Университет ИТМО  
Мегафакультет компьютерных технологий и управления  
Кафедра вычислительной техники

Отчет по практике

Выполнила:

Островская Т.Б.  
студентка гр. P3211

Научный руководитель:  
Гаврилов А.В.

Санкт-Петербург

2016 год

1. Датчики

Датчик — [средство измерений](https://ru.wikipedia.org/wiki/%D0%A1%D1%80%D0%B5%D0%B4%D1%81%D1%82%D0%B2%D0%BE_%D0%B8%D0%B7%D0%BC%D0%B5%D1%80%D0%B5%D0%BD%D0%B8%D0%B9), предназначенное для выработки сигнала измерительной информации в форме, удобной для передачи, дальнейшего преобразования, обработки и хранения, но не поддающейся непосредственному восприятию наблюдателем. Датчики различных параметров окружающей среды (например, температуры, освещенности, ускорения в трех направлениях) устанавливаются в устройство, и с помощью определенных API разработчики приложений могут использовать полученную с датчиков информацию. Необходимость в использовании датчиков связана с тем, что в небольшое по размеру устройство невозможно установить приборы, измеряющие количественно какой-либо фактор окружающей среды. Датчики являются их аналогами, работающими по схожим принципам, но имеющими отличное от приборов строение и гораздо меньший размер.

2. Различия прибора и одноименного датчика на примере акселерометра

Акселерометр – это прибор, предназначенный для измерения кажущегося ускорения. Кажущееся ускорение – это разница между гравитационным и истинным ускорениями объекта. Принципиально акселерометр состоит из пружины, подвижной массы и демпфера. Пружина крепится к неподвижной поверхности, к пружине крепится масса. С другой стороны ее поддерживает демпфер, который гасит собственные вибрации груза. Во время ускорения массы деформируется пружина. На этих деформациях и основываются показания прибора. Три таких прибора, объединенные в одну систему и сориентированные по осям позволяют получать информацию о положении предмета в трехмерном пространстве.

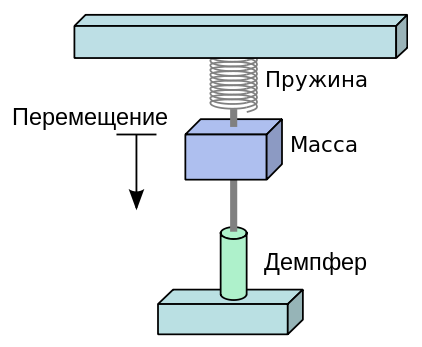


Рис. 1. Устройство акселерометра

Современные устройства имеют настолько небольшие размеры, что не могут содержать в себе три прибора-акселерометра. Решением этой проблемы стали компактные датчики-акселерометры, К неподвижному корпусу на упругих приставках, которые позволяют перемещение в определенных пределах, крепится перегородка с отведенными в сторону проводниками. Эти отводы размещаются между контактами, которые и снимают показания. При перемещении отводов напряженность поля вокруг контактов меняет свои характеристики, что и служит показателем для измерения. Важно отметить, что производить такие мелкие детали путем физической обработки материалов практически невозможно – их размеры слишком малы. Для производства этих устройств используются различные реакции силикона с другими веществами.

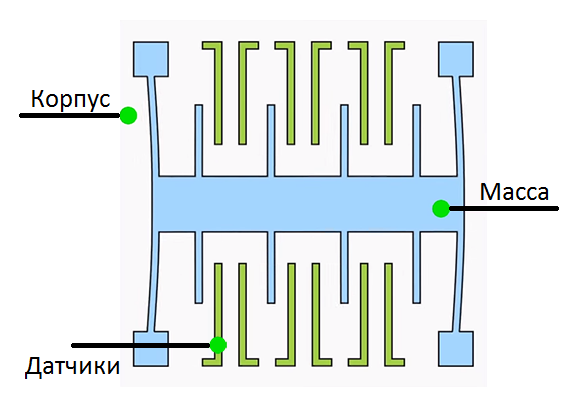


Рис. 2. Устройство датчика акселерометра

3. Датчики в ОС Android

Датчики ОС Android делятся на три категории: движения, положения и окружающей среды. Датчики эти могут быть самыми разными:

* Акселерометр
* Гироскоп
* Датчик освещения
* Датчик магнитных полей
* Акселерометр
* Барометр
* Датчик поднесения телефона к голове
* Датчик температуры аппарата
* Датчик температуры окружающей среды
* Измеритель относительной влажности

Естественно, их набор зависит от «комплектации» аппарата, но есть и датчики, присутствующие в большинстве смартфонов на Android — акселерометр и гироскоп.

За работу с датчиками (сенсорами) отвечает класс **SensorManager**, содержащий несколькоконстант, которые характеризуют различные аспекты системы датчиков Android, в том числе:

**Тип датчика**

Ориентация, акселерометр, свет, магнитное поле, близость, температура и т.д.

**Частота измерений**

Максимальная, для игр, обычная, для пользовательского интерфейса. Когда приложение запрашивает конкретное значение частоты отсчетов, с точки зрения сенсорной подсистемы это лишь рекомендация. Никакой гарантии, что измерения будут производиться с указанной частотой, нет.

**Точность**

Высокая, низкая, средняя, ненадежные данные.

