JavaFXBrowser Project Source Code

1. FXMLDocument.fxml

```
<?xml version="1.0" encoding="UTF-8"?>
<?import com.jfoenix.controls.JFXButton?>
<?import com.jfoenix.controls.JFXTextField?>
<?import javafx.geometry.Insets?>
<?import javafx.scene.control.ProgressBar?>
<?import javafx.scene.control.Tooltip?>
<?import javafx.scene.layout.BorderPane?>
<?import javafx.scene.layout.HBox?>
<?import javafx.scene.layout.VBox?>
<?import javafx.scene.web.WebView?>
<?import org.kordamp.ikonli.javafx.FontIcon?>
<BorderPane prefHeight="570.0" prefWidth="883.0"
xmlns="http://javafx.com/javafx/24.0.1" xmlns:fx="http://javafx.com/fxml/1"
fx:controller="javafxbrowser.FXMLDocumentController">
    <center>
        <VBox fillWidth="true" spacing="12">
            <padding>
                <Insets bottom="12" left="12" right="12" top="12" />
            </padding>
            <children>
                <!-- Search Row -->
                <HBox alignment="CENTER_LEFT" spacing="10.0">
                    <children>
                        <JFXTextField fx:id="textField" focusColor="#3967fc"
focusTraversable="false" onAction="#loadPage" promptText="Search"
styleClass="textfield" HBox.hgrow="ALWAYS" />

                        <JFXButton fx:id="settingBtn" focusTraversable="false"
onAction="#openSettings" prefHeight="40.0" prefWidth="50.0" ripplerFill="#a6a6a6"
styleClass="settingBtn" text="">
                            <graphic>
                                <FontIcon iconLiteral="fa-gear" iconSize="22"
styleClass="icon" />
                            </graphic>
                        </JFXButton>
                    </children>
                </HBox>
                <!-- Buttons Row -->
                <HBox alignment="CENTER_LEFT" spacing="10.0">
                    <children>
                        <JFXButton onAction="#loadPage" prefHeight="40.0"
prefWidth="50.0" styleClass="btn">
                            <graphic><FontIcon iconLiteral="fa-search"
iconSize="22" styleClass="icon" /></graphic>
                            <tooltip><Tooltip text="Search" /></tooltip>
                        </JFXButton>
                        <JFXButton onAction="#refreshPage" prefHeight="40.0"
prefWidth="50.0" styleClass="btn">
                            <graphic><FontIcon iconLiteral="fa-refresh"
iconSize="22" styleClass="icon" /></graphic>
                            <tooltip><Tooltip text="Refresh" /></tooltip>
                        </JFXButton>
                        <JFXButton onAction="#zoomIn" prefHeight="40.0"
prefWidth="50.0" styleClass="btn">
                            <graphic><FontIcon iconLiteral="fa-search-plus"
iconSize="22" styleClass="icon" /></graphic>
                            <tooltip><Tooltip text="Zoom In" /></tooltip>
                        </JFXButton>
                        <JFXButton onAction="#zoomOut" prefHeight="40.0"
prefWidth="50.0" styleClass="btn">
                            <graphic><FontIcon iconLiteral="fa-search-minus"
iconSize="22" styleClass="icon" /></graphic>
                            <tooltip><Tooltip text="Zoom Out" /></tooltip>
                        </JFXButton>
                        <JFXButton onAction="#displayHistory" prefHeight="40.0"
prefWidth="50.0" styleClass="btn">
                            <graphic><FontIcon iconLiteral="fa-history"
iconSize="22" styleClass="icon" /></graphic>
                            <tooltip><Tooltip text="History" /></tooltip>
                        </JFXButton>
                        <JFXButton onAction="#back" prefHeight="40.0"
prefWidth="50.0" styleClass="btn">
                            <graphic><FontIcon iconLiteral="fa-arrow-circle-o-
left" iconSize="22" styleClass="icon" /></graphic>
                            <tooltip><Tooltip text="Backward" /></tooltip>
                        </JFXButton>
                        <JFXButton onAction="#forward" prefHeight="40.0"
prefWidth="50.0" styleClass="btn">
                            <graphic><FontIcon iconLiteral="fa-arrow-circle-o-
right" iconSize="22" styleClass="icon" /></graphic>
                            <tooltip><Tooltip text="Forward" /></tooltip>
                        </JFXButton>
                    </children>
                </HBox>
                <!-- Progress Bar -->
                <ProgressBar fx:id="progressBar" maxWidth="Infinity"
prefHeight="4.0" styleClass="progressBar" visible="false" />

                <!-- WebView -->
                <WebView fx:id="webView" VBox.vgrow="ALWAYS" />
            </children>
        </VBox>
    </center>
</BorderPane>
```

