



サーバー管理

ActionScript リファレンスガイド

商標

1 Step RoboPDF、ActiveEdit、ActiveTest、Authorware、Blue Sky Software、Blue Sky、Breeze、Breezo、Captivate、Central、ColdFusion、Contribute、Database Explorer、Director、Dreamweaver、Fireworks、Flash、FlashCast、FlashHelp、Flash Lite、FlashPaper、Flex、Flex Builder、Fontographer、FreeHand、Generator、HomeSite、JRun、MacRecorder、Macromedia、MXML、RoboEngine、RoboHelp、RoboInfo、RoboPDF、Roundtrip、Roundtrip HTML、Shockwave、SoundEdit、Studio MX、UltraDev、および WebHelp は、Macromedia, Inc. の商標または登録商標であり、米国およびその他の法域で登録される場合があります。本マニュアルにおけるその他の製品名、ロゴ、デザイン、タイトル、語句は、Macromedia, Inc. またはその他の団体の商標、サービスマーク、または商号である場合があります。米国およびその他の国の特定の法域で登録されている場合があります。

サードパーティー情報

本マニュアルには、Macromedia 社が管理していない、サードパーティーの Web サイトへのリンクが掲載されていますが、Macromedia 社はいかなるリンク先サイトの内容についても責任を持ちません。本マニュアルに記載されているサードパーティーの Web サイトには、自己責任においてアクセスしてください。Macromedia 社はこれらのリンクを便宜上の目的においてのみ掲載しています。リンクを掲載することにより、Macromedia 社がこれらのサードパーティーのサイトの内容について何らかの責任を持つことを示すものではありません。



Sorenson™ Spark™ ビデオ圧縮および圧縮解凍テクノロジーは、Sorenson Media, Inc. のライセンス供与によって提供されます。

Copyright © 2002-2005 Macromedia, Inc. All rights reserved. 本マニュアルの全部または一部を Macromedia, Inc. の書面による事前の許可なしに、複製、複写、再製造、または翻訳すること、および電子的または機械的に読み取り可能な形に変換することは禁じられています。前述の内容にかかわらず、本マニュアルが付属する正規ソフトウェアの所有者または認証されたユーザーは、ソフトウェアの使用方法を習得することのみを目的として、本マニュアルの電子版を1部印刷することができます。それ以外の目的のために本マニュアルのいかなる部分も印刷、複製、配布、転売、および転送することは禁じられています。また、いかなる場合においても、本マニュアルの販売や有償のサポートサービスの提供など、営利目的で利用することは禁じられています。

マニュアル制作スタッフ

プロジェクト管理 : Suzanne Smith

執筆 : Jody Bleye

編集管理 : Rosana Francescato

編集 : Geta Carlson

制作管理 : Patrice O'Neill

メディアデザイン・制作 : Adam Barnett、Aaron Begley、Paul Benkman、John Francis、Mario Reynoso Erick Vera、ならびに Flash Media Server エンジニアリングおよび QA チームの協力に感謝します。

初版 : 2005 年 10 月

Macromedia, Inc.

601 Townsend St.

San Francisco, CA 94103

マクロメディア株式会社

東京都港区赤坂 2-17-22 赤坂ツインタワー本館 13F

目次

第1章：サーバー管理 ActionScript リファレンスガイド.....	5
サーバー管理 ActionScript API の使用	5
サーバー接続の確立	5
アプリケーションの単純な例	7
情報オブジェクト	9
メソッドの種類	9
addAdmin()	13
addApp()	14
addVHostAlias()	15
approveDebugSession()	16
changePswd()	17
gc()	18
getActiveInstances()	19
getAdaptors()	19
getAdminContext()	20
getAdmins()	21
getApps()	21
getAppStats()	22
getConfig()	24
getConfig2()	26
getGroupMembers()	29
getGroupStats()	30
getGroups()	32
getInstanceStats()	32
getIOStats()	34
getLicenseInfo()	36
getLiveStreams()	38
getLiveStreamStats()	39
getMsgCacheStats()	40
getNetStreams()	41
getNetStreamStats()	42
getRecordedStreams()	43
getRecordedStreamStats()	44
getScriptStats()	45

getServerStats()	46
getServices()	48
getSharedObjects()	48
getSharedObjectStats()	49
getUsers()	50
getUserStats()	51
getVHosts()	53
getVHostStats()	53
ping()	55
reloadApp()	56
removeAdmin()	57
removeApp()	58
removeVHostAlias()	59
restartVHost()	60
setConfig()	61
setConfig2()	64
startServer()	67
startVHost()	68
stopServer()	69
stopVHost()	70
unloadApp()	71

サーバー管理 ActionScript リファレンスガイド

Macromedia Flash Media Server には、高度なメソッドをまとめたサーバー管理 ActionScript のためのアプリケーションプログラミングインターフェイス (API) が含まれています。このサーバー管理 ActionScript API を使用して Macromedia Flash アプリケーションを作成することによって、Management Console (管理コンソール) の拡張、ユーザー独自の管理および監視ツールの作成が可能になります。

このドキュメントでは、このサーバー管理 ActionScript API について説明します。ActionScript については、『ActionScript 2.0 リファレンスガイド』を参照してください。Flash Media Server については、『Flash Media Server 2 クライアントサイド ActionScript リファレンスガイド』、『サーバーサーバーサイド ActionScript リファレンスガイド』、および『Flash Media Server 管理ガイド』を参照してください。

サーバー管理 ActionScript API の使用

サーバー管理 ActionScript API を使用するには、Flash と Flash Media Server がすでにインストールされており、ユーザーが Flash Media Server への管理者アクセス権限を持っている必要があります。また、ActionScript および Flash アプリケーションの作成に精通していることを前提としています。

サーバー接続の確立

Flash Media Server で使用されるデフォルトポートは 1935 ですが、サーバー管理 ActionScript API を使用するには、ポート 1111 上の管理サーバーを使用して Flash Media Server に接続する必要があります。

メモ

このマニュアルでは、管理サーバー用のデフォルトポート番号が変更されていないことを前提として作成されています。ポート番号を変更した場合は、ポート 1111 の代わりに有効なポート番号を使用してください。

NetConnection.connect メソッドを使用し、ホスト管理サーバーの URL、管理者のユーザー名、および管理者のパスワードという 3 つのパラメータを渡すことによって、サーバー接続が確立します。

接続先の管理サーバーホストが仮想ホスト(同一マシン上に存在する複数サービスの1つ)であるときは、URIの一部として仮想ホストのドメイン名またはIPアドレスを必ず指定してください(例、`rtmp://www.myVhost.com/admin:1111`)。このように指定することにより、管理ホストマシンに接続したときに、指定した仮想ホスト上で稼働中のサーバーのインスタンスに接続できます。

サーバーに接続できるのは、`Users.xml` 設定ファイルに定義されている有効な管理者のみです。したがって、`NetConnection.connect` メソッドには、2つの管理パラメータとして、有効な管理者のユーザー名およびパスワードを指定する必要があります。

次の例は、ユーザー名が `MHill` でパスワードが `635xjh27` である管理者が `localhost` 上のサーバーに接続するときの呼び出しを示します。

```
nc = new NetConnection();
nc.connect("rtmp://localhost:1111/admin", "MHill", "635xjh27");
```

また、権限のないユーザーが管理サーバーにアクセスすることを防止するには、ファイアウォールを使用して管理サーバーポート(ポート1111)へのアクセスを制限するとよいでしょう。Flash Media Serverのセキュリティの詳細については、www.macromedia.com/go/flashmediaserver_security_en でセキュリティについてのホワイトペーパーを参照してください。

シンタックスの例

サーバー管理 ActionScript API で呼び出しを実行するには、コードにコールバックハンドラとすべての必須パラメータを含める必要があります。

次の例では、`getAppStats` メソッドが呼び出され、実行中のアプリケーションのパフォーマンスデータを取得します。コールバックハンドラ `new receiveAppStats` が結果をキャプチャします。値 `ChatApp` は、アプリケーション名を指定するための必須なパラメータです。

```
nc = new NetConnection();
nc.connect("rtmp://localhost:1111/admin", "MHill", "635xjh27");

// Call getAppStats
nc.call("getAppStats", new receiveAppStats(), "ChatApp");
```

アプリケーションの単純な例

このセクションでは、`getAppStats` メソッドの呼び出しを行う単純な Flash アプリケーションのコードを紹介します。このコードをコピーして Flash に貼り付けると、単純な呼び出しがどのように行われるかを確認することができます。

Flash では、次の要素を使ってアプリケーションを生成します。

- Application Name という名前の入力フィールド (枠付き)
- `doGetAppStats()` メソッドを呼び出すボタン (後述のコードサンプルに表示)
- `outputBox` という名前の複数行の動的なテキストフィールド (枠付き)
- テキストフィールドの横のスクロールコンポーネント

この単純なインターフェイスを使って、サーバー管理 ActionScript API をテストできます。

次に、以下に示す ActionScript のサンプルコードを [Actions] パネルに入力します。このとき、`admin_name` と `admin_pass` を有効な管理者名とパスワードに置き換えてください。

メモ

継続文字 (`-`) は、コードが次の行に継続していることを表しています。この文字をコードに含めないでください。

```
/** Flash Media Server への接続を確立 **/  
  
nc = new NetConnection();  
nc.connect("rtmp://localhost:1111/admin","admin_name","admin_pass");  
  
/** GetAppStats などの API の呼び出し実行 **/  
function doGetAppStats() {  
    function onGetAppStats()  
    {  
        this.onResult = function(info)  
        {  
            if (info.code != "NetConnection.Call.Success")  
                outputBox.text = "Call failed:" + info.description;  
            else {  
                outputBox.text = "Info for "+appName.text+ " returned:"+ -  
newline;  
                printObj(info, outputBox);  
            }  
        }  
    }  
    nc.call("getAppStats", new onGetAppStats(), appName.text);  
}  
  
// この関数はオブジェクトを処理し、オブジェクト内の再帰的な  
// 出力オブジェクトを含め、すべての値を対象に出力する。  
// tabLevel パラメータは、体裁を整えるため使用される。  
function printObj(obj, destination, tabLevel)
```

```

{
  if (arguments.length < 2) {
    trace("ERROR! you need to supply a text object to output to");
    return;
  }
  if (arguments.length < 3)
    tabLevel = 0;

  for (var prop in obj) {
    for (var i = 0; i < tabLevel; i++) // 要求された数のタブ文字の挿入
      destination.text += "\t";

    destination.text += prop + " = " + obj[prop] + newline;
    if (typeof (obj[prop]) == "object") { // printObj の再帰的呼び出し
      printObj(obj[prop], destination, tabLevel+1 );
    }
  }
}

// サーバーへの接続でエラーが発生した場合にアラートを出す
nc.onStatus = function(info) {
  if (info.code == "NetConnection.Connect.Success") {
    trace("Connected! The call should work")
  } else {
    // サーバーの状態をテストするには、nc.isConnected を使用
    if (! nc.isConnected)
      trace("NO netConnection to server. Call will not work");
  }
}

```

情報オブジェクト

すべてのサーバ管理 ActionScript API メソッドでは、level、code、timestamp、data、description、および details というプロパティを使用して、情報オブジェクト内のデータを返します。すべての情報オブジェクトには、level、code、および timestamp プロパティが含まれます。一部のメソッドには、戻り値のデータ（オブジェクトまたは配列内に含まれている場合が多い）を含む data プロパティや、通常はエラー情報を提供する description および details プロパティが含まれます。

NetConnection オブジェクトには、『Flash Media Server 2 クライアントサイド ActionScript リファレンスガイド』に記載されている情報オブジェクトのほかに、サーバ管理 ActionScript API 専用の情報オブジェクトがあります。次の表は、通常返される情報オブジェクトの code プロパティ、level プロパティ、および意味について一覧表示しています。

Code プロパティ	Level プロパティ	意味
NetConnection.Call.Success	Status	呼び出しは成功です。 この情報オブジェクトに含まれる data プロパティでは、オブジェクトまたは配列の詳細な情報が返されます。
NetConnection.Admin.CommandFailed	Error	呼び出されたメソッドは存在しません。*
NetConnection.Call.Failed	Error	一般的な失敗が発生しました。*
NetConnection.Call.BadValue	Error	パラメータの値が無効です。*

* この情報オブジェクトには、description および detail プロパティが含まれ、失敗についてのより具体的な理由が文字列で示されます。

メソッドの種類

サーバ管理 ActionScript API に含まれるのは、次の3種類のメソッドです。

- クエリー：Flash Media Server とそのアプリケーション、および特定のアプリケーションインスタンスを監視します。
- 管理コマンド：管理ユーザーの追加、サーバ、仮想ホスト、およびアプリケーションの起動と終了など、Flash Media Server の管理タスクを実行します。
- 設定コマンド：getConfig2() コマンドによって、サーバの設定キーを参照できます。また、setConfig2() コマンドによって、キーの値を設定できます。

各メソッドの説明でアスタリスク(*)が付いているものは、サーバー管理者のみが使用できます。仮想ホスト管理者は、これらのメソッドを使用できません。ただし、仮想ホスト管理者でも制限付きでメソッドを使用できる場合もあります。このような制限については、メソッドの項目で説明しています。

メモ	任意のパラメータは角カッコ ([]) で囲まれています。これ以外のパラメータはすべて必須です。任意のパラメータを指定しない場合は、メソッドによってはデフォルト値が使用されます。たとえば、 <code>scope</code> パラメータに仮想ホストを指定しない場合は、Flash Media Server にログオンしたときに接続している仮想ホスト上でコマンドが実行されます。
----	---

