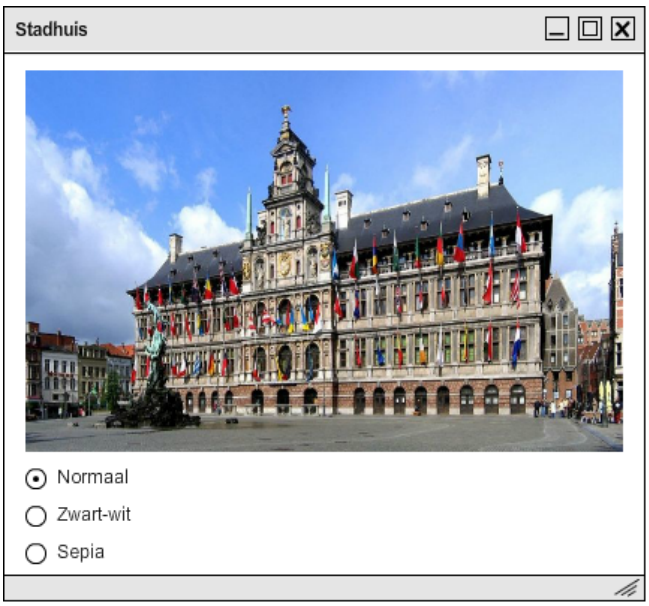
Karel de Grote Hogeschool

STADHUIS



In deze oefening maken we een applicatie waarmee we een visueel effect kunnen toepassen op een afbeeldling.

De voornaamste JavaFX klassen die we voor deze oefening nodig hebben zijn:

- `javafx.scene.layout.VBox`
- `javafx.scene.image.Image`
- `javafx.scene.image.ImageView`
- `javafx.scene.control.RadioButton`
- `javafx.scene.control.ToggleGroup`

Om de visuele effecten toe te passen gebruiken we:

- `javafx.scene.effect.ColorAdjust`
- `javafx.scene.effect.SepiaTone`

Raadpleeg in eerste instantie de JavaFX documentatie als je ergens vast zit!

1 WIREFRAME

Bouw de gegeven wireframe met een tool naar keuze.

2 HOOFDSCHERM AANMAKEN – MVP

Bij deze oefening is er geen of weinig achterliggende logica. Ze is bedoeld om de grafische capaciteiten van JavaFX in te oefenen en te demonstreren. We hebben, uitzonderlijk**(!)**, geen *model*.

De *view* klasse is **`StadhuisPane`**. Deze erft over van **`VBox`**. We werken deze klasse uit in punt 3.

De *presenter* klasse is **`Presenter`**. We werken deze klasse uit in punt 4.

De **`Main`** klasse is gegeven.

3 UI OPBOUWEN

De klasse **`StadhuisPane`**:

- Maak voor elk van de drie radiobuttons een attribuut aan van het type **`RadioButton`**.
  Maak voor de afbeelding een attribuut aan van het type **`ImageView`**.
- Maak voor elke radiobutton een getter met de juiste *access level*.
  Voor de control die de afbeelding toont hebben we geen getter nodig.

3.1 De methode initialiseNodes

- Initialiseer de vier controls
- Als je een afbeelding wil toekennen aan een **ImageView** dan heb je een **Image** nodig. Deze kan je aanmaken als volgt:
  - **new Image("be/kdg/stadhuis/view/images/stadhuis.jpg")**
- Zorg er voor dat de drie radiobuttons in eenzelfde **ToggleGroup** zitten. De **ToggleGroup** hoeft geen attribuut te zijn, je kan een lokale variabele gebruiken.
- De knop “Normaal” moet geselecteerd staan.
- Voorzie een ruimte van een 15-tal pixels tussen de verschillende controls. Dit doen we op **VBox**-niveau.

3.2 De methode layoutNodes

Voeg de vier controls toe aan de view. In een VBox is dit heel eenvoudig:

```
this.getChildren().add(mijnControl)
```

Dit zal er voor zorgen dat de componenten **in volgorde** onder elkaar staan.

3.3 De methode resetEffect

Deze methode zal opgeroepen worden door de presenter. Volgend stuk code zorgt voor een normale weergave van de afbeelding:

```
myImageView.setEffect(null);
```

3.4 De methode applyBlackAndWhiteEffect

Deze methode zal opgeroepen worden door de presenter. Volgend stuk code zorgt voor een zwart-wit weergave van de afbeelding:

```
ColorAdjust blackAndWhite = new ColorAdjust();
blackAndWhite.setSaturation(-1.0);
myImageView.setEffect(blackAndWhite);
```

3.5 De methode applySepiaEffect

Deze methode zal opgeroepen worden door de presenter. Volgend stuk code zorgt voor een sepia weergave van de afbeelding:

```
SepiaTone sepiaTone = new SepiaTone();
sepiaTone.setLevel(0.8);
this.imageView.setEffect(sepiaTone);
```

4 AFHANDELEN EVENTS

4.1 De methode addEventHandlers

Deze methode zorgt voor het toevoegen van de event handlers aan de radio buttons.

- Zorg voor een event handler voor elke radio button. Afhankelijk van de knop moet de juiste methode van **`StadhuisPane`** opgeroepen worden:
    - Normaal → methode **`resetEffect`**
    - Zwart-wit → methode **`applyBlackAndWhiteEffect`**
    - Sepia → methode **`applySepiaEffect`**