CHAPTER 5

デバイス管理関数

この章では、次のデバイス管理関数に関する情報を示します。

asyncDiscoverDevices

構文

```
(UserToken token, String subnet, String subnetMask,String group,DiscoveryAuthInfo
devAuthInfo,Device.TransportMethod[] transportMethods, String[] policyIDs,IDStatusListener
listener)throws RemoteException;
```

説明

この関数は、指定したサブネットで検出されたデバイスのインベントリを生成し、それらのデバイスを指定したデバイス グループに追加します。

Cisco License Manager は現在、エージェント非対応デバイスの検出に Cisco IOS Telnet と Secure Shell（SSH; セキュア シェル）をサポートしているため、検出時間が長くなります。ネットワーク デバイスが同じように設定されている場合は、トランスポート方法（HTTP、HTTPS、Telnet、または SSH）を指定することで、検出を高速化して検出処理時間を短縮できます。null または空の文字列の配列を渡した場合は、サポートされているすべての方法を使用してデバイスが検出されます。

この関数はノンブロッキングであり、呼び出し元に即座に要求 ID を返します。この関数を呼び出している間、クライアント プログラムは IDStatusListener インターフェイスを実装するリスナー オブジェクトを提供します。操作が完了すると、リスナー オブジェクトの onStatus() メソッドが呼び出されます。

入力パラメータ

パラメータ	タイプ	値	説明
token	UserToken、必須	—	ユーザの認証パスを表すトークン。これは、ユーザが login 関数を呼び出して、バックエンド サーバがそのユーザを認証した後に取得されます。
subnet	String、必須	ネットワーク アドレス	自動検出を実行するサブネット。
subnet_mask	String、必須	—	サブネット マスク。
group	String、必須	—	検出されたデバイスを追加するグループ。
dev_auth_info	DiscoveryAuthInfo、必須	—	DiscoveryAuthInfo は、検出されたデバイスと通信してライセンス情報を取得するときに使用されます。対象のネットワークに存在するデバイスに認証が必要でない場合、このパラメータは null に設定できます。 DiscoveryAuthInfo には、ユーザ名とパスワードのペアの配列とイネーブル パスワードの配列が含まれます。Cisco License Manager がデバイスと通信するとき、DiscoveryAuthInfo のインスタンスが一度に 1 つずつ試され、デバイスによって認証されるまでこれが続きます。Telnet を使用した場合、試行回数は 3 回だけです。
listener	IDStatusListener オブジェクト、必須	—	IDStatusListener インターフェイスを実装するオブジェクト。

パラメータ	タイプ	値	説明
transports	Device.TransportMethod の配列、省略可能	HTTP、TELNET、SSH	サブネット内のデバイスが同じデバイス検出方法を使用してセットアップされている場合は、指定された方法だけを使用してデバイス ライセンスがサポートされているかどうかが迅速に確認されるため、デバイス検出処理が高速になります。このパラメータは、使用する接続トランスポートを指定します。指定したトランスポートが入力順に使用されます。 null または長さが 0 の配列を指定した場合は、使用可能なすべての方法を使用してデバイスが検出されます。
Policy_ids	String の配列、必須	—	検出が完了した後に実行されるポリシー ID の配列。

戻り値

この関数は要求 ID を返します。操作が完了すると、onStatus() メソッドの入力パラメータとしてステータスが提供されます。リスナーは、IDStatusItem のリストを含む IDStatus を受信します。各 IDStatusItem には、検出されたデバイスのデバイス ID が含まれます。

エラーと例外

システム エラーによって操作が完了しなかった場合は、RemoteException がスローされます。

asyncPollDeviceLicenseInfo

構文

```
UserToken token, String jobGroup,String[] devIDs, IDStatusListener listener)
throws RemoteException;
```

