Index d'API pour les applications Windows de bureau

Article • 12/06/2023

Cet article fournit des liens vers la documentation de référence pour les API qui peuvent être utilisées dans les applications Windows de bureau.

Win32 (API Windows)

L'API Win32 (également appelée API Windows) est la plateforme native pour les applications Windows. Cette API est idéale pour les applications de bureau qui nécessitent un accès direct aux fonctionnalités et au matériel système. L'API Windows peut être utilisée dans toutes les applications de bureau, et les mêmes fonctions sont généralement prises en charge sur Windows 32 bits et 64 bits.

- Informations de référence sur l'API Win32 par fonctionnalité
- Référence de l'API Win32 par en-tête
- API Win32 et COM pour les applications UWP
- Bibliothèques parapluies Windows
- Ensembles d'API Windows

Windows Runtime (WinRT)

WinRT est la plateforme de pointe pour les applications et les jeux Windows 10, y compris les applications de bureau. L'API WinRT est adaptée aux applications C++ natives et de bureau managées qui nécessitent une interface utilisateur sophistiquée, une personnalisation des styles et des scénarios gourmands en graphismes.

- Informations de référence sur les API WinRT
- API WinRT pouvant être appelées à partir d'une application de bureau

.NET

Les bibliothèques de classes .NET permettent d'accéder aux fonctionnalités du système Windows et de l'interface utilisateur pour les applications de bureau managées, notamment les applications WPF et Windows Forms.

- API .NET

Commentaires

Cette page a-t-elle été utile ? Yes No

Obtenir de l’aide sur Microsoft Q&A

Index des API Windows

Article • 16/03/2023

Voici une liste du contenu de référence pour l'interface de programmation d'applications (API) Windows pour les applications de bureau et de serveur.

À l'aide de l'API Windows, vous pouvez développer des applications qui s'exécutent correctement sur toutes les versions de Windows tout en tirant parti des fonctionnalités propres à chaque version. (Notez que cela s'appelait auparavant API Win32. Le nom API Windows reflète plus précisément ses racines dans Windows 16 bits et sa prise en charge sur Windows 64 bits.)

Interface utilisateur

L'API d'interface utilisateur Windows crée et utilise des fenêtres pour afficher la sortie, inviter l'utilisateur à entrer et effectuer les autres tâches qui prennent en charge l'interaction avec l'utilisateur. La plupart des applications créent au moins une fenêtre.

- Accessibilité
- Gestionnaire de fenêtres de bureau (DWM)
- Services de globalisation
- Haute résolution
- Interface utilisateur multilingue (MUI)
- National Language Support (NLS)
- Éléments de l'interface utilisateur :
  - Boutons
  - Carets
  - Zones de liste déroulante
  - Boîtes de dialogue communes
  - Contrôles communs
  - Curseurs
  - Boîtes de dialogue
  - Modifier les contrôles
  - Contrôles d'en-tête
  - Icônes
  - Raccourcis clavier

    - Zones de liste
    - Contrôles d’affichage de liste
    - Menus
    - Barres de progression
    - Feuilles de propriétés
    - Contrôles d’édition enrichis
    - Barres de défilement
    - Contrôles statiques
    - Chaînes
    - Barres d'outils
    - Info-bulles
    - Barres de suivi
    - Contrôles tree-view
- Gestionnaire d’animations Windows
- Infrastructure du ruban Windows

Environnement Windows (Shell)

- Système de propriétés Windows
- Windows Shell
- Recherche Windows
- Consoles

Entrée utilisateur et messagerie

- Interaction utilisateur
    - Manipulation directe
    - Entrée manuscrite
    - Configuration des commentaires d’entrée
    - Contexte d’interaction
    - Pile d’entrée de périphérique pointeur
    - Messages et notifications d’entrée de pointeur
    - Entrée du contrôleur radial
    - Text Services Framework
    - Test d’accès tactile
    - Injection tactile
- Interaction utilisateur héritée
    - Entrée tactile

