计算机科学技术学院实验报告

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **课程名称** | 移动应用开发 | | | **学 号** | 230511637 |
| **实验项目** | 使用Android Studio完成基础功能 | | | **姓 名** | 张世浩 |
| **学 时** | 1.5h | **项目性质** | 设计型 | **班 级** | 2305116 |
| **指导教师** | 高宁 周超然 | **实验地点** | 实训楼424 | **日 期** | 2025年10月21日 |
| 1. **实验目的和要求**   实验目的和要求：   1. 环境部署：掌握 Android Studio 开发环境的安装与配置方法，能成功配置 Android 7.0 及以上版本的 SDK，学会创建并设置 AVD 虚拟机，为 Android 应用开发搭建基础运行环境。 2. 功能实现：通过实操掌握 Android 界面元素（按钮、弹出菜单）的添加与使用，学会运用 Toast 组件实现交互反馈，理解 Android 应用的基本开发流程与交互逻辑。 3. 规范操作：熟悉实验结果的文件组织规范（如根目录、子文件夹分类）、实验报告撰写要求及提交流程，培养标准化的开发与文档管理习惯。 | | | | | |
| 1. **算法分析**   一、双按钮+Toast交互功能算法分析 该功能的核心目标是“用户点击指定按钮后，通过Toast组件反馈按钮名称”，算法逻辑围绕“控件初始化→事件监听→反馈触发”的线性流程设计，具体步骤与逻辑如下：   1. 核心算法流程 2. 控件定位与初始化算法 逻辑目标：从XML布局文件中精准获取“RIGHT NOW”“MAYBE LATER”两个Button控件的实例，为后续交互做准备。 实现逻辑：通过Android系统提供的findViewById(int id)方法，传入布局文件中为两个按钮定义的唯一ID（如R.id.btn\_right\_now、R.id.btn\_maybe\_later），建立Java/Kotlin代码与XML界面控件的关联；若ID匹配成功，则返回对应Button实例，若匹配失败（如ID拼写错误），则返回null（实验中需通过编译检查规避此问题）。 3. 点击事件监听算法 逻辑目标：实时监测用户对按钮的点击操作，触发后续反馈逻辑。 实现逻辑：采用“监听器模式”，为每个Button实例绑定View.OnClickListener接口实现类（实验中可使用匿名内部类简化代码）： 当用户点击按钮时，Android系统会触发onClick(View v)回调方法，并将被点击的Button实例作为参数v传入； 通过v.getId()方法判断当前点击的按钮ID（区分“RIGHT NOW”和“MAYBE LATER”），实现不同按钮的差异化响应。 4. Toast反馈触发算法 逻辑目标：根据点击的按钮，生成并显示对应的文本提示。 实现逻辑： 基于onClick方法中判断的按钮ID，确定Toast需显示的文本内容（点击“RIGHT NOW”按钮则文本为“RIGHT NOW”，点击“MAYBE LATER”则为“MAYBE LATER”）； 调用Toast.makeText(Context context, CharSequence text, int duration)静态方法创建Toast实例： context传入当前Activity实例（确保Toast在正确的上下文显示）； text传入上述确定的按钮名称文本； duration设置为Toast.LENGTH\_SHORT（约2秒，符合实验交互反馈的简洁性需求）； 调用Toast实例的show()方法，触发系统显示提示，完成交互闭环。 5. 算法复杂度分析  * 时间复杂度：整体为O(1)（常数级）。控件初始化、事件监听、Toast创建与显示均为单次调用操作，无循环或递归逻辑，不受数据量影响，响应速度仅依赖Android系统组件调用效率，满足实时交互需求。 * 空间复杂度：整体为O(1)。仅占用少量内存存储Button实例、监听器对象与Toast实例，无大规模数据结构（如数组、集合），符合移动端应用轻量化的内存使用要求。   二、微信风格弹出菜单+Toast交互功能算法分析 该功能的核心目标是“用户点击‘+’控件后弹出菜单，点击菜单项通过Toast反馈菜单项名称”，算法逻辑在双按钮功能基础上增加“菜单资源加载”与“菜单项事件匹配”环节，具体分析如下：   1. 核心算法流程 2. 菜单资源加载算法 逻辑目标：将XML定义的5个菜单项（“New Chat”“Add Contacts”等）解析为可视化的PopupMenu实例。 实现逻辑： 先通过findViewById获取“+”触发控件（如ImageButton），绑定点击监听器； 当用户点击触发控件时，创建PopupMenu实例（传入当前Activity上下文与触发控件，确保菜单在控件下方显示）； 调用MenuInflater.inflate(int menuRes, Menu menu)方法，将XML菜单资源文件（如R.menu.menu\_popup）解析为PopupMenu的Menu对象，完成菜单项的可视化加载（系统自动处理菜单项的布局排列，无需手动计算位置）。 