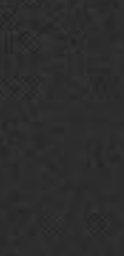
# Release with confidence

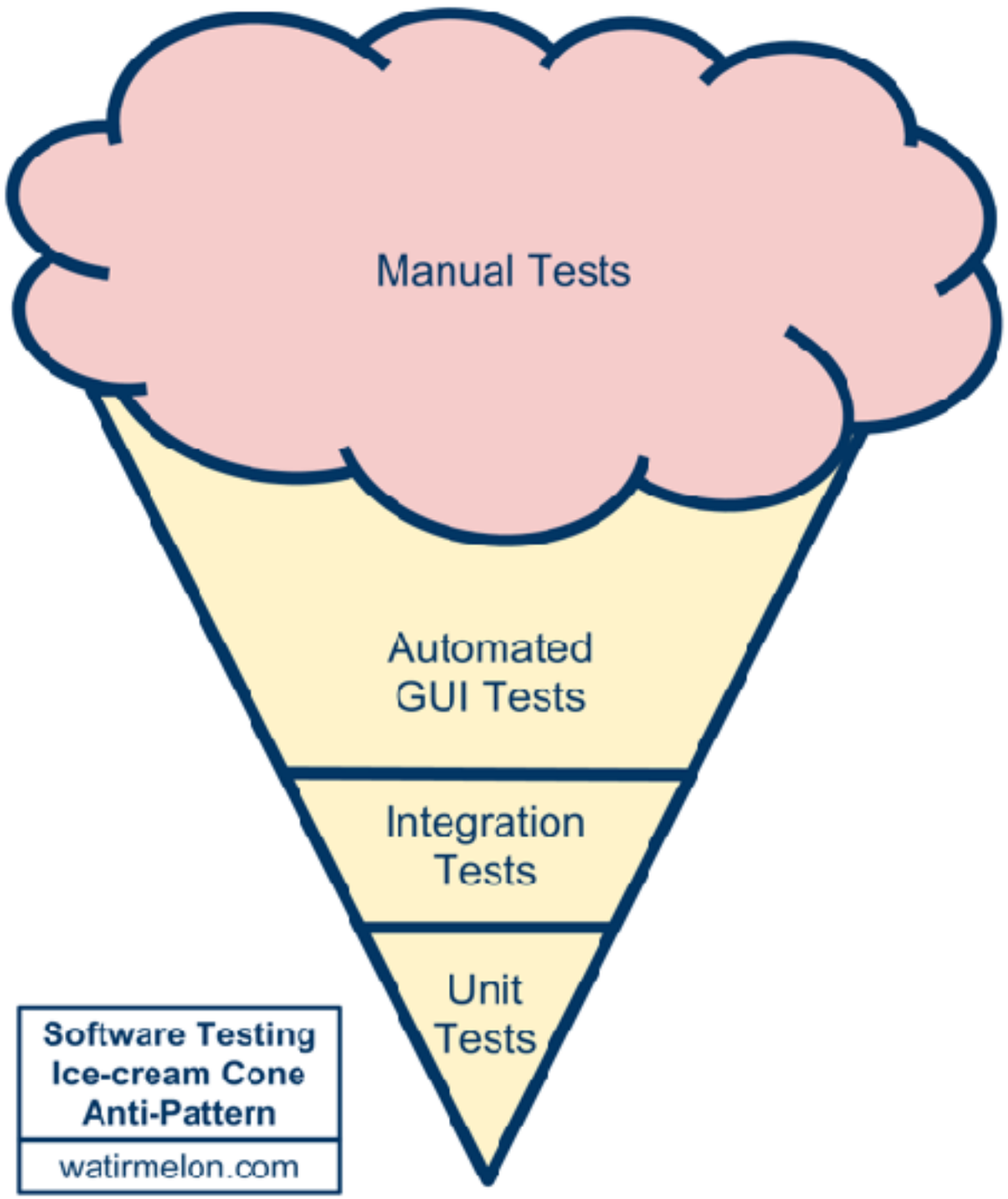


# Your goal ?

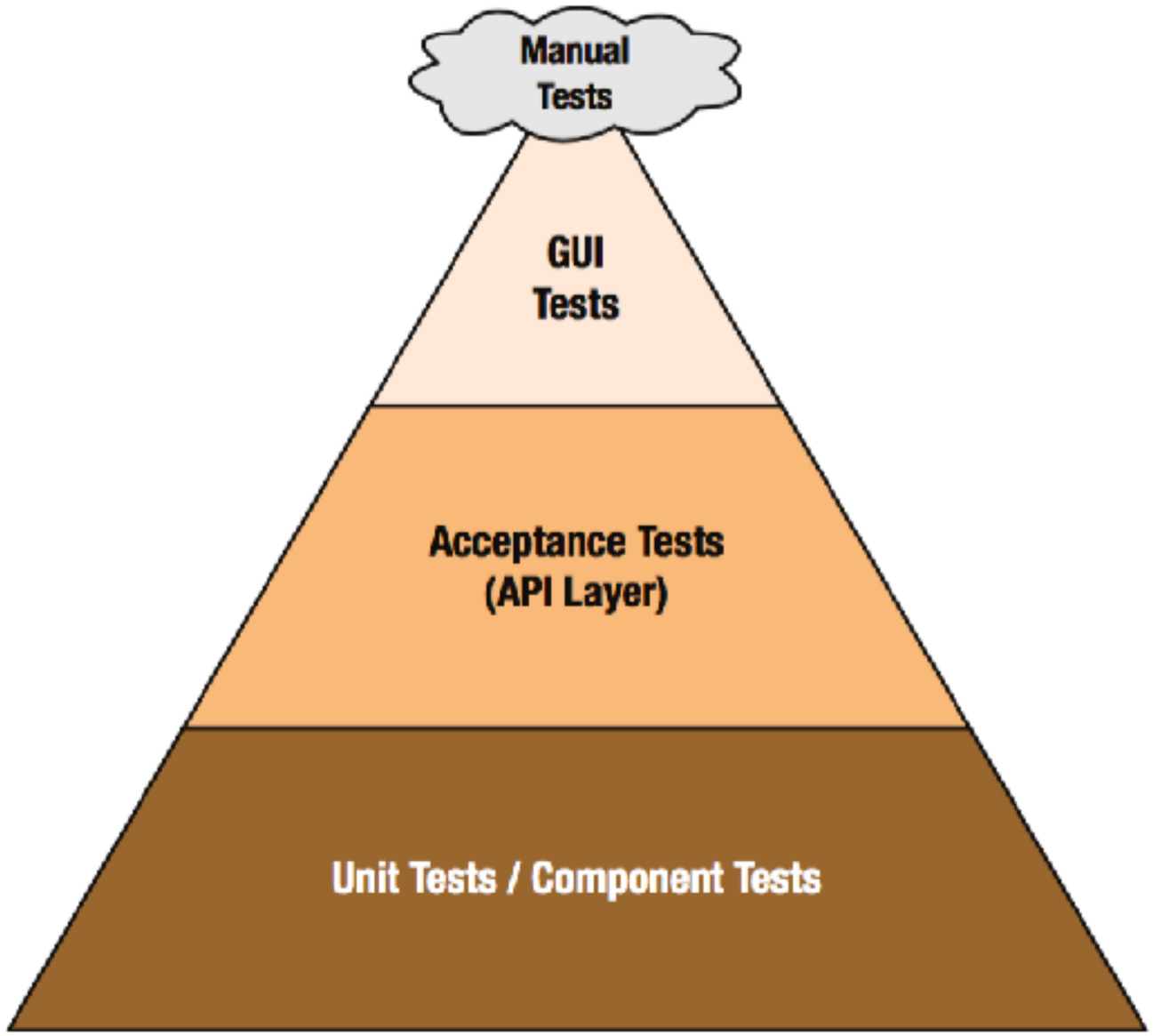
Release with confidence

# Testing

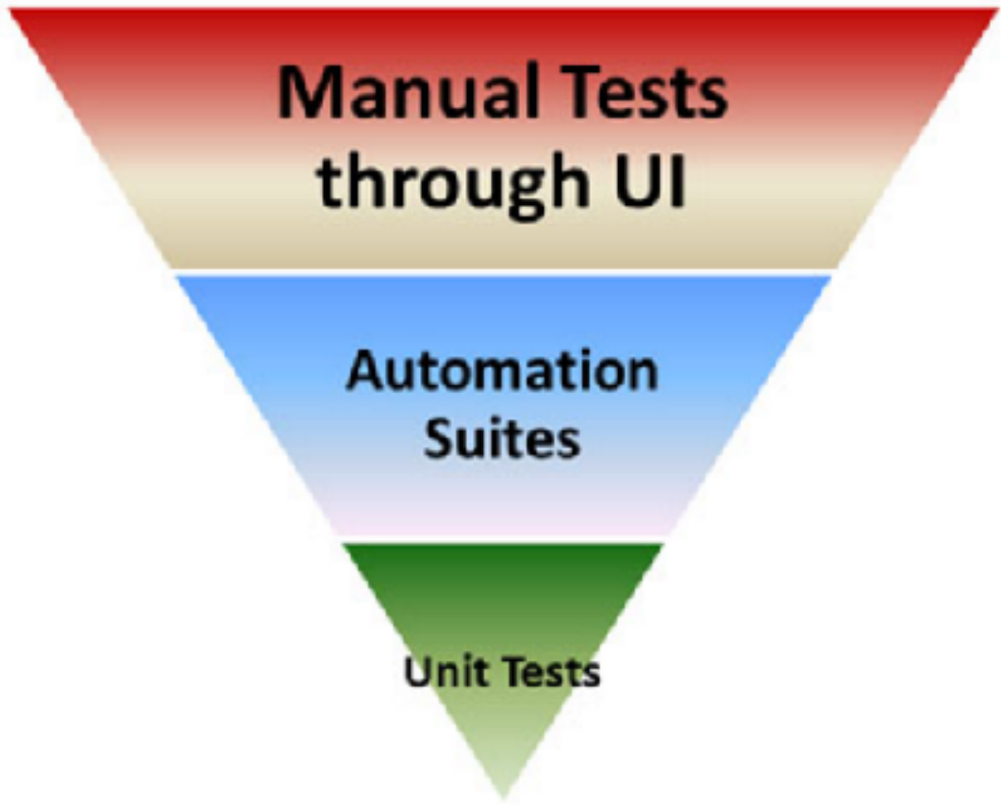
# Testing


Manual Tests
Automated GUI Tests
Integration Tests
Unit Tests
Software Testing Ice-cream Cone Anti-Pattern
watirmelon.com

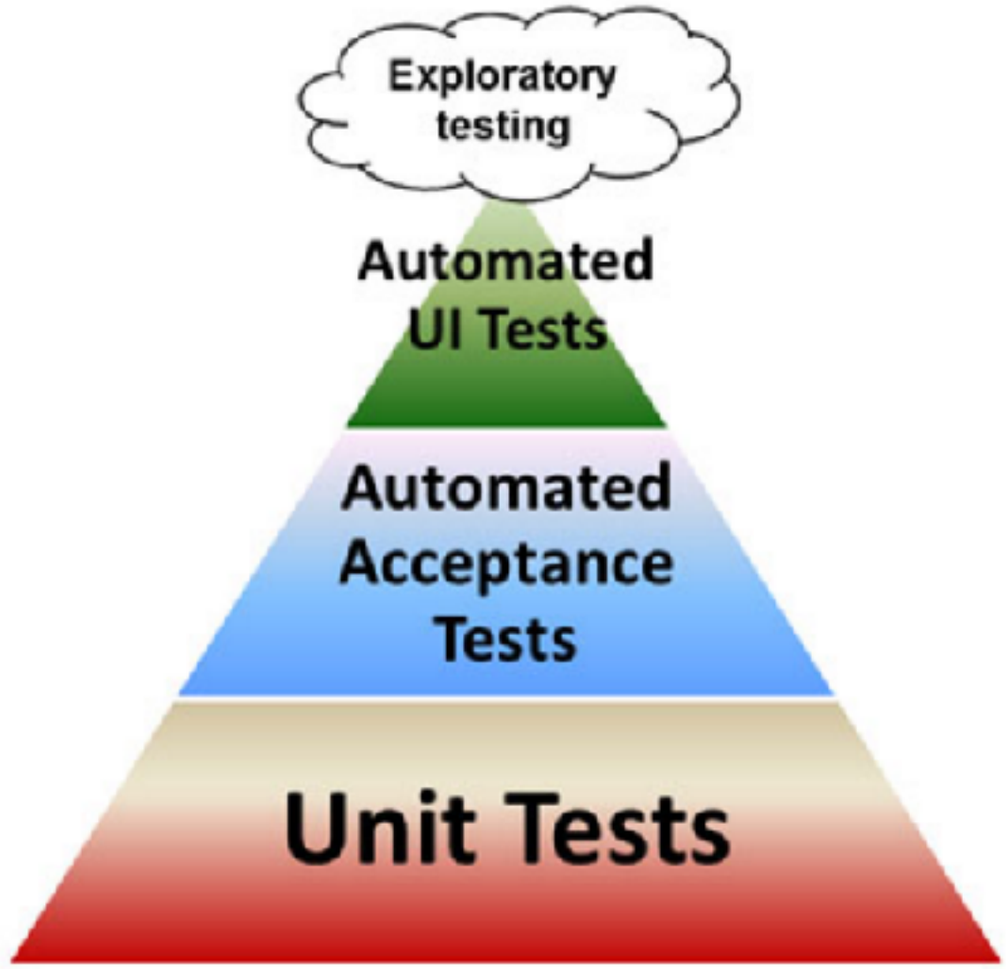
# Testing pyramid



# Testing pyramid

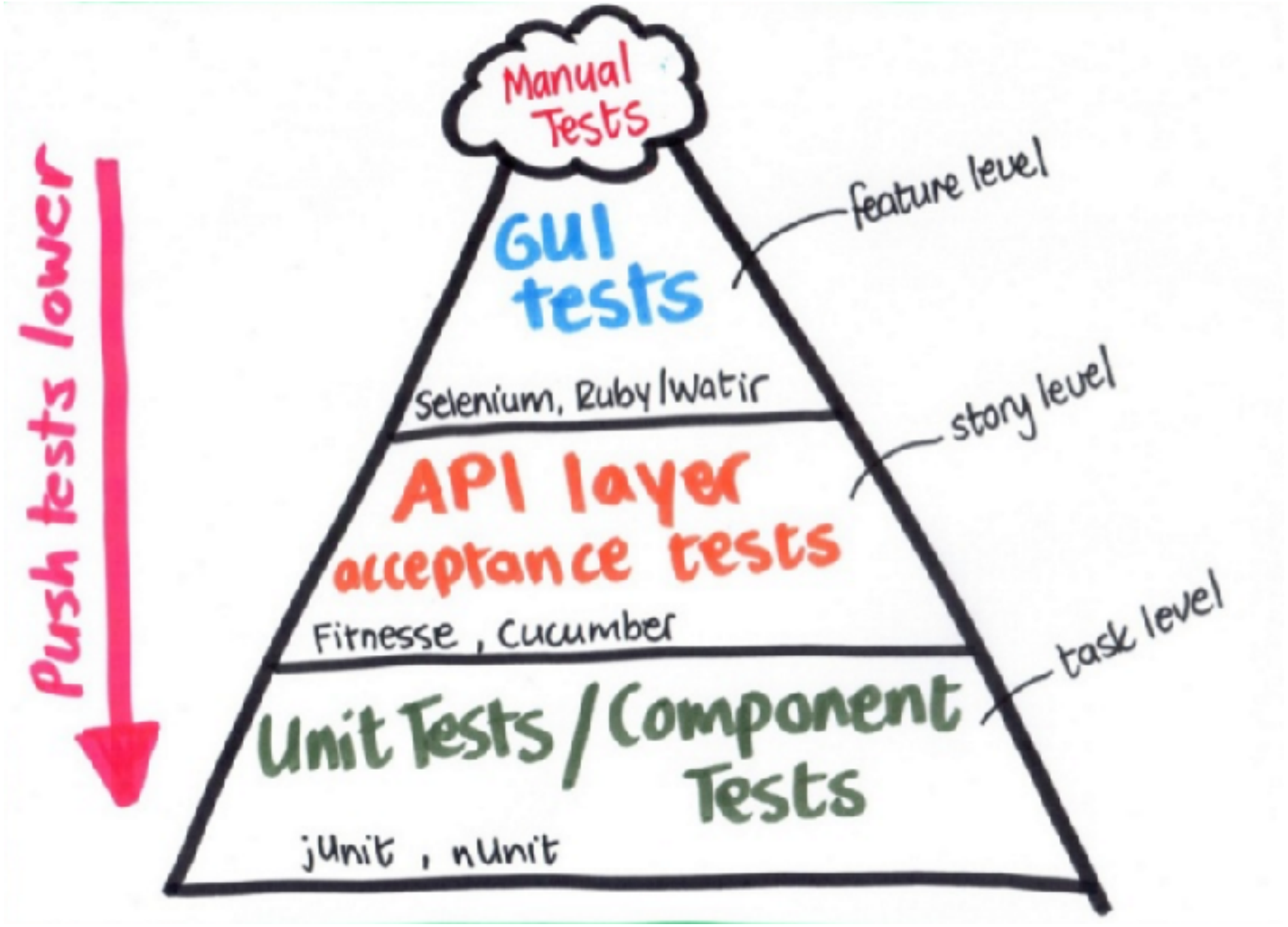



Traditional
(find bugs)

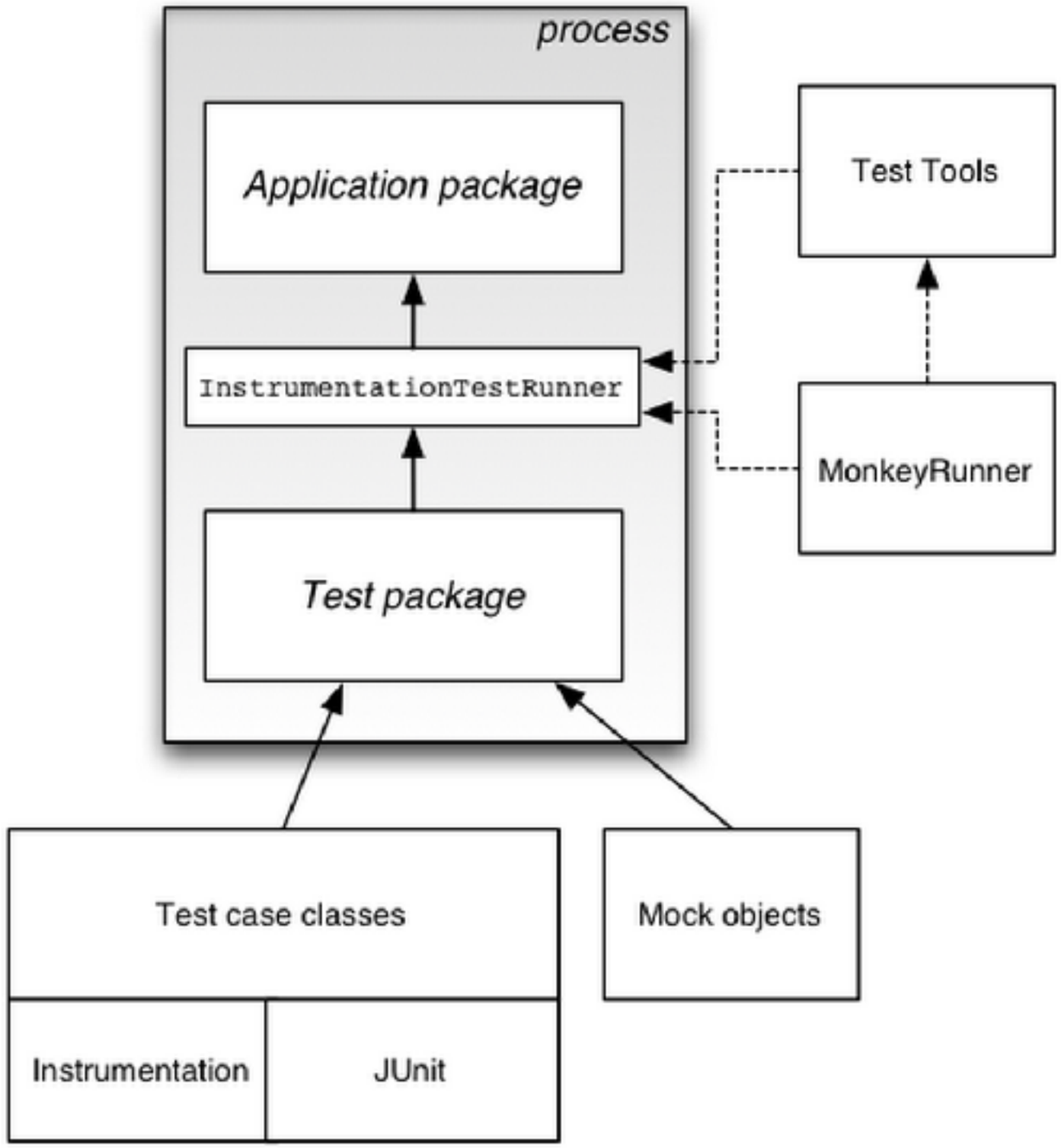



Agile
(prevent bugs)

# Testing pyramid



# Testing for android



# Testing for android

1. ต้องการ Emulator/Device
2. ไม่ต้องการ Emulator/Device

# 3rd-party tools

JUnit

JACOCO
Java Code Coverage

Hamcrest

mockito

espresso
UI TESTING FOR ANDROID


Stubby4J

Jenkins

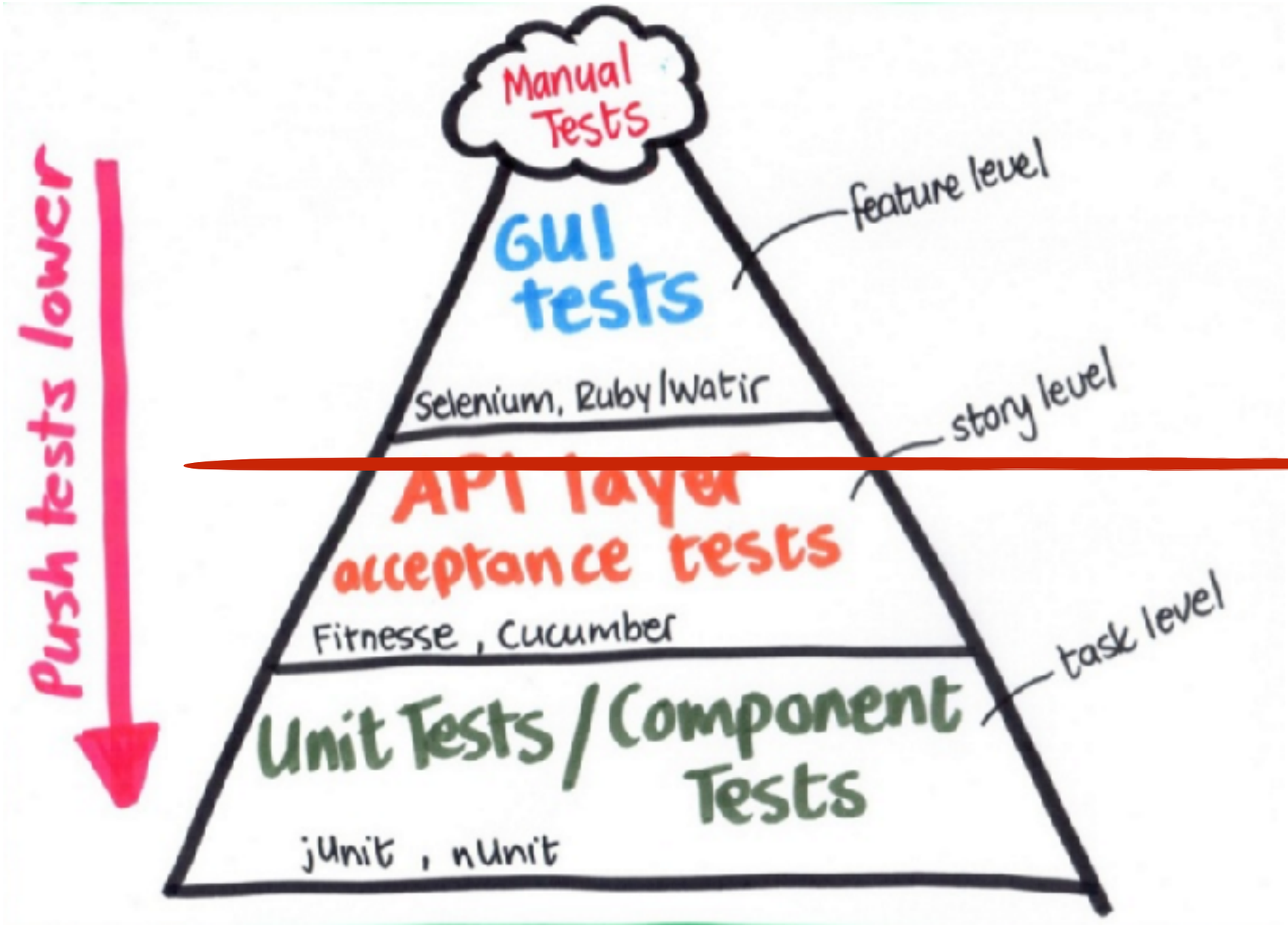
sonarqube

JFrog Artifactory

Supply

Screengrab

# Testing for android



# Test sizes

| Feature | Small | Medium | Large |
|---|---|---|---|
| Network access | No | localhost only | Yes |
| Database | No | Yes | Yes |
| File system access | No | Yes | Yes |
| Use external systems | No | Discouraged | Yes |
| Multiple threads | No | Yes | Yes |
| Sleep statements | No | Yes | Yes |
| System properties | No | Yes | Yes |
| Time limit (seconds) | 60 | 300 | 900+ |

http://googletesting.blogspot.com/2010/12/test-sizes.html

# We need small test !!

| Feature | Small | Medium | Large |
|---|---|---|---|
| Network access | No | localhost only | Yes |
| Database | No | Yes | Yes |
| File system access | No | Yes | Yes |
| Use external systems | No | Discouraged | Yes |
| Multiple threads | No | Yes | Yes |
| Sleep statements | No | Yes | Yes |
| System properties | No | Yes | Yes |
| Time limit (seconds) | 60 | 300 | 900+ |

http://googletesting.blogspot.com/2010/12/test-sizes.html

# Unit testing

Code เพื่อใช้ทดสอบ Code

# Unit testing

ไม่ต้องการ Emulator ในการทดสอบ

เร็วแน่นอน !!

# Why Unit testing

Catch more mistakes
earlier in the development process

# Why Unit testing

Confidently make more changes

# Why Unit testing

Build in regression testing

# Why Unit testing

Extend the life of your codebase

# Good Unit Testing

**F**ast

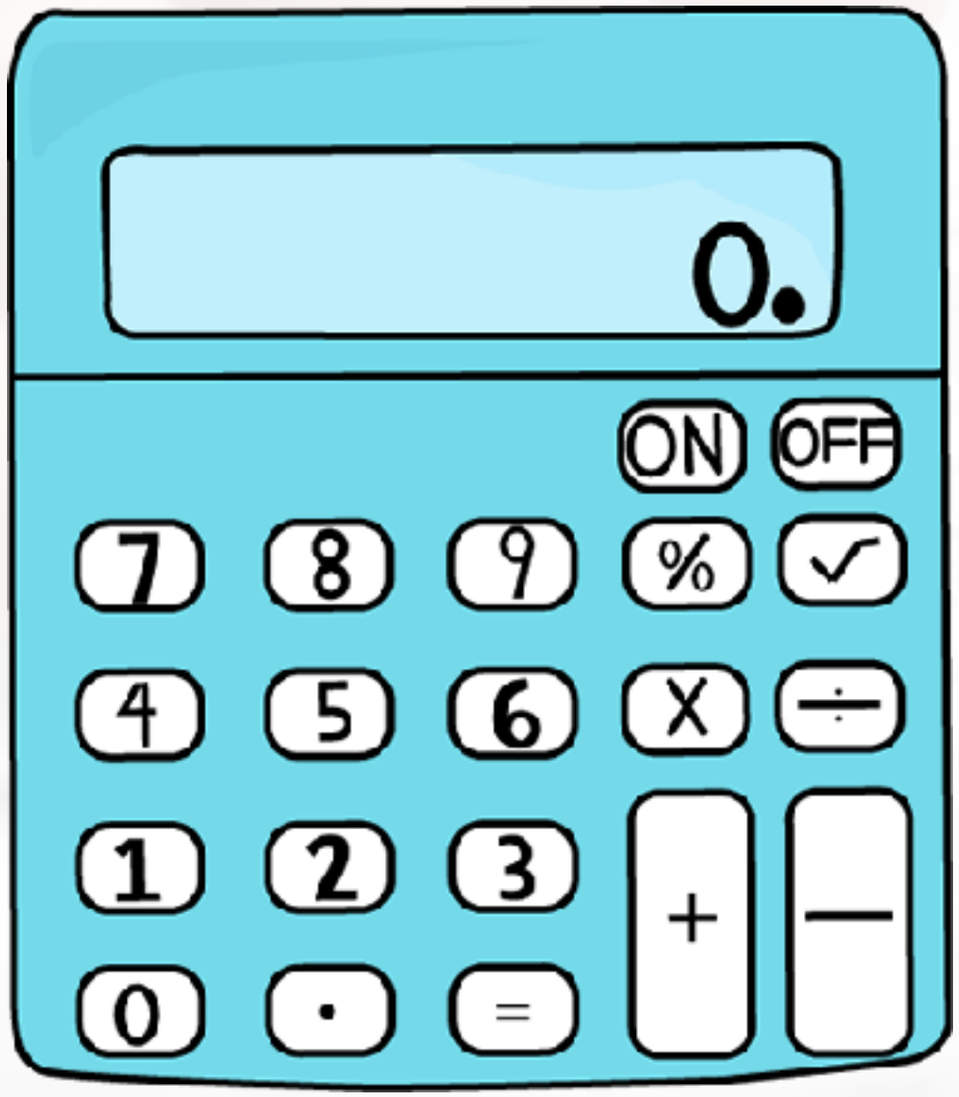
**I**solated

**R**epeatable

**S**elf-verifying

**T**imely

# Workshop



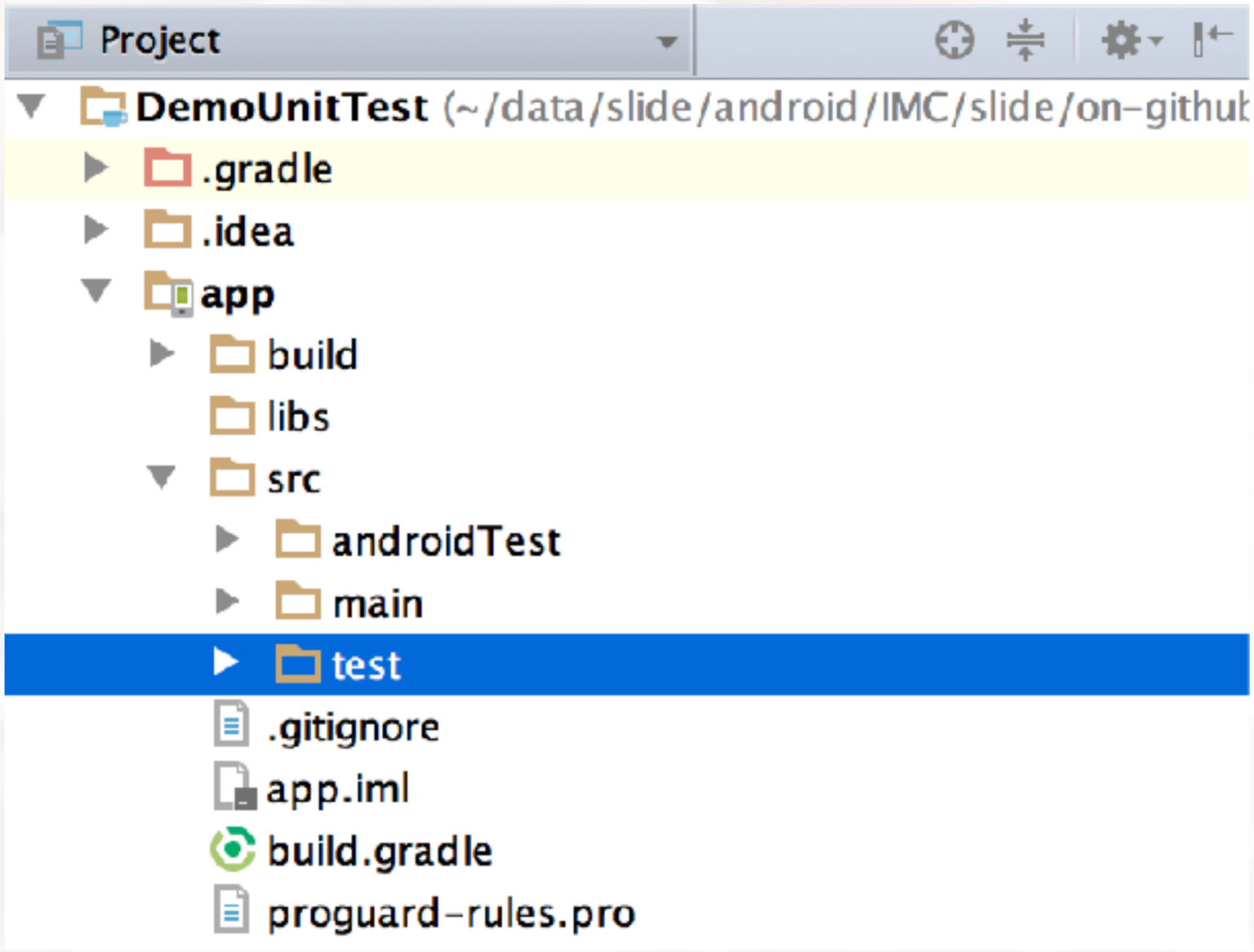
# Workshop

สร้าง project ใหม่ ชื่อ **Calculator**

เรียนรู้การสร้าง Unit test (jUnit4)

