

# Les objets-format en Java



Extrait de la hiérarchie des formats

Le concept du format de nombre  
Formatage de nombre décimal

Concept du format de date  
Formatage d'une date

Xavier HER

1



## L'objet « format »

- A quoi ça sert ?
- Exemple d'utilisation d'un objet format :

```
//Utilisation pour « formater » une chaîne de caractères
SimpleDateFormat form = new SimpleDateFormat("dd/MM/yyyy");

System.out.println("Aujourd'hui : " + form.format(new Date()));

//Affichage
// Aujourd'hui :14/09/2011

//Utilisation pour « parser » (ou scanner)
Date d2 = form.parse("13/09/2010");

System.out.println("il y a un an: " + form.format(d2));
//Affichage
// Aujourd'hui :13/09/2010
```

- En cas de pb : java.text.ParseException

XH

2



## Le concept du format

<i>Format</i> (from text)
<pre> + Format() + format(arg0 : Object) : String + format(arg0 : Object, arg1 : StringBuffer, arg2 : FieldPosition) : StringBuffer + parseObject(arg0 : String, arg1 : ParsePosition) : Object + parseObject(arg0 : String) : Object + clone() : Object </pre>

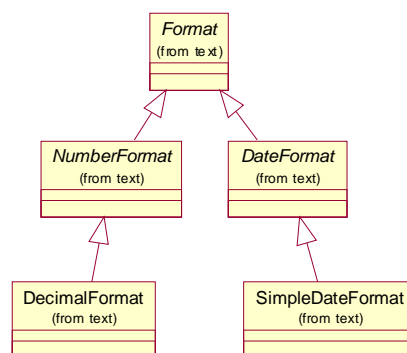
- La classe `java.text.Format` est tête de hiérarchie
- Avec un objet format, on peut
  - récupérer une chaîne formatée (méthode `format`),
  - « parser » une chaîne pour savoir s'il elle suit le format et éventuellement récupérer l'objet correspondant,
  - paramétrer ce format (`applyPattern`)
- La classe `Format` est abstraite
  - seule formatage possible jusqu'à J2SE 1.4 (compris)

XH

3



## L'API Java



- Diagramme de classe qui donne un extrait de la hiérarchie des formats de l'API Java
  - les classes les plus utiles
- Toutes les classes sont **abstraites** sauf les deux filles

XH

4



## Depuis Java 5, autre possibilité pour formater ou parser

- La classe `java.util.Formatter`
  - Nouveau objet pour formater une chaîne
  - Voir aussi le `printf` des flux

```
System.out.println( "-----Affichage avec printf:-----");
System.out.printf( "%tD \n" , new Date() );
System.out.printf( "%tF \n" , new Date() );
System.out.printf( "%1$te %1$tb %1$ty \n" , new Date() );
System.out.printf( "%1$tA %1$te %1$TB %1$tY \n" , new Date() );
System.out.printf( "%1$tr \n" , new Date() );
System.out.println( "-----" );
```

- La classe `java.util.Scanner`
  - Nouveau objet pour parser

XH

5



Format  
(from text)

## Le concept du format de nombre

- Classe `NumberFormat`
  - Classe abstraite
- Méthodes `getXXXInstance` sont statiques

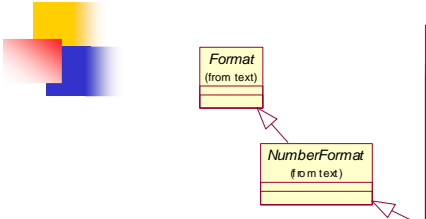
**NumberFormat**  
(from text)

```
+ INTEGER_FIELD : int = 0
+ FRACTION_FIELD : int = 1
```

```
+ NumberFormat()
+ format(arg0 : Object, arg1 : StringBuffer, arg2 : FieldPosition) : StringBuffer
+ parseObject(arg0 : String, arg1 : ParsePosition) : Object
+ format(arg0 : double) : String
+ format(arg0 : long) : String
+ format(arg0 : double, arg1 : StringBuffer, arg2 : FieldPosition) : StringBuffer
+ format(arg0 : long, arg1 : StringBuffer, arg2 : FieldPosition) : StringBuffer
+ parse(arg0 : String, arg1 : ParsePosition) : Number
+ parse(arg0 : String) : Number
+ isParseIntegerOnly() : boolean
+ setParseIntegerOnly(arg0 : boolean) : void
+ getInstance() : NumberFormat
+ getInstance(arg0 : Locale) : NumberFormat
+ getNumberInstance() : NumberFormat
+ getNumberInstance(arg0 : Locale) : NumberFormat
+ getCurrencyInstance() : NumberFormat
+ getCurrencyInstance(arg0 : Locale) : NumberFormat
+ getPercentInstance() : NumberFormat
+ getPercentInstance(arg0 : Locale) : NumberFormat
+ getAvailableLocales() : Locale[]
+ hashCode() : int
+ equals(arg0 : Object) : boolean
+ clone() : Object
+ isGroupingUsed() : boolean
+ setGroupingUsed(arg0 : boolean) : void
+ getMaximumIntegerDigits() : int
+ setMaximumIntegerDigits(arg0 : int) : void
+ getMinimumIntegerDigits() : int
+ setMinimumIntegerDigits(arg0 : int) : void
+ getMaximumFractionDigits() : int
+ setMaximumFractionDigits(arg0 : int) : void
+ getMinimumFractionDigits() : int
+ setMinimumFractionDigits(arg0 : int) : void
```

