VMware HTML Console SDK Guida alla programmazione

Per vSphere 5.5 e versioni successive (disponibilità generale)
e vCloud Director (Tech Preview)

Febbraio 2016

VMware HTML Console SDK
Guida alla programmazione

Sommario

HTML Console (WMKS) SDK Panoramica

HTML Console SDK è una libreria JavaScript implementata in base a webmks (WMKS), in grado di garantire gestione ed elaborazione dell'input tramite mouse, tastiera e tocco, nonché aggiornamenti dello schermo e modifiche del cursore durante una sessione desktop da un browser. Le applicazioni in esecuzione nel browser possono utilizzare JavaScript per accedere alle funzioni della console della macchina virtuale utilizzando l'API HTML Console SDK.

L'API HTML Console SDK contiene diversi metodi che possono essere utilizzati per connettersi e comunicare con una macchina virtuale. Attiva inoltre gli eventi per l'invio di notifiche agli utenti sulle modifiche dello stato della macchina virtuale. È possibile utilizzare questi metodi e callback per dare agli utenti finali la possibilità di gestire in remoto una macchina virtuale da qualsiasi sistema con il browser Web e il sistema operativo appropriati.

Browser supportati (inclusi quelli per iOS e Android)

- Internet Explorer 10+
- Firefox 24+
- Chrome 30+
- Safari 6.1+

Procedura di utilizzo dell'API HTML Console SDK

Solitamente, per utilizzare l'API HTML Console SDK, un'applicazione basata sul Web deve procedere come segue:

1. Caricare i file della libreria JavaScript HTML Console SDK.
2. Creare l'oggetto WMKS principale utilizzando il metodo factory createWMKS() con un determinato ID dell'elemento div nel DOM del documento e le opzioni per la creazione.
3. Registrare i callback JavaScript per rispondere agli eventi attivati da WMKS.
4. Utilizzare il metodo connect () HTML Console SDK per la connessione a una macchina virtuale di destinazione.
5. Utilizzare i metodi HTML Console SDK per interagire con la macchina virtuale connessa.
6. Utilizzare il metodo disconnect() HTML Console SDK per disconnettere la connessione (se attiva) e rimuovere il widget dall'elemento associato.

Posizionamento della console in una pagina Web

HTML Console SDK è disponibile nella sezione Download del sito Web www.vmware.com/it: Driver e strumenti per vSphere, vCloud Director, vRealize Automation e vCloud Air.

In ciascuna build sono disponibili un file wmks.min.js minimizzato e un file wmks.js non minimizzato. HTML Console SDK utilizza il widget jQuery per visualizzare la console VM e, di conseguenza, deve prima essere inclusa la libreria correlata a jQuery. Ecco un esempio:

1. Includere i componenti Javascript jQuery e jQuery UI nei tag script e il file CSS jQuery corrispondente nel tag foglio di stile del collegamento.
2. Includere wmks.js (o wmks.min.js) in un tag script e wmks-all.css nel tag foglio di stile del collegamento.
3. Specificare un div con ID appropriato.
4. Creare l'oggetto WMKS principale con il metodo factory WMKS.createWMKS().
5. Associare i callback per gli eventi WMKS.
6. Connettersi con l'URL di destinazione appropriato.

```
 <link rel="stylesheet" type="text/css" href="wmks-all.css" />

<script type="text/javascript" src="jquery-1.8.3.min.js"></script>
 <script type="text/javascript" src="jquery-ui.1.8.16.min.js"></script>
 <script type="text/javascript" src="wmks.js" type="text/javascript"></script>
<script>
        var wmks = WMKS.createWMKS("wmksContainer",{})
                    .register(WMKS.CONST.Events.CONNECTION_STATE_CHANGE,
                    function(event,data){
                        if(data.state == cons.ConnectionState.CONNECTED)
                        {
                         console.log("connection state change : connected");
                        }
                    });
        wmks.connect("ws://127.0.0.1:8080");
</script >
<div  id="wmksContainer" style="position:absolute;width:100%;height:100%"></div>
```

Connessione a una macchina virtuale remota

Modi per connettersi a una macchina virtuale remota:

WMKS può fungere da componente per offrire una connessione della console esclusivamente basata sul Web alle macchine virtuali gestite da vCenter Server utilizzando un proxy intermediario per gestire i requisiti di autenticazione.

