

신기술/기능 요구사항 지원 보고서  
JAVA1.5와 JAVA1.6의  
기능 분석 보고서(I)

한국소프트웨어진흥원  
공개SW기술지원센터

<Revision 정보>

일자	VERSION	변경내역	작성자
2007. 05. 14	0.1	초기 작성	허종윤

## 목 차

1. 문서 개요 .....	5
가. 문서의 목적 .....	5
나. 본 문서의 사용방법 .....	5
다. 참고 사항 .....	5
2. 새로운 패키지과 수정된 패키지 .....	6
가. 새로운 패키지 .....	6
나. 수정된 패키지 .....	6
3. 새로운 패키지 .....	8
3.1 PACKAGE java.text.spi .....	8
3.2 PACKAGE java.util.spi .....	13
4. 수정된 패키지 .....	15
4.1 PACKAGE java.applet .....	15
4.2 PACKAGE java.awt .....	16
4.3 PACKAGE java.awt.event .....	17
4.4 PACKAGE java.awt.font .....	17
4.5 PACKAGE java.awt.geom .....	18
4.6 PACKAGE java.awt.im .....	19
4.7 PACKAGE java.awt.im.spi .....	19
4.8 PACKAGE java.awt.image .....	19
4.9 PACKAGE java.awt.print .....	20
4.10 PACKAGE java.beans .....	20
4.11 PACKAGE java.beans.beancontext .....	21
4.12 PACKAGE java.io .....	21
4.13 PACKAGE java.lang .....	24
4.14 PACKAGE java.lang.annotation .....	26
4.15 PACKAGE java.lang.instrument .....	26
4.16 PACKAGE java.lang.management .....	27
4.17 PACKAGE java.lang.reflect .....	27
4.18 PACKAGE java.math .....	28
4.19 PACKAGE java.net .....	28
4.20 PACKAGE java.nio .....	29
4.21 PACKAGE java.nio.channels .....	30
4.22 PACKAGE java.nio.charset .....	30
4.23 PACKAGE java.rmi .....	31
4.24 PACKAGE java.rmi.activation .....	31

4.25 PACKAGE java.security .....	31
4.26 PACKAGE java.security.cert .....	33
4.27 PACKAGE java.security.interfaces .....	34
4.28 PACKAGE java.security.spec .....	34
4.29 PACKAGE java.text .....	34
4.30 PACKAGE java.util .....	35
4.31 PACKAGE java.util.concurrent .....	41
5. 성능비교 .....	45
5. 참고문헌 .....	45

## 1. 문서 개요

본 문서는 JAVA1.6을 중심으로 JAVA1.5와의 차이점과 새로운 기능들에 대한 분석을 통해 국내 JAVA개발자들의 기술 향상에 도움을 주고자 제작되었다.

### 가. 문서의 목적

다음과 같은 세부적인 목적을 달성하기 위하여 작성되었다.

- JAVA1.6 (JAVA6.0) 새로운 기능 분석
- 신기술의 장점 및 특이점 분석

### 나. 본 문서의 사용방법

다음과 같은 방법으로 사용할 수 있다.

- 국내 JAVA개발업체들에게 새롭게 추가된 기술을 소개할 수 있다.
- 사용자들에게 새로운 기능들에 대한 내용을 홍보할 수 있다.

### 다. 참고사항

- 본 문서는 JAVA1.6에 새로 추가된 2개의 PACKAGE와 기능이 수정된 31개의 PACKAGE를 분석하였고, 나머지 PACKAGE는 차후에 기능 분석할 예정이다.
- 개발지원시 시스템 관련사항, 기술적 배경 등 참고사항들을 기술한다.

## 2. 새로운 패키지과 수정된 패키지

### 가. 새로운 패키지

JAVA1.6에서는 JAVA1.5와 비교하여 다음과 같은 패키지가 추가되었다.

- java.text.spi
- java.util.spi

### 나. 수정된 패키지

JAVA1.6에서는 JAVA1.5와 다음과 같은 패키지가 수정되었다.

- java.applet
- java.awt
- java.awt.event
- java.awt.font
- java.awt.geom
- java.awt.im
- java.awt.im.spi
- java.awt.image
- java.awt.print
- java.beans
- java.beans.beancontext
- java.io
- java.lang
- java.lang.annotation
- java.lang.instrument
- java.lang.management
- java.lang.reflect
- java.math
- java.net
- java.nio
- java.nio.channels
- java.nio.charset
- java.rmi
- java.rmi.activation
- java.security
- java.security.cert
- java.security.interfaces
- java.security.spec
- java.text
- java.util
- java.util.concurrent

- ☐ java.util.concurrent.atomic
- ☐ java.util.concurrent.locks
- ☐ java.util.jar
- ☐ java.util.logging
- ☐ java.util.prefs
- ☐ java.util.regex
- ☐ java.util.zip
- ☐ javax.accessibility
- ☐ javax.crypto
- ☐ javax.crypto.interfaces
- ☐ javax.crypto.spec
- ☐ javax.imageio
- ☐ javax.imageio.metadata
- ☐ javax.net.ssl
- ☐ javax.print
- ☐ javax.security.auth
- ☐ javax.security.auth.callback
- ☐ javax.security.auth.kerberos
- ☐ javax.security.auth.login
- ☐ javax.security.auth.x500
- ☐ javax.security.cert
- ☐ javax.security.sasl
- ☐ javax.sound.midi
- ☐ javax.swing
- ☐ javax.swing.border
- ☐ javax.swing.colorchooser
- ☐ javax.swing.event
- ☐ javax.swing.plaf
- ☐ javax.swing.plaf.basic
- ☐ javax.swing.plaf.metal
- ☐ javax.swing.plaf.synth
- ☐ javax.swing.table
- ☐ javax.swing.text
- ☐ javax.swing.text.html
- ☐ javax.swing.tree
- ☐ javax.swing.undo

### 3. 새로운 패키지

#### 3.1 PACKAGE java.text.spi

CLASS 요약	
BreakIteratorProvider	BreakIterator class 기능향상
CollatorProvider	Collator class 기능향상
DateFormatProvider	DateFormat class 기능향상
DateFormatSymbolsProvider	DateFormatSymbol class 기능향상
DecimalFormatSymbolProvider	DecimalFormatSymbol class 기능향상
NumberFormatProvider	NumberFormat class 기능향상

##### 3.1.1 BreakIterator class

BreakIterator 클래스는, 텍스트내의 경계의 위치를 찾아내는 메소드를 구현합니다. BreakIterator 의 인스턴스는 현재의 위치를 유지해, 텍스트를 스캔 해 경계가 발생하는 문자의 인덱스를 돌려줍니다. 내부적으로는,BreakIterator 는 CharacterIterator 를 사용해 텍스트를 스캔 하기 위한(해), 이 프로토콜을 구현하는 임의의 오브젝트에 의해 보관 유지되는 텍스트를 스캔 할 수 있습니다. StringCharacterIterator 는,setText 에게 건네진 String 오브젝트의 스캔에 사용됩니다.

이 클래스에 의해 제공되는 팩토리 메소드를 사용해, 다양한 형태의 분할 반복자의 인스턴스를 생성합니다. 특히, 단어, 행, 문, 및 문자의 경계 해석을 실행하는 BreakIterator 를 생성하려면 , 각각 getWordIterator, getLineIterator, getSentenceliterator, 및 getCharacterIterator를 사용합니다. 단일의 BreakIterator 는, 1 개의 유닛(단어, 행, 문장등)만을 처리하기 때문에, 실행하는 유닛 경계 해석 마다 다른 반복자를 사용할 필요가 있습니다. 행의 경계 해석에서는, 텍스트 캐릭터 라인을 행 즉시 로 분할하는 위치를 판정합니다. 구두점 및 하이픈 자동 삽입 된 단어도, 기구에 의해 올바르게 처리됩니다.

문장의 경계 해석에서는, 숫자와 약어의 피리어드, 인용부호나 괄호 등에 계속되는 구두점의 올바른 해석에 대해 선택이 가능합니다.

단어의 경계 해석은, 검색 치환 기능으로 사용됩니다. 또, 텍스트 편집 어플리케이션내에서, 더블 클릭에 의해 단어가 선택 가능하게 됩니다. 단어 선택에서는, 구두점과 함께, 기호나 구두점등과 같이 단어가 일부에서 없는 문자, 전후에 분할이 있는 문자도 올바르게 해석됩니다.