サーバー監視用クエリー

次の表は、サーバーの監視に使用できるメソッドの一覧です。

メソッド	説明
<code>approveDebugSession()</code>	保留のデバッグセッション要求を許可して、選択されたアプリケーションに接続できるようにします。
<code>getActiveInstances()</code>	接続した仮想ホスト上で実行するすべてのアプリケーションインスタンス名を含む文字列の配列を返します。
<code>getAdaptors()</code>	アダプタ名の配列を返します。
<code>getAdminContext()</code>	管理者の管理コンテキスト (管理者タイプ、アダプタ名、および仮想ホスト名) を返します。
<code>getAdmins()</code>	Flash Media Server のすべての管理者を返します。
<code>getApps()</code>	インストールされているすべてのアプリケーションの名前を含む文字列の配列を返します。
<code>getAppStats()</code>	すべてのアプリケーションインスタンスについての包括的な情報を返します。
<code>getGroupMembers()</code>	特定グループに属するメンバーのリストを返します。
<code>getGroupStats()</code>	特定のグループ接続についての統計を返します。
<code>getGroups()</code>	特定のアプリケーションインスタンスへのグループ接続のリストを返します。
<code>getInstanceStats()</code>	実行中のアプリケーションインスタンス1つについて詳細な情報を返します。
<code>getIOStats()</code>	入出力情報 (入力バイト、出力バイトなど) を返します。*
<code>getLicenseInfo()</code>	ライセンスキー情報を返します。
<code>getLiveStreams()</code>	特定アプリケーションあてにパブリッシュされているすべてのライブストリームのリストを返します。
<code>getLiveStreamStats()</code>	ライブストリームの詳細情報を返します。

メソッド	説明
<code>getMsgCacheStats()</code>	サーバーのTCMessage キャッシュの統計を返します。
<code>getNetStreams()</code>	現在アプリケーションに接続しているすべてのネットワークストリームのリストを返します。
<code>getNetStreamStats()</code>	特定のネットワークストリームの詳細情報を返します。
<code>getRecordedStreamStats()</code>	記録ストリームの詳細情報を返します。
<code>getRecordedStreams()</code>	特定のアプリケーションインスタンスから再生中のすべての記録ストリーム名を含む配列を返します。
<code>getScriptStats()</code>	アプリケーションの指定したインスタンスで実行中のスクリプトのパフォーマンスデータを取得します。
<code>getServerStats()</code>	サーバーの実行時間、入出力およびメッセージキャッシュの統計など、サーバーの操作に関する状態と統計を取得します。*
<code>getServices()</code>	現在 Flash Media Server に接続しているすべてのサービスの名前を含む配列を返します。
<code>getSharedObjects()</code>	アプリケーションの指定したインスタンスで使用中のすべての永続的および非永続的共有オブジェクトのリストを返します。
<code>getSharedObjectStats()</code>	共有オブジェクトの詳細情報を返します。
<code>getUsers()</code>	指定したアプリケーションインスタンスに接続中のすべてのユーザーのリストを返します。
<code>getUserStats()</code>	指定したユーザーについての詳細情報を返します。
<code>getVHosts()</code>	指定したアダプタに定義される仮想ホストの配列を返します。
<code>getVHostStats()</code>	仮想ホストの統計を返します。
<code>ping()</code>	サーバーの状態を示す状態文字列を返します。

* このメソッドを使用できるのはサーバー管理者のみです。

サーバー管理用コマンド

次の表は、サーバーの管理に使用できるメソッドの一覧です。

メソッド	説明
<code>addAdmin()</code>	管理者をシステムに追加します。*
<code>addApp()</code>	新しいアプリケーションを追加します。
<code>addVHostAlias()</code>	仮想ホストにエイリアスを追加します。
<code>changePswd()</code>	システムの管理者用パスワードを変更します。
<code>gc()</code>	サーバーリソースのガーベージコレクションを実行します。*
<code>reloadApp()</code>	アプリケーションインスタンスがロードされている場合はアンロードし、続いてインスタンスを再ロードします。
<code>removeAdmin()</code>	管理者をシステムから削除します。*
<code>removeApp()</code>	アプリケーション、またはアプリケーションインスタンスを削除します。
<code>removeVHostAlias()</code>	仮想ホストからエイリアスを削除します。
<code>restartVHost()</code>	仮想ホストを再起動します。
<code>startServer()</code>	Flash Media Server を起動または再起動します。*
<code>startVHost()</code>	指定した仮想ホストが停止している場合は起動します。仮想ホストのディレクトリがファイルシステムに作成されている場合は、新しい仮想ホストを有効にします。*
<code>stopServer()</code>	Flash Media Server を終了します。*
<code>stopVHost()</code>	仮想ホストを停止します。
<code>unloadApp()</code>	すべて、またはいずれかのアプリケーションインスタンスをアンロードします。すべてのユーザーとの接続を切断します。

* このコマンドを使用できるのはサーバー管理者のみです。

サーバー設定用コマンド

次の表は、サーバーの設定に使用できるメソッドの一覧です。

メソッド	説明
<code>getConfig2()</code>	指定した設定キーの設定情報を返します。
<code>setConfig2()</code>	指定した設定キーに値を設定します。

このマニュアルでは、メソッドはアルファベット順に一覧表示されています。

addAdmin()

使用できるバージョン

- Flash Player 6
- Flash Communication Server MX 1.0

シンタックス

```
addAdmin(admin_name, password [,scope])
```

パラメータ

admin_name 追加された管理者のユーザー名を含む文字列。

password 管理者のパスワード。パスワードは Server.xml 設定ファイルに書き込まれる前にエンコーディングされます。

scope オプションのパラメータ。管理者がサーバー管理者であるか、または仮想ホスト管理者であるか(その場合は、どの仮想ホスト管理者であるか)を指定する文字列です。サーバー管理者を追加するには、「server」と指定します。

接続している仮想ホストに仮想ホスト管理者を追加するには、このパラメータを削除します。別の仮想ホストに仮想ホスト管理者を追加するには、追加先の仮想ホストを *adaptor_name/virtual_host_name* の形式で指定します。

戻り値

呼び出しが成功すると、サーバーは *level* プロパティの *status* および *code* プロパティの `NetConnection.Call.Success` を含む情報オブジェクトを応答として送信します。Server.xml 設定ファイルは、新しい管理者の情報を使用して更新されます。

呼び出しが失敗すると、サーバーは、*level* プロパティの *error* および *code* プロパティの `NetConnection.Admin.CommandFailed`、またはより具体的な値(存在する場合)を含む情報オブジェクトを応答として送信します。失敗の原因について説明する文字列を含む *description* プロパティが含まれているオブジェクトもあります。

指定した管理者がすでにシステム上に存在している場合は、コマンドは失敗します。

説明

管理者をシステムに追加します。指定するパラメータに応じて、サーバー管理者または仮想ホスト管理者を追加できます。

管理者をシステムに追加できるのは、サーバー管理者のみです。

例

次の3つの例では、addAdmin コマンドの呼び出しでのパラメータの指定方法を示しています。

```
/* ユーザー名が GLee、パスワードが boat4907 となるサーバー管理者を追加 */
nc.call("addAdmin", new onAddAdmin(), "GLee", "boat4907", "server");

/* ユーザー名が ChrisM、パスワードが tree2981 となる仮想ホスト管理者を追加 */
nc.call("addAdmin", new onAddAdmin(), "ChrisM", "tree2981");

/* ユーザー名が DHong、パスワードが wate3235 となる仮想ホスト管理者を */
/* 仮想ホスト tree.oak.com に追加 */
nc.call("addAdmin", new onAddAdmin(), "DHong", "wate3235", "_defaultRoot_/tree.oak.com");
```

addApp()

使用できるバージョン

- Flash Player 6
- Flash Communication Server MX 1.0

シンタックス

```
addApp(app_name)
```

パラメータ

app_name 追加するアプリケーション名を含む文字列。

戻り値

呼び出しが成功すると、サーバーは level プロパティの status および code プロパティの NetConnection.Call.Success を含む情報オブジェクトを応答として送信します。

呼び出しが失敗すると、サーバーは、level プロパティの error および code プロパティの NetConnection.Call.BadValue、またはより具体的な値 (存在する場合) を含む情報オブジェクトを応答として送信します。失敗の原因について説明する文字列を含む description プロパティが含まれているオブジェクトもあります。

説明

ディレクトリのツリーに新しいアプリケーションのディレクトリを作成することによって、仮想ホストに新しいアプリケーションを追加します。この新しいアプリケーションのディレクトリが作成されると、ユーザー (または、ファイルシステムへのアクセス権を持つ他の管理者) は、ディレクトリに必要なサーバーサイドスクリプトを追加できます。クライアントサイドのコードでは、この新しいアプリケーションのディレクトリを NetConnection.Connect の呼び出しの URI パラメータに使用します。

例

次の例では、接続中の仮想ホストに ChatApp アプリケーションを追加します。

```
nc = new NetConnection();
nc.connect("rtmp://localhost:1111/admin", "JLee", "x52z49ab");

nc.call("addApp", new onAddApp(), "ChatApp");
```

addVHostAlias()

使用できるバージョン

- Flash Player 6
- Flash Media Server 2

シンタックス

```
addVHostAlias(VHostName, AliasName, PersistValue)
```

パラメータ

VHostName エイリアスを追加する仮想ホストを示す文字列。

AliasName 指定した仮想ホストに追加するエイリアスを示す文字列。

PersistValue エイリアスの変更を設定ファイルに書き込むかどうかを示すブール値。"true" に設定すると、仮想ホストが次回再起動した後もエイリアスが保持されます。"false" に設定すると、仮想ホストが再起動したときにエイリアスは保持されません。

戻り値

呼び出しが成功すると、サーバーは、level プロパティの status および code プロパティの NetConnection.Call.Success を含む情報オブジェクトを応答として送信します。

説明

仮想ホストにエイリアスを追加します。エイリアスは、Flash Media Server の接続が確立されるときにターゲットとして使用される仮想ホストの代替名です。エイリアスを削除すると、接続が確立されるときにエイリアスを使用できなくなります。

この API が利用可能になるまでは、設定ファイルに手動でエイリアスを追加することが必要でした。

関連項目

[removeVHostAlias\(\)](#)

approveDebugSession()

使用できるバージョン

- Flash Player 6
- Flash Media Server 2

シンタックス

```
approveDebugSession(appInst, PIN)
```

パラメータ

appInst デバッグのための接続が保留になっているアプリケーションおよびインスタンスの名前を示す文字列。

PIN デバッグセッションの個人識別番号 (PIN) を示す数字。各デバッグ接続では、アプリケーションへの接続がキューに格納されるときにデバッグ番号が発行されます。この番号は、この API に含まれています。この API が処理されるとき、PIN が照合され、対応する接続が許可されます。これは、許可されないユーザーがデバッグ接続を実行することを防ぐためのセキュリティ上の対策です。

0 は、有効な *PIN* の値ではありません。セキュリティを確保するため、容易に推測できる数字を *PIN* に設定しないようにします。PIN はゼロ以外の数字であり、さらに +/- (2³¹ - 1) の範囲内 (プラス 2147483648 から マイナス 2147483648 の間) である必要があります。特定の PIN を持つ接続が保留中となっているときに、同一の PIN を持つ別の接続を受信した場合、2 番目の接続はセキュリティ対策のために拒否されます。

戻り値

呼び出しが成功すると、サーバーは `level` プロパティの `status`、および `code` プロパティの `NetConnection.Call.Success` を含む情報オブジェクトを応答として送信します。

説明

保留のデバッグセッション要求を許可して、選択されたアプリケーションに接続できるようにします。許可されたデバッグセッションは、`Services.onDebugConnect` ゲートウェイを介してアプリケーションに接続し、通常のクライアントとして動作できるようになります。

デバッグセッションを使用してアプリケーションに接続することにより、Management Console からアプリケーションのストリームや共有オブジェクトを表示できます。

例

次に、デバッグ接続の例を示します。

```
nc.connect("rtmp://serverName/appName%3F%5Ffcs%5Fdebugreq%5F%3D1234");  
// 元の文字列は _fcs_debugreq_=1234。
```

次に、デバッグ許可要求の例を示します。

```
nc_admin.call("approveDebugSession", null, appName/instName", 1234);
```

changePswd()

使用できるバージョン

- Flash Player 6
- Flash Communication Server MX 1.0.

