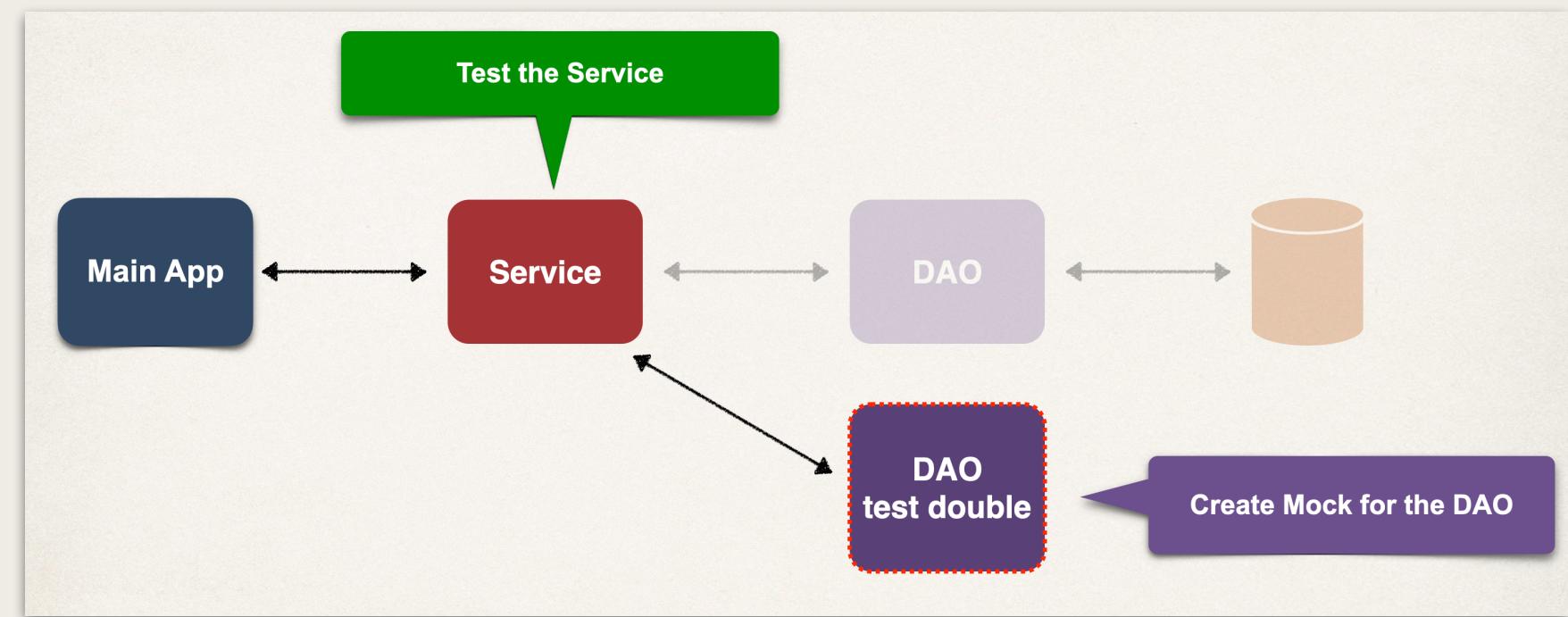


Development Process

Step-By-Step

1. Create Mock for DAO
2. Inject mock into Service
3. Set up expectations
4. Call method under test and assert results
5. Verify method calls

