/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

软件名称：机器人控制软件

版本： V1.0

作者： 中南民族大学

开发人员：汪红 后面是学生姓名

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/

AndroidManifest.xml 文件配置

<?xml version="1.0" encoding="utf-8"?>

<manifest xmlns:android="http://schemas.android.com/apk/res/android"

package="com.naorobot"

android:versionCode="1"

android:versionName="1.0" >

<uses-sdk

android:minSdkVersion="8"

android:targetSdkVersion="18" />

<!--权限可以获取使用Wi-Fi等WLAN无线网络-->

<uses-permission android:name="android.permission.ACCESS\_WIFI\_STATE"></uses-permission>

<!--允许程序改变Wi-Fi连接状态 -->

<uses-permission android:name="android.permission.CHANGE\_WIFI\_STATE"></uses-permission>

<!--屏幕保持唤醒 不锁屏 -->

<uses-permission android:name="android.permission.WAKE\_LOCK"></uses-permission>

<!--连接网络-->

<uses-permission android:name="android.permission.INTERNET" />

<application

android:allowBackup="true"

android:icon="@drawable/ic\_launcher"

android:label="@string/app\_name"

android:theme="@style/AppTheme" >

<activity

<!--设置打开进入界面的SplashActivity-->

android:name="com.naorobot.SplashActivity"

android:label="@string/app\_name" >

<intent-filter>

<action android:name="android.intent.action.MAIN" />

<category android:name="android.intent.category.LAUNCHER" />

</intent-filter>

</activity>

<!--添加MainActivity-->

<activity android:name=".MainActivity"/>

<!--添加SelectedActivity-->

<activity android:name=".SelectedActivity"/>

<!--添加SayActivity-->

<activity android:name=".SayActivity"/>

<!--添加MoveActivity-->

<activity android:name=".MoveActivity"/>

<!--添加VisionActivity-->

<activity android:name=".VisionActivity"/>

<!--添加J*oint*Activity-->

<activity android:name=".J*oint*Activity/>

</application>

</manifest>

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/

Layout文件夹下

activity\_main.xml界面文件配置

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:background="#e7e7e7" <!--背景颜色-->

android:orientation="vertical" >

<ImageView

android:layout\_width="132dp"

android:layout\_height="132dp"

android:layout\_gravity="center" <!--设置居中-->

android:layout\_marginTop="68dp"

android:src="@drawable/icon\_avatar\_login" />

<LinearLayout

android:id="@+id/login\_linearLayout"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_gravity="center"

android:layout\_marginLeft="15dp"

android:layout\_marginRight="15dp"

android:layout\_marginTop="8dp"

android:orientation="vertical" >

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:background="@drawable/login\_input" <!--设置界面图片->

android:orientation="vertical" >

<LinearLayout

android:id="@+id/userId\_LinearLayout"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:gravity="right"

android:orientation="horizontal" >

<!-- android:ems="10"表示EditText内最多只显示10个字符，超出不显示 -->

<!-- android:singleLine="true"表示不能全部显示时，后面用“…”来表示 -->

<EditText

android:id="@+id/EditText\_ip"

android:layout\_width="0dp"

android:layout\_height="44dp"

android:layout\_weight="1"

android:background="@null"

android:ems="10"

android:hint="@string/ip" <!--设置IP文字-->

android:paddingLeft="15dp"

android:paddingRight="15dp"

android:singleLine="true"

android:textColorHint="#999999"<!--设置文字属性-->

android:textSize="18sp"

>

<requestFocus />

</EditText>

</LinearLayout>

</LinearLayout>

<Button

android:id="@+id/login\_btnLogin"

android:layout\_width="match\_parent"

android:layout\_height="45dp"

android:layout\_marginTop="20dp"

android:background="@drawable/login\_button\_style"

android:text="@string/connet" <!--设置显示字符-->

android:textColor="#FFFFFF" <!--设置显示颜色-->

android:textSize="19sp"

android:onClick="onClickConnect"/> <!--触发函数 onClickConnect-->

</LinearLayout>

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="0dp"

android:layout\_weight="1"

android:orientation="vertical" />

</LinearLayout>

/layout下

main.xml 界面文件配置

main.xml<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:orientation="vertical"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:background="#eae7e7"> <!--背景颜色-->

<LinearLayout

android:layout\_marginTop="120dp"

android:layout\_gravity="center" <!--界面居中-->

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content">

<Button

android:id="@+id/say"

android:text="@string/speak" <!--设置显示字符-->

android:layout\_marginRight="1dp"

android:layout\_width="100dp"

android:layout\_height="100dp"

android:background="@drawable/slt" <!--设置背景文件-->

android:onClick="onSay"/> <!--触发函数onSay-->

<Button

android:text="@string/action"

android:id="@+id/move"

android:layout\_marginRight="1dp"

android:layout\_width="100dp"

android:layout\_height="100dp"

android:background="@drawable/slt" <!--设置背景文件-->

android:onClick="onMove"/> <!--触发函数onSay-->

</LinearLayout>

<LinearLayout

android:layout\_marginTop="1dp"

android:layout\_gravity="center" <!--界面居中-->

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content">

<Button

android:layout\_marginRight="1dp"

android:text="@string/eye" <!--设置显示字符-->

android:layout\_width="100dp"

android:layout\_height="100dp"

android:background="@drawable/slt"/> <!--设置背景文件-->

<Button

android:layout\_marginRight="1dp"

android:text="@string/joint" <!--设置显示字符-->

android:layout\_width="100dp"

android:layout\_height="100dp"

android:background="@drawable/slt"/> <!--设置背景文件-->

</LinearLayout>

</LinearLayout>

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/layout文件下

move.xml 主函数

<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:orientation="vertical"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent">

<LinearLayout

android:layout\_marginTop="5dp"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:orientation="horizontal">

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="速度："/> <!--设置显示文字-->

<SeekBar

android:id="@+id/seekbar"

android:layout\_width="300dp"

android:layout\_height="wrap\_content" />

</LinearLayout>

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:orientation="vertical">

<Button

android:id="@+id/Toward"

android:layout\_marginTop="80dp"

android:layout\_width="80dp"

android:layout\_height="80dp"

android:background="@drawable/btn\_zhixing"<!--设置背景按钮图片-->

android:layout\_gravity="center" <!--居中-->

android:onClick="onMove"/> <!--触发函数onMove-->

</LinearLayout>

<LinearLayout

android:layout\_marginTop="15dp"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:orientation="horizontal">

<Button

android:id="@+id/turnLeft"

android:layout\_marginLeft="5dp"

android:layout\_width="80dp"

android:layout\_height="80dp"

android:background="@drawable/btn\_left" <!--设置背景按钮图片-->

android:onClick="onMove"/> <!--触发函数onMove-->

<Button

android:id="@+id/pause"

android:layout\_marginLeft="36dp"

android:layout\_width="80dp"

android:layout\_height="80dp"

android:background="@drawable/pausebtn" <!--设置背景按钮图片-->

android:onClick="onMove"/> <!--触发函数onMove-->

<Button

android:id="@+id/turnRight"

android:layout\_marginLeft="25dp"

android:layout\_width="80dp"

android:layout\_height="80dp"

android:background="@drawable/btn\_right" <!--设置背景按钮图片-->

android:onClick="onMove" <!--触发函数onMove-->

/>

</LinearLayout>

</LinearLayout>

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/layout文件下

splash.xml 界面布局文件

<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android" xmlns:tools="http://schemas.android.com/tools"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

tools:context=".SplashActivity" >

<ImageView

android:background="@drawable/loading" <!--设置背景按钮图片-->

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:scaleType="fitCenter" /> <!--图片属性-->

</RelativeLayout>

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/layout文件下

joint.xml 界面布局文件

<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:tools="http://schemas.android.com/tools"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="vertical"

android:paddingBottom="@dimen/activity\_vertical\_margin"

android:paddingLeft="@dimen/activity\_horizontal\_margin"

android:paddingRight="@dimen/activity\_horizontal\_margin"

android:paddingTop="@dimen/activity\_vertical\_margin"

tools:context=".jointActivity" >

<!--头部摆动布局-->

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:orientation="vertical" >

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_marginTop="8dp"

android:orientation="horizontal" >

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_marginRight="8dp"

android:text="@string/label\_head\_yaw" /> <!--设置文字-->

<TextView

android:id="@+id/TextView\_joint\_0"

android:layout\_width="0dp"

android:layout\_height="wrap\_content"

android:layout\_weight="1" />

</LinearLayout>

<SeekBar

android:id="@+id/SeekBar\_joint\_0"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content" />

<!--头部上下抬起布局-->

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_marginTop="8dp"

android:orientation="horizontal" >

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_marginRight="8dp"

android:text="@string/label\_head\_pitch" /> <!--设置文字-- >

<TextView

android:id="@+id/TextView\_joint\_1"

android:layout\_width="0dp"

android:layout\_height="wrap\_content"

android:layout\_weight="1" />

</LinearLayout>

<SeekBar

android:id="@+id/SeekBar\_joint\_1"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content" />

<!--左肩斜度布局-->

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_marginTop="8dp"

android:orientation="horizontal" >

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_marginRight="8dp"

android:text="@string/label\_l\_shoulder\_pitch" /> <!--设置文字-- >

<TextView

android:id="@+id/TextView\_joint\_2"

android:layout\_width="0dp"

android:layout\_height="wrap\_content"

android:layout\_weight="1" />

</LinearLayout>

<SeekBar

android:id="@+id/SeekBar\_joint\_2"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content" />

<!--左肩转动布局-->

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_marginTop="8dp"

android:orientation="horizontal" >

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_marginRight="8dp"

android:text="@string/label\_l\_shoulder\_roll" /> <!--设置文字-- >

<TextView

android:id="@+id/TextView\_joint\_3"

android:layout\_width="0dp"

android:layout\_height="wrap\_content"

android:layout\_weight="1" />

</LinearLayout>

<SeekBar

android:id="@+id/SeekBar\_joint\_3"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content" />

<!--左肘部滚动布局-->

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_marginTop="8dp"

android:orientation="horizontal" >

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_marginRight="8dp"

android:text="@string/label\_l\_elbow\_yaw" /> <!--设置文字-- >

<TextView

android:id="@+id/TextView\_joint\_4"

android:layout\_width="0dp"

android:layout\_height="wrap\_content"

android:layout\_weight="1" />

</LinearLayout>

<SeekBar

android:id="@+id/SeekBar\_joint\_4"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content" />

