TESTNG? JUNIT?

WHAT IS THE BEST CHOICE FOR ME?

Corina Pip

@imalittletester
https://imalittletester.com/
https://github.com/iamalittletester

JUnit 5 vs? TestNG

- JUnit 5 – stable + experimental features
- JUnit 5 – depends heavily on annotations
- TestNG – test attributes and annotations

Dependencies needed

JUnit 5

```
<dependency>

<groupId>org.junit.jupiter</groupId>
  <artifactId>junit-jupiter</artifactId>
  <version>5.7.0</version>
  <scope>test</scope>
</dependency>
```

TestNG

```
<dependency>
  <groupId>org.testng</groupId>
  <artifactId>testng</artifactId>
  <version>7.3.0</version>
  <scope>test</scope>
</dependency>
```

Test method Annotations

JUnit 5

- @Test
- @ParameterizedTest
- @RepeatedTest
- Display name given by @DisplayName

```
@Test
@DisplayName("This desc")
```

TestNG

- @Test
- Display name given by 'description' attribute

```
@Test(description = "This desc")
```

Lifecycle annotations

JUnit 5

- @BeforeAll
- @BeforeEach
- @AfterAll
- @AfterEach

TestNG

- @BeforeClass
- @BeforeMethod
- @AfterClass
- @AfterMethod
- @Before/AfterSuite
- @Before/AfterTest
- @Before/AfterGroup

Naming, access modifiers

JUnit 5

- Name doesn't need to begin or end with 'test'* (*Maven Surefire)
- No need to specify the 'public' access modifier for class or methods; but not private
- Cannot return value
- If you don't want static before and after methods the class will be annotated:

```
@TestInstance(TestInstance.
Lifecycle.PER_CLASS)
```

TestNG

- Name does not need to include 'test'
- Must specify the 'public' access modifier for the class. Not needed for the methods
- Test method returns are ignored (except for a setting from testng.xml file)

JUnit 5

```
@TestInstance(TestInstance.
Lifecycle.PER_CLASS)
class Junit1 {

    @BeforeAll
    void beforeAll() {
        ...    }

    @BeforeEach
    void beforeEach() {
        ...    }

    @Test
    void first() {
        ...    }
}
```

TestNG

```
public class NG1 {

    @BeforeClass
    void beforeClass() {
        ...    }

    @BeforeMethod
    void beforeMethod() {
        ...    }

    @Test
    void first() {
        ...    }
}
```

Test run order

JUnit 5

- By:
  - Method name alphanumerically
  - Display name alphanumerically
  - Specified order (based on int)
  - Random

TestNG

- By:
  - Depends on method
  - Depends on group
  - Priority

Disabling tests

JUnit 5

- @Disabled – test class or method
- @DisabledOnOs, @EnabledOnOs
- @DisabledIfSystemProperty, @EnabledIfSystemProperty
- @DisabledIfEnvironmentVariable, @EnabledIfEnvironmentVariable
- @DisabledIf, @EnabledIf

TestNG

- @Ignore – entire class, package or method
- attribute enabled=false - disable entity annotated with @Test (class, method)

