CORE JAVA

With

SCJP / OCJP

Study Material

Chapter 18: Internationalization (I18N)

DURGA M.Tech

(Sun certified & Realtime Expert)

Ex. IBM Employee

Trained Lakhs of Students
for last 14 years across INDIA

India's No.1 Software Training Institute

DURGASOFT

www.durgasoft.com Ph: 9246212143 ,8096969696

Internationalization (I18N)

Agenda

1. Introduction
2. Locale
   - How to create a Locale object
   - Important methods of Locale class
3. NumberFormat
   - Getting NumberFormat object for the default Locale
   - Getting NumberFormat object for the specific Locale
   - Requirements
     - Write a program to display java number form into Italy specific form
     - Write a program to print a java number in INDIA, UK, US and ITALY currency formats
   - Setting Maximum, Minimum, Fraction and Integer digits
4. DateFormat
   - Getting DateFormat object for default Locale
   - Getting DateFormat object for specific Locale
   - Requirements
     - Write a program to represent current system date in all possible styles of us format
     - Write a program to represent current system date in UK, US and ITALY styles
   - Getting DateFormat object to get both date and time

Introduction

The process of designing a web application such that it supports various countries, various languages without performing any changes in the application is called Internationalization.

If the request is coming from India then the response should be in India specific form , and if the request is from US then the response should be in US specific form.

We can implement Internationalization by using the following classes.

They are:

1. Locale
2. NumberFormat
3. DateFormat

1. Locale:

- A Locale object can be used to represent a geographic (country) location (or) language.
- Locale class present in java.util package.
- It is a final class and direct child class of Object and , implements Cloneable, and Serializable Interfaces.

How to create a Locale object:

We can create a Locale object by using the following constructors of Locale class.

```
Locale l=new Locale(String language);
Locale l=new Locale(String language,String country);
```

Locale class already defines some predefined Locale constants. We can use these constants directly.

```
Example:
Locale. UK
Locale. US
Locale. ITALY
Locale. CHINA
```

Important methods of Locale class:

1. public static Locale getDefault()
2. public static void setDefault(Locale l)
3. public String getLanguage()
4. public String getDisplayLanguage(Locale l)
5. public String getCountry()
6. public String getDisplayCountry(Locale l)
7. public static String[] getISOLanguages()
8. public static String[] getISOCountries()
9. public static Locale[] getAvailableLocales()

```
Example for Locale:
import java.util.*;
class LocaleDemo{
public static void main(String args[]){
Locale l1=Locale.getDefault();
//System.out.println(l1.getCountry()+".....+l1.getLanguage())
;
//System.out.println(l1.getDisplayCountry()+".....+l1.getDisp
layLanguage());
Locale l2=new Locale("pa","IN");
Locale.setDefault(l2);
String[] s3=Locale.getISOLanguages();
for(String s4:s3)
```

```
{
//System.out.print("ISO language is   :");
//System.out.println(s4);
}
String[] s4=Locale.getISOCountries();
for(String s5:s4)
{
System.out.print("ISO Country is:");
System.out.println(s5);
}
Locale[] s=Locale.getAvailableLocales();
for(Locale s1:s)
{
//System.out.print("Available locales is:");
//System.out.println(s1.getDisplayCountry()+"......"+s1.getDis
playLanguage());
}}}
```

