Universidad Tecnológica de El Salvador

FACULTAD DE INFORMÁTICA Y CIENCIAS APLICADAS

ESCUELA DE INFORMÁTICA

ASIGNATURA: Electiva 1

SECCION:01

Documento:

Ejercicio para evaluación 2

TRABAJO PRESENTADO POR:

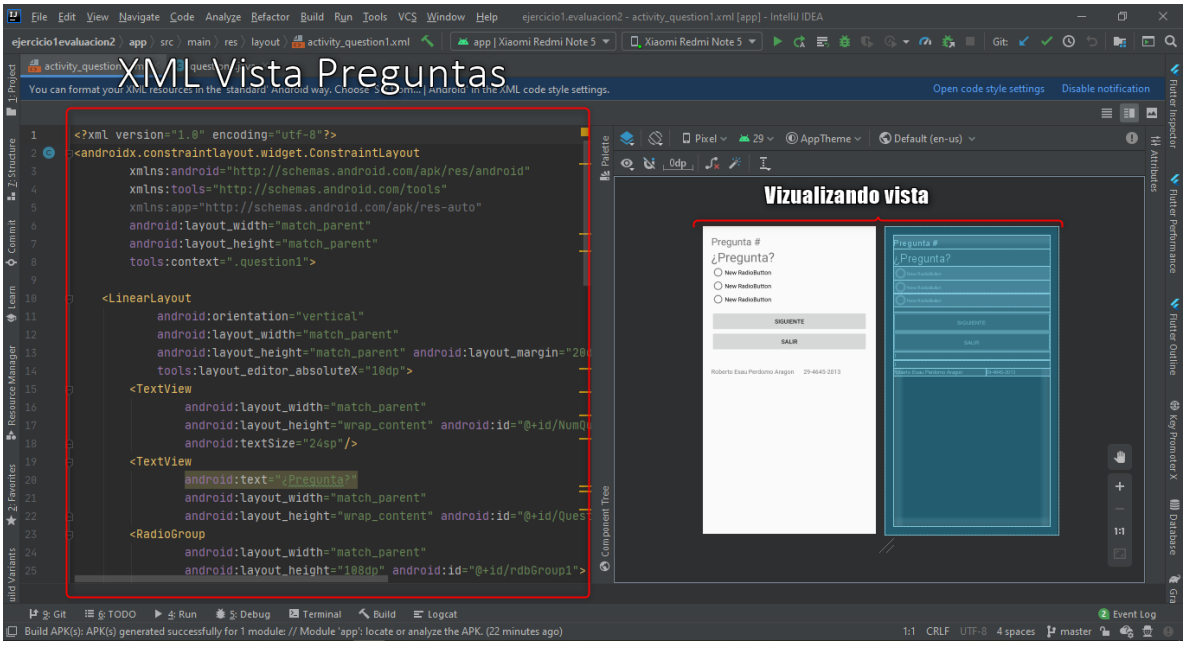
INTEGRANTES

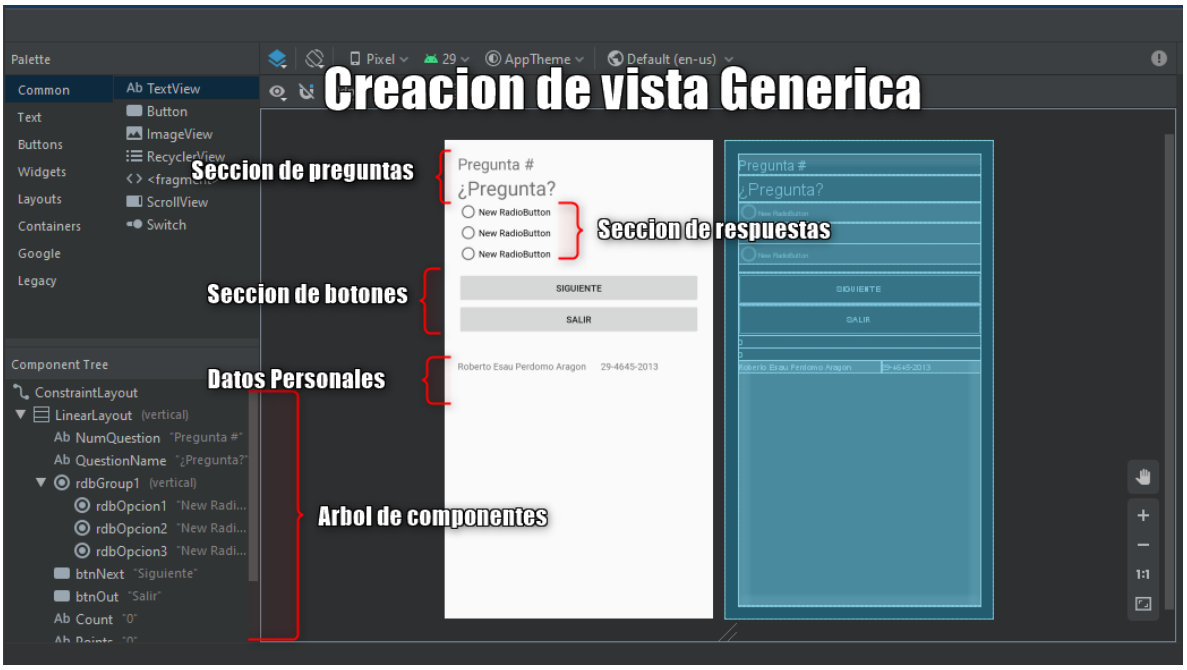
NOMBRES	CARNET
Perdomo Aragón, Roberto Esaú	29-4645-2013

