FIC – UDC – 1º cuatrimestre 17/18

Proyecto Programación de Sistemas

EasyArduino!

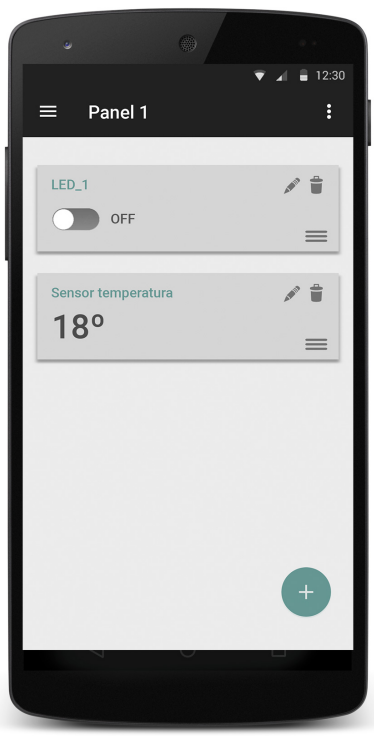
Jose Luis Álvarez Quiroga – joseluisalvquiroga@gmail.com
Matías Cubilla Currás – matias.cubillac@udc.es

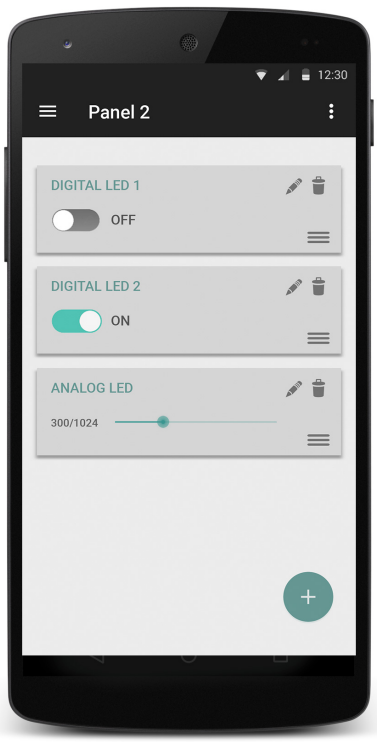
INTRODUCCIÓN

Esta app permitirá controlar una serie de componentes tales como LEDs, servo motores, diversos sensores, etc... conectados a un Arduino. Para ello, se mandarán comandos por puerto serie a través de USB desde nuestro dispositivo Android.

Nuestra aplicación dispondrá de dos elementos principales, a los que llamaremos paneles y controladores. Los paneles podemos entenderlos como mandos a distancia que incluyen diversos botones. Los controladores serían en ese caso cada uno de los botones, permitiéndonos interactuar con los distintos componentes que estén conectados al Arduino.

A continuación se muestra un prototipo de la aplicación donde podemos ver dos paneles, el de la izquierda con dos controladores, y del de la derecha con tres controladores:





Con esta aplicación pretendemos conseguir que el usuario pueda tener distintos arduinos con distintos circuitos en cada uno, pero con el mismo código fuente, y que pueda controlarlos desde la aplicación. Así mismo, también podría modificar el circuito de uno de ellos, sin tener que modificar el código .ino.
Hay otras aplicaciones que cumplen estos requisitos, pero creemos que la sencillez de uso de una aplicación como ésta debe ser algo esencial.

ANÁLISIS PRELIMINAR DE REQUISITOS

Las funcionalidades que se van a implementar, se dividen entre básicas y accesorio, ordenadas en cuánto a su prioridad.

Funcionalidades básicas

T1. Creación de la infraestructura de conexión Android-Arduino.
T2. Paso de mensajes a través de la infraestructura anteriormente creada.
T3. Implementación de controladores de envío de datos (digitales y analógicos)
T4. Gestión de controladores (Crear, modificar y eliminar)
T5. Implementación de paneles
T6. Gestión de paneles (Crear, modificar y eliminar)
T7. Almacenamiento de preferencias de usuario (paneles creados, controladores... etc)
T8. Implementación de controladores de lectura de datos (digitales y analógicos)

Dependencias:

T1->T2
T3->T4->T8
T5->T6

Funcionalidades accesorio

A1. Cerrar/Abrir conexiones serie con el Arduino al desconectar/conectar USB.
A2. Gestión dinámica de los paneles creados.
A3. Gestión dinámica de los controladores creados.
A4. Retocar interface y acciones para tener un aspecto más fluido.

