VMware HTML Console SDK
Manuel de programmation

Pour vSphere 5.5 et versions ultérieures (disponibilité générale)
et vCloud Director (aperçu technologique)

Octobre 2016

VMware HTML Console SDK
Manuel de programmation

Table des matières

HTML Console (WMKS) SDK Présentation

HTML Console SDK est une bibliothèque JavaScript mise en œuvre via WebMKS (WMKS), capable de fournir une souris, un clavier, des fonctions de gestion et de traitement de saisie tactile, des mises à jour d'écran, ainsi que des modifications de curseur pour une session de poste de travail lancée à partir d'un navigateur. JavaScript peut être utilisé par les applications qui s'exécutent dans un navigateur pour accéder aux fonctions de console de machine virtuelle à l'aide de l'API de HTML Console SDK.

Cette API contient diverses méthodes pouvant être utilisées pour établir une connexion et communiquer avec une machine virtuelle. Elle déclenche également des événements afin d'informer les utilisateurs des changements d'état de la machine virtuelle. Ces méthodes et ces rappels peuvent être utilisés pour permettre aux utilisateurs finaux de gérer une machine virtuelle à distance à partir de n'importe quel système doté du système d'exploitation et du navigateur Web appropriés.

Navigateurs pris en charge (y compris sur iOS et Android)

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

Procédure d'utilisation de l'API de HTML Console SDK

Pour utiliser l'API de HTML Console SDK, une application Web doit généralement procéder comme suit :

1. Charger les fichiers de bibliothèque JavaScript de HTML Console SDK.
2. Créer l'objet central WMKS à l'aide de la méthode de fabrique createWMKS() avec un ID d'élément div donné dans le DOM de document et les options de création.
3. Enregistrer des rappels JavaScript afin de répondre aux événements déclenchés par WMKS.
4. Établir une connexion avec une machine virtuelle cible à l'aide de la méthode connect() de HTML Console SDK.
5. Interagir avec la machine virtuelle connectée à l'aide des méthodes de HTML Console SDK.
6. Désactiver la connexion (le cas échant) à l'aide de la méthode disconnect() de HTML Console SDK et supprimer le widget de l'élément associé.

Mise en œuvre de la console sur une page Web

HTML Console SDK est disponible dans la section Téléchargements du site www.vmware.com/fr : pilotes et outils pour vSphere, vCloud Director, vRealize Automation et vCloud Air.

Chaque build contient une version réduite du fichier (wmks.min.js) et une version normale (wmks.js). HTML Console SDK utilise le widget jQuery pour afficher la console de machine virtuelle. Par conséquent, la bibliothèque associée à jQuery doit être ajoutée en premier lieu. Voici un exemple :

1. Inclure les composants JavaScript jQuery et jQuery UI dans les balises du script, puis le fichier css jQuery correspondant dans la balise de la feuille de style du lien.
2. Inclure wmks.js (ou wmks.min.js) dans une balise du script, puis wmks-all.css dans la balise de la feuille de style du lien.
3. Fournir un div avec un ID approprié.
4. Créer l'objet central WMKS avec la méthode de fabrique WMKS.createWMKS().
5. Lier les rappels pour les événements WMKS.
6. Se connecter avec l'URL de destination appropriée.

```
 <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>
```

Connexion à une machine virtuelle distante

Méthodes pour établir une connexion avec une machine virtuelle distante :

WMKS peut être utilisé comme un composant pour établir une connexion Web parfaite entre la console et les machines virtuelles gérées par vCenter Server, à l'aide d'un proxy intermédiaire pour gérer les conditions d'authentification.

Connectez-vous à la machine virtuelle à l'aide de la méthode suivante :

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

L'URL doit suivre le schéma suivant :

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

Déconnexion et destruction

Effectuez cette opération après en avoir terminé avec le composant WMKS. La destruction du WMKS entraînera également une déconnexion (le cas échéant) et supprimera le widget de l'élément associé.

