Assignment 3 – Mobile Services

Distributed and Mobile Systems

# Brief

The purpose of this assignment is to create an android application with services to communicate between two or more android devices. The purpose of the android application is up to the project team, but must show implementations of two services. These two services can be chosen from the following four: Messaging service, Bluetooth service, Location-based service, Near field communication.

* Messaging service: Utilizes SMS and/or MMS messaging between two mobile devices
* Bluetooth service: Uses the Client/Server communication between two devices via Bluetooth.
* Location-based service: Uses the device’s location to do some sort of operation
* Near Field Communication: uses tags to send information from one device to another.

# The Android Application

The android application the project team is developing is called “Pokimin”. It is a parody in the more popular Pokémon game. For the sake of demonstrating this application, Pokimin will only focus on the battle aspect of Pokémon with the inclusion of other features.

## Battle objective

A battle within the Pokimin application is between two players. Each player’s active minion will fight in this battle. The objective of this battle is to reduce the opponent’s minion’s health down to 0 before the opponent reduces your minion’s health down to 0. The last minion standing will announce their owner as the winner of the battle.

## Levelling minions

The player is given the opportunity to grow their minions by increasing their stats. This will be measured by the minion’s level. The higher the minion’s level, the more advantageous against the opponent’s minion during a battle. In order to increase the minions level, the player can choose to gain experience points either by battles with other players or by physically walking specific distances. Once the minion has sufficient experience points, the minion will level up. Therefore, increasing the stats of the minion.

# End User Instructions

## Initiating

The application can be accessed once it has been installed in a device. When the user launches the device, the user is directed to the main activity. If the user has recently installed the application, a dialog will appear to allow the user to register as a new player. Once the player has successfully registered, they will be directed to the main menu.

## https://scontent-syd1-1.xx.fbcdn.net/v/t35.0-12/13321243_10206420169889601_1624042390_o.png?oh=4ce85e89cfd9b82ec6ded7043890c82a&oe=574FAEA4Main Menu

Within the main menu, The player is provided 3 options:   
- Player – Directs the player to the player activity.  
- Minion – Directs the player to the minion activity.  
- Battle- Directs the player to the battle activity.

- Training- Directs the player to the training activity.

The player can choose one of these options by pressing them on their screen. Doing so will execute a function depending on what the player has chosen.

## Player Activity

The player will be directed to the player activity when selected from the main menu. This activity will provide details of the player. This includes their name, and battle history in terms of wins and losses. The player can backtrack once finished with this activity. Doing so will direct them back to the main menu.

## Minion Activity

The player will be directed to the minion activity when selected from the main menu. This activity will provide details about the player’s minion. This includes their name, element, level, experience, health, defence, attack, and speed

Once the player is done with the minion activity, the player can backtrack using the device’s back button. Doing so will direct the player back to the main menu.

## Battle Activity

The player will be directed to the battle activity when selected from the main menu. This activity simulates battles against other players who have the same application. The players must be able to connect through Bluetooth in order to battle each other.

### Search

Before the players can battle each other, they must first be able to connect to each other using Bluetooth. To do this, the player must search for a device that provides the correct service, which is the application. If the service is not found, the player can become the service, providing service for other devices, allowing others to connect to the player.

### https://scontent-syd1-1.xx.fbcdn.net/v/t35.0-12/13313706_10206420170329612_1633469317_o.png?oh=d8c98bca716809844228b21b716a343b&oe=57500A00Battle

Once a connection has been established, the player can now battle with his opponent. The battle is round-based, meaning each player must make an action before the next round begins. In each round, the player can select a skill for the minion to perform. This can be selected by the four skills selected at the bottom of the screen. Once a player has selected a move, they must wait for their opponent to also select a move. The result of these skills will show once the round has ended. The process will repeat until a minion has been eliminated from the game. The last minion standing announces their owner as the winner.

### Result

Once a battle is finished, both players will be directed to the result screen. This will show whether a player has won or lost from their previous battle. Once the player is finished with this screen, they can press the device’s back button to return to the main menu.

