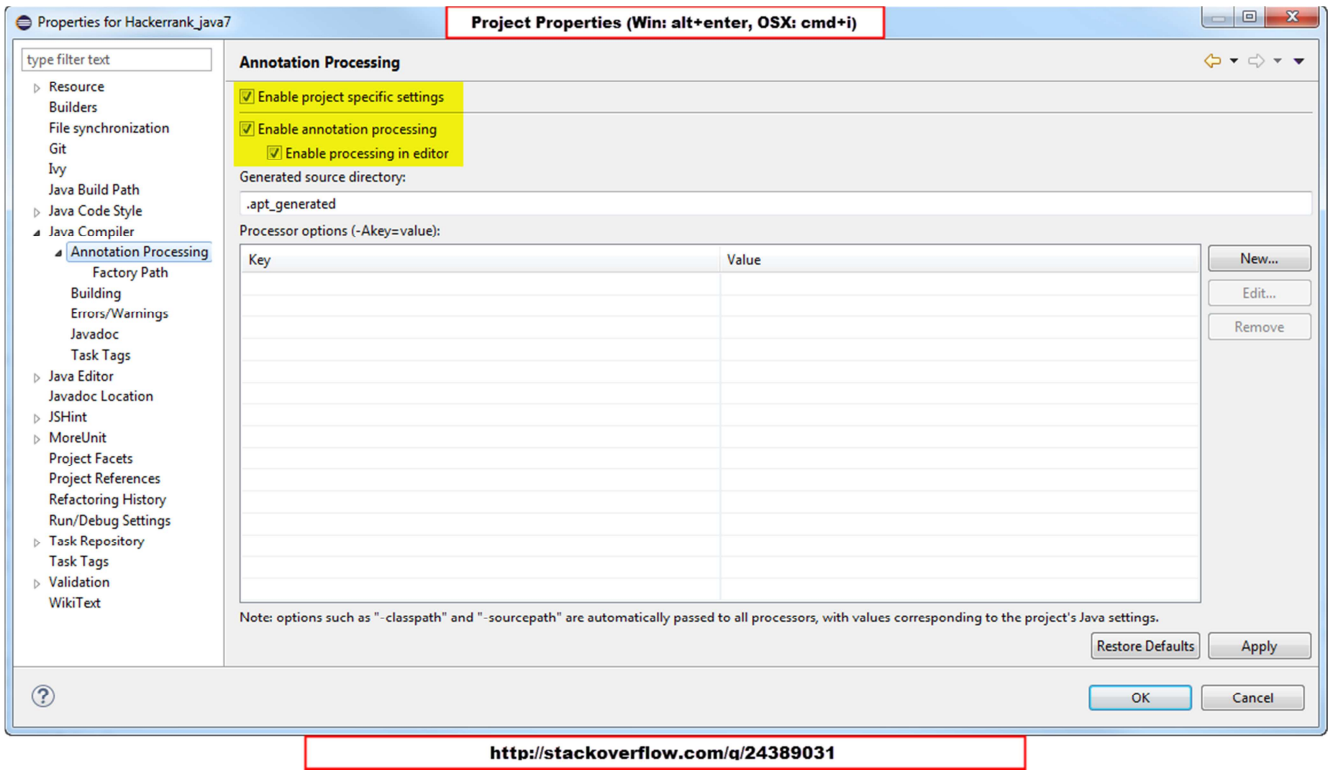
Setup JMH to run in Eclipse

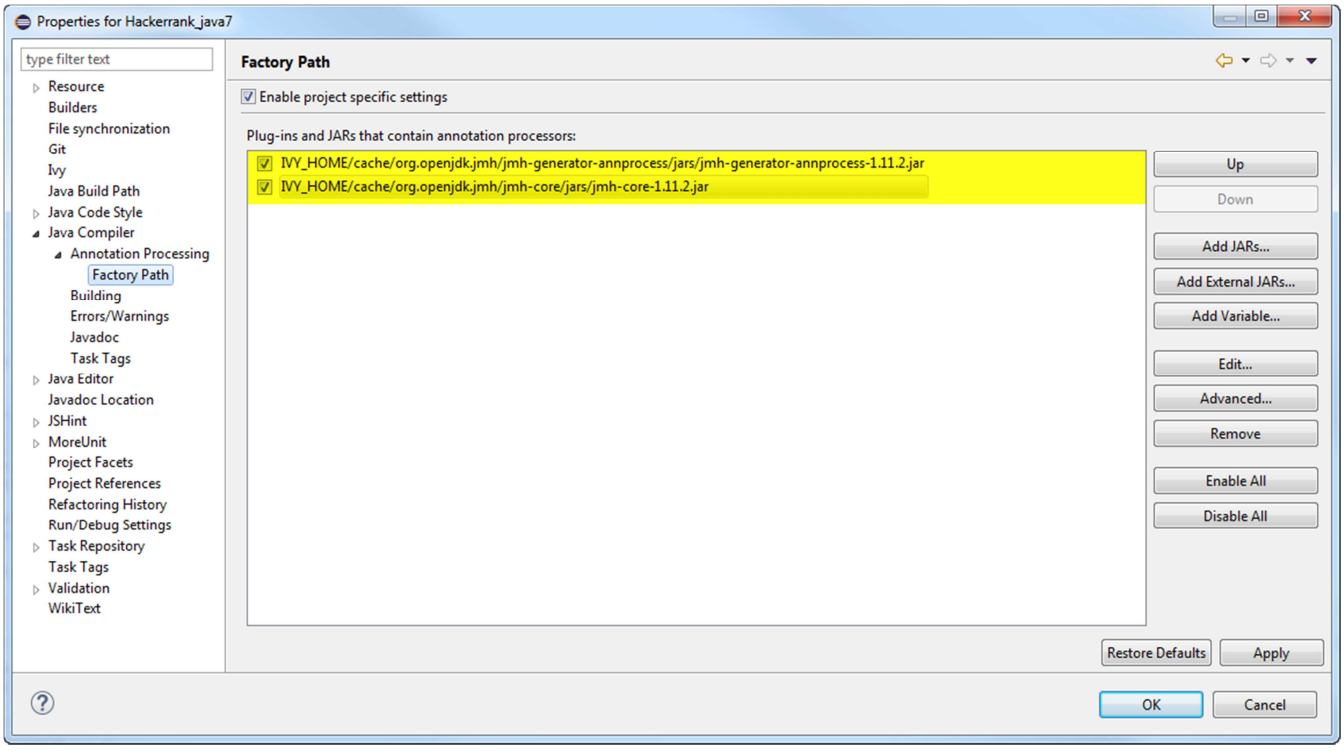
Depends on Ivy as dependency management:

```
    <configurations defaultconfmapping="compile->default">
        <conf name="compile" description="Required to compile application"/>
        <conf name="runtime" description="Additional run-time dependencies" extends="compile"/>
        <conf name="test"    description="Required for test only" extends="runtime"/>
        <conf name="benchmark"    description="Required for benchmarking only" extends="runtime"/>
    </configurations>
    <dependencies>
        <dependency org="org.openjdk.jmh" name="jmh-core" rev="1.11.2" conf="runtime->default,sources,javadoc"/>
        <dependency org="org.openjdk.jmh" name="jmh-generator-annprocess" rev="1.11.2" conf="runtime-
>default,sources,javadoc"/>
        <!-- <dependency org="org.openjdk.jmh" name="jmh-core-benchmarks" rev="1.11.2"  conf="runtime-
>default,sources,javadoc"/> -->
    </dependencies>
```

Project settings to set:



http://stackoverflow.com/q/24389031



Example Usage:

```
import java.util.ArrayList;
import java.util.List;
import java.util.Random;
import java.util.concurrent.TimeUnit;

import org.junit.Test;
import org.openjdk.jmh.annotations.Benchmark;
import org.openjdk.jmh.annotations.Level;
import org.openjdk.jmh.annotations.Mode;
import org.openjdk.jmh.annotations.Scope;
import org.openjdk.jmh.annotations.Setup;
import org.openjdk.jmh.annotations.State;
import org.openjdk.jmh.infra.Blackhole;
import org.openjdk.jmh.runner.Runner;
import org.openjdk.jmh.runner.options.Options;
import org.openjdk.jmh.runner.options.OptionsBuilder;
import org.openjdk.jmh.runner.options.TimeValue;

//http://stackoverflow.com/a/30486197
