

EASYMOCK - FIRST APPLICATION

http://www.tutorialspoint.com/easymock/easymock_first_application.htm

Copyright © tutorialspoint.com

Before going into the details of the EasyMock Framework, let's see an application in action. In this example, we've created a mock of Stock Service to get the dummy price of some stocks and unit tested a java class named Portfolio.

The process is discussed below in a step-by-step manner.

Step 1: Create a JAVA class to represent the Stock

File: Stock.java

```
public class Stock {
    private String stockId;
    private String name;
    private int quantity;

    public Stock(String stockId, String name, int quantity){
        this.stockId = stockId;
        this.name = name;
        this.quantity = quantity;
    }

    public String getStockId() {
        return stockId;
    }

    public void setStockId(String stockId) {
        this.stockId = stockId;
    }

    public int getQuantity() {
        return quantity;
    }

    public String getTicker() {
        return name;
    }
}
```

Step 2: Create an interface StockService to get the price of a stock

File: StockService.java

```
public interface StockService {
    public double getPrice(Stock stock);
}
```

Step 3: Create a class Portfolio to represent the portfolio of any client

File: Portfolio.java

```
import java.util.List;

public class Portfolio {
    private StockService stockService;
    private List stocks;

    public StockService getStockService() {
        return stockService;
    }

    public void setStockService(StockService stockService) {
        this.stockService = stockService;
    }
}
```

```

    }

    public List getStocks() {
        return stocks;
    }

    public void setStocks(List stocks) {
        this.stocks = stocks;
    }

    public double getMarketValue(){
        double marketValue = 0.0;

        for(Stock stock:stocks){
            marketValue += stockService.getPrice(stock) * stock.getQuantity();
        }
        return marketValue;
    }
}

```

Step 4: Test the Portfolio class

Let's test the Portfolio class, by injecting in it a mock of stockservice. Mock will be created by EasyMock.

File: PortfolioTester.java

```

import java.util.ArrayList;
import java.util.List;

import org.easymock.EasyMock;

public class PortfolioTester {
    Portfolio portfolio;
    StockService stockService;

    public static void main(String[] args){
        PortfolioTester tester = new PortfolioTester();
        tester.setUp();
        System.out.println(tester.testMarketValue()?"pass":"fail");
    }

    public void setUp(){
        //Create a portfolio object which is to be tested
        portfolio = new Portfolio();

        //Create the mock object of stock service
        stockService = EasyMock.createMock(StockService.class);

        //set the stockService to the portfolio
        portfolio.setStockService(stockService);
    }

    public boolean testMarketValue(){

        //Creates a list of stocks to be added to the portfolio
        List<Stock> stocks = new ArrayList<Stock>();
        Stock googleStock = new Stock("1","Google", 10);
        Stock microsoftStock = new Stock("2","Microsoft",100);

        stocks.add(googleStock);
        stocks.add(microsoftStock);

        //add stocks to the portfolio
        portfolio.setStocks(stocks);

        // mock the behavior of stock service to return the value of various stocks
        EasyMock.expect(stockService.getPrice(googleStock)).andReturn(50.00);
        EasyMock.expect(stockService.getPrice(microsoftStock)).andReturn(1000.00);
    }
}

```

```
// activate the mock
EasyMock.replay(stockService);

double marketValue = portfolio.getMarketValue();
return marketValue == 100500.0;
}
}
```

Step 5: Verify the result

Compile the classes using **javac** compiler as follows:

```
C:\EasyMock_WORKSPACE>javac Stock.java StockService.java Portfolio.java
PortfolioTester.java
```

Now run the PortfolioTester to see the result:

```
C:\EasyMock_WORKSPACE>java PortfolioTester
```

Verify the Output

```
pass
```