シンタックス

```
changePswd(admin_name, password [, scope])
```

パラメータ

admin_name パスワードを変更する管理者の名前を含む文字列。

password 管理者の新しいパスワードを含む文字列。

scope オプションのパラメータ。管理者がサーバー管理者であるか、または仮想ホスト管理者であるか(その場合は、どの仮想ホスト管理者であるか)を指定する文字列。

接続中の仮想ホスト上の管理者のパスワードを変更するには、このパラメータを削除します。異なる仮想ホスト上の管理者のパスワードを変更するには、*adaptor_name/virtual_hostname* の形式で異なる仮想ホストを指定します。

サーバー管理者のパスワードを変更するには、「server」と指定します。

戻り値

呼び出しが成功すると、サーバーは *level* プロパティの *status* および *code* プロパティの `NetConnection.Call.Success` を含む情報オブジェクトを応答として送信します。

呼び出しが失敗すると、サーバーは、*level* プロパティの *error* および *code* プロパティの `NetConnection.Admin.CommandFailed`、またはより具体的な値(存在する場合)を含む情報オブジェクトを応答として送信します。失敗の原因について説明する文字列を含む *description* プロパティが含まれているオブジェクトもあります。

指定した管理者が存在しない場合は、コマンドは失敗します。

説明

指定した管理者のパスワードを変更しますパスワードは `Server.xml` 設定ファイルに書き込まれる前にエンコーディングされます。

仮想ホスト管理者は、自分のパスワード以外は変更できません。

例

```
/* サーバー管理者 ASilva のパスワードを cbx5978y に変更 */
nc.call("changePswd", new onChangePswd(), "ASilva", "cbx5978y", "server");

/* 仮想ホスト管理者 JLee のパスワードを kbat3786 に変更 */
nc.call("changePswd", new onChangePswd(), "JLee", "kbat3786");

/* 仮想ホスト tree.oak.com 上の仮想ホスト管理者 JLee の */
/* パスワードを kbat3786 に変更 */
nc.call("changePswd", new onChangePswd(), "JLee", "kbat3786", "_defaultRoot_/~
tree.oak.com");
```

gc()

使用できるバージョン

- Flash Player 6
- Flash Communication Server MX 1.0

シンタックス

gc()

パラメータ

なし

戻り値

呼び出しが成功すると、サーバーは level プロパティの status および code プロパティの NetConnection.Call.Success を含む情報オブジェクトを応答として送信します。

呼び出しが失敗すると、サーバーは、level プロパティの error および code プロパティの NetConnection.Admin.CommandFailed、またはより具体的な値 (存在する場合) を含む情報オブジェクトを応答として送信します。失敗の原因について説明する文字列を含む description プロパティが含まれているオブジェクトもあります。

説明

閉じられたストリーム、アプリケーションインスタンス、非永続的共有オブジェクトなど、使用されなくなったすべてのサーバーリソースを収集し、削除します。呼び出しでこの処理に要する時間は、約 1 秒です。

この操作を実行できるのは、サーバー管理者のみです。

getActiveInstances()

使用できるバージョン

- Flash Player 6
- Flash Communication Server MX 1.0

シンタックス

```
getActiveInstances([processID])
```

パラメータ

processID Flash Media Server のコアプロセスのプロセス識別子を示す数字。このパラメータはオプションです。

戻り値

呼び出しが成功すると、サーバーは、`level` プロパティの `status`、`code` プロパティの `NetConnection.Call.Success`、および、サーバーまたは指定されたプロセスで実行中のすべてのアプリケーションインスタンスの名前を含む文字列の配列である `data` プロパティを含む情報オブジェクトを応答として送信します。

呼び出しが失敗すると、サーバーは、`level` プロパティの `error` および `code` プロパティの `NetConnection.Call.Failed`、またはより具体的な値 (存在する場合) を含む情報オブジェクトを応答として送信します。失敗の原因について説明する文字列を含む `description` プロパティが含まれているオブジェクトもあります。

説明

サーバー全体 (渡された *processID* がいない場合)、または *processID* で指定されたプロセスで実行中のすべてのアプリケーションインスタンスを含む文字列の配列を返します。

getAdaptors()

使用できるバージョン

- Flash Player 6
- Flash Communication Server MX 1.0

シンタックス

```
getAdaptors()
```

パラメータ

なし

戻り値

呼び出しが成功すると、サーバーは、`level` プロパティの `status`、`code` プロパティの `NetConnection.Call.Success`、および、すべてのアダプタの名前を含む文字列の配列である `data` プロパティを含む情報オブジェクトを応答として送信します。

説明

定義されているアダプタの配列を返します。このコマンドを実行できるのは、サーバー管理者のみです。

getAdminContext()

使用できるバージョン

- Flash Player 6
- Flash Communication Server MX 1.0

シンタックス

```
getAdminContext([adminName][, adaptorName][, vhostName])
```

パラメータ

adminName 管理者の名前。

adaptorName 管理者の検索を実行する代替アダプタ (`_defaultRoot_` 以外)。指定しない場合は、`_defaultRoot_` が使用されます。

vhostName 仮想ホスト管理者の検索を実行する代替の仮想ホスト (`_defaultVHost_` 以外)。指定しない場合は、サーバーレベルの管理者が使用されます。

戻り値

呼び出しが成功すると、サーバーは、`level` プロパティの `status`、`code` プロパティの `NetConnection.Call.Success`、およびオブジェクトである `data` プロパティを含む情報オブジェクトを応答として送信します。`data` オブジェクトには、以下のプロパティが含まれます。

プロパティ	説明
<code>admin_type</code>	管理者のタイプを示す文字列 (<code>server</code> または <code>vhost</code>)。
<code>adaptor</code>	ユーザーが管理者となっているアダプタの名前を示す文字列。
<code>vhost</code>	ユーザーが管理者となっている仮想ホストの名前を示す文字列。
<code>connected</code>	このプロパティは使用されなくなりました。したがって、常に <code>true</code> が返されます。

説明

指定したユーザーの管理権限、ユーザーが接続しているアダプタと仮想ホストの名前、ユーザーが現在 Flash Media Server に接続しているかどうかなど、管理者の管理コンテキストを取得します。

getAdmins()

使用できるバージョン

- Flash Player 6
- Flash Media Server 2

シンタックス

`getAdmins(adaptorName)`

パラメータ

adaptorName 仮想ホスト管理者の検索を実行する代替アダプタ (`_defaultRoot_` 以外)。指定しない場合は、`_defaultRoot_` が使用されます。

戻り値

呼び出しが成功すると、サーバーは、`level` プロパティの `status`、`code` プロパティの `NetConnection.Call.Success`、および配列である `data` プロパティを含む情報オブジェクトを応答として送信します。`data` オブジェクトには、以下のプロパティが含まれます。

プロパティ	説明
<code>server_admins</code>	管理者名の配列。
<code>admin name</code>	管理者名の文字列。
<code>vhost_admins</code>	仮想ホスト名の配列。
<code>vhost</code>	ユーザーが管理者となっている仮想ホストの名前を示す文字列。

説明

Flash Media Server のすべての管理者の配列を返します。返されるデータには、最上位レベルのグループが2つ(サーバーレベルの管理者および仮想ホストレベルの管理者)含まれます。サーバーレベルの管理者のグループは、名前の単純なリストです。仮想ホスト管理者は仮想ホスト別に分けられ、各仮想ホストについて管理者名がリストされます。

getApps()

使用できるバージョン

- Flash Player 6
- Flash Communication Server MX 1.0

シンタックス

`getApps()`

パラメータ

なし

戻り値

呼び出しが成功すると、サーバーは、`level` プロパティの `status`、`code` プロパティの `NetConnection.Call.Success`、および、インストールされているすべてのアプリケーションの名前を含む文字列の配列である `data` プロパティを含む情報オブジェクトを応答として送信します。

呼び出しが失敗すると、サーバーは、`level` プロパティの `error` および `code` プロパティの `NetConnection.Call.Failed`、またはより具体的な値（存在する場合）を含む情報オブジェクトを応答として送信します。失敗の原因について説明する文字列を含む `description` プロパティが含まれているオブジェクトもあります。

説明

インストールされているすべてのアプリケーションの名前を含む文字列の配列を返します。

関連項目

[getActiveInstances\(\)](#)、[getAppStats\(\)](#)、[getInstanceStats\(\)](#)

getAppStats()

使用できるバージョン

- Flash Player 6
- Flash Communication Server MX 1.0

シンタックス

```
getAppStats(app_name)
```

パラメータ

app_name パフォーマンスデータを必要とするアプリケーションの名前を含む文字列。

戻り値

呼び出しが成功すると、サーバーは、`level` プロパティの `status`、`code` プロパティの `NetConnection.Call.Success`、およびオブジェクトである `data` プロパティを含む情報オブジェクトを応答として送信します。`data` オブジェクトには、以下のプロパティが含まれます。

プロパティ	説明
<code>accepted</code>	アプリケーションが許可した接続の合計数を示す数字。
<code>bytes_in</code>	アプリケーションが読み取った合計バイト数を示す数字。

プロパティ	説明
bytes_out	アプリケーションが書き込んだ合計バイト数を示す数字。
connected	現在アクティブな接続の合計数を示す数字。
launch_time	アプリケーションの開始時間を示す ActionScript の Date オブジェクト。
msg_dropped	アプリケーションがドロップしたメッセージの合計数を示す数字。
msg_in	アプリケーションが処理したメッセージの合計数を示す数字。
msg_out	アプリケーションから送信されたメッセージの合計数を示す数字。
normal_connects	通常の接続の合計数を示す数字。
virtual_connects	リモートエッジによる接続の合計数を示す数字。
group_connects	接続されるリモートエッジの合計数を示す数字。
service_connects	サービスのための接続の合計数を示す数字。
service_requests	要求されたサービスの合計数を示す数字。
admin_connects	管理者による接続の合計数を示す数字。
debug_connects	デバッグのための接続の合計数を示す数字。
rejected	アプリケーションが拒否した接続要求の合計数を示す数字。
total_connects	アプリケーション開始時以降のアプリケーションへのソケット接続の合計数を示す数字。
total_disconnects	アプリケーション開始時以降のアプリケーションの接続解除の合計数を示す数字。
total_instances_loaded	アプリケーション開始時以降にロードされたインスタンスの合計数を示す数字。 このプロパティは、ロードされたアクティブなインスタンスの合計数ではありません。ロードされたアクティブなインスタンスの数を取得するには、total_instances_loaded の値から total_instances_unloaded の値を引きます。
total_instances_unloaded	アプリケーション開始時以降にアンロードされたインスタンスの合計数を示す数字。
up_time	アプリケーションの実行時間 (秒) を示す数字。

呼び出しが失敗すると、サーバーは、level プロパティの error および code プロパティの NetConnection.Call.Failed、またはより具体的な値 (存在する場合) を含む情報オブジェクトを応答として送信します。失敗の原因について説明する文字列を含む description プロパティが含まれているオブジェクトもあります。

説明

指定したアプリケーションのすべてのインスタンスの包括的なパフォーマンスデータを取得します。

関連項目

[getApps\(\)](#)、[getInstanceStats\(\)](#)

getConfig()

使用できるバージョン

- Flash Player 6
- Flash Communication Server MX 1.0

シンタックス

```
getConfig(key [,scope])
```

パラメータ

key 情報を取得する設定キーを指定する文字列。

キーは、スラッシュ (/) で区切られたサブキーのリストとして指定されます。最初のサブキーは、目的の設定キーを含む XML 設定ファイルを指定します。2 番目以降のサブキーは、XML 設定ファイルに関連したタグに対応します。したがって、サブキーの階層と名前は、XML ファイルのタグと一致することになります。

Flash Media Server には、Server.xml、Adaptor.xml、Vhost.xml、および Application.xml という 4 つのサーバー設定ファイルが含まれています。ユーザーに与えられた権限によっては、以下のリストに示すように、これらのファイルすべての設定キーを取得できます。

- Server.xml ファイルでは、最初のサブキーとして、Admin または Server を指定します。2 番目以降のすべてのキーは、Server.xml ファイルの Admin または Server タグに関連したタグに対応します。

Server タグの設定キーを表示できるのは、サーバー管理者のみです。

仮想ホスト管理者は、自分の仮想ホストについてのみ Admin タグの設定キーを表示できます。一部の機密性のある情報については、表示できない場合があります。たとえば、自分の仮想ホストの他の管理者名は表示できますが、これらの管理者のパスワードや権限の設定は表示できません。

- Adaptor.xml ファイルでは、最初のサブキーとして Adaptor:adaptor_name (adaptor_name はアダプタ名) を指定します。2 番目以降のすべてのキーは、Adaptor.xml ファイルの Adaptor タグに関連したタグに対応します。
- Vhost.xml では、最初のサブキーとして Adaptor:adaptor_name/VirtualHost:vhost_name (adaptor_name はアダプタ名、vhost_name は仮想ホスト名) を指定します。2 番目以降のすべてのキーは、Vhost.xml ファイルの VirtualHost タグに関連したタグに対応します。

- 管理サーバーにログオンしたときに接続した仮想ホスト上で実行中のアプリケーションの Application.xml ファイルについては、最初のサブキーとして `Application:app_name` (`app_name` はアプリケーション名) を指定します。

別の仮想ホストで実行中のアプリケーションの Application.xml ファイルのキーを取得するには、完全なキー (`Adaptor:adaptor_name/VirtualHost:vhost_name/Application:app_name`) を指定します。`scope` パラメータも指定する必要があります。

デフォルトの Application.xml ファイルを取得するには、コロン (:) と属性 `app_name` を付けずに「Application」と指定します。

`scope` 文字列。Server.xml ファイル、Adaptor.xml ファイル、または Vhost.xml ファイルの設定キーを取得するには、スラッシュ (/) を指定します。

Flash Media Server にログオンしたときに接続した仮想ホスト上で実行中のアプリケーションの Application.xml ファイルの設定キーを取得するには、このパラメータを削除します。

ヒント	接続しているアダプタまたは仮想ホストを確認するには、 <code>getAdminContext</code> メソッドを使用します。
-----	---

戻り値

呼び出しが成功すると、サーバーは、`level` プロパティの `status`、`code` プロパティの `NetConnection.Call.Success`、および指定したタグの XML 文字列である `data` プロパティを含む情報オブジェクトを応答として送信します。

