《移动应用软件开发》实验报告

年级、专业、班级	2021 级计算机科学与技术 3 班	姓名	刘成	学号	20215005
实验题目	数字华容道游戏 APP 开发				
实验时间	2023.11.5	实验地点	DS1401		
学年学期	2023-2024（1）	实验性质	□验证性 ■设计性 □综合性		

一、实验目的

1. 本次实验的目的是掌握华为 DevEco Studio 开发工具的基本操作，了解所开发 APP 项目的基本结构。掌握 Ability 框架和 UI 框架的使用，掌握常用组件和布局的使用，尤其图像组件的使用，位图的处理等。
2. 确定开发平台为 HarmonyOS，开发工具华为 DevEco Studio，建议选择最新版。可以使用 Java 语言，也可以使用 eTS 语言开发。
3. 设计完成一个数字华容道游戏 APP。

二、实验项目内容

从多张图片中，选择一张，分割为 4 块以上的小图片，打乱后分布，可以拼成原图，数字正确排列。

评分点：（1）可以从多张图片中选择一张图片，然后分割成小图片，每个图片上面**标记上数字**；（2）通过开始按钮开始游戏，把分隔后的图片打乱分布，显示在界面上，并开始游戏；（3）通过点击图片进行移动。（4）游戏过程中，自动判断是否完成游戏。（5）设置结束按钮提前结束游戏，显示正确结果；（6）APP 界面自行设计，不少于两个。（7）可以自行增加游戏计时、计分、关卡、动画、历史记录、从相册选择图片、游戏指南等功能，（8）复杂度、美观度、自定义功能、多设备协同等都是加分项。

提交：（1）本实验报告，（2）源代码压缩文件 zip，注意源代码加注释。（3）软件演示的 MP4 视频，视频大小不超过 20M。能够在搜狗浏览器正常播放。注意文件名称的规范性。文件名：学号姓名 2.doc，学号姓名 2.zip, 学号姓名 2.mp4。三个文件分别提交。

三、实验过程或算法（写明设计思想、程序的结构、功能关系图、类的说明和类之间的关系图、程序主要执行流程图，最后是核心源代码，截图等）

1. 设计思想

1.1 图片碎片及数字标记的实现方式

选取 5 张图片作为拼图的资源图片，通过编写 python 脚本，调用 PIL 库，对选取的图片进行裁剪缩放，以及对图片分割和标记的作用。

1.2 拼图难度和不同图片的选择实现方式

选取 5 张图片作为拼图资源，其中两张分割为 3*3 的碎片，2 张分割为 4*4 的碎片，最后一张分割为 5*5 的碎片，并为每一张图片单独制作游戏页面，调用不同的图片资源。在页面中通过用户自主选择（翻页）的方式，进入不同的关卡进行体验。

1.3 拼图堆积打乱、拼图移动、验证通关的实现方式

将拼图通过 DirectionalLayout 的布局方式，通过水平和垂直布局，将拼图碎片按顺序排列到页面中。首先，定义随机数，对拼图中每两个碎片判断是否进行交换，并使用数组 imageIndex 记录下打乱后的图片排列顺序，再对获取到的拼图碎片的位置渲染上对应的打乱后的图片资源，从而达到随机打乱的操作。

再使用 tmpImageIndex 对数组信息进行拷贝，以实现重试当前关卡、还原成最初形式的功能。定义变量 blackImg 用于记录空白的拼图碎片，在点击拼图碎片进行游戏时，首先判断当前拼图碎片和空白位置的距离是否为 1，如果不是则无法移动；若距离为 1，则修改空白位置的图片资源，并对原来图片碎片位置资源设置为不可见，同时修改 imageIndex 和 blackImg 等变量信息，从而达到拼图的效果。

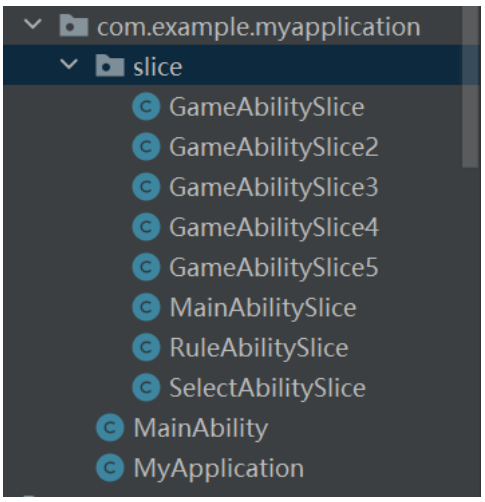
同时，在每次移动后判断当前的 imageIndex 中的数组排列顺序是否正确，如果正确则游戏结束，弹出提示框提示本次游戏的使用时间和累计步骤。

1.4 时间计时和步骤累计的实现方式

设置变量 cur 用于记录移动的操作次数，需要注意的是若移动不合法（即当前拼图碎片和空白位置的距离不为 1），不可以计算累计步骤；设置 text 子类 tickTimer 用于记录使用时间，将时间基准初始化为当前时间，需要注意的是每次进行重试、通关后再来一次操作时都需要重新初始化时间基准。

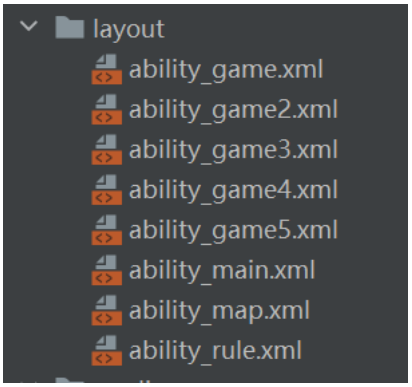
2. 程序结构

2.1 com.application.myapplication.slice 下的 java 程序

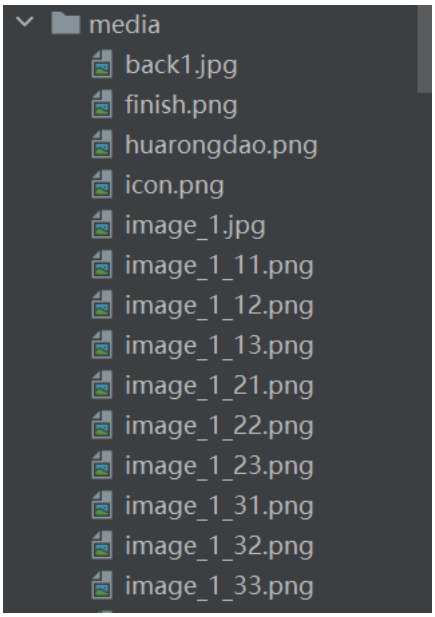


GameAbilitySlice(2,3,4,5)：游戏界面的程序代码
MainAbilitySlice：游戏首页的程序代码
RuleAbilitySlice：规则介绍界面的程序代码
SelectAbilitySlice：选择关卡界面的程序代码

