



# REPRODUCTOR DE VIDEO

Código del proyecto

Adrián Martínez Medina

18 de marzo de 2022

# Contenido

MediaPlayer.java.....	2
MediaPlayerController.java .....	3
MediaPlayer.fxml .....	12
AcercaDe.fxml .....	14
Style.css.....	16

# MediaPlayer.java

```
1 /**
2  * Clase java usada para ejecutar la app
3  *
4  * Crea la ventana, cargando todos los componentes
5  * a partir de la vista fxml que hemos creado y
6  * preparado en el scene builder
7  *
8  * @author amartinez
9  */
10 package mediaplayer;
11
12 // Listado de imports
13 import javafx.application.Application;
14 import javafx.fxml.FXMLLoader;
15 import javafx.scene.Parent;
16 import javafx.scene.Scene;
17 import javafx.scene.image.Image;
18 import javafx.scene.input.MouseEvent;
19 import javafx.stage.Stage;
20
21 public class MediaPlayer extends Application {
22
23     @Override
24     public void start(Stage stage) throws Exception {
25
26         // Creamos el parent root cargando los nodos desde el fxml
27         Parent root = FXMLLoader.load(getClass().getResource("MediaPlayer.fxml"));
28
29         /**
30          * Creamos el objeto que nos dará el contenido de la ventana a partir de
31          * los nodos pasados desde root
32          */
33         Scene scene = new Scene(root);
34
35         // A la escena le añadimos los valores descriptivos que queremos
36         stage.setTitle("Media Player hecho por Adrian");
37         stage.setFullScreenExitHint("Presione ESC or Doble Click para salir de la pantalla completa");
38
39         // Añadimos a la ventana nuestro icono personalizado
40         stage.getIcons().add(new Image("images/icon.png"));
41
42         /**
43          * Añadimos a la escena la capacidad de cambiar entre modo ventana y
44          * modo pantalla completa al hacer doble click sobre el video
45          */
46         scene.setOnMouseClicked((MouseEvent clicked) -> {
47             if (clicked.getClickCount() == 2) {
48                 if (!stage.isFullScreen()) {
49                     stage.setFullScreen(true);
50                 } else {
51                     stage.setFullScreen(false);
52                 }
53             }
54         });
55
56         /**
57          * Añadimos el contenido guardado en la escena a la ventana y la hacemos
58          * visible
59          */
60         stage.setScene(scene);
61         stage.show();
62
63     }
64
65     public static void main(String[] args) {
66
67         launch(args);
68
69     }
70
71 }
```

