Gabby
android learner

深入理解Android Sensor系统 (4.0)

2013-08-20 10:03 539人阅读 评论(0) 收藏 举报

分类：
Q_SENSOR (9)

http://blog.csdn.net/qianjin0703/article/details/7568641

曾几何时，本人写了一篇Android传感器初探"惊艳整个篮球场"...一转眼两年过去了，真是物逝人非，技术更新的快啊，如今都已经4.0巧克力冰激凌了...

0. 总论

本文希望分别从动态角度（应用程序进程）以及静态角度（框架体系架构）两方面来理解传感器系统。

1. 上层应用

从编写应用程序的角度来看，比较简单，大体分如下4步，便可得到一个传感器实时上报的数值并作处理，

1) 得到传感器服务 getSystemService(SENSOR_SERVICE);

得到一个SensorManager，用来管理分配调度处理Sensor的工作，注意它并不服务运行于后台，真正属于Sensor的系统服务是SensorService，终端下#service list可以看到sensorservice: [android.gui.SensorServer]。

2) 得到传感器类型 getDefaultSensor(Sensor.TYPE_GRAVITY);

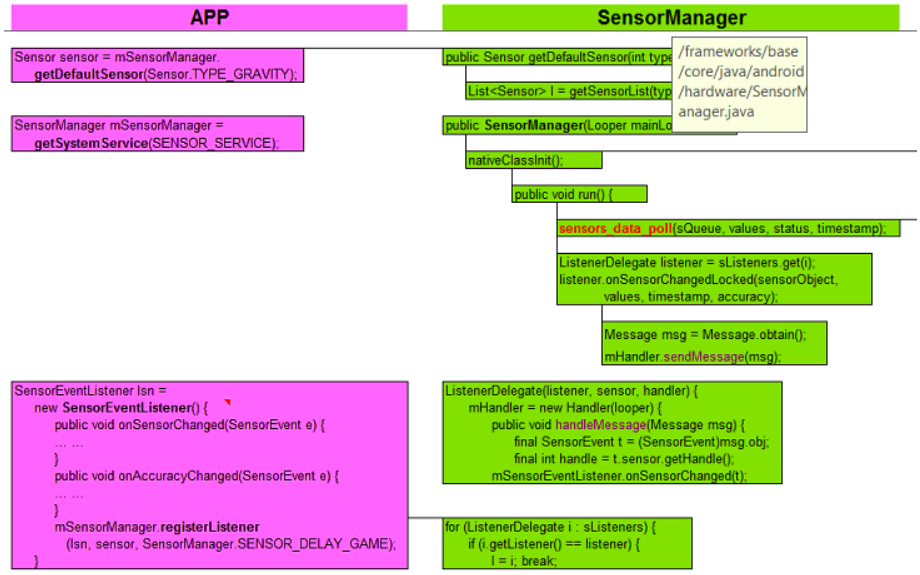
当然还有各种千奇百怪的传感器，可以查阅Android官网API或者源码Sensor.java。

3) 注册监听器 SensorEventListener

应用程序打开一个监听接口，专门处理传感器的数据，这个监听机制比较重要，被系统广泛使用。

4) 实现监听器的回调函数 onSensorChanged, onAccuracyChanged

例如对重力感应器的xyz值经算法变换得到左右上下前后方向等，就由这个回调函数实现。



1.1 进程空间

一个应用程序对应一个虚拟机实例，一个进程和一个主线程，通过主线程控制四大组件。

要想得到传感器数据，须要开一个子线程，通过pol轮训的方式监听底层上报的数据，再由消息机制把数据发送给主线程处理。

换句话说，应用程序有两个任务：

1）准备一个监听接口供主线程接受数据

2）开启一个子线程接受底层数据并发送到主线程，

而这些有关Sensor的具体任务，全部交给SensorManager统一管理。

LINUX_PRO (6)
OS_Install_Tips (12)
C++ (60)
HR (9)
health (1)
tv&&music&&fun (9)
compile (9)
并发处理相关 (8)
const&mutable (2)
callback_func (1)
ipc (1)
camera_framework (9)
sp&wp (3)
selinux (1)
initrc (3)
cmd (2)
virtual_func (2)
stl (1)
log (2)
mediaplayer (1)
parcel (1)
property (2)
debug (1)
busybox (1)
bionic (1)
zygote (2)
surface (2)
jni (1)
笔试题目 (3)
平台设备总线模型 (1)
iic (0)
字符驱动 (1)

