注：文档中所有的图片及代码截图皆为示意图，后期官方更新，具体以 HarmonyOS 官网发布内容为准。

一、视频播放

1. 介绍

本篇 Codelab 将实现的内容

HarmonyOS 是面向全场景多终端的分布式操作系统，使得应用程序的开发打破了智能终端互通的性能和数据壁垒，业务逻辑原子化开发，适配多端。通过一个简单应用开发，体验 HarmonyOS 的视频播放能力

您将建立什么

在这个 Codelab 中，你将创建 Demo Project，并将 Demo 编译成 Hap，此示例应用程序展示了如何播放视频。

您将会学到什么

- 如何创建一个 HarmonyOS Demo Project
- 如何构建一个 Hap 并且将其部署到智慧屏真机
- 通过此示例应用体验如何播放本地或者在线视频

2. 您需要什么

硬件要求

- 操作系统：Windows10 64 位
- 内存：8G 及以上。
- 硬盘：100G 及以上。
- 分辨率：1280*800 及以上

软件要求

- DevEco Studio：需手动下载安装，详细步骤请参考《DevEco Studio 使用指南》2.1.2
- JDK：DevEco Studio 自动安装。
- Node.js：请手动下载安装，详细步骤请参考《DevEco Studio 使用指南》2.1.3 下载和安装 Node.js。
- HarmonyOS SDK：待 DevEco Studio 安装完成后，利用 DevEco Studio 来加载 HarmonyOS SDK。详细步骤请参考《DevEco Studio 使用指南》2.1.6 加载 HarmonyOS SDK。
- Maven 库依赖包：如需手动拷贝和配置，详细步骤请参考《DevEco Studio 使用指南》2.3 离线方式配置 Maven 库。

需要的知识点

- Java 基础开发能力。

3. 能力接入准备

实现 HarmonyOS 应用开发，需要完成以下准备工作：

- 环境准备。
- 环境搭建。
- 创建项目
- 申请调试证书
- 应用开发

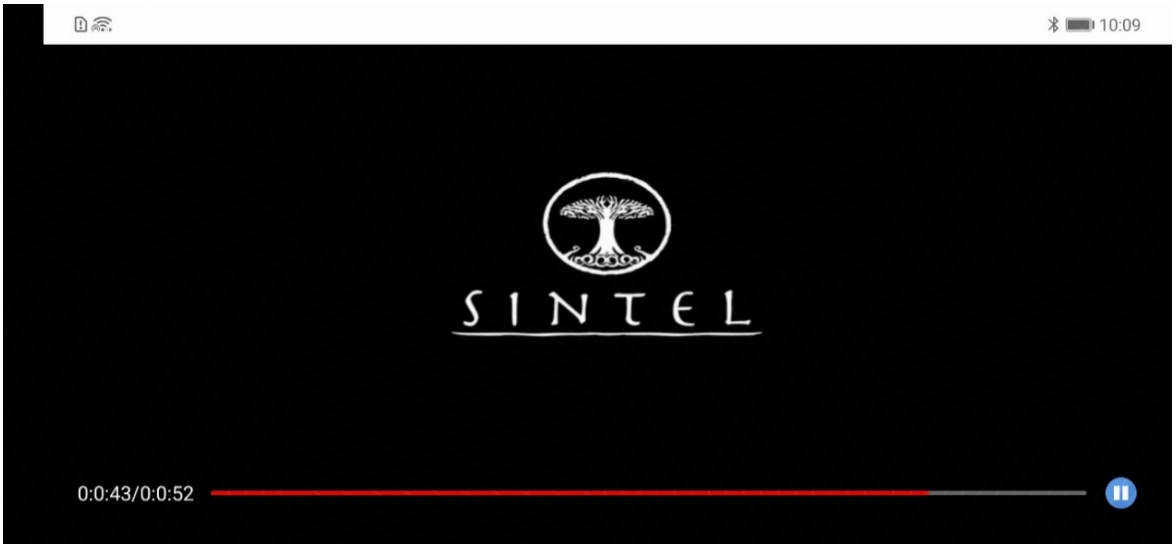
具体操作，请按照《DevEco Studio 使用指南》中详细说明来完成。

提示：需要通过注册成开发者才能完成集成准备中的操作。

4. 代码片段

1. 布局：

- 创建播放视频的 Ability

- public class VedioPlayAbilitySlice extends AbilitySlice implements SurfaceOps.Callback
- 布局截图
- 

- 布局代码：

```
//设置页面背景透明
WindowManager windowManager = WindowManager.getInstance();
Window window = windowManager.getTopWindow().get();
window.setTransparent(true);

//页面父布局
DependentLayout myLayout = new DependentLayout(this);
DependentLayout.LayoutConfig params = new
DependentLayout.LayoutConfig(MATCH_PARENT,
MATCH_PARENT);
myLayout.setLayoutConfig(params);
//显示视频的自定义 videoView
DependentLayout.LayoutConfig lpVideo = new
DependentLayout.LayoutConfig(MATCH_PARENT,
MATCH_PARENT);
videoView = new VideoView(this, this);
videoView.setHandler(handler);
myLayout.addComponent(videoView, lpVideo);

DependentLayout rlParent = new DependentLayout(this);
DependentLayout.LayoutConfig lpParent = new
DependentLayout.LayoutConfig(MATCH_PARENT,
WRAP_CONTENT);
lpParent.addRule(DependentLayout.ALIGN_PARENT_BOTTOM);
lpParent.leftMargin = ConvertUtils.dp2Px(40);
```

```
lpParent.rightMargin = ConvertUtils.dp2Px(40);
lpParent.bottomMargin = ConvertUtils.dp2Px(40);
myLayout.addComponent(rlParent, lpParent);
//显示播放暂停按钮
playBtn = new Image(this);
DependentLayout.LayoutConfig lpPlayBtn = new
DependentLayout.LayoutConfig(ConvertUtils.dp2Px(40),
ConvertUtils.dp2Px(40));
lpPlayBtn.addRule(DependentLayout.ALIGN_PARENT_RIGHT);
playBtn.setLayoutConfig(lpPlayBtn);
playBtn.setId(1112);
playBtn.setPixelMap(ResourceTable.Media_pause);
playBtn.setScaleType(Image.ScaleType.SCALE_TO_FULL);
playBtn.invalidate();
playBtn.setClickedListener(new Component.ClickedListener() {
@Override
public void onClick(Component component) {
    if (videoView.getPlayState() == VideoView.STATE_PLAYING) {
        videoView.pause();
        playBtn.setPixelMap(ResourceTable.Media_play);
    } else {
        videoView.start();
        playBtn.setPixelMap(ResourceTable.Media_pause);
    }
}
});
rlParent.addComponent(playBtn);
//显示当前视频播放时间
timePlay = new Text(this);
DependentLayout.LayoutConfig lpTimePlay = new
DependentLayout.LayoutConfig(WRAP_CONTENT,
WRAP_CONTENT);
lpTimePlay.addRule(DependentLayout.CENTER_VERTICAL);
timePlay.setLayoutConfig(lpTimePlay);
timePlay.setId(1111);
timePlay.setText("00:00/00:00");
timePlay.setTextSize(40);
timePlay.setTextColor(Color.WHITE);
rlParent.addComponent(timePlay);
// 显示播放进度条
roundProgressBar = new ProgressBar(this);
roundProgressBar.setProgressWidth(ConvertUtils.dp2Px(5));
roundProgressBar.setProgressColor(Color.RED);
roundProgressBar.setMax(100);
```

```
roundProgressBar.setProgress(0);
DependentLayout.LayoutConfig lpProgressBar = new DependentLayout.LayoutConfig(MATCH_PARENT, ConvertUtils.dp2Px(40));
lpProgressBar.addRule(DependentLayout.RIGHT_OF, timePlay.getId());
lpProgressBar.leftMargin = ConvertUtils.dp2Px(20);
lpProgressBar.rightMargin = ConvertUtils.dp2Px(60);
rlParent.addComponent(roundProgressBar, lpProgressBar);
```

2. 自定义 VideoView

- 继承父类 SurfaceProvider

```
public class VideoView extends SurfaceProvider implements Player.IPlayerCallback
```

- 初始化：

```
public VideoView(Context context, SurfaceOps.Callback callback) {
super(context);
player = new Player(getContext());
player.setPlayerCallback(this);
Optional<SurfaceOps> surfaceHolderOptional = getSurfaceOps();
SurfaceOps surfaceHolder = surfaceHolderOptional.get();
surfaceHolder.addCallback(callback);
setZOrderOnTop(false);
state = STATE_INIT;
}
```

- 播放视频

  如果播放的是在线视频，需要申请网络权限：

```
"reqPermissions": [
  {
    "name": "harmonyos.permission.INTERNET"
  }
]


public void playAssets(String fileName, boolean isLooping, SurfaceOps holder) {
      try {
//播放本地视频：
//player.setSource(getContext().getResourceManager().getRawFileEntry(fileName).openRawFileDescriptor());
```

```
//播放在线视频:
            player.setSource(new
Source("https://media.w3.org/2010/05/sintel/trailer.mp4"));
            player.setVideoSurface(holder.getSurface());
            player.enableSingleLooping(isLooping);
            player.enableScreenOn(true);
            player.prepare();
            initPlayViewSize();
            player.play();
            if (state != STATE_INIT) {
                player.rewindTo(currentTime * 1000);
            }
            state = STATE_PLAYING;

handler.sendEvent(InnerEvent.get(MESSAGE_UPDATE_PLAY_TIME)
);
        } catch (Exception e) {
            e.printStackTrace();
            Log.hiLog(e.toString());
        }
}

private void initPlayViewSize() {
        int videoWidth = player.getVideoWidth();
        int videoHeight = player.getVideoHeight();
        Log.hiLog("videoWidth:" + videoWidth + ", videoHeight:" +
videoHeight);
        if (videoWidth < videoHeight) {
            double scale = screenHeight * 1.f / videoHeight;
            double currHeight = videoHeight * scale;
            double currWidth = videoWidth * scale;
            setHeight(((int) currHeight));
            setWidth(((int) currWidth));
        } else {
            double scale = screenWidth * 1.f / videoWidth;
            double currHeight = videoHeight * scale;
            double currWidth = videoWidth * scale;
            setHeight(((int) currHeight));
            setWidth(((int) currWidth));
        }
}
```

- 暂停：

```
public void pause() {
if (state == STATE_PLAYING) {
```

```
            player.pause();
            state = STATE_PAUSE;
        }
    }
```

- 在 AbilitySlice 启动自动播放

```
    @Override
    public void surfaceCreated(SurfaceOps surfaceOps) {
    Log.hiLog("surfaceCreated");
    videoView.playAssets("resources/rawfile/VID_20200613_204240.mp4"
    , true, surfaceOps);
    }

    @Override
    public void surfaceDestroyed(SurfaceOps surfaceOps) {
    Log.hiLog("surfaceDestroyed");
    videoView.stop();
    }
```

