Unit Testing with Mock Objects

http://d3s.mff.cuni.cz

Pavel Parízek
parizek@d3s.mff.cuni.cz
class CustomerList {
    private DbConnection db;
    private Map<Integer, Customer> customers;

    public CustomerList(DbConnection db) {
        this.db = db;
        customers = new HashMap<Integer, Customer>();
    }

    public void add(Customer cr) {
        customers.put(cr.getID(), cr);
        db.update("INSERT INTO ..." + createSQL(cr));
    }

    public Customer get(int id) {
        if (customers.containsKey(id)) return customers.get(id);
        ResultSet rs = db.query("SELECT ... FROM ...");
        Customer cr = createObject(rs.get(0));
        return cr;
    }
}
Possible solutions

- Dummy object
  - Passed around (parameters) but never used

- Stub implementation
  - Predefined responses to some method calls

- Mock object
  - Stub that checks proper usage by object under test
  - Process: define expectations and later verify them
  - Concept represents limited “behavior verification”
Frameworks

- C#/.NET
  - Moq
    - https://github.com/moq/moq4
  - RhinoMocks
    - https://www.hibernate rhinos.com/oss/rhino-mocks

- Java
  - Mockito
    - http://site.mockito.org/
  - EasyMock
    - http://easymock.org/