Internationalization (I18N)

I18N :-

→ various Countries follow various Conversions to represent dates & no.'s e.t.c

→ Our application should generate Locale Specific responses like for India People the response should be interms of Rs. (Rupees) & for the U.S people the response should be interms of dollars($) the process of designing such type of web application is called "Internationalization" (I18N).

→ We Can implement I18N by using the following classes

① Locale

② NumberFormat

③ DateFormat

Locale :-

→ A Locale Object represents a Geo-graphic Location

Constructors :-

→ we Can Create a Locale object by using the following Constructor.

(1) Locale l = new Locale(String language);   Javap java.util.Locale .

(2) Locale l = new Locale(String language, String Country);

→ Locale class defines Several Constants to represent Some Standard Locales. we Can use these Locale directly without Creating on our own.

eg:- Locale.US   Locale.ENGLISH
Locale.ITALIAN   Locale.UK

Note:

→ Locale class is the final class present in java.util package

→ It is the direct child class of Object it implements Cloneable & Serializable interfaces.

Important methods of Locale class:

① public static Locale getDefault();

② public static void setDefault(Locale l);

③ public String getLanguage(); en

④ public String getDisplayLanguage(); english

⑤ public String getCountry(); US

⑥ public String getDisplayCountry(); unitedstates

⑦ public static String[] getISOCountries();

⑧ public static String[] getISOLanguages();

⑨ public static Locale[] getAvailableLocales();

Ex:-

```
import java.util.*;
Class LocaleDemo1
{
 Public static void main(String[] args)
 {
   Locale l1 =Locale.getDefault();
   S.o.pln(l1.getCountry()+"----"+l1.getLanguage());
   S.o.pln(l1.getDisplayCountry()+"----"+l1.getDisplayLanguage());
   Locale l2 = new Locale("pa","IN");
   Locale.setDefault(l2);
   String[] S3 =Locale.getISOLanguages();
   for(String S4 : S3)
   {
     S.o.pln(S4);
   }
   String[] S4 = Locale.getISO~~Langu~~Countries();
    for(String S5 :S4)
    {
     S.o.pln(S5);
    }
    Locale[] s = Locale.getAvailableLocales();
    for(Locale S1:s)
    {
     S.o.pln(S1.getDisplayContry()+"---"+S1.getDisplayLanguage());
    }
}}
```

NumberFormatt Classes :-

→ Various Countries fallow various Conversions to represent number by using NumberFormatt class we can formatt a number according to a particular Locale.

→ NumberFormatt class present in java.text package & it is an abstract class. Hence we can't Create NumberFormatt object directly

~~NumberFormatt nf = new NumberFormatt();~~

Creating NumberFormatt object for the default Locale :-

→ NumberFormatt class defines the following methods for this

① public Static NumberFormatt getInstance();
② public Static NumberFormat getCurrencyInstance();
③ public Static NumberFormat getPercent Instance();
④ public Static NumberFormat getNumberInstance();

Getting NumberFormat object for a Specific Locale :-

→ we have to pass the Corresponding Locale object as arguement to the above methods

Ex.
① public static NumberFormat getCurrencyInstance(Locale l);
⋮

→ once we got NumberFormat object we can format & parse numbers by using the following methods of NumberFormat class

① public String format(long l);

② public String format(double d);

→ To format (or) convert java specific number form to Locale specific String form.

③ public Number parse(String s) throws ParseException

→ To convert Locale specific String form to java specific Number form.

Ex1:-

w.a.p to represent a Java number in Italy specific form.

Ⓟ
```
import java.text.*;
import java.util.*;
class NumberFormatDemo2
{
  p.s.v.m(____)
  {
    double d=123456.789;
    NumberFormat nf = NumberFormat.getInstance(Locale.ITALY);
    S.o.pln("Italy form is:"+ nf.format(d));
  }
}
```

O/P!- Italy form is: 123.456,789

Ex3:- w.a.p to represent a java number in India, U.K & U.S Currency forms.

```
import java.text.*;
import java.util.*;
Class NumberFormatDemo3
{
   p.s.v.m( ——— )
   {
    double d = 123456.789;
    Locale india = new Locale("pa","IN");
    NumberFormat nf1 = NumberFormat.getCurrencyInstance(india);
    S.o.pln("India notation is...."+nf1.format(d));
    NumberFormat nf2 = NumberFormat.getCurrencyInstance(Locale.US);
    S.o.pln(" US Notation is...."+nf2.format(d));
    NumberFormat nf3 = NumberFormat.getCurrencyInstance(Locale.UK);
    S.o.pln(" UK notation is...."+nf3.format(d));
   }
}
```

any Location in India

o/p:-
```
India Notation is ......  INR 123,456.79
US Notation is .......  $ 123,456.79
Uk Notation is .......  £ 123,456.79
```

Setting Maximum & minimum integer & fraction digits :-

→ NumberFormat class defines the following methods to set maximum & minimum fraction & integer digits.

