蓝牙手柄控制

场景案例：

在某舞台现场，通过游戏手柄的摇杆来控制Pepper不同方位的移动或旋转以便让Pepper能“走”到舞台的正确位置上，继而开始主持或表演娱乐节目。

项目简介：

项目内使用蓝牙为手柄和Pepper机器人提供通信，在开始使用之前，请先确保蓝牙输入设备已与Pepper的蓝牙已保持连接（在Pepper平板里打开蓝牙并主动连接输入设备）。并且，关闭power hatch（充电口盖），否则 Pepper 无法移动。

您可直接clone项目在AndroidStudio内打开并在Pepper上直接运行它。

也可以参考下列步骤将该功能集成到您的项目里。

第一步：

将项目下pepper-gamepad这个Library导入您的项目。

第二步（确保您的项目已经是机器人项目）：

在onRobotFocusGained里禁用BasicAwareness，并实例化RemoteRobotController

```
override fun onRobotFocusGained(qiContext: QiContext) {
    val basicAwarenessHolder = HolderBuilder.with(qiContext)
        .withAutonomousAbilities(AutonomousAbilitiesType.BASIC_AWARENESS)
        .build()

    basicAwarenessHolder.async().hold().thenConsume {
        when {
            it.isSuccess -> Log.i(TAG, "BasicAwareness held with success")
            it.hasError() -> Log.e(TAG, "holdBasicAwareness error: " + it.errorMessage)
            it.isCancelled -> Log.e(TAG, "holdBasicAwareness cancelled")
        }
    }

    remoteRobotController = RemoteRobotController(qiContext)
}
```

第三步：

实现获取蓝牙输入设备传回来值的方法（左操作杆能够使Pepper进行不同方向的移动，右操作杆能够使Pepper进行旋转）。

```
private fun getCenteredAxis(event: MotionEvent, device: InputDevice, axis: Int): Float {
    val range: InputDevice.MotionRange? = device.getMotionRange(axis, event.source)

    // A joystick at rest does not always report an absolute position of
    // (0,0). Use the getFlat() method to determine the range of values bounding the joystick axis center.

    range?.apply {
        val value = event.getAxisValue(axis)

        // Ignore axis values that are within the 'flat' region of the joystick axis center.
        if (Math.abs(value) > flat) {
            return value
        }
    }
    return 0f
}
```

第四步：

重写Activity的onGenericMotionEvent方法，并通过MotionEvent的不同TAG来获取操作摇杆时传回来的值，并最终通过remoteRobotController.updateTarget来实现Pepper运动更新。

```
override fun onGenericMotionEvent(event: MotionEvent): Boolean {
    // Add null protection for when the controller disconnects
    val inputDevice = event.device ?: return super.onGenericMotionEvent(event)

    // Get left joystick coordinates
    val leftJoystickX = getCenteredAxis(event, inputDevice, MotionEvent.AXIS_X)
    val leftJoystickY = getCenteredAxis(event, inputDevice, MotionEvent.AXIS_Y)

    // Get right joystick coordinates
    val rightJoystickX = getCenteredAxis(event, inputDevice, MotionEvent.AXIS_Z)
    val rightJoystickY = getCenteredAxis(event, inputDevice, MotionEvent.AXIS_RZ)

    if (::remoteRobotController.isInitialized) {
        thread {
            remoteRobotController.updateTarget(leftJoystickX, leftJoystickY, rightJoystickX, rightJoystickY)
        }
    }
    return true
}
```