第十章java.lang套件

資訊科技系
林偉川

包裝器類別

- 可以包裝基本資料型別數值的類別，這些類別稱為包裝器（Wrappers）。
- 對應基本資料型別的包裝器類別為：Boolean、Byte、Short、Character、Integer、Long、Float和Double八個類別。
- 包裝器物件是不可變更的（immutable），物件建立後，其包裝的數值就不可改變。
- Byte、Short、Integer、Long、Float和Double六個類別都是和數值相關的類別，它們都繼承**Number**抽象類別。

2

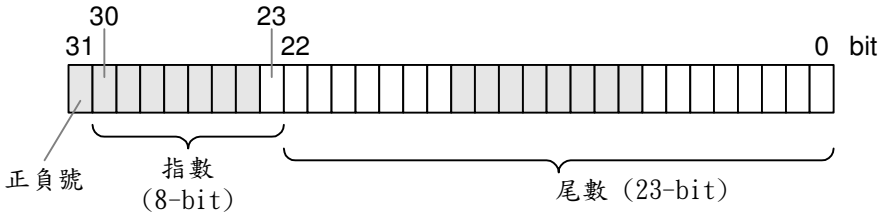
10-1-1 字串轉換成基本型別

類別	方法	說明
Boolean	boolean **getBoolean**(**String s**)	若字串s轉換成小寫後為"true"時，則傳回布林值true，否則傳回false。
Byte	byte **parseByte**(**String s**)	轉換成Byte型別。
Short	short **parseShort**(**String s**)	轉換成Short型別。
Integer	int **parseInt**(**String s**)	轉換成Integer型別。
Long	long **parseLong**(**String s**)	轉換成Long型別。
Float	float **parseFloat**(**String s**)	轉換成Float型別。
Double	double **parseDouble**(**String s**)	轉換成Double型別。

3

10-1-2 浮點數包裝器

- float型別各bit的意義(32 bits)



4

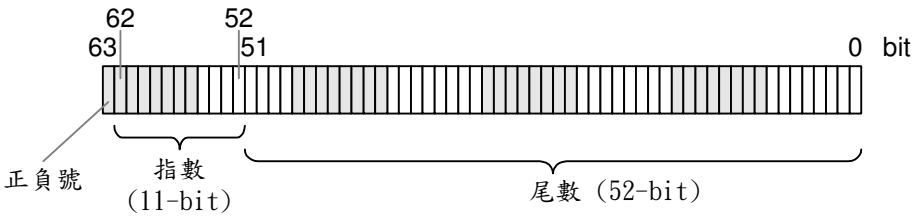
10-1-2 浮點數包裝器

- float數值表示方式的四個特列：
  - 0：無法使用2的次方表示，使用所有bits皆為0表示。
  - POSITIVE_INFINITY：以所有exponential-bits為1，其餘bits為0表示。
  - NEGATIVE_INFINITY：以sign-bit為1，所有exponential-bits為1，所有mantissa-bits為0表示。
  - NaN：以sign-bit為0，所有exponential-bits為1，mantissa-bits中左側第一個為1其餘為0表示。

5

10-1-2 浮點數包裝器

- double型別各bit的意義(64 bits)



6

10-1-3 自動包裝(Autoboxing)與解裝(Unboxing)

- 之前的寫法:

```
int i = 10;
// 轉成包裝器物件
Integer j = new Integer(i);
// 取得基本型別的值
int k = j.intValue();
```

- 有Autoboxing之後的寫法:

```
int i = 10;
Integer j = i;
int k = j;
```


10-1-3 自動包裝(Autoboxing)與解裝(Unboxing)

- 便利之餘要注意的是，每個基本型別的值都只自動包裝到對應的包裝器物件，跟自動型別轉換沒有直接的關係。
- 自動包裝與解裝的另一個問題是，可能在方法多載時造成曖昧不明的情況。發生這種情況時，可暫時忘掉自動包裝與解裝，再推敲結果。

