The 2nd Demonstration

문휘식, 김형규, 박정환, 김인교

ⓒSaebyeol Yu. Saebyeol's PowerPoint

목차

- DVM 개발 전체 과정
- Design Pattern 2개 적용, 전/후 모습, 이유, 장단점
  - Singleton Pattern
  - Template Method Pattern
- Clean Code & PMD 적용, 전/후 모습, 평가
- OOAD 개발방법론 장점 & 단점
- OOAD 적용가능성 & 개선점
- 실제 현업 기준 OOAD 선결 조건 및 지원방법

ⓒSaebyeol Yu. Saebyeol's PowerPoint

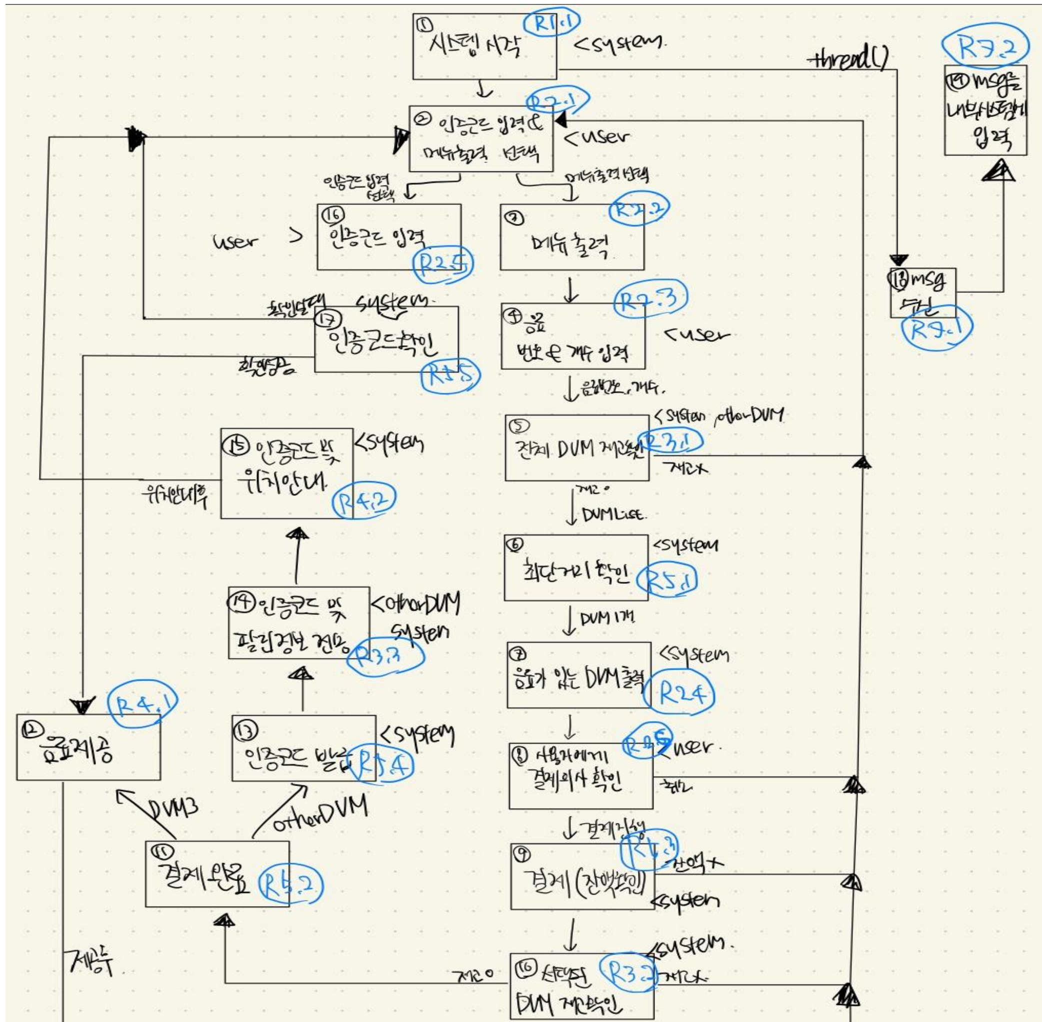
1

DVM 개발 전체 과정

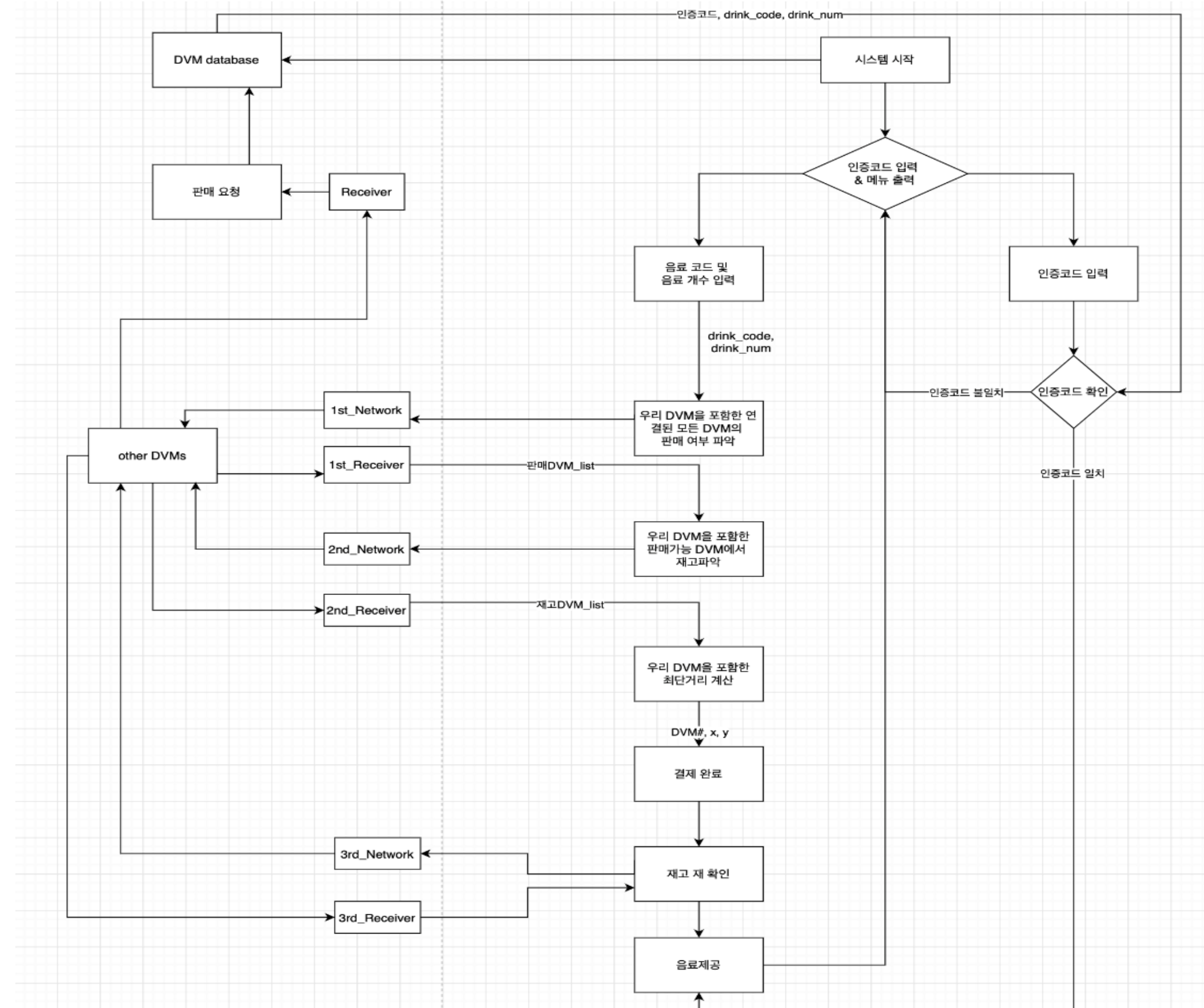
ⓒSaebyeol Yu. Saebyeol's PowerPoint

1 DVM 개발 전체 과정 Flow

기존



최종



2

Design Pattern 2개 적용, 전/후 모습, 이유, 장단점

ⓒSaebyeol Yu. Saebyeol's PowerPoint

2 Singleton Pattern

적용 전

```java
public class Dvmmain {
    ≗ lunarmoon7 *
    public static void main(String[] args) {
//      ServerThreadTest serverThreadTest = new ServerThreadTest();
//      ClientThreadTest clientThreadTest = new ClientThreadTest();
//      serverThreadTest.start();

//      DVM dvm = DVM.getInstance();
        DVM dvm = new DVM();
        Receiver receiver = new Receiver(dvm);
        Boolean networkConnect = true;  //임시로 일단 true라고 설정했습니다.
        if (networkConnect) {
            receiver.start();   //네트워크 확인 되면 시작
        }
        Admin admin = new Admin(networkConnect, dvm);     //admin에서 system start()
    }
}
```

