桂林电子科技大学2023-2024学年 第2学期

**Android应用开发 实验报告**

**课号：2322143         任课老师：刘斯韵**

|  |  |  |  |
| --- | --- | --- | --- |
| 项目名称 | **实验二、控件及布局** | | |
| 院系 | **计算机与信息安全学院** | 专业 | **物联网工程** |
| 学号 | 2200100501 | 姓名 | **韩睿鑫** |
| 实验日期 | **2024年4月2日** | | |

**一、实验目的**

掌握Activity布局的基本⽤法；

掌握Android中各个控件的基本使用。

**二、实验内容**

1. 创建不同Layout，熟悉LinearLayout、RelativeLayout、Gird Layout、TableLayout等布局的使用；

2. 对布局中的单位：mm\pt\px\dp\in\sp\wrap\_content\match\_parent\%等效果的理解和熟悉；

3. 搭建一个矩形填图游戏界面，如图2.1所示；

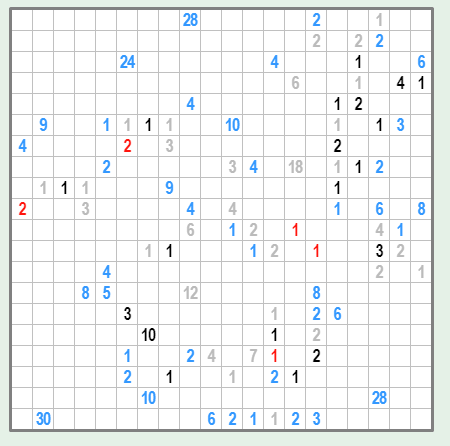


图2.1 矩形填图界面

4. 设置其中数字和空格的事件响应。

**三、实验过程和代码**

过程步骤和截图，以及代码分析

 第一步构建一个xml布局文件

|  |
| --- |
| <?*xml version*="1.0" *encoding*="utf-8"?> <RelativeLayout *xmlns:android*="http://schemas.android.com/apk/res/android"  *android:layout\_width*="match\_parent"  *android:layout\_height*="match\_parent"  *android:background*="#E0F5FF"  *xmlns:app*="http://schemas.android.com/apk/res-auto">   *<!-- 使用RelativeLayout实现游戏标题 -->* <TextView  *android:id*="@+id/text\_title"  *android:layout\_width*="wrap\_content"  *android:layout\_height*="wrap\_content"  *android:text*="矩形填图游戏"  *android:textSize*="24sp"  *android:layout\_centerHorizontal*="true"  *android:layout\_marginTop*="20dp"/>   *<!-- 使用LinearLayout实现游戏说明 -->* <LinearLayout  *android:id*="@+id/layout\_instructions"  *android:layout\_width*="match\_parent"  *android:layout\_height*="wrap\_content"  *android:orientation*="vertical"  *android:layout\_below*="@id/text\_title"  *android:layout\_marginTop*="20dp"  *android:layout\_marginHorizontal*="40dp">   <TextView  *android:layout\_width*="match\_parent"  *android:layout\_height*="wrap\_content"  *android:text*="游戏说明："  *android:textSize*="18sp"/>   <TextView  *android:layout\_width*="match\_parent"  *android:layout\_height*="wrap\_content"  *android:text*="1. 将所有矩形填满。"  *android:textSize*="16sp"/>   </LinearLayout>   <LinearLayout  *android:id*="@+id/grid\_layout"  *android:layout\_width*="match\_parent"  *android:layout\_height*="550dp"  *android:layout\_below*="@id/layout\_instructions"  *android:layout\_marginTop*="17dp"  *android:alignmentMode*="alignMargins"  *android:background*="#E1CEE6B2"  *android:layout\_margin*="7dp"  *android:orientation*="horizontal"  *android:rowOrderPreserved*="false">   <androidx.recyclerview.widget.RecyclerView  *android:id*="@+id/recycler\_view"  *android:layout\_width*="match\_parent"  *android:layout\_height*="match\_parent"  *android:layout\_margin*="9dp"  *android:layout\_marginStart*="9dp"  *android:layout\_marginTop*="9dp"  *android:layout\_marginEnd*="9dp"  *android:layout\_marginBottom*="9dp" />    </LinearLayout>   <TableLayout  *android:layout\_width*="match\_parent"  *android:layout\_height*="wrap\_content"  *android:layout\_alignParentBottom*="true">   <TableRow>   <Button  *android:id*="@+id/btn\_reset"  *android:layout\_width*="0dp"  *android:layout\_height*="wrap\_content"  *android:layout\_weight*="1"  *android:text*="重置"/>   <Button  *android:id*="@+id/btn\_hint"  *android:layout\_width*="0dp"  *android:layout\_height*="wrap\_content"  *android:layout\_weight*="1"  *android:text*="提示"/>   </TableRow>   </TableLayout>  </RelativeLayout> |

