David Javier Boquete Pulleiro 29BPDJ

Actividad 2: Bases de datos desde Android. Modificación.

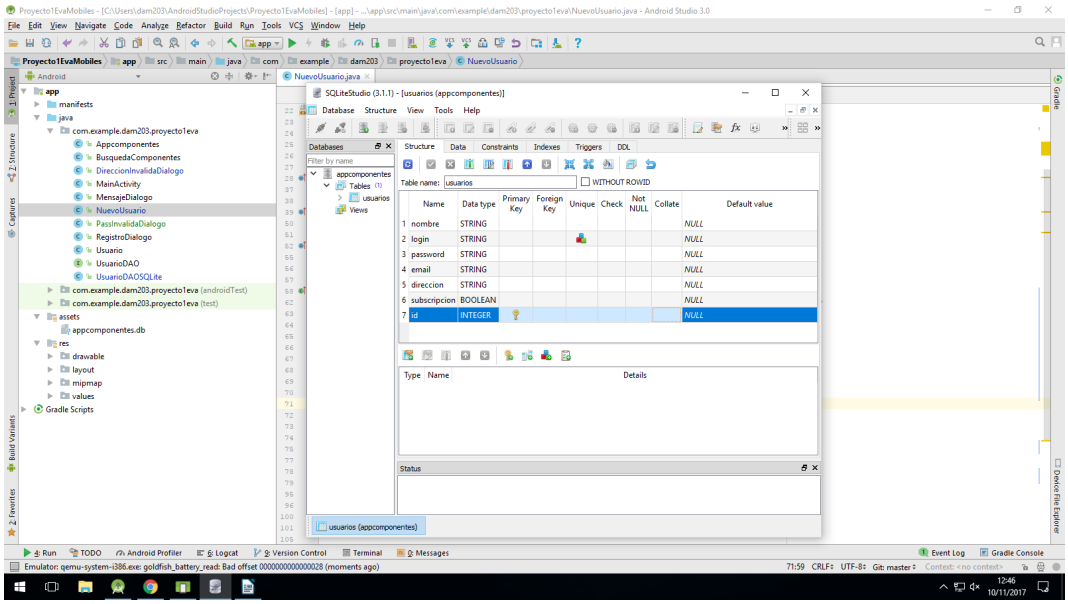
1. Instrucciones.

- Crear un proyecto Android Studio e implementar el programa que se describe a continuación.
- La entrega consistirá en enviar dos archivos separados:
  - Un archivo PDF con las capturas que se piden.
  - La carpeta del proyecto Android Studio en formato ZIP.
- Todos los textos que se muestren serán definidos como recursos String.
  - A excepción de los String que genere el programa como resultado de la ejecución.
- Todos los colores se asignarán mediante referencia a recursos definidos en el archivo `colors.xml`

