제목	Status바가 있는 그림판 만들기	과제번호	3
학과	독일어과	학번	201602173
이름	윤영택	제출일	20200521

1. 문제 정의

GrimPan3FXML을 수정하여 Event가 발생했을 때 즉시 반영하여 Screen Size와 Shape Count를 보여주는 Status Bar가 있는 그림판을 만든다.

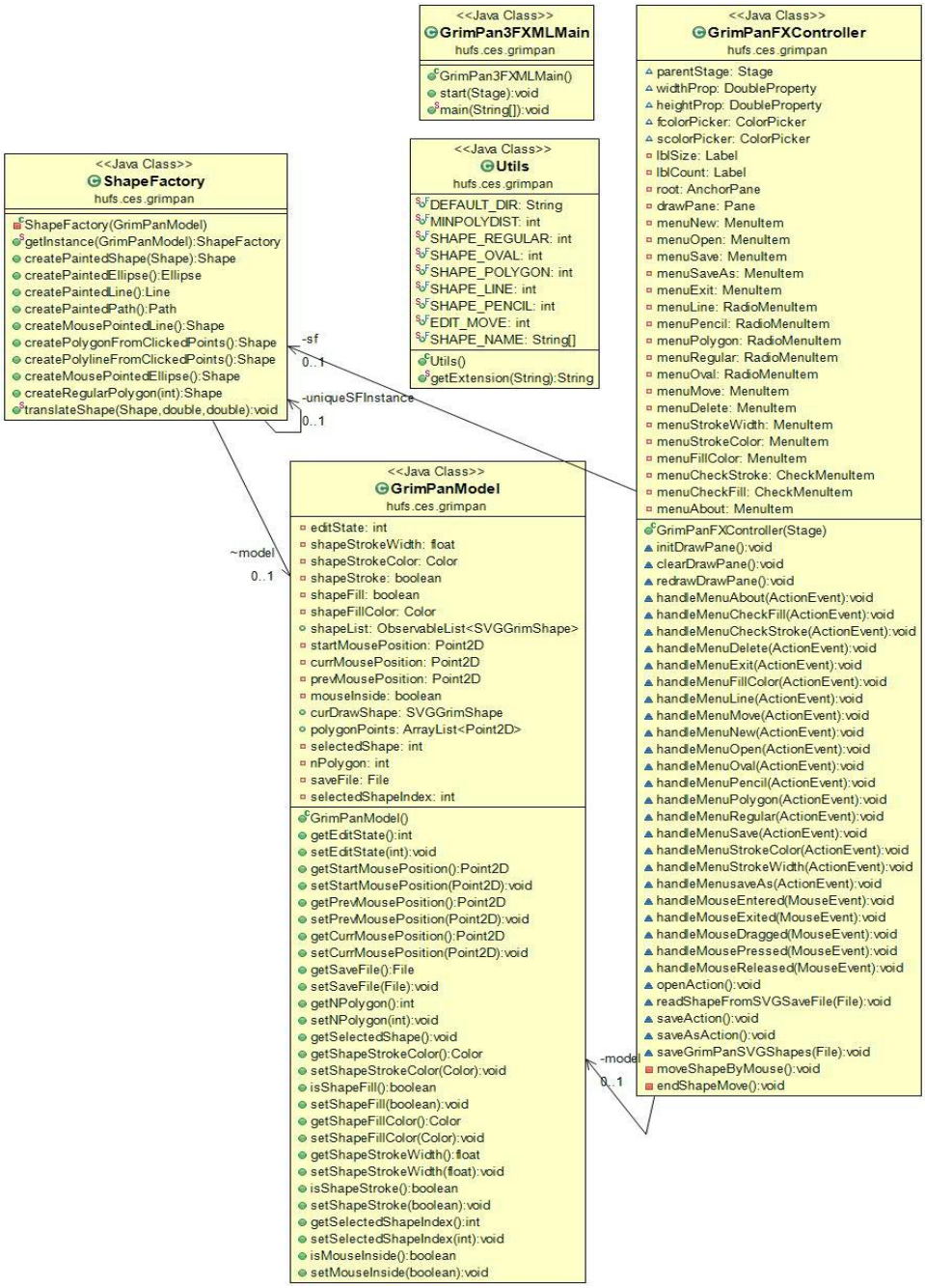
2-1. 문제 해결 방법

addListener메소드를 이용한다!