3. 响应遥控器点击

```
@Override
public boolean onKeyUp(int keyCode, KeyEvent keyEvent) {
    switch (keyCode) {
        case KeyEvent.KEY_DPAD_CENTER:
        case KeyEvent.KEY_ENTER:
            playBtn.performClick();
            return true;
        default:
                break;
    }
    return false;
}
```

4. 编译运行该应用

- 通过 hdc 连接大屏设备

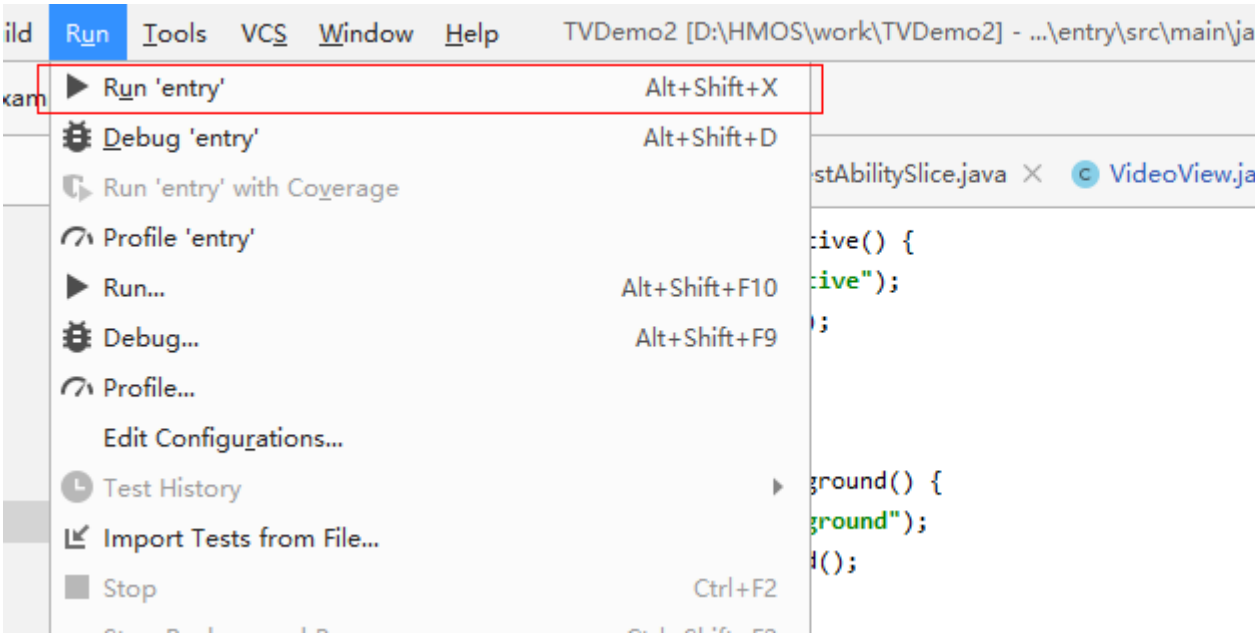
  先查看智慧屏 IP：

  ```
  大屏设置->"网络与连接"->"网络"->"有线网络"
  ```

  在 cmd 或者 IDE 的 Terminal 输入命令：

  ```
  hdc tconn 192.168.3.9:5555
  ```

- 运行 hap



5. 总结

你已经成功完成了 HarmonyOS 应用开发 E2E 体验，学到了：

- 如何创建一个 HarmonyOS Demo Project
- 如何构建一个 Hap 并且将其部署到真机
- 在 HarmonyOS 上如何使用 HarmonyOS 的视频播放的能力

二、基本控件

1. 介绍

本篇 Codelab 将实现的内容

HarmonyOS 是面向全场景多终端的分布式操作系统，使得应用程序的开发打破了智能终端互通的性能和数据壁垒，业务逻辑原子化开发，适配多端。通过一个简单应用开发，体验 HarmonyOS 的基本控件使用。

您将建立什么

在这个 Codelab 中，你将创建 Demo Project，并将 Demo 编译成 Hap，部署到智慧屏上，以实现 HarmonyOS 基本控件使用。

您将会学到什么

- 如何创建一个 HarmonyOS Demo Project
- 如何构建一个 Hap 并且将其部署到智慧屏远程模拟器上
- 通过此示例应用体验在 HarmonyOS 上如何使用基本控件，包括文本输入框，日期选择控件，单选按钮，下拉菜单和按钮。

2. 您需要什么

硬件要求

- 操作系统：Windows10 64 位
- 内存：8G 及以上。
- 硬盘：100G 及以上。
- 分辨率：1280*800 及以上

软件要求

- DevEco Studio：需手动下载安装，详细步骤请参考《DevEco Studio 使用指南》2.1.2
- JDK：DevEco Studio 自动安装。
- Node.js：请手动下载安装，详细步骤请参考《DevEco Studio 使用指南》2.1.3 下载和安装 Node.js。
- HarmonyOS SDK：待 DevEco Studio 安装完成后，利用 DevEco Studio 来加载 HarmonyOS SDK。详细步骤请参考《DevEco Studio 使用指南》2.1.6 加载 HarmonyOS SDK。
- Maven 库依赖包：如需手动拷贝和配置，详细步骤请参考《DevEco Studio 使用指南》2.3 离线方式配置 Maven 库。

需要的知识点

- Java 基础开发能力。

3. 能力接入准备

实现 HarmonyOS 应用开发，需要完成以下准备工作：

- 环境准备。
- 环境搭建。
- 创建项目
- 申请调试证书
- 应用开发

具体操作，请按照《DevEco Studio 使用指南》中详细说明来完成。

提示：需要通过注册成开发者才能完成集成准备中的操作。

4. 代码片段

1. 在 index.hml 文件中写入以下代码：

```
// input (text) 文本输入，输入用户名。
<div class="item-content">
        <input class="input-text" placeholder="{{userNamePromt}}"
onchange="getName" type="text"></input> // 设置占位内容：
userNamePromt，设置 input 类型为 text
</div>

// input (radio) 单选按钮，输入性别。
<div class="item-content">
      <label target="radio1">{{ $t('Strings.male') }}:</label>
      <input id="radio1" type="radio" name="radio"
value="{{ $t('Strings.male') }}" onchange="getGender"></input>
      <label target="radio2">{{ $t('Strings.female') }}:</label>
      <input id="radio2" type="radio" name="radio"
value="{{ $t('Strings.female') }}" onchange="getGender"></input>
      <label target="radio3">{{ $t('Strings.secret') }}:</label>
```

```
        <input id="radio3" type="radio" name="radio"
value="{{ $t('Strings.secret') }}" onchange="getGender"></input>
</div>

// picker  日期选择控件，输入日期。
<div class="item-content">
        <picker type="date" end="2020-01-01" selected="1990-01-01"
value="{{today}}" onchange="getDate"></picker>
</div>

// select   下拉菜单，输入学历。
<select class="select">
        <option value=" "> </option>
        <option
value="{{$t('Strings.graduated')}}">{{$t('Strings.graduated')}}</option>
        <option
value="{{$t('Strings.bachelor')}}">{{$t('Strings.bachelor')}}</option>
        <option value="{{$t('Strings.master')}}">{{$t('Strings.master')}}</option>
        <option value="{{$t('Strings.doctor')}}">{{$t('Strings.doctor')}}</option>
</select>
```

2. 在 index.js 文件中写入以下代码：

```
data: {
        userNamePromt:'这是占位符',
        today："2020-08-26"
},
getName(e){
        console.info("用户名是" + e.value);
},
getGender(e){
        console.info("性别是" + e.value);
},
getDate(e){
        console.info("选择日期是" + e.value);
},
```

3. 在 js/default/i18n/文件中写入以下代码：

在 zh-CN.json 文件中写入：

```
"Strings": {
            "hello": "您好",
                        "world": "世界",
                        "male": "男",
```

```
        "female": "女",
        "secret": "保密",
        "graduated": "本科",
        "bachelor": "学士",
        "master": "硕士",
        "doctor": "博士"
    },
```

相应的，在 en-US.json 文件中写入：

```
"Strings": {
        "hello": "Hello",
        "world": "World",
        "male": "male",
        "female": "female",
        "secret": "secret",
        "graduated": "graduated",
        "bachelor": "bachelor",
        "master": "master",
        "doctor": "doctor"
    },
```

4. 编译运行该应用

- 通过 hdc 连接大屏设备

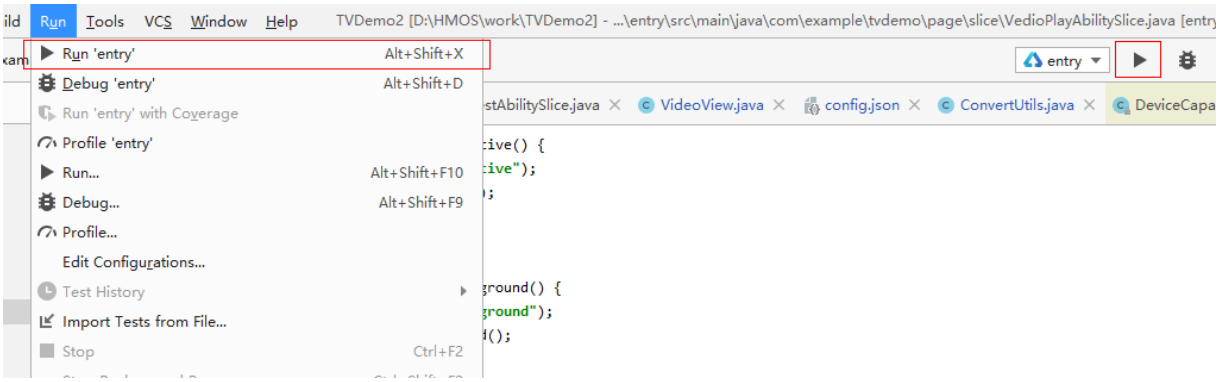
  先查看智慧屏 IP：

  ```
  大屏设置->"网络与连接"->"网络"->"有线网络"
  ```

  在 cmd 或者 IDE 的 Terminal 输入命令：

  ```
  hdc tconn 192.168.3.9:5555
  ```

- 运行 hap



5. 总结

您已经成功完成了 HarmonyOS 应用开发 E2E 体验，学到了：

- 如何创建一个 HarmonyOS Demo Project
- 如何构建一个 Hap 并且将其部署到真机
- 在 HarmonyOS 上如何使用基本控件，包括文本输入框，日期选择控件，单选按钮，下拉菜单和按钮。

三、应用偏好数据读写

1. 介绍

本篇 Codelab 将实现的内容

HarmonyOS 是面向全场景多终端的分布式操作系统，使得应用程序的开发打破了智能终端互通的性能和数据壁垒，业务逻辑原子化开发，适配多端。通过一个简单应用开发，体验 HarmonyOS 的轻量级偏好数据库能力

