

ORACLE

How to Be a Java Automated Testing Superstar!

Billy Korando

Oracle - Java Developer Advocate ☕🥑
@BillyKorando



Important Notes

- Ask questions
- Reach out:
Twitter: @BillyKorando
Email: billy.korando@oracle.com
- Key link 🙋 <https://github.com/wkorando/java-testing-superstar>



- <https://dev.java>
- <https://inside.java>
- <https://youtube.com/java>
- Inside Java Podcast
- Inside Java Newscast
- #SipOfJava
- #JEP Café

Why Write Automated Tests?

- Substantial time and effort to write tests
- Substantial time and effort to maintain tests

My code is perfect!
Why should I waste time writing tests?

Why Write Automated Tests?

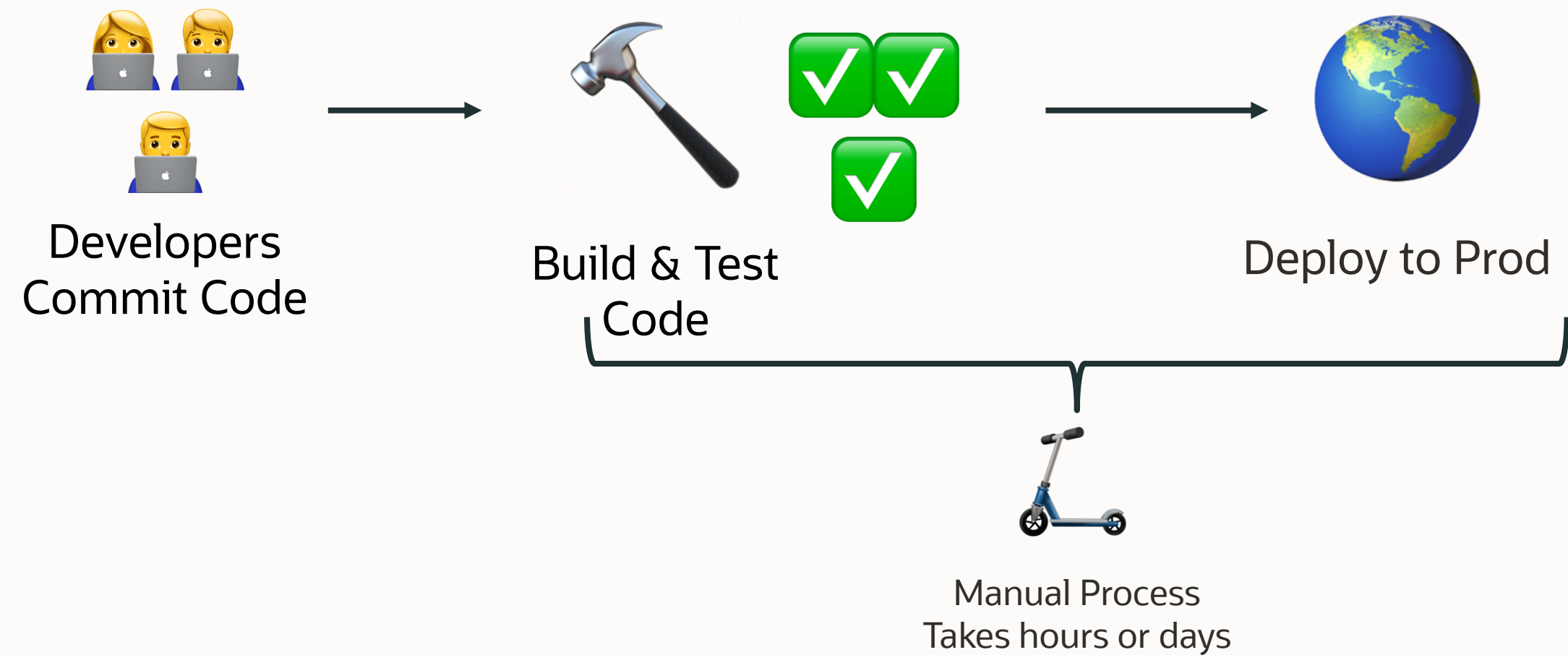
CI/CD Process =>



Automated Tests =>



Deployment Process

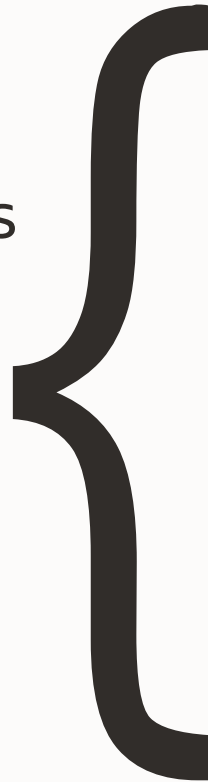


The Dependency Iceberg

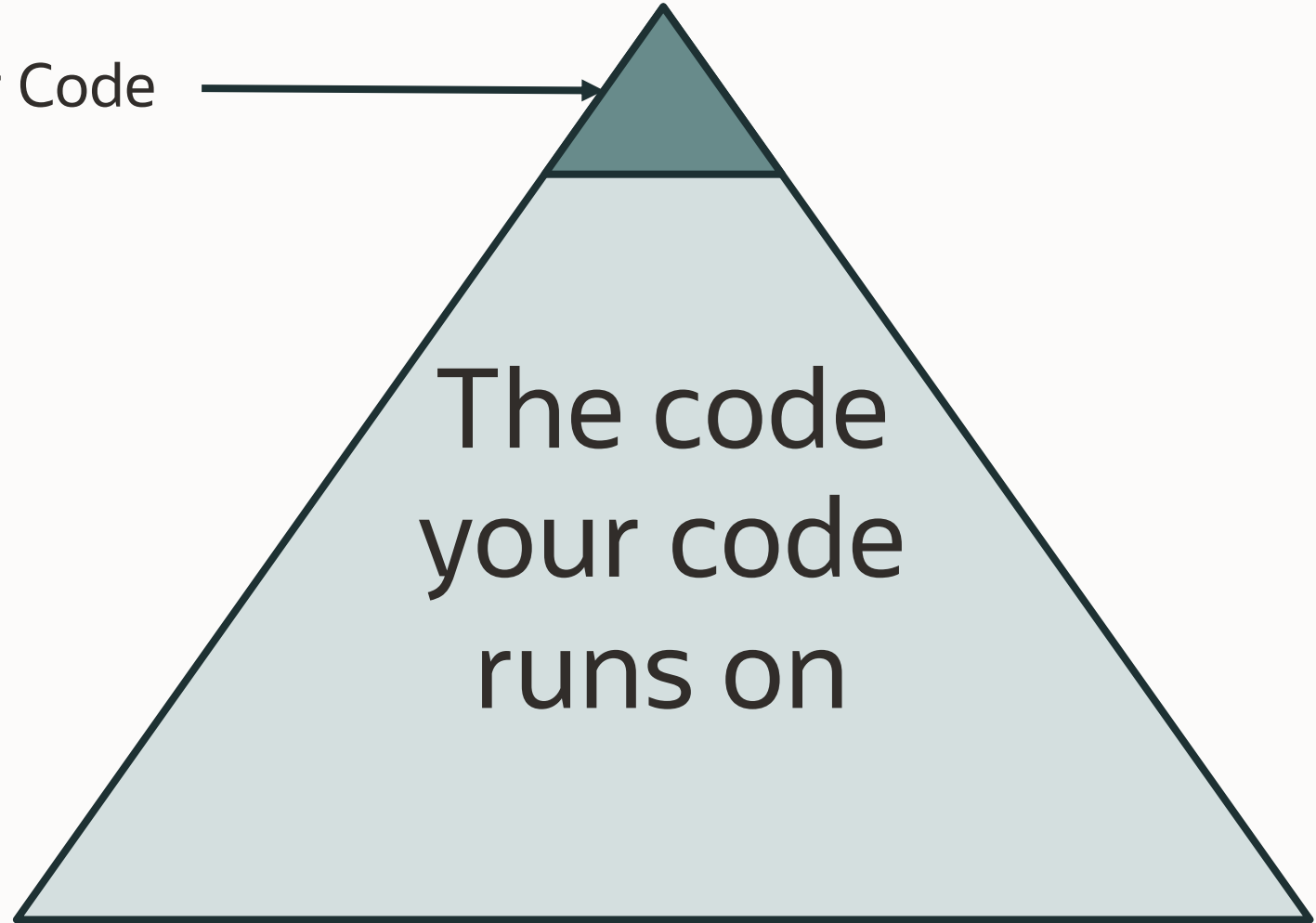
Your Code



- Security patches
- Performance enhancements
- Feature enhancements



