Обработка жестов - тема очень обширная, здесь я попробую отметить только основные моменты.

Что касается сложных геометрических жестов (например жест - квадрат и тп.), то подход здесь следующий:

1. Для начала необходимо создать специальный файл, который будет хранить жесты и связывать их с идентификаторами. Создается такой файл с помощью утилиты *Gestures Builder*.
2. В самом приложении необходимо разместить на разметке слой, который будет "принимать" жесты:
3. <android.gesture.GestureOverlayView
4. android:id="@+id/gestures"
5. android:layout\_width="fill\_parent"
6. android:layout\_height="fill\_parent">

</android.gesture.GestureOverlayView>

1. Регистрируем на этот слой слушатель OnGesturePerformedListener, который будет обрабатывать жесты и через колбэк onGesturePerformed возвращать идентификаторы распознаных жестов.

Подробнее смотрите, например, [эту статью](http://habrahabr.ru/post/120016/).

Для обнаружения простых жестов, вроде смахивания, все гораздо проще.

Вам нужно повесить слушатель OnTouchListener на свой элемент, который должен "понимать" смахивания. В колбэке интерфейса onTouch() регистрируется GestureDetector.OnGestureListener, который позволяет переопределить метод onFling() - обнаружение смахивания - в этом методе вы определяете логику действий при смахивании. Так же интерфейс распознает еще [несколько действий](http://developer.android.com/intl/ru/reference/android/view/GestureDetector.OnGestureListener.html): скроллинг, долгое касание и некоторые другие

Простой [пример](https://stackoverflow.com/a/4098447):

public class MyActivity extends Activity {

private void onCreate() {

final GestureDetector gdt = new GestureDetector(new GestureListener());

final ImageView imageView = (ImageView) findViewById(R.id.image\_view);

imageView.setOnTouchListener(new OnTouchListener() {

@Override

public boolean onTouch(final View view, final MotionEvent event) {

gdt.onTouchEvent(event);

return true;

}

});

}

private static final int SWIPE\_MIN\_DISTANCE = 120;

private static final int SWIPE\_THRESHOLD\_VELOCITY = 200;

private class GestureListener extends SimpleOnGestureListener {

@Override

public boolean onFling(MotionEvent e1, MotionEvent e2, float velocityX, float velocityY) {

if(e1.getX() - e2.getX() > SWIPE\_MIN\_DISTANCE && Math.abs(velocityX) > SWIPE\_THRESHOLD\_VELOCITY) {

return false; // справа налево

} else if (e2.getX() - e1.getX() > SWIPE\_MIN\_DISTANCE && Math.abs(velocityX) > SWIPE\_THRESHOLD\_VELOCITY) {

return false; // слева направо

}

if(e1.getY() - e2.getY() > SWIPE\_MIN\_DISTANCE && Math.abs(velocityY) > SWIPE\_THRESHOLD\_VELOCITY) {

return false; // снизу вверх

} else if (e2.getY() - e1.getY() > SWIPE\_MIN\_DISTANCE && Math.abs(velocityY) > SWIPE\_THRESHOLD\_VELOCITY) {

return false; // сверху вниз

}

return false;