22 de septiembre de 2020

Capturas Aplicación móvil

creación de vista





LO QUE CUENTA ES TU ACTITUD POSITIVA

Partes de código importantes

Para poder utilizar una vista genérica se necesito crear una pequeña base de información en este caso se utilizo una Lista Tipo “ModelQuestion” que es un modelo que guarda todo lo que incluye las Preguntas. Esta se inicializar al momento que creamos la vista.

```
1
2  // inicializamos una lista de preguntas
3  private List<ModelQuestion> ListQuestions
    = new ArrayList<ModelQuestion>();
4
5  // creando objetos para asignar los que usa
   remos en la lista

6  RadioGroup rdbGroup1;
7  RadioButton rdtQuestion1, rdtQuestion2,
   rdtQuestion3,rdtAnswerd;
8  TextView NameQuestion, NumQuestion,
   CountView,SumPoint;
9  Button Nextquestion, OutApp;
```

Modelo Tipo ModelQuestion

Modelo utilizado para manejar el objeto pregunta en la aplicación y así hacer la app más genérica

```java
package com.example.ejercicio1evaluacion2.Model;

// Modelo creado para guardar informacion de las preguntas
public class ModelQuestion {
    private int Idquestion;
    private int Numberquestion;
    private String Namequestion;
    private String AnswerText1;
    private String AnswerText2;
    private String AnswerText3;
    private int AnswerIdValue;


    public int getAnswerIdValue() {
        return AnswerIdValue;
    }

    public void setAnswerIdValue(int answerIdValue) {
        AnswerIdValue = answerIdValue;
    }

    public int getIdquestion() {
        return Idquestion;
    }

    public void setIdquestion(int idquestion) {
        Idquestion = idquestion;
    }

    public int getNumberquestion() {
        return Numberquestion;
    }

            ....
}
```

Métodos importantes

Encargado de crear la vista, en ella se inicializo los componentes de ellas y se crearon 2 eventos, uno cuando el usuario seleccionaba siguiente pregunta y otro para salir de la aplicación.

```java
RadioGroup rdbGroup1;
  RadioButton rdtQuestion1, rdtQuestion2, rdtQuestion3,rdtAnswerd;
  TextView NameQuestion, NumQuestion,CountView,SumPoint;
  Button Nextquestion, OutApp;

  @Override
  protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_question1);

      // Asignamos data a la lista creada
      AsignarList();

      //asignacion de objetos de la vista
      //Radiobuttons
      rdbGroup1 = (RadioGroup) findViewById(R.id.rdbGroup1);
      rdtQuestion1 = (RadioButton) findViewById(R.id.rdbOpcion1);
      rdtQuestion2 = (RadioButton) findViewById(R.id.rdbOpcion2);
      rdtQuestion3 = (RadioButton) findViewById(R.id.rdbOpcion3);

      //TextView
      NameQuestion = (TextView) findViewById(R.id.QuestionName);
      NumQuestion = (TextView) findViewById(R.id.NumQuestion);
      CountView = (TextView) findViewById(R.id.Count);
      SumPoint = (TextView) findViewById(R.id.Points);

      //Buttons
      Nextquestion = (Button) findViewById(R.id.btnNext);
      OutApp = (Button) findViewById(R.id.btnOut);

      //Inicializar con la primer pregunta
      AsignarPregunta(Integer.parseInt(CountView.getText().toString()));

      Nextquestion.setOnClickListener(new View.OnClickListener(){
          public void onClick(View v){
              SumarPuntos();
          }
      });
      OutApp.setOnClickListener(new View.OnClickListener(){
          public void onClick(View v){
              System.exit(0);
          }
      });

  }
```

Métodos para mostrar preguntas en la vista

La finalidad de este método es realizar toda la lógica de mostrar las preguntas en los componentes necesario en la vista y al finalizarlas calcular la nota final.