문자의 경계 해석에서는, 예를 들어 커서를 텍스트 캐릭터 라인에 따라 움직이는 것 같은 경우에, 유저가 예측하는 대로로 조작을 하도록(듯이) 합니다. 문자의 경계 해석에 의해, 문자의 포함 방법으로 의존하지 않고, 캐릭터 라인에 올바른 네비게이션이 가능하게 됩니다. 예를 들어, 액센트 첨부의 문자는, 기존 문자와 발음 구별 부호로서 포함되고 있는 경우가 있습니다. 유저의 문자에 대한 인식은 언어간에 다릅니다.

BreakIterator 는 자연어에서의 사용만을 상정하고 있습니다. 이 클래스는, 프로그램 언어를 토큰화하기 위해서(때문에)는 사용할 수 없습니다.

예)

```
public static void main(String args[]) {
    if (args.length == 1) {
```

```
String stringToExamine = args[0];
//print each word in order
BreakIterator boundary = BreakIterator.getWordInstance();
boundary.setText(stringToExamine);
printEachForward(boundary, stringToExamine);
//print each sentence in reverse order
boundary = BreakIterator.getSentenceInstance(Locale.US);
boundary.setText(stringToExamine);
printEachBackward(boundary, stringToExamine);
printFirst(boundary, stringToExamine);
printLast(boundary, stringToExamine);
}
}
```

### 3.1.2 Collator class

Collator 클래스는, 로케일에 의존하는 String 의 비교를 실시합니다. 이 클래스를 사용해, 자연 언어 텍스트의 검색과 소트 루틴을 구축합니다.

Collator 는, 추상 base class입니다. 서브 클래스에 의해, 특징의 조합 방법을 구현합니다. 현재는, 1 개의 서브 클래스 RuleBasedCollator 가 Java 2 플랫폼에서 제공되고 있어 광범위한 언어에 적용 가능합니다. 한층 더 특수한 필요성이 있는 경우에는, 다른 서브 클래스를 작성할 수 있습니다.

로케일에 의존하는 것 외의 클래스와 같이, static 팩토리 메소드 getInstance 를 사용해, 어느 로케일에 대한 적절한 Collator 오브젝트를 얻을 수 있습니다. 특징의 조합 방법의 상세를 이해하거나 그방법을 수정하거나 하는 경우에는, Collator 의 서브 클래스를 보는 것으로 충분합니다.

다음의 예는, 디폴트 로케일의 Collator 를 사용해 2 개의 캐릭터 라인을 비교하는 방법을 나타낸 것입니다.

```
// Compare two strings in the default locale
Collator myCollator = Collator.getInstance();
if( myCollator.compare("abc", "ABC") < 0 )
    System.out.println("abc is less than ABC");
else
    System.out.println("abc is greater than or equal to ABC");
//Get the Collator for US English and set its strength to PRIMARY
Collator usCollator = Collator.getInstance(Locale.US);
usCollator.setStrength(Collator.PRIMARY);
if( usCollator.compare("abc", "ABC") == 0 ) {
    System.out.println("Strings are equivalent");
}
```

### 3.1.3 DateFormat class

DateFormat 클래스는, 언어에 의존하지 않는 방법으로 일자 또는 시각을 포맷 및 해석하는, 일자/시각 포맷 서브 클래스의 추상 클래스입니다. SimpleDateFormat 등의 일자/시각 포맷 서브 클래스에 의해, 포맷 (일자 -> 텍스트), 해석 (텍스트 -> 일자), 및 정규화를 실시할 수가 있습니다. 일자는, Date 오브젝트 또는 1970 년 1 월 1 일 그리니치 표준시 00:00:00 부터의 밀리 세컨드로 표현됩니다.

DateFormat 는, 디폴트치, 지정된 로케일, 많은 포맷 스타일에 근거해, 디폴트의 일자/시각

포매터를 취득하는 많은 클래스 메소드를 제공합니다. 포맷 스타일에는 FULL, LONG, MEDIUM, 및 SHORT 가 있습니다. 이러한 스타일의 상세와 사용예에 대해서는, 메소드의 설명을 참조해 주세요.

DateFormat 는, 모든 로케일의 일자에 대해 포맷 및 해석을 지원합니다. 코드는, 달, 요일, 달력 포맷, 음력 태양력 등 로케일의 관습과는 완전하게 독립하고 있습니다.

현재의 로케일용으로 일자를 포맷 하기 위해서는, 다음의 static 팩토리 메소드의 1 개를 사용합니다.

```
myString = DateFormat.getDateInstance().format(myDate);
```

복수의 일자를 포맷 하는 경우에는, 그 포맷을 취득해, 그것을 여러 차례 사용하는 (분)편이 효율적입니다. 이것에 의해, 시스템이 로컬의 언어와 나라의 룰을 몇번이나 취득할 필요가 없어집니다.

```
DateFormat df = DateFormat.getDateInstance();
for (int i = 0; i < myDate.length; ++i) {
    output.println(df.format(myDate[i]) + " ");
}
```

어느 날 첨부를 다른 로케일에 대해서 포맷 하는 경우에는, getDateInstance()의 호출해에 그 일자를 지정합니다.

```
DateFormat df = DateFormat.getDateInstance(DateFormat.LONG, Locale.FRANCE);
```

게다가 해석을 위해서(때문에) DateFormat 를 사용할 수 있습니다.

```
myDate = df.parse(myString);
```

그 나라의 통상의 일자 포맷을 취득하려면, getDateInstance 를 사용합니다. 그 밖에도 사용 가능한 static 팩토리 메소드가 있습니다. 그 나라의 시각 포맷을 취득하려면, getTimeInstance 를 사용합니다. 일자와 시각 포맷을 취득하려면, getDateTimeInstance 를 사용합니다. 결과의 길이를 제어하기 위해서, 이러한 팩토리 메소드를 다른 옵션으로 건네줄 수가 있습니다 (SHORT 로부터 MEDIUM, LONG, FULL 에). 정확한 결과는 로케일에 의존합니다만, 일반적으로는 다음과 같이 됩니다.

- SHORT 는 모두 수치로, 12.13. 52 나 3:30pm 등
- MEDIUM 는 그것보다 긴 형식에서, Jan 12, 1952 등
- LONG 는 한층 더 긴 형식에서, January 12, 1952 나 3:30:32pm 등
- FULL 는 거의 완전하게 지정하는 형식에서, Tuesday, April 12, 1952 AD 나 3:30:42pm PST 등

필요에 따라서, 포맷에 타임 존을 설정할 수도 있습니다. 포맷이나 해석을 한층 더 제어하는 경우 (또는, 유저에게 그러한 제어를 주는 경우)에는, 팩토리 메소드로부터 얻은 DateFormat 을 SimpleDateFormat 에 캐스트 하는 방법을 시행할 수도 있습니다. 이 방법은 대부분의 나라에 유효하지만, 그렇지 않은 경우에 대비해, try 블록에 넣습니다.

게다가 parse 와 format 메소드의 형식을 ParsePosition 와 FieldPosition 와 함께 사용하

면, 다음 일 을 할 수가 있습니다.

- 캐릭터 라인의 각각의 부분을 단계적으로 해석한다
- 특정의 필드를 위치 맞춤 하거나 화면상의 어디가 선택 상태인가를 조사한다

### 3.1.4 DateFormatSymbols class

DateFormatSymbols 클래스는, 달, 요일, 타임 존 데이터 등, 지역 대응이 가능한 일자/시각 포맷 데이터를 캡슐화하기 위한 public 클래스입니다. DateFormat 과 SimpleDateFormat 은 DateFormatSymbols 를 사용해 이 정보를 캡슐화합니다.

일반적으로는, DateFormatSymbols 를 직접 사용하는 것은 하지 않습니다. 그 대신에, 가능한 한 DateFormat 클래스의 팩토리 메소드 getTimeInstance, getDateInstance, 또는 getDateTimeInstance 를 사용해 일자/시각 포매터를 작성해 주세요. 이러한 메소드를 사용하면(자), 그 포매터의 DateFormatSymbols 가 자동적으로 작성되므로, 새롭게 작성할 필요는 없습니다. 포매터가 작성되면(자), 그 포맷 패턴은 setPattern 메소드를 사용해 수정할 수가 있습니다.