2.2 resources.base.layout 下的布局文件



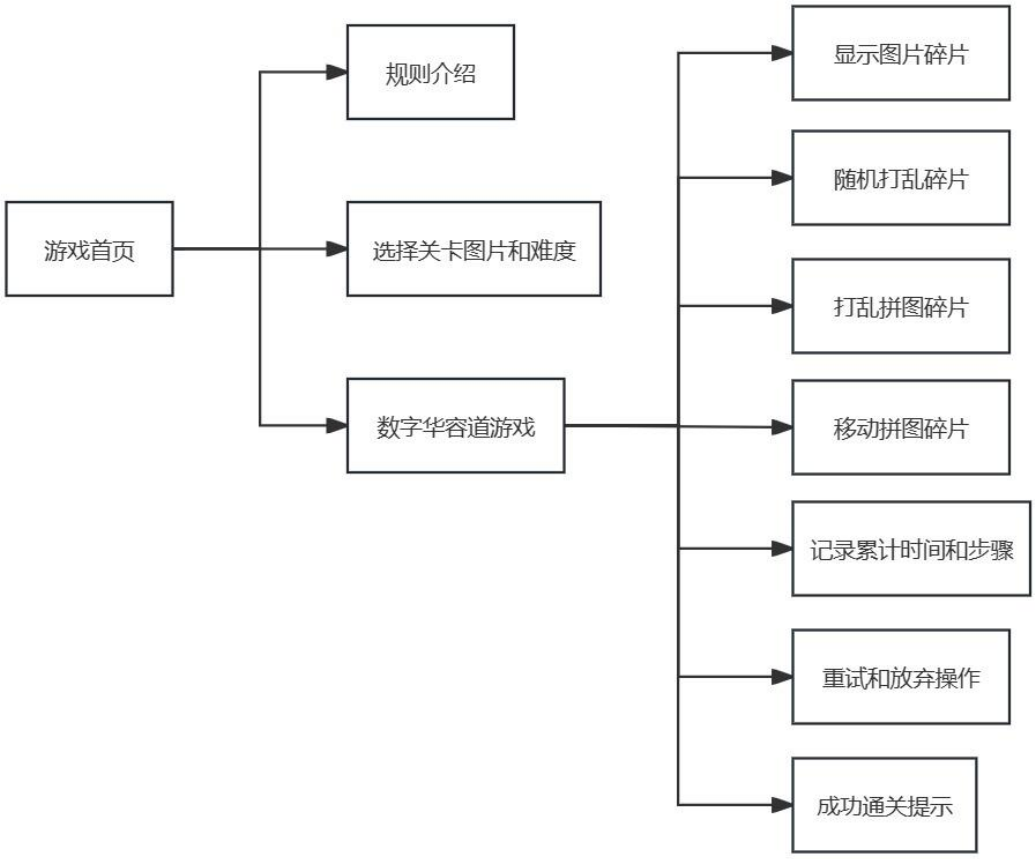
ability_game(2,3,4,5)：游戏界面的布局文件
ability_main：游戏首页的布局文件
ability_rule：规则介绍界面的布局文件
ability_map：选择关卡界面的布局文件
2.3 resources.base.media 中的图片资源文件（仅列举部分）



用于储存拼图图片和拼图碎片、各种提示字符的图片资源（返回、放弃、标题等）

2.4 config.json 全局设置文件

3. 功能关系图



4. 类的说明和类之间的关系图

4.1 GameAbilitySlice(2,3,4,5).java

5 张拼图(3*3，4*4，5*5)的核心操作代码,定义了资源初始化、随机打乱、拼图移动的点击事件、游戏结束的判断、重试操作、放弃操作以及重新打乱操作的方法函数。

4.2 MainAbilitySlice.java

定义了主界面中“开始游戏”和游戏规则的点击事件，定位到不同的页面。

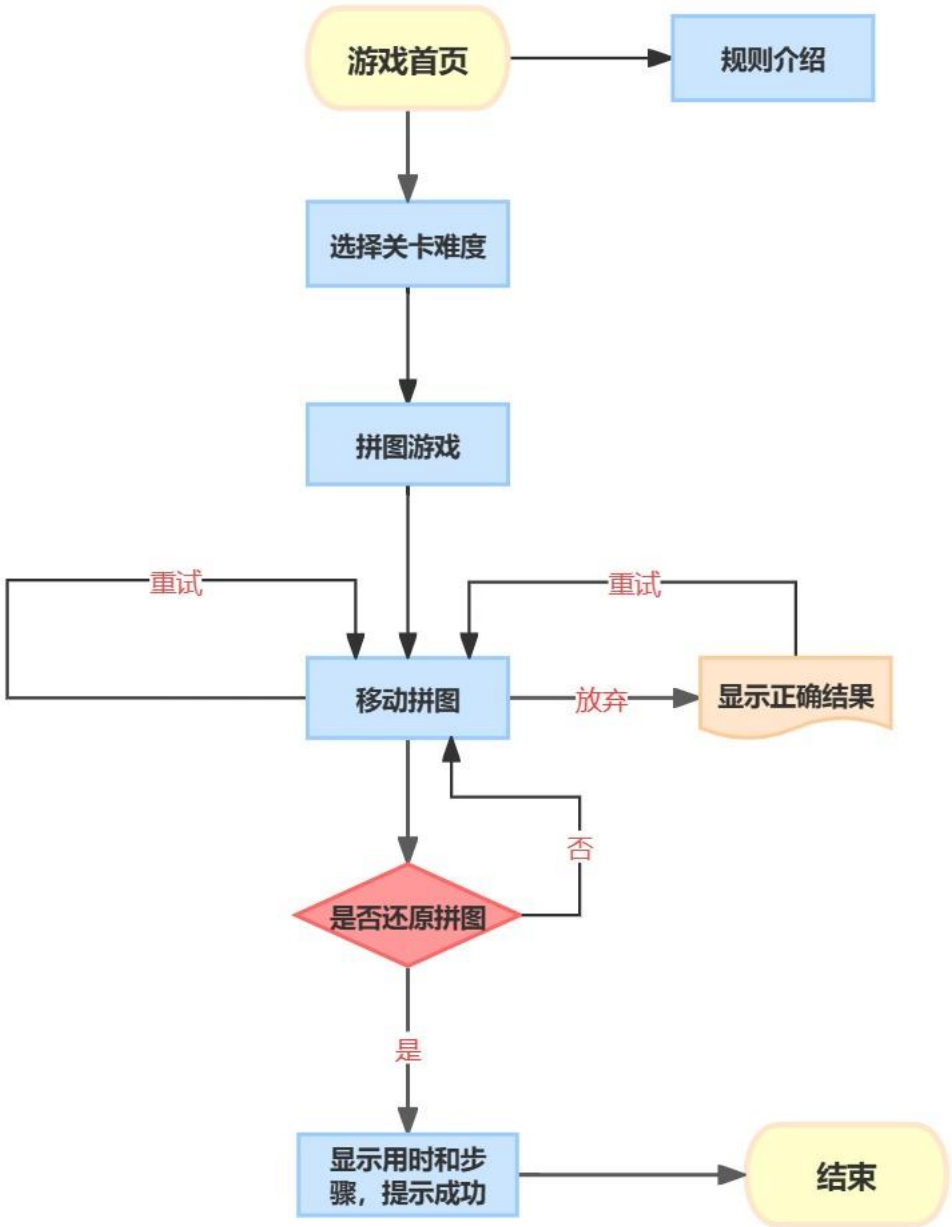
4.3 RuleAbilitySlice.java

定义了游戏规则介绍界面中“返回首页”的点击事件，定位到游戏首页。

4.4 SelectAbilitySlice.java

实现了选择关卡图片和难度的相关功能，定义了向左、向右选择图片的点击事件，以及预览图的图片资源更换的点击事件。同时定义了“返回主页”和“选好了”两个选项的点击事件。