Para ello utilizamos un contador que al superar las 5 preguntas hace invisibles los componentes pedidos por el cliente y en efecto mostrar nota obtenida y si el usuario logro aprobar.

```java
// Metodo encargado de mostrar las preguntas en pantalla y hacer evalucion final
    private void AsignarPregunta(int PreguntaActual)
    {
        String Estado = "Aprobado";
        int NotaMinima = 6;
        int ActualPoint = Integer.parseInt(SumPoint.getText().toString());
        int converPregunt = PreguntaActual;
        int count = converPregunt + 1;
        int TotalPreguntas = 5;

        if (count <= TotalPreguntas)
        {
            NameQuestion.setText(ListQuestions.get(converPregunt).getNamequestion());
            NumQuestion.setText("Pregunta # "+ ListQuestions.get(converPregunt).getNumberquestion());
            rdtQuestion1.setText(ListQuestions.get(converPregunt).getAnswerText1());
            rdtQuestion2.setText(ListQuestions.get(converPregunt).getAnswerText2());
            rdtQuestion3.setText(ListQuestions.get(converPregunt).getAnswerText3());
            CountView.setText(Integer.toString(count));
        }
    else
        {
            NumQuestion.setText("Nota Obtenida: " + ActualPoint +".00");

            if (ActualPoint < NotaMinima)
            {
                Estado = "Reprobado";
            }
            NameQuestion.setText("Estado: " + Estado);

            Nextquestion.setVisibility(View.INVISIBLE);
            rdtQuestion1.setVisibility(View.INVISIBLE);
            rdtQuestion2.setVisibility(View.INVISIBLE);
            rdtQuestion3.setVisibility(View.INVISIBLE);
        }

    }
```

Método para validar respuestas

Este Método para evaluar si la respuesta seleccionada por el usuario es creada, tomando las respuestas de la lista creada y llenada con data anteriormente y si el usuario no ha contestado le muestra un Toast pidiendo que por favor seleccione una respuesta.

además, suma la puntuación del usuario, si su respuesta es correcta.

```java
// Método encargado de evaluar la puntuacion de las preguntas
    private void SumarPuntos()
    {
        int valorCorrecto = 2;
        int SinEspecificar = -1;
        int RespuestaPreguntaActual;
        rdtAnswerd = (RadioButton) findViewById(rdbGroup1.getCheckedRadioButtonId());
        int Respuesta = rdbGroup1.indexOfChild(rdtAnswerd);
        int ActualPoint = Integer.parseInt(SumPoint.getText().toString());
        int PreguntaActual = Integer.parseInt(CountView.getText().toString());

        if (Respuesta != SinEspecificar){
            int calcularposicionenlista = PreguntaActual -1 ;
            RespuestaPreguntaActual = ListQuestions.get(calcularposicionenlista).getAnswerIdValue();

            if (Respuesta == RespuestaPreguntaActual)
            {
                ActualPoint += valorCorrecto;
                SumPoint.setText(Integer.toString(ActualPoint));
            }
            rdtAnswerd.setChecked(false);
            AsignarPregunta(PreguntaActual);
        }else{
            Toast.makeText(this, "Por favor seleccione una respuesta", Toast.LENGTH_LONG).show();
        }
    }
```

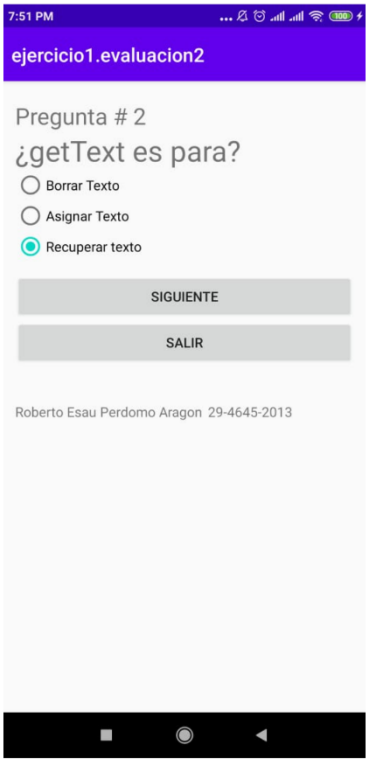
Método encargado de llenar lista general con las preguntas

