Usabilidad y Análisis de Juegos

Guillermo Jiménez Díaz (gjimenez@ucm.es)
Pilar Sancho Thomas (psancho@ucm.es)
Pedro Pablo Gómez Martín (pedrop@ucm.es)

Curso 2023-24

Tema 7: Internacionalización y Localización

La **localización** (o l10n) es el proceso de adaptar el software para una región específica (idioma y cultura) mediante la adaptación de sus contenidos a una configuración regional (o *locale*) y la traducción de los textos. No es solo la traducción de texto sino que es mucho más amplio.

La **internacionalización** (o i18n) es el proceso de diseñar software de manera que pueda admitir diferentes idiomas y regiones sin la necesidad de realizar cambios en el código.

Ambos son necesarios para adaptar un videojuego tanto al idioma de cada país como a los aspectos culturales relacionados con el lugar del mundo donde se va a lanzar el juego. Estos aspectos "culturales" no solo se limitan al idioma sino que van desde unidades de medida (que requieren conversiones, a veces complicadas), formatos de fechas, usos de plurales, géneros, genitivos (en algunos idiomas), referencias religiosas, gestos o símbolos, entre otros. Todo esto se aplica en lo que refiere sencillamente a textos, pero puede extenderse a aspectos de diseño de personajes, señalización, y otros elementos, algunos de los cuales veremos a lo largo de este tema.

Proceso

Es fundamental que, antes de desarrollar el videojuego, se planifique si va a ser o no localizado, ya que afectará profundamente al desarrollo del mismo. Podemos decir que el proceso de localización se divide en las siguientes fases:

1. Internacionalización del código y de la interfaz.
2. Preparación del kit de localización, es decir, de todos los archivos o assets que se enviarán a traducción. Este kit ha de incluir otros documentos y metadatos,

como la longitud máxima de las cadenas de texto traducidas, glosarios de términos, información de contexto, comentarios del formato, etc.
3. Traducción. Esto puede ser más extenso que solo traducir texto, ya que puede incluir doblaje de vídeo (*voice over*) y/o subtítulos.
4. Integración de los recursos localizados en el videojuego. Debería ser un proceso automático pero hay que tener en cuenta que puede involucrar la modificación de texturas o cinemáticas.
5. LQA o *Language Quality Assurance*, es decir, testear el videojuego desde un punto de vista de la localización: texto sin errores ortográficos; sustitución correcta de placeholders; texto correcto desde el punto de vista cultural, en cada punto del videojuego; que el texto no se corte ni se desborde en los espacios en los que ha de aparecer; audio correcto y sincronizado; subtítulos sincronizados con el texto; uso de terminología adecuada, etc.

Texto

Gran parte del proceso de localización consiste en la traducción de texto. Hay que tener en cuenta que los traductores solo traducen (no usan Unity o Photoshop) por lo que hay que mandarles el texto en un formato que sea entendible para ellos y que puedan traducir lo más rápidamente posible (el coste de una traducción puede rondar los 0,10 - 0,15€ por palabra). Esto implica que es necesario extraer todo el texto del videojuego y crear una serie de archivos de texto que se enviarán para ser traducidos. Existen una serie de pautas a tener en cuenta con respecto a estos archivos.

Codificación

La codificación hace referencia a cómo se guardan los caracteres en un archivo de texto (y a cómo se interpretan al leerlos). Es importante porque los caracteres tienen que almacenarse e interpretarse dentro del juego de la misma forma. Si no, se generarán errores en los símbolos. Además, cada codificación puede soportar distintos idiomas y puede ocupar distinto tamaño en memoria (especialmente relevante para las consolas).