您将建立什么

在这个 Codelab 中，你将创建 Demo Project，并将 Demo 编译成 Hap，此示例应用程序展示了如何使用轻量级偏好数据库。

您将会学到什么

- 如何创建一个 HarmonyOS Demo Project
- 如何构建一个 Hap 并且将其部署到智慧屏真机
- 通过此示例应用体验如何使用轻量级偏好数据库

2. 您需要什么

硬件要求

- 操作系统：Windows10 64 位
- 内存：8G 及以上。
- 硬盘：100G 及以上。
- 分辨率：1280*800 及以上

软件要求

- DevEco Studio：需手动下载安装，详细步骤请参考《DevEco Studio 使用指南》2.1.2
- JDK：DevEco Studio 自动安装。
- Node.js：请手动下载安装，详细步骤请参考《DevEco Studio 使用指南》2.1.3 下载和安装 Node.js。
- HarmonyOS SDK：待 DevEco Studio 安装完成后，利用 DevEco Studio 来加载 HarmonyOS SDK。详细步骤请参考《DevEco Studio 使用指南》2.1.6 加载 HarmonyOS SDK。
- Maven 库依赖包：如需手动拷贝和配置，详细步骤请参考《DevEco Studio 使用指南》2.3 离线方式配置 Maven 库。

需要的知识点

- Java 基础开发能力。

3. 能力接入准备

实现 HarmonyOS 应用开发，需要完成以下准备工作：

- 环境准备。

- 环境搭建。
- 创建项目
- 申请调试证书
- 应用开发

具体操作，请按照《DevEco Studio 使用指南》中详细说明来完成。

提示：需要通过注册成开发者才能完成集成准备中的操作。

4. 代码片段

1. 布局：

- 布局代码：

```
LayoutConfig config = new LayoutConfig(LayoutConfig.MATCH_PARENT, LayoutConfig.MATCH_PARENT);
myLayout.setLayoutConfig(config);
myLayout.setOrientation(Component.VERTICAL);
ShapeElement element = new ShapeElement();
element.setRgbColor(new RgbColor(255, 255, 255));
myLayout.setBackground(element);
log = createText("日志信息");
myLayout.addComponent(log);
writeBtn = createBtn("写入 preferences 数据", new RgbColor(0, 0, 255), 1002);
readBtn = createBtn("读取 preferences 数据", new RgbColor(0, 0, 255), 1003);
addObserver = createBtn("注册观察者", new RgbColor(255, 0, 0), 1004);
private Text createText(String title) {
Text text = new Text(this);
DirectionalLayout.LayoutConfig config = new DirectionalLayout.LayoutConfig(DirectionalLayout.LayoutConfig.MATCH_CONTENT, DirectionalLayout.LayoutConfig.MATCH_CONTENT);
text.setLayoutConfig(config);
text.setText(title);
text.setTextSize(48);
text.setTextColor(new Color(0xFF0000FF));
return text;
}
```

```
private Button createBtn(String title, RgbColor color, int id) {
Button btn = new Button(this);
LayoutConfig configBtn = new LayoutConfig(500, 100);
configBtn.topMargin = 30;
btn.setLayoutConfig(configBtn);
btn.setText(title);
btn.setId(id);
btn.setTextSize(48);
btn.setTextColor(new Color(0xFFFFFFFF));
ShapeElement elementBtn = new ShapeElement();
elementBtn.setRgbColor(color);
elementBtn.setCornerRadius(12);
btn.setBackground(elementBtn);
myLayout.addComponent(btn);
return btn;
}
```

2. Preferences 使用

- Preferences 初始化

```
private void initPreferences() {
DatabaseHelper databaseHelper = new DatabaseHelper(this);
String fileName = "user_info";
preferences = databaseHelper.getPreferences(fileName);
}
```

- 写文件：

```
preferences.putInt("age", Integer.parseInt(age.getText()));
preferences.putString("name", name.getText());
preferences.flushSync();
```

- 读文件：

```
int age = preferences.getInt("age", 0);
String name = preferences.getString("name", "");
ToastDialog toastDialog = new
ToastDialog(PreferencesAbilitySlice.this);
toastDialog.setText("read user data frome preferences name:" + name
+ ", age:" + age);
toastDialog.show();
```

- 观察者：

  注册：

```
counter = new PreferencesChangeCounter();
```

```
preferences.registerObserver(counter);
private class PreferencesChangeCounter implements
Preferences.PreferencesObserver {
    @Override
    public void onChange(Preferences preferences, String key) {
        if ("name".equals(key)) {
            String name = preferences.getString("name", "");
            log.setText("user data name is edit:" + name);
        }
        if ("age".equals(key)) {
            int age = preferences.getInt("age", 0);
            log.setText("user data age is edit:" + age);
        }
    }
}
```

删除：

```
preferences.unregisterObserver(counter);
```

2. 响应遥控器点击

```
private void addFocusChangedListener(Component view) {
    view.setFocusChangedListener(new
Component.FocusChangedListener() {
        @Override
        public void onFocusChange(Component component, boolean b) {
            ShapeElement shapeElement = (ShapeElement)
view.getBackgroundElement();
            if (b) {
                shapeElement.setStroke(10, new RgbColor(0, 0, 0));
                focusView = view;
            } else {
                shapeElement.setStroke(0, new RgbColor(0, 0, 0));
            }
        }
    });
}

@Override
public boolean onKeyUp(int keyCode, KeyEvent keyEvent) {
    switch (keyCode) {
        case KeyEvent.KEY_DPAD_CENTER:
        case KeyEvent.KEY_ENTER:
            if(focusView == writeBtn) {
```

```
                    preferences.putInt("age", index++);
                    preferences.putString("name", "张三");
                    preferences.flushSync();
                }
                if(focusView == readBtn) {
                    int age = preferences.getInt("age", 0);
                    String name = preferences.getString("name", "");
                    log.setText("read user data frome preferences name:" +
name + ", age:" + age);
                }
                if(focusView == addObserver) {
                    if (addObserver.getText().equals("注册观察者")) {
                        addObserver.setText("删除观察者");
                        // 向 preferences 实例注册观察者
                        counter = new PreferencesChangeCounter();
                        preferences.registerObserver(counter);

                    } else {
                        addObserver.setText("注册观察者");
                        // 向 preferences 实例注销观察者
                        preferences.unregisterObserver(counter);
                    }
                }
                return true;
            case KeyEvent.KEY_DPAD_UP:
                int position = views.indexOf(focusView.getId());
                if (position > 0) {
                    switch (position) {
                        case 1:
                            writeBtn.requestFocus();
                            break;
                        case 2:
                            readBtn.requestFocus();
                            break;
                        default:
                            break;
                    }
                }
                return true;
            case KeyEvent.KEY_DPAD_DOWN:
                position = views.indexOf(focusView.getId());
                if (position < 3) {
                    switch (position) {
```

```
                        case 0:
                            readBtn.requestFocus();
                            break;
                        case 1:
                            addObserver.requestFocus();
                            break;
                        default:
                            break;
                    }
                }
                return true;
            }
            return false;
        }
```

3.编译运行该应用

- 通过 hdc 连接大屏设备

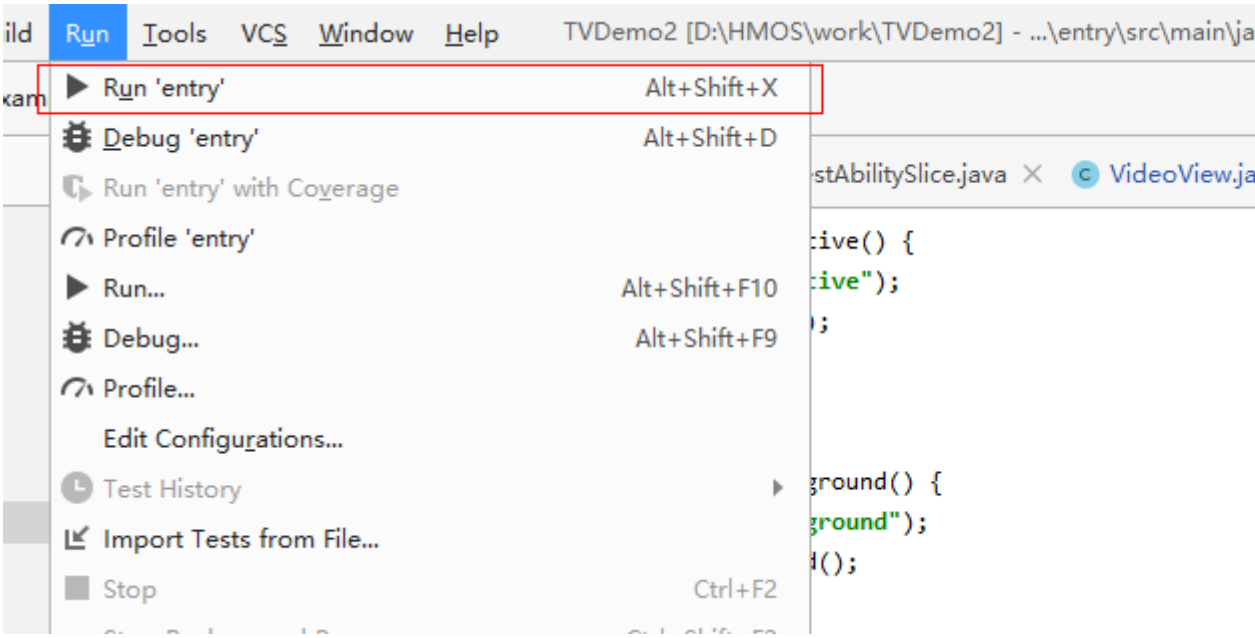
  先查看智慧屏 IP：

  ```
  大屏设置->"网络与连接"->"网络"->"有线网络"
  ```

  在 cmd 或者 IDE 的 Terminal 输入命令：

  ```
  hdc tconn 192.168.3.9:5555
  ```

- 运行 hap



5. 恭喜你

你已经成功完成了 HarmonyOS 应用开发 E2E 体验，学到了：

- 如何创建一个 HarmonyOS Demo Project
- 如何构建一个 Hap 并且将其部署到真机上
- 在 HarmonyOS 上如何使用 HarmonyOS 的轻量级偏好数据库

四、剪切板

1. 介绍

本篇 CodeLab 将实现的内容

HarmonyOS 是面向全场景多终端的分布式操作系统，使得应用程序的开发打破了智能终端互通的性能和数据壁垒，业务逻辑原子化开发，适配多端。通过一个简单应用开发，体验 HarmonyOS 的剪切板能力。

您将建立什么

在这个 CodeLab 中，你将创建 Demo Project，并将 Demo 编译成 HAP，此示例应用程序展示了如何使用 HarmonyOS 剪切板复制文字。

您将会学到什么

- 如何创建一个 HarmonyOS Demo Project
- 如何构建一个 HAP 并且将其部署到智慧屏上
- 通过此示例应用体验剪切板复制粘贴文本。

2. 您需要什么

硬件要求

- 操作系统：Windows10 64 位
- 内存：8G 及以上
- 硬盘：100G 及以上
- 分辨率：1280*800 及以上

软件要求

- 安装 DevEco Studio 和 Node.js，详情请参考下载和安装软件
- 设置 DevEco Studio 开发环境，DevEco Studio 开发环境需要依赖于网络环境，需要连接上网络才能确保工具的正常使用，可以根据如下两种情况来配置开发环境
  1. 如果可以直接访问 Internet，只需进行下载 HarmonyOS SDK 操作

