GROUPON

The Long Tail Treasure Trove

xbean-finder

Apache 2

http://tomee.apache.org/dev/xbean-finder.html

```java
@Plugin
public class FancyThing
{
    // ...
}
```

```java
Archive archive = new JarArchive(classloader, jar_url);
AnnotationFinder f = new AnnotationFinder(archive);
List<Class<?>> plugins = f.findAnnotatedClasses(Plugin.class)
```

Shrinkwrap Resolver

Apache 2

https://github.com/shrinkwrap/resolver

```
String coords = "org.apache.activemq:apollo-cli:1.6";
File[] files = Maven.resolver()
                    .resolve(coords)
                    .withTransitivity()
                    .asFile();

URL[] urls = new URL[files.length];
for (int i = files.length - 1; i >= 0; i--) {
    urls[i] = files[i].toURI().toURL();
    System.out.println(files[i].getAbsolutePath());
}

String cn = "org.apache.activemq.apollo.cli.commands.Run";
URLClassLoader cl = new URLClassLoader(urls);
Class<?> clz = cl.loadClass(cn);
System.out.println(clz);
```

zt-zip

Apache 2

https://github.com/zeroturnaround/zt-zip

```
ZipUtil.iterate(new File("demo.jar"), new ZipEntryCallback()
{
  public void process(InputStream in, ZipEntry ze)
                        throws IOException
  {
    if (ze.getName().endsWith(".class"))
    {
      System.out.println("Found " + ze.getName());
      IOUtils.copy(in, System.out);
    }
  }
});
```

Airline

Apache 2

https://github.com/airlift/airline

```
public class Git
{
    public static void main(String[] args)
    {
        CliBuilder<Runnable> builder = Cli.<Runnable>builder("not-git")
                .withDescription("the content tracker")
                .withDefaultCommand(Help.class)
                .withCommands(Help.class, Add.class);

        builder.withGroup("remote")
                .withDescription("Manage set of tracked repositories")
                .withDefaultCommand(RemoteShow.class)
                .withCommands(RemoteShow.class, RemoteAdd.class);

        Cli<Runnable> gitParser = builder.build();

        gitParser.parse(args).run();
    }
```

```
@Command(name = "add", description = "Add file contents to the index")
public class Add implements Runnable
{
  @Arguments(description = "Patterns of files to be added")
  public List<String> patterns;

  @Option(name = "-i", description = "Add modified contents interactively.")
  public boolean interactive;

  public void run()
  {
    System.out.println(getClass().getSimpleName());
  }
}
```

```
$ ./not-git help
usage: not-git [-v] <command> [<args>]

The most commonly used not-git commands are:
    add        Add file contents to the index
    help       Display help information
    remote     Manage set of tracked repositories

See 'not-git help <command>' for more information on a specific command.

$
```

really-executable-jars -maven-plugin

Apache 2

https://github.com/brianm/really-executable-jars-maven-plugin

```
<plugin>
  <groupId>org.skife.maven</groupId>
  <artifactId>really-executable-jar-maven-plugin</artifactId>
  <version>1.1.0</version>
  <configuration>
    <!-- as "java $flags -jar ..." -->
    <flags>-Xmx128M</flags>

    <!-- (optional) -->
    <programFile>not-git</programFile>
  </configuration>

  <executions>
    <execution>
      <phase>package</phase>
      <goals>
        <goal>really-executable-jar</goal>
      </goals>
    </execution>
  </executions>
</plugin>
```

```
$ ./not-git help
usage: not-git [-v] <command> [<args>]

The most commonly used not-git commands are:
    add        Add file contents to the index
    help       Display help information
    remote     Manage set of tracked repositories

See 'not-git help <command>' for more information on a specific command.

$
```

jansi

Apache 2

http://jansi.fusesource.org/

```
Ansi ansi = Ansi.ansi();
ansi.cursor(height(), 0);
ansi.eraseLine(Ansi.Erase.FORWARD);

ansi.a(label).a(" ");
ansi.fg(Ansi.Color.GREEN);
ansi.a("[");

ansi.fg(Ansi.Color.BLUE);
ansi.a("*");

ansi.fg(Ansi.Color.GREEN);
ansi.a("]");
ansi.reset();

System.out.print(ansi);
System.out.flush();
```


pancake:~/src/java-progress-bar<master>$

jmxutils

Apache 2

https://github.com/martint/jmxutils

```
class ManagedObject {

    @Managed
    public int getValue() { ... }

    @Managed
    public void setValue(int value) { ... }

    @Managed(description="do the operation")
    public void operation() { ... }
}

...

MBeanExporter ex =
   new MBeanExporter(ManagementFactory.getPlatformMBeanServer());

ex.export("example:name=Pancake", new ManagedObject());
```