Existen diferentes codificaciones y es necesario saber cuál elegir:

- ASCII: Primer estándar para almacenamiento de caracteres, que ocupa solo 1 byte. Guarda los 128 primeros caracteres más comunes en 7 bits. El 8º bit da acceso otros 128 caracteres especiales. Se hicieron páginas de códigos (*codepage*) que permitían modificar estos 128 caracteres especiales, de modo que era posible usar caracteres de distintos idiomas. De ahí surgieron variantes de ASCII, como la

ISO-8859 (o ISO latin 1), que se usa para la mayoría de los idiomas europeos (y que incluye el símbolo del euro).
- UNICODE: Almacena los caracteres en 16 bits, por lo que cada letra/símbolo ocupa 2 bytes en lugar de 1. Java usa directamente UNICODE por lo que las cadenas ocupan el doble que en otros lenguajes (a pesar de que muchos de los códigos no se usen).
- UTF-8: es una forma de codificar el UNICODE comprimiéndolo para ahorrar algo de espacio. Es una codificación de tamaño variable, de modo que los caracteres que están entre los 128 primeros (el ASCII) se envían en un solo byte y cualquier otro por encima del 128 utiliza 2 bytes. Es uno de los estándares más utilizados en web.
- Shift-JIS: Codificación de caracteres para japonés.
- GB 18030: Estándar para codificación de caracteres en chino.

Formato de los archivos

Los archivos con el texto que se van a traducir pueden estar en distintos formatos, los cuales tienen que poder ser editables por lo traductores. Se pueden usar archivos de texto plano, CSV (que se pueden abrir y editar con Excel fácilmente), archivos más estructurados como JSON o XML (que permiten organizar el texto a traducir y añadir etiquetas para hacer traducciones relacionadas con el género o los plurales) o formatos propios y que son importables en software profesional de traducción como el formato XLIFF (*XML Localisation Interchange File Format*)

Organización de los archivos

Dependiendo de la cantidad de texto y del número de idiomas a traducir puede bastar con enviar un único archivo o es necesario dividir el contenido textual del juego en distintos archivos, replicados a la vez para distintos idiomas, con el fin de que distintos traductores puedan trabajar simultáneamente. Esto implica que el kit de traducción será complejo y necesitará instrucciones claras de cómo ser usado.

Internacionalización

Como dijimos, la internacionalización implica diseñar (desde el punto de vista del software) nuestro juego de forma que sea localizable. Las buenas prácticas relacionadas con la internacionalización se pueden dividir en dos categorías:

Organización

Es una práctica bastante común que los assets que van a ser localizados se organicen en distintas carpetas, una para cada lenguaje. De esta forma se facilita la integración de los assets en el videojuego, así como la distribución de estos assets entre los responsables de traducción.

También es recomendable que el nombre de las carpetas de los idiomas sean los definidos por algunos de los ISO de códigos para representar idiomas, como el ISO 639-2

Programación

A continuación vamos a dar algunas pautas de programación y del videojuego en sí a tener en cuenta para la internacionalización.

Cadenas de texto

Es recomendable externalizar todas las cadenas de texto de nuestro videojuego. Sin embargo, mientras programamos (o los diseñadores crean los niveles) es necesario incluir las cadenas de texto finales. Para ello se suelen utilizar *tablas de cadenas*, diccionarios o bases de datos en los que se almacenan las cadenas usando un identificador único. Podemos tener una o varias tablas de cadenas, dependiendo del tamaño de nuestro videojuego.

En este caso, las cadenas de texto dejan de ir "incrustadas" en el código sino que se accede a ellas a través de la tabla de cadenas:

```
// Internacionalización en Unreal Engine
NSLOCTEXT("TablaDeCadenas","Clave","Texto por defecto")
```

En general, la concatenación de texto (o de diálogos) es un problema para la localización ya que el orden del texto puede variar en distintos idiomas. Para crear cadenas compuestas en las que entran en juego variables de nuestro código, en lugar de utilizar la concatenación podemos usar *cadenas interpoladas* (*interpolated strings*), que muchos lenguajes ya incorporan.

```
// En C#
var date = DateTime.Now;
string name = "Joe";
msg = $"Hello, {name}! Today is {date.DayOfWeek}, it's {date:HH:mm} now."

// En JS
```