Code coverage

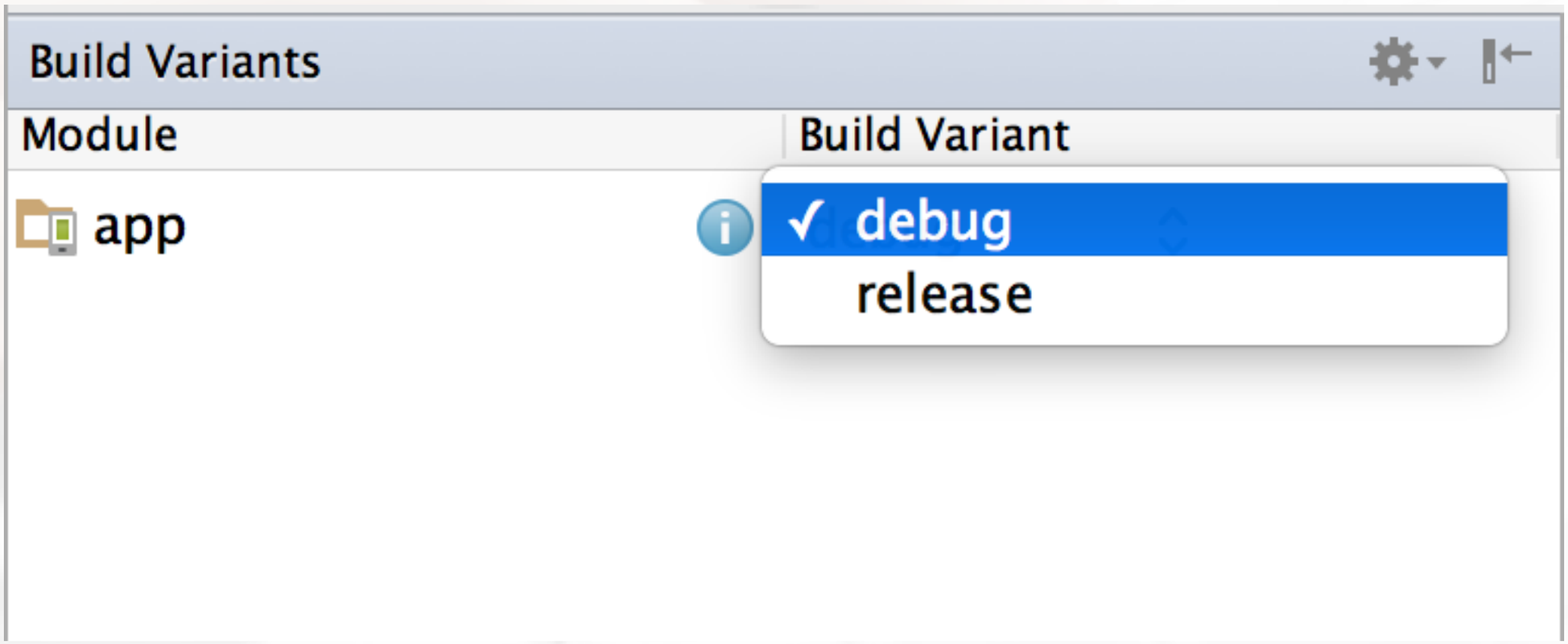
# Project structure



# build.gradle

```
dependencies {
    compile fileTree(dir: 'libs', include
    testCompile 'junit:junit:4.12'
    compile 'com.android.support:appcompa
}
```

# Build variant



# First unit test

```
public class CalculatorUnitTest {

    @Test
    public void addition_isCorrect() throws Exception {
        //Arrange
        Calculator calculator = new Calculator();
        //Act
        double actualResultOfAdd = calculator.add(1, 2);
        //Assert
        assertEquals(3, actualResultOfAdd, 0);
    }
}
```

# Naming test class and method

Express what the class should be able to do.

```
public class MyClassShould {

    @Test public void
    do_something_interesting() {

    }

}
```

Reads as a full sentence

Start with a verb.

# Naming test class and method

```
public class BankAccountShould{

    @Test public void
    have_balance_of_zero_when_created() {
        BankAccount bankAccount = new BankAccount();

        assertThat(bankAccount.balance(), is(0));
    }

    @Test public void
    have_the_balance_increased_after_a_deposit() {
given   BankAccount bankAccount = new BankAccount();

when    bankAccount.deposit(10);

then    assertThat(bankAccount.balance(), is(10));
    }

}
```

# Test creation order

Step 1: Name the class

```
public class BankAccountShould{

    @Test public void
    have_balance_increased_after_a_deposit() {
```

Step 2: Name the method

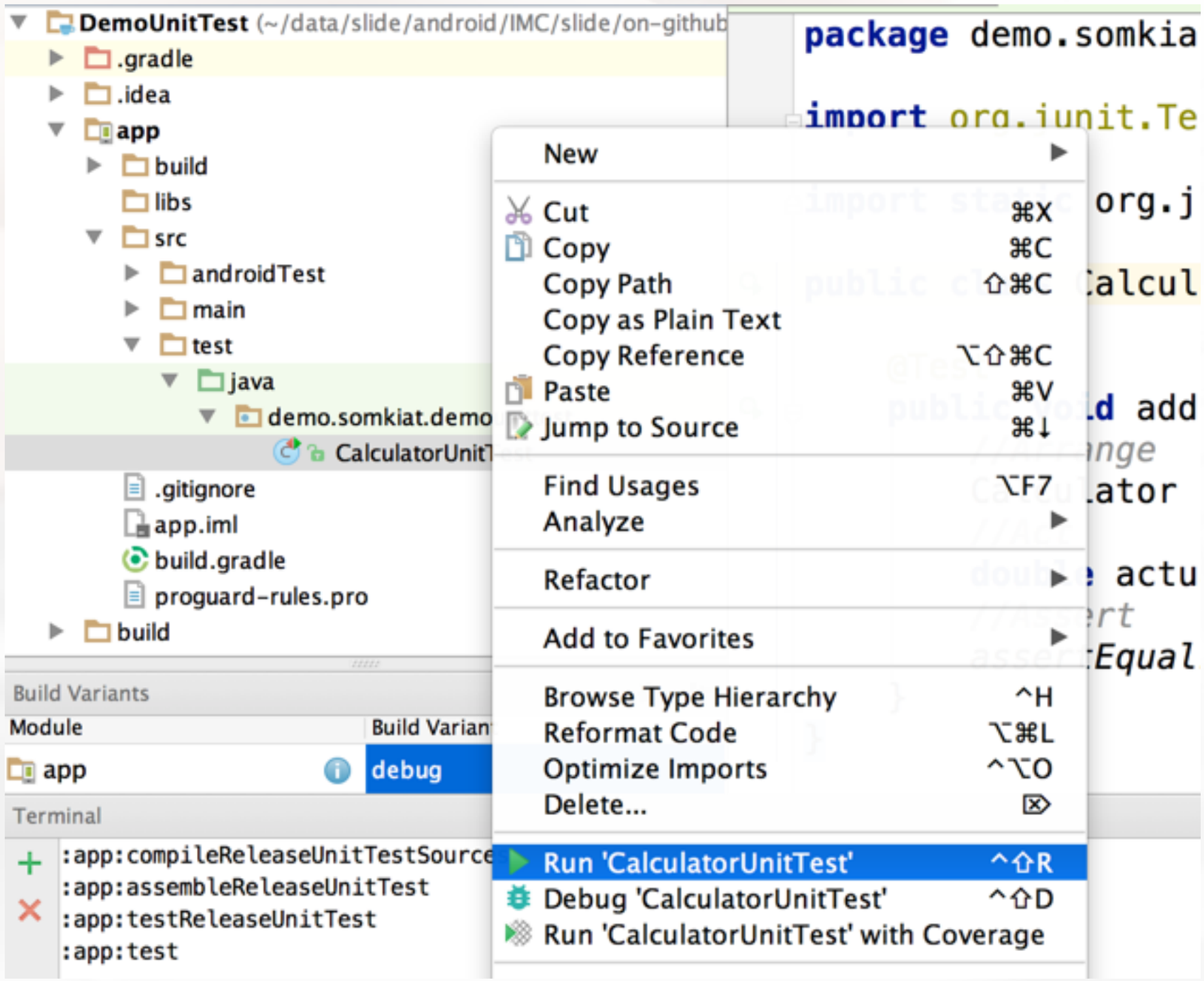
Step 5: Setup

```
given   BankAccount bankAccount = new BankAccount();

when    bankAccount.deposit(10);

then    assertThat(bankAccount.balance(), is(10));

    }

}
```

Step 4: Trigger the code

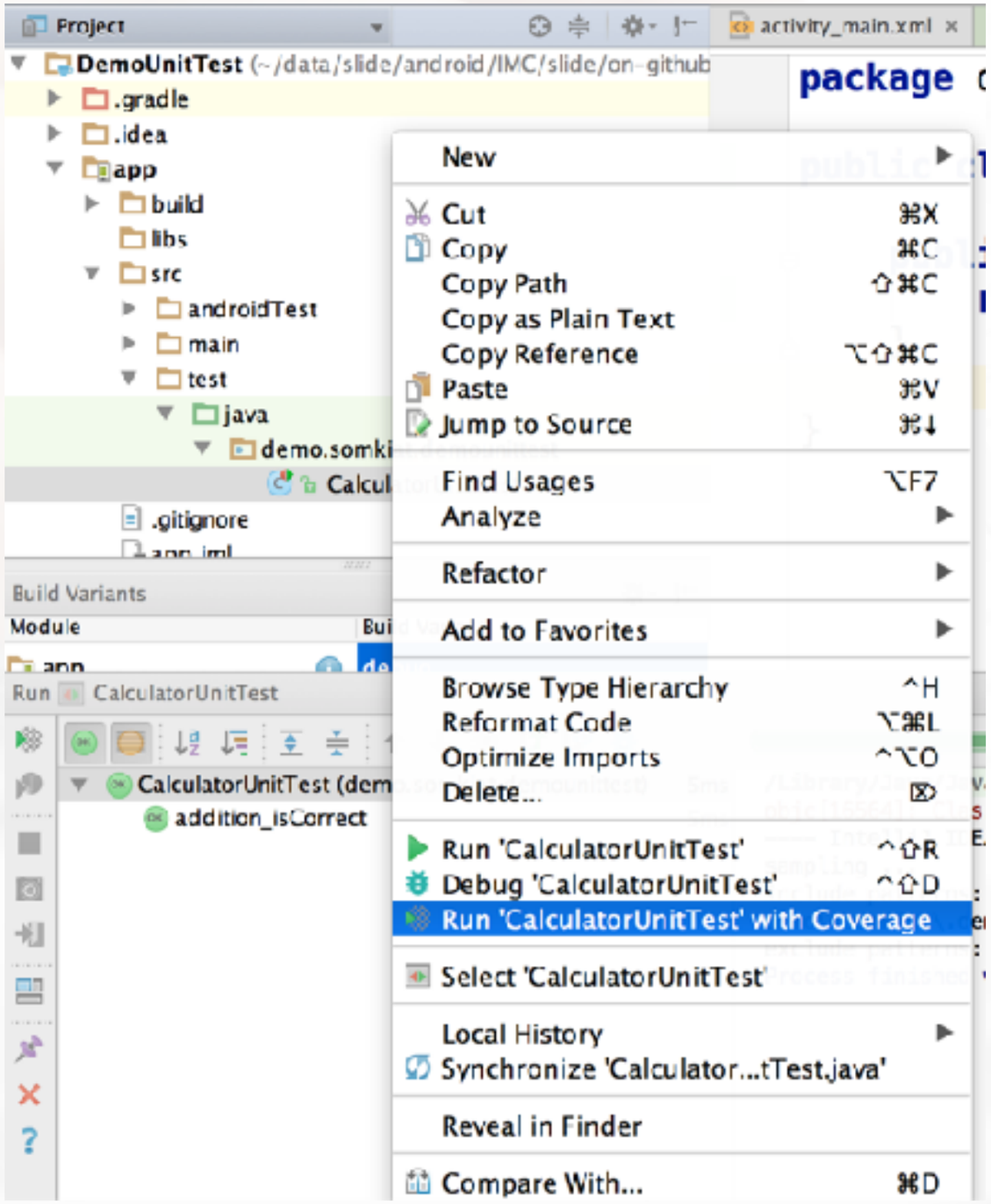
Step 3: Define what you are testing

# Run unit test

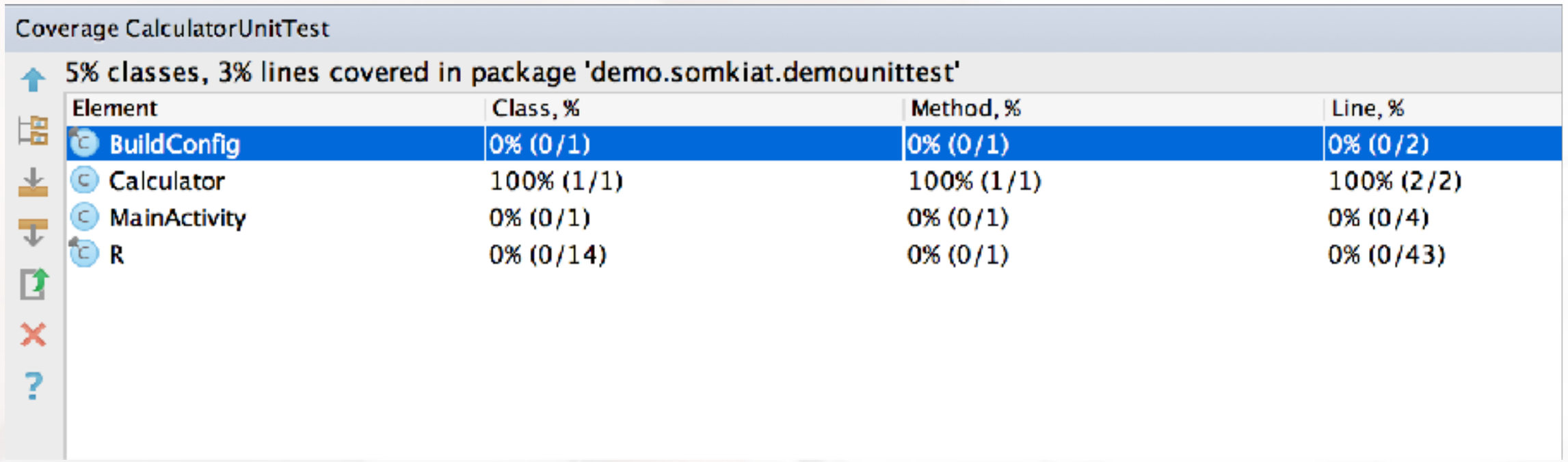


# Run unit test

# Run unit test with coverage



# Run unit test with coverage



# Run unit test

$./gradlew test --continue

# Show report

## Test Summary

| 1 | 0 | 0 | 0.002s |
|---|---|---|---|
| tests | failures | ignored | duration |

**100%**
successful

**Packages** Classes

| Package | Tests | Failures | Ignored | Duration | Success rate |
|---|---|---|---|---|---|
| demo.somkiat.demounittest | 1 | 0 | 0 | 0.002s | 100% |

Generated by Gradle 2.10 at May 7, 2016, 2:50:21 PM

# Run unit test with coverage

$./gradlew createDebugCoverageReport

# Enable code coverage

## build.gradle

```
buildTypes {
    debug {
        testCoverageEnabled = true
    }
    release {
        minifyEnabled false
        proguardFiles getDefaultProgu
    }
}
```

# Show report

debug > demo.somkiat.demounittest

## demo.somkiat.demounittest

| Element | Missed Instructions | Cov. | Missed Branches | Cov. | Missed | Cxty | Mi |
|---|---|---|---|---|---|---|---|
| MainActivity | | 0% | | n/a | 2 | 2 | |
| Calculator | | 0% | | n/a | 2 | 2 | |
| Total | 17 of 17 | 0% | 0 of 0 | n/a | 4 | 4 | |

# Run unit test with coverage

$./gradlew createDebugCoverageReport

Working with instrumented test (AndroidTest) !!

# Run unit test with coverage

Let’s start to hack jacoco !!

# Enable coverage for unit test

## build.gradle

```
apply plugin: 'jacoco'

task jacocoTestReport(type: JacocoReport, dependsOn:

    reports {
        xml.enabled = true
        html.enabled = true
    }

    jacocoClasspath = configurations['androidJacocoA

    def fileFilter = ['**/R.class', '**/R$*.class',
    def debugTree = fileTree(dir: "${buildDir}/inter
    def mainSrc = "${project.projectDir}/src/main/ja

    sourceDirectories = files([mainSrc])
    classDirectories = files([debugTree])
    executionData = files("${buildDir}/jacoco/testDe
```

# Run unit test with coverage

```
$./gradlew jacocoTestReport
```

# Show report

app > demo.somkiat.demounittest

## demo.somkiat.demounittest

| Element | Missed Instructions | Cov. | Missed Branches | Cov. | Missed | Cxty | Missed | Lines | Miss |
|---|---|---|---|---|---|---|---|---|---|
| MainActivity | | 0% | | n/a | 2 | 2 | 4 | 4 | |
| Calculator | | 100% | | n/a | 0 | 2 | 0 | 2 | |
| Total | 10 of 17 | 41% | 0 of 0 | n/a | 2 | 4 | 4 | 6 | |

# Jacoco android gradle

## jacoco-android-gradle-plugin

build passing | Codecov 98% | Download 0.1.1

A Gradle plugin that adds fully configured `JacocoReport` tasks for unit tests of each Android application and library project variant.

## Why

In order to generate JaCoCo unit test coverage reports for Android projects you need to create `JacocoReport` tasks and configure them by providing paths to source code, execution data and compiled classes. It can be troublesome since Android projects can have different flavors and build types thus requiring additional paths to be set. This plugin provides those tasks already configured for you.

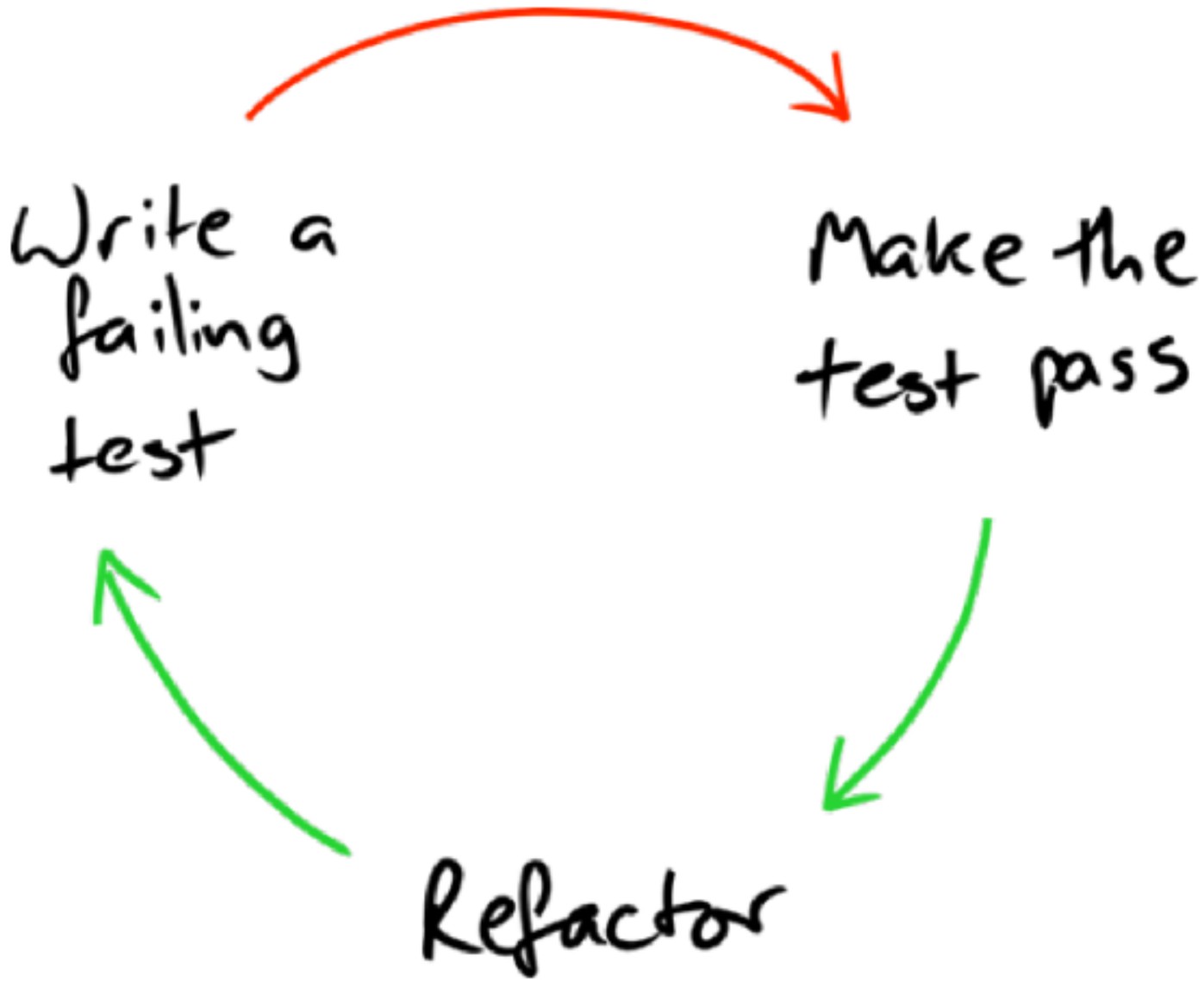
https://github.com/arturdm/jacoco-android-gradle-plugin