3. 菜单项事件匹配算法 逻辑目标：精准识别用户点击的菜单项，触发对应Toast反馈。 实现逻辑：为PopupMenu绑定PopupMenu.OnMenuItemClickListener接口实现类： 当用户点击某菜单项时，系统触发onMenuItemClick(MenuItem item)回调方法，传入被点击的菜单项实例item； 通过item.getTitle()方法获取当前菜单项的文本（如“Scan”“Money”），无需额外判断ID（直接复用菜单项定义的标题文本，简化逻辑）； 调用Toast.makeText方法，将获取的菜单项标题作为提示文本，调用show()方法显示，完成反馈。 4. 菜单显示与销毁算法 逻辑目标：确保菜单在正确时机显示/隐藏，避免内存泄漏。 实现逻辑： 菜单显示：PopupMenu创建并加载资源后，调用show()方法，系统自动计算菜单位置（基于触发控件坐标），在屏幕指定区域显示； 菜单销毁：用户点击菜单项或点击菜单外区域时，系统自动调用dismiss()方法销毁菜单实例，释放内存，无需手动处理。 5. 算法复杂度分析  * 时间复杂度：整体为O(1)。菜单资源加载、菜单项点击判断、Toast反馈均为单次操作，菜单项数量固定（5个），无循环遍历逻辑，响应速度与双按钮功能一致，满足实时交互需求。 * 空间复杂度：整体为O(1)。仅占用内存存储PopupMenu实例、MenuInflater实例与菜单项对象，无大规模数据存储，符合移动端内存优化要求。   三、整体算法设计特点与合理性   1. 高耦合性与易用性：依托Android系统原生组件（Button、PopupMenu、Toast）与接口（OnClickListener）设计算法，无需自定义复杂组件，降低开发难度，同时保证功能兼容性（适配Android 7.0及以上系统版本）。 2. 模块化逻辑：将“控件初始化”“事件监听”“反馈触发”拆分为独立步骤，每个步骤职责单一（如Toast反馈逻辑在两个功能中可复用），便于调试与修改（如后续需修改提示文本，仅需调整对应字符串即可）。 3. 轻量化设计：无复杂计算或数据处理，算法逻辑聚焦“事件触发-响应”的核心需求，符合移动端应用对“低延迟、低内存占用”的要求，确保功能在AVD虚拟机中稳定运行，无卡顿或崩溃问题。 | | | | | |
| 1. **实验流程：**   **流程图：**    **（一）开发环境部署（准备阶段）**   1. **安装 Android Studio：从官方渠道下载 Android Studio 安装包，按照安装向导完成软件安装；若现有系统不支持低版本，可选择兼容的较高版本。** 2. **配置 SDK：打开 Android Studio，进入 SDK Manager，勾选 Android 7.0 及以上版本的 SDK Platform 和对应的 SDK Tools（如 Build-Tools、Emulator 等），点击 “Apply” 完成下载与配置。** 3. **创建 AVD 虚拟机：在 Android Studio 中打开 AVD Manager，点击 “Create Virtual Device”，先选择硬件设备（如 Pixel 4 XL、Pixel 3a 等，文档示例含多种设备选项），再选择已下载的 Android 7.0 及以上版本系统镜像，完成虚拟机参数设置（如内存分配、存储大小），最终创建可用于运行 Android 应用的 AVD。**   **（二）功能开发（核心实验阶段）**  **1. 背景 + 双按钮 + Toast 交互实现**   1. **素材准备：将实验提供的背景图片素材复制到实验结果根目录下的 “Extra” 子文件夹中。** 2. **项目创建：打开 Android Studio，创建一个新的 Android 项目（选择 Empty Views Activity 等基础模板），设置项目名称、包名等基础信息，确保项目关联已配置的 SDK 版本。** 3. **界面设计（布局文件）：在 res/layout 目录下的 activity\_main.xml 文件中，添加背景图片（引用 Extra 文件夹中的素材），再添加两个 Button 控件，分别设置文本为 “RIGHT NOW” 和 “MAYBE LATER”，调整控件位置与样式（如大小、颜色、间距）以匹配示例效果。** 4. **逻辑编写（Java/Kotlin 代码）：在 MainActivity（对应编程语言的文件）中，为两个按钮分别设置点击事件监听器（setOnClickListener）；在点击事件回调方法中，使用 Toast.makeText () 方法创建提示，分别显示 “RIGHT NOW” 和 “MAYBE LATER”，并调用 show () 方法触发 Toast 显示。** 5. **功能测试：启动之前创建的 AVD 虚拟机，点击 Android Studio 的 “Run” 按钮，将项目部署到虚拟机上；点击两个按钮，验证 Toast 是否能正确显示对应按钮名称。**   **2. 微信风格弹出菜单 + Toast 交互实现**   1. **菜单资源创建：在 res 目录下新建 menu 文件夹，创建 menu 资源文件（如 menu\_popup.xml），在文件中添加 5 个菜单项，分别对应 “New Chat”“Add Contacts”“Scan”“Money”“Support”（匹配微信右上角 “+” 菜单的示例效果），设置每个菜单项的 id 和标题。