

AccessibilityNodeInfo

中国信息无障碍产品联盟&信息无障碍研究会 译制

20161130

目录

翻译声明.....	1
AccessibilityNodeInfo	2
1.开发指南.....	3
2.概要.....	4
2.1 嵌套类.....	4
2.2 常量.....	4
2.3 继承常量.....	8
2.4 字段.....	9
2.5 公有方法.....	9
2.6 继承方法.....	22
3.常量.....	25
3.1ACTION_ACCESSIBILITY_FOCUS	25
3.2ACTION_ARGUMENT_COLUMN_INT	25
3.3ACTION_ARGUMENT_EXTEND_SELECTION_BOOLEAN	25
3.4ACTION_ARGUMENT_HTML_ELEMENT_STRING	26
3.5ACTION_ARGUMENT_MOVEMENT_GRANULARITY_INT	27
3.7ACTION_ARGUMENT_PROGRESS_VALUE	27
3.8ACTION_ARGUMENT_ROW_INT.....	28
3.9ACTION_ARGUMENT_SELECTION_END_INT	28
3.10ACTION_ARGUMENT_SELECTION_START_INT	29
3.11ACTION_ARGUMENT_SET_TEXT_CHARSEQUENCE.....	29
3.12ACTION_CLEAR_ACCESSIBILITY_FOCUS	30
3.13ACTION_CLEAR_FOCUS	30
3.14ACTION_CLEAR_SELECTION.....	31
3.15ACTION_CLICK	31
3.16ACTION_COLLAPSE	31
3.17ACTION_COPY	31
3.18ACTION_CUT	32
3.19ACTION_DISMISS.....	32
3.20ACTION_EXPAND.....	32
3.21ACTION_FOCUS.....	32

3.22ACTION_LONG_CLICK	33
3.23ACTION_NEXT_AT_MOVEMENT_GRANULARITY	33
3.24ACTION_NEXT_HTML_ELEMENT	34
3.25ACTION_PASTE.....	35
3.26ACTION_PREVIOUS_AT_MOVEMENT_GRANULARITY	35
3.27ACTION_PREVIOUS_HTML_ELEMENT	36
3.28ACTION_SCROLL_BACKWARD	37
3.29ACTION_SCROLL_FORWARD.....	37
3.30ACTION_SELECT	38
3.31ACTION_SET_SELECTION.....	38
3.32ACTION_SET_TEXT	39
3.33FOCUS_ACCESSIBILITY	39
3.34FOCUS_INPUT	39
3.35MOVEMENT_GRANULARITY_CHARACTER	40
3.36MOVEMENT_GRANULARITY_LINE	40
3.37MOVEMENT_GRANULARITY_PAGE	40
3.38MOVEMENT_GRANULARITY_PARAGRAPH.....	41
3.39MOVEMENT_GRANULARITY_WORD	41
4.字段.....	42
5.公有方法.....	43
5.1addAction	43
5.2addAction	43
5.3addChild.....	44
5.4addChild	45
5.5canOpenPopup	45
5.6describeContents	45
5.7equals	46
5.8findAccessibilityNodeInfosByText.....	47
5.9findAccessibilityNodeInfosByViewId	47
5.10findFocus.....	48
5.11focusSearch	49
5.12getActionList.....	49
5.13getActions	50

5.14getBoundsInParent	51
5.15getBoundsInScreen	51
5.16getChild	51
5.17getChildCount	52
5.18getClassName	52
5.19getCollectionInfo	53
5.20getCollectionItemInfo	53
5.21getContentDescription	53
5.22getDrawingOrder	54
5.23getError	54
5.24getExtras	54
5.25getInputType	55
5.26getLabelFor	55
5.27getLabeledBy	56
5.28getLiveRegion	56
5.29getMaxTextLength	57
5.30getMovementGranularities	57
5.31getPackageName	57
5.32getParent	58
5.33getRangeInfo	58
5.34getText	58
5.35getTextSelectionEnd	59
5.36getTextSelectionStart	59
5.37getTraversalAfter	60
5.38getTraversalBefore	60
5.39getViewIdResourceName	61
5.40getWindow	61
5.41getWindowId	62
5.42hashCode	62
5.43isAccessibilityFocused	63
5.44isCheckable	63
5.45isChecked	63
5.46isClickable	64

5.47isContentInvalid	64
5.48isContextClickable	64
5.49isDismissable	65
5.50isEditable.....	65
5.51isEnabled.....	65
5.52isFocusable.....	66
5.53isFocused.....	66
5.53isImportantForAccessibility.....	66
5.54isLongClickable	67
5.55isMultiLine.....	67
5.56isPassword.....	68
5.57isScrollable.....	68
5.58isSelected	68
5.59isVisibleToUser	69
5.60obtain.....	69
5.61obtain.....	70
5.62obtain.....	70
5.63obtain.....	70
5.64performAction.....	71
5.65performAction.....	72
5.66recycle	72
5.67refresh	73
5.68removeAction.....	73
5.69removeAction.....	74
5.70removeChild.....	74
5.71removeChild.....	75
5.72setAccessibilityFocused	76
5.73setBoundsInParent	76
5.74setBoundsInScreen.....	77
5.75setCanOpenPopup.....	77
5.76setCheckable	78
5.77setChecked	78
5.78setClassName	79

5.79setClickable.....	79
5.80setCollectionInfo.....	80
5.81setCollectionItemInfo	80
5.82setContentDescription.....	81
5.83setContentInvalid	81
5.84setContextClickable	82
5.85setDismissable.....	82
5.86setDrawingOrder.....	82
5.87setEditable.....	83
5.88setEnabled	83
5.89setError.....	84
5.90setFocusable	84
5.91setFocused.....	85
5.92setImportantForAccessibility	86
5.93setInputType.....	86
5.94setLabelFor	86
5.95setLabelFor	87
5.96setLabeledBy.....	87
5.97setLabeledBy.....	88
5.98setLiveRegion	88
5.99setLongClickable	89
5.100setMaxTextLength	89
5.101setMovementGranularities	90
5.102setMultiLine	91
5.103setPackageName	91
5.104setParent.....	92
5.105setParent.....	92
5.106setPassword.....	93
5.107setRangeInfo	93
5.108setScrollable	94
5.109setSelected.....	94
5.110setSource	95
5.111 setSource	95

5.112setText	96
5.113 setTextSelection	96
5.114setTraversalAfter	97
5.115setTraversalAfter	97
5.116setTraversalBefore	98
5.117setTraversalBefore	99
5.118setViewIdResourceName	99
5.119setVisibleToUser	100
5.120toString.....	100
5.121writeToParcel	101

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：刘辉

审阅：李鸿利、刘彪、沈广荣

本文档翻译自谷歌官方文档《[AccessibilityNodeInfo](#)》。如您对翻译文档内容有异议，请将原文文档作为主要参考，原文版权由 Google 持有并保留。

本翻译文档使用请参见 [CC BY-NC-SA 3.0](#)。文档可以免费使用、分享，但请保留本链接，如您对内容上有任何的意见或疑问，请发送邮件至 liuhui@siaa.org.cn，我们只是希望文档内容能够统一完整，真正帮助开发者完善产品的信息无障碍。

AccessibilityNodeInfo

添加于 [API 级别 14](#)。

public class AccessibilityNodeInfo

extends [Object](#) implements [Parcelable](#)

[java.lang.Object](#)

↳ [android.view.accessibility.AccessibilityNodeInfo](#)

该类代表一个窗口内容节点和可以从源请求的操作。从 [AccessibilityService](#) 的角度看，一个窗口内容被呈现为一个无障碍节点信息树，该树可能与视图层次一一映射，也可能不与视图层次一一映射。换句话说，一个自定义视图可灵活地将自己报告为一个无障碍节点信息树。

一旦无障碍节点信息被发送给无障碍服务，该信息将会是不可改变的，且调用状态改变方法将会产生错误。

请参考 [AccessibilityService](#)，了解更多关于怎样获得将窗口内容处理为一个无障碍节点信息树，和关于安全模型的详细信息。

1.开发指南

更多关于让应用无障碍的信息，详见 [Android 无障碍开发指南](#)。

参见：

[AccessibilityService](#)

[AccessibilityEvent](#)

[AccessibilityManager](#)

2.概要

2.1 嵌套类

class	AccessibilityNodeInfo.AccessibilityAction 定义了一个可以在 AccessibilityNodeInfo 上执行的操作的类。
class	AccessibilityNodeInfo.CollectionInfo 具有一个节点是否是个集合的信息的类。
class	AccessibilityNodeInfo.CollectionItemInfo 具有一个节点是否是个集合项目的信息的类。
class	AccessibilityNodeInfo.RangeInfo 具有一个节点是否是个范围的信息的类。

2.2 常量

int	ACTION_ACCESSIBILITY_FOCUS 给节点添加无障碍焦点的操作。
String	ACTION_ARGUMENT_COLUMN_INT 让指定集合列在屏幕上可见的参数。
String	ACTION_ARGUMENT_EXTEND_SELECTION_BOOLEAN 当以一定粒度移动时，是否扩大选择范围或反之移除的参数。

String	<code>ACTION_ARGUMENT_HTML_ELEMENT_STRING</code> 要移动到的下一个/上一个 HTML 元素的参数。
String	<code>ACTION_ARGUMENT_MOVEMENT_GRANULARITY_INT</code> 当遍历节点文本的时，使用哪种移动粒度的参数。
String	<code>ACTION_ARGUMENT_PROGRESS_VALUE</code> 指定要设置的进度值的参数。
String	<code>ACTION_ARGUMENT_ROW_INT</code> 让指定集合行在屏幕上可见的参数。
String	<code>ACTION_ARGUMENT_SELECTION_END_INT</code> 指定选择结束的参数。
String	<code>ACTION_ARGUMENT_SELECTION_START_INT</code> 指定选择开始的参数。
String	<code>ACTION_ARGUMENT_SET_TEXT_CHARSEQUENCE</code> 指定要设置的文本内容的参数。
int	<code>ACTION_CLEAR_ACCESSIBILITY_FOCUS</code> 清除节点的无障碍焦点的操作。
int	<code>ACTION_CLEAR_FOCUS</code> 清除节点输入焦点的操作。
int	<code>ACTION_CLEAR_SELECTION</code> 取消选择节点的操作。

int	ACTION_CLICK 在节点信息上点击的操作。
int	ACTION_COLLAPSE 折叠一个可展开节点的操作。
int	ACTION_COPY 将当前选择拷贝到剪贴板的操作。
int	ACTION_CUT 剪切当前选择并将其放置到剪贴板的操作。
int	ACTION_DISMISS 关闭一个可关闭节点的操作。
int	ACTION_EXPAND 展开一个可展开节点的操作。
int	ACTION_FOCUS 给节点添加输入焦点的操作。
int	ACTION_LONG_CLICK 在节点上点击长按的操作。
int	ACTION_NEXT_AT_MOVEMENT_GRANULARITY 以给定移动粒度，请求去到该节点文本的下一个文本实体的操作。
int	ACTION_NEXT_HTML_ELEMENT 移动到给定类型的下一个 HTML 元素的操作。

int	ACTION_PASTE 粘贴当前剪贴板内容的操作。
int	ACTION_PREVIOUS_AT_MOVEMENT_GRANULARITY 以给定移动粒度，请求去到该节点文本的上一个文本实体的操作。
int	ACTION_PREVIOUS_HTML_ELEMENT 移动到给定类型的上一个 HTML 元素的操作。
int	ACTION_SCROLL_BACKWARD 向后滚动节点内容的操作。
int	ACTION_SCROLL_FORWARD 向前滚动节点内容的操作。
int	ACTION_SELECT 选择节点的操作。
int	ACTION_SET_SELECTION 设置选择项的操作。
int	ACTION_SET_TEXT 设置节点文本的操作。
int	FOCUS_ACCESSIBILITY 无障碍焦点。
int	FOCUS_INPUT 输入焦点。

int	MOVEMENT_GRANULARITY_CHARACTER 以字符为移动粒度位，遍历节点文本。
int	MOVEMENT_GRANULARITY_LINE 以行为移动粒度位，遍历节点文本。
int	MOVEMENT_GRANULARITY_PAGE 以页为移动粒度位，遍历节点文本。
int	MOVEMENT_GRANULARITY_PARAGRAPH 以段为移动粒度位，遍历节点文本。
int	MOVEMENT_GRANULARITY_WORD 以字词为移动粒度位遍历节点文本。