# แบบฝึกหัด

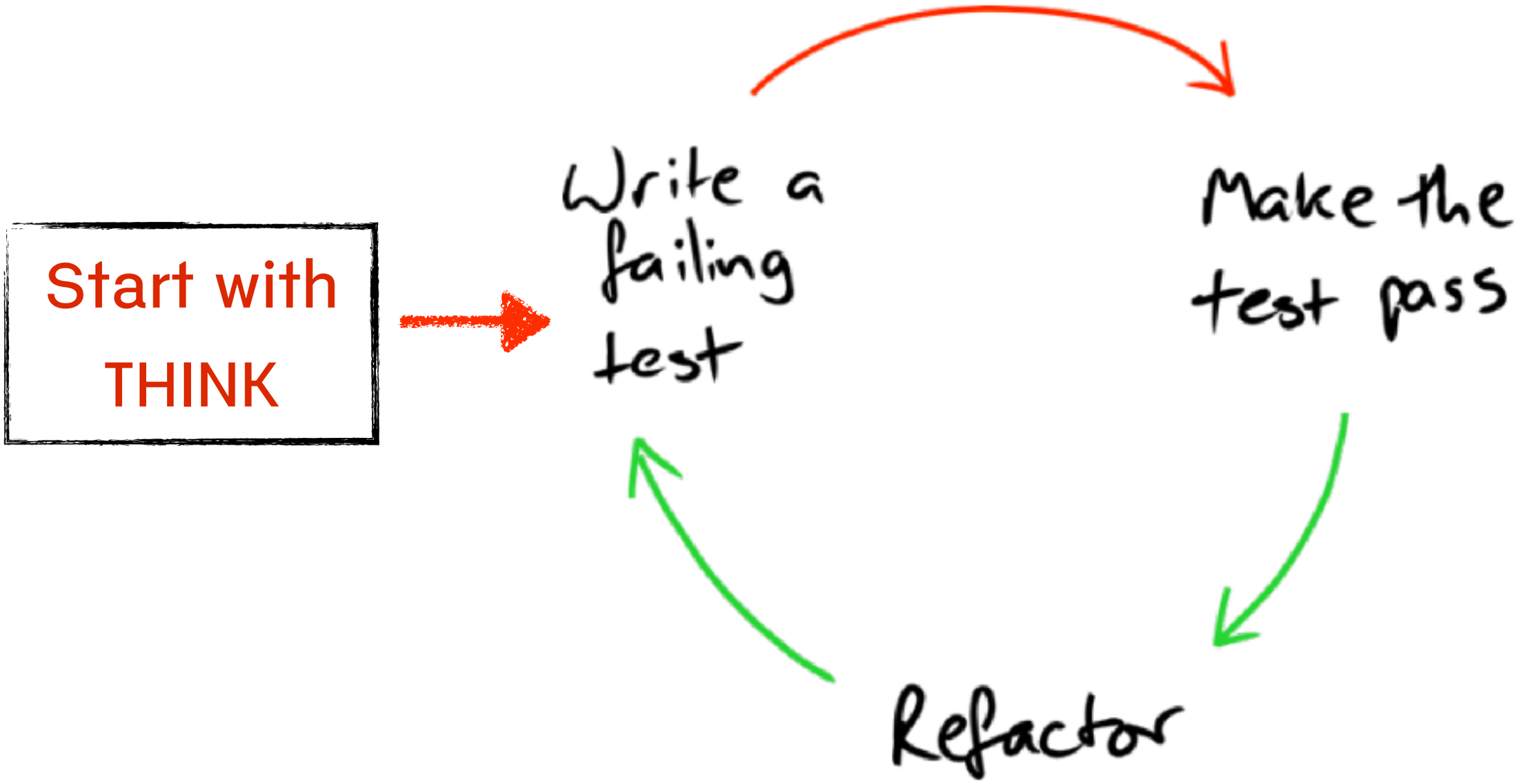
## บวก ลบ คูณ หาร

## Code coverage = 100% !!

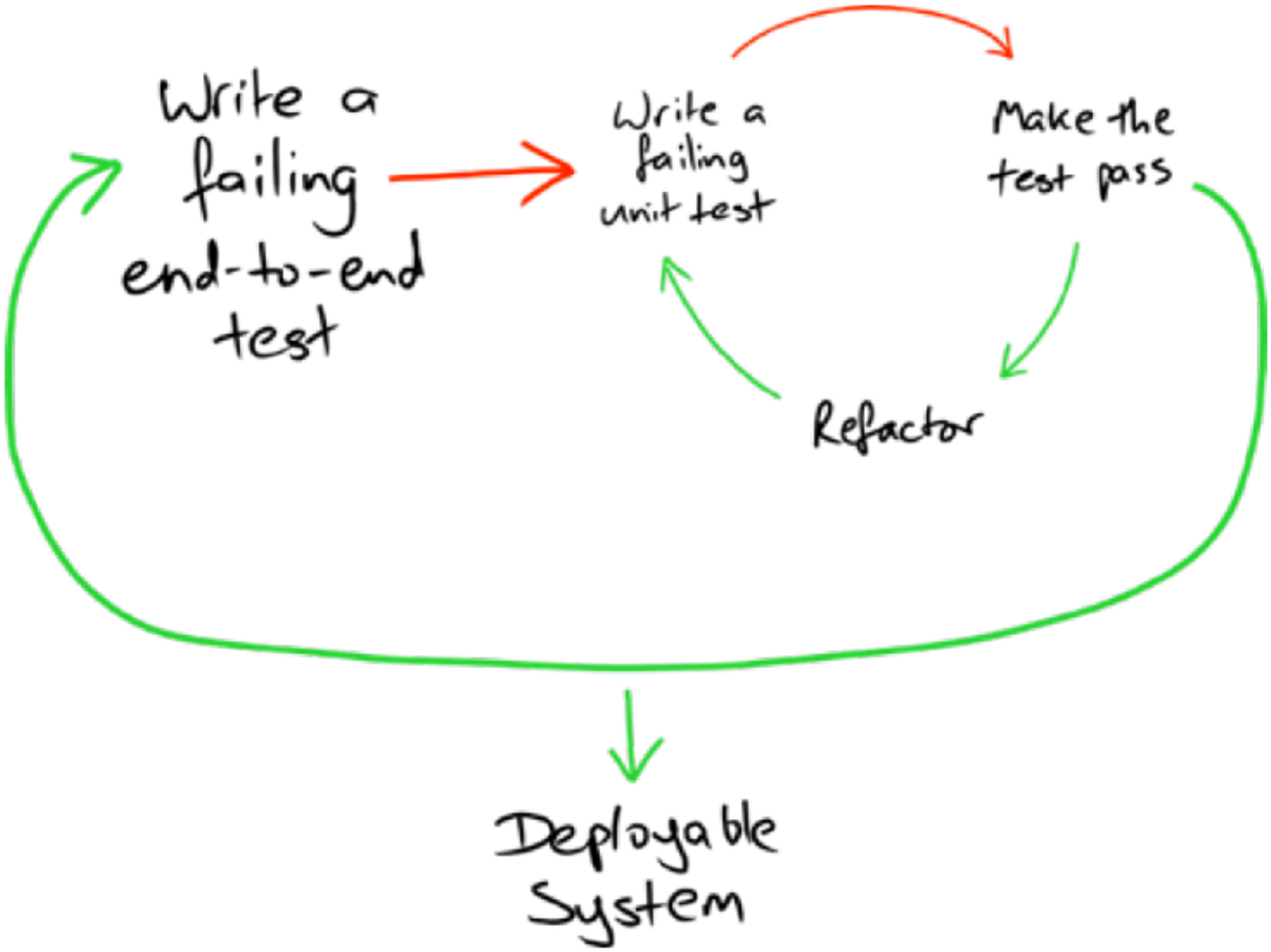
# ทำดีดี



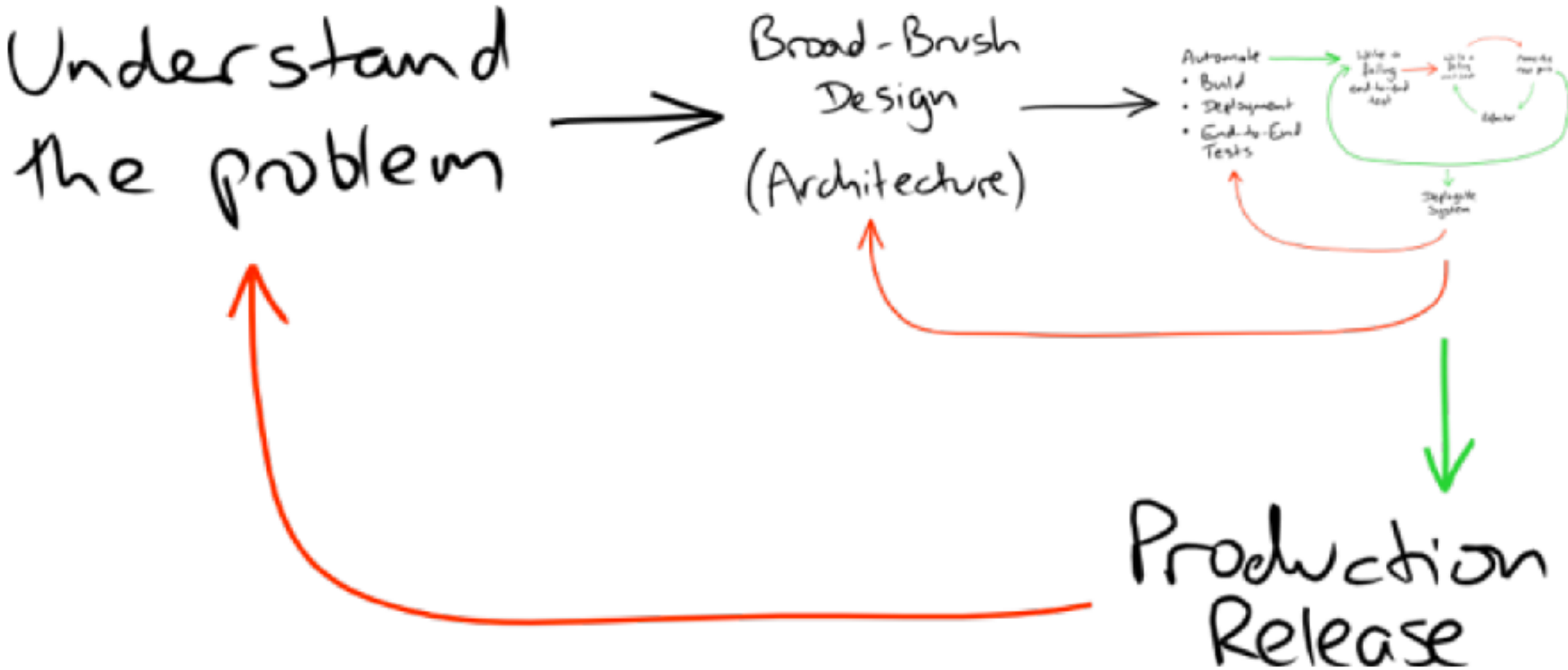
# คิดก่อนทำดีดี


Start with
THINK
Write a
failing
test
Make the
test pass
Refactor

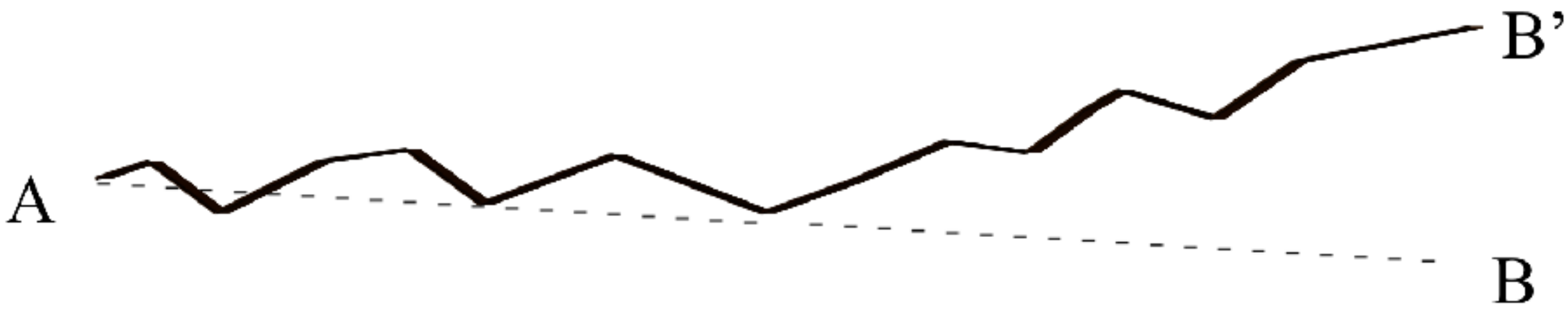
# ACCEPTANCE TEST DRIVEN DEVELOPMENT



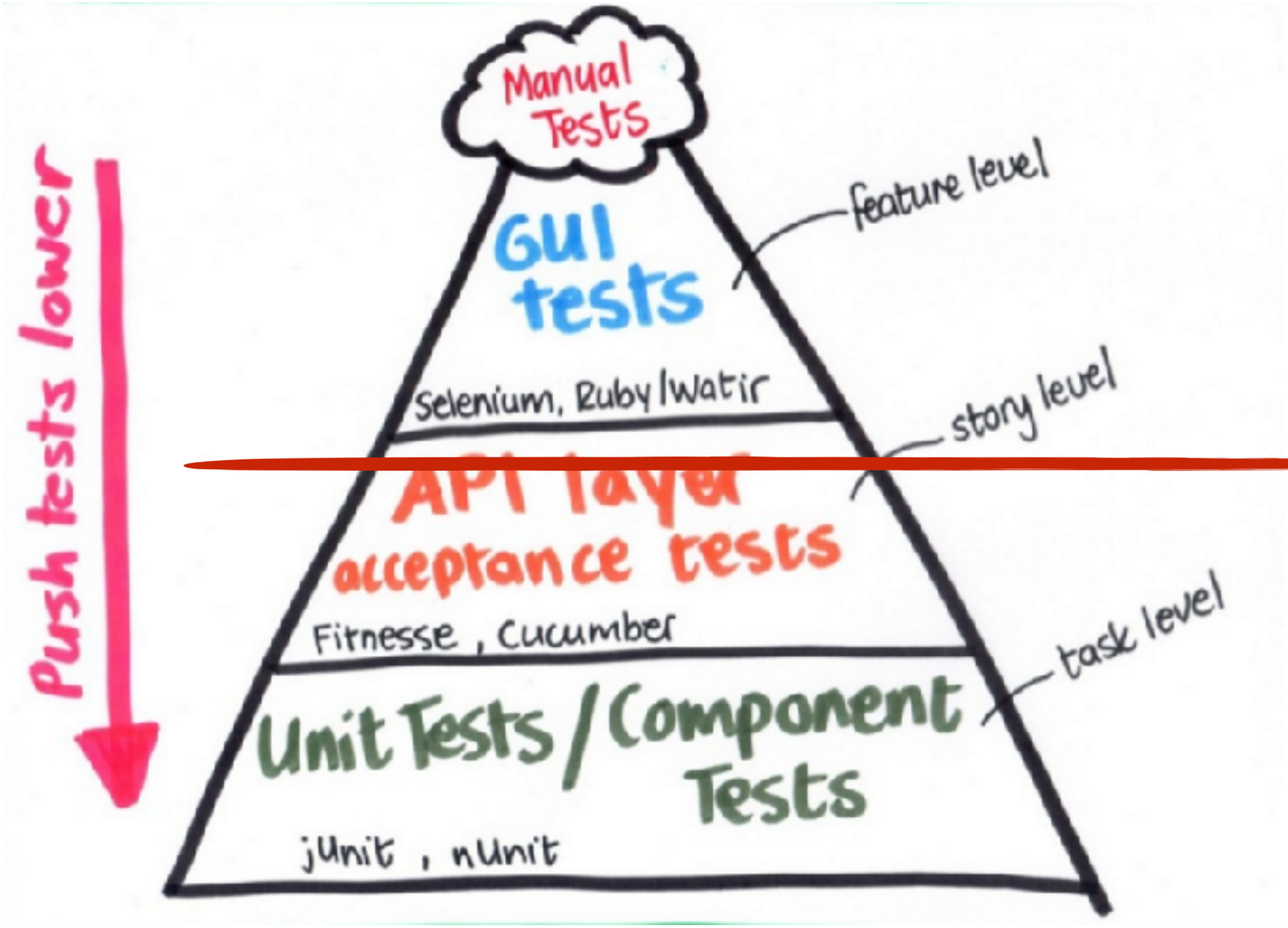
# LARGER FEEDBACK LOOP



# Small step



# Testing for android



# UI testing

ทำให้แน่ใจว่า App ใช้งานได้
บน device ต่าง ๆ

# UI testing

**Looks good but can’t use !!**

# UI testing

ต้องการทดสอบการทำงานของ Activity

ใช้เวลาการทดสอบนาน

จำนวนการทดสอบมีเท่าที่จำเป็น

# UI testing frameworks


UI Automator


Robotium

# UI testing


https://google.github.io/android-testing-support-library/docs/espresso/index.html

# Espresso ?

Android Testing Support Library

Simulate user interactions

Automatic synchronization of test action with app UI

# Espresso components

ViewMatcher

ViewAction

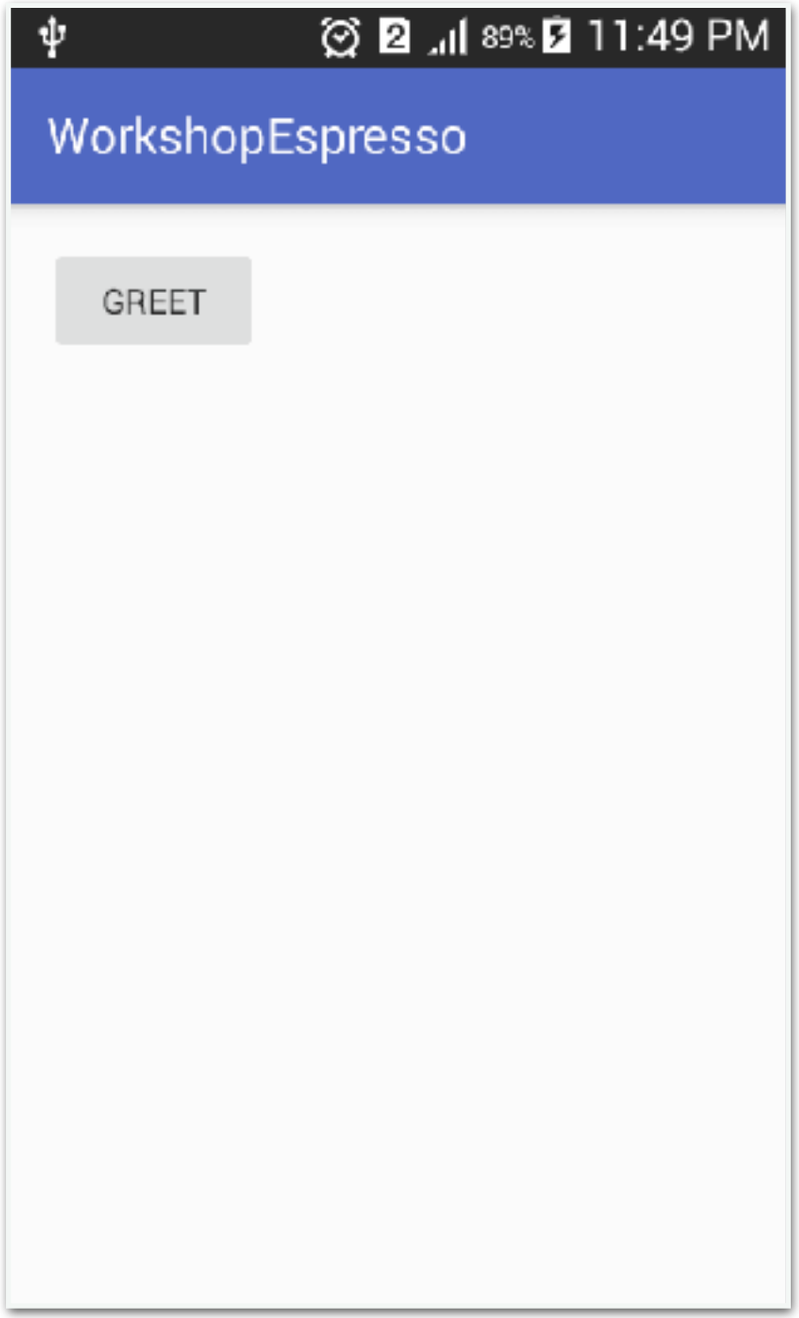
ViewAssertion

# Espresso

```
onView(ViewMatcher)
    .perform(ViewAction)
    .check(ViewAssertion)
```

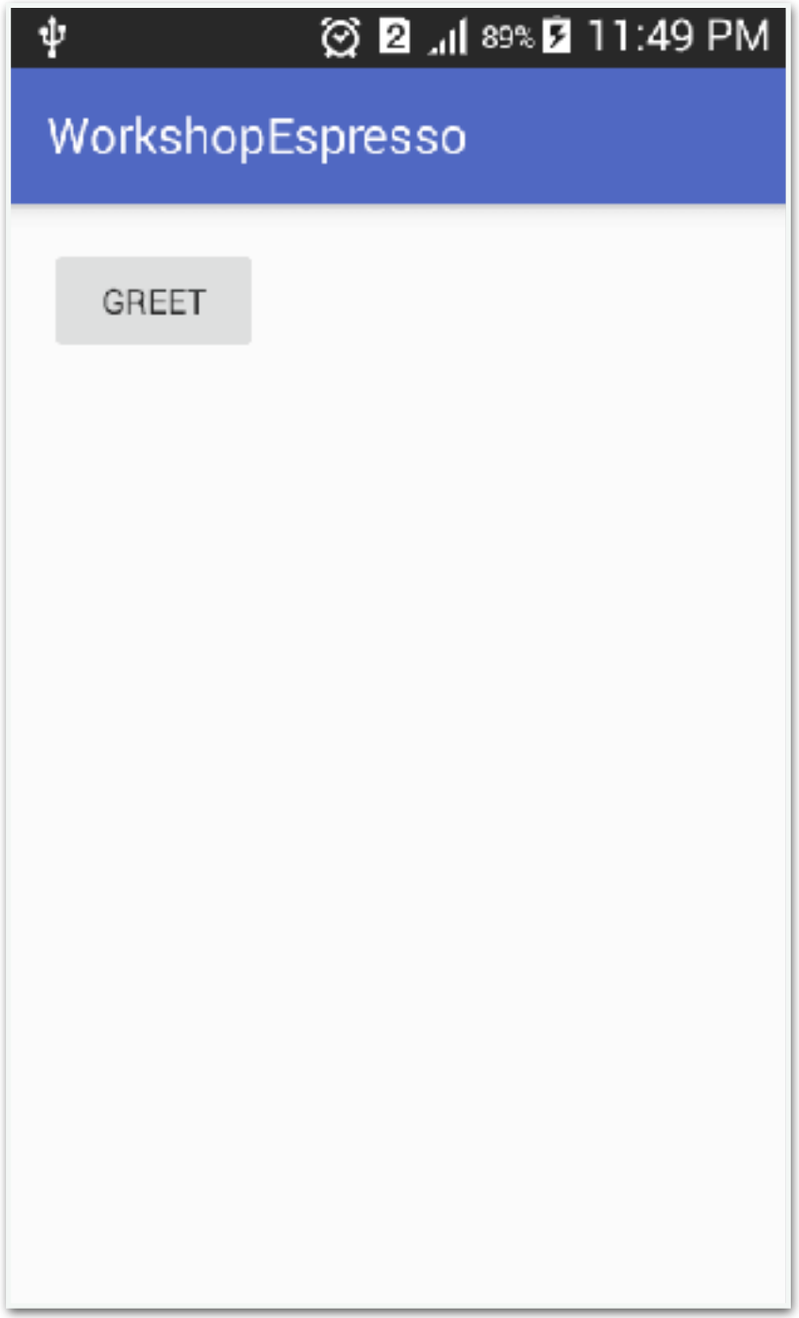
# Hello Espresso

```
onView(ViewMatcher)
      .perform(ViewAction)
      .check(ViewAssertion)
```



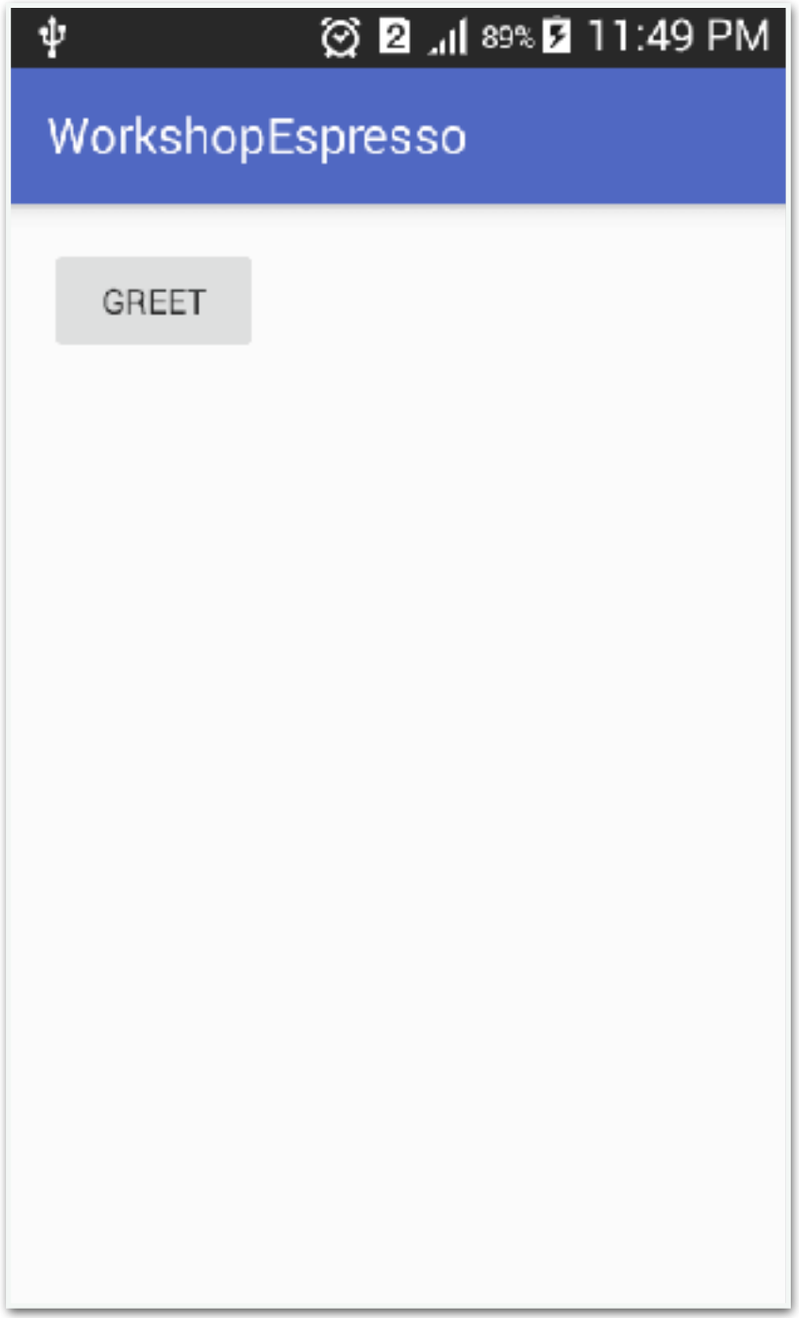
# ViewMatcher

```
onView(withId(R.id.greet_button))
      .perform(ViewAction)
      .check(ViewAssertion)
```



# ViewAction

```
onView(withId(R.id.greet_button))
      .perform(click())
      .check(ViewAssertion)
```

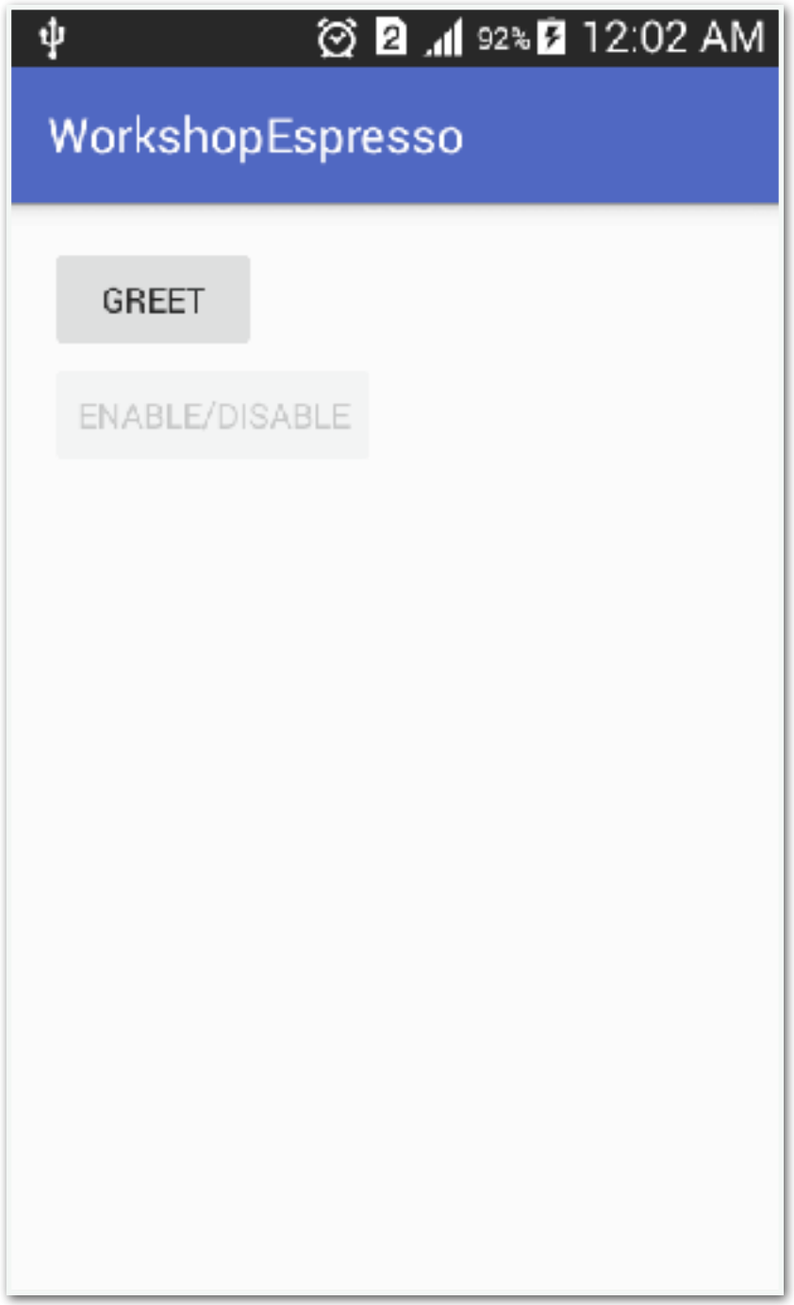


# ViewAssertion

```
onView(withId(R.id.greet_button))
    .perform(click())
    .check(matches(withText("สวัสดี")))
```

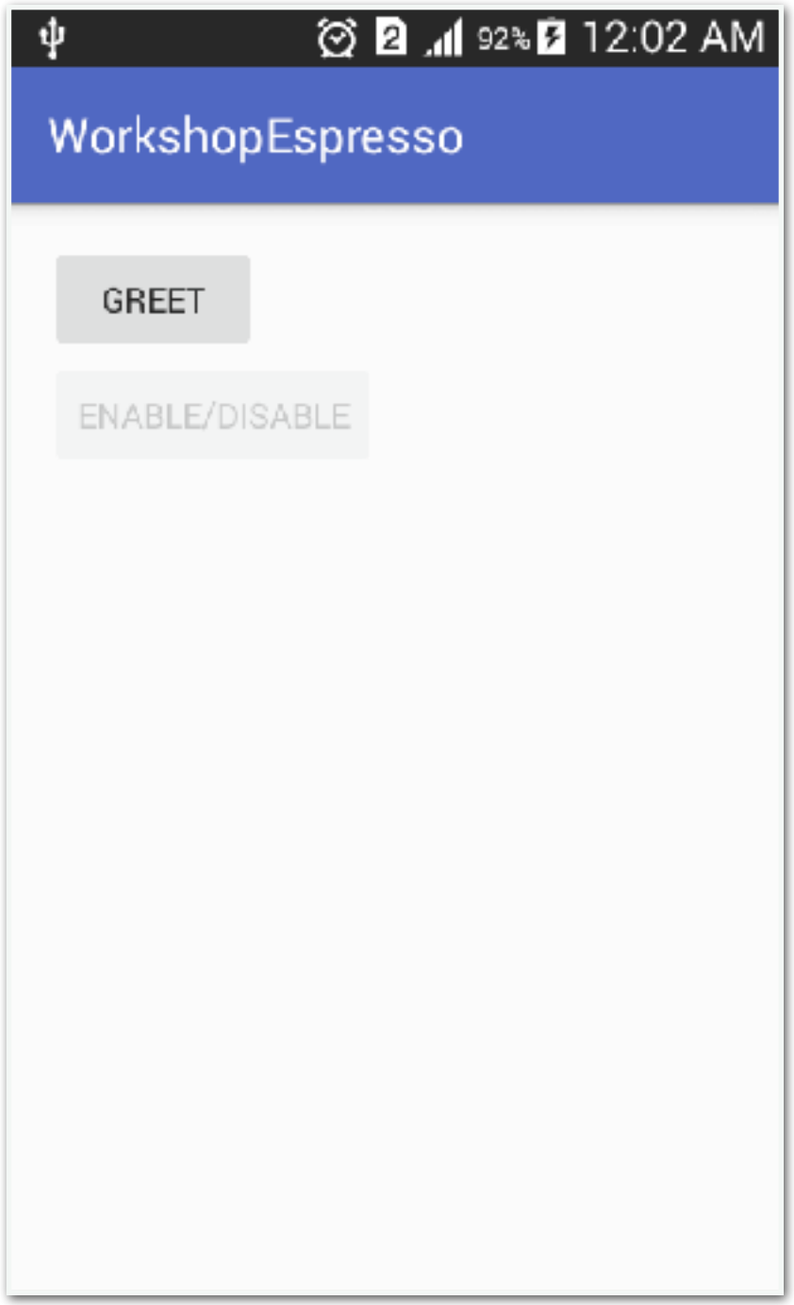
# ViewAssertion

```
onView(withId(R.id.state_button))
      .perform(click())
      .check(matches(not(isEnabled())))
```



# Hamcrest

```
onView(withId(R.id.state_button))
      .perform(click())
      .check(matches(not(isEnabled())))
```



# Espresso cheatsheet

## View Matchers

**USER PROPERTIES**

```
withId(...)
withText(...)
withTagKey(...)
withTagValue(...)
hasContentDescription(...)
withContentDescription(...)
withHint(...)
withSpinnerText(...)
hasLinks()
hasEllipsizedText()
hasMultilineText()
```

**HIERARCHY**

```
withParent(Matcher)
withChild(Matcher)
hasDescendant(Matcher)
isDescendantOfA(Matcher)
hasSibling(Matcher)
isRoot()
```

**INPUT**

```
supportsInputMethods(...)
hasImeAction(...)
```

**UI PROPERTIES**

```
isDisplayed()
isCompletelyDisplayed()
isEnabled()
hasFocus()
isClickable()
isChecked()
isNotChecked()
withEffectiveVisibility(...)
isSelected()
```

**CLASS**

```
isAssignableFrom(...)
withClassName(...)
```

**ROOT MATCHERS**

```
isFocusable()
isTouchable()
isDialog()
withDecorView()
isPlatformPopup()
```

**OBJECT MATCHER**

```
allOf(Matchers)
anyOf(Matchers)
is(...)
not(...)
endsWith(String)
startsWith(String)
instanceOf(Class)
```

**SEE ALSO**

```
Preference matchers
Cursor matchers
Layout matchers
```

## Data Options

```
inAdapterView(Matcher)
atPosition(Integer)
onChildView(Matcher)
```

## View Actions

**CLICK/PRESS**

```
click()
doubleClick()
longClick()
pressBack()
pressIMEActionButton()
pressKey([int/EspressoKey])
pressMenuKey()
closeSoftKeyboard()
openLink()
```

**GESTURES**

```
scrollTo()
swipeLeft()
swipeRight()
swipeUp()
swipeDown()
```

**TEXT**

```
clearText()
typeText(String)
typeTextIntoFocusedView(String)
replaceText(String)
```

## View Assertions

```
matches(Matcher)
doesNotExist()
selectedDescendantsMatch(...)
```

**LAYOUT ASSERTIONS**

```
noEllipsizedText(Matcher)
noMultiLineButtons()
noOverlaps([Matcher])
```

**POSITION ASSERTIONS**

```
isLeftOf(Matcher)
isRightOf(Matcher)
isLeftAlignedWith(Matcher)
isRightAlignedWith(Matcher)
isAbove(Matcher)
isBelow(Matcher)
isBottomAlignedWith(Matcher)
isTopAlignedWith(Matcher)
```

https://google.github.io/android-testing-support-library/docs/espresso/cheatsheet/index.html

# Hamcrest cheatsheet

### General purpose

```
is(T)
equalTo(T)
not(T)                                   : Matcher<T>

anything()
anything(String)                         : Matcher<Object>

any(Class<T>)
instanceOf(Class<?>)
isA(Class<T>)                            : Matcher<T>

nullValue()                              : Matcher<Object>
nullValue(Class<T>)                      : Matcher<T>
notNullValue()                           : Matcher<Object>
notNullValue(Class<T>)                   : Matcher<T>

sameInstance(T)
theInstance(T)                           : Matcher<T>

isIn(Collection<T>)
isIn(T[])
isOneOf(T...)
hasToString(String)
hasToString(Matcher<? super String>)     : Matcher<T>
```

### Combining multiple matchers

```
is(Matcher<T>)
not(Matcher<T>)                          : Matcher<T>

allOf(Matcher<? super T>...)
allOf(Iterable<Matcher<? super T>>)
anyOf(Matcher<? super T>...)
anyOf(Iterable<Matcher<? super T>>)      : Matcher<T>

both(Matcher<? super LHS>)
either(Matcher<? super LHS>)             : Matcher<LHS>

describedAs(String, Matcher<T>, Object...)  : Matcher<T>
```

### Strings

```
containsString(String)
startsWith(String)
endsWith(String)                         : Matcher<String>

equalToIgnoringCase(String)
equalToIgnoringWhiteSpace(String)        : Matcher<String>

isEmptyString()
isEmptyOrNullString()                    : Matcher<String>

stringContainsInOrder(Iterable<String>)  : Matcher<String>
```

### Iterables

```
everyItem(Matcher<U>)                    : Matcher<Iterable<U>>

hasItem(T)
hasItem(Matcher<? super T>)  : Matcher<Iterable<? super T>>

hasItems(T...)
hasItems(Matcher<? super T>...)          : Matcher<Iterable<T>>

emptyIterable()              : Matcher<Iterable<? extends E>>
emptyIterableOf(Class<E>)                : Matcher<Iterable<E>>

contains(E...)
contains(Matcher<? super E>...)
contains(Matcher<? super E>)
contains(List<Matcher<? super E>>)
                             : Matcher<Iterable<? extends E>>

containsInAnyOrder(T...)
containsInAnyOrder(Collection<Matcher<? super T>>)
containsInAnyOrder(Matcher<? super T>...)
containsInAnyOrder(Matcher<? super E>)
                             : Matcher<Iterable<? extends E>>

iterableWithSize(Matcher<? super Integer>)
iterableWithSize(int)                    : Matcher<Iterable<E>>
```

### Collections

```
hasSize(int)
hasSize(Matcher<? super Integer>)
                             : Matcher<Collection<? extends E>>

empty()                      : Matcher<Collection<? extends E>>
emptyCollectionOf(Class<E>)  : Matcher<Collection<E>>
```

### Arrays

```
array(Matcher<? super T>...)             : Matcher<T[]>

hasItemInArray(T)
hasItemInArray(Matcher<? super T>)       : Matcher<T[]>

arrayContaining(E...)
arrayContaining(List<Matcher<? super E>>)
arrayContaining(Matcher<? super E>...)   : Matcher<E[]>

arrayContainingInAnyOrder(E...)
arrayContainingInAnyOrder(Matcher<? super E>...)
arrayContainingInAnyOrder(Collection<Matcher<? super E>>)
                                         : Matcher<E[]>

arrayWithSize(int)
arrayWithSize(Matcher<? super Integer>)
emptyArray()                             : Matcher<E[]>
```

### Maps

```
hasEntry(K, V)
hasEntry(Matcher<? super K>, Matcher<? super V>)
                  : Matcher<Map<? extends K, ? extends V>>

hasKey(K)
hasKey(Matcher<? super K>)   : Matcher<Map<? extends K, ?>>

hasValue(V)
hasValue(Matcher<? super V>) : Matcher<Map<?, ? extends V>>
```

### Beans

```
hasProperty(String)
hasProperty(String, Matcher<?>)
samePropertyValuesAs(T)                  : Matcher<T>
```