//http://stackoverflow.com/q/24389031
public class BenchmarkExample {

    @Test
    public void launchBenchmark() throws Exception {

        Options opt = new OptionsBuilder()
                // Specify which benchmarks to run.
                // You can be more specific if you'd like to run only one
                // benchmark per test.
                .include(this.getClass().getName() + ".*")
                // Set the following options as needed
                .mode(Mode.AverageTime).timeUnit(TimeUnit.MICROSECONDS)
                .warmupTime(TimeValue.seconds(1)).warmupIterations(2)
                .measurementTime(TimeValue.seconds(1)).measurementIterations(2)
                .threads(2).forks(1).shouldFailOnError(true).shouldDoGC(true)
                // .jvmArgs("-XX:+UnlockDiagnosticVMOptions",
                // "-XX:+PrintInlining")
                // .addProfiler(WinPerfAsmProfiler.class)
                .build();

        new Runner(opt).run();
    }

    // The JMH samples are the best documentation for how to use it
    // http://hg.openjdk.java.net/code-tools/jmh/file/tip/jmh-
samples/src/main/java/org/openjdk/jmh/samples/
    @State(Scope.Thread)
    public static class BenchmarkState {
        List<Integer> list;

        @Setup(Level.Trial)
        public void initialize() {

            Random rand = new Random();

            list = new ArrayList<>();
            int counter = 1000;
            while (counter > 0) {
                list.add(rand.nextInt());
                counter--;
            }
        }
    }

    @Benchmark
    public void benchmark1(BenchmarkState state, Blackhole bh) {

        List<Integer> list = state.list;
        for (int i = 0; i < 1000; i++)
            bh.consume(list.get(i));
    }

}
```