Méthode de fabrique du WMKS

createWMKS()

Cette méthode doit être la première méthode lorsque vous utilisez le HTML Console SDK. Lorsqu'elle est utilisée, un widget capable d'afficher l'écran distant est généré, puis l'objet central WMKS est renvoyé. Cet objet peut utiliser toutes les API de HTML Console SDK pour se connecter à une machine virtuelle et effectuer des opérations.

Méthode	createWMKS(id, options)
Paramètre1	**id**(chaîne) : ID de l'élément div. Cet élément est le conteneur de zone utilisé pour afficher l'écran distant.
Paramètre2	**options**:(objet json) : les options de création affectent le comportement du WMKS. Lorsque cette méthode est vide, cela signifie « utiliser toutes les options par défaut ».
Valeur renvoyée	Objet central WMKS, capable d'utiliser toutes les API WMKS pour établir une connexion à une machine virtuelle et effectuer des opérations.
Exemple d'appel	var wmks =WMKS.createWMKS("container",{});

Options de création

Comme le WebMKS intégré à vSphere Web Client, WMKS dispose également de nombreuses options permettant de gérer son comportement. Vous trouverez ci-dessous une comparaison entre le SDK et le WebMKS intégré à vSphere Web Client :

rescale

Booléen : la valeur par défaut est « true » ; indique si l'écran distant doit être redimensionné pour s'adapter à la taille allouée du conteneur.

position
Énumération : peut prendre n'importe quelle valeur de WMKS.CONST.Position ; la valeur par défaut est WMKS.CONST.Position.CENTER. Elle indique la position (centre ou en haut à gauche) de l'écran distant dans le conteneur.

changeResolution
Booléen : la valeur par défaut est « true ». Lorsque l'option changeResolution contient la valeur « true », WMKS doit envoyer la demande de modification de la résolution à la machine virtuelle connectée, la résolution demandée (largeur et hauteur) étant identique à la taille allouée du conteneur.

Ces trois options suivent un ordre de procédure spécifique en fonction de leur priorité.

1. Vérifiez l'option **changeResolution**, et si elle contient la valeur « true », WebMKS envoie la demande de modification de résolution à la machine virtuelle connectée.
2. Si la demande échoue, les opérations **rescale** et **position** peuvent être utilisées pour effectuer les modifications correspondantes.

audioEncodeType
Énumération : peut prendre n'importe quelle valeur de WMKS.CONST.AudioEncodeType. Cette option indique le type de méthode d'encodage audio utilisé : vorbis, opus ou aac.

useNativePixels
Booléen : la valeur par défaut est « false ». Permet l'utilisation des tailles de pixel natives du périphérique. Sur les iPhone 4+ ou iPad 3+, cette option active le mode « Retina », qui offre un espace d'écran plus important pour l'invité, réduisant tous les éléments affichés.

useUnicodeKeyboardInput
Booléen : la valeur par défaut est « false ». Pour toutes les saisies de l'utilisateur, WMKS envoie deux types de message au serveur : codes de touche enfoncée du clavier ou Unicode. Dans ce contexte, la valeur « *true* » signifie que le client doit utiliser Unicode, si cela est possible.

useVNCHandshake
Booléen : la valeur par défaut est « true ». Permet l'établissement de liaison VNC standard. Cette option doit être utilisée lorsque le point de terminaison a recours à une authentification VNC standard. Définissez-la sur « false » si vous vous connectez à un proxy qui utilise authd pour effectuer l'authentification. Cette option n'établit pas de liaison VNC.

sendProperMouseWheelDeltas
Booléen : la valeur par défaut est « false ». Les précédentes versions de la bibliothèque normalisent les écarts des événements liés à la molette de la souris en l'une des trois valeurs suivantes : [-1, 0, 1]. Lorsque cette option est définie sur *true*, les véritables écarts par rapport au navigateur sont envoyés au serveur.

reverseScrollY
Booléen : la valeur par défaut est « false ». Si cette option est définie sur *true*, la valeur opposée de la molette de la souris est envoyée à la machine virtuelle connectée.

