2016

AccessibilityEvent

中国信息无障碍产品联盟&信息无障碍研究会　译制

20161101

目录

翻译声明

翻译机构：信息无障碍研究会（ARA）　中国信息无障碍产品联盟（CAPA）

译者：李鸿利

审阅：刘辉、刘彪、沈广荣

本文档翻译自谷歌官方文档《AccessibilityEvent》。如您对翻译文档内容有异议，请将原文文档做为主要参考，原文版权由 Google 持有并保留。

AccessibilityEvent

添加于 API 级别 4。

public final class AccessibilityEvent
extends AccessibilityRecord implements Parcelable

java.lang.Object
↳android.view.accessibility.AccessibilityRecord
↳android.view.accessibility.AccessibilityEvent

当用户界面发生某些明显的事件时，该类代表的无障碍事件会被系统发送。例如，一个按钮被点击，一个视图被聚焦等等。

一个无障碍事件由独立视图发送，独立视图使用它的状态数据填充事件，并从它的父视图请求发送事件到感兴趣的各方。在分发一个类似的请求给其父视图之前，父视图可以自由地为自己添加一个 AccessibilityRecord。父视图也可以选择不响应发送事件的请求。无障碍事件是由视图树中的最顶端视图发送的。因此，AccessibilityService 可以浏览无障碍事件中的所有记录，来获取更多关于发送事件的上下文信息。

一个无障碍事件的主要目的是向 AccessibilityService 暴露足够的信息，来为用户提供有意义的反馈。但是有时，无障碍服务可能需要比事件负载的更多的上下文信息。在这种情境下，该服务可以获得事件源，此事件源是一个可以用于浏览窗口内容的 AccessibilityNodeInfo（视图状态的快照)。请注意，必须明确请求访问事件源和窗口内容的权限。更多详细信息请参考 AccessibilityService。如果无障碍服务没有请求检索窗口内容的权限，事件不会包含事件源的引用。对于 TYPE_NOTIFICATION_STATE_CHANGED 类型的事件，事件源是永远不可获得的。

该类代表各种语义的不同无障碍事件类型。每一种事件类型都有一组相关联属性。换言之，每一种事件类型是由该类暴露出的属性子集表示其特征的。在此类中为每一种事件类型定义了相应的常量。下面所描述的是事件类型及其相关属性的规范。

1.开发者指南

关于创建和处理 AccessibilityEvents 的更多信息，请参阅无障碍开发者指南。

1.1 视图类型

1.1.1 点击视图

表示点击一个视图的事件，例如按钮（Button）、复合按钮（CompoundButton）等视图。

类型： TYPE_VIEW_CLICKED

属性：

- getEventType()-事件类型。
- getSource()-源信息（对于已注册的客户端）。
- getClassName()-源的类名。
- getPackageName()-源的包名。
- getEventTime()-事件时间。
- getText()-源的子树的文本。
- isEnabled()-源是否可用。
- isPassword()-源是否是个密码。
- isChecked()-源是否选中。
- getContentDescription()-源的内容描述。
- getScrollX()-使用像素表示源左边缘的偏移量(不包含 AdapterView 的子视图)。
- getScrollY()-使用像素表示源顶部边缘的偏移量(不包含 AdapterView 的子视图)。
- getFromIndex()-源第一个可见项的从零开始的索引，包含（用于 AdapterView 的子视图）。
- getToIndex()-源最后一个可见项的从零开始的索引，包含（用于 AdapterView 的子视图）。

- getItemCount()-源的总项目数（用于 AdapterView 的子视图）。

1.1.2 长按点击视图

表示长按点击一个视图的事件，例如按钮（Button）、复合按钮（CompoundButton）等视图。

类型：TYPE_VIEW_LONG_CLICKED。

属性：

- getEventType()-事件类型。
- getSource()-源信息（对于已注册的客户端）。
- getClassName()-源的类名。
- getPackageName()-源的包名。
- getEventTime()-事件时间。
- getText()-源的子树的文本。
- isEnabled()-源是否可用。
- isPassword()-源是否是个密码。
- isChecked()-源是否选中。
- getContentDescription()-源的内容描述。
- getScrollX()-使用像素表示源左边缘的偏移量(不包含 AdapterView 的子视图)。
- getScrollY()-使用像素表示源顶边缘的偏移量(不包含 AdapterView 的子视图)。
- getFromIndex()-源第一个可见项的从零开始的索引，包含（用于 AdapterView 的子视图）。
- getToIndex()-源最后一个可见项的从零开始的索引，包含（用于 AdapterView 的子视图）。
- getItemCount()-源的总项目数（用于 AdapterView 的子视图）。

1.1.3 选择视图

通常表示在 AdapterView 的上下文中选择一个项目的事件。

类型：TYPE_VIEW_SELECTED。

属性：

- getEventType()-事件类型。
- getSource()-源信息（对于已注册的客户端）。
- getClassName()-源的类名。
- getPackageName()-源的包名。
- getEventTime()-事件时间。
- getText()-源的子树的文本。
- isEnabled()-源是否可用。
- isPassword()-源是否是个密码。
- isChecked()-源是否选中。
- getItemCount() –源中可选择项目的数目。
- getCurrentItemIndex() – 当前已选择项目索引。
- getContentDescription()-源的内容描述。
- getScrollX()-使用像素表示源左边缘的偏移量(不包含 AdapterView 的子视图)。
- getScrollY()-使用像素表示源顶边缘的偏移量(不包含 AdapterView 的子视图)。
- getFromIndex()-源第一个可见项的从零开始的索引，包含（用于 AdapterView 的子视图）。
- getToIndex()-源最后一个可见项的从零开始的索引，包含（用于 AdapterView 的子视图）。
- getItemCount()-源的总项目数（用于 AdapterView 的子视图）。

