目录

[一、主要源代码： 1](#_Toc485394514)

[二、主要API说明： 10](#_Toc485394515)

[1、 FlightController类 10](#_Toc485394516)

[2、 CommonCallbacks类 10](#_Toc485394517)

[3、LocationCoordinate2D类 14](#_Toc485394518)

[4、 Aircraft类 15](#_Toc485394519)

[5、FlightControllerState类 16](#_Toc485394520)

[6、LocationCoordinate3D类 17](#_Toc485394521)

[7、MissionControl类 19](#_Toc485394522)

[8、HotpointMissionState类 21](#_Toc485394523)

[9、HotpointMissionOperator类 23](#_Toc485394524)

[10、HotpointMission类 24](#_Toc485394525)

## 一、主要源代码：

package com.dji.sdk.sample.demo.timeline;

|  |  |
| --- | --- |
|  | import android.app.Service; |
|  | import android.content.Context; |
|  | import android.support.annotation.NonNull; |
|  | import android.support.annotation.Nullable; |
|  | import android.util.AttributeSet; |
|  | import android.view.LayoutInflater; |
|  | import android.view.View; |
|  | import android.view.View.OnClickListener; |
|  | import android.widget.Button; |
|  | import android.widget.LinearLayout; |
|  | import android.widget.ProgressBar; |
|  | import android.widget.TextView; |
|  | import android.widget.Toast; |
|  | import com.dji.sdk.sample.R; |
|  | import com.dji.sdk.sample.internal.controller.DJISampleApplication; |
|  | import com.dji.sdk.sample.internal.utils.GeneralUtils; |
|  | import com.dji.sdk.sample.internal.utils.ToastUtils; |
|  | import com.dji.sdk.sample.internal.view.PresentableView; |
|  | import dji.common.error.DJIError; |
|  | import dji.common.gimbal.Attitude; |
|  | import dji.common.gimbal.Rotation; |
|  | import dji.common.mission.hotpoint.HotpointHeading; |
|  | import dji.common.mission.hotpoint.HotpointMission; |
|  | import dji.common.mission.hotpoint.HotpointStartPoint; |
|  | import dji.common.mission.waypoint.Waypoint; |
|  | import dji.common.mission.waypoint.WaypointAction; |
|  | import dji.common.mission.waypoint.WaypointActionType; |
|  | import dji.common.mission.waypoint.WaypointMission; |
|  | import dji.common.mission.waypoint.WaypointMissionFinishedAction; |
|  | import dji.common.mission.waypoint.WaypointMissionFlightPathMode; |
|  | import dji.common.mission.waypoint.WaypointMissionGotoWaypointMode; |
|  | import dji.common.mission.waypoint.WaypointMissionHeadingMode; |
|  | import dji.common.model.LocationCoordinate2D; |
|  | import dji.common.util.CommonCallbacks; |
|  | import dji.sdk.base.BaseProduct; |
|  | import dji.sdk.flightcontroller.FlightController; |
|  | import dji.sdk.mission.MissionControl; |
|  | import dji.sdk.mission.timeline.TimelineElement; |
|  | import dji.sdk.mission.timeline.TimelineEvent; |
|  | import dji.sdk.mission.timeline.Mission; |
|  | import dji.sdk.mission.timeline.actions.GimbalAttitudeAction; |
|  | import dji.sdk.mission.timeline.actions.GoHomeAction; |
|  | import dji.sdk.mission.timeline.actions.GoToAction; |
|  | import dji.sdk.mission.timeline.actions.HotpointAction; |
|  | import dji.sdk.mission.timeline.actions.RecordVideoAction; |
|  | import dji.sdk.mission.timeline.actions.ShootPhotoAction; |
|  | import dji.sdk.mission.timeline.actions.TakeOffAction; |
|  | import dji.sdk.products.Aircraft; |
|  | import java.util.ArrayList; |
|  | import java.util.LinkedList; |
|  | import java.util.List; |
|  | import java.util.concurrent.CountDownLatch; |
|  | import java.util.concurrent.TimeUnit; |
|  |  |
|  | /\*\* |
|  | \* Class for Timeline MissionControl. |
|  | \*/ |
|  | public class TimelineMissionControlView extends LinearLayout implements OnClickListener, PresentableView { |
|  |  |
|  | private MissionControl missionControl; |
|  | private FlightController flightController; |
|  |  |
|  | protected Button prepareBtn; |
|  | protected Button startBtn; |
|  | protected Button stopBtn; |
|  | protected Button pauseBtn; |
|  | protected Button resumeBtn; |
|  | protected Button cleanBtn; |
|  |  |
|  | protected TextView timelineInfoTV; |
|  | protected TextView runningInfoTV; |
|  | protected ProgressBar progressBar; |
|  |  |
|  | protected double homeLatitude = 181; |
|  | protected double homeLongitude = 181; |
|  |  |
|  | public TimelineMissionControlView(Context context) { |
|  | super(context); |
|  | initUI(context); |
|  | } |
|  |  |
|  | public TimelineMissionControlView(Context context, @Nullable AttributeSet attrs) { |
|  | super(context); |
|  | initUI(context); |
|  | } |
|  |  |
|  | public TimelineMissionControlView(Context context, @Nullable AttributeSet attrs, int defStyleAttr) { |
|  | super(context); |
|  | initUI(context); |
|  | } |
|  |  |
|  | private void setRunningResultToText(final String s) { |
|  | post(new Runnable() { |
|  | @Override |
|  | public void run() { |
|  | if (runningInfoTV == null) { |
|  | Toast.makeText(getContext(), "textview = null", Toast.LENGTH\_SHORT).show(); |
|  | } else { |
|  | runningInfoTV.append(s + "\n"); |
|  | } |
|  | } |
|  | }); |
|  | } |
|  |  |
|  | private void setTimelinePlanToText(final String s) { |
|  |  |
|  | post(new Runnable() { |
|  | @Override |
|  | public void run() { |
|  | if (timelineInfoTV == null) { |
|  | Toast.makeText(getContext(), "textview = null", Toast.LENGTH\_SHORT).show(); |
|  | } else { |
|  | timelineInfoTV.append(s + "\n"); |
|  | } |
|  | } |
|  | }); |
|  | } |
|  |  |
|  | private void initTimeline() { |
|  | if (!GeneralUtils.checkGpsCoordinate(homeLatitude, homeLongitude)) { |
|  | ToastUtils.setResultToToast("No home point!!!"); |
|  | return; |
|  | } |
|  |  |
|  | List<TimelineElement> elements = new ArrayList<>(); |
