

Aplicaciones Móviles

DOMICIANO RINCÓN

INGENIERÍA TELEMÁTICA
INGENIERÍA DE SISTEMAS
DISEÑO DE MEDIOS INTERACTIVOS

Referenciar Views

ANDROID

Referenciar Views

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    ...
    Button login_btn = findViewById(R.id.login_btn);
    ...
}
```

```
<VIEW> <referencia> =
findViewById(R.id.<IDENTIFICADOR>);
```

JAVA

```
<Button
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginTop="15dp"
    android:layout_marginLeft="50dp"
    android:layout_marginRight="50dp"
    android:textColor="#9F4898"
    android:background="@drawable/back_btn"
    android:text="Login"
    android:id="@+id/login_btn"
    style="@style/Widget.AppCompat.Button.Borderless"
/>
```

```
android:id="@+id/<IDENTIFICADOR>"
```

XML

Listeners

ANDROID

Listeners

GUI Listeners:

Interrupciones del usuario

EJEMPLOS

- x `onClickListener`
- x `onItemClickListener`
- x `onTouchListener`
- x `onKeyListener`

System Listeners:

Interrupciones del sistema

EJEMPLOS

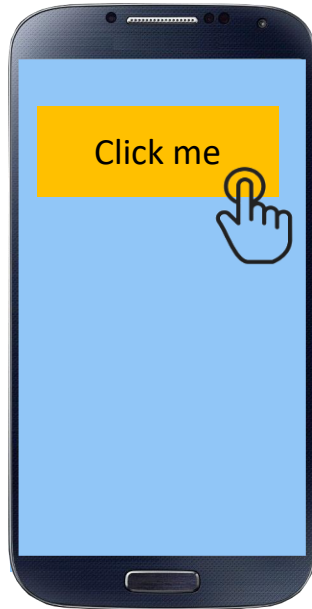
- x `onActivityResult`
- x `onRequestPermissionsResult`

Referenciación



```
Button boton = (Button) findViewById(R.id.boton);
```

OnClickListener



```
Button boton = (Button) findViewById(R.id.boton);  
  
boton.setOnClickListener(new View.OnClickListener() {  
    @Override  
    public void onClick(View v) {  
        accion(v);  
    }  
});
```



Ejecuta el método acción
cuando se toque el botón

OnTouchListener



```
TextView miText = (Button) findViewById(R.id.miText);
miText.setOnTouchListener(new OnTouchListener() {
    @Override
    public boolean onTouch(View v, MotionEvent event) {
        switch (event.getAction()) {
            case MotionEvent.ACTION_DOWN:

                return true;
            case MotionEvent.ACTION_MOVE:

                return true;
            case MotionEvent.ACTION_UP:

