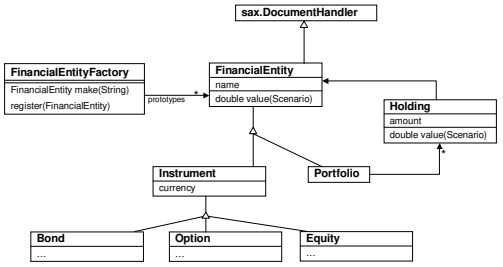
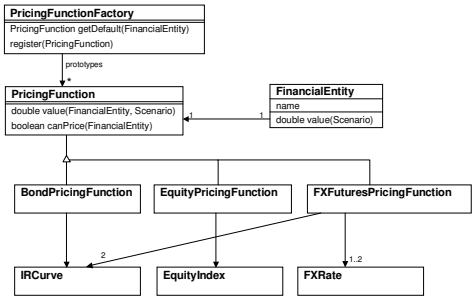
Ass 2/3 – One Solution

Instruments & Portfolios



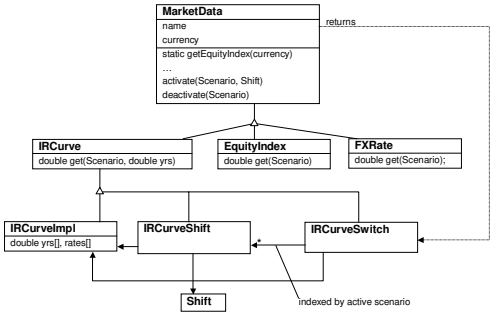
Pricing Functions



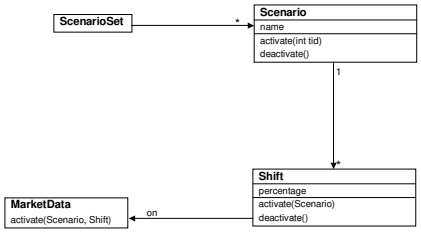
Pricing Function Code

```
public class EquityPricingFunction extends PricingFunction {
    public boolean canPrice(FinancialEntity fe) {
       return fe instanceof Equity;
    }

    public double value(FinancialEntity fe, Scenario s) {
        Equity e = (Equity)fe;
        ...
    }

    public initialize(FinancialEntity fe) {
        Equity e = (Equity)fe;
        equityIndex = MarketData.getEquityIndex(fe.currency());
    }

    private EquityIndex equityIndex;
}
```

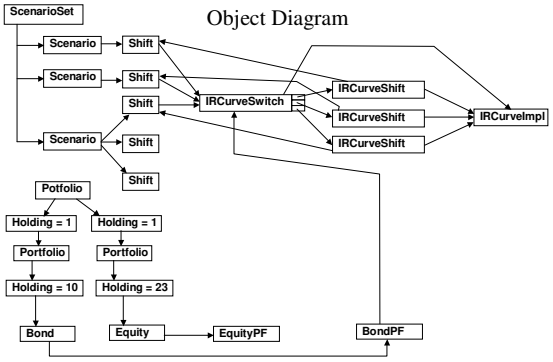

MarketData



Scenarios



Object Diagram



Scenario Activation

```
Thread.run() {
    int tid = ThreadSet.getNextId();
    while(ScenarioSet.stillMore()) {
       Scenario s = ScenarioSet.getNextScenarioToProcess();
       s.activate(tid);
       s.saveResult(Portfolio.getRoot().value(s));
       s.deactivate();
    }
}

Scenario.activate(tid) {
    for each Shift s {
        shift.activate(this);
    }
}

Shift.activate(Scenario s) {
    marketData.activate(s, this)
}

EquityIndexSwitch.activate(Scenario s, Shift shift) {
    activeScenarios[s.getId()] = new EquityIndexShift(shift);
}
```


MarketData Pricing Code

```
public class EquityIndexImpl extends EquityIndex {
    public double get(Scenario s) {
        return level;
    }
    double level;
}

public class EquityIndexShift extends EquityIndex {
    public double get(Scenario s) {
        return e.get(s) * (1+shift.getPercentage());
    }
    EquityIndex e;
    Shift shift;
}

public class EquityIndexSwitch extends EquityIndex {
    public double get(Scenario s) {
        if(s == null)
            return eq.get(s);
        else
            return activeScenarios[s.getId()].get(s);
    }
    EquityIndex activeScenarios[] = new EquityIndex[MAX_THREADS];
    EqutyIndex eq;
}
```