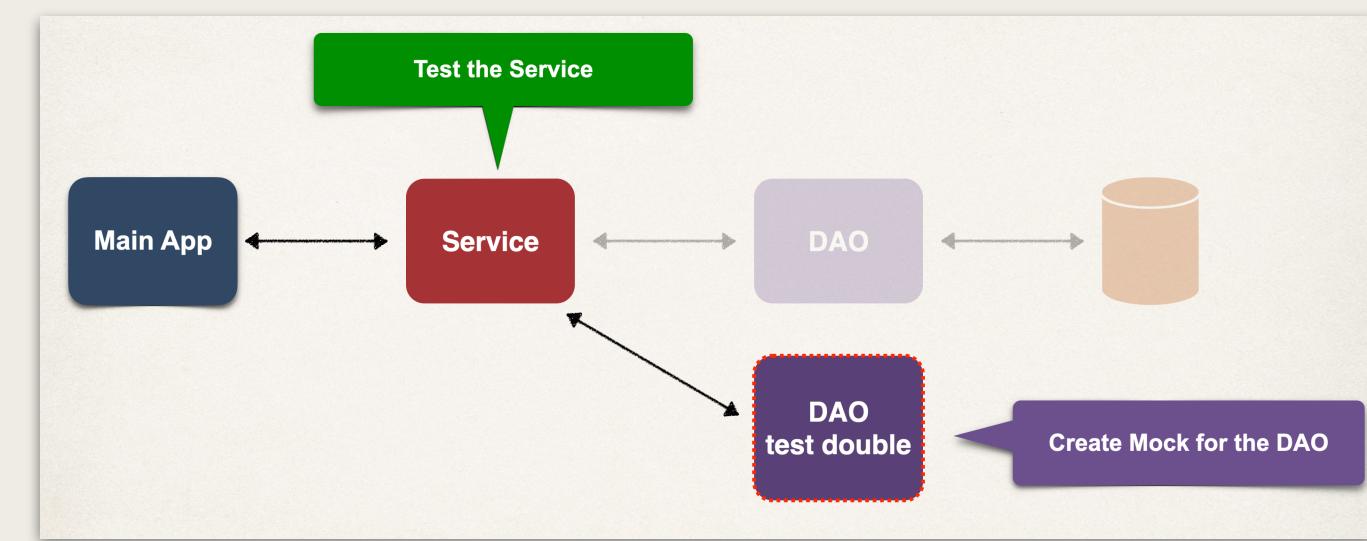


Step 1: Create Mock for the DAO

MockAnnotationTest.java

```
import org.mockito.Mock;  
...  
  
@SpringBootTest(classes=MvcTestingExampleApplication.class)  
public class MockAnnotationTest {  
  
    @Mock  
    private ApplicationDao applicationDao;  
  
    ...  
  
}
```

