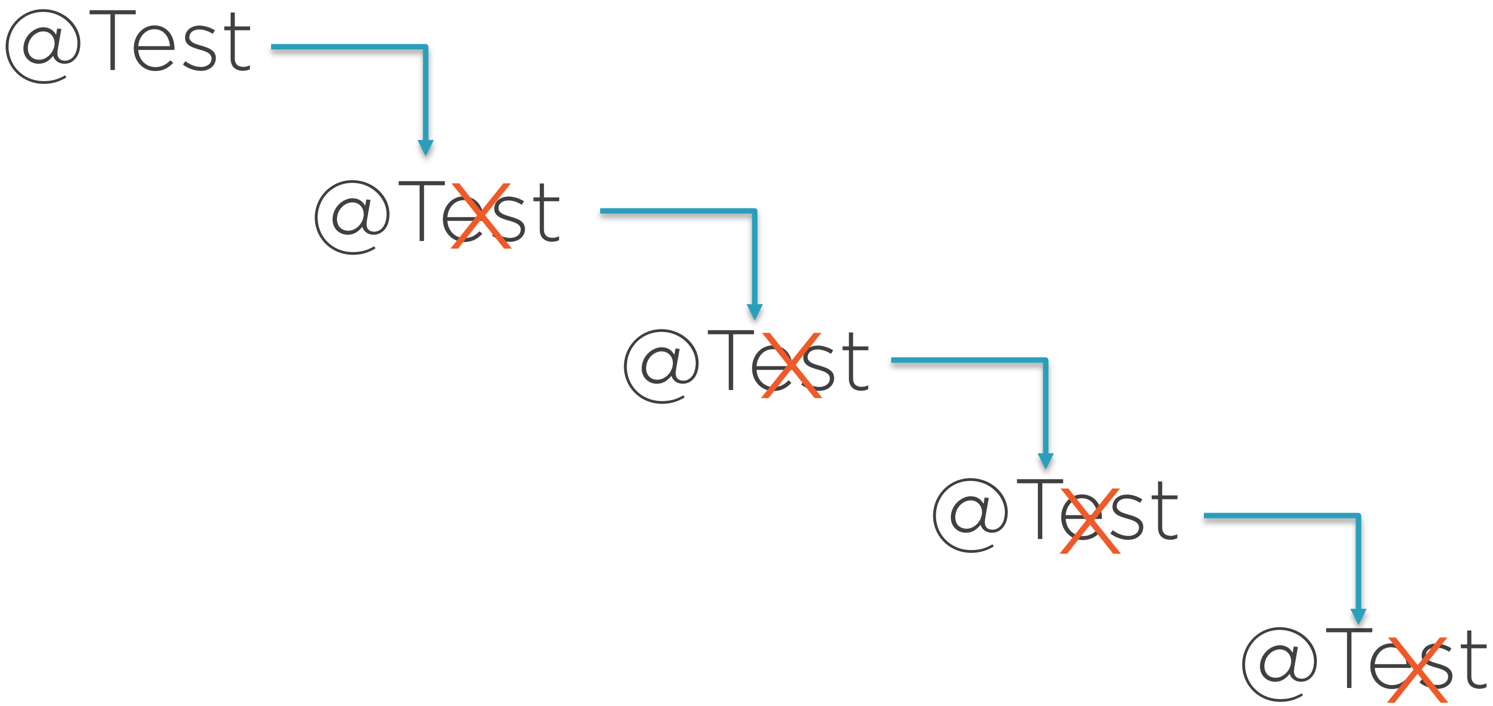
Test Organization

Andrejs Doronins
TEST AUTOMATION ENGINEER

> “Tests should run in any order any number of times and produce the same result using the same data.”

