Travail pratique 1 – Lecteur Multimedia utilisant ExoPlayer ( 10 % de la session )

Partie 2 – Grille de correction, seekBar activée par thread, liste complexe

|  |  |
| --- | --- |
| **Modèle** | |
| Lire les données du fichier JSON afin de les obtenir dans votre application à partir d'un serveur ( librairie Volley ) | 3/3 |
| Utiliser GSON pour simplifier la lecture | 2/2 |
| Classes du modèle représentant le fichier JSON | 1/1 |
| Le faire dans une classe autre que l'activité   * Utiliser le patron de conception Observer pour détecter l'obtention de la réponse de la requête ( Atelier 5 ) * BONUS : la classe lisant le fichier JSON pourrait être un Singleton | 3/3  +1 |
|  |  |
| **VUE** | |
| Utilisation d'un PlayerView auquel les contrôles ont été cachés | 1/1 |
| Présence des contrôles de base : |  |
| PLAY démarre la chanson courante | 2/2 |
| PAUSE arrête momentanément la diffusion | 2/2 |
| PLAY suite à un PAUSE , reprend où on a quitté | 2/2 |
| PRÉCÉDENT / SUIVANT avance à la chanson suivante / précédente  Qu'arrive-t-il s'il n'y a plus de chanson dans la playlist ? | 2/2 |
| AVANCER / RECULER de 10 secondes la chanson courante | 3/3 |
|  |  |
| Présence d'une SeekBar ( Thread / Runnable : à voir )   * Suit la diffusion de la chanson * S'arrête quand on clique sur Pause * Reprends quand on reclique sur Play * Revient au début si nouvelle chanson * Avance ou reculer si on clique sur AVANCER de 10 sec ou RECULER de 10 sec | 4/4 |
| SeekBar 🡪 avancer pour chercher dans la chanson  Seekbar fonctionne normalement ensuite | BONUS +1  1 |
| Affichage des infos de la chanson courante ( nom, artiste, etc ) | 2/2 |
| Développement d'un ListView avec SimpleAdapter ( Atelier 6, Annexe 5 ) dans la même activité ou une autre activité comprenant toutes les chansons sur le serveur | 5/5 |
| Gestion des événements sur un item du ListView afin de diffuser immédiatement cette Chanson | 2/2 |
|  |  |
| Utilisation d'événements sur le ExoPlayer – au choix | 2/2 |
|  |  |
| Redéfinition du thème afin de donner un look original / dynamique | 0/2 |
|  |  |
| En cas de fermeture de l'app, fermer l'exo player de manière correcte | 1/1 |
| Pouvoir redémarrer l'app à l'endroit où on l'a quittée ( endroit dans la chanson, chanson courante 🡪 sérialisation ) | 3/3 |
|  |  |
| Si on tourne le téléphone, la diffusion continue, l'apparence tient la route ( Annexe 4, page 7 ) | 1/1 |
| Prouver que vous êtes capable de faire un « boomerang » ( startActivityForResult, annexe 4 ) | /3 |
| Autres idées : volume, diffusion aléatoire, répétitive, faire des playlists selon le type de chansons, etc | BONUS ( jusqu’à +5 ) |
|  |  |
|  |  |
| Commentaires pertinents  Indentation  Bonnes pratiques de programmation | /4 |
| TOTAL | /50 |

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>

API / Javadoc :

<https://developer.android.com/reference/androidx/media3/common/package-summary>