



El futuro digital  
es de todos

MinTIC

# La Importancia de una Activity





El futuro digital  
es de todos

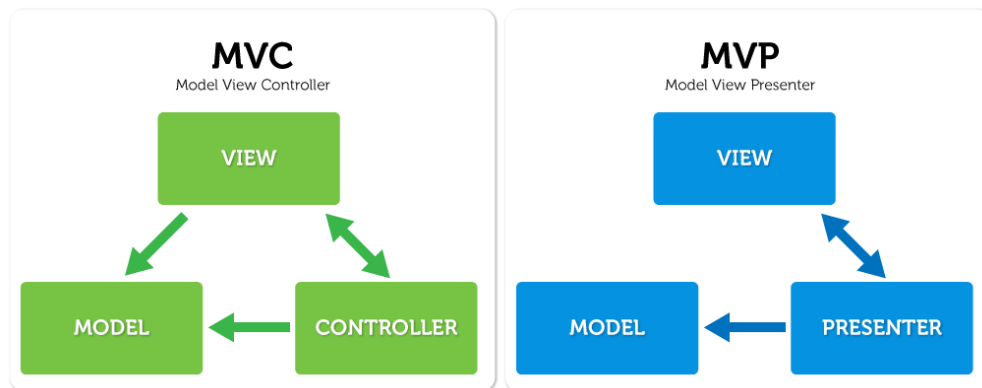
MinTIC

# Tema 1: Arquitectura modelo vista presentador para desarrollo de aplicaciones

# MVP

Los patrones de arquitectura, nos permiten mantener un proyecto limpio, escalable, fácil de mantener y de testear.

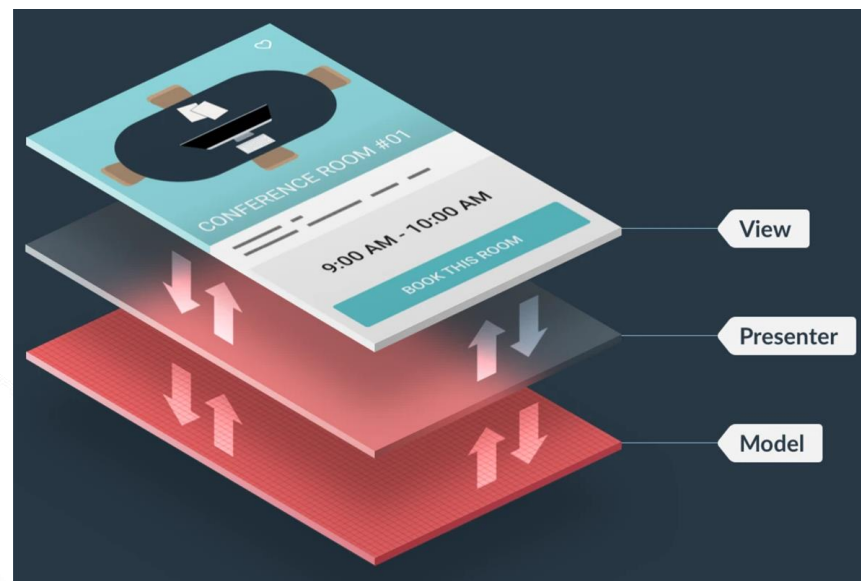
El MVP (Model View Presenter) es un patrón derivado del MVC (Model View Controller), que nos permite separar de forma muy clara nuestras vistas de la lógica de nuestras aplicaciones.



Fuente imagen: <https://medium.com/@tinmegali/model-view-presenter-mvp-in-android-part-1-441bfd7998fe>

# Capas del patrón MVP

- **Modelo:** Esta capa gestiona los datos. Son las clases que se denominan de lógica de negocio.
- **Vista:** Se encarga de mostrar los datos y reacciona con las acciones del usuario.  
En Android se encontrarán: Actividades (Activity), Fragmentos (Fragment), Vistas (View) y Diálogos (Dialog).
- **Presentador:** Se sitúa entre el modelo y la vista, permitiendo conectar la interfaz gráfica (Vista) con los datos, también se encarga de las tareas en segundo plano.