説明

この関数は、指定したデバイスと通信してそれらのデバイスのライセンス情報を取得し、その情報をインベントリに格納します。

この関数はノンブロッキングであり、呼び出し元に即座に要求 ID を返します。この関数を呼び出している間、クライアント プログラムは IDStatusListener インターフェイスを実装するリスナー オブジェクトを提供します。操作が完了すると、リスナー オブジェクトの onStatus() メソッドが呼び出されます。

入力パラメータ

パラメータ	タイプ	値	説明
token	UserToken、必須	—	ユーザの認証パスを表すトークン。これは、ユーザが login 関数を呼び出して、バックエンド サーバがそのユーザを認証した後に取得されます。
jobGroup	String、 必須		この非同期ジョブが属するグループ。null に設定した場合は、返されたジョブ ID がジョブ グループ名として使用されます。
dev_ids	String の配列、必須	—	デバイス ID の配列。
listener	IDStatusListener オブジェクト、必須	—	IDStatusListener インターフェイスを実装するオブジェクト。

戻り値

この関数は呼び出し元に要求 ID を返します。操作が完了すると、onStatus() メソッドの入力パラメータとしてステータスが提供されます。リスナーは、操作が完了したときに IDStatus を受信します。IDStatus には IDStatusItem のリストが含まれます。各 IDStatusItem には、デバイス ID、個々のポーリング操作のエラー コード、およびエラー メッセージが含まれます。

エラーと例外

システム エラーによって操作が完了しなかった場合は、RemoteException がスローされます。IDStatusListener の onStatus() メソッドの入力パラメータには、エラー コードとエラー メッセージが含まれます。IDStatus オブジェクトには複数のエラー コードとエラー メッセージが含まれる場合があります。次の例では、onStatus() メソッドでエラー コードとエラー メッセージを取得しています。

```
public void onStatus(IDStatus status) {

    // The general error code of the operation.
    int err_code = status.getErrorCode();

    // The general error message of the operation.
    String err_msg = status.getErrorMessage();

    // A list of status for each individual element in the
    // bulk operation.
    IDStatusItem[] items = status.getIDStatusItems()

    // Iterate through the list to get individual status.
    for (int i = 0; i < items.length(); i++) {

        // Get the individual object ID returned by the operation.
        String id = items[i].getID();

        // Get the individual error code corresponding to
        // the object ID.
        int item_err_code = items[i].getErrorCode();

        // Get the individual error message corresponding to
        // the object ID.
        String item_err_msg = items[i].getErrorMessage();
```

```
        }
    }
```

createDeviceByIPAddr

構文

```
DeviceStatus createDeviceByIPAddr(UserToken token, String ip, String group,
DiscoveryAuthInfo dev_auth_info, Device.TransportMethod[] transports) throws
RemoteException;
```

説明

この関数は、次のパラメータを使用して、インベントリ内にデバイス オブジェクトを作成します。

入力パラメータ

パラメータ	タイプ	値	説明
token	UserToken、必須	—	ユーザの認証パスを表すトークン。これは、ユーザが login 関数を呼び出して、バックエンド サーバがそのユーザを認証した後に取得されます。
ip	String、必須	IP アドレス	IP アドレス。このアドレスを使用してデバイス オブジェクトが作成されます。
group	String、必須	—	デバイス グループの名前。
dev_auth_info	DiscoveryAuthInfo の配列、必須	—	DiscoveryAuthInfo 値の配列は、検出されたデバイスと通信してライセンス情報を取得するときに使用されます。配列内の各項目を使用して、IP 配列に含まれる IP アドレスに接続します。 DeviceAuthentication には、ユーザ名、パスワード、およびイネーブル パスワードが含まれます。
transports	Device.TransportMethod の配列、省略可能	HTTP、HTTPS、TELNET、SSH	このパラメータは、各 IP エントリで使用する接続トランスポート方法を指定します。呼び出し元が各デバイスで使用される接続方法を知っていることが前提となります。