QA Wisdom



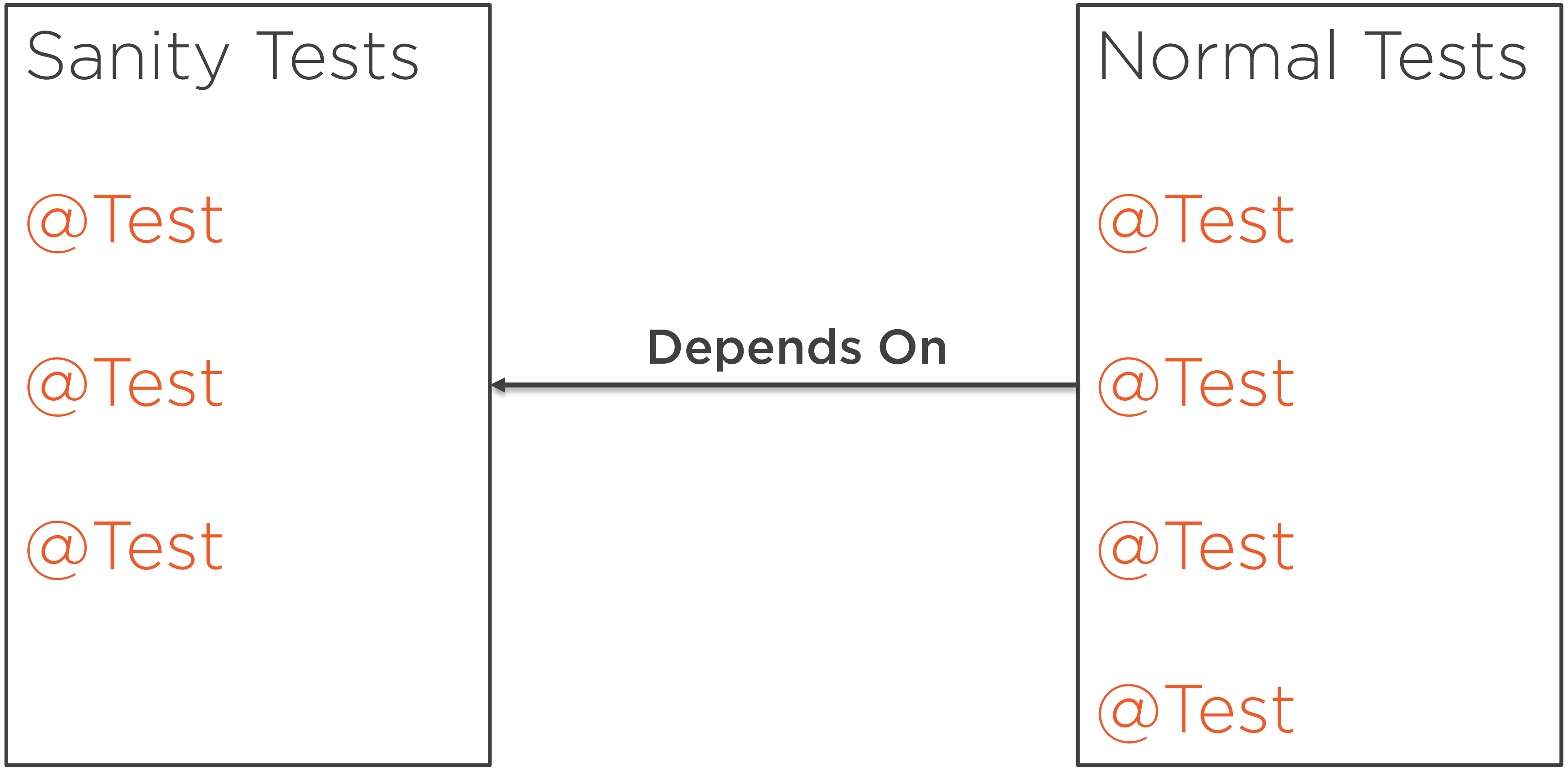
@Test

@Test

@Test

@Test

@Test



Sanity Tests

@Test

@Test

@Test

Depends On

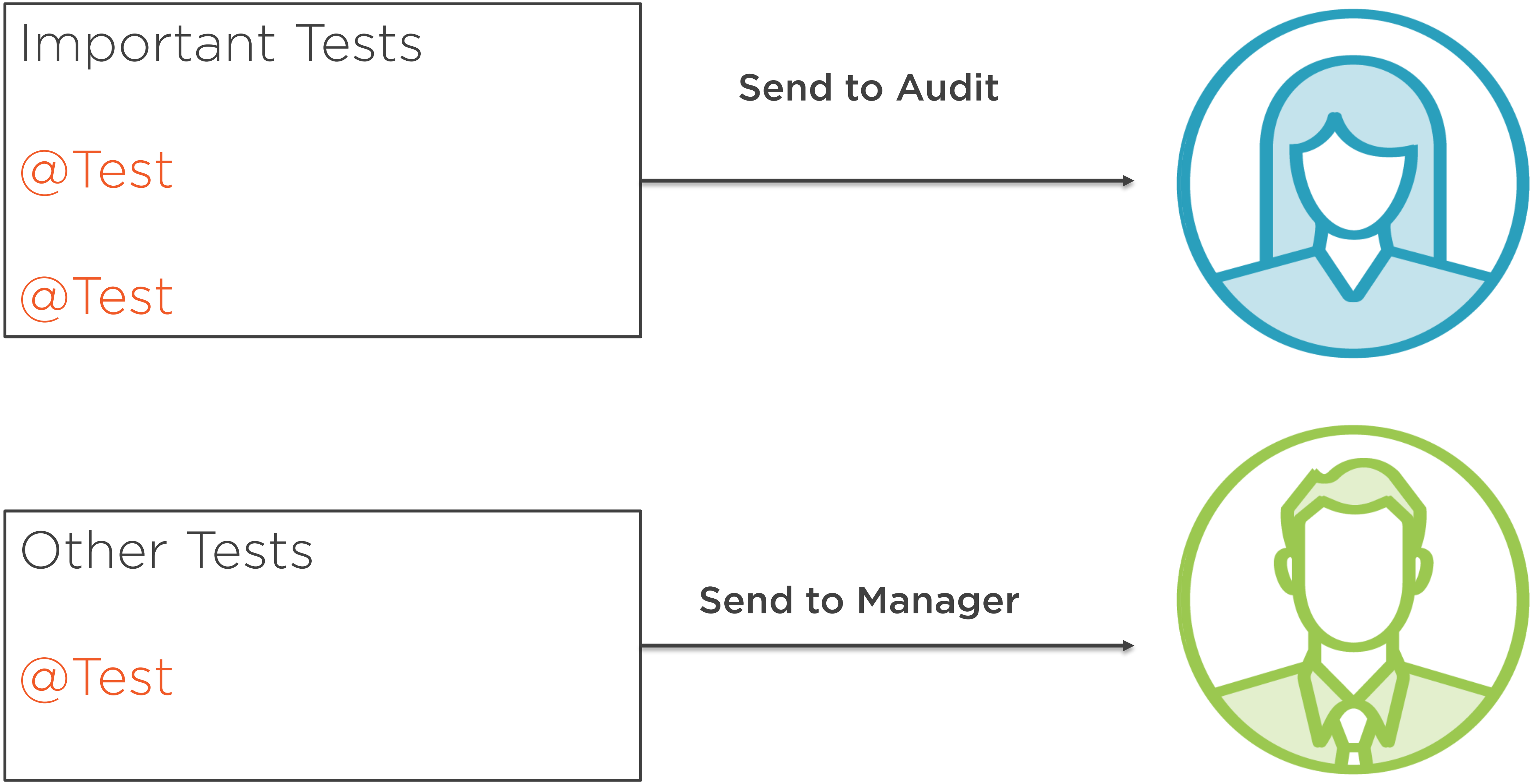
Normal Tests

@Test

@Test

@Test

@Test



Important Tests

@Test

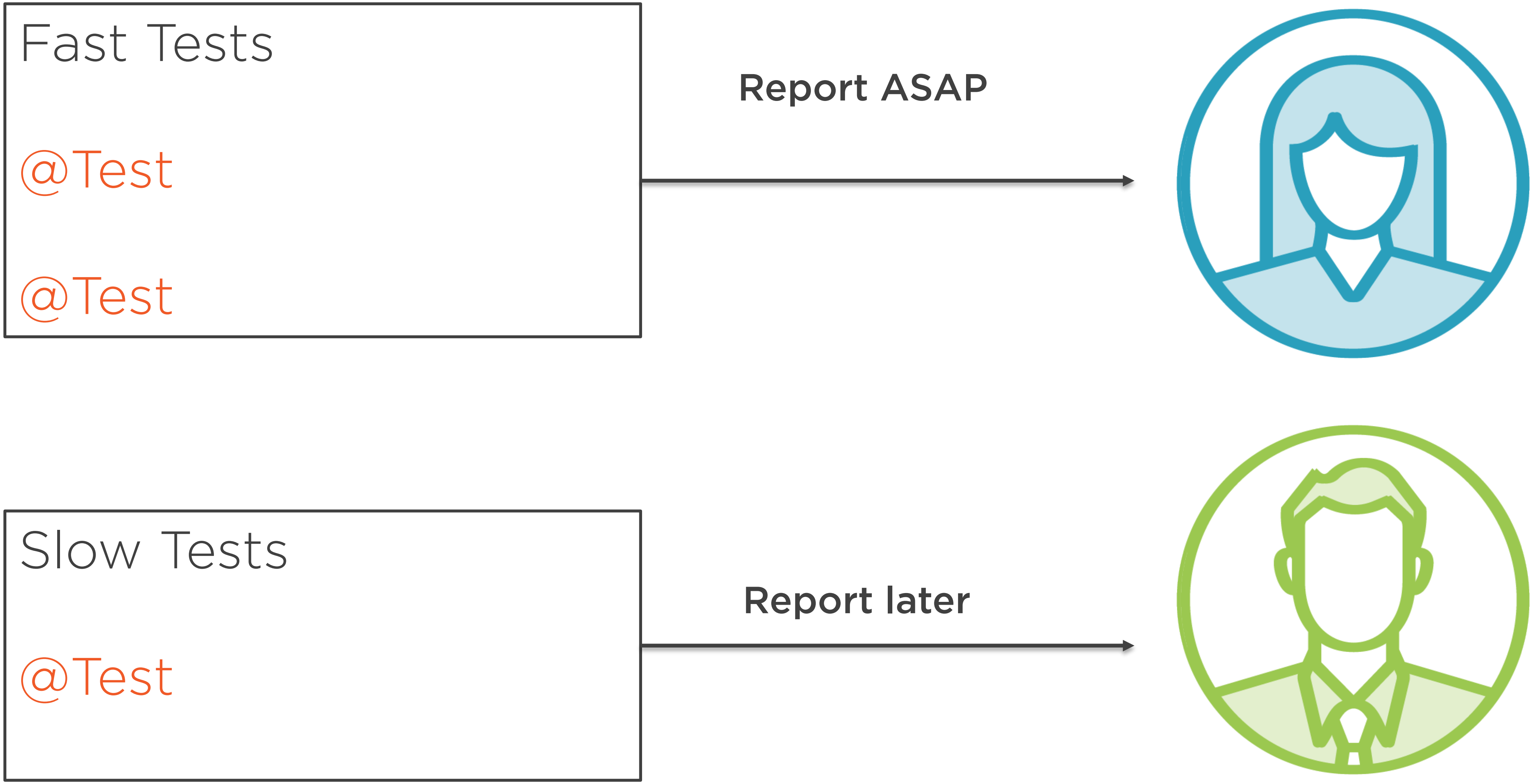
@Test

Send to Audit

Other Tests

@Test

Send to Manager



Fast Tests

@Test

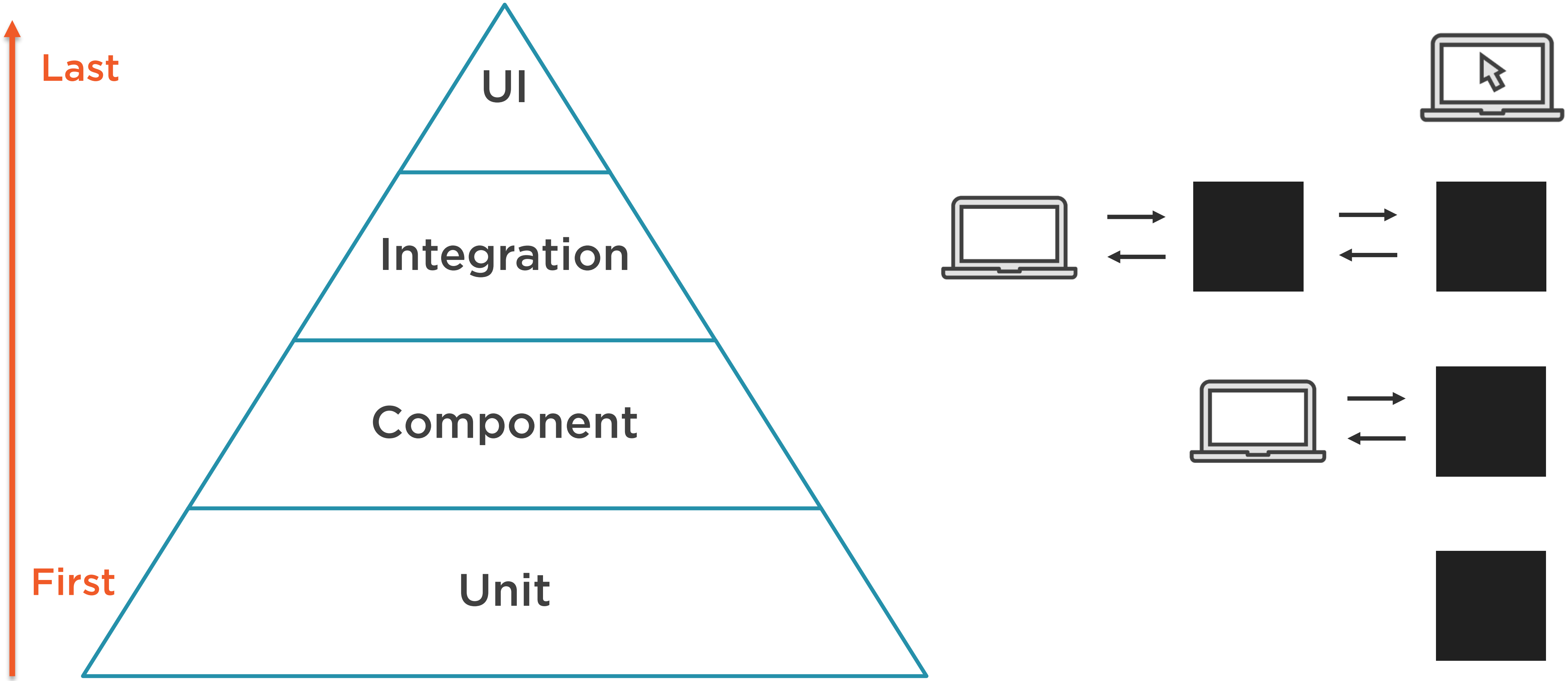
