





#### Lecture 8: Accessing the Web and Unit Tests





http://aiti.mit.edu

#### Interacting with the Web

### How to Access Web Content

- I. Give your app permission to access the web
- 2. Open a connection to a URL
- 3. Read data from the URL and store it somewhere
- 4. Display the data from the URL on your app

### How to Access Web Content

- I. Give your app permission to access the web
- 2. Open a connection to a URL
- 3. Read data from the URL and store it somewhere
- 4. Display the data from the URL on your app

#### Give your app permission to access the web

- Find AndroidManifest.xml file
- Navigate to the Permissions tab
- Select "Add → Uses Permission"
- Select android.permission.INTERNET from the drop-down menu

### How to Access Web Content

- I. Give your app permission to access the web
- 2. Open a connection to a URL
- 3. Read data from the URL and store it somewhere
- 4. Display the data from the URL on your app

# Open a connection to a URL

- A URL is a type of URI
- Sample code:

```
URL myURL;
myURL = new URL("http://myWebsite.com");
HttpURLConnection conn =
 (HttpURLConnection) url.openConnection();
```

### How to Access Web Content

- I. Give your app permission to access the web
- 2. Open a connection to a URL
- 3. Read data from the URL and store it somewhere
- 4. Display the data from the URL on your app

• Get data from the URL

InputStream in = conn.getInputStream();

• Get data from the URL

InputStream in = conn.getInputStream();

- Use a reader to convert the data into the format you want. Useful Java classes.
  - o InputStreamReader
  - BufferedReader
  - o StringBuffer
  - CharBuffer

#### Sample code (printing data out instead of storing)

```
BufferedReader in =
```

```
new BufferedReader(
```

```
new InputStreamReader(conn.getInputStream()));
String inputLine;
```

```
while ((inputLine = in.readLine()) != null) {
    System.out.println(inputLine);
}
```

- May encounter methods that throw exceptions, such as:
  - MalformedURLException
     (new URL() throws when the string isn't a URL)

#### IOException

(getInputStream() throws on bad connection)

• Handle them gracefully

• How should the app work without Internet?

### How to Access Web Content

- I. Give your app permission to access the web
- 2. Open a connection to a URL
- 3. Read data from the URL and store it somewhere
- 4. Display the data from the URL on your app

## Display the data from the URL on your app

Access your stored data and display it using whatever combination of layouts and widgets that you choose!

#### Unit Tests and JUnit

#### What are unit tests?

- Small pieces of code that test your code
  - Test the smallest testable piece (unit)
  - Tests interact with your main code

### Why unit test?

- Guarantee your code does what you say
- Uncover corner cases early on
  - Ensure graceful degradation (GPS unavailable?)
- Debug before you release
- Can help guide development
  - Test-driven development (write tests first)

### What goes into a unit test?

- Assertion A test of a single property or value (e.g. assert that "I+I" gives "2")
- Test Case A set of assertions that test a single function or use case
- Test Suite A collection of related Test Cases to run together
- Test Runner Code that runs the Test Suites
- Mock Object An object substituting for another (when the object itself is not being tested)

### Some Types of Assertions

- All can take an extra first argument String message to print out when the assertion fails
- org.junit.Assert.
  - o assertEquals(expected, actual)
    - Test that expected and actual are equal (.equals())
  - o assertTrue(condition)/assertFalse(condition)
    - Test that condition is true/false
  - o fail()
    - Always fail

See also: <u>http://junit.sourceforge.net/javadoc/org/junit/Assert.html</u>

### More Types of Assertions

- android.test.MoreAsserts.
  - o assertMatchesRegex(expectedRegex, actual)
    - Test that actual matches regular expression expectedRegex
  - o assertEmpty(iterable)
    - Test that iterable contains no objects
  - o assertContentsInOrder(iterable, expected...)
    assertContentsInAnyOrder(iterable, expected...)
    - Test that iterable contains exactly all of the remaining arguments in exact/any order and nothing else

See also: <u>http://developer.android.com/reference/android/test/MoreAsserts.html</u>

