JavaFX-- Properties

Tutorial 8 (7th April 2021)

Properties

- Sometimes redundancy/duplication of information is unavoidable
  - E.g. if you display information about the model using a JavaFX component (Label, Text). These components often maintain information themselves.
- Duplication gives rise to synchronisation problems
  - How do you keep software components up-to-date?
- Conflicts with the MVC-principle
- Solution: use the observer pattern to propagate changes of data
- Or better: use properties.

Properties: basic example

```
private static void testIntegerPropertyBinding() {
    IntegerProperty length  = new SimpleIntegerProperty (10);
    IntegerProperty width   = new SimpleIntegerProperty (30);
    IntegerProperty area    = new SimpleIntegerProperty ();

    area.bind(length.multiply(width));

    System.out.println("Area: " + area.getValue());

    length.setValue(20);

    System.out.println("Area: " + area.getValue());

    width.setValue(40);

    System.out.println("Area: " + area.getValue());
}
```

```
run:
Area: 300
Area: 600
Area: 800
BUILD SUCCESSFUL (total time: 0 seconds)
```

Properties: resizing an ellipse


```
public void start( Stage stage ) {
  Ellipse ellipse = new Ellipse( 100, 100 );
  ellipse.setFill( Color.RED );
  Pane root = new StackPane( ellipse );
  ellipse.radiusXProperty().bind(root.widthProperty().multiply(0.45));
  ellipse.radiusYProperty().bind(root.heightProperty().multiply(0.45));
  Scene scene = new Scene(root, 200, 100);
  stage.setTitle(this.getClass().getSimpleName());
  stage.setScene(scene);
  stage.show();
}
```

Listeners: resizing an ellipse

```
public void start( Stage stage ) {
  Ellipse ellipse = new Ellipse( 100, 100 );
  ellipse.setFill( Color.RED );
  Pane root = new StackPane( ellipse );
  root.widthProperty().addListener( (obs, oldVal, newVal) ->
          ellipse.radiusXProperty().setValue(newVal.doubleValue()*0.45));
  root.heightProperty().addListener( (obs, oldVal, newVal) ->
          ellipse.radiusYProperty().setValue(newVal.doubleValue()*0.45));
  Scene scene = new Scene(root, 200, 100);
  stage.setTitle(this.getClass().getSimpleName());
  stage.setScene(scene);
  stage.show();
}
```

Properties: resizing an ellipse (II)

```
public void start( Stage stage ) {
  Ellipse ellipse = new Ellipse( 100, 100 );
  ellipse.setFill( Color.RED );
  Pane root = new StackPane( ellipse );
  ellipse.radiusXProperty().bind( ellipse.layoutXProperty().multiply(0.9) );
  ellipse.radiusYProperty().bind( ellipse.layoutYProperty().multiply(0.9) );
  Scene scene = new Scene(root, 200, 100);
  stage.setTitle(this.getClass().getSimpleName());
  stage.setScene(scene);
  stage.show();
}
```

Properties: mixing ints and doubles

```
private static void testDoubleToIntegerBinding() {
    DoubleProperty  a     = new SimpleDoubleProperty ();
    IntegerProperty b     = new SimpleIntegerProperty (45);
    IntegerProperty c     = new SimpleIntegerProperty (30);

    a.bind( b.divide( c ) );

    System.out.println("a: " + a.getValue());
}
```

```
run-single:
a: 1.0
BUILD SUCCESSFUL (total time: 1 second)
```

Properties: auxiliary operations

```
private static void testDoubleToIntegerBinding() {
    DoubleProperty  adj  = new SimpleDoubleProperty (3);
    DoubleProperty  opp  = new SimpleDoubleProperty (4);
    DoubleProperty  hyp  = new SimpleDoubleProperty ();

    hyp.bind( ??? );

    System.out.println("hyp: " + hyp.getValue());

}
```

hypot

```
public static double hypot(double x,
                           double y)
```

Returns sqrt($x^2 + y^2$) without intermediate overflow or underflow.

Properties: auxiliary operations

```
private static void testDoubleToIntegerBinding() {
    DoubleProperty  adj  = new SimpleDoubleProperty (3);
    DoubleProperty  opp  = new SimpleDoubleProperty (4);
    DoubleProperty  hyp  = new SimpleDoubleProperty ();

    hyp.bind(Bindings.createDoubleBinding(() ->
            Math.hypot(adj.doubleValue(),opp.doubleValue()), adj, opp));

    System.out.println("hyp: " + hyp.getValue());
    adj.setValue(5);
    System.out.println("hyp: " + hyp.getValue());
    opp.setValue(12);
    System.out.println("hyp: " + hyp.getValue());
}
```

```
run-single:
hyp: 5.0
hyp: 6.4031242374328485
hyp: 13.0
BUILD SUCCESSFUL (total time: 1 second)
```

Finally