|  |  |
|  | missionControl = MissionControl.getInstance(); |
|  | final TimelineEvent preEvent = null; |
|  | MissionControl.Listener listener = new MissionControl.Listener() { |
|  | @Override |
|  | public void onEvent(@Nullable TimelineElement element, TimelineEvent event, DJIError error) { |
|  | updateTimelineStatus(element, event, error); |
|  | } |
|  | }; |
|  |  |
|  | //Step 1: takeoff from the ground |
|  | setTimelinePlanToText("Step 1: takeoff from the ground"); |
|  | elements.add(new TakeOffAction()); |
|  |  |
|  | //Step 2: reset the gimbal to horizontal angle in 2 seconds. |
|  | //setTimelinePlanToText("Step 2: reset the gimbal to horizontal angle in 2 seconds"); |
|  | //Attitude attitude = new Attitude(0, Rotation.NO\_ROTATION, Rotation.NO\_ROTATION); |
|  | //GimbalAttitudeAction gimbalAction = new GimbalAttitudeAction(attitude); |
|  | //gimbalAction.setCompletionTime(2); |
|  | //elements.add(gimbalAction); |
|  |  |
|  | //Step 3: Go 10 meters from home point |
|  | //setTimelinePlanToText("Step 3: Go 10 meters from home point"); |
|  | //elements.add(new GoToAction(new LocationCoordinate2D(homeLatitude, homeLongitude), 10)); |
|  |  |
|  | //Step 4: shoot 3 photos with 2 seconds interval between each |
|  | //setTimelinePlanToText("Step 4: shoot 3 photos with 2 seconds interval between each"); |
|  | //elements.add(new ShootPhotoAction(3, 2)); |
|  |  |
|  | //Step 5: shoot a single photo |
|  | //setTimelinePlanToText("Step 5: shoot a single photo"); |
|  | //elements.add(new ShootPhotoAction()); |
|  |  |
|  | //Step 6: start recording video |
|  | setTimelinePlanToText("Step 2: start recording video"); |
|  | elements.add(new RecordVideoAction(true)); |
|  |  |
|  |  |
|  | //Step 7: start a waypoint mission while the aircraft is still recording the video |
|  | setTimelinePlanToText("Step 3: start a waypoint mission while the aircraft is still recording the video"); |
|  | elements.add(Mission.elementFromWaypointMission(initTestingWaypointMission())); |
|  |  |
|  | //Step 8: stop the recording when the waypoint mission is finished |
|  | setTimelinePlanToText("Step 4: stop the recording when the waypoint mission is finished"); |
|  | elements.add(new RecordVideoAction(false)); |
|  |  |
|  | //Step 9: shoot a single photo |
|  | //setTimelinePlanToText("Step 9: shoot a single photo"); |
|  | //elements.add(new ShootPhotoAction()); |
|  |  |
|  | //Step 10: start a hotpoint mission |
|  | setTimelinePlanToText("Step 5: start a hotpoint mission to surround 360 degree"); |
|  | HotpointMission hotpointMission = new HotpointMission(); |
|  | hotpointMission.setHotpoint(new LocationCoordinate2D(homeLatitude, homeLongitude)); |
|  | hotpointMission.setAltitude(40); |
|  | hotpointMission.setRadius(10); |
|  | hotpointMission.setAngularVelocity(10); |
|  | HotpointStartPoint startPoint = HotpointStartPoint.NEAREST; |
|  | hotpointMission.setStartPoint(startPoint); |
|  | HotpointHeading heading = HotpointHeading.TOWARDS\_HOT\_POINT; |
|  | hotpointMission.setHeading(heading); |
|  | elements.add(new HotpointAction(hotpointMission, 360)); |
|  |  |
|  | //Step 11: go back home |
|  | setTimelinePlanToText("Step 6: go back home"); |
|  | elements.add(new GoHomeAction()); |
|  |  |
|  | if (missionControl.scheduledCount() > 0) { |
|  | missionControl.unscheduleEverything(); |
|  | missionControl.removeAllListeners(); |
|  | } |
|  |  |
|  | missionControl.scheduleElements(elements); |
|  | missionControl.addListener(listener); |
|  | } |
|  |  |
|  | private void updateTimelineStatus(@Nullable TimelineElement element, TimelineEvent event, DJIError error) { |
|  |  |
|  | if (element != null) { |
|  | if (element instanceof Mission) { |
|  | setRunningResultToText(((Mission) element).getMissionObject().getClass().getSimpleName() |
|  | + " event is " |
|  | + event.toString() |
|  | + " " |
|  | + (error == null ? "" : error.getDescription())); |
|  | } else { |
|  | setRunningResultToText(element.getClass().getSimpleName() |
|  | + " event is " |
|  | + event.toString() |
|  | + " " |
|  | + (error == null ? "" : error.getDescription())); |
|  | } |
|  | } else { |
|  | setRunningResultToText("Timeline Event is " + event.toString() + " " + (error == null |
|  | ? "" |
|  | : "Failed:" |
|  | + error.getDescription())); |
|  | } |
|  |  |
|  | } |
|  |  |
|  | private WaypointMission initTestingWaypointMission() { |
|  | if (!GeneralUtils.checkGpsCoordinate(homeLatitude, homeLongitude)) { |
|  | ToastUtils.setResultToToast("No home point!!!"); |
|  | return null; |
|  | } |
|  |  |
|  | WaypointMission.Builder waypointMissionBuilder = new WaypointMission.Builder().autoFlightSpeed(5f) |
|  | .maxFlightSpeed(10f) |
|  | .setExitMissionOnRCSignalLostEnabled(false) |
|  | .finishedAction(WaypointMissionFinishedAction.NO\_ACTION) |
|  | .flightPathMode(WaypointMissionFlightPathMode.NORMAL) |
|  | .gotoFirstWaypointMode(WaypointMissionGotoWaypointMode.SAFELY) |
|  | .headingMode(WaypointMissionHeadingMode.AUTO) |
|  | .repeatTimes(1);; |
|  | List<Waypoint> waypoints = new LinkedList<>(); |
|  |  |
|  | Waypoint northPoint = new Waypoint(homeLatitude + 10 \* GeneralUtils.ONE\_METER\_OFFSET, homeLongitude, 10f); |
|  | Waypoint eastPoint = |
|  | new Waypoint(homeLatitude, homeLongitude + 10 \* GeneralUtils.calcLongitudeOffset(homeLatitude), 20f); |
|  | Waypoint southPoint = new Waypoint(homeLatitude - 10 \* GeneralUtils.ONE\_METER\_OFFSET, homeLongitude, 30f); |
|  | Waypoint westPoint = |
