# Android平台输入法开发解析

Android平台自1.5版本之后开放了输入法框架（Input Method Framework，IMF），IMF是Android平台的特色设计。它的出现，为诞生不带实体键盘的设备提供了可能。同时，IMF开放了Android平台输入法的开发接口，为Android平台的输入法提供了可扩展性。

本文首先介绍Android平台IMF的构成、工作机制和流程，以便广大开发者能够更好的理解Android平台是如何实现文字输入。接下来，将以一个简单的例子，介绍如何进行简单的输入法开发，开发者应该可以对Android平台的输入法开发形成一个初步的认识。

## 1．Android平台IMF简介

Android平台的IMF按照功能分为三个主要模块，分别为：客户端控件，输入法服务（IMMS），以及输入法应用（IME）。如果要深刻的理解和掌握Android平台的输入法开发技巧，那么必须要了解Android平台IMF内部各个模块之间的工作机制。

客户端控件，是指具有文字编辑功能的系统控件（如EditText），是平台内部输入法应用人机交互的起点。输入法服务，是Android平台的底层基础服务之一，负责管理输入法，包括输入法的安装、注册、激活等。输入法应用，则是指平台内预置，或者后续安装的输入法程序。三个模块间的交互关系如下：

下面我们将分别介绍以上三个模块，并简要介绍各模块内主要使用的对象和服务。

### 1.1客户端控件

Android平台的客户端控件主要是TextView及其子类。客户端控件是输入法人机交互的起点。客户端控件与输入法服务和输入法应用都有交互操作。

以EditText为例，当客户端控件接受到焦点时，控件启动输入法应用并显示键盘；而失去焦点时，则隐藏键盘。另外，EditText控件收到长按事件时，弹出输入法选择菜单。这类操作属于客户端控件与输入法服务间的交互。

此外，客户端控件还可以向输入法应用传递文本状态，包括光标位置、文本选择等；接受并显示输入法应用反馈的输入文字。这类交互属于客户端控件与输入法应用的交互。

客户端控件对输入法服务和输入法应用的操作，都是通过对InputMethodManager实例的调用来实现的。InputMethodManager更像是一个供客户端控件使用API操作的集合，定义对输入法应用以及输入法服务的一系列操作。客户端控件会通过InputMethodManager.peekInstance() 来获取唯一的实例。

需要指出的是，各调用函数的具体操作并不是在InputMethodManager中具体实现的。Android平台的输入法框架定义了IIputMethodManager接口和IIputMethodSession接口分别定义了对输入法服务和输入法应用的操作。而InputMethodManagerService和InputMethodService则分别对这两个接口进行了具体的实现。这样通过AIDL机制，客户端控件便可以跨进程的调用其他service内的实现。

通过这样的机制，Android平台降低了输入法框架内各个模块间的耦合性，并且保证了输入法应用的可扩展性。客户端无需知道使用的是那一款输入法，从而实现了简单直接的控制。

另外还需要指出的，客户端控件通过IIputMethodSession对于输入法应用的交互是单向的，即只能向输入法应用传递信息，无法获取信息。客户端应用从输入法应用获取信息是通过InputConnection来实现的, 在启动输入法时，InputConnection由客户端控件创建，并传递给输入法应用，由输入法应用调用，进行信息反馈。

### 1.2输入法服务

输入法服务作为平台底层的一项基础服务，用来管理输入法应用。输入法服务的主要工作由InputMethdoManagerService（IMMS）完成。下面简要的介绍输入法服务如何实现输入法的安装、切换。

* **输入法安装**

IMMS内部包含一个receiver，注册接收所有程序包安装、卸载的消息。当收到此类消息后，IMMS会通过系统的PackageManager查询所有声明为InputMethod的程序，并生成一个系统可用的输入法列表，供用户选择使用。

* **输入法切换**

当用户选择了一款输入法后，IMMS将此输入法的ID保存为系统的默认输入法ID。这里的ID是该输入法在IMMS内维护的可用输入法列表中的位置ID。

当客户端通过InputMethodManager启动输入法应用时，IMMS将根据这个ID从输入法列表中取出输入法，并加载使用。

### 1.3输入法应用

输入法应用是具体处理用户输入行为的应用程序。为了能够在Android的输入法框架中良好的运行，所有的输入法应用都需要继承特定的service。Android平台的输入法框架为输入法应用定义了一个基类InputMethodService。InputMethodService提供了一个输入法的标准实现。定义了输入法生命周期内的重要函数，提供给开发人员进行相应的处理。