2. Programa

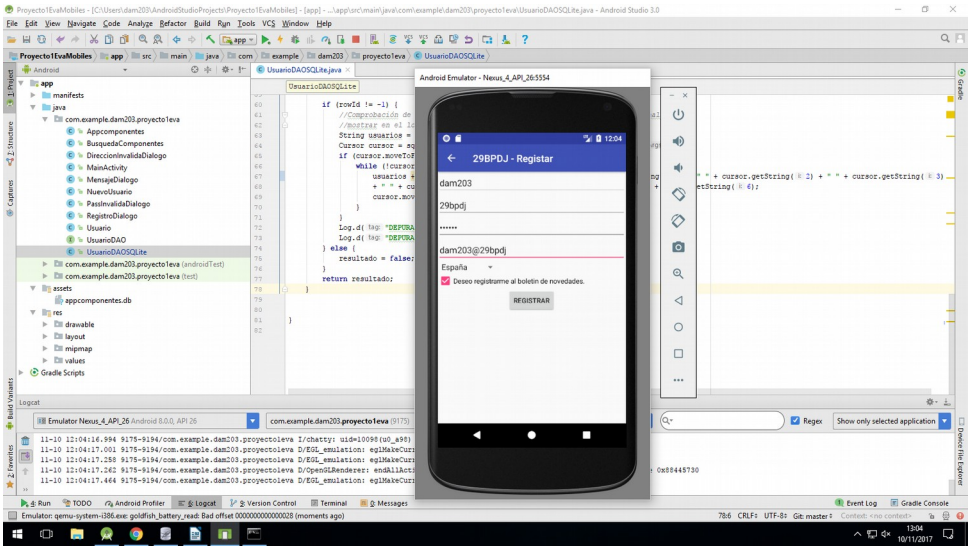
1. Aunque en la guía los strings se encuentran hard-coded para facilitar la lectura, cada alumno deberá crear los correspondientes recursos string y referenciarlos en el código.
2. Implementar la activity de registro de usuario. En su action bar deberán mostrar un título que dé a entender que es un formulario de registro.
3. Los elementos que deberá incluir obligatoriamente son:
   a. EditText:
      i. Nombre.
      ii. Login.
      iii. Password.
   b. Botón.
      i. Registro.
   c. Botón de navegación al formulario de login en la barra de título.
4. En la base de datos, añadir a la tabla *Usuario* tres campos dependiendo de la temática elegida para el proyecto. Realizar los mismo cambios en la clase *Usuario*.
   a. Ejemplos: campo boolean de aceptación de condiciones, elegir entre un conjunto de valores (colores, localidad, particular/empresa, tipo de documento de identidad, ...)
5. En la activity de registro de usuario, añadir tres elementos para poder recibir el valor de los tres nuevos campos añadidos a la tabla *Usuario*.

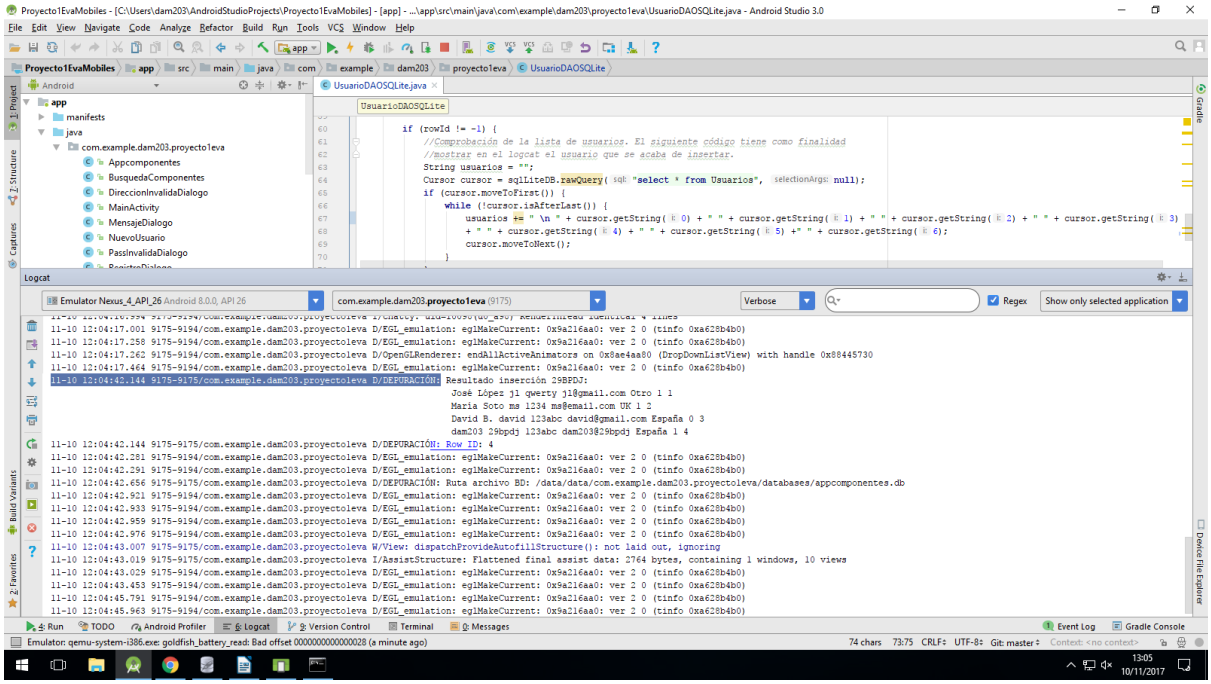
   a. Solamente uno de esos tres campos podrá ser un EditText.
   b. Ejemplos:
      i. Otros campos, por ejemplo, email.
      ii. Casilla de verificación.
         1. Por ejemplo, *He leído y acepto los términos del servicio*.
      iii. Spinner con una lista de opciones.
   c. Opcionalmente se podrán usar los siguientes elementos.
      i. Uso de un Listview (enlace).
         1. Si se quiere hacer una lista de multiselección, ver Anexo 2.
      ii. Opcionalmente, y para investigar por parte del alumno, diálogos *DatePickerDialog* o *TimePickerDialog*.
6. Implementar validación de campos.
   a. Usar el siguiente código de ejemplo que se indica en Anexo 1.
   b. Mientras no se supere la validación, el usuario no podrá ser creado.
   c. Ejemplos:
      i. Impedir contraseñas que no cumplan ciertos requisitos, por ejemplo, longitud mínima 6 caracteres.
      ii. Hacer que ciertos campos sean obligatorios, por ejemplo, login y password. Normalmente se marcarán con un * en la interfaz.
2. Mostrar el **identificador de alumno** en:
   a. Action bar de la activity de registro de usuario.
   b. Toast de usuario creado con éxito.
   c. Mensaje de depuración de logcat en el que se indica la filas de la tabla usuario tras la inserción. En concreto, a continuación del texto *"Resultado inserción"*
3. Capturas:
   a. Esquema de la tabla de usuarios. Si se ha implementado alguna tabla a mayores, incluir también la estructura.