    - Entrées du clavier
    - Entrées de la souris
    - Entrée brute

- Windows et messages :
    - Messages et files d'attente de messages
    - Windows
    - classes de fenêtre
    - Procédures de fenêtre
    - Minuteurs
    - Propriétés de la fenêtre
    - Hooks

Accès aux données et stockage

- Service de transfert intelligent en arrière-plan (BITS)

- Sauvegarde de données
    - Sauvegarde
    - Déduplication des données
    - Cliché instantané des volumes
    - Sauvegarde Windows Server

- Échange de données :
    - Presse-papiers
    - Dynamic Data Exchange (DDE)
    - Gestion dynamique de l'échange de données (DDEML)

- Gestion de répertoires

- Gestion des disques

- Système de fichiers DFS

- Réplication du système de fichiers DFS

- Moteur de stockage extensible

- Fichiers et E/S (système de fichiers local)

- API de bibliothèque de découverte iSCSI

- Fichiers hors connexion

- Packaging

- Compression différentielle à distance
- NTFS transactionnel
- Gestion des volumes
- Disque dur virtuel (VHD)
- Gestion de stockage Windows
- Windows Data Access Components
  - Microsoft ODBC (Open Database Connectivity)
  - Microsoft OLE DB
  - Microsoft ActiveX Data Objects (ADO)

Diagnostics

L'API Diagnostics vous permet de résoudre les problèmes d'application ou de système et de surveiller les performances.

- Récupération et redémarrage d'application
- Débogage
- Gestion des erreurs
- Journalisation des événements
- Suivi d'événements
- Profilage de compteur matériel (HCP)
- Network Diagnostics Framework (NDF)
- Moniteur réseau
- Compteurs de performance
- Journaux et alertes de performances (PLA)
- Traiter la capture instantanée
- État du processus (PSAPI)
- Gestion structurée des exceptions
- Moniteur système
- Wait Chain Traversal
- Rapport d'erreurs Windows
- Journal des événements Windows
- Plateforme de résolution des problèmes Windows

Graphisme et multimédia

Les API Graphics, multimédia, audio et vidéo permettent aux applications d'incorporer du texte mis en forme, des graphiques, de l'audio et de la vidéo.

- Audio de base
- Direct2D
- DirectComposition
- Directshow
- DirectWrite
- DirectX
- Graphics Device Interface (GDI)
- GDI+
- Streaming multimédia
- Microsoft Media Foundation
- Microsoft TV Technologies
- Opengl
- Configuration du moniteur
- Moniteurs d'affichage multiples
- Acquisition d'images
- Système de couleurs Windows
- Composant Imagerie Windows (WIC)
- Codec et DSP audio et vidéo Windows Media
- Windows Media Center
- Format Windows Media
- Services de partage de bibliothèque Windows Media
- Lecteur Windows Media
- Windows Media Services
- Windows Movie Maker
- Windows Multimedia

Appareils

- AllJoyn
- Ressources de communication
- Accès aux appareils
- Gestion des appareils
- Stockage étendu
- Découverte de fonctions
- Mastering d'images
- Lieu
- Base de données d'association PnP-X
- Impression

    - Spouleur d’impression
    - Imprimer le package de document
    - Spécification du schéma d’impression
    - Imprimer le ticket
    - XPS Print
- Détecteurs
- Sens (System Event Notification Service)
- Aide sur l’outil
- UPnP
- Services web sur les appareils
- Acquisition d’image Windows (WIA)
- Windows Media Gestionnaire de périphériques
- Appareils portables Windows

Services système

Les API Des services système permettent aux applications d’accéder aux ressources de l’ordinateur et aux fonctionnalités du système d’exploitation sous-jacent, telles que la mémoire, les systèmes de fichiers, les appareils, les processus et les threads.