retryConnectionInterval

Entier : la valeur par défaut est « -1 ». Intervalle (en millisecondes) qui s'écoule avant qu'une nouvelle tentative de connexion ne soit lancée lorsque la première tentative de connexion entre le client Web et le serveur a échoué. Une valeur inférieure à 0 indique qu'il n'y aura « aucune autre tentative ».

ignoredRawKeyCodes

Groupe : la valeur par défaut est vide. Tous ces codes de touche sont ignorés et ne sont pas envoyés au serveur.

fixANSIEquivalentKeys

Booléen : la valeur par défaut est « false ». Permet de corriger tous les codes de touche des dispositions de clavier américain non-ANSI afin qu'ils correspondent aux codes de touche équivalents des dispositions de clavier américain ANSI. Cette option tente de corriger toutes les touches enfoncées lorsque la disposition du clavier international du client comporte une touche qui figure également sur le clavier américain ANSI, mais à un autre emplacement ou qu'elle ne correspond pas à l'état Maj ou non Maj d'un clavier américain ANSI.

keyboardLayoutId

Chaîne : la valeur par défaut est « en-US » (Anglais (États-Unis)). Cette option fournit à l'invité des configurations de clavier de langues différentes. Les utilisateurs doivent déterminer la disposition de leur clavier, et conserver la même disposition pour les claviers des postes de travail local et distant. Dans cette version de HTML Console SDK, VMware ne prend pas en charge les périphériques mobiles. Deux types de code sont envoyés au SE invité. Dans le cadre d'un mappage international, seul vScancode est pris en charge.

Internet Explorer/Firefox/Chrome sont pris en charge sur Windows et Chrome/Safari sur Mac.
Ajoutez une liste de sélection au format html ou utilisez une autre méthode pour permettre aux utilisateurs de choisir la langue à utiliser. Dans le fichier js, utilisez l'API setOption comme ci-dessous :

```
<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>
        <option value="fr-FR">French</option>
        <option value="fr-CH">Swiss-French</option>
        <option value="de-CH">Swiss-German</option>
</select>
```

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

VCDProxyHandshakeVmxPath

Chaîne : la valeur par défaut est « null ». Appliquez le protocole VNC lors de la connexion. vncProtocol répondra à une demande de connexion pour un chemin VMX avec l'option VCDProxyHandshakeVmxPath fournie lors de la connexion à **vCloud Director**.

enableUint8Utf8

Booléen : la valeur par défaut est « false ». Si le projet n'offre pas de prise en charge efficace du protocole binaire, l'option enableUint8Utf8 est requise pour activer l'ancien protocole uint8utf8.

Gestion des événements WMKS

Liste des événements WMKS

WebMKS génère des événements lorsque l'état de la machine virtuelle actuellement connectée change, ou en réponse aux messages émis par celle-ci. Tous les événements WMKS sont répertoriés dans WMKS.CONST.Events.

Tous les gestionnaires d'événements disposent des deux paramètres suivants : **event** et **data**. Le paramètre **event** est un événement jQuery, et tous les paramètres spéciaux des événements sont stockés dans **data**.

connectionstatechange

L'événement connectionstatechange est généré en réponse au changement de l'état de la connexion, de « disconnected » à « connecting » ou de « connecting » à « connected ».

Les paramètres figurant dans **data** sont les suivants :

- state : peut prendre n'importe quelle valeur dans WMKS.CONST. ConnectionState peut prendre les valeurs suivantes :
  1. « connecting »
  2. « connected »
  3. « disconnected »
- Si le paramètre state a pour valeur WMKS.CONST. ConnectionState. CONNECTING, le paramètre data comprend deux autres paramètres supplémentaires, à savoir : ***vvc*** et ***vvcSession***.
- Si le paramètre state a pour valeur WMKS.CONST. ConnectionState. DISCONNECTED, le paramètre data comprend deux autres paramètres supplémentaires, ***reason*** et ***code***.

screensizechange

L'événement screensizechange est généré en réponse aux modifications apportées à la taille de l'écran de la machine virtuelle actuellement connectée.
Les paramètres figurant dans data sont les suivants :

