JAVAFX MEDIA SUPPORT

IS PART-AWESOME,
PART-LAME

THIS SUPPORT MAKES IT POSSIBLE
TO REPRESENT DATA WHERE THE MODEL
IS A MEDIA FILE

AUDIO AND VIDEO FILES CAN BE
OPENED, PLAYED AND INTERACTED
WITH IN JAVAFX

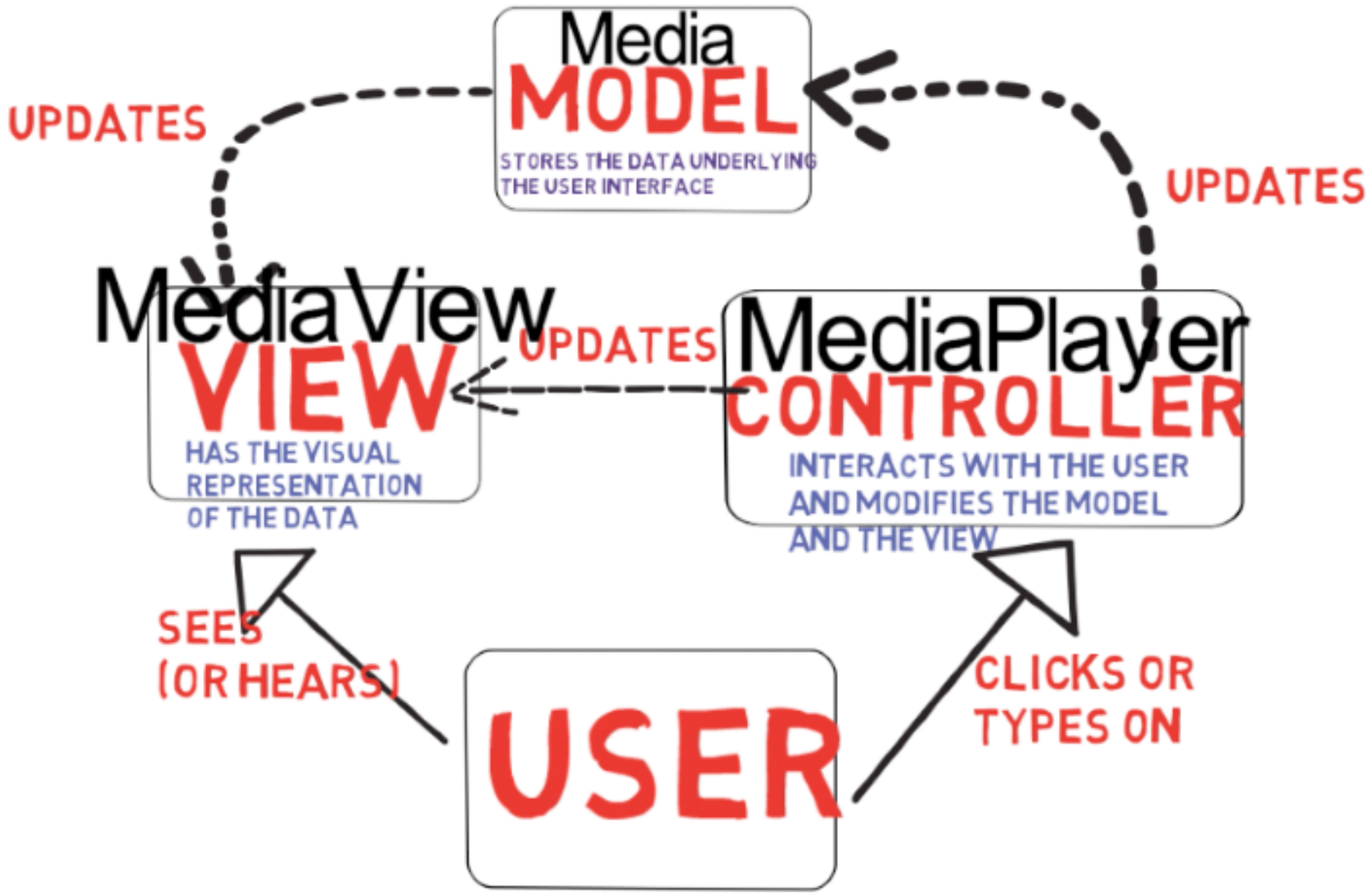
CREATING A MEDIA PLAYER,
ADDING SUBTITLES, AUDIO
EQUALIZATION ARE ALL
SUPER-SIMPLE

THIS IS AWESOME

THE ONE DOWNSIDE OF JAVAFX
IS THAT THIS SUPPORT IS MOSTLY
"READ-ONLY"

THIS IS LAME

YOU CAN'T CREATE MEDIA FILES
(FOR INSTANCE FROM SCREEN
CAPTURE OR A CAMERA), AND
YOU CAN'T SAVE MODIFICATIONS
THAT YOU MAKE TO MEDIA FILES


Media
MODEL
STORES THE DATA UNDERLYING
THE USER INTERFACE
UPDATES
UPDATES
MediaView
VIEW
HAS THE VISUAL
REPRESENTATION
OF THE DATA
UPDATES
MediaPlayer
CONTROLLER
INTERACTS WITH THE USER
AND MODIFIES THE MODEL
AND THE VIEW
SEES
(OR HEARS)
USER
CLICKS OR
TYPES ON