#### Mock Objects

- Objects that *implement* an *interface* (i.e. they look like the interface)
- But results of functions may be pre-defined (i.e. behavior is deterministic)

#### Mock Objects: Example

```
public interface ProxySettings {
    public abstract String fetchWithProxy(URL url);
}
public class Weather {
    public static String fetchCurrentWeather(
      String place, ProxySettings proxy) {
        /* ... */
        return proxy.fetchWithProxy(url);
    }
}
```

```
ProxySettings proxy = new DeviceProxySettings();
String s = Weather.fetchCurrentWeather("Mumbai", proxy);
Assert.assertEquals(s, "Rain");
```

#### Mock Objects: Example

#### Problem: DeviceProxySettings is device-specific! Also, if I don't use a proxy, I can't test!

Solution: Make a mock object (class: MockProxySettings)!

#### Mock Objects: Example

```
public class MockProxySettings implements ProxySettings {
    public String fetchWithProxy(URL url) {
        /* Fetch without a proxy! */
    }
}
```

```
ProxySettings proxy = new MockProxySettings();
String s = Weather.fetchCurrentWeather("Mumbai", proxy);
Assert.assertEquals(s, "Rain");
```

NOTE: What MockProxySettings does isn't important. What we care about is that fetchCurrentWeather works with a class that behaves like a ProxySettings interface.

### Unit Testing with JUnit

### Making a TestCase

```
public class Email {
    private String mSubject; // And so on...
    public Email(String from, String subject, String body) {
        mSubject = subject;
        // And so on...
    }
    public String getSubject() {
        return mSubject;
    }
}
```

### Making a TestCase

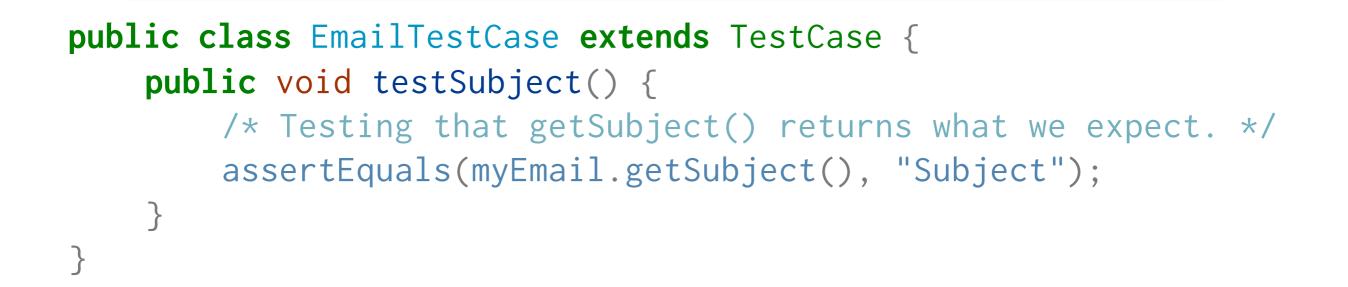
```
public class EmailTestCase extends TestCase {
    public Email myEmail;
```

}

```
protected void setUp() throws Exception {
    super.setUp();
    /* Set up the objects which will be tested. */
    myEmail = new Email("From", "Subject", "Body");
}
```

```
protected void tearDown() throws Exception {
    super.tearDown();
    /* Destroy the objects that were tested. */
}
```