- width : largeur (en pixels) de la machine virtuelle actuellement connectée.
- height : hauteur (en pixels) de la machine virtuelle actuellement connectée.

fullscreenchange

L'événement fullscreenchange est généré lorsque la console WMKS passe en mode plein écran ou le quitte.
Les paramètres figurant dans data sont les suivants :

- isFullScreen : booléen, la valeur « true » indique que le mode plein écran est activé, la valeur « false » indique que le mode plein écran est désactivé.

error

L'événement error est généré lorsqu'une erreur survient, comme un échec d'authentification, une erreur WebSocket ou de protocole.

Les paramètres figurant dans data sont les suivants :

- errorType : peut prendre n'importe quelle valeur dans WMKS.CONST.Events.ERROR :
  1. AUTHENTICATION_FAILED : « authenticationfailed »,
  2. WEBSOCKET_ERROR : « websocketerror »
  3. PROTOCOL_ERROR : « protocolerror »

keyboardledschanged

L'événement keyboardledschanged est généré lorsque l'état de verrouillage des voyants du clavier de la machine virtuelle distante change.

Les paramètres figurant dans data sont les suivants :

- La valeur du paramètre data est un entier : 1 correspond à la touche Défilement, 2 correspond à la touche Verr num, 4 correspond à la touche Verrouillage des majuscules.

heartbeat

L'événement heartbeat est généré lors de la réception d'un message de pulsation côté serveur.

- La valeur du paramètre data est un entier qui définit l'intervalle entre deux pulsations.

audio

L'événement audio est généré lors de la réception d'un message audio émis par le serveur.
Les paramètres figurant dans data sont les suivants :

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

copy

L'événement copy est généré lorsque le serveur envoie un événement de texte coupé.

- La valeur du paramètre data est la chaîne copiée.

toggle

L'événement toggle est généré lorsque le clavier, le clavier étendu ou le pavé tactile est affiché ou masqué.
Les paramètres figurant dans data sont les suivants :

- type : peut prendre les valeurs « KEYBOARD », « EXTENDED_KEYPAD », « TRACKPAD »
- visibility : Booléen, la valeur « true » correspond à l'option « afficher », la valeur « false » à l'option « masquer ».

Configuration de gestionnaires pour les événements WMKS

Dans HTML Console SDK, utilisez les méthodes register() et unregister() pour ajouter et supprimer des gestionnaires pour les événements WMKS.

register()

Cette méthode permet d'enregistrer un gestionnaire d'événements pour le WMKS.

Méthode	register(eventName, eventHandler)
Paramètre1	eventName(constante, toute valeur de WebMKS.Events)
Paramètre2	eventHandler(fonction de rappel javascript)
Valeur renvoyée	Aucune
Exemple d'appel	var connectionStateHandler = function(event, data){}; wmks.register(WebMKS.Events.CONNECTION_STATE_CHANGE, connectionStateHandler);

unregister()

Cette méthode permet d'annuler l'enregistrement du gestionnaire d'événements pour le WMKS.

Si elle ne contient aucun paramètre, tous les gestionnaires d'événements sont supprimés.

Si elle ne contient qu'un seul paramètre, tous les gestionnaires d'événements comportant l'événement dont le nom est spécifié sont supprimés.

Méthode	unregister(eventName, eventHandler)
Paramètre1	eventName(constante, toute valeur de WebMKS.Events)
Paramètre2	eventHandler(fonction de rappel javascript)
Valeur renvoyée	Aucune
Exemple d'appel	wmks.unregister(WebMKS.Events.CONNECTION_STATE_CHANGE, connectionStateHandler);

Méthodes de l'objet HTML Console SDK

Après avoir appelé la méthode createWMKS(), vous obtenez un objet central WMKS capable d'utiliser l'API WMKS pour établir une connexion avec une machine virtuelle et effectuer des opérations. Dans cet exemple, l'objet WMKS est nommé « wmks ».