文章存档

2016年04月 (32)
2016年03月 (6)
2016年02月 (13)
2015年12月 (43)
2015年11月 (9)

展开

阅读排行

请教：过量喝咖啡对心脏 (4385)
C++和JAVA的区别 -- 给初 (3934)
svn如何取消认证缓存设置 (3792)
android TP (2739)
Android LCD (2562)
高通平台android开发总结 (2280)
Android图形合成和显示系 (2258)
一个离职员工对中兴的回 (2237)
高通QPST Download使用 (2208)
android camera (2088)

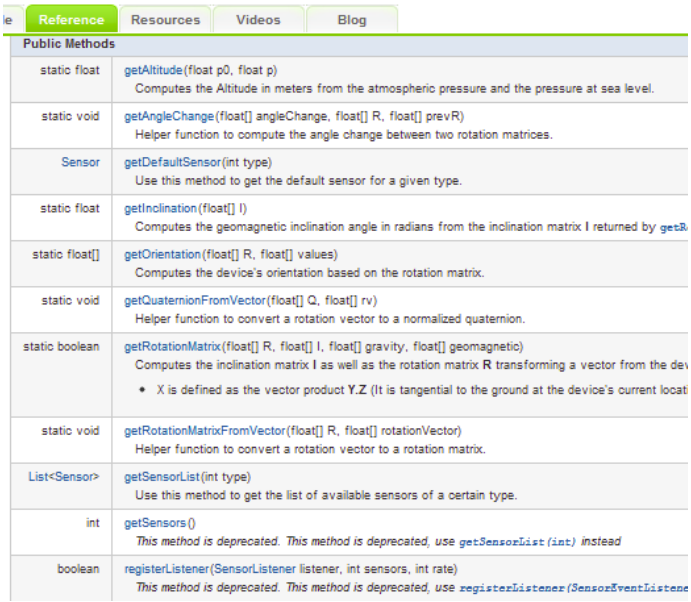
推荐文章

* CSDN日报20170828——《4个方法快速打造你的阅读清单》
* Android检查更新下载安装
* 动手打造史上最简单的Recycleview 侧滑菜单
* TCP网络通讯如何解决分包粘包问题
* 程序员的八重境界

1.2 Android SDK API

上层通过SDK API得到Java框架，这里，应用程序通过AIDL接口远程调用（RPC）得到SensorManager.

可查阅官网SDK API android.hardware.SensorManager类提供的方法。



Reference | Resources | Videos | Blog

Public Methods

static float	getAltitude(float p0, float p) Computes the Altitude in meters from the atmospheric pressure and the pressure at sea level.
static void	getAngleChange(float[] angleChange, float[] R, float[] prevR) Helper function to compute the angle change between two rotation matrices.
Sensor	getDefaultSensor(int type) Use this method to get the default sensor for a given type.
static float	getInclination(float[] I) Computes the geomagnetic inclination angle in radians from the inclination matrix I returned by getR
static float[]	getOrientation(float[] R, float[] values) Computes the device's orientation based on the rotation matrix.
static void	getQuaternionFromVector(float[] Q, float[] rv) Helper function to convert a rotation vector to a normalized quaternion.
static boolean	getRotationMatrix(float[] R, float[] I, float[] gravity, float[] geomagnetic) Computes the inclination matrix I as well as the rotation matrix R transforming a vector from the dev • X is defined as the vector product Y.Z (It is tangential to the ground at the device's current locat
static void	getRotationMatrixFromVector(float[] R, float[] rotationVector) Helper function to convert a rotation vector to a rotation matrix.
List<Sensor>	getSensorList(int type) Use this method to get the list of available sensors of a certain type.
int	getSensors() *This method is deprecated. This method is deprecated, use getSensorList(int) instead*
boolean	registerListener(SensorListener listener, int sensors, int rate) *This method is deprecated. This method is deprecated, use registerListener(SensorEventListene*

简单粗暴概括地讲，上层要拿底层数据，无外乎两部分，

1)设备的类型 (控制流)

getSensorList()得到传感器类型，取得sensorList列表的过程由上至下，java框架->JNI->本地框架 ->标准抽象层->设备驱动

2) 设备的数据 (数据流)

registerListener()注册监听器的时候会开启一个线程，其中sensor_data_poll同样由上至下最后拿到底层数据。

写到这里，摘抄别人的一段话...