- COM
- COM+
- Compression API
- Distributed Transaction Coordinator (DTC)
- Bibliothèques de liens dynamiques (DLL)
- API d’aide
- Communications interprocessus :
    - Mailslots
    - Canaux
- Gestionnaire de transactions du noyau (KTM)
- Gestion de la mémoire
- Enregistreur d’opérations
- Gestion de l’alimentation
- Services Bureau à distance
- Processus
- Services
- Synchronisation
- Threads
- Partage du bureau Windows
- Informations système Windows
    - Handle et objets

    - Registre
    - Time
    - Fournisseur de temps

Sécurité et identité

Les API de sécurité et d'identité activent l'authentification par mot de passe lors de l'ouverture de session, la protection discrétionnaire pour tous les objets système pouvant être partagés, le contrôle d'accès privilégié, la gestion des droits et l'audit de sécurité.

- Authentification
- Autorisation
- Inscription de certificat
- Cryptographie
- Chiffrement nouvelle génération (CNG)
- Services d'annuaire
    - Services de domaine Active Directory
    - Active Directory Service Interfaces (ADSI)
- Protocole EAP (Extensible Authentication Protocol)
- Hôte de protocole d'authentification extensible (EAPHost)
- Gestion des mots de passe MS-CHAP
- Protection d'accès réseau (NAP)
- Network Policy Server Extensions (NPS)
- Contrôle parental
- Fournisseurs WMI de sécurité
- Services de base TPM (TBS)
- Windows Biometric Framework

Installation d'application et maintenance

- Explorateur des jeux
- Assemblys côte à côte
- API d'empaquetage, de déploiement et de requête
- Licence de développeur
- Redémarrer le Gestionnaire
- Windows Installer

Administration et gestion système

Les interfaces d'administration système vous permettent d'installer, de configurer et de traiter des applications ou des systèmes.

- Fournisseur WMI de données de configuration de démarrage
- Clusters de basculement
- Gestionnaire de ressources du serveur de fichiers
- Stratégie de groupe
- Microsoft Management Console (MMC) 2.0
- NetShell
- Infrastructure de gestion des paramètres
- Journalisation de l'inventaire logiciel
- Gestion des licences des logiciels
- Redémarrer le Gestionnaire
- Infrastructure de gestion des paramètres
- Restauration du système
- Arrêt du système
- Planificateur de tâches
- Journalisation des accès utilisateur
- Windows Virtual PC
- Microsoft Virtual Server
- Fournisseur d'équilibrage de charge réseau
- Windows Defender WMI v2
- Services de déploiement Windows
- Windows Genuine Advantage
- Infrastructure de gestion Windows
- Windows Management Instrumentation (WMI)
- Gestion à distance de Windows
- Protection des ressources Windows
- Windows Server Update Services
- Outil d'évaluation du système Windows
- Agent Windows Update

Mise en réseau et Internet

Les API de mise en réseau permettent la communication entre les applications sur un réseau. Vous pouvez également créer et gérer l'accès aux ressources partagées, telles que les répertoires et les imprimantes réseau.

- DNS (Domain Name System)
- Protocole DHCP (Dynamic Host Configuration Protocol)
- Service de télécopie

- Assistant Connexion Internet
- Serveur HTTP
- Partage de connexion Internet et pare-feu
- Assistance IP
- Pare-feu de connexion Internet IPv6
- Base d'informations de gestion
- Message Queuing (MSMQ)
- Protocole MADCAP (Multicast Address Dynamic Client Allocation Protocol)
- Traduction d'adresses réseau (NAT)
- Network List Manager (NLM)
- Gestion du réseau
- Gestion du partage réseau
- Pair à pair
- Qualité de service (QOS)
- Appel de procédure distante
- Service de routage et d'accès à distance (RAS)
- Simple Network Management Protocol (SNMP)
- Gestion SMB
- Interfaces de programmation d'applications de téléphonie (TAPI)
- WebDAV
- Composant protocole WebSocket
- Mise en réseau sans fil :
  - Bluetooth
  - Irda
  - Haut débit mobile
  - Wifi natif
  - Windows Connect Now
  - Gestionnaire de connexions Windows
- Plateforme de filtrage Windows
- Pare-feu Windows avec fonctions avancées de sécurité
- Services HTTP Windows (WinHTTP)
- Windows Internet (WinINet)
- Windows Networking (WNet)
- Virtualisation de réseau Windows
- Plateforme RSS Windows
- Windows Sockets (Winsock)
- Windows Web Services
- Requête étendue HTTP XML