1.1.4 聚焦视图

表示聚焦一个视图的事件。

类型：TYPE_VIEW_FOCUSED。

属性：

- getEventType()-事件类型。
- getSource()-源信息（对于已注册的客户端）。
- getClassName()-源的类名。
- getPackageName()-源的包名。

- getEventTime()-事件时间。
- getText()-源的子树的文本。
- isEnabled()-源是否可用。
- isPassword()-源是否是个密码。
- isChecked()-源是否选中。
- getItemCount() –源中可选择项目的数目。
- getCurrentItemIndex() – 当前已选择项目索引。
- getContentDescription()-源的内容描述。
- getScrollX()-使用像素表示源左边缘的偏移量(不包含 AdapterView 的子视图)。
- getScrollY()-使用像素表示源顶边缘的偏移量(不包含 AdapterView 的子视图)。
- getFromIndex()-源第一个可见项的从零开始的索引，包含（用于 AdapterView 的子视图）。
- getToIndex()-源最后一个可见项的从零开始的索引，包含（用于 AdapterView 的子视图）。
- getItemCount()-源的总项目数（用于 AdapterView 的子视图）。

1.1.5 视图文本改变

表示一个编辑框（EditText）文本改变的事件。

类型：TYPE_VIEW_TEXT_CHANGED

属性：

- getEventType()-事件类型。
- getSource()-源信息（对于已注册的客户端）。
- getClassName()-源的类名。
- getPackageName()-源的包名。
- getEventTime()-事件时间。
- getText()-源的子树的文本。
- isEnabled()-源是否可用。
- isPassword()-源是否是个密码。
- isChecked()-源是否选中。

- getFromIndex()-文本改变起始索引。
- getAddedCount()-被添加的字符数。
- getRemovedCount()-被移除的字符数。
- getBeforeText()-改变之前的源文本。
- getContentDescription()-源的内容描述。

1.1.6 视图文本选择改变

表示一个编辑框（EditText）改变文本选择的事件。

类型：TYPE_VIEW_TEXT_SELECTION_CHANGED。

属性：

- getEventType()-事件类型。
- getSource()-源信息（对于已注册的客户端）。
- getClassName()-源的类名。
- getPackageName()-源的包名。
- getEventTime()-事件时间。
- getText()-源的子树的文本。
- isPassword()-源是否是个密码。
- getFromIndex()-选择开始索引。
- getToIndex()-选择结束索引。
- getItemCount()-源文本的长度。
- isEnabled()-源是否可用。
- getContentDescription()-源的内容描述。

1.1.7 以移动粒度遍历视图文本

表示以给定粒度遍历一个视图文本的事件。例如，移动到下一个词。

类型：TYPE_VIEW_TEXT_TRAVERSED_AT_MOVEMENT_GRANULARITY。

属性：

- getEventType()-事件类型。
- getSource()-源信息（对于已注册的客户端）。
- getClassName()-源的类名。

- getPackageName()-源的包名。
- getEventTime()-事件时间。
- getMovementGranularity()-设置视图文本被遍历的粒度。
- getText()-源的子树的文本。
- getFromIndex()-在移动中被跳过的文本开头。向前移动遍历文本时，它是起始点，但向后移动时不是。
- getToIndex()-在移动中被跳过的文本末尾。向前移动遍历文本时，它是结束点，但向后移动时不是。
- isPassword()-源是否是个密码。
- isEnabled()-源是否可用。
- getContentDescription()-源的内容描述。
- getMovementGranularity()-设置视图文本被遍历的粒度。
- getAction()-获得指定方向的遍历动作。

1.1.8 滚动视图

表示滚动视图的事件。如果源是 AdapterView 的子视图，滚动被报告在可见项组中——第一个可见项，最后一个可见项，和总项目——因为源并不知道其像素尺寸，其适配器负责创建视图。在所有其他情况下，滚动被报告为当前在 X 和 Y 轴的滚动，分别加上源高度的像素。

类型：TYPE_VIEW_SCROLLED。

属性：

- getEventType()-事件类型。
- getSource()-源信息（对于已注册的客户端）。
- getClassName()-源的类名。
- getPackageName()-源的包名。
- getEventTime()-事件时间。
- getText()-源的子树的文本。
- isEnabled()-源是否可用。
- getContentDescription()-源的内容描述。
- getScrollX()-使用像素表示源左边缘的偏移量(不包含 AdapterView 的子视图)。

- getScrollY()-使用像素表示源顶边缘的偏移量(不包含 AdapterView 的子视图)。
- getFromIndex()-源第一个可见项的从零开始的索引，包含（用于 AdapterView 的子视图)。
- getToIndex()-源最后一个可见项的从零开始的索引，包含（用于 AdapterView 的子视图）。
- getItemCount()-源的总项目数（用于 AdapterView 的子视图)。

注意：该事件类型不使用 View.dispatchPopulateAccessibilityEvent(AccessibilityEvent)分发给子视图，因此该事件源视图和它的子树不会接收 View.onPopulateAccessibilityEvent(AccessibilityEvent)的调用。对于这样的事件添加文本内容的首选方式是为源视图设置内容描述。

1.2 切换类型

1.2.1 窗口状态改变

表示打开弹窗（PopupWindow）、菜单（Menu）、对话框（Dialog）等的事件。

类型：TYPE_WINDOW_STATE_CHANGED。

属性：

- getEventType()-事件类型。
- getSource()-源信息（对于已注册的客户端）。
- getClassName()-源的类名。
- getPackageName()-源的包名。
- getEventTime()-事件时间。
- getText()-源的子树的文本。
- isEnabled()-源是否可用。