2. 如果网络不能直接访问 Internet，需要通过代理服务器才可以访问，请参考配置开发环境

需要的知识点

- Java 基础开发能力。

3. 能力接入准备

实现 HarmonyOS 应用开发，需要完成以下准备工作：

- 创建 TV 的工程
- 准备密钥和证书请求文件
- 申请调试证书
- 应用开发

具体操作，请按照《DevEco Studio 使用指南》中详细说明来完成。

4. 代码编写

实现"复制文本"功能，代码片段如下：

```
private static final String ADDITON_KEY = "ADDITION_KEY";

mPasteboard = SystemPasteboard.getSystemPasteboard(this);
PasteData pasteData = new PasteData();
pasteData.addTextRecord("copyText");
PacMap pacMap = new PacMap();
pacMap.putString(ADDITION_KEY, "ADDITION_VALUE_OF_TEXT");
pasteData.getProperty().setAdditions(pacMap);
pasteData.getProperty().setTag("USER_TAG");
pasteData.getProperty().setLocalOnly(true);
mPasteboard.setPasteData(pasteData);
mShowText.setText("copy text succeed");
```

实现"粘贴文本"功能，代码片段如下：

```
private static final String ADDITON_KEY = "ADDITION_KEY";

mPasteboard = SystemPasteboard.getSystemPasteboard(this);
mPasteData = mPasteboard.getPasteData();
```

```
mRecord = mPasteData.getRecordAt(0);
mShowText.append(mRecord.getPlainText().toString());
mShowText.append(mRecord.getMimeType());
mShowText.append(mPasteData.getProperty().getTag().toString());
PacMap pacMap = mPasteData.getProperty().getAdditions();
String extraInfo = pacMap.getString(ADDITON_KEY);
if ((extraInfo != null && !extraInfo.isEmpty())) {
      mShowText.setText("value is " + extraInfo);
}
```

提示：以上代码仅是 demo 演示参考使用

5. 编译运行

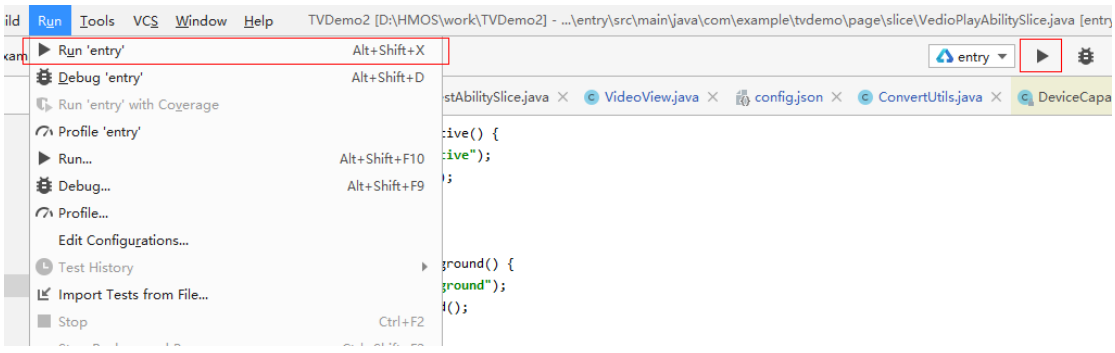
通过 hdc 连接大屏设备

先查看智慧屏 IP：大屏设置->"网络与连接"->"网络"->"无线网络"

在 cmd 或者 IDE 的 Terminal 输入命令：

```
hdc tconn 192.168.xxx.xxx:5555
```

运行



6. 恭喜您

您已经成功完成了 HarmonyOS 应用开发体验，学到了：

- 如何创建一个 HarmonyOS Demo Project
- 如何构建一个 HAP 并且将其部署到真机上
- 在 HarmonyOS 上如何使用剪切板复制粘贴文本

五、分布式任务调度

1. 介绍

本篇 CodeLab 将实现的内容

HarmonyOS 是面向全场景多终端的分布式操作系统，使得应用程序的开发打破了智能终端互通的性能和数据壁垒，业务逻辑原子化开发，适配多端。通过一个简单应用开发，体验 HarmonyOS 的分布式任务调度。

您将建立什么

在这个 CodeLab 中，你将创建 Demo Project，并将 Demo 编译成 HAP，此示例应用程序展示了如何使用分布式任务调度。

您将会学到什么

- 如何创建一个 HarmonyOS Demo Project
- 如何构建一个 HAP 并且将其部署到智慧屏真机
- 通过此示例应用体验如何使用分布式任务调度

2. 您需要什么

硬件要求

- 操作系统：Windows10 64 位
- 内存：8G 及以上
- 硬盘：100G 及以上
- 分辨率：1280*800 及以上

软件要求

- 安装 DevEco Studio 和 Node.js，详情请参考下载和安装软件
- 设置 DevEco Studio 开发环境，DevEco Studio 开发环境需要依赖于网络环境，需要连接上网络才能确保工具的正常使用，可以根据如下两种情况来配置开发环境
  1. 如果可以直接访问 Internet，只需进行下载 HarmonyOS SDK 操作

2. 如果网络不能直接访问 Internet，需要通过代理服务器才可以访问，请参考配置开发环境

需要的知识点

- Java 基础开发能力。

3. 能力接入准备

实现 HarmonyOS 应用开发，需要完成以下准备工作：

- 创建 TV 的工程
- 准备密钥和证书请求文件
- 申请调试证书
- 应用开发

具体操作，请按照《DevEco Studio 使用指南》中详细说明来完成。

4. 代码编写

分布式任务调用代码参考：

```
Intent intent = new Intent();
// BUNDLE_NAME 和 ABILITY_NAME 对应开发者需要进行分布式任务调度的 Ability 信息
Operation operation = new Intent.OperationBuilder()
        .withDeviceId(info.getDeviceId())
        .withBundleName(BUNDLE_NAME)
        .withAbilityName(ABILITY_NAME)
        .withFlags(Intent.FLAG_ABILITYSLICE_MULTI_DEVICE)
        .build();
intent.setOperation(operation);
try {
    // FLAGS_TO_QUERY 和 USERID_TO_QUERY 分别对应查询 ability 的 flags 和 userid，具体可以参考 API-DOC
    List<AbilityInfo> abilityInfos =
getBundleManager().queryAbilityByIntent(intent, 0, 0);
    if (abilityInfos != null && !abilityInfos.isEmpty()) {
        startAbility(intent);
    }
} catch (RemoteException re) {
```

```
        HiLog.error(TAG, "RemoteException");
}
```

提示：以上代码仅 demo 演示参考使用

5. 编译运行

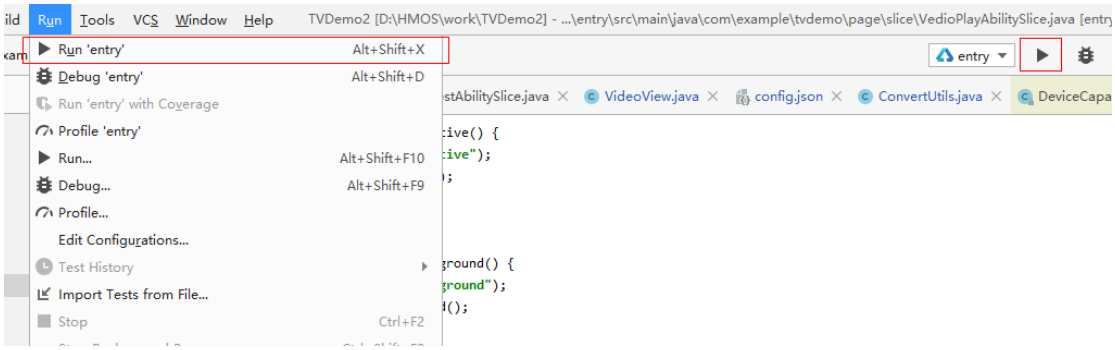
通过 hdc 连接大屏设备

先查看智慧屏 IP：大屏设置->"网络与连接"->"网络"->"无线网络"

在 cmd 或者 IDE 的 Terminal 输入命令：

```
hdc tconn 192.168.xxx.xxx:5555
```

运行



6. 恭喜您

您已经成功完成了 HarmonyOS 应用开发体验，学到了：

- 如何创建一个 HarmonyOS Demo Project
- 如何构建一个 HAP 并且将其部署到真机上
- 在 HarmonyOS 上如何使用分布式任务调度能力

六、元程序交互

1. 介绍

本篇 CodeLab 将实现的内容

HarmonyOS 是面向全场景多终端的分布式操作系统，使得应用程序的开发打破了智能终端互通的性能和数据壁垒，业务逻辑原子化开发，适配多端。通过一个简单应用开发，体验 HarmonyOS 的元程序调度能力。
•有界面元程序 A 拉起另外一个有界面元程序 B，在元程序 B 上可以读到的意图。
•元程序 B 可以回数据给元程序 A，元程序 A 收到的返回信息。

您将建立什么

在这个 CodeLab 中，你将创建 Demo Project，并将 Demo 编译成 HAP，此示例应用程序展示了如何使用有界面元程序。

您将会学到什么

- 如何创建一个 HarmonyOS Demo Project
- 如何通过实现界面跳转以及数据传递

2. 您需要什么

硬件要求

- 操作系统：Windows10 64 位
- 内存：8G 及以上
- 硬盘：100G 及以上
- 分辨率：1280*800 及以上

软件要求

- 安装 DevEco Studio 和 Node.js，详情请参考下载和安装软件
- 设置 DevEco Studio 开发环境，DevEco Studio 开发环境需要依赖于网络环境，需要连接上网络才能确保工具的正常使用，可以根据如下两种情况来配置开发环境

1. 如果可以直接访问 Internet，只需进行下载 HarmonyOS SDK 操作
2. 如果网络不能直接访问 Internet，需要通过代理服务器才可以访问，请参考配置开发环境

需要的知识点

- Java 基础开发能力。

3. 能力接入准备

实现 HarmonyOS 应用开发，需要完成以下准备工作：

- 创建 TV 的工程
- 准备密钥和证书请求文件
- 申请调试证书
- 应用开发

具体操作，请按照《DevEco Studio 使用指南》中详细说明来完成。

4. 代码编写

核心代码参考

```
Operation operation = new Intent.OperationBuilder()
        .withDeviceId("")
        .withBundleName("com.huawei.codelab")
        .withAbilityName("com.huawei.codelab.CalleeAbility")
        .build();
Intent intent = new Intent();
intent.setOperation(operation);
startAbility(intent);
```