                return false;
        }
    }
});
```


OnTouchListener



MotionEvent.ACTION_DOWN:

Ocurre cuando se toca el view.

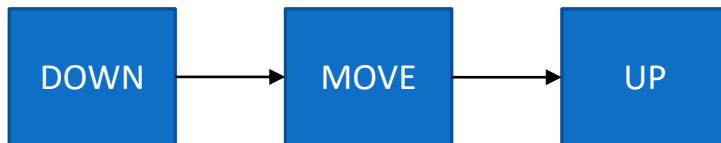
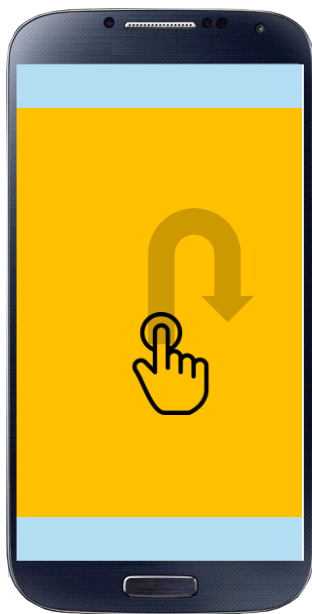
MotionEvent.ACTION_MOVE:

Ocurre cuando se arrastra el dedo luego de ser tocado el View

MotionEvent.ACTION_UP:

Ocurre cuando se levanta el dedo y se deja de tocar el view

OnTouchListener

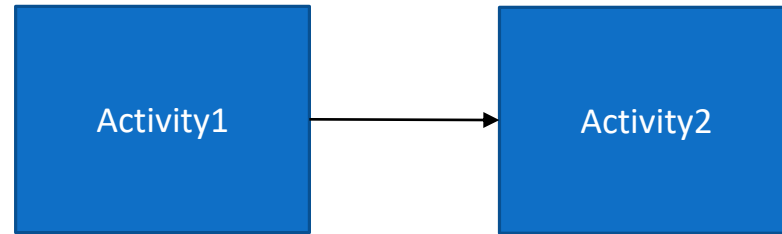


NOTA: Se retorna true para darle continuidad al gesto

Intents

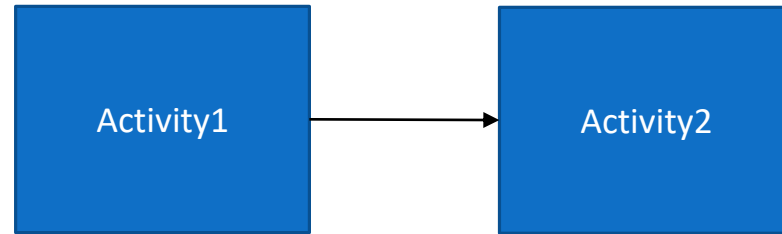
ANDROID

Intent



Se utiliza el *Intent* para navegar de una actividad a otra. Por ejemplo navegar de la actividad 1 a la 2

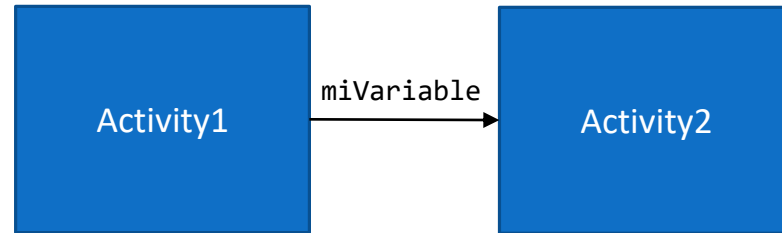
Intent



El código es muy simple. Por ejemplo, estando en la Activity1 se puede ir a la Activity2 así:

```
Intent i = new Intent(this, Activity2.class);  
startActivity(i);
```

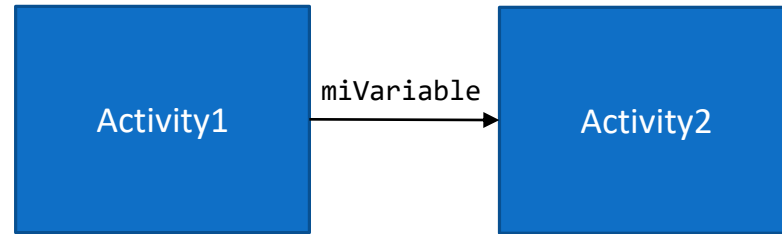
Intent



Se puede usar el intent para enviar variables de una actividad a la otra. Por ejemplo desde Activity1 puedo mandar un entero almacenado en miVariable:

```
Intent i = new Intent(this, Activity2.class);  
i.putExtra("<CLAVE>", miVariable);  
startActivityForResult(i, requestCode);
```

Intent



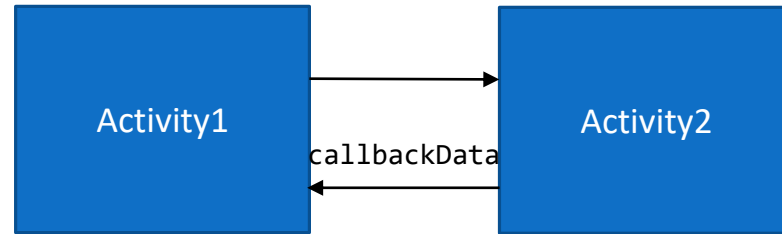
Y desde Activity2 se puede recibir esa variable, por ejemplo, en el onCreate().

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    ...
    int miVariable = getIntent().getExtras().getInt("actualColor");
    ...
}
```