2. NumberFormat:

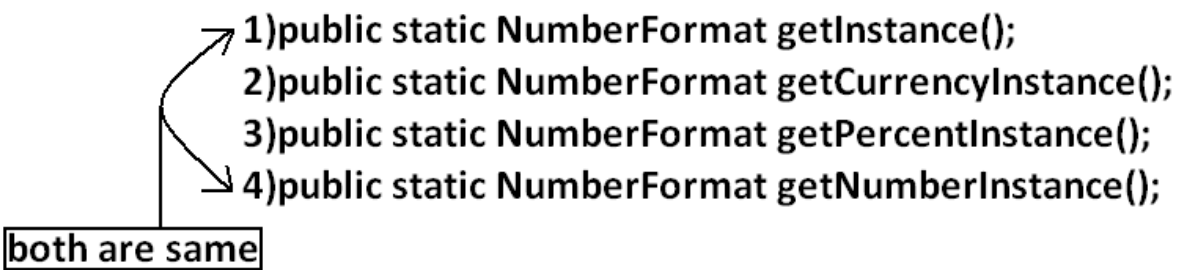
Various countries follow various styles to represent number.

```
Example:

double d=123456.789;

1,23,456.789------------INDIA
123,456.789-------------US
123.456,789-------------ITALY
```

- **By using NumberFormat class we can format a number according to a particular Locale.**
- **NumberFormat class present in java.Text package and it is an abstract class. Hence we can't create an object by using constructor.**
  NumberFormat nf=new NumberFormat(); --------invalid

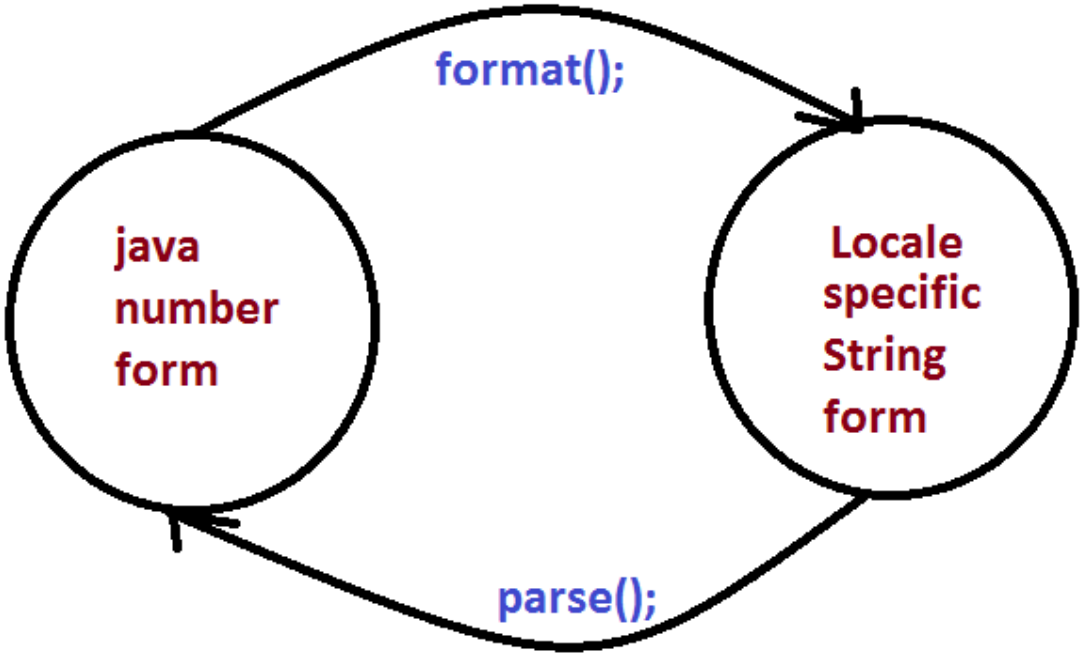
Getting NumberFormat object for the default Locale:

NumberFormat class defines the following methods for this.



