Objektovo orientované programovanie

Učiteľ:
Ing. Jozef Wagner, PhD.

Učebnica:
https://oop.wagjo.com/

OPG
Teória 24

1. Sledovanie zmien hodnôt
2. ObservableValue a addListener()
3. property objekt

Udalosti a sledovanie zmien

Keď užívateľ vykoná akciu (klik, klávesa), JavaFX vygeneruje udalosť, ktorú vieme spracovať

Niektoré komponenty v sebe obsahujú hodnotu - číslo alebo text.

Napríklad `textField` obsahuje text a `slider` číselnú hodnotu

Chceme byť informovaní, čo sa s hodnotou deje, ako sa zmenila

Doteraz sme to robili pomocou udalostí. Tento prístup však komplikuje kód a nerieši všetky situácie, ktoré môžu nastať

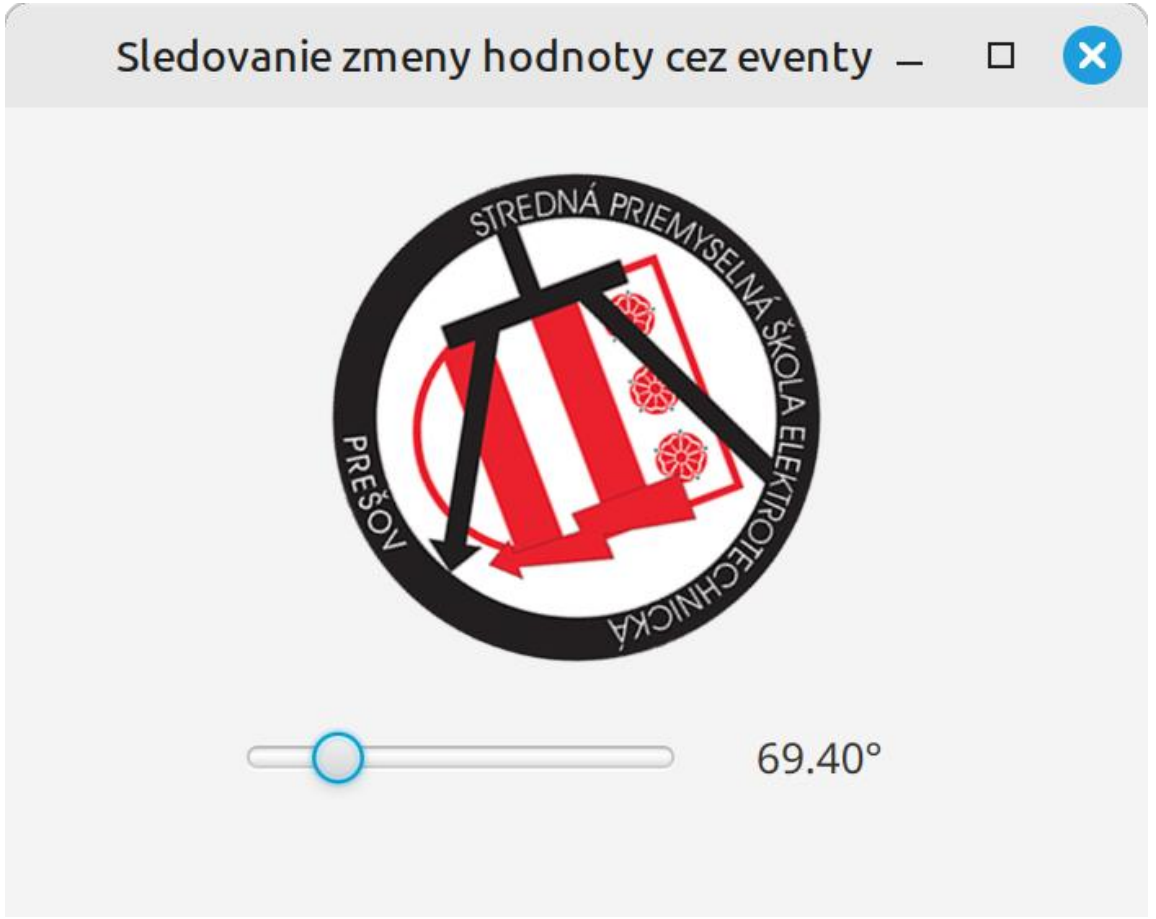
Udalosti a sledovanie zmien

Treba rozlišovať medzi:

- Užívateľ vykonal nejakú akciu
- Zmenila sa hodnota nejakého komponentu

JavaFX udalosti - eventy sú určené na ošetrenie akcií, nie na sledovanie zmien hodnôt!

