



Test the untestable

Johan Haleby and Jan Kronquist

www.powermock.org

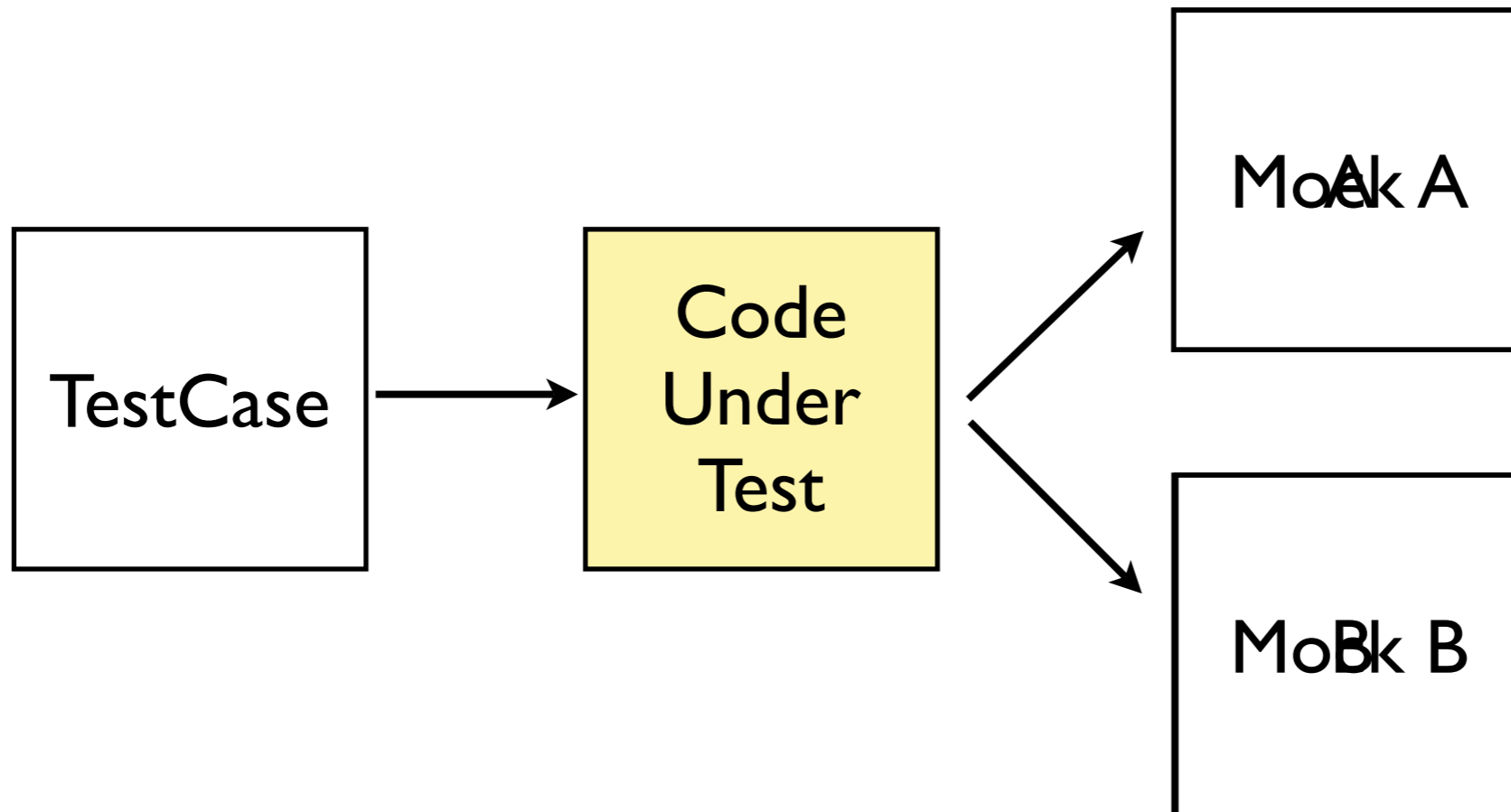
jayway

Audience survey

- :: Writing unit tests?
- :: Using mock framework?
- :: Using EasyMock?
- :: Frustrated with current limitations?

jayway

Automated + isolated testing



jayway

Mock objects and stubs

- :: Simulated objects that mimic behavior of real objects
- :: Stubs provide canned replies
- :: Mocks verify expectations

jayway

EasyMock

```
// Create a mock
SomeInterface mock = createMock(SomeInterface.class);

// Record behavior
expect(mock.doStuff("argument")).andReturn("returnValue");

// Replay behavior
replay(mock);

// Executing the code we want to test
assertEquals("returnValue", test.perform());

// Verify behavior
verify(mock);
```



