Smart Plug

App

Arquitectura

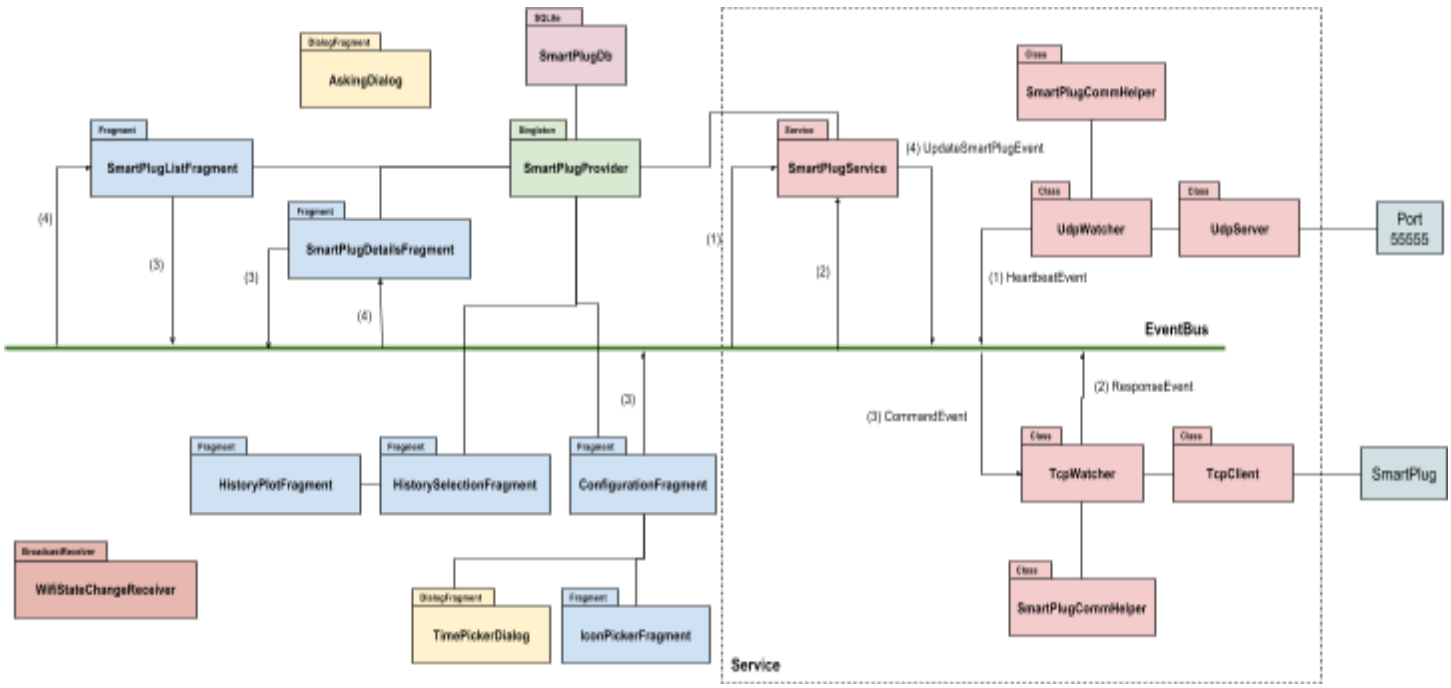
Autor
Ing. Mariano Mondani

Tabla de contenidos

Registro de cambios

Revisión	Cambios realizados	Fecha
1.0	Creación del documento	14/09/2016

Arquitectura App móvil



Servicio

Base de datos

La base de datos **SmartPlugDb** se va a implementar mediante SQLite consta de 4 tablas:

STATICINFO		
ID	**NAME**	**ICON_ID**
STRING	STRING	INT

ONOFFTIMES						
ID	**ENABLED_TIMES**	**MONDAY_LOAD_ON_TIME**	**MONDAY_LOAD_OFF_TIME**	**....**	**SUNDAY_LOAD_ON_TIME**	**SUNDAY_LOAD_OFF_TIME**
STRING	INT	STRING	STRING	….	STRING	STRING

MEASUREMENTS			
ID	**DATE**	**MEASUREMENT_TYPE**	**MEASUREMENTS**
STRING	STRING	INT	STRING

INSTANTANEOUSINFO									
ID	**IP**	**LAST_UPDATE**	**CONNECTION _STATE**	**LOAD_ STATE**	**VOLTAGE**	**CURRENT**	**POWER**	**TOTAL_ ENERGY**	**TIMEOUTS**
STRING	STRING	LONG	INT	INT	FLOAT	FLOAT	FLOAT	FLOAT	INT

Nombre: StaticInfo

Campos:

- id (string): es el ID único de cada SmartPlug. Por ejemplo 123456.
- name (string): nombre personalizado que se le asignó al nodo. Es el nombre que se va a mostrar en la aplicación.
- icon_id (integer): es el número de recurso del ícono elegido para representar a al Smart Plug.