Connettersi alla macchina virtuale utilizzando il seguente metodo:

```
wmks.connect(url);
```

L'URL deve rientrare in questo schema:

```
<ws | wss> :// <host:port>/ <path> /? <authentication info>
```

Disconnessione ed eliminazione

Chiamare questo comando al termine dell'utilizzo del componente WMKS. L'eliminazione di WMKS comporterà anche la disconnessione (se attiva) e la rimozione del widget dall'elemento associato.

Il metodo factory di WMKS

createWMKS()

Questo deve essere il primo metodo quando si utilizza HTML Console SDK. L'utilizzo di questo metodo genera il widget, che può mostrare lo schermo remoto e quindi restituire l'oggetto WMKS principale che può utilizzare tutte le API HTML Console SDK per connettersi a una macchina virtuale ed eseguire le operazioni.

Metodo	createWMKS(id, opzioni)
Parameter1	**id**(stringa): ID dell'elemento div. Questo elemento è il contenitore delle tele utilizzato per mostrare lo schermo remoto.
Parameter2	**options**:(oggetto json): la creazione di opzioni comprometterebbe il comportamento di WMKS. Vuoto significa "utilizza tutte le opzioni predefinite".
Valore restituito	L'oggetto WMKS principale, che può utilizzare tutte le API WMKS per connettersi a una macchina virtuale ed eseguire le operazioni.
Chiamata di esempio	var wmks =WMKS.createWMKS("container",{});

Opzioni per la creazione

Analogamente a webMKS integrato nel client Web vSphere, anche WMKS dispone di più opzioni mirate a controllarne il comportamento. Ecco il confronto tra SDK e webMKS integrati nel client Web vSphere:

rescale

Booleano: il valore predefinito è true e indica se ridimensionare lo schermo remoto per adattarlo alle dimensioni allocate del contenitore.

position

Enumerazione: può essere un qualsiasi valore di WMKS.CONST.Position, il valore predefinito è WMKS.CONST.Position.CENTER. Indica la posizione (centrale o superiore sinistra) che lo schermo remoto deve assumere nel contenitore.

changeResolution

Booleano: il valore predefinito è true. Se l'opzione changeResolution è true, WMKS invia la richiesta di risoluzione della modifica alla macchina virtuale connessa, con la risoluzione richiesta (larghezza e altezza) uguale alle dimensioni allocate del contenitore.

Queste tre opzioni seguono un ordine di procedura specifico basato sulla priorità.

1. Verificare l'opzione **changeResolution** e, se il valore è true, webmks invierà la richiesta di modifica della risoluzione alla macchina virtuale connessa.
2. Se la richiesta non va a buon fine, le operazioni **rescale** e **position** possono essere utilizzate per effettuare le modifiche corrispondenti.

audioEncodeType

Enumerazione: può essere qualsiasi valore di WMKS.CONST.AudioEncodeType. Indica quale tipo di metodo di codifica audio è in uso: vorbis, opus o aac.

useNativePixels

Booleano: il valore predefinito è false. Consente l'utilizzo delle dimensioni pixel native nel dispositivo. Per iPhone 4+ o iPad 3+, l'attivazione di questa opzione abilita la "modalità Retina", che offre più spazio sullo schermo per il guest, riducendo le dimensioni generali.

useUnicodeKeyboardInput

Booleano: il valore predefinito è false. Per tutti gli input dell'utente, WMKS può inviare due tipi di messaggi al server: codici scansione tastiera o Unicode. In questo caso *true* indica che è preferibile utilizzare Unicode.

useVNCHandshake

Booleano: il valore predefinito è true. Consente un handshake VNC standard. Questa opzione deve essere utilizzata quando l'endpoint utilizza l'autenticazione VNC standard. Impostare su false se ci si connette a un proxy che esegue l'autenticazione tramite authd e non esegue un handshake VNC.

sendProperMouseWheelDeltas

Booleano: il valore predefinito è false. Le versioni precedenti della libreria normalizzano i delta degli eventi della rotellina del mouse in uno dei tre valori di seguito: [-1, 0, 1]. Quando l'opzione è impostata su *true*, i delta effettivi del browser vengono inviati al server.