2.3 继承常量

继承自接口 [android.os.Parcelable](#)。

int	CONTENTS_FILE_DESCRIPTOR 用于 describeContents() 的描述位：标识可打包对象的封装代表包含一个文件描述符。
int	PARCELABLE_WRITE_RETURN_VALUE 用于 writeToParcel(Parcel, int) 的标识：被写入对象是个返回值，是个函数的结果，例如“ Parcelable someFunction() ”、“ void someFunction(out Parcelable) ”、或“ void someFunction(inout Parcelable) ”。

2.4 字段

public static final	CREATOR
Creator<AccessibilityNodeInfo>	

2.5 公有方法

void	<p>addAction(int action)</p> <p>该方法在 API 级别 21 被弃用，被 addAction(AccessibilityAction) 代替。</p>
void	<p>addAction(AccessibilityNodeInfo.AccessibilityAction action)</p> <p>添加一个可以在节点上执行的操作。</p>
void	<p>addChild(View root, int virtualDescendantId)</p> <p>添加一个虚拟子元素，作为给定 root 的后代。</p>
void	<p>addChild(View child)</p> <p>添加一个子元素。</p>
boolean	<p>canOpenPopup()</p> <p>获取该节点是否可以打开一个弹窗或对话框。</p>
int	<p>describeContents()</p> <p>描述被包含在可打包实例的封装代表中的</p>

	特殊对象的种类。
boolean	<code>equals(Object object)</code> 标识是否有某些其他对象“等同于”该对象。
<code>List<AccessibilityNodeInfo></code>	<code>findAccessibilityNodeInfosByText(String text)</code> 使用文本找到 <code>AccessibilityNodeInfo</code> 。
<code>List<AccessibilityNodeInfo></code>	<code>findAccessibilityNodeInfosByViewId(String viewId)</code> 使用完全合格视图 id 的源名称找到 <code>AccessibilityNodeInfo</code> ，完全合格 id 的样式如下 “package:id/id_resource_name”。
<code>AccessibilityNodeInfo</code>	<code>findFocus(int focus)</code> 找到具有指定焦点类型的视图。
<code>AccessibilityNodeInfo</code>	<code>focusSearch(int direction)</code> 在指定方向搜索具有输入焦点的最近的视图。
<code>List<AccessibilityNodeInfo.AccessibilityAction></code>	<code>getActionList()</code> 获取可以在该节点上执行的操作。
int	<code>getActions()</code> 该方法在 API 级别 21 被弃用。使用 <code>getActionList()</code> 代替。
void	<code>getBoundsInParent(Rect outBounds)</code> 获取父级坐标中的节点边界。

void	<code>getBoundsInScreen(Rect outBounds)</code> 获取屏幕坐标中的节点边界。
<code>AccessibilityNodeInfo</code>	<code>getChild(int index)</code> 获取给定索引下的子元素。
int	<code>getChildCount()</code> 获取子元素的数目。
<code>CharSequence</code>	<code>getClassName()</code> 获取该节点来自的类。
<code>AccessibilityNodeInfo.CollectionInfo</code>	<code>getCollectionInfo()</code> 如果节点是个集合，获取集合信息。
<code>AccessibilityNodeInfo.CollectionItemInfo</code>	<code>getCollectionItemInfo()</code> 如果该节点是个集合项目，获取该集合项目的信息。
<code>CharSequence</code>	<code>getContentDescription()</code> 获取该节点的内容描述。
int	<code>getDrawingOrder()</code> 获取该节点中视图的相对绘制顺序。
<code>CharSequence</code>	<code>getError()</code> 获取该节点的错误文本。
<code>Bundle</code>	<code>getExtras()</code> 获取具有附加数据的可选 bundle。

int	<p>getInputType()</p> <p>获取定义为 InputTyle 的源中的输入类型。</p>
AccessibilityNodeInfo	<p>getLabelFor()</p> <p>为无障碍目的, 获取作为标签代表该视图的节点信息。</p>
AccessibilityNodeInfo	<p>getLabeledBy()</p> <p>为无障碍目的, 获取作为标签代表该视图的节点信息。</p>
int	<p>getLiveRegion()</p> <p>获取该节点的实时区域模式。</p>
int	<p>getMaxTextLength()</p> <p>返回该节点的最大文本长度。</p>
int	<p>getMovementGranularities()</p> <p>获取遍历该节点文本的移动粒度。</p>
CharSequence	<p>getPackageName()</p> <p>获取该节点来自的包。</p>
AccessibilityNodeInfo	<p>getParent()</p> <p>获取父级。</p>
AccessibilityNodeInfo.RangeInfo	<p>getRangeInfo()</p> <p>如果该节点是个范围, 获取范围信息。</p>
CharSequence	<p>getText()</p>

	获取该节点的文本。
int	<code>getTextSelectionEnd()</code> 如果文本已被选择，获取文本选择结尾。
int	<code>getTextSelectionStart()</code> 获取文本选择起始或光标位置。
<code>AccessibilityNodeInfo</code>	<code>getTraversalAfter()</code> 当前节点被访问后，获取无障碍遍历中的下一个节点。
<code>AccessibilityNodeInfo</code>	<code>getTraversalBefore()</code> 当前节点被访问后，获取无障碍遍历中的上一个节点。
<code>String</code>	<code>getViewIdResourceName()</code> 获取源视图 id 的完全合格源名称。
<code>AccessibilityNodeInfo</code>	<code>getWindow()</code> 获取该节点所属的窗口。
int	<code>getWindowId()</code> 获取该信息来自的窗口的 id。
int	<code>hashCode()</code> 返回该对象的哈希编码值。
boolean	<code>isAccessibilityFocused()</code> 获取该节点是否可被无障碍聚焦。

boolean	isCheckedable() 获取该节点是否可选中。
boolean	isCheckeded() 获取该节点是否已选中。
boolean	isClickable() 获取该节点是否可点击。
boolean	isContentInvalid() 获取该节点的内容是否已失效。
boolean	isContextClickable() 获取该节点是否上下文可点击。
boolean	isDismissable() 获取该节点是否可被关闭。
boolean	isEditable() 获取该节点是否可编辑。
boolean	isEnabled() 获取该节点是否可用。
boolean	isFocusable() 获取该节点是否可聚焦。
boolean	isFocused() 获取该节点是否已被聚焦。

boolean	<p>isImportantForAccessibility()</p> <p>返回该节点是否来源于一个对无障碍相当重要的视图。</p>
boolean	<p>isLongClickable()</p> <p>获取该节点是否可长按点击。</p>
boolean	<p>isMultiLine()</p> <p>获取该节点是否是个多行可编辑文本。</p>
boolean	<p>isPassword()</p> <p>获取该节点是否是个密码。</p>
boolean	<p>isScrollable()</p> <p>获取该节点是否可滚动。</p>
boolean	<p>isSelected()</p> <p>获取该节点是否已被选择。</p>
boolean	<p>isVisibleToUser()</p> <p>获取该节点是否对用户可见。</p>
static AccessibilityNodeInfo	<p>obtain(View root, int virtualDescendantId)</p> <p>如果可获得，返回一个缓存实例，否则，返回一个新的并设置源。</p>
static AccessibilityNodeInfo	<p>obtain(AccessibilityNodeInfo info)</p> <p>如果可获得，返回一个缓存实例，或创建一个新的。</p>

static AccessibilityNodeInfo	<p>obtain()</p> <p>如果可获得，返回一个缓存实例，否则，返回一个新的。</p>
static AccessibilityNodeInfo	<p>obtain(View source)</p> <p>如果可获得，返回一个缓存实例，否则，返回一个新的并设置源。</p>
boolean	<p>performAction(int action, Bundle arguments)</p> <p>在该节点上执行一个操作。</p>
boolean	<p>performAction(int action)</p> <p>在该节点上执行一个操作。</p>
void	<p>recycle()</p> <p>返回一个实例重复使用。</p>
boolean	<p>refresh()</p> <p>刷新视图呈现的最新状态信息。</p>
boolean	<p>removeAction(AccessibilityNodeInfo.AccessibilityAction action)</p> <p>移除一个可以在该节点上执行的操作。</p>
void	<p>removeAction(int action)</p> <p>该方法在 API 级别 21 被弃用，使用 removeAction(AccessibilityAction)代替。</p>
boolean	<p>removeChild(View root, int virtualDescendantId)</p>

	移除给定 <code>root</code> 的一个虚拟子元素。
boolean	<code>removeChild(View child)</code> 移除一个子元素。
void	<code>setAccessibilityFocused(boolean focused)</code> 设置该节点可无障碍聚焦。
void	<code>setBoundsInParent(Rect bounds)</code> 在父级坐标中设置节点边界。
void	<code>setBoundsInScreen(Rect bounds)</code> 在屏幕坐标中设置节点边界。
void	<code>setCanOpenPopup(boolean opensPopup)</code> 设置该节点可打开一个弹窗或对话框。
void	<code>setCheckable(boolean checkable)</code> 设置该节点可选中。
void	<code>setChecked(boolean checked)</code> 设置该节点已选中。
void	<code>setClassName(CharSequence className)</code> 设置该节点来自的类。
void	<code>setClickable(boolean clickable)</code> 设置该节点可点击。
void	<code>setCollectionInfo(AccessibilityNodeInfo.CollectionInfo collectionInfo)</code>

	如果该节点是个集合，设置集合信息。
void	<p><code>setCollectionItemInfo(AccessibilityNodeInfo.CollectionItemInfo collectionItemInfo)</code></p> <p>如果该节点是个集合项目，设置该集合项目的信息。</p>
void	<p><code>setContentDescription(CharSequence contentDescription)</code></p> <p>设置该节点的内容描述。</p>
void	<p><code>setContentInvalid(boolean contentInvalid)</code></p> <p>设置节点内容无效。</p>
void	<p><code>setContextClickable(boolean contextClickable)</code></p> <p>设置该节点文本可点击。</p>
void	<p><code>setDismissable(boolean dismissable)</code></p> <p>设置该节点可以被关闭。</p>
void	<p><code>setDrawingOrder(int drawingOrderInParent)</code></p> <p>设置该节点相关视图的绘制顺序。</p>
void	<p><code>setEditable(boolean editable)</code></p> <p>设置该节点可编辑。</p>
void	<p><code>setEnabled(boolean enabled)</code></p> <p>设置该节点可用。</p>
void	<code>setError(CharSequence error)</code>

	设置该节点的错误文本。
void	<code>setFocusable(boolean focusable)</code> 设置该节点可聚焦。
void	<code>setFocused(boolean focused)</code> 设置该节点已聚焦。
void	<code>setImportantForAccessibility(boolean important)</code> 设置该节点对无障碍很重要。
void	<code>setInputType(int inputType)</code> 设置源中的输入类型为 <code>InputType</code> 。
void	<code>setLabelFor(View root, int virtualDescendantId)</code> 为无障碍目的, 设置可作为标签代表视图的信息。
void	<code>setLabelFor(View labeled)</code> 为无障碍目的, 设置可作为标签代表视图的信息。
void	<code>setLabeledBy(View label)</code> 为无障碍目的, 设置可作为标签代表视图的信息。
void	<code>setLabeledBy(View root, int virtualDescendantId)</code> 为无障碍目的, 设置可作为标签代表视图的信息。

