



MENÚS, BARRAS DE HERRAMIENTAS Y DIÁLOGOS EN JAVAFX: ANEXO INTERNACIONALIZACIÓN

Interfaces Persona Computador

Depto. Sistemas Informáticos y Computación

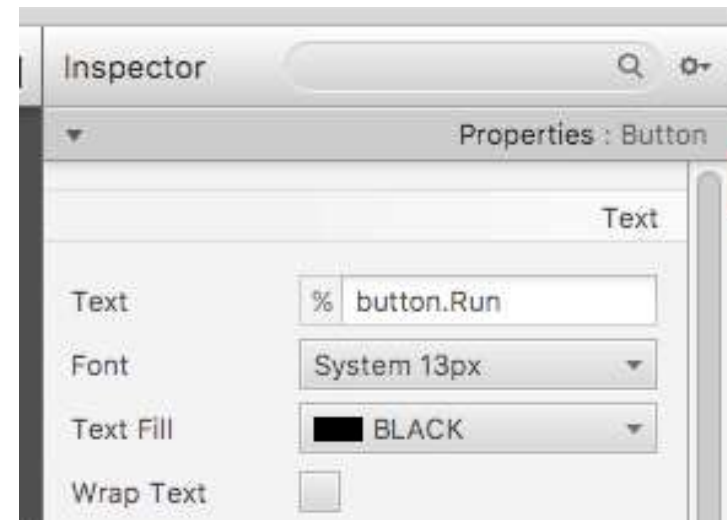
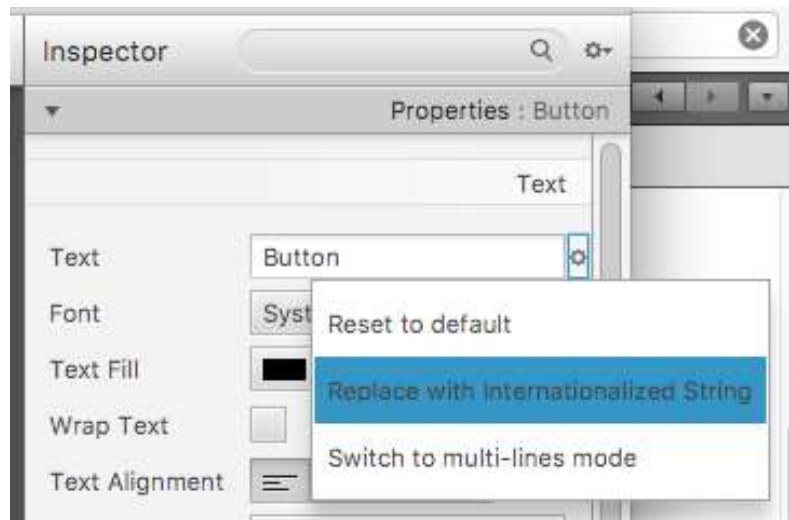
UPV

Índice

- Internacionalización
- Bibliografía

Internacionalización

- SceneBuilder facilita la tarea de traducir una aplicación permitiendo definir, en vez de la etiqueta de un control, una variable cuyo valor dependerá del idioma



Internacionalización

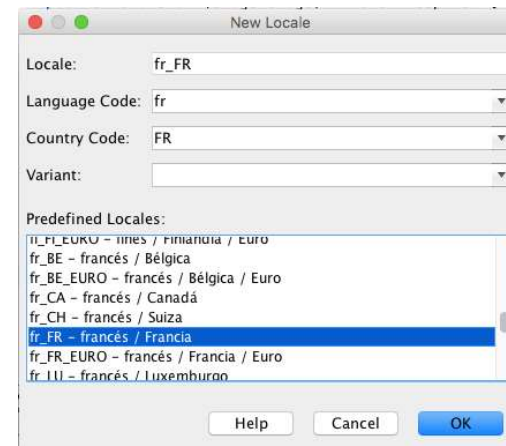
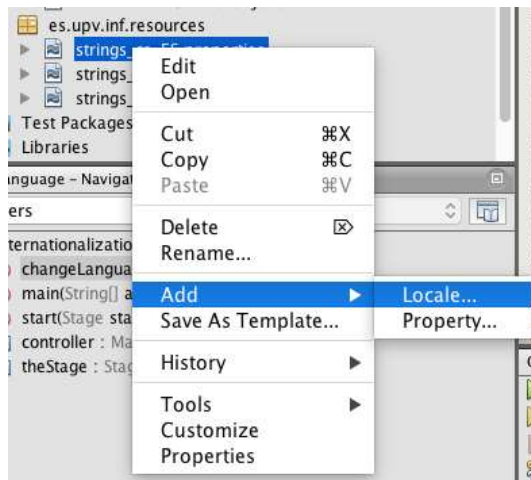
- En el fichero FXML, las etiquetas a traducir se distinguen por empezar por un signo %

```
<Button mnemonicParsing="false" text="%button.Run" />
```