5. 程序主要流程图



6. 核心代码

6.1.1 游戏首页布局代码

```
<?xml version="1.0" encoding="utf-8"?>
<DirectionalLayout
    xmlns:ohos="http://schemas.huawei.com/res/ohos"
    ohos:background_element="$media:back1"
    ohos:height="match_parent"
    ohos:width="match_parent"
    ohos:alignment="center"
    ohos:orientation="vertical">

    <Image
```

```
        ohos:height="match_content"
        ohos:width="match_content"
        ohos:image_src="$media:huarongdao"
        ohos:layout_alignment="center"
        ohos:bottom_margin="300px"
        />

    <Image
        ohos:id="$+id:start"
        ohos:height="match_content"
        ohos:width="match_content"
        ohos:image_src="$media:start"
        ohos:layout_alignment="center"
        ohos:top_margin="40px"
        />

    <Image
        ohos:id="$+id:rule"
        ohos:height="match_content"
        ohos:width="match_content"
        ohos:image_src="$media:rule"
        ohos:layout_alignment="center"
        ohos:top_margin="80px"
        />

</DirectionalLayout>
```

6.1.2 游戏首页逻辑代码

```
public class MainAbilitySlice extends AbilitySlice {
    @Override
    public void onStart(Intent intent) {
        super.onStart(intent);
        super.setUIContent(ResourceTable.Layout_ability_main);
        getWindow().addFlags(
                WindowManager.LayoutConfig.MARK_TRANSLUCENT_STATUS|
                WindowManager.LayoutConfig.MARK_TRANSLUCENT_NAVIGATION);

        // 获取“开始游戏”图片组件
        Image start = (Image) findComponentById(ResourceTable.Id_start);
        // 为“开始游戏”图片组件绑定点击事件，跳转到游戏界面
        start.setClickedListener(new Component.ClickedListener() {
            @Override
            public void onClick(Component component) {
                present(new SelectAbilitySlice(), new Intent());
```

```
            }
        });

        // 获取“游戏规则”图片组件
        Image rule = (Image) findComponentById(ResourceTable.Id_rule);
        // 为“游戏规则”图片组件绑定点击事件，跳转到游戏规则界面
        rule.setClickedListener(new Component.ClickedListener() {
            @Override
            public void onClick(Component component) {
                present(new RuleAbilitySlice(), new Intent());
            }
        });
    }
}
```

6.2.1 游戏规则布局代码

```
<DirectionalLayout
    xmlns:ohos="http://schemas.huawei.com/res/ohos"
    ohos:background_element="$media:back1"
    ohos:height="match_parent"
    ohos:width="match_parent"
    ohos:orientation="vertical">

    <Image
        ohos:height="match_content"
        ohos:width="match_content"
        ohos:image_src="$media:huarongdao"
        ohos:layout_alignment="horizontal_center"
        ohos:top_margin="300px"
        />

    <Image
        ohos:height="match_content"
        ohos:width="match_content"
        ohos:image_src="$media:introduce"
        ohos:layout_alignment="horizontal_center"
        ohos:top_margin="280px"
        ohos:start_margin="50px"
        />

    <Image
        ohos:id="$+id:returnHome"
        ohos:height="match_content"
        ohos:width="match_content"
```

```
        ohos:image_src="$media:returnHome"
        ohos:layout_alignment="horizontal_center"
        ohos:top_margin="150px"
        />

</DirectionalLayout>
```

6.2.2 游戏规则逻辑代码

```
public class RuleAbilitySlice extends AbilitySlice {
    @Override
    public void onStart(Intent intent) {
        super.onStart(intent);
        super.setUIContent(ResourceTable.Layout_ability_rule);

        // 获取“返回首页”图片组件
        Image returnHome = (Image)
findComponentById(ResourceTable.Id_returnHome);
        // 为“返回首页”图片组件绑定点击事件，跳转到返回首页界面
        returnHome.setClickedListener(new Component.ClickedListener() {
            @Override
            public void onClick(Component component) {
                present(new MainAbilitySlice(), new Intent());
            }
        });
    }
}
```

6.3.1 选择关卡难度布局代码

```
<?xml version="1.0" encoding="utf-8"?>
<DirectionalLayout
    xmlns:ohos="http://schemas.huawei.com/res/ohos"
    ohos:background_element="$media:back1"
    ohos:height="match_parent"
    ohos:width="match_parent"
    ohos:orientation="vertical">
    <Image
        ohos:height="match_content"
        ohos:width="match_content"
        ohos:image_src="$media:selectMap"
        ohos:layout_alignment="horizontal_center"
        ohos:top_margin="200px"
        />

    <DirectionalLayout
```

```
        xmlns:ohos="http://schemas.huawei.com/res/ohos"
        ohos:height="match_content"
        ohos:width="match_content"
        ohos:orientation="horizontal"
        ohos:layout_alignment="horizontal_center"
        ohos:top_margin="200px">
        <Image
            ohos:id="$+id:toLeft"
            ohos:height="180px"
            ohos:width="180px"
            ohos:image_src="$media:ToLeft"
            ohos:layout_alignment="left|vertical_center"
            ohos:scale_mode="stretch"
            ohos:right_margin="30px"
            />

        <Image
            ohos:id="$+id:imageSlider"
            ohos:height="700px"
            ohos:width="700px"
            ohos:image_src="$media:image_1"
            ohos:layout_alignment="horizontal_center"
            ohos:scale_mode="stretch"
            />

        <Image
            ohos:id="$+id:toRight"
            ohos:height="180px"
            ohos:width="180px"
            ohos:image_src="$media:toRight"
            ohos:layout_alignment="right|vertical_center"
            ohos:scale_mode="stretch"
            ohos:left_margin="30px"
            />

    </DirectionalLayout>

    <Image
        ohos:id="$+id:rank"
        ohos:height="match_content"
        ohos:width="match_content"
        ohos:image_src="$media:lv1"
        ohos:layout_alignment="horizontal_center"
        />
```

```
    <Image
        ohos:id="$+id:selectOver"
        ohos:height="match_content"
        ohos:width="match_content"
        ohos:image_src="$media:selectOver"
        ohos:layout_alignment="horizontal_center"
        ohos:top_margin="50px"
        />

    <Image
        ohos:id="$+id:returnHome"
        ohos:height="match_content"
        ohos:width="match_content"
        ohos:image_src="$media:returnHome"
        ohos:layout_alignment="horizontal_center"
        ohos:top_margin="30px"
        />

</DirectionalLayout>
```

