Generics

Bloh

Home

In release 1.5, generics were added to Java. Before generics, you had to cast every object you read from a collection. If someone accidentally inserted an object of the wrong type, casts could fail at runtime. With generics, you tell the compiler what types of objects are permitted in each collection. The compiler inserts casts for you automatically and tells you at compile time if you safer and clearer, but these benefits come with complications.

Item 23: Don’t use raw types in new code.

First, a few terms. A class or interface whose declaration has one or more type parameters is a generic class or interface. For example, as of release 1.5, the List interface has a single type parameter, E, representing the element type of the list. Technically the name of the interface is now List<E> (read "list of E"), but people often call it List for short. Generic classes and interfaces are collectively known as generic types.

Each generic type defines a set of parameterized types, which consist of the class or interface name followed by an angle-blracketed list of actual type parameters corresponding to the generic type’s formal type parameters. For example, List<String> (read "list of string") is a parameterized type representing a list whose elements are of type String. (String is the actual type parameter corresponding to the formal type parameter E.)

Finally, each generic type defines a raw type, which is the name of the generic type used without any accompanying actual type parameters. For example, the raw type corresponding to List<E> is List. Raw types behave as if all of the generic type information were erased from the type declaration. For all practical purposes, the raw type List behaves the same way as the interface type List did before generics were added to the platform.

Before release 1.5, this would have been an exemplary collection declaration:

```
// Now a raw collection type - don’t do this!
/**
 * My stamp collection. Contains only Stamp instances.
 */
private final Collection stamps = ... ;
```

If you accidentally put a coin into your stamp collection, the erroneous insertion compiles and runs without error:

Item 24: Eliminate unchecked warnings

When you program with generics, you will see many compiler warnings: unchecked cast warnings, unchecked method invocation warnings, unchecked generic array creation warnings, and unchecked conversion warnings. The more experience you aquire with generics, the fewer warnings you’ll get, but don’t expect newly written code that uses generics to compile cleanly.

Many unchecked warnings are easy to eliminate. For example, suppose you accidentally write this declaration: