Assignment 1: Bank

- **Interface Account**

```
public interface Account {

   String getNumber();  // returns the account number
   String getOwner();   // returns the account owner
   boolean isActive();  // returns the account state

   void deposit (double amount)
      throws IllegalArgumentException, InactiveException;

   void withdraw (double amount)
      throws IllegalArgumentException, OverdrawException,
             InactiveException;

   double getBalance();
}
```

And all methods declare to throw an IOException

Assignment 1: Bank

- **Interface Bank**

```
public interface Bank {
   // creates a new account and returns the account number
   String createAccount(String owner);

   // closes the specified account and returns success of
   // closing and passivates the account
   boolean closeAccount(String number);

   // returns a set which contains the account numbers of
   // all currently active accounts
   Set<String> getAccountNumbers();

   // returns a particular account or null, if number is not valid
   Account getAccount(String number);

   void transfer(Account a, Account b, double amount)
      throws IllegalArgumentException, OverdrawException,
             InactiveException;
}
```

Assignment 1: Bank

- **Interface BankDriver**

```
package bank;
public interface BankDriver {
   public void connect(String[] args);
   public void disconnect();
   public Bank getBank();
}
```

- **Start of the Application**

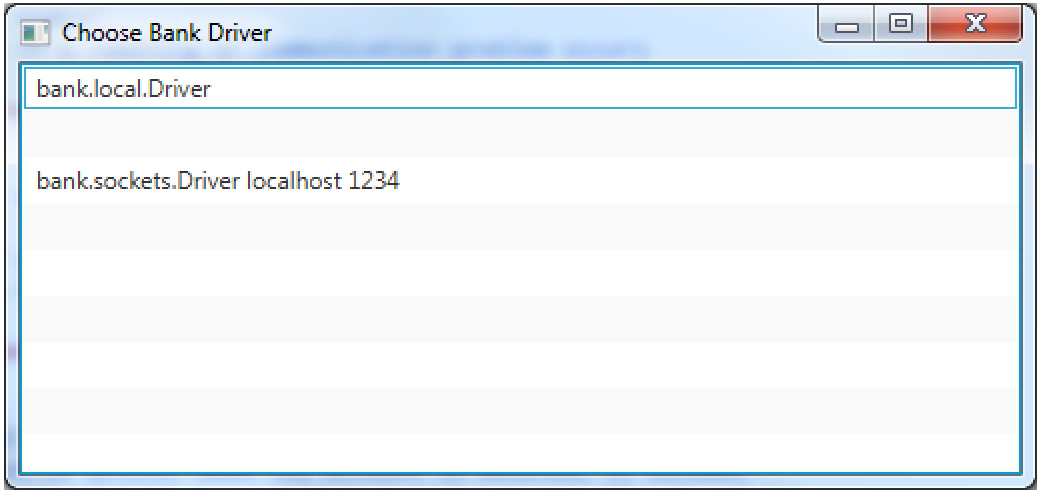
```
java bank.Client bank.local.Driver
java bank.Client bank.socket.Driver 147.86.8.30 8888
```

  - 1. Creates an instance of the specified driver
  - 2. invokes connect on the driver passing the additional arguments
  - 3. invokes getBank on the driver

