代码

canvas

```java
package top.mqxu.jfx.basic.control;

import javafx.application.Application;
import javafx.scene.Group;
import javafx.scene.Scene;
import javafx.scene.canvas.Canvas;
import javafx.scene.canvas.GraphicsContext;
import javafx.scene.paint.Color;
import javafx.scene.shape.ArcType;
import javafx.stage.Stage;

/**
 * @description: javafx canvas
 * @author: mqxu
 * @date: 2021/10/29
 */
public class CanvasApp extends Application {


    @Override
    public void start(Stage stage) {
        stage.setTitle("Canvas");
        Group root = new Group();
        Canvas canvas = new Canvas(500, 300);
        GraphicsContext gc = canvas.getGraphicsContext2D();
        drawShapes(gc);
        root.getChildren().add(canvas);
        stage.setScene(new Scene(root));
        stage.show();
    }

    private void drawShapes(GraphicsContext gc) {
        gc.setFill(Color.GREEN);
        gc.setStroke(Color.BLUE);
        gc.setLineWidth(5);
        gc.strokeLine(40, 10, 10, 40);
        gc.fillOval(10, 60, 30, 30);
        gc.strokeOval(60, 60, 30, 30);
        gc.fillRoundRect(110, 60, 30, 30, 10, 10);
        gc.strokeRoundRect(160, 60, 30, 30, 10, 10);
        gc.fillArc(10, 110, 30, 30, 45, 240, ArcType.OPEN);
        gc.fillArc(60, 110, 30, 30, 45, 240, ArcType.CHORD);
        gc.fillArc(110, 110, 30, 30, 45, 240, ArcType.ROUND);
        gc.strokeArc(10, 160, 30, 30, 45, 240, ArcType.OPEN);
        gc.strokeArc(60, 160, 30, 30, 45, 240, ArcType.CHORD);
        gc.strokeArc(110, 160, 30, 30, 45, 240, ArcType.ROUND);
        gc.fillPolygon(new double[]{10, 40, 10, 40},
                new double[]{210, 210, 240, 240}, 4);
        gc.strokePolygon(new double[]{60, 90, 60, 90},
                new double[]{210, 210, 240, 240}, 4);
        gc.strokePolyline(new double[]{110, 140, 110, 140},
                new double[]{210, 210, 240, 240}, 4);
    }

    public static void main(String[] args) {
        Application.launch(args);
    }
}
```

progressbar

```
package top.mqxu.jfx.basic.control;

import javafx.application.Application;
import javafx.beans.value.ObservableValue;
import javafx.concurrent.Task;
import javafx.event.ActionEvent;
import javafx.geometry.Insets;
import javafx.geometry.Pos;
import javafx.scene.Scene;
import javafx.scene.control.*;
import javafx.scene.layout.BorderPane;
```

```java
import javafx.scene.layout.ColumnConstraints;
import javafx.scene.layout.FlowPane;
import javafx.scene.layout.GridPane;
import javafx.stage.Stage;

/**
 * @description:
 * @author: mqxu
 * @date: 2021/10/29
 */
public class ProgressbarApp extends Application {

    private Task createWorker(final int numFiles) {
        return new Task() {
            @Override
            protected Object call() throws Exception {
                for (int i = 0; i < numFiles; i++) {
                    long elapsedTime = System.currentTimeMillis();

                    Thread.sleep(1 * 1000);

                    elapsedTime = System.currentTimeMillis() - elapsedTime;
                    String status = elapsedTime + " milliseconds";

                    updateMessage(status);
                    updateProgress(i + 1, numFiles);
                }
                return true;
            }
        };
    }

    private Task copyWorker;
    private int numFiles = 20;

    @Override
    public void start(Stage primaryStage) {
        final Label label = new Label("Progress:");
        final ProgressBar progressBar = new ProgressBar(0);
        final ProgressIndicator progressIndicator = new
ProgressIndicator(0);

        final Button startButton = new Button("Start");
        final Button cancelButton = new Button("Cancel");
        final TextArea textArea = new TextArea();

        startButton.setOnAction((ActionEvent event) -> {
            startButton.setDisable(true);

            progressBar.setProgress(0);
            progressIndicator.setProgress(0);

            textArea.setText("");
            cancelButton.setDisable(false);

            copyWorker = createWorker(numFiles);

            progressBar.progressProperty().unbind();
 
 progressBar.progressProperty().bind(copyWorker.progressProperty());
            progressIndicator.progressProperty().unbind();
 
 progressIndicator.progressProperty().bind(copyWorker.progressProperty());

            copyWorker.messageProperty().addListener(
                    (ObservableValue<? extends String> observable, String 
oldValue,
                     String newValue) -> {
                        textArea.appendText(newValue + "\n");
                    });

            new Thread(copyWorker).start();
        });

        cancelButton.setOnAction((ActionEvent event) -> {
            startButton.setDisable(false);
            cancelButton.setDisable(true);
            copyWorker.cancel(true);

            progressBar.progressProperty().unbind();
            progressBar.setProgress(0);
            progressIndicator.progressProperty().unbind();
            progressIndicator.setProgress(0);
            textArea.appendText("File transfer was cancelled.");
        });
        BorderPane root = new BorderPane();
        Scene scene = new Scene(root, 500, 250,
javafx.scene.paint.Color.WHITE);

        FlowPane topPane = new FlowPane(5, 5);
        topPane.setPadding(new Insets(5));
        topPane.setAlignment(Pos.CENTER);
        topPane.getChildren().addAll(label, progressBar,
progressIndicator);

        GridPane middlePane = new GridPane();
        middlePane.setPadding(new Insets(5));
        middlePane.setHgap(20);
        middlePane.setVgap(20);
        ColumnConstraints column1 = new ColumnConstraints(300, 400,
                Double.MAX_VALUE);
        middlePane.getColumnConstraints().addAll(column1);
        middlePane.setAlignment(Pos.CENTER);
        middlePane.add(textArea, 0, 0);

        FlowPane bottomPane = new FlowPane(5, 5);
        bottomPane.setPadding(new Insets(5));
        bottomPane.setAlignment(Pos.CENTER);
        bottomPane.getChildren().addAll(startButton, cancelButton);

        root.setTop(topPane);
        root.setCenter(middlePane);
        root.setBottom(bottomPane);

        primaryStage.setScene(scene);
        primaryStage.show();
    }

    public static void main(String[] args) {
        launch(args);
    }
}
```