1.2.2 窗口内容改变

表示窗口内容改变的事件。此改变可以是添加/删除视图，改变视图大小等。

注意：该事件不同于 TYPE_NOTIFICATION_STATE_CHANGED，仅为最后的无障碍事件的窗口源发送，其目的是通知客户端用户交互窗口中内容的改变。

类型：TYPE_WINDOW_CONTENT_CHANGED。

属性：

- getEventType()-事件的类型。
- getContentChangeTypes()-内容改变的类型。
- getSource()-源信息（对于已注册的客户端）。
- getClassName()-源的类名。
- getPackageName()-源的包名。
- getEventTime()-事件时间。

注意：此事件类型不使用 View.dispatchPopulateAccessibilityEvent(AccessibilityEvent)分发给子视图，因此此事件源的视图和它的子树不会接收 View.onPopulateAccessibilityEvent(AccessibilityEvent)的调用。对于这样的事件添加文本内容的首选方式是为源视图设置内容描述。

1.2.3 窗口改变

表示显示在屏幕上的窗口改变的事件，例如一个窗口出现、一个窗口消失、一个窗口大小改变、一个窗口层级改变等。

类型：TYPE_WINDOWS_CHANGED。

属性：

- getEventType()-事件类型。
- getEventTime()-事件时间。

注意：可以通过 getSource()为事件的窗口源检索 AccessibilityWindowInfo，来获取源节点，然后调用 AccessibilityNodeInfo.getWindow()获取窗口。还可以通过调用 android.accessibilityservice.AccessibilityService.getWindows()检索屏幕上的所

有窗口。

1.3 通知类型

1.3.1 通知状态改变

表示显示通知的事件。

类型：TYPE_NOTIFICATION_STATE_CHANGED。

属性：

- getEventType()-事件类型。
- getClassName()-源的类名。
- getPackageName()-源的包名。
- getEventTime()-事件时间。
- getParcelableData-发布的通知（Notification）。
- getText-提供更多上下文的文本。

注意：此事件类型不使用View.dispatchPopulateAccessibilityEvent(AccessibilityEvent)分发给子视图，因此此事件源的视图和它的子树不会接收View.onPopulateAccessibilityEvent(AccessibilityEvent)调用。对于这样的事件添加文本内容的首选方式是为源视图设置内容描述。

1.4 浏览类型

1.4.1 视图悬停进入

表示开始悬停视图的事件。悬停可能通过触摸浏览屏幕或通过点击设备生成。

类型：TYPE_VIEW_HOVER_ENTER。

属性：

- getEventType()-事件类型。
- getSource()-源信息（对于已注册的客户端）。
- getClassName()-源的类名。
- getPackageName()-源的包名。
- getEventTime()-事件时间。
- getText()-源的子树的文本。
- isEnabled()-源是否可用。
- getContentDescription()-源的内容描述。
- getScrollX()-使用像素表示源左边缘的偏移量(不包含 AdapterView 的子视图)。
- getScrollY()-使用像素表示源顶边缘的偏移量(不包含 AdapterView 的子视图)。
- getFromIndex()-源第一个可见项的从零开始的索引，包含（用于 AdapterView 的子视图）。
- getToIndex()-源最后一个可见项的从零开始的索引，包含（用于 AdapterView 的子视图）。
- getItemCount()-源的总项目数（用于 AdapterView 的子视图）。

1.4.2 视图悬停退出

表示停止悬停视图的事件。悬停可能通过触摸浏览屏幕或通过点击设备生成。

类型：TYPE_VIEW_HOVER_EXIT。

属性：

- getEventType()-事件类型。
- getSource()-源信息（对于已注册的客户端）。
- getClassName()-源的类名。
- getPackageName()-源的包名。
- getEventTime()-事件时间。
- getText()-源的子树的文本。
- isEnabled()-源是否可用。
- getContentDescription()-源的内容描述。
- getScrollX()-使用像素表示源左边缘的偏移量(不包含 AdapterView 的子视

图)。

- getScrollY()-使用像素表示源顶边缘的偏移量(不包含 AdapterView 的子视图)。
- getFromIndex()-源第一个可见项的从零开始的索引，包含（用于 AdapterView 的子视图)。
- getToIndex()-源最后一个可见项的从零开始的索引，包含（用于 AdapterView 的子视图）。
- getItemCount()-源的总项目数（用于 AdapterView 的子视图)。

