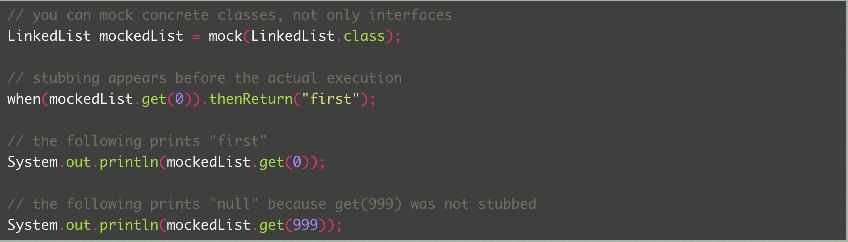
* What is CI, Why CI, What does CI actually do?
  + Continuous integration (CI) is the practice, in software engineering, of merging all developer working copies with a shared mainline several times a day
  + CI was originally intended to be used in combination with automated unit tests written through the practices of test-driven development. Initially this was conceived of as running all unit tests in the developer's local environment and verifying they all passed before committing to the mainline. This helps avoid one developer's work-in-progress breaking another developer's copy. If necessary, partially complete features can be disabled before committing using feature toggles. Later elaborations of the concept introduced build servers, which automatically ran the unit tests periodically or even after every commit and report the results to the developers. The use of build servers (not necessarily running unit tests) had already been practised by some teams outside the XP community. Nowadays, many organisations have adopted CI without adopting all of XP
  + Show <http://anjenkinsm.dev.activenetwork.com:8080/view/CIBS/job/CIBS-ActiveNetServlet/>, explain each component
    - **CheckStyle** - a static code analysis tool used in software development for checking if Java source code complies with coding rules. Code is checked using Sun coding convention with some customization introduced from AW.
    - **Findbugs** - a program which uses static analysis to look for bugs in Java code
    - **PMD** - is a source code analyzer. It finds common programming flaws like unused variables, empty catch blocks, unnecessary object creation, and so forth.
    - **UnitTest** – execute unit tests wrote in JUnit
    - **Test coverage** – execute unit tests with line/method/class test coverage calculated
* Gradle:  Enterprise Build Automation System
  + Compare to Maven, Ant
    - use groovy instead xml configurations, less code needed to achieve same thing compare to Maven and Ant
    - JVM based, friendly to other JVM based languages like java, groovy etc
    - better conflict resolution - explain transitive here
  + Current gradle implementation in ANET java projects
    - gradle-support: [https://fndsvn.dev.activenetwork.com/ActiveNet/core/](https://www.evernote.com/OutboundRedirect.action?dest=https%3A%2F%2Ffndsvn.dev.activenetwork.com%2FActiveNet%2Fcore%2F)
    - gradle build scripts in ActiveNetClients/ActiveNetPackage/ActiveNetServlet
    - show configurations for ActiveNetServlet/build.gradle
  + **Things need to be understood and practiced by each dev**
    - **How to build ActiveNet projects using gradle, how to run them** - refer to README.md in each project
    - **How to introduce a new third party dependency**
      * keep doing whatever you’re doing now include downloading jars and put int third party stuff
      * get gradle configs from <http://mvnrepository.com/>, add into build.gradle
    - **How to get latest changes from ActiveNetPackage and ActiveNetClients**
      * run ./g publish to upload the jar to <http://nexus.dev.activenetwork.com/nexus/index.html>
      * read more from README.md in ActiveNetPackage project
* Unit test frameworks
  + frameworks
    - [JUnit](http://junit.org/) - unit test framework
    - [Mockito](http://mockito.org/) - a mocking framework
    - [PowerMockito](https://www.evernote.com/OutboundRedirect.action?dest=https%3A%2F%2Fgithub.com%2Fjayway%2Fpowermock) - essential makeup for mock, especially for static method mocking
* Where to start unit test
  + projects and where to add unit test
    - ActiveNetServlet - /test
    - ActiveNetPackage - not enabled, but essentially needed
    - ActiveNetClients - not needed as there’re basically prototypes from 3rd party services(Offer, AA)
    - .Net projects - not covered in this phase
  + package and naming pattern
    - A test case usually covers test for a class
      * given src/ANServlet/ActiveNetServlet.java, unit test class should be test/ANServlet/ActiveNetServletTestCase.java
    - method naming
      * each test method should start with test, like ’testSomething()’, a better pattern would be test{methodName}{behavesWhat}{InWhatCondition}, for example HttpHelperTestCase*.testEncodeReturnRawForUnsupportedEncoding()*, so people who view the test method would get the behavior just from the name instead of diving into method body
  + annotations
    - @Test - this is a test
    - @Ignore - ignore from running
    - @Before - runs before every single test method
    - @After - runs after every single test method
    - @BeforeClass - a static method, run once before all test methods
    - @AfterClass - a static method, run once after all test methods
    - @Test( expected = NullPointerException.class) - this test throw a given exception
  + unit test
    - create a sample input, call the real method, assert the output
    - 
    - 
* Mock
  + Why
    - quick create a sample input with defined behavior
    - mock external dependencies, focus on current method only
    - mockup remote service invocation, db operations to speed up unit test, unit test has to be very fast or no one would actually like it
  + How
    - 
    - 
    - Refer to <http://site.mockito.org/mockito/docs/current/org/mockito/Mockito.html> for more official documents
    - Refer to [https://github.com/jayway/powermock/wiki/MockitoUsage](https://www.evernote.com/OutboundRedirect.action?dest=https%3A%2F%2Fgithub.com%2Fjayway%2Fpowermock%2Fwiki%2FMockitoUsage) on how to mock static methods using PowerMock
    - Refer to ActiveNetServlet/test for more real examples
* How to Run unit tests
  + IDE
    - eclipse/IDEA: right click in the source -> Run xxxxTestCase, or right click on a test method -> Run testxxxxxx
    - netbeans: [https://netbeans.org/kb/docs/java/junit-intro.html](https://www.evernote.com/OutboundRedirect.action?dest=https%3A%2F%2Fnetbeans.org%2Fkb%2Fdocs%2Fjava%2Fjunit-intro.html)
  + Command Line
    - windows
      * g test
      * g coverage
    - \*nix based
      * ./g test
      * ./g coverage
  + CI server
    - <http://anjenkinsm.dev.activenetwork.com:8080/view/CIBS/job/CIBS-ActiveNetServlet/>, run build every 15 mins if there’s new code checked in
    - it sends email if build is fail