### Comparables

```
comparesEqualTo(T extends Comparable<T>)
greaterThan(T extends Comparable<T>)
greaterThanOrEqualTo(T extends Comparable<T>)
lessThan(T extends Comparable<T>)
lessThanOrEqualTo(T extends Comparable<T>)    : Matcher<T>
```

### Numbers

```
closeTo(double, double)                  : Matcher<Double>
closeTo(BigDecimal, BigDecimal)          : Matcher<BigDecimal>
```

### Classes

```
typeCompatibleWith(Class<T>)  : Matcher<java.lang.Class<?>>
```

### EventObjects

```
eventFrom(Object)
eventFrom(Class<? extends EventObject>, Object)
                                         : Matcher<EventObject>
```

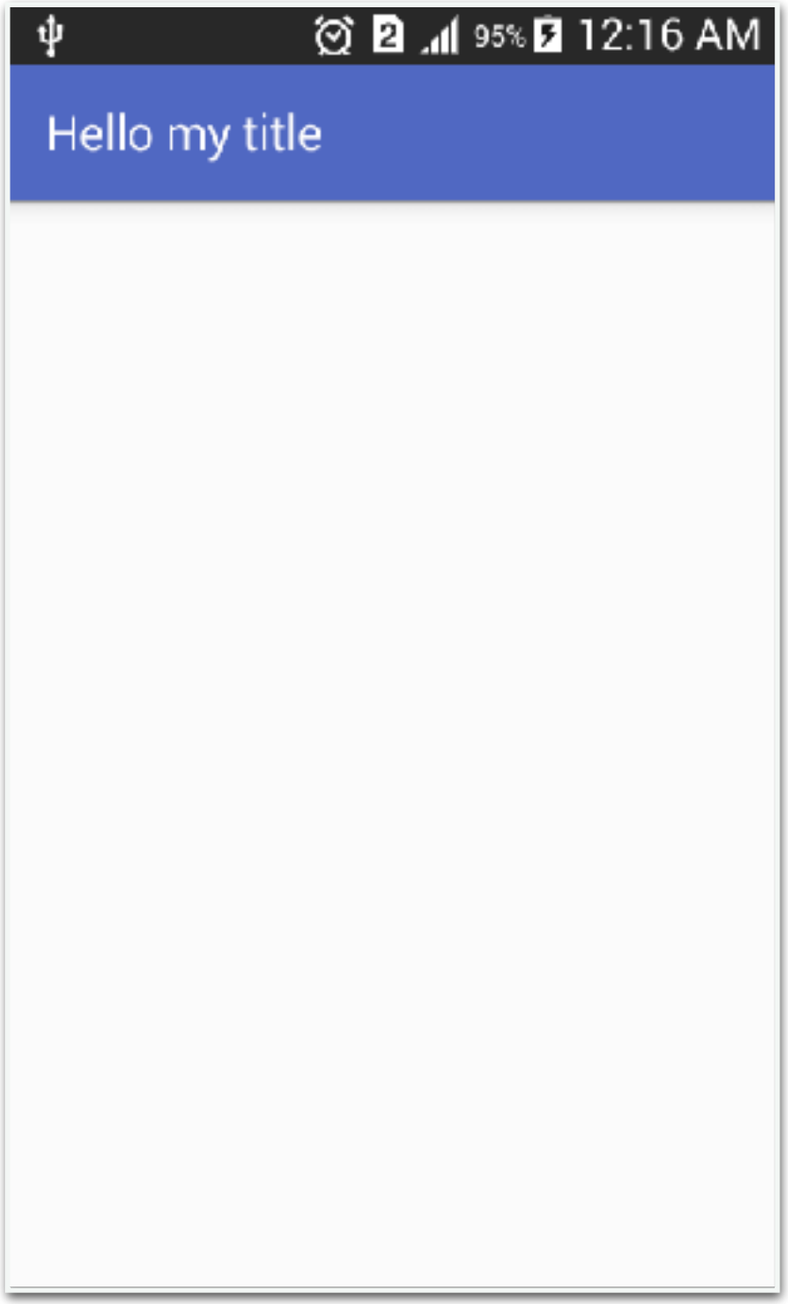
### DOM

```
hasXPath(String)
hasXPath(String, NamespaceContext)
hasXPath(String, Matcher<String>)
hasXPath(String, NamespaceContext, Matcher<String>)
                                         : Matcher<org.w3c.dom.Node>
```



http://www.marcphilipp.de/downloads/posts/2013-01-02-hamcrest-quick-reference/Hamcrest-1.3.pdf

# Matchers with Toolbar title



# Espresso with onData

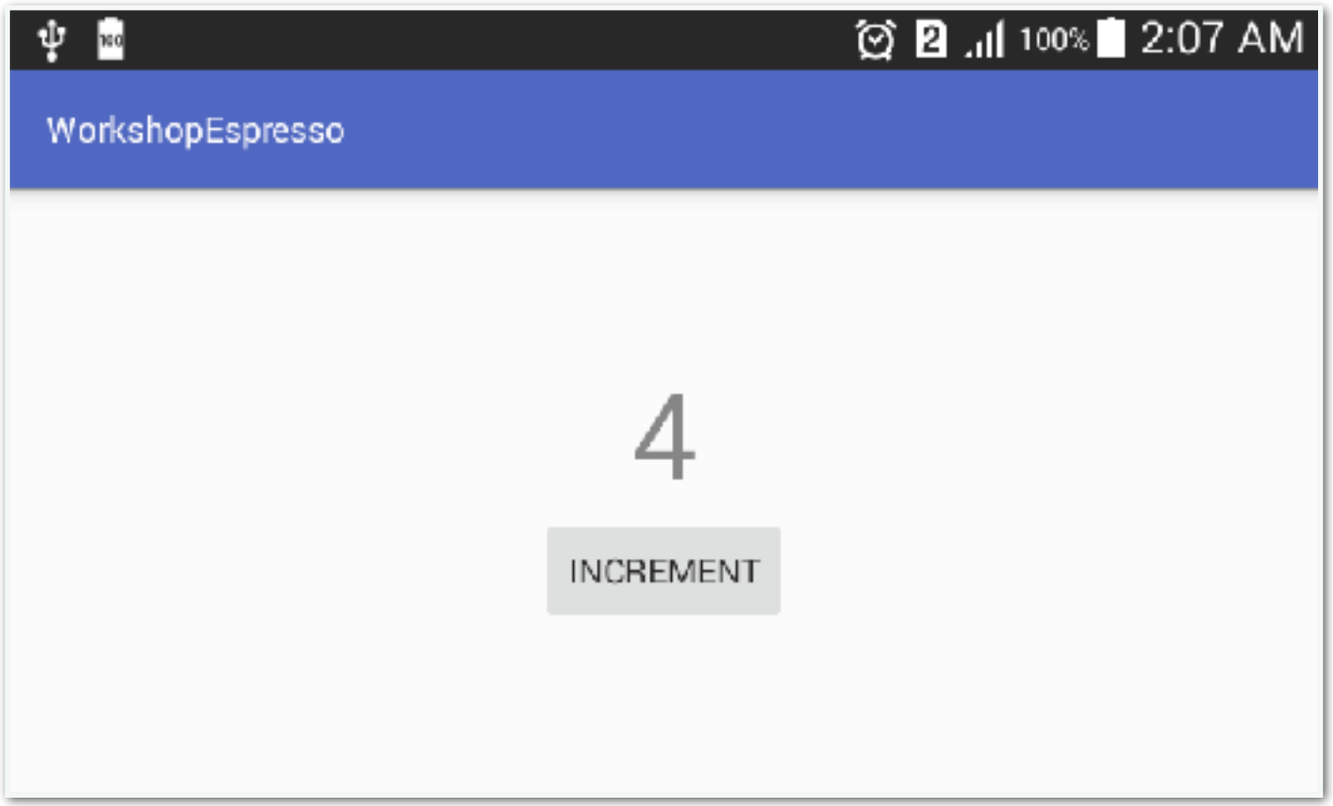
```
onData(ObjectMatcher)
        .DataOptions
        .perform(ViewAction)
        .check(ViewAssertion)
```

## Data Options

```
inAdapterView(Matcher)
atPosition(Integer)
onChildView(Matcher)
```

# Listview

# Rotate screen

# Example


6:00
Demo Testing
Type Operand 1
Type Operand 2
ADD
SUB
DIV
MUL
6:00
Demo Testing
1
2
ADD
SUB
DIV
MUL
6:00
Demo Testing
1
2
ADD
SUB
DIV
MUL
3.0

# Example

```
@RunWith(AndroidJUnit4.class)
@LargeTest
public class CalculatorAddTest {

    public static final String ONE = "1";
    public static final String TWO = "2";
    public static final String RESULT = "3.0";

    @Rule
    public ActivityTestRule<MainActivity> mActivityRule =
            new ActivityTestRule<MainActivity>(MainActivity.class);

    @Test
    public void calculatorAdd() {
        onView(withId(R.id.operand_one_edit_text)).perform(typeText(ONE));
        onView(withId(R.id.operand_two_edit_text)).perform(typeText(TWO));
        onView(withId(R.id.operation_add_button)).perform(click());
        onView(withId(R.id.operation_result_text_view)).check(matches(withText(RESULT)));
    }

}
```

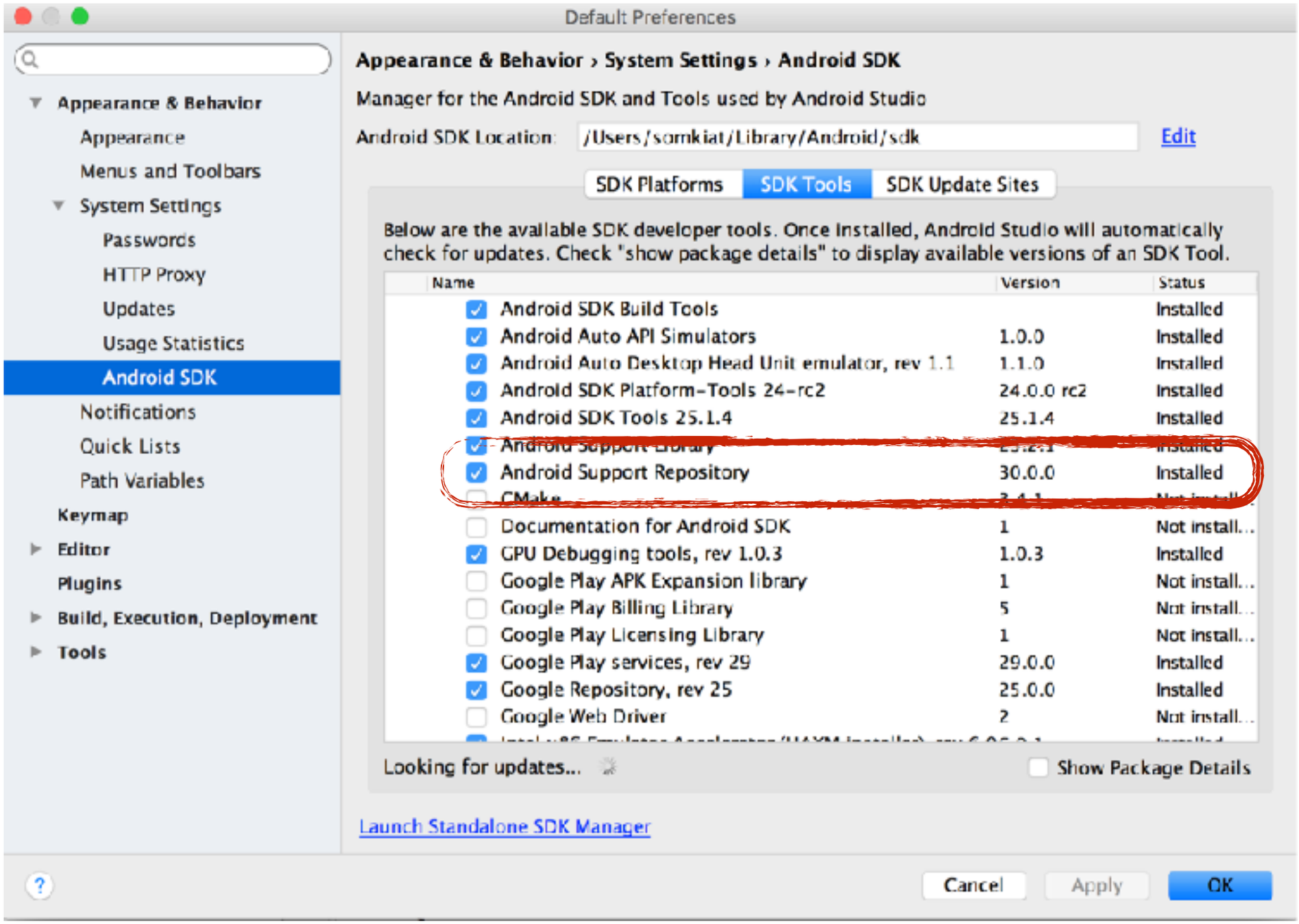
# Workshop

## Configuration espresso in project

## เรียนรู้การสร้าง UI test

## Code coverage

# ติดตั้ง Android support repository



# เพิ่ม Espresso library

```
dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    compile 'com.android.support:appcompat-v7:23.3.0'
    compile 'com.android.support:support-annotations:23.3.0'
    //Unit testing
    testCompile 'junit:junit:4.12'
    //UI testing with Espresso
    androidTestCompile 'com.android.support:support-annotations:23.3.0'
    androidTestCompile 'com.android.support.test:runner:0.5'
    androidTestCompile 'com.android.support.test:rules:0.5'
    androidTestCompile 'com.android.support.test.espresso:espresso-core:2.2.2'
}
```

# Add instrumentation runner

```
android {
    compileSdkVersion 23
    buildToolsVersion '24.0.0 rc3'

    defaultConfig {
        applicationId "demo.somkiat.demounittest"
        minSdkVersion 15
        targetSdkVersion 23
        versionCode 1
        versionName "1.0"

        testInstrumentationRunner "android.support.test.runner.AndroidJUnitRunner"
    }
}
```

# First UI test

```java
@RunWith(AndroidJUnit4.class)
@LargeTest
public class CalculatorAddTest {

    public static final String ONE = "1";
    public static final String TWO = "2";
    public static final String RESULT = "3.0";

    @Rule
    public ActivityTestRule<MainActivity> mActivityRule =
            new ActivityTestRule<MainActivity>(MainActivity.class);

    @Test
    public void calculatorAdd() {
        onView(withId(R.id.operand_one_edit_text)).perform(typeText(ONE));
        onView(withId(R.id.operand_two_edit_text)).perform(typeText(TWO));
        onView(withId(R.id.operation_add_button)).perform(click());
        onView(withId(R.id.operation_result_text_view)).check(matches(withText(RESULT)));
    }

}
```

# Run UI test

$./gradlew connectedAndroidTest

$./gradlew cAT

# Show report

## Test Summary

| 1 | 0 | 2.476s | 100% |
|---|---|---|---|
| tests | failures | duration | successful |

**Packages** Classes

| Package | Tests | Failures | Duration | Success rate |
|---|---|---|---|---|
| demo.somkiat.demounittest | 1 | 0 | 2.476s | 100% |

Generated by Gradle 2.10 at May 9, 2016, 6:22:50 AM

# Show coverage report

debug > demo.somkiat.demounittest

## demo.somkiat.demounittest

| Element | Missed Instructions | Cov. | Missed Branches | Cov. | Missed | Cxty |
|---|---|---|---|---|---|---|
| MainActivity | | 85% | | 50% | 1 | 6 |
| Calculator | | 100% | | n/a | 0 | 2 |
| Total | 13 of 91 | 86% | 1 of 2 | 50% | 1 | 8 |

# แบบฝึกหัด

## บวก ลบ คูณ หาร

## Code coverage = 100% !!

# Merge coverage unit + ui

```
def fileFilter = ['**/R.class', '**/R$*.class', '**/BuildConfig.*', '**/Manifest*
def debugTree = fileTree(dir: "${buildDir}/intermediates/classes/debug", excludes
def mainSrc = "${project.projectDir}/src/main/java"

sourceDirectories = files([mainSrc])
classDirectories = files([debugTree])
executionData = files(["${buildDir}/jacoco/testDebugUnitTest.exec",
                       "${buildDir}/outputs/code-coverage/connected/coverage.ec"
])
```

# Merge coverage unit + ui

$./gradlew clean
createDebugCoverageReport jacocoTestReport

# Merge coverage unit + ui

app > demo.somkiat.demounittest

## demo.somkiat.demounittest

| Element | Missed Instructions | Cov. | Missed Branches | Cov. | Missed | Cxty |
|---|---|---|---|---|---|---|
| MainActivity | | 85% | | 50% | 1 | 6 |
| Calculator | | 100% | | n/a | 0 | 3 |
| Total | 13 of 95 | 86% | 1 of 2 | 50% | 1 | 9 |

# Android unit testing

ใช้งาน JUnit 4.12

https://github.com/junit-team/junit4

# Android assertions

| Assertion | Description |
|---|---|
| `assertEquals` | Test that two values are the same |
| `assertTrue` | Test Boolean condition is true |
| `assertFalse` | Test Boolean condition is false |
| `assertNull` | Check that the object is null |
| `assertNotNull` | Check that the object is not null |
| `assertSame` | Test that both values refer to the same object reference |
| `assertNotSame` | Test that both values do not refer to the same object reference |
| `assertThat` | Test that the first value (object) matches the second value (or matcher) |
| `fail` | Test should always fail |

# Unit options

@Before
@After
@BeforeClass
@AfterClass
@Test
@Test(expected=exception)
@Test(timeout=ms)

# Grouping tests

| Feature | Small | Medium | Large |
|---|---|---|---|
| Network access | No | localhost only | Yes |
| Database | No | Yes | Yes |
| File system access | No | Yes | Yes |
| Use external systems | No | Discouraged | Yes |
| Multiple threads | No | Yes | Yes |
| Sleep statements | No | Yes | Yes |
| System properties | No | Yes | Yes |
| Time limit (seconds) | 60 | 300 | 900+ |

http://googletesting.blogspot.com/2010/12/test-sizes.html

# Example

```java
@SmallTest
@Test
public void addition_isCorrect() throws Exception {
    assertEquals(3, calculator.add(1, 2), 0);
}

@MediumTest
@Test
public void subtraction_isCorrect() throws Exception {
    assertEquals(1, calculator.sub(2, 1), 0);
}

@LargeTest
@Test
public void divide_isCorrect() throws Exception {
    assertEquals(2, calculator.div(2, 1), 0);
}
```

# ระดมสมอง

Unit test ?

UI test ?

https://plus.google.com/+AndroidDevelopers/posts/TPy1EeSaSg8

# Run with instrumentation runner

```
defaultConfig {
    applicationId "demo.somkiat.demounittest"
    minSdkVersion 15
    targetSdkVersion 23
    versionCode 1
    versionName "1.0"

    testInstrumentationRunner "android.support.test.runner.AndroidJUnitRunner"
    testInstrumentationRunnerArgument "size", "medium,large"
}
```

# Run test

$./gradlew clean cAT

# Parameterized tests

```
@Test
public void addition_isCorrect() throws Exception {
    assertEquals(3, calculator.add(1, 2), 0);
    assertEquals(3, calculator.add(2, 1), 0);
    assertEquals(10, calculator.add(8, 2), 0);
    assertEquals(5, calculator.add(-1, 6), 0);
}
```

## ได้กลิ่นอะไรไหม ?

# Parameterized tests

```
@Test
public void addition_isCorrect() throws Exception {
    assertEquals(3, calculator.add(1, 2), 0);
    assertEquals(3, calculator.add(2, 1), 0);
    assertEquals(10, calculator.add(8, 2), 0);
    assertEquals(5, calculator.add(-1, 6), 0);
}
```

มา Refactor code กัน

# 1. Add @RunWith(Parameterized.class)

```
@RunWith(Parameterized.class)
public class CalculatorAddUnitTest {
    Calculator calculator = new Calculator();
```

# 2. Add datas

```
@RunWith(Parameterized.class)
public class CalculatorAddUnitTest {
    Calculator calculator = new Calculator();

    @Parameters
    public static List<Object[]> data() {
        return Arrays.asList(new Object[][] {
                {1, 2, 3},
                {2, 1, 3},
                {8, 2, 10},
                {-1, 6, 5},
        });
    }
}
```

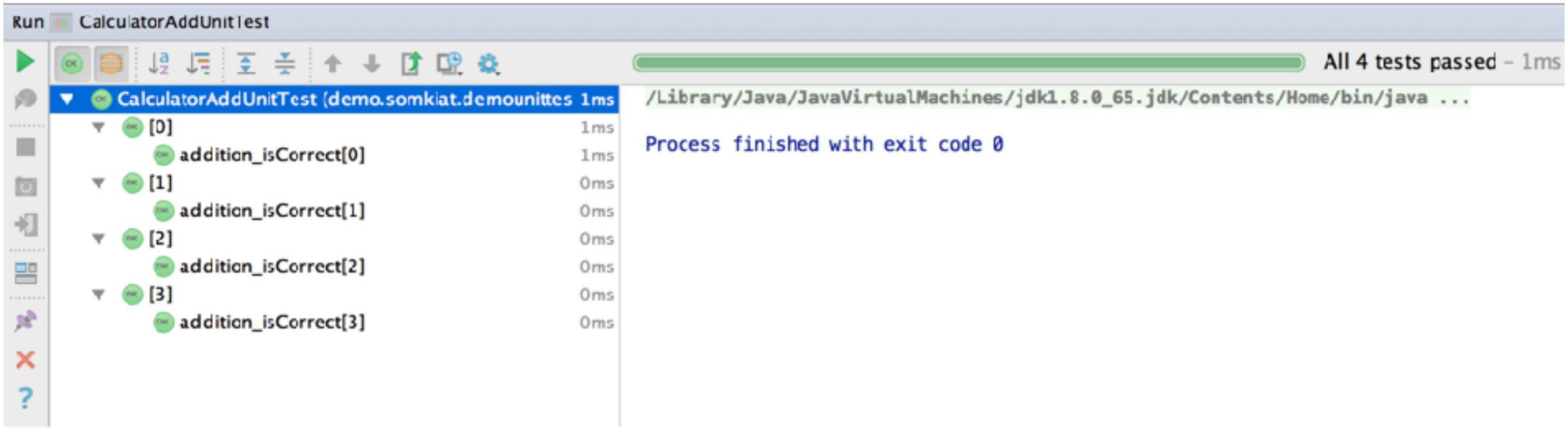
# 3. Add constructor

```
@RunWith(Parameterized.class)
public class CalculatorAddUnitTest {
    Calculator calculator = new Calculator();

    public CalculatorAddUnitTest(int mOperandOne,
                                 int mOperandTwo,
                                 int mExpectedResult) {
        this.mOperandOne = mOperandOne;
        this.mOperandTwo = mOperandTwo;
        this.mExpectedResult = mExpectedResult;
}
```

# 4. Use parameters in test

```
@RunWith(Parameterized.class)
public class CalculatorAddUnitTest {
    Calculator calculator = new Calculator();

    @Test
    public void addition_isCorrect(){
        assertEquals(this.mExpectedResult,
                calculator.add(this.mOperandOne, this.mOperandTwo), 0);
    }
```

# Run test & Show report



# Run test & Show report

## Class demo.somkiat.demounittest.CalculatorAddUnitTest

all > demo.somkiat.demounittest > CalculatorAddUnitTest

| 4 | 0 | 0 | 0.001s |
|---|---|---|---|
| tests | failures | ignored | duration |

**100%**
successful

### Tests

| Test | Duration | Result |
|---|---|---|
| addition_isCorrect[0] | 0.001s | passed |
| addition_isCorrect[1] | 0s | passed |
| addition_isCorrect[2] | 0s | passed |
| addition_isCorrect[3] | 0s | passed |

# แบบฝึกหัด

ทำการ run ทุก ๆ การทดสอบ
ทั้ง unit test และ ui test

# With or without TDD,

## unit testing needs to become part of your development process

# More testing

## Monkey testing

http://developer.android.com/tools/help/monkey.html

# Monkey testing

Stress testing

Command line tool

Random events

# How to use ?

$adb shell monkey -p <your package> -v <# of events>

# Workshop

$adb shell monkey -p **demo.somkiat.demounittesting** **-v 20000**

See result and fix it !!