적용 후

```java
public class Dvmmain {
    ≗ lunarmoon7 *
    public static void main(String[] args) {
        DVM dvm = DVM.getInstance();
        dvm.start();
    }
}
```

```java
public void start() {
    boolean networkCoonect = true;

    Admin admin = Admin.getInstance(networkCoonect, dvm);
    if(networkCoonect) {
        admin.systemStart();
    }
}
```

```java
    1개 사용 위치  ≗ lunarmoon7
    public static DVM getInstance() {
        return dvm;
    }
```

* 이유, 장/단점 써야 함.

ⓒSaebyeol Yu. Saebyeol's PowerPoint

2 Singleton Pattern

적용 전

```java
Admin admin = new Admin(networkCoonect, dvm);
```

```java
public Receiver(DVM dvm) {
    this.dvm = dvm;
    this.server = new DVMServer();
    this.serializer = new Serializer();
}
```

적용 후

```java
Admin admin = Admin.getInstance(networkCoonect, dvm);
```

```java
public static Admin getInstance(Boolean networkStatus, DVM dvm)
{
    admin = new Admin(networkStatus, dvm);
    return admin;
}
```

```java
private static Receiver receiver = new Receiver();

hwan-cs *
public static Receiver getInstance() {
    receiver.dvm = dvm;
    receiver.server = new DVMServer();
    receiver.serializer = new Serializer();
    return receiver;
}
```

* 이유, 장/단점 써야 함.

ⓒSaebyeol Yu. Saebyeol's PowerPoint

2 Singleton Pattern

적용 전

```java
public ServerThreadTest() {
}
```

적용 후

```java
private static ServerThreadTest serverThreadTest = new ServerThreadTest();
```

```java
private ServerThreadTest() {
}
```

```java
public static ServerThreadTest getInstance()
{
    return serverThreadTest;
}
```

* 이유, 장/단점 써야 함.

ⓒSaebyeol Yu. Saebyeol's PowerPoint

2 Singleton Pattern

적용 이유 & 장점

- 메모리 낭비를 방지
- 이미 생성된 인스턴스를 사용해서 속도 측면에서 이점이 있음
- 다른 클래스 간의 데이터 공유가 쉬워짐

단점

- 테스트가 어려움
- 생성자가 private이 되므로 상속에 제한
- 제대로 적용할 줄 모르면 코드가 안 돌아갈 수 있음

ⓒSaebyeol Yu. Saebyeol's PowerPoint

2 Template Method Pattern

```java
public abstract class AbstractDVMClass
{
    5개 사용 위치  1개 구현
    protected abstract void calcClosestDVMLoc();
    public void templateCalcClosestDVMLoc()
    {
        calcClosestDVMLoc();
    }
    4개 사용 위치  1개 구현
    protected abstract boolean checkOurDVMStock(String drinkCode, int drinkNum);
    public void templateCheckOurDVMStock(String drinkCode, int drinkNum)
    {
        checkOurDVMStock(drinkCode, drinkNum);
    }
    1개 사용 위치  1개 구현
    protected abstract void start();
    1개 사용 위치
    public void templateStart()
    {
        start();
    }
}
```

```java
public abstract class AbstractAdminClass
{
    2개 사용 위치  1개 구현
    protected abstract void refillDrink();
    public void templateRefillDrink()
    {
        refillDrink();
    }

}
```

ⓒSaebyeol Yu. Saebyeol's PowerPoint

2 Template Method Pattern

적용 이유 & 장점

- 중복되는 코드를 줄일 수 있음
- 추상클래스를 구현한 서브클래스의 역할을 줄여서 핵심 로직 관리가 용이
- 코드를 객체지향적으로 구성할 수 있음
- 쉬운 확장성