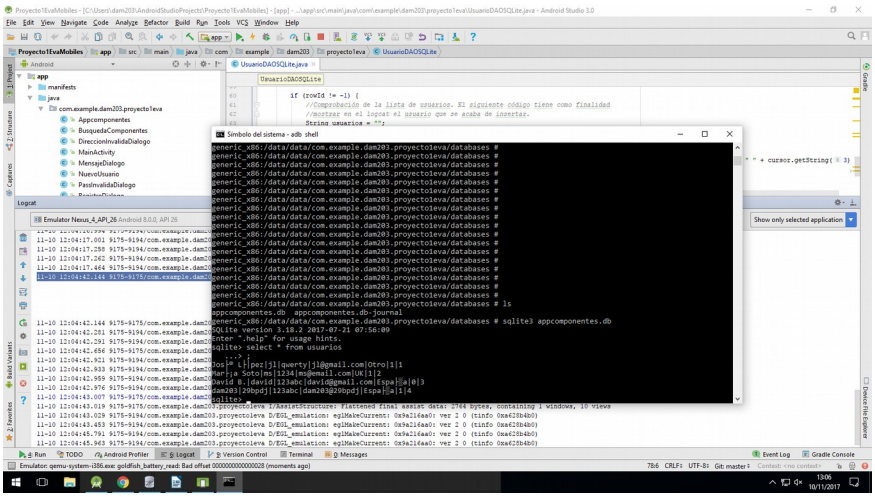
David Javier Boquete Pulleiro 29BPDJ

b. Capturas de los escenarios de ejecución representativos de la funcionalidad implementada.
   i. Junto con la captura de la ventana de inserción de usuario, será necesario incluir también la captura de los mensajes de depuración de logcat tras insertar un nuevo usuario pulsando el botón registro.





c. Contenido de la tabla de usuarios.
   i. Dos alternativas.
      1. Obtenerlo del logcat una vez se hayan realizado las ejecuciones y pruebas del programa.
      2. Conectándose al AVD desde la consola de comandos y ejecutando la respectiva consulta SQL.