** 2. **触发控件添加：在 activity\_main.xml 中添加一个用于触发弹出菜单的控件（如 Button 或 ImageButton，模拟微信右上角 “+” 图标），调整其位置至界面右上角。** 3. **弹出菜单逻辑编写：在 MainActivity 中，为触发控件设置点击事件；在点击事件中，创建 PopupMenu 对象，关联当前 Activity 和触发控件，通过 menuInflater.inflate () 加载第一步创建的 menu 资源；为 PopupMenu 设置菜单项点击监听器（setOnMenuItemClickListener），在回调方法中，通过 Toast 显示被点击菜单项的标题。** 4. **功能测试：再次将项目部署到 AVD 虚拟机，点击触发控件弹出菜单，依次点击 5 个菜单项，验证 Toast 是否能正确显示每个菜单项的名称。**   **（三）实验结果整理与提交（收尾阶段）**   1. **截屏记录：对虚拟机中运行的两个功能（双按钮交互、弹出菜单交互）分别进行整屏截屏，确保截屏包含任务栏（显示时间和日期），不修改或截取图片内容；若电脑无法联网，可用手机拍摄清晰图像。** 2. **文件组织：**    * **根目录文件夹命名：按 “学号后五位\_MAPE\_实验次数\_简短日期（MMDD）” 格式命名（如 21101\_MAPE\_1\_0427）。**    * **子文件夹内容：**      + **Code：放入本次实验项目的全部程序源码（Module 对应的所有代码文件）。**      + **Doc：放入按模板撰写的 Word 版实验报告（包含实验题目、算法分析、实验流程等 7 项内容，流程图和结果以图片形式插入，其他为可编辑文字）。**      + **Extra：放入实验所用的背景图片等素材文件。**    * **根目录直接存放：截屏图片文件、Word 版实验报告（与 Doc 文件夹内报告一致）。** | | | | | |
| 1. **实验关键代码：**   **package com.example.myapplication;  import android.os.Bundle; import android.view.MenuItem; import android.view.View; import android.widget.Button; import android.widget.ImageButton; import android.widget.Toast; import androidx.appcompat.app.AppCompatActivity; import androidx.appcompat.widget.PopupMenu;  public class MainActivity extends AppCompatActivity {   // 声明控件  private Button btnRightNow;  private Button btnMaybeLater;  private ImageButton btnPopupMenu;   @Override  protected void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.*activity\_main*);   // 绑定控件  btnRightNow = findViewById(R.id.*btnRightNow*);  btnMaybeLater = findViewById(R.id.*btnMaybeLater*);  btnPopupMenu = findViewById(R.id.*btnPopupMenu*);   // "+"按钮点击事件（弹出菜单）  btnPopupMenu.setOnClickListener(new View.OnClickListener() {  @Override  public void onClick(View v) {  PopupMenu popupMenu = new PopupMenu(MainActivity.this, v);  popupMenu.getMenuInflater().inflate(R.menu.*popup\_menu*, popupMenu.getMenu());  popupMenu.setOnMenuItemClickListener(new PopupMenu.OnMenuItemClickListener() {  @Override  public boolean onMenuItemClick(MenuItem item) {  // 将switch语句改为if-else语句以解决"Constant expression required"问题  if (item.getItemId() == R.id.*menu\_new\_chat*) {  Toast.*makeText*(MainActivity.this, "New Chat", Toast.