Create Mock for the DAO



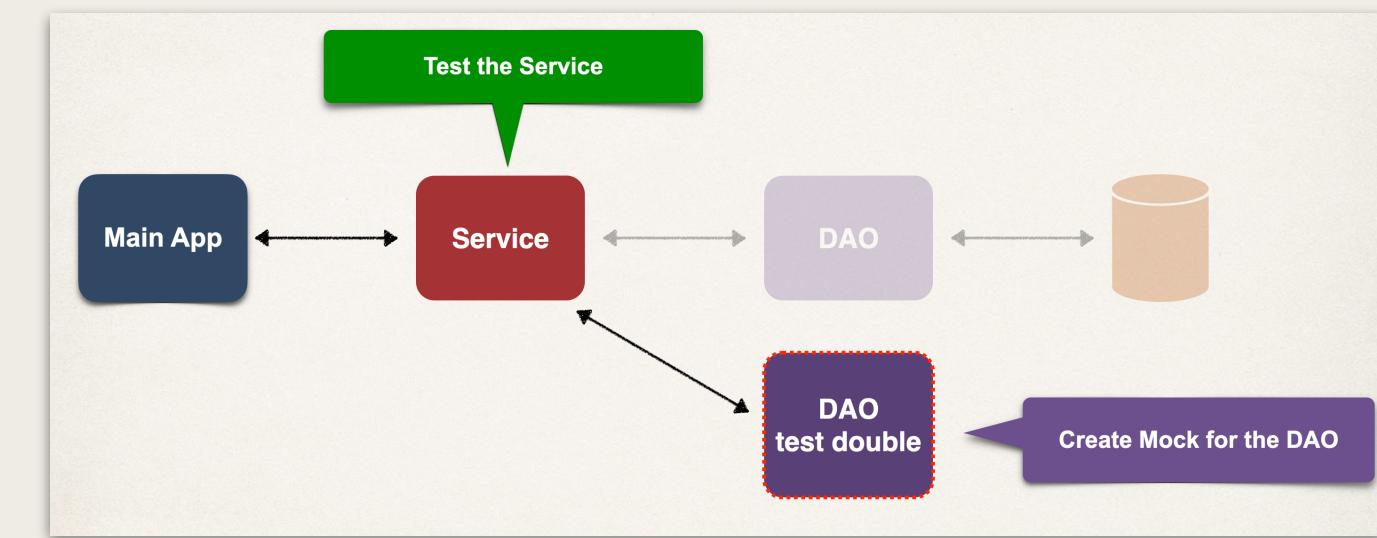
Step 2: Inject mock into Service

MockAnnotationTest.java

```
import org.mockito.Mock;
import org.mockito.InjectMocks;
...
@SpringBootTest(classes=MvcTestingExampleApplication.class)
public class MockAnnotationTest {

    @Mock
    private ApplicationDao applicationDao;

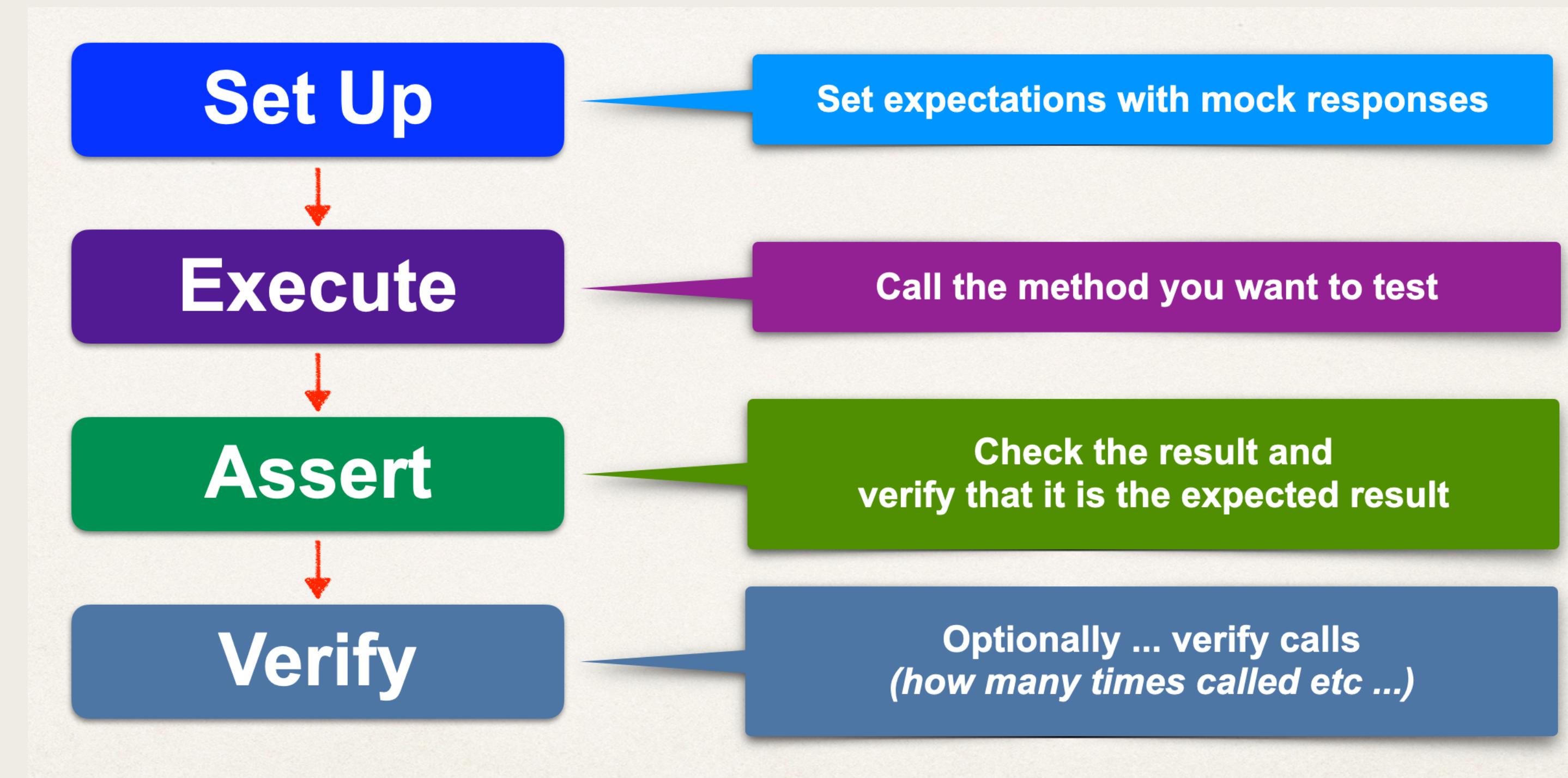
    @InjectMocks
    private ApplicationService applicationService;
    ...
}
```