Assignment 1: Bank

- **Start of the application**

```
java bank.Client
```



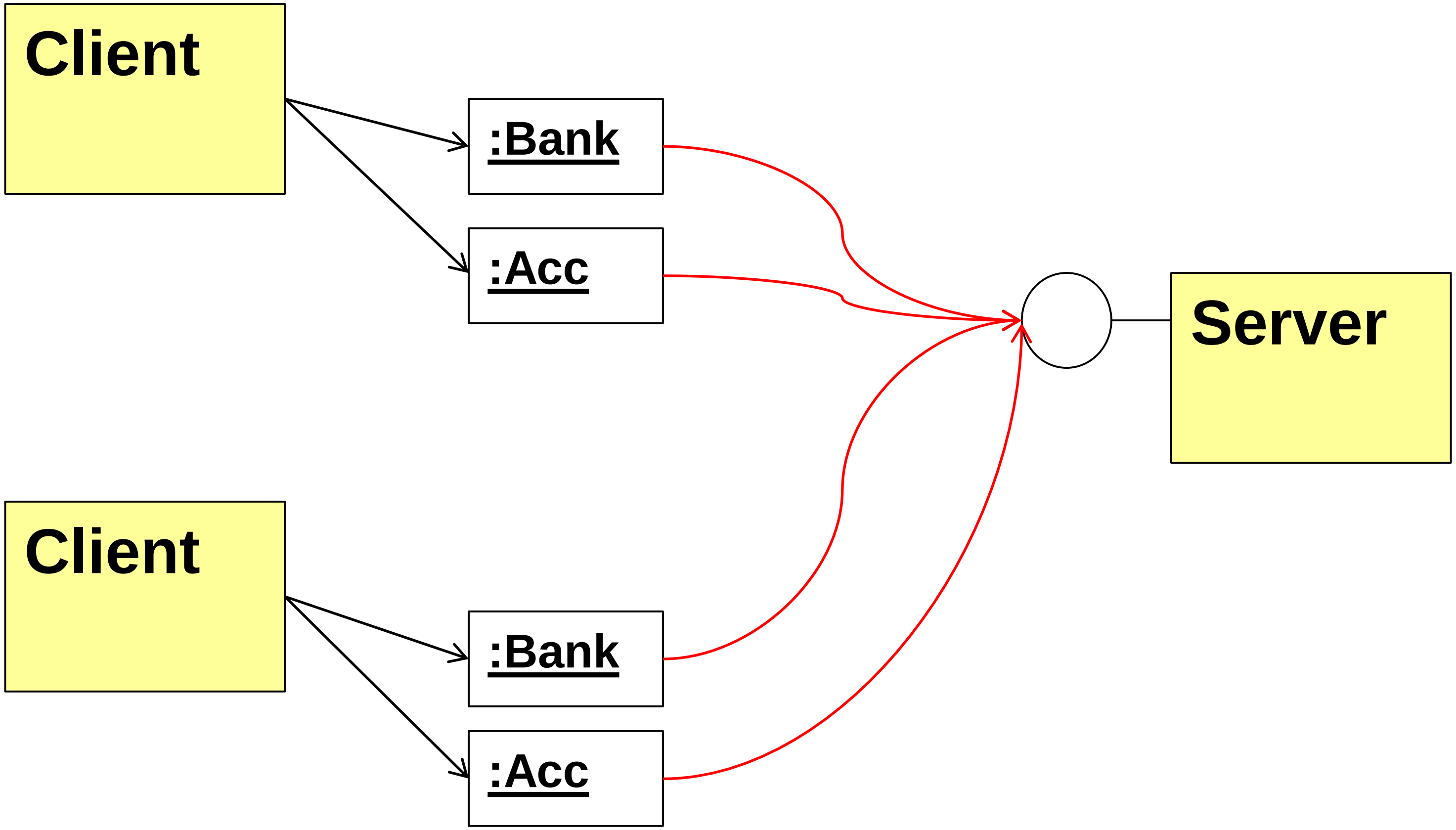
Choose Bank Driver

bank.local.Driver

bank.sockets.Driver localhost 1234

  - Arguments are stored in file `src/main/resources/Servers.txt`

Assignment 1: Socket Bank



Assignment 1: Socket Bank

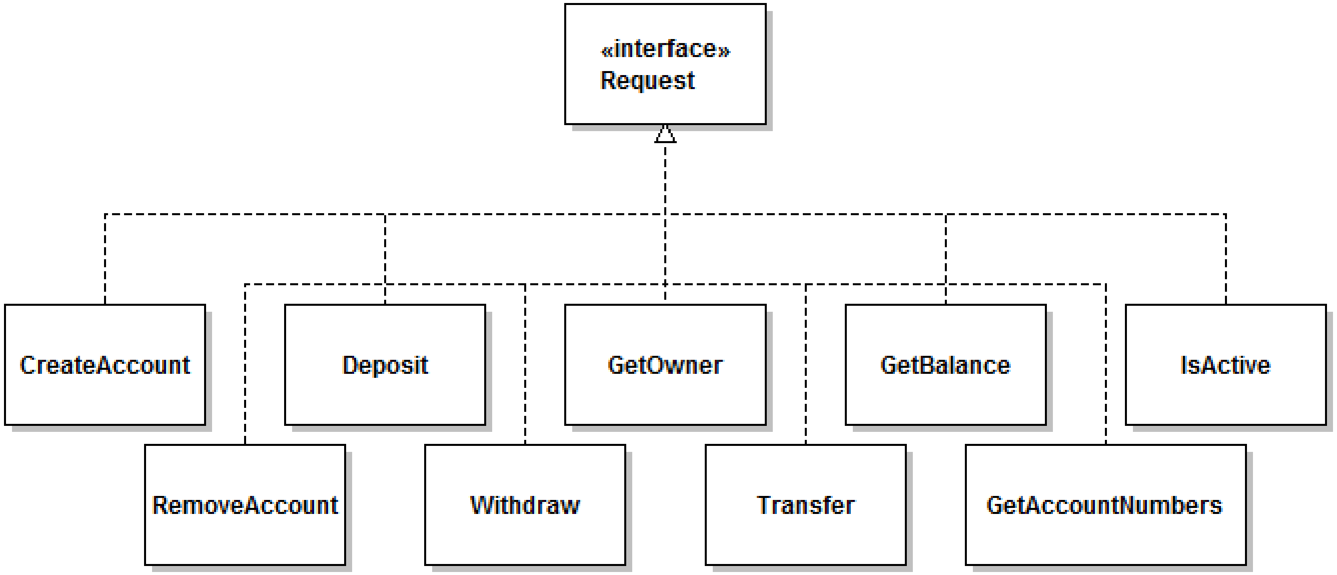
- **Text-Protocol**

```
=> "create" "Meier"
<= "ok" "22-345-0"

=> "remove" "22-345-0"
<= "true"

=> "get-acc-numbers"
<= 3 "11-345-0" "22-345-0" "33-345-0"

