# Ejercicio 1:

Desarrollar una aplicación que al apretar un botón:

- Imprima en el logcat el mensaje “El usuario apretó el botón”.
- Se muestre un Toast con el mensaje “Hola mundo!”

# Ejercicio 2:

Desarrollar una aplicación con dos Activity.

- La primera de ellas tendrá:
    - Un botón que permita obtener los datos de un contacto del dispositivo, y mostrarlos en un un TextView.
    - Un botón que lance la segunda Activity, que va a devolver el resultado de un multiplicación para que se muestre en un TextView.
- La segunda Activity tendrá:
    - 2 TextView que permitan el ingreso de dos valores.
    - 1 botón que calcule el producto de los valores ingresados anteriormente.
    - La Activity debe finalizar retornado el resultado de la multiplicación a la Activity que la invocó.

En ambos casos, se debe resguardar y restaurar los estados de las Activity para que al girar la pantalla del dispositivo no se pierdan los valores de los TextView.

# Ejercicio 3:

Desarrollar un segundero con tres botones y un texto, donde el primer botón sea un “comenzar” y los segundos comiencen a acumularse, luego un botón “parar” que detenga la cuenta, y por último un botón “resetear” que permita resetear el acumulador. El proceso de acumulación de segundos deberá realizarse con una AsyncTask. Por último, si la aplicación es cerrada y vuelta abrir, mediante la utilización de SharedPreferences el contador debe estar en el último estado dejado por el usuario.

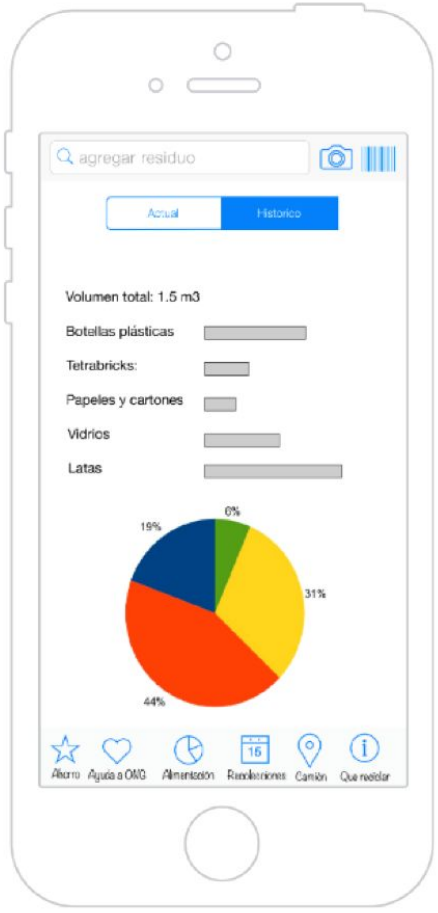
# Ejercicio 4

Se requieren implementar dos Background Servicios, el primero que extienda de IntentService y el segundo que extienda de Service. La lógica de negocio para estos servicios será la misma, recibirán en un intent un número de iteración, se domirá el thread por 3 segundos y luego se enviará un broadcast local con el número de iteración recuperado del Intent recibido.
Se deberá implementar una activity con 2 botones, que realizarán 4 llamadas a los servicio cada uno, y 2 campos de texto que mostran la iteración reciba mediante Broadcast receiver para cada uno de los servicios.

## Ejercicio 5

Se requiere implementar un Bound Service que permita conexión desde otros procesos/aplicaciones. Este servicio retornará un número aleatorio entre 0 y 100, el cual será enviado y mostrado en la activity del ejercicio anterior, para lo cual habrá que agregar un nuevo campo de texto.
Se deberá realizar una nueva aplicación que se enlace a este servicio y realice la llamada.

## Final




Se requiere implementar una aplicación que menaje y los reciclado de un usuario. Para esto cuando el usuario abra la aplicación podrá registrar un usuario. Lo cual podrá realizar enviando la petición al servidor provisto por la cátedra de esta forma:
Método POST
URL: http://localhost:8080/api/users
Body:
{"firstName":"Mauri","lastName":"Arroqui","mail":"mauriarroqui@gmail.com",**"username":"marroqui2"**,"address":{"department":"Tandil","number":874,"streetAddress":"Alberdi","city":"Tandil","state":"Buenos Aires","zipCode":"7000"}}

Este usuario deberá ser recordado por la aplicación cada vez que se abra nuevamente, a no ser que otro usuario sea registrado y ahora será este el que aparezca por defecto cuando la aplicación es abierta. Por simplicidad de la solución no es necesario que se permita seleccionar con que usuario iniciar.
Notar que el **username** será utilizado luego por todas las solicitudes HTTP.
Luego se podrán registrar reciclados para el usuario actual. El usuario podrá ir almacenando localmente en la aplicación la cantidad reciclada de cada uno de los componentes (bottles , tetrabriks, glass, paperboard, cans) hasta que finalmente decida enviarlo a servidor (el usuario lleva lo reciclado a punto limpio tandil o avisa al camión para que ya pueda pasar a buscarlo), lo cual será realizada enviando la petición al servidor de esta forma:
Método POST
URL: http://localhost:8080/api/users/**marroqui2**/recycling/
Body: {"bottles":1,"tetrabriks":5,"glass":3,"paperboard":4,"cans":2,"date":"2018-11-28"}
La aplicación deberá permitir dos tipos de visualizaciones. La primera que muestre el listado de todos los reciclados, el cual se realizará enviando la petición al servidor:
Método GET
URL http://localhost:8080/api/users/marroqui2/recycling/
La segunda que muestre el total reciclado hasta la actualidad, para lo cual se realizará enviando la petición al servidor:
GET
http://localhost:8080/api/users/marroqui2/total/

## Notas:

1. El servidor se levanta ejecutando java -jar garbage-recycler-0.0.1-SNAPSHOT.jar
2. Acceso a la base de datos en memoria:
   http://localhost:8080/h2-console
   Driver class: org.h2.Driver
   JDBC URL: jdbc:h2:mem:testdb
   User Name: sa
   Password:

Registro de un usuario
Almacenar reciclados
Botellas, tetrabricks, papeles, vidrios, latas.
Submitear un reciclado

Mostrar listas de reciclados
Mostrar un total de todo lo reciclado.