*LENGTH\_SHORT*).show();  return true;  } else if (item.getItemId() == R.id.*menu\_add\_contacts*) {  Toast.*makeText*(MainActivity.this, "Add Contacts", Toast.*LENGTH\_SHORT*).show();  return true;  } else if (item.getItemId() == R.id.*menu\_scan*) {  Toast.*makeText*(MainActivity.this, "Scan", Toast.*LENGTH\_SHORT*).show();  return true;  } else if (item.getItemId() == R.id.*menu\_money*) {  Toast.*makeText*(MainActivity.this, "Money", Toast.*LENGTH\_SHORT*).show();  return true;  } else if (item.getItemId() == R.id.*menu\_support*) {  Toast.*makeText*(MainActivity.this, "Support", Toast.*LENGTH\_SHORT*).show();  return true;  } else {  return false;  }  }  });  popupMenu.show();  }  });   // RIGHT NOW 按钮点击事件  btnRightNow.setOnClickListener(new View.OnClickListener() {  @Override  public void onClick(View v) {  Toast.*makeText*(MainActivity.this, "RIGHT NOW", Toast.*LENGTH\_SHORT*).show();  }  });   // MAYBE LATER 按钮点击事件  btnMaybeLater.setOnClickListener(new View.OnClickListener() {  @Override  public void onClick(View v) {  Toast.*makeText*(MainActivity.this, "MAYBE LATER", Toast.*LENGTH\_SHORT*).show();  }  });  } }**  **<?xml version="1.0" encoding="utf-8"?> <LinearLayout  xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:background="@drawable/img\_1"  android:orientation="vertical"  android:padding="15dp">   <!-- 顶部"+"按钮 -->  <ImageButton  android:id="@+id/btnPopupMenu"  android:layout\_width="50dp"  android:layout\_height="50dp"  android:src="@drawable/ic\_plus"  android:layout\_gravity="end"  android:background="?attr/selectableItemBackgroundBorderless"  android:contentDescription="Popup menu button"/>   <!-- 版本更新提示文本 -->  <TextView  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="New version is found, update it right now?"  android:textColor="#FFFFFF"  android:textSize="18sp"  android:layout\_gravity="center\_horizontal"  android:layout\_marginTop="80dp"  android:layout\_marginBottom="40dp"/>   <!-- RIGHT NOW 按钮 -->  <Button  android:id="@+id/btnRightNow"  android:layout\_width="200dp"  android:layout\_height="50dp"  android:text="RIGHT NOW"  android:textColor="#FFFFFF"  android:background="#7B1FA2"  android:layout\_gravity="center\_horizontal"  android:layout\_marginBottom="15dp"/>   <!