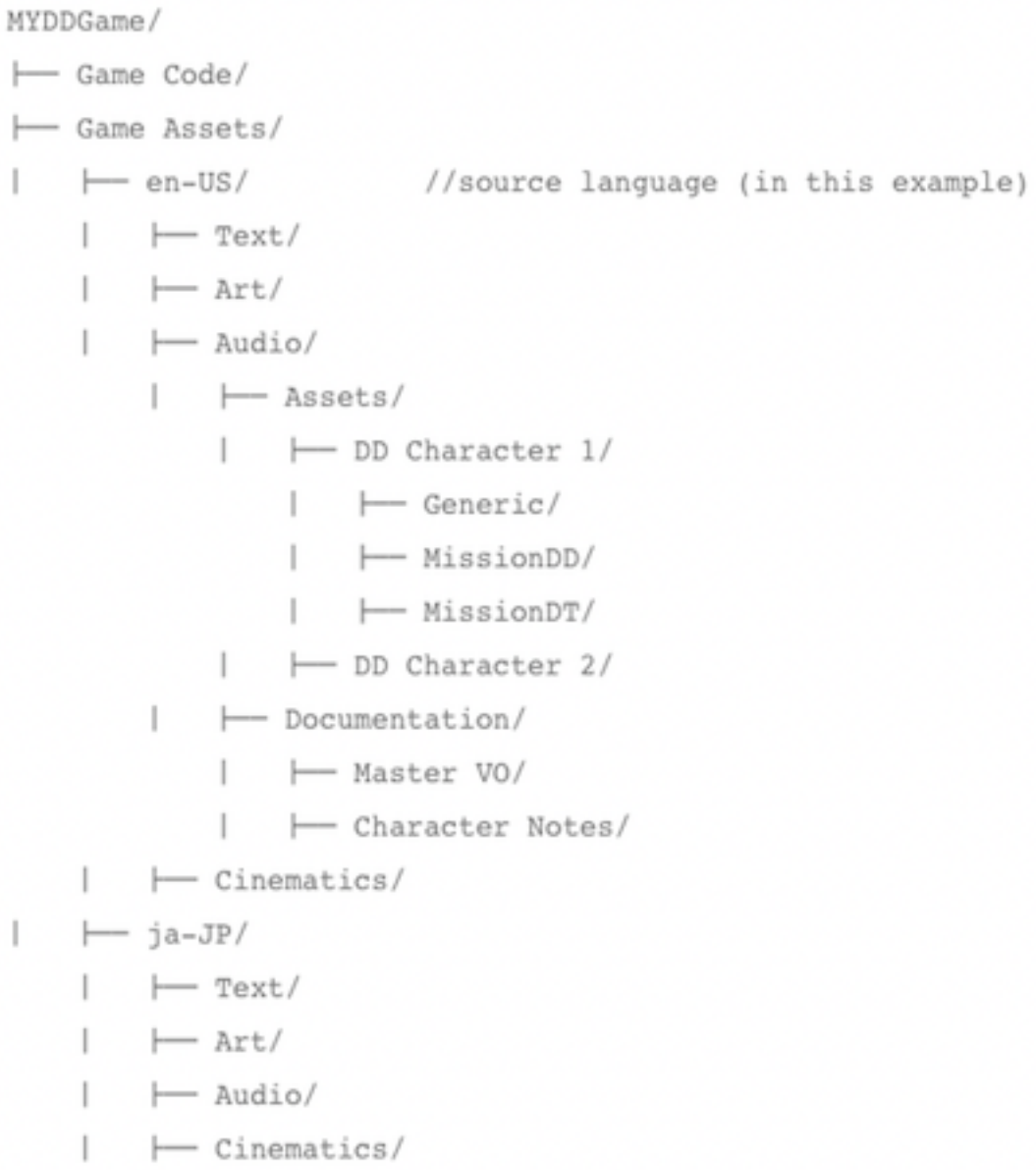

Figura 7.1: Organización de los assets

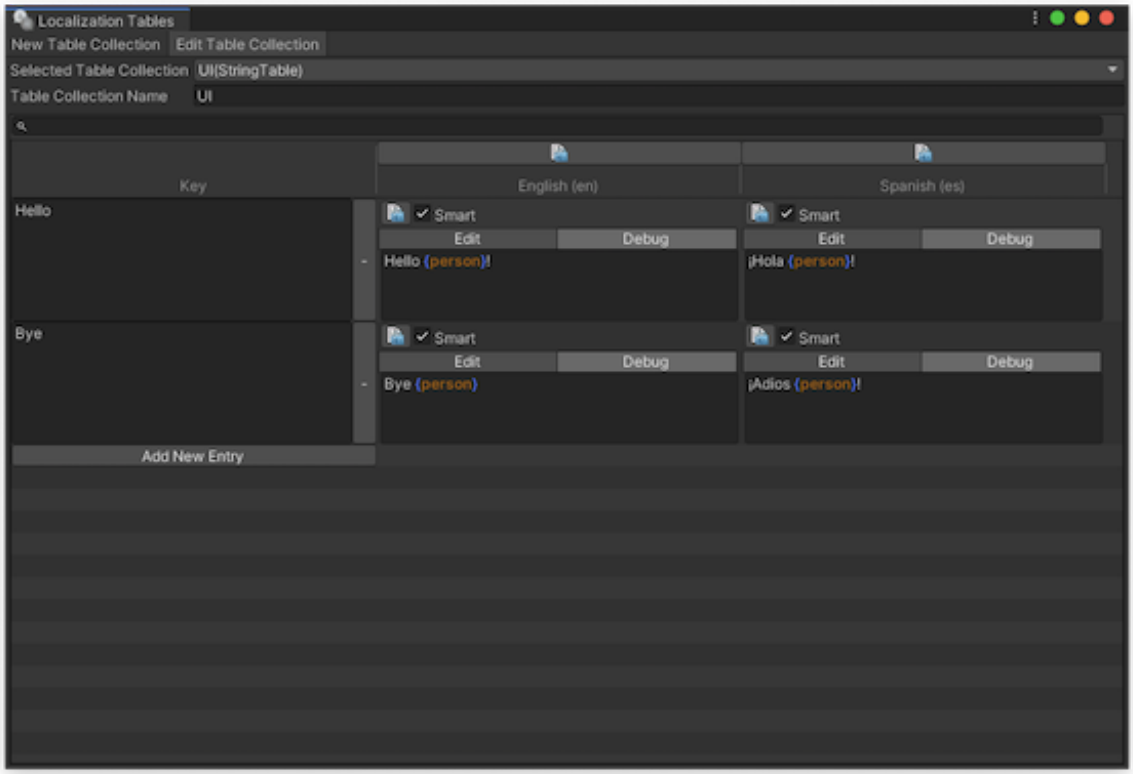

Figura 7.2: Tablas de cadenas en el paquete de Localización de Unity

```
let a = 5;
let b = 10;
console.log(`${a} + ${b} suma ${a + b}`);
```

Estas cadenas permiten indicar el orden de los parámetros dentro de la cadena o añadir algunas expresiones (como las condicionales). Así mismo, se pueden ampliar de modo que añadan etiquetas y códigos de control que permitan crear cadenas más complejas en las que se tenga en cuenta el número del contenido que se está escribiendo.

```
// Plurales con C#
// Si NumCats=1 la cadena es "There is one cat"
// Si NumCats>1 la cadena es "There are N cats"
$"There {(NumCats == 1 ? "is" : "are")} {NumCats} cat{(NumCats == 1 ? "" : "s")}");
```

```
// Cadenas de formato en Unreal
// Si NumCats=1 la cadena es "There is one cat"
// Si NumCats>1 la cadena es "There are N cats"
"There {NumCats}|plural(one=is,other=are) {NumCats}
{NumCats}|plural(one=cat,other=cats)"
```

Se puede usar una técnica similar para el género. En este caso, será necesario que tener información adicional sobre el género de lo que se desea representar.

```
\\ Cadenas para género en Unreal
\\ Ejemplo en francés, dependiendo del género
"{Gender}|gender(Le,La) {Gender}|gender(guerrier,guerrière)
est {Gender}|gender(fort,forte)"
```

Hay que tener en cuenta que los formatos de fecha, moneda o numéricos también varían de una región a otra.