利用RecylerView布局控件，创建item，绘制矩形格子

Item的xml布局如下：

|  |
| --- |
| <?*xml version*="1.0" *encoding*="utf-8"?> <RelativeLayout *xmlns:android*="http://schemas.android.com/apk/res/android"  *android:layout\_width*="47dp"  *android:layout\_height*="42dp"  *android:orientation*="horizontal"  *android:background*="@drawable/text\_bg">   <TextView  *android:id*="@+id/text\_hh"  *android:layout\_width*="wrap\_content"  *android:layout\_height*="wrap\_content"  *android:text*="123"  *android:layout\_centerInParent*="true"  *android:textSize*="20dp"  *android:textColor*="#117BCF"/> </RelativeLayout> |

为了使格子具有边框，使用shape自定义text的背景文件

|  |
| --- |
| <?*xml version*="1.0" *encoding*="utf-8"?> <shape *xmlns:android*="http://schemas.android.com/apk/res/android" >  <stroke  *android:width*="3px"  *android:color*="#696969"/><solid  *android:color*="#F8F8F8" /><padding  *android:left*="5dp"  *android:top*="5dp"  *android:right*="5dp"  *android:bottom*="5dp"/> </shape> |

接下来到主界面编辑了

|  |
| --- |
| class hhActivity : CommunicationActivity() {  private val dataSet = *mutableListOf*<String>()  private lateinit var recyclerView: RecyclerView  private lateinit var adapter: MyAdapter   override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  setContentView(R.layout.*test2*)  *//val randomNum = Random.nextInt(100).toString()  repeat*(10) **{** dataSet.add(" ")  **}** dataSet.add("1")  *repeat*(5) **{** dataSet.add(" ")  **}** dataSet.add("77")  *repeat*(20) **{** dataSet.add(" ")  **}** dataSet.add("22")  *repeat*(5) **{** dataSet.add(" ")  **}** dataSet.add("45")  *repeat*(7) **{** dataSet.add(" ")  **}** dataSet.add("56")  *repeat*(20) **{** dataSet.add(" ")  **}** dataSet.add("41")  *repeat*(4) **{** dataSet.add(" ")  **}** dataSet.add("11")  *repeat*(9) **{** dataSet.add(" ")  **}** dataSet.add("90")  *repeat*(5) **{** dataSet.add(" ")  **}** dataSet.add("64")  *repeat*(2) **{** dataSet.add(" ")  **}** recyclerView = findViewById(R.id.*recycler\_view*)  val layoutManager = GridLayoutManager(this, 8)  recyclerView.*layoutManager* = layoutManager   *// 创建并设置适配器* adapter = MyAdapter(this,dataSet)  recyclerView.*adapter* = adapter  }   fun addItem(view: View) {   adapter.notifyDataSetChanged()  } } |

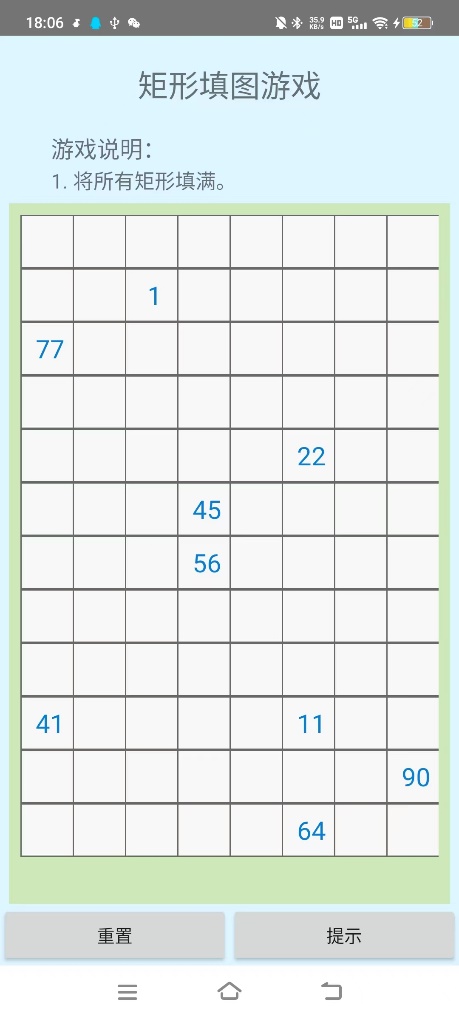
在主界面中找到recyclerView为其设置自定义的适配器，将其设置为网格布局且每行有8列，其中用repeat给dataSet数据列中传入初始数据。