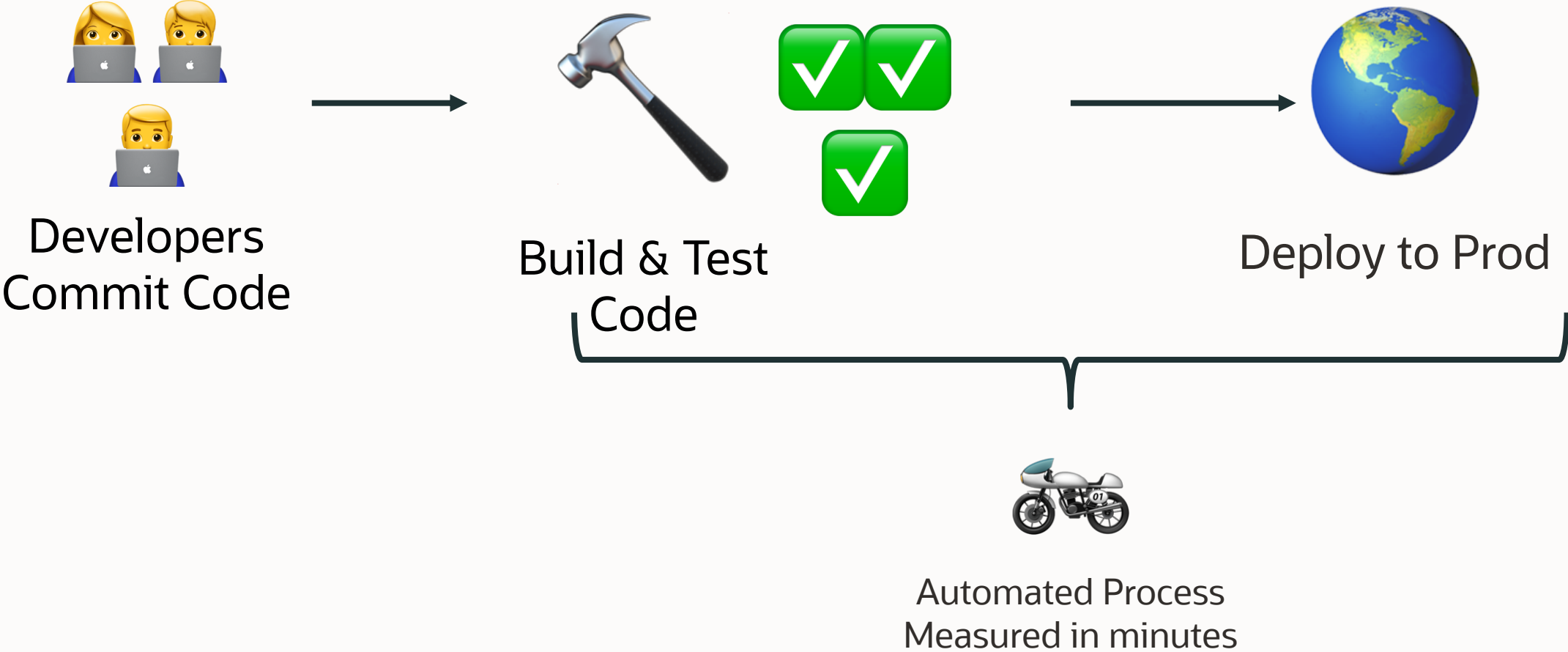
The code
your code
runs on



Manual Test Costs

The high effort of regression testing code disincentivizes keeping dependencies up to date

Deployment Process



Additional Benefits

Auditable

- Can review test code, reports, and logging

Repeatable

- Can re-run test suite to see if problems persists
- Can run test suite against other code base/version to see if problem exists

How Many Automated Test Should I Write?