# MediaPlayerController.java

```
1 /**
2  * Clase java usada para las funcionalidades de la app
3  *
4  * Hace a nuestra app capaz de pausar, activar,
5  * parar, acelerar, ralentizar, avanzar y retroceder
6  *
7  * Además, también habilitamos la barra de progreso
8  * del video y la de volumen. Por último también se crean
9  * los keyevents para la funcionalidades
10 *
11 * @author amartinez
12 */
13 package mediaplayer;
14
15 // Listado de imports
16 import java.io.File;
17 import java.io.IOException;
18 import java.net.URL;
19 import java.util.Arrays;
20 import java.util.List;
21 import java.util.ResourceBundle;
22 import javafx.animation.FadeTransition;
23 import javafx.beans.Observable;
24 import javafx.beans.binding.Bindings;
25 import javafx.beans.property.DoubleProperty;
26 import javafx.beans.value.ObservableValue;
27 import javafx.event.ActionEvent;
28 import javafx.fxml.FXML;
29 import javafx.fxml.FXMLLoader;
30 import javafx.fxml.Initializable;
31 import javafx.scene.Parent;
32 import javafx.scene.Scene;
33 import javafx.scene.control.Button;
34 import javafx.scene.control.MenuBar;
35 import javafx.scene.control.Slider;
36 import javafx.scene.image.Image;
37 import javafx.scene.input.KeyCode;
38 import javafx.scene.input.KeyEvent;
39 import javafx.scene.input.MouseEvent;
40 import javafx.scene.media.Media;
41 import javafx.scene.media.MediaPlayer;
42 import javafx.scene.media.MediaView;
43 import static javafx.scene.media.MediaPlayer.Status.PLAYING;
44 import javafx.stage.FileChooser;
45 import javafx.stage.Stage;
46 import javafx.util.Duration;
47 import javax.swing.JOptionPane;
48
49 public class MediaPlayerController implements Initializable {
50
51     // Variable que almacena la ruta al video abierto
52     @FXML
53     private String path;
54
55     /**
56      * Variable que almacena todas las extensiones de archivo válidas para
57      * nuestra aplicación en un array
58      */
59     @FXML
60     private List<String> extensions;
61
62     // Variable objeto, crea el mediaPlayer
63     @FXML
64     private MediaPlayer mediaPlayer;
65
66     // Variable objeto, crea el mediaView
67     @FXML
68     private MediaView mediaView;
69
70     // Variable objeto, crea el slider de volumen
71     @FXML
72     private Slider volumeSlider;
73
74     // Variable objeto, crea el slider de progreso de video
75     @FXML
76     private Slider progressBar;
77
78     // Botones de la aplicación
```

```

79  @FXML
80  private Button openFile, playVideo, stopVideo, slowVideo,
81      backVideo5, normalVideo, forwardVideo5, fastVideo, setVisibilityButtons;
82
83  // Menu bar de la aplicación
84  @FXML
85  private MenuBar menuBar;
86
87  // Booleana que usaremos para mostrar y ocultar los botones
88  @FXML
89  private boolean visible = true;
90
91  /**
92   * Método que usamos para añadir al boton de abrir archivo la capacidad de
93   * hacerlo.
94   *
95   * @param event
96   */
97  @FXML
98  private void OpenFileMethod(ActionEvent event) {
99
100      try {
101          /**
102           * llenamos el array de strings con las extensiones permitidas en
103           * nuestra app
104           */
105          extensions = Arrays.asList("*.mp4", "*.3gp", "*.mkv", "*.MP4", "*.MKV", "*.3GP", "*.flv", "*.wmv");
106
107          /**
108           * Creamos tanto el objeto filechooser para elegir archivo como el
109           * objeto extension filter para añadir el filtro de extensiones que
110           * hemos concretado en el array anterior
111           */
112          FileChooser fileChooser = new FileChooser();
113          FileChooser.ExtensionFilter filter = new FileChooser.ExtensionFilter("selecciona video", extensions);
114
115          // Añadimos el filtro de extensiones al filechooser
116          fileChooser.getExtensionFilters().add(filter);
117          fileChooser.setTitle("Selecciona un video");
118
119          /**
120           * Abrimos la ventana para elegir archivo y este se guarda como
121           * variable File
122           *
123           * Posteriormente guardamos en un string la ruta del archivo elegido
124           * para ser usada en el objeto Media
125           */
126          File file = fileChooser.showOpenDialog(null);
127          path = file.toURI().toString();
128
129          /**
130           * Siempre y cuando se haya elegido un archivo, se realizará el
131           * código contenido en el if
132           */
133          if (path != null) {
134              /**
135               * Creamos el objeto Media gracias al path, luego lo usamos para
136               * crear el MediaPlayer y finalmente creamos la vista del video
137               */
138              Media media = new Media(path);
139              mediaPlayer = new MediaPlayer(media);
140              mediaView.setMediaPlayer(mediaPlayer);
141
142              /**
143               * Las líneas que tenemos a continuación sirven para definir las
144               * dimensiones del video cargado ajustandolo al mediaView
145               */
146              DoubleProperty widthProp = mediaView.fitWidthProperty();
147              DoubleProperty heightProp = mediaView.fitHeightProperty();
148              widthProp.bind(Bindings.selectDouble(mediaView.sceneProperty(), "width"));
149              heightProp.bind(Bindings.selectDouble(mediaView.sceneProperty(), "height"));
150
151              // Damos un valor inicial al slider del volumen del video
152              volumeSlider.setValue(mediaPlayer.getVolume() * 100);
153
154              /**
155               * Creamos el listener que pertimirá ajustar el volumen del
156               * video según donde coloquemos el punto del slider del volumen
157               */

```

```

158         volumeSlider.valueProperty().addListener((Observable observable) -> {
159             mediaPlayer.setVolume(volumeSlider.getValue() / 100);
160         });
161
162         // Damos un valor inicial a la barra de progreso del video
163         mediaPlayer.currentTimeProperty().addListener((ObservableValue<? extends javafx.util.Duration> observable,
164             javafx.util.Duration oldValue, javafx.util.Duration newValue) -> {
165             progressBar.setValue(newValue.toSeconds());
166         });
167
168         /**
169          * Creamos los listener que permitirán ajustar el volumen del
170          * video según donde coloquemos el punto del slider del volumen
171          */
172         progressBar.setOnMousePressed((MouseEvent event1) -> {
173             mediaPlayer.seek(javafx.util.Duration.seconds(progressBar.getValue()));
174         });
175         progressBar.setOnMouseDragged((MouseEvent event1) -> {
176             mediaPlayer.seek(javafx.util.Duration.seconds(progressBar.getValue()));
177         });
178         mediaPlayer.setOnReady(() -> {
179             javafx.util.Duration total = media.getDuration();
180             progressBar.setMax(total.toSeconds());
181         });
182
183         // Por último, iniciamos la reproducción del video al abrirlo
184         mediaPlayer.play();
185     } catch (Exception e) {
186     }
187 }
188
189 @Override
190 public void initialize(URL url, ResourceBundle rb) {
191     // TODO
192 }
193
194 /**
195  * Método que usamos para añadir al botón de play & pause video la capacidad
196  * de hacerlo.
197  *
198  * @param event
199  */
200 @FXML
201 private void playVideo(ActionEvent event) {
202
203     /**
204      * Siempre y cuando se haya elegido previamente un video, se comprueba
205      * si el video está activo o pausado
206      *
207      * Si está activo, se pausa y si está pausado se reactiva
208      */
209     try {
210         if (mediaPlayer.getStatus() == PLAYING) {
211             mediaPlayer.pause();
212         } else {
213             mediaPlayer.play();
214         }
215     } catch (Exception e) {
216         JOptionPane.showMessageDialog(null, "Elige un video primero");
217     }
218 }
219
220 }
221
222 /**
223  * Método que usamos para añadir al botón de stop video la capacidad de
224  * hacerlo.
225  *
226  * @param event
227  */
228 @FXML
229 private void stopVideo(ActionEvent event) {
230
231     /**
232      * Siempre y cuando se haya elegido previamente un video, se detiene el
233      * video reseteándolo
234      */
235     try {