<!--左腕横摆布局-->

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_marginTop="8dp"

android:orientation="horizontal" >

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_marginRight="8dp"

android:text="@string/label\_l\_elbow\_roll" /> <!--设置文字-- >

<TextView

android:id="@+id/TextView\_joint\_5"

android:layout\_width="0dp"

android:layout\_height="wrap\_content"

android:layout\_weight="1" />

</LinearLayout>

<SeekBar

android:id="@+id/SeekBar\_joint\_5"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content" />

<!--右肩斜度布局-->

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_marginTop="8dp"

android:orientation="horizontal" >

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_marginRight="8dp"

android:text="@string/label\_l\_wrist\_yaw" /> <!--设置文字-- >

<TextView

android:id="@+id/TextView\_joint\_6"

android:layout\_width="0dp"

android:layout\_height="wrap\_content"

android:layout\_weight="1" />

</LinearLayout>

<SeekBar

android:id="@+id/SeekBar\_joint\_6"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content" />

<!--右肩转动布局-->

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_marginTop="8dp"

android:orientation="horizontal" >

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_marginRight="8dp"

android:text="@string/label\_r\_shoulder\_pitch" /> <!--设置文字-- >

<TextView

android:id="@+id/TextView\_joint\_7"

android:layout\_width="0dp"

android:layout\_height="wrap\_content"

android:layout\_weight="1" />

</LinearLayout>

<SeekBar

android:id="@+id/SeekBar\_joint\_7"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content" />

<!--右手横摆布局-->

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_marginTop="8dp"

android:orientation="horizontal" >

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_marginRight="8dp"

android:text="@string/label\_r\_shoulder\_roll" /> <!--设置文字-- >

<TextView

android:id="@+id/TextView\_joint\_8"

android:layout\_width="0dp"

android:layout\_height="wrap\_content"

android:layout\_weight="1" />

</LinearLayout>

<SeekBar

android:id="@+id/SeekBar\_joint\_8"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content" />

<!--右肘部滚动布局-->

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_marginTop="8dp"

android:orientation="horizontal" >

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_marginRight="8dp"

android:text="@string/label\_r\_elbow\_yaw" />

<TextView

android:id="@+id/TextView\_joint\_9"

android:layout\_width="0dp"

android:layout\_height="wrap\_content"

android:layout\_weight="1" />

</LinearLayout>

<SeekBar

android:id="@+id/SeekBar\_joint\_9"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content" />

<!--右肘部滚动布局-->

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_marginTop="8dp"

android:orientation="horizontal" >

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_marginRight="8dp"

android:text="@string/label\_r\_elbow\_roll" />

<TextView

android:id="@+id/TextView\_joint\_10"

android:layout\_width="0dp"

android:layout\_height="wrap\_content"

android:layout\_weight="1" />

</LinearLayout>

<SeekBar

android:id="@+id/SeekBar\_joint\_10"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content" />

<!--右腕横摆布局-->

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_marginTop="8dp"

android:orientation="horizontal" >

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_marginRight="8dp"

android:text="@string/label\_r\_wrist\_yaw" /> <!--设置文字-- >

<TextView

android:id="@+id/TextView\_joint\_11"

android:layout\_width="0dp"

android:layout\_height="wrap\_content"

android:layout\_weight="1" />

</LinearLayout>

<SeekBar

android:id="@+id/SeekBar\_joint\_11"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content" />

</LinearLayout>

</LinearLayout>

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/layout文件

vision.xml 界面布局文件

<?xml version="1.0" encoding="utf-8"?>

<LinearLayout

xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:tools="http://schemas.android.com/tools"

android:orientation="vertical"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:paddingLeft="@dimen/activity\_horizontal\_margin"

android:paddingRight="@dimen/activity\_horizontal\_margin"

android:paddingTop="@dimen/activity\_vertical\_margin"

android:paddingBottom="@dimen/activity\_vertical\_margin"

tools:context=".visionActivity">

<Button

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:text="@string/get\_Camera"

android:onClick="onClickGetImageRemote" /><!--触发函数onClickGetImageRemote-->

<!--图片资源获取-->

<ImageView

android:id="@+id/ImageView\_photo"

android:layout\_width="match\_parent"

android:layout\_height="0dp"

android:layout\_weight="1"

android:layout\_margin="8dp"

android:contentDescription="TODO"/>

</LinearLayout>

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/layout文件下

speak.xml界面布局文件

<?xml version="1.0" encoding="utf-8"?>

<LinearLayout

xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:tools="http://schemas.android.com/tools"

android:orientation="vertical"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:paddingLeft="@dimen/activity\_horizontal\_margin"

android:paddingRight="@dimen/activity\_horizontal\_margin"

android:paddingTop="@dimen/activity\_vertical\_margin"

android:paddingBottom="@dimen/activity\_vertical\_margin"

tools:context=".speakActivity">

<LinearLayout

android:layout\_width="match\_parent"

android:orientation="horizontal"

android:weightSum="4"

android:layout\_height="wrap\_content">

<EditText

android:id="@+id/EditText\_ip"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:hint="@string/input\_speak" />

</LinearLayout>

<Button

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:text="@string/speak"

android:onClick="onClickSpeak" /><!--触发函数onClickSpeak-- >

</LinearLayout>

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/drawable资源文件

btn\_left.xml 按钮属性配置

<!--向左按钮布局-- >

<?xml version="1.0" encoding="utf-8"?>

<selector xmlns:android="http://schemas.android.com/apk/res/android">

<item android:drawable="@drawable/turn\_left\_press"

android:state\_pressed="true"/>

<item android:drawable="@drawable/turn\_left"

android:state\_pressed="false"/>

</selector>

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/drawable资源文件

btn\_right.xml按钮属性配置

<!--向右按钮布局-- >

<?xml version="1.0" encoding="utf-8"?>

<selector xmlns:android="http://schemas.android.com/apk/res/android">

<item android:drawable="@drawable/turn\_right\_press"

android:state\_pressed="true"/>

<item android:drawable="@drawable/turn\_right"

android:state\_pressed="false"/>

</selector>

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/drawable资源文件

btn\_zhixing.xml按钮属性配置

<!--直行按钮布局-- >

<?xml version="1.0" encoding="utf-8"?>

<selector xmlns:android="http://schemas.android.com/apk/res/android">

<item android:drawable="@drawable/turn\_right\_press"

android:state\_pressed="true"/>

<item android:drawable="@drawable/turn\_right"

android:state\_pressed="false"/>

</selector>

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/drawable资源文件

pausebtn.xml 按钮属性配置

<!--暂停按钮布局-- >

<?xml version="1.0" encoding="utf-8"?>

<selector xmlns:android="http://schemas.android.com/apk/res/android">

<item android:drawable="@drawable/pause" android:state\_pressed="false"/>

<item android:drawable="@drawable/pause\_press" android:state\_pressed="true"/>

</selector>

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/values 文件下

string.xml 字符存储文件

<?xml version="1.0" encoding="utf-8"?>

<resources>

<string name="app\_name">nao机器人控制</string>

<string name="action\_settings">Settings</string>

<string name="hello\_world">Hello world!</string>

<!-- activity\_main -->

<string name="edittext\_ip\_hint">IP Address</string>

<string name="button\_connnect">Connnect</string>

<string name="label\_head\_yaw">头部摆动</string>

<string name="label\_head\_pitch">头部上下抬起</string>

<string name="label\_l\_shoulder\_pitch">左肩斜度</string>

<string name="label\_l\_shoulder\_roll">左肩转动</string>

<string name="label\_l\_elbow\_yaw">左手横摆</string>

<string name="label\_l\_elbow\_roll">左肘部滚动</string>

<string name="label\_l\_wrist\_yaw">左腕横摆</string>

<string name="label\_r\_shoulder\_pitch">右肩斜度</string>

<string name="label\_r\_shoulder\_roll">右肩转动</string>

<string name="label\_r\_elbow\_yaw">右手横摆</string>

<string name="label\_r\_elbow\_roll">右肘部滚动</string>

<string name="label\_r\_wrist\_yaw">右腕横摆</string>

<string name="speak">说</string>

<string name="action">动作</string>

<string name="eye">视觉</string>

<string name="joint">关节控制</string>

<string name="get\_Camera">获取摄像头</string>

<string name="input\_speak">输入说的话</string>

<string name="ip">Ip地址：</string>

<string name="connet">连接</string>

</resources>

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/ src 文件夹下

MainActivity.java 字符存储文件

package com.naorobot;

import java.io.File;

import android.app.Activity;

import android.content.Intent;

import android.os.Bundle;

import android.os.Looper;

import android.view.View;

import android.view.Window;

import android.widget.EditText;

import android.widget.Toast;

import com.aldebaran.qi.EmbeddedTools;

import com.aldebaran.qi.Session;

import com.naorobot.MainActivity;

import com.naorobot.R;

import com.naorobot.Statictext;

public class MainActivity extends Activity {

private EditText mEditTextIp;

@Override

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :onCreate \*/

/\*Descrp :初始化 \*/

/\*Input :Bundle. \*/

/\*Output :None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

requestWindowFeature(Window.FEATURE\_NO\_TITLE);//设置全屏

setContentView(R.layout.activity\_main);//加载界面activity\_main

mEditTextIp = (EditText) findViewById(R.id.EditText\_ip);//获取控件

//初始化 Embedded Tools

EmbeddedTools tools = new EmbeddedTools();

File dir = getApplicationContext().getCacheDir();

tools.overrideTempDirectory(dir);

tools.loadEmbeddedLibraries();

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :onCreate \*/

/\*Descrp : 按钮接口触发函数 \*/

/\*Input :View. \*/

/\*Output :None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

public void onClickConnect(View view) {

Thread thread = new Thread(new Runnable() {

@Override

public void run() {

Looper.prepare();//连接IP

Statictext.mQiSession = new Session();//创建连接管道

Statictext.Ip = "tcp://" + mEditTextIp.getText().toString()

+ ":9559";//ip 地址

try {

Statictext.mQiSession.connect(Statictext.Ip).get();//连接ip

} catch (Exception e) {

toastOnUiThread(e, "连接失败");

e.printStackTrace();