Несколько констант класса SensorManager, характеризующие типы датчиков:

* **TYPE\_ACCELEROMETER** - Измеряет ускорение в пространстве по осям X, Y, Z
* **TYPE\_GRAVITY** - Трёхосевой датчик силы тяжести. Как правило, это виртуальный датчик и представляет собой низкочастотный фильтр для показаний, возвращаемых акселерометром
* **TYPE\_GYROSCOPE** - Трёхосевой гироскоп, возвращающий текущее положение устройства в пространстве в градусах по трём осям. По другим данным, возвращает скорость вращения устройства по трём осям в радианах в секунду.
* **TYPE\_LIGHT** - Измеряет степень освещенности. Датчик окружающей освещенности, который описывает внешнюю освещенность в люксах. Этот тип датчиков обычно используется для динамического изменения яркости экрана.
* **TYPE\_PRESSURE** - Датчик атмосферного давления (барометр), возвращающий текущее давление в миллибарах. Можно определять высоту над уровнем моря, путём сравнения атмосферного давления в двух точках.

Для получения показаний датчиков к каком-либо классе необходимо, чтобы он имплементировал интерфейс android.hardware.SensorListener, включающий в себя два необходимых метода:

* Метод **onSensorChanged(int sensor, float values[])** вызывается всякий раз, когда изменяется значение датчика. Этот метод вызывается только для датчиков, контролируемых данным приложением. Аргументами метода являются номер датчика и массив, отражающих собственно значение датчика (наример, датчики ориентации и акселерометр дают по три значения данных каждый)
* Метод **onAccuracyChanged(int sensor,int accuracy)** вызывается при изменении точности показаний датчика.

4. Датчик ориентации

Отдельно можно выделить один из самых распространенных случаев, когда необходимо определить ориентацию девайса (например, ориентация устройства изменяется при повороте его на 90 градусов). Датчик ориентации — это комбинация датчика магнитного поля, выполняющего роль электронного компаса, и акселерометра, измеряющего наклон и вращение. Для снятия показаний с него необязательно реализовывать интерфейс SensorListener. Достаточно в классе активности создать поле типа OrientationEventListener. Это абстрактный вспомогательный класс для получения уведомлений от менеджера датчиков в случае изменения ориентации устройства. Он содержит 4 публичных метода:

|  |  |
| --- | --- |
| boolean | [canDetectOrientation](https://developer.android.com/reference/android/view/OrientationEventListener.html#canDetectOrientation())()  Определяет, способен или нет объект класса OrientationEventListener принимать уведомления об изменении ориентации устройства |
| void | [disable](https://developer.android.com/reference/android/view/OrientationEventListener.html#disable())()  Отключает OrientationEventListener |
| void | [enable](https://developer.android.com/reference/android/view/OrientationEventListener.html#enable())()  Активирует объект класса OrientationEventListene – после вызова метода он может вызывать метод onOrientationChanged() при изменении ориентации устройства |
| abstract void | [onOrientationChanged](https://developer.android.com/reference/android/view/OrientationEventListener.html#onOrientationChanged(int))(int orientation)  Вызывается в случае изменения ориентации устройства |

5. Использование датчика ориентации на примере приложения

Разработанное приложение является игрой для девайсов под ОС Android, суть которой заключается в следующем: изначально пользователь видит окно с приглашением к началу игры. В ходе игры пользователю необходимо передвигать основной объект (самолет), чтобы избежать надвигающихся на него препятствий (метеоритов). Передвижение самолета происходит засчет наклона устройства влево и вправо – чем сильнее наклон, тем быстрее летит самолет в указанном направлении.

Именно для обработки события изменения ориентации девайса в пространстве и используется датчик ориентации. Его показания принимаются с помощью класса OrientationEventListener (класс основной активности содержит поле этого типа). В методе onCreate() основной активности последовательно вызываются методы initOrientationEventListener() и enableOrientationEventListener() для инициализации и активации датчика соответственно.

При инициализации датчика происходит создание нового объекта класса OrientationEventListener и привязка к нему метода onOrientationChanged(int). Этот метод является обработчиком события изменения ориентации устройства, в качестве своего аргумента принимает целое число от 0 до 360 – угол поворота устройства относительно положительного направления оси OZ. При активации датчика происходит проверка, может ли датчик принимать показания. Эта проверка важна, т.к. в некоторых устройствах некоторые типы датчиков могут либо отсутствовать, либо быть отключены, либо не работать по другим причинам (например, при повреждении схемы). При прохождении этой проверки вызывается метод enable(), который активизирует датчик. После этого событие изменения ориентации устройства в пространстве будет отлавливаться обработчиком этого события.

В обработчике значение угла, принятое с датчика, передается в объект класса GameManager («движок» игры). Класс GameManager содержить объекты, которые отражаются на экране (самолет, метеориты, облака) и отвечает за изменение их координат и проверку коллизий – столкновений самолета с метеоритами. Этот класс также содержит поле currentAirplaneAngle – текущий угол наклона устройства. Он передается в объект класса Airplane, отвечающий за корректное отображение самолета. Там угол преобразуется в значение от -90 до 90 – угол наклона от положительного направления оси OZ в положительном или отрицательном направлении по оси OX. В течение игрового цикла происходит непрерывное обновление координат всех объектов. При обновлении координат самолета происходит проверка на то, чтобы объект не вышел за пределы дисплея. В случае, если он находится в рамках дисплея, к его абсциссе прибавляется значение угла, умноженного на скоростной коэффициент. В итоге получается, что при отрицательном угле картинка двигается влево, а при положительном – вправо. В случае, если самолет находится у правого края дисплея происходит запрет на передвижение самолета вправо – игнорируется обновление координаты X в случае наклона телефона в сторону положительного угла. Аналогично обрабатывается ситуация с левой границей дисплея.



Рис.3. Интерфейс игры