TestNG Cheat Sheet

TestNG Annotations

```
@Test             - Marks a method as a test case
@BeforeSuite      - Runs once before all tests in the suite
@AfterSuite       - Runs once after all tests in the suite
@BeforeTest       - Runs before any <test> tag in XML
@AfterTest        - Runs after all <test> tags in XML
@BeforeClass      - Runs before the first method in the class
@AfterClass       - Runs after all methods in the class
@BeforeMethod     - Runs before each @Test method
@AfterMethod      - Runs after each @Test method
@BeforeGroups     - Runs before the first method in a group
@AfterGroups      - Runs after the last method in a group
```

@Test Attributes

```
priority          - @Test(priority = 1) - Execution order (lower runs first)
enabled           - @Test(enabled = false) - Disable test
dependsOnMethods  - Depends on other test methods
groups            - Group test cases for selective execution
invocationCount   - Run the test multiple times
timeOut           - Fail if it takes longer than timeout (ms)
expectedExceptions  - Expect a specific exception to be thrown
```

Assertions

```
Assert.assertEquals(actual, expected);
Assert.assertTrue(condition);
Assert.assertFalse(condition);
Assert.assertNull(object);
Assert.assertNotNull(object);
Assert.fail("Force fail");
```

Sample testng.xml

```
<suite name="MySuite">
  <test name="MyTest">
    <classes>
      <class name="com.example.TestClass"/>
    </classes>
  </test>
</suite>
```

TestNG Cheat Sheet

Groups

```
@Test(groups = {"smoke"})
@Test(groups = {"regression"})

<groups>
  <run>
    <include name="smoke"/>
  </run>
</groups>
```

DataProvider Example

```
@DataProvider(name = "loginData")
public Object[][] getData() {
  return new Object[][] {{"user1", "pass1"}, {"user2", "pass2"}};
}

@Test(dataProvider = "loginData")
public void loginTest(String user, String pass) { ... }
```

Parallel Execution

```
<suite name="Suite" parallel="tests" thread-count="2">
  <test name="Test1">...</test>
  <test name="Test2">...</test>
</suite>
```

Maven Dependency

```
<dependency>
  <groupId>org.testng</groupId>
  <artifactId>testng</artifactId>
  <version>7.10.1</version>
  <scope>test</scope>
</dependency>
```

Listeners in TestNG

Implement ITestListener, ISuiteListener, or IInvokedMethodListener to hook into test events.

Example:
public class MyListener implements ITestListener {

```
    public void onTestSuccess(ITestResult result) {
        System.out.println("Test Passed: " + result.getName());
    }
}
```

Usage in testng.xml:

```
<listeners>
  <listener class-name="com.example.MyListener"/>
</listeners>
```

Retry Analyzer

Used to retry failed tests automatically.

Example:

```
public class RetryAnalyzer implements IRetryAnalyzer {
    private int count = 0;
    private static final int MAX_RETRY = 2;

    public boolean retry(ITestResult result) {
        if (count < MAX_RETRY) {
            count++;
            return true;
        }
        return false;
    }
}
```

Use in test method:

```
@Test(retryAnalyzer = com.example.RetryAnalyzer.class)
public void flakyTest() { ... }
```

Test Lifecycle Flow

Order of execution:

1. @BeforeSuite
2. @BeforeTest
3. @BeforeClass
4. @BeforeMethod
5. @Test
6. @AfterMethod
7. @AfterClass
8. @AfterTest
9. @AfterSuite