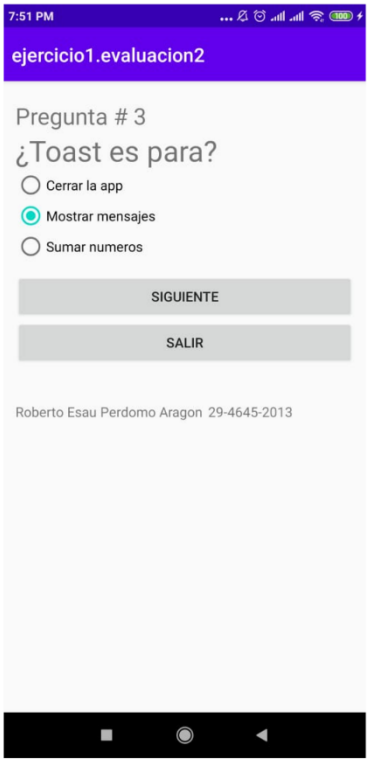
Utilizando el potencial de POO se instancias nuevos objetos a partir de la clase ModelQuestion que contiene las propiedad y métodos necesarios para cada pregunta.

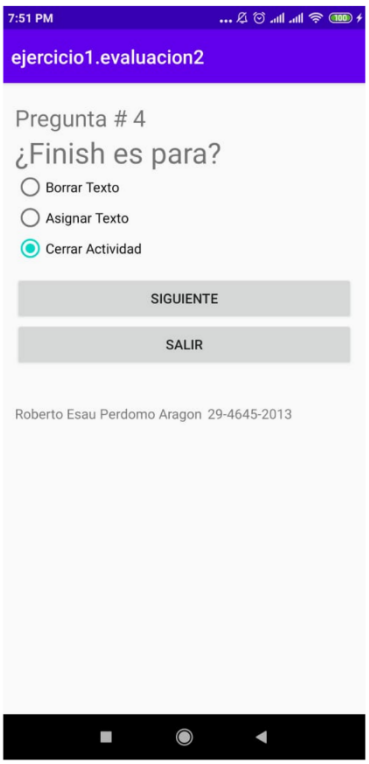
```
1  // Método encargado de asignar data de las preguntas
2     private void AsignarList()
3     {
4         ModelQuestion Pregunta1 = new ModelQuestion();
5             Pregunta1.setIdquestion(0);
6             Pregunta1.setNumberquestion(1);
7             Pregunta1.setNamequestion("¿setText es para?");
8             Pregunta1.setAnswerText1("Borrar Texto");
9             Pregunta1.setAnswerText2("Asignar Texto");
10            Pregunta1.setAnswerText3("Ocultar texto");
11            Pregunta1.setAnswerIdValue(1);
12        ListQuestions.add(Pregunta1);
13
14        ModelQuestion Pregunta2 = new ModelQuestion();
15            Pregunta2.setIdquestion(1);
16            Pregunta2.setNumberquestion(2);
17            Pregunta2.setNamequestion("¿getText es para?");
18            Pregunta2.setAnswerText1("Borrar Texto");
19            Pregunta2.setAnswerText2("Asignar Texto");
20            Pregunta2.setAnswerText3("Recuperar texto");
21            Pregunta2.setAnswerIdValue(2);
22        ListQuestions.add(Pregunta2);
23
24        ......
25    }
```

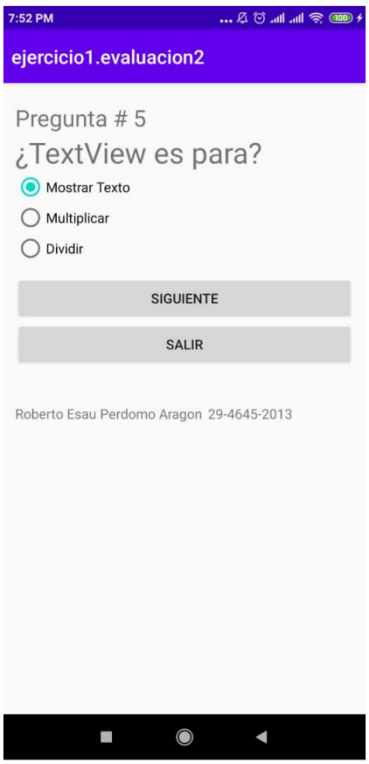
Capturas aplicación en funcionamiento

Contestando de forma correcta todas las preguntas









Cuando un usuario selecciona mal las respuestas, esta sería un ejemplo del resultado

LO QUE CUENTA ES TU ACTITUD POSITIVA

Bibliografía

Código Fuente creado por Estudiante

https://github.com/rperdomo24/Electiva1-Android