	信息。
void	<code>setLiveRegion(int mode)</code> 设置该节点的实时区域模式。
void	<code>setLongClickable(boolean longClickable)</code> 设置该节点可长按点击。
void	<code>setMaxTextLength(int max)</code> 设置最大文本长度，或无限制时为-1。
void	<code>setMovementGranularities(int granularities)</code> 设置遍历节点文本的移动粒度。
void	<code>setMultiLine(boolean multiLine)</code> 设置该节点为一个多行可编辑文本。
void	<code>setPackageName(CharSequence packageName)</code> 设置该节点来自的包。
void	<code>setParent(View root, int virtualDescendantId)</code> 设置给定 root 虚拟后代的父元素。
void	<code>setParent(View parent)</code> 设置父元素。
void	<code>setPassword(boolean password)</code> 设置该节点为密码。

void	<p><code>setRangeInfo(AccessibilityNodeInfo.RangeInfo rangeInfo)</code></p> <p>如果该节点是个范围，设置范围信息。</p>
void	<p><code>setScrollable(boolean scrollable)</code></p> <p>设置该节点可滚动。</p>
void	<p><code>setSelected(boolean selected)</code></p> <p>设置该节点是否已选择。</p>
void	<p><code>setSource(View source)</code></p> <p>设置源。</p>
void	<p><code>setSource(View root, int virtualDescendantId)</code></p> <p>为给定 root 虚拟后代设置源。</p>
void	<p><code>setText(CharSequence text)</code></p> <p>设置该节点的文本。</p>
void	<p><code>setTextSelection(int start, int end)</code></p> <p>设置文本选择的起始和结尾。</p>
void	<p><code>setTraversalAfter(View view)</code></p> <p>设置无障碍遍历中当前访问节点的下一个视图。</p>
void	<p><code>setTraversalAfter(View root, int virtualDescendantId)</code></p> <p>设置无障碍遍历中当前访问节点的下一个</p>

	视图。
void	<code>setTraversalBefore(View view)</code> 设置无障碍遍历中当前访问节点的上一个视图。
void	<code>setTraversalBefore(View root, int virtualDescendantId)</code> 设置无障碍遍历中当前访问节点的上一个视图。
void	<code>setViewIdResourceName(String viewIdResName)</code> 设置源视图 id 的完全合格源名称。
void	<code>setVisibleToUser(boolean visibleToUser)</code> 设置该节点对用户可见。
String	<code>toString()</code> 返回一个代表对象的字符串。
void	<code>writeToParcel(Parcel parcel, int flags)</code> 封装该对象到一个包中。

2.6 继承方法

继承自类 `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	<p>wait(long millis)</p> <p>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。</p>
final void	<p>wait()</p> <p>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。</p>

继承自接口 [android.os.Parcelable](#);

abstract int	<p>describeContents()</p> <p>描述被包含在可打包实例的封装代表中特殊对象的种类。</p>
abstract int	<p>writeToParcel(Parcel dest, int flags)</p> <p>将对象写入到一个包。</p>

3. 常量

3.1 ACTION_ACCESSIBILITY_FOCUS

添加于 [API 级别 16](#)。

`int ACTION_ACCESSIBILITY_FOCUS`

给节点添加无障碍焦点的操作。

常量值: 64 (0x00000040)

3.2 ACTION_ARGUMENT_COLUMN_INT

添加于 [API 级别 23](#)。

`String ACTION_ARGUMENT_COLUMN_INT`

让指定集合列在屏幕上可见的参数。

类型: `int`

操作:

- [ACTION_SCROLL_TO_POSITION](#)

参见:

[ACTION_SCROLL_TO_POSITION](#)

常量值: `"android.view.accessibility.action.ARGUMENT_COLUMN_INT"`

3.3 ACTION_ARGUMENT_EXTEND_SELECTION_BOOLEAN

添加于 [API 级别 18](#)。

[String](#) ACTION_ARGUMENT_EXTEND_SELECTION_BOOLEAN

当以一定粒度移动时，是否扩展选项或反之移除的参数。

类型：boolean

操作：

- [ACTION_NEXT_AT_MOVEMENT_GRANULARITY](#)
- [ACTION_PREVIOUS_AT_MOVEMENT_GRANULARITY](#)

参见：

[ACTION_NEXT_AT_MOVEMENT_GRANULARITY](#)

[ACTION_PREVIOUS_AT_MOVEMENT_GRANULARITY](#)

常量值："ACTION_ARGUMENT_EXTEND_SELECTION_BOOLEAN"

3.4 ACTION_ARGUMENT_HTML_ELEMENT_STRING

添加于 [API 级别 16](#)。

[String](#) ACTION_ARGUMENT_HTML_ELEMENT_STRING

要移动到的上一个/下一个 HTML 元素的参数。

类型：String

操作：

- [ACTION_NEXT_HTML_ELEMENT](#)
- [ACTION_PREVIOUS_HTML_ELEMENT](#)

参见：

[ACTION_NEXT_HTML_ELEMENT](#)

[ACTION_PREVIOUS_HTML_ELEMENT](#)

常量值: "ACTION_ARGUMENT_HTML_ELEMENT_STRING"

3.5 ACTION_ARGUMENT_MOVEMENT_GRANULARITY_INT

添加于 [API 级别 16](#)。

[String](#) ACTION_ARGUMENT_MOVEMENT_GRANULARITY_INT

当遍历节点文本时，使用哪种移动粒度的参数。

类型: int

操作:

- [ACTION_NEXT_AT_MOVEMENT_GRANULARITY](#)
- [ACTION_PREVIOUS_AT_MOVEMENT_GRANULARITY](#)

参见:

[ACTION_NEXT_AT_MOVEMENT_GRANULARITY](#)

[ACTION_PREVIOUS_AT_MOVEMENT_GRANULARITY](#)

常量值: "ACTION_ARGUMENT_MOVEMENT_GRANULARITY_INT"

3.7 ACTION_ARGUMENT_PROGRESS_VALUE

添加于 [API 级别 24](#)。

[String](#) ACTION_ARGUMENT_PROGRESS_VALUE

指定要设置的进度值的参数。

类型: float

操作:

[ACTION_SET_PROGRESS](#)

参见:

[ACTION_SET_PROGRESS](#)

常量值: "android.view.accessibility.action.ARGUMENT_PROGRESS_VALUE"

3.8 ACTION_ARGUMENT_ROW_INT

添加于 [API 级别 23](#)。

[String](#) ACTION_ARGUMENT_ROW_INT

指定集合行在屏幕上可见的参数。

类型: int

操作:

- [ACTION_SCROLL_TO_POSITION](#)

参见:

[ACTION_SCROLL_TO_POSITION](#)

常量值: "android.view.accessibility.action.ARGUMENT_ROW_INT"

3.9 ACTION_ARGUMENT_SELECTION_END_INT

添加于 [API 级别 18](#)。

[String](#) ACTION_ARGUMENT_SELECTION_END_INT

指定选择结束的参数。

类型：int

操作：

- [ACTION_SET_SELECTION](#)

参见：

[ACTION_SET_SELECTION](#)

常量值："ACTION_ARGUMENT_SELECTION_END_INT"

3.10 ACTION_ARGUMENT_SELECTION_START_INT

添加于 [API 级别 18](#)。

[String](#) ACTION_ARGUMENT_SELECTION_START_INT

指定选择起始的参数。

类型：int

操作：

- [ACTION_SET_SELECTION](#)

参见：

[ACTION_SET_SELECTION](#)

常量值："ACTION_ARGUMENT_SELECTION_START_INT"

3.11 ACTION_ARGUMENT_SET_TEXT_CHARACTERSEQUENCE

添加于 [API 级别 21](#)。

[String](#) ACTION_ARGUMENT_SET_TEXT_CHARSEQUENCE

指定要设置的文本内容的参数。

类型: CharSequence

操作:

- [ACTION_SET_TEXT](#)

参见:

[ACTION_SET_TEXT](#)

常量值: "ACTION_ARGUMENT_SET_TEXT_CHARSEQUENCE"

3.12 ACTION_CLEAR_ACCESSIBILITY_FOCUS

添加于 [API 级别 16](#)。

int ACTION_CLEAR_ACCESSIBILITY_FOCUS

清除节点无障碍焦点的操作。

常量值: 128 (0x00000080)

3.13 ACTION_CLEAR_FOCUS

添加于 [API 级别 14](#)。

int ACTION_CLEAR_FOCUS

清除节点输入焦点的操作。

常量值: 2 (0x00000002)

3.14 ACTION_CLEAR_SELECTION

添加于 [API 级别 14](#)。

int ACTION_CLEAR_SELECTION

取消选择节点的操作。

常量值: 8 (0x00000008)

3.15 ACTION_CLICK

添加于 [API 级别 16](#)。

int ACTION_CLICK

在节点信息上点击的操作。参见 [ACTION_CLICK](#)。

常量值: 16 (0x00000010)

3.16 ACTION_COLLAPSE

添加于 [API 级别 19](#)。

int ACTION_COLLAPSE

折叠一个可展开节点的操作。

常量值: 524288 (0x00080000)

3.17 ACTION_COPY

添加于 [API 级别 18](#)。

int ACTION_COPY

将当前选择拷贝到剪贴板的操作。

常量值：16384 (0x00004000)

3.18 ACTION_CUT

添加于 [API 级别 18](#)。

int ACTION_CUT

剪贴当前选项并放置到剪贴板的操作。

常量值：65536 (0x00010000)

3.19 ACTION_DISMISS

添加于 [API 级别 19](#)。

int ACTION_DISMISS

关闭一个可关闭节点的操作。

常量值：1048576 (0x00100000)

3.20 ACTION_EXPAND

添加于 [API 级别 19](#)。

int ACTION_EXPAND

展开一个可展开节点的操作。

常量值：262144 (0x00040000)

3.21 ACTION_FOCUS

添加于 [API 级别 14](#)。

`int ACTION_FOCUS`

给节点添加输入焦点的操作。

常量值: 1 (0x00000001)

3.22 ACTION_LONG_CLICK

添加于 [API 级别 16](#)。

`int ACTION_LONG_CLICK`

在节点上点击长按的操作。

常量值: 32 (0x00000020)

3.23 ACTION_NEXT_AT_MOVEMENT_GRANULARITY

添加于 [API 级别 16](#)。

`int ACTION_NEXT_AT_MOVEMENT_GRANULARITY`

以给定移动粒度，请求去到该节点文本的下一个文本实体的操作。例如，移动到下一个字、词等。

参数:

[ACTION_ARGUMENT_MOVEMENT_GRANULARITY_INT](#) <int>, [ACTION_ARGUMENT_EXTEND_SELECTION_BOOLEAN](#) <boolean>

样例: 移动到上一个字符，且不扩大选择。

```
Bundle arguments = new Bundle();  
arguments.putInt(AccessibilityNodeInfo.ACTION_ARGUMENT_MOVEMENT_
```



```
GRANULARITY_INT,AccessibilityNodeInfo.MOVEMENT_GRANULARITY_CHARACTER);  
  
arguments.putBoolean(AccessibilityNodeInfo.ACTION_ARGUMENT_EXTEND_SELECTION_BOOLEAN,false);  
  
info.performAction(AccessibilityNodeInfo.ACTION_NEXT_AT_MOVEMENT_GRANULARITY, arguments);
```

参见：

[ACTION_ARGUMENT_MOVEMENT_GRANULARITY_INT](#)

[ACTION_ARGUMENT_EXTEND_SELECTION_BOOLEAN](#)

[setMovementGranularities\(int\)](#)

[getMovementGranularities\(\)](#)

[MOVEMENT_GRANULARITY_CHARACTER](#)

[MOVEMENT_GRANULARITY_WORD](#)

[MOVEMENT_GRANULARITY_LINE](#)

[MOVEMENT_GRANULARITY_PARAGRAPH](#)

[MOVEMENT_GRANULARITY_PAGE](#)

常量值： 256 (0x00000100)

3.24 ACTION_NEXT_HTML_ELEMENT

添加于 [API 级别 16](#)。

`int ACTION_NEXT_HTML_ELEMENT`

移动到给定类型的下一个 HTML 元素的操作。例如，移动到 `BUTTON`、`INPUT`、`TABLE` 等。

参数：[ACTION_ARGUMENT_HTML_ELEMENT_STRING](#)

样例：

```
Bundle arguments = new Bundle();  
  
arguments.putString(AccessibilityNodeInfo.ACTION_ARGUMENT_HTML_ELEMENT_STRING, "BUTTON");  
  
info.performAction(AccessibilityNodeInfo.ACTION_NEXT_HTML_ELEMENT, arguments);
```

常量值：1024 (0x00000400)

3.25 ACTION_PASTE

添加于 [API 级别 18](#)。

int ACTION_PASTE

粘贴当前剪贴板内容的操作。

常量值：32768 (0x00008000)