특정의 로케일에 대한 특정의 포맷 패턴을 사용해 일자/시각을 작성하는 경우에는, 다음과 같이 합니다.

```
new SimpleDateFormat(aPattern, new DateFormatSymbols(aLocale)).
```

DateFormatSymbols 오브젝트는 복제 가능합니다. DateFormatSymbols 오브젝트를 취득하면(자), 그 일자/시각 포맷 데이터를 자유롭게 수정할 수가 있습니다. 예를 들어, 지역 대응된 일자/시각 포맷 패턴 문자를, 기억하기 쉬운 것에 옮겨놓을 수가 있습니다. 혹은, 대표적인 도시명을 기호의 것으로 바꿀 수가 있습니다.

추가 로케일에 대한 일자/시각 포맷 기능을 위해서(때문에), 새로운 DateFormatSymbols 서브 클래스를 추가해 SimpleDateFormat를 서포트할 수가 있습니다.

### 3.1.5 DecimalFormatSymbol class

DecimalFormatSymbol 클래스는, 수치를 포맷 할 경우에 DecimalFormat 로 필요한 기호 세트(소수점, 그룹화 단락 문자 등)를 나타냅니다. DecimalFormat 는, 그 자체를 위해서(때문에), 그 로케일 데이터로부터 DecimalFormatSymbols 의 인스턴스를 생성합니다. 이러한 기호의 어떤 것인지를 변경할 필요가 있는 경우는, DecimalFormat 로부터 DecimalFormatSymbols 오브젝트를 취득해, 수정할 수가 있습니다.

### 3.1.6 NumberFormat class

NumberFormat 클래스는, 모든 수치 포맷에 대한 abstract class 입니다. 이 클래스는, 수치의 포맷과 해석을 위한 인터페이스입니다. 한층 더 NumberFormat 는, 수치 포맷이 어느 로케일에 있어, 그 이름이 무엇으로 있을까를 알기 위한 메소드를 제공합니다.

NumberFormat를 사용하면, 임의의 로케일의 수치를 간단하게 포맷 하거나 해석하거나 할 수 있게 됩니다. 유저의 코드는, 10 진수의 소수점, 1000 의 단락지어, 사용하는 특정의 10 진수자에 대한 로케일의 룰 뿐만이 아니라, 숫자 포맷이 10 진수인가 여편가라고 하는 것 매음 도, 완전하게 독립하고 있습니다.

현행 로케일의 수치를 포맷 하는 경우는, 팩토리 클래스 메소드의 1 개를 사용합니다.

```
myString = NumberFormat.getInstance(). format(myNumber);
```

복수의 수치를 포맷 하는 경우에는, 그 포맷을 취득해, 그것을 여러 차례 사용하는 (분)편이 효율적입니다. 그러면, 로컬의 언어와 나라의 룰을 시스템이 여러 차례 취득할 필요가 없습니다.

```
NumberFormat nf = NumberFormat.getInstance();
for (int i = 0; i < a.length; ++i) {
    output.println(nf.format(myNumber[i]) + " ");
}
```

다른 로케일의 수치를 포맷 하는 경우는, getInstance 를 호출할 때 그것을 지정합니다.

```
NumberFormat nf = NumberFormat.getInstance(Locale.FRENCH);
```

게다가 NumberFormat 를 사용해 수치를 해석할 수가 있습니다.

```
myNumber = nf.parse(myString);
```

통상의 수치 포맷을 얻으려면, getInstance 나 getNumberInstance 를 사용합니다. 정수형 수치 포맷을 얻으려면, getIntegerInstance 를 사용합니다. 통화 수치 포맷을 얻으려면, getCurrencyInstance 를 사용합니다. 그리고, 퍼센트를 표시하는 포맷을 얻으려면, getPercentInstance 을 사용합니다. 이 포맷을 사용하면(자), 0.53 등의 소수점 이하는 53% 과 같이 표시됩니다.

게다가 setMinimumFractionDigits 등의 메소드를 사용하면, 수치의 표시를 제어할 수가 있습니다. 포맷이나 해석을 한층 더 제어하고 싶은 경우, 혹은 이러한 제어를 유저가 사용할 수 있도록 하고 싶은 경우는, 팩토리 메소드로부터 얻을 수 있는 NumberFormat 를 DecimalFormat 에 캐스트 할 수도 있습니다. 이것은 대부분의 로케일로 유효하지만, 유효하게 안 되는 로케일의 경우에 대비해, 이것은 try 블록으로 지정합니다.

NumberFormat 와 DecimalFormat 는, 포맷을 제어하는 것과 해석을 제어하는 것이 있도록 (듯이) 설계되고 있습니다. 다음에, 이러한 제어 메소드에 대해 상세하게 설명합니다.

setParseIntegerOnly 는 해석에만 작용합니다. 예를 들어, true 의 경우는, "3456.78" -> 3456 (해석 위치를 인덱스 6 의 직후에 그대로 남겨 둔다)이 됩니다. false 의 경우는, "3456.78" -> 3456.78 (해석 위치를 인덱스 8 의 직후에 그대로 남겨 둔다)이 됩니다. 이것은 포맷에 의존하지 않습니다. 소수점 이하의 자리수가 없는 경우에 소수점을 표시하지 않을 때는, setDecimalSeparatorAlwaysShown 를 사용합니다.

setDecimalSeparatorAlwaysShown 는 포맷에만 작용합니다. 다만, "#,##0. ##" (와)과 같은 패턴으로, 소수점 이하의 자리수가 없는 경우에 한정합니다. 예를 들어, true 의 경우, 3456.00 -> "3,456. " 됩니다. false 의 경우, 3456.00 -> "3456" 이 됩니다. 이것은 해석에 의존하지 않습니다. 소수점으로써 해석을 정지하는 경우는, setParseIntegerOnly 를 사용합니다.

게다가 parse 와 format 메소드의 형식을 ParsePosition 와 FieldPosition 와 함께 사용하면, 다음 일이 가능합니다.

- 캐릭터 라인의 각각의 부분을 단계적으로 해석한다
- 소수점등의 구역을 위치 맞춤 한다

예를 들어, 복수의 수치를 2 개의 방법으로 위치 맞춤 할 수가 있습니다.

1. 물건 스페이스 폰트를 사용해, 위치 맞춤에 스페이스를 들어갈 수 있는 경우에는, 포맷 호출로 FieldPosition 를 건너줄 수가 있습니다. 이 때, field = INTEGER\_FIELD 로 합니다. 출력으로는, getEndIndex 가, 정수의 마지막 문자와 소수점의 사이의 오프셋(offset)로 설정됩니다. 캐릭터 라인의 전에 (desiredSpaceCount - getEndIndex)의 스페이스를 추가해 주세요.
2. proportional font를 사용하는 경우에는, 스페이스를 추가하는 대신에, 캐릭터 라인의 폭을 그 처음부터 getEndIndex 까지 픽셀 단위로 측정합니다. 다음에, 텍스트를 그리기 전에, (desiredPixelWidth - widthToAlignmentPoint)만 펜을 이동합니다. 이것은, 소수점을 취급할 수 없습니다만, 마지막에 추가의 문자가 있는 경우에도 유효합니다. 예를 들어, -12 를 "(12)" 로 하도록(듯이), 부의 수에 괄호를 붙이는 경우입니다.

## 3.2 PACKAGE java.util.spi

CLASS 요약	
CurrencyNameProvider	Currency class 기능향상
LocaleNameProvider	Locale class 기능향상
LocaleServiceProvider	로케일서비스의 슈퍼클래스
TimeZoneNameProvider	TimeZone class 기능향상

### 3.2.1 Currency class

Currency 클래스는 통화를 나타냅니다. 통화는, ISO 4217 통화 코드로 식별됩니다. 클래스는, 지정된 통화의 Currency 인스턴스가 복수 존재하지 않게 설계되고 있습니다. 따라서, 클래스에는 public constructor이 없습니다. getInstance 메소드를 사용해, Currency 인스턴스를 취득합니다.

### 3.2.2 Locale class

Locale 클래스는, 특정의 지리적, 국가적, 또는 문화적 지역을 나타내기 위한 것입니다. 어느 조작으로 Locale를 필요로 하는 태스크가 있는 경우, 그 조작을 로케일에 의존하는 조작이라고 합니다. 이 경우, 정보는 Locale에 의해 유저에 맞추어 조정됩니다. 예를 들어, 수치를 표시하는 것은, 로케일에 의존하는 조작입니다. 이 수치는, 유저의 나라나 지역, 문화의 습관이나 규칙에 따라 포맷 할 필요가 있습니다.

Locale 오브젝트를 작성하려면, 이 클래스의 constructor을 사용합니다.