### Adding Tests



#### All tests start with test!

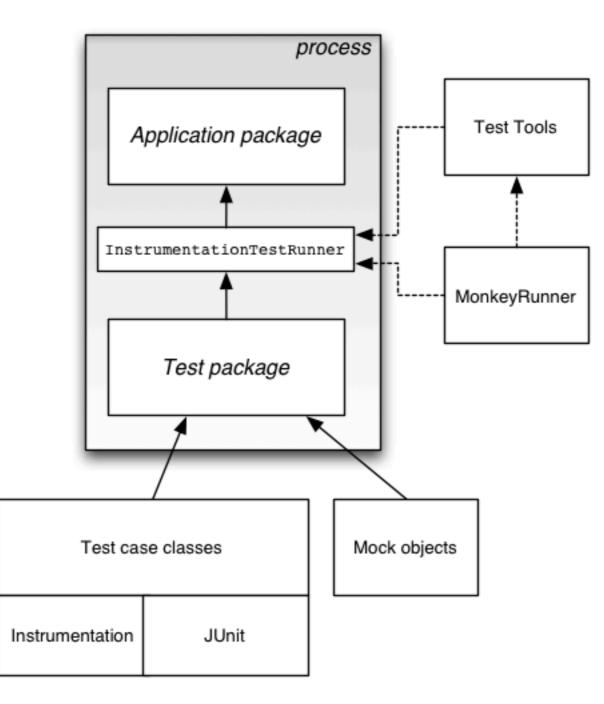
NOTE: TestCase extends [is a child class of] Assert, so assertEquals may be called without referring to Assert.

#### Collecting Tests in a TestSuite

```
public class MyTestSuiteBuilder {
    public static Test suite() {
        /* Create a TestSuite and add each test. */
        TestSuite suite = new TestSuite();
        suite.addTest(new EmailTest("testSubject"));
        return suite;
    }
}
                               or
    public static Test suite() {
        /* Create a TestSuite and add each test. */
        TestSuite suite = new TestSuite(EmailTest.class);
        return suite;
    }
```

#### Unit Testing on Android

### Unit Testing on Android

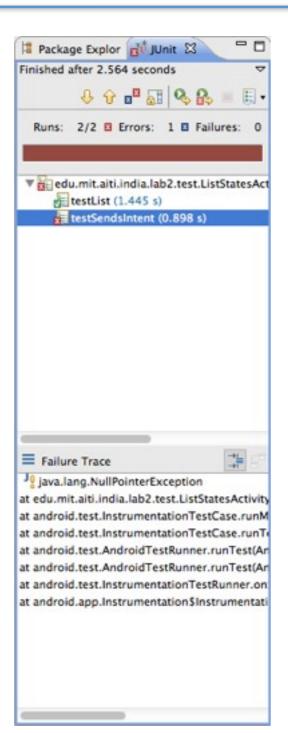


### Unit Testing in Eclipse

<ul> <li>O O J Java - Lab 2 Tests/src/edu/</li> <li>] ➡ □ □ ] ➡ J ➡ J ➡ J ➡ J</li> </ul>	👌 ] 🏇 • 🖸		{♥■■]2+?+*>	• = •
Package Explor ☎ 🗗 JUnit □ □ □ 🔄 🔄 ☑ ▽ ► 🕞 Lab 1 [lab1 56f21d3 (refs/tags/add	ListSta pacl	2 Lab 2 J <sup>Q</sup> 3 ListStatesActivityTestCase	*16	
<ul> <li>Lab 2 [lab2 solution]</li> <li>src</li> <li>&gt; edu.mit.aiti.india.lab2</li> <li>ListStatesActivity.java</li> </ul>	⊖ impo impo impo impo impo	<ul> <li>4 Navigation Examples</li> <li>5 Lab 1</li> <li>6 ILikeMumbai ILikeMumbai</li> <li>7 AITI Website AITI Website</li> </ul>	esActivity; pilActivity;	
<ul> <li>StateDetailActivity.java</li> <li>StateMapActivity.java</li> <li>gen [Generated Java Files]</li> <li>Android 4.1</li> <li>Android Dependencies</li> </ul>	impo impo impo impo impo	Run As Run Configurations Organize Favorites	<ul> <li>1 Android Application</li> <li>J<sup>Q</sup> 2 Android JUnit Test</li> <li>3 Java Applet</li> </ul>	∵#X
<ul> <li>assets</li> <li>bin</li> <li>libs</li> <li>res</li> <li>drawable-hdpi</li> </ul>	// An public	ActivityUnitTestCase runs the act c class ListStatesActivityTestCase rivate Intent mStartIntent;		X#7

