Sources:

GitHub Copilot

1. Using **Timeline, KeyValue, KeyFrame with Duration**

**A screenshot of a chat

AI-generated content may be incorrect.**

**A white background with black text

AI-generated content may be incorrect.**

**import javafx.animation.KeyFrame;**

**import javafx.animation.KeyValue;**

**import javafx.animation.Timeline;**

**import javafx.util.Duration;**

**import javafx.scene.layout.StackPane;**

**public void changeStackPaneHeight(StackPane stackPane) {**

**Timeline timeline = new Timeline();**

**KeyValue keyValue = new KeyValue(stackPane.prefHeightProperty(), 50);**

**KeyFrame keyFrame = new KeyFrame(Duration.seconds(2), keyValue);**

**timeline.getKeyFrames().add(keyFrame);**

**timeline.play();**

**}**

**This method will animate the height of the given StackPane from its current height to 50 over a duration of 2 seconds.**

In JavaFX, KeyFrame, KeyValue, and Timeline are used to create animations. Here's a brief explanation of each:

**1. KeyFrame**:

2. A KeyFrame defines a specific point in time within an animation and the values that should be applied to the properties of the animated objects at that time.

3. It consists of a Duration (the time at which the key frame occurs) and one or more KeyValue objects.

**4. KeyValue**:

5. A KeyValue represents the end value of a property at a specific point in time during the animation.

6. It links a property to a target value, which the property should reach at the specified time defined in the KeyFrame.

**7. Timeline**:

8. A Timeline is a sequence of KeyFrame objects that define the animation.

9. It controls the playback of the animation, including starting, stopping, and pausing the animation.

import **javafx.animation.KeyFrame**;

import **javafx.animation.KeyValue**;

import **javafx.animation.Timeline**;

import **javafx.util.Duration**;

import **javafx.scene.layout.StackPane**;

**public** **class** AnimationExample {

**public** **void** animateStackPaneHeight(**StackPane** stackPane) {

// Create a Timeline object

**Timeline** timeline = new Timeline();

// Define the end value for the height property

**KeyValue** keyValue = new KeyValue(stackPane.prefHeightProperty(), 50);

// Create a KeyFrame at 2 seconds with the end value

**KeyFrame** keyFrame = new KeyFrame(Duration.seconds(2), keyValue);

// Add the KeyFrame to the Timeline

timeline.getKeyFrames().add(keyFrame);

// Play the animation

timeline.play();

}

}

In this example:

• The Timeline object is created to manage the animation.

• A KeyValue is defined to set the prefHeightProperty of the StackPane to 50.

• A KeyFrame is created to apply the KeyValue at 2 seconds.

• The KeyFrame is added to the Timeline, and the animation is played.

Ex:

**package** application;

**import** javafx.application.Application;

**import** javafx.scene.Scene;

**import** javafx.stage.Stage;

**import** javafx.scene.control.Button;

**import** javafx.scene.control.Label;

**import** javafx.scene.layout.StackPane;

**import** javafx.scene.layout.VBox;

**import** javafx.scene.layout.HBox;

**import** javafx.scene.shape.Rectangle;

**import** javafx.animation.TranslateTransition;

**import** javafx.util.Duration;

**import** javafx.animation.KeyFrame;

**import** javafx.animation.KeyValue;

**import** javafx.animation.Timeline;

**public** **class** **Main** **extends** Application {

**@Override**

**public** **void** **start**(**Stage** primaryStage) {

**Button** **butt1** = **new** **Button**(**"Click Me!!!"**);

butt1.**setId**(**"butt1"**);

**Rectangle** **rect1** = **new** **Rectangle**(**100**, **100**);

**HBox** **hbox1** = **new** **HBox**(butt1, rect1);

butt1.**setOnAction**(**even**->{

{

// Create a Timeline object

**Timeline** **timeline** = **new** **Timeline**();

// Define the end value for the height property

**KeyValue** **keyValue** = **new** **KeyValue**(rect1.**heightProperty**(), 200);

// Create a KeyFrame at 2 seconds with the end value

**KeyFrame** **keyFrame** = **new** **KeyFrame**(**Duration**.***seconds***(**2**), keyValue);

// Add the KeyFrame to the Timeline

timeline.**getKeyFrames**().**add**(keyFrame);

// Play the animation

timeline.**play**();

} });

**Scene** **scene** = **new** **Scene**(hbox1,**400**,**400**);

scene.**getStylesheets**().**add**(**getClass**().**getResource**(**"application.css"**).**toExternalForm**());

primaryStage.**setTitle**(**"Program 1"**);

primaryStage.**setScene**(scene);

primaryStage.**show**();

}

**public** **static** **void** **main**(**String**[] args) {

***launch***(args);

}

}

Syntax in **application.css**

/\* JavaFX CSS - Leave this comment until you have at least create one rule which uses -fx-Property \*/

#butt1 {

-fx-font-size:20px;

-fx-text-fill:blue;

-fx-padding:10px;

}

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

1. Using **PauseTransition with Duration**

You can use PauseTransition to delay the visibility change

Ex:

PauseTransition **pauseTransition** = **new** PauseTransition(Duration.seconds(**2**));

pauseTransition.setOnFinished(**event** -> {

topNavigationBar1\_HBox1\_Dropdown\_Menu\_Tab\_HBox1.setVisible(**true**);

topNavigationBar1\_HBox1\_Dropdown\_Menu\_Tab\_HBox2.setVisible(**true**);

});

pauseTransition.play();

This code creates a PauseTransition that waits for 2 seconds before setting the visibility of the elements to true.