Hlavný dôvod je, že hodnota sa môže meniť aj iným spôsobom, ako udalosťou, ktorú zachytávame.

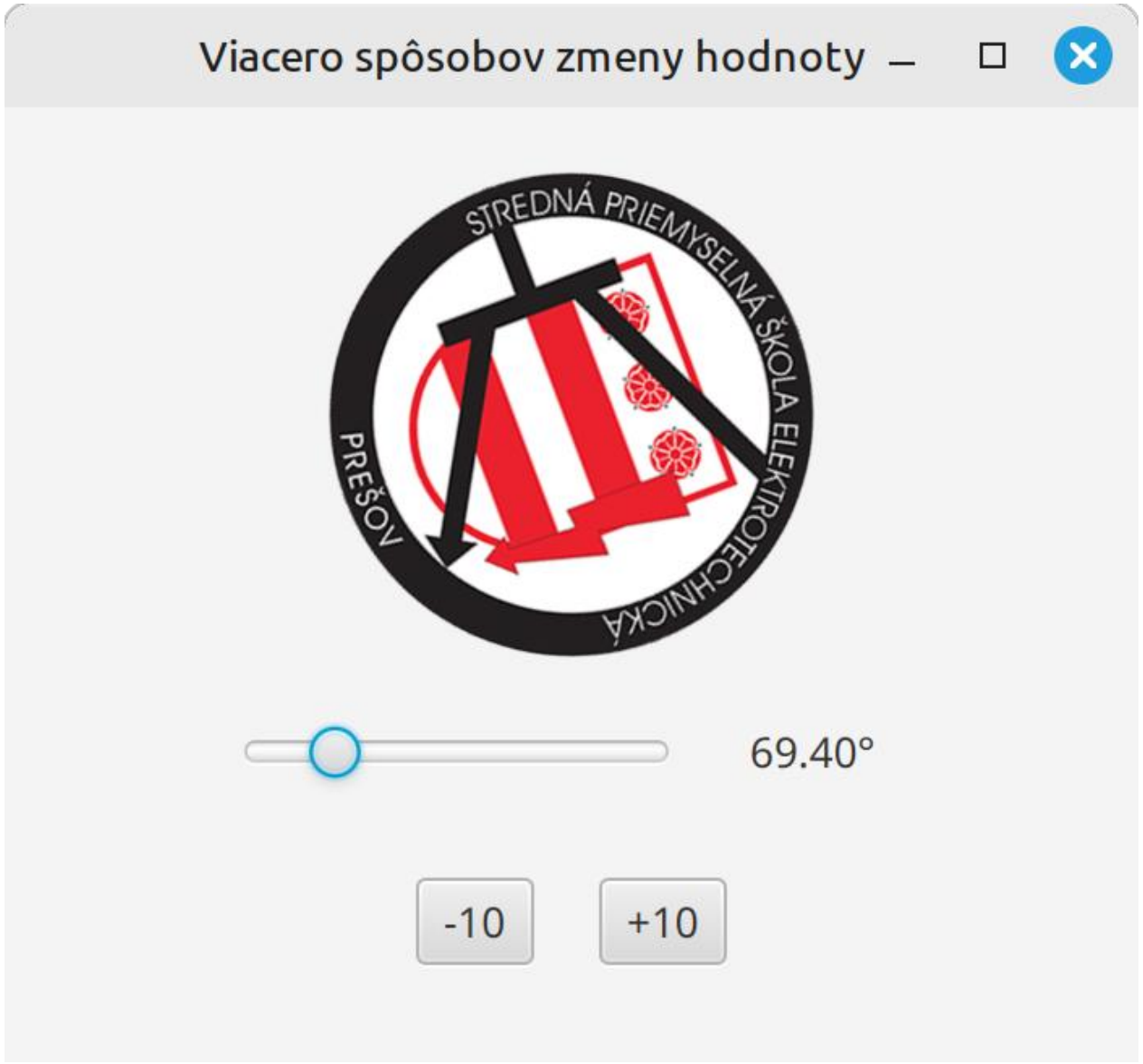



```
public class Controller {

    public ImageView obrazok;
    public Slider slider;
    public Label label;

    public void updatuj(MouseEvent mouseEvent) {
        double hodnota = slider.getValue();
        obrazok.setRotate(hodnota);
        label.setText(String.format("%.2f°", hodnota));
    }
}
```