Statictext.mQiSession = null;

return;

}

toastOnUiThread("连接成功");

Intent intent = new Intent();

intent.setClass(MainActivity.this, SelectedActivity.class);//跳转到

//selectedActivity

MainActivity.this.startActivity(intent);

MainActivity.this.finish();

}

});

thread.start();

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :toastOnUiThread \*/

/\*Descrp : toast 提示信息 \*/

/\*Input :Exception e, String str \*/

/\*Output :None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

private void toastOnUiThread(Exception e, String str) {

toastOnUiThread(str + "\n" + e.getMessage());

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :toast\_long \*/

/\*Descrp : toast 弹出时间 \*/

/\*Input :String str \*/

/\*Output :None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

private void toast\_long(String str) {

Toast.makeText(this, str, Toast.LENGTH\_LONG).show();

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :toastOnUiThread \*/

/\*Descrp :toast弹出 线程开启 \*/

/\*Input :String str \*/

/\*Output :None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

private void toastOnUiThread(String str) {

final String msg = str;

runOnUiThread(new Runnable() {//开启线程

@Override

public void run() {

toast\_long(msg);

}

});

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/src 文件下

SplashActivity.java 打开软件画面

package com.naorobot;

import android.app.Activity;

import android.content.Intent;

import android.os.Bundle;

import android.os.Handler;

import android.view.Window;

import android.view.WindowManager;

public class SplashActivity extends Activity {

private static final long SPLASH\_DELAY\_MILLIS = 3000;

@Override

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :onCreate \*/

/\*Descrp :初始化 \*/

/\*Input :Bundle. \*/

/\*Output :None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

// /getSupportActionBar().hide();

requestWindowFeature(Window.FEATURE\_NO\_TITLE);

getWindow().setFlags(WindowManager.LayoutParams.FLAG\_FULLSCREEN,

WindowManager.LayoutParams.FLAG\_FULLSCREEN);

setContentView(R.layout.splash);

new Handler().postDelayed(new Runnable() {

public void run() {

Intent intent = new Intent(SplashActivity.this,

MainActivity.class);

//

startActivity(intent);

SplashActivity.this.finish();

}

}, SPLASH\_DELAY\_MILLIS);

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/src 文件夹下

SelectedActivity.java 打开选择控制界面

package com.naorobot;

import com.naorobot.R;

import android.app.Activity;

import android.content.Intent;

import android.os.Bundle;

import android.os.Handler;

import android.view.View;

public class SelectedActivity extends Activity {

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.main);

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :onMove \*/

/\*Descrp : 按钮接口触发函数跳转到移动控制界面 \*/

/\*Input :View. \*/

/\*Output :None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

public void onMove(View view) {

new Handler().postDelayed(new Runnable() {

public void run() {

Intent intent = new Intent(SelectedActivity.this,

MoveActivity.class);

SelectedActivity.this.startActivity(intent);

SelectedActivity.this.finish();

}

}, 1000);

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :onSay \*/

/\*Descrp : 按钮接口触发函数跳转到说话控制界面 \*/

/\*Input :View. \*/

/\*Output :None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

public void onSay(iew view) {

new Handler().postDelayed(new Runnable() {

public void run() {

Intent intent = new Intent(SelectedActivity.this,

SayActivity.class);

//

SelectedActivity.this.startActivity(intent);

SelectedActivity.this.finish();

}

}, 1000);

}

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :*onvision* \*/

/\*Descrp : 按钮接口触发函数跳转到获取摄像头界面 \*/

/\*Input :View. \*/

/\*Output :None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

public void *onvision*(View view) {

new Handler().postDelayed(new Runnable() {

public void run() {

Intent intent = new Intent(SelectedActivity.this,

V*ision*Activity.class);

//

SelectedActivity.this.startActivity(intent);

SelectedActivity.this.finish();

}

}, 1000);

}

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :*onjoint* \*/

/\*Descrp : 按钮接口触发函数跳转到关节控制界面 \*/

/\*Input :View. \*/

/\*Output :None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

public void *onvision*(View view) {

new Handler().postDelayed(new Runnable() {

public void run() {

Intent intent = new Intent(SelectedActivity.this,

J*oint*Activity.class);

//

SelectedActivity.this.startActivity(intent);

SelectedActivity.this.finish();

}

}, 1000);

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/src文件夹下

SayActivity.java 说话界面控制代码

import java.io.File;

import com.aldebaran.qi.EmbeddedTools;

import com.aldebaran.qi.helper.proxies.ALTextToSpeech;

import android.app.Activity;

import android.os.Bundle;

import android.os.Looper;

import android.view.View;

import android.widget.EditText;

import android.widget.Toast;

public class SayActivity extends Activity {

EditText editText;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.speak);

EmbeddedTools tools = new EmbeddedTools();

File dir = getApplicationContext().getCacheDir();

tools.overrideTempDirectory(dir);

tools.loadEmbeddedLibraries();

editText=(EditText)findViewById(R.id.saytext);

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :onClickSay \*/

/\*Descrp : 按钮接口触发函数 开启线程让nao说话 \*/

/\*Input :View. \*/

/\*Output :None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

public void onClickSay(View view){

if ( Statictext.mQiSession == null ) {

Toast.makeText(this, "失败", Toast.LENGTH\_LONG).show();//如果失败提示toast

return;

}

Thread thread = new Thread( new Runnable() {

@Override

public void run() {

Looper.prepare();//初始化准备

robotSay();

}

});

thread.start();

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :robotSay \*/

/\*Descrp : 将要说的字符传给nao让它执行 \*/

/\*Input :None. \*/

/\*Output :None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

private void robotSay() {

try {

ALTextToSpeech tts = new ALTextToSpeech( Statictext.mQiSession );

tts.say(editText.getText().toString());

} catch (Exception e) {

e.printStackTrace();

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/src 文件夹下

MoveActivity.java 控制机器人移动

package com.naorobot;

import android.app.Activity;

import android.os.Bundle;

import android.view.View;

import android.widget.SeekBar;

import android.widget.Toast;

import com.naorobot.R;

import com.aldebaran.qi.CallError;

import com.aldebaran.qi.helper.proxies.ALMotion;

public class MoveActivity extends Activity implements SeekBar.OnSeekBarChangeListener {

public SeekBar seb;//控制速度

private static ALMotion mALMotion;//移动函数公共类

public static float V\_s=0; //当前速度

@Override

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :onCreate \*/

/\*Descrp :初始化 \*/

/\*Input :Bundle. \*/

/\*Output :None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.move);

seb=(SeekBar)findViewById(R.id.seekbar);

seb.setMax(1);

seb.setProgress(1);

seb.setOnSeekBarChangeListener(this);

try {

mALMotion = new ALMotion(Statictext.mQiSession);//创建一个移动线程

} catch (Exception e) {

e.printStackTrace();

}

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :onClickSay \*/

/\*Descrp : 按钮接口触发函数 控制nao机器人的移动 \*/

/\*Input :View. \*/

/\*Output :None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

public void onMove(View v) {

boolean toward=false; //前进控制

boolean left=false;//左转控制

boolean right=false;//右转控制

float x = 0f;

float y = 0f;

float theta = 0.0f;

switch (v.getId()) {

case R.id.turnLeft:

toward=true;

break;

case R.id.turnRight:

left=true;

break;

case R.id.Toward:

right=true;

break;

case R.id.pause:

robotStopMove();//停止函数

break;

}

if(toward){

x=V\_s;

}else if(left)

{

theta=V\_s;

}else if(right)

{

theta=-V\_s;

}

try {

mALMotion.moveToward(x, y, theta);//将参数属性传入移动控制函数中

} catch (CallError callError) {

callError.printStackTrace();

} catch (InterruptedException e) {

e.printStackTrace();

}

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :robotStopMove \*/

/\*Descrp : 按钮接口触发函数 停止机器人移动 \*/

/\*Input :None. \*/

/\*Output :None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

private void robotStopMove() {

try {

MoveActivity.mALMotion.stopMove();//停止函数

} catch (Exception e) {

toastOnUiThread(e);

e.printStackTrace();

}

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :toastOnUiThread \*/

/\*Descrp : toast ui 线程 \*/

/\*Input:Exception e \*/

/\*Output :None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

private void toastOnUiThread(Exception e) {

toastOnUiThread(e.getMessage());

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :toastOnUiThread \*/

/\*Descrp : toast ui 线程 \*/

/\*Input:String str \*/

/\*Output :None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

private void toastOnUiThread(String str) {

final String msg = str;

runOnUiThread(new Runnable() {

@Override

public void run() {

toast\_short(msg);

}

});

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :toast\_short \*/

/\*Descrp : toast 弹出时间 \*/

/\*Input:Exception e \*/

/\*Output :None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

private void toast\_short(String str) {

Toast.makeText(this, str, Toast.LENGTH\_SHORT).show();

}

@Override

public void onProgressChanged(SeekBar seekBar, int progress, boolean fromUser) {

}

@Override

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :onStartTrackingTouch \*/

/\*Descrp 滑动条触摸函数 \*/

/\*Input:SeekBar seekBar \*/

/\*Output :None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

public void onStartTrackingTouch(SeekBar seekBar) {

V\_s=seekBar.getProgress();

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/src文件夹下

JointActivity.java 手关节控制

package com.naorobot;

import android.app.Activity;

import android.content.SharedPreferences;

import android.content.res.Resources;

import android.os.Bundle;

import android.os.Looper;

import android.preference.PreferenceManager;

import android.util.Log;

import android.view.View;

import android.widget.EditText;

import android.widget.SeekBar;

import android.widget.TextView;

import android.widget.Toast;

import com.aldebaran.qi.EmbeddedTools;

import com.aldebaran.qi.Session;

import com.aldebaran.qi.helper.proxies.\*;

import java.io.File;

import java.util.ArrayList;

import java.util.List;

public class JointActivity extends Activity {

// 调试

private static final String TAG = "NAOJoint";

private static final boolean D = true;

// 字符

private static final String LF = "\n";

private static final String COMMA\_SPACE = ", ";