```

```

236         mediaPlayer.stop();
237     } catch (Exception e) {
238         JOptionPane.showMessageDialog(null, "Elige un video primero");
239     }
240 }
241
242
243 /**
244  * Método que usamos para añadir al boton de avanzar video la capacidad de
245  * hacerlo.
246  *
247  * @param event
248  */
249 @FXML
250 private void skip5(ActionEvent event) {
251
252     /**
253      * Siempre y cuando se haya elegido previamente un video, se adelanta el
254      * video 5 segundos
255      */
256     try {
257         mediaPlayer.seek(mediaPlayer.getCurrentTime().add(javaafx.util.Duration.seconds(5)));
258     } catch (Exception e) {
259         JOptionPane.showMessageDialog(null, "Elige un video primero");
260     }
261 }
262
263
264 /**
265  * Método que usamos para añadir al boton de acelerar video la capacidad de
266  * hacerlo.
267  *
268  * @param event
269  */
270 @FXML
271 private void furtherSpeedUpVideo(ActionEvent event) {
272
273     /**
274      * Siempre y cuando se haya elegido previamente un video, se acelera el
275      * video
276      */
277     try {
278         mediaPlayer.setRate(2);
279     } catch (Exception e) {
280         JOptionPane.showMessageDialog(null, "Elige un video primero");
281     }
282 }
283
284
285 /**
286  * Método que usamos para añadir al boton de atrasar video la capacidad de
287  * hacerlo.
288  *
289  * @param event
290  */
291 @FXML
292 private void back5(ActionEvent event) {
293
294     /**
295      * Siempre y cuando se haya elegido previamente un video, se atrasa el
296      * video 5 segundos
297      */
298     try {
299         mediaPlayer.seek(mediaPlayer.getCurrentTime().add(javaafx.util.Duration.seconds(-5)));
300     } catch (Exception e) {
301         JOptionPane.showMessageDialog(null, "Elige un video primero");
302     }
303 }
304
305
306 /**
307  * Método que usamos para añadir al boton de ralentizar video la capacidad
308  * de hacerlo.
309  *
310  * @param event
311  */
312 @FXML
313 private void furtherSlowDownVideo(ActionEvent event) {

```

```

314
315     /**
316     * Siempre y cuando se haya elegido previamente un video, se ralentiza
317     * el video
318     */
319     try {
320         mediaPlayer.setRate(0.5);
321     } catch (Exception e) {
322         JOptionPane.showMessageDialog(null, "Elige un video primero");
323     }
324 }
325
326
327 /**
328 * Método que usamos para añadir al boton de normalizar velocidad video la
329 * capacidad de hacerlo.
330 *
331 * @param event
332 */
333 @FXML
334 private void normalSpeedVideo(ActionEvent event) {
335
336     /**
337     * Siempre y cuando se haya elegido previamente un video, se resetea la
338     * velocidad de reproducción del video
339     */
340     try {
341         mediaPlayer.setRate(1);
342     } catch (Exception e) {
343         JOptionPane.showMessageDialog(null, "Elige un video primero");
344     }
345 }
346
347
348 /**
349 * Método que usamos para abrir la ventana Acerca De al hacer click en el
350 * menú de la app
351 *
352 * @param event
353 */
354 @FXML
355 private void openAcercaDe(ActionEvent event) {
356     try {
357         // Creamos el parent root cargando los nodos desde el fxml
358         Parent root = FXMLLoader.load(getClass().getResource("AcercaDe.fxml"));
359
360         /**
361         * Creamos el objeto que nos dará el contenido de la ventana a
362         * partir de los nodos pasados desde root
363         */
364         Scene scene = new Scene(root);
365
366         // Creamos la ventana acerca de
367         Stage acercaDe = new Stage();
368
369         // Añadimos a la ventana nuestro icono personalizado
370         acercaDe.getIcons().add(new Image("images/icon.png"));
371
372         /**
373         * Añadimos el contenido guardado en la escena a la ventana y la
374         * hacemos visible
375         */
376         acercaDe.setScene(scene);
377         acercaDe.show();
378     } catch (IOException e) {
379     }
380 }
381
382 /**
383 * Método que usamos para mostrar y ocultar todos los controles con el fin
384 * de que el video sea perfectamente visible sin molestias a gusto del
385 * usuario
386 *
387 * @param event
388 */
389 @FXML
390 private void hideShowControls(ActionEvent event) {
391

