RECURSION SOFTWARE, Inc.

Software for Software Developers™

JGL® toolkit

Collection and Algorithms for JAVA™ Developers

Release 5.0

JGL Release 5.0 is a high-performance, full featured, and easy to use extension to the Java™ Developer's Kit (JDK 5.0). It provides a set of collections, iterators and algorithms that augment and work seamlessly with the JDK collections API to provide advanced data management capabilities.

JGL Toolkit 5.0 supplements Sun's JDK 5.0 with the following:

- 28 additional collections
- 205 additional algorithm methods
- 8 additional comparators
- 31 additional iterators

In addition to Sun's JDK 5.0 features, JGL Toolkit 5.0 adds the following features:

- 6 buffers
- 30 algorithm classes
- 6 generators
- 61 functions
- 75 predicates
- Generic arithmetic and constraints framework.

JGL's 50+ algorithms can operate upon a wide variety of data structures. Function objects and predicates add further flexibility.

- JGL embraces and extends the JDK. It is compatible with the Collection, List, Iterator, and ListIterator classes.
- All JGL algorithms operate not only on JGL containers, but also on Java arrays of objects, Java arrays of primitives, and JDK Collections and Lists.
- JDK algorithms in java.util.Collections and java.util.Arrays operate on JGL containers.
- As long as you adhere to standard Java language conventions, you don't have to add any code to your classes in order to store them in a JGL container.
- JGL follows the same thread safety model as the JDK. That is, access to containers is not synchronized, by default, for performance reasons: you can make the synchronization decisions based on your application's usage patterns and performance. For simple synchronization needs, however, you can easily create a thread safe JGL container using the same wrapping mechanisms used for JDK collections. (e.g., java.util.Collections.synchronizedCollection).
- JGL containers support Serialization, a lightweight object persistence layer built into the Java language in JDK 1.1. The iterators for all JGL Sequences are also serializable.

Containers

JGL includes 6 highly optimized data structures that fill in some gaps in the Java 2 Collections API. Often, these take the form of high-level containers representing a specific usage pattern that are backed by a lower-level container. For example, JGL provides a Stack class, which has the traditional push and pop methods for a stack data structure. By using Stack instead of a standard LinkedList, for instance, you can write code that clearly communicates the intended use of that container.
Each JGL container has been engineered for performance and ease of use. In addition, all JGL containers support the appropriate standard Java Collection API interfaces and work just like the Collections API classes with respect to multithreaded situations. One of JGL's strengths is its seamless compatibility with the Java Collections API.

Algorithms

One of JGL's major features is its complement of over 50 reusable algorithms that can perform tasks ranging from a simple filtering operation to a complex quicksort. Instead of embedding algorithms into containers, JGL algorithms are encapsulated within their own classes and can operate upon a wide variety of data structures, including JGL containers, JDK collections, and native Java arrays. All algorithms that are closely related are encapsulated as class methods of a single class

JGL® toolkit

Collection and Algorithms for JAVA™ Developers

Function Objects and Predicates

More advanced users will find the power and flexibility of function objects and predicates the open door to using JGL® for enterprise development and extending JGL® to add new features. Function objects are used by selective algorithms so that programmers can encapsulate behaviors for use by the algorithm. For example, using a function object, you can execute a transformation upon a collection of objects in place. JGL 5.0 contains over 30 new reusable function objects. Predicates are used by both algorithms and containers for ordering and comparing objects. For example, using a predicate, you can order a set in ascending or descending order, or perhaps in an order based on a complex, domain-specific calculation.

Iterators

In addition to supporting the JDK `Iterator type`, JGL includes several powerful classes of iterators. For example, a `ReverseIterator` allows you to iterate backwards over a container. JGL also includes an iterator for stepping through an I/O stream.

Packages in JGL®

Table 1 provides a list of the JGL packages and their contents.

JGL® comes complete with the following:

- A single jar file containing the JGL classes
- Complete HTML API javadoc documentation
- A comprehensive HTML user guide packed with examples.

Table 1 : JGL Packages

Package	Contents
com.recursionsw.jgl	Interfaces, collection classes, and iterators
com.recursionsw.jgl.adapters	Java-native array adapters
com.recursionsw.jgl.algorithms	All JGL® algorithms
com.recursionsw.jgl.arithmetic	Generic arithmetic classes
com.recursionsw.jgl.collections	Iterators over the collections
com.recursionsw.jgl.constraints	JGL constraint framework and basic classes
com.recursionsw.jgl.fileutil	Useful file utility for iteration
com.recursionsw.jgl.functions	Function objects that can be used to modify algorithms
com.recursionsw.jgl.generators	All JGL generators
com.recursionsw.jgl.iterators	Specialized JGL iterators
com.recursionsw.jgl.predicates	Predicate classes used to affect element ordering within containers
com.recursionsw.jgl.util	A few miscellaneous utility classes, such as conditional enumeration

Recursion Software, Inc.
2591 North Dallas Parkway
Suite 200
Frisco, Texas 75034
1.800.727.8674
www.recursionsw.com

RECURSION SOFTWARE, Inc.