Inject mock dependencies

Note: Will only inject dependencies annotated with `@Mock` or `@Spy`

Step 3: Set up expectations



Step 3: Set up expectations

*when method
doSomeWork(...)
is called
then return "I am finished"*

```
import static org.mockito.Mockito.when;  
...  
  
String aResponse = "I am finished";  
  
when( doSomeWork() ).thenReturn( aResponse );
```

response

method

Real world analogy:
Theater - just read the script

Step 3: Set up expectations

MockAnnotationTest.java

```
import static org.mockito.Mockito.when;
import static org.junit.jupiter.api.Assertions.assertEquals;
...

@SpringBootTest(classes=MvcTestingExampleApplication.class)
public class MockAnnotationTest {

    @Mock
    private ApplicationDao applicationDao;

    @InjectMocks
    private ApplicationService applicationService;

    @Autowired
    private CollegeStudent studentOne;

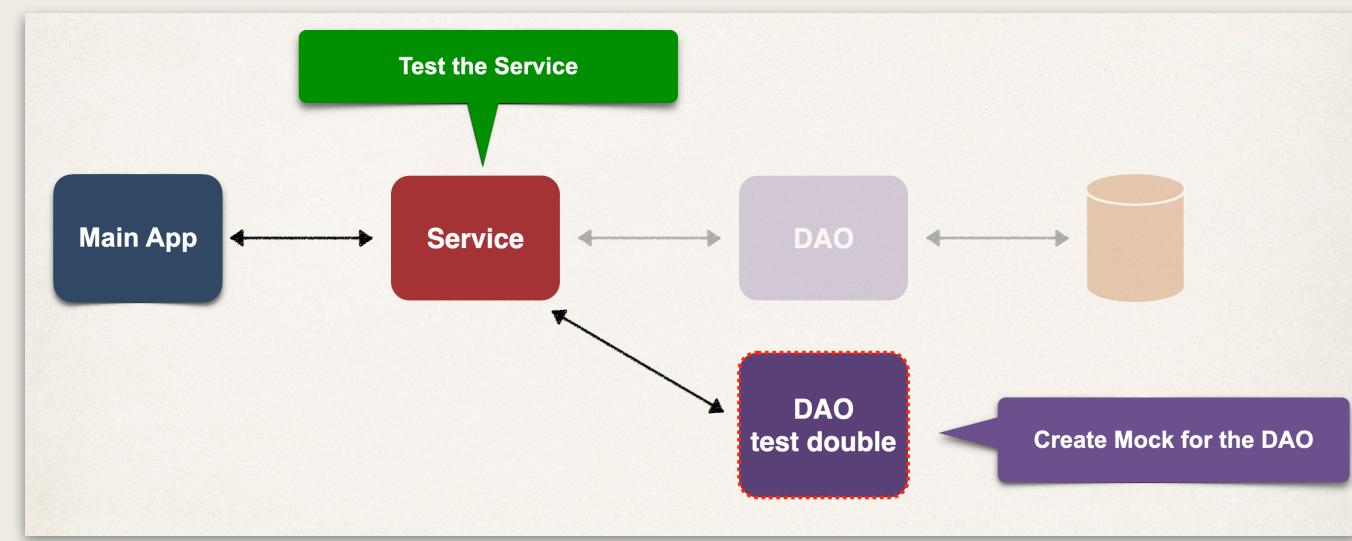
    @Autowired
    private StudentGrades studentGrades;

    @DisplayName("When & Verify")
    @Test
    public void assertEqualsTestAddGrades() {
        when(applicationDao.addGradeResultsForSingleClass(
            studentGrades.getMathGradeResults())).thenReturn(100.0);

        ...
    }
}
```