Getting NumberFormat object for the specific Locale:

- **The methods are exactly same but we have to pass the corresponding Locale object as argument.**
  Example: public static NumberFormat getNumberInstance(Locale l);
- **Once we got NumberFormat object we can call the following methods to format and parse numbers.**
  public String format(long l);
  public String format(double d);
- **To convert a number from java form to Locale specific form.**
  public Number parse(String source)throws ParseException
- **To convert from Locale specific String form to java specific form.**



Requirement: **Write a program to display java number form into Italy specific form.**

```
Example:
import java.util.*;
import java.text.*;
class NumberFormatDemo
{
```

```
public static void main(String args[]){
double d=123456.789;
NumberFormat nf=NumberFormat.getInstance(Locale.ITALY);
System.out.println("ITALY form is :"+nf.format(d));
}
}
Output:
ITALY form is :123.456,789
```

Requirement: **Write a program to print a java number in INDIA, UK, US and ITALY currency formats.**

```
Program:
import java.util.*;
import java.text.*;
class NumberFormatDemo
{
public static void main(String args[]){
double d=123456.789;
Locale INDIA=new Locale("pa","IN");
NumberFormat nf=NumberFormat.getCurrencyInstance(INDIA);
System.out.println("INDIA notation is :"+nf.format(d));

NumberFormat nf1=NumberFormat.getCurrencyInstance(Locale.UK);
System.out.println("UK notation is :"+nf1.format(d));
NumberFormat nf2=NumberFormat.getCurrencyInstance(Locale.US);
System.out.println("US notation is :"+nf2.format(d));
NumberFormat
nf3=NumberFormat.getCurrencyInstance(Locale.ITALY);
System.out.println("ITALY notation is :"+nf3.format(d));
}}
Output:
INDIA notation is: INR 123,456.79
UK notation is: ú123,456.79
US notation is: $123,456.79
ITALY notation is: Ç 123.456,79
```

Setting Maximum, Minimum, Fraction and Integer digits:

NumberFormat class defines the following methods for this purpose.

1. **public void setMaximumFractionDigits(int n);**
2. **public void setMinimumFractionDigits(int n);**
3. **public void setMaximumIntegerDigits(int n);**
4. **public void setMinimumIntegerDigits(int n);**

```
Example:
import java.text.*;
public class NumberFormatExample
{
public static void main(String[] args){
```

```
NumberFormat nf=NumberFormat.getInstance();
nf.setMaximumFractionDigits(3);
System.out.println(nf.format(123.4));
System.out.println(nf.format(123.4567));
nf.setMinimumFractionDigits(3);
System.out.println(nf.format(123.4));
System.out.println(nf.format(123.4567));
nf.setMaximumIntegerDigits(3);
System.out.println(nf.format(1.234));
System.out.println(nf.format(123456.789));
nf.setMinimumIntegerDigits(3);
System.out.println(nf.format(1.234));
System.out.println(nf.format(123456.789));
}}
Output:
123.4
123.457
123.400
123.457
1.234
456.789
001.234
456.789
```

3. DateFormat:

Various countries follow various styles to represent Date. We can format the date according to a particular locale by using DateFormat class.

DateFormat class present in java.text package and it is an abstract class.

Getting DateFormat object for default Locale:

DateFormat class defines the following methods for this purpose.

1)public static DateFormat getInstance();
2)public static DateFormat getDateInstance();
3)public static DateFormat getDateInstance(int style);

DateFormat.FULL---------->0
DateFormat.LONG---------->1
DateFormat.MEDIMUM---------->2
DateFormat.SHORT---------->3

