Взаимодействие с сенсорным экраном при программировании на android.

Реализация мультитач-жестов.

  Эта статья пригодится тем, кто хочет запрограммировать продвинутое сенсорное управление в своем приложении на android. Как показывает практика, интерфейс, построенный на взаимодействии с сенсорным экраном, очень прост, понятен и удобен.

   Существует 2 способа запрограммировать взаимодействие с сенсорным экраном: вactivity и внутри кода отдельного компонента. Второе обычно полезнее, поскольку тогда один раз запрограммированный компонент будет всегда слушаться нужных жестов.

   Итак, для отлавливания касаний экрана в activity, нужно ввести в ее код следующий метод (eclipse предложит имплементировать интерфейс onTouchListener):

**public** **boolean** onTouch(View arg0, MotionEvent arg1) {

   //code

}

В коде компонента следует аналогичное действие делается так:

**public** **boolean** onTouchEvent(MotionEvent event){

   //code

}

   Аргументами методов служат:

1)Компонент arg0(только в первом случае), над которым и было совершено касание

2)arg1 (во втором случае он называется event) – специальный класс, содержащий кучу параметров и типе прикосновения к сенсорному экрану — координаты, силу нажатия, тип нажатия

   Основные методы класса MotionEvent:

   Методы getX() и getY() возвращают координаты нажатия.

Метод getAction() возвращает тип нажатия пользователем. При проектировании приложения буз мультитача возможны следубщие его значения:

-константа ACTION\_DOWN : пользователь опускает палец на экран

-константа ACTION\_UP : юзер поднимает палец с экрана

-константа ACTION\_MOVE : пользователь перемещает палец по экрану.

Таким образом, если юзер проведет пальцем по дисплею, аозникнет целая куча событийonTouch : сначала ACTION\_DOWN, затем некоторое количество ACTION\_MOVE, затем ACTION\_UP.

   В более сложном случае, когда в управлении используется мультитач, типов событий больше (внимание, программирование мультитача имеет также один важный недостаток — его невозможно протестировать на эмуляторе):

ACTION\_POINTER\_1\_UP – поднятие 1-го пальца

ACTION\_POINTER\_2\_UP – поднятие 2-го пальца

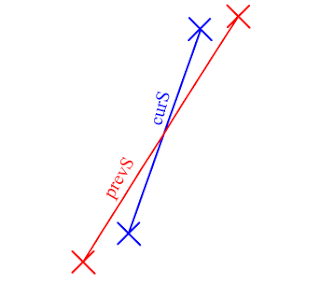
ACTION\_POINTER\_1\_DOWN – отпускание 1-го пальца

ACTION\_POINTER\_2\_DOWN – отпускание 2-го пальца

   Узнать количество пальцев на экране можно методом event.getPointerCount(). Координаты n-го пальца(считая от 0) находят методами getX(n) и getY(n)

   Теории достаточно, давайте программировать мультитач. Условимся, что мы его будем реализовывать в компоненте.

Реализуем следующий несложный интерфейс — двумя пальцами осуществляется просто масштабирование без переноса(чтобы не усложнять пример). Алгоритм таков:

[](http://3.bp.blogspot.com/-opzK7Ns9bnY/T-qYaNLXbEI/AAAAAAAAAHo/fojpJt07X1M/s1600/14.png)

   Пусть синие кресты — положение пальцев в момент времени 2, а красные — в момент времени 1 при их перемещении. Нам нужно рассчитать расстояния между пальцами в оба момента времени (curS и prevS – свойства компонента) и сравнить их — их отношение и будет масштабным коэффициентом.

   Нам, при этом, правда, придется как-то разделять различные акты масштабирования. Нам же не хочется, чтобы после масштабирования и отпускания пальцев при следующем масштабировании расстояние сравнивалось с последним из другого акта.

   Для этой цели не следует выполнять масштабирование, если prevS=0 && curS=0. Тогда нужно просто определить сами расстояния. В начале же акта надо обязательно обнулить эти значения. Код обработчика события:

**if** (event.getPointerCount()==*2*){

    double dx;

    double dy;

**if** (event.getAction()==event.Action\_MOVE){

**if** (prevS==0 && curS==0){

        dx=event.getX(0)-event.getX(1);

        dy=event.getY(0)-event.getY(1);

        curS=Math.*sqrt*(dx\*dx+dy\*dy);

        prevS=curS;

     }

**else**{

        prevS=curS;

        dx=event.getX(0)-event.getX(1);

        dy=event.getY(0)-event.getY(1);

        curS=Math.*sqrt*(dx\*dx+dy\*dy);

                    //Масштабирование с помощью curS и prevS

     }

    }

**if** (event.getAction()==event.ACTION\_POINTER\_2\_DOWN){

     curS=0;

     prevS=0;

    }

}

**You might also like:**

[Программирование под android. Полезный инструмент для ...](http://dmiter.blogspot.com/2012/06/android.html)

[Russian language lesson. Phonetic system. Consonants](http://dmiter.blogspot.com/2012/06/russian-language-lesson-phonetic-system.html)

[Пишем приложение на android для создания тестов. ...](http://dmiter.blogspot.com/2011/09/android-1.html)

[Linkwithin](http://www.linkwithin.com/learn?ref=widget)