Traditional assumptions

- :: Testable = good design
- :: Testable = easy to understand
- :: Testable = limitations

jayway

Problem – Static methods

- :: Not possible to mock
- :: Still 3rd party libraries that use them
 - :: Eclipse, Java ME

jayway

Problem – Static initializers

:: You cannot unit test a class with evil static initializer

```
public class EvilStaticInitializer {  
    static {  
        System.loadLibrary("evil.dll");  
    }  
}
```

jayway

Problem – Final class / method

- :: Final methods cannot be mocked
- :: Why use final?
 - :: Ensure that your classes are not misused
 - :: Correct object construction
- :: Examples: Wicket, Apache Mina

Jayway

Problem – Private methods/state

- :: Invocation is hard
- :: Mocking not possible
- :: State access not possible
- :: Private is used to enforce encapsulation

jayway

Problem – Using new

- :: Code not designed for testing
- :: Not everything needs to be injected

```
File file = new File(fileName);  
if (file.exists()) {  
    doStuff();  
}
```

Jayway

Traditional solutions

- :: Better design and IoC
- :: Wrap static methods
- :: Factory methods
- :: Package private instead of private
- :: Avoid final and static initializers

jayway

Static method – example

```
public class MessageDao {
    public void addMessage(Message message) throws RecordStoreException {
        RecordStore recordStore = null;
        try {
            recordStore = RecordStore.openRecordStore(name, true);

            ...

        } finally {
            if (recordStore != null) {
                recordStore.closeRecordStore();
            }
        }
    }
}
```

jayway

Solution 1 – Partial mock

```
public class MessageDao {  
    public void addMessage(Message message) throws RecordStoreException {  
        RecordStore recordStore = null;  
        try {  
            recordStore = openOrCreateRecordStore(name);  
  
            ...  
  
        } finally {  
            if (recordStore != null) {  
                recordStore.closeRecordStore();  
            }  
        }  
    }  
}  
  
protected RecordStore openOrCreateRecordStore(String name) {  
    return RecordStore.openRecordStore(name, true);  
}  
}
```

Jayway

Solution 2 – DI

```
public class MessageDao {  
    private RecordStoreFactory recordStoreFactory;  
  
    public MessageDao(RecordStoreFactory recordStoreFactory) {  
        this.recordStoreFactory = recordStoreFactory;  
    }  
  
    public void addMessage(Message message) throws RecordStoreException {  
        RecordStore recordStore = null;  
        try {  
            recordStore = recordStoreFactory.openOrCreateRecordStore(name);  
  
            ...  
        } finally {  
            if (recordStore != null) {  
                recordStore.closeRecordStore();  
            }  
        }  
    }  
}
```

The logo for Jayway, featuring the word "jayway" in a stylized, lowercase, red serif font.

Static method – example

```
public class MessageDao {  
    public void addMessage(Message message) throws RecordStoreException {  
        RecordStore recordStore = null;  
        try {  
            recordStore = RecordStore.openRecordStore(name, true);  
  
            ...  
  
        } finally {  
            if (recordStore != null) {  
                recordStore.closeRecordStore();  
            }  
        }  
    }  
}
```

jayway

Using PowerMock

```
public class MessageDao {
    public void addMessage(Message message) throws RecordStoreException {
        RecordStore recordStore = null;
        try {
            recordStore = RecordStore.openRecordStore(name, true);

            ...

        } finally {
            if (recordStore != null) {
                recordStore.closeRecordStore();
            }
        }
    }
}
```



PowerMock test case

```
@RunWith(PowerMockRunner.class)
@PrepareForTest(RecordStore.class)
public class MessageDaoTest {
    public void testAddMessage() throws Exception {
        mockStatic(RecordStore.class);

        // Record behavior
        expect(RecordStore.openRecordStore("name", true))
            .andReturn(recordStoreMock);

        // Replay behavior
        replay(RecordStore.class);

        // Executing the code we want to test
        Message message = new Message();
        messageDao.addMessage(message);
        ...

        // Verify behavior
        verify(RecordStore.class);
    }
}
```

The logo for Jayway, featuring the word "jayway" in a stylized, lowercase, red serif font.

PowerMock

- :: Drop in jar file
- :: Allow full semantics of the language
- :: Clean separation of test and production code
- :: Extends EasyMock
- :: Open source (Apache License 2.0)

jayway

DEMO

- :: Mock static method
- :: Mock construction
- :: Suppress static initializers

jayway

Traditional assumptions *are wrong*

- :: Testable \neq good design
- :: Testable \neq easy to understand
- :: Testable \neq limitations

The choice is yours!

jayway

Future

- :: Anonymous inner classes
- :: Improved mock creation
- :: Improved type safety
- :: Other test frameworks
- :: Other mock frameworks

jayway

Benefits

- :: Testable legacy code
- :: Third party code
- :: More design options

jayway

www.powermock.org

jayway