提示：以上代码仅 demo 演示参考使用

5. 编译运行

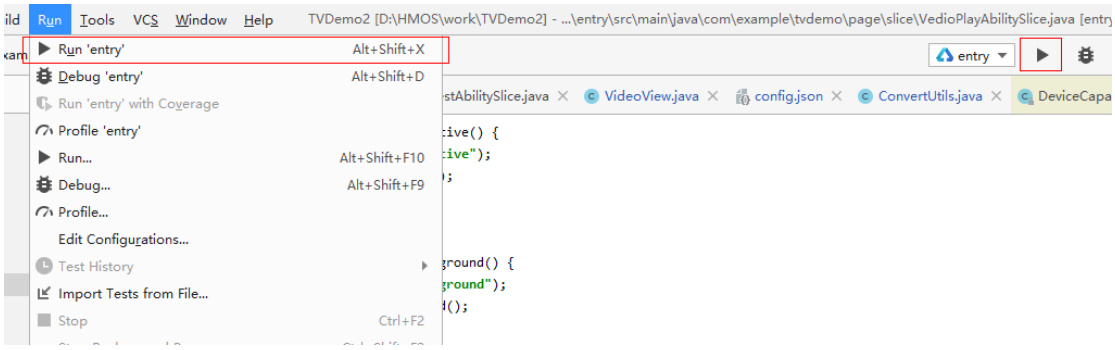
通过 hdc 连接大屏设备

先查看智慧屏 IP：大屏设置->"网络与连接"->"网络"->"无线网络"

在 cmd 或者 IDE 的 Terminal 输入命令：

```
hdc tconn 192.168.xxx.xxx:5555
```

运行



6. 恭喜您

您已经成功完成了 HarmonyOS 应用开发体验，学到了：

- 如何创建一个 HarmonyOS Demo Project
- 如何构建一个 HAP 并且将其部署到真机上
- 如何实现元能力界面跳转

七、UI 设计开发与预览

1. 介绍

通过智能表待办应用开发，让开发者了解智能表 HarmonyOS 应用开发的全流程，实现从工程创建到界面预览全过程。使用 HUAWEI DevEco Studio 开发 HarmonyOS 待办应用，完成工程创建、代码编辑，界面预览等开发过程。

您将建立什么

在这个 CodeLab 中，你将创建一个智能表的 Demo Project（harmony-todo），以及完成一个待办应用的页面的搭建和预览。

您将学到什么

- 如何搭建一个 APP 并添加页面布局
- 如何实时预览创建的页面布局信息
- 完成智能表应用的页面搭建和预览

硬件要求

- 操作系统：Windows10 64 位
- 内存：8G 及以上
- 硬盘：100G 及以上
- 分辨率：1280*800 及以上

软件要求

- 安装 DevEco Studio 和 Node.js，详情请参考下载和安装软件

提示：智能表 UI 开发的预览功能将在 Beta2 版本上线，当前只能在 CodeLab 现场体验尝鲜

- 设置 DevEco Studio 开发环境，DevEco Studio 开发环境需要依赖于网络环境，需要连接上网络才能确保工具的正常使用，可以根据如下两种情况来配置开发环境
- 如果可以直接访问 Internet，只需进行下载 HarmonyOS SDK 操作
- 如果网络不能直接访问 Internet，需要通过代理服务器才可以访问，请参考配置开发环境

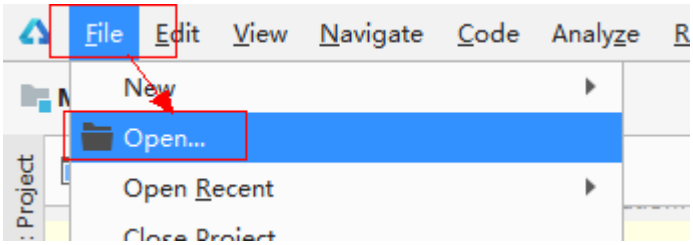
提示：下载 HarmonyOS SDK 时，需要下载 JS SDK 和 SDK Tools 中的 Previewer

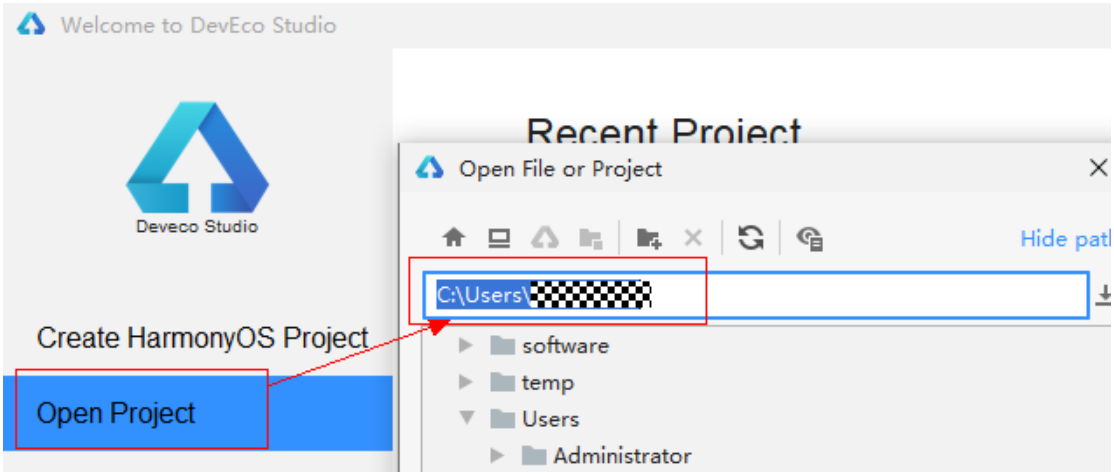
需要的知识点

- html、CSS、JavaScript 开发基础能力

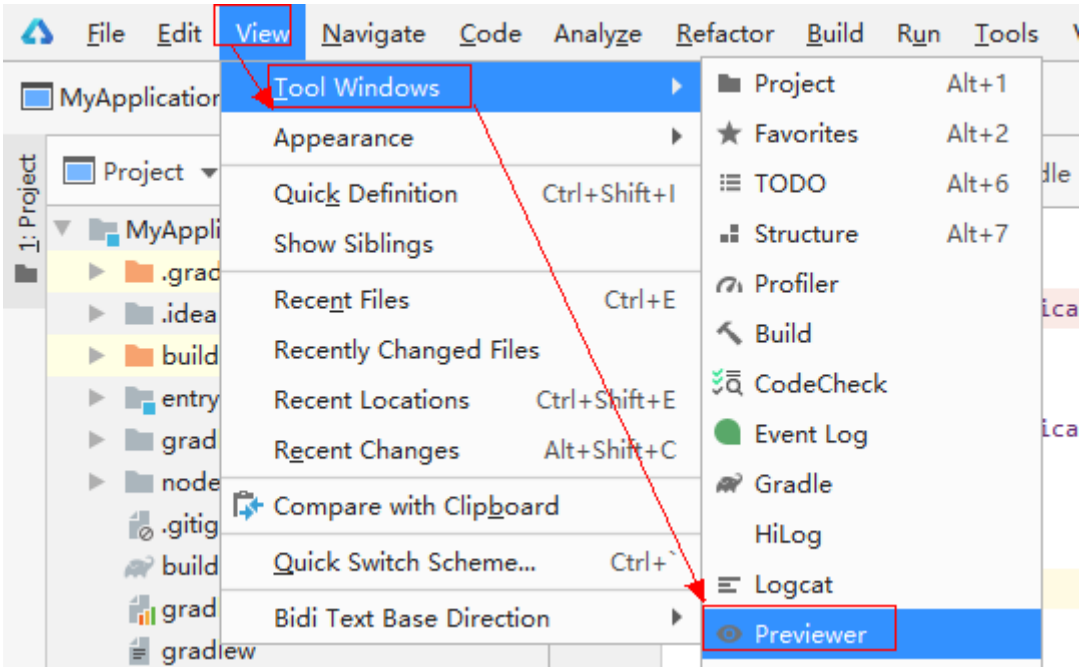
2. 代码开发

1.打开本地 Demo Project（harmony-todo）



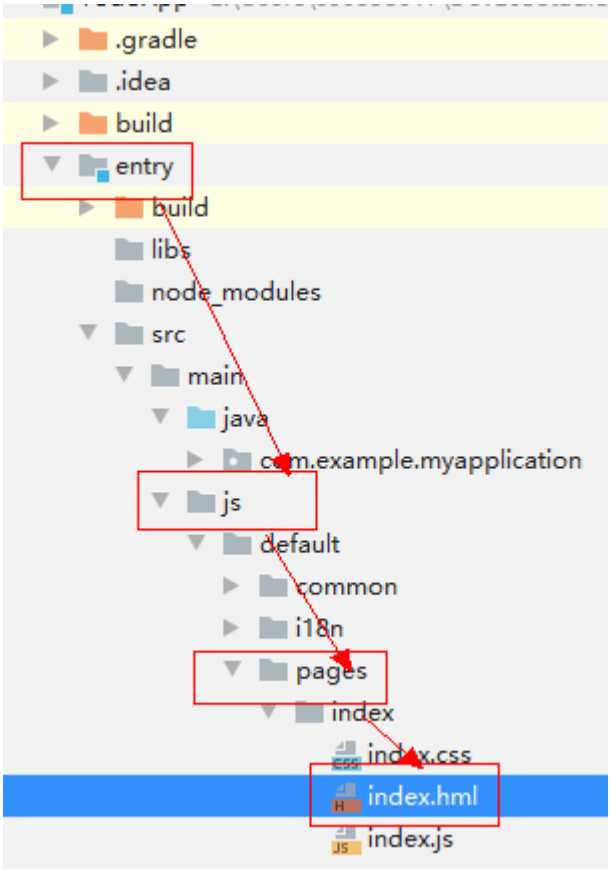


2.点击 Previewer 按钮，实时预览 Demo Project（harmony-todo）



提示：开发过程完成每一步点击保存之后即可在预览界面实时预览效果

3.为 index 页面（index.hml）添加布局信息



3.1 添加今日待办事项的列表

```
<todo-header title="今日待办" type="" @add-event="toAddEvent"></todo-
header>
<list-item for="{{todayList}}" class="tag-list-item"
clickeffect="false">
    <todo-list @complete-event="completeEvent" @delete-
event="deleteEvent" todo="{{$item}}"
            todos="{{todayList}}" index="{{$idx}}"></todo-list>
</list-item>
```

提示：**以上代码只是 demo 演示，产品化的代码需要使用国际化**

预览效果如下所示：

3.2 添加明日待办事项的列表

```
<todo-header title="明日待办" type="" @add-event="toAddEvent"></todo-header>
<list-item for="{{tomorrowList}}" class="tag-list-item" clickeffect="false">
    <todo-list @complete-event="completeEvent" @delete-event="deleteEvent" todo="{{$item}}"
            todos="{{tomorrowList}}" index="{{$idx}}"></todo-list>
</list-item>
```