# Good ideas in testing

**Multiple assertions** per test are OK

http://www.slideshare.net/alacenski/android-testing-strategies

# Good ideas in testing

Multiple assertions per test are OK

Test only one concept per test method

# Good ideas in testing

Multiple assertions per test are OK

Test only one concept per test method

You can write multiple test for one UI story

# Good ideas in testing

Multiple assertions per test are OK

Test only one concept per test method

You can write multiple test for one UI story

Don’t test other library’s code

# Good ideas in testing

Multiple assertions per test are OK

Test only one concept per test method

You can write multiple test for one UI story

Don’t test other library’s code

Look for ways to simplify your tests

# We need a continuous testing

Jenkins

Bamboo

TeamCity

go™

Visual Studio Team Foundation Server

Hudson

Travis

wercker

circleci

Jenkins

Bamboo

TeamCity

go

Visual Studio
Team Foundation Server

Hudson



Travis

wercker

circleci

# CI is about what people do not about what tools they use

# CI is a practice

## Discipline to **integrate frequently**

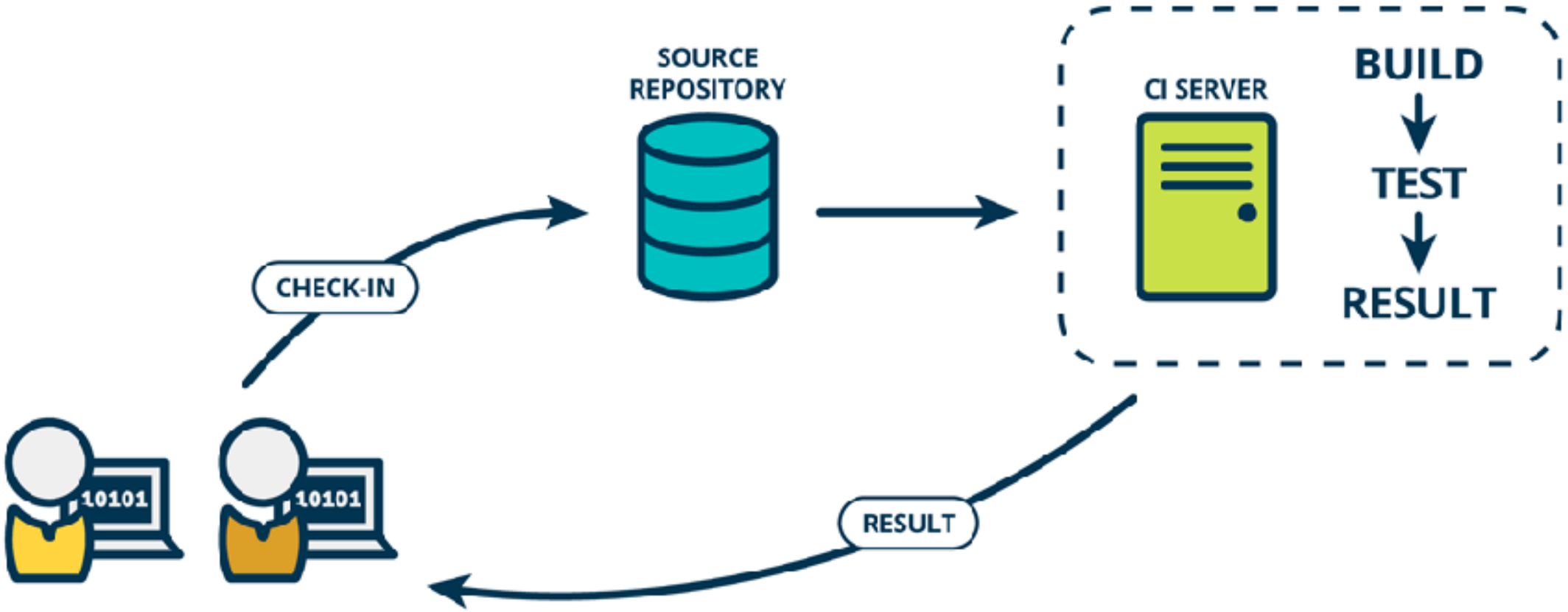
# CI is a practice

Strive to make **small change**

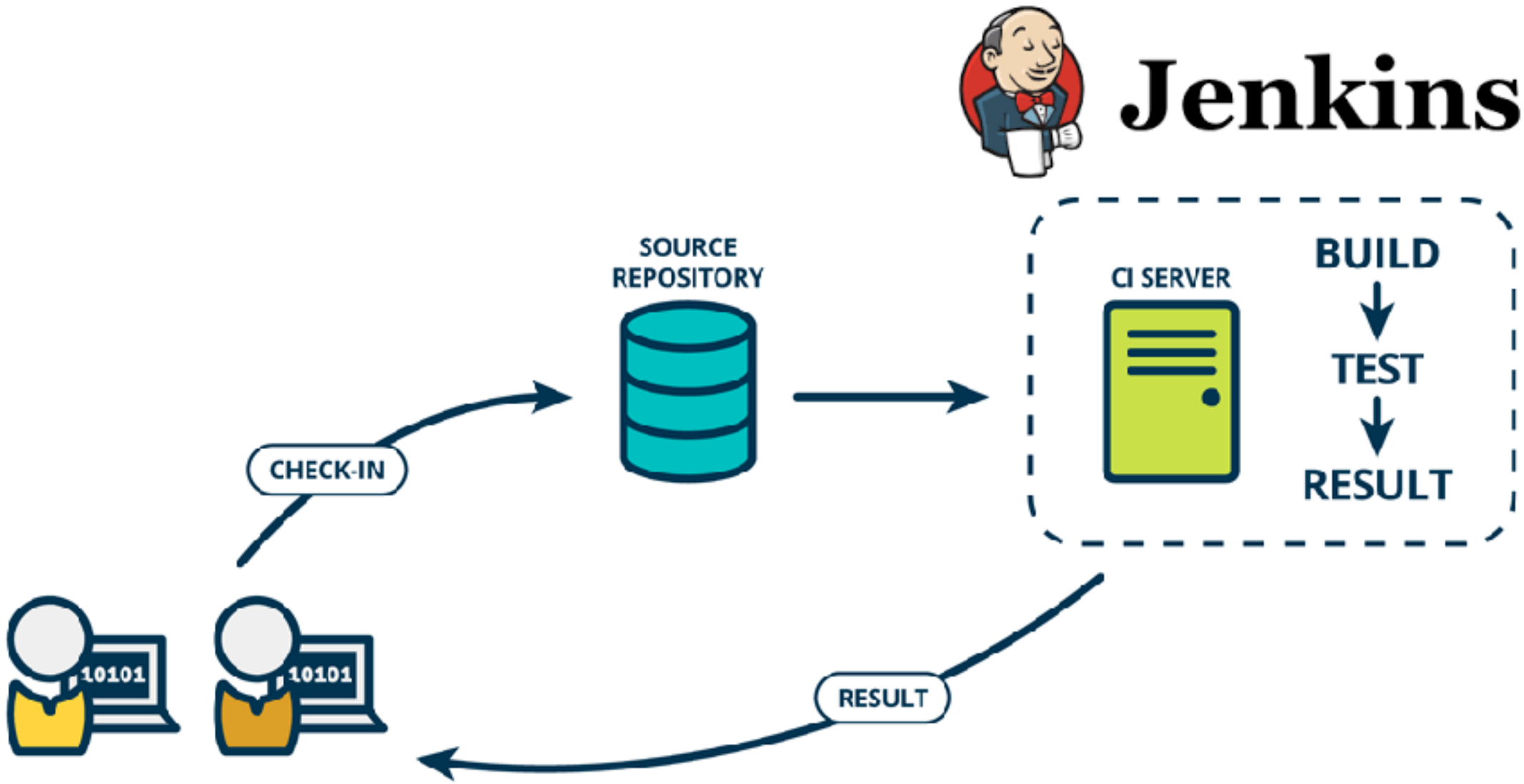
# CI is a practice

## Strive for **fast feedback**

# Continuous Integration



# Continuous Integration Server



# Workshop with Jenkins

ติดตั้ง Jenkins

ติดตั้ง plug-ins ต่าง ๆ

สร้างระบบการทดสอบแบบอัตโนมัติ

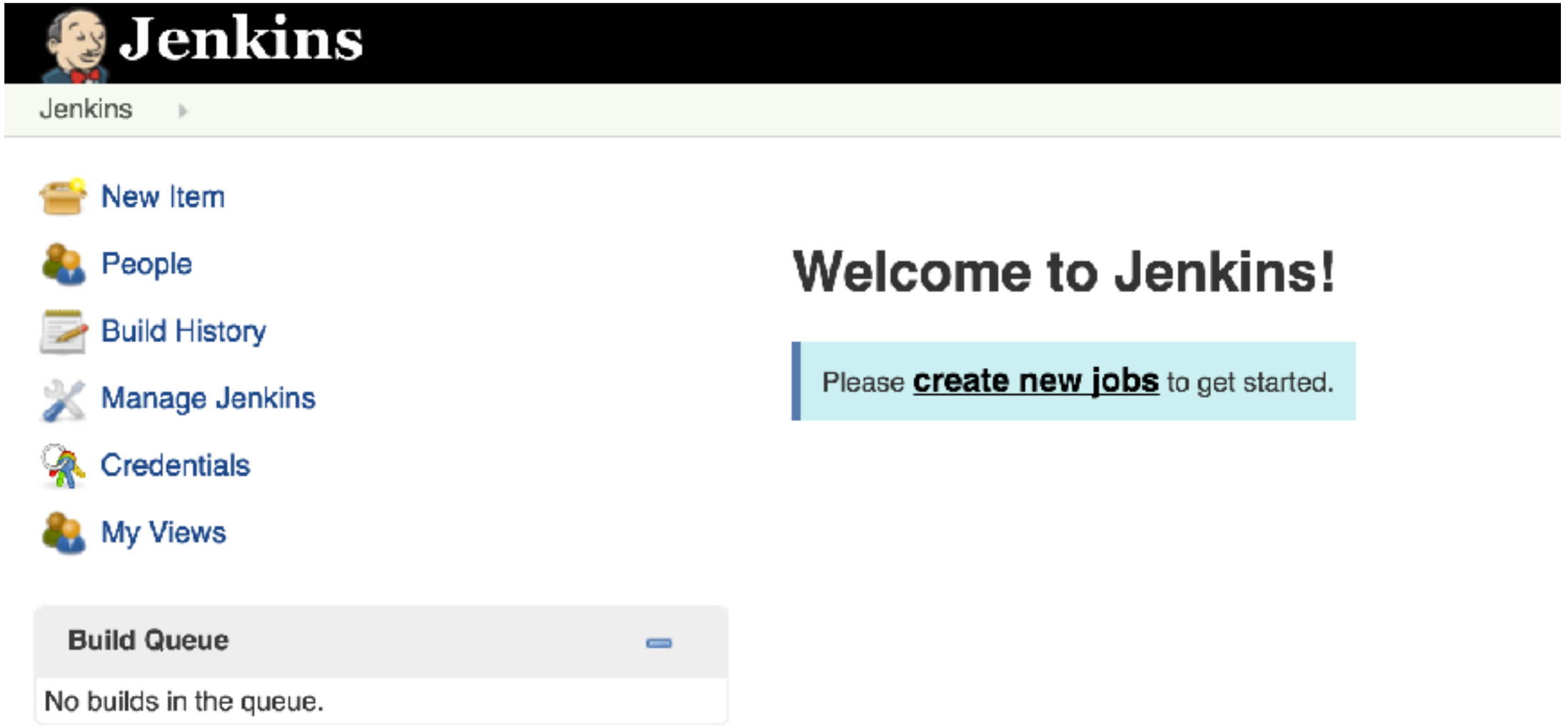
# Download Jenkins


https://jenkins-ci.org/

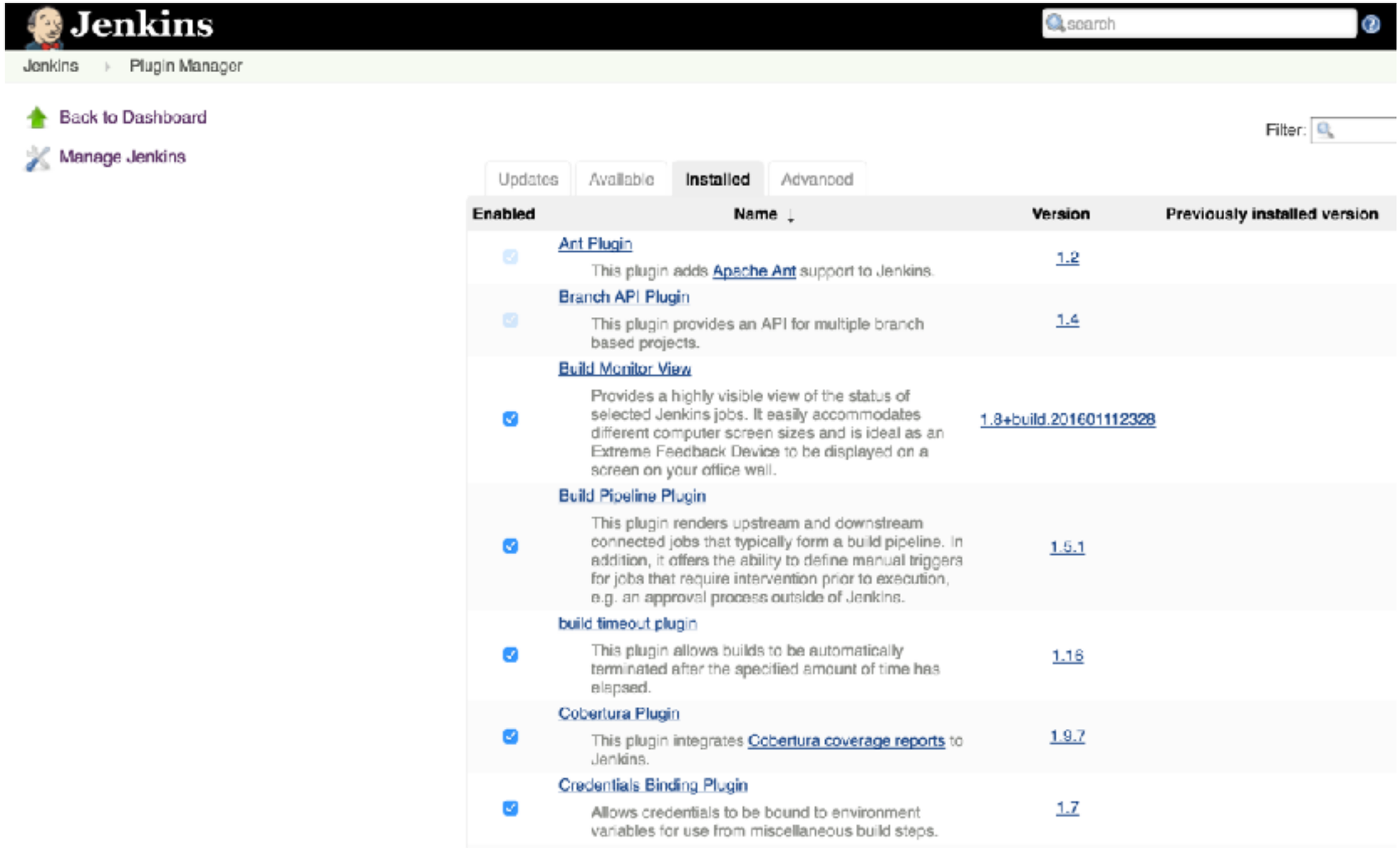
# Start Jenkins

$java -jenkins.war

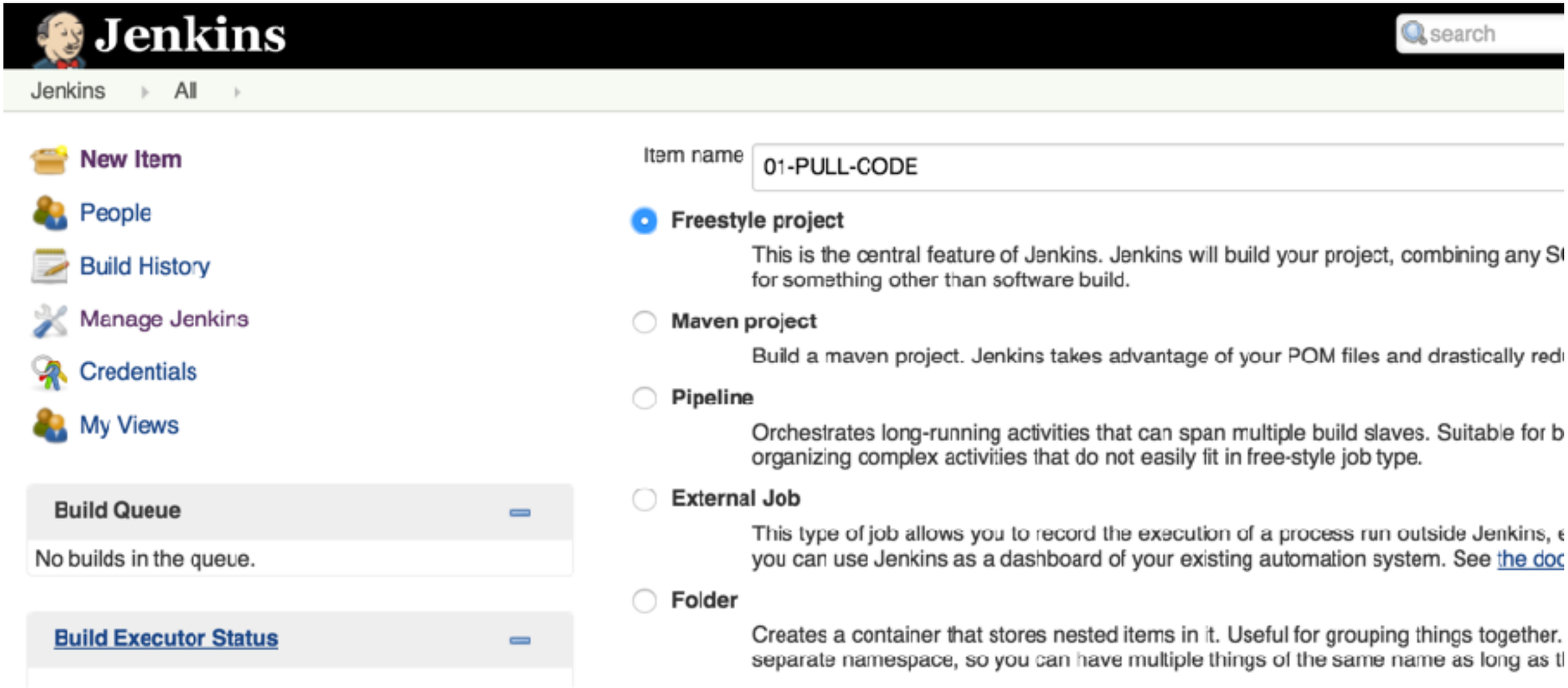
# Welcome to Jenkins



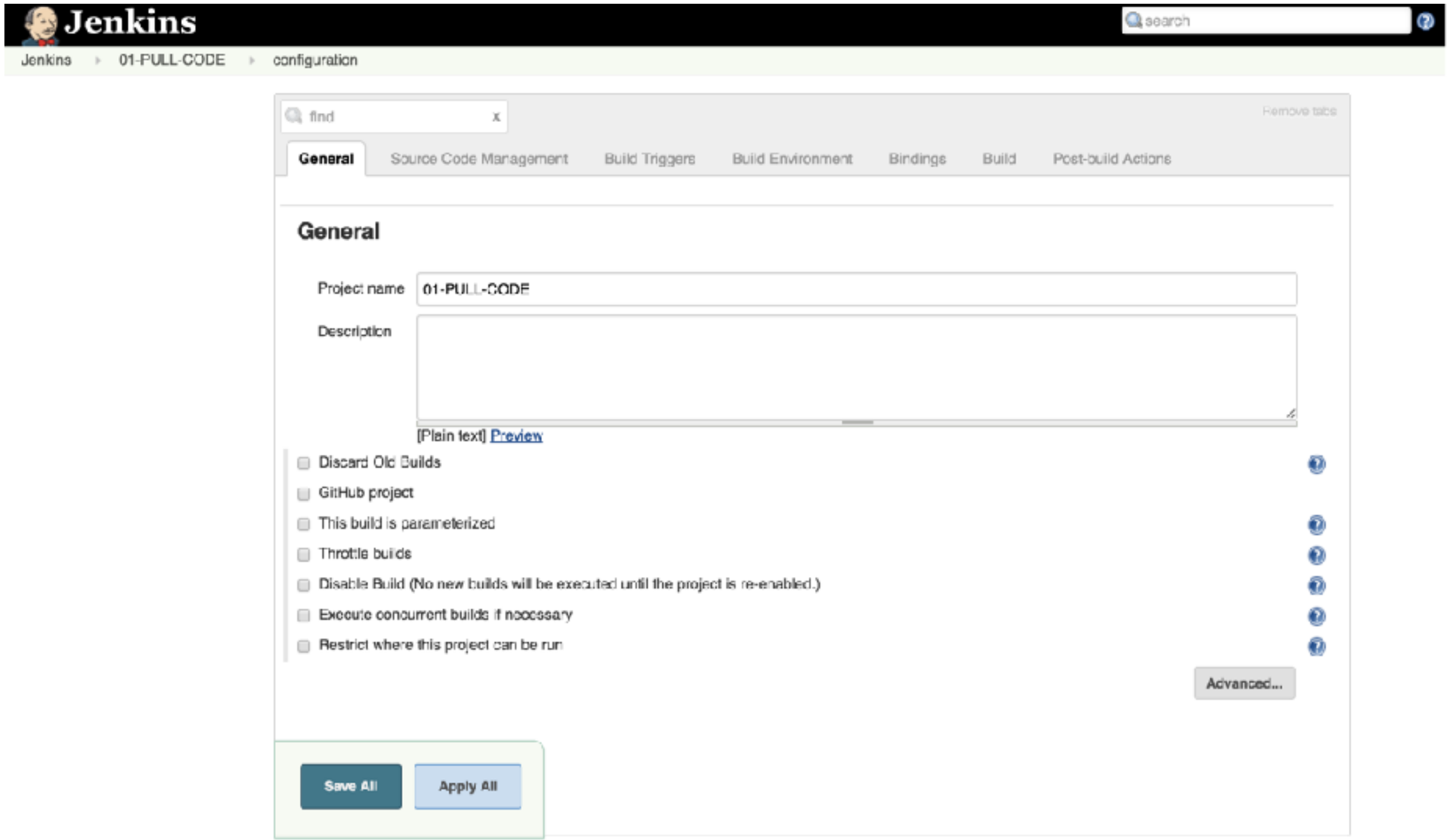
# Configure Jenkins



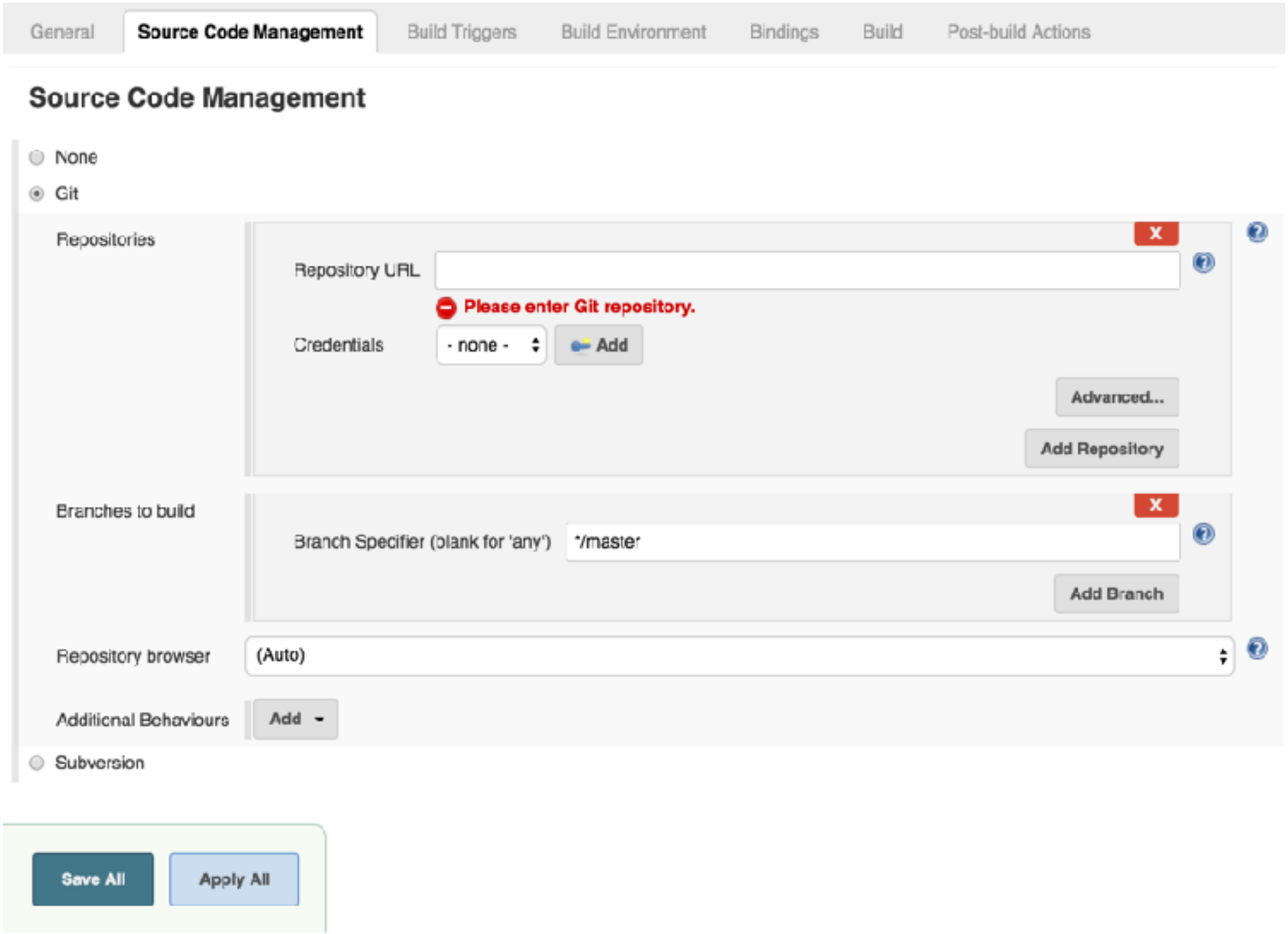
# Create automated job



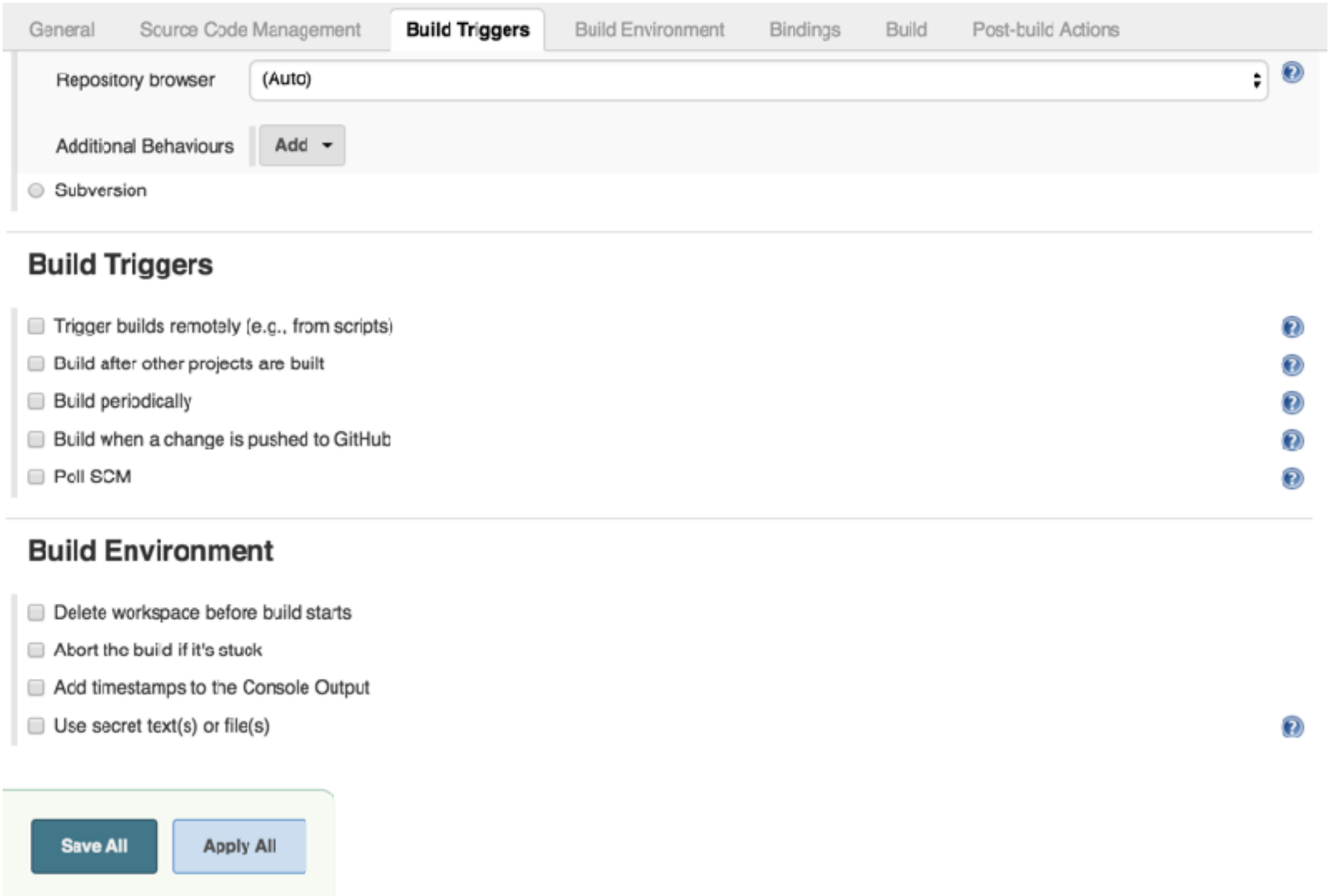
# Configure job



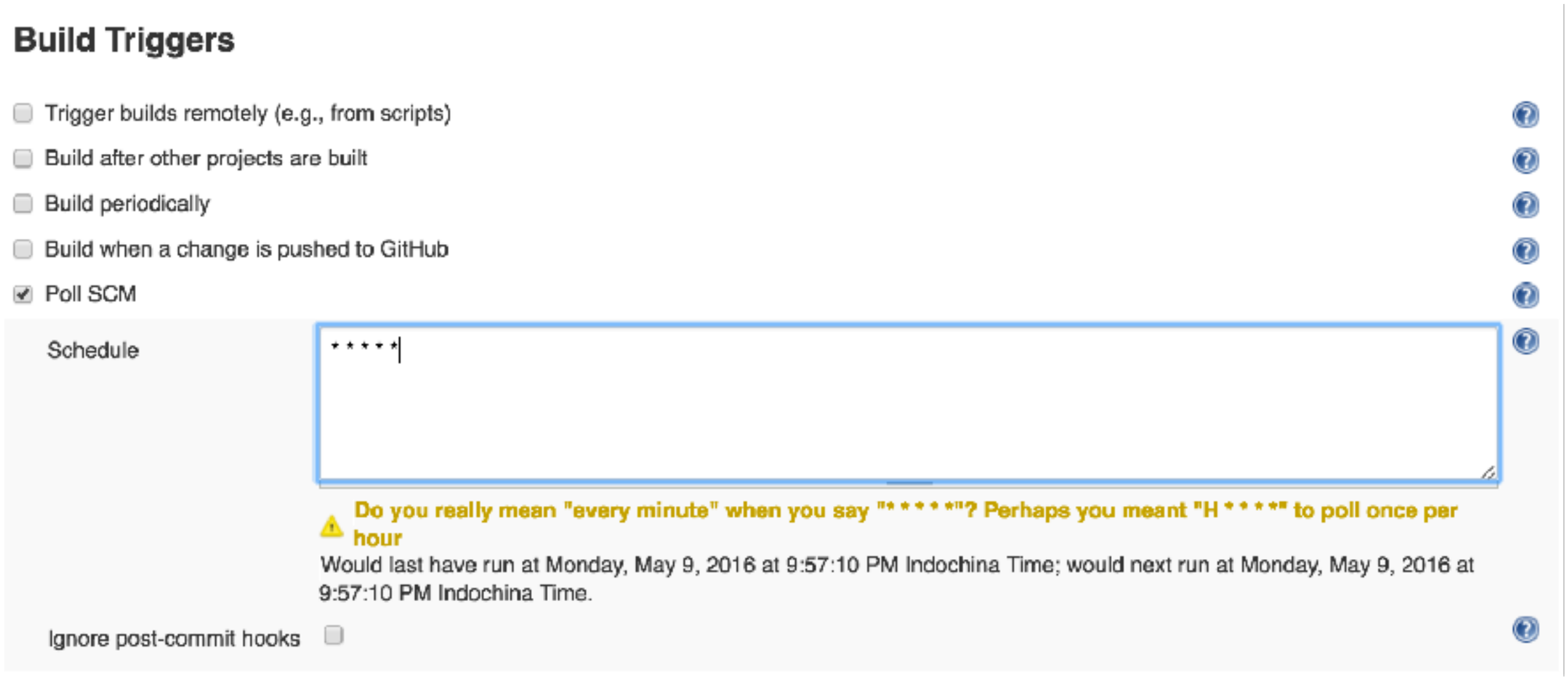
# Source code management



# Build triggers



# Build triggers with poll SCM



# Build

General | Source Code Management | Build Triggers | Build Environment | Bindings | **Build** | Post-build Actions

Schedule `* * * * *`

**Do you really mean "every minute" when you say "* * * * *"? Perhaps you meant "H * * * *" to poll once per hour**

Would last have run at Monday, May 9, 2016 at 9:57:10 PM Indochina Time; would next run at Monday, May 9, 2016 at 9:57:10 PM Indochina Time.

Ignore post-commit hooks

## Build Environment

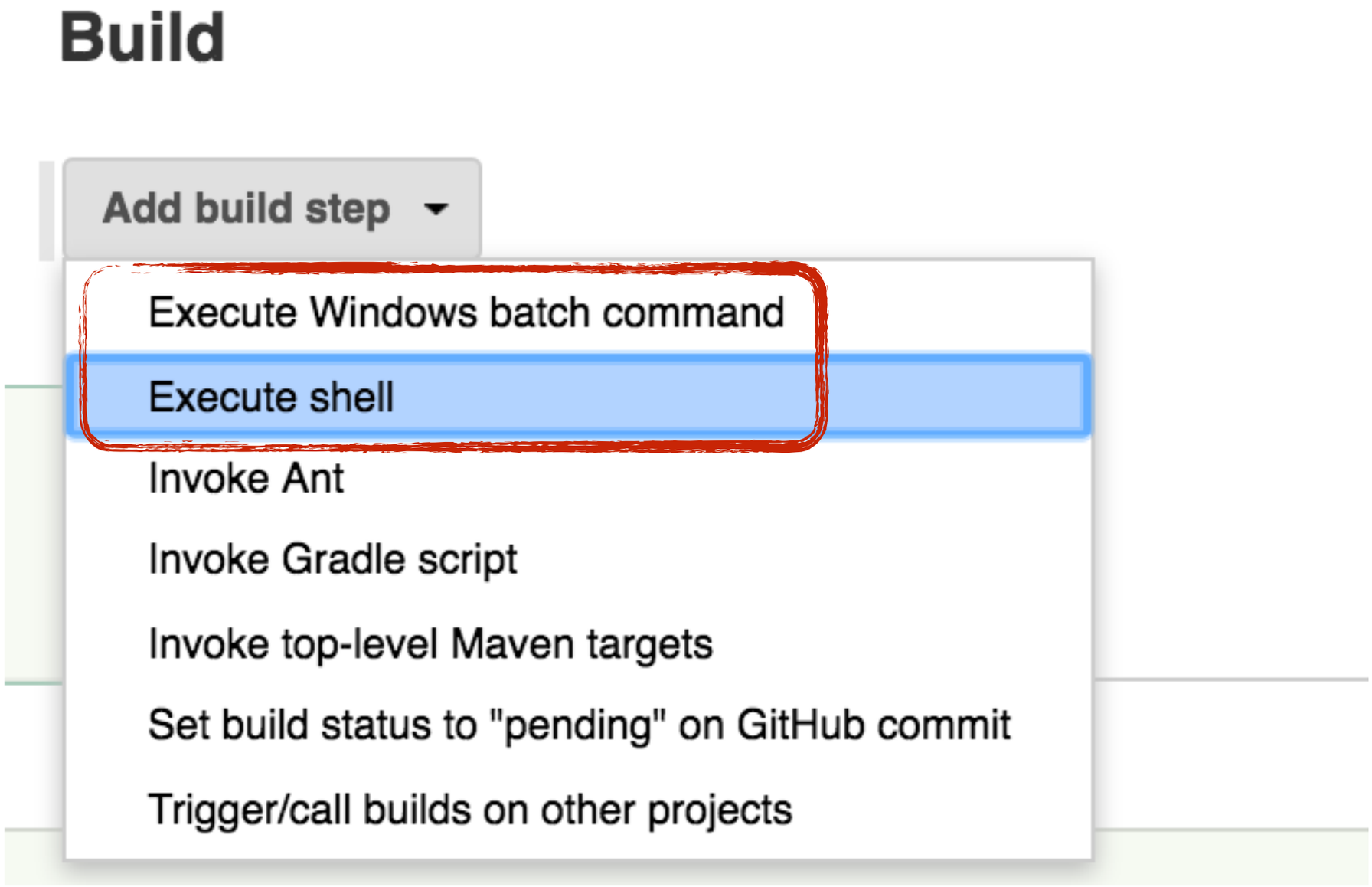
- Delete workspace before build starts
- Abort the build if it's stuck
- Add timestamps to the Console Output
- Use secret text(s) or file(s)