|  | new Waypoint(homeLatitude, homeLongitude - 10 \* GeneralUtils.calcLongitudeOffset(homeLatitude), 40f); |
|  |  |
|  | northPoint.addAction(new WaypointAction(WaypointActionType.GIMBAL\_PITCH, -60)); |
|  | southPoint.addAction(new WaypointAction(WaypointActionType.ROTATE\_AIRCRAFT, 60)); |
|  |  |
|  | waypoints.add(northPoint); |
|  | waypoints.add(eastPoint); |
|  | waypoints.add(southPoint); |
|  | waypoints.add(westPoint); |
|  |  |
|  | waypointMissionBuilder.waypointList(waypoints).waypointCount(waypoints.size()); |
|  | return waypointMissionBuilder.build(); |
|  | } |
|  |  |
|  | private void startTimeline() { |
|  | if (MissionControl.getInstance().scheduledCount() > 0) { |
|  | MissionControl.getInstance().startTimeline(); |
|  | } else { |
|  | ToastUtils.setResultToToast("Init the timeline first by clicking the Init button"); |
|  | } |
|  | } |
|  |  |
|  | private void stopTimeline() { |
|  | MissionControl.getInstance().stopTimeline(); |
|  | } |
|  |  |
|  | private void pauseTimeline() { |
|  | MissionControl.getInstance().pauseTimeline(); |
|  | } |
|  |  |
|  |  |
|  | private void resumeTimeline() { |
|  | MissionControl.getInstance().resumeTimeline(); |
|  | } |
|  |  |
|  | private void cleanTimelineDataAndLog() { |
|  | if (missionControl.scheduledCount() > 0) { |
|  | missionControl.unscheduleEverything(); |
|  | missionControl.removeAllListeners(); |
|  | } |
|  | runningInfoTV.setText(""); |
|  | timelineInfoTV.setText(""); |
|  | } |
|  |  |
|  |  |
|  |  |
|  | @Override |
|  | protected void onAttachedToWindow() { |
|  | super.onAttachedToWindow(); |
|  | BaseProduct product = DJISampleApplication.getProductInstance(); |
|  |  |
|  | if (product == null || !product.isConnected()) { |
|  | ToastUtils.setResultToToast("Disconnect"); |
|  | missionControl = null; |
|  | return; |
|  | } else { |
|  | missionControl = MissionControl.getInstance(); |
|  | if (product instanceof Aircraft) { |
|  | flightController = ((Aircraft) product).getFlightController(); |
|  | } |
|  | } |
|  | } |
|  |  |
|  | private void initUI(Context context) { |
|  | setClickable(true); |
|  | LayoutInflater layoutInflater = (LayoutInflater) context.getSystemService(Service.LAYOUT\_INFLATER\_SERVICE); |
|  | layoutInflater.inflate(R.layout.view\_timeline, this, true); |
|  |  |
|  | timelineInfoTV = (TextView) findViewById(R.id.tv\_timeline\_info); |
|  | runningInfoTV = (TextView) findViewById(R.id.tv\_running\_info); |
|  | prepareBtn = (Button) findViewById(R.id.btn\_timeline\_init); |
|  | startBtn = (Button) findViewById(R.id.btn\_timeline\_start); |
|  | stopBtn = (Button) findViewById(R.id.btn\_timeline\_stop); |
|  | pauseBtn = (Button) findViewById(R.id.btn\_timeline\_pause); |
|  | resumeBtn = (Button) findViewById(R.id.btn\_timeline\_resume); |
|  | cleanBtn = (Button) findViewById(R.id.btn\_timeline\_clean); |
|  |  |
|  | prepareBtn.setOnClickListener(this); |
|  | startBtn.setOnClickListener(this); |
|  | stopBtn.setOnClickListener(this); |
|  | pauseBtn.setOnClickListener(this); |
|  | resumeBtn.setOnClickListener(this); |
|  | cleanBtn.setOnClickListener(this); |
|  | } |
|  |  |
|  | @Override |
|  | public void onClick(View v) { |
|  |  |
|  | if (DJISampleApplication.getProductInstance() instanceof Aircraft && !GeneralUtils.checkGpsCoordinate( |
|  | homeLatitude, |
|  | homeLongitude) && flightController != null) { |
|  | final CountDownLatch cdl = new CountDownLatch(1); |
|  | flightController.getHomeLocation(new CommonCallbacks.CompletionCallbackWith<LocationCoordinate2D>() { |
|  | @Override |
|  | public void onSuccess(LocationCoordinate2D locationCoordinate2D) { |
|  | homeLatitude = locationCoordinate2D.getLatitude(); |
|  | homeLongitude = locationCoordinate2D.getLongitude(); |
|  | setTimelinePlanToText("home point latitude: " |
|  | + homeLatitude |
|  | + "\nhome point longitude: " |
|  | + homeLongitude); |
|  | } |
|  |  |
|  | @Override |
|  | public void onFailure(DJIError djiError) { |
|  | cdl.countDown(); |
|  | } |
|  | }); |
|  | try { |
|  | cdl.await(500, TimeUnit.MILLISECONDS); |
|  | } catch (InterruptedException e) { |
|  | e.printStackTrace(); |
|  | } |
|  |  |
|  | if (!GeneralUtils.checkGpsCoordinate(homeLatitude, homeLongitude)) { |
|  | ToastUtils.setResultToToast("Home coordinates not yet set..."); |
|  | return; |
|  | } |
|  | } |
|  | switch (v.getId()) { |
|  | case R.id.btn\_timeline\_init: |
|  | initTimeline(); |
|  | break; |
|  | case R.id.btn\_timeline\_start: |
|  | startTimeline(); |
|  | break; |
|  | case R.id.btn\_timeline\_stop: |
|  | stopTimeline(); |
|  | break; |
|  | case R.id.btn\_timeline\_pause: |
|  | ToastUtils.setResultToToast("Timeline just supports the pause on the pausable elements, such as hotpoint mission, waypoint mission"); |
|  | pauseTimeline(); |
|  | break; |
|  | case R.id.btn\_timeline\_resume: |
|  | ToastUtils.setResultToToast("Timeline just supports the resume on the pausable elements, such as hotpoint mission, waypoint mission"); |
|  | resumeTimeline(); |
|  | break; |
|  | case R.id.btn\_timeline\_clean: |
|  | cleanTimelineDataAndLog(); |
|  | break; |
|  | default: |
|  | break; |
|  | } |
|  | } |
|  |  |
|  | @Override |
|  | public int getDescription() { |
|  | return R.string.component\_listview\_timeline\_mission\_control; |
|  | } |
|  |  |
|  | @NonNull |
|  | @Override |
|  | public String getHint() { |
|  | return this.getClass().getSimpleName() + ".java"; |
|  | } |
|  | } |