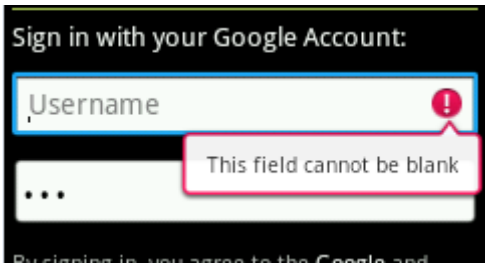


d. Enumerar las validaciones que se han implementado.
    1) Todos los campos con contenido.
    2) Spinner de direccion con una Direccion seleccionada.
    3) Contraseña con 6 o más caracteres.
    4) Excepcion controlada si el login del usuario ya existe.

3. Anexos.

Anexo 1: Mostrar errores en EditText.

- Es posible mostrar errores en los EditText, por ejemplo, debidos a validaciones, mediante el método *setError*.
- Ejemplo.
    - Resultado.



    - Código.

```
firstName.addTextChangedListener(new TextWatcher()  {

    @Override
    public void onTextChanged(CharSequence s, int start, int before, int count) {
    }

    @Override
    public void beforeTextChanged(CharSequence s, int start, int count, int after) {
    }
```

```
    @Override
    public void afterTextChanged(Editable s)  {
        if (firstName.getText().toString().length <= 0) {
            firstName.setError("Enter FirstName");
        } else {
            firstName.setError(null);
        }
    }
});
```

Anexo 2: ListView con multiselección.

- El hecho de poder seleccionar un conjunto de elementos de una lista podría implicar crear una nueva tabla en la base de datos ya que no se admiten campos multivaluados. Por ejemplo para una lista de intereses, se podría crear una tabla *Intereses* relacionada con *Usuario* a través del campo *id*. Los intereses se podrían representar con campos de tipo boolean, creando así un campo para cada tipo de interés que pueda seleccionar.
- Ejemplo (enlace).
    - En el ejemplo, para la programación del evento onItemClick (marcado en amarillo en el código), seguir mejor el método usado en la wiki enlace.
    - Step 1: setAdapter to your listview.

```
listView.setAdapter(new ArrayAdapter<String>(this, android.R.layout.simple_list_item_multiple_choice, GENRES));
```

    - Step 2: set choice mode for listview .The second line of below code represents which checkbox should be checked.

```
listView.setChoiceMode(ListView.CHOICE_MODE_MULTIPLE);
listView.setItemChecked(2, true);

listView.setOnItemClickListener(this);
private static  String[] GENRES = new String[] {
        "Action", "Adventure", "Animation", "Children", "Comedy", "Documentary", "Drama",
        "Foreign", "History", "Independent", "Romance", "Sci-Fi", "Television",
"Thriller"
};
```

    - Step 3: Checked views are returned in *SparseBooleanArray*, so you might use the below code to get key or values.The below sample are simply displayed selected names in a single String.

```
@Override
public void onItemClick(AdapterView<?> adapter, View arg1, int arg2, long arg3)
{

       SparseBooleanArray sp=getListView().getCheckedItemPositions();

       String str="";
       for(int i=0;i<sp.size();i++)
       {
              str+=GENRES[sp.keyAt(i)]+",";
       }
       Toast.makeText(this, ""+str, Toast.LENGTH_SHORT).show();
}
```

Anexo 3: Validación del formulario de creación de usuario.

- Pseudocódigo Java,

```
boolean campoLoginVacio(EditText e){
	boolean resultado=true;

	if (e.getText().toString().length()==0){
		resultado= false;
	}

	return resultado;
}

void clickBotonCrearUsuario(){
	boolean validacion=validacion();

	if (validacion)
		// Crear usuario
	else
		// Mostar mensaje de error
}

boolean validacion(){
	String mensaje="";
	if (!campoLoginVacio(edL)){
		//mensaje+=" El campo login está vacío";
		edL.setError("El campo login está vacío");
		validacion=false;
	}
	if (!campoPasswordLongitud(edP)){
		//mensaje+=" El campo login está vacío";
		edL.setError("El campo password tiene menos de 6 caracteres");
		validacion=false;
	}
}
```