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

언어의 인수는, 유효한 ISO 언어 코드입니다. 이러한 코드는, ISO-639 로 정의되는 2 자리수의 소문자입니다. 나라의 인수는, 유효한 ISO 국 코드입니다. 이러한 코드는, ISO-3166 로 정의되는 2 자리수의 대문자입니다. 코드의 완전한 리스트는, <http://www.iso.ch/iso/en/prods-services/iso3166ma/02iso-3166-code-lists/list-en1.html> 등, 다수의 사이트에서 입수할 수 있습니다.

변형의 인수는 벤더 또는 브라우저에 고유의 코드입니다. 예를 들어, Windows 에는 WIN, Macintosh 에는 MAC, POSIX 에는 POSIX 를 사용합니다. 변형이 2 개 있는 경우에는, 그것들을 밑줄로 단락지어, 중요한 것을 최초로 지정합니다. 예를 들어, Traditional Spanish (에스파니아어)의 조합을 사용하는 경우는, 언어, 나라, 변형의 파라미터로서 「es」, 「ES」, 및 「Traditional\_WIN」를 지정해 로케일을 구축합니다.

Locale 오브젝트는, 어느 지역을 나타내는 단순한 식별자이므로, Locale 를 구축할 때, 그것이 타당한가 어떤가의 검사는 행해지지 않습니다. 구축하는 Locale 에 대해 특정의 자원이 있을지 어떨지를 조사하려면, 이러한 자원의 조회를 실시할 필요가 있습니다. 예를 들어, NumberFormat 에, 그 getAvailableLocales 메소드를 사용해, 서포트하는 로케일을 조회합니다.

Locale 클래스에는, 몇개의 편리한 정수가 있으므로, 이것들을 사용해, 자주(잘) 사용되는 로케일의 Locale 오브젝트를 작성할 수가 있습니다. 예를 들어, 다음의 예는, 미국의 Locale 오브젝트를 작성하는 예입니다.

```
Locale.US
```

Locale 를 작성한 뒤는, 이 오브젝트에 그것 자신의 정보를 조회할 수가 있습니다. ISO 국 코드를 얻으려면 getCountry 를, ISO 언어 코드를 얻으려면 getLanguage 를 각각 사용합니다. getDisplayCountry 를 사용하면, 적합한 국명이 유저에게 표시됩니다. 똑같이, getDisplayLanguage 를 사용하면, 적합한 언어의 이름이 유저에게 표시됩니다. 흥미로운 것에, getDisplayXXX 메소드 자체도 로케일에 의존합니다. 이것에는 2 개의 버전이 있어, 한편은 디폴트의 로케일을 사용해, 한편은 인수로서 지정된 로케일을 사용합니다.

Java 2 플랫폼에는, 로케일에 의존하는 조작을 실시하는 몇개의 클래스가 있습니다. 예를 들어, NumberFormat 클래스는, 수치나 통화, 퍼센트를 로케일에 의존하는 형태로 포맷 합니다. NumberFormat 등의 클래스에는, 이 형태의 오브젝트를 작성하기 위한 간이 메소드가 몇개인가 있습니다. 예를 들어, NumberFormat 클래스에는, 디폴트의 NumberFormat 오브젝트를 작성하기 위해서 다음의 3 개의 메소드가 있습니다.

```
NumberFormat.getInstance()
NumberFormat.getCurrencyInstance()
NumberFormat.getPercentInstance()
```

이러한 메소드에는 2 개의 변형이 있습니다. 명시적인 로케일을 가지는 것과 가지지 않는 것입니다. 후자의 경우는 디폴트 로케일이 사용됩니다.

```
NumberFormat.getInstance(myLocale)
NumberFormat.getCurrencyInstance(myLocale)
NumberFormat.getPercentInstance(myLocale)
```

Locale 는, 어느 오브젝트 (NumberFormat)를 사용하는지를 지정하기 위한 기구입니다. 로케일은, 오브젝트를 지정하기 위한 단순한 기구이며, 오브젝트 자체의 컨테이너가 아닙니다.

### 3.2.3 TimeZone class

TimeZone 클래스는, 타임 존 오프셋(offset)를 나타냅니다. 또, 서머타임을 인식합니다.

통상은 getDefault를 사용해 TimeZone을 취득합니다. 이 메소드는, 프로그램을 실행하고 있는 장소의 타임 존에 근거한 TimeZone 를 작성합니다. 예를 들어, 일본에서 실행되고 있는 프로그램의 경우, getDefault는 일본 표준시를 기본으로 TimeZone 오브젝트를 작성합니다.

타임존 ID를 지정해 `getTimeZone` 를 사용해 `TimeZone` 를 취득할 수도 있습니다. 예를 들어, 태평양 표준시의 타임존 ID 는, 「America/Los\_Angeles」입니다. 따라서, 다음과 같이 입력해 PST `TimeZone` 오브젝트를 취득할 수 있습니다.

```
TimeZone tz = TimeZone.getTimeZone("America/Los_Angeles");
```

`getAvailableIDs` 메소드를 사용해, 서포트 떠날 수 있어 모든 타임존 ID를 조사할 수가 있습니다. 서포트되고 있는 ID 중에서, 목적의 `TimeZone` 를 선택할 수 있습니다. 서포트되고 있는 ID 의 어느 쪽인가에 따라 필요한 타임존이 나타내지지 않은 경우는, 커스텀 타임존 ID 를 지정해, `TimeZone`을 생성할 수가 있습니다. 커스텀 타임존 ID 의 구문은 다음과 같습니다.

*CustomID:*

GMT 부호 시 : 분

GMT 부호 시 분

GMT 부호 시

부호: + - 의 어느 쪽인지

때: 숫자 숫자 숫자

분: 숫자 숫자

숫자: 0 1 2 3 4 5 6 7 8 9

`Hours` 는 0 ~ 23 으로, `Minutes` 는 00 ~ 59 를 지정할 필요가 있습니다. 예를 들어, 「GMT+10」(와)과 「GMT+0010」은, 각각, GMT 의 10 시간전과 10 분전이 됩니다.

형식은 로케일에 의존하지 않습니다. 숫자는 Unicode 표준의 Basic Latin 블록으로부터 취득할 필요가 있습니다. 서머타임에의 이행 스케줄은, 커스텀 타임 존 ID 로 지정할 수 없습니다. 지정된 캐릭터 라인이 구문과 일치하지 않는 경우는, 「GMT」를 사용합니다.

`TimeZone` 를 작성할 때는, 지정된 커스텀 타임 존 ID 는, 다음의 구문으로 규정됩니다.

*NormalizedCustomID:*

GMT 부호 TwoDigitHours : 분

부호: + - 의 어느 쪽인지

TwoDigitHours: 숫자 숫자

분: 숫자 숫자

숫자: 0 1 2 3 4 5 6 7 8 9

예를 들어, `TimeZone.getTimeZone("GMT-8").getID()`는 GMT-08:00 를 돌려줍니다.

## 4. 수정된 패키지

### 4.1 PACKAGE java.applet

- `getLocale()`

## 4.2 PACKAGE java.awt

### 4.2.1 Desktop Class

- `browse()`, `edit()`, `getDesktop()`, `isDesktopSupported()`, `isSupported()`, `mail`, `open()`, `print()`

### 4.2.2 GridBagLayout Class

- `addLayoutComponent()`, `adjustForGravity()`, `arrangeGrid()`, `getConstraints()`, `getLayoutAlignmentX()`, `getLayoutAlignmentY()`, `getLayoutDimensions()`, `getLayoutInfo()`, `getLayoutInfo()`, `getLayoutOrigin()`, `getLayoutWeights()`, `GetMinSize()`, `invalidateLayout()`, `LayoutContainer()`, `location()`, `lookupConstraints()`, `maximumLayoutSize()`, `minumLayoutSize()`, `preferrrdLayoutSize()`, `removeLayoutComponebt()`, `setConstraints()`, `toString()`

### 4.2.3 SplashScreen Class

- `close()`, `getBounds()`, `getGraphics()`, `getImageURL()`, `getSize()`, `getSplashScreen()`, `isVisible()`, `setImageURL()`, `update()`

### 4.2.4 SystemTray Class

- `add()`, `addPropertyChangeListeners()`, `getSystemTray()`, `getTrayIconSize()`, `isSupported()`, `remove()`, `removePropertyChangeListeners()`