Math類別

- Math類別的常數

靜態常數	說明
double **E**	尤拉常數，自然對數的底數。
double **PI**	圓周率。

9

Math類別

- Math類別的三角函數方法

三角函數方法	說明
double **acos(double a)**	求反餘弦。
double **asin(double a)**	求反正弦。
double **atan(double a)**	求反正切。
double **atan2(double y, double x)**	以x和y為座標，求反正切。
double **cos(double a)**	以a為徑度求餘弦。
double **sin(double a)**	以a為徑度求正弦。
double **tan(double a)**	以a為徑度求正切。

10

Math類別

- Math類別的其它常用方法

方法	說明
double **ceil(double a)**	取得不小於a的double型別之整數。
double **exp(double a)**	求得e的a次方。
double **floor(double a)**	取得不大於a的double型別之整數。
double **log(double a)**	以e為底求對數值。
double **max(double a, double b)**	回傳a和b中較大者。
double **min(double a, double b)**	回傳a和b中較小者。
double **pow(double a, double b)**	求a的b次方。
double **random()**	取得介於0.0 ~ 1.0（小於1.0）之間的亂數值。
long **round(double a)**	四捨五入求整數值。
double **sqrt(double a)**	求a的平方根。

11

String類別

- **java.lang**是會自動被引入的套件，其中字串相關的類別為**String**和**StringBuffer**。
- String類別的常用建構子

String建構子	說明
String(byte[] bytes)	以byte陣列建立。
String(char[] value)	以字元陣列建立。
String(char[] value, int offset, int count)	以字元陣列中的部份內容建立。
String(String value)	指定字串常數建立。
String(StringBuffer buffer)	以StringBuffer物件建立。

12

- 直接讓String型別的參照變數指向字串常數

```
String str = "我愛Java!";
```

- String物件為不可變更的（immutable）物件，其實體內的字元或字串長度都不可變更。
- String物件的長度可以透過length()方法取得。
- 利用String物件的charAt()方法可以使用索引值取得字串裡的某個字元

```
字串名稱.charAt(索引)
```


10-3-1 字串的連接

- 使用加號「+」

```
str1 = str2 + str3;
```

- 使用「+=」

```
str1 += "ABC";
```

- 使用concat()方法

```
str1.concat(str2)
```

- String物件為不可變更的（immutable）物件，呼叫其物件方法都不會對原物件的內容造成影響。

10-3-2 尋找字元和子字串

- String類別的尋找字元和子字串的方法

尋找字元和子字串的方法	簡單說明
int indexOf (int ch)	尋找字元ch的位置。
int indexOf (int ch, int fromIndex)	從fromIndex開始尋找字元ch。
int indexOf (String str)	尋找字串str的位置。
int indexOf (String str, int fromIndex)	從fromIndex開始尋找字串str。
int lastIndexOf (int ch)	尋找最後一個字元ch的位置。
int lastIndexOf (int ch, int fromIndex)	從fromIndex開始尋找最後一個ch。
int lastIndexOf (String str)	尋找最後一個字串str的位置。
int lastIndexOf (String str, int fromIndex)	從fromIndex開始尋找最後一個str。

10-3-3 擷取子字串

- String類別的擷取子字串方法

方法	簡單說明
public String **substring**(int beginIndex)	傳回索引從beginIndex到最後的子字串。
public String **substring**(int beginIndex, int endIndex)	傳回索引從beginIndex到endIndex的子字串（不包含endIndex的字元）。

10-3-4 字串的比較

- 以比較運算子「==」判斷兩個String物件時，是判斷兩者是否指向相同的參考。
- String類別的equals()方法，其功能是比較兩個String物件的內容而不是參照。

```
字串一.equals(字串二)
```

- 如果要忽略英文字母的大小寫，可以使用equalsIgnoreCase()方法。

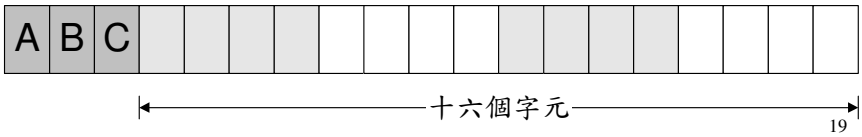
10-3-5 String類別的其它常用方法

方法	簡單說明
public static String valueOf (Object obj)	將物件轉成字串傳回。
public String toLowerCase ()	將字串內的字母轉換成小寫。
public String toUpperCase ()	將字串內的字母轉換成大寫。
public String replace (char oldChar, char newChar)	將字串內的所有oldChar字元換成newChar字元後，傳回整個字串。