단점

- 추상클래스를 구현한 서브클래스의 경우 다른 클래스의 상속이 불가능
- 추상 메소드가 많아지면서 클래스의 관리가 복잡해짐

ⓒSaebyeol Yu. Saebyeol's PowerPoint

3

Clean Code & PMD 적용, 전/후 모습, 평가

ⓒSaebyeol Yu. Saebyeol's PowerPoint

2 Clean Code 적용 & 전/후 모습 & 평가

변경 전 / **변경 후**

```
private HashMap<String, Message> ODRCHashMap;
```

- 상대 DVM에서 받은 인증코드 저장하는 변수
-> ODRCHashMap가 어떤 의미인지 이해하기 힘듬

```
private HashMap<String, Message> receivedVerifyCodeMap;
```

- 변수를 봤을 때 어떤 역할을 하는지 알 수 있게 변경
-> receivedVerifyCodeMap

```
private String[] calcDVMInfo
```

- 계산된 다른 DVM들의 정보를 저장하는 변수
-> 줄임 표현 calc를 사용

```
private String[] calculatedDVMInfo
```

- 명확한 의도가 보이도록 줄임 표현을 없앰.

```
public HashMap<String, Message> getODRCHashMap() {
    return ODRCHashMap;
}
```

- Getter함수 명을 보고
어떤 변수를 get하는 함수인지 알기 어려움

```
public HashMap<String, Message> getreceivedVerifyCodeMap() {
    return receivedVerifyCodeMap;
}
```

- 어떤 변수를 get하는지 알 수 있게 이름을 변경

ⓒSaebyeol Yu. Saebyeol's PowerPoint

2 Clean Code 적용 & 전/후 모습 & 평가

변경 전

```java
public String[] getCalcDVMInfo() {
    return this.calcDVMInfo;
}
```

- 줄임 표현을 사용한 getter 함수

```java
public Drink[] getDrinkList() {
    return this.drinkList;
}
```

(생략)

```java
private JTextField jtf;
```

- 특정 Dialog 클래스에서 인증코드를 입력할 수 있는 JTextField
-> 코드 작성자가 아니면 어떤 변수인지 파악하기 힘듬

변경 후

```java
public String[] getCalculatedDVMInfo() {
    return this.calculatedDVMInfo;
}
```

- 줄임 표현을 없앰

```java
public Drink[] getEntireDrinkList() {
    return this.entireDrinkList;
}
```

(생략)

```java
private JTextField verifyCodeField;
```

- 의도에 맞는 이름으로 변수명 변경

ⓒSaebyeol Yu. Saebyeol's PowerPoint

2 Clean Code 평가

Clean Code 적용 후 평가

- 변수 명이나 함수 명을 줄임 표현을 써서 작성하거나 의도에 맞지 않는 표현을 써서 작성을 하게 되면 코드를 보는 사람이 어떤 함수, 변수인지 이해하기에 매우 어렵다.
- 당연하게도 코드 작성자도 자신이 작성한 코드를 알아보지 못할 가능성이 높다.

- **하지만 Clean Code에 맞게 코드를 작성하게 되면, 의도가 명확해서 코드를 [읽기 & 이해] 하기가 쉽다.**
- **적용해보면서 느낀 것은 확실히 코드가 깔끔 해지고 잘 짜여진 느낌이 든다는 것이다.**

ⓒSaebyeol Yu. Saebyeol's PowerPoint

적용 전 PMD 결과 → 총 51개

(빨간 느낌표 만 처리)

- FieldNamingConventions (14 violations)
  - (14, 35) DVM
  - (9, 12) DialogConfirmPayment
  - (12, 30) Network
  - (14, 30) Network
  - (15, 30) Network
  - (16, 30) Network
  - (17, 30) Network
  - (18, 30) Network
  - (14, 30) Receiver
  - (15, 30) Receiver
  - (16, 30) Receiver
  - (17, 30) Receiver
  - (18, 30) Receiver
  - (19, 30) Receiver
- VariableNamingConventions (21 violations)
  - (14, 35) DVM
  - (9, 12) DialogConfirmPayment
  - (12, 30) Network
  - (14, 30) Network
  - (15, 30) Network
  - (16, 30) Network
  - (17, 30) Network
  - (18, 30) Network
  - (35, 38) Network.sendBroadcastMsg()
  - (103, 54) Network.sendSoldDrinkInfo()
  - (103, 39) Network.sendSoldDrinkInfo()
  - (14, 30) Receiver
  - (15, 30) Receiver
  - (16, 30) Receiver
  - (17, 30) Receiver
  - (18, 30) Receiver
  - (19, 30) Receiver
  - (50, 10) Receiver.handleStockCheckRequestAndSend()
  - (51, 10) Receiver.handleStockCheckRequestAndSend()
  - (84, 10) Receiver.handleSaleCheckRequestAndSend()
  - (85, 10) Receiver.handleSaleCheckRequestAndSend()

