

Java CE Options

Java CE options are visible every time you attempt to reverse source code. This section describes the **Resolve collection generic** option for Java-specific options. Other options are common for all languages and are described in [Reverse Options](#).

Resolve collection generics option

By default, this option is selected in the **Reverse Options** dialog. Java introduced parameterized types in JLS 3 and added type variables to all Java collections. With reverse engineering, it is possible to find which type is in a Java collection and make associations directly to the contained type, instead of the Java collection.

Reverse engineering allows you to find a Java parameterized collection and retrieves the Java type used in a container. This type is set as a *Type* to the UML Property. The container type is set to the UML Property/Java language property "Container" as a simple string.

Example

Source code sample

```
public class Test
{
    java.util.List<String> attribute;
}
```

UML Model



Attribute type, retrieved from collection

Specification of Property attribute

Language-specific code generation/reverse engineering options

Change code generation/reverse engineering options for the selected Property.

~attribute : java::lang::String [0..*]

Documentation/Hyperlinks

Usage in Diagrams

Inner Elements

Relations

Connectors

Tags

Constraints

C++ Language Properties

Traceability

C# Language Properties

Language Properties

Language Properties

Properties: Standard

CIL	
Java	
Transient modifier	not transient
Volatile modifier	not volatile
Container	java.util.Set<\$\$type\$\$>

Type here to filter properties

Close

Back

Forward

Help

Collection type in UML Property specification

Note, that *\$\$type\$\$* shows where it should be in the lined UML Property type on code generation.