- StringBuffer類別的建構子

建構子	簡單說明
StringBuffer ()	建立後，物件擁有16個字元大小的字串空間。
StringBuffer (int length)	建立擁有length個字元大小的字串空間。
StringBuffer (String str)	複製一份str內容，同時再加上16個字元大小的空間。

- StringBuffer物件會預留16個字元的空間



10-4-1 StringBuffer物件的容量與內容大小

方法	簡單說明
public int **length**()	取得內容字串的大小。
public void **setLength**(int newLength)	設定內容字串的大小為newLength。
public int **capacity**()	取得容量的大小。
public void **ensureCapacity**(int minCp)	設定最少有minCp個字元大小的容量。

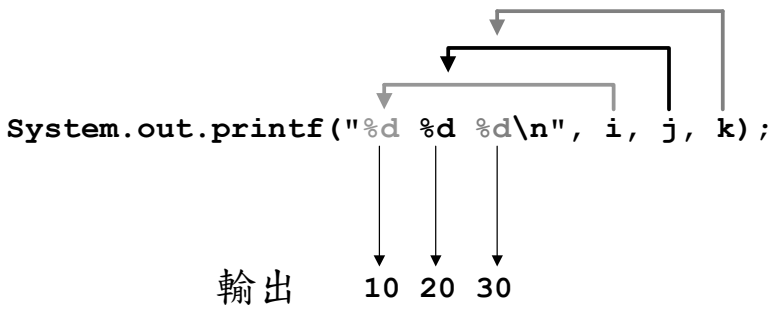
10-4-2 StringBuffer物件內容的變更

方法	簡單說明
void **setCharAt**(int index, char ch)	在位置index處，設定為ch字元。
StringBuffer **reverse**()	將整個字串頭尾顛倒過來。
StringBuffer **append**(Object obj)	將各類資料加到內容的後面。
StringBuffer **insert**(int offset, String str)	在位置offset插入字串str。
StringBuffer **delete**(int start, int end)	刪除start和end之間的字元。

21

10-5-1 System.out.printf()

- 格式對應字符和變數的對應關係



22

格式對應字符	說明
%%	輸出一個百分比符號（%）。
%b, %B	輸出 ”true” 或 ”false”。%B輸出全大寫。
%c, %C	輸出字元。
%d	十進位整數輸出。
%e, %E	浮點數以指數的方式輸出。
%f	浮點數不以指數的方式輸出。
%g, %G	取%e和%f之中較短者。
%h, %H	相當於使用Integer.toHexString(arg.hashCode())，以十六進位輸出參數的雜湊碼。
%n	平台上的換行。
%o	整數的八進位輸出。
%s, %S	字串。
%t, %T	時間格式。
%x, %X	整數的十六進位輸出。

10-5-2 java.util.Formatter類別

- 格式對應字符的完整語法格式，中括號包起來的表示為選擇性設定：

%[參數索引$][旗標][寬度][.準確位數]轉換控制字元

格式化輸出

- 格式旗標

格式旗標	說明
-	依寬度向左對齊，預設為向右對齊。
#	若使用%o輸出時前置為0，若使用%x輸出時前置為0x。
+	數值帶有正負號。
空白字元	正值前放置一個空白，對齊用。
(	以括號包裹負值。
0	左側補0。
,	使用區域字元。

25

格式化輸出

- 時間格式化對應字符

對應字符	說明
%tH	小時，00~23。
%tI	小時，01~12。
%tk	小時，0~23。
%tl	小時，1~12。
%tM	分鐘，00~59。
%tS	秒數，00~59。
%tL	毫秒，000~999。
%tN	奈秒，九位數表示。
%tp	locale的上午、下午。
%tz	和GMT時差。
%tZ	時區的簡寫名。
%ts	從1970的第一秒算起的秒數。
%tQ	從1970的第一毫秒算起的毫秒數。