Intents + callbacks

ANDROID

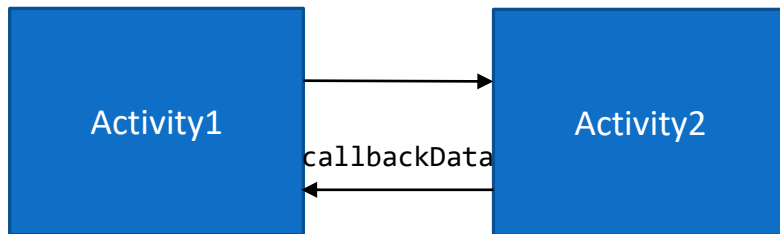
Intent



La Activity1 puede requerir datos a cualquier actividad que llame. Si llama a la Activity2 puede pedirle datos cuando esta última finalice su ejecución.

```
Intent i = new Intent(this, Activity2.class);  
startActivityForResult(i, REQUEST_CODE);
```

Intent

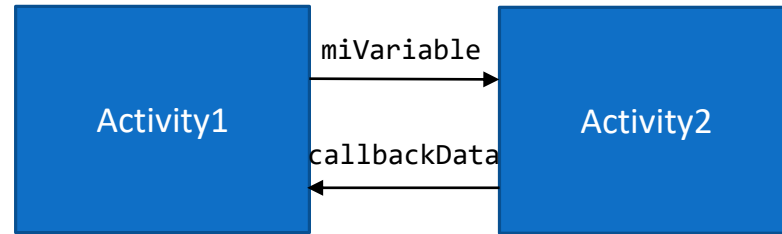


La Activity1 puede requerir datos a cualquier actividad que llame. Si llama a la Activity2 puede pedirle datos cuando esta última finalice su ejecución.

```
Intent i = new Intent(this, Activity2.class);  
startActivityForResult(i, REQUEST_CODE);
```

El REQUEST_CODE es un entero que define el desarrollador.

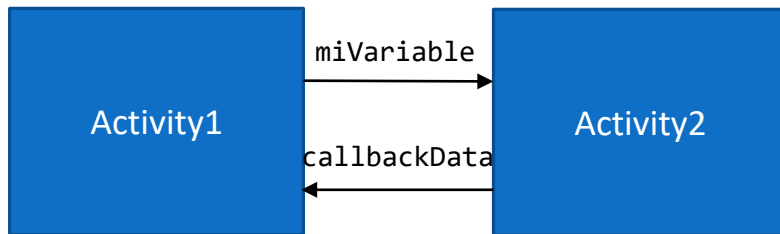
Intent



La Activity2 puede responder mediante un callback a la Activity1 en un evento como el toque de un botón:

```
Intent i = new Intent();  
i.putExtra("respuesta", callbackData);  
setResult(RESULT_OK, i);  
finish();
```

Intent

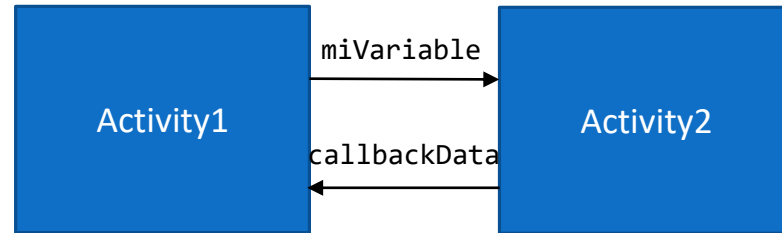


La Activity2 puede responder mediante un callback a la Activity1 en un evento como el toque de un botón:

```
Intent i = new Intent();  
i.putExtra("respuesta", callbackData);  
setResult(RESULT_OK, i);  
finish();
```

El intent nos permite devolver información gracias a los extras.

Intent

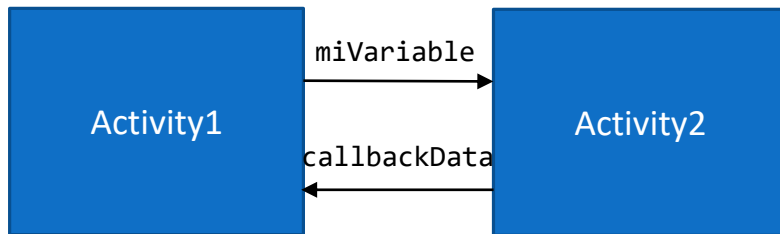


La Activity2 puede responder mediante un callback a la Activity1 en un evento como el toque de un botón:

```
Intent i = new Intent();  
i.putExtra("respuesta", callbackData);  
setResult(RESULT_OK, i);  
finish();
```

