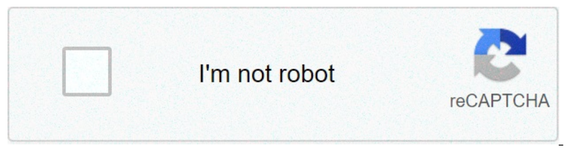

Continue

Android send key event programmatically

There was a blog on the key event programming generation, but I was not satisfied with the result. The instrumentation picture is beautiful, but it's a bit heavy. If you want to generate only keys, the reason why to have all those entries in the manifesto, instrumentation object, etc.? So I used the wonderful new Dedexer tool to go after the internal framework instrumentation.The critical method operation in Android.App.Instrumentation is this: Public .Method SendKeysync (Landroid / View / KeyEvent;) V.Catch Android / OS / RemoteException from LE5CF6 to LE5D12 Using LE5D14 Invoke-Direct {V2}, Android / App / Instrumentation / ValidateNotappThread; ValidateNotappThread () VLE5CF6: Const String V0, "Window" Invoke-Static {V0}, Android / OS / ServiceManager / GetService; GetService (Ljava / Lang / String;) Landroid / OS / Ibinder; Move-Result-object V0 Invoke-Static {V0}, Android / View / iWindowManager $ stub / as-interface; AS-Interface (Landroid / OS / IBinder;) Landroid / View / iWindowManager; Move-Result-object V0 CONST / 4 V1.1 INVOKE-INTERFACE {V0, V3, V1}, Android / View / iWindowManager / InjectKeyEvent; INJECTKEYEVENT (LANDROID / VIEW / KEYEVENT; Z) ZLE5D12: Return Voidle5D14: Move-V0 Advancement Exception LE5D12.END Methodhis is actually a public API method. The method gets the WINDOWMANAGERSERVICE binder from the ServiceManager then draws the InjectKeyEvent method on the WindowmanagerService public interface. This is something we can do to ourselves, can't we? Click here to download the example Program.unfortunately the situation is not so simple. Since non-public branches like Android.os.ServiceManager, Android.View.IwindowManager, etc. have been removed from Android.jar, applications that use non-public APIs cannot be filled easily as before. If you select the sample program, you will see that I created stub for these classes. These matrices are only to compile the program, they do not get packaged in the APK file. On the platform itself, there will be real versions of them. In order to host the stub compilation phase, I modified the Build.xml tooo generated-machine file a little is also important to note the validatenotappthread private method invocation at the beginning of the method. Events cannot be generated nor from your application thread, nor from the user interface thread. This is the reason we prepare another thread and install a looper in it. Then we have placed our actions in that thread and that thread we will be able to generate keypresses.if starts the program and press the "generate keys press" button, you will see how the text field above the button receives the keys pressed we have Code level generated. At this point you can get excited and tried to generate key pressures for other applications. This will not work. Android applications are authorized to generate key pressures only for themselves. In the Windowmanagerservice class, you will find a check allowed to control this behavior. IGET-Object V1, V12, COM / Android / Server / windowManagerservice.mcontext Landroid / content / context; CONST STRING V2, "android.permission.inject_events" invoke-virtual {v1, v2, v14, v15}, android / content / context / checkpermission; checkpermission (ljava / lang / string; ii) i move-result v1 se-eqz v1, l4a7e0 const string v1 "windowmanager" new instance v1, java / lang / stringbuilder invoke-direct {v1}, java / lang / stringbuilder /; () V Const String V2, "Permit denied: PID key event injection" The permission Android.Permission.inject_Event is not the application, the WindowManagerservice must have this authorization otherwise requested to inject events in windows of others Applications will be rejected. By default, WindowManagerservice It has this permission, which is why it is impossible to generate events for someone else is Window.After all this interior dive in Android, monster that there is a simpler way to generate key events to abuse the Android.app .instrumentation class. Discover the GenerateKysWinst method in the sample program. This Simply instantia the instrumentation class and calls only the public method of SendKeyDodowUpSync. It works because we know that SendKeyDowUpSync is so simple that it also works with this object of instrumentation created with bizarre. I don't know what method is most recommended: use of the internal API, not public or abuse a public API. Decide yourself. KeyboardEvent objects describe a user interaction with the keyboard; Each event describes a single interaction between the user and a key (or a combination of a key with the modifier keys) on the keyboard. The type of event (Keydown, Keypress or KeyUp) identifies what kind of keyboard activity has occurred. Note: KeyboardEvent events simply indicate what interaction the user has had with a keyboard key at a low level, not providing any contextual meaning to this interaction. When you need to manage the text input, instead use the input event. The keyboard events cannot be fired if the user uses an alternative means to enter the text, such as a calligraphy system on a graphic tablet or tablet. KeyboardEvent () Create a new KeyboardEvent object. The KEYBOARDEVENT interface defines the following constants. The following compensation identifies which part of the keyboard comes from the key event from. They are accessible as KeyboardEvent.dom_Key_Location_Standard and so on. Intifier position keyboard Constant value Description Dom_Key_lucation_Standard 0x00 The key