## 二、主要API说明：

### 1、 FlightController类

|  |
| --- |
| class FlightController extends BaseComponent |

|  |  |
| --- | --- |
| *Package:* | dji.sdk.flightcontroller |
| *Inherits From:* | [BaseComponent](https://developer.dji.com/cn/api-reference/android-api/Components/BaseComponent/DJIBaseComponent.html#djibasecomponent) |

##### *Description:*

此类包含飞行控制器组件，并提供不同的命令发送到飞行控制器的方法。此对象是可从飞机对象这是 BaseProduct 的一个子类。

###### method getHomeLocation

|  |
| --- |
| void getHomeLocation(@NonNull CompletionCallbackWith<LocationCoordinate2D> callback) |

|  |  |
| --- | --- |
| *Package:* | dji.sdk.flightcontroller |
| *SDK Key:* | [FlightControllerKey.HOME\_LOCATION](https://developer.dji.com/cn/api-reference/android-api/Components/KeyManager/DJIFlightControllerKey.html#flightcontrollerkey_home_location_inline) |

##### *Description:得到飞机的原点。*

##### *Input Parameters:*

|  |  |
| --- | --- |
| @NonNull [CompletionCallbackWith](https://developer.dji.com/cn/api-reference/android-api/Utils/DJICommonCallbacks.html#djicommoncallbacks_completioncallbackwith_interface)<[LocationCoordinate2D](https://developer.dji.com/cn/api-reference/android-api/Components/FlightController/DJIFlightController_DJILocationCoordinate2D.html#djiflightcontroller_djilocationcoordinate2d)> callback | *返回值的执行回调（s）。* |

### 2、 CommonCallbacks类

|  |
| --- |
| class CommonCallbacks |

|  |  |
| --- | --- |
| *Package:* | dji.common.util |

##### *Description:*

##### 用于返回异步操作结果常见的回调接口。

##### *Class Members:*

##### *Related:*

interface

[CompletionCallback](https://developer.dji.com/cn/api-reference/android-api/Utils/DJICommonCallbacks.html#djicommoncallbacks_completioncallback_interface_inline)

###### interface CompletionCallback

|  |
| --- |
| interface CompletionCallback |

|  |  |
| --- | --- |
| *Package:* | dji.common.util |

##### *Description:*

##### 不需要从操作返回信息时完成异步操作的完成回调。

##### *Interface Methods:*

#### Callback Interfaces

method

[onResult](https://developer.dji.com/cn/api-reference/android-api/Utils/DJICommonCallbacks.html#djicommoncallbacks_completioncallback_interface_onresult_inline)

###### method onResult

|  |
| --- |
| void onResult(DJIError error) |

|  |  |
| --- | --- |
| *Package:* | dji.common.util |

##### *Description:*

##### 当异步操作完成时调用。如果操作成功完成，错误将为null。在自己的代码中重载处理。

##### *Input Parameters:*

|  |  |
| --- | --- |
| [DJIError](https://developer.dji.com/cn/api-reference/android-api/Components/SDKError/DJIError.html#djierror) error | *The DJI error result* |

interface

[CompletionCallbackWith](https://developer.dji.com/cn/api-reference/android-api/Utils/DJICommonCallbacks.html#djicommoncallbacks_completioncallbackwith_interface_inline)

###### interface CompletionCallbackWith

|  |
| --- |
| interface CompletionCallbackWith<T> |

|  |  |
| --- | --- |
| *Package:* | dji.common.util |

##### *Description:*当需要一个参数（t）的信息需要从操作中返回时，异步操作的完成回调。参数t将在操作调用的地方定义。

##### *Interface Methods:*

#### Callback Interfaces

method

[onSuccess](https://developer.dji.com/cn/api-reference/android-api/Utils/DJICommonCallbacks.html#djicommoncallbacks_completioncallbackwith_interface_onsuccess_inline)