- Ošetríme viac udalostí a pridáme tlačidlá



```
public ImageView obrazok;
public Slider slider;
public Label label;

public void updatujMysou(MouseEvent mouseEvent) {
    updatujSlider();
}

public void updatujKlavesou(KeyEvent keyEvent) {
    updatujSlider();
}

public void updatujSlider() {
    double hodnota = slider.getValue();
    obrazok.setRotate(hodnota);
    label.setText(String.format("%.2f°", hodnota));
}
```

```
public void updatujSlider() {
    double hodnota = slider.getValue();
    obrazok.setRotate(hodnota);
    label.setText(String.format("%.2f°", hodnota));
}

public void plus10(ActionEvent actionEvent) {
    double hodnota = slider.getValue();
    hodnota = Math.min(hodnota + 10, 360);
    slider.setValue(hodnota);
    obrazok.setRotate(hodnota);
    label.setText(String.format("%.2f°", hodnota));
}

public void minus10(ActionEvent actionEvent) {
    double hodnota = slider.getValue();
    hodnota = Math.max(hodnota - 10, 0);
    slider.setValue(hodnota);
    obrazok.setRotate(hodnota);
    label.setText(String.format("%.2f°", hodnota));
}
```

ObservableValue

Na sledovanie zmeny hodnoty používame interface `ObservableValue`

Rozhranie `ObservableValue` umožňuje 'počúvať' na zmenu hodnoty

Cez `ObservableValue` si vieme zaregistrovať metódu, ktorá sa spustí, ak sa hodnota zmenila.

Počúvanie na zmenu hodnoty

Počúvanie na zmenu hodnoty sa robí pomocou metódy `addListener()`

JavaFX komponenty s hodnotou poskytujú objekt, ktorý podporuje `ObservableValue` rozhranie.

Takýto objekt sa volá `property`, získame ho väčšinou pomocou metódy `valueProperty()`

```
slider.valueProperty().addListener(
    (observable, oldValue, newValue) -> {
    ...
  });
```

Príklad

Určíme si, kde v našom programe bude hlavné miesto - objekt - v ktorom sa bude uchovávať hodnota natočenia.

Dobrým kandidátom na takýto účel je `slider`, ktorý v sebe uchováva `double` hodnotu.

- Pre aktualizáciu stavu aplikácie nám stačí počúvať na zmenu hodnoty na jednom mieste - slideri
- Ak chceme hodnotu meniť manuálne, budeme ju meniť v slideri

```
public void initialize() {
  slider.valueProperty().addListener(
    (observable, oldValue, newValue) -> {
    obrazok.setRotate(newValue.doubleValue());
    label.setText(String.format("%.2f°", newValue.doubleValue()));
  });
}

public void plus10(ActionEvent actionEvent) {
  double hodnota = slider.getValue();
  hodnota = Math.min(hodnota + 10, 360);
  slider.setValue(hodnota);
}

public void minus10(ActionEvent actionEvent) {
  double hodnota = slider.getValue();
  hodnota = Math.max(hodnota - 10, 0);
  slider.setValue(hodnota);
}
```

Záver

Porovnanie udalostí a sledovania zmien hodnoty:

- `addEventFilter` - zachytenie udalosti - **capturing**
- `addEventHandler` - ošetrenie udalosti - **handling**
- `addListener` - sledovanie zmeny hodnoty - **listening**

Nad udalosťou vieme zavolať `addEventFilter` a `addEventHandler`, nad `property` voláme `addListener`.