El método setResult permite responder con un estado y el intent

Intent

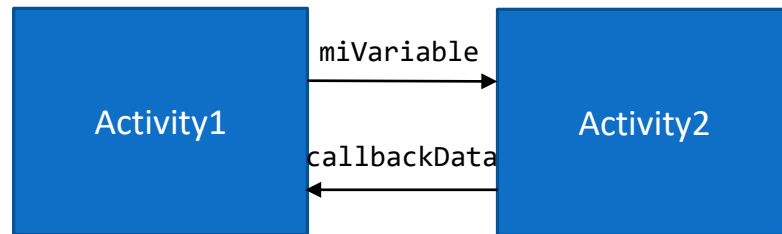


La Activity2 puede responder mediante un callback a la Activity1 en un evento como el toque de un botón:

```
Intent i = new Intent();  
i.putExtra("respuesta", callbackData);  
setResult(RESULT_OK, i);  
finish();
```

En este caso el estado es el
RESULT_OK

Intent

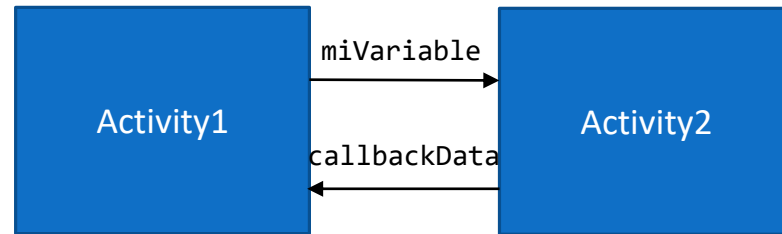


La Activity2 puede responder un dato usando un callback a la Activity1 en un evento, como el toque de un botón:

```
Intent i = new Intent();  
i.putExtra("respuesta", callbackData);  
setResult(RESULT_OK, i);  
finish();
```

Finalmente finish() permite cerrar la actividad.

Intent

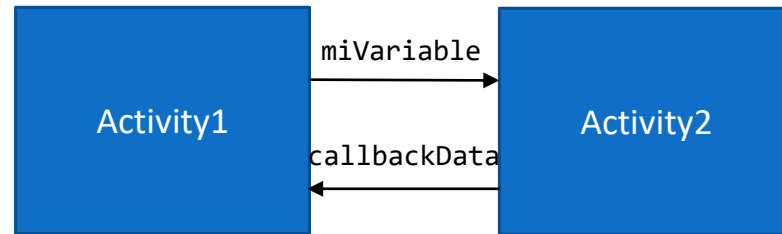


La Activity1 espera el dato sobrescribiendo el método onActivityResult:

```
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    if(requestCode == REQUEST_CODE && resultCode == RESULT_OK){
        ...
    }
}
```

Puede valer:
RESULT_OK
RESULT_CANCELED

Intent



La Activity1 espera el dato sobrescribiendo el método onActivityResult:

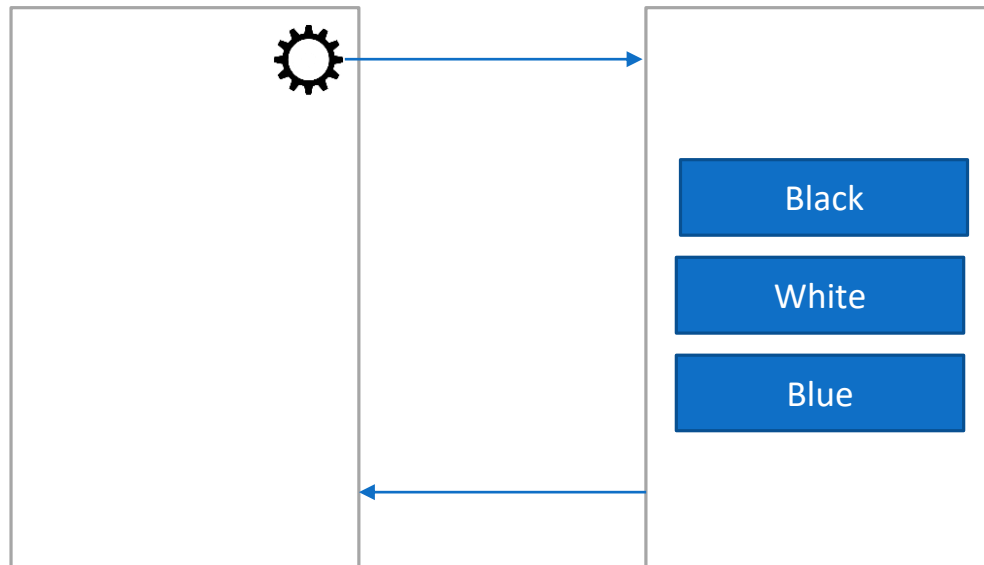
```
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    if(requestCode == REQUEST_CODE && resultCode == RESULT_OK){
        ...
    }
}
```

Corresponde al valor con el que inicialmente llamamos a Activity2

ACTIVIDAD EN CLASE

Cree una actividad principal que tenga un botón de configuración. El botón de configuración me permite cambiar el color de la actividad principal.