reverseScrollY

Booleano: il valore predefinito è false. Se il flag è impostato su *true*, invia alla macchina virtuale connessa il valore opposto della rotellina del mouse.

retryConnectionInterval

Numero intero: il valore predefinito è -1. L'intervallo (in millisecondi) per il nuovo tentativo di connessione quando il primo tentativo di configurare una connessione tra il client Web e il server non va a buon fine. Un valore inferiore a 0 indica che "non verrà eseguito un nuovo tentativo".

ignoredRawKeyCodes

Array: il valore predefinito è vuoto. Tutti questi codici verranno ignorati e non inviati al server.

fixANSIEquivalentKeys

Booleano: il valore predefinito è false. Consente la correzione di qualsiasi codice di layout statunitense non ANSI per la corrispondenza agli equivalenti codici di layout ANSI statunitensi. Tenta di correggere la pressione dei tasti se il layout della tastiera internazionale dell'utente contiene un tasto presente anche nella tastiera ANSI statunitense, ma in una posizione diversa o se non esiste corrispondenza con lo stato MAIUSC o NON MAIUSC di una tastiera ANSI statunitense.

keyboardLayoutId

Stringa: il valore predefinito è "en-US" (inglese Stati Uniti). Vengono offerte diverse configurazioni linguistiche della tastiera per il guest. Gli utenti devono determinare il layout della tastiera e mantenere lo stesso layout della tastiera del desktop remoto e del desktop locale. In questa versione di HTML console SDK, VMware non supporta i dispositivi mobili. Esistono due tipi di codici che possono essere inviati al sistema operativo guest. Il codice supportato per il mapping internazionale è vScancode.

Internet Explorer/Firefox/Chrome sono supportati su Windows e Chrome/Safari sono supportati su Mac.
Aggiungere un elenco di selezione in HTML o utilizzare altri modi per consentire agli utenti di scegliere la lingua che utilizzano. Nel file js utilizzare l'API setOption come segue:

```
<select id="selectLanguage">
        <option value="en-US">English</option>
        <option value="ja-JP_106/109">Japanese</option>
        <option value="de-DE">German</option>
        <option value="it-IT">Italian</option>
        <option value="es-ES">Spanish</option>
        <option value="pt-PT">Portuguese</option>
</select>
```

```
$('#selectLanguage').change(function(){
   if(!wmks) return;
   var keyboardLayoutId = $(this).find(":selected").val();
   wmks.setOption('keyboardLayoutId',keyboardLayoutId);
 });
```

VCDProxyHandshakeVmxPath

Stringa: il valore predefinito è null. Durante la connessione passare al protocollo VNC. vncProtocol risponderà a una richiesta di connessione per un percorso VMX con la stringa VCDProxyHandshakeVmxPath fornita durante la connessione a **vCloud Director**.

enableUint8Utf8

Booleano: il valore predefinito è false. Se il progetto non supporta correttamente il protocollo binario, per abilitare il protocollo uint8utf8 precedente è necessaria l'opzione enableUint8Utf8.

Gestione degli eventi WMKS

Elenco degli eventi WMKS

WebMKS genera eventi quando lo stato della macchina virtuale attualmente connessa cambia o in risposta ai messaggi della macchina virtuale attualmente connessa. Tutti gli eventi WMKS sono elencati in WMKS.CONST.Events.

Tutti i gestori di eventi hanno due parametri: **event** e **data**. **Event** è un evento jQuery e tutti i parametri speciali degli eventi vengono archiviati in **data**.

connectionstatechange

L'evento connectionstatechange viene generato in risposta a una modifica dello stato di connessione, ad esempio da "disconnesso" a "connessione in corso"o da "connessione in corso" a "connesso".

Parametri in **data**:

- state: può essere un qualsiasi valore in WMKS.CONST. ConnectionState può essere:
  1. "connessione in corso"
  2. "connesso"
  3. "disconnesso"
- Se lo stato è WMKS.CONST. ConnectionState. CONNESSIONE IN CORSO, in data esistono altri due parametri: ***vvc*** e ***vvcSession***.
- Se lo stato è WMKS.CONST. ConnectionState. DISCONNESSO, in data esistono altri due parametri: ***reason*** e ***code***.

screensizechange

L'evento screensizechange è generato in risposta alle modifiche della dimensione dello schermo della macchina virtuale attualmente connessa.
Parametri in data:

- width: la larghezza (in pixel) della macchina virtuale attualmente connessa.
- height: l'altezza (in pixel) della macchina virtuale attualmente connessa.

fullscreenchange

L'evento fullscreenChange viene generato quando la console WMKS esiste oppure passa alla modalità a schermo intero.
Parametri in data:

- isFullScreen: Booleano, true indica il passaggio alla modalità a schermo interno, false indica l'uscita dalla modalità a schermo intero.

error

L'evento error viene generato quando si verifica un errore, ad esempio un errore di autenticazione, un errore di WebSocket o un errore del protocollo.

Parametri in data:

- errorType: può essere un qualsiasi valore in WMKS.CONST.Events.ERROR:
  1. AUTHENTICATION_FAILED: "authenticationfailed",
  2. WEBSOCKET_ERROR: "websocketerror",
  3. PROTOCOL_ERROR: "protocolerror"

Keyboardledschanged

L'evento keyboardledschanged viene generato quando cambia lo stato di blocco del LED della tastiera della macchina virtuale remota.

Parametri in data:

- Data è un numero intero: 1 vuol dire BLOC SCORR, 2 vuol dire BLOC NUM, 4 vuol dire BLOC MAIUSC.

heartbeat

L'evento heartbeat viene generato quando si riceve un messaggio heartbeat lato server.

- Data è un numero intero che indica l'intervallo di heartbeat.

audio

L'evento audio viene generato quando si riceve un messaggio audio dal server.
Parametri in data:

- sampleRate
- numChannels
- containerSize
- sampleSize
- length
- audioTimestampLo
- audioTimestampHi
- frameTimestampLo
- frameTimestampHi
- flags
- data

copy

L'evento copy viene generato quando il server invia un evento di taglio testo.

- Data è la stringa che viene copiata.

toggle

L'evento toggle viene generato se vengono nascosti o visualizzati la tastiera, la tastiera estesa o il trackpad.
Parametri in data:

- type: può essere “KEYBOARD”, "EXTENDED_KEYPAD", "TRACKPAD"
- visibility: booleano, true indica "visualizza" e false indica "nascondi".

Impostazione dei gestori per gli eventi WMKS

In HTML Console SDK, specificare il metodo register() e unregister per aggiungere e rimuovere i gestori per gli eventi WMKS.

register()

Questo metodo è utilizzato per registrare il gestore degli eventi per WMKS.

Metodo	register(eventName, eventHandler)
Parameter1	eventName (costante, qualsiasi valore di WebMKS.Events)
Parameter2	eventHandler (funzione di callback javascript)
Valore restituito	Nessuno
Chiamata di esempio	var connectionStateHandler = function(event, data){}; wmks.register(WebMKS.Events.CONNECTION_STATE_CHANGE, connectionStateHandler);

unregister()

Questo metodo è utilizzato per annullare la registrazione di un gestore di eventi per WMKS.
In assenza di parametri, questo metodo rimuove tutti i gestori di eventi.
Nel caso di un solo parametro, rimuove tutti i gestori di eventi per lo specifico eventName.

Metodo	unregister(eventName, eventHandler)
Parameter1	eventName (costante, qualsiasi valore di WebMKS.Events)
Parameter2	eventHandler (funzione di callback javascript)
Valore restituito	Nessuno
Chiamata di esempio	wmks.unregister(WebMKS.Events.CONNECTION_STATE_CHANGE, connectionStateHandler);

Metodi dell'oggetto HTML Console SDK

Dopo aver richiamato il metodo createWMKS(), si ottiene un oggetto WMKS che può utilizzare l'API WMKS per connettersi a una macchina virtuale ed eseguire le operazioni. In questo esempio, l'oggetto WMKS verrà chiamato "wmks".

API generali

I metodi delle API generali offrono informazioni su WMKS. Questi metodi possono essere chiamati in qualsiasi momento, anche prima della connessione alla macchina virtuale di destinazione.

getVersion()

Recupera il numero di versione corrente di HTML Console SDK.

Metodo	getVersion()
Parametri	Nessuno
Valore restituito	Stringa. Contiene il numero della versione completa di HTML Console SDK.
Chiamata di esempio	var version = wmks.getVersion();

getConnectionState()

Recupera lo stato corrente della connessione.