Dependencias:

T1->A1
T4→A2

Posibles funcionalidades extra

- **Gráficas** con valores leídos de Arduino en **tiempo real**

  Crear gráficas para monitorizar en tiempo real los valores de los sensores. Solo contendrán los valores recibidos desde que se creó la gráfica, no se almacenarán datos en el dispositivo Android. Para acceder a dichas gráficas, los controladores marcados como de lectura analógica, dispondrán de un botón que creará esta nueva actividad.

- Creación de **código para el Arduino**

  Dispondremos para nuestra aplicación de una opción que permita crear el archivo .ino que ha de subirse al Arduino para poder comunicarse correctamente con la aplicación Android. Para ello, descargar el archivo en el propio teléfono Android no tendría mucho sentido, por lo que estamos planteándonos cual es la mejor opción.

PLANIFICACIÓN INICIAL

Iteraciones

El proyecto tendrá una evolución iteracional, añadiendo nuevas funcionalidades con cada versión de la aplicación.

Para la primera iteración, se implementará toda la parte de comunicación serie mediante Arduino y Android, permitiendo comandos básicos de envío de datos a Arduino, no de lectura. Se crearán también los primeros 'controladores' dentro de un panel de prueba. No se tendrá en cuenta el estado del UI ni UX de la aplicación para esta primera iteración.

En la segunda iteración del proyecto, nos centraremos en la gestión de la interfaz de usuario. Es decir, se le permitirá al usuario crear, modificar o eliminar paneles y modificar o eliminar controladores (la funcionalidad de crear paneles ya estará implementada en la iteración 1). También se almacenará en memoria los paneles y controladores creados por el usuario.

En la tercera iteración, se crearán controladores que permitan lectura de datos de Arduino en tiempo real.

En posteriores iteraciones, se llevarán a cabo las siguientes tareas en el orden descrito:

- Permitir al usuario modificar o eliminar un panel
- Permitir al usuario modificar o eliminar los controladores dentro de un panel
- Crear un UI y UX
- Funcionalidades extras descritas más adelante en este documento

Responsabilidades

Jose Luis Alvarez

- Paso de mensajes Android-Arduino
- Interfaz de usuario y UX
- Extras : Gráficas en tiempo real

Matías Cubilla

- Gestión de paneles y controladores
- Almacenamiento de datos en memoria del teléfono
- Broadcast Receivers
- Extras : Creación de código Arduino

CONSIDERACIONES DE DESARROLLO

Comunicación Android-Arduino

En nuestra aplicación, la comunicación entre Arduino y Android se llevará a cabo a través de USB. Sin embargo, en un futuro esto podría cambiar, e implementarse la comunicación mediante otras tecnologías como, por ejemplo, Bluetooth o wifi. Para permitir esta escalabilidad, crearemos un paquete llamado 'arduinomanager', donde se incluirá una interfaz, cada una de las implementaciones de dicha interfaz para cada tecnología usada (USB en nuestro caso) y clases adicionales para códigos de respuesta o interfaces que permitan crear listeners de la comunicación.

Protocolo de comunicación serie por USB

Para poder utilizar la comunicación serie por USB del dispositivo Android usaremos una librería externa, usbserial creada por felHR85 disponible en GitHub.

Arquitecturas utilizadas

En cuanto a arquitecturas utilizadas en nuestra aplicación, dispondremos de al menos dos actividades, una donde se muestren todos los paneles disponibles, y otra donde se muestren los controladores dentro de un panel.

Usaremos también broadcast receivers, uno escuchará las conexiones y desconexiones USB e iniciará o cancelará automáticamente la conexión con el dispositivo conectado, sólo en caso de que sea un Arduino. Otro broadcast receiver lo usaremos para gestionar la respuesta a los permisos para acceder a dispositivos USB.

