The java.lang Package

Java Certification Study Group

java.lang overview

- Object
- Math
- The wrapper classes
- String
- StringBuffer

java.lang (non-objectives)

- Cloneable
  - A class implements the Cloneable interface to indicate to the clone method in class Object that it is legal for that method to make a field-for-field copy of instances of that class.
- SecurityManager
  - The security manager is an abstract class that allows applications to implement a security policy
- Exceptions
  - The class Exception and its subclasses are a form of Throwable that indicates conditions that a reasonable application might want to catch.

java.lang (non-objectives)

- Errors
  - An Error is a subclass of Throwable that indicates serious problems that a reasonable application should not try to catch. Most such errors are abnormal conditions
- ClassLoader
  - The class ClassLoader is an abstract class. Applications implement subclasses of ClassLoader in order to extend the manner in which the Java Virtual Machine dynamically loads classes.

java.lang.Object

- Class Object is the root of the class hierarchy.
- Every class has Object as a superclass.
- All objects, including arrays, implement the methods of this class.

java.lang.Object (con’t)

- If a class doesn’t extend another class, then compiler extends it from Object.
- For instance:

```
public class Lala { // stuff }
```

is created by the compiler as:

```
public class Lala extends Object{ // stuff}
```

java.lang.Object Methods

- Thread control (chapter 7)
  - wait(), notify(), notifyAll()
- General
  - equals(), toString()
- Not tested
  - finalize(), hashCode(), clone(), getClass()

java.lang.Object Methods (con’t)

- public boolean equals(Object object)
  - should be overrided
  - provides “deep” comparison
  - not the same as ==!
    - == checks to see if the two objects are actually the same object
    - equals() compares the relevant instance variables of the two objects

java.lang.Object Methods (con’t)

- public String toString()
  - should be overrided
  - returns a textual representation of the object
  - useful for debugging

java.lang.Math

- The class Math contains methods for performing basic numeric operations such as the elementary exponential, logarithm, square root, and trigonometric functions.
- class is final
- constructor is private
- methods are static

java.lang.Math (con’t)

- public static final double E
  - as close as Java can get to e, the base of natural logarithms
- public static final double PI
  - as close as Java can get to pi, the ratio of the circumference of a circle to its diameter

java.lang.Math (con’t)

int, long, float, double	abs()	Returns absolute value
int, long, float, double	max(x1, x2)	Returns the greater of x1 and x2
int, long, float, double	min(x1, x2)	Returns the smaller of x1 and x2

java.lang.Math (con’t)

- double ceil(double d)
  - returns the smallest integer that is not less than d, and equal to a mathematical integer
  - `double x = Math.ceil( 423.3267 );  x == 424.0;`
- double floor(double d)
  - returns the largest integer that is not greater than d, and equal to a mathematical integer
  - `double x = Math.floor( 423.3267 );  x == 423.0;`
- double random()
  - returns a random number between 0.0 and 1.0
  - note: java.util.Random has more functionality

java.lang.Math (con’t)

- double sin(double d)
  - returns the sine of d
- double cos(double d)
  - returns the cosine of d
- double tan(double d)
  - returns the tangent of d
- double sqrt(double d)
  - returns the square root of d

java.lang Wrapper Classes

Primitive Data Type	Wrapper Class
`boolean`	`Boolean`
`byte`	`Byte`
`char`	`Character`
`short`	`Short`
`int`	`Integer`
`long`	`Long`
`float`	`Float`
`double`	`Double`

java.lang Wrapper Classes

- encapsulates a single, immutable value
- all wrapper classes can be constructed by passing the value to be wrapper to the constructor
  - double d = 453.2344;
  - Double myDouble = new Double(d);
- can also pass Strings that represent the value to be wrapped (doesn’t work for Character)
  - Short myShort = new Short(“2453”);
  - throws NumberFormatException

java.lang Wrapper Classes

- the values can be extracted by calling XXXvalue() where XXX is the name of the primitive type.
- wrapper classes useful for classes such as Vector, which only operates on Objects

java.lang.String

- uses 16-bit Unicode characters
- represents an immutable string
- Java programs have a pool of literals
  - when a String literal is created, it is created in the pool
  - if the value already exists, then the existing instance of String is used
  - both == and equals() work

.equals() and ==

- Note this example:

```
String s1 = “test”;
String s2 = “test”;
s1 == s2; // returns true
s1.equals( s2 ); // returns true

String s3 = “abc”;
String s4 = new String( s3 );
s3 == s4; // returns false
s3.equals( s4 ); // returns true
```

whole lotta String methods

- `char charAt( int index)`
- `String concat( String addThis )`
- `int compareTo( String otherString )`
- `boolean endsWith( String suffix )`
- `boolean equals( Object obj )`
- `boolean equalsIgnoreCase( String s )`
- `int indexOf( char c )`
- `int lastIndexOf( char c )`
- `int length()`
- `String replace( char old, char new )`
- `boolean startsWith( String prefix )`
- `String substring( int startIndex )`

a few more

- `String toLowerCase()`
- `String toString()`
- `String toUpperCase()`
- `String trim()`
- phew
- remember: Strings are immutable so these methods return new Strings instead of altering itself

java.lang.StringBuffer

- represents a String which can be modified
- has a capacity, which can grow dynamically
- Methods for the exam
- `StringBuffer append( String s )`
- `StringBuffer append( Object obj )`
- `StringBuffer insert( int offset, String s )`
- `StringBuffer reverse()`
- `StringBuffer setCharAt( int offset, char c )`
- `StringBuffer setLength( int newlength )`
- `String toString()`

java.lang.String Concatenation

- Java has overloaded the + operator for Strings
- `a+b+c` is interpreted as

```
new StringBuffer().append(a).append(b).append(
  c).toString()
```

References

- Roberts, Simon and Heller, Philip, Java 1.1 Certification Study Guide, 1997: SYBEX. ISBN: 0-7821-2069-5
- JDK 1.1.7 JavaDoc HTML Pages