### 4.2.5 TrayIcon Class

- `addActionListener()`, `addMouseListener()`, `addMouseMotionListener()`, `displayMessage()`, `getActionCommand()`, `getActionListeners()`, `setImage()`, `getMouseListeners()`, `getMouseMotionListener()`, `getPopupMenu()`, `getSize()`, `getToolTip()`, `isImageAutoSize()`, `removeActionListener()`, `removeMouseMotionListener()`, `removeMouseListener()`, `setActionCommand()`, `setImage()`, `setImageAutoSize()`, `setPopupMenu()`, `setToolTip()`

### 4.2.6 Component.BaselineResizeBehavior

- `valueOf()`, `values()`



### 4.3 PACKAGE java.awt.event

#### 4.3.1 MouseEvent Class

- getButton(), getClickCount(), getLocationOnScreen(), getMouseModifierText(), getPoint(), getX(), getXOnScreen(), getY(), getYOnScreen(), isPopupTrigger(), paramString(), translatePoint()

#### 4.3.2 MouseWheelEvent Class

- getScrollAmount(), getScrollType(), getUnitsToScroll(), getWheelRotation(), paramString()

### 4.4 PACKAGE java.awt.font

#### 4.4.1 LayoutPath Class

- pathToPoint(), pointToPath()

#### 4.4.2 FontRenderContext Class

- equals(), getAntiAliasingHint(), getFractionalMetricsHint(), getTransform(), getTransformType(), hashCode(), isAntiAliased(), isTransformed(), usesFractionalMetrics()

#### 4.4.3 GraphicAttribute Class

- draw(), getAdvance(), getAlignment(), getAscent(), getBounds(), getDescent(), getJustificationInfo(), getOutline()

#### 4.4.4 NumericShaper Class

- equals(), getContextualShaper(), getRanges(), getShaper(), hashCode(), isContextual(), shape(), toString()

#### 4.4.5 ShapeGraphicAttribute Class

- draw, equals, getAdvance, getAscent(), getBounds(), getDescent(), getOutline(), hashCode()

### 4.4.6 TextAttribute Class

- getMissingGlyphCode(), hasLayoutAttributes()

### 4.4.7 TextLayout Class

- clone(), draw(), equals(), getAdvance(), getAscent(), getBaseline(), getBaselineOffsets(), getBounds(), getCareInfo(), get-caretShape(), get-caretShapes(), getCharacterCount(), getCharacterLevel(), getDescent(), getJustifiedLayout(), getLayoutPath(), getLeading(), getLogicalHighlightShape(), getLogicalRangesForVisualSelection(), getNextLeftHit(), getNextRightHit(), getOutline(), getPixelBounds(), getVisibleAdvance(), getVisibleHighlightShape(), getVisualOtherHit(), handleJustify(), hashCode(), hitTestChar(), hitToPoint(), isLeftToRight(), isVertical(), toString()

### 4.4.8 TransformAttribute Class

- equals(), getTransform(), isIdentity()

### 4.5 PACKAGE java.awt.geom

#### 4.5.1 AffineTransform Class

- getQuadrantRotateInstance(), getRotateInstance(), invert(), QuadrantRotate(), rotate(), setToQuadrantRotation(), setToRotation()

#### 4.5.2 Arc2D.Double Class

- equals(), hashCode()

#### 4.5.3 Arc2D.Float Class

- equals(), hashCode()

#### 4.5.4 Arc2D Class

- equals(), hashCode()

#### 4.5.5 Area Class

- area(), subtract(), intersect(), exclusiveOr(), transform(), createTransformedArea(), contains(), intersects(),

#### 4.5.6 Ellipse2D.Float Class

- equals(), hashCode()

#### 4.5.7 Ellipse2D Class

- equals(), hashCode()

#### 4.5.8 RoundRectangle2D.Double Class

- equals(), hashCode()

#### 4.5.9 RoundRectangle2D.Float Class

- equals(), hashCode()

#### 4.5.10 RoundRectangle2D Class

- equals(), hashCode()

### 4.6 PACKAGE java.awt.im

#### 4.6.1 InputContext Class

- dispose()

### 4.7 PACKAGE java.awt.im.spi

#### 4.7.1 InputMethod Interface

- dispose()

### 4.8 PACKAGE java.awt.image

#### 4.8.1 BufferStrategy Class

- dispose(), getCapabilities(), getDrawGraphics()

#### 4.8.2 BufferedImage Class

- flush(), getCapabilities(), getScaledInstance()

#### 4.8.3 IndexColorModel Class

- getGreen(), getBlue(), getAlpha(), getRGB(), convertToIntDiscrete()

#### 4.8.4 SampleModel Class

- setSample()

### 4.9 PACKAGE java.awt.print

#### 4.9.1 PrinterJob Class

- getPageFormat(), getPrintService(), getPrinterJob()

### 4.10 PACKAGE java.beans

#### 4.10.1 EventHandler Class

- create()

#### 4.10.2 PropertyChangeEvent Class

- getNewValue()

#### 4.10.3 PropertyChangeSupport Class

- firePropertyChange(), fireIndexedPropertyChange(),

#### 4.10.4 PropertyEditorSupport Class

- isPaintable()

#### 4.10.5 VetoableChangeSupport Class

- fireVetoableChange()

### 4.11 PACKAGE java.beans.beancontext

#### 4.11.1 BeanContextSupport Class

- remove(), removeAll(), retainAll(),

### 4.12 PACKAGE java.io

#### 4.12.1 Console Class

- flush(), format(), printf(), reader(), readLine(), readPassword(), writer()

#### 4.12.2 IOException Class

- IOException()

#### 4.12.3 BufferedInputStream Class

- available()

#### 4.12.4 BufferedReader Class

- close(), mark(), markSupported(), read(), readLine(), ready(), reset(), skip()

#### 4.12.5 BufferedWriter Class

- close(), flush(), newLine(), write()

#### 4.12.6 ByteArrayInputStream Class

- available(), read()

#### 4.12.7 CharArrayReader Class

- close(), mark(), markSupported(), read(), ready(), reset(), skip()

#### 4.12.8 DataInputStream Class

- read()

#### 4.12.9 File Class

- canExecute(), getFreeSpace(), getTotalSpace(), getUsableSpace(), setExecutable(), setReadable(), setWritable(), toURL()

#### 4.12.10 FileInputStream Class

- available(), read()

#### 4.12.11 FilePermission Class

- FilePermission(), implies(), equals()

#### 4.12.12 FilterInputStream Class

- available(), read()

#### 4.12.13 FilterReader Class

- close(), mark(), markSupported(), read(), ready(), reset(), skip(), FilterReader()

#### 4.12.14 FilterWriter Class

- close(), flush(), write()

#### 4.12.15 InputStream Class

- available(), read()

#### 4.12.16 InputStreamReader Class

- close(), getEncoding(), read(), ready()

#### 4.12.17 LineNumberReader Class

- close(), getEncoding(), read(), ready()

#### 4.12.18 ObjectInputStream Class

- skipBytes(), readUnshared(), resolveClass()

#### 4.12.19 ObjectOutputStream.PutField Class

- put()

#### 4.12.20 ObjectOutputStream Class

- getName(), lookupAny()

#### 4.12.21 OutputStreamWriter Class

- close(), flush(), getEncoding(), write()

#### 4.12.22 PipedInputStream Class

- read(), available()

#### 4.12.23 PrintStream Class

- checkError(), clearError(), close(), flush(), print(), println(), setError(), write()

#### 4.12.24 PushbackInputStream Class

- available(), read(),

#### 4.12.25 PushbackReader Class

- PushbackReader(), read(), unread(), ready(), mark(), reset(), markSupported(), close(), skip()

#### 4.12.26 Reader Class

- read(), skip(), ready(), markSupported(), mark(), reset(), close()

#### 4.12.27 SequenceInputStream Class

- available(), read()

#### 4.12.28 Serializable Interface

- Serializable

#### 4.12.28 StringReader Class

- close(), mark(), markSupported(), read(), reset(),

#### 4.12.29 StringWriter Class

- write()

#### 4.12.30 Writer Class

- close(), flush(), write()

### 4.13 PACKAGE java.lang

#### 4.13.1 Boolean Class

- valueOf(), parseBoolean(),

#### 4.13.2 Class Class

- getMethod(), getAnnotation(), isAnnotationPresent()