The dafault style is Medium style

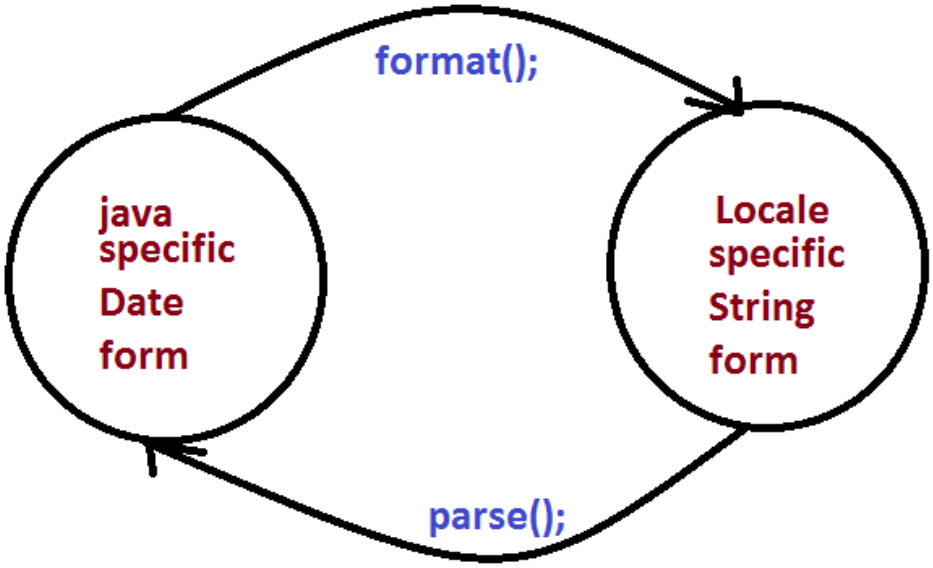
Getting DateFormat object for the specific Locale:

public static DateFormat getDateInstance(int style, Locale l);

Once we got DateFormat object we can format and parse Date by using the following methods.

public String format(Date date);
To convert the date from java form to locale specific string form.

public Date parse(String source)throws ParseException
To convert the date from locale specific form to java form.



Requirement: **Write a program to represent current system date in all possible styles of us format.**

- Questions and answers
- Full Form
- Study Material
- Interview question and answer
- Interview questions and answers
- Iso
- Date and time


```
Program:
import java.text.*;
import java.util.*;
public class DateFormatDemo
{
```

```
public static void main(String args[]){
System.out.println("full form is
:"+DateFormat.getDateInstance(0).format(new Date()));
System.out.println("long form is
:"+DateFormat.getDateInstance(1).format(new Date()));
System.out.println("medium form is
:"+DateFormat.getDateInstance(2).format(new Date()));
System.out.println("short form is
:"+DateFormat.getDateInstance(3).format(new Date()));
}
}
Output:
Full form is: Wednesday, July 20, 2011
Long form is: July 20, 2011
Medium form is: Jul 20, 2011
Short form is: 7/20/11
```

Note: **The default style is medium style.**

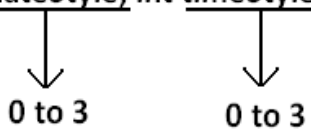
Requirement: **Write a program to represent current system date in UK, US and ITALY styles.**

```
Program:
import java.text.*;
import java.util.*;
public class DateFormatDemo
{
public static void main(String args[]){
DateFormat UK=DateFormat.getDateInstance(0,Locale.UK);
DateFormat US=DateFormat.getDateInstance(0,Locale.US);
DateFormat ITALY=DateFormat.getDateInstance(0,Locale.ITALY);
System.out.println("UK style is :"+UK.format(new Date()));
System.out.println("US style is :"+US.format(new Date()));
System.out.println("ITALY style is :"+ITALY.format(new
Date()));
}
}
Output:
UK style is: Wednesday, 20 July 2011
US style is: Wednesday, July 20, 2011
ITALY style is: mercoled 20 luglio 2011
```

Getting DateFormat object to get both date and time:

DateFormat class defines the following methods for this.

1)public static DateFormat getDateTimeInstance();
2)public static DateFormat getDateTimeInstance(int dateStyle, int timeStyle);
3)public static DateFormat getDateTimeInstance(int dateStyle, int timeStyle,Locale l);




```
Example:
import java.text.*;
import java.util.*;
public class DateFormatDemo
{
public static void main(String args[]){
DateFormat
ITALY=DateFormat.getDateTimeInstance(0,0,Locale.ITALY);
System.out.println("ITALY style is:"+ITALY.format(new
Date()));
}
}
Output:
ITALY style is: mercoled 20 luglio 2011 23.21.30 IST
```