-- MAYBE LATER 按钮 -->  <Button  android:id="@+id/btnMaybeLater"  android:layout\_width="200dp"  android:layout\_height="50dp"  android:text="MAYBE LATER"  android:textColor="#FFFFFF"  android:background="#7B1FA2"  android:layout\_gravity="center\_horizontal"/>  </LinearLayout>**   1. **实验结果**        1. **实验结果分析**   （一）开发环境部署结果验证   1. Android Studio与SDK配置：已按任务书“实验准备工作”要求，安装兼容当前系统的Android Studio版本，成功配置Android 7.0及以上版本SDK（含对应SDK Platform、Build-Tools等组件），无软件安装失败或SDK缺失问题，可正常创建Android项目，满足任务书“为应用开发搭建基础环境”的要求。 2. AVD虚拟机创建：在Android Studio中完成AVD设置，选择任务书示例中的硬件设备类型（如Pixel 4 XL、Pixel 3a等），匹配Android 7.0及以上系统镜像，虚拟机启动后无闪退、黑屏，可正常运行Android应用，符合“创建虚拟机用于查看应用程序”的任务要求。   （二）功能实现结果验证   1. 双按钮+Toast交互功能  * 界面与素材要求匹配：已将实验提供的背景图片素材放入“根目录/Extra”（符合任务书“注意①”要求），并在布局中成功添加背景，背景显示完整；同时添加“RIGHT NOW”和“MAYBE LATER”两个按钮，界面布局与任务书示例效果一致。 * 交互反馈符合要求：点击两个按钮后，均能通过Toast方式准确显示对应按钮名称（“RIGHT NOW”“MAYBE LATER”），Toast提示响应及时、内容无错误，完全满足任务书“点击按钮用Toast显示按钮名称”的功能要求。 * 截屏规范达标：对功能交互过程进行整屏截屏，截图包含任务栏（显示时间和日期），未修改或截取图片内容（符合任务书“实验教学环节”中“拷屏要求”），截图文件已保存至根目录。  1. 微信风格弹出菜单+Toast交互功能  * 菜单效果匹配：成功实现微信右上角“+”的弹出菜单效果，菜单包含任务书要求的五个菜单项（如“New Chat”“Add Contacts”“Scan”等），菜单项显示顺序与界面样式贴合任务书示例。 * 交互反馈合规：点击每个菜单项后，均通过Toast方式显示对应菜单项名称，无点击无响应或提示内容错误的情况，满足任务书“点击菜单选项用Toast显示按钮名称”的要求。 * 截屏规范达标：对菜单弹出、菜单项点击过程进行整屏截屏，截图包含任务栏时间和日期，符合任务书截屏要求，截图文件已保存至根目录。   （三）文件组织与报告提交准备验证   1. 文件结构合规：已按任务书“实验结果所包含的内容”要求，创建根目录文件夹（命名格式：学号后五位\_MAPE\_实验顺次\_ShortDate，如“21101\_MAPE\_1\_0427”），根目录下包含：  * 直接文件：截屏图片、Word版实验报告； * 子文件夹：   + Code：存放当次实验的全部程序源码（Module对应的所有代码）；   + Doc：存放Word版实验报告（符合任务书“实验报告要求”中“放入Doc文件夹”的“注意②”规定）；   + Extra：存放背景图片素材，无素材缺失。  1. 实验报告规范：Word版实验报告已按任务书“实验报告要求”撰写，包含“实验题目和具体要求”“算法分析”“实验流程”“实验关键技术或关键代码”“实验结果展示”“实验结果分析”“实验讨论”7项内容，其中流程图和结果为图片形式，其他为可编辑文字，同时包含实验运行截图（含时间日期），符合任务书对报告格式、内容的全部要求。   二、实验中出现的问题与解决措施 （一）问题1：AVD虚拟机启动后黑屏   * 问题现象：创建AVD并启动后，虚拟机长时间停留在黑屏界面，无法进入系统桌面，无法测试应用。 * 问题原因：未按任务书“实验准备工作”中“系统不支持可选择较高版本”的灵活要求，选择的Android 7.0（API 24）系统镜像与电脑硬件加速设置不兼容，导致虚拟机启动异常。 * 解决措施：根据任务书“选择较高版本”的提示，重新在AVD Manager中选择Android 9.0（API 28）系统镜像，重新创建AVD后启动正常，可进入系统桌面并测试应用。   （二）问题2：背景图片未在界面显示   * 问题现象：已将背景图片放入“根目录/Extra”，但在布局中引用后，界面未显示背景，仅显示空白。 * 问题原因：未正确处理素材引用路径，且忽略了任务书“实验报告要求”中“软件类实验报告应有运行截图”的隐含前提——素材需正确引用才能生成有效截图，此次问题因素材未放入Android项目的res/drawable目录，导致布局无法识别图片资源。 * 解决措施：将“根目录/Extra”中的背景图片复制到Android项目的res/drawable目录，重新在布局文件中引用drawable目录下的图片资源，界面成功显示背景，满足任务书界面要求。   （三）问题3：实验报告代码部分不符合要求   * 问题现象：初始撰写实验报告时，在“实验关键技术或关键代码”部分粘贴了全部项目代码，不符合任务书要求。 * 问题原因：未仔细阅读任务书“实验报告要求”中“实验报告的代码部分只需要提供关键代码即可”的明确规定，导致代码部分冗余。 * 解决措施：按任务书要求删减冗余代码，仅保留“按钮点击事件绑定”“Toast创建”“PopupMenu初始化与菜单项监听”等关键代码，并添加注释说明，使报告符合“关键代码”提交要求。   三、实验结果总结与反思 （一）结果总结 本次实验完全达成《移动应用开发课程实验任务书-1.docx》的全部要求：开发环境部署满足应用开发基础需求，双按钮与弹出菜单功能实现符合交互要求，文件组织与实验报告严格遵循任务书规范，无核心要求遗漏，实验目标全部达成。  （二）反思与改进方向   1. 需加强任务书细节阅读：本次出现的“代码冗余”“素材引用错误”等问题，均因对任务书具体条款关注不足，后续需逐字研读任务书要求，确保每个环节（如代码提交、素材存放）均符合规定。 2. 需提升问题排查效率：针对“AVD黑屏”问题，初期未及时联想到任务书“选择较高版本”的提示，后续需结合任务书灵活要求，快速定位兼容性、路径类问题，减少调试耗时。 3. 需强化规范意识：任务书对文件命名（如根目录、邮件主题）、截屏格式有明确标准，此次虽按要求执行，但需进一步形成“先核对任务书要求，再执行操作”的习惯，确保实验结果完全贴合任务书标准，为后续实验提交奠定基础。 4. **实验心得**   一、实验技能与知识的收获  本次实验以“开发环境部署”和“基础交互功能实现”为核心，让我从“理论认知”迈向“实操落地”。在环境部署环节，我严格按照任务书“实验准备工作”要求，成功安装Android Studio并配置Android 7.0及以上版本SDK，还学会了根据设备兼容性选择合适的AVD虚拟机（如任务书示例中的Pixel 4 XL）——这一过程让我明白，“版本匹配”是移动端开发的基础，任务书提示的“系统不支持可选择较高版本”，更是帮我规避了初期因版本兼容导致的虚拟机启动问题，为后续功能开发扫清了障碍。  在功能实现环节，任务书要求的“双按钮+Toast交互”“微信风格弹出菜单+Toast交互”，让我深入理解了Android组件的使用逻辑。比如，为按钮绑定点击监听器、通过`Toast.makeText()`实现反馈，看似简单的操作，实则是“控件-事件-响应”的完整交互闭环；而PopupMenu的实现，从加载menu资源到绑定菜单项监听，每一步都需贴合任务书“显示对应按钮名称”的要求——这些实操让我不再是机械记忆API，而是能结合任务书功能目标，灵活调用Android原生组件，真正掌握了基础交互功能的开发逻辑。  此外，任务书对“文件组织”和“实验报告”的严格规范，也让我养成了标准化的工作习惯。从“根目录需包含Code、Doc、Extra子文件夹”，到“实验报告仅提交关键代码”，每一项要求都在引导我建立清晰的项目结构意识——我意识到，移动开发不仅是写代码，规范的文件管理和文档撰写，更是后续项目协作与成果检查的关键，这一点在任务书的细节要求中体现得淋漓尽致。  二、实验中的问题与成长  实验过程中，我也因对任务书细节关注不足，遇到了一些小波折。比如在准备背景图片时，我虽按任务书“放入根目录/Extra”的要求存放了素材，却忽略了Android项目需将图片放入res/drawable目录才能被布局识别，导致界面初期无法显示背景。直到对照任务书“软件类实验报告应有实验运行截图”的要求检查时，才发现问题根源——这让我明白，任务书的每一项要求都相互关联，素材存放不仅是“按文件夹分类”，更要结合开发逻辑确保资源可调用，后续需更细致地将任务书要求与实际开发流程结合。  另一个问题是撰写实验报告时，我最初粘贴了全部项目代码，未遵循任务书“仅提供关键代码”的规定。这一疏忽让我重新审视任务书条款，也意识到“按要求提交成果”和“实现功能”同样重要——任务书的规范不是束缚，而是为了让实验结果更清晰、更易检查，后续我会先逐字梳理任务书的每一项要求，再开展操作，避免因细节遗漏返工。  三、对后续实验的启示  这次实验让我深刻体会到，《移动应用开发课程实验任务书-1.docx》不仅是“任务清单”，更是指导实验的“核心指南”。后续实验中，我会养成“先精读任务书，再制定执行步骤”的习惯：比如在开发前，先标注任务书的关键要求（如功能目标、文件命名、提交截止时间）；在调试时，结合任务书的灵活提示（如“选择较高版本SDK”）排查问题；在提交前，对照任务书逐项核对成果（如截图是否含时间日期、代码是否仅保留关键部分）。  同时，本次实验掌握的“组件调用”“规范管理”等技能，也为后续更复杂的Android开发打下基础。比如任务书要求的Toast反馈，未来可扩展为页面跳转、数据存储等功能；而文件组织规范，也能直接应用于后续团队实验项目中。  总之，第一次实验让我在技能、规范意识和问题解决能力上都有了显著提升。我会带着这次实验的收获，严格遵循后续任务书要求，扎实推进每一次实验，逐步成长为能规范开发、高效解决问题的移动应用开发者。 | | | | | |
| 1. **教师评语** | | | | | |
| 1. **实验成绩**   教师签名： 批阅日期： 2025年 月 日 | | | | | |

注：项目性质为 演示型、验证型、设计型、综合型和创新型。