我们在学习新系统时，首先映入眼帘的就是新概念。新名词，就如现在我们面临的Android大量的新名词，在程序员的世界都是从代码实践开始的，是从写应用开始去涉及。SDK给了我们一个概念，我们就在这个概念框架下，使用SDK给我提供的函数接口，数据结构，初始化过程等，我们最初的接触到原型就是"HelloWorld"之类的DEMO程序，我们在Hello world上去使用各种不同的接口函数，对于应用程序员来讲,他说看到的系统就是系统调用接口，及其编程开发流程。实际上只要一使用这些接口,就不得不接受一系列的概念,只有在这种概念系统下,我们才能工作。但是,实际上我们却忽略了这样的概念系统的理解,只是在编程接口的这个狭窄的空间去理解系统.我们理解系统在形成理解概念的空间只是微小的一角，很少有资料来介绍这种概念系统的形成和理解,编程接口只是这个概念空间一个，对外部的一个表征。我们可以抽象起来,以接口,协议和行为,来描述系统的情况。SDK API的实质向上层提供了一个语义接口，从而在层间实现了一个转义过程，同时又成为一个功能的集合体。但是我们很少这样跳出来看，我们到底是处于一种什么样的概念空间，SDK除了调用接口外，还给了我们怎样一种整体概念？目标系统的基本构架在本质上的东西就是一个概念系统到另一个概念系统的映射。让我们大脑理解的概念系统映射到计算机能实现的概念域的一个映射。我们假定这个概念域E,机器能够理解的概念域为M,我们的软件工程要做的事情实质就是：EàM领域的一个映射过程。

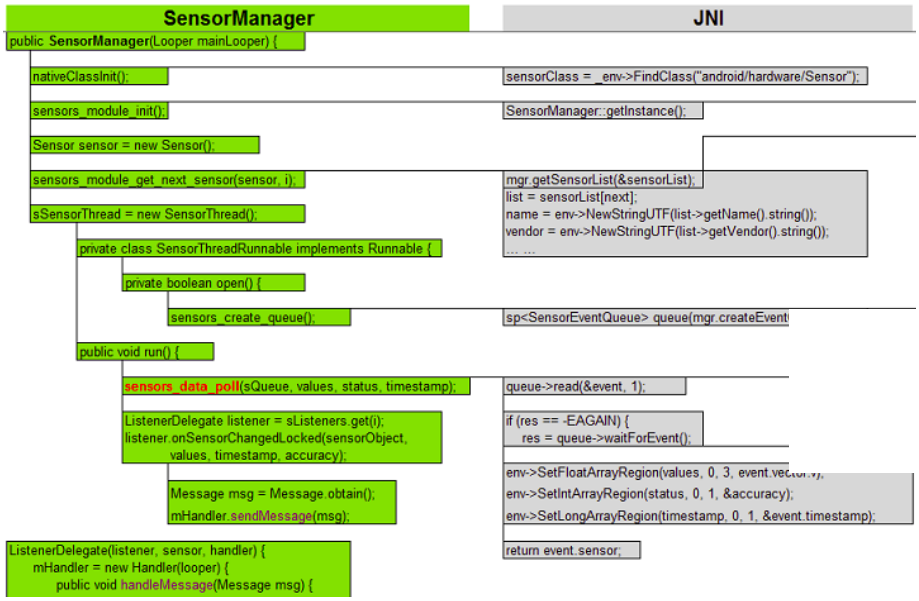
2 Java框架

上层由getSystemService(SENSOR_SERVICE);得到一个SensorManager实例，为上层提供方法。

除了上文的两个method，SensorManager本身的构造函数很有必要看下，

关闭

* 四大线程池详解



nativeClassInit(); 在JNI层得到android.hardware.Sensor的JNI环境指针env.

sensors_module_init(); 通过JNI调用本地框架，得到SensorService，SensorService初始化控制流各功能。

new Sensor(); 建立一个Sensor对象，可查阅官网API android.hardware.Sensor

sensors_module_get_next_sensor(); 上层得到设备支持的所有Sensor，并放入SensorList链表。

new SensorThread(); 创建Sensor线程，当应用程序registerListener()注册监听器的时候开启线程run()，注意当没有数据变化时，线程会阻塞。