1.4.3 触摸交互开始

表示开始一个触摸交互的事件，表示用户开始触摸屏幕。

类型：TYPE_TOUCH_INTERACTION_START。

属性：

- getEventType()-事件的类型。

注意：该事件只能被系统发送，且不被传递给视图树进行填充。

1.4.4 触摸交互结束

表示结束一个触摸交互的事件，表示用户停止触摸屏幕。

类型：TYPE_TOUCH_INTERACTION_END。

属性：

- getEventType()-事件的类型。

注意：该事件只能被系统发送，且不被传递给视图树进行填充。

1.4.5 触摸浏览手势开始

表示开始一个触摸浏览手势的事件。

类型：TYPE_TOUCH_EXPLORATION_GESTURE_START。

属性：

- getEventType()-事件的类型。

注意：该事件只能被系统发送，且不被传递给视图树进行填充。

1.4.6 触摸浏览手势结束

表示结束一个触摸浏览手势的事件。

类型：TYPE_TOUCH_EXPLORATION_GESTURE_END。

属性：

- getEventType()-事件的类型。

注意：该事件只被系统发送，而不是传递给视图树进行填充。

1.4.7 触摸手势检测开始

表示开始一个用户手势检测的事件。

类型：TYPE_GESTURE_DETECTION_START。

属性：

- getEventType()-事件的类型。

注意：该事件只能被系统发送，且不被传递给视图树进行填充。

1.4.8 触摸手势检测结束

表示结束一个用户手势检测的事件。

类型：TYPE_GESTURE_DETECTION_END。

属性：

- getEventType-事件的类型。

注意：该事件只能被系统发送，且不被传递给视图树进行填充。

1.5 杂类

1.5.1 通知

表示一个应用程序产生一个通知的事件。通常，该通知是有关某种上下文改变，对于没有事件代表的 UI 转换改变来说是一个不错的选择。例如，宣布一本书的一个新页面。

类型：TYPE_ANNOUNCEMENT。

属性：

- getEventType()-事件类型。
- getSource()-源信息（对于已注册的客户端）。
- getClassName()-源的类名。
- getPackageName()-源的包名。
- getEventTime()-事件时间。
- getText()-源的子树的文本。
- isEnabled()-源是否可用。

1.5.2 安全注意事项

因为一个包含其源隐私文本的事件可能会导致敏感信息的泄露，如密码。为了解决这个问题，为处理密码字段而发送的事件上不可包含密码文本。

参见：

AccessibilityManager

AccessibilityService

AccessibilityNodeInfo

2.摘要

2.1 常量

int	CONTENT_CHANGE_TYPE_CONTENT_DESCRIPTION TYPE_WINDOW_CONTENT_CHANGED 事件的改变类型：该节点的内容描述改变。
int	CONTENT_CHANGE_TYPE_SUBTREE TYPE_WINDOW_CONTENT_CHANGED 事件的改变类型：源节点子树上的一个节点添加或移除。
int	CONTENT_CHANGE_TYPE_TEXT TYPE_WINDOW_CONTENT_CHANGED 事件的改变类型：节点的文本改变。
int	CONTENT_CHANGE_TYPE_UNDEFINED TYPE_WINDOW_CONTENT_CHANGED 事件的改变类型：改变类型未定义。
int	INVALID_POSITION 无效的选择/焦点位置。
int	MAX_TEXT_LENGTH 文本字段的最大长度。
int	TYPES_ALL_MASK AccessibilityEvent 所有类型的掩码。

int	TYPE_ANNOUNCEMENT 表示一个应用程序产生一个通知的事件。
int	TYPE_ASSIST_READING_CONTEXT 表示辅助用户读取当前屏幕上下文的事件。
int	TYPE_GESTURE_DETECTION_END 表示结束手势检测的事件。
int	TYPE_GESTURE_DETECTION_START 表示开始手势检测的事件。
int	TYPE_NOTIFICATION_STATE_CHANGED 表示显示通知（Notification）的事件。
int	TYPE_TOUCH_EXPLORATION_GESTURE_END 表示结束一个触摸浏览手势的事件。
int	TYPE_TOUCH_EXPLORATION_GESTURE_START 表示开始一个触摸浏览手势的事件。
int	TYPE_TOUCH_INTERACTION_END 表示用户结束触摸屏幕的事件。
int	TYPE_TOUCH_INTERACTION_START 表示用户开始触摸屏幕的事件。
int	TYPE_VIEW_ACCESSIBILITY_FOCUSED 表示获得无障碍焦点的事件。

int	TYPE_VIEW_ACCESSIBILITY_FOCUS_CLEARED 表示清除无障碍焦点的事件。
int	TYPE_VIEW_CLICKED 表示点击一个视图的事件，例如按钮（Button）、复合按钮（CompoundButton）等视图。
int	TYPE_VIEW_CONTEXT_CLICKED 表示上下文点击视图的事件。
int	TYPE_VIEW_FOCUSED 表示为一个视图设置输入焦点的事件。
int	TYPE_VIEW_HOVER_ENTER 表示悬停进入视图的事件。
int	TYPE_VIEW_HOVER_EXIT 表示悬停退出视图的事件。
int	TYPE_VIEW_LONG_CLICKED 表示长按点击一个视图的事件，例如按钮（Button）、复合按钮（CompoundButton）等视图。
int	TYPE_VIEW_SCROLLED 表示滚动一个视图的事件。
int	TYPE_VIEW_SELECTED 表示通常在 AdapterView 中选择一个项目的事件。
int	TYPE_VIEW_TEXT_CHANGED

	表示一个编辑框（EditText）内文本改变的事件。
int	TYPE_VIEW_TEXT_SELECTION_CHANGED 表示在一个编辑框（EditText）里改变选择的事件。
int	TYPE_VIEW_TEXT_TRAVERSED_AT_MOVEMENT_GRANULARITY 表示以给定移动粒度遍历视图文本的事件。
int	TYPE_WINDOWS_CHANGED 表示屏幕上显示窗口改变的事件。
int	TYPE_WINDOW_CONTENT_CHANGED 表示改变窗口内容和指定事件源的子树的事件。
int	TYPE_WINDOW_STATE_CHANGED 表示打开一个弹窗（PopupWindow）、菜单（Menu）、对话框（Dialog）等的事件。

2.2 继承常量

来自接口 android.os.Parcelable。

2.3 字段

Public static final Creator<AccessibilityEvent>	CREATOR

2.4 公有方法

Void	appendRecord(AccessibilityRecord record) 在事件记录的末尾添加一个 AccessibilityRecord。
Int	describeContents() 在 Parcelable 实例的编组代表中描述特殊对象包含的种类。
static String	eventTypeToString(int eventType) 返回代表事件类型的字符串。
int	getAction() 获取触发该事件的执行操作。
int	getContentChangeTypes() 获取由 TYPE_WINDOW_CONTENT_CHANGED 事件标识的变更类型的位掩码。
Long	getEventTime() 获取此事件的发送时间。
int	getEventType() 获取事件类型。
int	getMovementGranularity() 获取遍历的移动粒度。
CharSequence	getPackageName() 获取源的包名。
AccessibilityRecord	getRecord(int index)

