


Localización

Fundamentos

- Consiste en desarrollar aplicaciones para múltiples idiomas y adaptadas a determinadas localizaciones geográficas.
 - Los textos se incluyen en archivos de recursos que son cargados dinámicamente en tiempo de ejecución
 - La clase Locale permite establecer la localización geográfica e idioma con el que se va a trabajar
- 

Archivos de recursos

➤ Para cada idioma, se creará un archivo de texto `.properties` con parejas clave=valor que incluyan los diferentes textos a mostrar.

➤ Los archivos se nombrarán:

`nombrearchivo_prefijo.properties`

códigos de prefijo según
norma ISO 639-1

➤ Ejemplos:

`mensajes_es.properties`

`dato1=Introduce tu nombre`
`dato2=Introduce tu edad`

`mensajes_en.properties`

`dato1=type your name`
`dato2=type your age`

Objeto Locale

- La clase `java.util.Locale` representa una localización e idioma
- Se puede crear un objeto `Locale` con los constructores:

```
Locale(String idioma)  
Locale(String idioma, String pais)
```

- También se puede usar la constantes `Locale.US`, `Locale.CANADA`, `Locale.FRENCH`, etc.

- Ejemplos:

```
Locale l1 = new Locale("fr", "FR");  
Locale l2 = new Locale("en", "US");
```

- Localización por defecto:

```
static void setDefault(Locale newLocale)  
static void setDefault(Locale.Category category, Locale newLocale)
```

Enumeración: `DISPLAY`, `FORMAT`

Carga del archivo de recursos

➤ Para obtener el archivo de recursos asociado a una determinada localización debemos crear un objeto **ResourceBundle**:

si no existe archivo para la localización indicada, intenta el de la localización por defecto, sino el predeterminado (sin prefijo)

```
ResourceBundle rb=ResourceBundle.getBundle(String nombreamplio, Locale locale)
```

➤ Ejemplos:

```
ResourceBundle rb1=ResourceBundle.getBundle("mensajes", Locale.US);  
// localización por defecto  
ResourceBundle rb2=ResourceBundle.getBundle("mensajes", Locale.getDefault());
```

➤ Para obtener los textos se emplea el método **getString(String clave)** de **ResourceBundle**:

```
System.out.println("1: "+rb1.getString("dato1"));  
System.out.println("2: "+rb1.getString("dato2"));
```

Revisión conceptos

Suponiendo que tenemos los archivos de recursos `mensajes.properties`, `mensajes_es.properties` y `mensajes_en.properties`, ¿Qué archivo se cargará al ejecutar esta instrucción?:

```
ResourceBundle messages= ResourceBundle.getBundle("mensajes",new Locale("de"));
```

Respuesta

Como no existe un archivo de recursos para ese idioma, se cargará el archivo de recursos de la localización predeterminada. Si no es "es" ni "en", entonces se cargará `mensajes.properties`

Formateado de datos

➤ El paquete `java.text` incluye las siguientes clases para el formateado de números y fechas según una determinada localización:

- **NumberFormat.** Establece un formato para número:

```
double salario=1256.678;  
NumberFormat nf=NumberFormat.getCurrencyInstance(Locale.GERMANY);  
System.out.println(nf.format(salario)); //1.256,68 €
```

- **DateFormat.** Permite formatear fechas

```
DateFormat df=DateFormat.getDateInstance(DateFormat.FULL, Locale.US);  
System.out.println(df.format(new Date())); // Tuesday, May 4, 2021
```

- **DateTimeFormatter.** Formateado de nuevas clases de fecha

```
LocalDate ld=LocalDate.now();  
DateTimeFormatter dtf=DateTimeFormatter.ofPattern("dd/MM/yyyy");  
System.out.println(dtf.format(ld)); //07/05/2021
```

`java.time.format`

Revisión conceptos

Indica cuales de las siguientes instrucciones permiten crear un objeto para formateado de fechas al estilo dia/mes/año:

- a. `DateFormat df=DateFormat.getInstance(DateFormat.FULL, new Locale("es"));`
- b. `SimpleDateFormat sdf=new SimpleDateFormat("dd/MM/yyyy");`
- c. `DateTimeFormatter dtf=DateTimeFormatter.ofPattern("dd/MM/yyyy");`
- d. `DateFormat df=new DateFormat("dd/MM/yyyy");`

Respuesta

- a. Incorrecta. El formato largo muestra el nombre del mes al completo y el día de la semana
- b. Correcta. `SimpleDateFormat` es una subclase de `DateFormat` que ofrece un constructor para proporcionar la cadena de formato
- c. Correcta. Crea correctamente un `DateTimeFormatter` que se emplea para el formateado de nuevas fechas
- d. Incorrecta. `DateFormat` es abstracta, no permite crear objetos con constructor público