=> "withdraw" "22-345-0" "100.0"
<= true

=> "withdraw" "22-345-0" "100.0"
<= false "OverdrawException"
```

Assignment 1: Socket Bank

- **Object-Protocol**

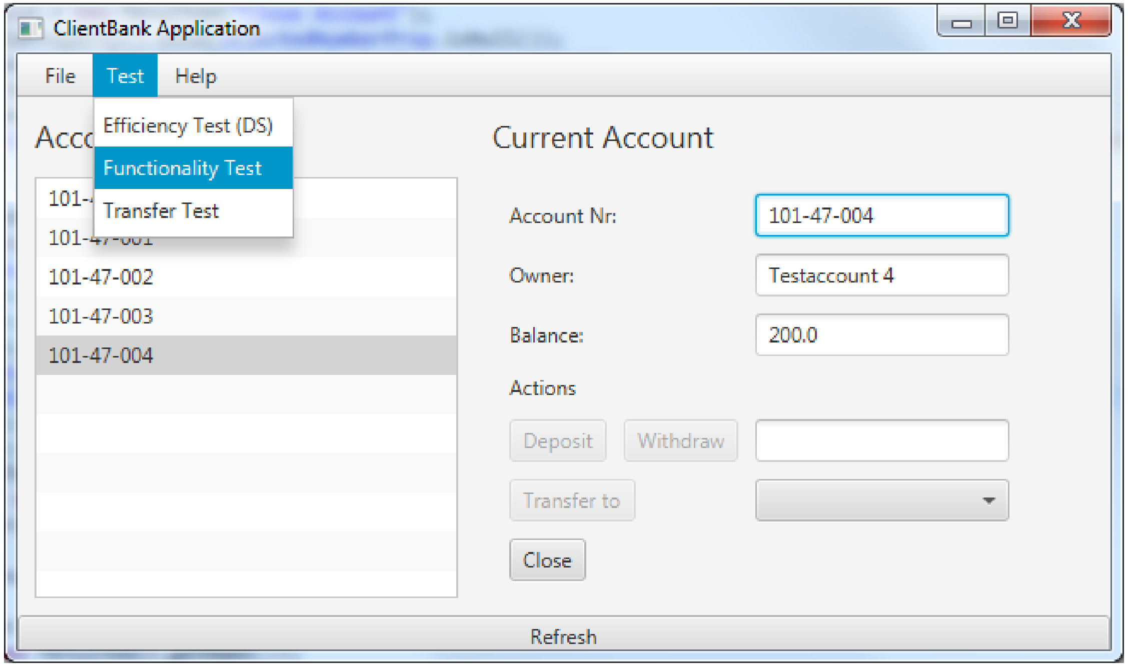
```
class CreateCommand implements Serializable {
  String name;
}
```



Assignment 1: Socket Bank

- **Tests**
  - Since the implementation of the bank is defined and loaded at runtime (using reflection) the use of JUnit is not trivial

    => Tests are provided as a menu entry



Account Nr: 101-47-004

Owner: Testaccount 4

Balance: 200.0