3.26 ACTION_PREVIOUS_AT_MOVEMENT_GRANULARITY

添加于 [API 级别 16](#)。

int ACTION_PREVIOUS_AT_MOVEMENT_GRANULARITY

以给定移动粒度，请求去到该节点文本的上一个文本实体的操作。例如，移动到下一个字、词等。

参数：[ACTION_ARGUMENT_MOVEMENT_GRANULARITY_INT](#)<,
[ACTION_ARGUMENT_EXTEND_SELECTION_BOOLEAN](#)

样例：移动到下一个字，且不扩大选择。

```
Bundle arguments = new Bundle();

arguments.putInt(AccessibilityNodeInfo.ACTION_ARGUMENT_MOVEMENT_
GRANULARITY_INT,

AccessibilityNodeInfo.MOVEMENT_GRANULARITY_CHARACTER);

arguments.putBoolean(AccessibilityNodeInfo.ACTION_ARGUMENT_EXTEND_
SELECTION_BOOLEAN, false);

info.performAction(AccessibilityNodeInfo.ACTION_PREVIOUS_AT_MOVEME
NT_GRANULARITY,arguments);
```

参见：

[ACTION_ARGUMENT_MOVEMENT_GRANULARITY_INT](#)

[ACTION_ARGUMENT_EXTEND_SELECTION_BOOLEAN](#)

[setMovementGranularities\(int\)](#)

[getMovementGranularities\(\)](#)

[MOVEMENT_GRANULARITY_CHARACTER](#)

[MOVEMENT_GRANULARITY_WORD](#)

[MOVEMENT_GRANULARITY_LINE](#)

[MOVEMENT_GRANULARITY_PARAGRAPH](#)

[MOVEMENT_GRANULARITY_PAGE](#)

常量值： 512 (0x00000200)

3.27 ACTION_PREVIOUS_HTML_ELEMENT

添加于 [API 级别 16](#)。

int ACTION_PREVIOUS_HTML_ELEMENT

移动到给定类型的上一个 HTML 元素的操作。例如，移动到 BUTTON、INPUT、TABLE 等。

参数: ACTION_ARGUMENT_HTML_ELEMENT_STRING

样例:

```
Bundle arguments = new Bundle();  
  
arguments.putString(AccessibilityNodeInfo.ACTION_ARGUMENT_HTML_ELEMENT_STRING, "BUTTON");  
  
info.performAction(AccessibilityNodeInfo.ACTION_PREVIOUS_HTML_ELEMENT, arguments);
```

常量值: 2048 (0x00000800)

3.28 ACTION_SCROLL_BACKWARD

添加于 API 级别 16。

int ACTION_SCROLL_BACKWARD

向后滚动节点内容的操作。

常量值: 8192 (0x00002000)

3.29 ACTION_SCROLL_FORWARD

添加于 API 级别 16。

int ACTION_SCROLL_FORWARD

向前滚动节点内容的操作。

常量值: 4096 (0x00001000)

3.30 ACTION_SELECT

添加于 [API 级别 14](#)。

int ACTION_SELECT

选择节点的操作。

常量值: 4 (0x00000004)

3.31 ACTION_SET_SELECTION

添加于 [API 级别 18](#)。

int ACTION_SET_SELECTION

设置选择项的操作。执行该操作，并且无参数清除选项。

参数: [ACTION_ARGUMENT_SELECTION_START_INT](#),
[ACTION_ARGUMENT_SELECTION_END_INT](#)

样例:

```
Bundle arguments = new Bundle();

arguments.putInt(AccessibilityNodeInfo.ACTION_ARGUMENT_SELECTION_S
TART_INT, 1);
arguments.putInt(AccessibilityNodeInfo.ACTION_ARGUMENT_SELECTION_E
ND_INT, 2);

info.performAction(AccessibilityNodeInfo.ACTION_SET_SELECTION,
arguments);
```

参见:

[ACTION_ARGUMENT_SELECTION_START_INT](#)

[ACTION_ARGUMENT_SELECTION_END_INT](#)

常量值: 131072 (0x00020000)

3.32 ACTION_SET_TEXT

添加于 [API 级别 21](#)。

int ACTION_SET_TEXT

设置节点文本的操作。在没有参数的情况下执行该操作，使用 `null` 或者空 `CharSequence` 将会清除文本。该操作也将会把光标放置到文本末尾。

参数: [ACTION_ARGUMENT_SET_TEXT_CHARSEQUENCE](#)

样例:

```
Bundle arguments = new Bundle();

arguments.putCharSequence(AccessibilityNodeInfo.ACTION_ARGUMENT_SET
_TEXT_CHARSEQUENCE, "android");

info.performAction(AccessibilityNodeInfo.ACTION_SET_TEXT, arguments);
```

常量值: 2097152 (0x00200000)

3.33 FOCUS_ACCESSIBILITY

添加于 [API 级别 16](#)。

int FOCUS_ACCESSIBILITY

无障碍焦点。

常量值: 2 (0x00000002)

3.34 FOCUS_INPUT

添加于 [API 级别 16](#)。

int FOCUS_INPUT

输入焦点。

常量值: 1 (0x00000001)

3.35 MOVEMENT_GRANULARITY_CHARACTER

添加于 [API 级别 16](#)。

int MOVEMENT_GRANULARITY_CHARACTER

以字符为移动粒度位，遍历节点文本。

常量值: 1 (0x00000001)

3.36 MOVEMENT_GRANULARITY_LINE

添加于 [API 级别 16](#)。

int MOVEMENT_GRANULARITY_LINE

以行为移动粒度位，遍历节点文本。

常量值: 4 (0x00000004)

3.37 MOVEMENT_GRANULARITY_PAGE

添加于 [API 级别 16](#)。

int MOVEMENT_GRANULARITY_PAGE

以页为移动粒度位，遍历节点文本。

常量值: 16 (0x00000010)

3.38 MOVEMENT_GRANULARITY_PARAGRAPH

添加于 [API 级别 16](#)。

int MOVEMENT_GRANULARITY_PARAGRAPH

以段为移动粒度位，遍历节点文本。

常量值: 8 (0x00000008)

3.39 MOVEMENT_GRANULARITY_WORD

添加于 [API 级别 16](#)。

int MOVEMENT_GRANULARITY_WORD

以字词为移动粒度位，遍历节点文本。

常量值: 2 (0x00000002)

4. 字段

[Creator<AccessibilityNodeInfo>](#) CREATOR

参见：

[Parcelable.Creator](#)

5.公有方法

5.1addAction

添加于 [API 级别 14](#)。

`void addAction (int action)`

注：该方法在 API 级别 21 的时候被弃用。该方法被 [addAction\(\[AccessibilityAction\]\(#\)\)](#)代替。

添加一个可以在节点上执行的操作。

注意：不能从一个 [AccessibilityService](#) 调用。在被发送到无障碍服务之前，该类是不可改变的。

参数：

<code>action</code>	<code>int</code> ：操作。
---------------------	-----------------------

抛出：

IllegalStateException	如果从一个无障碍服务调用。
IllegalArgumentException	如果该参数不是个标准操作。

5.2addAction

添加于 [API 级别 21](#)。

`void addAction (AccessibilityNodeInfo.AccessibilityAction action)`

添加一个可以在节点上执行的操作。

使用 [AccessibilityNodeInfo.AccessibilityAction](#) 的静态常量添加一个标准操作。

添加一个自定义操作，通过从应用中传递一个源 id 作为操作 id 和描述操作的可选标签，创建一个新的 `AccessibilityNodeInfo.AccessibilityAction`。重写一个标准操作，作为标准操作 id，例如 `ACTION_CLICK` 和描述操作的可选标签。

注意：不能从一个 `AccessibilityService` 调用。在被发送到无障碍服务之前，该类是不可改变的。

参数：

Action	<code>AccessibilityNodeInfo.AccessibilityAction</code> : 操作。
--------	--

抛出：

<code>IllegalStateException</code>	如果从一个无障碍服务调用。
------------------------------------	---------------

5.3addChild

添加于 [API 级别 16](#)。

`void addChild (View root, int virtualDescendantId)`

添加一个虚拟子元素，作为给定 `root` 的后代。如果 `virtualDescendantId` 是 `NO_ID`，该 `root` 作为一个子元素被添加。

一个虚拟后代是一个虚构视图，为无障碍目的，将自己报告为视图层次的一部分。该操作可以使绘制复杂内容的自定义视图，将自己报告为虚拟视图树，然后传递其逻辑结构。

参数：

root	<code>View</code> : 虚拟子树的 root。
virtualDescendantId	<code>int</code> : 虚拟子元素的 id。

5.4addChild

添加于 [API 级别 14](#)。

void addChild ([View](#) child)

添加一个子元素。

注意：不能从一个 [AccessibilityService](#) 调用。在被发送到无障碍服务之前，该类是不可改变的。

参数：

child	View ：子元素。
-------	----------------------------

抛出：

IllegalStateException	如果从一个无障碍服务调用。
---------------------------------------	---------------

5.5canOpenPopup

添加于 [API 级别 19](#)。

boolean canOpenPopup ()

获取该节点是否可以打开一个弹窗或对话框。

返回值：

boolean	该节点是否打开了一个弹窗。
---------	---------------

5.6describeContents

添加于 [API 级别 14](#)。

`int describeContents ()`

描述被包含在可打包实例的封装代表中的特殊对象的种类。例如，如果该对象在 `writeToParcel(Parcel, int)` 的输出中包含一个文件描述符，该方法的返回值必须包含 `CONTENTS_FILE_DESCRIPTOR` 位。

参数：

<code>int</code>	一个位掩码，标识被该 <code>Parcelable</code> 对象实例封装的指定对象类型的设置。
------------------	--

5.7 equals

添加于 [API 级别 14](#)。

`boolean equals (Object object)`

标识某些其他对象是否“等同于”该对象。

该 `equals` 方法在非空对象索引上实现等价关系：

- **自反**：对于任何非空索引值 `x`，`x.equals(x)` 应该返回 `true`。
- **对称**：对于任何非空索引值 `x` 和 `y`，当且仅当 `y.equals(x)` 返回 `true` 时，`x.equals(y)` 应该返回 `true`。
- **传递**：对于任何非空索引值 `x`、`y` 和 `z`，如果 `x.equals(y)` 返回 `true` 且 `y.equals(z)` 返回 `true`，则 `x.equals(z)` 应该返回 `true`。
- **一致**：对于任何非空索引值 `x` 和 `y`，`x.equals(y)` 的多次调用一致返回 `true`，或一致返回 `false`，在已修改对象的 `equals` 比较中，未提供任何信息。
- 对于任何非空索引值 `x`，`x.equals(null)` 应该返回 `false`。

`Object` 类的 `equals` 方法在对象上实现了最可能区分的等价关系；因此，对于任何非空索引值 `x` 和 `y`，当且只有当 `x` 和 `y` 指向同一对象的时候，该方法返回 `true` (`x==y` 具有相同的值)，返回 `true`。

注意，一般来说，无论该方法何时被重写，必须重写 `hashCode` 方法，保持

hashCode 方法的总合约，这说明等价对象必须有等价哈希编码。

参数：

object	Object: 要比较的索引对象。
--------	-------------------

返回值：

boolean	如果该对象与 obj 参数相同，返回 true，否则，返回 false。
---------	--------------------------------------

5.8 findAccessibilityNodeInfosByText

添加于 [API 级别 14](#)。

[List<AccessibilityNodeInfo>](#) findAccessibilityNodeInfosByText ([String](#) text)

使用文本找到 [AccessibilityNodeInfo](#)。与之匹配的是不区分大小写的情况。该检索与该信息相关联，例如该信息是遍历树的根。

注意：通过调用 [recycle\(\)](#)回收接收到的信息，是客户端的责任，可以避免创建多个实例。

参数：

Text	String: 检索文本。
------	---------------

返回值：

List<AccessibilityNodeInfo>	节点信息列表。
---	---------

5.9 findAccessibilityNodeInfosByViewId

添加于 [API 级别 18](#)。

[List<AccessibilityNodeInfo>](#) findAccessibilityNodeInfosByViewId([String](#) viewId)

使用完全合格视图 id 的源名字找到 [AccessibilityNodeInfo](#)，完全合格 id 的样式为“package:id/id_resource_name”。例如，如果目标应用的包是“foo.bar”并且 id 源名字是“baz”，则完全合格源 id 为“foo.bar:id/baz”。