6.3.2 选择关卡难度逻辑代码

```
public class SelectAbilitySlice extends AbilitySlice {
    private int cur = 1;    // 记录当前图片，实现图片转换

    @Override
    public void onStart(Intent intent) {
        cur = 1;
        super.onStart(intent);
        super.setUIContent(ResourceTable.Layout_ability_map);

        Image curImage = findComponentById(ResourceTable.Id_imageSlider);
        Image curRank = findComponentById(ResourceTable.Id_rank);
        curImage.setPixelMap(ResourceTable.Media_image_1);
        curRank.setPixelMap(ResourceTable.Media_lv1);

        // 绑定向右事件
        Image toRight = findComponentById(ResourceTable.Id_toRight);
        toRight.setClickedListener(new Component.ClickedListener() {
            @Override
            public void onClick(Component component) {
                if (cur == 1) {     // 第二张图
                    cur += 1;
                    curImage.setPixelMap(ResourceTable.Media_image_2);
```

```
                curRank.setPixelMap(ResourceTable.Media_lv1);
            } else if (cur == 2) {      // 第三张图
                cur += 1;
                curImage.setPixelMap(ResourceTable.Media_image_3);
                curRank.setPixelMap(ResourceTable.Media_lv2);
            } else if (cur == 3) {     // 第三张图
                cur += 1;
                curImage.setPixelMap(ResourceTable.Media_image_4);
                curRank.setPixelMap(ResourceTable.Media_lv2);
            } else if (cur == 4) {      // 第四张图
                cur += 1;
                curImage.setPixelMap(ResourceTable.Media_image_5);
                curRank.setPixelMap(ResourceTable.Media_lv3);
            }
        }
    });

    // 绑定向左事件
    Image toLeft = findComponentById(ResourceTable.Id_toLeft);
    toLeft.setClickedListener(new Component.ClickedListener() {
        @Override
        public void onClick(Component component) {
            if (cur == 3) {     // 第二张图
                cur -= 1;
                curImage.setPixelMap(ResourceTable.Media_image_2);
                curRank.setPixelMap(ResourceTable.Media_lv1);
            } else if (cur == 2) {      // 第一张图
                cur -= 1;
                curImage.setPixelMap(ResourceTable.Media_image_1);
                curRank.setPixelMap(ResourceTable.Media_lv1);
            } else if (cur == 4) {      // 第三张图
                cur -= 1;
                curImage.setPixelMap(ResourceTable.Media_image_3);
                curRank.setPixelMap(ResourceTable.Media_lv2);
            }else if (cur == 5) {      // 第四张图
                cur -= 1;
                curImage.setPixelMap(ResourceTable.Media_image_4);
                curRank.setPixelMap(ResourceTable.Media_lv3);
            }
        }
    });


    // 获取“选好了”图片组件
```

```
        Image start = (Image) findComponentById(ResourceTable.Id_selectOver);
        // 为“选好了”图片组件绑定点击事件，跳转到游戏界面
        start.setClickedListener(new Component.ClickedListener() {
            @Override
            public void onClick(Component component) {
                if (cur == 1) {
                    present(new GameAbilitySlice(), new Intent());
                } else if (cur == 2) {
                    present(new GameAbilitySlice2(), new Intent());
                } else if (cur == 3) {
                    present(new GameAbilitySlice3(), new Intent());
                } else if (cur == 4) {
                    present(new GameAbilitySlice4(), new Intent());
                } else if (cur == 5) {
                    present(new GameAbilitySlice5(), new Intent());
                }
            }
        });

        // 获取“返回首页”图片组件
        Image returnHome = (Image)
findComponentById(ResourceTable.Id_returnHome);
        // 为“返回首页”图片组件绑定点击事件，跳转到返回首页界面
        returnHome.setClickedListener(new Component.ClickedListener() {
            @Override
            public void onClick(Component component) {
                present(new MainAbilitySlice(), new Intent());
            }
        });
    }
}
```