Run like any other app

### Unit Testing in Eclipse



#### View tests which fail

And why

website] O O New	
)-HEAD] Select a wizard	EI EI
ngs/addin mples ma	
Wizards:	
type filter text	
🕨 🧀 General	
The Android	
Android Activity	(8-1
Android Application Project	8= (
Android Object	
Android Project from Existing Code	
Android Sample Project	<u>►</u>
Android Test Project	• • • •
Android XML File	
Android XML Layout File	
CVS	
s/ianjaco	LogCat
f4449602 's/ianjacc	🛃 🖏 -
< Back Next > Cancel	Finish
status notifica Or a Status Notification	lan lacobi ≤pipian@mit.edu:

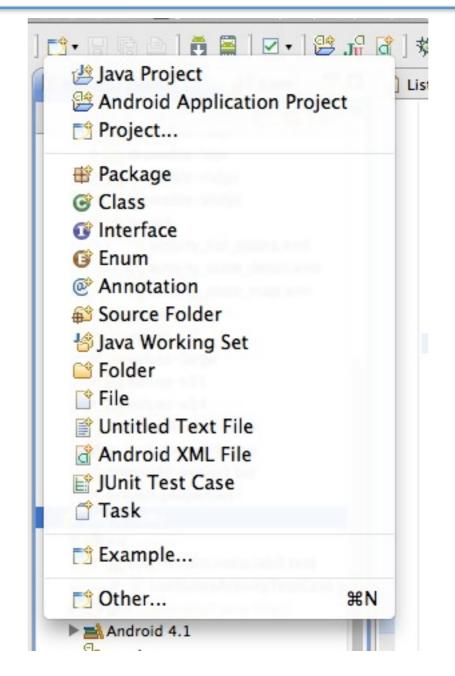
#### Make an Android Test Project

ebsite]	00	New Android Test Project				
HEAD] s/addin	Create Android Project Select project name and type of project					
	Project Name:	Lab 2 Test	-			
	Use default	t location				
	Location: Working sets	/Users/lanjacobi/Documents/AITI Eclipse Workspace/Lab 2 T Browse				
	ect to working sets					
	Working sets	s: [\$] [Select				
nples						
npiesi						
			: 23			
	?	< Back Next > Cancel Finish	or			

#### Name it after your project (Yes, this means you need 2 git repositories)

000	New Android Test Project
Select Test Target Choose a project to test	
<ul> <li>This project</li> <li>An existing Android project</li> </ul>	roject:
E Lab 1	
Lab 2 Lab 2 Tests	
Avigation Examples	
?	< Back Next > Cancel Finish

#### Choose the project to test



#### Add a new JUnit Test Case

	(edu)	⊖ ○ ⊖ New JUnit Test Case	
29		JUnit Test Case Select the name of the new JUnit test case. You have the options to specify the class under test and on the next page, to select methods to be tested.	E
Taat Daalsa aa	ibsit IEAE i/ade		Browse
Test Package –		Package. edu.mit.aiti.india.lab2.test	Browse
Test Class Name	ra	Name. ListStatesActivityTest Superclass: android.test.AndroidTestCase	Test Superclass
		Which method stubs would you like to create? setUpBeforeClass() tearDownAfterClass() setUp() tearDown() constructor Do you want to add comments? (Configure templates and default value here) Generate comments	
		Class under test: edu.mit.aiti.india.lab2.ListStatesActivity	Class being tested
		edu.mit.aiti.india.la     dRuntime	