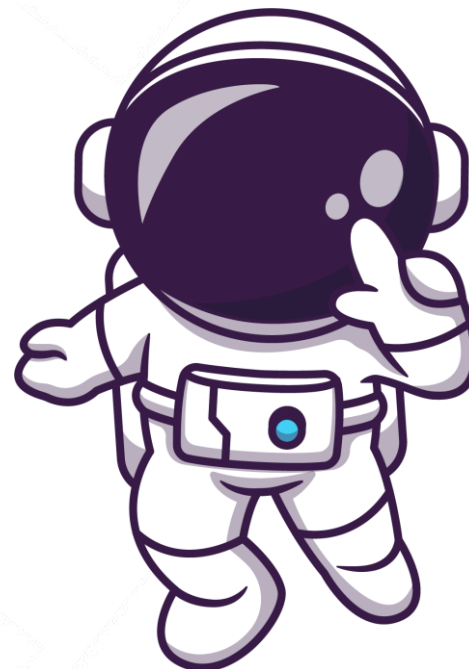


Fuente imagen: <https://bit.ly/3w1Swjc>



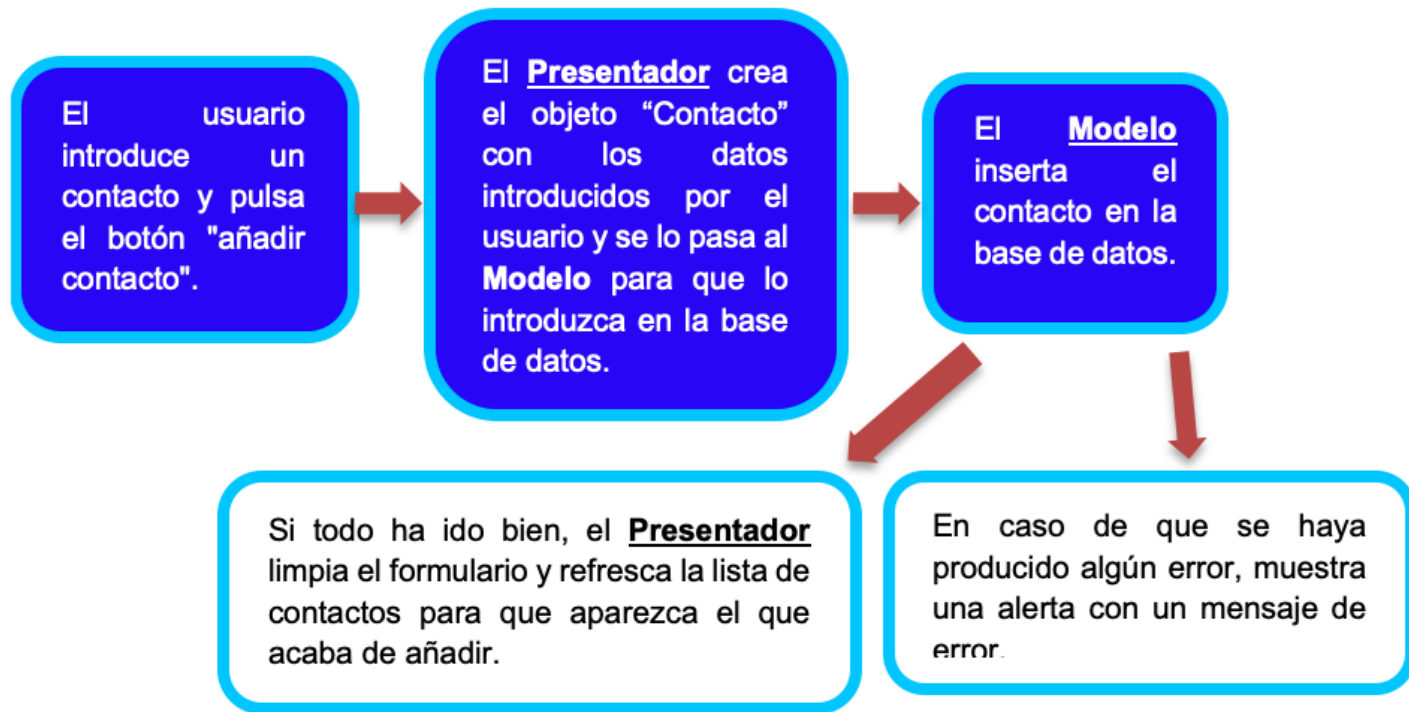
# MVP en Android

Android no nos ofrece de forma nativa la posibilidad de desarrollar nuestras aplicaciones bajo el patrón MVP, de hecho viola mucho de sus principios básicos. Aun así podemos llevar a cabo alguna aproximación para este fin. Vamos a ver un posible ejemplo de implementación en el que el usuario dispone de un formulario donde puede introducir contactos.