//参数

private static final float DEG\_TO\_RAD = (float)Math.PI / 180f ;

private static final float RAD\_TO\_DEG = 180f / (float)Math.PI ;

private static final int RES\_TOAST = 1;

private static final int RES\_JOINT\_INFO = 2;

// 关节相关参数

private static final int NUM\_JOINT = 15;

private static final int NUM\_VIEW = 15;

private static final String[] JOINT\_NAMES =

{ "HeadYaw", "HeadPitch",

"LShoulderPitch", "LShoulderRoll", "LElbowYaw", "LElbowRoll", "LWristYaw",

"RShoulderPitch", "RShoulderRoll", "RElbowYaw", "RElbowRoll", "RWristYaw",

"HipRoll", "HipPitch", "KneePitch",

"LHand", "RHand" };

//参数数组

private static final float[][] JOINT\_VALEES =

{

// HeadYaw

{-2.0856686f, 2.0856686f, 0f},

// HeadPitch

{-0.70685834f, 0.6370452f, 0f},

// LShoulderPitch

{-2.0856686f, 2.0856686f, 0f},

// LShoulderRoll 8deg

{0.008726646f, 1.5620697f, 0.14f},

// LElbowYaw

{-2.0856686f, 2.0856686f, 0f},

// LElbowRoll 30deg

{-1.5620697f, -0.008726646f, -0.523f},

// LWristYaw

{-1.8238691f, 1.8238691f, 0f},

// RShoulderPitch

{-2.0856686f, 2.0856686f, 0f},

// RShoulderRoll -8deg

{-1.5620697f, -0.008726646f, -0.14f},

// RElbowYaw

{-2.0856686f, 2.0856686f, 0f},

// RElbowRoll 30deg

{0.008726646f, 1.5620697f, 0.523f},

// RWristYaw

{-1.8238691f, 1.8238691f, 0f},

// HipRoll

{-0.51487213f, 0.51487213f, 0f},

// HipPitch

{-1.0384709f, 1.0384709f, 0f},

// KneePitch

{-0.51487213f, 0.51487213f, 0f}

};

private static final String BODY\_NAME\_JOINT = "Body";

private static final boolean USE\_SENSORS = true;

private static final float STIFFNESSE = 1.0f;

private static final float SPEED = 0.2f;

private static final float ANGLE\_ERROR = 10f;

//控件数组

private TextView[] mTextViewJoints = new TextView[NUM\_VIEW];

private SeekBar[] mSeekBarJoints = new SeekBar[NUM\_VIEW];

// object

private SharedPreferences mPreferences;

private Resources mResources;

// nao API

private ALMotion mALMotion;

//相关参数全局变量初始化

private String mAddr = "";

private int mJointNum = 0;

private boolean[] isJointStiffnesses = new boolean[NUM\_JOINT];

private String mToast = "";

private String mJointInfo = "";

// 弧度

private float[] mJointAngles = new float[NUM\_JOINT];

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :onCreate \*/

/\*Descrp :初始化 \*/

/\*Input :Bundle. \*/

/\*Output :None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.joint);

initrobot();

mPreferences = PreferenceManager.getDefaultSharedPreferences(this);

mResources = getResources();

initViewJoints();

// Embedded 工具初始化

EmbeddedTools tools = new EmbeddedTools();

File dir = getApplicationContext().getCacheDir(););

tools.overrideTempDirectory(dir);

tools.loadEmbeddedLibraries();

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :initViewJoints \*/

/\*Descrp :界面控件初始化 \*/

/\*Input :None. \*/

/\*Output :None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

private void initViewJoints() {

String pkg = getPackageName();

for ( int i = 0; i < NUM\_VIEW; i++ ) {

initViewJoint( i, pkg );

}

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :initViewJoints \*/

/\*Descrp :界面控件初始化 \*/

/\*Input :int n, String pkg. \*/

/\*Output :None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

private void initViewJoint( int n, String pkg ) {

float joint\_min = JOINT\_VALEES[n][0];

float joint\_max = JOINT\_VALEES[n][1];

float joint\_default = JOINT\_VALEES[n][2];

int max\_deg = (int)( RAD\_TO\_DEG \* ( joint\_max - joint\_min ));

int progress\_deg = (int)( RAD\_TO\_DEG \* ( joint\_default - joint\_min ));

float angle\_rad = joint\_default;

log\_d( "initViewJoint " + n + " " +max\_deg + " " + progress\_deg + " " + angle\_rad );

mJointAngles[n] = angle\_rad;

isJointStiffnesses[n] = false;

// TextView

String name\_text = "TextView\_joint\_" + n ;

int id\_text = mResources.getIdentifier( name\_text, "id", pkg );

mTextViewJoints[n] = (TextView) findViewById( id\_text );

setTextViewJoint( n, angle\_rad );

// SeekBar

String name\_seek = "SeekBar\_joint\_" + n ;

int id\_seek = mResources.getIdentifier( name\_seek, "id", pkg );

mSeekBarJoints[n] = (SeekBar) findViewById( id\_seek );

mSeekBarJoints[n].setMax( max\_deg );

mSeekBarJoints[n].setProgress( progress\_deg );

mSeekBarJoints[n].setTag( n );

//控件响应监听函数

mSeekBarJoints[n].setOnSeekBarChangeListener(

new SeekBar.OnSeekBarChangeListener() {

@Override

//控件响应监听函数

public void onStartTrackingTouch( SeekBar seek ) {

// noting to do

}

@Override

//控件响应监听函数

public void onProgressChanged( SeekBar seek, int progress, boolean touch ) {

procProgressChanged( seek, progress );

}

@Override

public void onStopTrackingTouch( SeekBar seek ) {

procStopTrackingTouch( seek );

}

});

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :procProgressChanged \*/

/\*Descrp :SeekBar 控件数值改变 \*/

/\*Input :SeekBar seek, int progress \*/

/\*Output :None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

private void procProgressChanged( SeekBar seek, int progress ) {

int tag = (int)seek.getTag();

float min\_seek = JOINT\_VALEES[tag][0];

// margin 1 degree

float min = min\_seek + DEG\_TO\_RAD;

float max = JOINT\_VALEES[tag][1] - DEG\_TO\_RAD;

float angle\_rad = DEG\_TO\_RAD \* (float)progress + min\_seek;

if ( angle\_rad < min ) {

angle\_rad = min;

}

if ( angle\_rad > max ) {

angle\_rad = max;

}

mJointAngles[tag] = angle\_rad;

setTextViewJoint( tag, angle\_rad );

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :procStopTrackingTouch \*/

/\*Descrp :SeekBar控件响应结束后开启命令执行线程 \*/

/\*Input :SeekBar seek \*/

/\*Output :None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

private void procStopTrackingTouch( SeekBar seek ) {

int tag = (int)seek.getTag();

mJointNum = tag;

startThread();

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :setTextViewJoint \*/

/\*Descrp :将当前角度显示在界面上 \*/

/\*Input :int n, float angle\_rad \*/

/\*Output :None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

private void setTextViewJoint( int n, float angle\_rad ) {

int angle\_deg = (int)( RAD\_TO\_DEG \* angle\_rad );

log\_d( "setTextViewJoint " + n + " " + angle\_rad + " " + angle\_deg );

mTextViewJoints[n].setText( Integer.toString(angle\_deg) );

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :showInfo \*/

/\*Descrp :显示信息 \*/

/\*Input :None. \*/

/\*Output :None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

private void showInfo() {

int angle\_deg = 0;

for ( int i = 0; i < NUM\_VIEW; i++ ) {

angle\_deg = (int)( RAD\_TO\_DEG \* mJointAngles[i] );

mTextViewJoints[i].setText( Integer.toString(angle\_deg) );

}

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :startThread \*/

/\*Descrp :开启控制线程 \*/

/\*Input :None. \*/

/\*Output :None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

private void startThread() {

Thread thread = new Thread( new Runnable() {

@Override

public void run() {

Looper.prepare();

robotSetAngle();

}

});

thread.start();

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :initrobot \*/

/\*Descrp :初始化参数通信管道 \*/

/\*Input :None. \*/

/\*Output :None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

private boolean initrobot( ) {

try {

mALMotion = new ALMotion(Statictext.mQiSession );

} catch (Exception e) {

if (D) e.printStackTrace();

}

List<String> list\_body = robotGetBodyNames( BODY\_NAME\_JOINT );

mJointInfo = debugBodyNames( BODY\_NAME\_JOINT, list\_body ) + LF;

mJointInfo += robotGetLimitsAngles();

procOnUiThread( RES\_JOINT\_INFO );

mAddr = Statictext.Ip;

startThread();//开区控制线程

return true;

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :robotGetLimitsAngles \*/

/\*Descrp :获取角度信息 \*/

/\*Input :None. \*/

/\*Output : String . \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

private String robotGetLimitsAngles() {

List<List<Float>> list\_list\_limits = null;

List<Float> list\_angle = null;

String name = "";

float angle\_rad = 0;

String msg = "";

for ( int i = 0; i < NUM\_JOINT; i++ ) {

name = JOINT\_NAMES[i];

list\_list\_limits = robotGetLimits( name );

list\_angle = robotGetAngles( name, USE\_SENSORS );//将角度信息传给nao

angle\_rad = convertAngle( list\_angle );

setJointAngle( i, angle\_rad );//设置关节角度

msg += debugLimits( name, list\_list\_limits ) + LF;

msg += debugAngle( name, angle\_rad ) + LF;

}

return msg;

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :robotSetAngle \*/

/\*Descrp :设置关节角度 \*/

/\*Input :None. \*/

/\*Output : None . \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

private void robotSetAngle() {

String name = JOINT\_NAMES[ mJointNum ];

float angle\_rad = mJointAngles[ mJointNum ];

if ( !isJointStiffnesses[ mJointNum ] ) {

isJointStiffnesses[ mJointNum ] = true;

robotSetStiffness( name, STIFFNESSE );

}

robotSetAngleInterpolationWithSpeed( name, angle\_rad, SPEED );//设置关节角度执行的速度

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :robotSetStiffness \*/

/\*Descrp :机器人状态设置 \*/

/\*Input : String name, float stiffnesse. \*/

/\*Output : None . \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

private void robotSetStiffness( String name, float stiffnesse ) {

log\_d( "robotSetStiffness " + name + " " + stiffnesse );

try {

mALMotion.setStiffnesses( name, stiffnesse );