Nombre: OnOffTimes

Campos:

- id (string): es el ID único de cada SmartPlug. Por ejemplo 123456.
- enabled_times (integer): es un byte que indica qué días están habilitados para la programación horaria. El bit 0 corresponde al Domingo y el bit 6 al Sábado.
- monday_load_on_time (string): indica la hora a la que se debe encender la carga el día Lunes. El formato es: HH:MM.
- monday_load_off_time
- tuesday_load_on_time
- tuesday_load_off_time
- wednesday_load_on_time
- wednesday_load_off_time
- thursday_load_on_time
- thursday_load_off_time
- friday_load_on_time
- friday_load_off_time
- saturday_load_on_time
- saturday_load_off_time
- sunday_load_on_time

- sunday_load_off_time

Nombre: Measurements

Campos:

- id (string): es el ID único de cada SmartPlug. Por ejemplo 123456.
- date (string): fecha a la que corresponden las mediciones históricas. El formato es: DD/MM/AA.
- measurement_type (integer): indica el tipo de medición:
  - 1: mediciones de potencia activa por hora.
  - 2: mediciones de energía por hora.
- measurements (string): contiene las 24 mediciones correspondientes a las 24 horas del día indicado en date. Están separadas por un "-" y cada medición está expresada como un número decimal. La unidad es Watt o kWh.

Nombre: InstantaneousInfo

Campos:

- id (string): es el ID único de cada SmartPlug. Por ejemplo 123456.
- ip (string): IP en la que se encuentra el Smart Plug.
- last_update (long): fecha y hora en la que se recibió el último heartbeat desde el Smart Plug. El formato es: DD/MM/AA - HH:mm:ss.
- connection_state (integer): indica el estado de la comunicación con el Smart Plug:
  - 0: comunicación OK.
  - 1: error en la comunicación.
- load_state (integer): indica si el Smart Plug tiene encendida la carga o no:
  - 0: carga apagada.
  - 1: carga encendida.
- voltage (float): último valor de tensión medido. Está expresado como un número decimal. La unidad es Volt.
- current (float): último valor de tensión medido. Está expresado como un número decimal. La unidad es Ampere.
- power (float): último valor de potencia activa medido. Está expresado como un número decimal. La unidad es Watt.
- total_energy (float): energía total acumulada por el Smart Plug. Está expresada como un número decimal. La unidad es kWh.
- timeouts (integer): cantidad de timeouts que hubo en la comunicación TCP. Se incrementa en 1 cada vez que hay un timeout o un error al intentar enviar un comando y se resetea a 0 cada vez que se envìa un comando exitósamente.

Eventos

Los distintos componentes de la App se van a comunicar a través de **EventBus**, librería desarrollada por greenrobot. Los eventos que se intercambian son instancias de clases de java.
Las clases que representan estos eventos son:

Nombre: HeartbeatEvent
Descripción: Se genera cada vez que llega un heartbeat de algún Smart Plug. Tiene la principal función de indicar que sigue existiendo el Smart Plug y de actualizar la IP en la que se lo puede encontrar.
Variables: ● id (string): ID del Smart Plug que envió el heartbeat. Formato: 123456. ● ip (string): dirección IP desde la que llegó el heartbeat. ● date (date): fecha y hora en la que llegó el heartbeat.

Nombre: ResponseEvent
Descripción: Se genera cuando se recibe la respuesta a un comando enviado a un Smart Plug.
Variables: ● id (string): ID del Smart Plug que envió la respuesta. ● data (byte[]): payload que acompaña a la respuesta recibida.

Nombre: CommandEvent
Descripción: Se genera cada vez que se debe enviar un mensaje a un SmartPlug.
Variables: ● id (string): ID al que se debe enviar el comando. ● data (byte[]): data a cargar como payload del paquete TCP.

Nombre: UpdateSmartPlugEvent
Descripción:

Se genera cada vez que se actualiza la información de un nodo.
Variables: ● id (string): ID del Smart Plug cuya información fue actualizada.

Nombre: AllMessagesSentEvent
Descripción: Se genera cuando se terminan de enviar todos los mensajes TCP hacia los Smart Plugs.
Variables:

Nombre: TcpTimeoutEvent
Descripción: Se genera cuando se produce un timeout o un error en la conexión con un Smart Plug al enviarle un comando.
Variables: ● id (string): ID del Smart Plug que produjo el timeout o el error.

Nombre: WifiStateUpdate
Descripción: Se genera cuando se produce un cambio en la conexión a WiFi, ya sea que se conectó a una red o que se desconectó.
Variables: ● state (boolean): true si está conectado a WiFi. false en caso contrario.