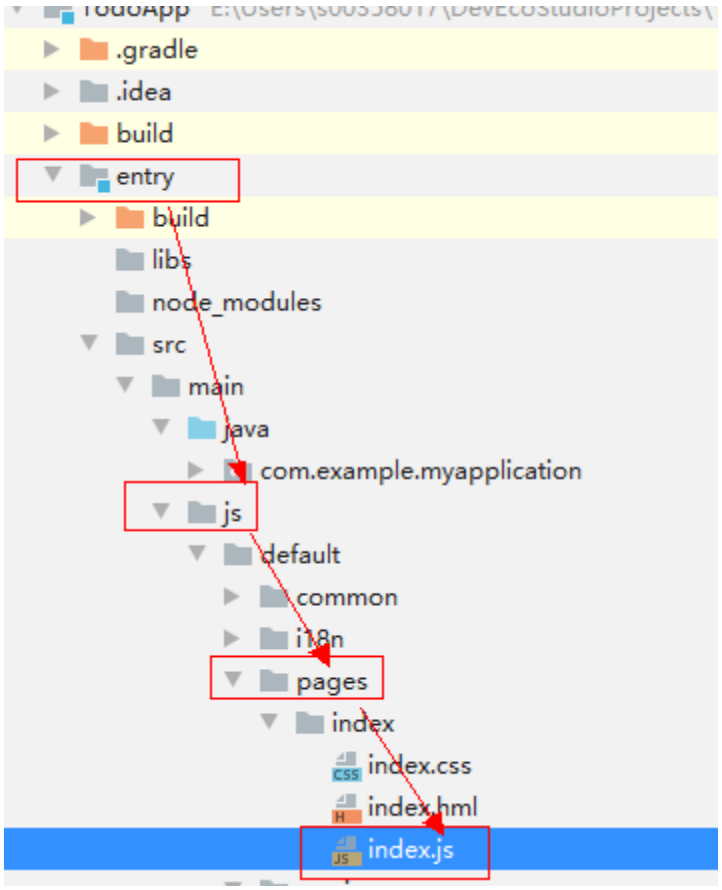
提示：**以上代码只是 demo 演示，产品化的代码需要使用国际化**

3.3 添加即将来临待办事项的列表

```
<todo-header title="即将来临" type="" @add-event="toAddEvent"></todo-
header>
<list-item for="{{laterList}}" class="tag-list-item"
clickeffect="false">
    <todo-list @complete-event="completeEvent" @delete-
event="deleteEvent" todo="{{$item}}"
            todos="{{laterList}}" index="{{$idx}}"></todo-list>
</list-item>
```

提示：**以上代码只是 demo 演示，产品化的代码需要使用国际化**

4.添加逻辑代码(index.js)



4.1 添加待办事项完成的逻辑代码

```
completeEvent(clicked) {
    var lists = clicked.detail.lists;
```

```
    var eid = clicked.detail.id;
    datahelper.completeEvent(eid, lists);
},
```

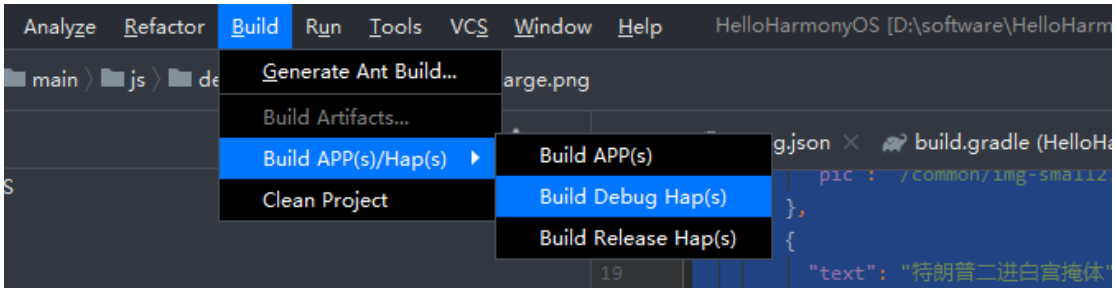
4.2 添加删除待办事项的逻辑代码

```
deleteEvent(clicked){
    var index = clicked.detail.index;
    var lists = clicked.detail.lists;
    lists.splice(index, 1);
},
```

预览效果如下图所示：

3. 编译构建

点击 Build > Build APP(s)/Hap(s) > Build Debug Hap(s)，打包生成 hap 包



4. 恭喜您

您已经成功完成了 CodeLab，并学到了：

- 如何开发一个智能表应用
- 如何实时预览开发的 UI 界面
- 完成 UI 设计开发与预览的整体流程。

八、HelloWorld 应用开发 E2E 体验

1. 介绍

智慧屏是首个搭载 HarmonyOS 的终端产品，通过 HarmonyOS 智慧屏多页签应用开发模板，让开发者了解 HarmonyOS 应用开发的全流程，20 分钟快速上手，实现从工程创建到应用运行全过程。

您将建立什么

在这个 CodeLab 中，您将使用 HUAWEI DevEco Studio 开发 HarmonyOS 智慧屏多页签应用，完成工程创建、编译构建，并实现 HarmonyOS 智慧屏部署和运行。

您将学到什么

- 如何创建一个 HarmonyOS Project
- 编译构建 hap 包
- 将 hap 包部署到智慧屏远程模拟器上，并运行

硬件要求

- 操作系统：Windows10 64 位
- 内存：8G 及以上
- 硬盘：100G 及以上
- 分辨率：1280*800 及以上

软件要求

- 安装 DevEco Studio 和 Node.js，详情请参考下载和安装软件
- 设置 DevEco Studio 开发环境，DevEco Studio 开发环境需要依赖于网络环境，需要连接上网络才能确保工具的正常使用，可以根据如下两种情况来配置开发环境
  1.如果可以直接访问 Internet，只需进行下载 HarmonyOS SDK 操作
  2.如果网络不能直接访问 Internet，需要通过代理服务器才可以访问，请参考配置开发环境