} catch (Exception e) {

if (D) e.printStackTrace();

toastOnUiThread( e, "Motion failed");

}

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :robotSetAngleInterpolationWithSpeed \*/

/\*Descrp :设置执行属性参数 \*/

/\*Input : String name, float angle, float speed. \*/

/\*Output : None . \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

private void robotSetAngleInterpolationWithSpeed( String name, float angle, float speed ) {

log\_d("robotSetAngleInterpolationWithSpeed " + name + " " + angle + " " + speed);

try {

mALMotion.angleInterpolationWithSpeed( name, angle, speed );

} catch (Exception e) {

if (D) e.printStackTrace();

toastOnUiThread( e, "Motion failed" );

}

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :robotGetBodyNames \*/

/\*Descrp :获取关节的数组 \*/

/\*Input : String name. \*/

/\*Output : List<String>. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

private List<String> robotGetBodyNames( String name ) {

log\_d( "robotGetBodyNames " + name );

List<String> list = new ArrayList<String>();

try {

list = mALMotion.getBodyNames( name );

} catch (Exception e) {

if (D) e.printStackTrace();

toastOnUiThread( e, "Motion failed");

}

log\_d( "getBodyNames " + list );

return list;

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :robotGetLimits \*/

/\*Descrp :获取最大限制参数 \*/

/\*Input : String name. \*/

/\*Output : List<List<Float>> . \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

@SuppressWarnings("unchecked")

private List<List<Float>> robotGetLimits( String name ) {

log\_d( "robotGetLimits " + name );

List<List<Float>> list\_list = new ArrayList<List<Float>>();

try {

list\_list = (List<List<Float>>) mALMotion.getLimits( name );

} catch (Exception e) {

if (D) e.printStackTrace();

toastOnUiThread( e, "Motion failed" );

}

log\_d( "getLimits " + list\_list );

return list\_list;

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :robotGetAngles \*/

/\*Descrp :获取角度参数 \*/

/\*Input : String name, boolean useSensors. \*/

/\*Output : List<Float>. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

private List<Float> robotGetAngles( String name, boolean useSensors ) {

log\_d( "robotGetAngles " + name );

List<Float> list = new ArrayList<Float>();

try {

list = mALMotion.getAngles( name, useSensors );

} catch (Exception e) {

if (D) e.printStackTrace();

toastOnUiThread( e, "Motion failed" );

}

log\_d( "getAngles " + list );

return list;

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :robotGetAngles \*/

/\*Descrp : 转换角度 \*/

/\*Input : List<Float> list. \*/

/\*Output : float . \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

private float convertAngle( List<Float> list ) {

if (( list == null )||( list.size() == 0 )) {

log\_d( "not get robotGetAngles" );

return ( ANGLE\_ERROR + 1 );

}

return list.get(0);

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :setJointAngle \*/

/\*Descrp : 转换角度 \*/

/\*Input : int n, float angle\_rad. \*/

/\*Output : None . \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

private void setJointAngle( int n, float angle\_rad ) {

if ( angle\_rad < ANGLE\_ERROR ) {

mJointAngles[n] = angle\_rad;

}

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :debugBodyNames \*/

/\*Descrp :调试身体名字 \*/

/\*Input : int n, float angle\_rad. \*/

/\*Output : None . \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

private String debugBodyNames( String name, List<String> list ) {

String msg = "BodyNames " + name + ": ";

if (( list == null )||( list.size() == 0 )) {

log\_d( "not getBodyNames" );

return msg;

}

for ( String str: list ) {

msg += str+ COMMA\_SPACE;

}

log\_d(msg);

return msg;

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :debugLimits \*/

/\*Descrp :调试参数限度 \*/

/\*Input : String name, List<List<Float>> list\_list. \*/

/\*Output : String . \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

private String debugLimits( String name, List<List<Float>> list\_list ) {

String msg = name + " limits: ";

if (( list\_list == null )||( list\_list.size() == 0 )) {

log\_d( "not get robotGetLimits" );

return msg;

}

msg += debugListFloat( list\_list.get(0) );

log\_d( msg );

return msg;

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :debugAngle \*/

/\*Descrp :调试角度 \*/

/\*Input : String name, List<List<Float>> list\_list. \*/

/\*Output : String . \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

private String debugAngle( String name, float angle\_rad ) {

String msg = name + " angle: ";

if ( angle\_rad < ANGLE\_ERROR ) {

msg += angle\_rad;

}

log\_d( msg );

return msg;

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :debugListString \*/

/\*Descrp :调试list<string>数组. \*/

/\*Input :List<String> list. \*/

/\*Output : String . \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

private String debugListString( List<String> list ) {

if (( list == null )||( list.size() == 0 )) {

return "";

}

String msg = "";

for ( String str: list ) {

msg += str + COMMA\_SPACE;

}

return msg;

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :debugListFloat \*/

/\*Descrp :调试list<Float>数组. \*/

/\*Input :List<Float> list . \*/

/\*Output : String . \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

private String debugListFloat( List<Float> list ) {

if (( list == null )||( list.size() == 0 )) {

return "";

}

String msg = "";

for ( Float f: list ) {

msg += f + COMMA\_SPACE;

}

return msg;

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :toastOnUiThread \*/

/\*Descrp :toast界面线程. \*/

/\*Input :Exception e, String res\_id. \*/

/\*Output : None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

private void toastOnUiThread( Exception e, String res\_id) {

String str = res\_id;

toastOnUiThread( str+ LF + e.getMessage() );

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :toastOnUiThread \*/

/\*Descrp :toast界面线程. \*/

/\*Input :int res\_id. \*/

/\*Output : None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

private void toastOnUiThread( int res\_id ) {

String str = getString(res\_id);

toastOnUiThread( str );

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :toastOnUiThread \*/

/\*Descrp :toast界面线程. \*/

/\*Input : String str. \*/

/\*Output : None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

private void toastOnUiThread( String str ) {

mToast = str;

procOnUiThread( RES\_TOAST );

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :procOnUiThread \*/

/\*Descrp :ui界面线程. \*/

/\*Input : int \_res. \*/

/\*Output : None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

private void procOnUiThread( int \_res ) {

log\_d( "procOnUiThread " + \_res );

final int res = \_res;

runOnUiThread( new Runnable() {

@Override

public void run() {

procUi( res );

}

});

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :procUi \*/

/\*Descrp :ui界面选择. \*/

/\*Input : int \_res. \*/

/\*Output : None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

private void procUi( int res ) {

switch( res ) {

case RES\_TOAST:

toast\_short( mToast );

break;

case RES\_JOINT\_INFO:

showInfo();

break;

}

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :toast\_short \*/

/\*Descrp :toast时间为short. \*/

/\*Input : int \_res. \*/

/\*Output : None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

private void toast\_short( String str ) {

Toast.makeText(this, str, Toast.LENGTH\_SHORT).show();

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :log\_d \*/

/\*Descrp :log. \*/

/\*Input : String str. \*/

/\*Output : None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

private void log\_d( String str ) {

if (D) Log.d( TAG, str );

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/src文件夹下

FileUtility.java 文件操作

package com.naorobot;

import android.graphics.Bitmap;

import android.os.Environment;

import android.util.Log;

import java.io.BufferedOutputStream;

import java.io.File;

import java.io.FileOutputStream;

import java.text.SimpleDateFormat;

import java.util.Date;

/\*\*

\* FileUtility

\*/

public class FileUtility {

// 文件参数群居变量

private static final String TAG = Constant.TAG;

private static final boolean D = Constant.DEBUG;

private static final String CHAR\_DOT = ".";

private static final String EXT\_JPEG = "jpg";

private static final String EXT\_DATA = "dat";

private static final int JPEG\_QUALITY = 100;

//时间

private SimpleDateFormat mFormat = new SimpleDateFormat( "yyyyMMddHHmmss" );

//文件路径

private File mDir;

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :FileUtility \*/

/\*Descrp :初始参数设置 \*/

/\*Input : String sub. \*/

/\*Output : None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

public FileUtility( String sub ) {

String path = Environment.getExternalStorageDirectory() + "/" + sub;

File dir = new File( path );

if ( !dir.exists() ) {

dir.mkdirs();

}

mDir = dir;

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :writeJpeg \*/

/\*Descrp :写入图片 \*/

/\*Input : String name, Bitmap bitmap \*/

/\*Output : None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

public void writeJpeg( String name, Bitmap bitmap ) {

File file = getFile( name, EXT\_JPEG );

writeJpeg( file, bitmap );

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :writeData \*/

/\*Descrp :写数据 \*/

/\*Input : String name, byte[] bytes \*/

/\*Output : None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

public void writeData( String name, byte[] bytes ) {

File file = getFile( name, EXT\_DATA );

writeData( file, bytes );

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :writeJpeg \*/

/\*Descrp :写入图片 \*/

/\*Input : String name, Bitmap bitmap \*/

/\*Output : None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

public void writeJpeg( File file, Bitmap bitmap ) {

log\_d( "writeJpeg " + file );

try {

FileOutputStream fos = new FileOutputStream( file );

bitmap.compress( Bitmap.CompressFormat.JPEG, JPEG\_QUALITY, fos );

fos.close();

} catch (Exception e) {

if (D) e.printStackTrace();

}

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :writeData \*/

/\*Descrp :写数据 \*/

/\*Input : File file, byte[] bytes \*/

/\*Output : None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

public void writeData( File file, byte[] bytes ) {

log\_d( "writeData " + file );

try {

FileOutputStream fos = new FileOutputStream( file );

BufferedOutputStream bos = new BufferedOutputStream( fos );//创建输出buffer

bos.write( bytes );

bos.flush();

bos.close();

} catch(Exception e) {

if (D) e.printStackTrace();

}

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name : File getJpegFile \*/

/\*Descrp :获取图片文件 \*/

/\*Input : String name \*/

/\*Output : None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

public File getJpegFile( String name ) {

return getFile( name, EXT\_JPEG );

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name : getDataFile \*/

/\*Descrp :获取文件数据 \*/

/\*Input : String name \*/

/\*Output : File . \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

public File getDataFile( String name ) {

return getFile( name, EXT\_DATA );

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name : getFile \*/

/\*Descrp :获取文件 \*/

/\*Input : String name, String ext. \*/

/\*Output : File . \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

public File getFile( String name, String ext ) {

String filename = name + CHAR\_DOT + ext;