```



```

392 try {
393     /**
394      * Objetos transiciones que nos permiten hacer visibles
395      * e invisibles posteriormente a los controles de nuestra
396      * aplicación cambiando su visibilidad
397      */
398     FadeTransition fadeTransitionVisibility = new FadeTransition(Duration.millis(200), setVisibilityButtons);
399     FadeTransition fadeTransitionOpenFile = new FadeTransition(Duration.millis(200), openFile);
400     FadeTransition fadeTransitionPlayVideo = new FadeTransition(Duration.millis(200), playVideo);
401     FadeTransition fadeTransitionStopVideo = new FadeTransition(Duration.millis(200), stopVideo);
402     FadeTransition fadeTransitionSlowVideo = new FadeTransition(Duration.millis(200), slowVideo);
403     FadeTransition fadeTransitionBackVideo5 = new FadeTransition(Duration.millis(200), backVideo5);
404     FadeTransition fadeTransitionNormalVideo = new FadeTransition(Duration.millis(200), normalVideo);
405     FadeTransition fadeTransitionForwardVideo5 = new FadeTransition(Duration.millis(200), forwardVideo5);
406     FadeTransition fadeTransitionFastVideo = new FadeTransition(Duration.millis(200), fastVideo);
407     FadeTransition fadeTransitionVolumeSlider = new FadeTransition(Duration.millis(200), volumeSlider);
408     FadeTransition fadeTransitionProgressBar = new FadeTransition(Duration.millis(200), progressBar);
409     FadeTransition fadeTransitionMenuBar = new FadeTransition(Duration.millis(200), menuBar);
410
411     if (visible) {
412         // Ponemos la visibilidad de los controles a 0
413         fadeTransitionVisibility.setToValue(0.2);
414         fadeTransitionOpenFile.setToValue(0);
415         fadeTransitionPlayVideo.setToValue(0);
416         fadeTransitionStopVideo.setToValue(0);
417         fadeTransitionSlowVideo.setToValue(0);
418         fadeTransitionBackVideo5.setToValue(0);
419         fadeTransitionNormalVideo.setToValue(0);
420         fadeTransitionForwardVideo5.setToValue(0);
421         fadeTransitionFastVideo.setToValue(0);
422         fadeTransitionVolumeSlider.setToValue(0);
423         fadeTransitionProgressBar.setToValue(0);
424         fadeTransitionMenuBar.setToValue(0);
425
426         // Ejecutamos la transición
427         fadeTransitionVisibility.play();
428         fadeTransitionOpenFile.play();
429         fadeTransitionPlayVideo.play();
430         fadeTransitionStopVideo.play();
431         fadeTransitionSlowVideo.play();
432         fadeTransitionBackVideo5.play();
433         fadeTransitionNormalVideo.play();
434         fadeTransitionForwardVideo5.play();
435         fadeTransitionFastVideo.play();
436         fadeTransitionVolumeSlider.play();
437         fadeTransitionProgressBar.play();
438         fadeTransitionMenuBar.play();
439
440         /**
441          * al completarse la última transición cambiamos el
442          * valor de la booleana visible para que al volver
443          * a hacer clic se lleve a cabo el efecto contrario
444          * a hacer clic se lleve a cabo el efecto contrario
445          */
446         fadeTransitionMenuBar.setOnFinished(e -> {
447             visible = false;
448         });
449     }
450     if (!visible) {
451         // Ponemos la visibilidad de los controles a 1
452         fadeTransitionVisibility.setToValue(1);
453         fadeTransitionOpenFile.setToValue(1);
454         fadeTransitionPlayVideo.setToValue(1);
455         fadeTransitionStopVideo.setToValue(1);
456         fadeTransitionSlowVideo.setToValue(1);
457         fadeTransitionBackVideo5.setToValue(1);
458         fadeTransitionNormalVideo.setToValue(1);
459         fadeTransitionForwardVideo5.setToValue(1);
460         fadeTransitionFastVideo.setToValue(1);
461         fadeTransitionVolumeSlider.setToValue(1);
462         fadeTransitionProgressBar.setToValue(1);
463         fadeTransitionMenuBar.setToValue(1);
464
465         // Ejecutamos la transición
466         fadeTransitionVisibility.play();
467         fadeTransitionOpenFile.play();
468         fadeTransitionPlayVideo.play();
469         fadeTransitionStopVideo.play();
470         fadeTransitionSlowVideo.play();
471         fadeTransitionBackVideo5.play();
472         fadeTransitionNormalVideo.play();
473         fadeTransitionForwardVideo5.play();
474         fadeTransitionFastVideo.play();
475         fadeTransitionVolumeSlider.play();
476         fadeTransitionProgressBar.play();
477         fadeTransitionMenuBar.play();