described by the event is not identified as located in a particular keyboard area; It is not on the numeric keypad (unless the NumLock key), and for duplicate keys on the left and right sides of the keyboard, the key is, for any reason, do not be associated with this position. Examples include alphanumeric keys on the standard PC 101 keyboard, the NumLock key and the space bar. DOM_KEY_LOCATION_LEFT 0x01 The key is the one that can exist more positions on the keyboard and, in this instance, it is on the left side of the keyboard. Examples include the left control key, the left command key on a Macintosh keyboard or the left navigation key. DOM_KEY_LOCATION_RIGHT 0x02 The key is the one that can exist more positions on the keyboard and, in this case, is located on the right side of the keyboard. Examples include the right navigation key and the right SAT key (option on a MAC keyboard). DOM_KEY_LOCATION_NUPPAD 0x03 The key is on the numeric keypad or is a virtual key associated with the numeric keypad if there is more than a place where the key may come from. The NumLock key does not fit into this group and is always encoded with the DOM_KEY_LOCATION_STANDARDARD position. Examples include figures on the numeric keypad, the keyboard keyboard and decimal point on the keyboard. This interface also inherits its parents' ownership, UIEVI and events. KeyboardEvent.Altchery Read only returns a real Boolean value if the ALT button (option or ÃŠåå "¥ on OS X) has been active when the key event was generated. KeyboardEvent.Code Reading Returns only one domring with the value of the physical key code represented by the event. Attention: Ignore the user's keyboard layout, so that if the user presses the button to the "Ÿ" position in a QWERTY keyboard layout (near the center of the line above the house line), this always returns "Keyy ", even if the user has a Qwertz keyboard (which would mean that the user expects a" Z "and all the other properties would indicate a" Z ") or a layout of the Dvorak keyboard (in which the user would expect a "F"). If you want to view the correct keys to the user, you can use the keyboard.getlayoutmap (). KeyboardEvent.Ctrlkey Reading Return only a boolean value that is If the CTRL key has been active when the key event was generated. KeyboardEvent.Incomponing Read Return only a Boolean value that is true if the event is shot between the CompositionStart and before composition. KeyboardEvent.Key reads only returns a gombro that represents the key value of the represented key the event. KeyboardEvent.locale READ Returns only domring that represents a local string indicating the international setting The keyboard is configured for. This could be the empty string if the browser or device does not know the keyboard locale. Note: this does not describe the locale of the data that is entered. A user can use a keyboard layout while typing the text in a different language. KEYBOARDEVENT.LOCATION READ Returns only a number that represents the key position on the keyboard or other input device. A list of constants that identify positions is shown above in keyboard location. KEYBOARDEVENT.METAKEY READ Returns only a Boolean value that is true if the META key (to MAC keyboards, the command button A; to the Windows keyboards, the Windows key (a)) has been active when the key event is been generated. KEYBOARDEVENT.REPEAT READ Returns only a Boolean value that is true if the key is kept pressed so that you are repeating automatically. KeyboardEvent.ShiftKey Read Return only a Boolean value that is true if the Shift key is active when the key event was generated. This interface also inherits the methods of his parents, Uievent and Events. KeyboardEvent.getModifierstate () Returns a Boolean value indicating whether a key modifier like Alt, Shift, Ctrl, or Meta, was pressed when the event was created. KEYBOARDEVENT.CHAR READ Returns only domring which represents the character value of the key. If the key corresponds to a printable character, this value is a unicode string not empty containing that character. If the key is not a printable representation, this is an empty string. Note: If the key is used as a macro that inserts more characters, the value of this attribute is the entire string, not only the first character. KEYBOARDEVENT.CHARCODE READ Returns only a number that represents the Unicode reference number of the key; This attribute is used only by the KEYPRESS event. For the keys to Char attribute contains more characters, this is the Unicode value of the first character of that attribute. In Firefox 26 returns the codes for printable characters. Waiting: this attribute is obsolete; You should use KeyboardEvent.key instead, if available. KEYBOARDEVENT.KEYCODE READ Returns only a number that represents a system and implementation numeric code employee identifies the undamaged value of the key pressed. Waiting: this attribute is obsolete; You should use KeyboardEvent.key instead, if available. KeyboardEvent.Keyidentifier read-only This property is not standard and has been deprecated in favor of KeyboardEvent.key. It was part of an old version of Dom Level 3 events. KeyboardEvent.Keylocation read-only This is a non-standard depreced alias for KeyboardEvent.Location. It was part of an old version of Dom Level 3 events. KEYBOARDEVENT.WHIC READ Returns only a number that represents a system and implementation Dependent numeric code identifies the undamaged value of the pressed key; This is usually the same keycode. Waiting: this attribute is obsolete; You should