CodingWeek 2019

TELECOM Nancy - 16/12 au 20/12

TelecomNancy PodcastCollector

> TelecomNancy PodcastCollector est une application qui permet gérer ses collections de podcasts. Cette application permet d'écouter ses podcasts préférés, de classer ceux-ci et d'en découvrir de nouveaux.

Sujet

L'objectif de ce projet est donc de réaliser une application permettant de gérer des collections de podcasts [1]. Il existe déjà bon nombre d'applications de ce genre. Si vous n'êtes pas familier avec ce genre d'applications nous vous invitons vivement à étudier le fonctionnement des applications existantes. L'application devra permetttre en s'abonnant à des flux de données généralement au format RSS ou Atom disponible sur Internet, d'écouter immédiatement ou de télécharger des émissions audio (et éventuellement vidéo).

Les principales fonctionnalités attendues sont :

- l'abonnement/le désabonnement à un flux d'émissions
- l'écoute immédiate d'une émission (lecture, pause, reprise d'une lecture)
- l'écoute différée après téléchargement d'une émission. Il doit donc être possible de télécharger les émissions pour pouvoir les écouter de manière déconnectée
- la présentation à l'utilisateur des différentes informations sur un flux d'émission ou sur une émission en particulier (description, illustration, durée)
- la découverte de nouveaux flux d'émissions par la recherche dans des listes pré-établies
- le classement des flux d'émissions par catégories
- la recherche locale d'un flux d'émissions
- la mise à jour des flux d'émissions (émissions nouvellement disponibles, qui ne sont plus disponibles, etc.)
- la différenciation des émissions déjà écoutées de celles qui sont nouvelles

Il est possible d'envisager un certain nombre d'extensions aux fonctionnalités basiques présentées ci-dessus.

- la recommendation de nouveaux flux par rapport aux préfèrences de l'utiliisateur
- l'exportation, le partage et l'importation de flux d'émissions ou d'émissions en particulier
- la génération manuelle ou automatique d'une liste de lecture des émissions
- le support des émissions vidéos et non plus uniquement audio
- la gestion d'émissions ou de flux d'émissions favoris
- ...

1. https ://fr.wikipedia.org/wiki/Podcasting

Organisation

Constitution des groupes

Les groupes sont constitués de 4 élèves qui ne sont pas issus obligatoirement du même groupe de TD. Il vous est donc demandé pour le jeudi 12 décembre 2019 au plus tard de fournir les informations concernant votre groupe en remplissant le formulaire dédié à l'adresse https://forms.gle/KEbJyLYnTnoSMesV6.

Il vous sera demandé de fournir :

- le nom de votre groupe,
- sa composition (pour chaque membre du groupe : son nom, prénom, son adresse email (terminant par `@telecomnancy.eu`).

Ces informations nous permettront de vous créer un projet sur le serveur GitLab de l'école dédié à votre groupe.

Présence et assiduité

La semaine a été dans sa très large majorité bloquée pour vous permettre de vous concentrer sur cette épreuve. Des salles de TD et TP seront réservées spécifiquement pour l'épreuve (aile est du 1er étage du bâtiment). Il vous est demandé d'indiquer dans quelle salle votre groupe se trouve.

Encadrement

Chaque groupe se verra assigner un enseignant référent. Chaque groupe rencontrera son enseignant à intervalles réguliers. Au cours des journées, plusieurs enseignants seront disponibles pour répondre à vos questions. Ceux-ci seront joignables à travers le forum ou dans une salle dont le numéro vous sera communiqué ultérieurement.

Travail et collaboration

Suite à l'enregistrement de votre groupe, un dépôt privé Git vous sera créé où vous déposerez le code source et les différentes informations nécessaires à la compilation et à l'exécution de votre application (Java).
Il est nécessaire de bien organiser le contenu de ce dépôt et de "commiter" régulièrement afin que l'ensemble des membres du groupe ait accès aux données les plus récentes. Il sera tenu compte des contributions de chacun des membres du groupe. Si un des membres n'a pas su "commiter" ses contributions, on considérera sa participation comme nulle.