API déconseillées ou héritées

Les technologies et API suivantes sont obsolètes ou qui ont été remplacées ou dépréciées à partir des systèmes d'exploitation client et serveur Windows.

- Directmusic
- Directsound
- Le Kit de développement logiciel (SDK) Microsoft UDDI est désormais inclus dans Microsoft BizTalk Server.
- Network Dynamic Data Exchange (DDE)
- Service d'installation à distance : utilisez plutôt les services de déploiement Windows .
- Service de disque virtuel (VDS) : utilisez plutôt Gestion du stockage Windows .
- Services Terminal Server : utilisez les services Bureau à distance.
- Gestionnaire de droits Windows Media
- Windows Messaging (MAPI) : utilisez Office MAPI à la place.
- Plateforme de gadgets Windows : créez des applications UWP à la place.
- Windows Sidebar : créez des applications UWP à la place.
- Windows SideShow : aucun remplacement.
- Effets bitmap WPF

Ensembles d'API Windows

Article • 01/06/2023

> **ⓘ Important**
>
> Les informations contenues dans cette rubrique s'appliquent à toutes les versions de Windows 10 et versions ultérieures. Nous ferons référence à ces versions ici en tant que « Windows », en appelant toutes les exceptions si nécessaire.

Toutes les versions de Windows partagent une base commune de composants de système d'exploitation appelée système *d'exploitation principal* (dans certains contextes, cette base commune est également appelée *OneCore*). Dans les principaux composants du système d'exploitation, les API Win32 sont organisées en groupes fonctionnels appelés *ensembles d'API*.

L'objectif d'un ensemble d'API est de fournir une séparation architecturale de la DLL hôte dans laquelle une API Win32 donnée est implémentée et du contrat fonctionnel auquel appartient l'API. Le découplage que fournissent les ensembles d'API entre l'implémentation et les contrats offre de nombreux avantages en matière d'ingénierie pour les développeurs. En particulier, l'utilisation d'ensembles d'API dans votre code peut améliorer la compatibilité avec les appareils Windows.

Les ensembles d'API répondent spécifiquement aux scénarios suivants :

- Bien que l'étendue complète de l'API Win32 soit prise en charge sur les PC, seul un sous-ensemble de l'API Win32 est disponible sur d'autres appareils Windows tels que HoloLens, Xbox et d'autres appareils. Le nom de l'ensemble d'API fournit un mécanisme de requête pour détecter correctement si une API est disponible sur un appareil donné.
- Certaines implémentations d'API Win32 existent dans des DLL avec des noms différents sur différents appareils Windows. L'utilisation de noms d'ensembles d'API au lieu de noms de DLL lors de la détection de la disponibilité de l'API et du délai de chargement des API fournit un itinéraire correct vers l'implémentation, quel que soit l'endroit où l'API est réellement implémentée.

Pour plus d'informations, consultez Opération de chargeur d'ensemble d'API et Détecter la disponibilité d'un ensemble d'API.

Les ensembles d'API et les dll sont-ils identiques ?

Non : un nom de jeu d'API est un *alias virtuel* pour un fichier physique `.dll` . Il s'agit d'une technique de masquage d'implémentation, dans laquelle vous n'avez pas besoin de savoir exactement quel module héberge les informations.