戻り値

この関数は DeviceStatus オブジェクトを返します。次の例では、ステータスに含まれるエラー コード、エラー メッセージ、および戻りオブジェクトを取得しています。

```
DeviceStatus status = createDevicesByIPAddr(…, …);

// The general error code of the operation.
int err_code = status.getErrorCode();

// The general error message of the operation.
String err_msg = status.getErrorMessage();

// A list of status for each individual element in the
```

```
    // bulk operation.
    DeviceStatusItem[] items = status.getDeviceStatusItems()

    // Iterate through the list to get individual status.
for (int i = 0; i < items.length(); i++) {

    // Get the individual object returned by the operation.
    Device device = items[i].getDevice();

    // Get the individual error code corresponding to
    // the object.
        int item_err_code = items[i].getErrorCode();

    // Get the individual error message corresponding to
    // the object.
        String item_err_msg = items[i].getErrorMessage();
    }
```

エラーと例外

システム エラーによって操作が完了しなかった場合は、RemoteException がスローされます。

操作エラーが発生した場合は、エラーに関する情報を含む DeviceStatus オブジェクトが返されます。個々の要素のステータスを調べるには、DeviceStatus に含まれる DeviceStatusItem 配列を反復処理する必要があります。各 DeviceStatusItem には、Device オブジェクト、エラー コード、およびエラー メッセージが含まれます。

createDevicesByUDI

構文

```
DeviceStatus createDevicesByUDI(UserToken token, String[] udis) throws RemoteException;

DeviceStatus createDevicesByUDI(UserToken token, String[] udis, String group) throws
RemoteException;
```

説明

この関数は、指定した Unique Device Identifier（UDI; 固有デバイス識別情報）を使用して、インベントリ内にデバイス オブジェクトを作成します。2 番目の関数形式では、それに加えて、作成されたデバイスを指定したグループに追加します。グループが指定されていない場合は、デフォルト グループにデバイスが追加されます。

入力パラメータ

パラメータ	タイプ	値	説明
token	UserToken、必須	—	ユーザの認証パスを表すトークン。これは、ユーザが login 関数を呼び出して、バックエンド サーバがそのユーザを認証した後に取得されます。
udis	String の配列、必須	x21 ～ x7A の範囲の ASCII 文字を含む最大 64 文字の文字列	UDI の配列。各 UDI を使用してデバイス オブジェクトが作成されます。
group	String、必須	—	デバイス グループの名前。

戻り値

この関数は DeviceStatus オブジェクトを返します。次の例では、ステータスに含まれるエラー コード、エラー メッセージ、および戻りオブジェクトを取得しています。

```
    DeviceStatus status = createDevicesByUDI(…, …);

    // The general error code of the operation.
    int err_code = status.getErrorCode();

    // The general error message of the operation.
    String err_msg = status.getErrorMessage();

    // A list of status for each individual element in the
    // bulk operation.
    DeviceStatusItem[] items = status.getDeviceStatusItems()

    // Iterate through the list to get individual status.
for (int i = 0; i < items.length(); i++) {

    // Get the individual object returned by the operation.
    Device device = items[i].getDevice();

    // Get the individual error code corresponding to
    // the object.
        int item_err_code = items[i].getErrorCode();

    // Get the individual error message corresponding to
    // the object.
        String item_err_msg = items[i].getErrorMessage();
    }
```

エラーと例外

システム エラーによって操作が完了しなかった場合は、RemoteException がスローされます。

操作エラーが発生した場合は、エラーに関する情報を含む DeviceStatus オブジェクトが返されます。個々の要素のステータスを調べるには、DeviceStatus に含まれる DeviceStatusItem 配列を反復処理する必要があります。各 DeviceStatusItem には、Device オブジェクト、エラー コード、およびエラー メッセージが含まれます。