API générales

Les méthodes d'API à usage général fournissent des informations sur WMKS. Ces méthodes peuvent être appelées à tout moment, y compris avant la connexion à la machine virtuelle cible.

getVersion()

Permet de récupérer le numéro de version actuel de HTML ConsoleSDK.

Méthode	getVersion()
Paramètres	Aucun
Valeur renvoyée	Chaîne. Contient le numéro de version complet de HTML Console SDK.
Exemple d'appel	var version = wmks.getVersion();

getConnectionState()

Permet de récupérer l'état actuel de la connexion.

Méthode	getConnectionState()
Paramètres	Aucun
Valeur renvoyée	Constante. Toute valeur dans WMKS. CONST.ConnectionState.
Exemple d'appel	var state = wmks.getConnectionState();

API relatives au cycle de vie

connect()

Permet de connecter le WMKS à une machine virtuelle distante via l'URL WebSocket, et de configurer l'IU.

Méthode	connect()
Paramètre	URL WebSocket, chaîne au format : <ws | wss> :// <host:port>/ <path> /? <authentication info>
Valeur renvoyée	aucune
Exemple d'appel	wmks.connect("ws://localhost:8080");

disconnect()

Permet de se déconnecter de la machine virtuelle distante et de détruire l'IU.

Méthode	disconnect()
Paramètre	Aucun
Valeur renvoyée	Aucune
Exemple d'appel	wmks.disconnect();

destroy()

Permet de mettre fin à la connexion (le cas échéant) à la machine virtuelle et de supprimer le widget de l'élément associé. Les utilisateurs doivent appeler cette méthode avant de supprimer l'élément du DOM.

Méthode	disconnect()
Paramètre	Aucun
Valeur renvoyée	Aucune
Exemple d'appel	wmks.destroy();

API relatives à l'affichage

setRemoteScreenSize()

Envoie une demande afin de définir la résolution de l'écran de la machine virtuelle actuellement connectée. Si les paramètres de largeur (width) et de hauteur (height) appliqués sont supérieurs à ceux de la taille allouée au widget WMKS, le dimensionnement sera normalisé.

Remarque : cette méthode ne fonctionne correctement que lorsque l'option changeResolution est définie sur « *true* ».

Méthode	setRemoteScreenSize(width,height)
Paramètre1	width(int) représente la largeur souhaitée de la machine virtuelle connectée, en pixels
Paramètre2	height(int) représente la hauteur souhaitée de la machine virtuelle connectée, en pixels
Valeur renvoyée	Aucune
Exemple d'appel	wmks.setRemoteScreeSize(800,600);

getRemoteScreenSize()

Permet de récupérer la largeur et la hauteur de l'écran de la machine virtuelle actuellement connectée, en pixels.

Méthode	getRemoteScreenSize()
Paramètre	Aucun
Valeur renvoyée	Objet au format {width:, height:}
Exemple d'appel	var size = wmks.getRemoteScreenSize();

updateScreen()

Permet de modifier la résolution ou de redimensionner l'écran distant afin qu'il corresponde à la taille actuellement allouée.

Le comportement de la méthode updateScreen dépend des options changeResolution, rescale et position :

1) Si l'option changeResolution est définie sur « true », la demande de modification de la résolution est envoyée à la machine virtuelle connectée, la résolution demandée (largeur et hauteur) étant identique à la taille allouée du conteneur.

2) Vérifiez l'option rescale : si elle est définie sur « true », l'écran distant est redimensionné afin de s'adapter à la taille allouée du conteneur.

3) Vérifiez l'option de position : si la taille de l'écran distant diffère de la taille allouée du conteneur, l'écran distant sera alors placé au centre ou en haut à gauche du conteneur en fonction de la valeur de cette option.

Méthode	updateScreen()
Paramètre	Aucun
Valeur renvoyée	Aucune
Exemple d'appel	wvmrc.updateScreen();

API relatives au plein écran

canFullScreen()

Indique si le mode plein écran est activé sur ce navigateur. Le mode plein écran est désactivé sur Safari car, pour des raisons de sécurité, la saisie clavier n'est pas prise en charge en mode plein écran.