La technique permet de refactoriser les modules (fractionnement, consolidation, renommage, etc.) sur différentes versions et éditions de Windows. De plus, vos applications sont toujours liées et sont toujours routées vers le code approprié au moment de l'exécution.

Pourquoi les ensembles d'API ont-ils `.dll` leur nom ? La raison en est la façon dont le *chargeur DE DLL* est implémenté. Le chargeur est la partie du système d'exploitation qui charge les DLL et/ou résout les références aux DLL. Et au niveau du serveur frontal, le chargeur exige que toute chaîne passée à LoadLibrary se termine par « .dll ». Mais après ce front-end, le chargeur peut supprimer ce suffixe et interroger la base de données du jeu d'API avec la chaîne obtenue.

LoadLibrary (et retarder le chargement) réussit avec un nom d'ensemble d'API (avec le « .dll » dans celui-ci) ; mais il n'y a pas nécessairement de fichier avec ce nom n'importe où sur le PC.

Liaison de bibliothèques de parapluies

Pour faciliter la restriction de votre code aux API Win32 prises en charge dans le système d'exploitation principal, nous fournissons une série de *bibliothèques parapluie*. Par exemple, une bibliothèque parapluie nommée `OneCore.lib` fournit les exportations pour le sous-ensemble d'API Win32 communes à tous les appareils Windows.

Pour plus d'informations, consultez Bibliothèques parapluies Windows.

Noms des contrats des ensembles d'API

Les ensembles d'API sont identifiés par un nom de contrat fort qui suit ces conventions standard reconnues par le chargeur de bibliothèque.

- Le nom doit commencer par la chaîne **api-** ou **ext-**.
  - Les noms qui commencent par **api** représentent des API qui sont garanties pour exister sur toutes les versions de Windows.

    - Les noms commençant par **ext-** représentent des API qui n'existent peut-être pas sur toutes les versions de Windows.
- Le nom doit se terminer par la séquence **l<n-n-n>><>,<** où **n** se compose de chiffres décimaux.
- Le corps du nom peut être des caractères alphanumériques ou des tirets (-).
- Le nom ne respecte pas la casse.

Voici quelques exemples de noms de contrats d'ensemble d'API :

- **api-ms-win-core-ums-l1-1-0**
- **ext-ms-win-com-ole32-l1-1-5**
- **ext-ms-win-ntuser-window-l1-1-0**
- **ext-ms-win-ntuser-window-l1-1-1**

Vous pouvez utiliser un nom d'ensemble d'API dans le contexte d'une opération de chargeur telle que LoadLibrary ou P/Invoke au lieu d'un nom de module DLL pour garantir un itinéraire correct vers l'implémentation, quel que soit l'endroit où l'API est réellement implémentée sur l'appareil actuel. Toutefois, dans ce cas, vous devez ajouter la chaîne **.dll** à la fin du nom du contrat. Il s'agit d'une exigence du chargeur pour fonctionner correctement et n'est pas considérée comme faisant partie du nom du contrat. Bien que les noms de contrats semblent similaires aux noms de DLL dans ce contexte, ils sont fondamentalement différents des noms de modules DLL et ne font pas directement référence à un fichier sur disque.

À l'exception de l'ajout de la chaîne **.dll** dans les opérations du chargeur, les noms de contrats des ensembles d'API doivent être considérés comme un identificateur immuable qui correspond à une version de contrat spécifique.

Identification des ensembles d'API pour les API Win32

Pour déterminer si une API Win32 particulière appartient à un ensemble d'API, consultez le tableau des exigences dans la documentation de référence de l'API. Si l'API appartient à un ensemble d'API, le tableau des exigences dans l'article répertorie le nom du jeu d'API et la version Windows dans laquelle l'API a été introduite pour la première fois dans l'ensemble d'API. Pour obtenir des exemples d'API qui appartiennent à un ensemble d'API, consultez les articles suivants :

- AllowSetForegroundWindow
- FindWindowsEx
- GetClassFile

Contenu de cette section

- Opération du chargeur de l'ensemble d'API
- Détecter la disponibilité des ensembles d'API
- Bibliothèques parapluies Windows

Commentaires

Cette page a-t-elle été utile ? Yes No

Obtenir de l'aide sur Microsoft Q&A

Opération du chargeur de l'ensemble d'API

Article • 01/06/2023

> **ⓘ Important**
>
> Les informations contenues dans cette rubrique s'appliquent à toutes les versions de Windows 10 et versions ultérieures. Nous ferons référence à ces versions ici en tant que « Windows », en appelant toutes les exceptions si nécessaire.