2. FXMLDocumentController.java

```java
package javafxbrowser;

import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import java.net.URL;
import java.time.LocalDateTime;
import java.util.ArrayList;
import java.util.List;
import java.util.ResourceBundle;
import javafx.application.Platform;
import javafx.beans.property.SimpleStringProperty;
import javafx.collections.ObservableList;
import javafx.fxml.FXML;
import javafx.fxml.FXMLLoader;
import javafx.fxml.Initializable;
import javafx.geometry.Insets;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.control.ProgressBar;
import javafx.scene.control.TableColumn;
import javafx.scene.control.TableRow;
import javafx.scene.control.TableView;
import javafx.scene.control.TextField;
import javafx.scene.layout.HBox;
import javafx.scene.layout.VBox;
import javafx.scene.web.WebEngine;
import javafx.scene.web.WebHistory;
import javafx.scene.web.WebView;
import javafx.stage.Modality;
import javafx.stage.Stage;

public class FXMLDocumentController implements Initializable {

    @FXML
    private WebView webView;

    @FXML
    private TextField textField;

    @FXML
    private ProgressBar progressBar;

    private WebEngine engine;
    private WebHistory history;

    private String homePage;

    private double zoom;
    private static final String HISTORY_FILE = "history.txt";

    public static class HistoryEntry {
        public String url;
        public String visitedDate;

        public HistoryEntry(String url, String visitedDate) {
            this.url = url;
            this.visitedDate = visitedDate;
        }
    }

    @Override
    public void initialize(URL url, ResourceBundle rb) {
        engine = webView.getEngine();
        zoom = 1;
        homePage = "www.google.com";
        textField.setText(homePage);
        setupLoadingIndicator();
        loadPage();
        Platform.runLater(() -> {
            if (textField.getScene() != null) {
                textField.getScene().getStylesheets().add(
                        getClass().getResource("light.css").toExternalForm()
                );
            }
        });
    }

    public void loadPage() {
        String url = textField.getText().trim();
        if (!url.startsWith("http://") && !url.startsWith("https://")) {
            url = "http://" + url;
        }
        engine.load(url);
        saveHistoryEntry(url, LocalDateTime.now().toString());
    }

    public void refreshPage() {
        engine.reload();
    }

    public void zoomIn() {
        zoom += 0.25;
        webView.setZoom(zoom);
    }

    public void zoomOut() {
        zoom -= 0.25;
        webView.setZoom(zoom);
    }

    private void saveHistoryEntry(String url, String timestamp) {
        try (BufferedWriter writer = new BufferedWriter(new
FileWriter(HISTORY_FILE, true))) {
            writer.write(url + "|" + timestamp);
            writer.newLine();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    private List<HistoryEntry> loadHistoryFromFile() {
        List<HistoryEntry> historyList = new ArrayList<>();
        try (BufferedReader reader = new BufferedReader(new
FileReader(HISTORY_FILE))) {
            String line;
            while ((line = reader.readLine()) != null) {
                String[] parts = line.split("\\|");
                if (parts.length == 2) {
                    historyList.add(new HistoryEntry(parts[0], parts[1]));
                }
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
        return historyList;
    }

    private void saveAllHistoryToFile(List<HistoryEntry> historyList) {
        try (BufferedWriter writer = new BufferedWriter(new
FileWriter(HISTORY_FILE, false))) {
            for (HistoryEntry entry : historyList) {
                writer.write(entry.url + "|" + entry.visitedDate);
                writer.newLine();
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    public void displayHistory() {
        List<HistoryEntry> savedHistory = loadHistoryFromFile();

        Stage historyStage = new Stage();
        historyStage.setTitle("Browsing History");
        historyStage.initModality(Modality.APPLICATION_MODAL);
        historyStage.setWidth(600);
        historyStage.setHeight(400);

        TableView<HistoryEntry> table = new TableView<>();

        TableColumn<HistoryEntry, String> urlCol = new TableColumn<>("URL");
        urlCol.setCellValueFactory(data -> new
SimpleStringProperty(data.getValue().url));
        urlCol.setPrefWidth(400);
        TableColumn<HistoryEntry, String> dateCol = new TableColumn<>("Last
Visited");
        dateCol.setCellValueFactory(data -> new
SimpleStringProperty(data.getValue().visitedDate));
        dateCol.setPrefWidth(180);

        table.getColumns().addAll(urlCol, dateCol);
        table.setColumnResizePolicy(TableView.CONSTRAINED_RESIZE_POLICY);

        table.getItems().setAll(savedHistory);

        table.setRowFactory(tv -> {
            TableRow<HistoryEntry> row = new TableRow<>();
            row.setOnMouseClicked(event -> {
                if (event.getClickCount() == 2 && !row.isEmpty()) {
                    HistoryEntry entry = row.getItem();
                    if (entry != null) {
                        textField.setText(entry.url);
                        loadPage();
                        historyStage.close();
                    }
                }
            });
            return row;
        });

        HBox buttonRow = new HBox(10);
        Button deleteBtn = new Button("Delete Selected");
        Button clearBtn = new Button("Clear All");

        deleteBtn.setOnAction(e -> {
            HistoryEntry selected = table.getSelectionModel().getSelectedItem();
            if (selected != null) {
                table.getItems().remove(selected);
                saveAllHistoryToFile(new ArrayList<>(table.getItems()));
            }
        });

        clearBtn.setOnAction(e -> {
            table.getItems().clear();
            saveAllHistoryToFile(new ArrayList<>());
        });

        buttonRow.getChildren().addAll(deleteBtn, clearBtn);

        VBox layout = new VBox(10, table, buttonRow);
        layout.setPadding(new Insets(10));
        Scene scene = new Scene(layout);
        historyStage.setScene(scene);
        historyStage.showAndWait();
    }

    public void back() {
        history = engine.getHistory();
        ObservableList<WebHistory.Entry> entries = history.getEntries();
        history.go(-1);
        textField.setText(entries.get(history.getCurrentIndex()).getUrl());
    }

    public void forward() {
        history = engine.getHistory();
        ObservableList<WebHistory.Entry> entries = history.getEntries();
        history.go(1);
        textField.setText(entries.get(history.getCurrentIndex()).getUrl());
    }

    public void setupLoadingIndicator() {
        engine = webView.getEngine();
        engine.getLoadWorker().stateProperty().addListener((obs, oldState,
newState) -> {
            switch (newState) {
                case RUNNING:
                    progressBar.setVisible(true);
                    break;
                case SUCCEEDED:
                case FAILED:
                case CANCELLED:
                    progressBar.setVisible(false);
                    break;
                default:
                    break;
            }
        });

progressBar.progressProperty().bind(engine.getLoadWorker().progressProperty());
    }

    public void openSettings() {
        try {
            FXMLLoader loader = new
FXMLLoader(getClass().getResource("Settings.fxml"));
            Parent root = loader.load();
            SettingsController settingsController = loader.getController();
            settingsController.setMainScene(textField.getScene());
            Stage settingsStage = new Stage();
            settingsStage.setTitle("Settings");
            settingsStage.initModality(Modality.APPLICATION_MODAL);
            settingsStage.setScene(new Scene(root));
            settingsStage.showAndWait();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

3. JavaFXBrowser.java