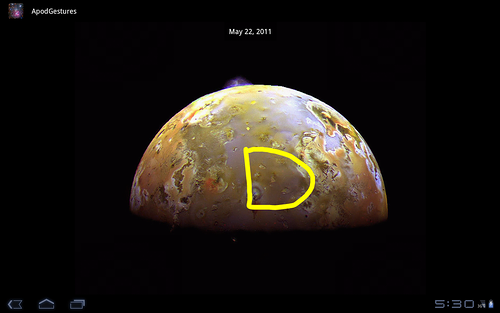
}

}

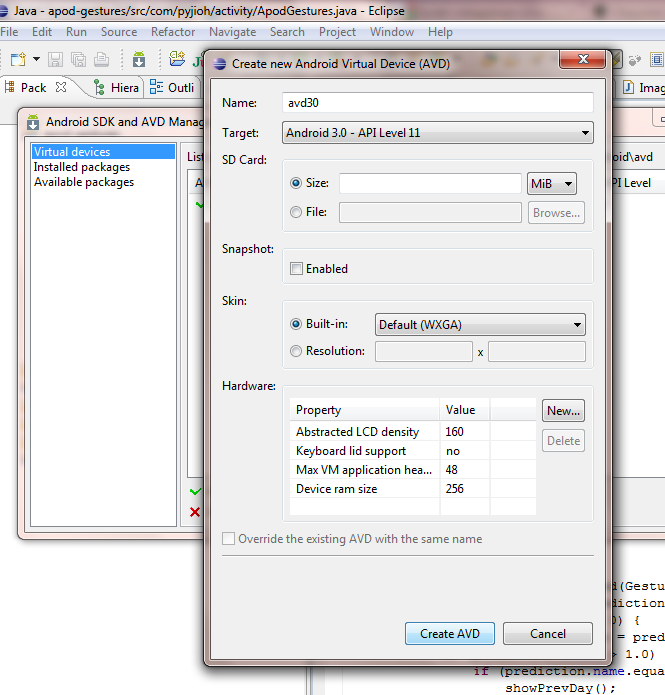
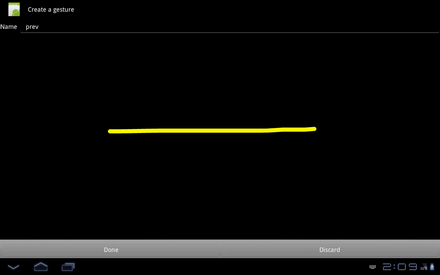
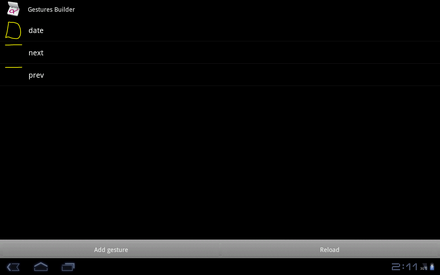
}

Более глубокое погружение в тему распознавания простых жестов - смотрите, например, [эту статью](http://www.ibm.com/developerworks/ru/library/j-mobileforthemasses2/index.html).

Введение в Gestures

*или толкование жестов :)*  
  
[](http://img-fotki.yandex.ru/get/4527/11245482.2/0_63c5e_eb3a19bf_orig)  
  
Сейчас мы на боевом примере поработаем с жестами в среде Android. Приложение будет клиентом сайта [Astronomy Picture of the Day](http://apod.nasa.gov/apod/) by NASA. На этом сайте ребята каждый день выкладывают какую-нибудь замечательную картинку, связанную с астрономией. Жестами мы будем ходить вперед/назад и вызывать диалог выбора даты. А чтобы было еще интересней — напишем его для Honeycomb.  
  
Статья состоит из двух частей: первая — покажет, как создавать жесты с помощью приложения Gestures Builder и выгружать их из эмулятора в отдельный бинарный файл. Во второй части, мы загрузим его в наше приложение и начнем использовать.

Создание жестов

Пакет [android.gesture](http://developer.android.com/reference/android/gesture/package-summary.html) появился в API версии 1.6 и призван существенно облегчить обработку жестов и уменьшить количество кода, который будут писать программисты. Вместе с тем, в эмуляторе версии 1.6 и выше появилось предустановленное приложение Gestures Builder, с его помощью мы можем создавать набор подготовленных жестов и добавлять их в виде бинарного ресурса в своё приложение. Для этого мы сделаем эмулятор с вмонтированным образом SD карты, куда будет сохранятся файл с жестами (без карты Gestures Builder скажет, что ему некуда записывать жесты)  
  
**Шаг 1.** Создадим img образ SD карты с помощью *mksdcard*. Эту утилиту можно найти в папке <путь куда установлен android-sdk>\tools. Файл с жестами будет весить немного, но эмулятор android имеет ограничение на минимальный размер карты что-то около 8-9 мегабайт. Поэтому мы создадим с запасом и каким-нибудь ровным размером, например, 64 мб. Пишем:  
  
*mksdcard -l mySdCard 64M gestures.img*  
  
**Шаг 2.** Сделаем эмулятор запустим его с образом gestures.img. Действуем как на картинке:  
  
  
  
Чтобы запустить эмулятор с образом зайдем в <путь куда установлен android-sdk>\tools и выполним:  
  