XH

6



## Formatage de nombre décimal (virgule flottante)

- Classe DecimalFormat
- A ne pas utiliser directement ?
- Classe concrète permettant de :
  - récupérer une chaîne qu'il aurait formatée (format),
  - « parser » une chaîne pour savoir s'elle elle suit le format et récupérer l'objet wrapper
  - Possibilité de paramétrer le format (applyPattern)
  - Préfix-suffix

```


classDiagram
    class Format {
        <<from text>>
    }
    class NumberFormat {
        <<from text>>
    }
    class DecimalFormat {
        <<from text>>
    }
    Format <|-- NumberFormat
    NumberFormat <|-- DecimalFormat
  
```

**DecimalFormat (from text)**

```

+ DecimalFormat()
+ DecimalFormat(arg0 : String)
+ DecimalFormat(arg0 : String, arg1 : DecimalFormatSymbols)
+ format(arg0 : double, arg1 : StringBuffer, arg2 : FieldPosition) : StringBuffer
+ format(arg0 : long, arg1 : StringBuffer, arg2 : FieldPosition) : StringBuffer
+ parse(arg0 : String, arg1 : ParsePosition) : Number
+ getDecimalFormatSymbols() : DecimalFormatSymbols
+ setDecimalFormatSymbols(arg0 : DecimalFormatSymbols) : void
+ getPositivePrefix() : String
+ setPositivePrefix(arg0 : String) : void
+ getNegativePrefix() : String
+ setNegativePrefix(arg0 : String) : void
+ getPositiveSuffix() : String
+ setPositiveSuffix(arg0 : String) : void
+ getNegativeSuffix() : String
+ setNegativeSuffix(arg0 : String) : void
+ getMultiplier() : int
+ setMultiplier(arg0 : int) : void
+ getGroupingSize() : int
+ setGroupingSize(arg0 : int) : void
+ isDecimalSeparatorAlwaysShown() : boolean
+ setDecimalSeparatorAlwaysShown(arg0 : boolean) : void
+ clone() : Object
+ equals(arg0 : Object) : boolean
+ hashCode() : int
+ toPattern() : String
+ toLocalizedPattern() : String
+ applyPattern(arg0 : String) : void
+ applyLocalizedPattern(arg0 : String) : void
+ setMaximumIntegerDigits(arg0 : int) : void
+ setMinimumIntegerDigits(arg0 : int) : void
+ setMaximumFractionDigits(arg0 : int) : void
+ setMinimumFractionDigits(arg0 : int) : void
  
```

XH 7



## Exemples et documentation

- Voir la démo (classe TestDate)
- Voir les exemples fournis sur le ftp
- Voir la javadoc de DecimalFormat,...

XH 8

## Concept du format de date

- Classe DateFormat
  - Classe abstraite
- tolérant (lenient) par défaut
- Méthodes getXXXInstance sont statiques

XH

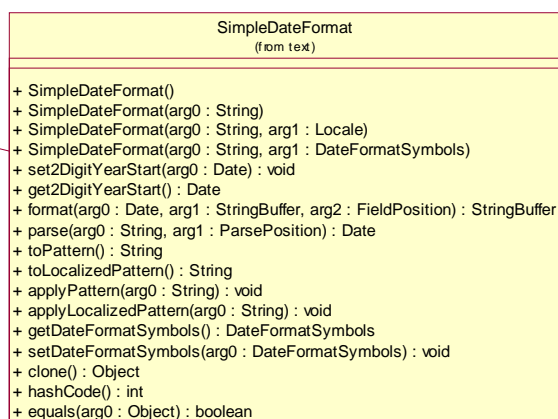


9

## Formatage d'une date

- Classe SimpleDateFormat
- Classe concrète permettant de :
  - récupérer une chaîne qu'elle aura formatée (format),
  - « parser » une chaîne de date pour savoir s'il elle suit le format et récupérer un objet Date,
  - Possibilité de paramétrer le format (applyPattern)
  - Préfix-suffix

XH



10



XH

11