Les ensembles d'API s'appuient sur la prise en charge du système d'exploitation dans le chargeur de bibliothèque pour introduire efficacement une redirection d'espace de noms de module dans le processus de liaison de bibliothèque. Le nom du contrat du jeu d'API est utilisé par le chargeur de bibliothèque pour effectuer une redirection du runtime de la référence vers un fichier binaire hôte cible qui héberge l'implémentation appropriée du jeu d'API.

Lorsque le chargeur rencontre une dépendance vis-à-vis d'un ensemble d'API au moment de l'exécution, le chargeur consulte les données de configuration dans l'image pour identifier le fichier binaire hôte d'un jeu d'API. Ces données de configuration sont **appelées schéma d'ensemble d'API**. Le schéma est assemblé en tant que propriété du système d'exploitation, et le mappage entre les jeux d'API et les fichiers binaires peut différer selon les fichiers binaires inclus dans un appareil donné. Le schéma permet à une fonction importée dans un seul fichier binaire d'être routée correctement sur différents appareils, même si les noms de module de l'hôte binaire ont été renommés ou complètement refactoris sur différents appareils Windows.

Windows prend en charge deux techniques standard pour consommer et interagir avec les ensembles d'API : **le transfert direct** et le **transfert inverse**.

Transfert direct

Dans cette configuration, le code consommateur importe directement le nom d'un module d'ensemble d'API. Cette importation est résolue en une seule opération et constitue la méthode la plus efficace avec le moins de surcharge. D'un point de vue conceptuel, cette résolution peut pointer vers différents fichiers binaires sur différents appareils Windows, comme illustré dans l'exemple suivant :

Ensemble d'API importé : **api-feature1-l1-1-0.dll**

- PC Windows ->**feature1.dll**
- HoloLens ->**feature1_holo.dll**
- IoT ->**feature1_iot.dll**

Étant donné que les mappages sont conservés dans un référentiel de données de schéma personnalisé, cela signifie qu'un nom d'ensemble d'API qui se termine par **.dll** ne fait pas directement référence à un fichier sur disque. La **partie.dll** du nom de l'ensemble d'API n'est qu'une convention requise par le chargeur. Le nom de l'ensemble d'API ressemble davantage à un alias ou à un nom virtuel pour un fichier DLL physique. Cela rend le nom portable sur toute la gamme d'appareils Windows.

Transfert inverse

Bien que les noms des ensembles d'API fournissent un espace de noms stable pour les modules sur les appareils, il n'est pas toujours pratique de convertir tous les binaires vers ce nouveau système. Par exemple, une application peut avoir été utilisée depuis de nombreuses années et la recompilation des fichiers binaires de l'application peut ne pas être possible. En outre, certaines applications devront peut-être continuer à s'exécuter sur des systèmes créés avant l'introduction de jeux d'API spécifiques.

Pour prendre en charge ce niveau de compatibilité, un système de *redirecteurs* est fourni sur tous les appareils Windows qui couvrent un sous-ensemble de la surface de l'API Win32. Ces redirecteurs utilisent les noms de module qui ont été introduits sur les PC Windows et tirent parti du système d'ensemble d'API pour assurer la compatibilité entre tous les appareils Windows.