*emulator -avd avd30 -sdcard gestures.img*  
  
**Шаг 3.** Эмулятор запущен и нам нужно сделать свои жесты. Для этого откроем Gestures Builder нажмем Add gesture и перед нами откроется редактор жеста. Отрисуем слева направо жест с нажатой левой клавишей мыши и вверху дадим ему имя — *prev*. Done. Затем по аналогии (в другую сторону) создаем жест *next*и в качестве последнего, напишем что-то похожее на латинскую D и назовем *date*. На картинках ниже — как мы это делаем (кликабельно):  
  
[](http://habrastorage.org/storage/c8d309a1/8bead35f/27474399/34b56afa.png)  
  
[](http://habrastorage.org/storage/f274ee8e/d9cf0fad/df2e79b0/09ba1bb8.png)  
  
[](http://habrastorage.org/storage/f93c1622/a04b0b89/4166169a/b14126bc.png)  
  
**Шаг 4.** У запущенного эмулятора мы забираем бинарный ресурс с жестами, выполнив в директории tools:  
  
*adb pull /sdcard/gestures gestures*  
  
После чего файл gestures должен появиться. Мы положим его в свой проект по пути res\raw\gestures.

Использование жестов

Наше приложение будет называться ApodGestures. Во второй части я коснусь только тем, связанных с жестами. Весь программный код можно взять на [code.google.com](http://code.google.com/p/apod-gestures/) из Mercurial репозитория, либо выполнить в консоли *hg clone*[*pyJIoH@apod-gestures.googlecode.com/hg*](https://pyJIoH@apod-gestures.googlecode.com/hg/)*apod-gestures*, либо скачать архив во вкладке [Downloads](http://code.google.com/p/apod-gestures/downloads/list).   
  
ApodGestures будет коннектиться к сайту [APOD](http://apod.nasa.gov/apod/), выкачивать астрономическую картинку в отдельном потоке и загружать ее в ImageView. С помощью жестов мы сделаем навигацию: вперед, назад и диалог выбора любой даты.  
  
В принципе, здесь не используются какие-то специальные фичи из Honeycomb, поэтому, снизив версию, мы без проблем запустим приложение на телефонах. Ограничения скорее накладывал сам Honeycomb, т.к. c этой версии и выше ужесточилась политика того, что можно делать в основном потоке. Здесь я столкнулся с примером NetworkOnMainThreadException при работе в главном потоке HttpClient'а. Почитать можно [тут](http://android-developers.blogspot.com/2010/12/new-gingerbread-api-strictmode.html).  
  
Создав проект, положим наш бинарный ресурс по пути *apod-gestures\res\raw\gestures*.  
  
ImageView в xml layout'е поместим на android.gesture.GestureOverlayView, которое является прозрачным наложением, способным определять ввод жестов. Код будет таким:

<?xml version="1.0" encoding="utf-8"?>

<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="fill\_parent"

android:layout\_height="fill\_parent">

<android.gesture.GestureOverlayView

android:id="@+id/gestures"

android:layout\_width="fill\_parent"

android:layout\_height="fill\_parent"

android:layout\_weight="1.0">

<ImageView

android:layout\_gravity="center\_vertical"

android:id="@+id/ImageView"

android:layout\_width="fill\_parent"

android:layout\_height="fill\_parent"

/>

<Button

android:id="@+id/button\_select\_date"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_gravity="center\_horizontal|top"

android:padding="12dip"

android:background="#AA000000"

android:textColor="#ffffffff"

android:text="Feb 10, 2011"

/>

</android.gesture.GestureOverlayView>

</FrameLayout>

Далее добавим в ApodGestures activity поле: private GestureLibrary mGestureLib;   
В него мы загрузим наши жесты из raw ресурса:

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.main);

mGestureLib = GestureLibraries.fromRawResource(this, R.raw.gestures);

if (!mGestureLib.load()) {

finish();

}

GestureOverlayView gestures = (GestureOverlayView) findViewById(R.id.gestures);

gestures.addOnGesturePerformedListener(this);

refreshActivity();