Feign

Apache 2

https://github.com/Netflix/feign

```
public interface GitHub {
  @RequestLine("GET /repos/{owner}/{repo}/contributors")
  List<Contributor> contributors(@Named("owner") String owner,
                                 @Named("repo") String repo);
}

public class Contributor {
  String login;
  int contributions;
}

GitHub gh = Feign.create(GitHub.class,
                         "https://api.github.com",
                         new GsonModule());
List<Contributor> contributors = gh.contributors("netflix",
                                                 "feign");
for (Contributor c : contributors) {
  System.out.println(c.login + " (" + c.contributions + ")");
}
```

Jerry/Lagarto

BSD

http://jodd.org/doc/jerry/

```
Jerry j = Jerry.jerry(
            NetUtil.downloadString("http://skife.org/")
          );

j.$("#page h1 a").each(new JerryFunction() {
  @Override
  public boolean onNode(final Jerry $this,
                        final int index)
  {
    System.out.println($this.text());
    return true;
  }
});
```

jchronic

Apache 2

https://github.com/samtingleff/jchronic

```
Chronic.parse("two months ago this friday");
Chronic.parse("5pm");
Chronic.parse("three minutes from now");
Chronic.parse("8/23/2013 4am");
```

assertj

Apache 2

https://github.com/joel-costigliola/assertj-core

```
assertThat(frodo.getName()).startsWith("Fro")
                           .endsWith("do")
                           .isEqualToIgnoringCase("frodo");


assertThat(fellowshipOfTheRing).hasSize(9)
                               .contains(frodo, sam)
                               .doesNotContain(sauron);
```

vtte

Apache 2

https://github.com/hgschmie/vtte

```
VeryTrivialTemplatingEngine vtte =
  new VeryTrivialTemplatingEngine("Hello, {name}[ {lastName}]!");

String one = vtte.render(ImmutableMap.of("name", "John"));
assertThat(one).isEqualTo("Hello John!");

String two = vtte.render(ImmutableMap.of("name", "John",
                                         "lastName", "Doe"));
assertThat(two).isEqualTo("Hello, John Doe!");
```

tape

Apache 2

http://square.github.io/tape/

```
Path tmp = Files.createTempFile("queue", ".tmp");
Files.delete(tmp);

QueueFile qf = new QueueFile(tmp.toFile());

qf.add(new byte[]{1, 2, 3});
qf.add(new byte[]{4, 5, 6});

assertThat(qf.peek()).isEqualTo(new byte[]{1, 2, 3});
qf.remove();

assertThat(qf.peek()).isEqualTo(new byte[]{4, 5, 6});
qf.remove();

assertThat(qf.peek()).isNull();

qf.close();
Files.delete(tmp);
```

connector/mxj

GPL

http://dev.mysql.com/doc/connector-mxj/en/connector-mxj.html

```
MysqldResource mysqld = new MysqldResource(empty_dir);

Map<String, String> props =
    ImmutableMap.of(MysqldResourceI.PORT, "8876",
                    MysqldResourceI.INITIALIZE_USER, "true",
                    MysqldResourceI.INITIALIZE_USER_NAME, "user",
                    MysqldResourceI.INITIALIZE_PASSWORD, "pass",
                    "default-time-zone", "+00:00");
mysqld.start("mysql-daemon", props);

assertThat(mysqld.isRunning()).isTrue();
while (!mysqld.isReadyForConnections()) { Thread.sleep(100); }

MysqlDataSource ds = new MysqlDataSource();
ds.setURL("jdbc:mysql://localhost:8876/foo?createDatabaseIfNotExist=true");
ds.setUser("user");
ds.setPassword("pass");

// use the datasource :-)
```