6.4.1 拼图游戏界面布局代码（以 3*3 为例）

```
<?xml version="1.0" encoding="utf-8"?>
<DirectionalLayout
    xmlns:ohos="http://schemas.huawei.com/res/ohos"
    ohos:height="match_parent"
    ohos:width="match_parent"
    ohos:orientation="vertical"
    ohos:background_element="$media:back1"
    ohos:padding="30px">

    <Image
        ohos:height="match_content"
```

```
        ohos:width="match_content"
        ohos:image_src="$media:huarongdao"
        ohos:layout_alignment="horizontal_center"
        ohos:top_margin="180px"
        />

    <DirectionalLayout
        xmlns:ohos="http://schemas.huawei.com/res/ohos"
        ohos:height="match_content"
        ohos:width="match_parent"
        ohos:orientation="horizontal"
        ohos:alignment="horizontal_center"
        ohos:top_margin="150px">
        <Image
            ohos:id="$+id:pic_ib11"
            ohos:height="300px"
            ohos:width="300px"
            ohos:image_src="$media:image_1_11"
            ohos:top_margin="40px"
            ohos:scale_mode="stretch"
            ohos:margin="5px"
            />
        <Image
            ohos:id="$+id:pic_ib12"
            ohos:height="300px"
            ohos:width="300px"
            ohos:image_src="$media:image_1_12"
            ohos:top_margin="40px"
            ohos:scale_mode="stretch"
            ohos:margin="5px"
            />
        <Image
            ohos:id="$+id:pic_ib13"
            ohos:height="300px"
            ohos:width="300px"
            ohos:image_src="$media:image_1_13"
            ohos:top_margin="40px"
            ohos:scale_mode="stretch"
            ohos:margin="5px"
            />
    </DirectionalLayout>

    <DirectionalLayout
        xmlns:ohos="http://schemas.huawei.com/res/ohos"
```

```
        ohos:height="match_content"
        ohos:width="match_parent"
        ohos:orientation="horizontal"
        ohos:alignment="horizontal_center"
        >
        <Image
            ohos:id="$+id:pic_ib21"
            ohos:height="300px"
            ohos:width="300px"
            ohos:image_src="$media:image_1_21"
            ohos:top_margin="40px"
            ohos:scale_mode="stretch"
            ohos:margin="5px"
            />
        <Image
            ohos:id="$+id:pic_ib22"
            ohos:height="300px"
            ohos:width="300px"
            ohos:image_src="$media:image_1_22"
            ohos:top_margin="40px"
            ohos:scale_mode="stretch"
            ohos:margin="5px"
            />
        <Image
            ohos:id="$+id:pic_ib23"
            ohos:height="300px"
            ohos:width="300px"
            ohos:image_src="$media:image_1_23"
            ohos:top_margin="40px"
            ohos:scale_mode="stretch"
            ohos:margin="5px"
            />
    </DirectionalLayout>

    <DirectionalLayout
        xmlns:ohos="http://schemas.huawei.com/res/ohos"
        ohos:height="match_content"
        ohos:width="match_parent"
        ohos:orientation="horizontal"
        ohos:alignment="horizontal_center">
        <Image
            ohos:id="$+id:pic_ib31"
            ohos:height="300px"
            ohos:width="300px"
```

```
            ohos:image_src="$media:image_1_31"
            ohos:top_margin="40px"
            ohos:scale_mode="stretch"
            ohos:margin="5px"
            />
        <Image
            ohos:id="$+id:pic_ib32"
            ohos:height="300px"
            ohos:width="300px"
            ohos:image_src="$media:image_1_32"
            ohos:top_margin="40px"
            ohos:scale_mode="stretch"
            ohos:margin="5px"
            />
        <Image
            ohos:id="$+id:pic_ib33"
            ohos:height="300px"
            ohos:width="300px"
            ohos:image_src="$media:image_1_33"
            ohos:top_margin="40px"
            ohos:scale_mode="stretch"
            ohos:margin="5px"
            ohos:visibility="invisible"
            />
    </DirectionalLayout>

    <DirectionalLayout
        xmlns:ohos="http://schemas.huawei.com/res/ohos"
        ohos:height="match_content"
        ohos:width="match_parent"
        ohos:alignment="horizontal_center"
        ohos:orientation="horizontal"
        ohos:top_margin="-20px">
        <Image
            ohos:id="$+id:reset"
            ohos:height="400px"
            ohos:width="550px"
            ohos:image_src="$media:reset"
            />
        <Image
            ohos:id="$+id:finish"
            ohos:height="400px"
            ohos:width="550px"
            ohos:image_src="$media:finish"
```

```
        />
</DirectionalLayout>

<DirectionalLayout
    xmlns:ohos="http://schemas.huawei.com/res/ohos"
    ohos:height="match_content"
    ohos:width="match_parent"
    ohos:orientation="horizontal"
    ohos:top_margin="-150px">
    <Image
        ohos:height="400px"
        ohos:width="550px"
        ohos:image_src="$media:time"
        ohos:start_margin="150px"
        />
    <TickTimer
        ohos:id="$+id:time_string"
        ohos:height="400px"
        ohos:width="match_content"
        ohos:text="00:00"
        ohos:text_size="100px"
        ohos:start_margin="50px"
        />
</DirectionalLayout>

<DirectionalLayout
    xmlns:ohos="http://schemas.huawei.com/res/ohos"
    ohos:height="match_content"
    ohos:width="match_parent"
    ohos:orientation="horizontal"
    ohos:top_margin="-200px">
    <Image
        ohos:height="400px"
        ohos:width="550px"
        ohos:image_src="$media:step"
        ohos:start_margin="150px"
        />
    <Text
        ohos:id="$+id:step_string"
        ohos:height="400px"
        ohos:width="match_content"
        ohos:text="0"
        ohos:text_size="100px"
        ohos:start_margin="110px"
```

```
            />
    </DirectionalLayout>

</DirectionalLayout>
```

