



# TodoAppV01

## Documentación Técnica


Aplicación de gestión de tareas desarrollada en Android Studio con  
Java

 Android Studio

 Java

 GitHub

 Seguridad

 GFPI-F-135 V01

### 1. Introducción

**TodoAppV01** es una aplicación móvil desarrollada como parte del componente formativo **GFPI-F-135 V01**, cuyo objetivo es gestionar tareas diarias aplicando conceptos clave de desarrollo de software, arquitectura por componentes y buenas prácticas de codificación.

Este documento detalla todos los aspectos técnicos del proyecto, desde requerimientos hasta patrones de diseño, cumpliendo con los estándares profesionales de desarrollo Android.

---

### 2. Requerimientos del Sistema

Tipo	Requerimiento
<b>Funcionales</b>	Agrear, marcar como completada y eliminar tareas
<b>No Funcionales</b>	Interfaz intuitiva, rendimiento óptimo, compatible con Android 8.0+
<b>Almacenamiento</b>	Memoria temporal (futuro: Room)
<b>Dispositivos</b>	Emulador y físico



### 3. Funcionamiento de Android Studio y su SDK

#### **Android Studio**

IDE oficial para desarrollo Android. Incluye editor de código, depurador, emulador y diseñador visual.

#### **SDK de Android**

Conjunto de herramientas y bibliotecas que permiten desarrollar apps para diferentes versiones de Android.

## Gradle

Sistema de construcción que gestiona dependencias y compila el proyecto.

## Emulador

Permite probar la app en diferentes dispositivos virtuales sin hardware físico.

## 4. Diagrama de Clases

```
+-----+
|  TaskItem  |
+-----+
| - title: String |
| - completed: boolean |
+-----+
| + getTitle()    |
| + isCompleted() |
| + setCompleted() |
+-----+

+-----+
|  TaskAdapter  |
+-----+
```

```

| - tasks: List |
| - context: Context |
+-----+
| + onCreateViewHolder() |
| + onBindViewHolder() |
| + getItemCount() |
+-----+

```

```

+-----+
| MainActivity |
+-----+
| - recyclerView |
| - addButton |
| - inputField |
| - taskList |
+-----+
| + addTask() |
| + updateUI() |
+-----+

```

## 5. Diagrama de Paquetes

```

com.example.todoappv01
├── model/           → TaskItem.java
├── adapter/         → TaskAdapter.java
├── ui/              → MainActivity.java
└── utils/           → (futuro)

```



## 6. Diagrama de Componentes

```
[Usuario]
  ↓
[activity_main.xml]
  ↓
[MainActivity] ↔ [TaskAdapter]
  ↓
[TaskItem]
```



## 7. Capas del Software

Capa	Componentes	Responsabilidad
<b>Presentación</b>	MainActivity, XMLs	Interacción con el usuario
<b>Lógica de Negocio</b>	TaskAdapter	Gestión de la lista
<b>Modelo de Datos</b>	TaskItem	Representa una tarea
<b>Acceso a Datos</b>	ArrayList	Almacenamiento temporal

## ⚙️ 8. Metodología de Desarrollo

Análisis → Diseño → Desarrollo → Pruebas → Documentación → Git

Metodología ágil iterativa con enfoque en buenas prácticas y cumplimiento del componente formativo.

## 🗺️ 9. Mapa de Navegación

```
[ Pantalla Principal ]  
    ↓  
[ Lista de Tareas ]  
    ↓  
[ Botón + → Añadir Tarea ]
```

Interfaz simple y funcional, ideal para MVP.

## </> 10. Codificación por Módulos

Módulo	Archivo	Lenguaje

Presentación	activity_main.xml, MainActivity.java	Java + XML
Modelo	TaskItem.java	Java
Adaptador	TaskAdapter.java	Java
Recursos	strings.xml, colors.xml	XML

✅ Todos los módulos codificados en **Java**, lenguaje seleccionado.

## 11. Control de Versiones (Git)

**Repositorio:** [github.com/fergmoon1/ToDoAppV01](https://github.com/fergmoon1/ToDoAppV01)

**Flujo:** Commits descriptivos, rama `main`

```
git commit -m "feat: implementar RecyclerView"
git commit -m "fix: evitar tareas vacías"
git commit -m "docs: actualizar README"
```

## 12. Librerías de la Capa de Presentación

Librería	Uso
RecyclerView (AndroidX)	Mostrar lista de tareas

ConstraintLayout	Diseño responsive y flexible
AppCompatActivity	Base para la actividad principal
Button, EditText	Componentes de UI

☒ Todas parte del **SDK de Android**.

## 13. Frameworks Utilizados

Capa	Framework	Uso
UI	Android SDK	Interfaz de usuario
Layout	AndroidX	Componentes modernos y compatibles
Futuro	Room	Persistencia de datos
Futuro	Navigation Component	Navegación entre pantallas

## 14. Reutilización de Componentes

Componente	Reutilizable en
------------	-----------------



task_item.xml	Otras listas (notas, contactos)
TaskAdapter	Cualquier lista de objetos
TaskItem	Como modelo base para otras entidades

✓ Diseño modular y escalable.

## ★ 15. Buenas Prácticas de Codificación

- ✓ Nombres descriptivos: `addTask()`, `isCompleted()`
- ✓ Comentarios claros y código limpio
- ✓ Indentación consistente (4 espacios)
- ✓ Uso de `final` cuando aplica
- ✓ Funciones cortas y reutilizables
- ✓ Separación de responsabilidades

## 📁 16. Estructura de Paquetes

```
com.example.todoappv01
```

```
├─ model/           → Fácil de entender: contiene modelos
├─ adapter/         → Adaptadores de UI
└─ ui/              → Actividades y controladores
```

- ✓ Paquetes con nombres de **fácil entendimiento**.

## 17. Patrones de Diseño

### ViewHolder

Optimiza el rendimiento de RecyclerView.

### Separación de Responsabilidades

Cada clase tiene una única función.

### Singleton (futuro)

Para acceso global a datos.

- ✓ Aplicados según la arquitectura por componentes.

🔧 Desarrollado con Android Studio | Java | Git

© 2025 TodoAppV01. Todos los derechos reservados.

**Documento generado según GFPI-F-135 V01**