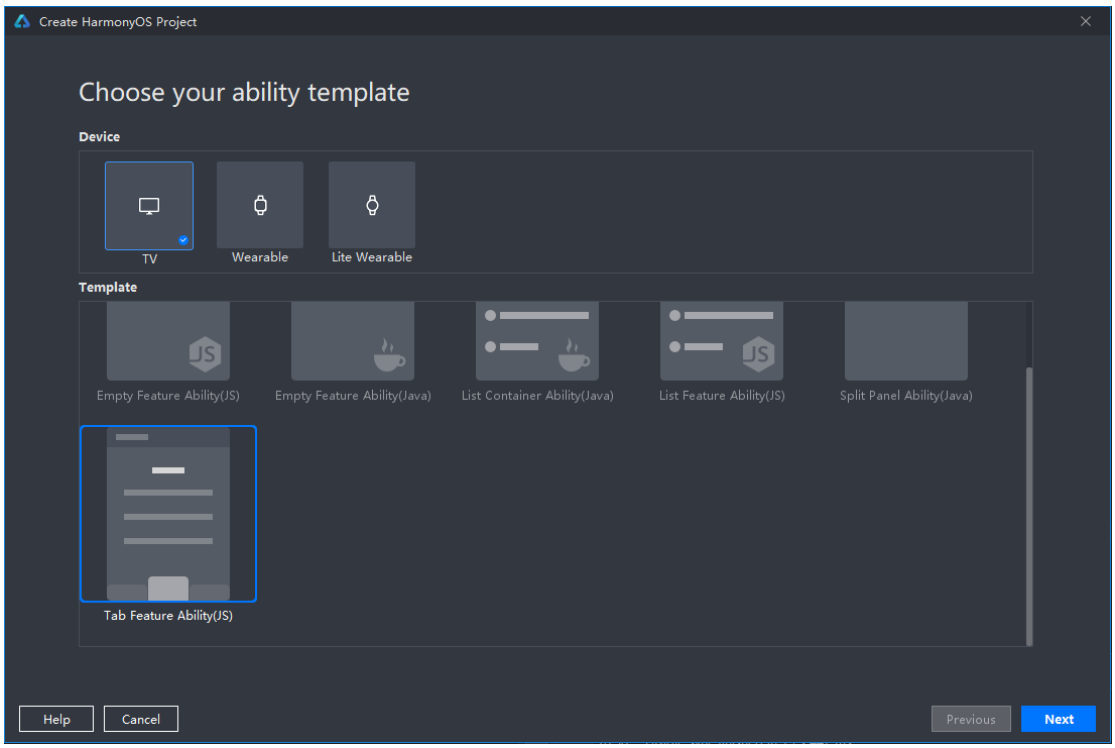
技能要求

- Java 基础开发能力
- JavaScript/HML(HarmonyOS Markup Language) /CSS 基础开发能力

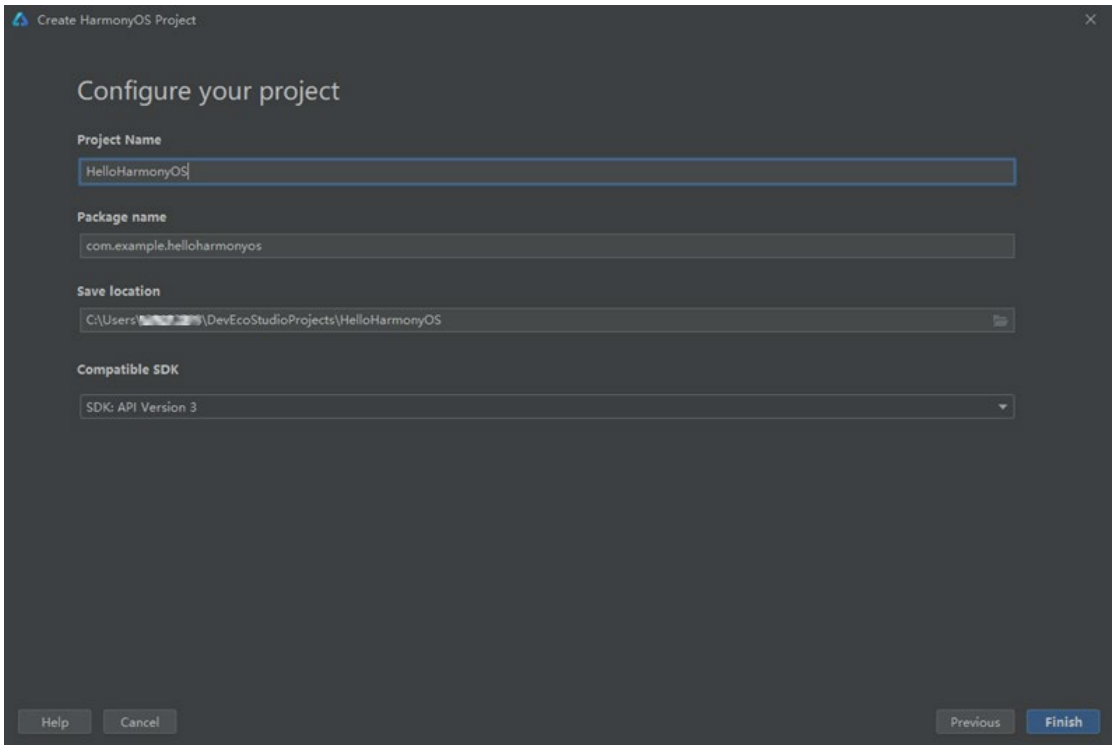
2. 代码开发

1. 工程创建

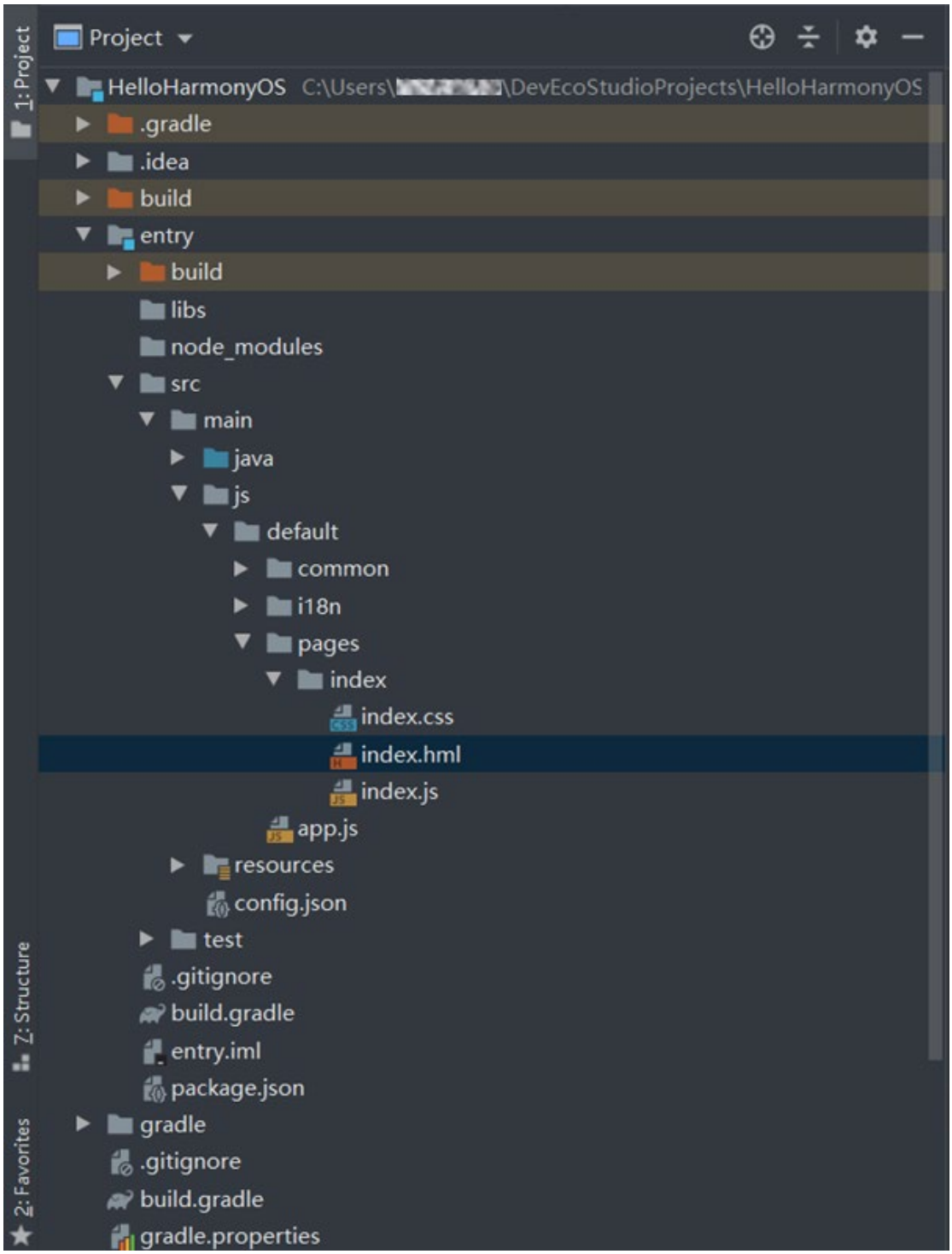
在 File > New > New Project 来创建一个新工程，选择 **Tab Feature Ability(JS)模板：**



填写工程基本信息，如命名工程名和包名：



工程创建完成后，目录结构如下：



目录结构中文件分类及作用：

- .hml 文件布局结构描述文件
- .css 页面样式描述文件
- .js 页面显示和用户交互文件

- app.js 用于全局应用生命周期管理
- pages 用于存放组件页面
- common 用于存放公共资源文件。如：媒体资源、自定义组件和 JS 文件
- resources 用于存放资源配置文件。如：全局样式、多分辨率加载等配置文件
- i18n 用于存放全球化资源

2. 编码

本次 CodeLab 需要编写布局文件、添加图片资源以及修改模块配置。

2.1 开发多页签界面布局及全球化资源

1、编辑布局文件 entry/src/main/js/default/pages/index/index.hml

```
<div class="container">
    <div class="tv_box">
        <div class="title_box">
            <text class="title">{{$t('Strings.title')}}
            </text>
            <button type="circle" icon="{{icon_src}}"
class="setting_box" onfocus="iconFocusFunc"
                onblur="iconBlurFunc"></button>
        </div>
        <tabs class="tab_box">
            <tab-bar mode="scrollable" class="bar_box">
                <block for="{{item in $t('Strings.tab')}}">
                    <text class="tab_text">{{item}}
                    </text>
                </block>
            </tab-bar>
            <tab-content>
                <block for="[1,2,3,4,5,6,7,8,9]">
                    <div class="content_box">
                        <list class="content_img">
                            <block for="{{imgIndex in
$t('Strings.images')}}">
                                <list-item type="listItem"
class="list_img">
                                    <image focusable="true"
class="tab_img" src="{{imgIndex}}"></image>
                                </list-item>
                            </block>
```

```
                        </list>
                        <div class="subtitle_box">
                            <text
class="subtitle">{{$t('Strings.subtitle')}}
                            </text>
                        </div>
                        <list class="img_list">
                            <block for="{{detailItem in
$t('Strings.details')}}">
                                <list-item type="listItem"
class="list_box">
                                    <image focusable="true"
class="img_img" src="{{detailItem.pic}}"></image>
                                    <text
class="img_text">{{detailItem.text}}
                                    </text>
                                </list-item>
                            </block>
                        </list>
                    </div>
                </block>
            </tab-content>
        </tabs>
    </div>
</div>
```

2、编辑全球化资源文件 entry/src/main/js/default/i18n/zh-CN.json

```
{
    "Strings": {
        "title": "华为智慧屏",
        "subtitle": "历史观看",
        "tab": [
            "首页",
            "电影",
            "电视剧",
            "购物",
            "溪村风景",
            "图册",
            "少儿",
            "VIP"
        ],
        "images": [
            "/common/img-large1.png",
```

```
            "/common/img-large2.png",
            "/common/img-large3.png",
            "/common/img-large4.png"
        ],
        "text": "文本内容",
        "details": [
            {
                "text": "花园",
                "pic": "/common/img-small1.png"
            },
            {
                "text": "风景一角",
                "pic": "/common/img-small2.png"
            },
            {
                "text": "蓝天白云",
                "pic": "/common/img-small3.png"
            },
            {
                "text": "池塘",
                "pic": "/common/img-small4.png"
            },
            {
                "text": "办公一角",
                "pic": "/common/img-small5.png"
            }
        ]
    },
    "Files": {}
}
```

2.2 添加图片资源

将 CodeLab PC 桌面 CodeLab/common 目录下的资源拷贝到 entry/src/main/js/default/commom。

注：可以在 common 目录点击右键，通过 Show in Explorer 快速进入目录。

2.3 修改模块配置文件

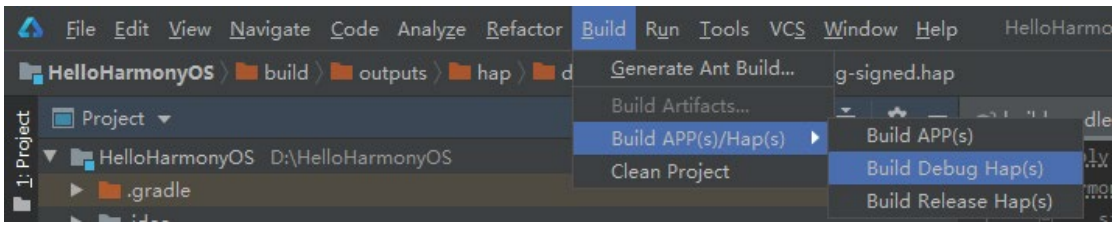
修改 entry/src/main/config.json 文件，将 designWidth 的值修改为 1024，并将 autoDesignWidth 的值修改为 false。

```
"js": [
```

```
    {
        "pages": [
            "pages/index/index"
        ],
        "name": "default",
        "window": {
            "designWidth": 1024,
            "autoDesignWidth": false
        }
    }
]
```