Metodo	getConnectionState()
Parametri	Nessuno
Valore restituito	Costante. Qualsiasi valore in WMKS. CONST.ConnectionState.
Chiamata di esempio	var state = wmks.getConnectionState();

API relative al ciclo di vita

connect()

Connette WMKS a una macchina virtuale remota tramite l'URL WebSocket e configura l'interfaccia utente.

Metodo	connect()
Parametro	URL WebSocket, il tipo è la stringa nel formato: <ws | wss> :// <host:port>/ <path> /? <authentication info>
Valore restituito	Nessuno
Chiamata di esempio	wmks.connect("ws://localhost:8080");

disconnect()

Esegue la disconnessione dalla macchina virtuale remota ed elimina l'interfaccia utente.

Metodo	disconnect()
Parametro	Nessuno
Valore restituito	Nessuno
Chiamata di esempio	wmks.disconnect();

destroy()

Interrompe la connessione (se attiva) con la macchina virtuale e rimuove il widget dall'elemento associato. I clienti devono chiamare questo parametro prima di rimuovere l'elemento dal DOM.

Metodo	destroy()
Parametro	Nessuno
Valore restituito	Nessuno
Chiamata di esempio	wmks.destroy();

API correlate allo schermo

setRemoteScreenSize()

Invia una richiesta per impostare la risoluzione dello schermo della macchina virtuale attualmente connessa. Qui, se i parametri relativi all'altezza e alla larghezza sono maggiori rispetto alla dimensione allocata del widget WMKS, il ridimensionamento verrà normalizzato.

Nota: questo metodo può funzionare correttamente solo quando l'opzione changeResolution è impostata su *true*.

Metodo	setRemoteScreenSize(width,height)
Parameter1	width rappresenta la larghezza desiderata della macchina virtuale connessa in pixel
Parameter2	height rappresenta l'altezza desiderata della macchina virtuale connessa in pixel
Valore restituito	Nessuno
Chiamata di esempio	wmks.setRemoteScreeSize(800,600);

getRemoteScreenSize()

Recupera l'altezza e la larghezza in pixel dello schermo della macchina virtuale attualmente connessa.

Metodo	getRemoteScreenSize()
Parametro	Nessuno
Valore restituito	Oggetto nel formato {width:, height:}
Chiamata di esempio	var size = wmks.getRemoteScreenSize();

updateScreen()

Modifica la risoluzione o ridimensiona lo schermo remoto per adattarlo alla dimensione attualmente allocata.

Il comportamento di updateScreen dipende dall'opzione changeResolution, dal ridimensionamento e dalla posizione:

1) Se l'opzione changeResolution è true, invia la richiesta di modifica della risoluzione alla macchina virtuale connessa e la risoluzione richiesta (larghezza e altezza) è uguale alle dimensioni allocate del contenitore.

2) Verificare l'opzione di ridimensionamento: se è impostata su true, ridimensionare lo schermo remoto per adattarlo alle dimensioni allocate del contenitore.

3) Verificare l'opzione relativa alla posizione: se le dimensioni dello schermo remoto non corrispondono a quelle allocate del contenitore, lo schermo remoto verrà posizionato al centro o in alto a sinistra del contenitore in base al relativo valore.

Metodo	updateScreen()
Parametro	Nessuno
Valore restituito	Nessuno
Chiamata di esempio	wvmrc.updateScreen();

API correlate alla modalità a schermo intero

canFullScreen()

Indica se la funzionalità a schermo intero è abilitata nel browser. La modalità a schermo intero è disabilitata in Safari poiché non supporta l'input tastiera in modalità a schermo intero per motivi di sicurezza.

Metodo	canFullScreen()
Parametro	Nessuno
Valore restituito	Booleano. True indica che è possibile, false indica che non è possibile.
Chiamata di esempio	wmks.canFullScreen();

isFullScreen()

Indica se il browser è in modalità a schermo intero.

Metodo	isFullScreen()
Parametro	Nessuno
Valore restituito	Booleano. True indica che è attiva la modalità a schermo intero, false indica che non è attiva.
Chiamata di esempio	wmks.isFullScreen();

enterFullScreen()

Forza il browser a passare alla modalità a schermo intero, se supportata. Nella modalità a schermo intero verrà visualizzato solo lo schermo remoto.

