# MediaContentView.SetMedia(mediaFile)

Aufgerufen von FileTableView (Haupt-Aufruf für mainMediaContentView)

* Und MediaContentView (zur Weitergabe an Voll-Bild-Instanz hasFullScreenMediaContentView=fullScreenStage != null)
* Und UnDeleteView (um gelöschte Medien anzuzeigen)

Probiere nacheinander

photoViewer.setMediaFileIfCompatible(mediaFile)

playerViewer.setMediaFileIfCompatible(mediaFile)

# setMediaFileIfCompatible(mediaFile)

Grundidee: das eigentliche Laden führt der Viewer aus, denn nur er weiß, wie sein „Media“ aussieht und wie/ob es gecached werden kann (bisher werden nur Fotos gecached)

Retry ist rekursiv implementiert: mediaFile.getCachedOrLoadMediaContent ruft getViewerSpecificMediaContent auf. Dieser installiert einen ErrorHandler, der wiederum mediaFile.getCachedOrLoadMediaContent auruft.

Der Retry-Counter wird in mediaFile verwaltet. Bevor der ErrorHandler in Viewer.getViewerSpecificMediaContent das MediaFile mit mediaFile.GetCachedOrLoadMediaContent neu lädt, muss der ErrorHandler mediaFile.retryCounterNotExceeded() prüfen.

|  |  |
| --- | --- |
| photoViewer | playerViewerFX |
| Super: currentlyShowedMediaFile=mediaFile | = |
| instanceof ImageFile? | Instanceof PlayableFile? |
|  | resetPlayer() |
| mediaFile.getCachedOrLoadMediaContent (isRetry=false) | = |
| resetLoadRetryCounter | = |
| mediaCache.maintainCacheSizeByFlushingOldest | = |
| photoViewer.getViewerSpecificMediaContent | playerViewerFX…. |
| new Image(mediaFile) | New MediaPlayer(new Media) |
| image.exceptionProperty.AddListener{  mediaFile.getCachedOrLoadMediaContent  maintainCacheSizeByFlushingOldest  getViewerSpecificMediaContent  newImage + Errorhandler  refreshViewIfCurrentMediaFile  } | mediaPlayer.setOnError{  resetPlayer()  System.gc()  runLater{  =  =  }  } |
| refreshViewIfCurrentMediaFile 🡪 setImage | = 🡪 install handler + mediaView.setMediaPlayer() |