Java.util PACKAGE

JAVA.UTIL PACKAGE

- Java.util package contains the collections framework, legacy collection classes, event model, date and time facilities, internationalization, and miscellaneous utility classes.

Java.util.Calendar Class

The **java.util.calendar** class is an abstract class that provides methods for converting between a specific instant in time and a set of calendar fields such as YEAR, MONTH, DAY_OF_MONTH, HOUR, and so on, and for manipulating the calendar fields, such as getting the date of the next week.

Following are the important points about Calendar:

- This class also provides additional fields and methods for implementing a concrete calendar system outside the package.
- Calendar defines the range of values returned by certain calendar fields.

- **Class declaration**
- Following is the declaration for **java.util.Calendar** class:

public abstract class Calendar

extends Object

implements Serializable, Cloneable, Comparable<Calendar>

JAVA.UTIL.STRINGTOKENIZER CLASS

- The **java.util.StringTokenizer** class allows an application to break a string into tokens.
- This class is a legacy class that is retained for compatibility reasons although its use is discouraged in new code.
- Its methods do not distinguish among identifiers, numbers, and quoted strings.
- This class methods do not even recognize and skip comments.

- **Class declaration**
- Following is the declaration for **java.util.StringTokenizer** class:

```
public class StringTokenizer extends Object
implements Enumeration<Object>
```

- Class constructors
- **StringTokenizer(String str)**

  This constructor a string tokenizer for the specified string.

- **StringTokenizer(String str, String delim)**

  This constructor constructs string tokenizer for the specified string.

- **StringTokenizer(String str, String delim, boolean returnDelims)**

  This constructor constructs a string tokenizer for the specified string.

- Class methods
- **int countTokens()**

  This method calculates the number of times that this tokenizer's nextToken method can be called before it generates an exception.
- **boolean hasMoreElements()**

  This method returns the same value as the hasMoreTokens method.
- **boolean hasMoreTokens()**

  This method tests if there are more tokens available from this tokenizer's string.
- **Object nextElement()**

  This method returns the same value as the nextToken method, except that its declared return value is Object rather than String.
- **String nextToken()**

  This method returns the next token from this string tokenizer.
- **String nextToken(String delim)**

  This method returns the next token in this string tokenizer's string.

Java.util.Random Class

- The **java.util.Random** class instance is used to generate a stream of pseudorandom numbers.Following are the important points about Random:
- The class uses a 48-bit seed, which is modified using a linear congruential formula.
- The algorithms implemented by class Random use a protected utility method that on each invocation can supply up to 32 pseudorandomly generated bits.

- Class declaration
- Following is the declaration for **java.util.Random** class:

  public class Random extends Object implements Serializable

- Class constructors

  Random() This creates a new random number generator.
- **Random(long seed)** This creates a new random number generator using a single long seed.

- **Class methods**
- **protected int next(int bits)**: This method generates the next pseudorandom number.
- **boolean nextBoolean()**: This method returns the next pseudorandom, uniformly distributed boolean value from this random number generator's sequence.
- **void nextBytes(byte[] bytes)**: This method generates random bytes and places them into a user-supplied byte array.
- **double nextDouble()**: This method returns the next pseudorandom, uniformly distributed double value between 0.0 and 1.0 from this random number generator's sequence.
- **float nextFloat()**: This method returns the next pseudorandom, uniformly distributed float value between 0.0 and 1.0 from this random number generator's sequence.

- **double nextGaussian()**: This method returns the next pseudorandom, Gaussian ("normally") distributed double value with mean 0.0 and standard deviation 1.0 from this random number generator's sequence.
- **int nextInt()**: This method returns the next pseudorandom, uniformly distributed int value from this random number generator's sequence.
- **int nextInt(int n)**: This method returns a pseudorandom, uniformly distributed int value between 0 (inclusive) and the specified value (exclusive), drawn from this random number generator's sequence.
- **long nextLong()**: This method returns the next pseudorandom, uniformly distributed long value from this random number generator's sequence.
- **void setSeed(long seed)**: This method sets the seed of this random number generator using a single long seed.

Java.util.Date Class

- The **java.util.Date** class represents a specific instant in time, with millisecond precision.

- Class declaration

  Following is the declaration for **java.util.Date** class:

  public class Date extends Object implements Serializable, Cloneable, Comparable<Date>

- Class constructors
- **Date()**: This constructor allocates a Date object and initializes it so that it represents the time at which it was allocated, measured to the nearest millisecond.

- **Date(long date)**: This constructor allocates a Date object and initializes it to represent the specified number of milliseconds since the standard base time known as "the epoch", namely January 1, 1970, 00:00:00 GMT.

- Class methods
- **boolean after(Date when)**: This method tests if this date is after the specified date.
- **boolean before(Date when)**: This method tests if this date is before the specified date.
- **Object clone()**: This method return a copy of this object.
- **int compareTo(Date anotherDate)**: This method compares two Dates for ordering.
- **boolean equals(Object obj)**: This method compares two dates for equality.
- **long getTime()**: This method returns the number of milliseconds since January 1, 1970, 00:00:00 GMT represented by this Date object.
- **void setTime(long time)**: This method sets this Date object to represent a point in time that is time milliseconds after January 1, 1970 00:00:00 GMT.

Regular Expressions

- Java provides the java.util.regex package for pattern matching with regular expressions. Java regular expressions are very similar to the Perl programming language and very easy to learn.
- A regular expression is a special sequence of characters that helps you match or find other strings or sets of strings, using a specialized syntax held in a pattern. They can be used to search, edit, or manipulate text and data.

- The java.util.regex package primarily consists of the following three classes:
- **Pattern Class:** A Pattern object is a compiled representation of a regular expression. The Pattern class provides no public constructors. To create a pattern, you must first invoke one of its public static compile methods, which will then return a Pattern object. These methods accept a regular expression as the first argument.
- **Matcher Class:** A Matcher object is the engine that interprets the pattern and performs match operations against an input string. Like the Pattern class, Matcher defines no public constructors. You obtain a Matcher object by invoking the matcher method on a Pattern object.
- **PatternSyntaxException:** A PatternSyntaxException object is an unchecked exception that indicates a syntax error in a regular expression pattern.