De nuevo, muchos lenguajes disponen de librerías (o están integrados en el lenguaje, como las clases del espacio de nombres `System.Globalization` de C#) que permiten hacer conversiones entre estos formatos de manera sencilla. Sin embargo, existen otras conversiones que hay que tener en cuenta:

- Unidades de peso, velocidad o temperatura pueden implicar conversiones de valores.
- Los precios pueden implicar conversiones de moneda ($2 no es lo mismo que 2 yenes, sino que debería convertirse a unos 220 yenes)

Por último, hay que tener en cuenta cómo se corta una palabra en textos multilínea en función del idioma. Para ello se pueden incorporar caracteres especiales ocultos que indiquen por dónde se corta una palabra (*super^cali^fragilistic^expi^ali^docious*) A esto

	US English	*French*	*German*	*Spanish*	*Italian*	*Japanese*
Date	mm/dd/yyyy	dd-mm-yyyy	yyyy-mm-dd	dd/mm/yyyy	dd/mm/yyyy	yyyy年mm月dd日
Time	hh:mm:ss am/pm (12-hour clock)	hh:mm:ss (24-hour clock)	hh:mm:ss (24-hour clock)	hh:mm:ss (24-hour clock)	hh:mm:ss (24-hour clock)	hh:mm:ss (24-hour clock)
Decimal Separator	period (.)	comma (,)	comma (,)	comma (,)	comma (,)	period (.)
Thousand Separator	comma (,)	space ( )	space ( ) or period (.)	space ( )	space ( ) or period (.)	comma (,)
Number Example	15,631.87	15 631,87	15 631,87 or 15.631,87	15 631,87	15 631,87 or 15.631,87	15,631.87
Currency	$12,345.67	12 345,67 €	12 345,67 €	12 345,67 €	€ 12.345,67	¥ 12,345
Ordinals	1st 2nd 3rd ...	1er 2e 3e ... or 1re 2e 3e ...	1. 2. 3. ...	1º 2º 3º ... or 1ª 2ª 3ª ...	1º 2º 3º ... or 1ª 2ª 3ª ...	1目 2目 3目 ...

Figura 7.3: Algunos formatos estándar para distintas regiones

hay que añadir que algunas first parties (Nintendo, Sony...) ponen sus propias reglas a este respecto (como no poder partir palabras).

Fuentes y texturas

El texto traducido puede tener que ser incorporado a una textura. En este caso, hay que preverlo durante la creación de la textura y separar las capas de texto adecuadamente. Si se hace correctamente, esta incorporación de texto se puede hacer de manera automática durante la build.

En cuanto al uso de fuentes, lo más importante es que la fuente que utilicemos tenga los caracteres de los idiomas en los que vamos a localizar el videojuego (faltas de ortografía como eliminación de acentos o ñ pueden ser aceptables si el texto es legible sin ellos). Una vez seleccionada, será necesario decidir cómo incorporarla al juego:

- Como texturas de fuentes: Los caracteres que vamos a usar están dibujados en una textura. Podemos convertir fuentes en texturas. Son más eficientes de dibujar y de usar si solo hay un tipo de letra y con un único formato. En caso de que hubiese varios, sería necesario un archivo por cada tamaño/formato.
- Como archivos de fuentes TrueType (ANSI), OpenType (UNICODE): Tienen información de cómo se dibuja la fuente (vectorial). Son menos eficientes de dibujar pero ahorran memoria en caso de tener que presentar en el videojuego textos con distintos tamaños y formatos.