Haremos uso también de las siguientes arquitecturas:
- Menús contextuales para modificaciones y eliminaciones de paneles y controladores.
- Fragments para disponer de layouts dinámicos en ciertas partes de la aplicación.
- Preferencias compartidas y MySQLite para almacenamiento de datos
- Threads para tareas que conlleven una carga de trabajo elevada, a pesar de que la librería 'usbserial' ya se encarga de gestionar la comunicación serie por USB de forma asíncrona.

EVOLUCIÓN DEL PROYECTO

PRIMERA ITERACIÓN
En esta primera iteración del proyecto disponemos de una aplicación con funcionalidades básicas. Al conectar mediante USB un Arduino a nuestro dispositivo, la aplicación se lanzará y se iniciará la conexión. Una vez hecho esto, si el Arduino dispone de nuestro código, podremos crear controladores de envío de datos, tanto analógicos como digitales, completamente operativos.

Funcionalidades implementadas:

- Creación infraestructura básica de la conexión con Arduino
- Definir la estructura de los mensajes.
- Código .ino para el Arduino que se encarge de gestionar las peticiones.
- Envío de mensajes a través de la infraestructura
- Implementación de un panel estático inicial.
- Implementación de dos tipos de controladores de envío de datos (analógico y digital)
- Broadcast Receiver para conexiones y desconexiones USB

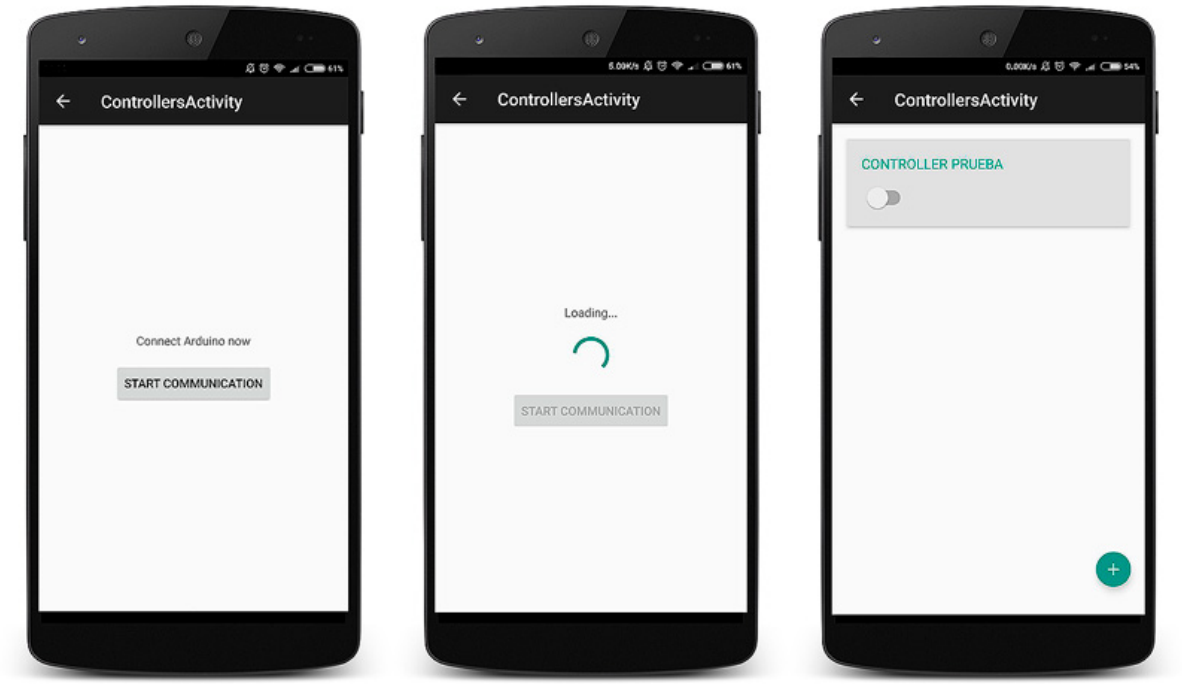
Hito: Entrega 09/11/2017.