# MVP en Android







El futuro digital  
es de todos

MinTIC

# Tema 2: Definición y creación de actividades (Activity)



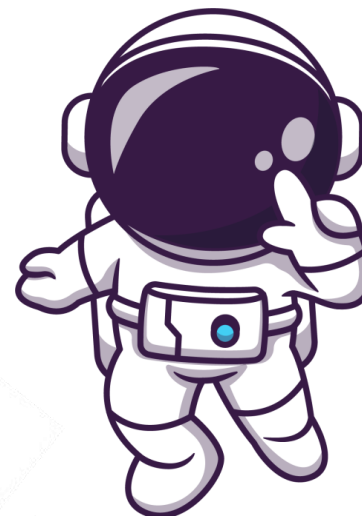


# Actividad = Pantallas

Una aplicación en Android va a estar formada por un conjunto de elementos básicos de visualización (textos, botones, imágenes, etc), coloquialmente conocidos como pantallas de la aplicación. En Android cada uno de estos elementos, o pantallas, se conoce como actividad. Su función principal es la creación de la interfaz de usuario.

Una aplicación suele necesitar varias actividades para crear la interfaz de usuario. Las diferentes actividades creadas serán independientes entre sí, aunque todas trabajarán para un objetivo común.

Una actividad se define en una clase descendiente de Activity y utiliza un layout que define su apariencia.