为了帮助开发者了解输入法应用的工作流程。我们首先看一下输入法应用的生命周期：

* 当用户触发输入法显示的时候（客户端控件获得焦点），InputMethodService启动。首先调用onCreate() 函数，该函数在输入法第一次启动的时候调用，适合用来做一些初始化的设置，与其他service相同；
* 调用onCreateInputView() 函数，在该函数中创建KeyboardView并返回；
* 调用onCreateCandidatesView()函数，在该函数中创建候选区实现并返回；
* 调用onStartInputView()函数来开始输入内容，
* 输入结束后调用onFinishInput()函数来结束当前的输入，
* 如果移动到下一个输入框则重复调用onStartInputView和onFinishInput函数；
* 在输入法关闭的时候调用onDestroy() 函数。



InputMethodService实现了两个重要的接口，InputMethod和InputMethodSession。

* **InputMethod**

InputMethod接口定义了一套操纵输入法应用的方法。如，bindInput, hideInput, startInput等。为了系统安全，这类接口只有系统可以访问，客户端控件无法直接调用这个接口。所有的输入法应用都需要客户端控件具有BIND\_INPUT\_METHOD权限，作为系统的安全机制，否则将无法与输入法服务交互。

* **InputMethodSession**

InputMethodSession作为InputMethod的辅助接口类，为客户端控件开放了可直接调用的函数接口。包括向输入法应用分发键盘事件，更新光标位置，更新编辑区域内选择的问题信息等。

## 2．Android平台输入法开发方法

通过对Android平台输入法框架的介绍，相信大家已经初步了解了Android平台内输入法相关模块间的交互和工作机制。

Android平台内的输入法开发主要包括上层界面UI的开发，以及底层输入法引擎的开发。本文所介绍的开发技术包括如何在Android平台下如何构建输入法应用，如何设计键盘，进行界面开发。

值得说明的是，输入法的很多核心功能都是体现在底层输入法引擎中的。输入法引擎的功能包括根据输入字符获取候选词以及联想词，调整管理词库词频等等。简而言之，输入法引擎是一个语言邻域专用的数据库引擎，根据用户按键输入，在语言数据库中查询出候选，供用户选择。这些与Android平台的开发是相互独立的，在此不做细致的说明。

下面我们以简单的实例，说明如何开发一个Android平台的输入法应用。

### 2.1.配置服务

输入法应用在Android系统中是一个service。与其他service一样，输入法需要通过在AndroidManifest.xml中进行service定义。示例如下：

<application android:label="@string/english\_ime\_name">

<service android:name="LatinIME"

android:label="@string/english\_ime\_name"

android:permission="android.permission.BIND\_INPUT\_METHOD">

<intent-filter>

<action android:name="android.view.InputMethod" />

</intent-filter>

<meta-data

android:name="android.view.im"

android:resource="@xml/method" />

</service>

</application>

该service具有BIND\_INPUT\_METHOD权限，表明这是一个输入法服务。在intent-filter中使用android.view.InputMethod action来定义，最后通过name为android.view.im的meta-data来描述该输入法的一些属性，meta-data引用的是一个XML文件，该文件是输入法的配置文件，用来配置一些信息，例如是否为默认输入法，是否具有配置Activity来配置输入法的一些选项，如果指定了配置Activity则在系统设置界面中的输入法设置中可以启动该Activity来设置输入法的配置项。

### 2.2继承InputMethodService

通过扩展android.inputmethodservice.InputMethodService可以很容易的实现一个输入法服务，InputMethodService提供了一些系统回调函数，可以按照需要来实现。上面已经进行了详细的解析，再此不做过多赘述。

构成一个输入法应用，最重要的界面元素包括软键盘区域和候选词区域。InputMethodService为这两个区域设置了专门的回调函数，以便开发者灵活的定制并加载资源文件。下面针对这两个区域的回调函数，做一个简单的介绍。

#### 2.2.1 onCreateInputView

该函数在输入区域，（比如虚拟键盘）第一次被显示的时候被调用，仅调用一次。显示的View通过return返回，默认的返回是null，当返回为null时，什么也不显示。