6.4.2 拼图游戏逻辑布局代码（以 3*3 为例）

```
public class GameAbilitySlice extends AbilitySlice {
    Image ib11, ib12, ib13, ib21, ib22, ib23, ib31, ib32, ib33;     // 定义图片碎片
    Image reset;       // 定义重试按钮
    Image finish;       // 定义结束按钮

    private int imageX = 3;     // 每行图片数量
    private int imageY = 3;     // 定义每列数量
    private int imageCount = imageX * imageY;       // 定义图片总数
    private int blackSwap = imageCount - 1;         // 定义缺少图片的位置，默认最后一个
    private int blackImgid = ResourceTable.Id_pic_ib33;        // 定义缺少图片的 id，默认是最后一个
    private int steps;      // 定义步骤计数
    private TickTimer tickTimer;        // 定义计时器
    private int[] image = {     // 存放图片碎片的数组
            ResourceTable.Media_image_1_11, ResourceTable.Media_image_1_12, ResourceTable.Media_image_1_13,
            ResourceTable.Media_image_1_21, ResourceTable.Media_image_1_22, ResourceTable.Media_image_1_23,
            ResourceTable.Media_image_1_31, ResourceTable.Media_image_1_32, ResourceTable.Media_image_1_33
    };

    private int[] imageIndex = new int[image.length];       // 定义图片下标数组，随机化排列顺序
    private int[] tmpImageIndex = new int[image.length];        // 定义另一个下标数组保存信息
    private int giveUpFlag;     // 定义放弃按钮

    @Override
    public void onStart(Intent intent) {
        super.onStart(intent);
        super.setUIContent(ResourceTable.Layout_ability_game);

        initComponent();
        disruptRandom();
    }
```

```
    @Override
    public void onActive() {
        super.onActive();
    }

    @Override
    public void onForeground(Intent intent) {
        super.onForeground(intent);
    }

    private void disruptRandom() {      //随机打乱数组中的元素，以不规则的形式进行图片显示
        for (int i = 0; i < imageIndex.length; i++) {
            imageIndex[i] = i;
        }
        //规定 20 次，随机选择两个角标对应的值进行交换
        int rand1, rand2;
        for (int j = 0; j < 20; j++) {
            //随机生成第一个角标，生成 0—8 之间的随机数
            rand1 = (int) (Math.random() * (imageIndex.length - 1));
//Math.random()生成的是 0—1 之间的随机数，再乘以最大值减去最小值（即 8-0），最后整体加上最小值 0
            //第二次随机生成的角标不能和第一次相同，如果相同就不方便交换
            do {
                rand2 = (int) (Math.random() * (imageIndex.length - 1));
            } while (rand1 == rand2);
            //交换数组两个角标上对应的值
            swap(rand1, rand2);
        }
        // 将随即后的数组存储下来
        System.arraycopy(imageIndex, 0, tmpImageIndex, 0, imageIndex.length);

        //随机排列到指定的控件上
        ib11.setPixelMap(image[imageIndex[0]]);
        ib12.setPixelMap(image[imageIndex[1]]);
        ib13.setPixelMap(image[imageIndex[2]]);
        ib21.setPixelMap(image[imageIndex[3]]);
        ib22.setPixelMap(image[imageIndex[4]]);
        ib23.setPixelMap(image[imageIndex[5]]);
        ib31.setPixelMap(image[imageIndex[6]]);
        ib32.setPixelMap(image[imageIndex[7]]);
        ib33.setPixelMap(image[imageIndex[8]]);
```

```
    }

    private void swap(int rand1, int rand2) {        //交换数组指定角标的数据
        int temp = imageIndex[rand1];
        imageIndex[rand1] = imageIndex[rand2];
        imageIndex[rand2] = temp;
    }

    private void initComponent() {       // 初始化控件
        ib11 = findComponentById(ResourceTable.Id_pic_ib11);
        ib12 = findComponentById(ResourceTable.Id_pic_ib12);
        ib13 = findComponentById(ResourceTable.Id_pic_ib13);
        ib21 = findComponentById(ResourceTable.Id_pic_ib21);
        ib22 = findComponentById(ResourceTable.Id_pic_ib22);
        ib23 = findComponentById(ResourceTable.Id_pic_ib23);
        ib31 = findComponentById(ResourceTable.Id_pic_ib31);
        ib32 = findComponentById(ResourceTable.Id_pic_ib32);
        ib33 = findComponentById(ResourceTable.Id_pic_ib33);
        finish = findComponentById(ResourceTable.Id_finish);
        reset = findComponentById(ResourceTable.Id_reset);

        ib11.setVisibility(Component.VISIBLE);
        ib12.setVisibility(Component.VISIBLE);
        ib13.setVisibility(Component.VISIBLE);
        ib21.setVisibility(Component.VISIBLE);
        ib22.setVisibility(Component.VISIBLE);
        ib23.setVisibility(Component.VISIBLE);
        ib31.setVisibility(Component.VISIBLE);
        ib32.setVisibility(Component.VISIBLE);
        ib33.setVisibility(Component.INVISIBLE);

        // 重置放弃按钮
        giveUpFlag = 0;
        finish.setPixelMap(ResourceTable.Media_finish);
        // 重置步骤和计时器
        steps = 0;
        tickTimer = findComponentById(ResourceTable.Id_time_string);
        tickTimer.setBaseTime(System.currentTimeMillis());
        tickTimer.start();

        // 为各个控件绑定点击事件
        ib11.setClickedListener(new Component.ClickedListener() {
            @Override
            public void onClick(Component component) {
```

```
            move(ResourceTable.Id_pic_ib11, 0);
        }
    });
    ib12.setClickedListener(new Component.ClickedListener() {
        @Override
        public void onClick(Component component) {
            move(ResourceTable.Id_pic_ib12, 1);
        }
    });
    ib13.setClickedListener(new Component.ClickedListener() {
        @Override
        public void onClick(Component component) {
            move(ResourceTable.Id_pic_ib13, 2);
        }
    });
    ib21.setClickedListener(new Component.ClickedListener() {
        @Override
        public void onClick(Component component) {
            move(ResourceTable.Id_pic_ib21, 3);
        }
    });
    ib22.setClickedListener(new Component.ClickedListener() {
        @Override
        public void onClick(Component component) {
            move(ResourceTable.Id_pic_ib22, 4);
        }
    });
    ib23.setClickedListener(new Component.ClickedListener() {
        @Override
        public void onClick(Component component) {
            move(ResourceTable.Id_pic_ib23, 5);
        }
    });
    ib31.setClickedListener(new Component.ClickedListener() {
        @Override
        public void onClick(Component component) {
            move(ResourceTable.Id_pic_ib31, 6);
        }
    });
    ib32.setClickedListener(new Component.ClickedListener() {
        @Override
        public void onClick(Component component) {
            move(ResourceTable.Id_pic_ib32, 7);
        }
```

```
        });
        ib33.setClickedListener(new Component.ClickedListener() {
            @Override
            public void onClick(Component component) {
                move(ResourceTable.Id_pic_ib33, 8);
            }
        });

        // 绑定放弃按钮
        finish.setClickedListener(new Component.ClickedListener() {
            @Override
            public void onClick(Component component) {
                if (giveUpFlag == 0) {
                    giveUp();
                    finish.setPixelMap(ResourceTable.Media_leave);
                    giveUpFlag = 1;
                } else {
                    present(new SelectAbilitySlice(), new Intent());
                }

            }
        });

        // 绑定重试按钮
        reset.setClickedListener(new Component.ClickedListener() {
            @Override
            public void onClick(Component component) {
                restart();
            }
        });

    }

    public void move(int imageId, int site) {
        int siteX = site / imageX;        //判断选中的图片在第几行，取整来判断
        int siteY = site % imageY;        //判断选中的图片在第几列，取整来判断
        // 获取空白区域的坐标
        int blackX = blackSwap / imageX;
        int blackY = blackSwap % imageY;

        // 判断移动的条件
        int x = Math.abs(siteX - blackX);
        int y = Math.abs(siteY - blackY);
        if ((x == 0 && y == 1) || (x == 1 && y == 0)) {
```

```
            Image currentImage = findComponentById(imageId);      // 通过 id 查找到这个可以移动图片控件
            currentImage.setVisibility(Component.INVISIBLE);      // 该可移动按钮不在显示图片
            Image blackImage = findComponentById(blackImgid);     // 查找空白区域的按钮
            blackImage.setPixelMap(image[imageIndex[site]]);      // 将空白按钮设置为显示图片
            blackImage.setVisibility(Component.VISIBLE);        // 移动之前是不可见的，移动之后将控件设置为可见
            // 上面的交换并没有存在数组之中，要调用 swap 函数，将改变角标的过程记录在存储图片位置的数组当中
            swap(site, blackSwap);
            steps += 1;
            //新的空白区域位置更新
            blackSwap = site;
            blackImgid = imageId;
        }
        Text step_string = findComponentById(ResourceTable.Id_step_string);
        step_string.setText(Integer.toString(steps));
        //判断本次移动后是否完成拼图游戏
        judgeGameOver();
    }

    private void judgeGameOver() {      // 判断游戏是否结束
        boolean loop = true;      // 定义标志位
        //对存放图片角标的数组 imageIndex 进行判断
        for (int i = 0; i < imageIndex.length; i++) {
            if (imageIndex[i] != i) {
                loop = false;
                break;
            }
        }
        if (loop) {
            //拼图成功
            ib11.setClickable(false);
            ib12.setClickable(false);
            ib13.setClickable(false);
            ib21.setClickable(false);
            ib22.setClickable(false);
            ib23.setClickable(false);
            ib31.setClickable(false);
            ib32.setClickable(false);
            ib33.setClickable(false);
```

```
            // 显示之前隐藏的拼图
            ib33.setPixelMap(image[8]);
            ib33.setVisibility(Component.VISIBLE);
            // 弹出提示框
            CommonDialog cd = new CommonDialog(this);
            cd.setSize(1000, 450);
            cd.setCornerRadius(60);
            cd.setTitleText("    恭喜你成功过关");
            cd.setContentText("    累计用时: " + tickTimer.getText() + "\n
累计步骤: " + steps);
            cd.setContentText("    累计用时: " + tickTimer.getText() + "\n
累计步骤: " + steps);
            cd.setTitleIcon(ResourceTable.Media_pass, 1);
            // 返回选图
            cd.setButton(0, "返回选图", new IDialog.ClickedListener() {
                @Override
                public void onClick(IDialog iDialog, int i) {
                    cd.destroy();
                    present(new SelectAbilitySlice(), new Intent());
                }
            });
            // 再试一次
            cd.setButton(1, "再试一次", new IDialog.ClickedListener() {
                @Override
                public void onClick(IDialog iDialog, int i) {
                    cd.destroy();
                    again();
                }
            });
//            cd.setAutoClosable(true);
            cd.show();
            // 重置计时器和步骤
            resetStepTime();
        }
    }

    public void giveUp() {      // 定义放弃图片组件
        // 还原拼图
        ib11.setPixelMap(image[0]);
        ib12.setPixelMap(image[1]);
        ib13.setPixelMap(image[2]);
        ib21.setPixelMap(image[3]);
        ib22.setPixelMap(image[4]);
        ib23.setPixelMap(image[5]);
```

```
        ib31.setPixelMap(image[6]);
        ib32.setPixelMap(image[7]);
        ib33.setPixelMap(image[8]);
        // 设置为图片不可点击
        ib11.setClickable(false);
        ib12.setClickable(false);
        ib13.setClickable(false);
        ib21.setClickable(false);
        ib22.setClickable(false);
        ib23.setClickable(false);
        ib31.setClickable(false);
        ib32.setClickable(false);
        ib33.setClickable(false);
        // 显示之前隐藏的拼图
        ib33.setVisibility(Component.VISIBLE);
        //最后一次选中的空白区域显示出来
        Image currentImage = findComponentById(blackImgid);
        currentImage.setVisibility(Component.VISIBLE);

        // 重置计时器和步骤
        resetStepTime();

    }

    public void restart() {     // 游戏重试控件定义
        // 拼图游戏重新开始，允许玩家重新触碰按钮
        ib11.setClickable(true);
        ib12.setClickable(true);
        ib13.setClickable(true);
        ib21.setClickable(true);
        ib22.setClickable(true);
        ib23.setClickable(true);
        ib31.setClickable(true);
        ib32.setClickable(true);
        ib33.setClickable(true);

        //还原被点击的图片按钮变成初始化的模样
        //最后一次选中的空白区域显示出来
        Image currentImage = findComponentById(blackImgid);
        currentImage.setVisibility(Component.VISIBLE);

        //定义一个新的图片按钮，设置为第九个，让其隐藏
        Image blackBtn = findComponentById(ResourceTable.Id_pic_ib33);
        blackBtn.setVisibility(Component.INVISIBLE);
```

```
        //初始化空白区域的按钮 id
        blackImgid = ResourceTable.Id_pic_ib33;
        blackSwap = imageCount - 1;

        //将拼图重新还原
        System.arraycopy(tmpImageIndex, 0, imageIndex, 0, imageIndex.length);
        //随机排列到指定的控件上
        ib11.setPixelMap(image[imageIndex[0]]);
        ib12.setPixelMap(image[imageIndex[1]]);
        ib13.setPixelMap(image[imageIndex[2]]);
        ib21.setPixelMap(image[imageIndex[3]]);
        ib22.setPixelMap(image[imageIndex[4]]);
        ib23.setPixelMap(image[imageIndex[5]]);
        ib31.setPixelMap(image[imageIndex[6]]);
        ib32.setPixelMap(image[imageIndex[7]]);
        ib33.setPixelMap(image[imageIndex[8]]);

        // 重置计时器和步骤
        resetStepTime();
        tickTimer.setBaseTime(System.currentTimeMillis());
        tickTimer.start();

        // 重置放弃按钮
        giveUpFlag = 0;
        finish.setPixelMap(ResourceTable.Media_finish);
    }

    public void again() {      // 游戏再来一次定义
        //将状态还原
        //拼图游戏重新开始，允许玩家重新触碰按钮
        ib11.setClickable(true);
        ib12.setClickable(true);
        ib13.setClickable(true);
        ib21.setClickable(true);
        ib22.setClickable(true);
        ib23.setClickable(true);
        ib31.setClickable(true);
        ib32.setClickable(true);
        ib33.setClickable(true);

        //最后一次选中的空白区域显示出来
        Image currentImage = findComponentById(blackImgid);
        currentImage.setVisibility(Component.VISIBLE);
```

```
        //定义一个新的图片按钮，设置为第九个，让其隐藏
        Image blackBtn = findComponentById(ResourceTable.Id_pic_ib33);
        blackBtn.setVisibility(Component.INVISIBLE);

        //初始化空白区域的按钮 id
        blackImgid = ResourceTable.Id_pic_ib33;
        blackSwap = imageCount - 1;

        //将拼图重新打乱
        disruptRandom();

        // 重置计时器和步骤
        resetStepTime();
        tickTimer.setBaseTime(System.currentTimeMillis());
        tickTimer.start();

        // 重置放弃按钮
        giveUpFlag = 0;
        finish.setPixelMap(ResourceTable.Media_finish);

    }

    public void resetStepTime() {
        steps = 0;
        Text step_string = findComponentById(ResourceTable.Id_step_string);
        step_string.setText(Integer.toString(steps));

        tickTimer.stop();
        tickTimer.setText("00:00");
    }
}
```