- ¡Cuidado!: en SceneBuilder, no se puede escribir directamente el nombre de una variable con un % delante: hay que seleccionar la opción “Replace with Internationalized String” e introducir el nombre de la variable, sin %

Internacionalización

- Cada traducción de las variables a un idioma dado se almacena en un fichero de propiedades (también llamado *resource bundle* en Java)
- En NetBeans:
 - File\New File...\Other\Properties File
 - Su nombre debe tener la forma: <base>_<idioma>_<pais>.properties
 - Haciendo clic derecho sobre el fichero se pueden añadir más:



Internacionalización

- La clase Locale en Java representa una región geográfica, política o cultural
- Hay operaciones que dependen de un objeto Locale
 - Por ejemplo, representar un número: 3,1416 o 3.1416
- Contiene varios campos con información sobre el idioma (“en”, “ja”...), alfabeto (latino, cirílico...), país o región (“ES”, “GB”...), etc
- En tiempo de ejecución, Java selecciona un Locale teniendo en cuenta la configuración de la máquina

```
NumberFormat defNum = NumberFormat.getInstance();
NumberFormat usNum = NumberFormat.getInstance(US);
NumberFormat jpNum = NumberFormat.getInstance(JAPAN);
System.out.println("DEF Num: " + defNum.format(3.1416));
System.out.println("US Num: " + usNum.format(3.1416));
System.out.println("JAPAN Num: " + jpNum.format(3.1416));

NumberFormat defCurr = NumberFormat.getCurrencyInstance();
NumberFormat usCurr = NumberFormat.getCurrencyInstance(US);
NumberFormat jpCurr = NumberFormat.getCurrencyInstance(JAPAN);
System.out.println("DEF Currency: " + defCurr.format(3.1416));
System.out.println("US Currency: " + usCurr.format(3.1416));
System.out.println("JAPAN Currency: " + jpCurr.format(3.1416));

NumberFormat defPercent = NumberFormat.getPercentInstance();
NumberFormat usPercent = NumberFormat.getPercentInstance(US);
NumberFormat jpPercent = NumberFormat.getPercentInstance(FRENCH);
System.out.println("DEF Percent: " + defPercent.format(0.1416));
System.out.println("US Percent: " + usPercent.format(0.1416));
System.out.println("JAPAN Percent: " + jpPercent.format(0.1416));

Locale defaultLoc = Locale.getDefault();
System.out.println(defaultLoc);
System.out.println(defaultLoc.getDisplayCountry());
System.out.println(defaultLoc.getDisplayLanguage());
System.out.println(defaultLoc.getDisplayName());
```

```
DEF Num: 3,142
US Num: 3.142
JAPAN Num: 3.142

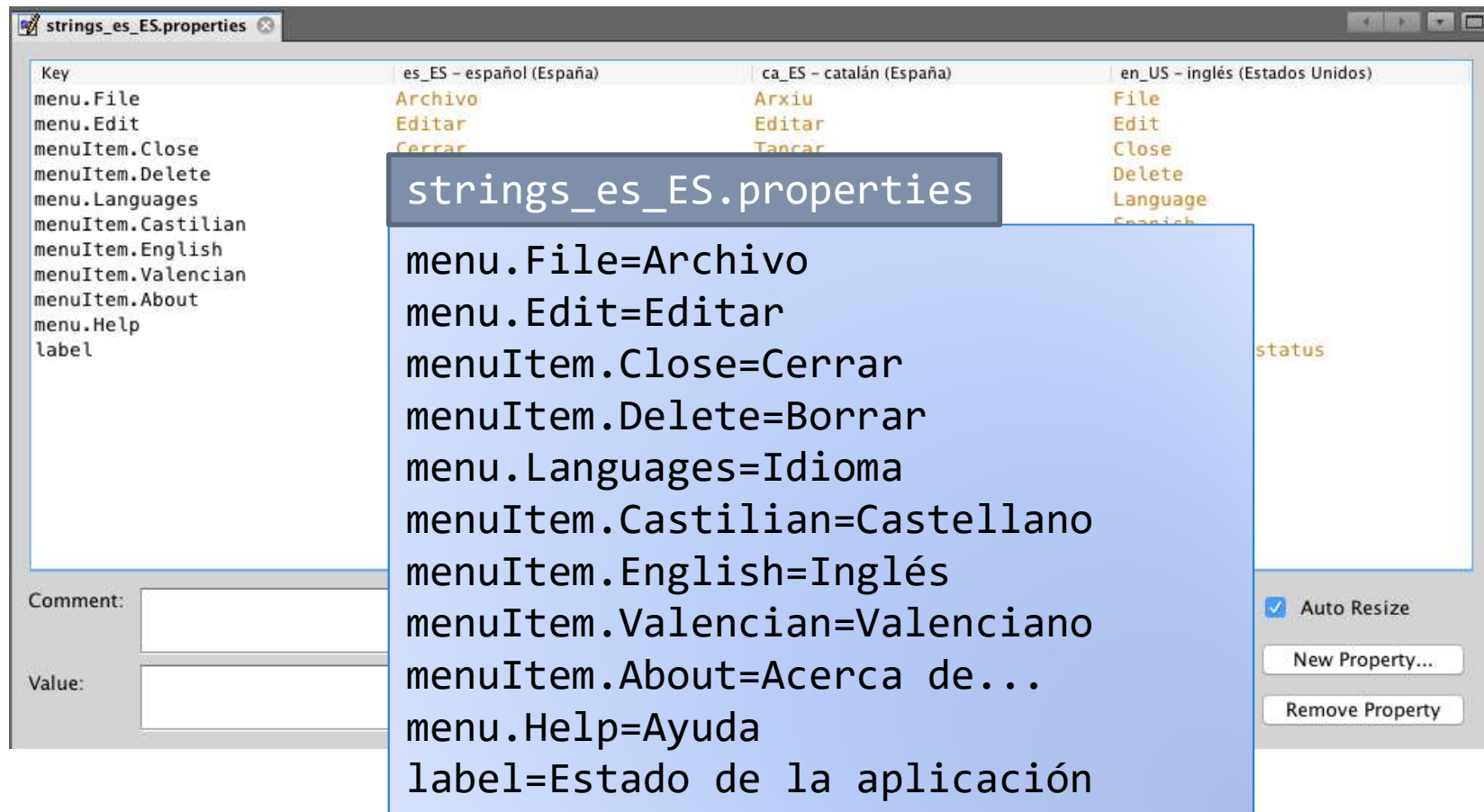
DEF Currency: 3,14 €
US Currency: $3.14
JAPAN Currency: ¥3

DEF Percent: 14%
US Percent: 14%
JAPAN Percent: 14 %

es_ES
España
español
español (España)
```