```java
package javafxbrowser;

import javafx.application.Application;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.stage.Stage;

public class JavaFXBrowser extends Application {

    @Override
    public void start(Stage stage) throws Exception {
        Parent root =
FXMLLoader.load(getClass().getResource("FXMLDocument.fxml"));
        Scene scene = new Scene(root);
        stage.setScene(scene);
        stage.show();
    }

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

---

4. Settings.fxml

```xml
<?xml version="1.0" encoding="UTF-8"?>
<?import com.jfoenix.controls.JFXToggleButton?>
<?import javafx.scene.control.Label?>
<?import javafx.scene.layout.HBox?>
<?import javafx.scene.layout.VBox?>
<VBox fx:id="background" alignment="CENTER" prefHeight="120" prefWidth="300.0"
spacing="20" xmlns="http://javafx.com/javafx/24.0.1"
xmlns:fx="http://javafx.com/fxml/1"
fx:controller="javafxbrowser.SettingsController">
    <HBox alignment="CENTER" spacing="10">
        <Label fx:id="lightLabel" text="Light" textFill="#121212" />
        <JFXToggleButton fx:id="themeToggle" focusTraversable="false" text=""
toggleColor="#0497ec" toggleLineColor="#90caf9" />
        <Label fx:id="darkLabel" text="Dark" textFill="#121212" />
    </HBox>
</VBox>
```