	获取给定索引下的记录。
int	getRecordCount() 获取包含在事件中的记录数。
void	initFromParcel(Parcel parcel) 从 Parcel 创建一个新实例。
static AccessibilityEvent	obtain(AccessibilityEvent event) 如果可获得，返回一个缓存实例，否则创建一个新的。
static AccessibilityEvent	obtain() 如果可获得，返回一个缓存实例，否则实例化一个新的。
static AccessibilityEvent	obtain(int eventType) 如果可获得，返回一个缓存实例，否则实例化一个具有类型属性的新的实例。
void	recycle() 回收一个实例重复使用。
void	setAction(int action) 设置触发该事件的执行操作。
void	setContentChangeTypes(int changeTypes) 设置被一个 TYPE_WINDOW_CONTENT_CHANGED 事件标识的节点树改变的位掩码。
void	setEventTime(long eventTime) 设置事件被发送的时间。

void	setEventType(int eventType) 设置事件类型。
void	setMovementGranularity(int granularity) 设置遍历的移动粒度。
void	setPackageName(CharSequence packageName) 设置源的包名。
String	toString() 返回代表对象的字符串。
void	writeToParcel(Parcel parcel, int flags) 整合该对象到 Parcel。

2.5 继承方法

来自 android.view.accessibility.AccessibilityRecord 类。

int	getAddedCount() 获取被添加的字符数。
CharSequence	getBeforeText() 获取改变之前的文本。
CharSequence	getClassName() 获取源的类名。
CharSequence	getContentDescription()

	获取源的描述。
int	getCurrentItemIndex() 在可访问的项列表中获取源的索引。
int	getFromIndex() 获取变更序列第一个字符的索引，或文本选择的开端,或滚屏时第一个可见项的索引。
int	getItemCount() 获取可访问的项目数。
int	getMaxScrollX() 获取源左边缘的最大滚动偏移像素数。
int	getMaxScrollY() 获取源顶部边缘的最大滚动偏移像素数。
Parcelable	getParcelableData() 获取 Parcelable 数据。
int	getRemovedCount() 获取删除的字符数。
int	getScrollX() 获取源左边缘的滚动偏移像素数。
int	getScrollY() 获取源顶部边缘的滚动偏移像素数。
AccessibilityNodeInfo	getSource()

	获取事件源的 AccessibilityNodeInfo。
List<CharSequence>	getText() 获取事件文本。
int	getToIndex() 获取文本选择结束的索引，或滚屏时最后一个可见项的索引。
int	getWindowId() 获取事件来源的窗口 ID。
boolean	isChecked() 获取源是否选中。
boolean	isEnabled() 获取源是否可用。
boolean	isFullScreen() 获取源是否占据整个屏幕。
boolean	isPassword() 获取源是否是一个密码字段。
boolean	isScrollable() 获取源是否可滚动。
static AccessibilityRecord	obtain() 如果可获得，返回一个缓存实例，否则实例化一个新的。
static	obtain(AccessibilityRecord record)

AccessibilityRecord	如果可获得，返回一个缓存实例，否则实例化一个新的。
void	recycle() 返回一个实例重复使用。
void	setAddedCount(int addedCount) 设置被添加的字符数。
void	setBeforeText(CharSequence beforeText) 在改变之前设置文本。
void	setChecked(boolean isChecked) 如果源可选中，设置。
void	setClassName(CharSequence className) 设置源的类名。
void	setContentDescription(CharSequence contentDescription) 设置源的描述。
void	setCurrentItemIndex(int currentItemIndex) 设置可访问的项列表中源的索引。
void	setEnabled(boolean isEnabled) 如果源可用，设置。
void	setFromIndex(int fromIndex) 设置已变更序列的第一个字符索引，或文本选择的开端，或滚屏时第一个可见项的索引。
void	setFullScreen(boolean isFullScreen)

	如果源可占用整个屏幕，设置。
void	setItemCount(int itemCount) 设置可访问的项目数。
void	setMaxScrollX(int maxScrollX) 设置源左边缘的最大滚动偏移像素数。
void	setMaxScrollY(int maxScrollY) 设置源顶部边缘的最大滚动偏移像素数。
void	setParcelableData(Parcelable parcelableData) 设置事件的 Parcelable 数据。
void	setPassword(boolean isPassword) 如果源是一个密码字段，设置。
void	setRemovedCount(int removedCount) 设置删除的字符数。
void	setScrollX(int scrollX) 设置源左边缘的滚动偏移像素数。
void	setScrollY(int scrollY) 设置源顶部边缘的滚动偏移像素数。
void	setScrollable(boolean scrollable) 如果源可滚动，设置。
void	setSource(View root, int virtualDescendantId)

	设置源为给定 root 的虚拟子节点。
void	setSource(View source) 设置事件源。
void	setToIndex(int toIndex) 设置文本选择结束的索引，或滚屏时最后一个可见项的索引。
String	toString() 返回对象的字符串表示。

来自类 java.lang.Object

Object	clone() 创建并返回该对象的复本。
boolean	equals(Object obj) 标识某些其他对象是否"等同于"该对象。
void	finalize() 当垃圾收集器确认再也没有该对象的引用时，垃圾收集器调用该方法。
final Class<?>	getClass() 返回该对象的运行类。
int	hashCode() 为该对象返回哈希编码值。
final void	Notify()

