



L10N + I18N

PROYECTO DE PLATAFORMAS ABIERTAS

PROYECTO | **Objetivos** | y delimitaciones

Objetivo General

- Crear una base que permita expandir un proyecto escrito en código C a distintos idiomas y de la manera más flexible posible.

Objetivos Específicos

- Investigar librerías externas escritas en código C que permitan implementar los conceptos de "internacionalización" y "localización".
- Comparar fortalezas y debilidades de las librerías encontradas para aspectos como almacenamiento de traducciones, conversión y formato, así como compatibilidad Unicode para multi-plataformas.

Delimitaciones

- Backend, Frontend, Mobile o Cross-platform son posibles alcances, pero C está más delimitado a Backend.
- Hay que tener en cuenta que, si se considera algún GUI como Frontend, la elección de una librería podría cambiar según los requerimientos.

i18n + L10n

Definiciones

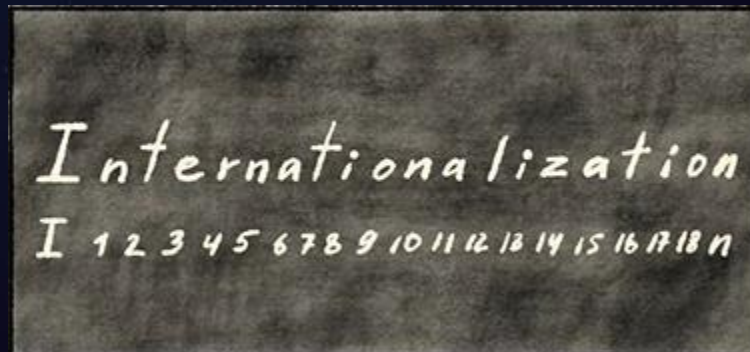
y responsabilidades

¿En educación superior?

- ¿Programa de estudios en el extranjero?
- ¿Captación de estudiantes internacionales?
- ¿Colaboración con socios internacionales?
- No tienen nada que ver en el contexto de la preparación técnica de software para la localización.

Internacionalización (i18n)

- La internacionalización (i18n) es el proceso de diseñar y desarrollar un producto para que sea adaptable a diversos idiomas y contextos culturales.
- i18n, porque la palabra tiene 18 letras entre la "i" y la "n".



Localización (L10n)

- La localización (L10n) es el siguiente paso: la personalización del producto para cumplir con los requisitos legales de un mercado o región, como la traducción de textos, formato de fechas y números, o adaptación de elementos visuales.

i18n + L10n

Beneficios

justificación y responsables

Justificación

- Gestionar traducciones manualmente puede convertirse rápidamente en una pesadilla logística.
- Al internacionalizar desde el principio, se crea una base flexible que permite crecer en cualquier dirección, y ganar ventaja al expandir el producto.

¿Beneficios?

- 1. Ahorro de costes: la inversión en tiempo y dinero es menor si se planifica al inicio en lugar de adaptar el producto después.
- 2. Mejor experiencia de usuario: resulta más familiar, independientemente de la ubicación del usuario.
- 3. Entrada más rápida al mercado: localizar sin grandes cambios permite una expansión más rápida.

¿Quién es responsable?

- Durante la fase de internacionalización, los desarrolladores e ingenieros de software crean un producto flexible desde el principio.
- En la localización, los traductores y diseñadores adaptan el producto a las necesidades específicas de cada mercado o región.

i18n + L10n

Escenarios

y aplicaciones

Backend i18n

- 1. C/C++: ICU4C, gettext, libiconv / genrb, xgettext, setlocale
- 2. Java: ICU4J, Spring Boot / Locale, ResourceBundle
- 3. PHP: Symfony (ICU support), Laravel / setlocale, Gettext
- 4. Python: PyICU, FastAPI, Django / LocaleMiddleware

Frontend i18n

- 1. React: React-intl, i18next, Next.js, NextGlobeGen
- 2. Angular (XLIFF format): Transloco
- 3. Vue and Nuxt: vue-i18n plugin, Nuxt.js
- 4. Svelte, EmberJS, Aurelia
- 5. jQuery.i18n, librería creada por la fundación Wikimedia
- 6. Polyglot.js, Globalize
- 7. HTML: Lokalise, Gatsby (traducciones estáticas)