File file = new File( mDir, filename );

return file;

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :getName \*/

/\*Descrp :获取文件 \*/

/\*Input : None. \*/

/\*Output : String . \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

public String getName() {

String name = mFormat.format( new Date() );

return name;

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :log\_d \*/

/\*Descrp :log. \*/

/\*Input : String. \*/

/\*Output : None . \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

private void log\_d( String str ) {

if (D) Log.d(TAG, str);

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/src文件下

MediaUtility.java android Media控制公共类

import android.content.Context;

import android.media.MediaScannerConnection;

import android.net.Uri;

import android.util.Log;

import java.io.File;

public class MediaUtility {

//全局变量定义

private static final String TAG = Constant.TAG;

private static final boolean D = Constant.DEBUG;

private static final String[] MIME\_TYPES = { "image/jpeg" };

private Context mContext;

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :MediaUtility. \*/

/\*Descrp :构造函数初始化 \*/

/\*Input : Context. \*/

/\*Output : None . \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

public MediaUtility( Context context ) {

mContext = context;

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :register. \*/

/\*Descrp :文件记录注册 \*/

/\*Input : File . \*/

/\*Output : None . \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

public void register( File file ) {

log\_d( "register " + file );

register( file.getAbsolutePath() );

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :register. \*/

/\*Descrp :文件记录注册 \*/

/\*Input : String. \*/

/\*Output : None . \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

public void register( String file ) {

log\_d( "register " + file );

String[] paths = { file };

MediaScannerConnection.scanFile( mContext, paths, MIME\_TYPES, mListener );

}

/\*\*

\* OnScanCompletedListener

\*/

MediaScannerConnection.OnScanCompletedListener mListener = new MediaScannerConnection.OnScanCompletedListener() {

@Override

public void onScanCompleted( String path, Uri uri ) {

log\_d( "onScanCompleted " + path + " " + uri );

}

};

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :log\_d \*/

/\*Descrp :log. \*/

/\*Input : String. \*/

/\*Output : None . \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

private void log\_d( String str ) {

if (D) Log.d(TAG, str);

}

}

/src文件下

ImageUtility.java 图片操作公共类

import android.content.Context;

import android.content.SharedPreferences;

import android.graphics.Bitmap;

import android.graphics.BitmapFactory;

import android.graphics.ImageFormat;

import android.graphics.Rect;

import android.graphics.YuvImage;

import android.os.Handler;

import android.os.Looper;

import android.os.Message;

import android.preference.PreferenceManager;

import android.util.Log;

import android.widget.Toast;

import java.io.ByteArrayOutputStream;

import java.io.File;

import java.nio.ByteBuffer;

import java.util.List;

public class ImageUtility {

// 全局变量参数

private static final String TAG = Constant.TAG;

private static final boolean D = Constant.DEBUG;

// YUV

private static final int[] YUV\_STRIDES = null;

private static final int JPEG\_RECT\_LEFT = 0;

private static final int JPEG\_RECT\_TOP = 0;

private static final int JPEG\_QUALITY = 80;

private static final int JPEG\_BYTE\_OFFSET = 0;

// RGB

private static final byte RGB\_ALPHA = (byte)0xff;

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :getBitmapFromYuv \*/

/\*Descrp :获取Bitmap从Yuv \*/

/\*Input : ByteBuffer buf, int format, int width, int height \*/

/\*Output :Bitmap. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

public Bitmap getBitmapFromYuv( ByteBuffer buf, int format, int width, int height ) {

byte[] bytes\_jpeg = yuvToJpeg( buf.array(), format, width, height );

Bitmap bitmap = jpegToBitmap( bytes\_jpeg );

return bitmap;

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :yuvToJpeg \*/

/\*Descrp :将yuv转换为Jpeg \*/

/\*Input : byte[] bytes, int format, int width, int height \*/

/\*Output :byte[]. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

private byte[] yuvToJpeg( byte[] bytes, int format, int width, int height ) {

YuvImage image = new YuvImage( bytes, format, width, height, YUV\_STRIDES );

ByteArrayOutputStream baos = new ByteArrayOutputStream();

Rect rect = new Rect( JPEG\_RECT\_LEFT, JPEG\_RECT\_TOP, width, height );

image.compressToJpeg( rect, JPEG\_QUALITY, baos );

byte[] data = baos.toByteArray();

return data;

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :jpegToBitmap \*/

/\*Descrp :将ARBG转换为Bitmap \*/

/\*Input : byte[]. \*/

/\*Output :Bitmap. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

private Bitmap jpegToBitmap( byte[] bytes ) {

BitmapFactory.Options options = new BitmapFactory.Options();

options.inPreferredConfig = Bitmap.Config.ARGB\_8888;

Bitmap bitmap = BitmapFactory.decodeByteArray( bytes, JPEG\_BYTE\_OFFSET, bytes.length, options );

return bitmap;

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :getBitmapFromRgb \*/

/\*Descrp :获取Bitmap从rgb \*/

/\*Input : ByteBuffer buf\_rgb, int width, int height \*/

/\*Output :Bitmap. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

public Bitmap getBitmapFromRgb( ByteBuffer buf\_rgb, int width, int height ) {

ByteBuffer buf\_argb = rbgToArgb( buf\_rgb, width \* height );

Bitmap bitmap = argbToBitmap( buf\_argb, width, height );

return bitmap;

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :argbToBitmap \*/

/\*Descrp :将ARBG转换为Bitmap \*/

/\*Input : ByteBuffer buf, int size \*/

/\*Output :ByteBuffer. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

private Bitmap argbToBitmap( ByteBuffer buf, int width, int height ) {

buf.position(0);

Bitmap bitmap = Bitmap.createBitmap( width, height, Bitmap.Config.ARGB\_8888 );

bitmap.copyPixelsFromBuffer( buf );

return bitmap;

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :rbgToArgb \*/

/\*Descrp :将RBG转换为ARBG \*/

/\*Input : ByteBuffer buf, int size \*/

/\*Output :ByteBuffer. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

private ByteBuffer rbgToArgb( ByteBuffer buf, int size ) {

ByteBuffer buf\_new = ByteBuffer.allocate( 4\*size );

for ( int i=0; i<size; i++ ) {

buf\_new.put( RGB\_ALPHA ); // A

buf\_new.put( buf.get( 3\*i + 0 ) ); // R

buf\_new.put( buf.get( 3\*i + 1 ) ); // G

buf\_new.put( buf.get( 3\*i + 2 ) ); // B

}

return buf\_new;

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :log\_d \*/

/\*Descrp :log. \*/

/\*Input : String. \*/

/\*Output : None . \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

private void log\_d( String str ) {

if (D) Log.d( TAG, str );

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/src文件下

Visionactivity.java 视觉获取函数

import android.app.Activity;

import android.content.SharedPreferences;

import android.graphics.Bitmap;

import android.graphics.ImageFormat;

import android.os.Bundle;

import android.preference.PreferenceManager;

import android.util.Log;

import android.view.View;

import android.widget.EditText;

import android.widget.ImageView;

import android.widget.Toast;

import com.aldebaran.qi.EmbeddedTools;

import com.aldebaran.qi.Session;

import com.aldebaran.qi.helper.proxies.ALVideoDevice;

import java.io.File;

import java.nio.ByteBuffer;

import java.util.List;

/\*\*

\* MainActivity

\*/

public class VisionActivity extends Activity {

// debug

private static final String TAG = Constant.TAG;

private static final boolean D = Constant.DEBUG;

private SharedPreferences mPreferences;

private ImageUtility mImageUtility;

private static final int RESOLUTION\_VGA = 2; // 640 x 480

// photo

private static final String PHOTO\_DIR = "nao";

// image remote

private static final String GVM\_NAME = "android\_client";

// class object

private RobotController mRobotController;

private FileUtility mFileUtility;

private MediaUtility mMediaUtility ;

// View

private EditText mEditTextIp;

private ImageView mImageViewPhoto;

private ByteBuffer mByteBuffer;

private static final int COLOR\_SPACE\_YUV422 = 9;

private static final int COLOR\_SPACE = COLOR\_SPACE\_YUV422;

// frames per second

private static final int FPS\_5 = 5;

private Bitmap mBitmap;

private ALVideoDevice mALVideoDevice;

private int YUV\_FORMAT = ImageFormat.YUY2;

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :onCreate \*/

/\*Descrp :初始化 \*/

/\*Input :Bundle. \*/

/\*Output :None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

mMediaUtility = new MediaUtility( this );

mFileUtility = new FileUtility( PHOTO\_DIR );

initRobot();//初始化

mImageViewPhoto = (ImageView) findViewById( R.id.ImageView\_photo );

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :initRobot \*/

/\*Descrp :初始化 \*/

/\*Input :None. \*/

/\*Output :None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* private void initRobot() {

try {

mALVideoDevice = new ALVideoDevice(Statictext.mQiSession );

} catch (Exception e) {

if (D) e.printStackTrace();

}

mPreferences = PreferenceManager.getDefaultSharedPreferences(this);

mImageUtility = new ImageUtility();

EmbeddedTools tools = new EmbeddedTools();

File dir =this.getCacheDir();

log\_d( "Extracting libraries in " + dir.getAbsolutePath() );

tools.overrideTempDirectory(dir);

tools.loadEmbeddedLibraries();

procImageRemoteChanged( mByteBuffer, mBitmap );

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :robotGetImageRemote \*/

/\*Descrp ：获取图片 \*/

/\*Input :ImageRemoteResult. \*/

/\*Output :None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

private ImageRemoteResult robotGetImageRemote( String name ) {

log\_d("robotGetImageRemote " + name);

ImageRemoteResult result = null;

List<Object> list = null;

try {

list = (List<Object>) mALVideoDevice.getImageRemote( name );

} catch (Exception e) {

if (D) e.printStackTrace();

}

log\_d("getImageRemote " + list);

if (( list != null )&&( list.size() > 0 )) {

result = new ImageRemoteResult( list );

}

return result;

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :robotUnsubscribe \*/

/\*Descrp ：video取消获取资源 \*/

/\*Input :ImageRemoteResult. \*/

/\*Output :None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* private void robotUnsubscribe( String name ) {

log\_d( "robotUnsubscribe " + name );

try {

mALVideoDevice.unsubscribe(name);

} catch (Exception e) {

if (D) e.printStackTrace();

}

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :getImageRemoteBitmap \*/

/\*Descrp ：获取Bitmap图片 \*/

/\*Input :None. \*/

/\*Output :None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

private void getImageRemoteBitmap() {

mByteBuffer = null;

mBitmap = null;

ImageRemoteResult result = null;

String client = robotSubscribe( GVM\_NAME, RESOLUTION\_VGA, COLOR\_SPACE, FPS\_5 );

if (( client != null )&&( client.length() > 0 )) {

result = robotGetImageRemote( client );

// WIN DEATH !