Metodo	enterFullScreen()
Parametro	Nessuno
Valore restituito	Nessuno
Chiamata di esempio	wmks.enterFullScreen();

exitFullScreen()

Forza il browser a uscire dalla modalità a schermo intero.

Metodo	exitFullScreen()
Parametro	Nessuno
Valore restituito	Nessuno
Chiamata di esempio	wmks.exitFullScreen();

API correlate all'input

sendInputString()

Invia una stringa come input tastiera al server.

Metodo	sendInputString(string)
Parametri	Stringa
Valore restituito	Nessuno.
Chiamata di esempio	wmks.sendInputString("test");

sendKeyCodes()

Invia una serie di codici speciali alla macchina virtuale. Viene selezionato un array di codici speciali e vengono inviati gli eventi KeyDown per ciascun array nell'ordine elencato. Successivamente vengono inviati gli eventi KeyUp per ciascun array nell'ordine inverso. È possibile inviare combinazioni di tasti, ad esempio CTRL-ALT-CANC.

Metodo	sendKeyCodes ()
Parametri	Array. Ciascun elemento nell'array può essere un codice o un Unicode. Codice per nessun tasto stampabile, Unicode per un carattere stampabile. Se si utilizza Unicode, il valore negativo, ad esempio -118, vuol dire "v"
Valore restituito	Nessuno.
Chiamata di esempio	wmks.sendKeyCodes([17,18,46]) ; // CTRL + ALT + CANC

sendCAD()

Invia una sequenza di tasti CTRL-ALT-CANC alla macchina virtuale attualmente connessa.

Metodo	sendCAD()
Parametri	Nessuno.
Valore restituito	Nessuno.
Chiamata di esempio	wmks.sendCAD();

API correlate ai dispositivi mobili

enableInputDevice()

Abilita il dispositivo di input (tastiera, tastiera estesa, trackpad) nel dispositivo mobile e lo prepara all'utilizzo.

Metodo	enableInputDevice (deviceType)
Parametri	deviceType:(Costante) Qualsiasi valore in WMKS.CONST.InputDeviceType.
Valore restituito	Nessuno.
Chiamata di esempio	wmks.enableInputDevice(WMKS. CONST .InputDeviceType. TASTIERA) ;

disableInputDevice()

Disabilita il dispositivo di input (tastiera, tastiera estesa, trackpad) nel dispositivo mobile e lo elimina.

Metodo	disableInputDevice (deviceType)
Parametri	deviceType:(Costante) Qualsiasi valore in WMKS.CONST.InputDeviceType.
Valore restituito	Nessuno.
Chiamata di esempio	wmks.disableInputDevice(WMKS. CONST .InputDeviceType. KEYBOARD) ;

showKeyboard()

Consente di visualizzare la tastiera in un dispositivo mobile.

Metodo	showKeyboard()
Parametri	Nessuno.
Valore restituito	Nessuno.
Chiamata di esempio	wmks.showKeyboard();

hideKeyboard()

Consente di nascondere la tastiera in un dispositivo mobile.

Metodo	hideKeyboard()
Parametri	Nessuno.
Valore restituito	Nessuno.
Chiamata di esempio	wmks.hideKeyboard();

toggleExtendedKeypad()

Consente di visualizzare/nascondere la tastiera estesa nel dispositivo mobile a seconda della visibilità corrente.

Metodo	toggleExtendedKeypad()
Parametri	Una mappa può includere minToggleTime(ms), ad esempio {minToggleTime: 50}, se l'utente prova a chiamare questo metodo di attivazione con una frequenza eccessiva. Se la durata è inferiore a minToggleTime, l'esecuzione non sarà possibile.
Valore restituito	Nessuno.
Chiamata di esempio	wmks.toggleExtendedKeypad();

toggleTrackpad()

Consente di visualizzare/nascondere il trackpad nel dispositivo mobile a seconda della visibilità corrente.