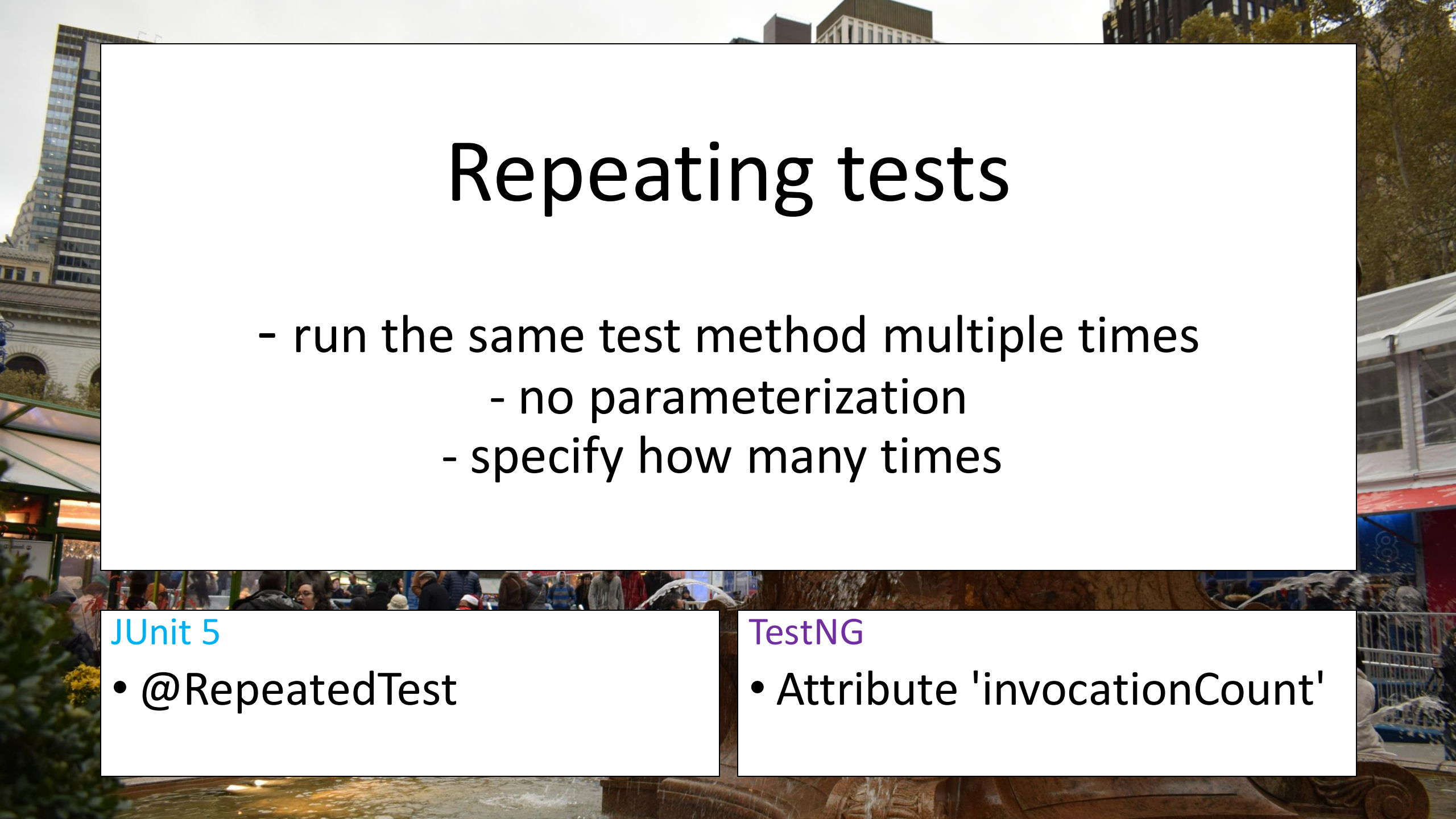
Parameterized tests

JUnit 5

- Pass inline constant values for 1 parameter (@ValueSource)
- Pass inline constant values for several parameters (@CsvSource)
- Read values from CSV file (@CsvFileSource)
- Values from Enum (@EnumSource)
- Receive values from method (@MethodSource)

TestNG

- Provide a single value for the param from the XML suite file
- Receive values from method