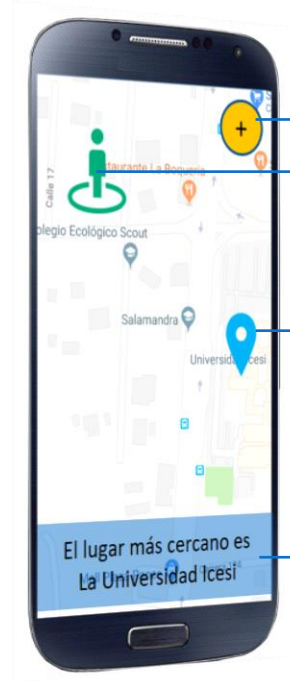
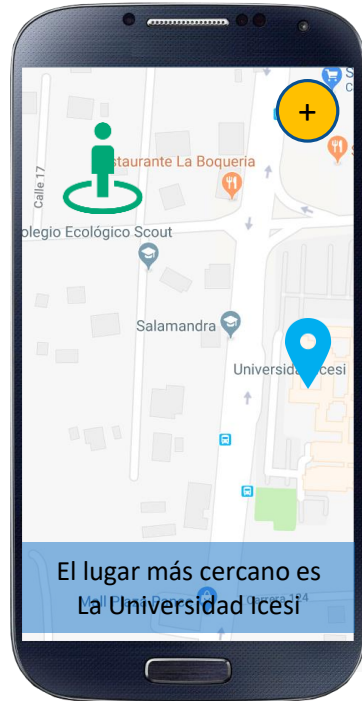
La actividad tendrá cuatro botones y para seleccionar el color, el usuario debe arrastrar el botón hasta la zona inferior



RETO 1

FECHA DE ENTREGA
8 DE SEPTIEMBRE

RETO 1: GOOGLE MAPS



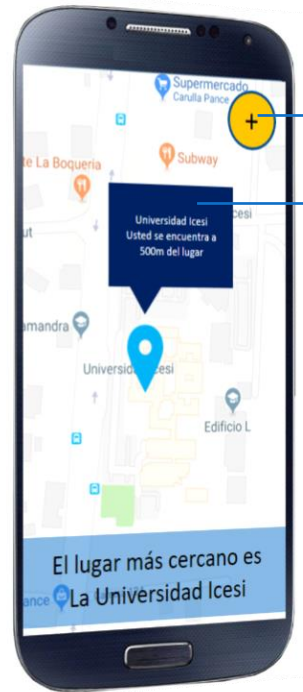
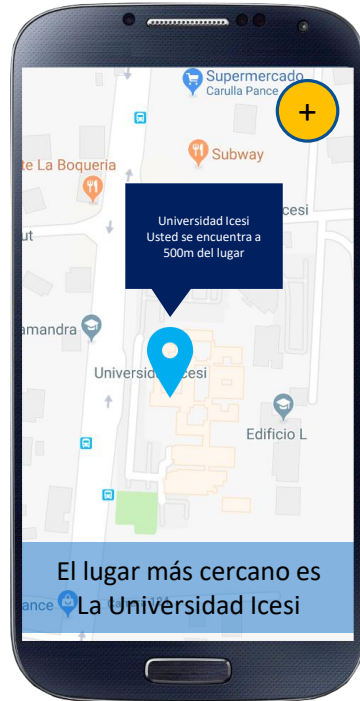
Permite agregar nuevos lugares para que la app los recuerde