接下来是为其自定义的适配器

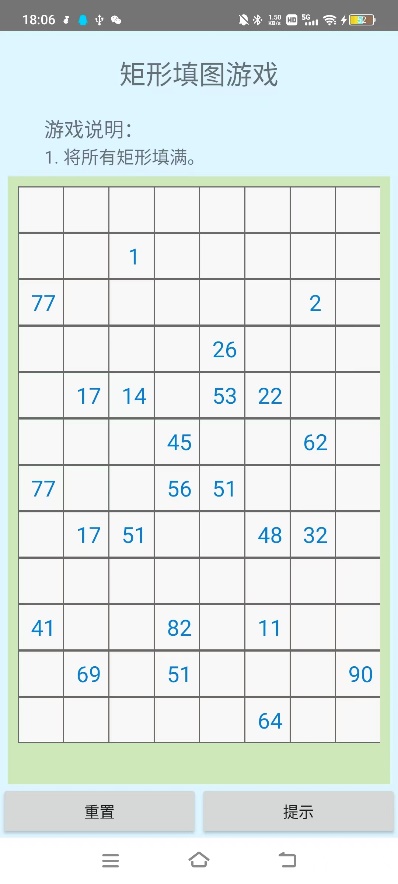
|  |
| --- |
| class MyAdapter(private val context: Context, private val dataSet: *List*<String>) :  RecyclerView.Adapter<MyAdapter.ViewHolder>() {   *// 创建 ViewHolder，绑定 item 布局* override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): ViewHolder {  val view = LayoutInflater.from(context).inflate(R.layout.*hh\_item*, parent, false)  return ViewHolder(view)  }   *// 绑定 ViewHolder 中的数据* override fun onBindViewHolder(holder: ViewHolder, position: Int) {  val item = dataSet[position]  holder.bind(item)  }   *// 返回数据集合的大小* override fun getItemCount(): Int {  return 96  }   *// ViewHolder 内部类* inner class ViewHolder(itemView: View) : RecyclerView.ViewHolder(itemView) {  private val textView: TextView = itemView.findViewById(R.id.*text\_hh*)   fun bind(item: String) {  textView.*text* = item  *// 设置点击事件监听器* itemView.setOnClickListener **{** val randomNumber = generateRandomNumber()  textView.*text* = "$randomNumber"  **}** }  }   private fun generateRandomNumber(): Int {  *// 生成一个在 0 到 100 之间的随机数（不包括 100）* return Random.nextInt(100)  } } |

继承父类RecyclerView.Adapter，实现其中的方法，设置ViewHolder内部类,实现每个item的点击监听事件，这里的监听事件是生成随机数并且在item中的text中显示出来。

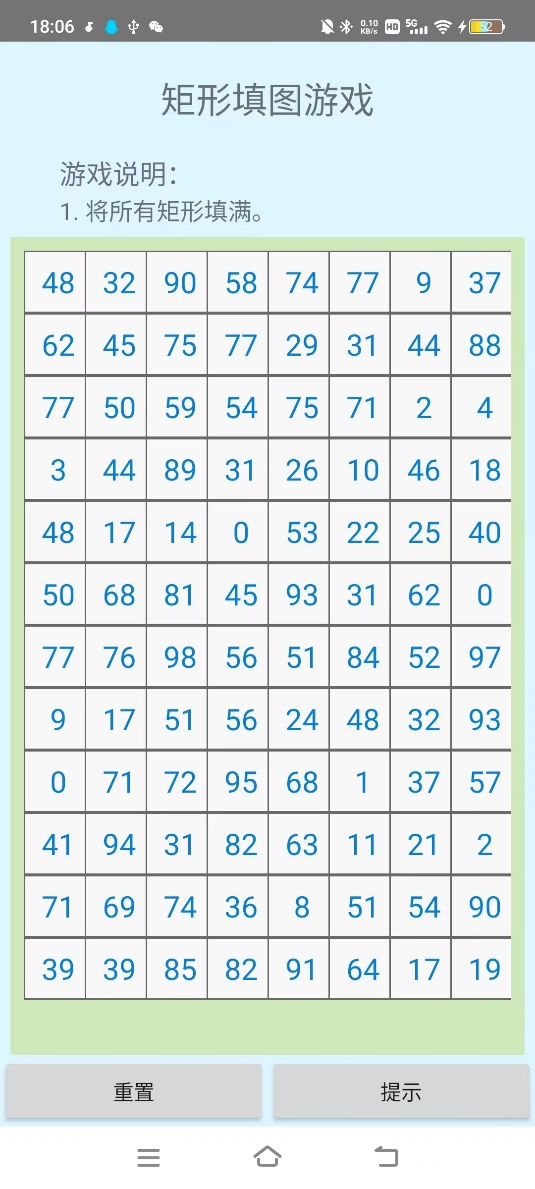
 运行图片如下：



 点击多次后：



全部点击后：



**四、遇到问题和解决办法**

在遇到问题后我在csdn上寻求解决办法，学会了如何使用shape自定义一个边框，一开始我设置了100个textview在Gird Layout中，但是这样做代码效率太低了，而且写的布局不美观，因此我找到了RecyclerView，利用其网格布局实现矩形格子的设计，大大避免了代码冗杂，减少内存。

在今后的学习中我会更加注重学习android，利用一些组件、控件、自定义控件布局等等实现优化我的APP，使其更加完美！