ness-pg-embedded

Apache 2

https://github.com/NessComputing/components-ness-pg

```
try (EmbeddedPostgreSQL db = EmbeddedPostgreSQL.start()) {
  try (Handle h = DBI.open(db.getPostgresDatabase())) {
    h.execute("CREATE DATABASE breakfast");
    h.execute("CREATE USER brianm with password 'secret'");
    h.execute("grant all privileges on database breakfast to brianm");
  }

  try (Handle h = DBI.open(db.getDatabase("brianm", "breakfast"))) {
    h.execute("create table food (id serial primary key, name text)");
    h.execute("insert into food (name) values ('pancake')");
  }
}
```

slice

Apache 2

https://github.com/airlift/slice

```
Slice slice = Slice.toUnsafeSlice(
                  ByteBuffer.allocateDirect(SOME_BIG_NUM)
              );

slice.fill((byte) 0);

slice.setBytes(4, new byte[]{1, 2, 3, 4, 5});
byte[] bytes = slice.getBytes(5, 3);

assertThat(bytes).isEqualTo(new byte[]{2, 3, 4});
```

Paranamer

BSD

http://paranamer.codehaus.org/

```java
@Test
public void testByteCodeReadingParanamer() throws Exception
{
  Paranamer p = new BytecodeReadingParanamer();

  Method greet = getClass().getMethod("greet", String.class);
  String[] names = p.lookupParameterNames(greet);

  assertThat(names).isEqualTo(new String[]{"name"});
}


public String greet(String name)
{
  return String.format("Hello, %s!", name);
}
```

sshj

Apache 2

https://github.com/shikhar/sshj

```
SSHClient ssh = new SSHClient();
ssh.addHostKeyVerifier(new PromiscuousVerifier());

ssh.connect("localhost", 2222);

ssh.authPassword("brianm", "abc123");
Session s = ssh.startSession();

Session.Command cmd = s.exec("ls -l /");
cmd.join();

String out = CharStreams.toString(
    new InputStreamReader(cmd.getInputStream(),
                          Charsets.UTF_8));

assertThat(out).isEqualTo("Hello, world!");
s.close();
ssh.close();
```

sshd-core

Apache 2

http://mina.apache.org/sshd-project/

```
SshServer server = SshServer.setUpDefaultServer();
server.setPort(2222);
server.setKeyPairProvider(new SimpleGeneratorHostKeyProvider());
server.setPasswordAuthenticator(new AllowAnyone());
server.setCommandFactory(new HelloWorldCommandFactory());
server.start();
...
public class HelloWorldCommandFactory implements CommandFactory {
  @Override
  public Command createCommand(final String command) {
    return new BaseCommand() {
      @Override
      public void start(final Environment env) throws IOException
      {
        out.write("Hello, world!".getBytes(Charsets.UTF_8));
        out.flush();
        exit.onExit(0);
      }
    };
  }
}
```

jline2

BSD

http://jline.github.io/jline2/

```
ConsoleReader reader = new ConsoleReader(System.in,
                                         System.out);
reader.setPrompt("$ ");
reader.addCompleter(new StringsCompleter(args));

String line;
while ((line = reader.readLine()) != null) {
  if (line.trim().equalsIgnoreCase("quit")) {
    System.exit(0);
  }
  System.out.printf("=> \"%s\"\n", line);
}
```

zt-exec

Apache 2

https://github.com/zeroturnaround/zt-exec

```
StartedProcess p = new ProcessExecutor()
                           .command("ls", "/tmp/")
                           .readOutput(true)
                           .exitValues(0)
                           .addListener(new ProcessListener()
                           {
                              // ...
                           })
                           .start();
String out = p.future().get(1, TimeUnit.SECONDS)
                       .outputUTF8();
```

jBCrypt

BSD

http://www.mindrot.org/projects/jBCrypt/

```
String hashed = BCrypt.hashpw("my password!",
                              BCrypt.gensalt());

if (BCrypt.checkpw("my password?", hashed)) {
   System.out.println("You are in!");
} else {
 System.out.println("No Login For You!");
}
```