MEDIA CLASSES YOU SHOULD REMEMBER ARE

MEDIA OBJECTS CAN BE OPENED FROM VARIOUS FILE FORMATS - NOTABLY MP3 AND MP4 AMONG OTHERS

Media — ENCAPSULATES AN AUDIO OR VIDEO FILE. HAS DURATION, TRACKS, METADATA

MediaPlayer — ALLOWS THE USER TO DO ALL OF THE THINGS USERS DO WITH MEDIA - PLAY, PAUSE, STOP, SEEK, CHANGE VOLUME, ADD SUBTITLE MARKERS, EQUALIZE TRACKS

MediaView — AUDIO MEDIA NEED NOT HAVE A VIEW, BUT VIDEO MEDIA MUST - THE MEDIAVIEW IS WHERE THE VIDEO ACTUALLY APPEARS ON SCREEN

MEDIA CLASSES YOU SHOULD REMEMBER ARE

MEDIA OBJECTS CAN BE OPENED FROM VARIOUS FILE FORMATS – NOTABLY MP3 AND MP4 AMONG OTHERS

Media ENCAPSULATES AN AUDIO OR VIDEO FILE. HAS DURATION, TRACKS, METADATA

BTW, TO PLAY AN AUDIO FILE IN A QUICK-AND-DIRTY MANNER WITHOUT SETTING UP A PLAYER, THERE IS A HANDY LITTLE CLASS CALLED **AudioClip**

MediaPlayer ALLOWS THE USER TO DO ALL OF THE THINGS USERS DO WITH MEDIA – PLAY, PAUSE, STOP, SEEK, CHANGE VOLUME, ADD SUBTITLE MARKERS, EQUALIZE TRACKS

MediaView AUDIO MEDIA NEED NOT HAVE A VIEW, BUT VIDEO MEDIA MUST – THE MEDIAVIEW IS WHERE THE VIDEO ACTUALLY APPEARS ON SCREEN

THE MEDIPLAYER CLASS HAS 2 PROPERTIES WORTH PAYING ATTENTION TO

STATUS – WHICH IS SET USING METHODS LIKE PLAY(), PAUSE, STOP(), AND READ USING GETSTATUS()

CURRENTTIMEPROPERTY – WHICH TELLS WHAT POINT IN THE SONG OR VIDEO WE ARE AT

THIS CRUCIAL PROPERTY IS READ-ONLY (SET VIA SEEK), WHICH MAKES BINDING TO IT QUITE COMPLICATED

NOW THERE IS SOMETHING THAT IS RATHER COMPLICATED -

WHICH IS HAVING A SLIDER THAT MOVES ALONG AS THE VIDEO PLAYS

THE SLIDER SHOULD AS EXPECTED, I.E. IT SHOULD MOVE ALONG AS THE VIDEO PLAYS,

..AND ALSO, AND IF THE USER MOVES THE SLIDER, THE VIDEO SHOULD SKIP TO THE WHERE THE USER LEFT IT

NOW THIS COULD HAVE BEEN AS SIMPLE AS A BIDIRECTIONAL BINDING BETWEEN THE CURRENTIMEPROPERTY OF THE MEDIAPLAYER AND THE VALUE OF THE SLIDER

TO UPDATE THE SLIDER AS THE TRACK PLAYS..

```
InvalidationListener positionSliderListener = new InvalidationListener() {
    @Override
    public void invalidated(Observable observable) {
        Runnable r = () -> {
            if (!positionSlider.isValueChanging()) {
                positionSlider.setValue(mediaPlayer.getCurrentTime().toMillis() /
                        mediaPlayer.getTotalDuration().toMillis());
            }
        };
        Platform.runLater(r);
    }
};
mediaPlayer.currentTimeProperty().addListener(positionSliderListener);
```

LISTEN ON THE CURRENTIMEPROPERTY OF THE MEDIAPLAYER

IN THAT LISTENER, EACH TIME THE TRACK MOVES, UPDATE THE VALUE OF THE SLIDER

AND BE SURE TO WRAP THIS INSIDE A CALL TO PLATFORM.RUNLATER

WE ALSO NEED TO UPDATE THE TRACK WHEN THE USER MOVES THE SLIDER

ADD A LISTENER SO WE ARE ALERTED WHENEVER THE USER MOVES THE SLIDER

```
positionSlider.valueProperty().addListener((obsVal, oldValue, newValue) -> {
    if (mediaPlayer == null) {
        return;
    }
    MediaPlayer.Status status = mediaPlayer.getStatus();
    mediaPlayer.pause();
    Duration seekPosition = Duration.millis(positionSlider.getValue() *
            mediaPlayer.getTotalDuration().toMillis());
    mediaPlayer.seek(seekPosition);
    if (status == MediaPlayer.Status.PLAYING) {
        mediaPlayer.play();
    }

});
```

PAUSE THE AUDIO OR VIDEO

FIND THE POINT IN THE VIDEO THAT CORRESPONDS TO THE SLIDER VALUE

SEEK TO THAT POINT

AND RESUME IF IT WAS PLAYING

IN THAT LISTENER -