	唤醒在对象监视器上等待的单线程。
final void	notifyAll() 唤醒在对象监视器上等待的所有线程。
String	toString() 返回对象的字符串表示。
final void	wait(long millis, int nanos) 导致当前线程等待，直到另一线程为该对象调用 Notify()方法或 notifyAll()方法，或某些其他线程中断当前线程，或一定数量的实时运行已经停止。
final void	wait(long millis) 导致当前线程等待，直到另一线程为该对象调用 Notify()方法或 notifyAll()方法，或一定数量的实时运行已经停止。
final void	wait() 导致当前线程等待，直到另一线程为该对象调用 Notify()方法或 notifyAll()方法。

来自接口 android.os.Parcelable

abstract int	describeContents() 描述被包含在 Parcelable 实例的编组代表中的各类特殊对象。
abstract void	writeToParcel(Parcel dest, int flags) 整合该对象到一个 Parcel。

3.常量

3.1CONTENT_CHANGE_TYPE_CONTENT_DESCRIPTION

添加于 API 级别 19。

int CONTENT_CHANGE_TYPE_CONTENT_DESCRIPTION

TYPE_WINDOW_CONTENT_CHANGED 事件的改变类型:该节点的内容描述改变。

常量值：4 (0x00000004)

3.2CONTENT_CHANGE_TYPE_SUBTREE

添加于 API 级别 19。

int CONTENT_CHANGE_TYPE_SUBTREE

TYPE_WINDOW_CONTENT_CHANGED 事件的改变类型：源节点子树上一个节点添加或移除。

常量值: 1 (0x00000001)

3.3CONTENT_CHANGE_TYPE_TEXT

添加于 API 级别 19。

int CONTENT_CHANGE_TYPE_TEXT

TYPE_WINDOW_CONTENT_CHANGED 事件的改变类型:节点的文本改变。

常量值: 2 (0x00000002)

3.4CONTENT_CHANGE_TYPE_UNDEFINED

添加于 API 级别 19。

int CONTENT_CHANGE_TYPE_UNDEFINED

TYPE_WINDOW_CONTENT_CHANGED 事件的改变类型：改变类型未定义。

常量值: 0 (0x00000000)

3.5INVALID_POSITION

添加于 API 级别 4。

int INVALID_POSITION

无效的选择/焦点位置。

参见:

getCurrentItemIndex()

常量值: -1 (0xffffffff)

3.6MAX_TEXT_LENGTH

添加于 API 级别 4。

int MAX_TEXT_LENGTH

文本域的最大长度。

参见：

getBeforeText()

注意：自从在无障碍事件里包含的文本的长度没有限制后，该常量就不再需要了。

常量值: 500 (0x000001f4)

3.7TYPES_ALL_MASK

添加于 API 级别 4。

int TYPES_ALL_MASK

所有 AccessibilityEvent 类型的掩码。

参见:

TYPE_VIEW_CLICKED

TYPE_VIEW_LONG_CLICKED

TYPE_VIEW_SELECTED

TYPE_VIEW_FOCUSED

TYPE_VIEW_TEXT_CHANGED

TYPE_WINDOW_STATE_CHANGED

TYPE_NOTIFICATION_STATE_CHANGED

TYPE_VIEW_HOVER_ENTER

TYPE_VIEW_HOVER_EXIT

TYPE_TOUCH_EXPLORATION_GESTURE_START

TYPE_TOUCH_EXPLORATION_GESTURE_END

TYPE_WINDOW_CONTENT_CHANGED

TYPE_VIEW_SCROLLED

TYPE_VIEW_TEXT_SELECTION_CHANGED

TYPE_ANNOUNCEMENT

TYPE_VIEW_TEXT_TRAVERSED_AT_MOVEMENT_GRANULARITY

TYPE_GESTURE_DETECTION_START

TYPE_GESTURE_DETECTION_END

TYPE_TOUCH_INTERACTION_START

TYPE_TOUCH_INTERACTION_END

TYPE_WINDOWS_CHANGED

TYPE_VIEW_CONTEXT_CLICKED

常量值: -1 (0xffffffff)

3.8TYPE_ANNOUNCEMENT

添加于 API 级别 16。

int TYPE_ANNOUNCEMENT

表示一个应用程序产生一个通知的事件。

常量值: 16384 (0x00004000)

3.9TYPE_ASSIST_READING_CONTEXT

添加于 API 级别 23。

int TYPE_ASSIST_READING_CONTEXT

表示辅助用户读取当前屏幕上下文的事件。

常量值: 16777216 (0x01000000)

3.10TYPE_GESTURE_DETECTION_END

添加于 API 级别 17。

int TYPE_GESTURE_DETECTION_END

表示结束手势检测的事件。

常量值: 524288 (0x00080000)

3.11TYPE_GESTURE_DETECTION_START

添加于 API 级别 17。

int TYPE_GESTURE_DETECTION_START

表示开始手势检测的事件。

常量值: 262144 (0x00040000)

3.12TYPE_NOTIFICATION_STATE_CHANGED

添加于 API 级别 4。

int TYPE_NOTIFICATION_STATE_CHANGED

表示显示一个通知（Notification）的事件。

常量值: 64 (0x00000040)

3.13TYPE_TOUCH_EXPLORATION_GESTURE_END

添加于 API 级别 14。

int TYPE_TOUCH_EXPLORATION_GESTURE_END

表示结束一个触摸浏览手势的事件。

常量值: 1024 (0x00000400)

3.14TYPE_TOUCH_EXPLORATION_GESTURE_START

添加于 API 级别 14。

int TYPE_TOUCH_EXPLORATION_GESTURE_START

表示开始一个触摸浏览手势的事件。

常量值: 512 (0x00000200)