joda-money

Apache 2

http://www.joda.org/joda-money/

```
Money money = Money.parse("USD 5.23");
Money more = money.plus(Money.of(CurrencyUnit.USD,
                                 new BigDecimal("1.23")));

Money java_zone = more.convertedTo(CurrencyUnit.getInstance("NOK"),
                                   new BigDecimal("6.07"),
                                   RoundingMode.HALF_UP);

assertThat(java_zone.toString()).isEqualTo("NOK 39.21");
```

jnr-ffi

Apache 2

https://github.com/jnr/jnr-ffi

```
public static interface LibC {
   int puts(String s);
}


LibC libc = LibraryLoader.create(LibC.class).load("c");

libc.puts("Hello, World");
```

sqlite-jdbc

Apache 2

https://bitbucket.org/xerial/sqlite-jdbc

```
Connection connection = DriverManager.getConnection("jdbc:sqlite:sample.db");
```

java-classmate

Apache 2

https://github.com/cowtowncoder/java-classmate

```
public interface Foo
{
  public List<String> bar();
}
```

```
TypeResolver tr = new TypeResolver();
MemberResolver mr = new MemberResolver(tr);

ResolvedType rt = tr.resolve(Foo.class);
ResolvedTypeWithMembers rtm = mr.resolve(rt, null, null);

Class<?> param_type = null;
for (ResolvedMethod member : rtm.getMemberMethods()) {
  if ("bar".equals(member.getName())) {
    ResolvedType arg_type = member.getReturnType();
    param_type = arg_type.getTypeParameters()
                         .get(0)
                         .getErasedType();
  }
}

assertThat(param_type).isEqualTo(String.class);
```

jackson-module-afterburner

Apache 2

https://github.com/FasterXML/jackson-module-afterburner

```
ObjectMapper mapper = new ObjectMapper()
mapper.registerModule(new AfterburnerModule());
```

jackson-dataformat-yaml

Apache 2

https://github.com/FasterXML/jackson-dataformat-yaml

```
public class Foo
{
  public List<String> things = Lists.newArrayList();
}
```

```
ObjectMapper m = new ObjectMapper(new YAMLFactory());
Foo foo = m.readValue("things:\n" +
                      "  - hello\n" +
                      "  - world\n",
                      Foo.class);

assertThat(foo.things).containsExactly("hello",
                                       "world");
```

unix4j

BSD

https://code.google.com/p/unix4j/

```
String out = Unix4j.fromStrings("hello world",
                                "hello frog",
                                "bye dinner")
                   .grep("hello")
                   .sed("s/o/O/")
                   .xargs()
                   .toStringResult();

assertThat(out).isEqualTo("hellO wOrld hellO frOg");
```

parboiled

Apache 2

https://github.com/sirthias/parboiled/

```
@BuildParseTree
public class PredicateParser extends BaseParser<Object> {

    @SuppressWarnings("InfiniteRecursion")
    public Rule Predicate() {
        return Sequence(Expression(),
                        OptionalWhitespace(),
                        Optional(Sequence(And(),
                                          OptionalWhitespace(),
                                          Predicate())), EOI);
    }

    public Rule Expression() {
        return Sequence(Atom(),
                        OptionalWhitespace(),
                        Operator(),
                        OptionalWhitespace(),
                        Atom());
    }
```

```
PredicateParser p = Parboiled.createParser(PredicateParser.class);
ParsingResult<Object> out = new ReportingParseRunner<>(
    p.Predicate()).run("event.sleepState == 'tired' " +
                       "&& ian != 'wombat squirrel 7'");
```

MVEL

Apache 2

https://github.com/mvel/mvel

```
boolean not_brian = MVEL.eval("name != 'Brian'",
                              ImmutableMap.of("name", "Ian"),
                              Boolean.class);

assertThat(not_brian).isTrue();
```

typesafe-config

Apache 2

https://github.com/typesafehub/config

```
foo {
  // important!
  bar = 10
  baz = 12
}
```

```
# or props style
foo.bar = 10
foo.baz - 12
```

```
Config conf = ConfigFactory.load();
int bar1 = conf.getInt("foo.bar");
Config foo = conf.getConfig("foo");
int bar2 = foo.getInt("bar");
```

Thank You!