（注） この API は、メンバ デバイスを含むデバイスの作成に使用できます。スイッチ スタックにマスター スイッチとメンバ スイッチを含む API コードの例を次に示します。

```
// Create 3 device objects, where dev_objs[0] is the master switch,
// dev_objs[1] is member switch 1 and dev_objs[2] is member switch 2.

String dev_udi[] = {“MasterSwitch”, “MemberSwitch1”, “MemberSwitch2”};
Device[] dev_objs =
       LicenseManager.createDeviceByUDI(token, dev_ids[]);

// Associate 2 member switches to their master switch.
dev_objs[0].member_device_ids = new String[2];
dev_objs[0].member_device_ids[0] = dev_ids[1];
dev_objs[0].member_device_ids[1] = dev_ids[2];

// Write the changes to the data storage.
LicenseManager.writeDevice(token, dev_ids);
```

deleteDevices

構文

```
IDStatus deleteDevices(UserToken token, String[] dev_ids) throws RemoteException;
```

説明

この関数は、指定したデバイス ID を使用して、インベントリからデバイス オブジェクトを削除します。すべてのデバイスはグループに属しています。ユーザ定義のグループに属していないデバイスは、デフォルト グループに属しています。

入力パラメータ

パラメータ	タイプ	値	説明
token	UserToken、必須	—	ユーザの認証パスを表すトークン。これは、ユーザが login 関数を呼び出して、バックエンド サーバがそのユーザを認証した後に取得されます。
dev_ids	String の配列、必須	—	デバイス ID の配列。

戻り値

この関数は IDStatus オブジェクトを返します。次の例では、ステータスに含まれるエラー コード、エラー メッセージ、および戻りオブジェクトを取得しています。

```
    IDStatus status = deleteDevices(…, …);

    // The general error code of the operation.
    int err_code = status.getErrorCode();

    // The general error message of the operation.
    String err_msg = status.getErrorMessage();

    // A list of status for each individual element in the
    // bulk operation.
    IDStatusItem[] items = status.getIDStatusItems()

    // Iterate through the list to get individual status.
for (int i = 0; i < items.length(); i++) {

    // Get the individual ID returned by the operation.
    String id = items[i].getID();

    // Get the individual error code corresponding to
    // the ID.
        int item_err_code = items[i].getErrorCode();

    // Get the individual error message corresponding to
    // the ID.
        String item_err_msg = items[i].getErrorMessage();
    }
```

エラーと例外

システム エラーによって操作が完了しなかった場合は、RemoteException がスローされます。

入力配列の要素でエラーが発生した場合は、エラーに関する情報を含む Status オブジェクトが返されます。

getActiveRMADevices

構文

```
DeviceStatus getActiveRMADevices(UserToken token) throws RemoteException;
```

説明

この関数は、Cisco License Manager サーバ ストレージに格納されている、Return Material Authorization（RMA）としてマークされた接続済みデバイスのリストを返します。

入力パラメータ

パラメータ	タイプ	値	説明
token	UserToken、必須	—	ユーザの認証パスを表すトークン。これは、ユーザが login 関数を呼び出して、バックエンド サーバがそのユーザを認証した後に取得されます。

戻り値

この関数は DeviceStatus を返します。この DeviceStatus には、デバイス オブジェクトとエラー メッセージを含む DeviceStatusItem[] が含まれます。

エラーと例外

システム エラーによって操作が完了しなかった場合は、RemoteException がスローされます。

getAllowedOperationByDevicePlatform

構文

```
Device.LicenseOperation[] getAllowedOperationByDevicePlatform(UserToken token,
Device.DevicePlatform platform) throws RemoteException;
```

説明

この関数は、指定したプラットフォームで許可されている操作の配列を返します。

入力パラメータ

パラメータ	タイプ	値	説明
token	UserToken、必須	—	ユーザの認証パスを表すトークン。これは、ユーザが login 関数を呼び出して、バックエンド サーバがそのユーザを認証した後に取得されます。
platform	DevicePlatform 列挙体、必須	CSL、ACE、URLF、CISCO12K、IPS、SANOSDCOS、PIXASA、CSCSSM、MSE	デバイス プラットフォーム。次のいずれかを指定できます。CSL、ACE、URLF、CISCO12K、IPS、SANOSDCOS、PIXASA、CSCSSM、MSE。