Este marcador está puesto donde se encuentre ubicada la persona

El marcador está puesto donde se encuentre ubicada la persona

El cajón de información dirá información acerca del estado de la persona

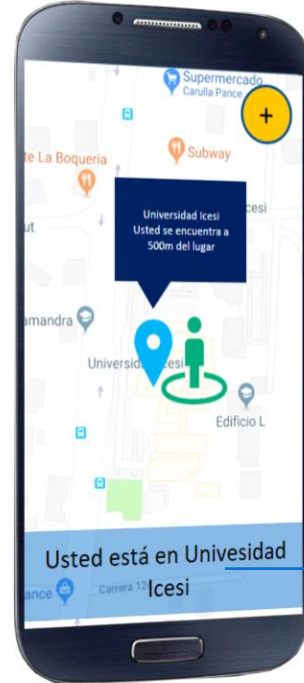
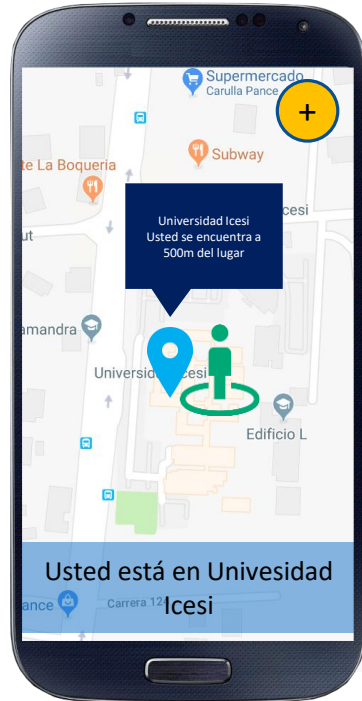
RETO 1: GOOGLE MAPS



La implementación es libre, pero poder normar y marcar el lugar.

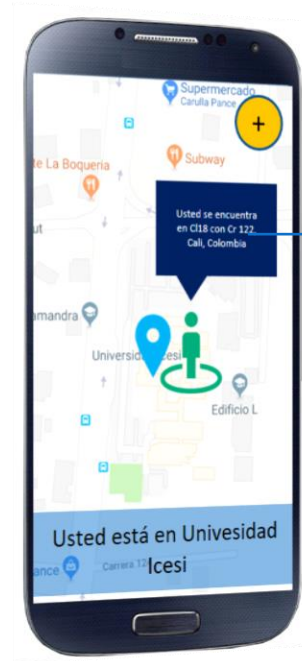
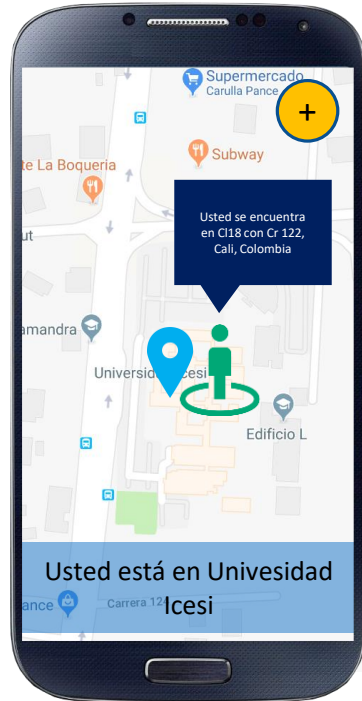
La información del marcado que debe ofrecer es a cuántos metros se encuentra el usuario del lugar.

RETO 1: GOOGLE MAPS



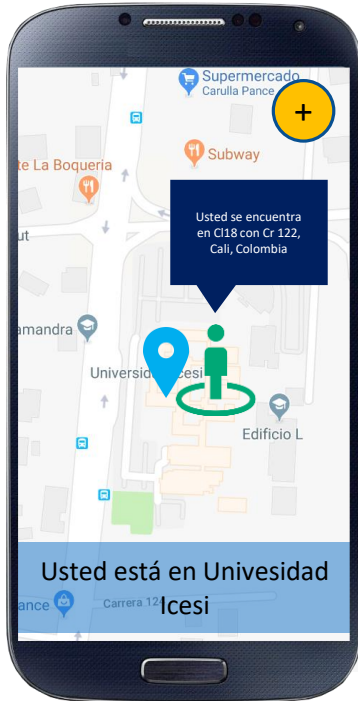
Si se encuentra muy cerca al punto, el cajón de información le debe decir a cuánto está.

RETO 1: GOOGLE MAPS



Cuando se click en el marcador de la persona, puede ver la dirección en la que se encuentra.

RETO 1: GOOGLE MAPS



Deben implementarlo usando
Google Maps SDK for Android

<https://developers.google.com/maps/documentation/android-sdk/intro>