###### method onSuccess

|  |
| --- |
| void onSuccess(T t) |

|  |  |
| --- | --- |
| *Package:* | dji.common.util |

##### *Description:*

##### 当异步操作成功完成时调用。在自己的代码中重载处理。

##### *Input Parameters:*

|  |  |
| --- | --- |
| T t | *The parameter being returned.* |

method

[onFailure](https://developer.dji.com/cn/api-reference/android-api/Utils/DJICommonCallbacks.html#djicommoncallbacks_completioncallbackwith_interface_onfailure_inline)

###### method onFailure

|  |
| --- |
| void onFailure(DJIError error) |

|  |  |
| --- | --- |
| *Package:* | dji.common.util |

##### *Description:*

##### 当异步操作不能正常完成时调用。在自己的代码中重载处理。

##### *Input Parameters:*

|  |  |
| --- | --- |
| [DJIError](https://developer.dji.com/cn/api-reference/android-api/Components/SDKError/DJIError.html#djierror) error | *The DJI error result* |

interface

[CompletionCallbackWithTwoParam](https://developer.dji.com/cn/api-reference/android-api/Utils/DJICommonCallbacks.html#djicommoncallbacks_completioncallbackwithtwoparam_interface_inline)

###### interface CompletionCallbackWithTwoParam

|  |
| --- |
| interface CompletionCallbackWithTwoParam<X, Y> |

|  |  |
| --- | --- |
| *Package:* | dji.common.util |

##### *Description:*当需要两个参数（x，y）的信息需要从操作返回时，异步操作的完成回调。参数x和y将在调用操作的地方定义

##### *Interface Methods:*

#### Callback Interfaces

method

[onSuccess](https://developer.dji.com/cn/api-reference/android-api/Utils/DJICommonCallbacks.html#djicommoncallbacks_completioncallbackwithtwoparam_interface_onsuccess_inline)

method

[onFailure](https://developer.dji.com/cn/api-reference/android-api/Utils/DJICommonCallbacks.html#djicommoncallbacks_completioncallbackwithtwoparam_interface_onfailure_inline)

###### method onFailure

|  |
| --- |
| void onFailure(DJIError error) |

|  |  |
| --- | --- |
| *Package:* | dji.common.util |

##### *Description:*

##### 当异步操作不能正常完成时调用。在自己的代码中重载处理。

##### *Input Parameters:*

|  |  |
| --- | --- |
| [DJIError](https://developer.dji.com/cn/api-reference/android-api/Components/SDKError/DJIError.html#djierror) error | *The DJI error result* |

### 3、LocationCoordinate2D类

|  |
| --- |
| class LocationCoordinate2D |

|  |  |
| --- | --- |
| *Package:* | dji.common.model |

##### *Description:*

##### 表示二维位置坐标的类

##### *Class Members:*

Constructor

method

[LocationCoordinate2D](https://developer.dji.com/cn/api-reference/android-api/Components/FlightController/DJIFlightController_DJILocationCoordinate2D.html#djiflightcontroller_djilocationcoordinate2d_constructor_inline)

###### method LocationCoordinate2D

|  |
| --- |
| LocationCoordinate2D(double latitude, double longitude) |

|  |  |
| --- | --- |
| *Package:* | dji.common.model |

##### *Description:*

##### 构建一个djilocationcoordinate2d对象的经度和纬度。

##### *Input Parameters:*

|  |  |
| --- | --- |
| double latitude | *Latitude in degrees.* |
| double longitude | *Longitude in degrees.* |

Latitude

method

[getLatitude](https://developer.dji.com/cn/api-reference/android-api/Components/FlightController/DJIFlightController_DJILocationCoordinate2D.html#djiflightcontroller_djilocationcoordinate2d_getlatitude_inline)

###### method getLatitude

|  |
| --- |
| double getLatitude() |

|  |  |
| --- | --- |
| *Package:* | dji.common.model |

##### *Description:*

##### 返回维度。

##### *Return:*

|  |  |
| --- | --- |
| double | *The double value of latitude.* |

Longitude

method

[getLongitude](https://developer.dji.com/cn/api-reference/android-api/Components/FlightController/DJIFlightController_DJILocationCoordinate2D.html#djiflightcontroller_djilocationcoordinate2d_getlongitude_inline)

###### method getLongitude

|  |
| --- |
| double getLongitude() |

|  |  |
| --- | --- |
| *Package:* | dji.common.model |

##### *Description:*

返回经度。

##### *Return:*

|  |  |
| --- | --- |
| double | *The double value of longitude.* |

### 4、 Aircraft类

|  |
| --- |
| class Aircraft extends BaseProduct |

|  |  |
| --- | --- |
| *Package:* | dji.sdk.products |
| *Inherits From:* | [BaseProduct](https://developer.dji.com/cn/api-reference/android-api/BaseClasses/DJIBaseProduct.html#djibaseproduct) |

##### *Description:*飞机产品类，包括基本产品信息和对所有部件的访问（如飞行控制器、电池等）。这个对象是在djisdkmanager getProduct访问。

##### *Class Members:*

Components

method

[getFlightController](https://developer.dji.com/cn/api-reference/android-api/Products/Aircraft/DJIAircraft.html?search=getflightcontroller&i=0&#djiaircraft_flightcontroller_inline)

###### method getFlightController

|  |
| --- |
| FlightController getFlightController() |

|  |  |
| --- | --- |
| *Package:* | dji.sdk.products |

##### *Description:*返回飞机飞行控制器的一个实例。

##### *Return:*

|  |  |
| --- | --- |
| [FlightController](https://developer.dji.com/cn/api-reference/android-api/Components/FlightController/DJIFlightController.html#djiflightcontroller) | *DJIFlightController component.* |