戻り値

この関数は、指定したデバイス プラットフォームで許可されている操作を含む Device.LicenseOperation の配列を返します。デバイス プラットフォームが不明な場合は、空の配列が返されます。

エラーと例外

システム エラーによって操作が完了しなかった場合は、RemoteException がスローされます。

getDiscoveryAuthInfo

構文

```
DiscoveryAuthInfo getDiscoveryAuthInfo (UserToken token) throws RemoteException;
```

説明

この関数は、検出時に管理者ユーザによって設定されたユーザ名、パスワード、およびイネーブル パスワードのコレクションを返します。この関数を呼び出す権限を得るには、管理者ロールを使用してログインする必要があります。

入力パラメータ

パラメータ	タイプ	値	説明
token	UserToken、必須	—	ユーザの認証パスを表すトークン。これは、ユーザが login 関数を呼び出して、バックエンド サーバがそのユーザを認証した後に取得されます。

戻り値

この関数は DiscoveryAuthInfo オブジェクトを返します。

エラーと例外

システム エラーによって操作が完了しなかった場合は、RemoteException がスローされます。

getDiscoverySettings

構文

```
DiscoverySetting[] getDiscoverySettings (UserToken token) throws RemoteException;
```

説明

この関数は、Cisco License Manager 3.0 がインストールされた後のすべての検出情報を取得します。DiscoverySetting はデータ ストレージに含まれるデータ構造体で、各検出操作のネットワーク IP、ネットワーク マスク、TransportMethods、および DiscoveryAuthInfo を格納しています。

入力パラメータ

パラメータ	タイプ	値	説明
token	UserToken、必須	—	ユーザの認証パスを表すトークン。これは、ユーザが login 関数を呼び出して、バックエンド サーバがそのユーザを認証した後に取得されます。

戻り値

この関数は DiscoverySetting 配列を返します。これは管理者ロールを持つユーザだけが取得できます。ユーザが管理者ユーザでない場合、この関数は null 値を返します。この関数呼び出しの前に検出操作が 1 度も実行されていない場合は、空の配列が返されます。

次の例は、DiscoverySetting の値を取得する方法を示します。

```
DiscoverySetting[] hist = licenseManager.getDiscoverySettings(_token);

If(hist==null)
{
//no discovery occurred, therefore, no DiscoveryHistory return;
}
for (int i = 0; i < hist.length(); i++)

{
// Get the individual object returned by the operation.
DiscoveryAuthInfo _authinfo=hist[i].getDiscoveryAuthInfo();
String net_ip=hist[i].getNetIpAddr();
String net_mask=hist[i].getNetMask();
}
```

エラーと例外

システム エラーによって操作が完了しなかった場合は、RemoteException がスローされます。

getDiscoveryTransports

構文

```
Device.TransportMethod[] getDiscoveryTransports (UserToken token) throws RemoteException;
```

説明

この関数は、検出時に管理者ユーザによって設定されたトランスポート方法の配列を返します。この API は管理者ユーザだけが実行できます。

入力パラメータ

パラメータ	タイプ	値	説明
token	UserToken、必須	—	ユーザの認証パスを表すトークン。これは、ユーザが login 関数を呼び出して、バックエンド サーバがそのユーザを認証した後に取得されます。

戻り値

この関数は TransportMethods の配列を返します。

エラーと例外

システム エラーによって操作が完了しなかった場合は、RemoteException がスローされます。

listAllDevicesInGroup

構文

```
public IDPagingInfo listAllDevicesInGroup(UserToken token,String group,Pagination
pageinfo)throws RemoteException;
```