注意：通过调用 [recycle\(\)](#) 回收接收到的信息，是客户端的责任，可以避免创建多个实例。

注意：该 API 的主要用途是 UI 测试自动化，和当配置自己的 [AccessibilityService](#)，客户端不得不设置 [FLAG_REPORT_VIEW_IDS](#)，来报告一个 [AccessibilityNodeInfo](#) 的源视图 id。

参数：

<code>viewId</code>	String: 要寻找的视图 id 的完全合格源名字。
---------------------	-----------------------------

返回值：

<code>List<AccessibilityNodeInfo></code>	节点信息列表。
--	---------

5.10 findFocus

添加于 [API 级别 16](#)。

[AccessibilityNodeInfo](#) findFocus (int focus)

找到具有指定焦点类型的视图。检索起始于该节点信息代表的视图。

参数：

<code>focus</code>	int: 要寻找的焦点。 FOCUS_INPUT 或 FOCUS_ACCESSIBILITY 中的一个。
--------------------	--

返回值：

AccessibilityNodeInfo	已聚焦视图的节点信息，或为空。
---------------------------------------	-----------------

参见：

[FOCUS_INPUT](#)

[FOCUS_ACCESSIBILITY](#)

5.11 focusSearch

添加于 [API 级别 16](#)。

[AccessibilityNodeInfo](#) focusSearch (int direction)

在指定方向搜索具有输入焦点的最近的视图。

参数：

direction	int：方向。可以是以下其中一个： FOCUS_DOWN 、 FOCUS_UP 、 FOCUS_LEFT 、 FOCUS_RIGHT 、 FOCUS_FORWARD 、 FOCUS_BACKWARD 。
-----------	---

返回值：

AccessibilityNodeInfo	具有无障碍焦点的视图节点信息。
---------------------------------------	-----------------

5.12 getActionList

添加于 [API 级别 21](#)。

[List<AccessibilityNodeInfo.AccessibilityAction>](#) getActionList ()

获取可以在节点上执行的操作。

返回值：

List<AccessibilityNodeInfo.AccessibilityAction>	
---	--

5.13getActions

添加于 [API 级别 14](#)。

`int getActions ()`

注意：该方法在 [API 级别 21](#) 被弃用。使用 [getActionList \(\)](#)代替。

获取可以在节点上执行的操作。

返回值：

<code>int</code>	操作的位掩码。
------------------	---------

参见：

[ACTION_FOCUS](#)

[ACTION_CLEAR_FOCUS](#)

[ACTION_SELECT](#)

[ACTION_CLEAR_SELECTION](#)

[ACTION_ACCESSIBILITY_FOCUS](#)

[ACTION_CLEAR_ACCESSIBILITY_FOCUS](#)

[ACTION_CLICK](#)

[ACTION_LONG_CLICK](#)

[ACTION_NEXT_AT_MOVEMENT_GRANULARITY](#)

[ACTION_PREVIOUS_AT_MOVEMENT_GRANULARITY](#)

[ACTION_NEXT_HTML_ELEMENT](#)

[ACTION_PREVIOUS_HTML_ELEMENT](#)

[ACTION_SCROLL_FORWARD](#)

[ACTION_SCROLL_BACKWARD](#)

5.14getBoundsInParent

添加于 [API 级别 14](#)。

`void getBoundsInParent (Rect outBounds)`

获取父级坐标中的节点边界。

参数：

outBounds	Rect：输出节点边界。
-----------	--------------

5.15getBoundsInScreen

添加于 [API 级别 14](#)。

`void getBoundsInScreen (Rect outBounds)`

获取屏幕坐标中的节点边界。

参数：

outBounds	Rect：输出节点边界。
-----------	--------------

5.16getChild

添加于 [API 级别 14](#)。

`AccessibilityNodeInfo getChild (int index)`

获取给定索引下的子元素。

注意：通过调用 `recycle()`回收接收到的信息，是客户端的责任，可以避免创建多个实例。

参数：

<code>index</code>	<code>int</code> : 子元素索引。
--------------------	---------------------------

返回值:

<code>AccessibilityNodeInfo</code>	子元素节点。
------------------------------------	--------

抛出:

<code>IllegalStateException</code>	如果从一个无障碍服务调用。
------------------------------------	---------------

5.17 getChildCount

添加于 [API 级别 14](#)。

`int getChildCount ()`

获取子元素的数量。

返回值:

<code>int</code>	子元素数量。
------------------	--------

5.18 getClassName

添加于 [API 级别 14](#)。

`CharSequence getClassName ()`

获取该节点来自的类。

返回值:

<code>CharSequence</code>	类名。
---------------------------	-----

5.19 getCollectionInfo

添加于 [API 级别 19](#)。

[AccessibilityNodeInfo.CollectionInfo](#) getCollectionInfo ()

如果节点是个集合，获取集合信息。一个集合子集总是一个集合项目。

返回值：

AccessibilityNodeInfo.CollectionInfo	集合信息。
--	-------

5.20 getCollectionItemInfo

添加于 [API 级别 19](#)。

[AccessibilityNodeInfo.CollectionItemInfo](#) getCollectionItemInfo ()

如果节点是个集合项目，获取集合项目信息。一个集合项目总是一个集合的子集。

返回值：

AccessibilityNodeInfo.CollectionItemInfo	集合项目信息。
--	---------

5.21 getContentDescription

添加于 [API 级别 14](#)。

[CharSequence](#) getContentDescription ()

获取该节点的内容描述。

返回值：

CharSequence	内容描述。
--------------	-------

5.22getDrawingOrder

添加于 [API 级别 24](#)。

获取该节点中视图的相对绘制顺序。

绘制顺序只决定于节点的父级，所以该索引只与其兄弟姐妹相关。

在一些情景下，绘制顺序是基本同步的，所以两个兄弟姐妹的返回值相同是可能的。返回值会被跳过也是可能的。

返回值：

int	该节点对应的视图，相对于其兄弟界面的绘制位置。
-----	-------------------------

5.23getError

添加于 [API 级别 21](#)。

[CharSequence](#) `getError ()`

获取该节点的错误文本。

返回值：

CharSequence	错误文本。
------------------------------	-------

5.24getExtras

添加于 [API 级别 19](#)。

[Bundle](#) `getExtras ()`

获取具有附加数据的可选 **bundle**。该包是闲时创建的且永不为 **null**。

注意：为了避免冲突，推荐使用应用的包名作为关键字字首，如果从不同应用发出的相同关键词有不同的含义，可能会导致无障碍服务困扰。

返回值：

Bundle	包。
---------------	----

5.25getInputType

添加于 [API 级别 19](#)。

int getInputType ()

获取定义为 **InputType** 的源中的输入类型。

返回值：

int	输入类型。
------------	-------

5.26getLabelFor

添加于 [API 级别 17](#)。

AccessibilityNodeInfo getLabelFor ()

为无障碍目的，获取作为标签代表该视图的节点信息。

注意：通过调用 **recycle()**回收接收到的信息，是客户端的责任，可以避免创建多个实例。

返回值：

AccessibilityNodeInfo	标签信息。
------------------------------	-------

5.27 getLabeledBy

添加于 [API 级别 17](#)。

[AccessibilityNodeInfo](#) getLabeledBy ()

为无障碍目的，获取作为标签代表该视图的节点信息。

注意：通过调用 [recycle\(\)](#)回收接收到的信息，是客户端的责任，可以避免创建多个实例。

返回值：

AccessibilityNodeInfo	标签。
---------------------------------------	-----

5.28 getLiveRegion

添加于 [API 级别 19](#)。

int getLiveRegion ()

获取该节点的实时区域模式。

一个实时区域是一个包含对用户来说重要信息的节点，且当其改变时应该告知用户。例如，在一个登录界面，有一个呈现“密码错误”通知的 `TextView`，该视图应该使用 [ACCESSIBILITY_LIVE_REGION_POLITE](#) 模式被标记为一个实时区域。

这是无障碍服务的责任，控制 [TYPE_WINDOW_CONTENT_CHANGED](#) 事件标识实时区域节点和其子元素的变化。

返回值：

int	实时区域模式，或如果视图不是个实时区域，返回 ACCESSIBILITY_LIVE_REGION_NONE 。
-----	---

参见：

[getAccessibilityLiveRegion](#)

5.29 getMaxTextLength

添加于 [API 级别 21](#)。

`int getMaxTextLength ()`

返回该节点的最大文本长度。

返回值：

<code>int</code>	最大文本长度，无限制返回-1。
------------------	-----------------

参见：

[setMaxTextLength\(int\)](#)

5.30 getMovementGranularities

添加于 [API 级别 16](#)。

`int getMovementGranularities ()`

获取遍历该节点文本的移动粒度。

返回值：

<code>int</code>	粒度的位掩码。
------------------	---------

5.31 getPackageName

添加于 [API 级别 14](#)。

[CharSequence](#) `getPackageName ()`

获取该节点来自的包。

返回值：

CharSequence	包名。
------------------------------	-----

5.32getParent

添加于 [API 级别 14](#)。

[AccessibilityNodeInfo](#) getParent ()

获取父级。

注意：通过调用 [recycle\(\)](#)回收接收到的信息，是客户端的责任，可以避免创建多个实例。

返回值：

AccessibilityNodeInfo	父级。
---------------------------------------	-----

5.33getRangeInfo

添加于 [API 级别 19](#)。

[AccessibilityNodeInfo.RangeInfo](#) getRangeInfo ()

如果节点是个范围，获取范围信息。

返回值：

AccessibilityNodeInfo.RangeInfo	父级。
---	-----

5.34getText

添加于 [API 级别 14](#)。

[CharSequence](#) `getText ()`

获取该节点的文本。

返回值：

CharSequence	文本。
------------------------------	-----

5.35 `getTextSelectionEnd`

添加于 [API 级别 18](#)。

`int` `getTextSelectionEnd ()`

如果文本已被选择，获取文本选择结尾。

如果没有文本被选择，该方法和 [getTextSelectionStart\(\)](#) 返回相同的值：光标的当前位置。

返回值：

<code>int</code>	选择文本的末尾，如果没有选择文本，返回光标位置，或如果不存在选择文本和光标，返回-1。
------------------	---

5.36 `getTextSelectionStart`

添加于 [API 级别 18](#)。

`int` `getTextSelectionStart ()`

获取文本选择起始或光标位置。

如果没有文本被选择，该方法和 [getTextSelectionEnd \(\)](#) 返回相同值：光标的当前位置。

返回值:

int	选择文本的起始, 如果没有选择文本, 返回光标位置, 或如果不存在选择文本和光标, 返回-1。
-----	---

5.37getTraversalAfter

添加于 [API 级别 22](#)。

[AccessibilityNodeInfo](#) getTraversalAfter ()

当前节点被访问后, 获取无障碍遍历中的下一个节点。一个屏幕阅读器必须在访问该节点内容之前, 访问其他节点的内容。

返回值:

AccessibilityNodeInfo	如果存在, 返回后续节点, 否则, 返回 null。
---------------------------------------	----------------------------

参见:

[setTraversalAfter\(android.view.View\)](#)

[setTraversalAfter\(android.view.View, int\)](#)

5.38getTraversalBefore

添加于 [API 级别 22](#)。

[AccessibilityNodeInfo](#) getTraversalBefore ()

当前节点被访问后, 获取无障碍遍历中的上一个节点。屏幕阅读器必须访问该节点的信息, 在访问先前节点内容之前。

返回值:

AccessibilityNodeInfo	如果存在, 返回后续节点, 否则, 返回 null。
---------------------------------------	----------------------------

参见:

[setTraversalBefore\(android.view.View\)](#)

[setTraversalBefore\(android.view.View, int\)](#)

5.39 getViewIdResourceName

添加于 [API 级别 18](#)。

[String](#) getViewIdResourceName ()

获取源视图 id 的完全合格源名称。

注意：该 API 的主要用途是 UI 测试自动化，和当配置自己的 [AccessibilityService](#)，客户端不得不设置 [FLAG_REPORT_VIEW_IDS](#)，来报告一个 [AccessibilityNodeInfo](#) 的源视图 id。