## Build

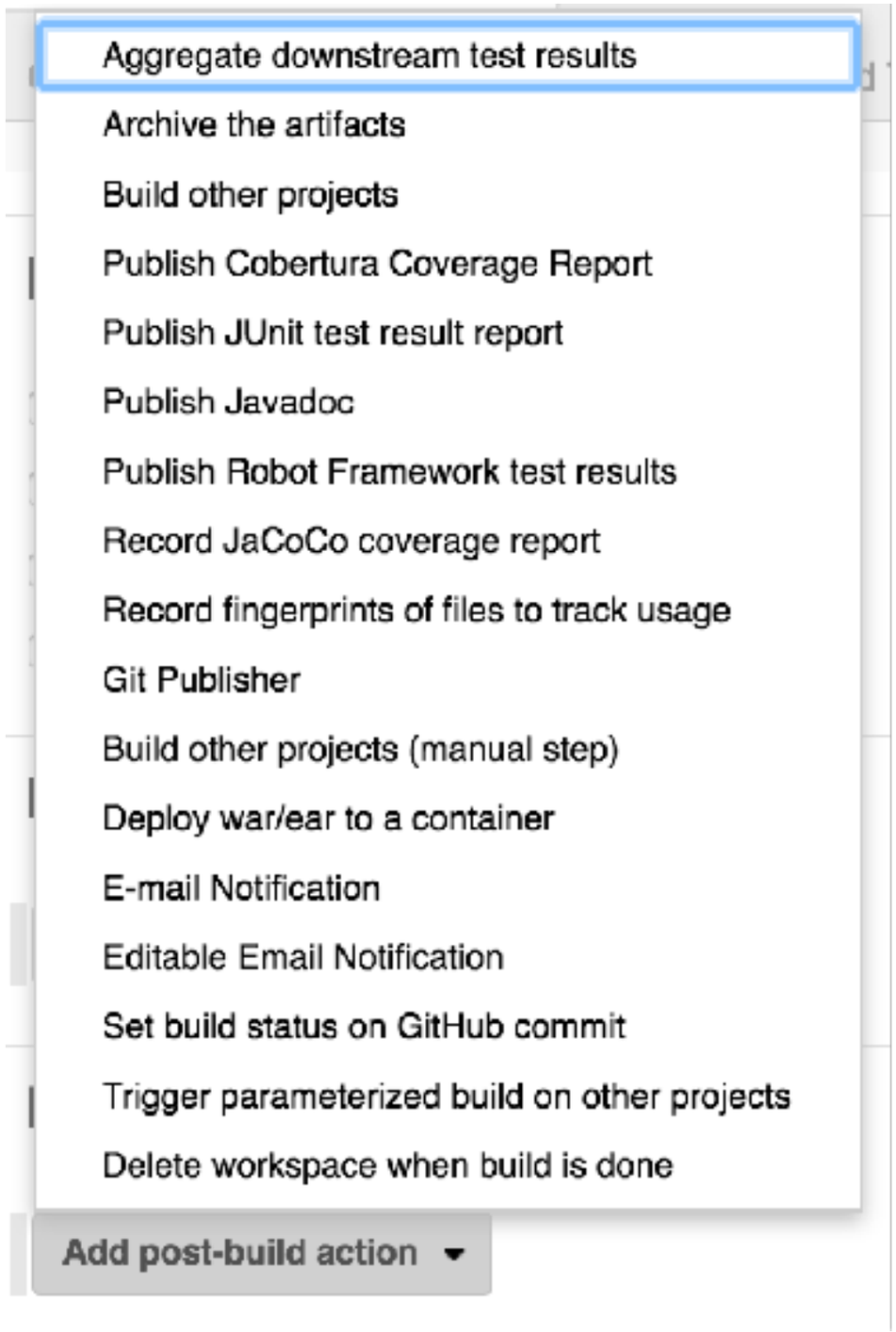
Add build step

Save All | Apply All

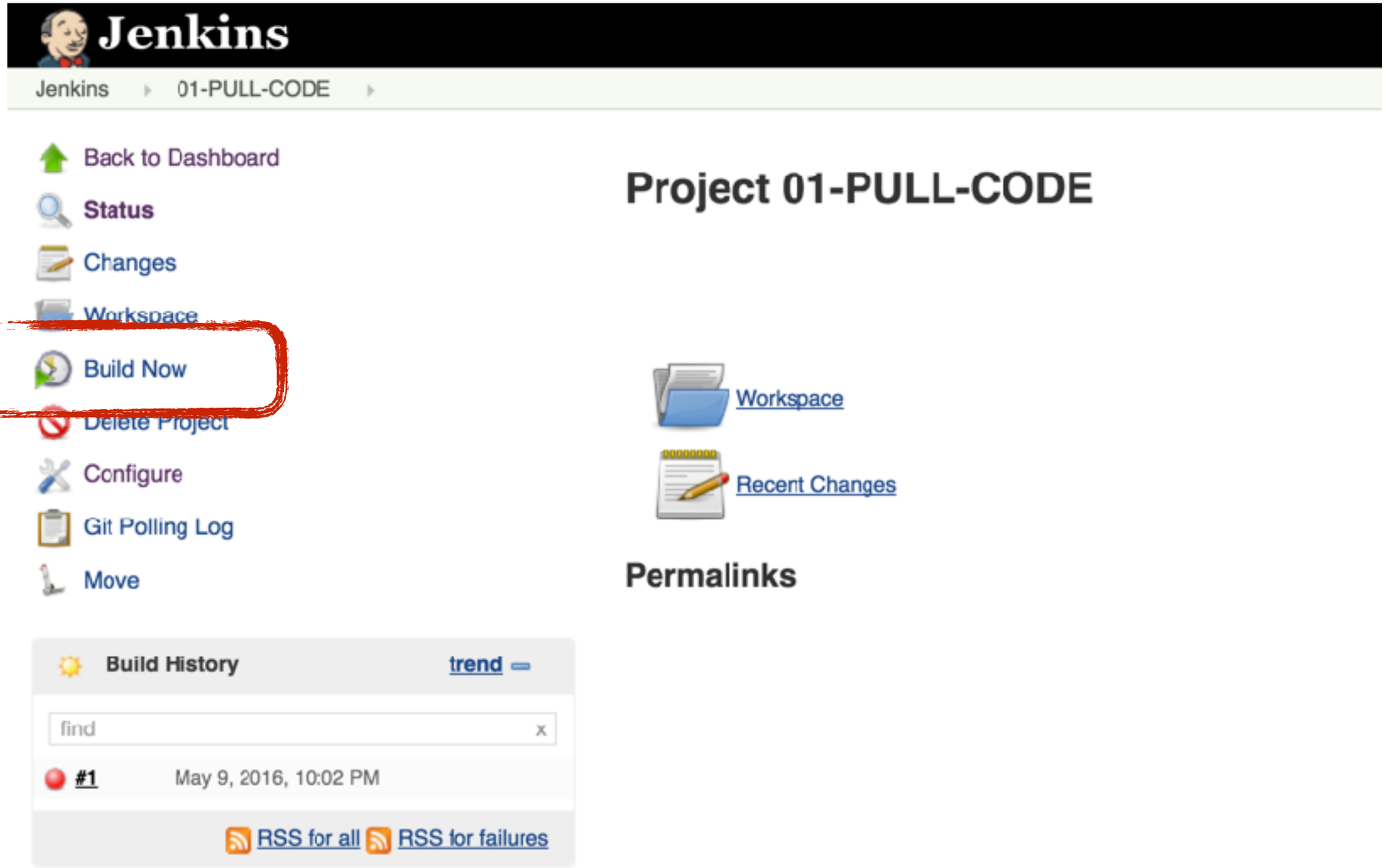
# Build with command line



# Post build actions



# Save and Build Now !!



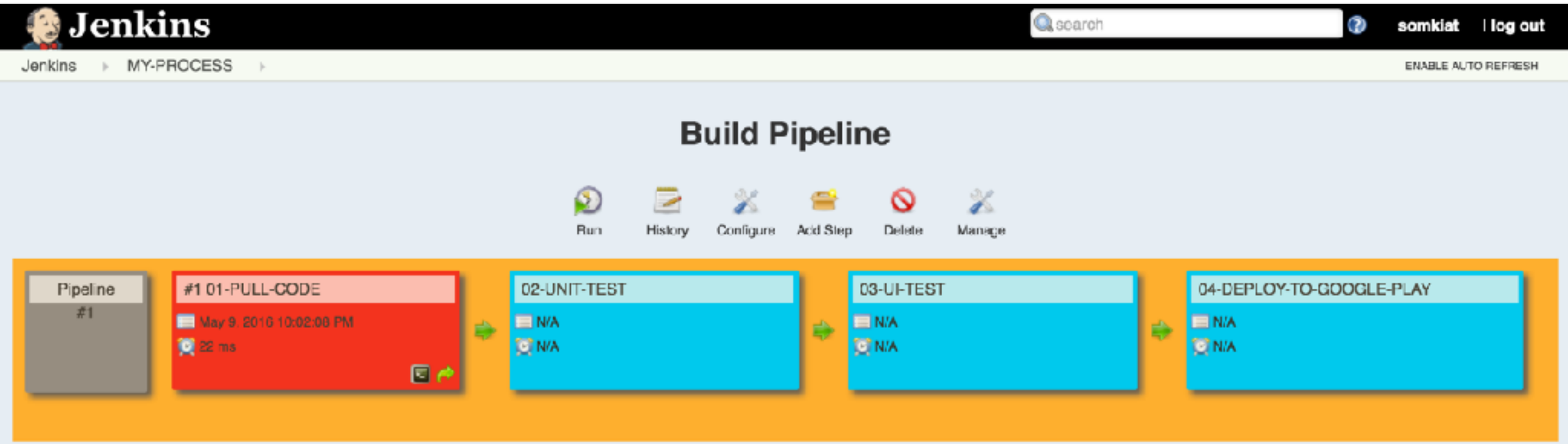
# แบบฝึกหัด

## ออกแบบ Build pipeline ของระบบงาน

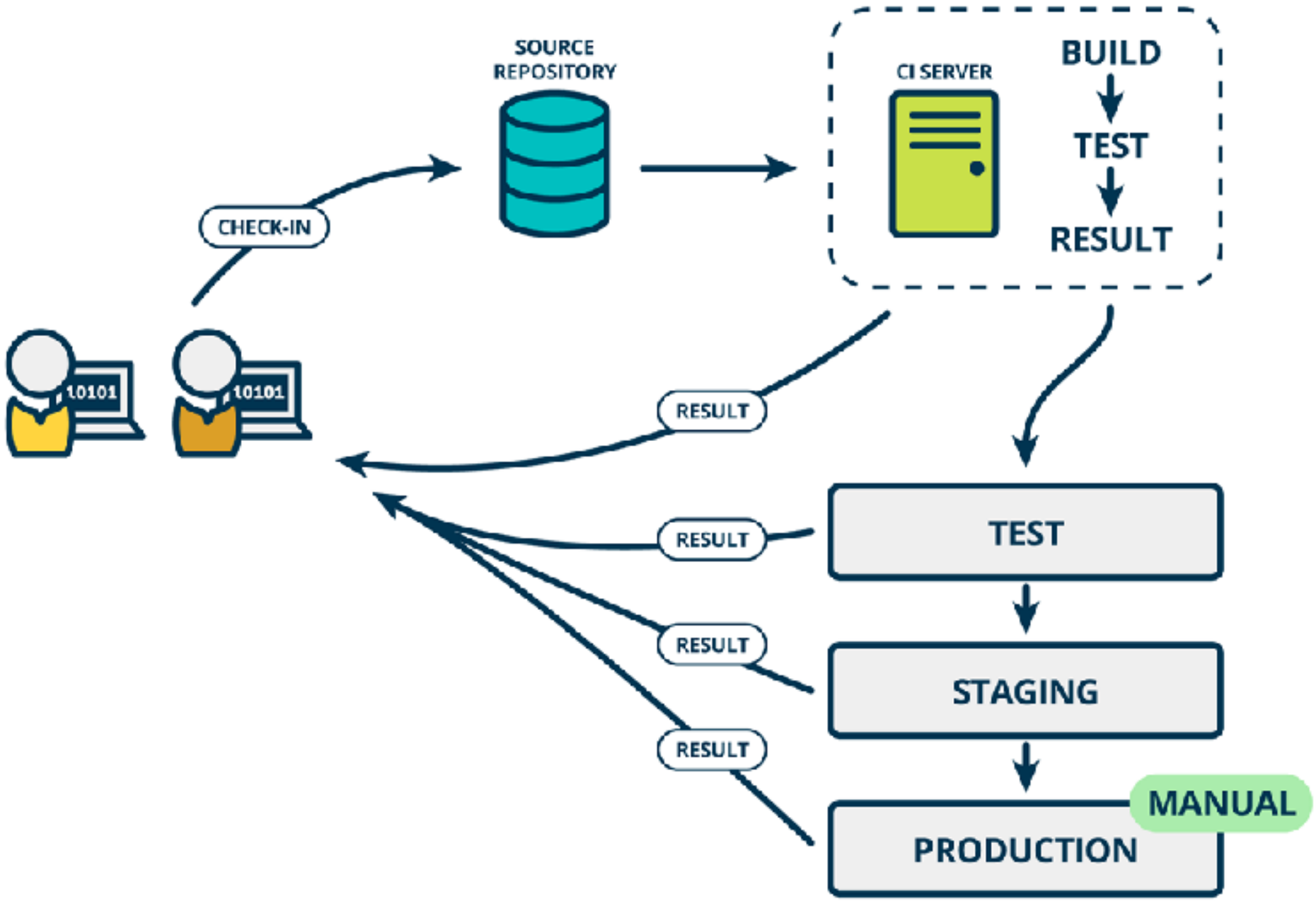
# แบบฝึกหัด

สร้าง Job ตาม build pipeline ใน Jenkins

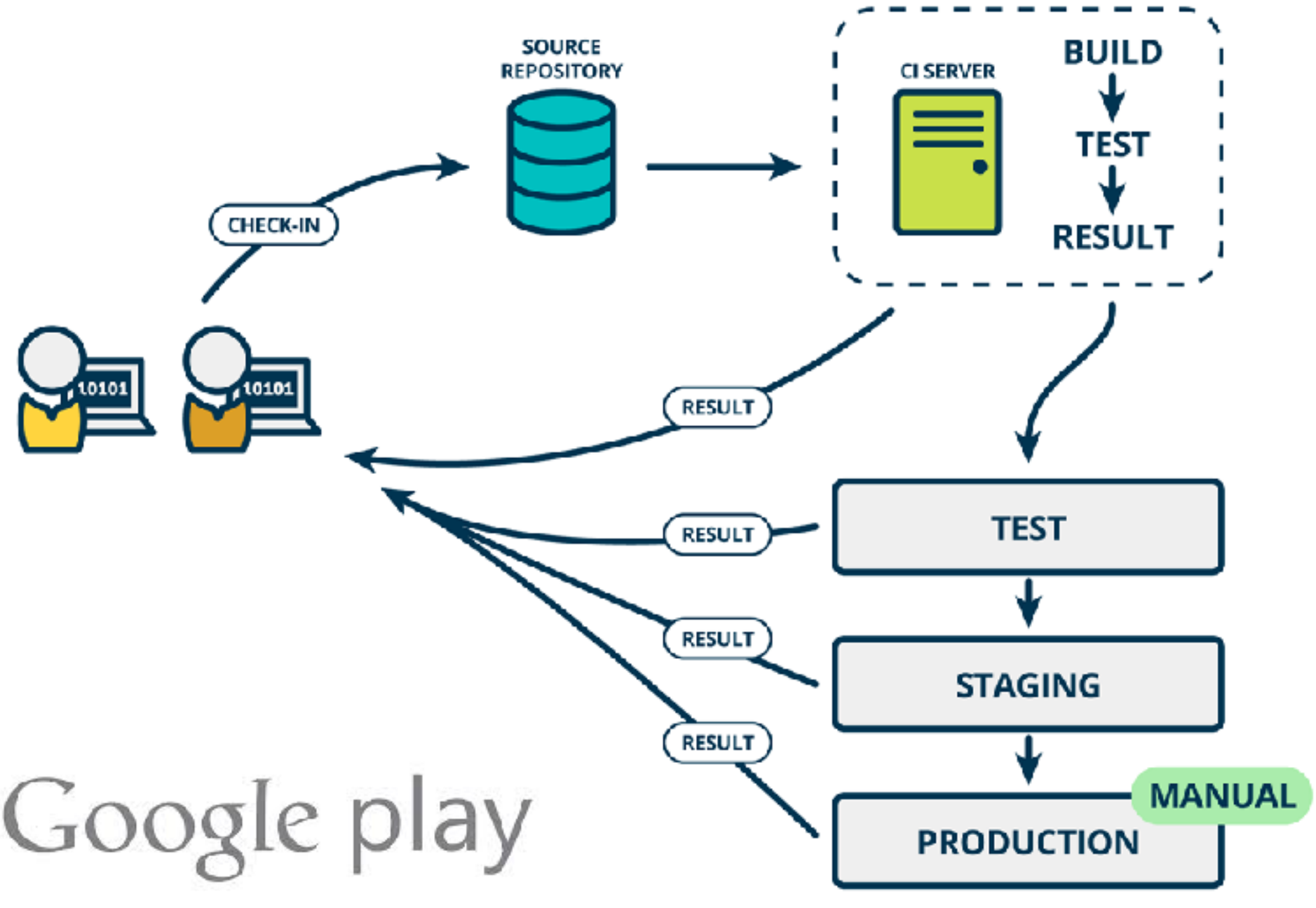
# Build pipeline



# Continuous Delivery



# Continuous Delivery

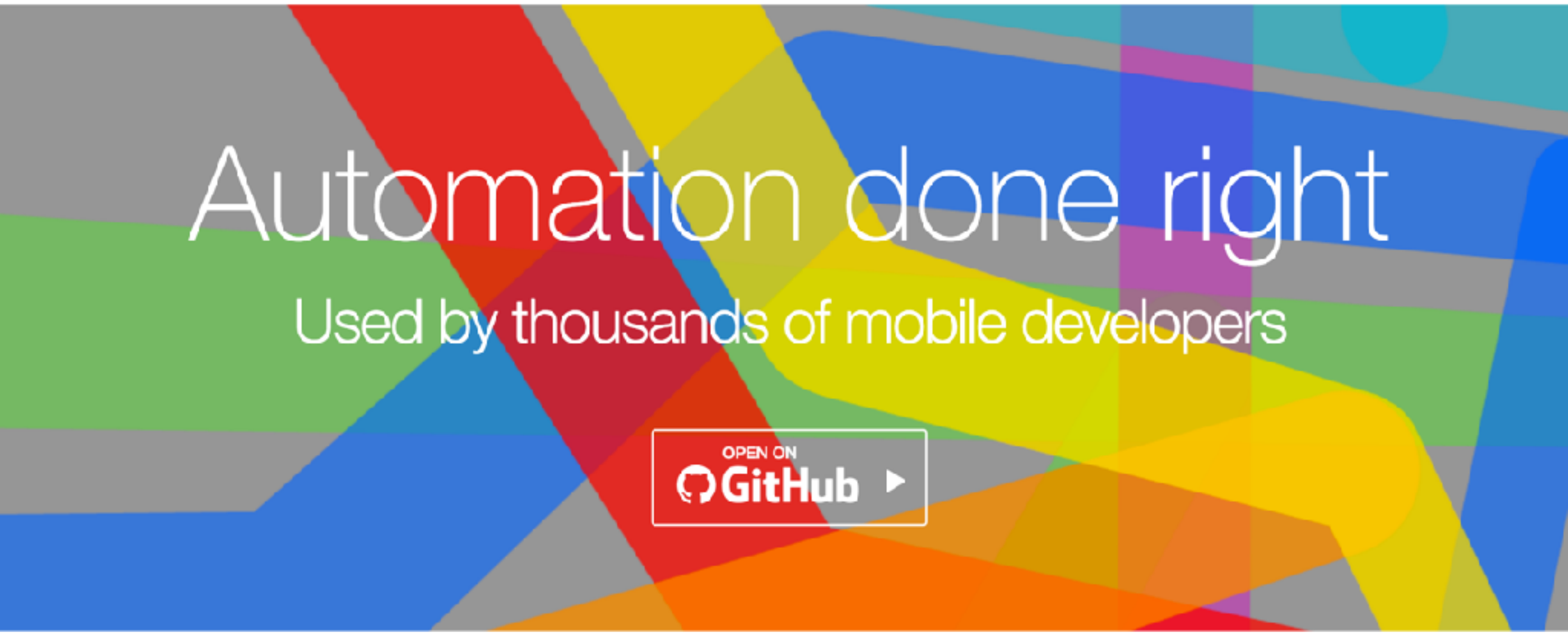



Google play

# Fastlane.tools

fastlane

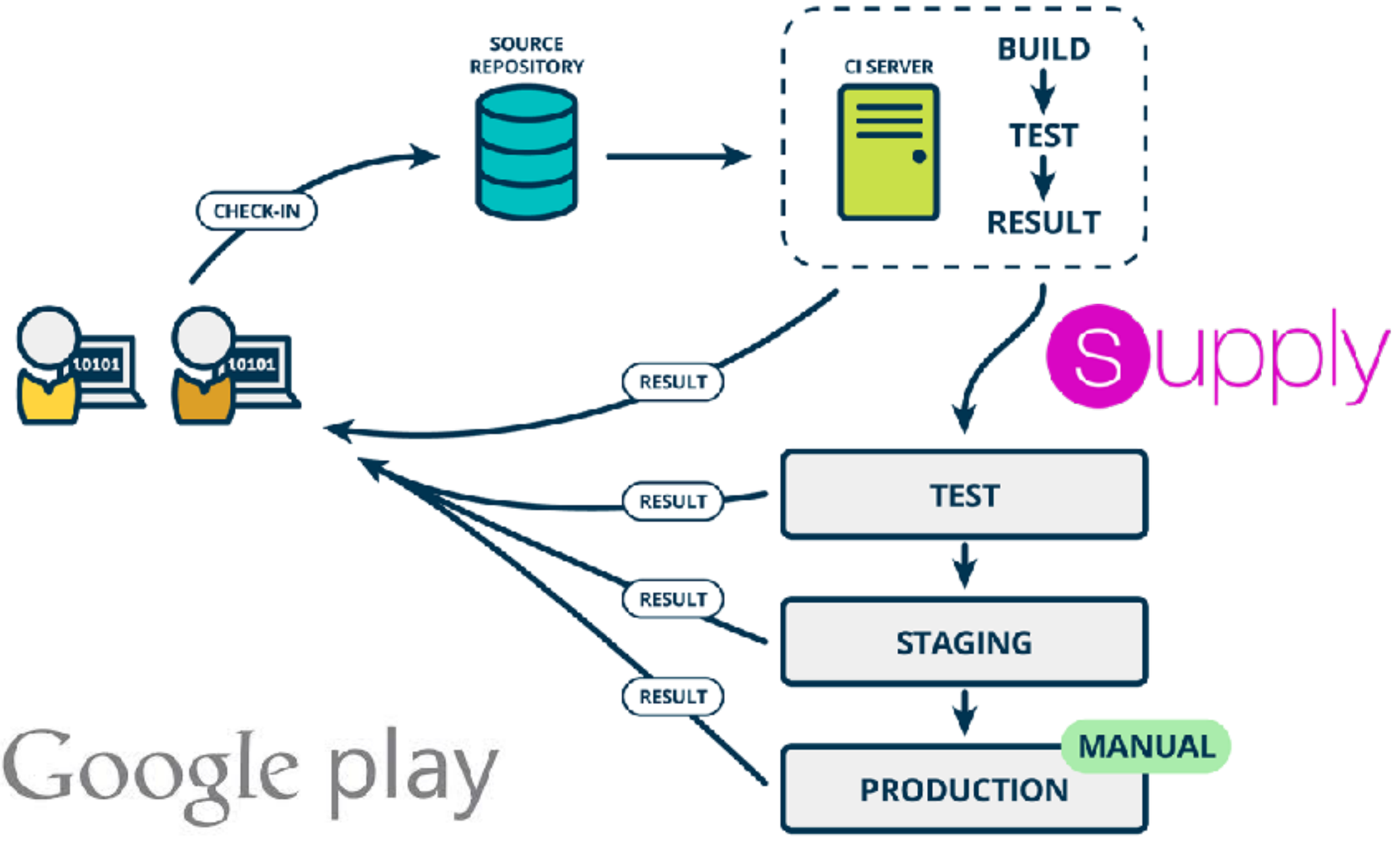
GitHub



# Installation

$sudo gem install fastlane --verbose

# Continuous Delivery



# Workshop with Supply

สมัคร Developer account

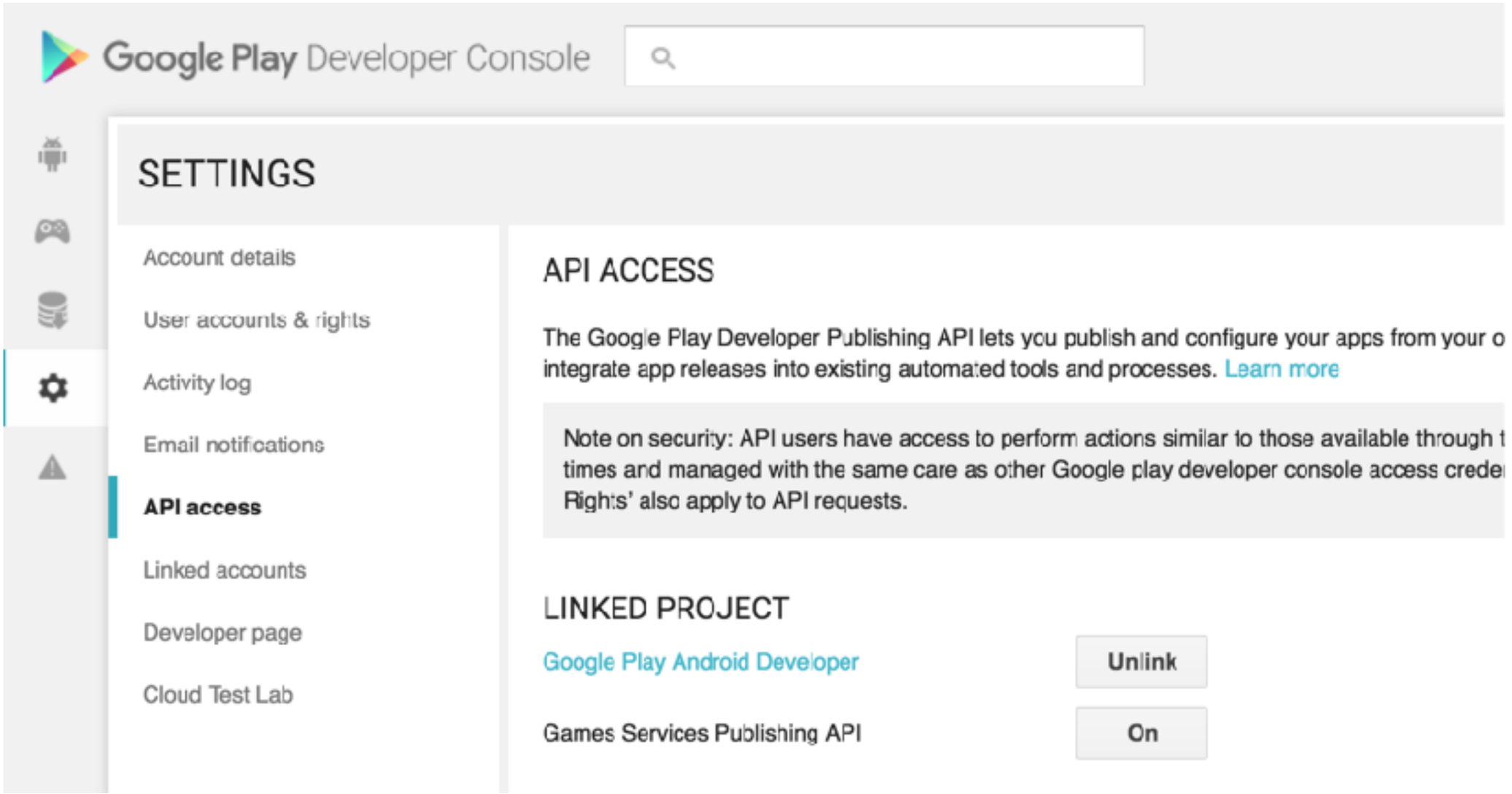
สร้าง Service account สำหรับ publish api

สร้าง Service account key

Upload APK to Google Play

# Create service account

https://developers.google.com/android-publisher/



# Create service account

## SERVICE ACCOUNTS

Service accounts allow access to the Google Play Developer Publishing API on behalf of an application rath
accessing the API from an unattended server, such as an automated build server (e.g. Jenkins). All actions
You can configure fine grained permissions for the service account on the 'User Accounts & Rights' page.

| EMAIL | PERMISSIONS |
| --- | --- |
| @developer.gserviceaccount.com | Grant access |

Create Service Account

# Create service account key

API key
Identifies your project using a simple API key to check quota and access.
For APIs like Google Translate.

OAuth client ID
Requests user consent so your app can access the user's data.
For APIs like Google Calendar.

Service account key
Enables server-to-server, app-level authentication using robot accounts.
For use with Google Cloud APIs.

Help me choose
Asks a few questions to help you decide which type of credential to use.

Create credentials

# Create service account key

Create service account key

**Service account**

New service account

**Service account name**

fastlane-01

**Service account ID**

fastlane-01 @api-5228745590174066006-637549.iam.gserviceaccou

**Key type**

Downloads a file that contains the private key. Store the file securely because this key can't be recovered if lost.

- (●) JSON
  Recommended
- ( ) P12
  For backward compatibility with code using the P12 format

Create Cancel

# Create fastlane project

$fastlane init

# Build signed APK

$./gradlew assembleRelease

# Upload APK to Google play

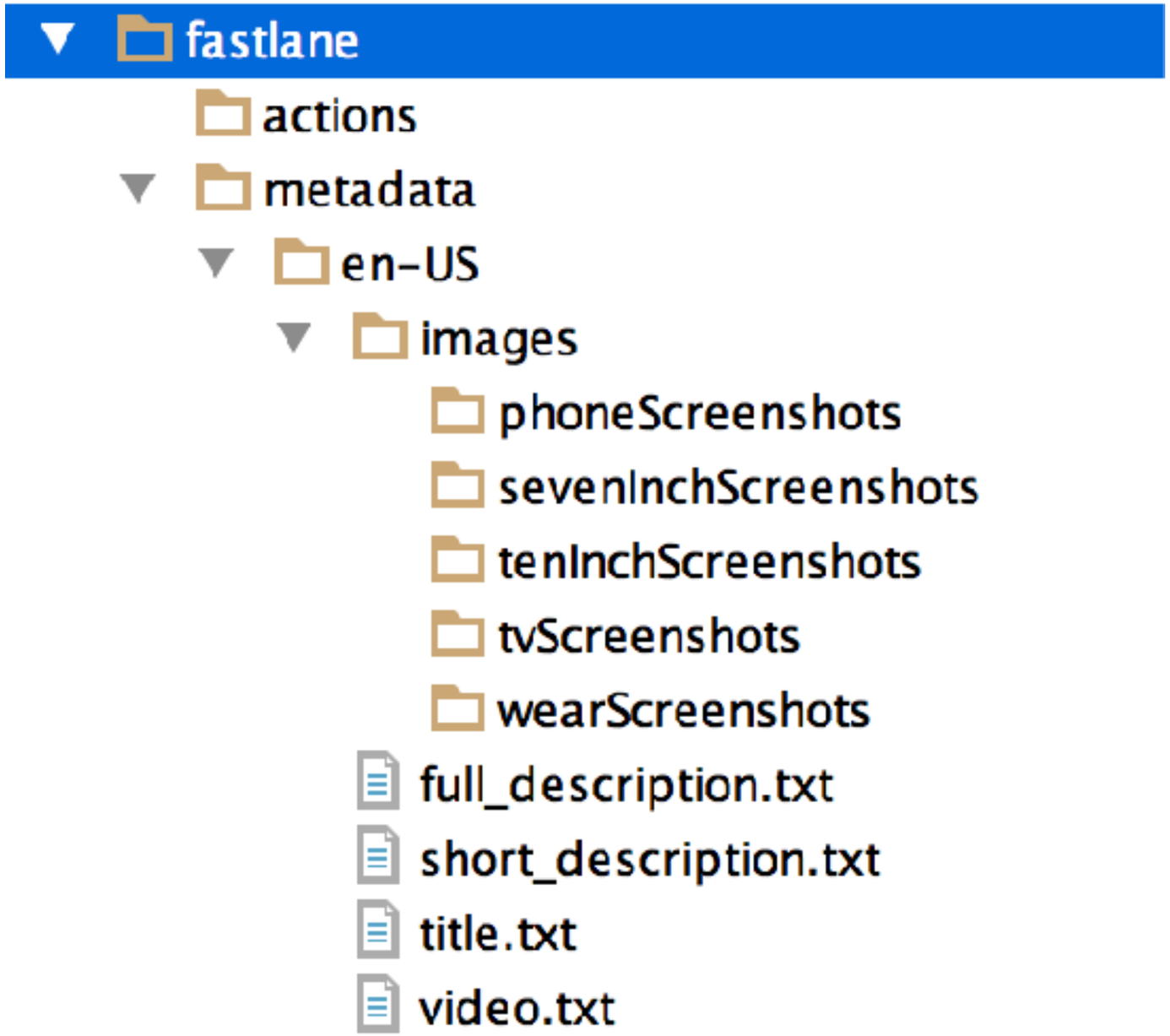
Manual process

# Initial supply

## $supply init

```
Writing to metadata/en-US/title.txt...
Writing to metadata/en-US/short_description.txt...
Writing to metadata/en-US/full_description.txt...
Writing to metadata/en-US/video.txt...
Due to the limit of the Google Play API `supply` can't download your existing screenshots...
Due to the limit of the Google Play API `supply` can't download your existing feature graphics...
Downloading icon for en-US...
Downloading promoGraphic for en-US...
Downloading tvBanner for en-US...
Successfully stored metadata in 'metadata'
```