```

```

478         /**
479         * al completarse la última transición cambiamos el
480         * valor de la booleana visible para que al volver
481         * a hacer clic se lleve a cabo el efecto contrario
482         */
483         fadeTransitionMenuBar.setOnFinished(e -> {
484             visible = true;
485         });
486     }
487 } catch (Exception e) {
488 }
489
490 }
491
492 /**
493  * Método que contiene todos los keylisteners de la app, haciendo posible la
494  * ejecución de las funciones descritas anteriormente segun la combinación
495  * de teclas pulsada.
496  *
497  * @param ke
498  */
499 @FXML
500 private void handleKeyPressed(KeyEvent ke) {
501
502     if (ke.getCode().equals(KeyCode.P) && ke.isControlDown()) {
503         try {
504             if (mediaPlayer.getStatus() == PLAYING) {
505                 mediaPlayer.pause();
506             } else {
507                 mediaPlayer.play();
508             }
509         } catch (Exception e) {
510             JOptionPane.showMessageDialog(null, "Elige un video primero");
511         }
512     }
513
514     if (ke.getCode().equals(KeyCode.S) && ke.isControlDown()) {
515         try {
516             mediaPlayer.stop();
517         } catch (Exception e) {
518             JOptionPane.showMessageDialog(null, "Elige un video primero");
519         }
520     }
521
522     if (ke.getCode().equals(KeyCode.RIGHT) && ke.isControlDown()) {
523         try {
524             mediaPlayer.seek(mediaPlayer.getCurrentTime().add(javaafx.util.Duration.seconds(5)));
525         } catch (Exception e) {
526             JOptionPane.showMessageDialog(null, "Elige un video primero");
527         }
528     }
529
530     if (ke.getCode().equals(KeyCode.LEFT) && ke.isControlDown()) {
531         try {
532             mediaPlayer.seek(mediaPlayer.getCurrentTime().add(javaafx.util.Duration.seconds(-5)));
533         } catch (Exception e) {
534             JOptionPane.showMessageDialog(null, "Elige un video primero");
535         }
536     }
537
538     if (ke.getCode().equals(KeyCode.UP) && ke.isControlDown()) {
539         try {
540             mediaPlayer.setRate(2);
541         } catch (Exception e) {
542             JOptionPane.showMessageDialog(null, "Elige un video primero");
543         }
544     }
545
546     if (ke.getCode().equals(KeyCode.DOWN) && ke.isControlDown()) {
547         try {
548             mediaPlayer.setRate(0.5);
549         } catch (Exception e) {
550             JOptionPane.showMessageDialog(null, "Elige un video primero");
551         }
552     }
553
554     if (ke.getCode().equals(KeyCode.N) && ke.isControlDown()) {
555         try {
556             mediaPlayer.setRate(1);
557         } catch (Exception e) {
558             JOptionPane.showMessageDialog(null, "Elige un video primero");
559         }
560     }
561

```

```

562 if (ke.getCode().equals(KeyCode.A) && ke.isControlDown()) {
563     try {
564         Parent root = FXMLLoader.load(getClass().getResource("AcercaDe.fxml"));
565
566         Scene scene = new Scene(root);
567         Stage acercaDe = new Stage();
568
569         acercaDe.setTitle("Media Player hecho por Adrian");
570         acercaDe.getIcons().add(new Image("images/icon.png"));
571
572         acercaDe.setScene(scene);
573         acercaDe.show();
574     } catch (IOException e) {
575     }
576 }
577
578 if (ke.getCode().equals(KeyCode.H) && ke.isControlDown()) {
579     try {
580         FadeTransition fadeTransitionVisibility = new FadeTransition(Duration.millis(200), setVisibilityButtons);
581         FadeTransition fadeTransitionOpenFile = new FadeTransition(Duration.millis(200), openFile);
582         FadeTransition fadeTransitionPlayVideo = new FadeTransition(Duration.millis(200), playVideo);
583         FadeTransition fadeTransitionStopVideo = new FadeTransition(Duration.millis(200), stopVideo);
584         FadeTransition fadeTransitionSlowVideo = new FadeTransition(Duration.millis(200), slowVideo);
585         FadeTransition fadeTransitionBackVideo5 = new FadeTransition(Duration.millis(200), backVideo5);
586         FadeTransition fadeTransitionNormalVideo = new FadeTransition(Duration.millis(200), normalVideo);
587         FadeTransition fadeTransitionForwardVideo5 = new FadeTransition(Duration.millis(200), forwardVideo5);
588         FadeTransition fadeTransitionFastVideo = new FadeTransition(Duration.millis(200), fastVideo);
589         FadeTransition fadeTransitionVolumeSlider = new FadeTransition(Duration.millis(200), volumeSlider);
590         FadeTransition fadeTransitionProgressBar = new FadeTransition(Duration.millis(200), progressBar);
591         FadeTransition fadeTransitionMenuBar = new FadeTransition(Duration.millis(200), menuBar);
592
593         if (visible) {
594             fadeTransitionVisibility.setToValue(0.2);
595             fadeTransitionOpenFile.setToValue(0);
596             fadeTransitionPlayVideo.setToValue(0);
597             fadeTransitionStopVideo.setToValue(0);
598             fadeTransitionSlowVideo.setToValue(0);
599             fadeTransitionBackVideo5.setToValue(0);
600             fadeTransitionNormalVideo.setToValue(0);
601             fadeTransitionForwardVideo5.setToValue(0);
602             fadeTransitionFastVideo.setToValue(0);
603             fadeTransitionVolumeSlider.setToValue(0);
604             fadeTransitionProgressBar.setToValue(0);
605             fadeTransitionMenuBar.setToValue(0);
606
607             fadeTransitionVisibility.play();
608             fadeTransitionOpenFile.play();
609             fadeTransitionPlayVideo.play();
610             fadeTransitionStopVideo.play();
611             fadeTransitionSlowVideo.play();
612             fadeTransitionBackVideo5.play();
613             fadeTransitionNormalVideo.play();
614             fadeTransitionForwardVideo5.play();
615             fadeTransitionFastVideo.play();
616             fadeTransitionVolumeSlider.play();
617             fadeTransitionProgressBar.play();
618             fadeTransitionMenuBar.play();
619
620             fadeTransitionMenuBar.setOnFinished(e -> {
621                 visible = false;
622             });
623         }
624         if (!visible) {
625             fadeTransitionVisibility.setToValue(1);
626             fadeTransitionOpenFile.setToValue(1);
627             fadeTransitionPlayVideo.setToValue(1);
628             fadeTransitionStopVideo.setToValue(1);
629             fadeTransitionSlowVideo.setToValue(1);
630             fadeTransitionBackVideo5.setToValue(1);
631             fadeTransitionNormalVideo.setToValue(1);
632             fadeTransitionForwardVideo5.setToValue(1);
633             fadeTransitionFastVideo.setToValue(1);
634             fadeTransitionVolumeSlider.setToValue(1);
635             fadeTransitionProgressBar.setToValue(1);
636             fadeTransitionMenuBar.setToValue(1);
637