#### 4.13.3 ClassLoader Class

- `setPackageAssertionStatus()`

#### 4.13.4 Comparable Interface

- `compareTo()`

#### 4.13.5 Double Class

- `MIN_NORMAL`, `MAX_EXPONENT`, `MIN_EXPONENT`

#### 4.13.6 Enum Class

- `finalize()`, `compareTo()`

#### 4.13.7 Float Class

- `MIN_NORMAL`, `MAX_EXPONENT`, `MIN_EXPONENT`

#### 4.13.8 InterruptedException Class

- `InterruptedException`

#### 4.13.9 Math Class

- `acos()`, `asin()`, `atan()`, `atan2()`, `copySign()`, `getExponent()`, `nextAfter()`, `nextUp()`, `scalb()`

#### 4.13.10 Object Class

- `clone()`, `equals()`, `finalize()`, `getClass()`, `hashCode()`, `notify()`, `toString()`, `wait()`

#### 4.13.11 Package Class

- `getAnnotation()`, `isAnnotationPresent()`

#### 4.13.12 Runtime Class

- `load()`

#### 4.13.13 StrictMath Class

- `acos()`, `asin()`, `atan()`, `atan2()`, `copySign()`, `getExponent()`, `nextAfter()`, `nextUp()`, `scalb()`

#### 4.13.14 String Class

- `isEmpty()`, `startsWith()`

#### 4.13.15 System Class

- `console()`, `getenv()`

#### 4.13.16 Thread Class

- `dumpStack()`, `sleep()`, `dumpStack()`, `join()`, `isInterrupted()`

#### 4.13.17 ThreadLocal Class

- `initialValue()`, `remove()`, `get()`, `set()`

### 4.14 PACKAGE `java.lang.annotation`

#### 4.14.1 ElementType Enum

- `field`

#### 4.14.2 RetentionPolicy Enum

- `finalize`

### 4.15 PACKAGE `java.lang.instrument`

#### 4.15.1 ClassLoaderConfigurationError Class

- `fillInStackTrace()`, `getCause()`, `getLocalizedMessage()`, `getMessage()`, `getStackTrace()`, `initCause()`, `printStackTrace()`, `printStackTrace()`, `printStackTrace()`, `setStackTrace()`, `toString()`

#### 4.15.2 Instrumentation Interface

- addTransformer(), appendToBootstrapClassLoaderSearch(), appendToSystemClassLoaderSearch(), isNativeMethodPrefixSupported(), isRetransformClassesSupported(), redefineClasses(), retransformClasses(), setNativeMethodPrefix()

### 4.16 PACKAGE java.lang.management

#### 4.16.1 LockInfo Class

- form(), getClassName(), getIdentityGashCode(), toString()

#### 4.16.2 MonitorInfo Class

- form(), getLockedStackDepth(), getLockedStackFrame()

#### 4.16.3 ManagementFactory Class

- getPlatformMBeanServer(), newPlatformMXBeanProxy()

#### 4.16.4 MemoryNotificationInfo Class

- form()

#### 4.16.5 ThreadInfo Class

- form(), getLockInfo(), getLockName(), getLockOwnerId(), getLockOwnerName(),

#### 4.16.6 ThreadMXBean Interface

- findDeadlockedThreads(), getAllLocakedMonitors(), getAllLockedSynchronizers(), isObjectMonitorUsageSupported(), isSynchronizerUsageSupported()

### 4.17 PACKAGE java.lang.reflect

#### 4.17.1 AccessibleObject Class

- getAnnotation()

#### 4.17.2 AnnotatedElement Interface

- getAnnotation()

#### 4.17.3 Constructor Class

- getAnnotation(), getDeclaredAnnotations()

#### 4.17.4 Field Class

- getGenericType(), getAnnotation(), getDeclaredAnnotations()

#### 4.17.5 Method Class

- getAnnotation(), getDeclaredAnnotations()

### 4.18 PACKAGE java.math

#### 4.18.1 BigDecimal Class

- negate(), longValue(), byteValueExtract()

### 4.19 PACKAGE java.net

#### 4.19.1 CookieManager Class

- get(), getCookieStore(), put(), setCookiePolicy()

#### 4.19.2 CookiePolicy Interface

- shouldAccept()

#### 4.19.3 CookieStore Interface

- add(), get(), getCookies(), getURLs(), remove(), removeAll()

#### 4.19.4 HttpCookie Class

- clone(), domainMatches(), equals(), getComment(), getCommentURL(), getDiscard(), getDomain(), getMaxAge(), getName(), getPath(), getPortList(), getSecure(),

getValue(), getVersion(), hasExpired(), hashCode(), parse(), setComment(),  
setCommentURL(), setDiscard(), setDomain(), setMaxAge(), setPath(),  
setPortLisr(), setSecure(), setValue(), setVersion(), toString()

#### 4.19.5 IDN Class

- toASCII, toUnicode

#### 4.19.6 InterfaceAddress Class

- equals(), getAddress(), getNetworkPrefixLength(), hashCode(), toString()

#### 4.19.7 NetworkInterface Class

- getHardwareAddress(), getInterfaceAddresses(), getMTU(), getParent(),  
getSubInterface(), isLoopBack(), isPointToPoint(), isUp(), isVirtual(),  
supportsMulticast()

#### 4.19.8 Proxy Class

- toString()

#### 4.19.10 Socket Class

- setTrafficClass(),

#### 4.19.11 URI Class

- getHost(), getRawHost(), toASCIIString(), toRIString(), toString()

### 4.20 PACKAGE java.nio

#### 4.20.1 Buffer Class

- array(), arrayOffset(), hasArray(), isDirect()

#### 4.20.2 ByteBuffer Class

- hasArray(), array(), arrayOffset(), compact(), isDirect(), compareTo()

#### 4.20.3 CharBuffer Class

- wrap(), hasArray(), array(), arrayOffset(), isDirect(), compareTo(), subSequence()

#### 4.20.4 DoubleBuffer Class

- hashArray(), array(), arrayOffset(), isDirect(), compareTo()

#### 4.20.5 FloatBuffer Class

- hashArray(), array(), arrayOffset(), isDirect(), compareTo()

#### 4.20.6 IntBuffer Class

- hashArray(), array(), arrayOffset(), isDirect(), compareTo()

#### 4.20.7 LongBuffer Class

- hashArray(), array(), arrayOffset(), isDirect(), compareTo()

#### 4.20.7 ShortBuffer Class

- hashArray(), array(), arrayOffset(), isDirect(), compareTo()

### 4.21 PACKAGE java.nio.channels

#### 4.21.1 FileChannel Class

- lock(), tryLock()

### 4.22 PACKAGE java.nio.charset

#### 4.22.1 Charset Class

- isSupported(), forName(), contains()

#### 4.22.2 CharsetDecoder Class

- flush()

#### 4.22.3 CharsetEncoder Class

- flush()

### 4.23 PACKAGE java.rmi

#### 4.23.1 MarshalledObject Class

- get(), MarshalledObject()

### 4.24 PACKAGE java.rmi.activation

#### 4.24.1 Activatable Class

- exportObject()

#### 4.24.2 ActivationDesc Class

- MarshalledObject()

#### 4.24.3 ActivationGroup Class

- activeObject()

#### 4.24.4 ActivationGroupDesc Class

- MarshalledObject()

#### 4.24.5 ActivationMonitor Class

- activeObject()

### 4.25 PACKAGE java.security

#### 4.25.1 PolicySpi Class

- engineGetPermissions(), engineImplies(), engineRefresh()

#### 4.25.2 URIPParameter Class

- getURI()

#### 4.25.3 AccessControlContext Class

- checkPermission()

#### 4.25.4 AccessController Class

- checkPermission(), doPrivilegedWithCombiner(), doPrivileged()

#### 4.25.5 AlgorithmParameterGenerator Class

- getInstance()

#### 4.25.6 AlgorithmParameters Class

- getInstance()

#### 4.25.7 KeyFactory Class

- getInstance()

#### 4.25.8 KeyPairGenerator Class

- getInstance()

#### 4.25.9 KeyStore Class

- engineLoad(), engineGetEntry()

#### 4.25.10 MessageDigest Class

- getInstance()

#### 4.25.10 Permissions Class

- implies()

#### 4.25.11 Policy Class



- getInstance(), getParameters(), getPermissions(), getProvider(), getType()

#### 4.25.11 Provider Class

- putAll(), entrySet(), values()

#### 4.25.12 SecureRandom Class

- getInstance(), nextBytes()

#### 4.25.13 SecureRandomSpi Class

- engineNextBytes()

#### 4.25.14 Security Class

- insertProviderAt()

#### 4.25.15 Signature Class

- getInstance()

