Rapport - Projet Applications mobiles

Najmeddine Ayoub
Hamza Zougari Belkhayat
Othmane Mokrane

Département Sciences du Numérique
2021-2022

Table des matières

1 Introduction

Le but de ce projet est de développer une application mobile qui permet d'échanger des données entre deux smartphone **Android** via une connexion Bluetooth, à travers des objets socket. l'application sera installer sur les deux smartphone afin d'échanger les informations relatives à l'état du processus de l'appareil distant.

2 Monitoring de processus à distance

La première activité à créer est une activité de monitoring qui permat au clic sur le bouton d'afficher la liste des processus actifs du client, pour cela, on créé une interface avec un bouton au milieu, et ensuite lui associer un *listener* qui permet de lancer l'affichage des information des application installées sur le client à savoir : leurs noms, leurs *pid*, et la valeur du *RSS* monitorée.

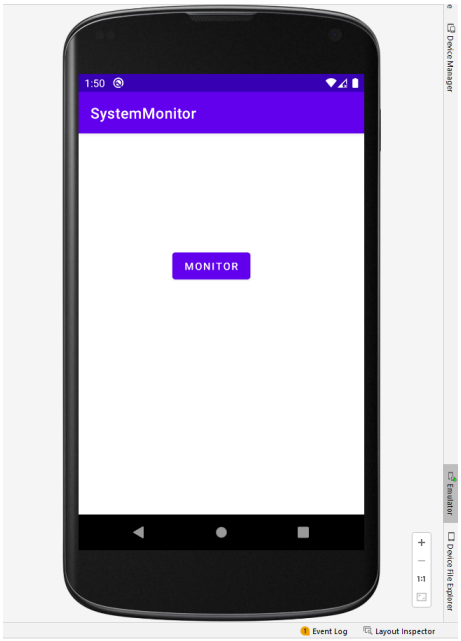

FIGURE 1 – La première activité.

Afin d'afficher les processus du système nous procédons à la préparation de l'interface graphique de notre seconde activité, celle-ci est constitué d'un *layout* principale de type *ScrollView*, ce widget est un conteneur qui contient une barre de défilement, on ajoute par la suite un *LinearLatout* dans le widget précedent pour le pouvoir peupler avec la liste des processus du système.

Nous avons récupérer le nom de l'application en cours d'execution ainsi que sa consommation mémoire, en utilisant les appels système pour récupérer les informations relatifs aux processus (appel système *ps*).

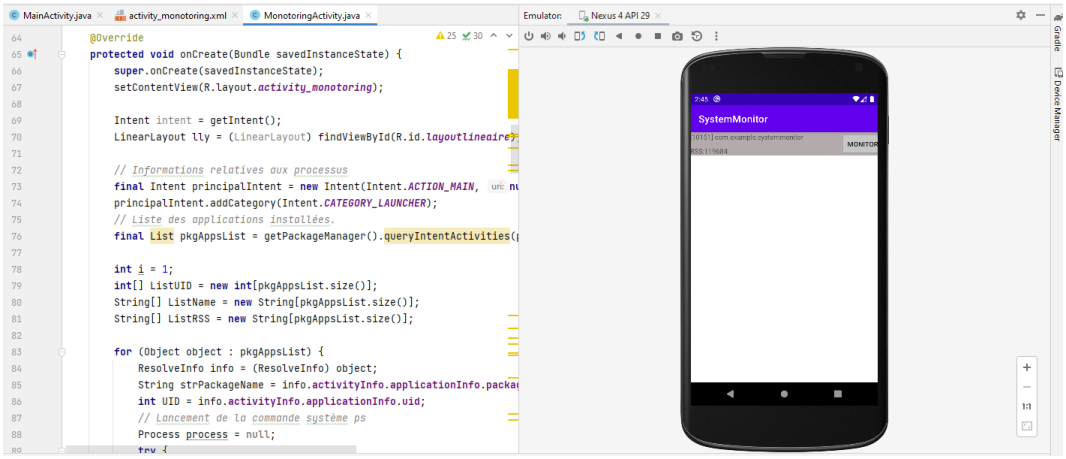

FIGURE 2 – Affichage des informations relatives aux processus actifs.

3 Mise en place de la connexion Bluetooth

Dans cette section on s'interesse à l'échange des information relatifs aux application actifs entre les appareils Client/Serveur. Cet échange se fait par l'ouverture d'un socket Bluetooth. L'ouverture du socket de communication passe par une étape de connexion.

L'application démarre sur un écran d'accueil oú l'utilisateur initialise la connexion Bluetooth. Sur cet écran on a positionné deux bouton, un bouton pour choisir le mode Client, et un bouton pour choisir le mode serveur Bluetooth.