## Training Activity

The player will be directed to the training activity when selected from the main menu. This activity will enable the user to train the minion by walking certain distances.

The train button will be presented to the player to allow them to gain experience. The player must travel a certain distance in order for the device to generate experience for their minion. Note: the generated experience will occur in intervals to prevent players from gaining excessive experience in short time periods.

The player can backtrack using the device’s back button. Doing so will direct the player back to the main menu.

# Services

The two services being implementing into the Pokimin application are:

* Bluetooth service: Messages are sent from one device to another to simulate a player verses player battle.
* Location-based service: Used to generate experience for the player’s active minion. A player must travel a specific distance in order to gain experience. Though, this calculation occurs during intervals to prevent the player from gaining excessive experience in a short time period.

## Bluetooth

This application uses client-server communication when simulating a battle between two devices. Both roles may seem similar in terms of the messages being passed between the client and the server, but the server contains more functionalities.

As a player is directed to the battle activity, they start as a client. This is because it is intended for a client to discover a server for efficiency. If the client fails to discover a server, they have the choice to become a server themselves or continue discovering. Choosing to become the server will convert the device from being a client to being a server.

### Client

The client uses a custom BroadcastReceiver class to discover Bluetooth devices. This class necessary actions that relate to discovering a Bluetooth device. These actions include:  
- BluetoothDevice.ACTION\_FOUND: registers the found device and adds it into an ArrayList that  
 stores all found devices  
- BluetoothAdapter.ACTION\_DISCOVERY\_STARTED: does not perform any special functions other  
 than notifying the player that the discovery has started.  
- BluetoothAdapter.ACTION\_DISCOVERY\_FINISHED: notifies the devices that the discovery has   
 been completed, allowing them to continue with their run() method.

Once the device discovery has been completed, the client will try to connect to each device, investigating if any device has the correct service. This is done by attempting to create a BluetoothSocket with the discovered devices using the same SERVICE\_UUID. Once a socket has been successfully established, the communication between the client and the server commences.

Messages can be send to and received from the server. These two functions are run in different threads to reduce conflict of sending and receiving messages. The client sends two types of messages:  
- InitMessage: Sends the player’s information to the server and register the client as the  
 server’s opponent.  
- SkillMessage: Sends the selected skill from the client. This includes the information of the skill as  
 well as the minion’s information to enable the server to calculate the outcome of the skill.

The client can also receive messages from the server. These messages include:  
- InitMessage: to register the server as an opponent of the client during a battle. This will modify  
 the view to change the opponent’s name.  
- BattleMessage: the result of the round after each player has selected a skill. This will modify the  
 view to show a message in the middle of the screen and may change the ProgressViews of the  
 two minions.  
- ResultMessage: will notify the device to indicate the battle is over. The player will be directed to  
 the result screen and show whether the player has won or lost the game.

### Server

The server is very similar to the client, but includes many functions to calculate skills and send messages to the client providing information such as round results and who has won the battle.

As a server, they can become discoverable for 30 seconds to allow clients to discover this device as a server. A BluetoothServerSocket is used to accept connections from the client that is searching for the same SERVICE\_UUID that has been registered to the server socket. Once a connection has been successful, the server starts two threads to receive and respond to the client.

Messages being sent to the client is similar to the messages being send from the client to the server. This is because both client and server are versing each other during a battle. Therefore, should have similarities in messages being sent.

When receiving messages from the client, the server processes these messages and sends an additional response to the client. This is because it is the server’s responsibility to calculate the results of the battle and notify both the client and server of these results. The responding messages are as follows:  
- BattleMessage: This response is sent once the client has sent a SkillMessage to the server and   
 also having the server choose a skill to use. The skills chosen from both the client and the server  
 calculated on the two minions, depending on the chosen skill. Once the calculation is complete,  
 it updates the view of the server side and sends a BattleMessage to the client side to update   
 the view of the client side.  
- ResultMessage: This response is sent after the calculation of the SkillMessage has been  
 completed, and a minion’s health has been reduced to 0 or below. This will change the screen  
 of the server to the Result screen to show whether the server has won or lost the battle. A   
 ResultMessage is also sent to the client to change their screen to indicate either the client has  
 won or lost the battle.