6.5 拼图分割碎片 python 代码

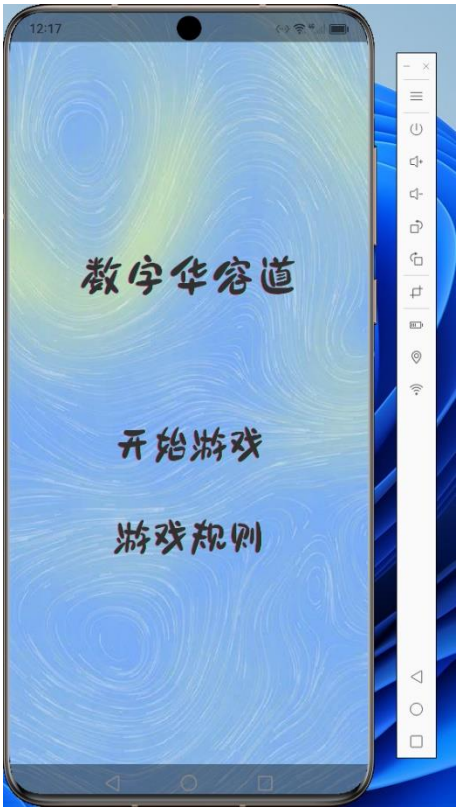
```
import os

from PIL import Image, ImageDraw, ImageFont
def crop_and_resize_to_square(input_image_path, output_image_path, size):
    original_image = Image.open(input_image_path)
    width, height = original_image.size

    if width > height:
        # 原始图像宽度大于高度，裁剪宽度
        left = (width - height) // 2
```

```
        right = left + height
        top = 0
        bottom = height
    else:
        # 原始图像高度大于宽度，裁剪高度
        top = (height - width) // 2
        bottom = top + width
        left = 0
        right = width

    cropped_image = original_image.crop((left, top, right, bottom))
    squared_image = cropped_image.resize((size, size))
    squared_image.save(output_image_path)


def split_image_into_grid(input_image_path, output_folder, rows, cols):
    original_image = Image.open(input_image_path)
    width, height = original_image.size
    cell_width = width // cols
    cell_height = height // rows

    font = ImageFont.load_default()
    font_size = 36  # 设置字体大小

    for row in range(rows):
        for col in range(cols):
            left = col * cell_width
            top = row * cell_height
            right = left + cell_width
            bottom = top + cell_height

            cell_image = original_image.crop((left, top, right, bottom))
            cell_name = f"image_5_{row + 1}{col + 1}"

            # 在小区域图片的中间绘制数字
            draw = ImageDraw.Draw(cell_image)
            font = ImageFont.truetype("arial.ttf", font_size)  # 使用指定字体
和字体大小
            text = str(row*5 + col + 1)
            text_width, text_height = draw.textsize(text, font)
            x = (cell_width - text_width) // 2
            y = (cell_height - text_height) // 2
            draw.text((x, y), text, (255, 255, 255), font=font)
```