呼び出しが失敗すると (つまり、指定した設定キーが検出されない場合)、サーバーは空の文字列を返します。

説明

この API は使用されなくなりました。代わりに、`getConfig2()` を使用してください。指定した設定ファイルの指定した設定キーの情報を取得します。Flash Media Server には、Server.xml、Adaptor.xml、Vhost.xml、および Application.xml という 4 つの設定ファイルが含まれています。

仮想ホスト管理者は、自分の仮想ホストの Vhost.xml ファイルと Application.xml ファイルの設定キーを表示できます。

Server.xml および Adaptor.xml ファイルの設定キーの大部分は、サーバー管理者のみが表示できます。

XML 設定ファイルについては、『Flash Media Server 管理ガイド』を参照してください。

ヒント	XML ツリーの同じレベルに同じ名前の XML タグが共存することも可能です。設定ファイル内では、XML タグの名前属性を使用して、このようなタグを区別する必要があります。たとえば、複数の <code>VirtualHost</code> タグについては、 <code><VirtualHost name="www.redpin.com"></VirtualHost></code> という形式を使用します。 <code>getConfig</code> コマンドを呼び出して設定のサブキーを指定するときは、タグ名、コロン、正しい <code>name</code> 属性の順に定義することによって必要なタグを指定することができます (例、 <code>Admin/Adaptor:_defaultRoot_/VirtualHost:www.redpin.com.</code>)。
-----	--

例

次の例では、4つのXMLファイルそれぞれについて、設定キーの取得方法を示します。

```
// サーバーへの接続を確立
nc = newNetConnection();
nc.connect("rtmp://localhost:1111/admin", "JLee", "xml472dy");

// 仮想ホスト管理者は、Server.xml でキーを検索
key = "Admin/Server/UserList/User:JLee/Password";
nc.call("getConfig", new onGetConfig(), key, "/");

// サーバー管理者は、Server.xml でキーを検索
key = "Server/LicenseInfo";
nc.call("getConfig", new onGetConfig(), key, "/");

// Adaptor.xml でキーを検索
key = "Adaptor:_defaultRoot_/HostPortList/HostPort";
nc.call("getConfig", new onGetConfig(), key, "/");

■ // Vhost.xml でキーを検索
key = "Adaptor:_defaultRoot_/VirtualHost:_defaultVhost_/RecordAccessLog";
nc.call("getConfig", new onGetConfig(), key, "/");

// 管理サーバーにログオンしたときに接続した
// 仮想ホスト上のアプリケーションの Application.xml でキーを検索。
// 直前のサブキーおよび 2 番目のパラメータ "/" は必須ではない。
key = "Application:FinanceApp/RecordAppLog";
nc.call("getConfig", new onGetConfig(), key);

// 別の仮想ホスト上のアプリケーションの Application.xml でキーを検索
key = "Adaptor:_defaultRoot_/VirtualHost:www.redpin.com/Application:ChatApp/RecordAppLog";
nc.call("getConfig", new onGetConfig(), key, "/");
```

関連項目

[getAdminContext\(\)](#)、[setConfig\(\)](#)

getConfig2()

使用できるバージョン

- Flash Player 6
- Flash Media Server 2

シンタックス

```
getConfig2(key, scope)
```

パラメータ

key 情報を取得する設定キーを指定する文字列。

キーは、スラッシュ (/) で区切られたサブキーのリストとして指定されます。最初のサブキーは、目的の設定キーを含む XML 設定ファイルを指定します。2 番目以降のサブキーは、XML 設定ファイルに関連したタグに対応します。したがって、サブキーの階層と名前は、XML ファイルのタグと一致することになります。同一名で同一の親を持つタグが複数存在する場合は、"name" 属性を指定することによって、これらのタグを区別できます。*key* パラメータでは、コロンで区切ったタグ名に "name" 属性を追加します。指定したタグがリーフノードである場合は、そのタグデータが返されます。指定したタグがリーフノードでない場合は、タグ全体が XML 文字列として返されます。

scope key パラメータに指定された設定タグを検索する設定ファイルを指定する文字列。

Flash Media Server には、Server.xml、Users.xml、Logger.xml、Adaptor.xml、Vhost.xml、および Application.xml という 6 つのサーバー設定ファイルが含まれています。ユーザーに与えられた権限によっては、以下のリストに示すように、これらのファイルすべての設定キーを取得できます。

- "/" は、Server.xml を指定します。
- "Users" は、サーバー管理者用の Users.xml を指定します。
- "Logger" は、Logger.xml を指定します。
- "Adaptor:<adaptor_name>" は、Adaptor.xml を指定します。<adaptor_name> はアダプタ名です。このファイルにアクセスできるのは、サーバー管理者の特権を持つユーザーのみです。<adaptor_name> が、呼び出し側が接続しているアダプタの名前でない場合は、呼び出しは失敗します。
- "Adaptor:<adaptor_name>/VHost:<vhost_name>" は、VHost.xml を指定します。<vhost_name> は、当該の仮想ホスト名です。呼び出し側が接続しているアダプタの名前が <adaptor_name> ではない場合や、呼び出し側が接続している仮想ホストの名前が <vhost_name> がない場合は、呼び出しは失敗します。
- "Adaptor:<adaptor_name>/VHost:<vhost_name>/Users.xml" は、仮想ホスト管理者の Users.xml を指定します。
- "Adaptor:<adaptor_name>/VHost:<vhost_name>/App[:<app_name>]" は、Application.xml を指定します。<app_name> が指定されていない場合は、デフォルトの Application.xml ファイルが使用されます。指定されている場合は、指定したアプリケーション用の Application.xml が使用されます。指定されたアプリケーションが定義されていない場合や、そのアプリケーション用の Application.xml ファイルがない場合は、呼び出しは失敗します。

ヒント

接続しているアダプタまたは仮想ホストを確認するには、`getAdminContext()` メソッドを呼び出します。

戻り値

呼び出しが成功すると、サーバーは、level プロパティの status、code プロパティの NetConnection.Call.Success、および指定したタグの XML 文字列である data プロパティを含む情報オブジェクトを応答として送信します。

指定したタグがリーフノードである場合は、タグデータが返されます。指定したタグがリーフノードでない場合は、タグ自体が返されます。たとえば、必要なタグが <foo>bar</foo> である場合は、"bar" が返されます。ただし、必要なタグに子タグ (<foo><bar>foobar</bar></foo>) が含まれる場合は、"<foo><bar>foobar</bar></foo>" が返されます。

呼び出しが失敗すると (つまり、指定した設定キーが検出されない場合)、サーバーは空の文字列を返します。

説明

指定した設定ファイルの指定した設定キーの情報を取得します。Flash Media Server には、Server.xml、Users.xml、Logger.xml、Adaptor.xml、Vhost.xml、および Application.xml という 6 つのサーバー設定ファイルが含まれています。

仮想ホスト管理者は、自分の仮想ホストの Vhost.xml ファイルと Application.xml ファイルの設定キーを表示できます。

Server.xml および Adaptor.xml ファイルの設定キーの大部分は、サーバー管理者のみが表示できます。

XML 設定ファイルについては、『Flash Media Server 管理ガイド』を参照してください。

ヒント	XML ツリーの同じレベルに同じ名前前の XML タグが共存することも可能です。設定ファイル内では、XML タグの名前属性を使用して、このようなタグを区別する必要があります。たとえば、複数の VirtualHost タグについては、<VirtualHost name="www.redpin.com"></VirtualHost> という形式を使用します。getConfig コマンドを呼び出して設定のサブキーを指定するときは、タグ名、コロン、正しい name 属性の順に定義することによって必要なタグを指定することができます (例、Admin/Adaptor:_defaultRoot_/VirtualHost:www.redpin.com.)。
-----	---

例

次の例では、さまざまな設定ファイルから XML データを取得します。

```
tSocket = new NetConnection();
tSocket.connect("rtmp://localhost/admin", "user", "password");
```

```
// Server.xml でキーを検索
key = "Server/LicenseInfo";
tSocket.call("getConfig2", new onGetConfig(), key, "/");
```

```
// Adaptor.xml でキーを検索
key = "HostPortList/HostPort";
scope = "Adaptor:_defaultRoot_";
tSocket.call("getConfig2", new onGetConfig(), key, scope);
```

```
// Vhost.xml でキーを検索
key = "AppsDir";
scope = "Adaptor:_defaultRoot_/VHost:_defaultVHost_";
tSocket.call("getConfig2", new onGetConfig(), key, scope);

// アプリケーション "foo" の Application.xml でキーを検索
key = "Process/Scope";
scope = "Adaptor:_defaultRoot_/VHost:_defaultVHost_/App:foo";
tSocket.call("getConfig2", new onGetConfig(), key, scope);

// デフォルトの Application.xml でキーを検索
key = "Process/Scope";
scope = "Adaptor:_defaultRoot_/VHost:_defaultVHost_/App";
tSocket.call("getConfig2", new onGetConfig(), key, scope);

// Vhost.xml 全体を返す
key = "";
scope = "Adaptor:_defaultRoot_/VHost:_defaultVHost_";
tSocket.call("getConfig2", new onGetConfig(), key, scope);
```

関連項目

[getAdminContext\(\)](#)、[setConfig2\(\)](#)

getGroupMembers()

使用できるバージョン

- Flash Player 6
- Flash Media Server 2

シンタックス

```
getGroupMembers(app_instance, groupNumber)
```

パラメータ

app_instance グループが接続するアプリケーションインスタンスの名前を含む文字列 (*application_name/instance_name* の形式)。アプリケーションのデフォルトのインスタンスについてのパフォーマンス統計を取る場合でも、アプリケーション名とインスタンス名の両方をスラッシュ (/) で区切って指定する必要があります。

たとえば、アプリケーション ChatApp のデフォルトのインスタンスを指定する場合は、「ChatApp/_defInst_」と指定します。

groupNumber グループのクライアント ID を示す数字。

戻り値

呼び出しが成功すると、サーバーは、`level` プロパティの `status`、`code` プロパティの `NetConnection.Call.Success`、および、このグループから接続するすべてのクライアントのクライアント ID を含む配列である `data` プロパティを含む情報オブジェクトを応答として送信します。

説明

特定グループに属するメンバーのリストを返します。グループ接続は、リモートのエッジサーバーからオリジンサーバーへのマルチブックス接続です。各グループ接続は、このサーバーのプロキシとして稼動している別の Flash Media Server に接続しているユーザーが少なくとも 1 人存在することを示します。

関連項目

[getGroups\(\)](#)、[getGroupStats\(\)](#)

getGroupStats()

使用できるバージョン

- Flash Player 6
- Flash Media Server 2

シンタックス

```
getGroupStats(app_instance, groupName)
```

パラメータ

app_instance グループが接続するアプリケーションインスタンスの名前を含む文字列 (*application_name/instance_name* の形式)。アプリケーションのデフォルトのインスタンスについてのパフォーマンス統計を取る場合でも、アプリケーション名とインスタンス名の両方をスラッシュ (/) で区切って指定する必要があります。

たとえば、アプリケーション `ChatApp` のデフォルトのインスタンスを指定する場合は、「`ChatApp/_defInst_`」と指定します。

groupName グループのクライアント ID を示す数字。

戻り値

呼び出しが成功すると、サーバーは、level プロパティの status、code プロパティの NetConnection.Call.Success、およびクライアントのパフォーマンスデータを含むオブジェクトである data プロパティを含む情報オブジェクトを応答として送信します。data オブジェクトには、以下のプロパティが含まれます。

プロパティ	説明
connect_time	アプリケーションがサーバーに接続した時間を示す ActionScript の Date オブジェクト。
protocol	クライアントがサーバーに接続するために使用するプロトコルを示す文字列 (rtmp または rtmpt)。
msg_in	アプリケーションが処理したメッセージの合計数を示す数字。
msg_out	アプリケーションから送信されたメッセージの合計数を示す数字。
msg_dropped	アプリケーションがドロップしたメッセージの合計数を示す数字。
bytes_in	アプリケーションが読み取った合計バイト数を示す数字。
bytes_out	アプリケーションが書き込んだ合計バイト数を示す数字。
msg_queue	クライアントメッセージキューの統計を示すオブジェクト。
total_queues	クライアントに対するキューの合計数を示す数字。
audio	すべてのオーディオキューに含まれるオーディオメッセージの合計数を示す数字。
video	すべてのビデオキューに含まれるビデオメッセージの合計数を示す数字。
other	"other" キューに含まれるコマンドおよびデータメッセージの合計数を示す数字。
stream_ids	数字の配列 (ストリーム ID)。
members_count	当該グループ接続でマルチプレックス接続を実行しているクライアント数を示す数字。

説明

特定のグループ接続についての統計を取得します。これは、複数の接続についてマルチプレックス接続を実行し、members_count という独自の統計を含んでいる特殊な接続です。グループ接続は、サーバー間でプロキシとして確立されます。

関連項目

[getGroups\(\)](#)、[getGroupMembers\(\)](#)

getGroups()

使用できるバージョン

- Flash Player 6
- Flash Media Server 2

シンタックス

```
getGroups(app_instance)
```

パラメータ

app_instance パフォーマンス統計の対象となるアプリケーションインスタンスの名前を含む文字列 (*application_name/instance_name* の形式) アプリケーションのデフォルトのインスタンスについてのパフォーマンス統計を取る場合でも、アプリケーション名とインスタンス名の両方をスラッシュ (/) で区切って指定する必要があります。

