Introduction to Architectural Testing

Understanding Architectural Testing

Benefits of ArchUnit

1. Automated Architecture Enforcement
2. Easy Integration
3. Prevent Architectural Drift
4. Custom Rule Definition

Key Features

1. Layered Architecture Validation
2. Dependency Rules
3. Coding Standards

ArchUnit Maven Dependency

```
<dependencies>
    <dependency>
        <groupId>com.tngtech.archunit</groupId>
        <artifactId>archunit-junit5</artifactId>
        <version>1.2.0</version>
        <scope>test</scope>
    </dependency>
</dependencies>
```

Simple Rule to Enforce Package Dependencies

```
import com.tngtech.archunit.core.importer.ClassFileImporter;
import com.tngtech.archunit.lang.ArchRule;
import static com.tngtech.archunit.lang.syntax.ArchRuleDefinition.noClasses;

public class ArchitectureTest {

    @Test
    public void servicesShouldNotDependOnRepositories() {
        ArchRule rule = noClasses()
            .that().resideInAPackage("..service..")
            .should().dependOnClassesThat()
            .resideInAPackage("..repository..");

        rule.check(new ClassFileImporter().importPackages("com.yourcompany.yourproject"));
    }
}
```

Defining Layer Interactions in ArchUnit

```
LayeredArchitecture layeredArchitecture = layeredArchitecture()
    .layer("Controllers").definedBy("..controller..")
    .layer("Services").definedBy("..service..")
    .layer("Repositories").definedBy("..repository..")
    .whereLayer("Controllers").mayNotBeAccessedByAnyLayer()
    .whereLayer("Services").mayOnlyBeAccessedByLayers("Controllers")
    .whereLayer("Repositories").mayOnlyBeAccessedByLayers("Services");
```

Cyclic Dependency Check in ArchUnit

```
ArchRule noCycles = slices().matching("..myapp.(*)..").should().beFreeOfCycles();
```

Best Practices in Architectural Testing with ArchUnit

1. Start Small and Simple
2. Gradually Increase Complexity
3. Document Your Rules Clearly
4. Integrate with Continuous Integration
5. Collaborate and Review
6. Avoid Over-specification