当输入区域显示后，可以通过实现[onEvaluateInputViewShown()](file:///E:\SDK\dr24622-OMS1.5-tiger-android-sdk_eng.sdk_windows-x86\android-sdk__windows\docs\reference\android\inputmethodservice\InputMethodService.html#onEvaluateInputViewShown())来控制。如果想替换显示的输入区域可以通过[setInputView(View)](file:///E:\SDK\dr24622-OMS1.5-tiger-android-sdk_eng.sdk_windows-x86\android-sdk__windows\docs\reference\android\inputmethodservice\InputMethodService.html#setInputView(android.view.View))实现。

生成输入区域的View通过LayoutInflater加载Layout中的资源文件来实现。在资源文件中或者是一个KeyboardView或者是由若干个KeyboardView组成的Container。

通常在这个函数中都需要注册键盘事件的监听器setOnKeyboardActionListener。这个监听器用来处理以下键盘事件：

[onKey](file:///E:\SDK\dr24622-OMS1.5-tiger-android-sdk_eng.sdk_windows-x86\android-sdk__windows\docs\reference\android\inputmethodservice\KeyboardView.OnKeyboardActionListener.html#onKey(int, int[]))(int primaryCode, int[] keyCodes)

[onPress](file:///E:\SDK\dr24622-OMS1.5-tiger-android-sdk_eng.sdk_windows-x86\android-sdk__windows\docs\reference\android\inputmethodservice\KeyboardView.OnKeyboardActionListener.html#onPress(int))(int primaryCode)

[onRelease](file:///E:\SDK\dr24622-OMS1.5-tiger-android-sdk_eng.sdk_windows-x86\android-sdk__windows\docs\reference\android\inputmethodservice\KeyboardView.OnKeyboardActionListener.html#onRelease(int))(int primaryCode)

[onText](file:///E:\SDK\dr24622-OMS1.5-tiger-android-sdk_eng.sdk_windows-x86\android-sdk__windows\docs\reference\android\inputmethodservice\KeyboardView.OnKeyboardActionListener.html#onText(java.lang.CharSequence))([CharSequence](file:///E:\SDK\dr24622-OMS1.5-tiger-android-sdk_eng.sdk_windows-x86\android-sdk__windows\docs\reference\java\lang\CharSequence.html) text)

#### 2.2.2 onCreateCandidatesView

该函数在联想词条第一次被显示的时候调用，仅调用一次。显示的View通过return返回。如果没有联想条，则返回null。

控制是否显示联想词条，可以通过[setCandidatesViewShown(boolean)](file:///E:\SDK\dr24622-OMS1.5-tiger-android-sdk_eng.sdk_windows-x86\android-sdk__windows\docs\reference\android\inputmethodservice\InputMethodService.html#setCandidatesViewShown(boolean))来实现。如果想替换联想词条则通过实现[setCandidatesView(View)](file:///E:\SDK\dr24622-OMS1.5-tiger-android-sdk_eng.sdk_windows-x86\android-sdk__windows\docs\reference\android\inputmethodservice\InputMethodService.html#setCandidatesView(android.view.View)).来实现。

构造联想词条，通过LayoutInflater来加载资源文件，生成View。

### 2.3软键盘区域

在Android 平台中软键盘是很容易实现的，通过Keyboard 类可以创建软键盘，该类从XML文件中读取软键盘信息。软键盘信息包括有多少行，每行有多少按键，每个按键代表什么内容等等。

#### 2.3.1 Keyboard

Keyboard通过加载XML文件，构建虚拟键盘。XML文件的内容示例如下：

<Keyboard

android:keyWidth="10%p"

android:horizontalGap="0px"

android:verticalGap="0px"

android:keyHeight="@dimen/key\_height"<!—在Value中预定义的常量-->

>

<Row android:verticalGap="@dimen/key\_padding">

<Key android:codes="113" android:keyLabel="q" android:keyEdgeFlags="left" android:popupKeyboard="@xml/kbd\_popup\_template"

android:popupCharacters="@string/alternates\_for\_q"/>

……

</Row>

borqs:keyWidth，borqs:horizontalGap，borqs:verticalGap，borqs:keyHeight分别用来表示横向纵向的间距以及按键长宽。

Row用来表示行，每行包括若干Key。最左侧和最右侧的Key需要用android:keyEdgeFlags标示。android:popupKeyboard和android:popupCharacters用来标示按键后弹出的小键盘的资源id和显示的文字。