### 4.26 PACKAGE java.security.cert

#### 4.26.1 CertPathBuilder Class

- getInstance(), getDefaultType()

#### 4.26.2 CertPathValidator Class

- getInstance(), getDefaultType()

#### 4.26.3 CertStore Class

- getInstance(), getDefaultType()

#### 4.26.3 CertificateFactory Class

- getInstance()

### 4.27 PACKAGE java.security.interfaces

#### 4.27.1 DSAKeyPairGenerator Interface

- initialize()

### 4.28 PACKAGE java.security.spec

#### 4.28.1 EncodedKeySpec Class

- EncodedKeySpec()

### 4.29 PACKAGE java.text

#### 4.29.1 Normalizer.Form Enum

- valueOf(), values()

#### 4.29.2 Normalizer Class

- isNormalized(), normalize()

#### 4.29.3 AttributedString Class

- iAttributedString(), addAttribute(), getIterator()

#### 4.29.4 BreakIterator Class

- getCharacterInstance(), getLineInstance(), getSentenceInstance(), getWordInstance(),

#### 4.29.5 CollationKey Class

- compareTo(), toByteArray()

#### 4.29.6 DateFormat Class

- getAvailableLocales()

#### 4.29.7 DateFormatSymbols Class

- `getAvailableLocales()`, `getInstance()`, `getZoneStrings()`, `setZoneStrings()`

#### 4.29.8 DecimalFormat Class

- `getDecimalFormatSymbols()`, `getRoundingMode()`, `setRoundingMode()`

#### 4.29.9 DecimalFormatSymbols Class

- `getAvailableLocales()`, `getExponentSeparator()`, `getInstance()`, `setExponentSeparator()`

#### 4.29.10 MessageFormat Class

- `setLocal()`, `parse()`

#### 4.29.11 NumberFormat Class

- `getRoundingMode()`, `setRoundingMode()`

### 4.30 PACKAGE java.util

#### 4.30.1 AbstractMap.SimpleEntry Class

- `equals()`, `getKey()`, `getValue()`, `hashCode()`, `setValue()`, `toString()`

#### 4.30.2 AbstractMap.SimpleImmutableEntry Class

- `equals()`, `getKey()`, `getValue()`, `hashCode()`, `setValue()`, `toString()`

#### 4.30.3 ArrayDeque Class

- `add()`, `addFirst()`, `addLast()`, `clear()`, `clone()`, `contains()`, `descendingIterator()`, `element()`, `getFirst()`, `getLast()`, `isEmpty()`, `iterator()`, `offer()`, `offerFirst()`, `offerLast()`, `peek()`, `peekFirst()`, `peekLast()`, `poll()`, `pollFirst()`, `pollLast()`, `pop()`, `push()`, `remove()`, `removeFirst()`, `removeFirstOccurrence()`, `removeLast()`, `removeLastOccurrence()`, `size()`, `toArray()`

#### 4.30.4 Deque Interface

- `add()`, `addFirst()`, `addLast()`, `contains()`, `descendingIterator()`, `element()`, `getFirst()`, `getLast()`, `iterator()`, `offer()`, `offerFirst()`, `offerLast()`, `peek()`, `peekFirst()`, `peekLast()`

- `poll()`, `pollFirst()`, `pollLast()`, `pop()`, `push()`, `remove()`, `removeFirst()`, `removeFirstOccurrence()`, `removeLast()`, `removeLastOccurrence()`, `size()`

#### 4.30.5 NavigableMap Interface

- `ceilingEntry()`, `ceilingKey()`, `descendingEntrySet()`, `descendingKeySet()`, `firstEntry()`, `floorEntry()`, `floorKey()`, `lastEntry()`, `lowerEntry()`, `lowerKey()`, `navigableHeadMap()`, `navigableSubMap()`, `pollFirstEntry()`, `pollLastEntry()`

#### 4.30.6 NavigableSet Interface

- `ceiling()`, `descendingIterator()`, `floor()`, `higher()`, `iterator()`, `lower()`, `navigableHeadSet()`, `navigableSubSet()`, `navigableTailSet()`, `pollFirst()`, `pollLast()`

#### 4.30.7 ResourceBundle.Control Class

- `getCandidateLocales()`, `getControl()`, `getFallbackLocale()`, `getFormats()`, `getNoFallbackControl()`, `getTimeToLive()`, `needsReload()`, `newBundle()`, `toBundleName()`, `toResourceName()`

#### 4.30.8 Service Class

- `iterator()`, `lookup()`, `lookupInstalled()`, `reload()`, `toString()`

#### 4.30.9 ServiceConfigurationError Class

- `fillInStackTrace()`, `getCause()`, `getLocalizedMessage()`, `getMessage()`, `getStackTrace()`, `initCause()`, `printStackTrace()`, `printStackTrace()`, `printStackTrace()`, `setStackTrace()`, `toString()`

#### 4.30.10 AbstractCollection Class

- `removeAll()`, `iterator()`, `size()`, `isEmpty()`, `contains()`, `toArray()`, `add()`, `remove()`, `containsAll()`, `addAll()`, `retainAll()`, `clear()`, `toString()`

#### 4.30.11 AbstractList Class

- `add()`, `get()`, `set()`, `add()`, `remove()`, `indexOf()`, `lastIndexOf()`, `clear()`, `addAll()`, `iterator()`, `listIterator()`, `subList()`, `equals()`, `hashCode()`, `removeRange()`

#### 4.30.12 AbstractQueue Class

- add(), remove(), element(), clear(), addAll()

#### 4.30.13 AbstractSequentialList Class

- get(), set(), add(), remove(), addAll(), iterator(), listIterator()

#### 4.30.14 AbstractSet Class

- equals(), hashCode(), removeAll()

#### 4.30.15 ArrayList Class

- ensureCapacity(), size(), isEmpty(), contains(), indexOf(), lastIndexOf(), clone(), toArray(), get(), set(), add(), remove(), addAll(), removeRange()

#### 4.30.16 Arrays Class

- binarySearch(), fill(), copyOf(), copyOfRange(), asList(), deepEquals()

#### 4.30.17 BitSet Class

- flip(), set(), clear(), nextSetBit(), hashCode(), clone()

#### 4.30.18 Calendar Class

- getDisplayNames()

#### 4.30.19 Collection Interface

- contains(), toArray(), add(), remove(), containsAll(), addAll(), removeAll(), retainAll(), clear(), equals(),

#### 4.30.20 Collections Class

- asLifoQueue(), newSetFromMap()

#### 4.30.21 Comparator Interface

- equals()

#### 4.30.22 EnumMap Class

- get()

#### 4.30.23 GregorianCalendar Class

- getActualMaximum()

#### 4.30.24 HashMap Class

- clear(), entrySet(), get(), keySet(), putAll(), removeAll(), values()

#### 4.30.25 HashSet Class

- hashSet(), size(), isEmpty(), contains(), add(), remove(), clear(), clone()

#### 4.30.26 HashMap Class

- clear(), entrySet(), get(), keySet(), putAll(), removeAll(), values()

#### 4.30.27 HashMap Class

- clear(), entrySet(), get(), keySet(), putAll(), removeAll(), values()

#### 4.30.28 Hashtable Class

- containsValue(), entrySet(), get(), keySet(), putAll(), values()

#### 4.30.29 IdentityHashMap Class

- clear(), entrySet(), get(), putAll(), values()

#### 4.30.30 LinkedHashMap Class

- clear(), get()

#### 4.30.31 LinkedList Class

- addFirst(), addLast(), descendingIterator(), indexOf(), offerFirst(), offerLast(), peekFirst(), peekLast(), poll(), pollFirst(), pollLast(), pop(), push(), remove(), removeFirstOccurrence(), removeLastOccurrence(), toArray()

#### 4.30.32 List Interface

- indexOf(), lastIndexOf(), remove(), removeAll(), toArray()

#### 4.30.33 ListIterator Class

- set(), add()

#### 4.30.34 ListResourceBundle Class

- getContents(), getKeys(), handleKeySet()

#### 4.30.35 Locale Class

- getAvailableLocales(), getLanguage(), getISO3Language(), getISO2Country(), getDisplayCountry(), equals()

#### 4.30.36 Map.Entry Class

- getKey(), getValue(), setValue(), equals(), hashCode()

#### 4.30.37 Map Interface

- clear(), entrySet(), get(), keySet(), remove()

#### 4.30.38 PriorityQueue Class

- add(), comparator(), contains(), peek(), poll()