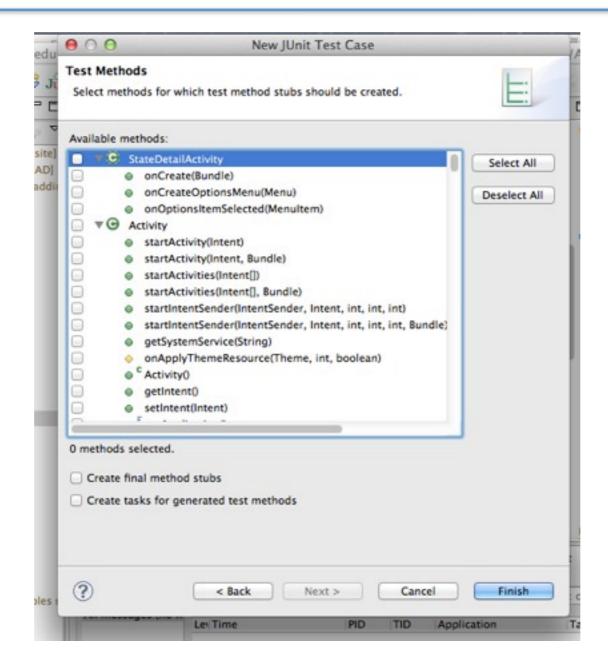
Set the properties of the Test Case

### Aside: Test Case Superclasses

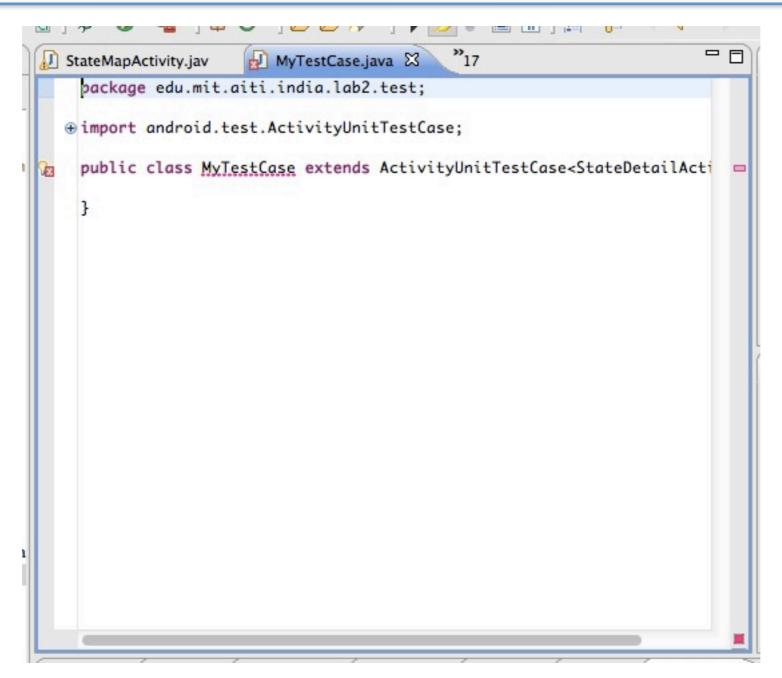
- TestCase JUnit default (not for Android classes)
- AndroidTestCase Android default
- ActivityInstrumentationTestCase2<T> Test Activities in Android environment
- ActivityUnitTestCase<T> Test Activities in standalone environment (e.g. to test Intents)
- ServiceTestCase Test Services
- ProviderTestCase2 Test Content Providers

#### What's with the <T>?

- <T> is used in "generic templates"
- T substituted with a class (e.g. 
   MyActivity>)
- Used with classes to specify the type of class being operated on
  - (e.g. ArrayList<String> is an ArrayList of String objects)



Select what functions you want to test (if any)



#### Basic Unit Test created!

### A Live Activity Unit Test!

#### References

- HttpURLConnection (Android APIs) <<u>http://developer.android.com/reference/java/net/HttpURLConnection.html</u>>
- Simple HttpURLConnection example
   <<u>http://digiassn.blogspot.in/2008/10/java-simple-httpurlconnection-example.html</u>>
- How do you GET/POST? See Tim Bray's "HttpURLConnection's Dark Secrets" <<u>http://www.tbray.org/ongoing/When/201x/2012/01/17/HttpURLConnection</u>>
- JUnit Cookbook
   <<a href="http://junit.sourceforge.net/doc/cookbook/cookbook.htm">http://junit.sourceforge.net/doc/cookbook/cookbook.htm</a>
- Android Testing
   <<a href="http://developer.android.com/tools/testing/index.html">http://developer.android.com/tools/testing/index.html</a>
- Unit Testing Best Practices
   <<u>http://www.bobmccune.com/2006/12/09/unit-testing-best-practices/</u>>