### 5、FlightControllerState类

|  |
| --- |
| class FlightControllerState |

|  |  |
| --- | --- |
| *Package:* | dji.common.flightcontroller |

##### *Description:*

此类表示飞行控制器的当前状态。

###### method getAircraftLocation

|  |
| --- |
| LocationCoordinate3D getAircraftLocation() |

|  |  |
| --- | --- |
| *Package:* | dji.common.flightcontroller |
| *SDK Key:* | [FlightControllerKey.AIRCRAFT\_LOCATION\_LATITUDE](https://developer.dji.com/cn/api-reference/android-api/Components/KeyManager/DJIFlightControllerKey.html#flightcontrollerkey_aircraft_location_latitude_inline), [FlightControllerKey.AIRCRAFT\_LOCATION\_LONGITUDE](https://developer.dji.com/cn/api-reference/android-api/Components/KeyManager/DJIFlightControllerKey.html#flightcontrollerkey_aircraft_location_longitude_inline),[FlightControllerKey.ALTITUDE](https://developer.dji.com/cn/api-reference/android-api/Components/KeyManager/DJIFlightControllerKey.html#flightcontrollerkey_altitude_inline) |

##### *Description:*获取作为坐标的飞机当前位置。如果位置无效，则为0。

##### *Return:*

|  |  |
| --- | --- |
| [LocationCoordinate3D](https://developer.dji.com/cn/api-reference/android-api/Components/FlightController/DJIFlightController_DJILocationCoordinate3D.html#djiflightcontroller_djilocationcoordinate3d) | *飞机作为坐标的当前位置。* |

###### method getFlightMode

|  |
| --- |
| FlightMode getFlightMode() |

|  |  |
| --- | --- |
| *Package:* | dji.common.flightcontroller |
| *SDK Key:* | [FlightControllerKey.FLIGHT\_MODE](https://developer.dji.com/cn/api-reference/android-api/Components/KeyManager/DJIFlightControllerKey.html#flightcontrollerkey_flight_mode_inline) |

##### *Description:*

飞行控制器飞行模式。更多信息，见http://wiki.dji.com/en/index.php/Phantom\_3\_Professional-Aircraft

##### *Return:*

|  |  |
| --- | --- |
| [FlightMode](https://developer.dji.com/cn/api-reference/android-api/Components/FlightController/DJIFlightController_DJIFlightControllerCurrectState.html#djiflightcontroller_djiflightcontrollerflightmode) | *An enum value of the*[*FlightMode*](https://developer.dji.com/cn/api-reference/android-api/Components/FlightController/DJIFlightController_DJIFlightControllerCurrectState.html#djiflightcontroller_djiflightcontrollerflightmode)*.* |

### 6、LocationCoordinate3D类

|  |
| --- |
| class LocationCoordinate3D |

|  |  |
| --- | --- |
| *Package:* | dji.common.flightcontroller |

##### *Description:*这是一个显示位置、纬度、经度、高度的结构。

##### *Class Members:*

Get Latitude

method

[LocationCoordinate3D](https://developer.dji.com/cn/api-reference/android-api/Components/FlightController/DJIFlightController_DJILocationCoordinate3D.html#djiflightcontroller_djilocationcoordinate3d_constructor_inline)

###### method LocationCoordinate3D

|  |
| --- |
| LocationCoordinate3D(double latitude, double longitude, float altitude) |

|  |  |
| --- | --- |
| *Package:* | dji.common.flightcontroller |

##### *Description:*构造纬度、经度和高度的三维坐标。

##### *Input Parameters:*

|  |  |
| --- | --- |
| double latitude | *纬度* |
| double longitude | *经度* |
| float altitude | *高度* |

method

[getLatitude](https://developer.dji.com/cn/api-reference/android-api/Components/FlightController/DJIFlightController_DJILocationCoordinate3D.html#djiflightcontroller_djilocationcoordinate3d_getlatitude_inline)

###### method getLatitude

|  |
| --- |
| double getLatitude() |

|  |  |
| --- | --- |
| *Package:* | dji.common.flightcontroller |

##### *Description:*

返回纬度。

##### *Return:*

|  |  |
| --- | --- |
| double | *A double value of the latitude.* |

Get Longitude

method

[getLongitude](https://developer.dji.com/cn/api-reference/android-api/Components/FlightController/DJIFlightController_DJILocationCoordinate3D.html#djiflightcontroller_djilocationcoordinate3d_getlongitude_inline)

###### method getLongitude

|  |
| --- |
| double getLongitude() |

|  |  |
| --- | --- |
| *Package:* | dji.common.flightcontroller |

##### *Description:*

返回经度。

##### *Return:*

|  |  |
| --- | --- |
| double | *A double value of the longitude.* |

Get Altitude

method

[getAltitude](https://developer.dji.com/cn/api-reference/android-api/Components/FlightController/DJIFlightController_DJILocationCoordinate3D.html#djiflightcontroller_djilocationcoordinate3d_getaltitude_inline)

###### method getAltitude

|  |
| --- |
| float getAltitude() |

|  |  |
| --- | --- |
| *Package:* | dji.common.flightcontroller |

##### *Description:*

##### 返回飞机相对于起飞位置的相对高度，用气压计测量。

##### *Return:*

|  |  |
| --- | --- |
| float | *A float value of the relative altitude of the aircraft relative to take off location.* |

### 7、MissionControl类

|  |
| --- |
| class MissionControl implements TimelineElementFeedback |

|  |  |
| --- | --- |
| *Package:* | dji.sdk.mission |

##### *Description:*

任务控制实际执行的任务。无论是单任务可以运行在专用的使命运营商，或一系列任务和行动可以运行连续使用时间表。  
  
Mission操作负责执行一项任务。一个任务操作有API来控制任务（例如启动、停止、暂停等），并通过侦听器提供任务状态。  
  
任务控制的时间轴是由一系列的时间元素可以点任务，热点任务（通过热点行动hotpointaction）或遵循timelineelement协议对象。missionaction是时间元素，withaircraftyawaction和gimbalattitudeaction子类是具体的行动的例子。如果一个任务在一个时间线上执行，任务操作员仍然可以用来获取任务的状态信息以及改变任务执行参数。  
  
