PowerMock

PowerMock is an open-source Java framework used for creating a mock object in unit testing. It extends other mocking frameworks such as EasyMock and Mockito to enhance the capabilities. The PowerMock framework uses a custom classloader and bytecode manipulation techniques to enable the mocking of static methods, final classes, final methods, private methods, constructor, and removal of static initializers. The main aim of PowerMock is to extend the existing APIs with some methods and annotations to provide extra features that make unit testing quite easy.

The PowerMock framework provides a class called PowerMockito used to create mock objects and initiates verification and expectation. The PowerMockito provides the functionality to work with the Java reflection API.

PowerMock dependencies (Download updated dependencies by clicking on below points).

- Powermock-api-mockito
- Powermock-module-junit4

PowerMock annotations

1. **@RunWith(PowerMockRunner.class):** @RunWith annotation is similar to what we did in Mockito. Instead of using Mockito, we will use a PowerMockRunner this time, which will enable PowerMockito APIs in the test.

2. **@PrepareForTest:** It tells PowerMock to prepare some classes for testing. It can be applied to both the test classes and the individual test methods. It includes classes with final, static, private, or native methods that can be mocked.

   @PrepareForTest was provided a fullyQualifiedNames package with a wildcard. This informs PowerMockito which classes to prepare with Java Reflection API for testing.

Mocking final methods

```
package org.ram.examples;
public class ClassWithFinalMethods {
   public final String printMessage(String message) {
       return message;
   }
}
```

```
package org.ram.examples;
import org.junit.Assert;
import org.junit.Test;
import org.junit.runner.RunWith;
import org.powermock.api.mockito.PowerMockito;
import org.powermock.core.classloader.annotations.PrepareForTest;
import org.powermock.modules.junit4.PowerMockRunner;

@RunWith(PowerMockRunner.class)
@PrepareForTest(fullyQualifiedNames = "org.ram.examples.*")
public class PowermockFinalMethodTest {
   @Test
   public void testPrintMessage() throws Exception {
       String message="Hello world";

                                       ClassWithFinalMethods     mockObject=
PowerMockito.mock(ClassWithFinalMethods.class);


PowerMockito.when(mockObject.printMessage(message)).thenReturn(message);

       ClassWithFinalMethods object= new ClassWithFinalMethods();

       String actualMessage=object.printMessage(message);
       Assert.assertEquals(message,actualMessage);

   }
}
```

Mocking static methods

```
package org.ram.examples;

public class ClassWithStaticMethod {
   public static String showMessage(String message){
       return message;
   }
}
```

```
package org.ram.examples;
import org.junit.Assert;
import org.junit.Test;
import org.junit.runner.RunWith;
import org.powermock.api.mockito.PowerMockito;
```

```
import org.powermock.core.classloader.annotations.PrepareForTest;
import org.powermock.modules.junit4.PowerMockRunner;

@RunWith(PowerMockRunner.class)
@PrepareForTest(fullyQualifiedNames = "org.ram.examples.*")
public class PowermockStaticMethodTest {
   @Test
   public void testShowMessageStaticMethod(){
       String message="Hello world";
       PowerMockito.mockStatic(ClassWithStaticMethod.class);

PowerMockito.when(ClassWithStaticMethod.showMessage(message)).thenRetu
rn(message);
                                                                    String
actualMessage=ClassWithStaticMethod.showMessage(message);
       Assert.assertEquals(message,actualMessage);
   }
}
```

Mocking private methods

```
package org.ram.examples;
public class ClassWithPrivateMethods {
   private String privateShowMessage(String message){
       return message;
   }
   public String callPrivateShowMessage(String message){
       return privateShowMessage(message);
   }
}
```

```
package org.ram.examples;
import org.junit.Assert;
import org.junit.Test;
import org.junit.runner.RunWith;
import org.powermock.api.mockito.PowerMockito;
import org.powermock.core.classloader.annotations.PrepareForTest;
import org.powermock.modules.junit4.PowerMockRunner;
@RunWith(PowerMockRunner.class)
@PrepareForTest(fullyQualifiedNames = "org.ram.examples.*")
public class PowermockPrivateMethodTest{
```

```
    @Test
    public void testPrivateMethod() throws Exception {
        String actualMessage= "this is actual message";
        String expectedMessage="this is expected message";

                ClassWithPrivateMethods   mock=   PowerMockito.spy(new
ClassWithPrivateMethods());

PowerMockito.doReturn(expectedMessage).when(mock,"privateShowMessag
e",actualMessage);
        String actual=mock.callPrivateShowMessage(actualMessage);
        Assert.assertEquals(expectedMessage,actual);

    }
}
```