Vous devrez :

- documenter la planification et l'avancement du projet ;
- vous assurer des livraisons (releases) quotidiennes et de la bonne utilisation de Git ;
- mettre en œuvre des tests et des procédures assurant la qualité du logiciel ;
- documenter votre architecture et votre conception pour les rendre compréhensibles ;
- documenter l'installation et l'exécution de votre application.

Ces différents points seront pris en compte pour un tiers de la note.

Travail préparatoire

Vous disposez de quelques jours pour vous mettre à niveau sur les technologies que vous devrez utiliser :

- renforcement de vos connaissances sur l'utilisation de JavaFX et le fonctionnement interne de Git
- mise en place de l'environnement de travail (outils de développement, gestion de versions et de bases de données, etc.) ;
- remise à niveau sur le fonctionnement d'un serveur de base de données et de l'accès à celui-ci à travers le langage Java ;
- étude des technologies liées qui pourraient être utilisées.

Quand vous utilisez Git faites attention à ne pas publier vos clés d'accès aux APIs que vous utilisez [a]. Pour cela, il convient de placer ces clés dans un fichier de configuration et de filtrer ce fichier en utilisant le fichier `.gitignore`. Vous devez vous arranger pour que chacun utilise ses propres clés d'accès aux API. Si possible, prévoyez des clés d'accès pour vos démonstrations, sinon documentez la procédure pour obtenir et fournir ces clés.

a. Ars Technica. *PSA : Don't upload your important passwords to* GitHub. Janvier 2013

Évaluations et rendus

Le développement se fait de **façon itérative et incrémentale**. Vous devez produire un **logiciel opérationnel chaque jour**. Nous vous recommandons d'utiliser un outil tel que Gradle pour automatiser la compilation, l'exécution des tests et la construction de votre application. Vous y ajouter des fonctionnalités en fonction d'une feuille de route (roadmap) que vous aurez définie à l'avance et que vous ferez évoluer en fonction de votre avancée. Il n'est pas question d'avoir plusieurs morceaux de programmes fonctionnant indépendamment les uns des autres. Le programme doit être intégré chaque jour et vous devez à tout moment pouvoir faire une démonstration de la dernière livraison. Le document `README` du dépôt doit indiquer comment exécuter ce programme.

Rendus intermédiaires

Chaque groupe devra effectuer un rendu journalier sous la forme de commits Git (cela ne signifie pas qu'il faut se limiter à un commit par jour dans le dépôt, au contraire). Ce rendu devra être étiqueté (au moyen de la commande `git tag` dans la branche master) selon la convention de nommage suivante : `RELEASE_DAY_X` où X indique le jour de la semaine (lundi : 1, mardi : 2, etc.). Il s'agira d'une livraison, donc d'un système déployable et opérationnel. Par défaut, la commande `git push` ne partage pas les tags, il faut explicitement partager ces tags (en utilisant l'option `--tags`).

La contribution de chacun des participants d'un groupe pourra être mesurée à son activité sur GitHub, i.e. le nombre de commit qu'il aura pu faire et leur taille. Il ne sera pas admis que certains d'entre vous n'aient pas commité de code à leur nom. Il est donc important de bien configurer votre client Git afin que vos commits vous soient attribués [a].

a. Vérifiez les variables `user.name` et `user.email` au moyen de la commande `git-config --list`.

Rendu final

Le rendu final sera étiqueté (toujours la branche master) selon la convention de nommage suivante : `RELEASE_FINAL`.

Il devra comporter a minima :

- les documents d'analyse et de conception que vous aurez réalisés ;
- les documents de gestion de projet ;
- le code source de l'application client lourd (Java), les instructions textuelles (texte brut ou fichier PDF) indiquant comment compiler, configurer et exécuter cette application (précisant les dépendances externes et comment avoir accès à ces dépendances).