Internacionalización


- NetBeans tiene un editor que facilita la traducción de las cadenas (botón derecho sobre un *bundle*, Open):



Internacionalización

- En la clase principal, al cargar el FXML:

```
public void start(Stage stage) throws Exception {  
    // Seleccionar el locale por defecto  
    Locale locale = Locale.getDefault();  
    // Cargar el bundle (p.e., strings_es_ES.properties)  
    ResourceBundle bundle =  
        ResourceBundle.getBundle("es.upv.inf.resources.strings", locale);  
    // Pasar el bundle al FXMLLoader  
    Parent root =  
        FXMLLoader.load(getClass().getResource("MainWindow.fxml"), bundle);  
    Scene scene = new Scene(root);  
    stage.setScene(scene);  
    stage.show();  
}  
  
// En el controlador...  
@Override  
public void initialize(URL url, ResourceBundle rb) {
```



Internacionalización

- Con los pasos anteriores, todas las cadenas de la interfaz definidas mediante variables tomarán el valor almacenado en el bundle
- Como en el controlador se recibe el bundle, le podemos pedir cadenas traducidas en tiempo de ejecución (por ejemplo, para ir cambiando el valor de una etiqueta en tiempo de ejecución):

```
private ResourceBundle bundle;
@FXML private Label status;
@Override
public void initialize(URL url, ResourceBundle rb) {
    bundle = rb;
}
private void updateLabel(String key) {
    status.setText(bundle.getString(key));
}
```

Bibliografía

- Internacionalización
 - http://docs.oracle.com/javafx/scenebuilder/1/user_guide/i18n-support.htm
 - <https://docs.oracle.com/javase/8/docs/api/java/util/Locale.html>