Other Testing Techniques

EJBContainer

The `EJBContainer` API is a JavaEE API enriched by some OpenEJB features to make the testing easier.

It starts a container (embedded for case we are interested in) scanning the classpath. This operation can be slow and if you go with this solution maybe think to start it only once for all tests.

Sample

```java
import org.junit.After;
import org.junit.AfterClass;
import org.junit.Before;
import org.junit.BeforeClass;
import org.junit.Test;

import javax.ejb.embeddable.EJBContainer;
import javax.inject.Inject;
import javax.naming.NamingException;

import static org.junit.Assert.assertTrue;

public class ATest {
    @Inject
    private MyCDIBean aBean;

    @PersistenceContext
    private EntityManager em;

    @Resource
    private DataSource ds;

    @BeforeClass
    public static void start() throws NamingException {
        container = EJBContainer.createEJBContainer();
    }

    @AfterClass
    public static void shutdown() {
        if (container != null) {
            container.close();
        }
    }

    @Before
    public void inject() throws NamingException {
        container.getContext().bind("inject", this);
    }

    @After
    public void reset() throws NamingException {
        container.getContext().unbind("inject");
    }

    @Test
    public void aTest() {
        // ...
    }
}
```

It will use `createEJBContainer()` method to start the container and application, and `close()` to shutdown it.

OpenEJB provides the `bind("inject")` hack to be able to get injection in the test class.

OpenEJB JUnit

`openejb-junit` is another artifact providing some facilities for testing.

EJBContainer Rule

```
@Properties({
    @Property(key = DeploymentFilterable.CLASSPATH_EXCLUDE, value = "jar:.*"),
    @Property(key = DeploymentFilterable.CLASSPATH_INCLUDE, value = ".*openejb-
junit.*")
})
public class TestEJBContainerDefaultConfig {
    @Rule
    public final EJBContainerRule containerRule = new EJBContainerRule(this);

    @org.apache.openejb.junit.jee.resources.TestResource
    private Context ctx;

    @org.apache.openejb.junit.jee.resources.TestResource
    private java.util.Properties props;

    @org.apache.openejb.junit.jee.resources.TestResource
    private EJBContainer container;


    @Test
    public void configIsHere() {
        // ...
    }
}
```

TIP there is the equivalent runner: `@RunWith(EJBContainerRunner.class)`

InjectRule: injections for EJBContainerRule

```java
@Properties({
    @Property(key = DeploymentFilterable.CLASSPATH_EXCLUDE, value = "jar:.*"),
    @Property(key = DeploymentFilterable.CLASSPATH_INCLUDE, value = ".*myjar.*")
})
public class TestEJBContainerRule {
    @ClassRule
    public static final EJBContainerRule CONTAINER_RULE = new EJBContainerRule();

    @Rule
    public final InjectRule injectRule = new InjectRule(this, CONTAINER_RULE);

    @EJB
    private BasicEjbLocal ejb;

    @Test
    public void aTest() {
        // ...
    }
}
```

TIP an alternative in `openejb-core` is to use `org.apache.openejb.Injector.inject(instance)`