```

```
638         fadeTransitionVisibility.play();
639         fadeTransitionOpenFile.play();
640         fadeTransitionPlayVideo.play();
641         fadeTransitionStopVideo.play();
642         fadeTransitionSlowVideo.play();
643         fadeTransitionBackVideo5.play();
644         fadeTransitionNormalVideo.play();
645         fadeTransitionForwardVideo5.play();
646         fadeTransitionFastVideo.play();
647         fadeTransitionVolumeSlider.play();
648         fadeTransitionProgressBar.play();
649         fadeTransitionMenuBar.play();
650
651         fadeTransitionMenuBar.setOnFinished(e -> {
652             visible = true;
653         });
654     }
655 } catch (Exception e) {
656 }
657 }
658
659 }
660
661 }
```

# MediaPlayer.fxml

```
1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <?import javafx.geometry.Insets?>
4 <?import javafx.scene.control.Button?>
5 <?import javafx.scene.control.Menu?>
6 <?import javafx.scene.control.MenuBar?>
7 <?import javafx.scene.control.MenuItem?>
8 <?import javafx.scene.control.Slider?>
9 <?import javafx.scene.effect.InnerShadow?>
10 <?import javafx.scene.layout.BorderPane?>
11 <?import javafx.scene.layout.HBox?>
12 <?import javafx.scene.layout.StackPane?>
13 <?import javafx.scene.layout.VBox?>
14 <?import javafx.scene.media.MediaView?>
15
16 <BorderPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity" minWidth="-Infinity" onKeyPressed="#handleKeyPressed"
prefHeight="400.0" prefWidth="600.0" styleSheets="@style.css" xmlns="http://javafx.com/javafx/17"
xmlns:fx="http://javafx.com/fxml/1" fx:controller="mediaplayer.MediaPlayerController">
17     <center>
18         <StackPane fx:id="pane" prefHeight="300.0" prefWidth="600.0" BorderPane.alignment="CENTER">
19             <children>
20                 <MediaView fx:id="mediaView" fitHeight="300.0" fitWidth="600.0">
21                     <effect>
22                         <InnerShadow />
23                     </effect>
24                 </MediaView>
25                 <VBox alignment="BOTTOM_CENTER" prefHeight="39.0" prefWidth="600.0">
26                     <children>
27                         <Slider fx:id="progressBar" prefHeight="0.0" prefWidth="600.0" />
28                         <HBox alignment="BOTTOM_CENTER" opacity="0.8" prefHeight="39.0" prefWidth="576.0">
29                             <children>
30                                 <Button fx:id="openFile" alignment="BOTTOM_LEFT" mnemonicParsing="false" onAction="#OpenFileMethod"
prefWidth="35.0">
31                                     <HBox.margin>
32                                         <Insets bottom="7.0" right="14.0" />
33                                     </HBox.margin>
34                                 </Button>
35                                 <Button fx:id="playVideo" alignment="BOTTOM_LEFT" mnemonicParsing="false" onAction="#playVideo"
prefWidth="35.0">
36                                     <HBox.margin>
37                                         <Insets bottom="7.0" right="7.0" />
38                                     </HBox.margin>
39                                 </Button>
40                                 <Button fx:id="stopVideo" alignment="BOTTOM_LEFT" mnemonicParsing="false" onAction="#stopVideo"
prefWidth="35.0">
41                                     <HBox.margin>
42                                         <Insets bottom="7.0" right="14.0" />
43                                     </HBox.margin>
44                                 </Button>
45                                 <Button fx:id="slowVideo" alignment="BOTTOM_LEFT" mnemonicParsing="false"
onAction="#furtherSlowDownVideo" prefWidth="35.0">
46                                     <HBox.margin>
47                                         <Insets bottom="7.0" right="7.0" />
48                                     </HBox.margin>
49                                 </Button>
50                                 <Button fx:id="backVideo5" alignment="BOTTOM_LEFT" mnemonicParsing="false" onAction="#back5"
prefWidth="35.0">
```

```

51         <HBox.margin>
52             <Insets bottom="7.0" right="7.0" />
53         </HBox.margin>
54     </Button>
55     <Button fx:id="normalVideo" alignment="BOTTOM_LEFT" mnemonicParsing="false" onAction="#normalSpeedVideo"
prefWidth="35.0">
56         <HBox.margin>
57             <Insets bottom="7.0" right="7.0" />
58         </HBox.margin>
59     </Button>
60     <Button fx:id="forwardVideo5" alignment="BOTTOM_LEFT" mnemonicParsing="false" onAction="#skip5"
prefWidth="35.0">
61         <HBox.margin>
62             <Insets bottom="7.0" right="7.0" />
63         </HBox.margin>
64     </Button>
65     <Button fx:id="fastVideo" alignment="BOTTOM_LEFT" mnemonicParsing="false"
onAction="#furtherSpeedUpVideo" prefWidth="35.0">
66         <HBox.margin>
67             <Insets bottom="7.0" right="21.0" />
68         </HBox.margin>
69     </Button>
70     <Slider fx:id="volumeSlider" stylesheets="@style.css">
71         <HBox.margin>
72             <Insets bottom="10.0" />
73         </HBox.margin>
74     </Slider>
75     <Button fx:id="setVisibilityButtons" alignment="BOTTOM_LEFT" mnemonicParsing="false"
onAction="#hideShowControls" prefWidth="35.0">
76         <HBox.margin>
77             <Insets bottom="7.0" left="21.0" />
78         </HBox.margin>
79     </Button>
80 </children>
81 </HBox>
82 </children>
83 </VBox>
84 <MenuBar fx:id="menuBar" prefHeight="0.0" prefWidth="600.0" StackPane.alignment="TOP_CENTER">
85     <menus>
86         <Menu mnemonicParsing="false" text="Ayuda">
87             <items>
88                 <MenuItem fx:id="menuAcercaDe" mnemonicParsing="false" onAction="#openAcercaDe" text="Acerca De" />
89             </items>
90         </Menu>
91     </menus>
92 </MenuBar>
93 </children>
94 </StackPane>
95 </center>
96 </BorderPane>