Metodo	toggleTrackpad()
Parametri	Una mappa può includere minToggleTime(ms), ad esempio {minToggleTime: 50}, se l'utente prova a chiamare questo metodo di attivazione con una frequenza eccessiva. Se la durata è inferiore a minToggleTime, l'esecuzione non sarà possibile.
Valore restituito	Nessuno.
Chiamata di esempio	wmks.toggleTrackpad();

toggleRelativePad ()

Consente di visualizzare/nascondere il trackpad del mouse relativo sullo schermo. Invierà l'evento del mouse relativo anziché l'evento del mouse assoluto dopo l'apertura del pad relativo. Può essere utilizzato quando nel sistema operativo guest remoto non sono installati VMware Tools. Aggiungere un pulsante toggleRelativeMouse nell'interfaccia utente e associarlo all'API toggleRelativePad(). Dopo aver fatto clic, il trackpad relativo visualizzerà e invierà l'evento del mouse relativo al guest remoto. Questa operazione è utile quando il sistema operativo guest non installa VMware Tools e non può accettare l'evento del mouse assoluto. Se infatti il server non restituisce alcuna immagine del cursore, è necessario visualizzare l'immagine del cursore locale. È quindi necessario aggiungere wmks-all.css nelle cartelle css e img. La cartella di riferimento è wmkssdk.

Metodo	toggleRelativePad ()
Parametri	Una mappa può includere minToggleTime(ms), ad esempio {minToggleTime: 50}, se l'utente prova a chiamare questo metodo di attivazione con una frequenza eccessiva. Se la durata è inferiore a minToggleTime, l'esecuzione non sarà possibile.
Valore restituito	Nessuno.
Chiamata di esempio	wmks. toggleRelativePad ();

API correlate alle opzioni

setOption()

Consente di modificare il valore dell'opzione. Solo le opzioni elencate di seguito possono utilizzare questo metodo:

- rescale
- position
- changeResolution
- useNativePixels
- reverseScrollY
- fixANSIEquivalentKeys
- sendProperMouseWheelDeltas
- keyboardLayoutId

Metodo	setOption(optionName, optionValue)
Parameter1	optionName(stringa del nome dell'opzione)
Parameter2	optionValue
Valore restituito	Nessuno
Chiamata di esempio	wmks.setOption("changeResolution",false);

Appendice

Costanti utilizzate in WMKS

- **Position**: WMKS mostra lo schermo remoto della macchina virtuale in proporzione uguale. Di conseguenza, dopo il ridimensionamento, lo schermo remoto potrebbe continuare ad avere dimensioni diverse rispetto al contenitore. Qui ***Position*** offre due possibili opzioni per impostare lo schermo nella stessa posizione del contenitore.

- **ConnectionState**: Esistono 3 possibili stati di connessione quando si cerca di eseguire la connessione alla macchina virtuale remota.

- **Events**: Tutti i nomi degli eventi che WMKS può attivare sono elencati qui.

- **ErrorType**: possibili tipi di errori nel ciclo di vita di WMKS.

- **InputDeviceType**: HTML Console SDK supporta la visualizzazione delle console delle macchine virtuali nei dispositivi mobili e in questo campo vengono elencati i possibili dispositivi di input.

(Andare alla pagina successiva per l'elenco delle costanti)

```
Position: {
   CENTER:   0,
   LEFT_TOP: 1
},

ConnectionState: {
   CONNECTING: "connecting",
   CONNECTED: "connected",
   DISCONNECTED: "disconnected"
},

Events: {
   CONNECTION_STATE_CHANGE: "connectionstatechange",
   REMOTE_SCREEN_SIZE_CHANGE: "screensizechange",
   FULL_SCREEN_CHANGE: "fullscreenchange",
   ERROR: "error",
   KEYBOARD_LEDS_CHANGE: "keyboardledschanged",
   HEARTBEAT: "heartbeat",
   AUDIO: "audio",
   COPY: "copy",
   TOGGLE: "toggle"
},

ErrorType: {
   AUTHENTICATION_FAILED: "authenticationfailed",
   WEBSOCKET_ERROR: "websocketerror",
   PROTOCOL_ERROR: "protocolerror"
},

AudioEncodeType: {
   VORBIS: "vorbis",
   OPUS: "opus",
   AAC: "aac"
},

InputDeviceType: {
   KEYBOARD: 0,
   EXTENDED_KEYBOARD: 1,
   TRACKPAD: 2
}
```