// sleep( IMAGE\_WAIT );

robotUnsubscribe( client );

}

if (( result != null )&&( result.buf != null )&&( result.buf.limit() > 0 )) {

mByteBuffer = result.buf;

mBitmap = mImageUtility.getBitmapFromYuv( mByteBuffer, YUV\_FORMAT, result.width, result.height );

// mBitmap = mImageUtility.getBitmapFromRgb( mByteBuffer, result.width, result.height );

}

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :robotSubscribe \*/

/\*Descrp ：nao摄像头调取 \*/

/\*Input :String gvmName, int resolution, int colorSpace, int fps \*/

/\*Output :string. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

private String robotSubscribe( String gvmName, int resolution, int colorSpace, int fps ) {

log\_d("robotSubscribe " + gvmName + " " + resolution + " " + colorSpace + " " + fps);

String str = "";

try {

str = mALVideoDevice.subscribe( gvmName, resolution, colorSpace, fps );

} catch (Exception e) {

if (D) e.printStackTrace();

}

log\_d( "subscribe " + str );

return str;

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :procImageRemoteChanged \*/

/\*Descrp ：图片格式转换 \*/

/\*Input :ByteBuffer buf, Bitmap bitmap \*/

/\*Output :None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

private void procImageRemoteChanged( ByteBuffer buf, Bitmap bitmap ) {

log\_d( "procImageRemoteChanged " + buf + " " + bitmap );

String name = mFileUtility.getName();

if ( buf != null ) {

mFileUtility.writeData( name, buf.array() );

}

if ( bitmap != null ) {

mImageViewPhoto.setImageBitmap( bitmap );

File file = mFileUtility.getJpegFile( name );

mFileUtility.writeJpeg( file, bitmap );

mMediaUtility.register( file );

toast\_short("Photo saved");

} else {

toast\_short("getImageRemote failed");

}

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :onClickGetImageRemote \*/

/\*Descrp ：按钮触发函数 \*/

/\*Input :View \*/

/\*Output :None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

public void onClickGetImageRemote( View view ) {

log\_d("onClickGetImageRemote");

mRobotController.getImageRemote();

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :toast\_short \*/

/\*Descrp ：toast弹出时间 \*/

/\*Input :String \*/

/\*Output :None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

private void toast\_short( String str ) {

Toast.makeText(this, str, Toast.LENGTH\_SHORT).show();

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :toast\_short \*/

/\*Descrp ：toast弹出时间 \*/

/\*Input :int \*/

/\*Output :None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

private void toast\_short( int res\_id ) {

Toast.makeText(this, res\_id, Toast.LENGTH\_SHORT).show();

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :log\_d \*/

/\*Descrp ：log \*/

/\*Input :String \*/

/\*Output :None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

private void log\_d( String str ) {

if (D) Log.d( TAG, str );

}

/\*\*

\* --- class ImageRemoteResult ---

\* [0] : width; [1] : height; [2] : number of layers; [3] : ColorSpace; [4] : time stamp (highest 32 bits); [5] : time stamp (lowest 32 bits); [6] : array of size height \* width \* nblayers containing image data; [7] : cameraID; [8] : left angle; [9] : top angle; [10] : right angle; [11] : bottom angle;

\*/

private class ImageRemoteResult {

public int width = 0;

public int height = 0;

public int layers = 0;

public int color = 0;

public int time1 = 0;

public int time2 = 0;

public long time = 0;

public int id = 0;

public float left = 0;

public float top = 0;

public float right = 0;

public float bottom = 0;

ByteBuffer buf = null;

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :ImageRemoteResult \*/

/\*Descrp ：初始化 \*/

/\*Input :List<Object> list \*/

/\*Output :None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

public ImageRemoteResult( List<Object> list ) {

log\_d("ImageRemoteResult " + list);

width = (int) list.get(0);

height = (int) list.get(1);

layers = (int) list.get(2);

color = (int) list.get(3);

time1 = (int) list.get(4);

time2 = (int) list.get(5);

time = ((long)time1 << 32 ) + (long)time2;

buf = (ByteBuffer) list.get(6);

id = (int) list.get(7);

left = (float) list.get(8);

top = (float) list.get(9);

right = (float) list.get(10);

bottom = (float) list.get(11);

debug();

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :debug \*/

/\*Descrp ：初始化 \*/

/\*Input :None. \*/

/\*Output :None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

private void debug() {

String msg = "";

msg += " width=" + width;

msg += " height=" + height;

msg += " layers=" + layers;

msg += " color=" + color;

msg += " time=" + time;

msg += " id=" + id;

msg += " left=" + left;

msg += " top=" + top;

msg += " right=" + right;

msg += " bottom=" + bottom;

msg += " buffer=" + buf;

log\_d( msg );

}

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/src文件下

Statictex.java 公共类

package com.naorobot;

import com.aldebaran.qi.Session;

public class Statictext {

public static Session mQiSession;

public static String Ip;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/libs第三方库文件

连接nao和线上兼容android高版本文件

java-naoqi-sdk-2.1.4-android.jar//第三方库文件

armeabi-v7a//兼容android高版本文件

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

软件名称：软件后台代码

版本： V1.0

作者： 中南民族大学

开发人员：汪红 后面是学生姓名

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/

tcp\_iocp.h windows iocp服务器模型搭建头文件

**#ifndef \_\_TCP\_IOCP\_H\_\_**

**#define \_\_TCP\_IOCP\_H\_\_**

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name : **start\_server** \*/

/\*Descrp ：创建服务器端口 \*/

/\*Input :**int**. \*/

/\*Output :None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**void start\_server(int port);**

**#endif**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/

tcp\_iocp.c windows iocp服务器模型搭建

#include<stdio.h>

#include<string.h>

#include <stdlib.h>

#include "tcp\_iocp.h"

#include <WinSock2.h>

#include <mswsock.h>

#include <windows.h>

#pragma comment(lib, "ws2\_32.lib")

#pragma comment(lib, "odbc32.lib")

#pragma comment(lib, "odbccp32.lib")

#include "tcp\_session.h"

enum {

IOCP\_ACCPET = 0,

IOCP\_RECV,

IOCP\_WRITE,

};

#define MAX\_RECV\_SIZE 8192

struct io\_package {

WSAOVERLAPPED overlapped;

int opt; // 标记一下我们当前的请求的类型;

int accpet\_sock;

WSABUF wsabuffer;

char pkg[MAX\_RECV\_SIZE];

};

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :post\_accept \*/

/\*Descrp ：端口接收函数 \*/

/\*Input :SOCKET l\_sock, HANDLE iocp \*/

/\*Output :None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

static void

post\_accept(SOCKET l\_sock, HANDLE iocp) {

struct io\_package\* pkg = malloc(sizeof(struct io\_package));

memset(pkg, 0, sizeof(struct io\_package));

pkg->wsabuffer.buf = pkg->pkg;

pkg->wsabuffer.len = MAX\_RECV\_SIZE - 1;

pkg->opt = IOCP\_ACCPET;

DWORD dwBytes = 0;

SOCKET client = WSASocket(AF\_INET, SOCK\_STREAM, IPPROTO\_TCP, NULL, 0, WSA\_FLAG\_OVERLAPPED);

int addr\_size = (sizeof(struct sockaddr\_in) + 16);

pkg->accpet\_sock = client;

AcceptEx(l\_sock, client, pkg->wsabuffer.buf, 0/\*pkg->wsabuffer.len - addr\_size\* 2\*/,

addr\_size, addr\_size, &dwBytes, &pkg->overlapped);

}

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :post\_recv \*/

/\*Descrp ：异步发送请求 \*/

/\*Input :SOCKET client\_fd, HANDLE iocp \*/

/\*Output :None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

static void

post\_recv(SOCKET client\_fd, HANDLE iocp) {

// 异步发送请求;

struct io\_package\* io\_data = malloc(sizeof(struct io\_package));

// 清0的主要目的是为了能让overlapped清0;

memset(io\_data, 0, sizeof(struct io\_package));

io\_data->opt = IOCP\_RECV;

io\_data->wsabuffer.buf = io\_data->pkg;

io\_data->wsabuffer.len = MAX\_RECV\_SIZE - 1;

// 发送了recv的请求;

//

DWORD dwRecv = 0;

DWORD dwFlags = 0;

int ret = WSARecv(client\_fd, &(io\_data->wsabuffer),

1, &dwRecv, &dwFlags,

&(io\_data->overlapped), NULL);

}

void

start\_server(int port) {

init\_session\_manager(1);

WSADATA data;

WSAStartup(MAKEWORD(2, 2), &data);

// 新建一个完成端口;

SOCKET l\_sock = INVALID\_SOCKET;

HANDLE iocp = CreateIoCompletionPort(INVALID\_HANDLE\_VALUE, NULL, 0, 0);

if (iocp == NULL) {

goto failed;

}

// 创建一个线程

// CreateThread(NULL, 0, ServerWorkThread, (LPVOID)iocp, 0, 0);

// end

// 创建监听socket

l\_sock = socket(AF\_INET, SOCK\_STREAM, 0);

if (l\_sock == INVALID\_SOCKET) {

goto failed;

}

// bind socket

struct sockaddr\_in s\_address;

memset(&s\_address, 0, sizeof(s\_address));

s\_address.sin\_family = AF\_INET;

s\_address.sin\_addr.s\_addr = inet\_addr("127.0.0.1");

s\_address.sin\_port = htons(port);

if (bind(l\_sock, (struct sockaddr \*) &s\_address, sizeof(s\_address)) != 0) {

goto failed;

}

if (listen(l\_sock, SOMAXCONN) != 0) {

goto failed;