説明

この関数は、ステータス、総数、および指定したデバイス グループに属するデバイス ID の配列を含む IDPagingInfo オブジェクトを返します。グループが null の場合は、デフォルト グループに含まれるデバイス ID が返されます。

有効なオフセットと最大数を指定してページ分割オプションを設定できます。次の例は、オフセットと最大数を指定してページ分割を設定する方法を示します。

1 ページ目を取得するには、次のようにします。

```
Pagination p = new Pagination(0, 10)
```

5 ページ目を取得するには、次のようにします。

```
Pagination p = new Pagination(50, 10)
```

ページ分割せずにすべてのレコードを取得するには、次のようにします。

```
Pagination p = new Pagination(0, -1)
```

入力パラメータ

パラメータ	タイプ	値	説明
token	UserToken、必須	—	ユーザの認証パスを表すトークン。これは、ユーザが login 関数を呼び出して、バックエンド サーバがそのユーザを認証した後に取得されます。

パラメータ	タイプ	値	説明
group	String、必須	—	デバイス グループの名前。グループ化されていないデバイスを検索する場合は null。
pageinfo	DataObject、必須	—	offset と max を含むページ分割オブジェクト。offset は、この関数によって取得する、最初のレコード セットからの相対的なオフセットを指定します。max は、取得するレコードの最大数を指定します。max を -1 に設定すると、値が -1 であるすべてのレコードが返されます。

戻り値

この関数は、デバイス ID の配列を含む IDPagingInfo オブジェクトを返します。offset がレコード数よりも大きい場合は、サイズが 0 の配列が返されます。操作エラーが発生した場合、Status オブジェクトにはエラー コードとエラー メッセージが含まれます。

エラーと例外

システム エラーによって操作が完了しなかった場合は、RemoteException がスローされます。エラーが発生した場合、この関数は null を返します。

listAllGroupsByDevice

構文

```
String[] listAllGroupsByDevice(UserToken token, String dev_id) throws RemoteException;
```

説明

この関数は、指定したデバイスが属するグループの配列を返します。

入力パラメータ

パラメータ	タイプ	値	説明
token	UserToken、必須	—	ユーザの認証パスを表すトークン。これは、ユーザが login 関数を呼び出して、バックエンド サーバがそのユーザを認証した後に取得されます。
dev_ids	String、必須	—	デバイス ID の配列。

戻り値

この関数は、デバイスが有効な場合、グループの配列を返します。

エラーと例外

システム エラーによって操作が完了しなかった場合は、RemoteException がスローされます。

エラーが発生した場合、この関数は null を返します。

listAllDevices

構文

```
listAllDevices(UserToken token,Pagination pageinfo) throws RemoteException;
```

説明

この関数は、インベントリからすべてのデバイスを取得します。

入力パラメータ

パラメータ	タイプ	値	説明
token	UserToken、必須	—	ユーザの認証パスを表すトークン。これは、ユーザが login 関数を呼び出して、バックエンド サーバがそのユーザを認証した後に取得されます。
pageinfo	DataObject、必須	—	offset と max を含むページ分割オブジェクト。offset は、この関数によって取得する、最初のレコード セットからの相対的なオフセットを指定します。max は、取得するレコードの最大数を指定します。max を -1 に設定すると、値が -1 であるすべてのレコードが返されます。

戻り値

この関数は、デバイスの配列を含むデバイス ページング情報オブジェクトを返します。offset がレコード数よりも大きい場合は、サイズが 0 の配列が返されます。

エラーと例外

操作エラーが発生した場合、Status オブジェクトにはエラー コードとエラー メッセージが含まれます。

システム エラーによって操作が完了しなかった場合は、RemoteException がスローされます。

listApplicableDevicesBySKU

構文

```
public String[] listApplicableDevicesBySKUs(UserToken token,SKU[] skus, String
group)throws RemoteException;
```