```

# AcercaDe.fxml

```
1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <?import javafx.geometry.Insets?>
4 <?import javafx.scene.control.Hyperlink?>
5 <?import javafx.scene.control.Separator?>
6 <?import javafx.scene.layout.BorderPane?>
7 <?import javafx.scene.layout.ColumnConstraints?>
8 <?import javafx.scene.layout.GridPane?>
9 <?import javafx.scene.layout.RowConstraints?>
10 <?import javafx.scene.text.Font?>
11 <?import javafx.scene.text.Text?>
12
13 <BorderPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity" minWidth="-Infinity" onKeyPressed="#handleKeyPressed"
  prefHeight="400.0" prefWidth="600.0" stylesheets="@style.css" xmlns="http://javafx.com/javafx/17"
  xmlns:fx="http://javafx.com/fxml/1" fx:controller="mediaplayer.MediaPlayerController">
14   <top>
15     <Text strokeType="OUTSIDE" strokeWidth="0.0" text="Acerca De" underline="true" BorderPane.alignment="CENTER">
16       <font>
17         <Font name="Arial Black" size="36.0" />
18       </font>
19       <BorderPane.margin>
20         <Insets top="20.0" />
21       </BorderPane.margin>
22     </Text>
23   </top>
24   <bottom>
25     <Text strokeType="OUTSIDE" strokeWidth="0.0" text="(c) Copyright amartinez Junio, 2022. Todos los derechos reservados"
  wrappingWidth="568.724609375" BorderPane.alignment="CENTER">
26       <BorderPane.margin>
27         <Insets bottom="10.0" />
28       </BorderPane.margin>
29       <font>
30         <Font name="Arial Black" size="12.0" />
31       </font>
32     </Text>
33   </bottom>
34   <center>
35     <GridPane BorderPane.alignment="CENTER">
36       <columnConstraints>
37         <ColumnConstraints hgrow="SOMETIMES" minWidth="10.0" prefWidth="100.0" />
38       </columnConstraints>
39       <rowConstraints>
40         <RowConstraints maxHeight="99.0" minHeight="0.0" prefHeight="0.0" vgrow="SOMETIMES" />
41         <RowConstraints maxHeight="274.0" minHeight="10.0" prefHeight="274.0" vgrow="SOMETIMES" />
42         <RowConstraints maxHeight="61.0" minHeight="0.0" prefHeight="0.0" vgrow="SOMETIMES" />
43       </rowConstraints>
44       <children>
45         <GridPane GridPane.rowIndex="1">
46           <columnConstraints>
47             <ColumnConstraints hgrow="SOMETIMES" minWidth="10.0" prefWidth="100.0" />
48           </columnConstraints>
49           <rowConstraints>
50             <RowConstraints maxHeight="62.0" minHeight="10.0" prefHeight="62.0" vgrow="SOMETIMES" />
51             <RowConstraints maxHeight="68.0" minHeight="0.0" prefHeight="23.0" vgrow="SOMETIMES" />
52             <RowConstraints maxHeight="73.0" minHeight="0.0" prefHeight="23.0" vgrow="SOMETIMES" />
53             <RowConstraints maxHeight="148.0" minHeight="10.0" prefHeight="124.0" vgrow="SOMETIMES" />
54             <RowConstraints maxHeight="115.0" minHeight="10.0" prefHeight="42.0" vgrow="SOMETIMES" />
55           </rowConstraints>
56           <children>
57             <Text strokeType="OUTSIDE" strokeWidth="0.0" text="Reproductor de video hecho por Adrián Martínez Medina">
58               <font>
59                 <Font name="Arial Black" size="14.0" />
60               </font>
61             <GridPane.margin>
62               <Insets left="20.0" />
```



```

63         </GridPane.margin>
64     </Text>
65     <Text strokeType="OUTSIDE" strokeWidth="0.0" text="Versión: 1.0" wrappingWidth="108.0" GridPane.rowIndex="1">
66         <font>
67             <Font name="Arial Black" size="14.0" />
68         </font>
69         <GridPane.margin>
70             <Insets left="20.0" />
71         </GridPane.margin>
72     </Text>
73     <Text strokeType="OUTSIDE" strokeWidth="0.0" text="Build: 20220318-001" wrappingWidth="204.599609375"
GridPane.rowIndex="2">
74         <font>
75             <Font name="Arial Black" size="14.0" />
76         </font>
77         <GridPane.margin>
78             <Insets left="20.0" />
79         </GridPane.margin>
80     </Text>
81     <Text strokeType="OUTSIDE" strokeWidth="0.0" text="Toda la documentación junto al proyecto está disponible en un
repositorio de Github con acceso gratuito, para acceder haz click en el siguiente link:" wrappingWidth="482.0" y="1.0"
GridPane.rowIndex="3">
82         <font>
83             <Font name="Arial Black" size="14.0" />
84         </font>
85         <GridPane.margin>
86             <Insets left="20.0" />
87         </GridPane.margin>
88     </Text>
89     <Text strokeType="OUTSIDE" strokeWidth="0.0" text="Esta aplicación ha sido desarrollada con JavaFX:" y="1.0"
GridPane.rowIndex="4">
90         <font>
91             <Font name="Arial Black" size="14.0" />
92         </font>
93         <GridPane.margin>
94             <Insets left="20.0" />
95         </GridPane.margin>
96     </Text>
97     <Hyperlink prefHeight="28.0" prefWidth="316.0" text="https://github.com/AdriMartinezMedina/TFG_MediaPlayer"
GridPane.rowIndex="3" GridPane.valignment="BOTTOM">
98         <GridPane.margin>
99             <Insets bottom="28.0" left="250.0" />
100         </GridPane.margin>
101     </Hyperlink>
102     <Hyperlink layoutX="20.0" layoutY="209.0" prefHeight="23.0" prefWidth="104.0" text="https://openjfx.io/"
GridPane.halignment="RIGHT" GridPane.rowIndex="4">
103         <GridPane.margin>
104             <Insets right="100.0" />
105         </GridPane.margin>
106     </Hyperlink>
107 </children>
108 </GridPane>
109 <Separator prefWidth="200.0" GridPane.rowIndex="2" />
110 <Separator layoutX="10.0" layoutY="268.0" prefWidth="200.0" />
111 </children>
112 </GridPane>
113 </center>
114 </BorderPane>