たとえば、アプリケーション ChatApp のデフォルトのインスタンスを指定する場合は、「ChatApp/_defInst_」と指定します。

戻り値

呼び出しが成功すると、サーバーは、level プロパティの status、code プロパティの NetConnection.Call.Success、および、このアプリケーションに接続するすべてのグループのクライアント ID の配列である data プロパティを含む情報オブジェクトを応答として送信します。

説明

特定のアプリケーションインスタンスへのグループ接続のリストを返します。グループ接続は、リモートのエッジサーバーからオリジンサーバーへのマルチプレックス接続です。各グループ接続は、このサーバーのプロキシとして稼働している別の Flash Media Server に接続しているユーザーが少なくとも 1 人存在することを示します。

関連項目

[getGroupMembers\(\)](#)、[getGroupStats\(\)](#)

getInstanceStats()

使用できるバージョン

- Flash Player 6
- Flash Communication Server MX 1.0

シンタックス

```
getInstanceStats(app_instance)
```

パラメータ

`app_instance` パフォーマンス統計の対象となるアプリケーションインスタンスの名前を含む文字列 (`application_name/instance_name` の形式) アプリケーションのデフォルトのインスタンスについてのパフォーマンス統計を取る場合でも、アプリケーション名とインスタンス名の両方をスラッシュ (/) で区切って指定する必要があります。

たとえば、アプリケーション ChatApp のデフォルトのインスタンスを指定する場合は、「ChatApp/_defInst_」と指定します。

戻り値

呼び出しが成功すると、サーバーは、`level` プロパティの `status`、`code` プロパティの `NetConnection.Call.Success`、およびオブジェクトである `data` プロパティを含む情報オブジェクトを応答として送信します。`data` オブジェクトには、以下のプロパティが含まれます。

プロパティ	説明
<code>launch_time</code>	インスタンスの開始時刻を示す ActionScript の Date オブジェクト。
<code>up_time</code>	インスタンスの実行時間 (秒) を示す数字。
<code>msg_in</code>	アプリケーションインスタンスが処理したメッセージの合計数を示す数字。
<code>msg_out</code>	アプリケーションインスタンスが送信したメッセージの合計数を示す数字。
<code>msg_dropped</code>	アプリケーションインスタンスがドロップしたメッセージの合計数を示す数字。
<code>bytes_in</code>	アプリケーションインスタンスが読み込んだ合計バイト数を示す数字。
<code>bytes_out</code>	アプリケーションインスタンスが書き込んだ合計バイト数を示す数字。
<code>accepted</code>	アプリケーションが許可した接続の合計数を示す数字。
<code>rejected</code>	アプリケーションが拒否した接続要求の合計数を示す数字。
<code>connected</code>	現在アクティブな接続の合計数を示す数字。
<code>total_connects</code>	インスタンス開始時以降のアプリケーションインスタンスへのソケット接続の合計数を示す数字。
<code>total_disconnects</code>	インスタンス開始時以降のアプリケーションインスタンスのソケット接続解除の合計数を示す数字。

プロパティ	説明
<code>script</code>	<p>スクリプトエンジンのパフォーマンスデータを含むオブジェクト。以下は、スクリプトオブジェクトのプロパティです。</p> <p><code>time_high_water_mark</code>: スクリプトがイベントの実行に要した最長時間 (秒) を示す数字。</p> <p><code>queue_size</code>: スクリプトエンジンのキューに現在入っているイベントの合計数を示す数字。</p> <p><code>total_processed</code>: スクリプトエンジンが処理したイベントの合計数を示す数字。</p> <p><code>total_process_time</code>: <code>total_processed</code> に示された数のイベント処理に要した時間 (秒) を示す数字。</p> <p><code>queue_high_water_mark</code>: キューに入れられたイベントの最大数を示す数字。</p>
<code>normal_connects</code>	通常の接続の合計数を示す数字。
<code>virtual_connects</code>	リモートエッジによる接続の合計数を示す数字。
<code>group_connects</code>	接続されるリモートエッジの合計数を示す数字。
<code>service_connects</code>	サービスのための接続の合計数を示す数字。
<code>service_requests</code>	要求されたサービスの合計数を示す数字。
<code>admin_connects</code>	管理者による接続の合計数を示す数字。
<code>debug_connects</code>	デバッグのための接続の合計数を示す数字。

呼び出しが失敗すると、サーバーは、`level` プロパティの `error` および `code` プロパティの `NetConnection.Call.Failed`、またはより具体的な値 (存在する場合) を含む情報オブジェクトを応答として送信します。失敗の原因について説明する文字列を含む `description` プロパティが含まれているオブジェクトもあります。

説明

指定したアプリケーションインスタンスについて、パフォーマンスデータを取得します。

特定のスクリプトのパフォーマンス情報のみが必要な場合には、`getScriptStats` メソッドを使用します。

関連項目

[getActiveInstances\(\)](#)、[getAppStats\(\)](#)、[getScriptStats\(\)](#)

getIOStats()

使用できるバージョン

- Flash Player 6
- Flash Communication Server MX 1.0

シンタックス

getIOStats()

パラメータ

なし

戻り値

呼び出しが成功すると、サーバーは、level プロパティの status、code プロパティの NetConnection.Call.Success、およびオブジェクトである data プロパティを含む情報オブジェクトを応答として送信します。data オブジェクトには、以下のプロパティが含まれます。

プロパティ	説明
msg_in	アダプタが処理したメッセージの合計数を示す数字。
msg_out	アダプタから送信されたメッセージの合計数を示す数字。
bytes_in	アダプタが読み取った合計バイト数を示す数字。
bytes_out	アダプタが書き込んだ合計バイト数を示す数字。
reads	システムの読み取り呼び出しの合計数を示す数字。
writes	システムの書き込みの合計数を示す数字。
connected	アダプタへの現在アクティブなソケット接続数を示す数字。
total_connects	アダプタ開始時以降のアダプタへのソケット接続の合計数を示す数字。
total_disconnects	アダプタのソケット接続解除の合計数を示す数字。
msg_dropped	ドロップされたメッセージの合計数を示す数字。
tunnel_bytes_in	トンネリングヘッダーの入力バイト数 (RTMP ペイロードを超えるオーバーヘッド) を示す数字。
tunnel_bytes_out	トンネリングヘッダーの出力バイト数 (逆方向のオーバーヘッド) を示す数字。
tunnel_requests	トンネリング要求の累計数を示す数字。
tunnel_responses	トンネリング応答の累計数を示す数字。
tunnel_idle_requests	ペイロードのないトンネリング要求 (ネットワークチャッターのオーバーヘッド) の数を示す数字。
tunnel_idle_responses	ペイロードのないトンネリング応答 (ネットワークチャッターのオーバーヘッド) の数を示す数字。
normal_connects	通常の接続の合計数を示す数字。
virtual_connects	リモートエッジによる接続の合計数を示す数字。
group_connects	接続されるリモートエッジの合計数を示す数字。

プロパティ	説明
<code>service_connects</code>	サービスのための接続の合計数を示す数字。
<code>service_requests</code>	要求されたサービスの合計数を示す数字。
<code>admin_connects</code>	管理者による接続の合計数を示す数字。
<code>debug_connects</code>	デバッグのための接続の合計数を示す数字。

呼び出しが失敗すると、サーバーは、`level` プロパティの `error` および `code` プロパティの `NetConnection.Call.Failed`、またはより具体的な値 (存在する場合) を含む情報オブジェクトを応答として送信します。失敗の原因について説明する文字列を含む `description` プロパティが含まれているオブジェクトもあります。

説明

接続しているアダプタのネットワーク入出力の特性についての詳細情報を返します。

この操作を実行できるのは、サーバー管理者のみです。

getLicenseInfo()

使用できるバージョン

- Flash Player 6
- Flash Communication Server MX 1.0

シンタックス

`getLicenseInfo()`

パラメータ

なし

戻り値

呼び出しが成功すると、サーバーは、`level` プロパティの `status`、`code` プロパティの `NetConnection.Call.Success`、およびオブジェクトである `data` プロパティを含む情報オブジェクトを応答として送信します。`data` オブジェクトには、以下のプロパティが含まれます。

プロパティ	説明
<code>name</code>	製品名の文字列 (例、Flash Media Server)。
<code>version</code>	バージョン番号の文字列 (例、1.0)。
<code>build</code>	ビルド番号の文字列。
<code>copyright</code>	著作権情報の文字列。

プロパティ	説明
key	Server.xml ファイルに設定されているライセンスキーの文字列。 ライセンスキーが複数ある場合は、すべての有効なライセンスキーをセミコロン (;) で連結した文字列となります。
type	ライセンスタイプを示す数字 ライセンスが複数ある場合は、すべて同じタイプの番号となります。可能な値は以下のとおりです。 0= 商用版 1= エデュケーション版
family	ライセンスファミリーを示す数字。これによって、他のプロパティが決定します。ライセンスが複数ある場合は、すべて同じファミリーの番号となります。可能な値は以下のとおりです。 0= パーソナル版 1= プロフェッショナル版
edition	エディション (シングル、試用、ベータ、無制限など) を示す数字。ライセンスが複数ある場合は、すべて同じエディションの番号となります。
max_connections	許可されているソケット接続の最大数を示す数字。これは、ライセンスファミリーによって決定します。 ライセンスが複数ある場合は、この数字はすべてのライセンスの max_connections の合計値となります。
max_adaptors	サーバーに設定可能なアダプタ (ネットワークカード) の最大数を示す数字。これは、ライセンスファミリーによって決定します。ライセンスファミリーがパーソナル版である場合は、max_adaptors の値は1となります。
max_vhosts	許可されている仮想ホストの最大数を示す数字。これは、ライセンスファミリーによって決定します。ライセンスファミリーがパーソナル版である場合は、max_vhosts の値は1となります。
max_cpu	許可されている CPU の最大数を示す数字。これは、ライセンスファミリーによって決定します。 この数字が1より大きい場合は、複数プロセッサのコンピュータ上でサーバーを実行できます。

プロパティ	説明
<code>max_bandwidth</code>	最大帯域幅 (Mbps) を示す数字。ライセンスが複数ある場合は、この数字はすべてのライセンスの <code>max_bandwidth</code> の合計値となります。
<code>key_details</code>	<p>各ライセンスキーの情報を含む配列。ライセンスキーが複数ある場合は、各ライセンスキーについて配列エレメントが1つ存在します。</p> <p>各配列エレメントは、以下のプロパティを含むオブジェクトです。</p> <p><code>key</code>: <code>Server.xml</code> のライセンスキーのセットを示す文字列。</p> <p><code>type</code>: 数字 (0= 商用版、1= エデュケーション版)。</p> <p><code>family</code>: 数字 (0= パーソナル版、1= プロフェッショナル版)。</p> <p><code>edition</code>: 数字。</p> <p><code>max_connections</code>: ライセンスで許可されているソケット接続の最大数を示す数字。</p> <p><code>max_adaptors</code>: ライセンスで許可されているアダプタ (ネットワークカード) の最大数を示す数字。ライセンスファミリーがアダプタである場合は、この値は1となります。</p> <p><code>max_vhosts</code>: ライセンスで許可されている各アダプタの仮想ホストの最大数を示す数字。ライセンスファミリーがパーソナル版である場合は、この値は1となります。</p> <p><code>max_cpu</code>: 許可されている CPU の最大数を示す数字。この数字が1より大きい場合は、複数プロセッサのコンピュータ上でサーバーを実行できます。</p> <p><code>max_bandwidth</code>: 最大帯域幅 (Mbps) を示す数字。</p> <p><code>product_code</code>: ライセンスファミリーによって決定する数字。</p> <p><code>valid</code>: ブール値 (1= <code>true</code>: ライセンスは有効、0= <code>false</code>: ライセンスは無効)。</p>

呼び出しが失敗すると、サーバーは、`level` プロパティの `error` および `code` プロパティの `NetConnection.Call.Failed`、またはより具体的な値 (存在する場合) を含む情報オブジェクトを応答として送信します。失敗の原因について説明する文字列を含む `description` プロパティが含まれているオブジェクトもあります。

説明

ライセンスで許可されている最大帯域幅、接続、アダプタ、仮想ホスト、および CPU の最大数など、ライセンスに関する完全な情報を取得します。すべてのライセンスについてライセンス情報のファミリーが最初に作成され、続いて各ライセンスの情報が作成されます。

getLiveStreams()

使用できるバージョン

- Flash Player 6
- Flash Communication Server MX 1.0

シンタックス

`getLiveStreams(app_instance)`

パラメータ

app_instance アプリケーションインスタンスの名前を含む文字列 (*application_name/instance_name* の形式)。

戻り値

呼び出しが成功すると、サーバーは、`level` プロパティの `status`、`code` プロパティの `NetConnection.Call.Success`、および、指定したアプリケーションインスタンスにパブリッシュされているすべてのライブストリームの名前を含む文字列の配列である `data` プロパティを含む情報オブジェクトを応答として送信します。

呼び出しが失敗すると、サーバーは、`level` プロパティの `error` および `code` プロパティの `NetConnection.Call.Failed`、またはより具体的な値 (存在する場合) を含む情報オブジェクトを応答として送信します。失敗の原因について説明する文字列を含む `description` プロパティが含まれているオブジェクトもあります。

説明

指定したアプリケーションインスタンスにパブリッシュされているすべてのライブストリームの名前を含む文字列の配列を取得します。

関連項目

[getLiveStreamStats\(\)](#)

getLiveStreamStats()