}

// start

CreateIoCompletionPort((HANDLE)l\_sock, iocp, (DWORD)0, 0);

post\_accept(l\_sock, iocp);

// end

DWORD dwTrans;

struct session\* s;

struct io\_package\* io\_data;

printf("服务器启动..\n");

while (1) {

clear\_offline\_session();

// 阻塞函数，当IOCP唤醒这个线程来处理已经发生事件

s = NULL;

dwTrans = 0;

io\_data = NULL;

int ret = GetQueuedCompletionStatus(iocp, &dwTrans, (LPDWORD)&s, (LPOVERLAPPED\*)&io\_data, WSA\_INFINITE);

if (ret == 0) {

printf("iocp error");

continue;

}

// IOCP端口唤醒了一个工作线程，

printf("IOCP have event\n");

if (dwTrans == 0 && io\_data->opt == IOCP\_RECV) { // socket 关闭

close\_session(s);

free(io\_data);

continue;

}// end

switch (io\_data->opt) {

case IOCP\_RECV: { // 完成端口意味着数据已经读好

io\_data->pkg[dwTrans] = 0;

printf("IOCP recv %d，%s\n", dwTrans, io\_data->pkg);

// 当读的请求完成后，你必须要再加一个读的请求;

DWORD dwRecv = 0;

DWORD dwFlags = 0;

int ret = WSARecv(s->c\_sock, &(io\_data->wsabuffer),

1, &dwRecv, &dwFlags,

&(io\_data->overlapped), NULL);

}

break;

case IOCP\_ACCPET://创建接收端口

{

int client\_fd = io\_data->accpet\_sock;

int addr\_size = (sizeof(struct sockaddr\_in) + 16);

struct sockaddr\_in\* l\_addr = NULL;

int l\_len = sizeof(struct sockaddr\_in);

struct sockaddr\_in\* r\_addr = NULL;

int r\_len = sizeof(struct sockaddr\_in);

GetAcceptExSockaddrs(io\_data->wsabuffer.buf,

0, /\*io\_data->wsabuffer.len - addr\_size \* 2, \*/

addr\_size, addr\_size,

(struct sockaddr\*\*)&l\_addr, &l\_len,

(struct sockaddr\*\*)&r\_addr, &r\_len);

struct session\* s = save\_session(client\_fd, inet\_ntoa(r\_addr->sin\_addr), ntohs(r\_addr->sin\_port));

CreateIoCompletionPort((HANDLE)client\_fd, iocp, (DWORD)s, 0);

post\_recv(client\_fd, iocp);

post\_accept(l\_sock, iocp);

}

break;

}

}

failed:

if (iocp != NULL) {

CloseHandle(iocp);

}

if (l\_sock != INVALID\_SOCKET) {

closesocket(l\_sock);

}

WSACleanup();

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/

tcp\_session.h 连接管理头文件

#ifndef \_\_TCP\_SESSION\_H\_\_

#define \_\_TCP\_SESSION\_H\_\_

struct session {

char c\_ip[32];

int c\_port;

int c\_sock;

int removed;

void\* player; // 指向玩家数据;

struct session\* next;

};

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :init\_session\_manager \*/

/\*Descrp :初始化连接管理 \*/

/\*Input :None. \*/

/\*Output :None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void init\_session\_manager();

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :exit\_session\_manager \*/

/\*Descrp :退出连接管理 \*/

/\*Input :None. \*/

/\*Output :None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void exit\_session\_manager();

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :save\_session \*/

/\*Descrp :有客服端进来，保存这个sesssion \*/

/\*Input :int c\_sock, char\* ip, int port \*/

/\*Output :struct session\* \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

struct session\* save\_session(int c\_sock, char\* ip, int port);

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :close\_session \*/

/\*Descrp :关掉sesssion \*/

/\*Input :struct session\* s \*/

/\*Output :None \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void close\_session(struct session\* s);

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :foreach\_online\_session \*/

/\*Descrp : 遍历我们session集合里面的所有session \*/

/\*Input :int(\*callback)(struct session\* s, void\* p), void\*p \*/

/\*Output :None \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void foreach\_online\_session(int(\*callback)(struct session\* s, void\* p), void\*p);

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :session\_on\_recv \*/

/\*Descrp : 处理我们的数据 \*/

/\*Input :struct session\* s \*/

/\*Output :None \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void session\_on\_recv(struct session\* s);

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :clear\_offline\_session \*/

/\*Descrp : 清除我们的session \*/

/\*Input :None \*/

/\*Output :None \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void clear\_offline\_session();

#endif

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/

tcp\_session.c 连接实现

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

#ifdef WIN32

#include <WinSock2.h>

#include <Windows.h>

#endif

#include "tcp\_session.h"

#define MAX\_SESSION\_NUM 6000

#define my\_malloc malloc

#define my\_free free

#define MAX\_RECV\_BUFFER 8096

struct {

struct session\* online\_session;

struct session\* cache\_mem;

struct session\* free\_list;

char recv\_buffer[MAX\_RECV\_BUFFER];

int readed; // 当前已经从socket里面读取的数据;

int has\_removed;

int prot\_mode; // 0 表示二进制协议，size + 数据的模式

// 1,表示文本协议，以回车换行来分解收到的数据为一个包

}session\_manager;

static struct session\* cache\_alloc() {

struct session\* s = NULL;

if (session\_manager.free\_list != NULL) {

s = session\_manager.free\_list;

session\_manager.free\_list = s->next;

}

else { // 调用系统的函数 malloc

s = my\_malloc(sizeof(struct session));

}

memset(s, 0, sizeof(struct session));

return s;

}

static void cache\_free(struct session\* s) {

// 判断一下，是从cache分配出去的，还是从系统my\_malloc分配出去的？

if (s >= session\_manager.cache\_mem && s < session\_manager.cache\_mem + MAX\_SESSION\_NUM) {

s->next = session\_manager.free\_list;

session\_manager.free\_list = s;

}

else {

my\_free(s);

}

//

}

void init\_session\_manager() {

memset(&session\_manager, 0, sizeof(session\_manager));

// 将6000个session一次分配出来。

session\_manager.cache\_mem = (struct session\*)my\_malloc(MAX\_SESSION\_NUM \* sizeof(struct session));

memset(session\_manager.cache\_mem, 0, MAX\_SESSION\_NUM \* sizeof(struct session));

// end

for (int i = 0; i < MAX\_SESSION\_NUM; i++) {

session\_manager.cache\_mem[i].next = session\_manager.free\_list;

session\_manager.free\_list = &session\_manager.cache\_mem[i];

}

}

void exit\_session\_manager() {

}

struct session\* save\_session(int c\_sock, char\* ip, int port) {

struct session\* s = cache\_alloc();

s->c\_sock = c\_sock;

s->c\_port = port;

int len = strlen(ip);

if (len >= 32) {

len = 31;

}

strncpy(s->c\_ip, ip, len);

s->c\_ip[len] = 0;

s->next = session\_manager.online\_session;

session\_manager.online\_session = s;

return s;

}

void foreach\_online\_session(int(\*callback)(struct session\* s, void\* p), void\*p) {

if (callback == NULL) {

return;

}

struct session\* walk = session\_manager.online\_session;

while (walk) {

if (walk->removed == 1) {

walk = walk->next;

continue;

}

if (callback(walk, p)) {

return;

}

walk = walk->next;

}

}

void close\_session(struct session\* s) {

s->removed = 1;

session\_manager.has\_removed = 1;

printf("client %s:%d exit\n", s->c\_ip, s->c\_port);

}

static void text\_process\_package(struct session\* s) {

}

static void bin\_process\_package(struct session\* s) {

if (session\_manager.readed < 4) { // 保留好数据，return

return;

}

// 二进制分包的规则

int\* pack\_size = (int\*)session\_manager.recv\_buffer;

int pack\_len = (\*pack\_size);

if (pack\_len > MAX\_RECV\_BUFFER) {

goto pack\_failed;

}

// end

// 包没有收完

int total = 0; // 总共处理掉的包的数据大小

while ((session\_manager.readed - total) >= pack\_len) {

// 包收完了,结合玩家进行处理。

// end

// 移动到下一个包进行处理

total += pack\_len;

if (session\_manager.readed - total < 4) {

if (session\_manager.readed > total) {

memmove(session\_manager.recv\_buffer, session\_manager.recv\_buffer + total, (session\_manager.readed - total));

}

session\_manager.readed -= total;

return;

}

pack\_size = (int\*)(session\_manager.recv\_buffer + total);

pack\_len = (\*pack\_size);

if (pack\_len > MAX\_RECV\_BUFFER) {

goto pack\_failed;

}

if ((session\_manager.readed - total) < pack\_len) {

memmove(session\_manager.recv\_buffer, session\_manager.recv\_buffer + total, (session\_manager.readed - total));

session\_manager.readed -= total;

return;

}

// end

}

return;

pack\_failed:

close\_session(s);

}

// 处理我们的socket数据

void session\_on\_recv(struct session\* s) {

int readed = recv(s->c\_sock, session\_manager.recv\_buffer + session\_manager.readed, MAX\_RECV\_BUFFER - session\_manager.readed, 0);

if (readed <= 0) { // 客服端已经关闭了socket

close\_session(s);

return;

}

session\_manager.readed += readed;

if (session\_manager.prot\_mode == 0) {

bin\_process\_package(s);

}

else {

text\_process\_package(s);

}

}

void clear\_offline\_session() {

if (session\_manager.has\_removed == 0) {

return;

}

struct session\*\* walk = &session\_manager.online\_session;

while (\*walk) {

struct session\* s = (\*walk);

if (s->removed) {

\*walk = s->next;

s->next = NULL;

closesocket(s->c\_sock);

s->c\_sock = 0;

// 释放session

cache\_free(s);

}

else {

walk = &(\*walk)->next;

}

}

session\_manager.has\_removed = 0;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/

main.c 主函数

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

#include "class\_code/tcp\_iocp.h"

/\*\*\*FUNC+\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Name :main \*/

/\*Descrp :main process \*/

/\*Input :None. \*/

/\*Output :None. \*/

/\*Return :None. \*/

/\*\*\*FUNC-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

int main(int argc, char\*\* argv) {

start\_server(5150);

return 0;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*