Méthode	canFullScreen()
Paramètre	Aucun
Valeur renvoyée	Booléenne. La valeur « true » indique que le mode plein écran peut être activé, la valeur « false » qu'il ne peut pas l'être.
Exemple d'appel	wmks.canFullScreen();

isFullScreen()

Indique si le navigateur est en mode plein écran.

Méthode	isFullScreen()
Paramètre	Aucun
Valeur renvoyée	Booléenne. La valeur « true » indique que le mode plein écran est activé, la valeur « false » qu'il est désactivé.
Exemple d'appel	wmks.isFullScreen();

enterFullScreen()

Force le navigateur à passer au mode plein écran, si celui-ci est pris en charge. En mode plein écran, seul l'écran distant est affiché.

Méthode	enterFullScreen()
Paramètre	Aucun
Valeur renvoyée	Aucune
Exemple d'appel	wmks.enterFullScreen();

exitFullScreen()

Force le navigateur à quitter le mode plein écran.

Méthode	exitFullScreen()
Paramètre	Aucun
Valeur renvoyée	Aucune
Exemple d'appel	wmks.exitFullScreen();

API relatives à la saisie

sendInputString()

Envoie au serveur la saisie clavier sous forme de chaîne.

Méthode	sendInputString(string)
Paramètres	Chaîne
Valeur renvoyée	Aucune.
Exemple d'appel	wmks.sendInputString("test");

sendKeyCodes()

Envoie une série de codes de touche spéciaux à la machine virtuelle. Un groupe de codes de touche spéciaux est défini et des touches fléchées vers le bas sont envoyées pour chacun d'entre eux, dans l'ordre indiqué. Des touches fléchées vers le haut sont ensuite envoyées pour chacun d'entre eux, dans le sens inverse. Cette méthode peut être utilisée pour envoyer des combinaisons de touches telles que Ctrl+Alt+Suppr.

Méthode	sendKeyCodes ()
Paramètres	Groupe. Chaque élément du groupe peut être un code de touche ou un Unicode. Code de touche pour les touches non imprimables, Unicode pour les caractères imprimables. Si vous optez pour Unicode, utilisez des valeurs négatives. Par exemple, -118 indique la touche « v »
Valeur renvoyée	Aucune.
Exemple d'appel	wmks.sendKeyCodes([17,18,46]) ; // Ctrl+Alt+Suppr

sendCAD()

Envoie la séquence de touches Ctrl+Alt+Suppr à la machine virtuelle actuellement connectée.

Méthode	sendCAD()
Paramètres	Aucun.
Valeur renvoyée	Aucune.
Exemple d'appel	wmks.sendCAD();

API relatives au périphérique mobile

enableInputDevice()

Permet d'activer le dispositif de saisie (clavier, clavier étendu, pavé tactile) sur le périphérique mobile, et de l'initialiser pour l'utiliser.

Méthode	enableInputDevice (deviceType)
Paramètres	deviceType :(constante) Toute valeur dans WMKS.CONST.InputDeviceType.
Valeur renvoyée	Aucune.
Exemple d'appel	wmks.enableInputDevice(WMKS. CONST .InputDeviceType. KEYBOARD) ;

disableInputDevice()

Permet de désactiver le dispositif de saisie (clavier, clavier étendu, pavé tactile) sur le périphérique mobile, et de le détruire.

Méthode	disbleInputDevice (deviceType)
Paramètres	deviceType :(constante) Toute valeur dans WMKS.CONST.InputDeviceType.
Valeur renvoyée	Aucune.
Exemple d'appel	wmks.disableInputDevice(WMKS. CONST .InputDeviceType. KEYBOARD) ;

showKeyboard()

Permet d'afficher le clavier sur un périphérique mobile.

Méthode	showKeyboard()
Paramètres	Aucun
Valeur renvoyée	Aucune.
Exemple d'appel	wmks.showKeyboard();

hideKeyboard()

Permet de masquer le clavier sur un périphérique mobile.