HTMLEditor

```
package top.mqxu.jfx.basic.control;
```

```java
import javafx.application.Application;
import javafx.scene.Scene;
import javafx.scene.layout.TilePane;
import javafx.scene.web.HTMLEditor;
import javafx.stage.Stage;

/**
 * @description: HtmlEdit
 * @author: mqxu
 * @date: 2021/10/29
 */
public class HtmlEdit extends Application {

    @Override
    public void start(Stage stage) {

        // set title for the stage
        stage.setTitle("HTMLEditor");

        // HTML text
        String text = "<html><body><h1>学习之路</h1></body></html>";

        // create a tile pane
        TilePane tilepane = new TilePane();

        // HTML editor
        HTMLEditor htmleditor = new HTMLEditor();

        // set html text
        htmleditor.setHtmlText(text);

        // add html editor
        tilepane.getChildren().add(htmleditor);

        // create a scene
        Scene scene = new Scene(tilepane, 600, 500);

        // set the scene
        stage.setScene(scene);

        stage.show();
    }
    public static void main(String[] args) {
        launch(args);
    }
}
```

HTMLEditor

加依赖

```
<dependency>
  <groupId>org.openjfx</groupId>
  <artifactId>javafx-web</artifactId>
  <version>17.0.0.1</version>
</dependency>
```

package-info.java引入

```
 requires javafx.web;
```

代码

```
package top.mqxu.jfx.basic.control;

import javafx.application.Application;
import javafx.scene.Scene;
import javafx.scene.layout.TilePane;
import javafx.scene.web.HTMLEditor;
import javafx.stage.Stage;

/**
 * @description: HtmlEditor
 * @author: mqxu
 * @date: 2021/10/29
 */
public class HtmlEditor extends Application {

    @Override
```

```java
    public void start(Stage stage) {

        // set title for the stage
        stage.setTitle("HTMLEditor");

        // HTML text
        String text = "<html><body><h1>学习之路</h1></body></html>";

        // create a tile pane
        TilePane tilepane = new TilePane();

        // HTML editor
        HTMLEditor htmleditor = new HTMLEditor();

        // set html text
        htmleditor.setHtmlText(text);

        // add html editor
        tilepane.getChildren().add(htmleditor);

        // create a scene
        Scene scene = new Scene(tilepane, 600, 500);

        // set the scene
        stage.setScene(scene);

        stage.show();
    }

    public static void main(String[] args) {
        launch(args);
    }
}
```