```



# Style.css

```
1 #pane {
2     -fx-background-color: black;
3 }
4
5 #openFile {
6     -fx-background-image: url("../images/of.png");
7     -fx-background-size: 20 20;
8     -fx-background-repeat: no-repeat;
9     -fx-background-position: center;
10 }
11
12 #playVideo {
13     -fx-background-image: url("../images/playpause.png");
14     -fx-background-size: 20 20;
15     -fx-background-repeat: no-repeat;
16     -fx-background-position: center;
17 }
18
19 #stopVideo {
20     -fx-background-image: url("../images/nogo.png");
21     -fx-background-size: 20 20;
22     -fx-background-repeat: no-repeat;
23     -fx-background-position: center;
24 }
25
26 #slowVideo {
27     -fx-background-image: url("../images/goback.png");
28     -fx-background-size: 20 20;
29     -fx-background-repeat: no-repeat;
30     -fx-background-position: center;
31 }
32
33 #fastVideo {
34     -fx-background-image: url("../images/goforward.png");
35     -fx-background-size: 20 20;
36     -fx-background-repeat: no-repeat;
37     -fx-background-position: center;
38 }
39
40 #normalVideo {
41     -fx-background-image: url("../images/go.png");
42     -fx-background-size: 20 20;
43     -fx-background-repeat: no-repeat;
44     -fx-background-position: center;
45 }
46
47 #backVideo5 {
48     -fx-background-image: url("../images/b5.png");
49     -fx-background-size: 20 20;
50     -fx-background-repeat: no-repeat;
51     -fx-background-position: center;
52 }
53
54 #forwardVideo5 {
55     -fx-background-image: url("../images/f5.png");
56     -fx-background-size: 20 20;
57     -fx-background-repeat: no-repeat;
58     -fx-background-position: center;
59 }
60
61 #setVisibilityButtons {
62     -fx-background-image: url("../images/eye.png");
63     -fx-background-size: 20 20;
64     -fx-background-repeat: no-repeat;
65     -fx-background-position: center;
66 }
```