Entregable:

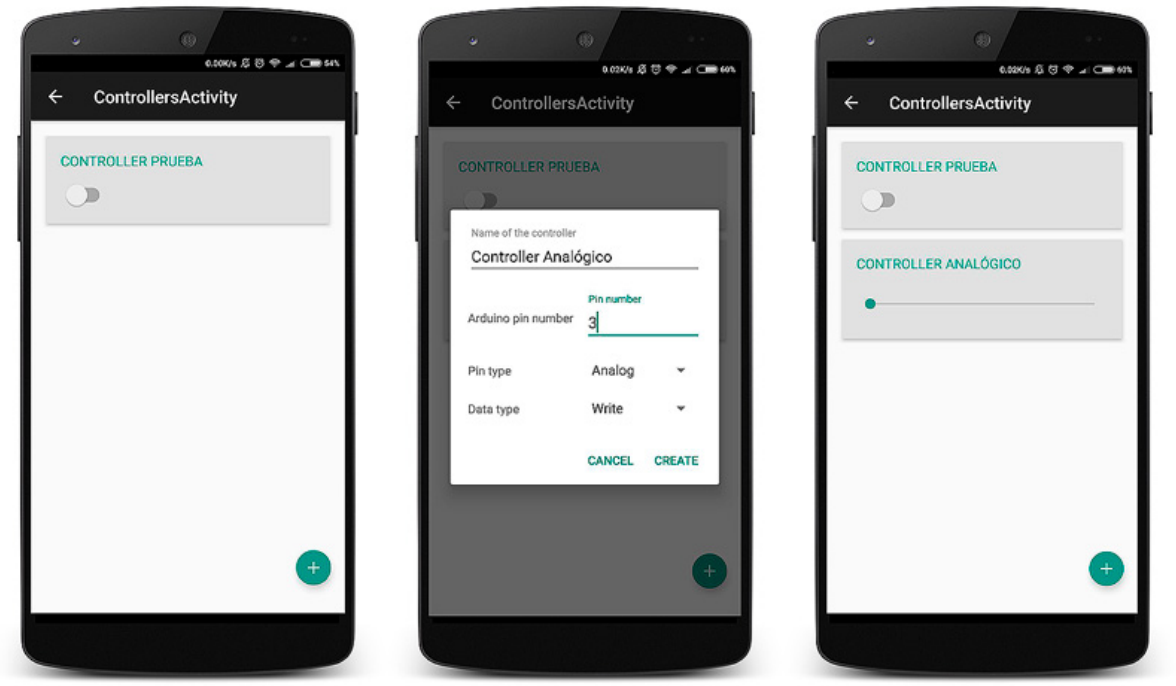
- Documentación : Este archivo
- Aplicación : EasyArduino-v0.0.1.apk

Pruebas llevadas a cabo:

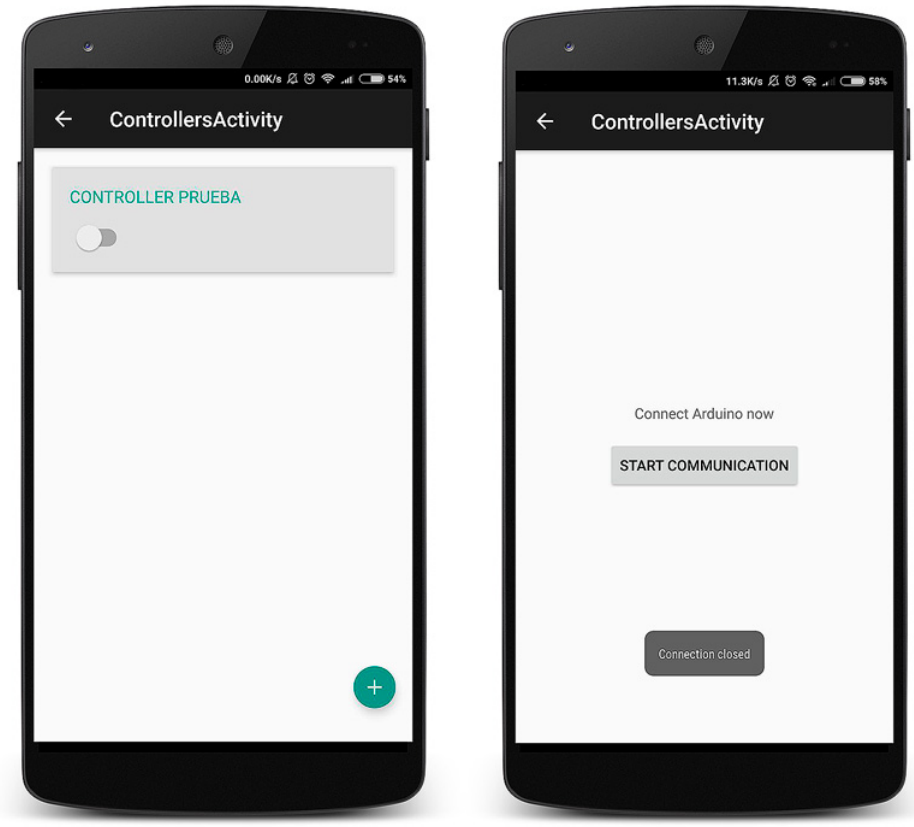
- Establecimiento de **conexión** correcta.



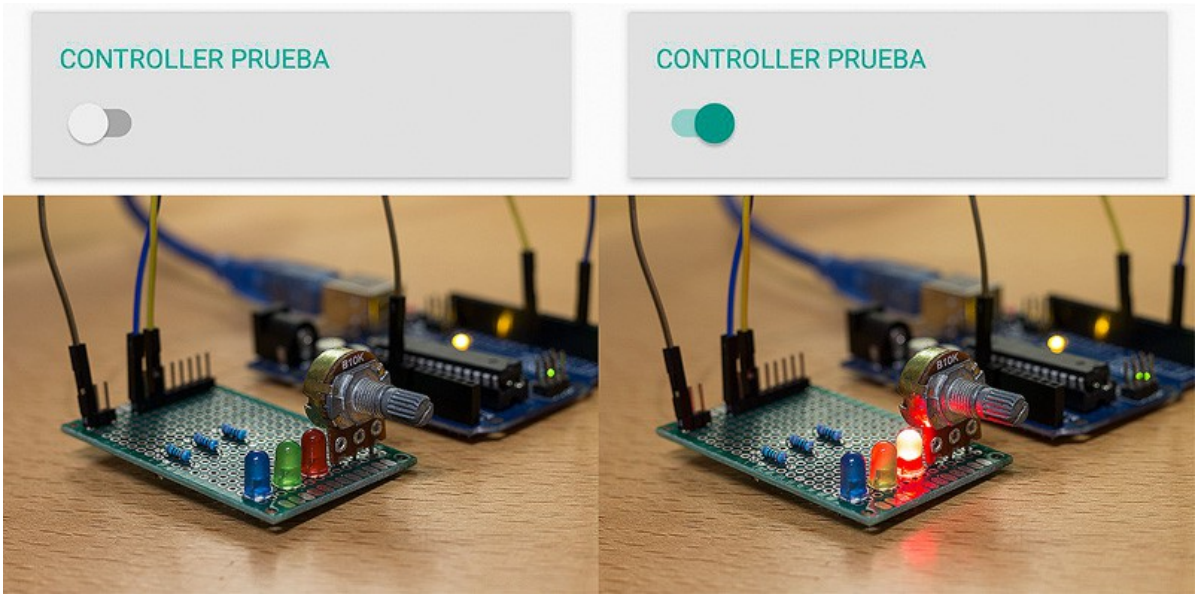
- **Creación** correcta de **controladores**



- Funcionamiento de **broadcast Receiver** tanto para abrir como cerrar la conexión (En la imagen se muestra como al desconectar el Arduino la conexión se cierra automáticamente)



- Funcionamiento **controlador digital** con LED.



- Funcionamiento **controlador analógico** con LED