Mobile i18n

- 1. Ionic framework
- 2. React Native
- 3. Xamarim
- 4. Android: Kotlin
- 5. iOS: Lokalise
- Cross-platform i18n
- 1. Flutter (ARB files)

i18n + L10n | Buenas prácticas | en gestión de traducciones

Archivos/Recursos

- Un archivo por idioma: facilita modificar idiomas sin afectar otros idiomas.
- Nomenclatura ISO: es_ES (español), en_US (inglés), ayuda a identificar el idioma.
- Nombres únicos: home_en.json, menu_en.json, en lugar de en.json repetidos por directorio.

Formato de traducciones

- Identificar formatos de archivo según el estándar de programación simplifica la localización.
- C: .po/.pot/.mo/.res
- Java: .properties
- .NET: .resx
- Ruby on Rails: .yaml
- Javascript: .json
- Angular: .xliff
- Flutter: .arb

Archivos/Recursos

- Formato único: si el código está escrito en JSON, no lo complique; use JSON para las traducciones en lugar de crear otros formatos, eso ayuda a simplificar el proceso.
- Nombres coherentes: misma estructura en todas las rutas de recursos. Ejemplo: idioma / país / variante (opcional).

i18n + L10n | Bosquejo | del inglés "mockup"



i18n + L10n | Configuración | para gettext

Habilitar los idiomas

- Editar /etc/locale.gen

Regenerar los idiomas

- sudo locale-gen

Chequear disponibles

- locale -a

```
[diego@pereira ~]$ sudo vim /etc/locale.gen
[diego@pereira ~]$ sudo locale-gen
[diego@pereira ~]$ locale -a
Generating locales...
  ar_AE.UTF-8... done
  de_DE.UTF-8... done
  en_US.UTF-8... done
  es_ES.UTF-8... done
  fr_FR.UTF-8... done
  hi_IN.UTF-8... done
  ja_JP.UTF-8... done
  ru_RU.UTF-8... done
  sr_RS.UTF-8... done
  zh_CN.UTF-8... done
  zh_TW.UTF-8... done
Generation complete.
[diego@pereira ~]$
```


i18n + L10n | Configuración | para gettext

Extraer_(Strings)

- libglade-xgettext,
xml-i18n-extract,
intltool-extract

Traducir los .po

- xgettext genera los .po

Generar los .mo

- msgfmt genera los .mo

```
[diego@pereira ~]$ ls -las /usr/share/locale/
total 4
0 drwxr-xr-x 1 root root 1704 Sep 19 08:55 .
0 drwxr-xr-x 1 root root 4732 Sep 19 08:55 ..
0 drwxr-xr-x 1 root root  56 Aug 21 12:00 ar
0 drwxr-xr-x 1 root root  22 Sep  6 11:44 de_DE
0 drwxr-xr-x 1 root root  56 Aug 21 12:00 en_US
0 drwxr-xr-x 1 root root  70 Aug 21 12:00 es
0 drwxr-xr-x 1 root root  22 Sep  4 10:33 fr_FR
0 drwxr-xr-x 1 root root  56 Aug 21 12:00 hi
0 drwxr-xr-x 1 root root  90 Aug 21 12:00 ja
0 drwxr-xr-x 1 root root  22 Aug 21 12:46 ru_RU
0 drwxr-xr-x 1 root root  90 Aug 21 12:00 sr
0 drwxr-xr-x 1 root root  70 Aug 21 12:00 zh_CN
0 drwxr-xr-x 1 root root  70 Aug 21 12:00 zh_TW
[diego@pereira ~]$
```

