|  |  |  |
| --- | --- | --- |
| **Nombre**:  María Fernanda Villegas Casco. | | **Matrícula**:  2817368. |
| **Nombre del curso:**  Desarrollo de aplicaciones en plataforma Android | **Nombre del profesor**:  Carolina de la Torre Hernández | |
| **Módulo**: Introducción al ambiente de Android | **Actividad**: Investigación de elementos Android | |
| **Fecha**: Jueves, 19 de Agosto de 2021. | | |
| **Bibliografía**:  *Understand the Activity Lifecycle |*. (s. f.). Android Developers. Recuperado 19 de agosto de 2021, de <https://developer.android.com/guide/components/activities/activity-lifecycle#java>  B. (2017). *Máster en Desarrollo de Aplicaciones Android - Layouts*. Android Curso. <http://www.androidcurso.com/index.php/114>  *Descripción general de eventos de entrada | Desarrolladores de Android*. (s. f.). Android Developers. Recuperado 20 de agosto de 2021, de https://developer.android.com/guide/topics/ui/ui-events?hl=es-419 | | |

**Objetivo:**

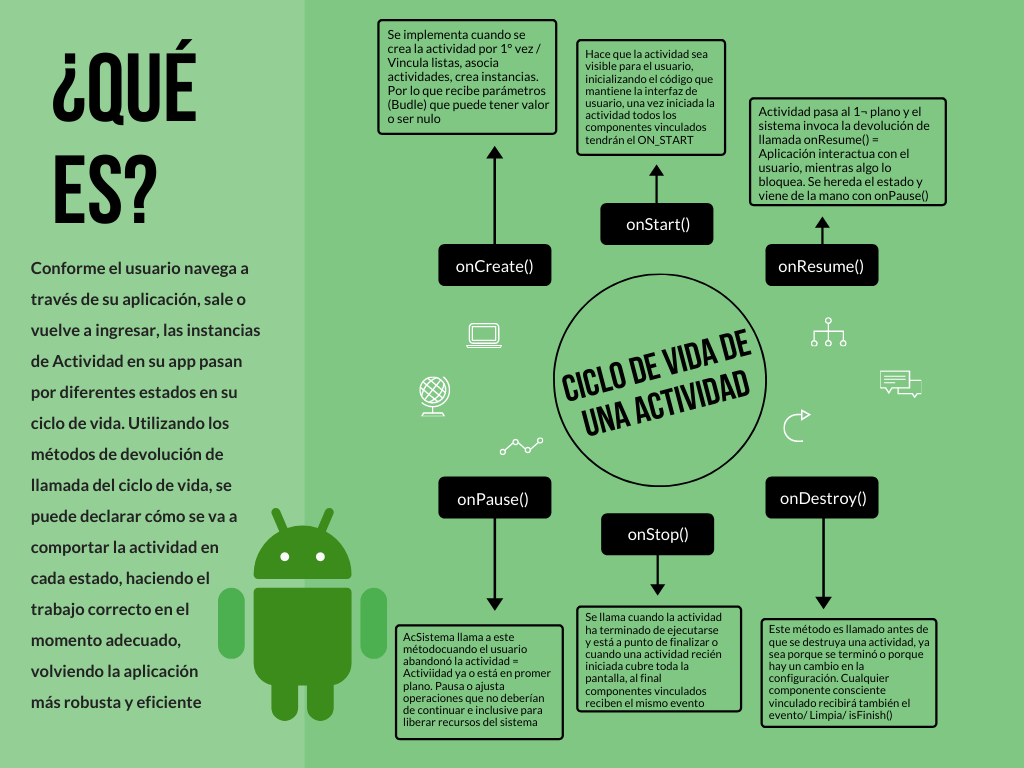
Investigar elementos utilizados en la sintaxis de Android para poderlos aplicar posteriormente en el desarrollo de una aplicación.

**Procedimiento:**

1. Investiga en [developer.android.com](javascript:;) cada uno de los métodos definidos en el ciclo de vida de una actividad, describiendo cuando se usa.
2. En el mismo sitio deberás investigar el nombre de la clase y el nombre y descripción de al menos tres métodos para los siguientes elementos de interfaz gráfica:
   1. Layout
   2. Texto Editable
   3. Botón
   4. Etiqueta
   5. Imagen

**Resultados:**

PARTE 1:



PARTE 2: Elementos

**-Layout-**

*layoutopt*es una herramienta de línea de comandos que se encarga de optimizar los diseños y las jerarquías de las aplicaciones y algunos de sus métodos son los siguientes:

* **LinearLayout**: Dispone los elementos en una fila/columna.
* **TableLayout**: Distribuye los elementos de forma tabular.
* **RelativeLayout**: Dispone los elementos en herencia o simir a otro.
* ***ConstraintLayout:*** Mejor que el anterior ya que puedes realizar una edición visual desde el editor.
* **FrameLayout**: Permite el cambio dinámico de los elementos que contiene.
* ***AbsoluteLayout:*** Posiciona los elementos de forma absoluta.

**-Texto Editable-**

*EditText* es un TextView pero aquí su apariencia se modificó para actuar como campo de texto, donde el usuario puede editar su contenido para especificar datos

* **getText():** Retorna el valor de un texto si está como EditText
* **android:textSize** Cambia el tamaño del texto a más grande o más chico
* **textAppearance**: Crea un cambio en la apariencia del texto ya sea con un estilo del sistema o propio.
* ***AutoCompleteTextView:*** Despliega sugerencias de palabras mientras el usuario pone caracteres
* ***Date:*** Pone el formato del texto como fecha

**-Texto Editable-**

*Button* es una clase con la cuál podrás hacer que se genera una acción cuando el usuario lo toque

* **ImageButton:**  Una imagen que sirva como botón
* **onClick()** Cuando se toca un elemento y queremos que tenga una acción
* **onLongClick()** Cuando se mantiene un elemento precionado y queremos que tenga una acción
* ***onFocusChange():*** Se llama este método cuando el usuario navega hacia el elemento o sale de este utilizando teclas de navegación o seguimiento
* ***onTouch:*** Muestra un True/False o 1 o 0 (booleano)

**-Etiqueta-**

Describen el propósito o la acción asociados con los Views o clases

* **Android:contentDescription:**  Incluye etiquetas de contenidos para ImageView, ImageButton o CkeckBox
* **Android:hint:** Para editTexts o TextViews que sean editables se utiliza ese atributo
* **Andorid:labelFor:** Indica que la view no debe funcionar como etiqueta
* **<include>** Agrega componente reutilizable
* **<merge>** Quita los grupos de vista que sean redundantes dentro de la jerarquía de vistas cuando se incluyen dentro de otro diseño

**-Imagen-**

ImageView hace referencia la imagen dentro de Android Studio

* **Android:layout\_width & Android:layout\_height:**  para poner la el tamaño de la imagen
* **takePicture(Executor, OnImageCapturedCallback):** Proporciona un búfer de la imagen y en la memoria
* **Android:orientation=** para vertical u horizontal
* **Android:id=””** para reconocer por un nombre clave a la imagen
* **Resouces.getDravable():** Retorna un objeto “drawable” asociado con un ID particular y con un estilo para el tema específico

**Conclusión:**

Gracias a esta actividad aprendí las bases del ciclo de vida de las aplicaciones Android, que pude comparar con el desarrollo en Swift.   
Me di cuenta que se utiliza Java o Kotlin, así que tengo mucho por aprender y espero con ansias poder poner ya mis manos en el código.