UNIVERSIDAD DE MÁLAGA

Programación Bluetooth en Java

OBJETIVO

Programación en Java de aplicaciones sobre Bluetooth utilizando la API BlueCove (disponible en el campus virtual).

Se realizarán ejercicios que pongan en práctica el descubrimiento de dispositivos y la búsqueda de servicios, así como la comunicación entre un cliente y servidor a través de distintas aplicaciones Bluetooth desarrolladas en Java.

En todas las aplicaciones desarrolladas se deberán gestionar errores, y se deberá solicitar al usuario la información pertinente y mostrar los mensajes oportunos al usuario. La forma en que se implemente cada ejercicio (ej. cada ejercicio en un método o una clase) queda a elección del alumno, siempre que se especifique claramente.

Para la entrega de la práctica se requiere el código Java junto con una pequeña memoria descriptiva con los posibles comentarios, problemas encontrados, capturas de pantalla o resultados de su funcionamiento.

INFORMACIÓN DEL DISPOSITIVO DE COMUNICACIÓN BLUETOOTH

1. Utiliza la documentación de la API y los métodos de la clase `LocalDevice` para obtener información acerca de la configuración de tu dispositivo Bluetooth local ¿Qué información puedes obtener?

DESCUBRIMIENTO DE DISPOSITIVOS REMOTOS

2. Implementación de una aplicación que permita descubrir todos los dispositivos bluetooth visibles cercanos. Muestra su nombre y su dirección Bluetooth.
3. Implementación de una aplicación que permita descubrir un dispositivo concreto (por dirección Bluetooth y/o *friendly name*).

DESCUBRIMIENTO DE SERVICIOS

4. Implementación de una aplicación que permita descubrir todos los servicios (de clase público) de todos los dispositivos cercanos.
5. Implementación de una aplicación que permita descubrir todos los servicios (de clase público) de un dispositivo concreto (especificado por dirección Bluetooth y/o *friendly name*).
6. Implementación de una aplicación que permita descubrir un servicio con un nombre concreto en un dispositivo concreto.

Haz uso de los siguientes UUIDs en la búsqueda de servicios (parámetros de SearchServices para especificar los atributos que requerimos de cada servicio buscado y la clase de servicio):

```
static int SERVICE_NAME_ATTRID = 0x0100;
…
// service search parameters
UUID uuids[] = new UUID[1];
uuids[0] = new UUID(0x1002); //PublicBrowseGroup
//uuids[0] = new UUID(0x1101);
//SerialPort (necesario para el siguiente ejercicio)

int attridset[] = new int[1];
attridset[0] = SERVICE_NAME_ATTRID;
```

Puedes consultar la especificación de los UUIDs en Bluetooth:

https://www.bluetooth.com/specifications/assigned-numbers/service-discovery/

COMUNICACIÓN CLIENTE/SERVIDOR CON BLUETOOTH A TRAVÉS DE RFCOMM

7. Implementa una aplicación cliente-servidor que permita el envío de mensajes de texto entre ellos (como un chat textual). Ambos procesos (cliente y servidor) son procesos Java distintos que se ejecutan en máquinas diferentes para comprobar la comunicación inalámbrica.

   Por un lado, la clase del servidor ya implementada se encuentra disponible en el Campus Virtual (Servidor.java). Este publica el servicio de chat con la siguiente URL:

```
String url = "btspp://localhost:" + new UUID(0x1101).toString() +
   ";name=chat";
```

   Desde el cliente, tras encontrar al dispositivo servidor, encuentra el servicio de chat (de clase puerto-serie) a través del nombre que el servidor le ha asignado (en el ejemplo a continuación, “chat”). Finalmente, usa su URL de conexión para conectarte desde el cliente y poder intercambiar información (echa un vistazo a los apuntes para identificar la función para obtener esa URL a partir del registro de servicio, una vez que lo encontramos).

   Puedes reutilizar en parte el código desarrollado en los apartados anteriores (para la búsqueda del servidor y del servicio), y basarte en el código del servidor para el envío/recepción de texto.

   Ayuda: en caso de no disponer de dos máquinas para realizar el ejercicio, es posible usar el emulador de la pila de protocolos proporcionado también por la distribución de Bluecove (con el fichero **bluecove-emu.jar**, disponible en el Campus Virtual).

Concretamente, importa el nuevo fichero jar en el proyecto Java y utiliza como base la clase **EmulatorChatServer** (también disponible en el Campus Virtual).

Si echas un vistazo a cómo está estructurado el código, verás que el programa ejecuta un thread para el servidor bluetooth (que en este caso es emulado para que funcione en la misma máquina que el cliente, y cuya funcionalidad es la misma que la de la clase Servidor.java).

En este caso, solo hay que completar el método **startClient()**, que representa el thread del cliente y ya posee implementada la búsqueda de dispositivos y servicios.

8. **Ejercicio opcional:** una vez que tengas desarrollada la funcionalidad del cliente, integra su funcionalidad (y la del servidor) con una interfaz gráfica para probar la comunicación entre ambos interlocutores.

   Para ello, importa el fichero ChatUI.jar disponible en el Campus Virtual, que contiene las clases necesarias para usar la interfaz (ya implementada). La clase **EjemploChatWindow**, por su parte, hace uso de esta interfaz y sirve como plantilla donde integrar el código Java de cliente y servidor (en dos clases independientes).

   Como alternativa al uso de dos equipos, al igual que ocurre con el ejercicio 7, puedes hacer uso del emulador de la pila de protocolos de Bluetooth para ejecutar el cliente y el servidor en la misma máquina, y además integrar una interfaz gráfica. Para ello, puedes basarte en la clase **EmulatorChatUI** disponible en el Campus Virtual (a importar en un proyecto donde también se incluya la librería del emulador y la interfaz gráfica), donde la funcionalidad del cliente queda implementada en el método **startClient()**.

   Echa un vistazo al Campus Virtual para más instrucciones sobre este ejercicio y descargarte estos recursos.

REFERENCIAS

[1] Bluecove: Java library for Bluetooth (con ejemplos de código): http://bluecove.org/

[2] Documentación de BlueCove: http://www.bluecove.org/bluecove/apidocs/