説明

この関数は、指定したグループに含まれるデバイス ID のうち、指定した SKU 内のすべてのフィーチャと一致するライセンス可能なフィーチャを持つデバイス ID の配列を返します。グループが null の場合は、すべてのグループから一致するデバイス ID を返します。

入力パラメータ

パラメータ	タイプ	値	説明
token	UserToken、必須	—	ユーザの認証パスを表すトークン。これは、ユーザが login 関数を呼び出して、バックエンド サーバがそのユーザを認証した後に取得されます。
sku	SKU、必須	有効な SKU オブジェクト	指定された SKU オブジェクト。
group	String	—	グループ名。

戻り値

指定したグループに含まれるデバイス ID のうち、指定した SKU 内のすべてのフィーチャと一致するライセンス可能なフィーチャを持つデバイス ID の配列。入力 SKU 配列が null または空の場合、またはどの入力 SKU にもフィーチャが含まれていない場合は、すべてのデバイスが返されます。一致するデバイスが見つからない場合は、長さが 0 の配列が返されます。

エラーと例外

SKU が null の場合、この API は null を返します。SKU 内のプラットフォームが null または不明な場合、または SKU プラットフォームと一致するデバイスがない場合には、長さが 0 の空の配列が返されます。

システム エラーによって操作が完了しなかった場合は、RemoteException がスローされます。

（注）　デバイスから取得されるライセンス可能なフィーチャは SWIFT から取得されるものとは異なるため、SKU フィルタリングは非 CSL デバイスには適用できません。これは PIX、ASA、FWSM、および URLF に当てはまります。

listDeviceIdsByFilter

構文

```
String[] listDeviceIdsByFilter(UserToken token, String device_type, String device_model,
String[] features) throws RemoteException;
```

説明

この関数は、device_type、device_model、およびライセンス フィーチャを含むデバイス ID のリストを返します。

フィルタリングを行わないフィルタには null を設定できます。すべてのフィルタを null に設定すると、その操作はエラーと見なされ、null が返されます。

入力パラメータ

パラメータ	タイプ	値	説明
token	UserToken、必須	—	ユーザの認証パスを表すトークン。これは、ユーザが login 関数を呼び出して、バックエンド サーバがそのユーザを認証した後に取得されます。
device_type	String、省略可能	—	デバイス タイプを表す文字列。
device_model	String、省略可能	—	デバイス モデルを表す文字列。
features	String の配列、省略可能	—	ライセンス フィーチャの名前を表す文字列の配列。

戻り値

この関数は文字列の配列を返します。各文字列は 1 つのデバイス ID を表します。

エラーと例外

システム エラーによって操作が完了しなかった場合は、RemoteException がスローされます。

エラーが発生した場合、この関数は null を返します。長さが 0 の配列は、デバイスが見つからなかったことを意味します。

listRunningAsyncOperationJobRecords

構文

```
ClmJobStatus listRunningAsyncOperationJobRecords(UserToken token, boolean
include_all_user) throws RemoteException;
```

説明

この関数は、ユーザが開始した実行中の非同期操作に関する情報を含むすべての Cisco License Manager ジョブを返します。

入力パラメータ

パラメータ	タイプ	値	説明
token	UserToken、必須	—	ユーザの認証パスを表すトークン。これは、ユーザが login 関数を呼び出して、バックエンド サーバがそのユーザを認証した後に取得されます。
Include_all_user	Boolean、必須	True、False	true を指定した場合はすべてのレコードが返されます。false を指定した場合は、ユーザ名を含むレコードだけが返されます。

戻り値

この関数は ClmStatus オブジェクトを返します。

エラーと例外

関数全体のエラーが発生した場合、この関数は、SUCCESS 以外の値を含む ClmJobStatus オブジェクトを返します。ClmJob は、ClmJobStatusItem 配列から取得できます。