站在上层的角度，我们看不到SensorManager做了两件重要的工作，即

1) 得到本地框架SensorManager.cpp，获得Sensor真正的后台服务SensorService

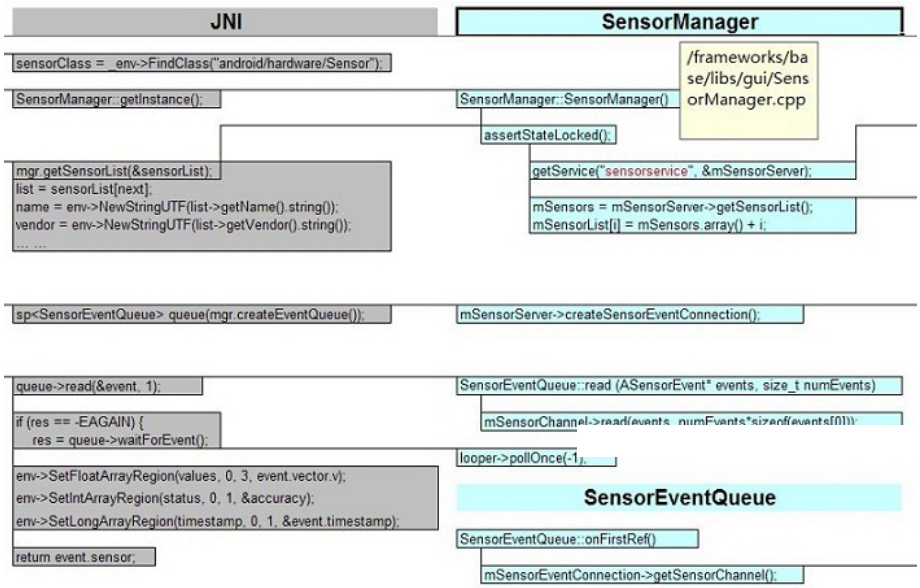
2) 创建EventQueue用来管理数据，这个是4.0新版本的一大变化，这个变化，导致抽象层有关数据流的内容全部重写...

3 本地框架

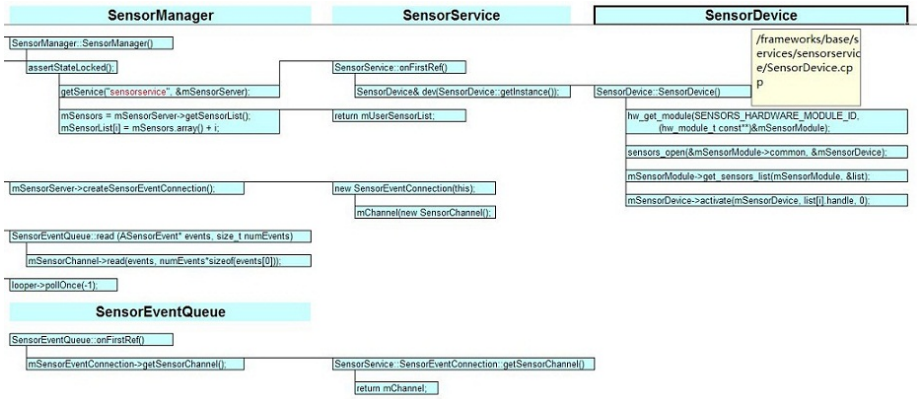
为了不陷入茫茫的代码海洋，这里简单的做个总结，

SensorManager负责控制流，通过C/S binder机制与SensorService通信。

SensorEventQueue负责数据流，通过管道机制读写底层数据。



关闭



SensorSevice把任务交给SensorDevice，SensorDevice调用标准的抽象层接口。
Sensor架构的抽象层接口是最标准的一种，它很好的实现了抽象层与本地框架的分离。
当然还有其它类型的抽象层接口，这里不作讨论。

顶 0 踩 0

上一篇 设备驱动外传 - 触摸屏的校正原理
下一篇 Android 输入系统

相关文章推荐

- 深入理解Android Sensor系统 (4.0)
- Android应用程序开发以及背后的设计思想深度剖析
- 轻松拿下Linux进程、线程和调度
- 一招学会Android自定义控件
- android4.0系统结构分析
- Android获取手机所有Sensor（传感器）并测试数...
- 30天掌握机器学习升级版
- 从零练就iOS高手
- Android开发-Sensor传感器-AndroidStudio(二)小方
- Android框架浅析之锁屏(Keyguard)机制原理
- Python网络爬虫快速入门实战
- Android 4.0系统自带图标
- android Sensor系统综述
- Android应用开发详解
- 最适合自学的C++基础知识
- android sensor hal