Set up expectations
for mock response

when method
addGradeResultsForSingleClass(...)
is called
then return 100.0



Step 4: Call method under test and assert results

MockAnnotationTest.java

```
import static org.mockito.Mockito.when;
import static org.junit.jupiter.api.Assertions.assertEquals;
...
@SpringBootTest(classes= MvcTestingExampleApplication.class)
public class MockAnnotationTest {

    @Mock
    private ApplicationDao applicationDao;

    @InjectMocks
    private ApplicationService applicationService;

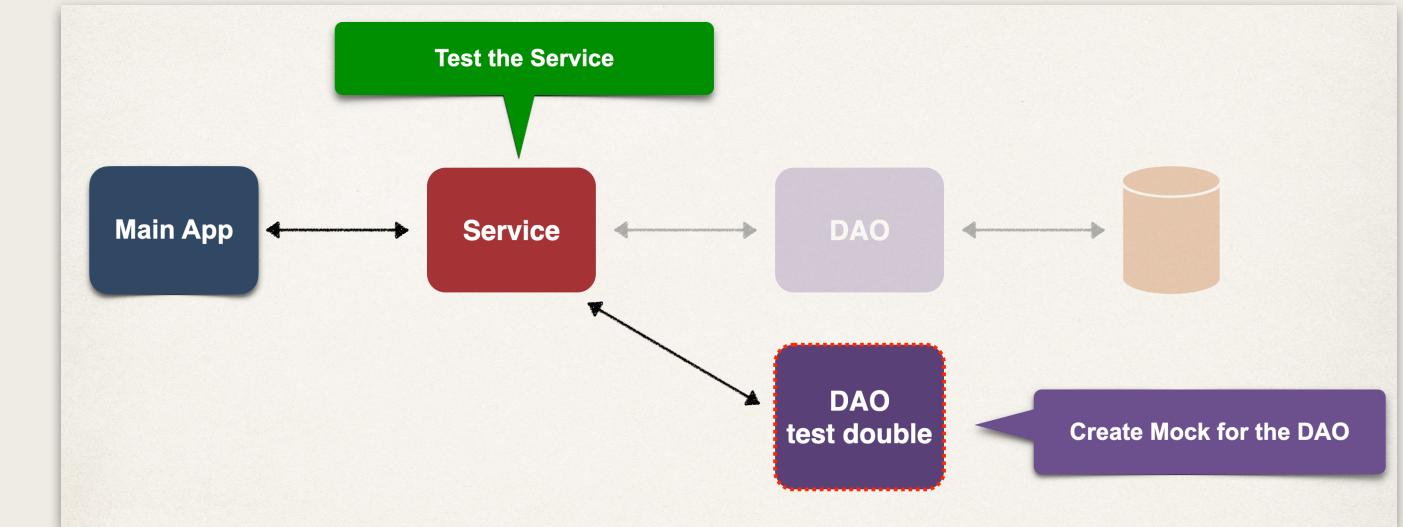
    @Autowired
    private CollegeStudent studentOne;

    @Autowired
    private StudentGrades studentGrades;

    @DisplayName("When & Verify")
    @Test
    public void assertEqualsTestAddGrades() {
        when(applicationDao.addGradeResultsForSingleClass(
                studentGrades.getMathGradeResults())).thenReturn(100.0);

        assertEquals(100.0, applicationService.addGradeResultsForSingleClass(
                studentOne.getStudentGrades().getMathGradeResults()));

    }
}
```