#### 4.30.39 Properties Class

- stringPropertyNames(), propertyNames()

#### 4.30.40 PropertyResourceBundle Class

- getKeys(), handleKeySet()

#### 4.30.41 Queue Interface

- add(), offer(), peek(), poll()

#### 4.30.42 Random Class

- nextGaussian()

#### 4.30.43 ResourceBundle Class

- clearCache(), containsKey(), getBundle(), getKeys(), getLocale(), getObject(), getString(), getStringArray(), handleGetObject(), handleKeySet(), keySet(), setParent()

#### 4.30.44 Scanner Class

- reset(), useDelimiter(), useLocale(), useRadix()

#### 4.30.45 ResourceBundle Class

- clearCache(), containsKey(), getBundle(), getKeys(), getLocale(), getObject(),

#### 4.30.46 Set Interface

- size(), isEmpty(), contains(), iterator(), toArray(), add(), remove(), containsAll(), addAll(), retainAll(), removeAll(), clear(), equals(), hashCode()

#### 4.30.47 SortedMap Interface

- comparator(), entrySet(), firstKey(), headMap(), keySet(), lastKey(), subMap(), tailMap(), values()

#### 4.30.48 SortedSet Interface

- comparator(), first(), headSet(), last(), subSet(), tailSet()

#### 4.30.49 TimeZone Class

- getAvailableIDs(), getDisplayName()

#### 4.30.50 TreeMap Class

- ceilingEntry(), ceilingKey(), clear(), comparator(), descendingEntrySet(), descendingKeySet(), entrySet(), firstEntry(), firstKey(), floorEntry(), floorKey(), get(), headMap(), higherEntry(), higherKey(), keySet(), lastEntry(), lastKey(), lowerEntry(), lowerKey(), navigableHeadMap(), navigableSubMap(), navigableTailMap(), pollFirstEntry(), pollLastEntry(), subMap(), tailMap(), values()

#### 4.30.51 TreeSet Class

- ceiling(), comparator(), descendingIterator(), first(), floor(), headSet(), higher(), iterator(), lower(), navigableHeadSet(), navigableSubSet(), navigableTailSet(), pollFirst(), pollLast(), subSet(), tailSet()

#### 4.30.52 Vector Class

- contains(), indexOf(), lastIndexOf()

#### 4.30.53 WeakHashMap Class

- clear(), entrySet(), get(), keySet(), putAll(), remove(), values()

### 4.31 PACKAGE java.util.concurrent

#### 4.31.1 BlockingDeque Interface

- add(), addFirst(), addLast(), contains(), element(), iterator(), offer(), offerFirst(), offerLast(), peek(), poll(), pollFirst(), pollLast(), push(), put(), putFirst(), putLast(), remove(), removeFirstOccurrence(), size(), take(), takeFirst(), takeLast()

#### 4.31.2 ConcurrentNavigableMap Interface

- headMap(), navigableHeadMap(), navigableSubMap(), navigableTailMap(), subMap(), tailMap()

#### 4.31.3 ConcurrentSkipListMap Class

- ceilingEntry(), ceilingKey(), clear(), clone(), comparator(), containsKey(), containsValue(), descendingEntrySet(), descendingKeySet(), entrySet(), equals(),

firstEntry(), firstKey(), floorEntry(), floorKey(), get(), headMap(), higherEntry(), higherKey(), isEmpty(), keySet(), lastEntry(), lastKey(), lowerEntry(), lowerKey(), navigableHeadMap(), navigableSubMap(), navigableTailMap(), pollFirstEntry(), pollLastEntry(), put(), putIfAbsent(), remove(), replace(), size(), subMap(), tailMap(), values()

#### 4.31.4 ConcurrentSkipListSet Class

- add(), ceiling(), clear(), clone(), comparator(), contains(), descendingIterator(), equals(), first(), floor(), headSet(), higher(), isEmpty(), iterator(), last(), lower(), navigableHeadSet(), navigableSubSet(), navigableTailSet(), pollFirst(), pollLast(), remove(), removal(), size(), subSet(), tailSet()

#### 4.31.5 LinkedBlockingDeque Class

- add(), addFirst(), addLast(), clear(), contains(), descendingIterator(), drainTo(), element(), getFirst(), getLast(), iterator(), offer(), offerFirst(), offerLast(), peek(), peekFirst(), peekLast(), poll(), pollFirst(), pollLast(), pop(), push(), put(), putFirst(), putLast(), remainingCapacity(), remove(), removeFirst(), removeFirstOccurrence(), removeLast(), removeLastOccurrence(), size(), take(), takeFirst(), takeLast(), toArray(), toString()

#### 4.31.6 RunnableFuture Interface

- run()

#### 4.31.7 RunnableScheduledFuture Interface

- isPeriodic()

#### 4.31.8 AbstractExecutorService Class

- invokeAll(), invokeAny(), newTaskFor(), submit()

#### 4.31.9 ArrayBlockingQueue Class

- add(), contains(), drainTo(), offer(), peek(), poll(), put(), remainingCapacity(), take(), toArray(), toString()

#### 4.31.10 BlockingQueue Interface

- add(), contains(), drainTo(), offer(), poll(), put(), remainingCapacity(), remove(), take()

#### 4.31.11 CompletionService Interface

- submit(), take(), poll()

#### 4.31.12 ConcurrentHashMap Class

- clear(), entrySet(), get(), keySet(), remove(), replace(), values()

#### 4.31.13 ConcurrentLinkedQueue Class

- add(), contains(), isEmpty(), offer(), peek(), poll(), remove(), toArray()

#### 4.31.14 ConcurrentMap Interface

- remove(), replace()

#### 4.31.15 CopyOnWriteArrayList Class

- addAll(), addAllAbsent(), containsAll(), equals(), hashCode(), indexOf(), isEmpty(), iterator(), lastIndexOf(), listiterator(), remove(), removeAll(), subList(), toArray(), toString()

#### 4.31.16 CopyOnWriteArraySet Class

- add(), addAll(), clear(), contains(), containsAll(), isEmpty(), iterator(), remove(), removeAll(), retainAll(), size(), toArray()

#### 4.31.17 CyclicBarrier Class

- await(), isBroken(), getNumberWaiting()

#### 4.31.18 DelayQueue Class

- add(), drainTo(), iterator(), offer(), peek(), poll(), put(), remove(), take(), toArray()

#### 4.31.19 ExcutorCompletionService Class

- submit()

#### 4.31.20 ExecutorService Interface

- invokeAll(), invokeAny(), submit()

#### 4.31.21 Executors Class

- newFixedThreadPool(),

#### 4.31.22 Future Interface

- cancel(), isCanceled(), isDone()

#### 4.31.21 FutureTask Class

- run(), isCancelled(), isDone(), cancel(), get(), set(), setException()

#### 4.31.22 LinkedBlockingQueue Class

- drainTo(), offer(), peek(), poll(), put(), remainingCapacity(), take(), toArray()

#### 4.31.23 PriorityBlockingQueue Class

- add(), comparator(), contains(), drainTo(), offer(), peek(), poll(), put(), take(), toArray()

#### 4.31.24 ScheduledExecutorService Interface

- schedule(), schduleAtFixedRate(), scheduleWithFixedDelay()

#### 4.31.25 ScheduledThreadPoolExecutor Class

- decorateTask(), schdule(), scheduleAtFixedRate(), scheduleWithFixedDelay(), execute(), submit(), setContinueExistingPeriodicTasksAfterShutdownPolicy(), shutdownNow()

#### 4.31.26 Semaphore Class

- drainPermits()

#### 4.31.27 SynchronousQueue Class

- drainTo(), put(), offer(), take(), contains(), remove(), containsAll(), toArray()

#### 4.31.28 ThreadPoolExecutor Class

- allowCoreThreadTimeOut(), shutdownNow(), allowCoreThreadTimeOut(),  
setKeepAliveTime(), beforeExecute(), afterExecute()

#### 4.31.29 TimeUnit Enum

- convert(), sleep(), timedJoin(), timedWait(), toDays(), toHours(), toMinutes()

## 5. 성능비교

자바 1.6과 자바 1.5의 성능 벤치마크 결과 가상머신(VM) 속도에서 40~100%에 이르는 현저한 성능향상이 있었고, collection 같은 코어 라이브러리에서도 최적화를 통해 성능향상을 이룰 수 있습니다.

## 6. 참고문헌

- <http://java.sun.com/javase/6/webnotes/features.html>
- <http://www.javalobby.org/java/forums/t66270.html>