement like:

```
<!--#include file="text/abc.rez"-->
```

The basic format of these REZ files is:

[type] id = quoted text

- type is optional
- id is a programmer's identifier
- quoted text is either single or double quoted text for translation

Each module of the site would have a text folder where these include files are kept, i.e. the REZ files are kept in a text folder off the main website folders.

Advantages

There are several advantages to using this approach. On the one hand, it encourages the developers to code internationally, which makes the site easier to translate (less cost and faster turn-around). On the other hand, it removes the need for engineers to review code. This is important, as even with a smart script filter you cannot remove the need for an engineering review. Enforcing proper coding practices is the safest, cleanest and most cost-effective way to avoid having to touch the source code. It sometimes takes a long time to track down whether or not a quoted string needs to be translated—you sometimes have to backtrack through several files to see where and how a string is used. By externalizing the script text into resource files, you can avoid having to ask the localization vendor to check script code. You can also use comments in the resource files to add context to the strings, which is a great help to the translators.

File format examples

Java Script (.JS) include file:

```
var NewMail = 'You have new mail in your Inbox';  
var WarnDelete = 'Warning: This action will delete all files';  
var PromptContinue = 'Do you want to continue?';  
var ErrorUpload = 'You may only upload one file at a time.';
```

ASP Script (.ASP) include file #1:

```
ButtonNext = "Next"
```

```
ButtonPrevious = "Previous"
```

```
ButtonResubmit = "Resubmit"
```

```
ButtonConfirmChanges = "Confirm Changes"
```

ASP Script (.ASP) include file #2:

```
<%
```

```
Const NoValuesEntered = "No values entered."
```

```
Const AnlComplete = "Analysis complete. Click Next to continue."
```

```
Const WrongFormat = "Warning: Expecting REZ format, found "
```

```
Const Points = "(N)orth (S)outh (E)ast (W)est"
```

```
%>
```

Concatenation restrictions for Visual Basic

To make it easier to write automated tools for localization of these files, it would be prudent to write VB concatenation in the following style:

```
NL = Chr(13) & Chr(10)
```

```
MsgText = "Thank you for visiting SDL International." & NL
```

```
MsgText = MsgText & "We hope you found the site informative. "
```

```
MsgText = MsgText & "Please come back soon." & NL & NL
```

This is easier to handle than

```
NL = Chr(13) & Chr(10)
```

```
MsgText = "Thank you for visiting SDL International." & NL & _
```

```
"We hope you found the site informative. " & _
```

```
"Please come back soon." & NL & NL
```

Pre-translation testing

Pre-translation testing is the process of exercising the site's user interface, localizability, and site stability prior to localization. This is done by quickly editing all of the strings in the project to:

- include some extended characters (e.g. é, ñ or ö) or Asian characters
- increase the length of the individual terms and paragraphs

During the design phase, it may be desirable to run a prototype site through pre-translation testing to ensure that the design is flexible for all the terms to be translated. For more complex sites, pre-translation testing can be used to test dynamically generated data or to ensure that the controls can display extended characters correctly.

The benefit of pre-translation testing is that the process can be iterated or run at different stages of the project to identify and resolve internationalization issues before they impact the localizer. Thus, pre-translation testing saves money and shortens time-to-market by avoiding the need to fix bugs later in the project. This testing will often have a beneficial effect on the overall quality of the international product as well. Pre-translation testing would determine:

- whether any strings are truncated
- whether all the strings are accessible to the localizers
- whether keyboard shortcuts can be localized
- whether characters display correctly in HTML and on all controls/elements of the website
- whether characters display correctly in and out of a database

That being said, pre-translation testing cannot test everything; human testing and running a pilot project are still necessary to deliver a high-quality website.

Localization kits

Why write localization kits?

To ensure that the website is localized correctly, it is necessary to provide instructions to the localizers, testers and engineers. Localizers need information on what to localize, for which audience to localize and, in most cases, what not to touch in the files.

Who to write the localization kits for

As mentioned above, the localization kit is aimed at localizers, testers and engineers. When the localization is outsourced, the project managers may also need instructions. The style and content of the kit should reflect the target audience. For example, for project managers it is not necessary to go into the details of what needs to be changed. Project managers will be interested in an overview, the number of files and the quantity of words to localize. Engineers would prefer the instructions to be specific to the files and engineering tasks required.

How to write localization kits

The details above should provide some guidance as to how to go about writing a localization kit. The following general steps also apply:

- prepare the project (internationalization)
- research the hurdles (normally this would go in a specification)
- identify the scope (file list, word count etc.)
- identify the target audience (technical, non-technical etc.)
- write instructions for each specific group of people working on the project
- run a pilot project to test the localization kit.

References

Developing International Software by Nadine Kano

Designing a Globalized and Localizable Web Site by Sjoert Ebben and Gwyneth Marshall

The Localization Process: Globalizing Your Code and Localizing Your Site by Sjoert Ebben and Gwyneth Marshall

Going Global: Not for the Halfhearted by Tapani Tuominen

How Wide a World? Localizing Web Sites by Amy Burns

SDL is the leader in Global Information Management (GIM) solutions that empower organizations to accelerate the delivery of high-quality multilingual content to global markets. Its enterprise software and services integrate with existing business systems to manage the delivery of global information from authoring to publication and throughout the distributed translation supply chain.

Global industry leaders rely on SDL to provide enterprise software or hosted services for their GIM processes, including ABN-Amro, Best Western, Bosch, Canon, Chrysler, CNH, Hewlett-Packard, Microsoft, Philips, SAP, Sony, SUN Microsystems and Virgin Atlantic.

SDL has implemented more than 400 enterprise GIM solutions, has deployed over 150,000 software licenses across the GIM ecosystem and provides access to on-demand translation portals for 10 million customers per month. Over 1,000 service professionals deliver consulting, implementation and language services through its global infrastructure of more than 50 offices in 30 countries.

For more information, visit www.sdl.com.