

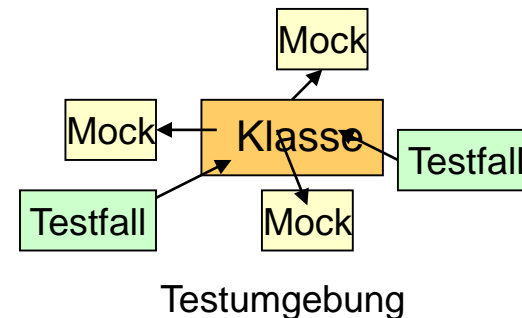
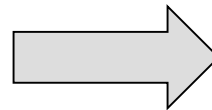
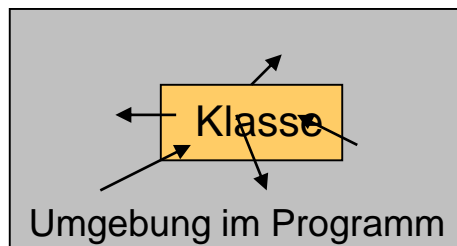
Verteilte Verarbeitung

Mocking



Umgebung: Stubs und Mocks

- Klassen / Module benötigen für die Modultests andere Klassen, Nachbarsysteme, ...
 - Häufig nicht verfügbar, nicht fertig
 - Verhalten häufig schwer kontrollierbar
 - Ausnahmesituationen schwer herstellbar
- Idee: benötigte Klassen / Nachbarsysteme etc. durch Dummies / Stubs ersetzen
 - Interface für diese Klasse/Nachbarsystem einziehen
 - Dummy implementiert das Interface wenn nicht möglich: Dummy ist von Klasse abgeleitet
 - Alternative: Mock-Objekt-Frameworks: z.B. Generische Mock-Objekte (Capture/Replay): <http://site.mockito.org/>



Testumgebung

Umgebung über Mock-Objekte simulieren

- Mock / Dummy-Objekte
= **Platzhalter** für „echte“ Objekte mit derselben Schnittstelle
- Wenn Umgebung der zu testende Klasse ...
 - zu komplex, um einfachen Testfall zu erstellen
 - für Test technisch / fachlich nicht oder nur aufwändig herstellbar
 - unerwünschte Seiteneffekte hat
 - zu langsam ist
(da am Netzwerk / an der Datenbank / am Nachbarsystem)
 - Sonderfälle (insbes. Fehlerzustände) nicht erstellen kann

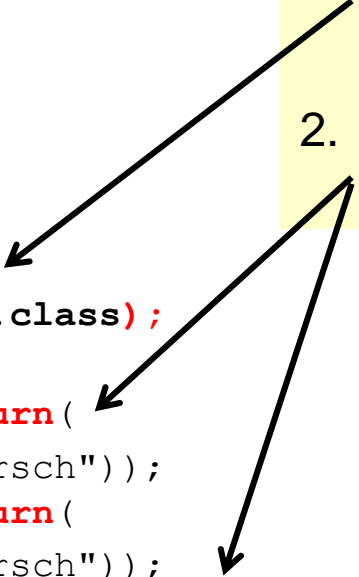
Umgebung über Mock-Objekte simulieren: Beispiel Mockito

```
import static org.mockito.Mockito.*;
// ...

public class AnwendungsfallTest {
    private IPersonenVerwaltung verwaltung;
    private Anwendungsfall afall;

    @Before
    public void setUp() {
        verwaltung = mock(IPersonenVerwaltung.class);

        when(verwaltung.findById(2L)).thenReturn(
            new Person(2L, "Bertram", "Barsch"));
        when(verwaltung.findById(3L)).thenReturn(
            new Person(3L, "Bettina", "Barsch"));
        when(verwaltung.findByName("Barsch")).thenReturn(
            Arrays.asList( ... )
    }
}
```

1. Mock Objekt aus Interface / Klasse erzeugen
 2. Mock Verhalten definieren
- 

Umgebung über Mock-Objekte simulieren: Beispiel Mockito

```
private IPersonenVerwaltung verwaltung;  
private Anwendungsfall afall;  
  
@Test  
public void findPerson() {  
    Person bertram = afall.findPerson(2L);  
    assertThat(bertram.getVorname(),  
                equalTo("Bertram"));  
    verify(verwaltung, times(1)).findById(2L);  
}
```

1. Mock Objekt verwenden (eigentlich indirekt)
2. Prüfen, ob Mock-Objekt vom getesteten Objekt wie erwartet verwendet wurde