使用できるバージョン

- Flash Player 6
- Flash Communication Server MX 1.0

シンタックス

`getLiveStreamStats(app_instance, stream_name)`

パラメータ

app_instance アプリケーションインスタンスの名前を含む文字列 (*application_name/instance_name* の形式)。

stream_name ストリームの名前を含む文字列。

戻り値

呼び出しが成功すると、サーバーは、level プロパティの status、code プロパティの NetConnection.Call.Success、およびオブジェクトである data プロパティを含む情報オブジェクトを応答として送信します。data オブジェクトには、以下のプロパティが含まれます。

プロパティ	説明
publisher	<p>パブリッシュする側の統計を示すオブジェクト。オブジェクトには、以下のプロパティが含まれます。</p> <p>name: パブリッシュされたライブストリーム名を示す文字列。</p> <p>time: ストリームがパブリッシュされた時刻を示す Date オブジェクト このプロパティは publish_time と重複しており、後方互換の目的のために存在します。</p> <p>type: パブリッシュする側のストリームのタイプを示す文字列。値は "publishing" となります。</p> <p>client: パブリッシュする側のクライアント ID を示す数字。</p> <p>stream_id: パブリッシュする側のストリーム ID を示す数字。</p> <p>publish_time: ストリームがパブリッシュされた日時を示す Date オブジェクト。</p> <p>client_type: パブリッシュするクライアントのタイプを示す文字列。</p> <p>publish_time: ストリームがパブリッシュされた日時を示す Date オブジェクト。</p>
subscribers	<p>サブスクライブする側の統計を示す配列。配列には、以下のプロパティを含むオブジェクトである subscriber プロパティが含まれます。</p> <p>client: ユーザー ID を示す数字。</p> <p>subscribe_time: ユーザーがストリームをサブスクライブした日時を示す Date オブジェクト。</p>

呼び出しが失敗すると、サーバーは、level プロパティの error および code プロパティの NetConnection.Call.Failed、またはより具体的な値 (存在する場合) を含む情報オブジェクトを応答として送信します。失敗の原因について説明する文字列を含む description プロパティが含まれているオブジェクトもあります。

説明

ライブストリームの詳細情報を返します。

関連項目

[getLiveStreams\(\)](#)

getMsgCacheStats()

使用できるバージョン

- Flash Player 6
- Flash Communication Server MX 1.0

シンタックス

```
getMsgCacheStats()
```

パラメータ

なし

戻り値

呼び出しが成功すると、サーバーは、level プロパティの status、code プロパティの NetConnection.Call.Success、およびオブジェクトである data プロパティを含む情報オブジェクトを応答として送信します。data オブジェクトには、以下のメッセージキャッシュの統計が含まれます。

プロパティ	説明
allocated	割り当てられているメッセージオブジェクトの合計数を示す数字。
reused	使用されているオブジェクトの合計数を示す数字。
size	現在のサイズを示す数字。

説明

サーバーの TCMessage キャッシュの統計を返します。この操作を実行するには、サーバー管理者の特権が必要です。

getNetStreams()

使用できるバージョン

- Flash Player 6
- Flash Communication Server MX 1.0

シンタックス

```
getNetStreams(app_instance)
```

パラメータ

app_instance アプリケーションインスタンスの名前を含む文字列 (*application_name/instance_name* の形式)。

戻り値

呼び出しが成功すると、サーバーは、level プロパティの status、code プロパティの NetConnection.Call.Success、および数字の配列である data プロパティを含む情報オブジェクトを応答として送信します。数字列は、指定したアプリケーションインスタンスに接続中のすべてのネットワークストリームにサーバーが割り当てた ID を示します。

呼び出しが失敗すると、サーバーは、`level` プロパティの `error` および `code` プロパティの `NetConnection.Call.Failed`、またはより具体的な値 (存在する場合) を含む情報オブジェクトを応答として送信します。失敗の原因について説明する文字列を含む `description` プロパティが含まれているオブジェクトもあります。

説明

指定したアプリケーションインスタンスに接続中のすべてのネットワークストリームに対してサーバーが割り当てた ID を示す数字の配列を返します。

関連項目

[getNetStreamStats\(\)](#)

getNetStreamStats()

使用できるバージョン

- Flash Player 6
- Flash Communication Server MX 1.0

シンタックス

```
getNetStreamStats(app_instance, netstream_ID)
```

パラメータ

app_instance アプリケーションインスタンスの名前を含む文字列 (*application_name/instance_name* の形式)。

netstream_ID ネットワークストリームの ID を示す数字、またはネットワークストリームの ID を示す数字の配列。

現在接続しているすべてのネットワークストリームの情報を取得するには、*netstream_ID* パラメータに値 `-1` を指定します。

戻り値

呼び出しが成功すると、サーバーは、`level` プロパティの `status`、`code` プロパティの `NetConnection.Call.Success`、および、ネットワークストリームの統計の配列である `data` プロパティを含む情報オブジェクトを応答として送信します。配列内の各エレメントは、以下のプロパティを含むオブジェクトです。

プロパティ	名前
<code>stream_id</code>	ストリーム ID を示す数字。
<code>name</code>	ストリーム名の文字列。ストリームがアイドル状態の場合は <code>empty</code> となります。

プロパティ	名前
<code>type</code>	ストリームのタイプを示す文字列。可能な値は以下のとおりです。 idle publishing playing live play recorded
<code>client</code>	ユーザー ID を示す数字。
<code>time</code>	ActionScript の Date オブジェクト。可能な値は以下のとおりです。 type の値が idle であるとき、値は 0 になります。 type の値が publishing であるとき、値はストリームがパブリッシュされた時刻となります。 type の値が playing live であるとき、値はストリーム再生が開始した時刻となります。 type の値が play recorded であるとき、値はストリーム再生が開始した時刻となります。

呼び出しが失敗すると、サーバーは、`level` プロパティの `error` および `code` プロパティの `NetConnection.Call.Failed`、またはより具体的な値 (存在する場合) を含む情報オブジェクトを応答として送信します。失敗の原因について説明する文字列を含む `description` プロパティが含まれているオブジェクトもあります。

説明

指定したアプリケーションインスタンスに接続しているネットワークストリームの詳細情報を取得します。

関連項目

[getNetStreams\(\)](#)

getRecordedStreams()

使用できるバージョン

- Flash Player 6
- Flash Media Server 2

シンタックス

```
getRecordedStreams(app_name)
```

パラメータ

app_name アプリケーションまたはアプリケーションインスタンスの名前を含む文字列 (`application_name[/instance_name]` の形式)。

戻り値

呼び出しが成功すると、サーバーは、level プロパティの status、code プロパティの NetConnection.Call.Success、および、すべての記録ストリームの名前の配列である data プロパティを含む情報オブジェクトを応答として送信します。記録ストリームの拡張名が使用されます。名前は keyName?type:streamName の形式でエンコーディングされます。keyName は仮想キー、type はストリームのタイプ (flv、mp3 など)、streamName はストリームのテキスト名を表します。次の表は、プロパティについて説明しています。

プロパティ	名前
streamName	記録ストリームの名前を示す文字列。
type	記録ストリームのタイプを示す文字列。
keyName	記録ストリームの仮想キーを示す文字列。

説明

特定のアプリケーションインスタンスから再生中のすべての記録ストリームの名前を含む配列を返します。

関連項目

[getRecordedStreamStats\(\)](#)

getRecordedStreamStats()

使用できるバージョン

- Flash Player 6
- Flash Media Server 2

シンタックス

```
getRecordedStreamStats(app_instance, stream_name)
```

パラメータ

app_instance アプリケーションインスタンスの名前を含む文字列 (*application_name/instance_name* の形式)。

stream_name ストリームの名前を含む文字列。ストリームの仮想キーやタイプがデフォルトではない場合は、ストリーム名の一部としてエンコーディングする必要があります。キーとタイプは、key?type:name の形式でエンコーディングします (例、on2key?flv:myStream)。

戻り値

呼び出しが成功すると、サーバーは、level プロパティの status、code プロパティの NetConnection.Call.Success、および、次のプロパティを含むオブジェクトである data プロパティを含む情報オブジェクトを応答として送信します。

プロパティ	名前
cache_bytes	記録ストリームのキャッシュされた合計バイト数を示す数字。
cache_segments	記録ストリームのキャッシュされたセグメント数を示す数字。
cache_hits	キャッシュ内の記録ストリームのヒット数を示す数字。
cache_misses	記録ストリームのキャッシュミスの数を示す数字。
modified_time	ファイルの最終変更日を示す ActionScript の Date オブジェクト。
size	記録ファイルのバイト数を示す数字。
length	ファイルの長さ (秒) を示す数字。

説明

記録ストリームの詳細情報を返します。

関連項目

[getRecordedStreams\(\)](#)

getScriptStats()

使用できるバージョン

- Flash Player 6
- Flash Communication Server MX 1.0

シンタックス

`getScriptStats(app_instance)`

パラメータ

app_instance アプリケーションインスタンスの名前を含む文字列 (*application_name/instance_name* の形式)。

戻り値

呼び出しが成功すると、サーバーは、`level` プロパティの `status`、`code` プロパティの `NetConnection.Call.Success`、およびオブジェクトである `data` プロパティを含む情報オブジェクトを応答として送信します。`data` オブジェクトには、以下のプロパティが含まれます。

プロパティ	説明
<code>time_high_water_mark</code>	スクリプトがイベントの実行に要した最長時間 (秒) を示す数字。
<code>queue_size</code>	スクリプトエンジンのキューに現在入っているイベントの合計数を示す数字。
<code>total_processed</code>	スクリプトエンジンが処理したイベントの合計数を示す数字。
<code>total_process_time</code>	<code>total_processed</code> での数のイベント処理に要した時間 (秒) を示す数字。
<code>queue_high_water_mark</code>	キューに入れられたイベントの最大数を示す数字。

呼び出しが失敗すると、サーバーは、`level` プロパティの `error` および `code` プロパティの `NetConnection.Call.Failed`、またはより具体的な値 (存在する場合) を含む情報オブジェクトを応答として送信します。失敗の原因について説明する文字列を含む `description` プロパティが含まれているオブジェクトもあります。

説明

アプリケーションの指定したインスタンスで実行中のスクリプトのパフォーマンスデータを取得します。

getServerStats()

使用できるバージョン

- Flash Player 6
- Flash Communication Server MX 1.0

シンタックス

```
getServerStats()
```

パラメータ

なし

戻り値

呼び出しが成功すると、サーバーは、level プロパティの status、code プロパティの NetConnection.Call.Success、およびオブジェクトである data プロパティを含む情報オブジェクトを応答として送信します。data オブジェクトには、以下のプロパティが含まれます。

プロパティ	説明
launchTime	サーバーが開始した日時を示す ActionScript の Date オブジェクト。
up_time	サーバーの実行時間 (秒) を示す数字。
io	入出力の統計。以下のプロパティを使用したオブジェクトとして返されます。 msg_in: サーバーが処理したメッセージの合計数を示す数字。 msg_out: サーバーから送信されたメッセージの合計数を示す数字。 bytes_in: サーバーが読み取った合計バイト数を示す数字。 bytes_out: サーバーが書き込んだ合計バイト数を示す数字。 reads: システムの読み取り呼び出しの合計数を示す数字。 writes: システムの書き込みの合計数を示す数字。 connected: サーバーへのアクティブなソケット接続の合計数を示す数字。 total_connects: サーバーへのソケット接続の合計数を示す数字。 total_disconnects: サーバーのソケット接続解除の合計数を示す数字。
msg_cache	Flash Media Server メッセージパケットキャッシュの統計。以下のプロパティを使用したオブジェクトとして返されます。 allocated: 割り当てられているメッセージオブジェクトの合計数を示す数字。 reused: 再使用されたオブジェクトの合計数を示す数字。 size: キャッシュサイズ (メッセージパケット数) を示す数字。
memory_usage	数字。Microsoft Windows NT 4.0 では、使用中の物理メモリの最近の 1,000 ページ分の概算の比率を示します。 Windows 2000 または Windows XP では、使用中の合計物理メモリの概算の比率を示します。
cpu_usage	Flash Media Server (システム全体ではない) で使用中の CPU の概算の比率を示す数字。

呼び出しが失敗すると、サーバーは、level プロパティの error および code プロパティの NetConnection.Call.Failed、またはより具体的な値 (存在する場合) を含む情報オブジェクトを応答として送信します。失敗の原因について説明する文字列を含む description プロパティが含まれているオブジェクトもあります。

説明

サーバーの実行時間、入出力およびメッセージキャッシュの統計など、サーバーの状態およびサーバー操作に関する統計を取得します。

この操作を実行できるのは、サーバー管理者のみです。

サーバーの入出力特性の情報のみが必要な場合には、getIOStats メソッドを使用します。

関連項目

[getIOStats\(\)](#)

getServices()

使用できるバージョン

- Flash Player 6
- Flash Media Server 2

シンタックス

```
getServices()
```

パラメータ

なし

戻り値

呼び出しが成功すると、サーバーは、`level` プロパティの `status`、`code` プロパティの `NetConnection.Call.Success`、およびサービス名を含む文字列の配列である `data` プロパティを含む情報オブジェクトを応答として送信します。

説明

現在 Flash Media Server に接続しているすべてのサービスの名前を含む配列を返します。

特別なアプリケーションが接続し、Flash Media Server にサービスを提供する場合があります。Flash Media Server 向けサービスの接続は、現時点では Java または C SDK から提供されます。通常の接続はサービスとして登録できません。各アプリケーションが、これらのサービスの使用を要求すると、要求されたサービスがアプリケーションに提供されます。

getSharedObjects()