i18n + L10n

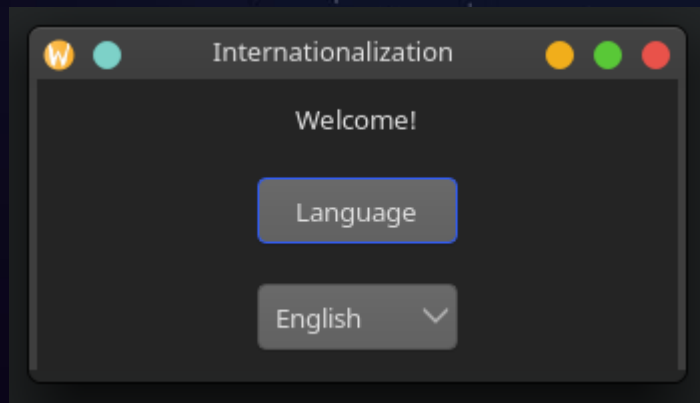
Ejecución

para gettext

Ejemplo no habilitado

- es_ES no habilitado en /etc/locale.gen

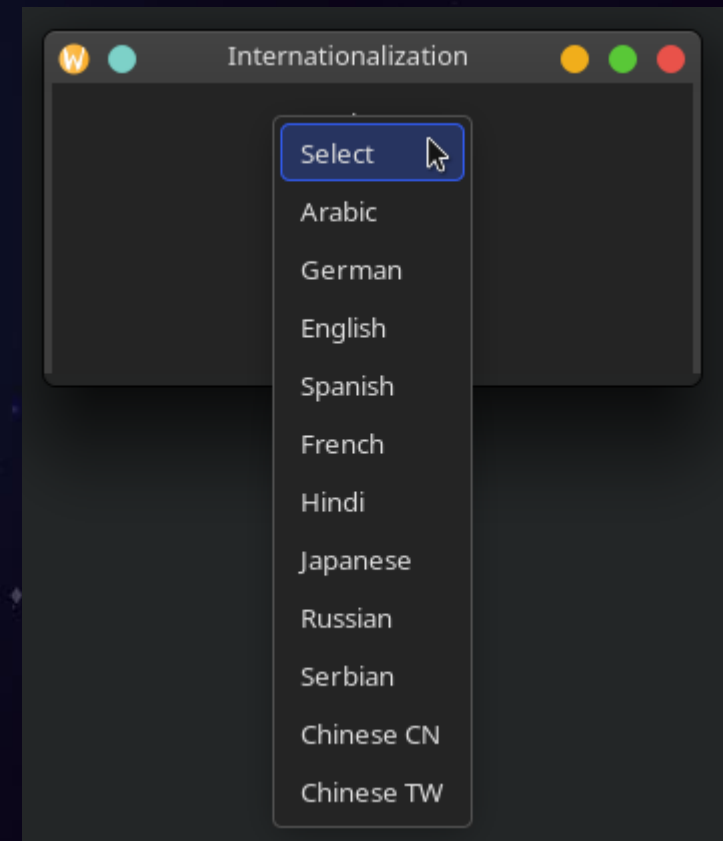
```
LC_ALL locale is currently 'en_US.UTF-8'  
Failed to change LC_ALL locale from 'en_US.UTF-8' to 'es_ES.UTF-8'  
LANG environment for GTK success updated to 'en_US.UTF-8'  
UTF-8 in locale 'es' has length '20' for 'Internationalization'  
UTF-8 in locale 'es' has length '8' for 'Welcome!'  
UTF-8 in locale 'es' has length '8' for 'Language'
```



Inicia con default

- En este caso el default es inglés.