# Fastlane structure



# Deploy to alpha

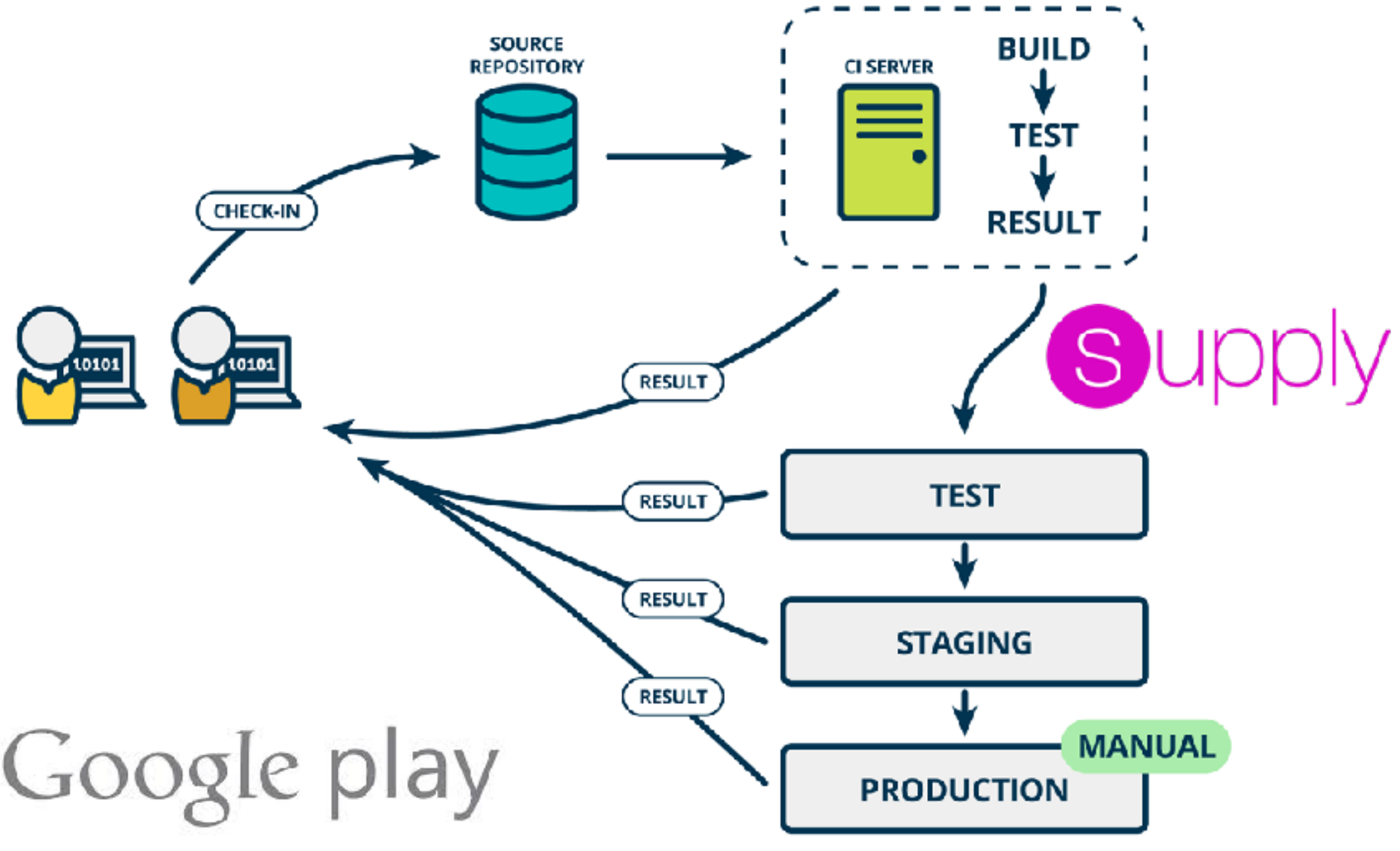
```
desc "Deploy a new version to the Google Play"
lane :deploy do
  gradle(task: "assembleRelease")
  supply(track: "alpha")
end
```

# Deploy to alpha

## $fastlane deploy

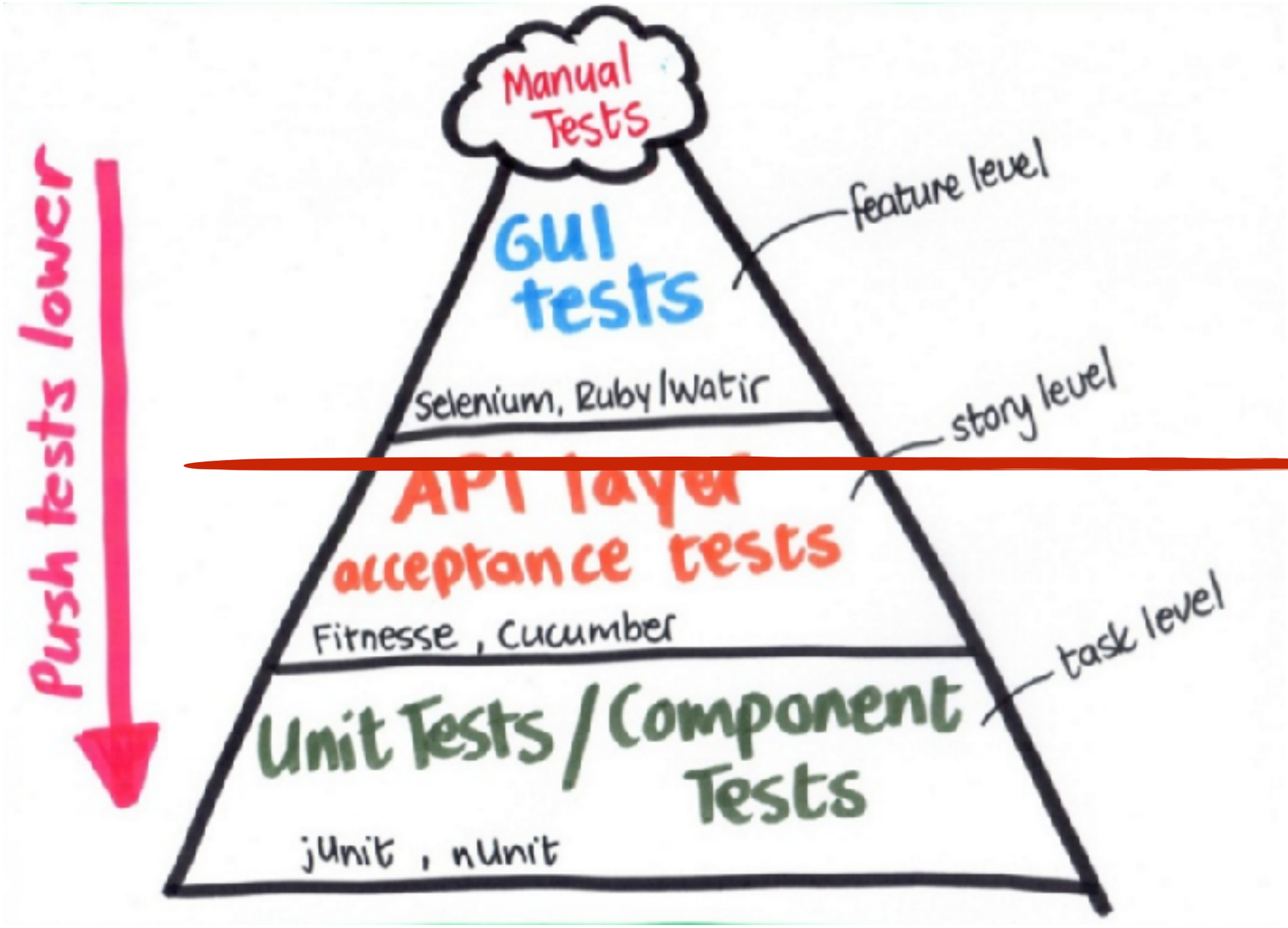
```
+----------------------+-----------------------------------+
|                Summary for supply 0.6.2                  |
+----------------------+-----------------------------------+
| track                | alpha                             |
| package_name         | demo.somkiat.demounittest         |
| rollout              | 1.0                               |
| json_key             | xxx-126555491470.json             |
| apk                  | app/build/outputs/apk/app-release.apk |
| skip_upload_apk      | false                             |
| skip_upload_metadata | false                             |
| skip_upload_images   | false                             |
| skip_upload_screenshots | false                          |
+----------------------+-----------------------------------+

[12:21:33]: Preparing apk at path 'app/build/outputs/apk/app-release.apk' for upload...
[12:22:04]: Updating track 'alpha'...
[12:22:06]: Uploading all changes to Google Play...
[12:22:09]: Successfully finished the upload to Google Play

+------+-------------------------------------+-------------+
|                    fastlane summary                       |
+------+-------------------------------------+-------------+
| Step | Action                              | Time (in s) |
+------+-------------------------------------+-------------+
| 1    | Verifying required fastlane version | 0           |
| 2    | default_platform                    | 0           |
| 3    | gradle                              | 15          |
| 4    | supply                              | 38          |
+------+-------------------------------------+-------------+

[12:22:09]: fastlane.tools finished successfully 🎉
```

# Let’s fun


SOURCE
REPOSITORY
CI SERVER
BUILD
TEST
RESULT
CHECK-IN
10101
10101
RESULT
RESULT
RESULT
RESULT
TEST
STAGING
PRODUCTION
MANUAL
Supply
Google play

# Mocking

# Testing for android



# We need small test !!

| Feature | Small | Medium | Large |
|---|---|---|---|
| Network access | No | localhost only | Yes |
| Database | No | Yes | Yes |
| File system access | No | Yes | Yes |
| Use external systems | No | Discouraged | Yes |
| Multiple threads | No | Yes | Yes |
| Sleep statements | No | Yes | Yes |
| System properties | No | Yes | Yes |
| Time limit (seconds) | 60 | 300 | 900+ |

http://googletesting.blogspot.com/2010/12/test-sizes.html

# Interact with others

Classes

Shared preferences

Time

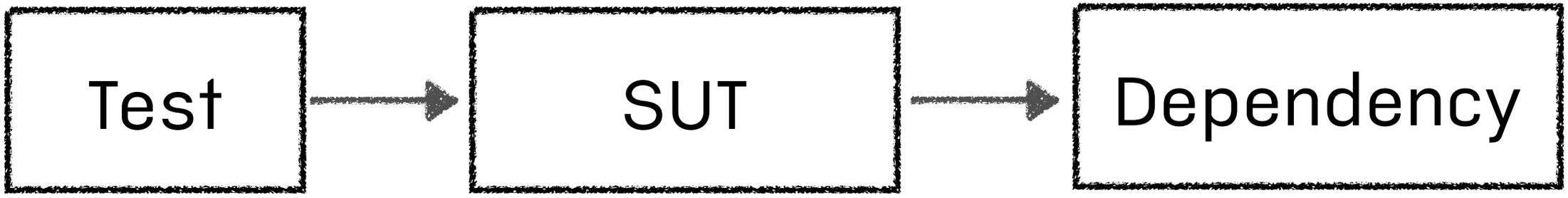
SQLite database

Network

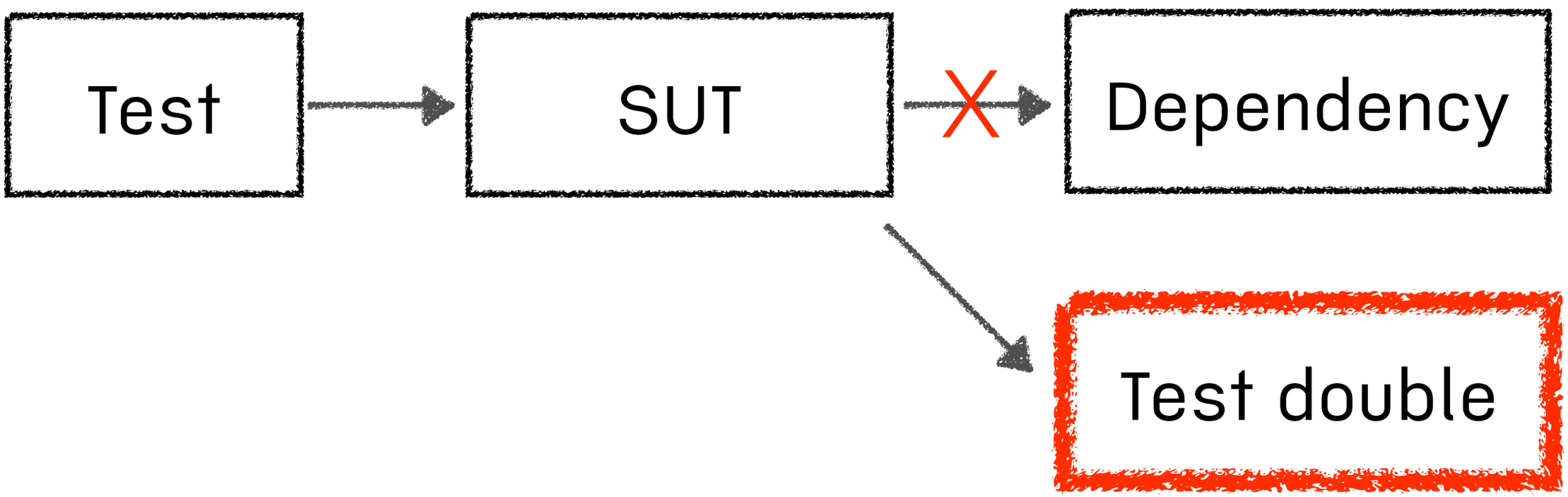
# Interact with others

How to break dependencies ?

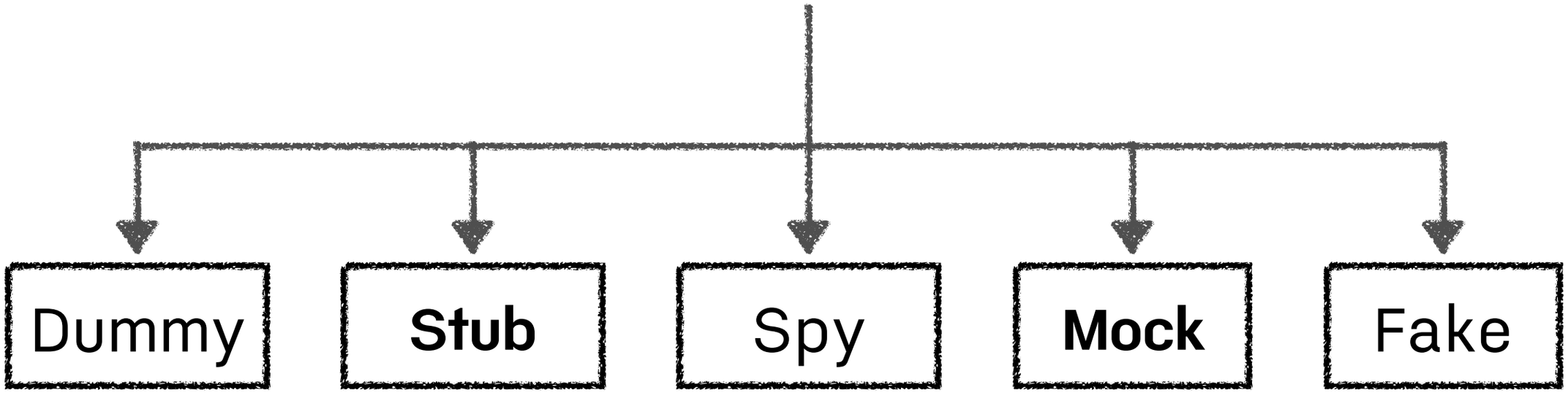
# Interact with others



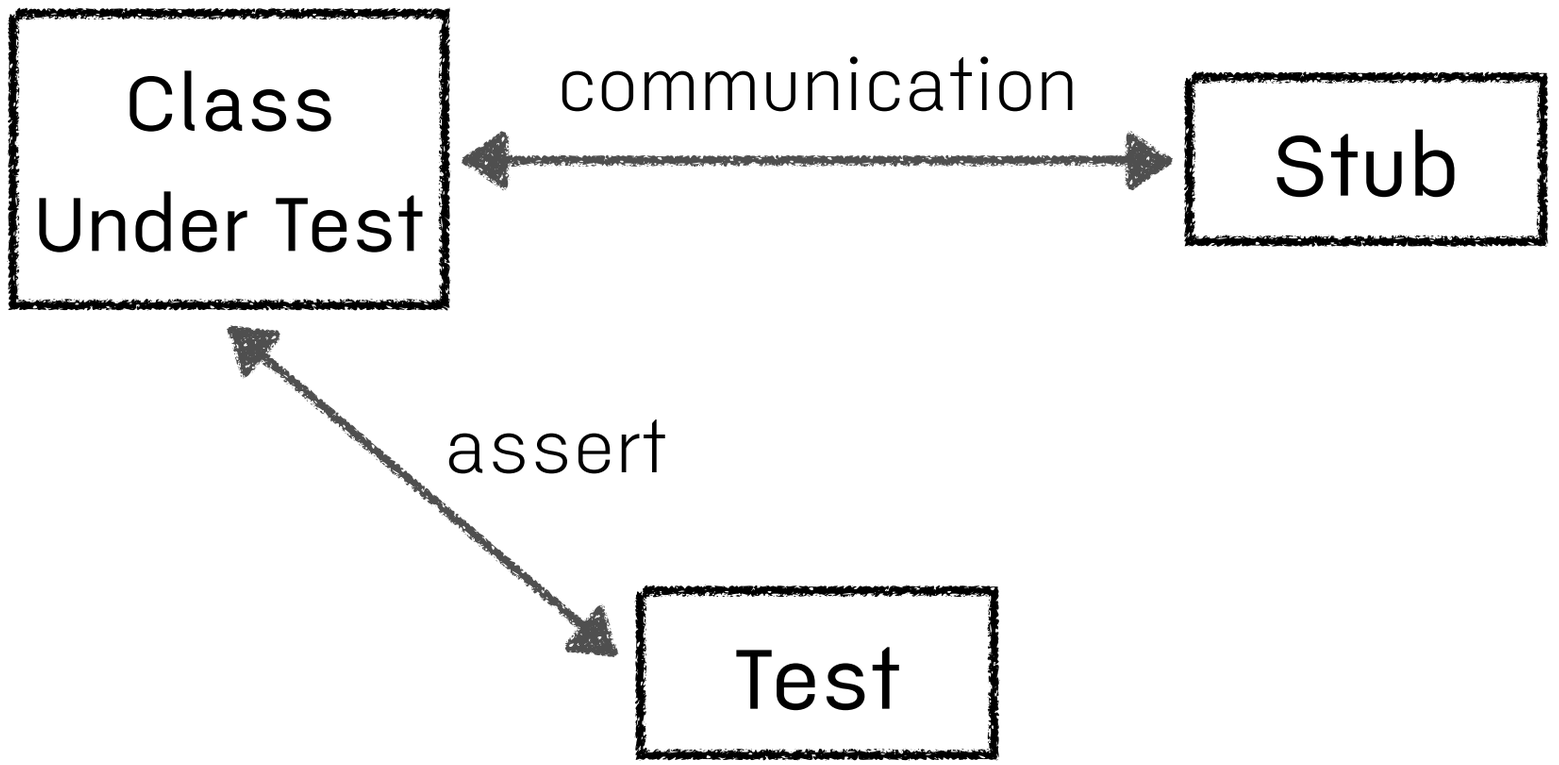
# Interact with others



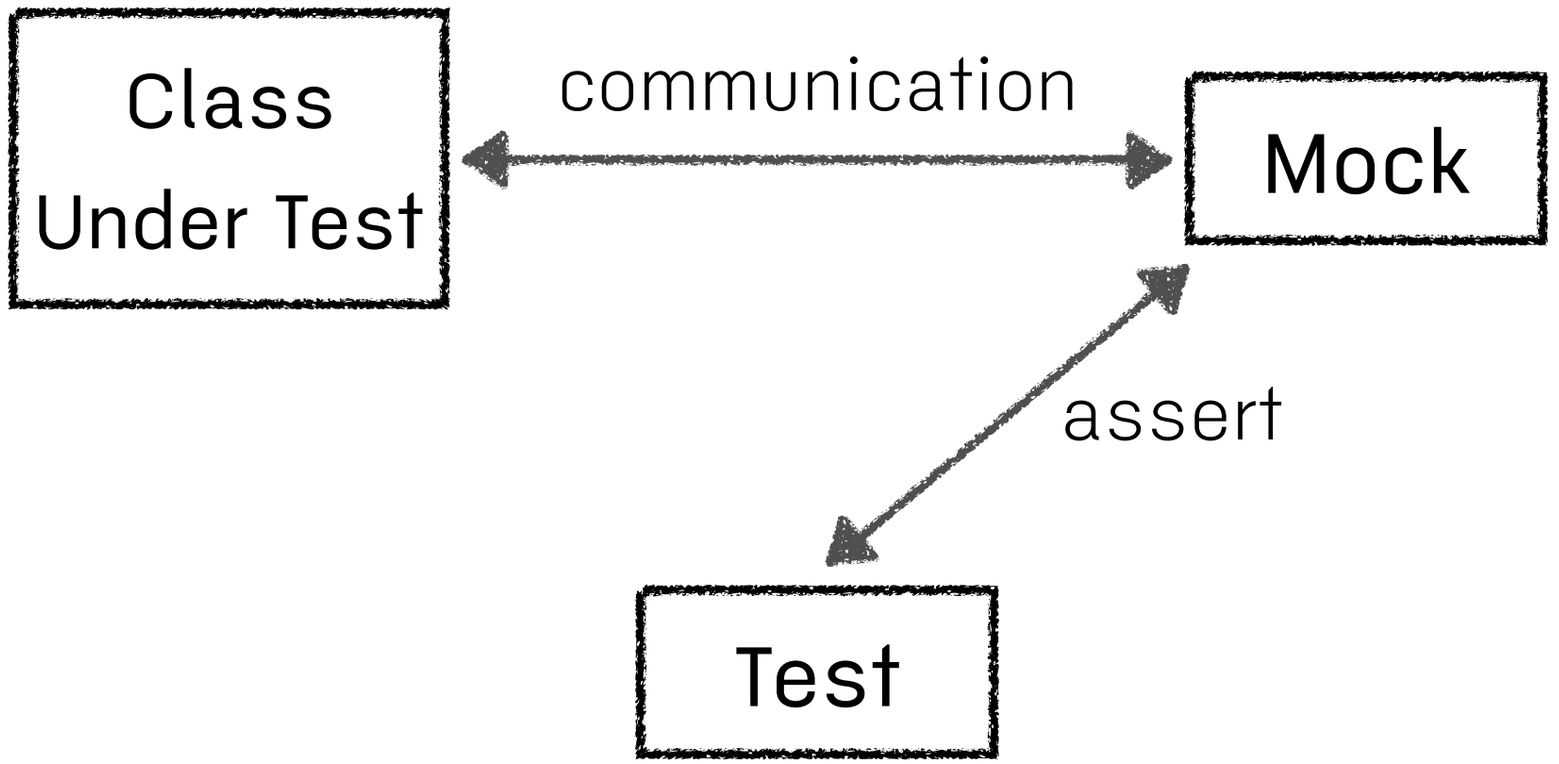
# Test double



# Stub



# Mock

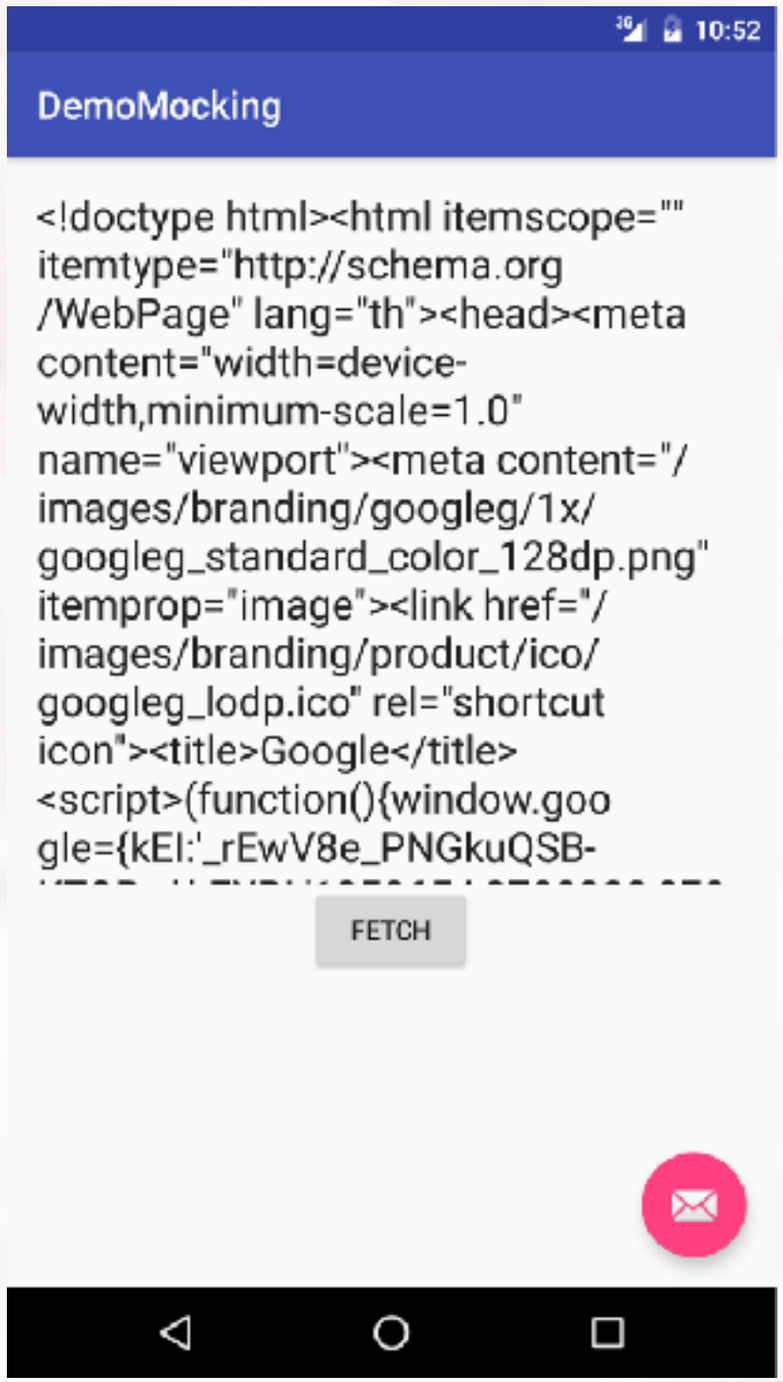


# Workshop with Mocking

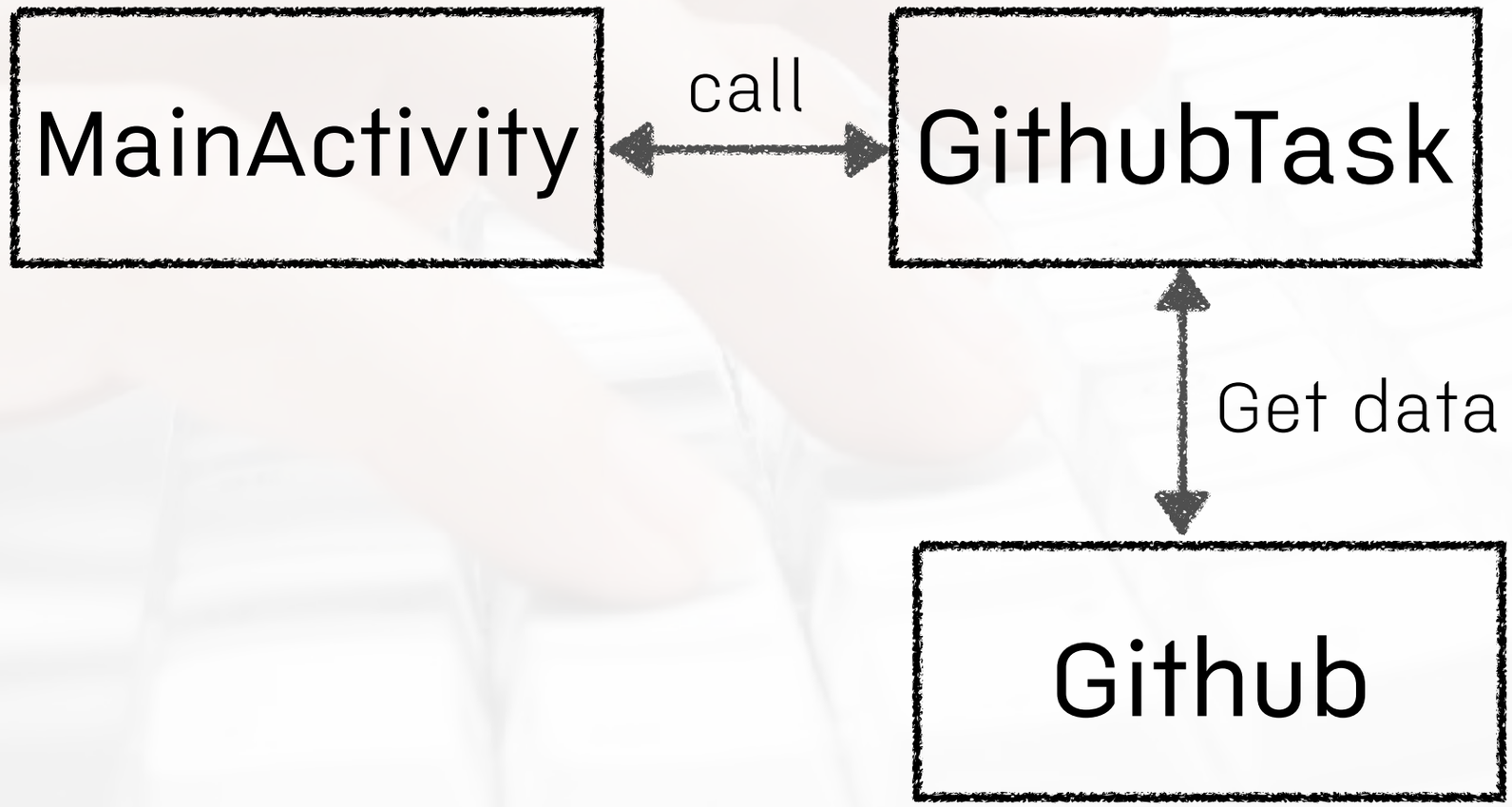
Network with HttpURLConnection

Network with Retrofit2

# HttpURLConnection



# HttpURLConnection



# Working with Mockito

```
dependencies {

    //Unit test and Mocking
    testCompile 'junit:junit:4.12'
    testCompile 'org.hamcrest:hamcrest-library:1.3'
    testCompile 'org.mockito:mockito-core:1.10.19'
```

# Example :: RunWith Mockito

```java
@RunWith(MockitoJUnitRunner.class)
public class GithubTaskTest {

    public GitHubTask gitHubTask = Mockito.mock(GitHubTask.class);

    @Before
    public void setUp() {
        try {
            Mockito.when(gitHubTask.loadFromNetwork("http://www.google.com"))
                    .thenReturn("<!doctype html><html itemscope=\"\" itemtype=\"h
        } catch (IOException e) {
            fail();
        }
    }

    @Test
    public void setGitHubTaskTest_ReturnsTrue() {
        try {
            assertThat(gitHubTask.loadFromNetwork("http://www.google.com"),
                    containsString("doctype"));
        } catch (IOException e) {
            fail();
        }
    }
}
```

# Example :: Create mock

```
@RunWith(MockitoJUnitRunner.class)
public class GithubTaskTest {

    public GitHubTask gitHubTask = Mockito.mock(GitHubTask.class);

    @Before
    public void setUp() {
        try {
            Mockito.when(gitHubTask.loadFromNetwork("http://www.google.com"))
                    .thenReturn("<!doctype html><html itemscope=\"\" itemtype=\"h
        } catch (IOException e) {
            fail();
        }
    }

    @Test
    public void setGitHubTaskTest_ReturnsTrue() {
        try {
            assertThat(gitHubTask.loadFromNetwork("http://www.google.com"),
                    containsString("doctype"));
        } catch (IOException e) {
            fail();
        }
    }
}
```

# Example :: Create stub

```
@RunWith(MockitoJUnitRunner.class)
public class GithubTaskTest {

    public GitHubTask gitHubTask = Mockito.mock(GitHubTask.class);

    @Before
    public void setUp() {
        try {
            Mockito.when(gitHubTask.loadFromNetwork("http://www.google.com"))
                    .thenReturn("<!doctype html><html itemscope=\"\" itemtype=\"h
        } catch (IOException e) {
            fail();
        }
    }

    @Test
    public void setGitHubTaskTest_ReturnsTrue() {
        try {
            assertThat(gitHubTask.loadFromNetwork("http://www.google.com"),
                    containsString("doctype"));
        } catch (IOException e) {
            fail();
        }
    }
}
```