- FormalParameterNamingConventions (3 violations)
  - (35, 38) Network.sendBroadcastMsg()
  - (103, 54) Network.sendSoldDrinkInfo()
  - (103, 39) Network.sendSoldDrinkInfo()
- LocalVariableNamingConventions (4 violations)
  - (50, 10) Receiver.handleStockCheckRequestAndSend()
  - (51, 10) Receiver.handleStockCheckRequestAndSend()
  - (84, 10) Receiver.handleSaleCheckRequestAndSend()
  - (85, 10) Receiver.handleSaleCheckRequestAndSend()

- ConstructorCallsOverridableMethod (9 violations)
  - (12, 4) Admin.Admin()
  - (35, 3) Controller.Controller()
  - (38, 9) DialogClosetDVM.DialogClosetDVM()
  - (14, 9) DialogOption.DialogOption()
  - (36, 9) DialogPrintMenu.DialogPrintMenu()
  - (37, 9) DialogPrintMenu.DialogPrintMenu()
  - (38, 9) DialogPrintMenu.DialogPrintMenu()
  - (15, 9) DialogVerficationCode.DialogVerficationCode()
  - (26, 3) Network.Network()

ⓒSaebyeol Yu. Saebyeol's PowerPoint

적용 후 PMD 결과 → 총 6개

ConstructorCallsOverridableMethod (6 violations)
- (38, 9) DialogClosetDVM.DialogClosetDVM()
- (14, 9) DialogOption.DialogOption()
- (36, 9) DialogPrintMenu.DialogPrintMenu()
- (37, 9) DialogPrintMenu.DialogPrintMenu()
- (38, 9) DialogPrintMenu.DialogPrintMenu()
- (15, 9) DialogVerficationCode.DialogVerficationCode()

적용 후 아직 남아있는 Violation의 경우 UI를 담당하는 Dialog 클래스에서 발생했다.
(시스템 내장 함수를 각 클래스의 생성자에서 호출해서 오류가 발생)

→ 코드 수정 후 UI가 돌아가지 않으면 로직이 원활히 돌아가는지 알 수 없음 & 메인 로직을 담당하는 클래스가 아니어서 수정하지 않음.

ⓒSaebyeol Yu. Saebyeol's PowerPoint

4

OOAD 개발방법론

장점 & 단점

ⓒSaebyeol Yu. Saebyeol's PowerPoint

4 장점

장점

문휘식

- 이제까지 해왔던 팀 프로젝트는 문서작업 없이 바로 코드 구현으로 들어갔었는데, OOAD를 적용한 이번 프로젝트는 처음에 문서 작업부터 구현까지 체계적으로 방향이 짜여져 있어서 좋았다.
- 또한, 각 단계마다 iteration이 있어서 처음에 충분히 작성하지 못했더라도 다음 Iteration에 작성이 가능해서 좋았다.
- 같은 이야기일 수 있지만, Method 작성을 하면서 순간 어떻게 작동 하는거였지? 하는 생각이 들 때 이전에 작성했던 Sequence Diagram, Use-case 같은 것들을 보면서 작동원리를 정확히 짚고 넘어갈 수 있어서 좋았다.

김인교

- 확실히 기존의 코딩보다 코드가 잘못되었을 때, 고쳐야 하는 부분을 쉽게 알 수 있었습니다.
- 각 클래스가 어떤 역할을 하는지 넓은 시각으로 볼 수 있어서 어려운 과정을 쉽게 생각할 수 있었습니다.