时间线元素也可以有触发器（触发器）与它们关联。触发器是与时间轴元素并行运行的独立对象。触发器对象用于在满足一组标准时启动动作。例如，一个触发器可以用来监视电池级别，然后在阈值通过后启动一个动作。

###### method getHotpointMissionOperator

|  |
| --- |
| @NonNull  HotpointMissionOperator getHotpointMissionOperator() |

|  |  |
| --- | --- |
| *Package:* | dji.sdk.mission |

##### *Description:*

##### 返回操作热点任务。

##### *Return:*

|  |  |
| --- | --- |
| [HotpointMissionOperator](https://developer.dji.com/cn/api-reference/android-api/Components/Missions/DJIHotpointMissionOperator.html#djihotpointmissionoperator) | *A Hotpoint mission operator object.* |

###### method getCurrentState

|  |
| --- |
| @NonNull  HotpointMissionState getCurrentState() |

|  |  |
| --- | --- |
| *Package:* | dji.sdk.mission.hotpoint |

##### *Description:*

##### 执行任务的热点执行当前的状态。

##### *Return:*

|  |  |
| --- | --- |
| [HotpointMissionState](https://developer.dji.com/cn/api-reference/android-api/Components/Missions/DJIHotpointMissionState.html#djihotpointmissionstate) | *The state of the Hotpoint mission.* |

###### method addListener

|  |
| --- |
| void addListener(@NonNull HotpointMissionOperatorListener listener) |

|  |  |
| --- | --- |
| *Package:* | dji.sdk.mission.hotpoint |

##### *Description:*

##### 添加监听器接收所有的热点事件任务操作。

##### *Input Parameters:*

|  |  |
| --- | --- |
| @NonNull [HotpointMissionOperatorListener](https://developer.dji.com/cn/api-reference/android-api/Components/Missions/DJIHotpointMissionOperator.html#djihotpointmissionoperatorlistener) listener | *监听在热点任务操作。* |

### 8、HotpointMissionState类

|  |
| --- |
| class HotpointMissionState extends MissionState |

|  |  |
| --- | --- |
| *Package:* | dji.common.mission.hotpoint |
| *Inherits From:* | MissionState |

##### *Description:*

热点任务操作的状态。

##### *Class Members:*

#### Members

Value

final

[UNKNOWN](https://developer.dji.com/cn/api-reference/android-api/Components/Missions/DJIHotpointMissionState.html#djihotpointmissionstate_unknown_inline)

###### final UNKNOWN

|  |
| --- |
| static final HotpointMissionState UNKNOWN = new HotpointMissionState("UNKNOWN") |

|  |  |
| --- | --- |
| *Package:* | dji.common.mission.hotpoint |

##### *Description:*

操作的状态未知。这是刚创建操作员时的初始状态。

final

[DISCONNECTED](https://developer.dji.com/cn/api-reference/android-api/Components/Missions/DJIHotpointMissionState.html#djihotpointmissionstate_disconnected_inline)

###### final DISCONNECTED

|  |
| --- |
| static final HotpointMissionState DISCONNECTED = new HotpointMissionState("DISCONNECTED") |

|  |  |
| --- | --- |
| *Package:* | dji.common.mission.hotpoint |

##### *Description:*

移动设备和飞机之间的连接断了。

final

[RECOVERING](https://developer.dji.com/cn/api-reference/android-api/Components/Missions/DJIHotpointMissionState.html#djihotpointmissionstate_recovering_inline)

###### final RECOVERING

|  |
| --- |
| static final HotpointMissionState RECOVERING = new HotpointMissionState("RECOVERING") |

|  |  |
| --- | --- |
| *Package:* | dji.common.mission.hotpoint |

##### *Description:*

移动设备和飞机之间的连接正在恢复中。此时，操作员正在从飞机上同步状态。

final

[NOT\_SUPPORTED](https://developer.dji.com/cn/api-reference/android-api/Components/Missions/DJIHotpointMissionState.html#djihotpointmissionstate_notsupported_inline)

###### final NOT\_SUPPORTED

|  |
| --- |
| static final HotpointMissionState NOT\_SUPPORTED = new HotpointMissionState("NOT\_SUPPORTED") |

|  |  |
| --- | --- |
| *Package:* | dji.common.mission.hotpoint |

##### *Description:*

连接产品不支持热点的使命。

final

[READY\_TO\_EXECUTE](https://developer.dji.com/cn/api-reference/android-api/Components/Missions/DJIHotpointMissionState.html#djihotpointmissionstate_readytoexecute_inline)

###### final READY\_TO\_EXECUTE

|  |
| --- |
| static final HotpointMissionState READY\_TO\_EXECUTE = new HotpointMissionState("READY\_TO\_EXECUTE") |

|  |  |
| --- | --- |
| *Package:* | dji.common.mission.hotpoint |

##### *Description:*

任务完全上传，飞机准备开始执行。

final

[INITIAL\_PHASE](https://developer.dji.com/cn/api-reference/android-api/Components/Missions/DJIHotpointMissionState.html#djihotpointmissionstate_initialphase_inline)

###### final INITIAL\_PHASE

|  |
| --- |
| static final HotpointMissionState INITIAL\_PHASE = new HotpointMissionState("INITIAL\_PHASE") |

|  |  |
| --- | --- |
| *Package:* | dji.common.mission.hotpoint |

##### *Description:*

飞机朝着热点任务的开始点。

final

[EXECUTING](https://developer.dji.com/cn/api-reference/android-api/Components/Missions/DJIHotpointMissionState.html#djihotpointmissionstate_executing_inline)

###### final EXECUTING

|  |
| --- |
| static final HotpointMissionState EXECUTING = new HotpointMissionState("EXECUTING") |

|  |  |
| --- | --- |
| *Package:* | dji.common.mission.hotpoint |

##### *Description:*

执行成功启动。

final

[EXECUTION\_PAUSED](https://developer.dji.com/cn/api-reference/android-api/Components/Missions/DJIHotpointMissionState.html#djihotpointmissionstate_executionpaused_inline)