@Test

Report ASAP

Slow Tests

@Test

Report later



Last

UI

Integration

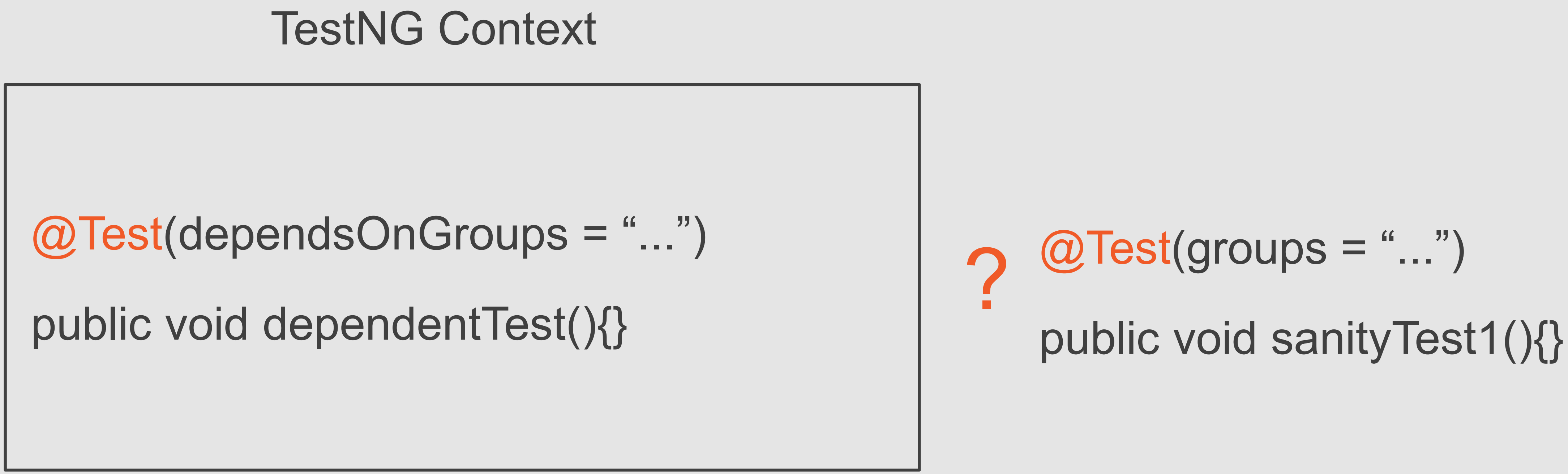
Component

Unit

First

```
@Test(groups = “...”)
public void sanityTest1(){}
```

```
@Test(dependsOnGroups = “...”)
public void dependentTest(){}
```



TestNG Context

```
@Test(dependsOnGroups = “...”)
public void dependentTest(){}
```

?

```
@Test(groups = “...”)
public void sanityTest1(){}
```

TestNG Context

```
@Test(dependsOnGroups = “...”)
public void dependentTest(){}
```

```
@Test(groups = “...”)
public void sanityTest1(){}
```

Setup vs. Sanity

@Before____

Initialize test-related Objects and dependencies

Check test-related resources

@Test(groups="sanity")

Check system basic functionality

```
@Test(groups = ...)
public class SanityTest(){

@Test

@Test

}
```

```
@Test(dependsOnGroups = ...)
public class NormalTest(){

@Test(priority = 1)

@Test(priority = 2)

}
```

Features to Explore

Annotations: @BeforeTest, @Parameters, @Factory

Fields: @Test(invocationCount =..., threadPoolSize =...)

Reports

ITestListener

Summary

Organize test into packages

Group in XML suites

Further TestNG features