システム エラーによって操作が完了しなかった場合は、RemoteException がスローされます。

readDevices

構文

```
DeviceStatus readDevices(UserToken token, String[] dev_ids) throws RemoteException;
```

説明

この関数は、指定したデバイス ID を使用して、インベントリからデバイス オブジェクトの配列を取得します。

入力パラメータ

パラメータ	タイプ	値	説明
token	UserToken、必須	—	ユーザの認証パスを表すトークン。これは、ユーザが login 関数を呼び出して、バックエンド サーバがそのユーザを認証した後に取得されます。
dev_ids	String の配列、必須	—	デバイス ID の配列。

戻り値

この関数は DeviceStatus オブジェクトを返します。次の例では、ステータスに含まれるエラー コード、エラー メッセージ、および戻りオブジェクトを取得しています。

```
        DeviceStatus status = readDevices(…, …);

        // The general error code of the operation.
        int err_code = status.getErrorCode();

        // The general error message of the operation.
        String err_msg = status.getErrorMessage();

        // A list of status for each individual element in the
        // bulk operation.
        DeviceStatusItem[] items = status.getDeviceStatusItems()

        // Iterate through the list to get individual status.
    for (int i = 0; i < items.length(); i++) {

        // Get the individual object returned by the operation.
        Device device = items[i].getDevice();

        // Get the individual error code corresponding to
        // the object.
            int item_err_code = items[i].getErrorCode();

        // Get the individual error message corresponding to
        // the object.
            String item_err_msg = items[i].getErrorMessage();
```

```
}
```

エラーと例外

システム エラーによって操作が完了しなかった場合は、RemoteException がスローされます。

操作エラーが発生した場合は、エラーに関する情報を含む DeviceStatus オブジェクトが返されます。個々の要素のステータスを調べるには、DeviceStatus に含まれる DeviceStatusItem 配列を反復処理する必要があります。各 DeviceStatusItem には、Device オブジェクト、エラー コード、およびエラー メッセージが含まれます。

writeDevices

構文

```
IDStatus writeDevices(UserToken token, Device[] devices) throws RemoteException;
```

説明

この関数は、指定したデバイス オブジェクトをインベントリに書き込みます。入力デバイス オブジェクトには、createDevices() 関数によって返された新しいデバイスのインスタンス、または readDevices() 関数によってインベントリから取得された既存のインスタンスを指定できます。

入力パラメータ

パラメータ	タイプ	値	説明
token	UserToken、必須	—	ユーザの認証パスを表すトークン。これは、ユーザが login 関数を呼び出して、バックエンド サーバがそのユーザを認証した後に取得されます。
devices	デバイスの配列、必須	—	デバイス オブジェクトの配列。デバイス クラスの新しいインスタンス、または readDevices() 関数によってインベントリから取得された既存のインスタンスを指定できます。

戻り値

この関数は IDStatus オブジェクトを返します。次の例では、ステータスに含まれるエラー コード、エラー メッセージ、および戻りオブジェクトを取得しています。

```
    IDStatus status = writeDevices(…, …);

    // The general error code of the operation.
    int err_code = status.getErrorCode();

    // The general error message of the operation.
    String err_msg = status.getErrorMessage();

    // A list of status for each individual element in the
    // bulk operation.
    IDStatusItem[] items = status.getIDStatusItems()

    // Iterate through the list to get individual status.
for (int i = 0; i < items.length(); i++) {

    // Get the individual ID returned by the operation.
```

```
String id = items[i].getID();

// Get the individual error code corresponding to
// the ID.
    int item_err_code = items[i].getErrorCode();

// Get the individual error message corresponding to
// the ID.
    String item_err_msg = items[i].getErrorMessage();
}
```

エラーと例外

システム エラーによって操作が完了しなかった場合は、RemoteException がスローされます。

入力配列の要素でエラーが発生した場合は、エラーに関する情報を含む Status オブジェクトが返されます。