L'opération de chargeur se comporte comme suit :

1. Sur un appareil autre qu'un PC Windows, le chargeur se voit présenter une dépendance de nom de module de PC Windows héritée qui n'est pas présente sur l'appareil.
2. Le chargeur localise un redirecteur de jeu d'API pour ce module et le charge en mémoire.
3. Le redirecteur a un mappage pour l'ensemble d'API pour la fonction donnée appelée.
4. Le chargeur trouve le fichier binaire hôte approprié pour l'appareil donné.

D'un point de vue conceptuel, le mappage se présente comme suit :

DLL importée : **feature1.dll**

- PC Windows ->**feature1.dll**
- HoloLens -> **redirecteurfeature1.dll** ->**api-feature1-l1-1-0.dll** ->**feature1_holo.dll**

- IoT -> **redirecteurfeature1.dll** ->**api-feature1-l1-1-0.dll** ->**feature1_iot.dll**

Le résultat final est fonctionnellement identique au transfert direct, mais il l'accomplit d'une manière qui optimise la compatibilité des applications.

> **Notes**
>
> Le transfert inverse fournit une couverture uniquement pour un sous-ensemble de la surface de l'API Win32. Il n'autorise pas les applications qui ciblent les versions de bureau de Windows à s'exécuter sur tous les appareils Windows.

Commentaires

Détecter la disponibilité des ensembles d'API

Article • 01/06/2023

Dans certains cas, un nom de contrat d'ensemble d'API donné peut être intentionnellement mappé à un nom de module vide sur certains appareils Windows. Les raisons de cela varient, mais un exemple courant est qu'une fonctionnalité coûteuse en termes de ressources système peut être supprimée du système d'exploitation Windows lorsqu'elle est configurée pour un appareil à ressources limitées. Cela pose un défi pour les applications de gérer correctement les fonctionnalités facultatives au niveau de l'API.

L'approche traditionnelle pour tester si une API Win32 est disponible consiste à utiliser LoadLibrary ou GetProcAddress. Toutefois, il ne s'agit pas d'un moyen fiable pour tester les ensembles d'API en raison de la prise en charge du transfert inverse dans Windows 10 et versions ultérieures. Lorsque le transfert inverse est appliqué à une API donnée, **LoadLibrary** ou **GetProcAddress** peut se résoudre en pointeur de fonction valide, même dans les cas où l'implémentation interne a été supprimée. Dans ce cas, le pointeur de fonction pointe vers une fonction stub qui retourne simplement une erreur.

Pour détecter ce cas, vous pouvez utiliser la fonction IsApiSetImplemented pour interroger la disponibilité sous-jacente d'une implémentation d'API donnée. Ce test valide que l'appel de cette fonction entraîne l'exécution d'une implémentation fonctionnelle de l'API.

L'exemple de code suivant montre comment utiliser **IsApiSetImplemented** pour déterminer si la fonction WTSEnumerateSessions est disponible sur l'appareil actuel avant de l'appeler.

c++

```
#include <windows.h>
#include <stdio.h>
#include <Wtsapi32.h>

int __cdecl wmain(int /* argc */, PCWSTR /* argv */ [])
{
    PWTS_SESSION_INFO pInfo = {};
    DWORD count = 0;

    if (!IsApiSetImplemented("ext-ms-win-session-wtsapi32-l1-1-0"))
    {
        wprintf(L"IsApiSetImplemented on ext-ms-win-session-wtsapi32-l1-1-0
returns FALSE\n");
```

```
    }
    else
    {
        if (WTSEnumerateSessionsW(WTS_CURRENT_SERVER_HANDLE, 0, 1, &pInfo,
&count))
        {
            wprintf(L"SessionCount = %d\n", count);

            for (ULONG i = 0; i < count; i++)
            {
                PWTS_SESSION_INFO pCurInfo = &pInfo[i];
                wprintf(L"    %s: ID = %d, state = %d\n", pCurInfo-
>pWinStationName,
                    pCurInfo->SessionId, pCurInfo->State);
            }

            WTSFreeMemory(pInfo);
        }
        else
        {
            wprintf(L"WTSEnumerateSessions failure : %x\n", GetLastError());
        }
    }

    return 0;
}
```