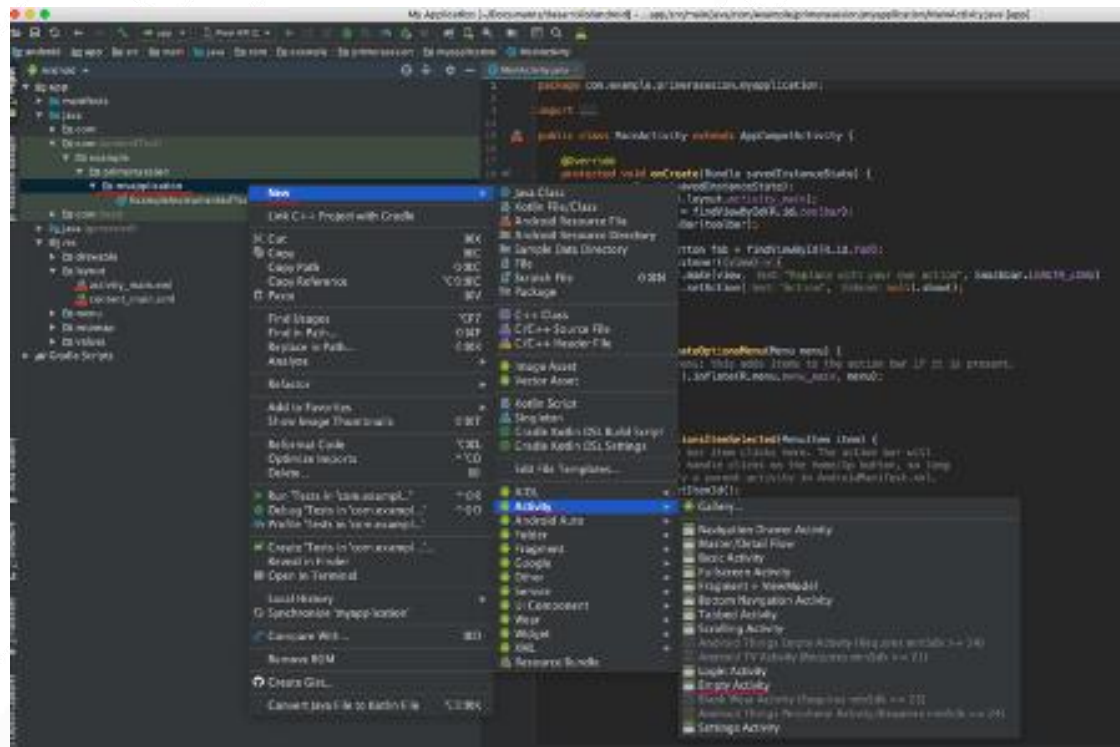




# Creación de una Actividad (Activity)

Si utilizamos como ejemplo la implementación de una nueva Actividad en un proyecto ya disponible, realizaríamos los siguientes pasos:

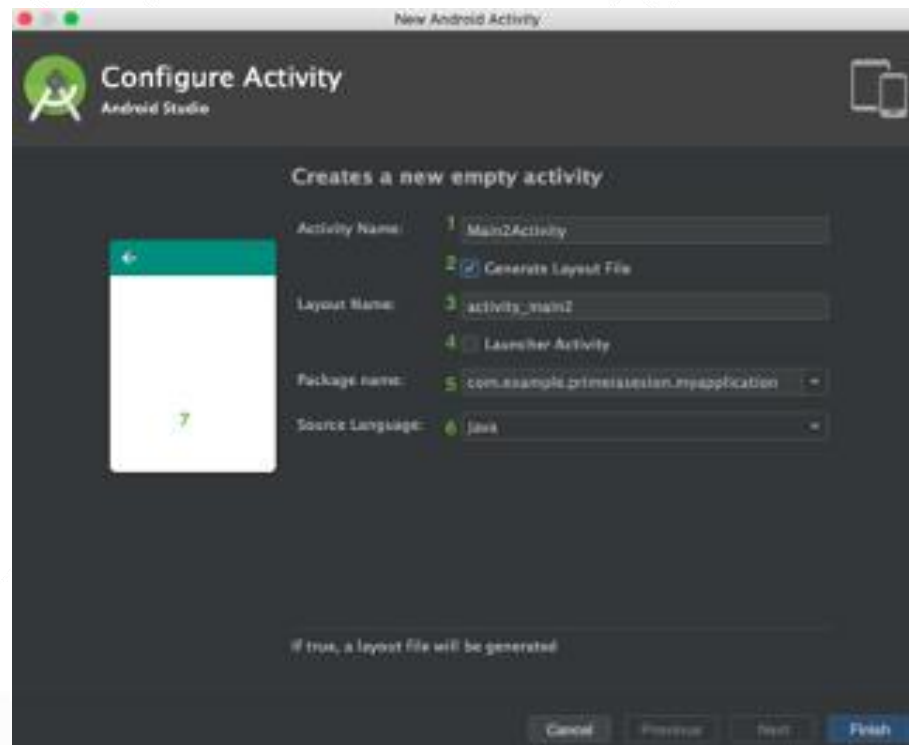
**PASO 1:** Pulsamos click derecho sobre el package del proyecto donde se encuentran las clases del mismo.





# Creación de una Actividad (Activity)

**PASO 2:** seguimos la ruta  
New › Activity y seleccionamos  
la opción Empty Activity  
(permitirá crear una plantilla con  
un diseño básico en el proyecto).