Seleccionar idioma



i18n + L10n

Configuración

para ICU

Instalación

- `sudo pacman -S icu`

Flags de compilación

- `icu-config --ldflags`

Estándar de C

- `icu-config --cflags`

```
[diego@pereira ~]$ sudo pacman -S icu
[sudo] password for diego:
warning: icu-76.1-1 is up to date -- reinstalling
resolving dependencies...
looking for conflicting packages...

Package (1)  Old Version  New Version  Net Change
core/icu     76.1-1       76.1-1       0.00 MiB

Total Installed Size: 41.76 MiB
Net Upgrade Size:     0.00 MiB

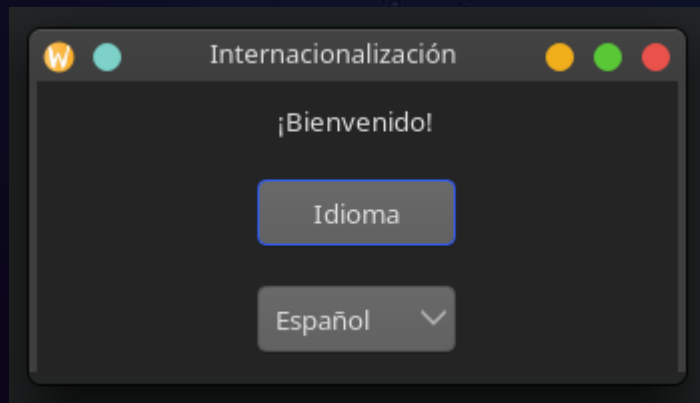
:: Proceed with installation? [Y/n] Y
[diego@pereira ~]$ icu-config --ldflags --ldflags-icuio --ldflags-toolutil
-L/usr/lib -licui18n -licuuc -licudata -licuio -licutu
[diego@pereira ~]$ icu-config --cflags
-std=c11
[diego@pereira ~]$
```

i18n + L10n | Ejecución | para ICU

Ejemplo no habilitado

- es_ES no habilitado en /etc/locale.gen

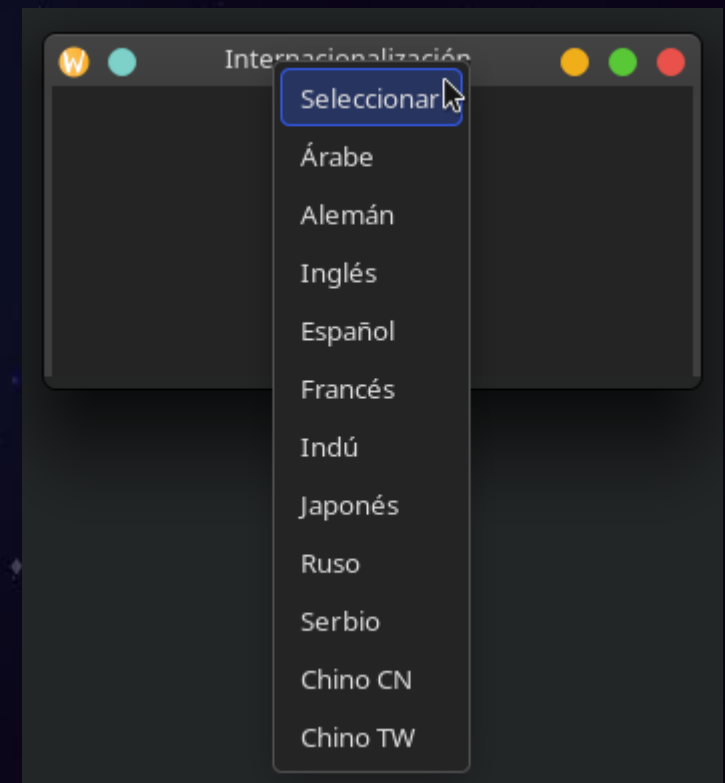
```
LC_ALL locale is currently 'en_US.UTF-8'
Failed to change LC_ALL locale from 'en_US.UTF-8' to 'es_ES.UTF-8'
LANG environment for GTK success updated to 'en_US.UTF-8'
UNICODE in locale 'es' has length '20' for 'Internacionalización'
  UTF-8 in locale 'es' has length '21' for 'Internacionalización'
UNICODE in locale 'es' has length '12' for '¡Bienvenido!'
  UTF-8 in locale 'es' has length '13' for '¡Bienvenido!'
UNICODE in locale 'es' has length '6' for 'Idioma'
  UTF-8 in locale 'es' has length '6' for 'Idioma'
```



Usa el recurso de ICU

- Aún así puede cargar el idioma.

Seleccionar idioma



¡GRACIAS!



DIEGO PEREIRA ES.



+ (506) 7166-7835



orcux86@gmail.com

WWW.DIEGOPEREIRA.ES.TL