Méthode	hideKeyboard()
Paramètres	Aucun.
Valeur renvoyée	Aucune.
Exemple d'appel	wmks.hideKeyboard();

toggleExtendedKeypad()

Permet d'afficher/de masquer le clavier étendu sur le périphérique mobile selon la visibilité actuelle.

Méthode	toggleExtendedKeypad()
Paramètres	Une carte peut inclure la méthode minToggleTime(ms) comme suit : {minToggleTime: 50} si l'utilisateur tente trop fréquemment d'appeler cette méthode de basculement, une durée inférieure à minToggleTime empêchera alors l'exécution de la méthode.
Valeur renvoyée	Aucune.
Exemple d'appel	wmks.toggleExtendedKeypad();

toggleTrackpad()

Permet d'afficher/de masquer le pavé tactile sur le périphérique mobile selon la visibilité actuelle.

Méthode	toggleTrackpad()
Paramètres	Une carte peut inclure la méthode minToggleTime(ms) comme suit : {minToggleTime: 50} si l'utilisateur tente trop fréquemment d'appeler cette méthode de basculement, une durée inférieure à minToggleTime empêchera alors l'exécution de la méthode.
Valeur renvoyée	Aucune.
Exemple d'appel	wmks.toggleTrackpad();

toggleRelativePad ()

Permet d'afficher/de masquer le pavé tactile de souris relative sur l'écran. Envoie un événement de position de souris relative plutôt qu'un événement de position de souris absolue, après l'ouverture de ce pavé relatif. Cette méthode peut être utilisée lorsque le SE invité distant ne dispose pas de VMware Tools. Ajoutez un bouton toggleRelativeMouse dans l'IU, liez-le à l'API toggleRelativePad(). Une fois ce bouton activé, le pavé tactile relatif s'affiche et envoie un événement de position de souris relative à l'invité distant. Cela est utile lorsque le SE invité ne dispose pas de VMware Tools et n'est pas en mesure d'accepter un événement de position de souris relative. En effet, s'il n'y a aucune image de curseur provenant du serveur, une image de curseur locale doit être affichée. Ainsi, des fichiers wmks-all.css doivent être ajoutés au produit, sous les dossiers css et img. Ces fichiers se trouvent tous sous le dossier wmkssdk.

Méthode	toggleRelativePad ()
Paramètres	Une carte peut inclure la méthode minToggleTime(ms) comme suit : {minToggleTime: 50} si l'utilisateur tente trop fréquemment d'appeler cette méthode de basculement, une durée inférieure à minToggleTime empêchera alors l'exécution de la méthode.
Valeur renvoyée	Aucune.
Exemple d'appel	wmks. toggleRelativePad ();

API relatives aux options

setOption()

Permet de modifier la valeur de l'option. Seules les options répertoriées ci-dessous peuvent utiliser cette méthode :

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

Méthode	setOption(optionName, optionValue)
Paramètre1	optionName(chaîne du nom de l'option)
Paramètre2	optionValue
Valeur renvoyée	Aucune
Exemple d'appel	wmks.setOption("changeResolution",false);

Annexe

Constantes utilisées dans WMKS

- **Position** : WMKS affiche l'écran distant de la machine virtuelle en proportion égale. Par conséquent, après le redimensionnement, la taille de l'écran distant peut être toujours différente de la taille du conteneur. La constante ***Position*** fournit deux options possibles pour placer l'écran à la position définie dans le conteneur.

- **ConnectionState** : il existe 3 états de connexion possibles lorsque vous tentez de vous connecter à la machine virtuelle distante.

- **Events** : tous les noms d'événement pouvant être déclenchés via WMKS sont répertoriés ici.

- **ErrorType** : types d'erreur possibles dans le cycle de vie de WMKS.

- **InputDeviceType** : HTML Console SDK prend en charge l'affichage des consoles de machine virtuelle sur les périphériques mobiles, et ce champ répertorie les dispositifs de saisie possibles.

(Reportez-vous à la page suivante pour obtenir la liste des constantes.)

```
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
}
```