Por último, las fuentes usadas pueden ser de ancho fijo o variables. Hay idiomas en

los que es necesario que sean de ancho fijo (como en japonés o coreano) pero en otros idiomas puede facilitar la lectura las de ancho variable. Éstas presentan un problema a tener en cuenta de cara a la información que se pasa al traductor, ya que en general, es necesario saber cuál es la longitud máxima del texto que se va a traducir. Si usamos una fuente de ancho fijo, esta longitud se puede medir en caracteres. Pero con una fuente de ancho variable, es necesario proporcionar herramientas adicionales con las que pueda ver cuál es la longitud real del texto traducido.

Interfaz

También a la hora de diseñar la interfaz de juego es necesario tener en cuenta la localización. En general, un texto en inglés o en japonés es más corto que un texto en español o en alemán (que puede tener, además, palabras mucho más largas).

Generalmente, es recomendable dejar espacio adicional (30-50 %) por problemas de tamaño de palabras en otros idiomas (ver problema de la figura)

Figura 7.4: Problema de localización en la interfaz de Le havre

Normalmente, el uso de fuentes de tamaño variable ayuda con este problema. Así mismo, es recomendable que las cajas de texto tengan un tamaño que sea adaptable al contenido

o añadir barras de scroll a las cajas que de texto que lo permitan. Como vimos antes, es un error grave que un texto se corte o se trunque.

También hay que dejar altura suficiente para letras más complejas (sobre todo para localizaciones en lenguas asiáticas), especialmente si los videojuegos son para consolas (potencialmente poca resolución o con el usuario alejado).

A veces también se puede reducir el tamaño de la fuente pero en este caso hay que tener cuidado para que el texto sea legible y, sobre todo, que cumpla los estándares de accesibilidad.

Por último, hay que tener en cuenta los teclados en pantalla, sobre todo en dispositivos móviles, el tamaño del teclado y la ubicación de las teclas dentro del mismo puede ser diferente en cada idioma (Ej. Prueba a cambiar de idioma en el teclado de Whatsapp). Esto implica que hay que tener cuidado a la hora de mapear las teclas de dicho teclado con el código de la tecla pulsada.

Otros aspectos a tener en cuenta

- Si hay vídeos con voz en off, es recomendable alargar el vídeo para los idiomas "más largos"
- Suele ser más caro localizar video prerenderizado que el generado con el propio motor (*lip sync*). Lo más "barato" es incorporar un sistema de subtítulos, lo que implica tener que añadir códigos de inicio y de fin, así como segmentar las líneas manualmente, usar códigos de colores para diálogos con varios personajes, etc.
- Hay que tener cuidado con tonos del habla, acentos, errores... así como con personajes famosos localmente.
- En los sistemas de Voice Over, suele ser necesario hacer un postprocesado de los archivos de audio localizados (eliminación de ruido, ecualización, añadir silencios...)

Referencias

- "The Absolute Minimum Every Software Developer Absolutely, Positively Must Know About Unicode and Character Sets (No Excuses!)", Joel Spolsky
- W3C Internationalization
- Localización de videojuegos, Pablo Muñoz Sánchez, Ed. Síntesis (2019)
- Best Practices for Game Localization. IGDA Game Localization SIG (2012)
- Localization of 'Cyberpunk 2077': Technology, Tools, and Approach. Tommi

Nykopp, Mikolaj Szwed (CD Projekt Red) GDC 2023. Una charla similar está disponible en https://www.youtube.com/watch?v=Nmh1KyCvWok

- The Key Lessons of 'Cyberpunk 2077 Phantom Liberty' Localization. GDC 2024.
- Design with Localization in Mind Approaches to Translation Damage Control Anton Mukhataev (Nine Rock Games). GDC 2024.
- Localización en Unreal Engine
- Paquete de Localización en Unity
- Plataformas de traducción y localización: Transifex, Crowdin