3.15TYPE_TOUCH_INTERACTION_END

添加于 API 级别 17。

int TYPE_TOUCH_INTERACTION_END

表示用户结束触摸屏幕的事件。

常量值: 2097152 (0x00200000)

3.16TYPE_TOUCH_INTERACTION_START

添加于 API 级别 17。

int TYPE_TOUCH_INTERACTION_START

表示用户开始触摸屏幕的事件。

常量值: 1048576 (0x00100000)

3.17TYPE_VIEW_ACCESSIBILITY_FOCUSED

添加于 API 级别 16。

int TYPE_VIEW_ACCESSIBILITY_FOCUSED

表示获取无障碍焦点的事件。

常量值: 32768 (0x00008000)

3.18TYPE_VIEW_ACCESSIBILITY_FOCUS_CLEARED

添加于 API 级别 16。

int TYPE_VIEW_ACCESSIBILITY_FOCUS_CLEARED

表示清除无障碍焦点的事件。

常量值: 65536 (0x00010000)

3.19TYPE_VIEW_CLICKED

添加于 API 级别 4。

int TYPE_VIEW_CLICKED

表示点击一个视图的事件，例如按钮（Button）、复合按钮（CompoundButton）等视图。

常量值: 1 (0x00000001)

3.20TYPE_VIEW_CONTEXT_CLICKED

添加于 API 级别 23。

int TYPE_VIEW_CONTEXT_CLICKED

表示点击一个视图上下文的事件。

常量值: 8388608 (0x00800000)

3.21TYPE_VIEW_FOCUSED

添加于 API 级别 4。

int TYPE_VIEW_FOCUSED

表示给一个视图设置输入焦点的事件。

常量值: 8 (0x00000008)

3.22TYPE_VIEW_HOVER_ENTER

添加于 API 级别 14。

int TYPE_VIEW_HOVER_ENTER

表示悬停进入视图的事件。

常量值: 128 (0x00000080)

3.23TYPE_VIEW_HOVER_EXIT

添加于 API 级别 14。

int TYPE_VIEW_HOVER_EXIT

表示悬停退出视图的事件。

常量值: 256 (0x00000100)

3.24TYPE_VIEW_LONG_CLICKED

添加于 API 级别 4。

int TYPE_VIEW_LONG_CLICKED

表示长按点击一个视图的事件，例如按钮（Button）、复合按钮（CompoundButton）等视图。

常量值: 2 (0x00000002)

3.25TYPE_VIEW_SCROLLED

添加于 API 级别 14。

int TYPE_VIEW_SCROLLED

表示滚动一个视图的事件。

常量值: 4096 (0x00001000)

3.26TYPE_VIEW_SELECTED

添加于 API 级别 4。

int TYPE_VIEW_SELECTED

表示通常在 AdapterView 的上下文中选择一个项目的事件。

常量值: 4 (0x00000004)

3.27TYPE_VIEW_TEXT_CHANGED

添加于 API 级别 4。

int TYPE_VIEW_TEXT_CHANGED

表示一个编辑框（EditText）文本改变的事件。

常量值: 16 (0x00000010)

3.28TYPE_VIEW_TEXT_SELECTION_CHANGED

添加于 API 级别 14。

int TYPE_VIEW_TEXT_SELECTION_CHANGED

表示在一个编辑框（EditText）里选择改变的事件。

常量值: 8192 (0x00002000)

3.29TYPE_VIEW_TEXT_TRAVERSED_AT_MOVEMENT_GRANULARITY

添加于 API 级别 16。

int TYPE_VIEW_TEXT_TRAVERSED_AT_MOVEMENT_GRANULARITY

表示以给定移动粒度遍历视图文本的事件。

常量值: 131072 (0x0020000)

3.30TYPE_WINDOWS_CHANGED

添加于 API 级别 21。

int TYPE_WINDOWS_CHANGED

表示屏幕上显示窗口改变的事件。

常量值: 4194304 (0x00400000)

3.31TYPE_WINDOW_CONTENT_CHANGED

添加于 API 级别 14。

int TYPE_WINDOW_CONTENT_CHANGED

表示改变窗口内容和指定事件源的子树的事件。

常量值: 2048 (0x00000800)

3.32TYPE_WINDOW_STATE_CHANGED

添加于 API 级别 4。

int TYPE_WINDOW_STATE_CHANGED

表示打开弹窗（PopupWindow）、菜单（Menu）、对话框（Dialog）等的事

件。

常量值: 32 (0x00000020)

4.字段

4.1CREATOR

添加于 API 级别 4。

Creator<AccessibilityEvent> CREATOR

参见:

Parcelable.Creator

5.公有方法

5.1appendRecord

添加于 API 级别 14。

appendRecord (AccessibilityRecord record)

添加一个 AccessibilityRecord 到事件记录的结尾。

参数：

record	AccessibilityRecord：要添加的记录。

抛出：

IllegalStateException	如果被一个无障碍服务调用。

5.2describeContents

添加于 API 级别 4。

int describeContents ()

描述被包含在 Parcelable 实例的编组表示中的特殊对象种类。例如，如果对象在 writeToParcel(Parcel, int)的输出里包括一个文件描述符，该方法的返回值必须包括 CONTENTS_FILE_DESCRIPTOR 位。

返回值：

int	标识被该包对象实例编组的特殊对象类型设置的位掩码。

5.3eventTypeToString

添加于 API 级别 14。

String eventTypeToString (int eventType)

返回一个事件类型的字符串表示。例如，TYPE_VIEW_CLICKED 的字符串表示为“TYPE_VIEW_CLICKED”。

参数：

eventType	int：事件类型。

返回值：

String	字符串表示。

5.4getAction

添加于 API 级别 16。

int getAction()