Repeating tests

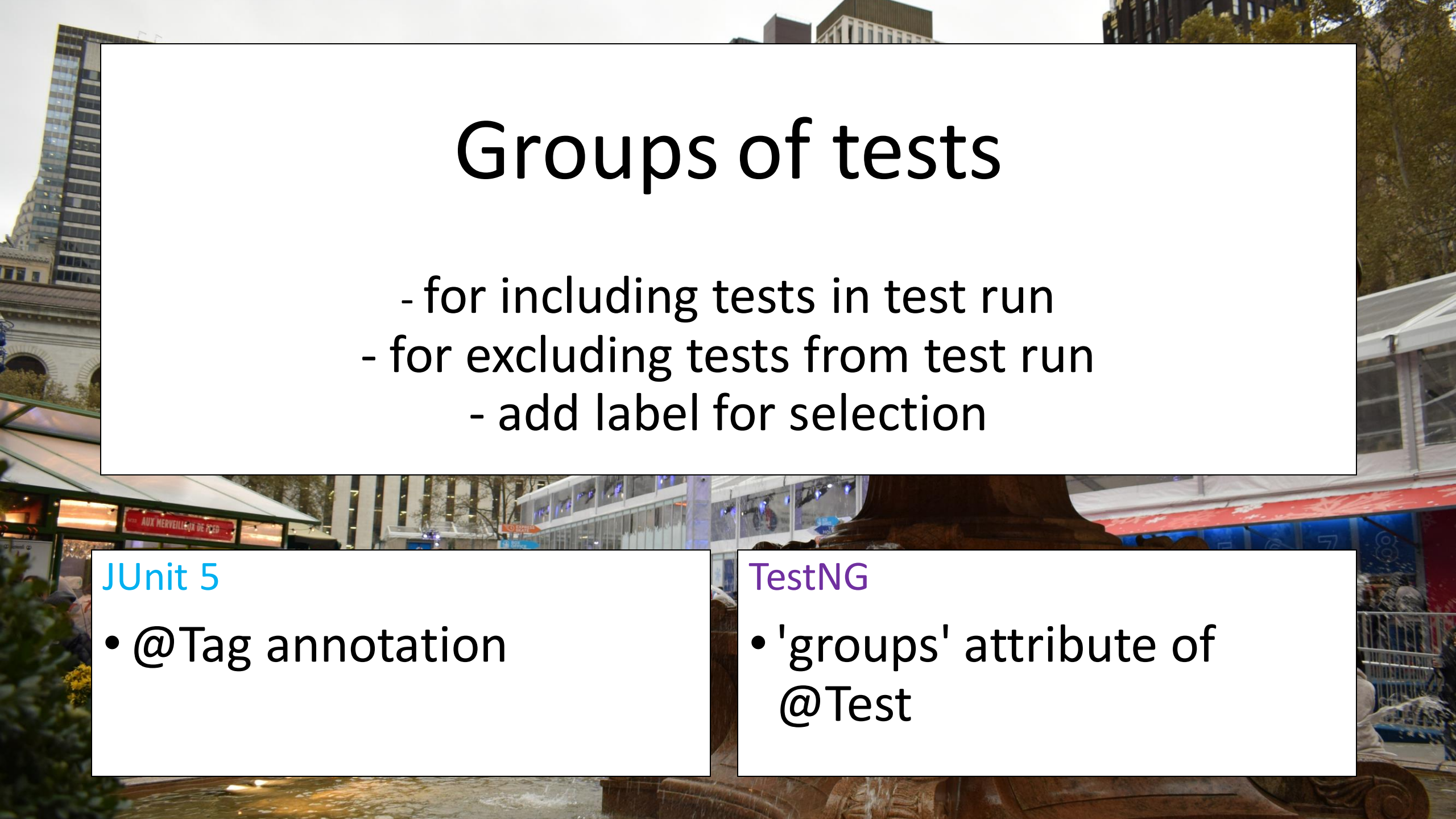
- run the same test method multiple times
- no parameterization
- specify how many times

JUnit 5

- @RepeatedTest

TestNG

- Attribute 'invocationCount'