使用できるバージョン

- Flash Player 6
- Flash Communication Server MX 1.0

シンタックス

```
getSharedObjects(app_instance)
```

パラメータ

app_instance アプリケーションインスタンスの名前を含む文字列 (*application_name/instance_name* の形式)。

戻り値

呼び出しが成功すると、サーバーは、level プロパティの status、code プロパティの NetConnection.Call.Success、およびオブジェクトである data プロパティを含む情報オブジェクトを応答として送信します。data オブジェクトには、以下のプロパティが含まれます。

プロパティ	説明
persistent	永続的な共有オブジェクトの名前を含む文字列の配列。
volatile	永続的でない共有オブジェクトの名前を含む文字列の配列。

呼び出しが失敗すると、サーバーは、level プロパティの error および code プロパティの NetConnection.Call.Failed、またはより具体的な値 (存在する場合) を含む情報オブジェクトを応答として送信します。失敗の原因について説明する文字列を含む description プロパティが含まれているオブジェクトもあります。

存在しないアプリケーションまたはインスタスの名前を指定すると、呼び出しは失敗します。

説明

現在アクティブなすべての共有オブジェクトの名前を取得します。

関連項目

[getSharedObjectStats\(\)](#)

getSharedObjectStats()

使用できるバージョン

- Flash Player 6
- Flash Communication Server MX 1.0

シンタックス

```
getSharedObjectStats(app_instance, object_name, persistence)
```

パラメータ

app_instance アプリケーションインスタスの名前を含む文字列 (*application_name/instance_name* の形式)。

object_name 共有オブジェクトの名前を含む文字列。すべてのアクティブな共有オブジェクトの名前を取得するには、getSharedObjects コマンドを使用します。

persistence ブール値。永続的な共有オブジェクトは true (1)、永続的でない共有オブジェクトは false (0) となります。

戻り値

呼び出しが成功すると、サーバーは、`level` プロパティの `status`、`code` プロパティの `NetConnection.Call.Success`、およびオブジェクトである `data` プロパティを含む情報オブジェクトを応答として送信します。`data` オブジェクトには、以下のプロパティが含まれます。

プロパティ	説明
<code>version</code>	共有オブジェクトのバージョン番号を示す数字。
<code>num_properties</code>	共有オブジェクトのプロパティの合計数を示す数字。
<code>msg_in</code>	共有オブジェクトが受信したメッセージの合計数を示す数字。
<code>msg_out</code>	共有オブジェクトから送信された合計数を示す数字。
<code>total_connects</code>	共有オブジェクトへの接続の合計数を示す数字。
<code>total_disconnects</code>	共有オブジェクトの接続解除の合計数を示す数字。
<code>connected</code>	アクティブなサブスクリブする側の数を示す数字。
<code>resync_depth</code>	再同期化まで保持されるバージョン間の相違の最大値を示す数字。サーバーのバージョン番号とクライアントのバージョン番号の間の相違が <code>resync_depth</code> の値よりも大きい場合は、Flash Media Server からはバージョン間の変更部分のみが送信されます。

呼び出しが失敗すると、サーバーは、`level` プロパティの `error` および `code` プロパティの `NetConnection.Call.Failed`、またはより具体的な値 (存在する場合) を含む情報オブジェクトを応答として送信します。失敗の原因について説明する文字列を含む `description` プロパティが含まれているオブジェクトもあります。

説明

共有オブジェクトの詳細情報を取得します。

関連項目

[getSharedObjects\(\)](#)

getUsers()

使用できるバージョン

- Flash Player 6
- Flash Communication Server MX 1.0

シンタックス

```
getUsers(app_instance)
```

パラメータ

app_instance アプリケーションインスタンスの名前を含む文字列 (*application_name/instance_name* の形式)。

戻り値

呼び出しが成功すると、サーバーは、`level` プロパティの `status`、`code` プロパティの `NetConnection.Call.Success`、および、指定したアプリケーションインスタンスに接続しているユーザーの ID を示す文字列の配列である `data` プロパティを含む情報オブジェクトを応答として送信します。ユーザー ID は、サーバーが割り当てます。

呼び出しが失敗すると、サーバーは、`level` プロパティの `error` および `code` プロパティの `NetConnection.Call.Failed`、またはより具体的な値 (存在する場合) を含む情報オブジェクトを応答として送信します。失敗の原因について説明する文字列を含む `description` プロパティが含まれているオブジェクトもあります。

説明

指定したアプリケーションインスタンスに接続しているすべてのユーザーに対してサーバーが割り当てた ID を示す文字列の配列を取得します。

関連項目

[getUserStats\(\)](#)

getUserStats()

使用できるバージョン

- Flash Player 6
- Flash Communication Server MX 1.0

シンタックス

```
getUserStats(app_instance, user_ID)
```

パラメータ

app_instance アプリケーションインスタンスの名前を含む文字列 (*application_name/instance_name* の形式)。

user_ID サーバーが割り当てたユーザー ID を含む文字列。`getUsers` `getUsers` コマンドを使用すると、ユーザー ID を取得できます。

戻り値

呼び出しが成功すると、サーバーは、level プロパティの status、code プロパティの NetConnection.Call.Success、およびオブジェクトである data プロパティを含む情報オブジェクトを応答として送信します。data オブジェクトには、以下のプロパティが含まれます。

プロパティ	説明
connect_time	アプリケーションの指定したインスタンスにユーザーが接続している時間 (秒) を示す ActionScript の Date オブジェクト。
msg_in	ユーザーが処理したメッセージの合計数を示す数字。
msg_out	ユーザーから送信されたメッセージの合計数を示す数字。
msg_dropped	ユーザーがドロップしたメッセージの合計数を示す数字。
bytes_in	ユーザーが読み取った合計バイト数を示す数字。
bytes_out	ユーザーが書き込んだ合計バイト数を示す数字。
msg_queue	クライアントのメッセージキューの統計を含むオブジェクト msg_queue には、以下のプロパティが含まれます。 total_queues: クライアント用のキューの合計数。 audio: すべてのオーディオキューに含まれるオーディオメッセージの合計数。 video: すべてのビデオキューに含まれるビデオメッセージの合計数。 other: other キューに含まれるコマンドおよびデータメッセージの合計数。
protocol	クライアントがサーバー (RTMP または RTMPT) に接続するために使用するプロトコルを示す文字列。
stream_ids	ストリーム ID を示す数字の配列。

呼び出しが失敗すると、サーバーは、level プロパティの error および code プロパティの NetConnection.Call.Failed、またはより具体的な値 (存在する場合) を含む情報オブジェクトを応答として送信します。失敗の原因について説明する文字列を含む description プロパティが含まれているオブジェクトもあります。

説明

指定したユーザーの接続についてのパフォーマンスデータを取得します。

関連項目

[getUsers\(\)](#)

getVHosts()

使用できるバージョン

- Flash Player 6
- Flash Communication Server MX 1.0

シンタックス

```
getVHosts([adaptorName])
```

パラメータ

adaptorName オプションのパラメータ。アダプタのユーザー名を含む文字です。指定しない場合は、"`_defaultRoot_`"であると見なされます。

戻り値

呼び出しが成功すると、サーバーは、`level` プロパティの `status`、`code` プロパティの `NetConnection.Call.Success`、および、すべての仮想ホストの名前を含む文字列の配列である `data` プロパティを含む情報オブジェクトを応答として送信します。

呼び出しが失敗すると、サーバーは、`level` プロパティの `error` および `code` プロパティの `NetConnection.Call.Failed`、またはより具体的な値 (存在する場合) を含む情報オブジェクトを応答として送信します。失敗の原因について説明する文字列を含む `description` プロパティが含まれているオブジェクトもあります。

説明

指定したアダプタに定義される仮想ホストの配列を返します。このコマンドを実行できるのは、サーバー管理者のみです。

getVHostStats()

使用できるバージョン

- Flash Player 6
- Flash Communication Server MX 1.0

シンタックス

```
getVHostStats([adaptorName, vhostName])
```

パラメータ

adaptorName オプションのパラメータ。アダプタのユーザー名を含む文字列です。指定しない場合は、"`_defaultRoot_`" が使用されます。

`vhostName` オプションのパラメータ。仮想ホストのユーザー名を含む文字列です。指定しない場合は、"`_defaultVHost_`" が使用されます。

戻り値

呼び出しが成功すると、サーバーは、`level` プロパティの `status`、`code` プロパティの `NetConnection.Call.Success`、および、仮想ホストのパフォーマンスデータを含むオブジェクトである `data` プロパティを含む情報オブジェクトを応答として送信します。`data` オブジェクトには、以下のプロパティが含まれます。

プロパティ	説明
<code>msg_in</code>	仮想ホストが処理したメッセージの合計数を示す数字。
<code>msg_out</code>	仮想ホストから送信されたメッセージの合計数を示す数字。
<code>msg_dropped</code>	仮想ホストがドロップしたメッセージの合計数を示す数字。
<code>bytes_in</code>	仮想ホストが読み取った合計バイト数を示す数字。
<code>bytes_out</code>	仮想ホストが書き込んだ合計バイト数を示す数字。
<code>accepted</code>	仮想ホストが許可した接続の合計数を示す数字。
<code>rejected</code>	仮想ホストが拒否した接続要求の合計数を示す数字。
<code>connected</code>	現在アクティブな接続の合計数を示す数字。
<code>total_apps</code>	インスタンスが作成されたアプリケーションの合計数を示す数字。
<code>total_connects</code>	サーバーへの接続の合計数を示す数字。
<code>total_disconnects</code>	サーバーの接続解除の合計数を示す数字。
<code>total_instances_loaded</code>	ロードされたインスタンスの合計数を示す数字。 このプロパティは、ロードされているアクティブなインスタンスの合計数ではありません。ロードされているアクティブなインスタンスの合計数を取得するには、 <code>total_instances_loaded</code> の値から <code>total_instances_unloaded</code> の値を引きます。
<code>total_instances_unloaded</code>	アンロードされたインスタンスの合計数を示す数字。
<code>bw_in</code>	現在の入力帯域幅 (bps) を示す数字。
<code>bw_out</code>	現在の出力帯域幅 (bps) を示す数字。
<code>tunnel_bytes_in</code>	トンネルで読み取られた合計バイト数を示す数字。
<code>tunnel_bytes_out</code>	トンネルで書き込まれた合計バイト数を示す数字。
<code>tunnel_requests</code>	現在の要求数を示す数字。
<code>tunnel_idle_requests</code>	現在アイドル状態の要求数を示す数字。
<code>tunnel_idle_responses</code>	現在アイドル状態の応答数を示す数字。
<code>normal_connects</code>	通常の接続の合計数を示す数字。

プロパティ	説明
<code>virtual_connects</code>	リモートエッジによる接続の合計数を示す数字。
<code>group_connects</code>	接続されるリモートエッジの合計数を示す数字。
<code>service_connects</code>	サービスのための接続の合計数を示す数字。
<code>service_requests</code>	要求されたサービスの合計数を示す数字。
<code>admin_connects</code>	管理者による接続の合計数を示す数字。
<code>debug_connects</code>	デバッグのための接続の合計数を示す数字。

呼び出しが失敗すると、サーバーは、`level` プロパティの `error` および `code` プロパティの `NetConnection.Call.Failed`、またはより具体的な値 (存在する場合) を含む情報オブジェクトを応答として送信します。失敗の原因について説明する文字列を含む `description` プロパティが含まれているオブジェクトもあります。

説明

指定した仮想ホストのすべてのアプリケーションのすべてのインスタンスについて包括的なパフォーマンスデータを返します。このコマンドを実行できるのは、サーバー管理者のみです。

ping()

使用できるバージョン

- Flash Player 6
- Flash Communication Server MX 1.0

シンタックス

`ping()`

パラメータ

なし

戻り値

呼び出しが成功すると、サーバーは、`level` プロパティの `status`、`code` プロパティの `NetConnection.Call.Success`、および `Date` オブジェクトである `timestamp` プロパティを含む情報オブジェクトを応答として送信します。`Date` オブジェクトは、コマンドが実行された時刻を示します。

呼び出しが失敗すると、サーバーは、level プロパティの error および code プロパティの NetConnection.Call.Failed、またはより具体的な値 (存在する場合) を含む情報オブジェクトを応答として送信します。失敗の原因について説明する文字列を含む description プロパティが含まれているオブジェクトもあります。

説明

サーバーが実行していることを確認します。サーバーは状態メッセージを使用して応答します。このコマンドを使用すると、サーバーの状態を確認できます。

reloadApp()

使用できるバージョン

- Flash Player 6
- Flash Communication Server MX 1.0

シンタックス

```
reloadApp( app_name )
```

パラメータ

app_name アプリケーションまたはアプリケーションインスタンスの名前を含む文字列 (*application_name*[/*instance_name*] の形式) デフォルトのアプリケーションインスタンスを再ロードするには、アプリケーション名のみを指定します。

戻り値

呼び出しが成功すると、サーバーは level プロパティの status および code プロパティの NetConnection.Call.Success を含む情報オブジェクトを応答として送信します。

呼び出しが失敗すると、サーバーは、level プロパティの error および code プロパティの NetConnection.Admin.Command.Failed、またはより具体的な値 (存在する場合) を含む情報オブジェクトを応答として送信します。失敗の原因について説明する文字列を含む description プロパティが含まれているオブジェクトもあります。