## Location-based service.

This application uses location-based service when training the minion. When the player starts the training, the device turns on location-based service and gets the physical location via GPS provider. Because GPS signal can be inaccurate inside buildings especially at high floors, it is recommended that the player starts the training outdoors, or at least at ground floor whenever possible.

The location-based service uses LocationManager class to request location updates at set intervals, and calls a LocationListener every time the interval has passed. Ten metres must be passed within an interval of twenty seconds in order to activate the LocationListener. This is done in order to minimize unintended LocationListener calls by GPS location signal fluctuations.

All these location-based service methods are implemented inside a LocationController custom class. When the training activity starts, a LocationController instance is created and grabs the location data with LocationManager. It then saves the current location into the database so that changes are saved. All experience points gathered from training are permanently saved.

### Client

The client uses a custom BroadcastReceiver class to discover Bluetooth devices. This class necessary actions that relate to discovering a Bluetooth device. These actions include:  
- BluetoothDevice.ACTION\_FOUND: registers the found device and adds it into an ArrayList that  
 stores all found devices  
- BluetoothAdapter.ACTION\_DISCOVERY\_STARTED: does not perform any special functions other  
 than notifying the player that the discovery has started.  
- BluetoothAdapter.ACTION\_DISCOVERY\_FINISHED: notifies the devices that the discovery has   
 been completed, allowing them to continue with their run() method.

Once the device discovery has been completed, the client will try to connect to each device, investigating if any device has the correct service. This is done by attempting to create a BluetoothSocket with the discovered devices using the same SERVICE\_UUID. Once a socket has been successfully established, the communication between the client and the server commences.

Messages can be send to and received from the server. These two functions are run in different threads to reduce conflict of sending and receiving messages. The client sends two types of messages:  
- InitMessage: Sends the player’s information to the server and register the client as the  
 server’s opponent.  
- SkillMessage: Sends the selected skill from the client. This includes the information of the skill as  
 well as the minion’s information to enable the server to calculate the outcome of the skill.

The client can also receive messages from the server. These messages include:  
- InitMessage: to register the server as an opponent of the client during a battle. This will modify  
 the view to change the opponent’s name.  
- BattleMessage: the result of the round after each player has selected a skill. This will modify the  
 view to show a message in the middle of the screen and may change the ProgressViews of the  
 two minions.  
- ResultMessage: will notify the device to indicate the battle is over. The player will be directed to  
 the result screen and show whether the player has won or lost the game.

### Server

The server is very similar to the client, but includes many functions to calculate skills and send messages to the client providing information such as round results and who has won the battle.

As a server, they can become discoverable for 30 seconds to allow clients to discover this device as a server. A BluetoothServerSocket is used to accept connections from the client that is searching for the same SERVICE\_UUID that has been registered to the server socket. Once a connection has been successful, the server starts two threads to receive and respond to the client.

Messages being sent to the client is similar to the messages being send from the client to the server. This is because both client and server are versing each other during a battle. Therefore, should have similarities in messages being sent.

When receiving messages from the client, the server processes these messages and sends an additional response to the client. This is because it is the server’s responsibility to calculate the results of the battle and notify both the client and server of these results. The responding messages are as follows:  
- BattleMessage: This response is sent once the client has sent a SkillMessage to the server and   
 also having the server choose a skill to use. The skills chosen from both the client and the server  
 calculated on the two minions, depending on the chosen skill. Once the calculation is complete,  
 it updates the view of the server side and sends a BattleMessage to the client side to update   
 the view of the client side.  
- ResultMessage: This response is sent after the calculation of the SkillMessage has been  
 completed, and a minion’s health has been reduced to 0 or below. This will change the screen  
 of the server to the Result screen to show whether the server has won or lost the battle. A   
 ResultMessage is also sent to the client to change their screen to indicate either the client has  
 won or lost the battle.