ⓒSaebyeol Yu. Saebyeol's PowerPoint

장점

김형규
- 요구 사항을 만족시키는 SW를 체계적으로 개발이 가능하다.
- 어떤 생각으로 이 시스템을 생각했지? 라는 생각이 들 때 문서를 보면 다시 기억나면서 작성이 가능해진다.

박정환
- 각 단계마다 업그레이드 되면서 재미있는 개발이 가능해진다.

ⓒSaebyeol Yu. Saebyeol's PowerPoint

3 단점

단점

문휘식

- 반복적인 문서 작업과 지나치게 많은 문서량으로 인해 지칠 수 있다.
- 그리고 어떠한 것을 변경하면 이전에 작성했던 연관된 모든 문서를 수정해야 하는 번거로움이 존재한다.
- 초기(OOPT 1000)에서 충분히 논의가 이루어지지 않고 문서를 작성하게 된다면 후에 많은 어려움이 있다.
- 또한 연관된 문서 수정에서 어떤 문서의 어디 부분을 수정해야 하는지 찾는데에 번거로움이 있다.

김인교

- 함수들을 잘게 쪼개서 만들다보니함수들의 개수가 많아서 복잡했다.

- 함수를 만들 때, 함수의 기능을 오해하지 않도록 함수의 이름을 신중하게 지어야 했다. 그래서 이런 부분에서 신경이 쓰였다.

int

4 단점

김형규

- 초기에 팀원과의 소통이 이루어지지 않으면 중간에 수정하기가 힘들어진다.

박정환

- 반드시 지켜야하는 요구사항이 앞에서 제대로 정의되지 않으면 추후에 힘들어진다.

ⓒSaebyeol Yu. Saebyeol's PowerPoint

5

OOAD 적용 가능성 & 개선점

ⓒSaebyeol Yu. Saebyeol's PowerPoint

4 적용가능성

이번 학기 수업에서는 처음이기도 하고 시간적인 부담감이 컸었지만, 앞으로 어떠한 프로젝트(팀, 개인)든간에 이번 학기에서 느꼈던 아쉬운 부분들을 개선한 후
OOAD 방법론을 적용해서 문서작업부터 구현까지 체계적으로
수행할 수 있을 것 같다.

기회가 된다면 이번 학기에 다 배우지 못한 OOAD를 더 깊게 배워서 아쉬움을 느끼지 않도록 적용 해 보고싶다.
또한, 현업에서도 적용할 수 있겠다고 느꼈다.

ⓒSaebyeol Yu. Saebyeol's PowerPoint

4 개선점

3개월이라는 짧은 시간에 문서 작업, 구현까지 모두 해야해서 시간적인 부담감이 컸다.
그래서 조금 더 여유롭게 OOAD에 대해서 배우고 이해도를 높인 후에 적용할 수 있었으면 하는 아쉬움이 남았다.

PFR에서 설명이 좀 모호하거나 없는 부분이 프로젝트 수행에 있어서 헷갈리게 했다.
이런 부분은 더 구체화된 설명이 있으면 좋겠다.

문서 수정에서 Github 코드 관리 처럼 문서를 관리해주는 툴(?) 같은게 있으면 수정에 더 수월했을 것 같다.

ⓒSaebyeol Yu. Saebyeol's PowerPoint

6

실제 현업 기준 OOAD 선결 조건 및 지원 방법

ⓒSaebyeol Yu. Saebyeol's PowerPoint

5 OOAD 선결 조건

객체 지향 개발에 대한 이해도, 체계적인 문서 작업 능력, Github 같은 버전 관리 툴 사용 경험(필수는 아니지만 있으면 매우 유용)이 있으면 좋다.

언어(Java, C++ 등)에 대한 이해도, 코드 작성 능력이 있어야 한다고 생각한다.

추가로 현업에서는 당연하겠지만 의지와 뒷받침해주는 기본적인 실력이 있어야 한다고 생각한다.

ⓒSaebyeol Yu. Saebyeol's PowerPoint

5 OOAD 지원방법

아무래도 의견은 많을수록 좋다고 생각하기 때문에,
문서에 대해 다른 사람들의 생각을 듣고 피드백이 이
루어지면서
문서를 보다 완성도 있게 보충할 수 있으면 좋을 것 같
다.

ⓒSaebyeol Yu. Saebyeol's PowerPoint

Thanks!