Overview

Java 5 annotations

Java 5 annotations are standardized through the JSR-175 specification. When you compile annotated source code the `javac` compiler will embed the annotations in the compiled class, and this information can then be read back using the reflection API.

Annotations in Java 5 are first class citizen through the `@interface` keyword.

```
// A Java 5 Annotation
public @interface Asynchronous {
    int timeout() default 5;
    String label();
}

// A Java 5 annotated method
@Asynchronous(timeout=5, label="will run for
a while")
public Object someMethod() {
    ...
}
```

backport175 annotations for Java 1.3/1.4

backport175 is an implementation of the JSR-175 specification for Java 1.3/1.4 which provides the same user experience as regular Java 5 annotations.

Annotations are defined using a regular Java interface (hence providing strongly typed access to annotation values), and annotated source code appears in the form of doclet in the JavaDoc part.

The backport175 compiler (available through command line or as an Ant task) allows you to post-compile your classes to embed the annotation information inside the class' bytecode. The annotations are bytecode compatible with regular Java 5 annotations and therefore can be treated the same by tools, regular Java reflection (on Java 5) etc.

Note: it is advised to have a space between the annotation and its value as below though not mandatory.

```
// A Java 1.3/1.4 Annotation with
backport175
public interface Asynchronous {
    /**
     * Default value is supported through
this specific annotation
     *
     *
@org.codehaus.backport175.DefaultValue (5)
     */
    int timeout();

    String label();
}

// A Java 1.3/1.4 annotated method
/**
 * @Asynchronous (timeout=5, label="will run
for a while")
 */
public Object someMethod() {
    ...
}
```

The annotations can then be accessed using the API in the `org.codehaus.backport175.reader.Annotations` class which allows to retrieve reflectively both regular Java 5 annotations and Java 1.3/1.4 annotations in a consistent way, allowing you to adopt annotation driven development even without Java 5, while having a simple migration path. For more information on how to access annotations, read this section.