Assert results

*The service uses the DAO ...
that has been set up to return 100.0*

Step 5: Verify method calls

MockAnnotationTest.java

```
import static org.mockito.Mockito.when;
import static org.mockito.Mockito.verify;
import static org.mockito.Mockito.times;
...

@SpringBootTest(classes= MvcTestingExampleApplication.class)
public class MockAnnotationTest {

    @Mock
    private ApplicationDao applicationDao;

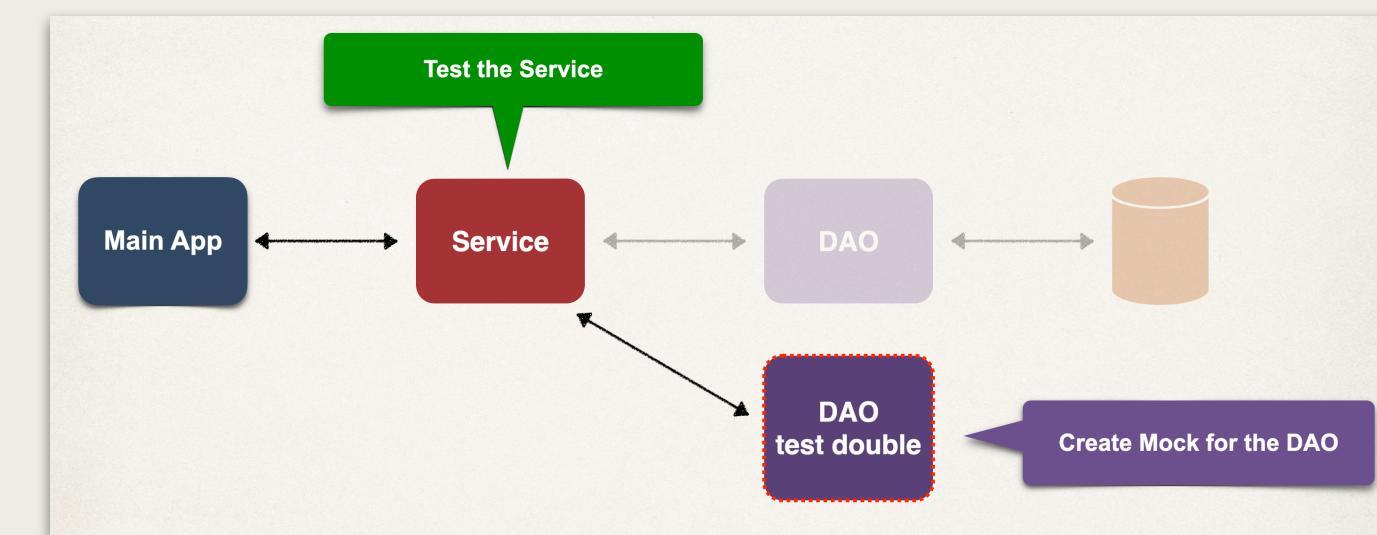
    @InjectMocks
    private ApplicationService applicationService;

    ...

    @DisplayName("When & Verify")
    @Test
    public void assertEqualsTestAddGrades() {