3. 编译构建

点击 Build > Build APP(s)/Hap(s) > Build Debug Hap(s) 进行代码编译构建，

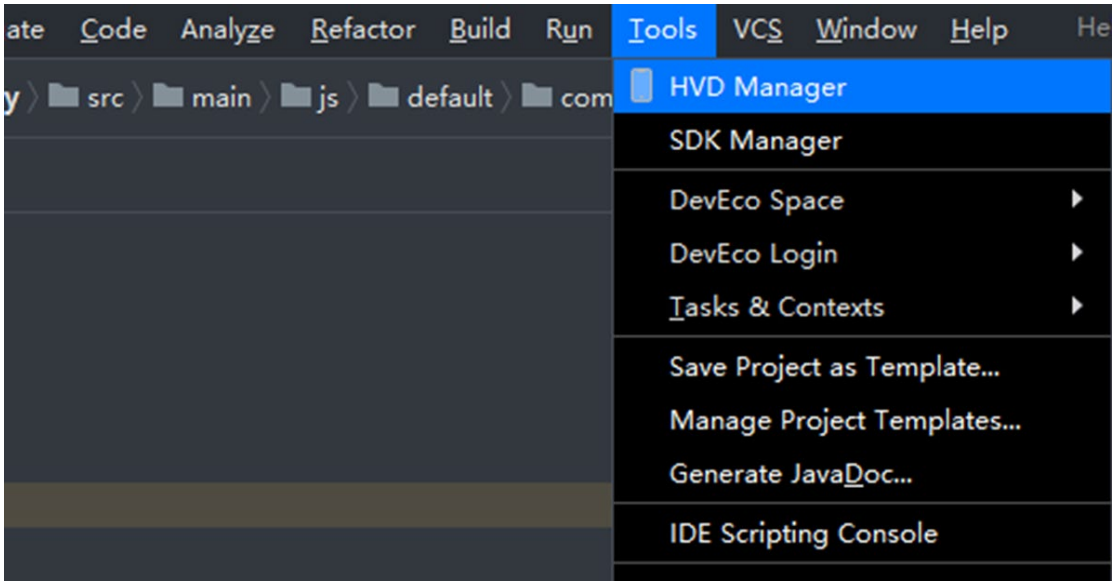


等待系统编译，在控制台看到编译成功提示信息即可。

4. 部署并运行

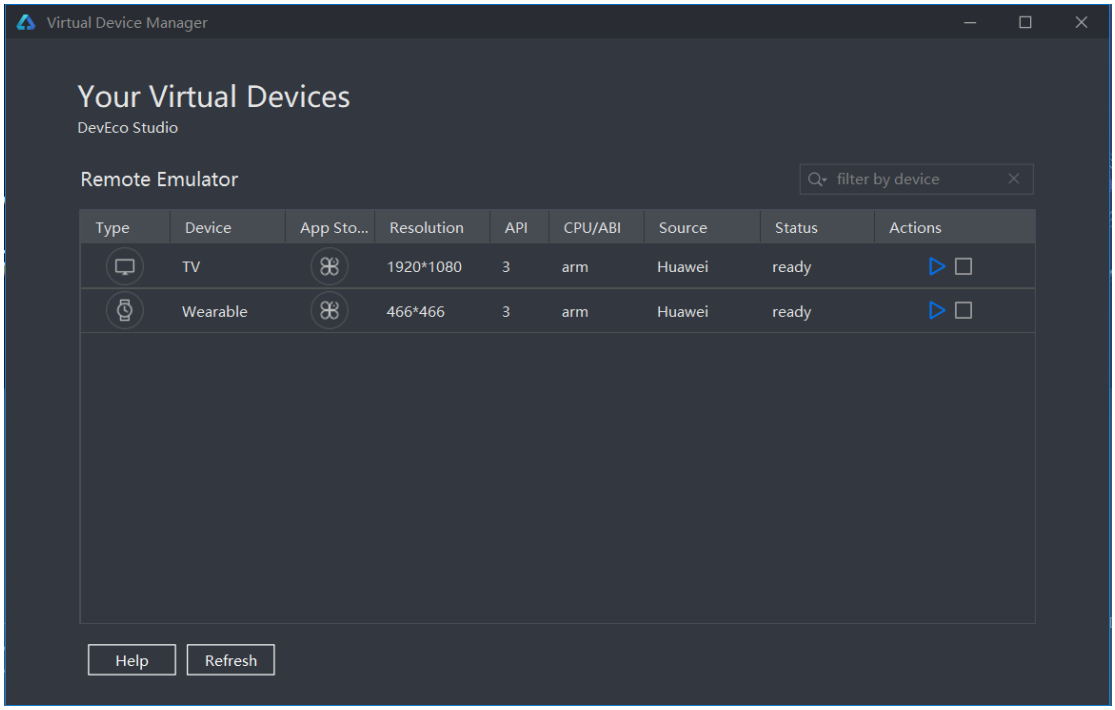
1. 启动模拟器

点击 Tools>HVD Manager 启动模拟器。

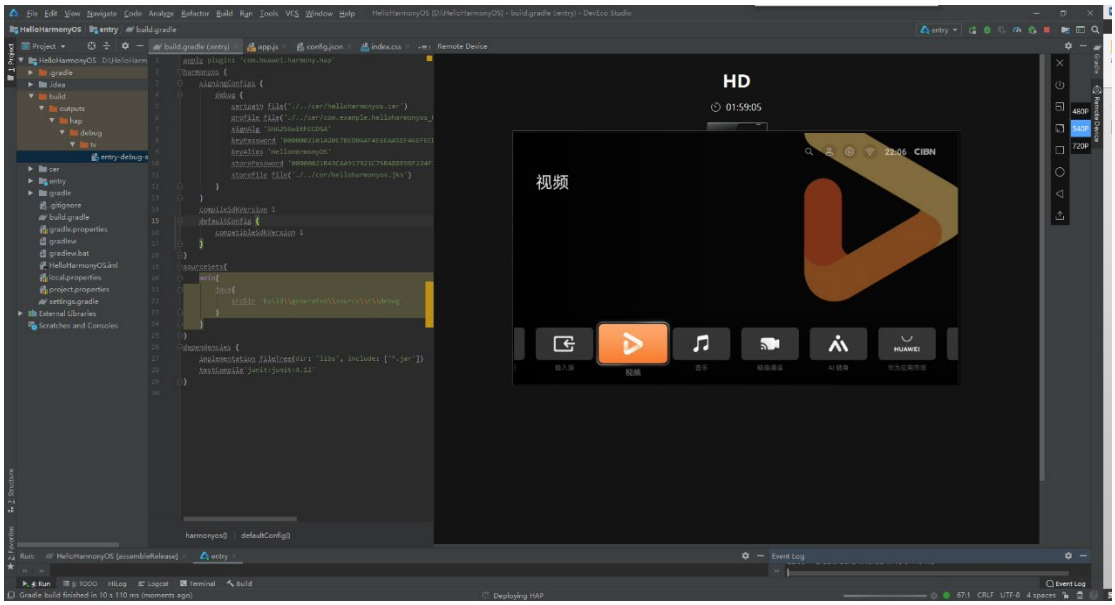


注：使用模拟器需要登录实名认证的华为开发者账号，请按照提示完成登录及授权。

选择智慧屏模拟器（TV），点击启动按钮，如下图所示。

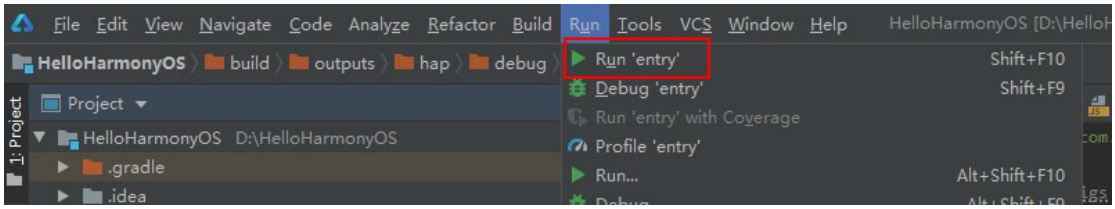


待模拟器启动成功后，出现下图所示的界面。

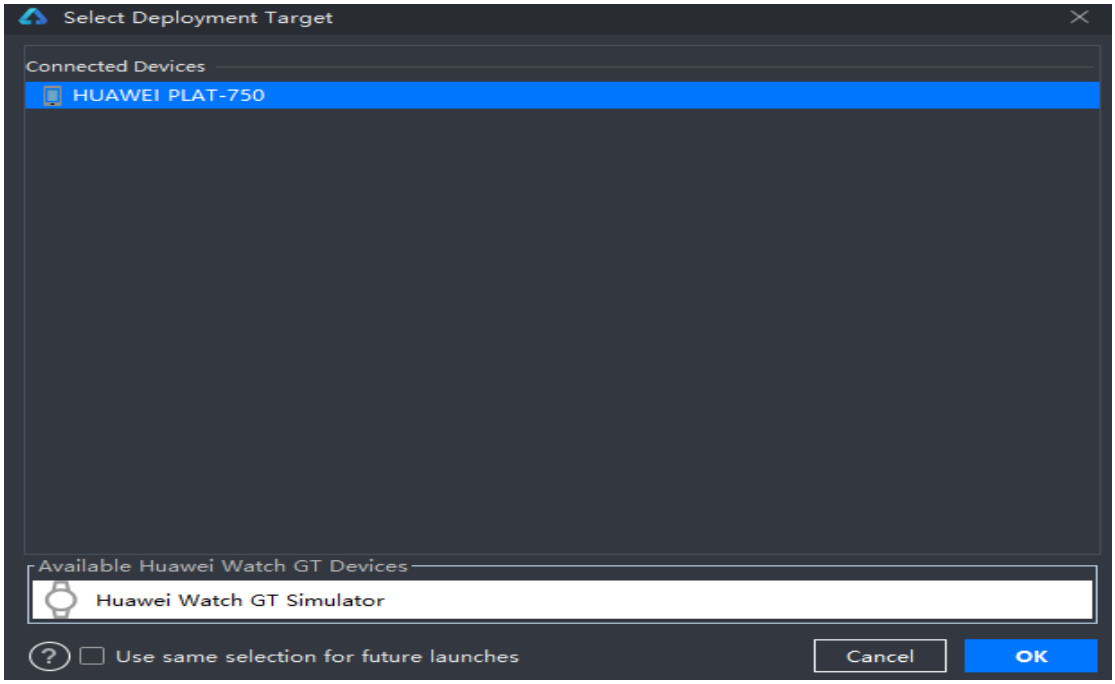


2. 部署应用

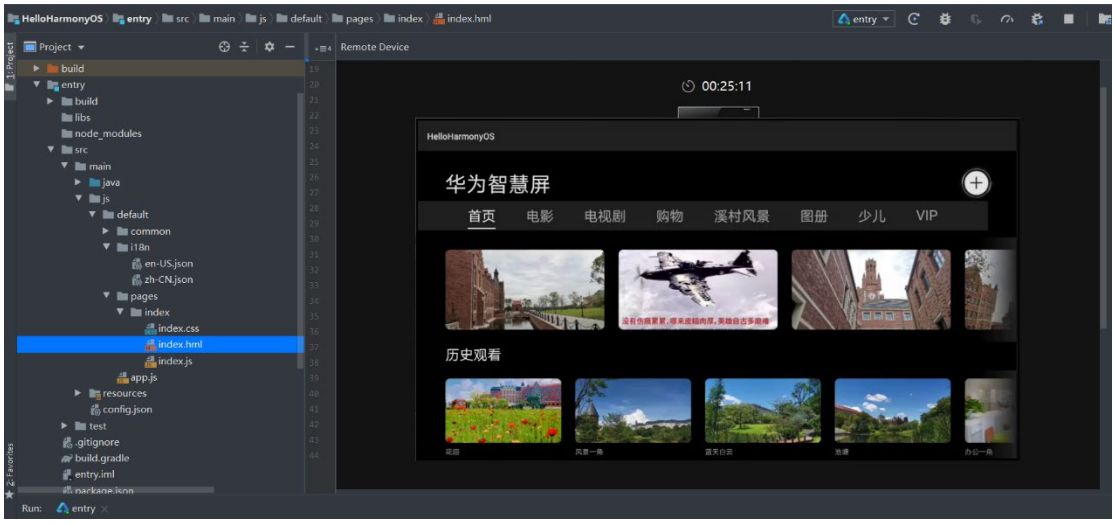
点击 **Run > Run 'Entry'**，部署应用。



选择模拟器设备。



应用程序运行如下



至此，您已经成功开发出第一个 HarmonyOS 应用，欢迎进入 HarmonyOS 世界！

5. 恭喜您

您已经成功完成了 HelloWorld 应用开发 E2E 体验，并学到了：

- 如何创建一个 HarmonyOS Project
- 编译构建 hap 包
- 将 hap 包部署到智慧屏远程模拟器上并运行

更多最新的鸿蒙相关技术文章、课程、直播等信息，欢迎关注HarmonyOS社区