26

格式化輸出

- 日期格式化對應字符

對應字符	說明
%tB	locale的月份名（完整名）。
%tb	locale的月份名（簡寫名）。
%th	同「%tb」。
%tA	locale的星期幾（完整名）。
%ta	locale的星期幾（簡寫名）。
%tC	西元四位數年除以100的商。
%tY	西元四位數年。
%ty	西元兩位數年。
%tj	一年中的第幾天，001~366。
%tm	月份，01~12。
%td	一月中的第幾日，01~31。
%te	一月中的第幾日，1~31。

27

格式化輸出

- 時間或日期組合格式化對應字符

對應字符	說明
%tR	相當於「%tH:%tM」。
%tT	相當於「%tH:%tM:%tS」。
%tr	相當於「%tI:%tM:%tS %Tp」。
%tD	相當於「%tm/%td/%ty」。
%tF	相當於「%tY-%tm-%td」。
%tc	完整的時間和日期，相當於「%ta %tb %td %tT %tZ %tY」。

28

正規表示法

- J2SE 支援正規表示法的套件為 java.util.regex套件，其中包含兩個類別Pattern和Matcher。
- 在Java使用正規表示法，必須先以Pattern類別的靜態方法compile()，由正規表示法的字串建立一個Pattern物件，再使用Pattern物件的matcher(CharSequence input)方法做比對，然後回傳Matcher物件。

29

正規表示法

- 單一字元的表示方式

\b	字的界限如空白	\bbe符合者為"be"、"being"；"abed"則不符合。
\B	「非」字的界限	\Bbe符合者為" abed"；"be"則不符合。
\d	數字0~9	\d\d符合者為"22"；"2c"則不符合。
\D	「非」數字	\D\D符合者為"ac"；"2c"則不符合。
\s	一個空白（space）	a\sbar符合者為"a bar"；"abar"則不符合。
\S	「非」空白	a\Sbar符合者為"a-bar"；"abar"和"a bar"不符合。
\w	字母、數字或底線（_）	c\w符合者為"c7"；"c#"和"c-"不符合。
\W	「非」字母、數字或底線	c\W符合者為"c%"；"ca"和"c_"不符合。
.	任何字元（不包含換行）	a..符合者可為"a12"、"ap+"、"a##"。
[]	中括號中任一字元	b[ae]d符合者可為"bad"、"bed"。
[^]	不包含中括號中任一字元	b[^ae]d符合者可為"b-d"、"bod"；"bad"和"bed"不符合。

30

正規表示法

- 多字元表示方式(貪婪計量子)

*	重複0次或多次	lo*p符合者可 為"lp"、"lop"、"loop"、"loop"。
?	重複0次或1次	lo?p符合者為"lp"、"lop"。
+	重複1次或多次	lo+p符合者可 為"lop"、"loop"、"looop"。
{n}	重複n次	ba{2}d符合者為"baad"。
{n,}	重複n次或以上	ba{2,}d符合者可為"baad"、"baaad"。
{n,m}	重複n次至m次之間	ba{1,2}d符合者為"bad"、"baad"。

31

正規表示法

- 貪婪計量子（Greedy quantifiers）會儘量找尋較長的字串。例如表示式為「lo*」，當搜尋的對象為 "looop" 時，搜尋到的會是 "looo"，而不是 "loo"、"lo" 或 "l"。

- 貪婪計量子後面接個「?」時，會變成自閉計量子（Reluctant quantifiers）儘量找尋較短的字串。例如表示式為「lo+?」，當搜尋的對象為 "looop" 時，搜尋到的會是 "lo"，而不是 "loo" 或 "looo"。

32

- 位置表示方式

表示法	說明	例子
^	字首	^pos符合者可為”pose”；”apos”不符合。
$	字尾	ring$符合者為”spring”；” ringer”不符合。

33