获取触发该事件的执行操作。

返回值：

int	操作。

5.5getContentChangeTypes

添加于 API 级别 19。

int getContentChangeTypes ()

获取由 TYPE_WINDOW_CONTENT_CHANGED 事件标识的改变类型的位掩码。一个单一事件可能代表多种变化类型。

返回值：

Int	变化类型的位掩码。一个或多个的： ● CONTENT_CHANGE_TYPE_CONTENT_DESCRIPTION ● CONTENT_CHANGE_TYPE_SUBTREE ● CONTENT_CHANGE_TYPE_TEXT ● CONTENT_CHANGE_TYPE_UNDEFINED

5.6getEventTime

添加于 API 级别 4。

long getEventTime ()

获取该事件的发送时间。

返回值：

long	事件时间。

5.7getEventType

添加于 API 级别 4。

int getEventType ()

获取事件类型。

返回值：

int	事件类型。

5.8getMovementGranularity

添加于 API 级别 16。

int getMovementGranularity ()

获取遍历的移动粒度。

返回值：

int	粒度。

5.9getPackageName

添加于 API 级别 4。

CharSequence getPackageName ()

获取源的包名。

返回值：

CharSequence	包名。

5.10getRecord

添加于 API 级别 4。

AccessibilityRecord getRecord (int index)

获取给定索引下的记录。

参数：

index	int：索引。

返回值：

AccessibilityRecord	指定索引下的记录。

5.11getRecordCount

添加于 API 级别 14。

int getRecordCount ()

获取被包含在事件中的记录数。

返回值：

int	记录数。

5.12initFromParcel

添加于 API 级别 4。

void initFromParcel (Parcel parcel)

从 Parcel 创建一个新实例。

参数：

parcel	Parcel：一个包括 AccessibilityEvent 状态的包。

5.13obtain

添加于 API 级别 14。

AccessibilityEvent obtain (AccessibilityEvent event)

如果可获得，返回一个缓存实例或创建一个新实例。返回实例从给定事件初始化。

参数：

event	AccessibilityEvent：其他的事件。

返回值：

AccessibilityEvent	一个实例。

5.14obtain

添加于 API 级别 4。

AccessibilityEvent obtain ()

如果可获得，返回一个缓存实例或实例化一个新的。

返回值：

AccessibilityEvent	一个实例。

5.15obtain

添加于 API 级别 4。

AccessibilityEvent obtain (int eventType)

如果可获得，返回一个缓存实例或实例化一个新的并设置它的类型属性。

参数：

eventType	int：事件类型。

返回值：

AccessibilityEvent	一个实例。

5.16recycle

添加于 API 级别 4。

void recycle ()

回收一个实例重复使用。

注意：在调用此函数后，不能触摸该对象。

抛出：

IllegalStateException	如果该事件已经被回收。

5.17setAction

添加于 API 级别 4。

void setAction (int action)

设置触发此事件的执行操作。

在 AccessibilityNodeInfo 中定义的有效操作:

- ACTION_ACCESSIBILITY_FOCUS
- ACTION_CLEAR_ACCESSIBILITY_FOCUS
- ACTION_CLEAR_FOCUS
- ACTION_CLEAR_SELECTION
- ACTION_CLICK
- 等。

参数：

action	int：操作。

抛出：

IllegalStateException	如果被一个无障碍服务调用。

参见:

performAction(int)

5.18setContentChangeTypes

添加于 API 级别 19。

void setContentChangeTypes (int changeTypes)

设置由一个 TYPE_WINDOW_CONTENT_CHANGED 事件标识的节点树改变的位掩码。

参数：

changeTypes	int：改变类型的位掩码。

抛出：

IllegalStateException	如果被一个无障碍服务调用。

参见:

getContentChangeTypes（）

5.19setEventTime

添加于 API 级别 4。

void setEventTime (long eventTime)

设置事件被发送的时间。

参数：

eventTime	long：事件时间。

抛出：

IllegalStateException	如果被一个无障碍服务调用。

5.20setEventType

添加于 API 级别 4。

void setEventType (int eventType)

设置事件类型。

参数：

eventType	int：事件类型。

抛出：

IllegalStateException	如果被一个无障碍服务调用。

5.21setMovementGranularity

添加于 API 级别 16。

void setMovementGranularity (int granularity)

设置遍历的移动粒度。

参数：

granularity	int：粒度。

抛出：

IllegalStateException	如果被一个无障碍服务调用。

5.22setPackageName

添加于 API 级别 4。

void setPackageName (CharSequence packageName)

设置源的包名。

参数：

packageName	CharSequence：包名。

抛出：

IllegalStateException	如果被一个无障碍服务调用。

5.23toString

添加于 API 级别 4。

String toString ()

返回对象的字符串表示。一般情况下，toString 方法返回一个“文本表示”该对象的字符串。结果应该是一个简洁但容易阅读的信息表示。建议所有子类重写该方法。

toString 方法为类 Object 返回一个字符串，包含实例对象的类名、字符‘@’和

该对象哈希编码的无符号十六进制表示。换言之，该方法返回一个字符串的值等于：

getClass().getName() + '@' + Integer.toHexString(hashCode())

返回值：

String	对象的字符串表示。

5.24writeToParcel

添加于 API 级别 4。

void writeToParcel (Parcel parcel, int flags)

整组该对象到一个 Parcel。

参数：

parcel	Parcel：对象应该被写入的 Parcel。
flags	int: 关于对象应该如何被写入的附加标识。可能是 0 或 PARCELABLE_WRITE_RETURN_VALUE。