Les derniers commit auront lieu avant 15h le vendredi 20 décembre 2019.

Il ne sera pas possible de faire de démonstration le dernier jour. Vous nous rendrez une démo de 10 mn de votre application sous la forme d'un screencast présentant ses fonctionnalités. Vous pouvez utiliser un outil de capture d'écran tel que OBS Studio https://obsproject.com/ ou tout autre outil. Le plus simple est de déposer votre vidéo sur une plateforme en ligne telle que YouTube ou Dailymotion et de publier ensuite le lien dans votre fichier `README` de votre dépôt Git (ce n'est pas un problème si le commit publiant ce lien est réalisé après l'heure de rendu limite de votre projet).

Cette vidéo devra être produite également le vendredi 20 décembre 2019. Elle pourra être mise en ligne après 15h mais avant le lendemain midi.

Critères d'évaluation

- résultats concrets présentés à la fin de la semaine ;
- respect des méthodes ;
- organisation du travail ;
- mise en œuvre de tests démontrant le fonctionnement du logiciel ;
- qualité du développement (principe de conception mis en œuvre, architecture adaptée).
- invention de nouveaux services et caractère innovant du logiciel ;

Communication avec les encadrants

Afin de poser vos questions et de discuter durant la semaine, nous avons crée sur la plateforme de messagerie instantanée de l'école (Mattermost) une équipe avec différents canaux. N'hésitez pas à y poser des questions sur le sujet ou sur des points techniques.

https://gl-mattermost.telecomnancy.univ-lorraine.fr/codingweek2k19

Afin de rejoindre l'équipe Mattermost dédiée à la codingweek, vous devrez utiliser le lien d'invitation ci-dessous à votre première connexion.

https://gl-mattermost.telecomnancy.univ-lorraine.fr/signup_user_complete/?id=u9xgyoeybtnr9dq6s9tyt4rtwo

Quelques pointeurs

Vous trouverez dans cette section divers pointeurs sur des librairies que vous pouvez utiliser. Vous pouvez bien entendu utiliser d'autres librairies que celles présentées.

HttpClient. Depuis Java 11, une API HTTPClient est intégrée à l'API Java pour réaliser des requêtes HTTP et permettre à vos programmes de converser avec des serveurs Web. Un tutoriel d'introduction de cette API est disponible en ligne (https://openjdk.java.net/groups/net/httpclient/intro.html). Vous pouvez également utiliser la librairie Apache HTTPComponents (https://hc.apache.org/).

Parseur XML. Les fichiers de syndication au format RSS et Atom sont des documents XML. Pour lire et analyser ceux-ci, vous pouvez utiliser l'API XML Java pour écrire un analyseur DOM (Document Object Model) ou StAX (Streaming API for XML). Le tutoriel écrit par J.-M. Doudoux (https://www.jmdoudoux.fr/java/dej/chap-stax.htm) est un très bon point de départ. Vous pouvez également si vous le souhaitez ré-utiliser une librairie existante telle que ROME (https://rometools.github.io/rome).

Podcasts. Il existe de nombreux sites proposant des podcasts tels que les sites de Radio France. Vous pouvez directement intégrer les liens vers ces flux dans votre application. Vous pouvez également faire appel à un service intermédiaire proposant déjà une liste de fournisseurs de podcasts. À titre d'exemple, la plateforme ListenNotes fournit une API (https://www.listennotes.com/api/docs) permettant d'obtenir des flux selon différents critères.

Multimedia. L'API Media incluse (javafx.media) dans la boîte à outils JavaFX devrait vous permettre de lire les principaux flux audio et vidéo diffusés dans les podcasts de votre application (https://openjfx.io/javadoc/11/javafx.media/javafx/scene/media/package-summary.html).

Gradle / JavaFX. Le projet herbergé sur GitHub (https://github.com/Typhon0/boilerplategradlejdk13) peut vous servir de point de départ pour configurer un projet Gradle support Java (ici la version 13, mais facilement adaptable à la version 11) et JavaFX.