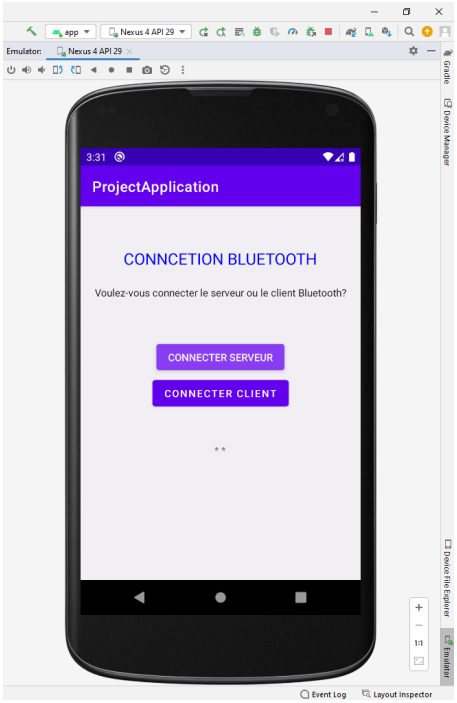

FIGURE 3 – Écran d'invite de connexion.

Pour passer en mode Serveur l'utilisateur appuie sur le bouton **CONNECTER SERVER**, tandis que pour passer en mode client il choisit le bouton **COONNECTER CLIENT**. Pendant que la connexion s'établit, l'affichage est modifié selon les écrans suivants.

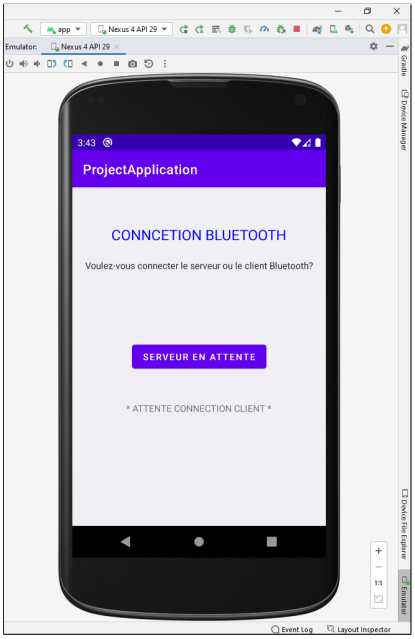

FIGURE 4 – Attente connexion client.

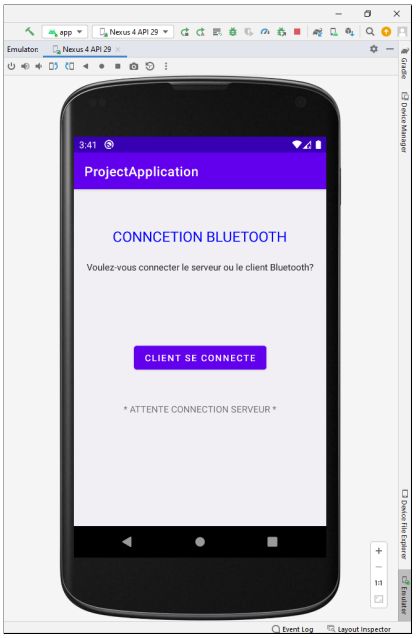

FIGURE 5 – Attente connexion Serveur.

Pour obtenir le socket de communication, le serveur utilise un objet 'serveur de socket' *BluetoothServerSocket* qui attend la connexion d'un client pour retourner un socket Bluetooth. Ce serveur de socket est ensuite mis en écoute. Quand une requête de connexion arrive d'un client, ce 'serveur de socket' retourne un socket de communication de type BluetoothSocket. Ce socket est prêt à être manipulé pour échanger des messages avec le client.

Ce socket est manipulé via un thread *ConnectedThread* pour envoyer les messages vers le l'appareil client. Pour que les données reçus dans le thread de réception soient exploitable par une activité, on associe un *handler* au thread.

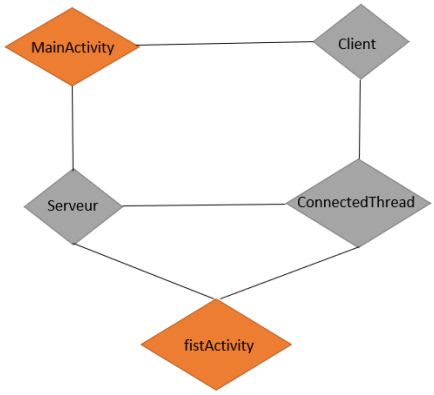

FIGURE 6 – Architecture de l'application.

3.1 Class MainActivity

La class principal offre à l'utilisateur la possibilité de choisir entre l'un des deux mode de transfert de données(émission/réception), ensuite identifie l'affichage associé à son choix en attente de l'établissement de la connexion bluetooth.

Du coté client on doit récupérer la liste des appareils appairer par Bluetooth et choisir le bon device et puis le lacement du thread Client *ClientClass* et du thread Serveur. Quant au serveur le thread *ConnectedThread* lance un appel à la méthode *server_ socket.accept()* et récupère le socket retourné pour le stocker dans un attribut d'une classe static *SocketHandler*.

3.2 Class firstActivity

Cette class permet d'extraire les informations nécessaire à l'affichage des processus actif comme décrit à la section 2.