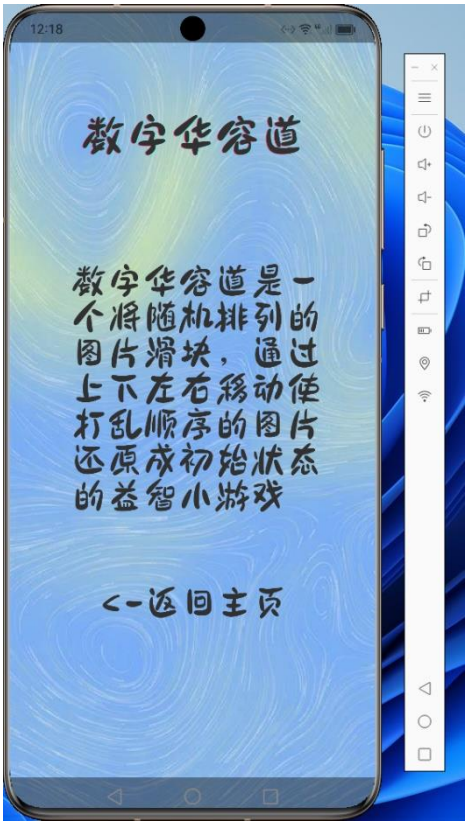
```
            cell_image.save(os.path.join(output_folder, f"{cell_name}.png"))


if __name__ == "__main__":
    input_image_path = "images5.jpg"  # 输入图像文件的路径
    output_image_path = "image_5.jpg"  # 重置为正方形后的图像保存路径
    output_folder = "image_cells"  # 保存分割小区域的文件夹路径
    square_size = 300  # 正方形的大小
    rows, cols = 5, 5  # 行数和列数

    # 裁剪并重置图像为正方形
    crop_and_resize_to_square(input_image_path, output_image_path,
square_size)

    # 创建保存小区域的文件夹
    os.makedirs(output_folder, exist_ok=True)

    # 分割图像为小区域并在中间绘制数字
    split_image_into_grid(output_image_path, output_folder, rows, cols)
```

7. 运行截图

7.1 游戏首页



7.2 规则介绍



7.3 选择关卡图片（仅展示部分）

7.3 拼图游戏界面（仅展示部分）

7.4 通关界面

四、实验结果及分析和（或）源程序调试过程（界面截图和文字）、实验总结与体会

1.实验结果分析

本次实验是使用 DevEco Studio 的 harmony 开发系统完成小游戏数字华容道的设计实验。实验已完成从多张图片中，选择一张，分割为 4 块以上的小图片，打乱后分布，可以拼成原图，数字正确排列的基本任务。同时实现了（1）从多张图片中选择并标记数字；（2）通过开始按钮开始游戏，打乱拼图显示在界面上；（3）通过点击图片进行移动；（4）游戏过程自动判断是否完成；（5）设置了重试和放弃按钮提前结束游戏；（6）设计了游戏主界面、规则介绍界面、关卡难度选择界面以及拼图游戏界面；（7）额外增加了游戏计时、步骤统计、关卡选择、游戏指南的功能。完成预期的实验目标。

2.实验总结与体会

学习了使用 java 语言在 DevEco Studio 的鸿蒙软件上的开发操作，虽然有很多和安卓开发类似的地方，例如与安卓开发类似的文件结构，媒体资源的存储和读取方式，布局文件中元素属性的接口有很多相同或者相近的地方，java 逻辑设计中对界面元素的获取和设置方式也是类似的，但是也有不同的地方，如 DirectionalLayout 的线性布局方式,ResourceTable 这种为每个媒体文件和页面元素分配 id 的资源管理列表，都是鸿蒙开发特有的方式。

具体来说，首先是在 configs.json 的全局配置文件中，可以设置全局样式，例如 androidhwext:style/Theme.Emui.Light.NoTitleBar 可以设置为白色背景的无标题栏的样式，配合逻辑代码中 getWindow().addFlags()接口的调用可以设置全屏的样式，让页面更精致。

在页面布局文件的设计方面，通过线性布局中 vertical 和 horizontal 的排列方式的组合，可以实现拼图按照对应顺序排列（例如 3*3）的效果。同时根据 margin 和 padding 的设置，能让页面布局更加美观。其中遇到的问题有对 match_parent 和 match_content 的理解问题，导致页面一直调整不出理想的效果，通过反复尝试最终解决问题。

在逻辑代码的实现方面，在 onstart 方法中添加页面打开时的业务逻辑，首先对涉及到的变量进行初始化操作，若需要跳转页面，需要使用到 present 接口来实现。若要对页面中元素绑定点击事件，则需要使用接口 findComponentById(ResourceTable.Id_xxx)来定位到页面中的控件资源，然后通过 setClickedListener()来绑定点击事件。我的实现代码中主要涉及到图片控件的图片资源的更换，通过 setPixelMap()接口来实现。在整个拼图游戏的逻辑设计中，用到多次使用随机数来随机交换拼图碎片，从而达到打乱的效果。在图片移动过程中，通过 setVisibility()来实现对应图片资源是否可视。同时每次移动后判断是否完成拼图还原，如果还原则用 CommonDialog()来弹出提示框。我还学到了使用 text 的子类 tickTimer 来进行计时，对应的 setBaseTime()函数来定义事件基准，实现计时的功能。

实验报告填写说明：

1、第一、二部分由老师提供；

2、第三部分填写源程序或者算法，清单文件，资源文件等。源程序要符合程序编写风格

（缩进、注释等）；

3、第四部分主要填写程序调试运行过程、结果（截图）、解决问题的方法、总结和体会等；

4、报告规范：包含报告页眉、报告的排版、内容是否填写，命名是否规范等。