填写完成构建键盘的XML文件后，需要编写一个类来进行具体的操作处理。这个类继承自Keyboard类。在这个类中，需要实现的方法有：

* 构造函数：通过构造函数加载键盘资源文件
* [createKeyFromXml](file:///E:\SDK\dr24622-OMS1.5-tiger-android-sdk_eng.sdk_windows-x86\android-sdk__windows\docs\reference\android\inputmethodservice\Keyboard.html#createKeyFromXml(android.content.res.Resources, android.inputmethodservice.Keyboard.Row, int, int, android.content.res.XmlResourceParser))：如果需要在按键类增加某些特定属性，可以通过继承Keyboard.Key类来实现，这时应相应的在此函数中增加这些特色属性的初始化工作。

#### 2.3.2 KeyboardView

KeyboardView用来渲染一个虚拟键盘，并监控键盘上面的点击和触摸事件。输入法应用需要构建一个继承自KeyboardView的键盘视图类。在这个类中处理所有来自键盘的事件。

重要的函数包括：

* setKeyboard:用来绑定Keyboard和KeyboardView。
* onLongPress：用来处理长按事件
* [getOnKeyboardActionListener](file:///E:\SDK\dr24622-OMS1.5-tiger-android-sdk_eng.sdk_windows-x86\android-sdk__windows\docs\reference\android\inputmethodservice\KeyboardView.html#getOnKeyboardActionListener())：

在KeyboardView中有一个监听器——OnKeyboardActionListener。通过这个函数可以获得此键盘视图的监听器。监听器通过OnKey函数分发处理键盘点击事件。

* getKeyboard:通过此函数获取和view绑定的键盘类

### 2.4候选词区域

对于中文输入法来说候选区域是一个特别重要的内容，拿拼音输入法来说用户输入拼音会出现多个候选词语，通过对候选词语的合理安排是一个输入法是否好用的重要评判标准之一。但是也有特殊情况下不需要候选区域的，例如输入数字或则密码。候选区域通过继承View实现。

#### 2.4.1 CandidateView

候选栏不同于键盘，Android输入法框架中没有为他封装特定的类。输入法候选栏需要集成View，独立实现。

在这个类中，要处理所有点击在候选栏的操作。包括点击候选词的操作。输入法应用在初始化时，会将自己的引用或者专门用来处理点击候选词操作的listener传递给CandidateView中。具体的选中候选词的操作要通过调用输入法应用服务内部的函数完成。

CandidateView中一个重要的函数是onDraw()，通过此函数绘画候选栏。CandidateView中需要设置用来保存候选词页数的变量，通过页数，计算候选词显示的词语个数，已调整宽度。通过Canvas的drawText方法写出各候选词条。

CandidateView中还可以定义一个PopupWindow用来显示点中后弹出的文字内容。这个弹出窗口可以在CandidateView中构建，也可以由CandidateContainer传递进来。

#### 2.4.2 CandidateContainer

之所有构建一个容器，是为了保存多个CandidateView。候选词条上的左右方向的翻页按钮，也是通过这个容器来管理的。这个类不是Android平台框架实现的类，借鉴了Android平台自带的LatinIME，我认为这是一个不错的实现方式。细化了CandidateView的功能，处理了除候选词外的其他逻辑操作。

另外使用CandidateContainer还有一个优点，即，通过保存两个CandidateView，翻页处理时可以马上显示，而不需要重新加载，提高了显示效率。CandidateContainer是通过ViewFlipper来管理多个View的。

## 3． 总结：

 从以上对于输入法框架的介绍，以及输入法开发方法的介绍可以看出，Android平台提供的良好的输入法框架可以很方便的开发第三方输入法，开发者只需要关注于输入法的业务逻辑实现，界面部分交给Android来做就可以了，这样可以极大的节省开发人员的精力和开发成本。

对于目前流行的大屏幕触摸屏智能终端，输入法应用的功能以及人机交互流程的设计，很大程度上影响着终端的用户体验。可以说输入法是大屏幕触摸屏智能终端最重要的应用模块之一。Android平台在不断改造输入法应用的同时，也开放了平台的输入法模块的开发合作，为广大开发者以及专业厂商创造了参与Android平台输入法开发的机会。