El futuro digital  
es de todos

MinTIC

# Tema 3: Ciclo de vida de una actividad en Android

# Ciclo de Vida de una Activity

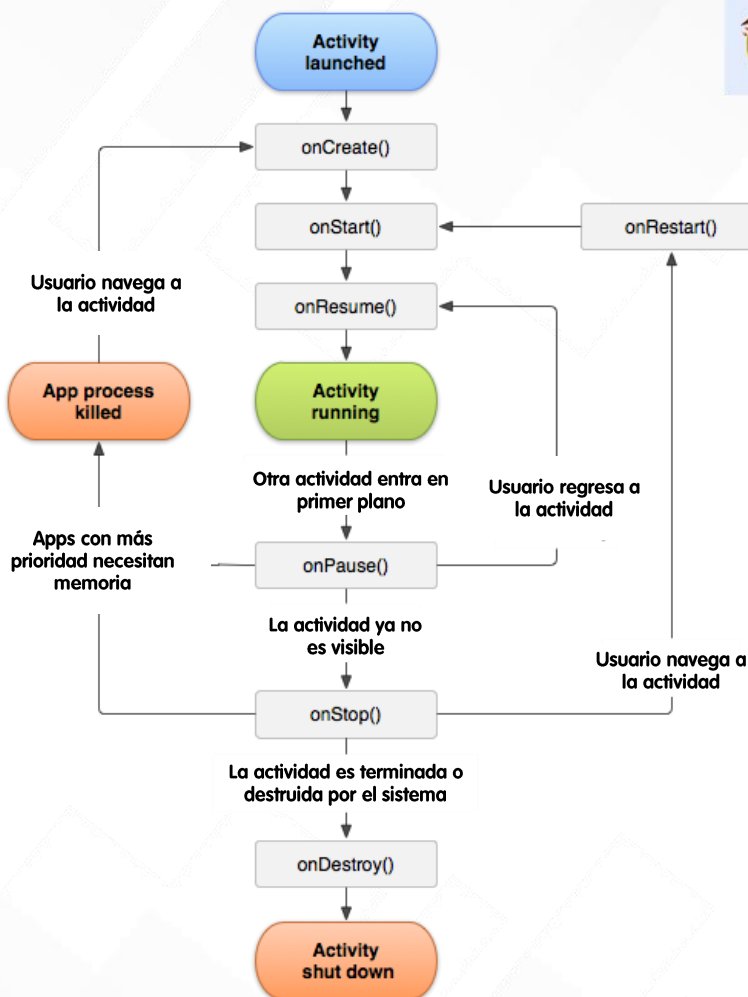
Una actividad pasa por varios estados que juegan un papel importante en su desarrollo, estos estados o callbacks van desde antes que se inicie una actividad, a el momento en que se está ejecutando y hasta después que se finaliza ésta.

El principal estado de una actividad es cuando se encuentra activa, esto quiere decir que durante todo su proceso de vida, la mayor parte de su tiempo lo pasa en este estado. Una actividad está activa cuando se encuentra en primer plano y el usuario puede interactuar con ella.





# Ciclo de Vida de una Activity





# Ciclo de Vida de una Activity

## onCreate()

Este método es llamado justo cuando se lanza la actividad. Dentro de este método es donde se coloca el código que inicializa las principales funciones:

- Infla los widgets y los posiciona en la pantalla.
- Obtiene las referencias de estos widgets.
- Asigna listeners a los widgets para que el usuario pueda interactuar con ellas.

## onStart()

Este método es llamado después de **onCreate()** y es el momento en el que la actividad está a punto de volverse visible al usuario.

## onResume()

Es llamado después de **onStart()**. En este punto la actividad por fin está en primer plano y se vuelve visible al usuario, este puede interactuar con los elementos o widgets que existan en la pantalla.





# Ciclo de Vida de una Activity

## onPause()

Es llamado cuando la actividad pierde el foco o deja de estar en primer plano. Esto puede ser causado por dos acciones:

- El usuario presiona el botón Home.
- El usuario presiona el botón de aplicaciones recientes.

La actividad se vuelve parcialmente visible. En este punto el rumbo de la actividad podría tomar dos caminos:

**onResume()**

**onStop()**

## onStop()

Este es llamado cuando la actividad deja de ser visible, puede ser causado porque la actividad ha sido destruida, una nueva actividad está siendo lanzada o una actividad existente vuelve a estar en primer plano. Éste método puede tomar dos caminos:

**onRestart()**

**onDestroy()**



# Ciclo de Vida de una Activity

## onRestart()

Es llamado después de **onStop()** si es que el usuario eligió poner nuevamente la actividad en primer plano.

## onDestroy()

Es llamado cuando el usuario elige dar por finalizada su interacción con la actividad. Este método permite liberar recursos que se han estado utilizando por la actividad.

En general una actividad puede contenerse en 4 estados:

Estado	¿En memoria?	¿Visible para el usuario?	¿En primer plano?
Inexistente	No	No	No
Detenida	Sí	No	No
Pausada	Sí	Parcialmente	No
En ejecución	Sí	Sí	Sí