---

5. SettingsController.java

```java
package javafxbrowser;

import com.jfoenix.controls.JFXToggleButton;
import javafx.fxml.FXML;
import javafx.scene.Scene;
import javafx.scene.control.Label;
import javafx.scene.layout.VBox;

public class SettingsController {
    @FXML
    private JFXToggleButton themeToggle;

    @FXML
    private VBox background;

    @FXML
    private Label lightLabel;

    @FXML
    private Label darkLabel;

    private Scene mainScene;

    public void setMainScene(Scene scene) {
        this.mainScene = scene;
        if (mainScene.getStylesheets().stream().anyMatch(s ->
s.contains("dark.css"))) {
            themeToggle.setSelected(true);
        } else {
            themeToggle.setSelected(false);
        }

        themeToggle.selectedProperty().addListener((obs, oldVal, isSelected) -> {
            if (mainScene != null) {
                mainScene.getStylesheets().removeIf(s -> s.contains("light.css")
|| s.contains("dark.css"));

                if (isSelected) {

mainScene.getStylesheets().add(getClass().getResource("dark.css").toExternalForm()
);
                    background.setStyle("-fx-background-color: #3D3D3D;");
                    lightLabel.setStyle("-fx-text-fill: #E8E8E8;");
                    darkLabel.setStyle("-fx-text-fill: #E8E8E8");
                } else {

mainScene.getStylesheets().add(getClass().getResource("light.css").toExternalForm(
));
                    background.setStyle("-fx-background-color: #ffffff;");
                    lightLabel.setStyle("-fx-text-fill: #121212;");
```

```
                        darkLabel.setStyle("-fx-text-fill: #121212");
                    }
                }
            });
        }
    }
```

6. light.css

```
.root {
    -fx-background-color: #FAFAFA; /* light bg */
    -fx-text-fill: #212121;
}
/* Buttons */
.btn {
    -fx-background-color: #5C6BC0;
    -fx-background-radius: 10px;
    -fx-border-radius: 10px;
    -fx-pref-width: 60px;
    -fx-pref-height: 40px;
}
.btn:hover {
    -fx-background-color: #7986CB;
    -fx-cursor: hand;
    -fx-scale-x: 1.05;
    -fx-scale-y: 1.05;
}
.btn:pressed {
    -fx-background-color: #3F51B5;
    -fx-scale-x: 0.95;
    -fx-scale-y: 0.95;
}
/* Settings button (round) */
.settingBtn {
    -fx-background-color: #E0E0E0;
    -fx-background-radius: 30px;
    -fx-border-radius: 30px;
}
/* Textfield */
.textfield {
    -fx-background-color: white;
    -fx-text-fill: #212121;
    -fx-background-radius: 6px;
    -fx-border-radius: 6px;
    -fx-prompt-text-fill: #9E9E9E;
    -fx-padding: 5 10 5 10;
}
/* Progress Bar */
.progressBar {
    -fx-background-color: transparent;
    -fx-pref-height: 4px;
}
.progressBar .track {
    -fx-background-color: transparent;
}
.progressBar .bar {
    -fx-background-color: #29B6F6;
}
/* Icons */
.icon {
    -fx-icon-color: #212121;
}
```

7. dark.css

```
.root {
    -fx-background-color: #121212; /* dark bg */
    -fx-text-fill: #E0E0E0;
}
/* Buttons */
.btn {
    -fx-background-color: #3949AB;
    -fx-background-radius: 10px;
    -fx-border-radius: 10px;
    -fx-pref-width: 60px;
    -fx-pref-height: 40px;
}
.btn:hover {
    -fx-background-color: #5C6BC0;
    -fx-cursor: hand;
    -fx-scale-x: 1.05;
    -fx-scale-y: 1.05;
}
.btn:pressed {
    -fx-background-color: #283593;
    -fx-scale-x: 0.95;
    -fx-scale-y: 0.95;
}
/* Settings button (round) */
.settingBtn {
    -fx-background-color: #1E1E1E;
    -fx-background-radius: 30px;
    -fx-border-radius: 30px;
}
/* Textfield */
.textfield {
    -fx-background-color: #1E1E1E;
    -fx-text-fill: #E0E0E0;
    -fx-background-radius: 6px;
    -fx-border-radius: 6px;
    -fx-prompt-text-fill: #9E9E9E;
    -fx-padding: 5 10 5 10;
}
/* Progress Bar */
.progressBar {
    -fx-background-color: transparent;
    -fx-pref-height: 4px;
}
.progressBar .track {
    -fx-background-color: transparent;
}
.progressBar .bar {
    -fx-background-color: #81D4FA;
}
/* Icons */
.icon {
    -fx-icon-color: #E0E0E0;
}
```

File directory structure

```
JavaFXBrowser
├── Source Packages
│   └── javafxbrowser
│       ├── FXMLDocument.fxml
│       ├── FXMLDocumentController.java
│       ├── JavaFXBrowser.java
│       ├── Settings.fxml
│       ├── SettingsController.java
│       ├── dark.css
│       ├── light.css
│       └── style.css
├── Test Packages
├── Libraries
│   ├── jfoenix-8.0.10.jar
│   ├── ikonli-javafx-2.4.0.jar
│   ├── ikonli-fontawesome-pack-2.4.0.jar
│   ├── ikonli-core-2.4.0.jar
│   └── JDK 1.8
└── Test Libraries
```