Travail pratique 1 – Lecteur Multimedia utilisant ExoPlayer

( partie 1 de 2 )



Objectifs :

* Utiliser les classes reliées à ExoPlayer pour diffuser de la musique
* Utiliser les librairies Volley et Klaxon afin de récupérer les ressources musicales sur un serveur distant
* Développer une application fonctionnelle avec un thème original
* Utiliser le langage Kotlin
* Séparer autant que possible le modèle de données de la vue en utilisant le modèle de conception Observer
* Gérer le cycle de vie de l’app pour que l’application soit agréable à utiliser

## Partie 1 de 2 – Lecteur comme tel, opérations sur le player

Créez un nouveau projet API 26.

Développez votre idée de lecteur. Vous avez la liberté d’explorer la grande bibliothèque de classes ExoPlayer. Les fichiers à lire sont présents sur le serveur :

<https://api.jsonbin.io/v3/b/680a6a1d8561e97a5006b822?meta=false>

Autre serveur disponible : <https://api.npoint.io/d4c29479e010376e6847>

Si le serveur ne répond pas : Arrêter l'émulateur. Le fermer. Le redémarrer seul et ensuite démarrer votre application

Attention ! Ce fichier JSON est appelé à être modifié à plusieurs reprises…la structure des objets ne changera pas cependant.

## Le modèle

1 – récupérer le jason puis fair un toast …?

Vous devrez lire les données en utilisant les librairies Volley et Klaxon de manière à vous bâtir une liste d’objets représentant votre librairie de fichiers multimédias. Cette partie doit être faite dans une classe n’étant pas une Activité.

Vous devrez trouver un moyen de détecter dans votre Vue que la réponse du serveur a été bien reçue dans le modèle 🡪 à venir à l’atelier 3

Cette / ces classes pourraient être l’occasion de filtrer les fichiers multimédias par catégorie ( artiste, album, genre… )

## La Vue

La Vue comprendra le ExoPlayer et les interfaces graphiques. Tel que vu dans l’annexe 4, on doit lier le ExoPlayer avec le composant graphique PlayerView. **Cependant, désactiver les contrôles automatiques sur le PlayerView en appliquant la méthode :**

***vue.setUseController(false);* où vue est l’objet PlayerView**

le but étant que vous développiez une interface originale où des boutons / autres composants peuvent être reliés à des actions ( clic sur un bouton 🡪 appel de la méthode play sur l’objet Player )

Autres directives sur la Vue

* Évitez de changer les couleurs au cas par cas, utilisez les thèmes, les styles et les StateListDrawables au besoin
* Faites en sorte que tourner le téléphone n’interrompt pas la musique diffusée ( fin de l’**annexe 4** )
* Vous pouvez avoir une ou plusieurs activités, une bien entendue doit être destinée aux contrôles et à diffusion de la musique.

## Vos premiers objectifs

* Lire le fichier JSON avec les chansons ( Atelier 1 )
* Utiliser Klaxon pour décomposer / recomposer les objets ( Atelier 1B )
* Présenter l’ensemble des chansons avec une liste complexe ( Annexe 5, Atelier 4 )
* Contrôles de base ( PLAY, SUIVANT, PRÉCÉDENT, PAUSE, AVANCER DE 10 SECONDES … ) ( Atelier 2 )
* Détecter l’obtention de la réponse du modèle dans la vue ( Atelier 3 )

Cette partie ci-haut devrait être terminée le 22 octobre environ

Partie 2 : à venir : cours du 22 octobre

* Détecter les secondes passées pour faire avancer la seekBar ( Handler, Runnable, Thread ) car il n’y a pas d’écouteur dédié pour cela dans l’API d’ExoPlayer.
* Opération « Seek » pour avancer / reculer dans la chanson avec la SeekBar
* Playlistes
* Autres
* Grille de correction

## Utilisation de l’AI dans le travail

* Votre code doit être commenté de manière à comprendre votre démarche, surtout dans les algorithmes. Si je ne comprends pas un passage de votre code ( car il n’est pas commenté en autres ) en relation avec la matière vue au cours, j’enlèverai ce code de votre application et les points s’y rattachant.
* Le but du travail est d’évaluer votre capacité à utiliser les concepts vus dans les annexes /ateliers précédents

## Plagiat

Tel que mentionné dans les [règles d’encadrement départementales](https://ericlabonte.com/annexes/PIEA_encadrements(janvier2024).pdf) le plagiat sur un autre travail entraînera la note 0.

Le logiciel MOSS développé à l’université Stanford sera utilisé <https://theory.stanford.edu/~aiken/moss/>. Si la comparaison de deux travaux identifie un nombre de lignes identiques considérablement supérieur à la moyenne des comparaisons des autres paires de travaux, les deux travaux obtiendront la note 0.

## Événements sur le ExoPlayer :

<https://developer.android.com/media/media3/exoplayer/listening-to-player-events?hl=fr>