        when(applicationDao.addGradeResultsForSingleClass(
            studentGrades.getMathGradeResults())).thenReturn(100.0);

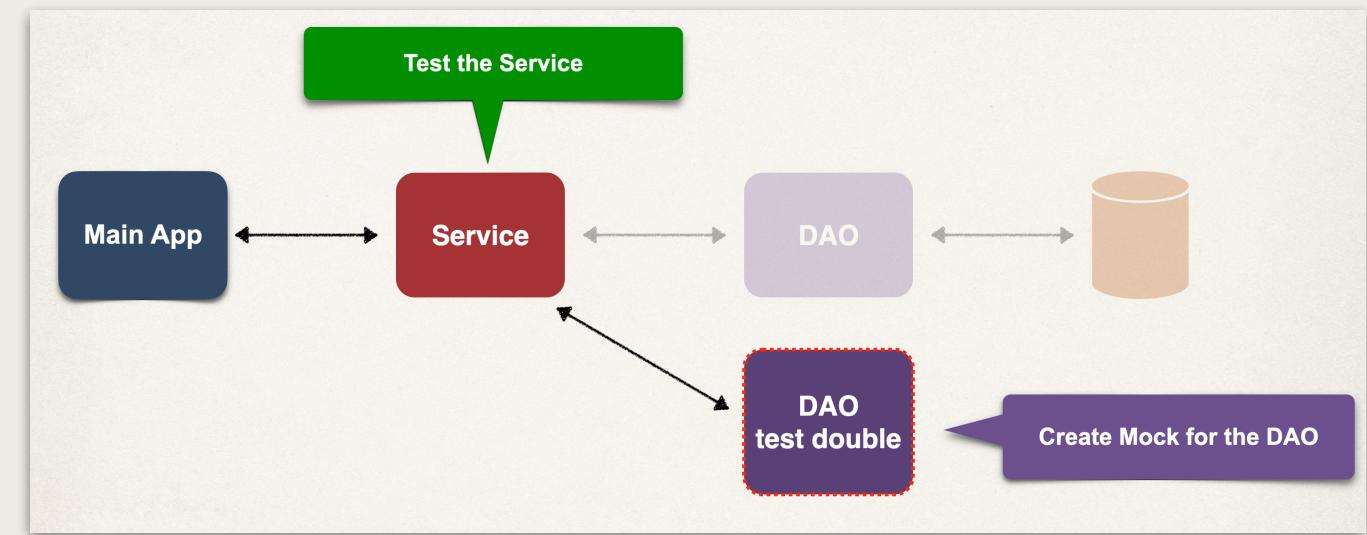
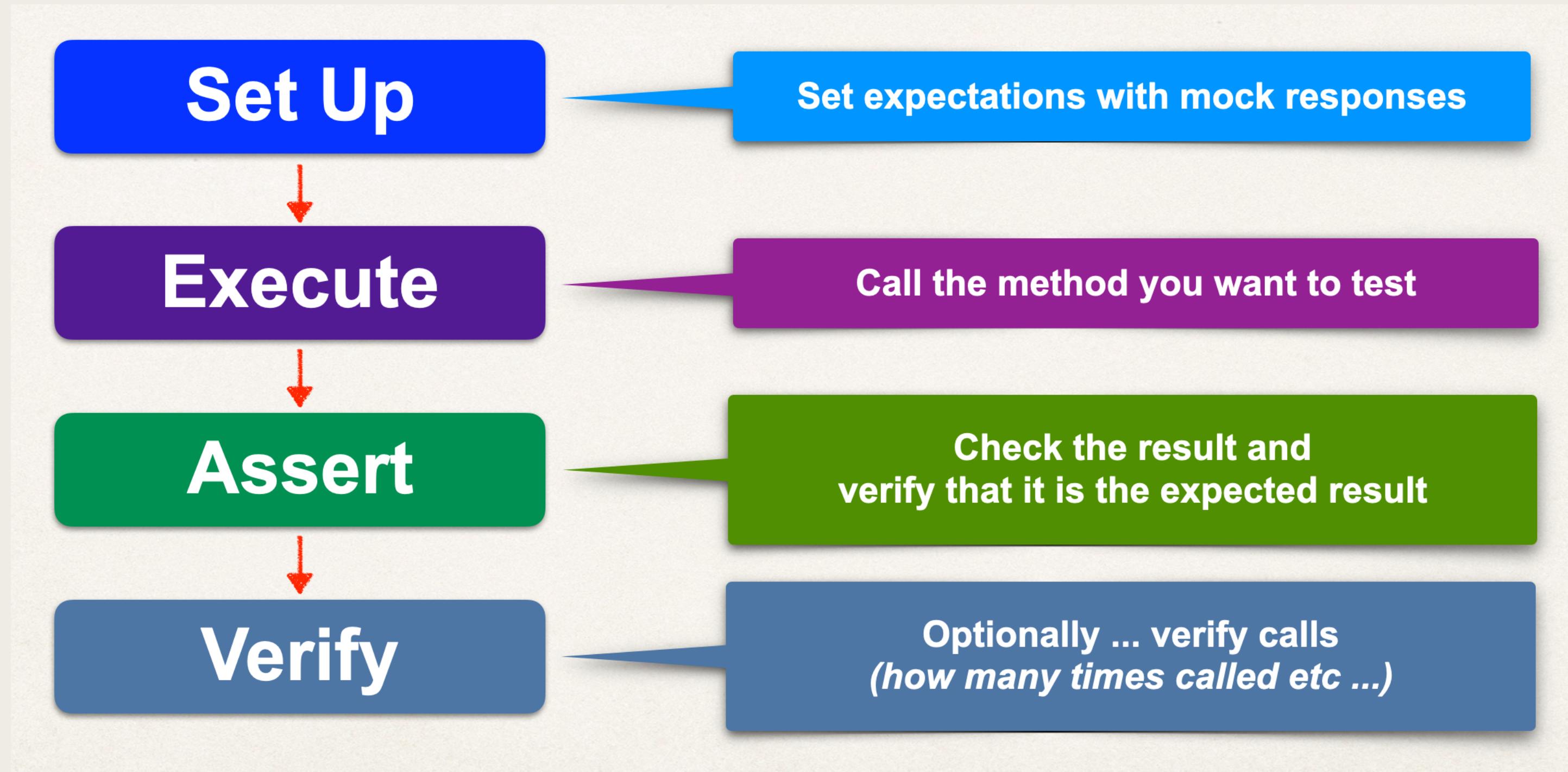
        assertEquals(100.0, applicationService.addGradeResultsForSingleClass(
            studentOne.getStudentGrades().getMathGradeResults()));

        verify(applicationDao,
            times(1)).addGradeResultsForSingleClass(studentGrades.getMathGradeResults());
    }
}
```



Verify the DAO method
was called 1 time

Recap



Mockito Resources

- Additional features
 - Stubs, spies
 - Argument matchers, Answers
- ...

site.mockito.org