Curs practic de Java Curs 10

Cristian Frăsinaru

acf@infoiasi.ro

Facultatea de Informatică

Universitatea "Al. I. Cuza" laşi

Internaționalizarea

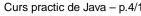
Cuprins

- Introducere
- Clasa Locale
- Formatarea numerelor şi datelor
- Localizarea mesajelor
- Compararea şirurilor

Introducere

Ce este internaționalizarea?

- Internaţionalizarea pregătirea unei aplicaţii astfel încât să poată fi personalizată funcţie de o regiune geografică, politică sau culturală.
- Localizarea adaptarea unei aplicaţii internaţionalizate la o anumită regiune.



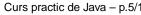
Clasa Locale

Un obiect de tip **Locale**:

- Reprezintă o regiune geografică, politică sau culturală specifică
- Va fi utilizat în operaţii regional-sensibile cum ar fi:
 - formatarea datelor şi numerelor
 - afişarea mesajelor

Crearea

```
Locale(String language)
Locale(String language, String country)
Locale(String language, String country, String variant)
```



Determinarea regiunilor

```
import java.util.Locale;
public class TestLocale {
  public static void main(String args[]) {
   System.out.println("Default locale:");
   localeInfo(Locale.getDefault());
   System.out.println("Available locales:");
   Locale available[] = Locale.getAvailableLocales();
   for(int i=0; i<available.length; i++)</pre>
   localeInfo(available[i]);
  static void localeInfo(Locale locale) {
    System.out.println(locale.getDisplayLanguage() + "\n" +
      locale.getDisplayCountry());
```

Formatarea și parsarea

Formatarea datelor

Formatarea numerelor

```
// Regiunea: Canada :)
locale = Locale.CANADA;
string = NumberFormat.getNumberInstance(locale).format(-1234.56);
// -1,234.56

// Regiunea: Romania :(
locale = new Locale("ro", "RO");
string = NumberFormat.getNumberInstance(locale).format(-1234.56);
// -1.234,56

// Formatare pentru regiunea curenta
string = NumberFormat.getNumberInstance().format(-1234.56);
```

Formatarea monedei

```
locale = Locale.CANADA;
 string = NumberFormat.getCurrencyInstance(locale).format(123.45);
  // $123.45
  locale = new Locale("ro", "RO");
  string = NumberFormat.getCurrencyInstance(locale).format(123.45);
  // 123,45 LEI
  // Parsarea unui sir ce reprezinta o suma de bani
 try {
   Number number = NumberFormat.
                      getCurrencyInstance(locale).parse("$123.45");
if (number instanceof Long) {
// Valoare de tip Long
} else {
// Valoare de tip Double
  } catch (ParseException e) { ... }
```

Localizarea mesajelor

Definirea fişierelor de resurse

MyResources_ro.properties

hello=Salut bye=La revedere

MyResources_fr.properties

hello=Bonjour bye=Au Revoir

MyResources.properties

hello=Hello bye=Goodbye

Mesaje parametrizate:

welcome=User $\{0\}$ logged in at $\{1\}$ from IP $\{2\}$



Crearea unei clase cu resurse

```
package demo;
import java.util.*;
public class MyResources_ro extends ListResourceBundle {
  public Object[][] getContents() {
    return contents;
  }
  static final Object[][] contents = {
    {"hello", "Salut"},
    {"bye", "La revedere"}
    };
}
```

Clasa ResourceBundle

```
String baseName = "MyResources";
try {
  // Selectam fisierul cu resurse implicit
 ResourceBundle rb = ResourceBundle.getBundle(baseName);
 String key = "hello";
 String s = rb.getString(key); // Hello
 key = "bye";
 s = rb.getString(key);  // Goodbye
  // Selectam alt fisier cu resurse
 rb = ResourceBundle.getBundle(baseName, Locale.FRENCH);
 key = "hello";
 s = rb.getString(key); // Bonjour
 key = "bye";
 s = rb.getString(key); // Au Revoir
} catch (MissingResourceException e) {
  // Fisierul cu resurse sau cheia nu exista
```

Compararea şirurilor

```
Locale locale = new Locale("ro", "RO");
Collator collator = Collator.getInstance(locale);
int compare = collator.compare(sir1, sir2);
if (compare < 0) {
    // sir1 < sir2
} else if (compare > 1) {
    // sir1 > sir2
} else {
    // sir1 = sir2
}
```

"ramură" < "rămurică" < "răţuşcă" < "repede" < "rîu"