Event를 처리하는 각각의 addListener메소드 안에 Label.setText()구문을 추가한다. 단, .setText()메소드의 parameter로 들어가는 width와 height는 double타입이고 count는 int타입이므로 String.valueOf()를 사용하여 String타입으로 변환시켜준다.

2-2. 설계 결과(프로그램 구성도, 필요 함수, 클래스 설명)

1) 프로그램 구성도



2) 필요함수, 클래스 설명

1) GrimpanFXController
메인프로그램의 루트 컨테이너 역할을 하는 클래스이다.

2) public GrimPanFXController(Stage stage)
GrimpanFXController의 생성자로서, fxml파일을 load하여 해당 클래스에 적용하는 역할을 한다. 또한 Sceen Size와 Shape Count에 관련된 Event를 처리하기도 하며 initDrawPane() 메소드를 호출한다.

3) public void initDrawPane()
그림판 객체를 초기화하는 역할을 하는 메소드이다.

3. 결론 및 소감

3.1 과제 해결 중 생긴 문제와 해결 과정

문제 1: 클래스를 run하자 "오류: 기본 클래스 hufs.ces.grimpan.GrimPan3FXMLMain을(를) 찾거나 로드할 수 없습니다."라는 에러가 발생해서 실행되지 않았다.

-> 수업 중에 해결방법을 알게 된 문제였다. batik-all-1.10jar이라는 외부 라이브러리가 프로젝트의 Build Path에 포함되어 있었지만, 해당 라이브러리가 존재하지 않아서 실행되지 않았던 것이었다. 해당 라이브러리를 다운받아 해당 경로에 맞춰서 넣어주니 정상적으로 실행되었다.

문제 2: Binding expression을 FXML Document에 적용시키려 했으나 실패했다.

-> JavaScript처럼 Binding expression을 사용하여 FXML Document내부에 값을 전달하려 했으나, StackOverflow에는 모두 fxml파일이 load되면서 값을 전달하는 방식밖에 없었다. 이 방법은 Reactive하게 Event를 처리하기에는 적합하지 않다고 생각되었다. 결국 Binding expression을 포기하고 addListener메소드 안에 Label.setText()구문을 삽입했다. 이 문제를 해결하는 과정에서 많은 시간이 소요됐다.

문제 3: addListener 안에 Label.setText()을 넣자 수많은 예외가 발생했다.

-> 처음엔 addListener 안에서 Event가 발생할 때 Text를 set하기 위해서는 생성자보다 위에서 lblSize와 lblCount를 멤버변수(객체변수)로 선언해야 한다고 생각했다. 그래서 @FXML형태로 선언되어 있는 lblSize와 lblCount의 위치를 상단으로 옮겼다. 그러자 수많은 예외들이 발생했다. .setText()메소드를 사용하는 방식에는 문제가 없었다. 구글링을 한 결과, 그 원인을 찾아냈다. lblSize와 lblCount가 fxml을 load하는 과정에서 setting되어야 하는데, 클래스 내부의 멤버변수로 선언함으로써 setting이 제대로 이루어지지 않아서 Syntatx Error는 발생하지 않았지만 실행과정에서 에러가 발생했던 것이었다. 해결방법은 간단했다. lblSize와 lblCount를 원래 위치로 두고, addListener 안에 Lable.setText()구문을 넣었더니 정상적으로 실행되었다.

3.2 배운 점 & 소감

Property와 Binding, 그리고 Event를 처리하는 addListener의 개념과 사용법을 이해할 수 있게 되었다. 문제를 해결하는 과정에서 이들이 Reactive한 프로그래밍에선 필수 불가결한 존재라는 것을 알게 되었고 관련 API와 자료들을 찾아봤다. 이번 과제에서 내가 직접 코드를 작성한 부분은 아주 적었지만, 아주 중요한 개념들을 깨닫게 되었다는 점에서 의미 있었다. 그리고 Java8 API를 계속해서 찾아보는 습관을 기르게 되었다.