}

Чтобы начать отслеживать вводимые жесты имплементируем нашей activity OnGesturePerformedListener и реализуем метод onGesturePerformed:

public void onGesturePerformed(GestureOverlayView overlay, Gesture gesture) {

ArrayList<Prediction> predictions = mGestureLib.recognize(gesture);

if (predictions.size() > 0) {

Prediction prediction = predictions.get(0);

if (prediction.score > 1.0) {

if (prediction.name.equals(Gestures.prev))

showPrevDay();

else if (prediction.name.equals(Gestures.next))

showNextDay();

else if (prediction.name.equals(Gestures.date))

showSelectDateDialog();

}

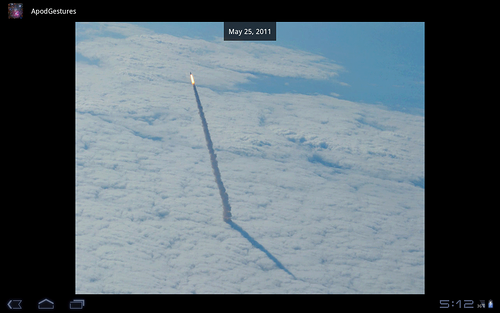
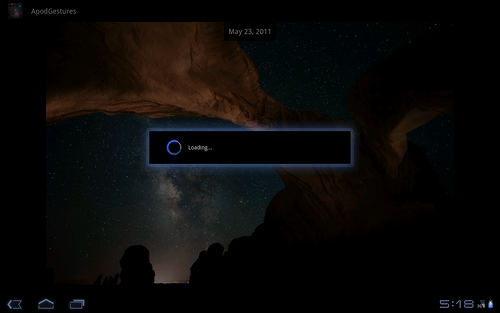
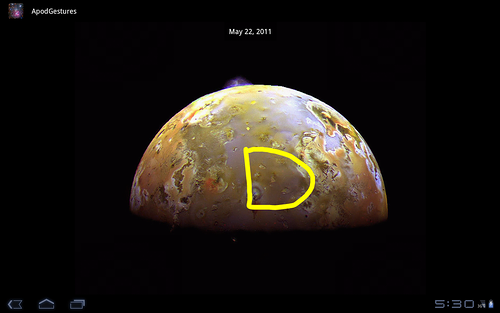
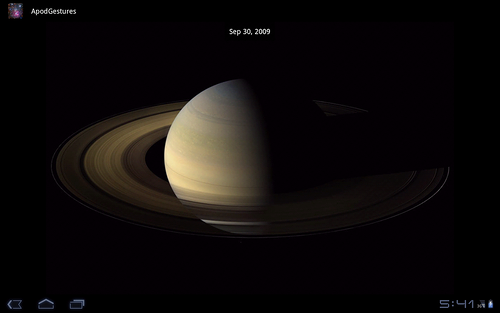
}

}

Как видите, имя которое мы вводили в приложении Gestures Builder является идентификатором жестов. Gestures — это наш класс с константами, где имена забиты фиксировано. Если кого-то напрягают жирные желтые линии, любой цвет мы можем задать в xml для GestureOverlayView, свойства android:gestureColor и android:uncertainGestureColor. UncertainGestureColor — это цвет рисующийся на view, когда нет уверенности, что это жест.  
  
Можно сделать цвета серыми и прозрачными:

android:gestureColor="#AA000000"

android:uncertainGestureColor="#AA000000">

Пример того, что получилось в итоге (все еще кликабельно).  
  
[](http://img-fotki.yandex.ru/get/4426/11245482.2/0_63c60_fe65f8fc_orig)  
  
[](http://img-fotki.yandex.ru/get/4527/11245482.3/0_63c61_70f5340a_orig)  
  
[](http://img-fotki.yandex.ru/get/4424/11245482.3/0_63c62_5e38cad5_orig)  
  
[](http://img-fotki.yandex.ru/get/4527/11245482.3/0_63c63_8df5613f_orig)  
  
[](http://img-fotki.yandex.ru/get/4527/11245482.3/0_63c64_8ffc0ad8_orig)  
  
Надеюсь Вам понравилось.  
  
UPD. Добавил apk файл во вкладку [Download](http://code.google.com/p/apod-gestures/downloads/list). Для установке требуется Android 3.0 и выше.