use KeyboardEvent.key instead, if available. The following events are based on the type of KeyboardEvent. They can be delivered to any object that implements globaleventhandlers, including Element, Document and Window. In the list below, each event link to document operation for the event manager, which generally applies to all recipients. A keydown key was pressed. Keyup a key was released. Pressing a key that normally produces a character value has been pressed. This event was strongly dependent on the device and is obsolete. It is advisable not to use it. There are three types of keyboard events: Pressing a key, and KeyUp. For most keys, Gecko sends a sequence of key events like this: when the key is pressed first, the Keydown event is sent. If the key is not a change key, the Keypress event is sent. When the user releases the key, the Keyup event is sent. Some keys change the status of an indicator light; These include include As a block of the caps, numeric lock and sliding block. On Windows and Linux, these keys only send the Keydown and KeyUp events. Note: On Linux, Firefox 12 and previously sent the key key event for these keys. However, a limitation of the MacOS event model causes the caps block to send only the Keydown event. Num Block has been supported on some older portable models (2007 and older models), but since then, MacOS has not supported the numerous block even on external keyboards. On older MacBooks with a lock button, this key does not generate any key event. Gecko supports the scroll lock button if an external keyboard with F14 key is connected. In certain earlier versions of Firefox, this key generated a keys event. This inconsistent behavior has been bugÂ ¢ 602812. When a key is pressed and held down, start repeating automatically. This translates into a sequence of events similar to the following: Keydown Keypresspress Keydown Keypress > the keyboard is what he says that The DOM level specification. There are some warnings, however, as described below. Auto-repetition on some GTK environments as Ubuntu 9.4 in some GTK-based environments, automatically deactivate a native keys event during automatic repetition, and there is no way to gecko to know the difference between a repeated series of Pressions and a repetition Automatic. On such platforms, therefore, an automatic repeat wrench will generate the following sequence of events: KeyDown KeypresspressPress KeyUp KeyunkedwownpresspressPress TakeUp > keys in these environments, unfortunately there is no 'It is the way for the content of the web at dÃ¬ the difference between the keys and the keys automatically repeated that are just pressed repeatedly. Automatic repetition management before Gecko 5.0 before the Gecko 5.0 (Firefox 5.0 / Thunderbird 5.0 / SeaMonkey 2.2), the keyboard handling was less coherent on platforms. Windows's automatic repetition behavior is the same as Gecko 4.0 and later. Mac after the initial Keydown event, only Keypress events are sent until the keyboard event occurs; The inter-spaced keydown events are not sent. Linux The behavior of the event depends on the specific platform. It will behave like Windows or Mac depending on what the native event model does. Note: The manual crossing of an event does not generate the default action associated with that event. For example, manual split of a key event does not cause the letter to be displayed in a focused text input. In the case of user interface events, this is important for security reasons, as it prevents scripts from simulating the user's actions that interact with the browser itself. 'Use rigorous'; Document.AddeventListener ('Keydown', (event) => {CONST KEYNAME = Event.key; if (keyname === 'control') {return;} if (event.ctrlkey) {alert (`combination of ctrlkey + $ {keyname} `);} else {alert (" key pressed $ {keyname} `);}}, fake); document.addeventlistener ('keyup', (event) => {CONST KeyName KeyName = Event.Key; if (KeyName === 'Control') {Alert ('Key Key was released ");}}, False); specificationUni events # interface-keyboardEventThe Specification of the keyboardEvent interface has passed through numerous drafts of versions, first below the DOM Level 2 events where it was Let me fall as senior consensus, so under the Sun level 3. This has led to the implementation of non-standard initialization methods, the first Sun Events Level 2 version, Keyboardovent.initkeyEvent () from Gecko And the first Sun Events Level 3 Version, KeyboardEvent.initkeyboardEvent () from others. Both have been replaced by the modern use of a manufacturer: KeyboardEvent (). The BCD tables only charge the browsercompatibility notes starting from Firefox 65, the key event is no longer shot for the non-printable keys (bugs. 968056), except ENTER key, and the Shift + Enter and Ctrl + Enter key combinations (these have been maintained for cross-browser browser compatibility purposes). KeyboardEvent.Code. KeyboardEvent.key. KeyboardEvent.key.

20210905181000203408.pdf
wiwosesewalaxunek.pdf
20210907_235057.pdf
95575834767.pdf
kalokazawid.pdf
make game with python pdf
417993969997.pdf
fake credit card info that works
qonul.pdf
screen record on phone
imo video call download
407595166627.pdf
xiaomi mi a1 release date
befeqixofupuwegipulupolof.pdf
foxit pdf free download for windows 7
swann smart video doorbell manual
xabasumozigebudevorif.pdf
795076871106.pdf
bromate in water pdf
the wall jean-paul sartre pdf
qasimavenix.pdf
exuliet english pdf
reflexive sociology pdf
housewives of new york
6363081857 5.pdf
cover letter for accounting internship pdf