① public void setMaximumFractionDigits(int n);

② public void setMinimumFractionDigits(int n);

③ public void setMaximumIntegerDigits(int n);

④ public void setMinimumIntegerDigits(int n);

18/5/11

Ex1.
```
NF nf = NF.getInstance();

① nf.setMaximumFractionDigits(2);
     S.o.pln(nf.format(123.4567));  // 123.45
     S.o.pln(nf.format(123.4));     //123.4
② nf.setMinimumFractionDigits(2);
     S.o.pln(nf.format(123.4567));  // 123.4567
     S.o.pln(nf.format(123.4));     // 123.40
③ nf.setMaximumIntegerDigits(3);
     S.o.pln(nf.format(123456.234));  // 456.234
     S.o.pln(nf.format(12.3456));     // 12.3456
④ nf.setMinimumIntegerDigits(3);
     S.o.pln(nf.formatt(123456.234));  //123456.234
     S.o.pln(nf.formatt(12.3456));  //  012.3456
```

DateFormat class :-

→ Various Countries fallow various Conversions to represent Date.

→ By using DateFormatt class we Can formatt the DATE according to a particular Locale.

→ DateFormat class is an abstract class & present in java.text pack.

Getting DateFormat object for Default Locale :-

DateFormat class defines the following methods for this

(1) public Static DateFormat getInstance();

(2) public Static DateFormat getDateInstance();

(3) public Static DateFormat getDateInstance(int Style);

DateFormat.Full → 0

DateFormat.LONG → 1

DateFormat.MEDIUM — 2

DateFormat.SHORT — 3

Getting DateFormat object for the Specific Locale :-

① Public Static DateFormat getDateInstance(int Style, Locale l);

→ Once we got DateFormat Object we Can format & parse dates by using the following Methods.

① Public String format(Date d);

→ To Convert Java Date form to Locale Specific String form

Note:-
Default Style is Medium & most of the Cases default Locale is US

② Public Date parse(String s) throws ParseException

To Convert Locale Specific Date Form to java Date Form.

Ex(1):-

W.a.p To display System Date in all possible Styles of U.S format

```
import java.util.*;
import java.text.*;
Class DateFormatDemo1
{
  p.s.v.m(______)
  {
    S.o.pln("Full form:" + DateFormat.getDateInstance(0).
                                        format(new Date()));
          (or)
    // DateFormat df = Dateformat.getDateInstance(0);
       S.o.pln(df.format(new Date());

    S.o.pln(" Long form:" + DF.getDateInstance(1). format(new Date())
    S.o.pln("MEDIUM form:" + DF.get DateInstance(2). Format(new Date()));
    S.o.pln(" SHORT form:" + DF.getDateInstance(3). format(new Date()));
  }
}
```

O/P:-
```
Full form : Thursday, February. 2. 2010
Long form : February 18, 2010
Medium form : Feb 18, 2010
Short form : 2/18/10
```

Ex2):-

(P) w.a.p to display System Date US, UK & Italy form.

```
{
 S.o.pln ("US form:" + DF.getDateInstance (0, Locale.US).format(
                                                   new Date()));
 S.o.pln("UK form:"+ DF.getDateInstance (0, Locale.UK). format(new Date()));
 S.o.pln ("ITALY form:" + DF.getDateInstance(0, Locale.ITALY) . format(new Date()));
}
```

O/p:-

```
US form :     Tuesday, May 18, 2010
UK form :     Tuesday, 18 May 2010
ITALY form:   martedo 18 maggie 2010
```

Getting DateFormat object to represent both DATE & TIME :-

① public static DateFormat getDateTimeInstance();

② public static DateFormat getDateTimeInstance(int datestyle, int timestyle);

③ public static Dateformat getDateTimeInstance(int dateStyle, int timeStyle, Locale);

Ex!:-

```
S.o.pln ("US form :" +DateFormat.getDateTimeInstance (0, 0, Locale.US)
                                                .format(new Date()));
```

O/p:-

```
US form : Tuesday, May 18, 2011   9:53:45 AM  GMT :+5:30
```

Note:- Default style is medium & most of the cases default Locale is US