###### final EXECUTION\_PAUSED

|  |
| --- |
| static final HotpointMissionState EXECUTION\_PAUSED = new HotpointMissionState("EXECUTION\_PAUSED") |

|  |  |
| --- | --- |
| *Package:* | dji.common.mission.hotpoint |

##### *Description:*热点的使命是成功暂停。用户可以调用简历继续执行。

### 9、HotpointMissionOperator类

|  |
| --- |
| class HotpointMissionOperator |

|  |  |
| --- | --- |
| *Package:* | dji.sdk.mission.hotpoint |

##### *Description:*

##### 热点任务操作控制的唯一对象，运行和监控热点任务。它可以访问frommissioncontrol。*Class Members:*

Current State

method

[getCurrentState](https://developer.dji.com/cn/api-reference/android-api/Components/Missions/DJIHotpointMissionOperator.html#djihotpointmissionoperator_currentstate_inline)

###### method getCurrentState

|  |
| --- |
| @NonNull  HotpointMissionState getCurrentState() |

|  |  |
| --- | --- |
| *Package:* | dji.sdk.mission.hotpoint |

##### *Description:*

##### 执行热点执行任务的当前状态。

##### *Return:*

|  |  |
| --- | --- |
| [HotpointMissionState](https://developer.dji.com/cn/api-reference/android-api/Components/Missions/DJIHotpointMissionState.html#djihotpointmissionstate) | *The state of the Hotpoint mission.* |

### 10、HotpointMission类

|  |
| --- |
| class HotpointMission |

|  |  |
| --- | --- |
| *Package:* | dji.common.mission.hotpoint |

##### *Description:*

##### 这类代表一个热点的使命。在一个热点的使命，飞机多次飞绕指定点恒定半径的圆称为热点。用户可以控制飞机绕飞与一个特定的半径和高度关注的焦点。在执行过程中，用户还可以使用物理遥控器来修改其半径和速度。它不是由Mavic Pro使用WiFi连接时支持。*Class Members:*

Coordinate

method

[setHotpoint](https://developer.dji.com/cn/api-reference/android-api/Components/Missions/DJIHotPointMission.html?search=hotpointheading&i=0&#djihotpointmission_hotpoint_inline)

###### method setHotpoint

|  |
| --- |
| void setHotpoint(LocationCoordinate2D hotpoint) |

|  |  |
| --- | --- |
| *Package:* | dji.common.mission.hotpoint |

##### *Description:*

##### 设置热点坐标。

##### *Input Parameters:*

|  |  |
| --- | --- |
| [LocationCoordinate2D](https://developer.dji.com/cn/api-reference/android-api/Components/FlightController/DJIFlightController_DJILocationCoordinate2D.html#djiflightcontroller_djilocationcoordinate2d) hotpoint | *A LocationCoordinate2D object of hotpoint.* |

method

[getHotpoint](https://developer.dji.com/cn/api-reference/android-api/Components/Missions/DJIHotPointMission.html?search=hotpointheading&i=0&#djihotpointmission_gethotpoint_inline)

###### method getHotpoint

|  |
| --- |
| LocationCoordinate2D getHotpoint() |

|  |  |
| --- | --- |
| *Package:* | dji.common.mission.hotpoint |

##### *Description:*

##### 获取热点的坐标。

##### *Return:*

|  |  |
| --- | --- |
| [LocationCoordinate2D](https://developer.dji.com/cn/api-reference/android-api/Components/FlightController/DJIFlightController_DJILocationCoordinate2D.html#djiflightcontroller_djilocationcoordinate2d) | *A object of*[*LocationCoordinate2D*](https://developer.dji.com/cn/api-reference/android-api/Components/FlightController/DJIFlightController_DJILocationCoordinate2D.html#djiflightcontroller_djilocationcoordinate2d)*.* |

Start Point

method

[setStartPoint](https://developer.dji.com/cn/api-reference/android-api/Components/Missions/DJIHotPointMission.html?search=hotpointheading&i=0&#djihotpointmission_startpoint_inline)

###### method setStartPoint

|  |
| --- |
| void setStartPoint(HotpointStartPoint startPoint) |

|  |  |
| --- | --- |
| *Package:* | dji.common.mission.hotpoint |

##### *Description:*

##### 在开始热点任务时将飞机的初始点设置在圆形飞行路径上。

##### *Input Parameters:*

|  |  |
| --- | --- |
| [HotpointStartPoint](https://developer.dji.com/cn/api-reference/android-api/Components/Missions/DJIHotPointMission.html#djihotpointmission_djihotpointstartpoint) startPoint | *An enum value of*[*HotpointStartPoint*](https://developer.dji.com/cn/api-reference/android-api/Components/Missions/DJIHotPointMission.html#djihotpointmission_djihotpointstartpoint)*.* |

method

[getStartPoint](https://developer.dji.com/cn/api-reference/android-api/Components/Missions/DJIHotPointMission.html?search=hotpointheading&i=0&#djihotpointmission_getstartpoint_inline)

###### method getStartPoint

|  |
| --- |
| HotpointStartPoint getStartPoint() |

|  |  |
| --- | --- |
| *Package:* | dji.common.mission.hotpoint |

##### *Description:*

##### 在开始热点任务时，在环形飞行路径上获取飞机的初始点。

##### *Return:*

|  |  |
| --- | --- |
| [HotpointStartPoint](https://developer.dji.com/cn/api-reference/android-api/Components/Missions/DJIHotPointMission.html#djihotpointmission_djihotpointstartpoint) | *An enum value of*[*HotpointStartPoint*](https://developer.dji.com/cn/api-reference/android-api/Components/Missions/DJIHotPointMission.html#djihotpointmission_djihotpointstartpoint)*.* |