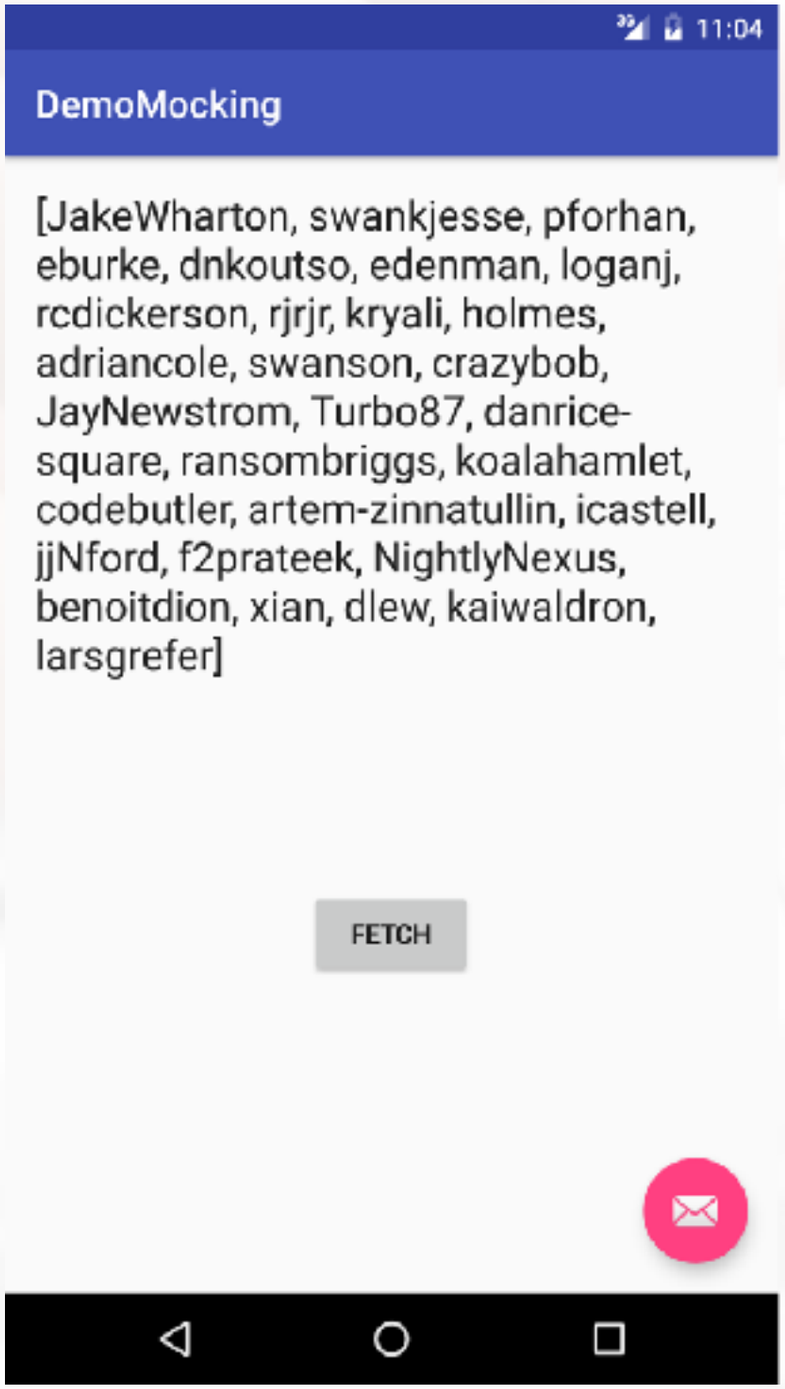
# Example :: Write test

```
@RunWith(MockitoJUnitRunner.class)
public class GithubTaskTest {

    public GitHubTask gitHubTask = Mockito.mock(GitHubTask.class);

    @Before
    public void setUp() {
        try {
            Mockito.when(gitHubTask.loadFromNetwork("http://www.google.com"))
                    .thenReturn("<!doctype html><html itemscope=\"\" itemtype=\"h
        } catch (IOException e) {
            fail();
        }
    }

    @Test
    public void setGitHubTaskTest_ReturnsTrue() {
        try {
            assertThat(gitHubTask.loadFromNetwork("http://www.google.com"),
                    containsString("doctype"));
        } catch (IOException e) {
            fail();
        }
    }
}
```

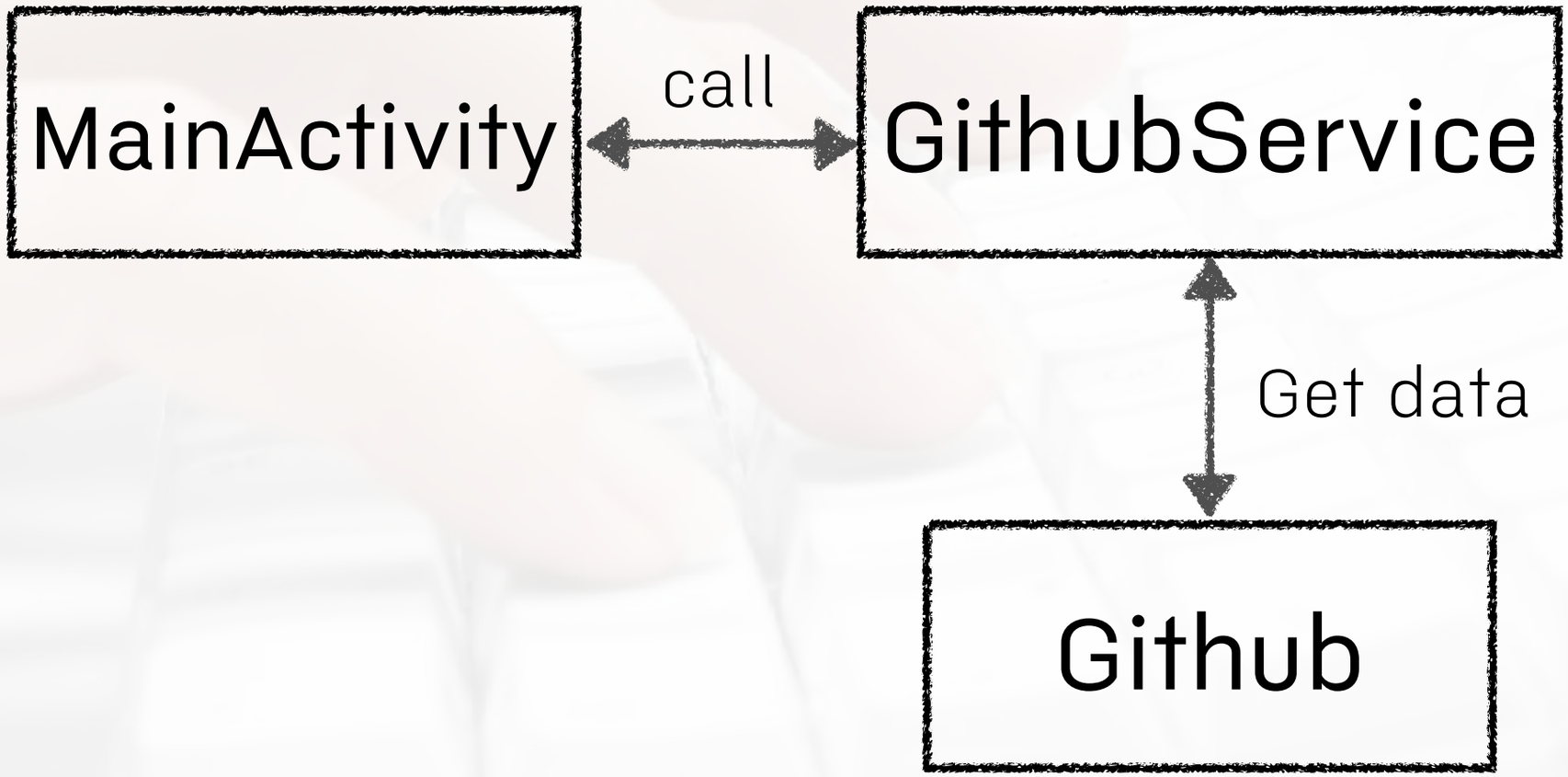
# Working with hardcode

ลองทำดูสิ !!

# Retrofit2


11:04
DemoMocking
[JakeWharton, swankjesse, pforhan, eburke, dnkoutso, edenman, loganj, rcdickerson, rjrjr, kryali, holmes, adriancole, swanson, crazybob, JayNewstrom, Turbo87, danrice-square, ransombriggs, koalahamlet, codebutler, artem-zinnatullin, icastell, jjNford, f2prateek, NightlyNexus, benoitdion, xian, dlew, kaiwaldron, larsgrefer]
FETCH

# Retrofit2



# Workshop with Retrofits

## MockWebServer

## MockRetrofit

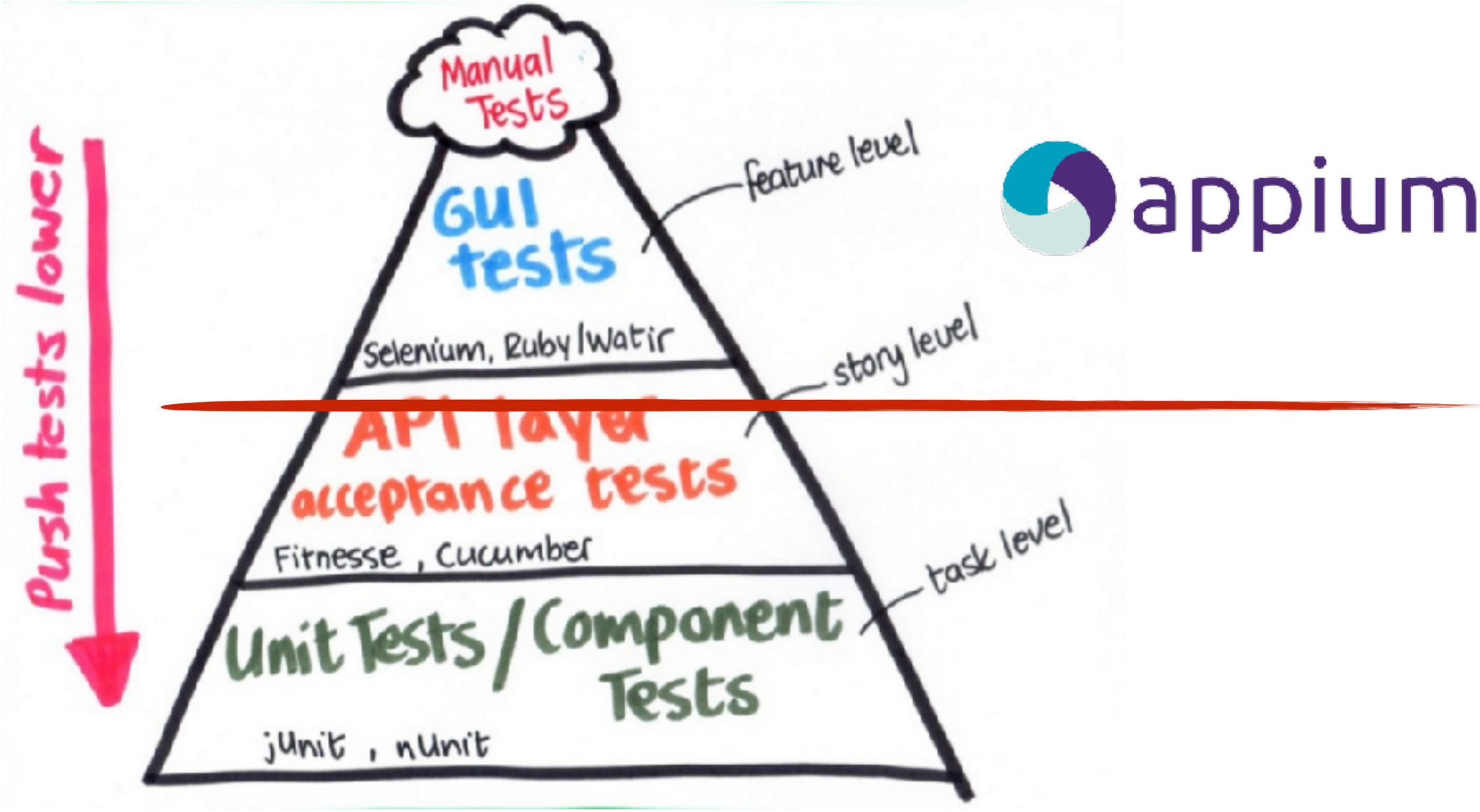
# Resources

https://github.com/up1/course-automated-testing-for-android-app

# New topics

# Appium for Android app


http://appium.io/


Manual Tests
GUI tests
Selenium, Ruby/Watir
API layer
acceptance tests
Fitnesse, Cucumber
Unit Tests / Component Tests
jUnit, nUnit
feature level
story level
task level
Push tests lower
appium

# What is Appium ?

Test automation tool for mobile app

Open source

Support native, hybrid and mobile web

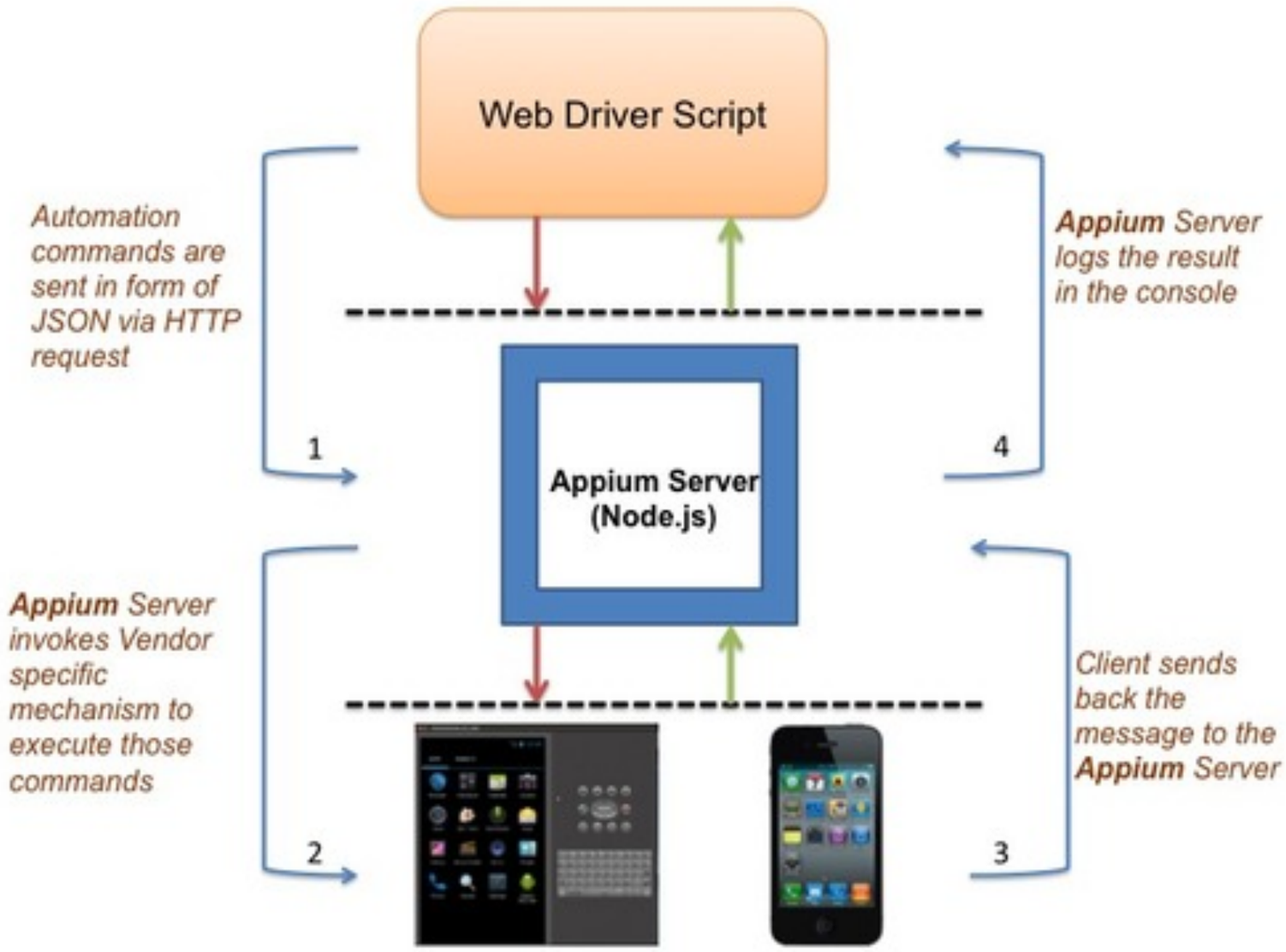
Run on actual device, emulator and simulator

Support android and iOS

# Appium philosophy

You should be able to reuse code between android and iOS

# Appium architecture




https://nishantverma.gitbooks.io/appium-for-android

# Install appium server

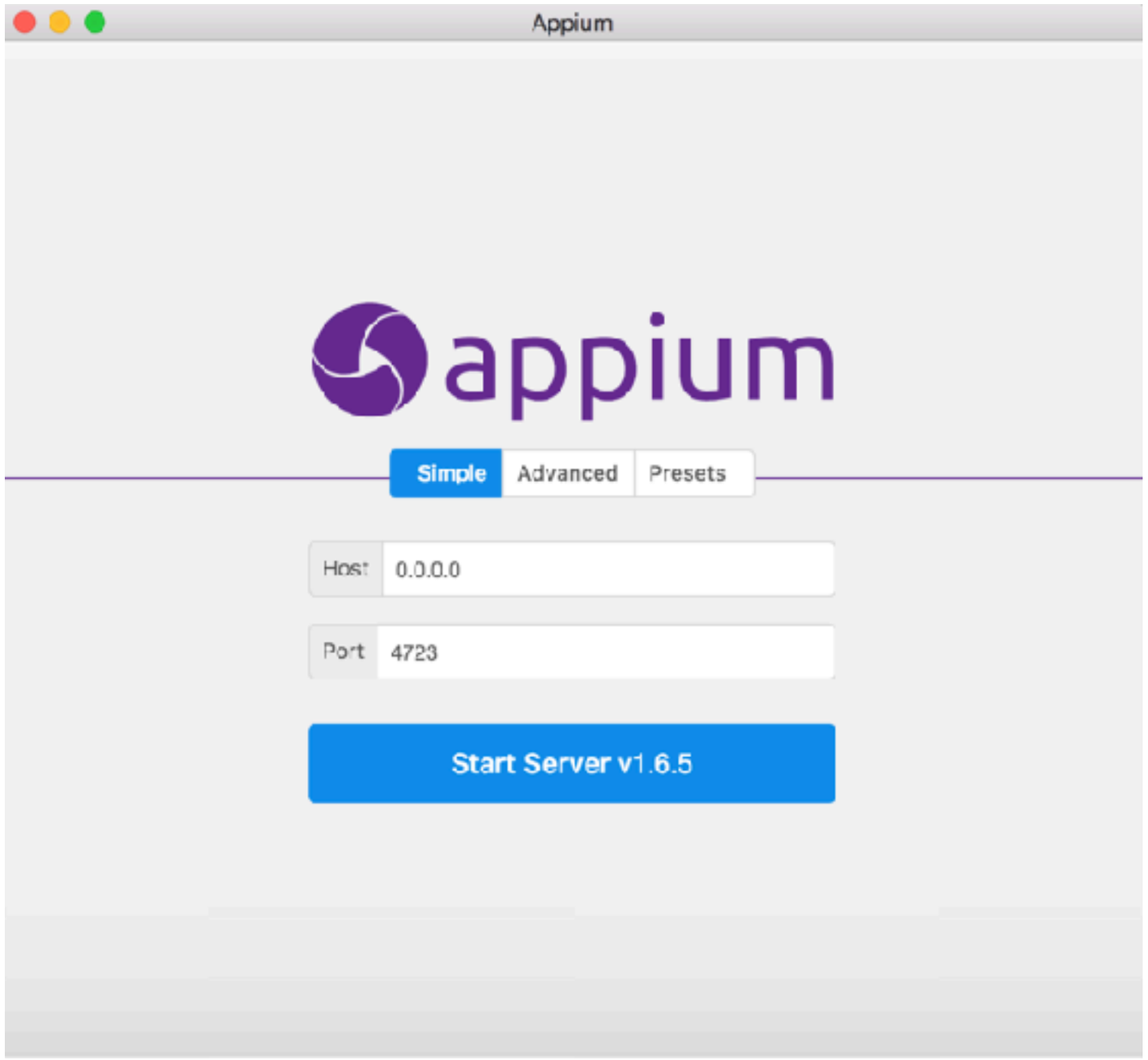
via app

via NPM (Node package manager)

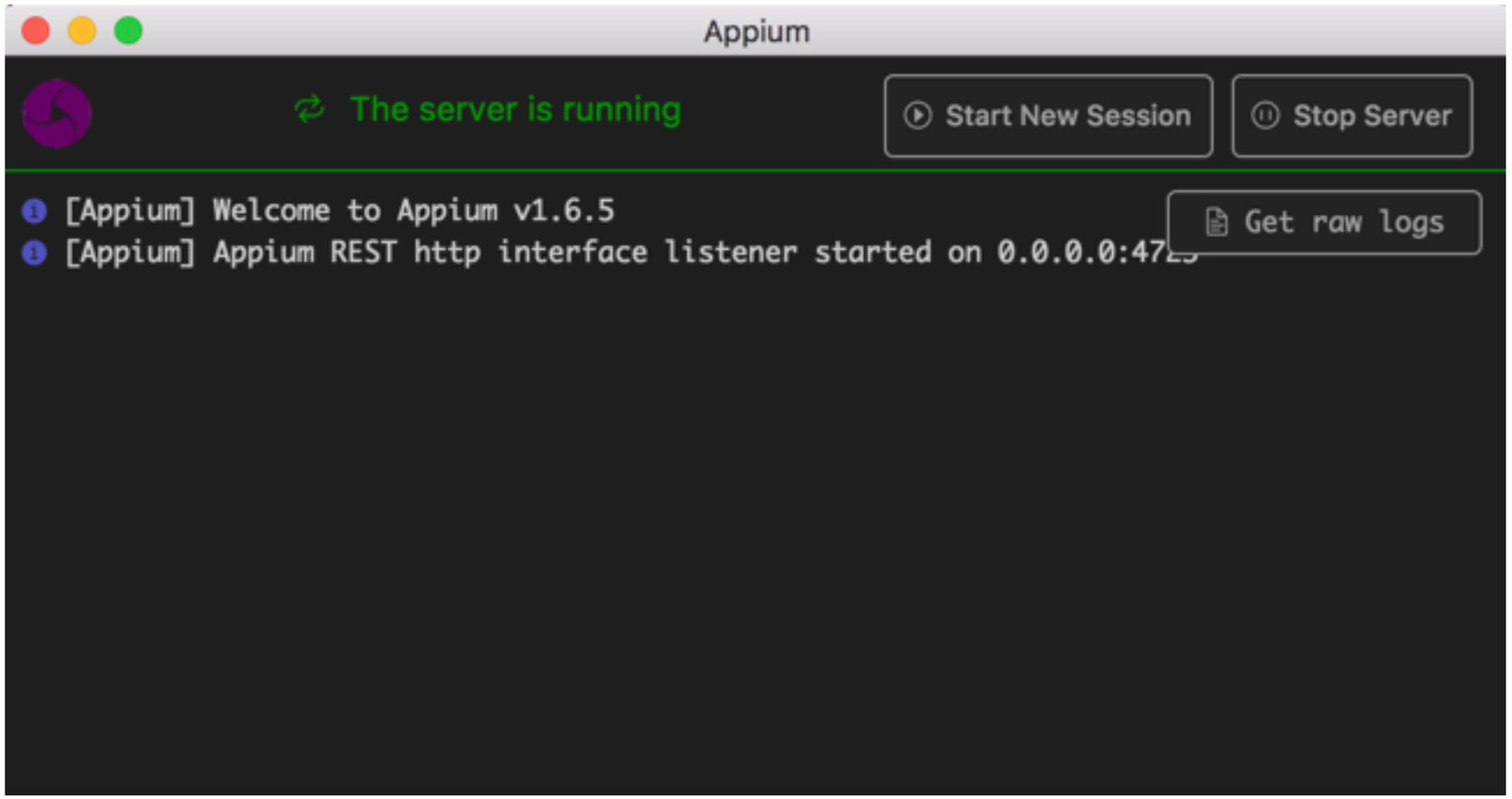
# Install via App

http://appium.io/

# Install via App



# Start server

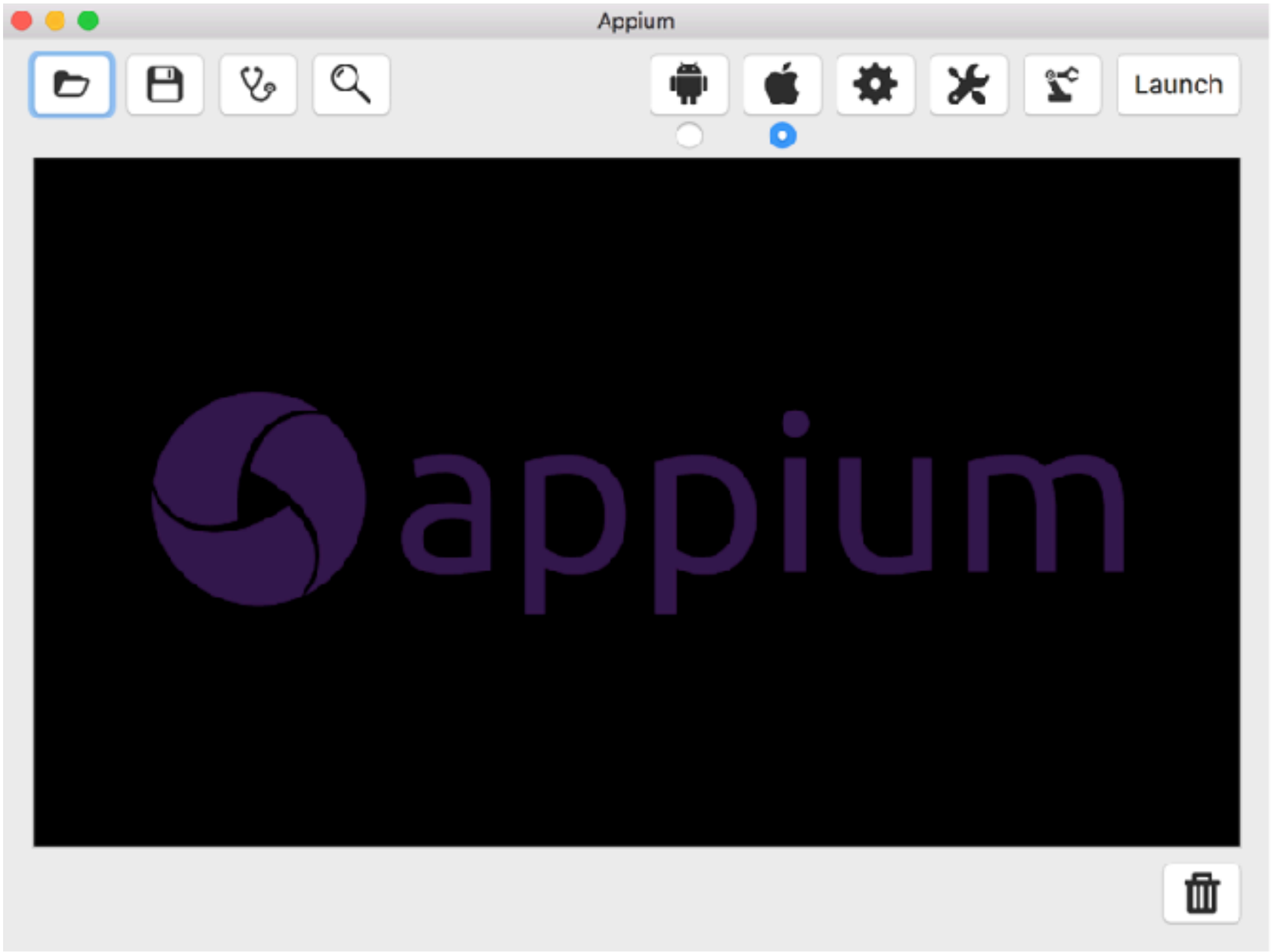


# Install via NPM

$npm install -g appium

$appium

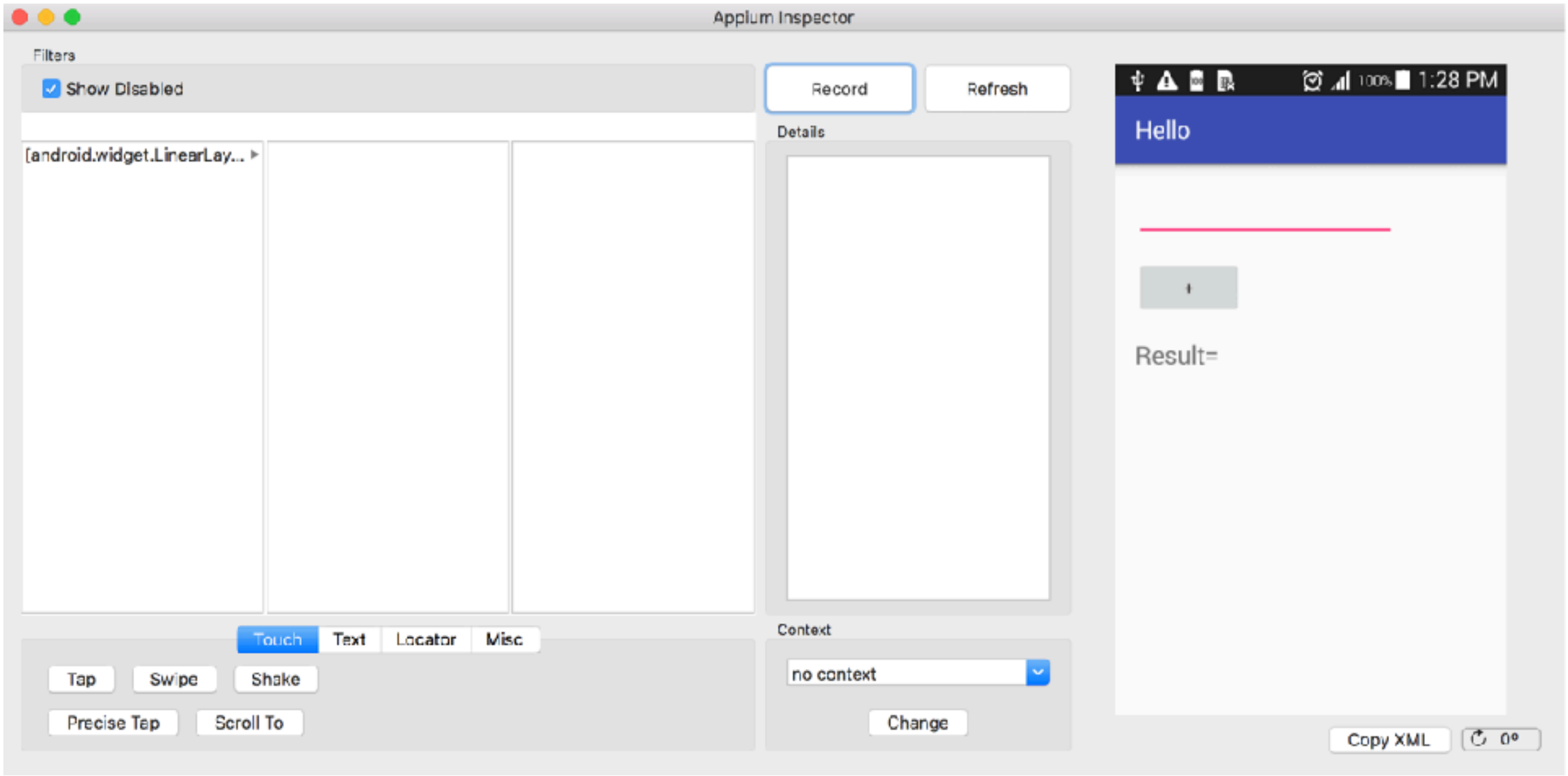
# Install appium GUI




https://bitbucket.org/appium/appium.app/downloads/appium.dmg

# Setting for android

# Appium Inspector



# List of android devices

$./adb devices