String	id 源名称。
------------------------	---------

5.40 getWindow

添加于 [API 级别 21](#)。

[AccessibilityWindowInfo](#) getWindow ()

获取该节点所属的窗口。

返回值:

AccessibilityWindowInfo	窗口。
---	-----

参见:

[getWindows\(\)](#)

5.41 getWindowId

添加于 [API 级别 14](#)。

`int getWindowId ()`

获取该信息来自的窗口的 id。

返回值：

<code>int</code>	窗口 id。
------------------	--------

5.42 hashCode

添加于 [API 级别 14](#)。

`int hashCode ()`

返回该对象的哈希编码值。该方法支持哈希表格的优点，例如 [HashMap](#) 提供的用途。

hashCode 的一般效用：

- 在一个 JAVA 应用执行过程中，无论何时在同一对象上被激活超过一次，`hashCode` 方法必须一致性地返回相同整数，该整数在 `equals` 比较中不提供任何关于被修改对象的信息。从一个应用的一次执行到相同应用的另一次执行，该整数不需要保持一致。
- 如果根据 `equals(Object)` 方法，两个对象相同，然后在两个对象上分别调用 `hashCode` 方法，一定会生成相同的整数结果。
- 如果依据 `equals(java.lang.Object)` 方法，两个对象不相同，不需要在两个对象上分别调用 `hashCode` 方法生成不同的整数结果。但是，开发者应该清楚知道，为不同对象生成不同整数结果会提升哈希表格的表现。

作为一个合理的实践，被 `Object` 对象定义的 `hashCode` 方法，为不同对象返回不同的整数。（这通常通过将转换对象外部地址为一个整体实现，但是该实现

技术不是 Java™ 编程语言必须的。)

返回值:

int	该对象的哈希代码值。
-----	------------

5.43isAccessibilityFocused

添加于 [API 级别 16](#)。

boolean isAccessibilityFocused ()

获取该节点是否可被无障碍聚焦。

返回值:

boolean	如果该节点可被无障碍聚焦，返回 true。
---------	-----------------------

5.44isCheckable

添加于 [API 级别 14](#)。

boolean isCheckable ()

获取该节点是否可选中。

返回值:

boolean	如果该节点可选中，返回 true。
---------	-------------------

5.45isChecked

添加于 [API 级别 14](#)。

boolean isChecked ()

获取该节点是否已选中。

返回值：

boolean	如果该节点已选中，返回 true。
---------	-------------------

5.46isClickable

添加于 [API 级别 14](#)。

boolean isClickable ()

获取该节点是否可点击。

返回值：

boolean	如果该节点可点击，返回 true。
---------	-------------------

5.47isContentInvalid

添加于 [API 级别 19](#)。

boolean isContentInvalid ()

获取该节点的内容是否已失效。例如，一个日期数据格式不正确。

返回值：

boolean	如果该节点文本无效。
---------	------------

5.48isContextClickable

添加于 [API 级别 23](#)。

`boolean isContextClickable ()`

获取该节点是否上下文可点击。

返回值：

<code>boolean</code>	如果该节点上下文可点击，返回 <code>true</code> 。
----------------------	------------------------------------

5.49isDismissable

添加于 [API 级别 19](#)。

`boolean isDismissable ()`

获取该节点是否可被关闭。

返回值：

<code>boolean</code>	如果该节点可被关闭。
----------------------	------------

5.50isEditable

添加于 [API 级别 18](#)。

`boolean isEditable ()`

获取该节点是否可编辑。

返回值：

<code>boolean</code>	如果该节点可编辑，返回 <code>true</code> ，否则返回 <code>false</code> 。
----------------------	--

5.51isEnabled

添加于 [API 级别 14](#)。

`boolean isEnabled ()`

获取该节点是否可用。

返回值：

<code>boolean</code>	如果该节点可用，返回 <code>true</code> 。
----------------------	--------------------------------

5.52isFocusable

添加于 [API 级别 14](#)。

`boolean isFocusable ()`

获取该节点是否可聚焦。

返回值：

<code>boolean</code>	如果该节点可聚焦，返回 <code>true</code> 。
----------------------	---------------------------------

5.53isFocused

添加于 [API 级别 14](#)。

`boolean isFocused ()`

获取该节点是否已被聚焦。

返回值：

<code>boolean</code>	如果该节点已被聚焦，返回 <code>true</code> 。
----------------------	----------------------------------

5.53isImportantForAccessibility

添加于 [API 级别 24](#)。

`boolean isImportantForAccessibility ()`

返回该节点是否来源于一个对无障碍相当重要的视图。

返回值：

<code>boolean</code>	如果该节点起源于一个对无障碍很重要的视图，返回 <code>true</code> ，否则，返回 <code>false</code> 。
----------------------	---

参见：

[isImportantForAccessibility\(\)](#)

5.54isLongClickable

添加于 [API 级别 14](#)。

`boolean isLongClickable ()`

获取该节点是否可长按点击。

返回值：

<code>boolean</code>	如果该节点可长按点击，返回 <code>true</code> 。
----------------------	-----------------------------------

5.55isMultiLine

添加于 [API 级别 19](#)。

`boolean isMultiLine ()`

获取该节点是否是个多行可编辑文本。

返回值：

boolean	如果该节点是个多行文本，返回 true。
---------	----------------------

5.56isPassword

添加于 [API 级别 14](#)。

boolean isPassword ()

获取该节点是否是个密码。

返回值：

boolean	如果该节点是个密码，返回 true。
---------	--------------------

5.57isScrollable

添加于 [API 级别 14](#)。

boolean isScrollable ()

获取该节点是否可滚动。

返回值：

boolean	如果该节点可滚动，返回 true，否则，返回 false。
---------	-------------------------------

5.58isSelected

添加于 [API 级别 14](#)。

boolean isSelected ()

获取该节点是否已被选择。

返回值:

boolean	如果该节点已被选择, 返回 true。
---------	---------------------

5.59 isVisibleToUser

添加于 [API 级别 16](#)。

boolean isVisibleToUser ()

获取该节点是否对用户可见。

返回值:

Boolean	该节点是否对用户可见。
---------	-------------

5.60 obtain

添加于 [API 级别 16](#)。

[AccessibilityNodeInfo](#) obtain ([View](#) root, int virtualDescendantId)

如果可获得, 返回一个缓存实例, 否则创建一个新的并设置源。

参数:

root	View : 虚拟子树的根。
virtualDescendantId	int: 虚拟后代的 id。

返回值:

AccessibilityNodeInfo	一个实例。
---------------------------------------	-------

参见:

`setSource(View, int)`

5.61obtain

添加于 [API 级别 14](#)。

[AccessibilityNodeInfo](#) obtain ([AccessibilityNodeInfo](#) info)

如果可获得，返回一个缓存实例，或创建一个新的。返回的实例初始化自一个给定 root。

参数：

info	AccessibilityNodeInfo ：其他信息。
------	--

返回值：

AccessibilityNodeInfo	一个实例。
---------------------------------------	-------

5.62obtain

添加于 [API 级别 14](#)。

[AccessibilityNodeInfo](#) obtain ()

如果可获得，返回一个缓存实例，否则，返回一个新的。

返回值：

AccessibilityNodeInfo	一个实例。
---------------------------------------	-------

5.63obtain

添加于 [API 级别 14](#)。

[AccessibilityNodeInfo](#) obtain ([View](#) source)

如果可获得，返回一个缓存实例，否则，返回一个新的并设置源。

参数：

source	View ：源视图。
--------	-------------------

返回值：

AccessibilityNodeInfo	一个实例。
---------------------------------------	-------

参见：

[setSource\(View\)](#)

5.64performAction

添加于 [API 级别 16](#)。

boolean performAction (int action, [Bundle](#) arguments)

在该节点上执行一个操作。

注意： 只有从一个 [AccessibilityService](#) 请求时，操作才会被执行。

参数：

action	int ：要执行的操作。
arguments	Bundle ：具有额外参数的包。

返回值：

boolean	如果操作被执行，返回 true 。
---------	--------------------------

抛出：

<code>IllegalStateException</code>	如果在 <code>AccessibilityService</code> 外被调用。
------------------------------------	---

5.65performAction

添加于 [API 级别 14](#)。

`boolean performAction (int action)`

在该节点上执行一个操作。

注意：只有从一个 [AccessibilityService](#) 请求时，操作才会被执行。

参数：

<code>action</code>	<code>int</code> : 要执行的操作。
<code>arguments</code>	<code>Bundle</code> : 具有额外参数的包。

返回值：

<code>boolean</code>	如果操作被执行，返回 <code>true</code> 。
----------------------	--------------------------------

抛出：

<code>IllegalStateException</code>	如果在 <code>AccessibilityService</code> 外被调用。
------------------------------------	---

5.66recycle

添加于 [API 级别 14](#)。

`void recycle ()`

返回一个实例重复使用。

注意：在调用该功能之后，不能再触摸该对象。

抛出：

<code>IllegalStateException</code>	如果该信息已被回收。
------------------------------------	------------

5.67refresh

添加于 [API 级别 18](#)。

`boolean refresh ()`

刷新视图呈现的最新状态信息。

注意：如果该方法返回 `false`，该信息是过时的，因为它呈现的视图不存在于视图树中，且应该被回收。

返回值：

<code>boolean</code>	刷新是否成功。
----------------------	---------

5.68removeAction

添加于 [API 级别 21](#)。

`boolean removeAction (AccessibilityNodeInfo.AccessibilityAction action)`

移除一个可以在该节点上执行的操作。如果该操作未被添加到该节点，调用该方法没有任何效果。

注意：不能从一个 [AccessibilityService](#) 调用。该类在传递给一个 `AccessibilityService` 之前是不可变更的。

参数：

<code>action</code>	<code>AccessibilityNodeInfo.AccessibilityAction</code> ：要移除的操作。
---------------------	---

返回值：

boolean	从操作列表中被移除的操作。
---------	---------------

抛出：

IllegalStateException	如果从一个无障碍服务调用。
-----------------------	---------------

5.69removeAction

添加于 [API 级别 21](#)。

void removeAction (int action)

注意：该方法在 API 级别 21 被弃用。使用 [removeAction\(AccessibilityAction\)](#) 代替。

移除一个可以在该节点上执行的操作。如果该操作未被添加到该节点，调用该方法没有任何效果。

注意：不能从一个 [AccessibilityService](#) 调用。该类在传递给一个 AccessibilityService 之前是不可变更的。

参数：

action	int：要移除的操作。
--------	-------------

抛出：

IllegalStateException	如果从一个无障碍服务调用。
-----------------------	---------------

5.70removeChild

添加于 [API 级别 21](#)。

boolean removeChild (View root, int virtualDescendantId)

移除给定 `root` 的一个虚拟子元素。如果子元素不是之前被添加到该节点的，调用该方法无效果。

参数：

<code>root</code>	View: 虚拟子树的 <code>root</code> 。
<code>virtualDescendantId</code>	int: 虚拟子元素的 <code>id</code> 。

返回值：

<code>boolean</code>	如果子元素存在，返回 <code>true</code> 。
----------------------	--------------------------------

参见：

[addChild\(View, int\)](#)

5.71removeChild

添加于 [API 级别 21](#)。

`boolean removeChild (View child)`

移除一个子元素。如果子元素先前未被添加到该节点，调用该方法无效果。

注意： 不能从一个 [AccessibilityService](#) 调用。该类在传递给一个 `AccessibilityService` 之前是不可变更的。

参数：

<code>child</code>	View: 子元素。
--------------------	-------------------

返回值：

<code>boolean</code>	如果子元素存在，为 <code>true</code> 。
----------------------	-------------------------------

抛出：

<code>IllegalStateException</code>	如果从一个无障碍服务调用。
------------------------------------	---------------

5.72 setAccessibilityFocused

添加于 [API 级别 16](#)。

`void setAccessibilityFocused (boolean focused)`

设置该节点可无障碍聚焦。

注意： 不能从一个 [AccessibilityService](#) 调用。在被传递到一个 [AccessibilityService](#) 之前，该类不可改变。

参数：

<code>focused</code>	<code>boolean</code> ：如果该节点被无障碍聚焦，为 <code>true</code> 。
----------------------	---