Whatever gives you confidence to
deploy to PROD without manual
intervention

Manual Testing

Manual testing still needed.

But it's exploratory testing.

How to be an Automated Testing Superstar?

How to be an Automated Testing Superstar?

- ✓ Treat Test Code like “Production” Code
- ✓ Write Human Readable Tests

AssertJ – Fluent Assertions

```
@Test
public void testFindByValidRoomType() {
    RoomRepo repo = mock(RoomRepo.class);
    RoomServiceImpl service = new RoomServiceImpl(repo, roomTypes);
    when(repo.findRoomsByRoomType("Single"))
        .thenReturn(Arrays.asList(//
            new Room(1L, "100", "Single", new BigDecimal(145.99))));
    List<Room> rooms = service.findRoomsByType("Single");

    assertEquals(1, rooms.size());
}
```

AssertJ – Fluent Assertions

```
@Test
public void testFindByValidRoomType() {
    RoomRepo repo = mock(RoomRepo.class);
    RoomServiceImpl service = new RoomServiceImpl(repo, roomTypes);
    when(repo.findRoomsByRoomType("Single"))
        .thenReturn(Arrays.asList(//
            new Room(1L, "100", "Single", new BigDecimal(145.99))));
    List<Room> rooms = service.findRoomsByType("Single");

    assertThat(rooms.size()).isEqualTo(1);
}
```


Upgrade to JUnit 5

- Parallel test execution
- Repeated tests
- Extensions
- Selective disabling
- Default test methods
- Test ordering
- And more!



How to be an Automated Testing Superstar?

 Write Reliable Tests

Test Containers

```
public class ITCustomerJUnit5Repo {

    public static class Initializer implements ApplicationContextInitializer<ConfigurableApplicationContext> {
        @Override
        public void initialize(ConfigurableApplicationContext applicationContext) {
            TestPropertyValues.of("spring.datasource.url=" + postgres.getJdbcUrl(), //
                "spring.datasource.username=" + postgres.getUsername(), //
                "spring.datasource.password=" + postgres.getPassword(),
                "spring.jpa.database-platform=org.hibernate.dialect.PostgreSQL95Dialect") //
                .applyTo(applicationContext);
        }
    }

    static DockerImageName postgresHoteldbLatest = DockerImageName.parse("postgres-hoteldb:latest")
        .asCompatibleSubstituteFor("postgres");
    private static PostgreSQLContainer<?> postgres = new PostgreSQLContainer<>(postgresHoteldbLatest);

    @RegisterExtension
    static SpringTestContainersExtension extension = new SpringTestContainersExtension(postgres, true);

    ...
}
```

Test Containers

- Test against local containerized instances of remote services
- Don't have to worry about:
 - Service being down
 - Test data going missing
 - Maintain local instance

JFRUnit

```
@Test
@EnableEvent("jdk.ObjectAllocationInNewTLAB")
@EnableEvent("jdk.ObjectAllocationOutsideTLAB")
public void checkAllocationEvents() throws Exception {
    ...
}
```

JFRUnit

- Capture JDK Flight Recorder (JFR) Events to track system performance behavior
- Check JVM behavior consistent across systems and environments



Contract Driven Development (Spring Cloud Contract)

```
name: find-produce-by-name
description: Find find all produce that matches supplied name
request:
  method: GET
  url: /api/v1/produce/Apple
response:
  status: 200
  body:
    - id: 1
      name: "Apple"
      subName: "Granny Smith"
      quantity: 100
    - id: 2
      name: "Apple"
      subName: "Gala"
      quantity: 50
  headers:
    Content-Type: application/json;charset=UTF-8
```

Contract Driven Development (Spring Cloud Contract)

- Contracts validate service fulfill defined behavior
- Contracts can be used to setup up mock of service



Code Examples

JUnit 5, Mockito, AssertJ, and Test Containers:

<https://github.com/wkorando/welcome-to-junit5-v2>

Spring Cloud Contract Example:

<https://github.com/wkorando/collaborative-contract-driven-development-2-0>

JFRUnit:

<https://github.com/moditect/jfrunit>

Thank you

ORACLE