Commentaires

Cette page a-t-elle été utile ? Yes No

Obtenir de l'aide sur Microsoft Q&A

Bibliothèques parapluies Windows

Article • 01/06/2023

> **ⓘ Important**
>
> Les informations contenues dans cette rubrique s'appliquent à toutes les versions de Windows 10 et versions ultérieures. Nous ferons référence à ces versions ici en tant que « Windows », en appelant toutes les exceptions si nécessaire.

Une *bibliothèque parapluie* est une bibliothèque de liens statiques unique qui exporte un sous-ensemble d'API Win32. Par exemple, une bibliothèque parapluie nommée **OneCore.lib** fournit les exportations pour le sous-ensemble d'API Win32 communes à tous les appareils Windows.

Les API d'une bibliothèque parapluie peuvent être implémentées sur une plage de modules (où un module est un ensemble d'API ou une DLL). Mais la bibliothèque parapluie extrait ces détails loin de vous, ce qui rend votre application plus portable entre les versions du système d'exploitation. Dans votre application de bureau ou pilote, il vous suffit de lier la bibliothèque parapluie qui contient l'ensemble des API qui vous intéressent, et c'est tout ce que vous avez à faire.

Bibliothèque	Description
OneCore.lib	Fournit les exportations pour le sous-ensemble d'API Win32 communes à tous les appareils Windows 10 et versions ultérieures. Lier `OneCore.lib` (et aucune autre bibliothèque) pour accéder à ces API. Si vous liez `OneCore.lib` et que vous appelez uniquement les API Win32 dans cette bibliothèque, votre application de bureau ou votre pilote se chargera correctement sur tous les appareils Windows 10 et versions ultérieures.
OneCore_apiset.lib	Fournit la même couverture que `OneCore.lib`, mais utilise le transfert direct d'ensemble d'API. La liaison `OneCore_apiset.lib` sera compatible uniquement avec la version de Windows, ou une version ultérieure, pertinente pour la version du KIT de développement logiciel (SDK) que vous ciblez.

Bibliothèque	Description
OneCoreUap.lib	Fournit les exportations pour le sous-ensemble d'API Win32 communes à tous les appareils Windows 10 et ultérieurs qui prennent en charge le Windows Runtime (WinRT). Lier `OneCoreUap.lib` (et aucune autre bibliothèque) pour accéder à ces API. Si vous liez `OneCore.lib`, et que vous appelez uniquement les API Win32 dans cette bibliothèque, votre application de bureau ou votre pilote se chargera correctement sur tous les appareils Windows 10, et les versions ultérieures, qui prennent en charge UWP.
OneCoreUAP_apiset.lib	Fournit la même couverture que `OneCoreUAP.lib`, mais utilise le transfert direct d'ensemble d'API. La liaison `OneCoreUAP_apiset.lib` sera compatible uniquement avec la version de Windows, ou une version ultérieure, pertinente pour la version du KIT de développement logiciel (SDK) que vous ciblez.

Commentaires

API WinRT pouvant être appelées à partir d'une application de bureau

Article • 03/06/2023

La plupart des API Windows Runtime (WinRT) peuvent être utilisées par les applications de bureau (.NET et C++natives). Toutefois, certaines classes WinRT sont conçues pour et ne sont prises en charge que pour une utilisation dans les applications UWP. Cela inclut CoreDispatcher, CoreWindow, ApplicationView et certaines classes associées. D'autres classes WinRT fonctionnent dans les applications de bureau, à l'exception de certains membres.

Pour plus d'informations, consultez API Windows Runtime non prises en charge dans les applications de bureau. Lorsque cela s'applique, cet article suggère d'autres API qui permettent d'obtenir les mêmes fonctionnalités que les API non prises en charge.

Commentaires