抛出：

<code>IllegalStateException</code>	如果从一个无障碍服务调用。
------------------------------------	---------------

5.73 setBoundsInParent

添加于 [API 级别 14](#)。

`void setBoundsInParent (Rect bounds)`

在父级坐标中设置节点边界。

注意： 不能从一个 [AccessibilityService](#) 调用。在被传递到一个 [AccessibilityService](#) 之前，该类不可改变。

参数：

<code>bounds</code>	<code>Rect</code> ：节点边界。
---------------------	--------------------------

抛出:

IllegalStateException	如果从一个无障碍服务调用。
---------------------------------------	---------------

5.74 setBoundsInScreen

添加于 [API 级别 14](#)。

`void setBoundsInScreen (Rect bounds)`

在屏幕坐标中设置节点边界。

注意：不能从一个 [AccessibilityService](#) 调用。在被传递到一个 [AccessibilityService](#) 之前，该类不可改变。

参数:

<code>bounds</code>	Rect : 节点边界。
---------------------	------------------------------

抛出:

IllegalStateException	如果从一个无障碍服务调用。
---------------------------------------	---------------

5.75 setCanOpenPopup

添加于 [API 级别 19](#)。

`void setCanOpenPopup (boolean opensPopup)`

设置该节点可打开一个弹窗或对话框。

注意：不能从一个 [AccessibilityService](#) 调用。在被传递到一个 [AccessibilityService](#) 之前，该类不可改变。

参数:

<code>opensPopup</code>	boolean: 如果该节点打开一个弹窗。
-------------------------	------------------------------

5.76 setCheckable

添加于 [API 级别 14](#)。

`void setCheckable (boolean checkable)`

设置该节点可选中。

注意： 不能从一个 [AccessibilityService](#) 调用。在被传递到一个 [AccessibilityService](#) 之前，该类不可改变。

参数：

<code>checkable</code>	boolean: 如果该节点可选中，为 <code>true</code> 。
------------------------	--

抛出：

IllegalStateException	如果从一个无障碍服务调用。
---------------------------------------	---------------

5.77 setChecked

添加于 [API 级别 14](#)。

`void setChecked (boolean checked)`

设置该节点已选中。

注意： 不能从一个 [AccessibilityService](#) 调用。在被传递到一个 [AccessibilityService](#) 之前，该类不可改变。

参数：

<code>checked</code>	boolean: 如果该节点被选中，为 <code>true</code> 。
----------------------	--

抛出：

IllegalStateException	如果从一个无障碍服务调用。
---------------------------------------	---------------

5.78 setClassName

添加于 [API 级别 14](#)。

void setClassName ([CharSequence](#) className)

设置该节点来自的类。

注意： 不能从一个 [AccessibilityService](#) 调用。在被传递到一个 [AccessibilityService](#) 之前，该类不可改变。

参数：

className	CharSequence ：类名。
-----------	-----------------------------------

抛出：

IllegalStateException	如果从一个无障碍服务调用。
---------------------------------------	---------------

5.79 setClickable

添加于 [API 级别 14](#)。

void setClickable (boolean clickable)

设置该节点可点击。

注意： 不能从一个 [AccessibilityService](#) 调用。在被传递到一个 [AccessibilityService](#) 之前，该类不可改变。

参数：

clickable	boolean: 如果该节点是可点击, 为 true。
-----------	-----------------------------

抛出:

IllegalStateException	如果从一个无障碍服务调用。
-----------------------	---------------

5.80 setCollectionInfo

添加于 [API 级别 19](#)。

void setCollectionInfo ([AccessibilityNodeInfo.CollectionInfo](#) collectionInfo)

如果该节点是个集合, 设置集合信息。一个集合子集通常也是个集合项目。

注意: 不能从一个 [AccessibilityService](#) 调用。在被传递到一个 AccessibilityService 之前, 该类不可改变。

参数:

collectionInfo	AccessibilityNodeInfo.CollectionInfo : 集合信息。
----------------	--

5.81 setCollectionItemInfo

添加于 [API 级别 19](#)。

void setCollectionItemInfo ([AccessibilityNodeInfo.CollectionItemInfo](#) collectionItemInfo)

如果该节点是个集合项目, 设置集合项目的信息。一个集合项目总是一个集合的子集。

注意: 不能从一个 [AccessibilityService](#) 调用。在被传递到一个 AccessibilityService 之前, 该类不可改变。

参数:

collectionItemInfo	AccessibilityNodeInfo.CollectionItemInfo
--------------------	--

5.82 setContentDescription

添加于 [API 级别 14](#)。

void setContentDescription ([CharSequence](#) contentDescription)

设置该节点的内容描述。

注意： 不能从一个 [AccessibilityService](#) 调用。在被传递到一个 AccessibilityService 之前，该类不可改变。

参数：

contentDescription	CharSequence ：内容描述。
--------------------	-------------------------------------

抛出：

IllegalStateException	如果从一个无障碍服务调用。
---------------------------------------	---------------

5.83 setContentInvalid

添加于 [API 级别 19](#)。

void setContentInvalid (boolean contentInvalid)

设置节点内容无效。例如，一个日期数据格式不正确。

注意： 不能从一个 [AccessibilityService](#) 调用。在被传递到一个 AccessibilityService 之前，该类不可改变。

参数：

contentInvalid	boolean：如果该节点内容无效。
----------------	--------------------

5.84 setContextClickable

添加于 [API 级别 23](#)。

`void setContextClickable (boolean contextClickable)`

设置该节点文本可点击。

注意： 不能从一个 [AccessibilityService](#) 调用。在被传递到一个 [AccessibilityService](#) 之前，该类不可改变。

参数：

<code>contextClickable</code>	boolean： 如果该节点文本可点击，为 <code>true</code> 。
-------------------------------	--

抛出：

IllegalStateException	如果从一个无障碍服务调用。
---------------------------------------	---------------

5.85 setDismissable

添加于 [API 级别 19](#)。

`void setDismissable (boolean dismissable)`

设置该节点可以被关闭。

注意： 不能从一个 [AccessibilityService](#) 调用。在被传递到一个 [AccessibilityService](#) 之前，该类不可改变。

参数：

<code>dismissable</code>	boolean： 如果该节点可以被关闭。
--------------------------	-----------------------------

5.86 setDrawingOrder

添加于 [API 级别 24](#)。

```
void setDrawingOrder (int drawingOrderInParent)
```

设置该节点相关视图的绘制顺序。

注意：不能从一个 [AccessibilityService](#) 调用。在被传递到一个 [AccessibilityService](#) 之前，该类不可改变。

抛出：

IllegalStateException	如果从一个无障碍服务调用。
---------------------------------------	---------------

5.87setEditable

添加于 [API 级别 18](#)。

```
void setEditable (boolean editable)
```

设置该节点可编辑。

注意：不能从一个 [AccessibilityService](#) 调用。在被传递到一个 [AccessibilityService](#) 之前，该类不可改变。

参数：

editable	boolean: 如果该节点可编辑，为 true。
--------------------------	----------------------------------

抛出：

IllegalStateException	如果从一个无障碍服务调用。
---------------------------------------	---------------

5.88setEnabled

添加于 [API 级别 14](#)。

`void setEnabled (boolean enabled)`

设置该节点可用。

注意： 不能从一个 [AccessibilityService](#) 调用。在被传递到一个 [AccessibilityService](#) 之前，该类不可改变。

参数：

<code>enabled</code>	<code>boolean</code> ：如果该节点可用，为 <code>true</code> 。
----------------------	---

抛出：

IllegalStateException	如果从一个无障碍服务调用。
---------------------------------------	---------------

5.89setError

添加于 [API 级别 21](#)。

`void setError (CharSequence error)`

设置该节点的错误文本。

注意： 不能从一个 [AccessibilityService](#) 调用。在被传递到一个 [AccessibilityService](#) 之前，该类不可改变。

参数：

<code>error</code>	<code>CharSequence</code> ：错误文本。
--------------------	----------------------------------

抛出：

IllegalStateException	如果从一个无障碍服务调用。
---------------------------------------	---------------

5.90setFocusable

添加于 [API 级别 14](#)。

```
void setFocusable (boolean focusable)
```

设置该节点可聚焦。

注意：不能从一个 [AccessibilityService](#) 调用。在被传递到一个 [AccessibilityService](#) 之前，该类不可改变。

参数：

<code>focusable</code>	<code>boolean</code> ：如果该节点可聚焦，为 <code>true</code> 。
------------------------	--

抛出：

IllegalStateException	如果从一个无障碍服务调用。
---------------------------------------	---------------

5.91 setFocused

添加于 [API 级别 14](#)。

```
void setFocused (boolean focused)
```

设置该节点已聚焦。

注意：不能从一个 [AccessibilityService](#) 调用。在被传递到一个 [AccessibilityService](#) 之前，该类不可改变。

参数：

<code>focused</code>	<code>boolean</code> ：如果该节点被聚焦，为 <code>true</code> 。
----------------------	--

抛出：

IllegalStateException	如果从一个无障碍服务调用。
---------------------------------------	---------------

5.92setImportantForAccessibility

添加于 [API 级别 24](#)。

`void setImportantForAccessibility (boolean important)`

设置该节点对无障碍很重要。

注意： 不能从一个 [AccessibilityService](#) 调用。在被传递到一个 [AccessibilityService](#) 之前，该类不可改变。

参数：

<code>important</code>	<code>boolean</code> :如果该节点对无障碍很重要，为 <code>true</code> ，否则，为 <code>false</code> 。
------------------------	---

5.93setInputType

添加于 [API 级别 19](#)。

`void setInputType (int inputType)`

设置源中的输入类型为 [InputType](#)。

注意： 不能从一个 [AccessibilityService](#) 调用。在被传递到一个 [AccessibilityService](#) 之前，该类不可改变。

参数：

<code>inputType</code>	<code>int</code> : 输入类型。
------------------------	--------------------------

抛出：

IllegalStateException	如果从一个无障碍服务调用。
---------------------------------------	---------------

5.94setLabelFor

添加于 [API 级别 17](#)。

```
void setLabelFor (View root, int virtualDescendantId)
```

为无障碍目的，设置可作为标签代表视图的信息。如果 `virtualDescendantId` 为 `NO_ID`，根被设置为标签。

一个虚拟后代是一个虚构视图，为无障碍目的，将自己报告为视图层次的一部分。该操作可以使绘制复杂内容的自定义视图，将自己报告为虚拟视图树，然后传递其逻辑结构。

注意： 不能从一个 [AccessibilityService](#) 调用。在被传递到一个 `AccessibilityService` 之前，该类不可改变。

参数：

<code>root</code>	View: 虚拟后代作为标签的根视图。
<code>virtualDescendantId</code>	int: 虚拟后代的 id。

5.95setLabelFor

添加于 [API 级别 17](#)。

```
void setLabelFor (View labeled)
```

为无障碍目的，设置可作为标签代表视图的信息。

参数：

<code>labeled</code>	View: 该信息会被作为视图的标签。
----------------------	----------------------------

5.96setLabeledBy

添加于 [API 级别 17](#)。

`void setLabeledBy (View label)`

为无障碍目的，设置可作为标签代表视图的信息。

参数：

label	View：标记该节点的源视图。
-------	-----------------

5.97setLabeledBy

添加于 [API 级别 17](#)。

`void setLabeledBy (View root, int virtualDescendantId)`

为无障碍目的，设置可作为标签代表视图的信息。如果 `virtualDescendantId` 为 [NO_ID](#)，根被设置为标签。

一个虚拟后代是一个虚构视图，为无障碍目的，将自己报告为视图层次的一部分。该操作可以使绘制复杂内容的自定义视图，将自己报告为虚拟视图树，然后传递其逻辑结构。

注意： 不能从一个 [AccessibilityService](#) 调用。在被传递到一个 [AccessibilityService](#) 之前，该类不可改变。

参数：

root	View：虚拟后代作为标签的根视图。
virtualDescendantId	int：虚拟后代的 id。

5.98setLiveRegion

添加于 [API 级别 19](#)。

`void setLiveRegion (int mode)`

设置该节点的实时区域模式。

注意： 不能从一个 [AccessibilityService](#) 调用。在被传递到一个 AccessibilityService 之前，该类不可改变。

参数：

<code>mode</code>	<code>int</code> ：实时区域模式，或如果视图不是个实时区域时，为 ACCESSIBILITY_LIVE_REGION_NONE 。
-------------------	---