Groups of tests

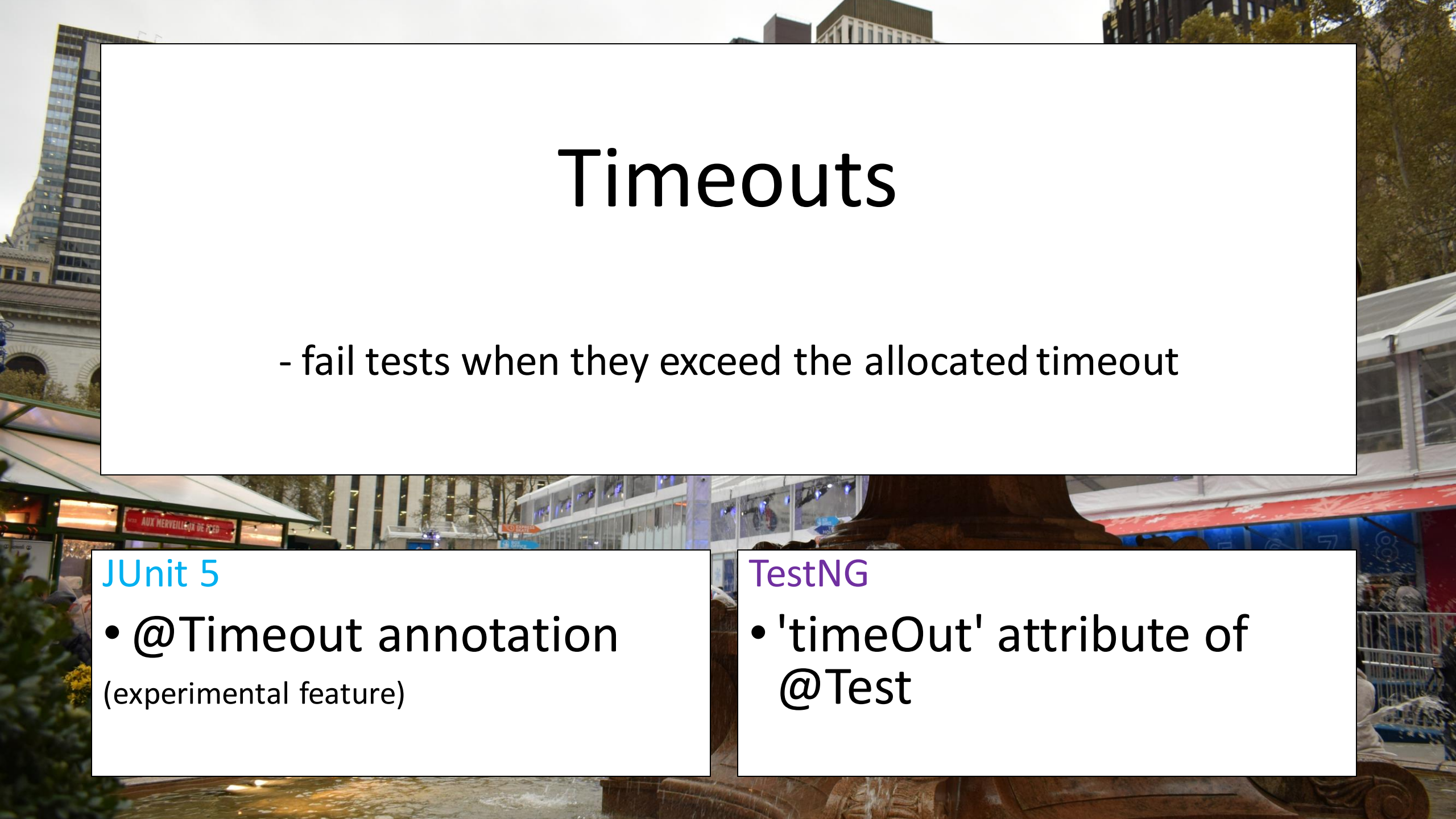
- for including tests in test run
- for excluding tests from test run
- add label for selection

JUnit 5

- @Tag annotation

TestNG

- 'groups' attribute of @Test



Timeouts

- fail tests when they exceed the allocated timeout

JUnit 5

- @Timeout annotation

(experimental feature)

TestNG

- 'timeOut' attribute of @Test

Assertions

JUnit 5

- JUnit specific (4 + 5) (Assertions class)
  - assertEquals, assertTrue, assertArraysEqual, assertTimeout, assertThrows, fail, assertAll
- Can be coupled with external libraries: Hamcrest, AssertJ, Truth

TestNG

- TestNG specific (Assert class):
  - assertEquals, assertTrue, assertEqualsNoOrder, assertThrows, fail
- SoftAssertion
- Can use JUnit assertions

Test suites

JUnit 5

- Can define suite class and run with JUnit platform runner
- Can include or exclude packages, classes, tags
- Currently buggy

TestNG

- XML suites
- Select packages, classes, groups
- Provide order
- Can declare programatically, less used

Listeners and reporting

JUnit 5

- TestWatcher interface: testAborted, testDisabled, testSuccessful, testFailed

TestNG

- ITestListener: onStart, onFinish, onTestFailure, onTestSkipped, onTestFailedButWithinSuccessPercentage, onTestSuccess

Running tests in parallel

JUnit 5

- Settings required + specify ExecutionMode (concurrent or same thread)

(experimental feature)

TestNG

- On the DataProvider level
- In the suite file, specify parallelism level: method, class, test

JUnit 5 specifics

- Assumptions: run tests or parts of tests only if boolean condition is met; abort otherwise
- Nested tests
- Create your own composed annotation

TestNG specifics

- RetryAnalyzer: retry the test method if a failure occurred

Thank you!

Corina Pip

@imalittletester

https://imalittletester.com/

https://github.com/iamalittletester