参见：

[setAccessibilityLiveRegion\(int\)](#)

5.99setLongClickable

添加 [API 级别 14](#)。

`void setLongClickable (boolean longClickable)`

设置该节点可长按点击。

注意： 不能从一个 [AccessibilityService](#) 调用。在被传递到一个 AccessibilityService 之前，该类不可改变。

参数：

<code>longClickable</code>	<code>boolean</code> ：如果该节点可长按点击，为 <code>true</code> 。
----------------------------	--

抛出：

IllegalStateException	如果从一个无障碍服务调用。
---------------------------------------	---------------

5.100setMaxTextLength

添加于 [API 级别 21](#)。


```
void setMaxTextLength (int max)
```

设置最大文本长度，或无限制时为-1。

一般情况下，用来标识一个可编辑文本框有输入字符数目限制。

注意：不能从一个 [AccessibilityService](#) 调用。在被传递到一个 [AccessibilityService](#) 之前，该类不可改变。

参数：

<code>max</code>	<code>int</code> ：最大文本长度。
------------------	---------------------------

抛出：

IllegalStateException	如果从一个无障碍服务调用。
---------------------------------------	---------------

参见：

[getMaxTextLength\(\)](#)

5.101 setMovementGranularities

添加于 [API 级别 16](#)。

```
int getMovementGranularities ()
```

设置遍历节点文本的移动粒度。

注意：不能从一个 [AccessibilityService](#) 调用。在被传递到一个 [AccessibilityService](#) 之前，该类不可改变。

参数：

<code>granularities</code>	<code>int</code> ：移动粒度的位掩码。
----------------------------	-----------------------------

抛出：

<code>IllegalStateException</code>	如果从一个无障碍服务调用。
------------------------------------	---------------

5.102 setMultiLine

添加于 [API 级别 19](#)。

`void setMultiLine (boolean multiLine)`

设置该节点为一个多行可编辑文本。

注意： 不能从一个 [AccessibilityService](#) 调用。在被传递到一个 [AccessibilityService](#) 之前，该类不可改变。

参数：

<code>multiline</code>	<code>boolean</code> ：如果该节点是个多行文本，返回 <code>true</code> 。
------------------------	--

5.103 setPackageName

添加于 [API 级别 14](#)。

`void setPackageName (CharSequence packageName)`

设置该节点来自的包。

注意： 不能从一个 [AccessibilityService](#) 调用。在被传递到一个 [AccessibilityService](#) 之前，该类不可改变。

参数：

<code>packageName</code>	<code>CharSequence</code> ：包名。
--------------------------	--------------------------------

抛出：

<code>IllegalStateException</code>	如果从一个无障碍服务调用。
------------------------------------	---------------

5.104setParent

添加于 [API 级别 16](#)。

`void setParent (View root, int virtualDescendantId)`

设置给定 root 虚拟后代的父元素。如果 virtualDescendantId 等于 NO_ID，该 root 被设置为父级。

一个虚拟后代是一个虚构视图，为无障碍目的，将自己报告为视图层次的一部分。该操作可以使绘制复杂内容的自定义视图，将自己报告为虚拟视图树，然后传递其逻辑结构。

注意： 不能从一个 [AccessibilityService](#) 调用。在被传递到一个 AccessibilityService 之前，该类不可改变。

参数：

root	View: 虚拟子树的 root。
virtualDescendantId	int: 虚拟后代的 id。

5.105setParent

添加于 [API 级别 14](#)。

`void setParent (View parent)`

设置父元素。

注意： 不能从一个 [AccessibilityService](#) 调用。在被传递到一个 AccessibilityService 之前，该类不可改变。

参数：

parent	View: 父级。
--------	-----------

抛出：

IllegalStateException	如果从一个无障碍服务调用。
---------------------------------------	---------------

5.106 setPassword

添加于 [API 级别 14](#)。

`void setPassword (boolean password)`

设置该节点为密码。

注意： 不能从一个 [AccessibilityService](#) 调用。在被传递到一个 [AccessibilityService](#) 之前，该类不可改变。

参数：

<code>password</code>	<code>boolean</code> ：如果该节点是个密码，为 <code>true</code> 。
-----------------------	---

抛出：

IllegalStateException	如果从一个无障碍服务调用。
---------------------------------------	---------------

5.107 setRangeInfo

添加于 [API 级别 19](#)。

`void setRangeInfo (AccessibilityNodeInfo.RangeInfo rangeInfo)`

如果该节点是个范围，设置范围信息。

注意： 不能从一个 [AccessibilityService](#) 调用。在被传递到一个 [AccessibilityService](#) 之前，该类不可改变。

参数：

rangeInfo	AccessibilityNodeInfo.RangeInfo: 范围信息。
-----------	--

5.108 setScrollable

添加于 [API 级别 14](#)。

void setScrollable (boolean scrollable)

设置该节点可滚动。

注意： 不能从一个 [AccessibilityService](#) 调用。在被传递到一个 AccessibilityService 之前，该类不可改变。

参数：

scrollable	boolean: 如果该节点是个密码，为 true。
------------	----------------------------

抛出：

IllegalStateException	如果从一个无障碍服务调用。
---------------------------------------	---------------

5.109 setSelected

添加于 [API 级别 14](#)。

void setSelected (boolean selected)

设置该节点是否已选择。

注意： 不能从一个 [AccessibilityService](#) 调用。在被传递到一个 AccessibilityService 之前，该类不可改变。

参数：

selected	boolean: 如果该节点已被选定，为 true。
----------	----------------------------

抛出：

[IllegalStateException](#)

如果从一个无障碍服务调用。

5.110 setSource

添加于 [API 级别 14](#)。

`void setSource (View source)`

设置源。

注意：不能从一个 [AccessibilityService](#) 调用。在被传递到一个 [AccessibilityService](#) 之前，该类不可改变。

参数：

`source`

View：信息源。

5.111 setSource

添加于 [API 级别 16](#)。

`void setSource (View root, int virtualDescendantId)`

为给定 root 虚拟后代设置源。如果 virtualDescendantId 等于 [NO_ID](#)，该 root 被设置为源。

一个虚拟后代是一个虚构视图，为无障碍目的，将自己报告为视图层次的一部分。该操作可以使绘制复杂内容的自定义视图，将自己报告为虚拟视图树，然后传递其逻辑结构。

注意：不能从一个 [AccessibilityService](#) 调用。在被传递到一个 [AccessibilityService](#) 之前，该类不可改变。

参数：

root	View: 虚拟子树的 root。
virtualDescendantId	int: 虚拟后代的 id。

5.112 setText

添加于 [API 级别 14](#)。

void setText ([CharSequence](#) text)

设置该节点的文本。

注意： 不能从一个 [AccessibilityService](#) 调用。在被传递到一个 AccessibilityService 之前，该类不可改变。

参数：

text	CharSequence : 文本。
------	------------------------------------

抛出：

IllegalStateException	如果从一个无障碍服务调用。
---------------------------------------	---------------

5.113 setTextSelection

添加于 [API 级别 18](#)。

void setTextSelection (int start, int end)

设置文本选择的起始和结尾。

注意： 不能从一个 [AccessibilityService](#) 调用。在被传递到一个 AccessibilityService 之前，该类不可改变。

参数：

start	int: 文本选择起始。
end	int: 文本选择结尾。

抛出:

IllegalStateException	如果从一个无障碍服务调用。
-----------------------	---------------

5.114 setTraversalAfter

添加于 [API 级别 22](#)。

void setTraversalAfter ([View](#) view)

设置无障碍遍历中下一个被访问的视图。一个屏幕阅读器在访问该节点内容之前，必须访问另一个节点的内容。

注意： 不能从一个 [AccessibilityService](#) 调用。在被传递到一个 AccessibilityService 之前，该类不可改变。

参数:

view	View : 前一个视图。
------	-------------------------------

参见:

[getTraversalAfter\(\)](#)

5.115 setTraversalAfter

添加于 [API 级别 22](#)。

void setTraversalAfter ([View](#) root, int virtualDescendantId)

设置无障碍遍历中下一个被访问的视图。一个屏幕阅读器在访问该节点内容

之前，必须访问另一个节点的内容。如果 `virtualDescendantId` 等于 `NO_ID`，`root` 被设置为前任。

一个虚拟后代是一个虚构视图，为无障碍目的，将自己报告为视图层次的一部分。该操作可以使绘制复杂内容的自定义视图，将自己报告为虚拟视图树，然后传递其逻辑结构。

注意： 不能从一个 `AccessibilityService` 调用。在被传递到一个 `AccessibilityService` 之前，该类不可改变。

参数：

<code>root</code>	View: 虚拟子树的 <code>root</code> 。
<code>virtualDescendantId</code>	Int: 虚拟后代的 <code>id</code> 。

5.116 setTraversalBefore

添加于 [API 级别 22](#)。

`void setTraversalBefore (View view)`

设置无障碍遍历中当前访问节点的上一个视图。一个屏幕阅读器在访问上一个节点之前，必须访问该节点的内容。

注意： 不能从一个 `AccessibilityService` 调用。在被传递到一个 `AccessibilityService` 之前，该类不可改变。

参数：

<code>view</code>	view: 提供先前节点的视图。
-------------------	-------------------------

参见：

[getTraversalBefore\(\)](#)

5.117setTraversalBefore

添加于 [API 级别 22](#)。

`void setTraversalBefore (View root, int virtualDescendantId)`

设置无障碍遍历中当前访问节点的上一个视图。一个屏幕阅读器在访问上一个节点之前，必须访问该节点的内容。如果 `virtualDescendantId` 等于 `NO_ID`，`root` 被设置为后任。

一个虚拟后代是一个虚构视图，为无障碍目的，将自己报告为视图层次的一部分。该操作可以使绘制复杂内容的自定义视图，将自己报告为虚拟视图树，然后传递其逻辑结构。

注意： 不能从一个 [AccessibilityService](#) 调用。在被传递到一个 [AccessibilityService](#) 之前，该类不可改变。

参数：

<code>root</code>	View: 虚拟子树的 <code>root</code> 。
<code>virtualDescendantId</code>	Int: 虚拟后代的 <code>id</code> 。

5.118setViewIdResourceName

添加于 [API 级别 18](#)。

`void setViewIdResourceName (String viewIdResName)`

设置源视图 `id` 的完全合格源名称。

注意： 不能从一个 [AccessibilityService](#) 调用。在被传递到一个 [AccessibilityService](#) 之前，该类不可改变。

参数：

<code>viewIdResName</code>	String: <code>id</code> 源名称。
----------------------------	-------------------------------------

5.119setVisibleToUser

添加于 [API 级别 16](#)。

`void setVisibleToUser (boolean visibleToUser)`

设置该节点对用户可见。

注意： 不能从一个 [AccessibilityService](#) 调用。在被传递到一个 [AccessibilityService](#) 之前，该类不可改变。

参数：

<code>visibleToUser</code>	boolean： 该节点是否对用户可见。
----------------------------	-----------------------------

抛出：

IllegalStateException	如果从一个无障碍服务调用。
---------------------------------------	---------------

5.120toString

添加于 [API 级别 14](#)。

[String](#) `toString ()`

返回一个代表对象的字符串。一般来说，`toString` 方法返回一个字符串，以文本代表该对象。结果应该是个精简却翔实的代表，且容易被用户阅读。推荐所有子类重写该方法。

`Object` 类的 `toString` 方法返回一个包含实例对象类名的字符串，包含字符“@”和对象哈希码的十六进制代表。换句话说，该方法返回的字符串等于下值：

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

返回值：

String	对象的字符串代表。
------------------------	-----------

5.121writeToParcel

添加于 [API 级别 14](#)。

void writeToParcel ([Parcel](#) parcel, int flags)

封装该对象到一个包中。

注意：在将该实例写入到一个包后，该实例可被再使用。调用该功能之后，一定不能触摸该对象。

参数：

parcel	parcel：该对象应该被写入的包。
flags	int：该对象应该被怎样写入的额外标识。可能是 0 或是 PARCELABLE_WRITE_RETURN_VALUE 。