説明

指定したアプリケーションまたはアプリケーションインスタンスが実行中の場合は終了し、再ロードします。すべてのユーザーとの接続は解除されます。

コマンド実行後、ユーザーはアプリケーションまたはアプリケーションインスタンスに再接続する必要があります。

このコマンドを使用すると、アプリケーションインスタンスをプリロードできます。また、アプリケーションの設定が変更されたり、アプリケーションに関連付けられているスクリプトが変更されたときにこのコマンドを使用すると、アプリケーションインスタンスを再ロードできます。

関連項目

[unloadApp\(\)](#)

removeAdmin()

使用できるバージョン

- Flash Player 6
- Flash Communication Server MX 1.0

シンタックス

```
removeAdmin(admin_name [, scope])
```

パラメータ

admin_name 削除対象となる管理者のユーザー名を含む文字列。

scope オプションのパラメータ。削除する管理者を指定する文字列です。

接続している仮想ホストから仮想ホスト管理者を削除するには、このパラメータを削除します 別の仮想ホストから仮想ホスト管理者を削除するには、仮想ホストを *adaptor_name/virtual_hostname* の形式で指定します。

サーバー管理者を削除するには、「server」と指定します。

戻り値

呼び出しが成功すると、サーバーは `level` プロパティの `status` および `code` プロパティの `NetConnection.Call.Success` を含む情報オブジェクトを応答として送信します。

呼び出しが失敗すると、サーバーは、`level` プロパティの `error` および `code` プロパティの `NetConnection.Admin.Command.Failed`、またはより具体的な値 (存在する場合) を含む情報オブジェクトを応答として送信します。失敗の原因について説明する文字列を含む `description` プロパティが含まれているオブジェクトもあります。

指定した管理者が存在しない場合は、呼び出しは失敗します。

説明

システムから管理者を削除します。指定したパラメータに応じて、サーバー管理者または仮想ホスト管理者を削除できます。

システムから管理者を削除できるのは、サーバー管理者のみです。

例

次の例では、`removeAdmin` メソッドの使用法を示します。

```
/* サーバー管理者 DYoung を削除 */
nc.call("removeAdmin", new onRemoveAdmin(), "DYoung", "server");

/* 仮想ホスト管理者 LPark を削除 */
nc.call("removeAdmin", new onRemoveAdmin(), "LPark");

/* 仮想ホスト tree.oak.com から仮想ホスト管理者 JGarcia を削除 */
nc.call("removeAdmin", new onRemoveAdmin(), "JGarcia", "_defaultRoot/" +
tree.oak.com");
```

関連項目

[addAdmin\(\)](#)

removeApp()

使用できるバージョン

- Flash Player 6
- Flash Communication Server MX 1.0

シンタックス

```
removeApp(app_name)
```

パラメータ

app_name 削除するアプリケーション名またはアプリケーションインスタンス名を含む文字列 (*application_name*[/*instance_name*] の形式)。

戻り値

呼び出しが成功すると、サーバーは `level` プロパティの `status` および `code` プロパティの `NetConnection.Call.Success` を含む情報オブジェクトを応答として送信します。

呼び出しが失敗すると、サーバーは、`level` プロパティの `error` および `code` プロパティの `NetConnection.Admin.Command.Failed`、またはより具体的な値 (存在する場合) を含む情報オブジェクトを応答として送信します。失敗の原因について説明する文字列を含む `description` プロパティが含まれているオブジェクトもあります。

説明

仮想ホストから指定したアプリケーションまたはアプリケーションインスタンスを削除します。最初に、指定したアプリケーションのすべてのインスタンスがアンロードされ、続いて、アプリケーションのディレクトリが仮想ホストから削除されます。アプリケーションインスタンスを指定する場合は、対象となるインスタンスのみがアンロードされて削除され、続いて、このインスタンスのすべてのストリームと共有オブジェクトが削除されます。

例

次の例では、アプリケーション `ChatApp` 全体を削除する方法、および、アプリケーション `ChatApp` のインスタンス `Instance1` のみを指定して削除する方法を示します。

```
nc = new NetConnection();
nc.connect("rtmp://localhost:1111/admin", "JGarcia", "ezcabby1");

/* アプリケーションとすべてのアプリケーションインスタンスを削除 */
nc.call("removeApp", new onRemoveApp(), "ChatApp");

/* 指定したインスタンスのみを削除 */
nc.call("removeApp", new onRemoveApp(), "ChatApp/Instance1");
```

関連項目

[addApp\(\)](#)

removeVHostAlias()

使用できるバージョン

- Flash Player 6
- Flash Media Server 2

シンタックス

```
removeVHostAlias(VHostName, AliasName)
```

パラメータ

VHostName エイリアスを削除する仮想ホストを示す文字列。

AliasName 指定した仮想ホストから削除するエイリアスを示す文字列。

戻り値

呼び出しが成功すると、サーバーは、`level` プロパティの `status` および `code` プロパティの `NetConnection.Call.Success` を含む情報オブジェクトを応答として送信します。

説明

仮想ホストからエイリアスを削除します。エイリアスは、Flash Media Server 接続が確立されるときにターゲットとして使用される仮想ホストの代替名です。エイリアスを削除した後に確立される接続では、この名前は使用できなくなります。

関連項目

[addVHostAlias\(\)](#)

restartVHost()

使用できるバージョン

- Flash Player 6
- Flash Communication Server MX 1.0

シンタックス

```
restartVHost([scope])
```

パラメータ

scope オプションのパラメータ。再起動する仮想ホストを指定する文字列です。

接続している仮想ホストを再起動するには、このパラメータを削除します。

別の仮想ホストを再起動するには、仮想ホストを *adaptor_name/virtual_hostname* の形式で指定します。

戻り値

呼び出しが成功すると、サーバーは *level* プロパティの *status* および *code* プロパティの `NetConnection.Call.Success` を含む情報オブジェクトを応答として送信します。

呼び出しが失敗すると、サーバーは、*level* プロパティの *error* および *code* プロパティの `NetConnection.Admin.Command.Failed`、またはより具体的な値 (存在する場合) を含む情報オブジェクトを応答として送信します。失敗の原因について説明する文字列を含む *description* プロパティが含まれているオブジェクトもあります。

説明

現在実行している仮想ホストを再起動します 仮想ホストを再起動すると、ユーザーとの接続がすべて切断され、現在ロードされているアプリケーションがすべてアンロードされ、この仮想ホストの設定ファイルが再ロードされます。

仮想ホストの設定ファイルを変更する場合にこのコマンドを使用すると、サーバーを再起動せずに仮想ホストを再起動できます。

仮想ホストを再起動するときは、そのつどユーザーは再接続する必要があります このコマンドを使用する前には、接続しているユーザーに再起動することを通知してください。

仮想ホスト管理者が再起動できるのは、自分が接続している仮想ホストのみです。接続している仮想ホスト以外の仮想ホストを再起動できるのは、サーバー管理者のアクセス権限が必要となります。

ヒント

停止している仮想ホストを起動するには、`startVHost()` コマンドを使用します。このコマンドを実行できるのは、サーバー管理者のみです。

例

次の例では、接続している仮想ホストを再起動する呼び出し、および、デフォルトのアダプタ上の仮想ホスト `tree.oak.com` を指定して再起動する呼び出しを示します。

```
/* 接続している仮想ホストを再起動 */  
nc.call("restartVHost", new onRestartVHost());
```

```
/* デフォルトのアダプタ上の仮想ホスト tree.oak.com を再起動 */  
nc.call("restartVHost", new onRestartVHost(), "_defaultRoot_/tree.oak.com");
```

関連項目

[reloadApp\(\)](#)、[startVHost\(\)](#)、[stopVHost\(\)](#)

setConfig()

使用できるバージョン

- Flash Player 6
- Flash Communication Server MX 1.0

シンタックス

```
setConfig(key, key_value, [scope])
```

パラメータ

key 値を変更する設定キーを指定する文字列。

キーは、スラッシュ (/) で区切られたサブキーのリストとして指定されます。最初のサブキーは、目的の設定キーを含む XML 設定ファイルを指定します。2 番目以降のサブキーは、XML 設定ファイルに関連したタグに対応しています。したがって、サブキーの階層と名前は、XML ファイルのタグと一致することになります。

Flash Media Server には、`Server.xml`、`Adaptor.xml`、`Vhost.xml`、および `Application.xml` という 4 つのサーバー設定ファイルが含まれています。ユーザーに与えられた権限によっては、これらのファイルすべての設定キーの値を変更できます。

- Server.xml ファイルでは、最初のサブキーとして、Admin または Server を指定します。2 番目以降のすべてのキーは、Server.xml ファイルの <Admin> または <Server> タグに関連したタグに対応します。

<Server> タグの設定キーを設定できるのは、サーバー管理者のみです。

仮想ホスト管理者は、自分の仮想ホストについてのみ、設定キーを設定できます。一部の機密性のある情報については、設定できない場合があります。たとえば、自分のパスワードは設定できませんが、他の仮想ホスト管理者のパスワードや権限は設定できません。

- Adaptor.xml ファイルでは、最初のサブキーとして `Adaptor:adaptor_name(adaptor_name` はアダプタ名) を指定します。2 番目以降のすべてのキーは、Adaptor.xml ファイルの <Adaptor> タグに関連したタグに対応します。
- Vhost.xml では、最初のサブキーとして `Adaptor:adaptor_name/VirtualHost:vhost_name` (`vhost_name` は仮想ホスト名) を指定します。2 番目以降のすべてのキーは、Vhost.xml ファイルの <VirtualHost> タグに関連したタグに対応します。
- 管理サーバーにログオンしたときに接続した仮想ホスト上で実行中のアプリケーションの Application.xml ファイルについては、最初のサブキーとして `Application:app_name` (`app_name` はアプリケーション名) を指定します。

別の仮想ホストで実行中のアプリケーションの Application.xml ファイルのキーを取得するには、完全なキー (`Adaptor:adaptor_name/VirtualHost:vhost_name/Application:app_name`) を指定します。`scope` パラメータも指定する必要があります (この項目のコード例を参照)。

デフォルトの Application.xml ファイルを取得するには、コロン (:) と属性 `app_name` を付けずに「Application」と指定します。

メモ

サブキーを指定しても、対応するタグが XML ファイルに存在しない場合は、XML ファイルに新しいタグが作成されます。

`key_value` 指定した設定キーの設定に使用される値の文字列。

`scope` 文字列。Server.xml ファイル、Adaptor.xml ファイル、または Vhost.xml ファイルの設定キーを変更するには、このパラメータにスラッシュ (/) を指定します。

Flash Media Server にログオンしたときに接続した仮想ホスト上で実行中のアプリケーションの Application.xml ファイルの設定キーを変更するには、このパラメータを削除します。

ヒント

接続しているアダプタまたは仮想ホストを確認するには、`getAdminContext` メソッドを使用します。

戻り値

呼び出しが成功すると、サーバーは `level` プロパティの `status` および `code` プロパティの `NetConnection.Call.Success` を含む情報オブジェクトを応答として送信します。設定キーの値は変更されません。

呼び出しが失敗すると、サーバーは、level プロパティの error、code プロパティの NetConnection.Call.BadValue、および失敗の原因について説明する文字列を含む description プロパティを含む情報オブジェクトを応答として送信します。

指定した設定キーが検出されない場合、または設定キーの値を変更する権限をユーザーが持っていない場合は、呼び出しは失敗します。

説明

この API は使用されなくなりました。代わりに、[setConfig2\(\)](#) を使用してください。指定した設定ファイルの指定した設定キーの値を変更します。Flash Media Server には、Server.xml、Adaptor.xml、Vhost.xml、および Application.xml という 4 つの設定ファイルが含まれています。

仮想ホスト管理者は、自分の仮想ホストの Vhost.xml ファイルと Application.xml ファイルの設定キーの値を変更できます。

Server.xml および Adaptor.xml ファイルの大部分の設定キーの値は、サーバー管理者のみが変更できます。

XML 設定ファイルについては、『Flash Media Server 管理ガイド』を参照してください。

ヒント

XML ツリーの同じレベルに同じ名前の XML タグが共存することも可能です。設定ファイル内では、XML タグの名前属性を使用して、このようなタグを区別する必要があります。たとえば、複数の仮想ホストについては、`<VirtualHost name="www.redpin.com"></VirtualHost>` という形式を使用します。`<VirtualHost name="www.redpin.com"></VirtualHost>` setConfig コマンドを呼び出して設定のサブキーを指定するとき、タグ名、コロン、正しい名前属性の順に定義することによって、必要なタグを指定することができます。たとえば、以下のように指定できます。

```
Adaptor:_defaultRoot_/VirtualHost:www.redpin.com.
```

例

次の例では、4 つの XML ファイルそれぞれについて、設定キーに新しい値を設定する方法を示します。

```
// サーバーへの接続を確立
nc = newNetConnection();
nc.connect("rtmp://localhost:1111/admin", "LGreen", "123jn098");

// 仮想ホスト管理者は、Server.xml のキーを変更
// ユーザー LGreen のパスワードを strawman28 に設定
key = "Admin/Server/UserList/User:LGreen/Password";
val = "strawman28"
nc.call("setConfig", new onSetConfig(), key, val, "/");

// サーバー管理者は、Server.xml のキーを変更
// Set LicenseInfo を Helloworld に設定
key = "Server/LicenseInfo";
val = "Helloworld";
nc.call("setConfig", new onSetConfig(), key, val, "/");
```

```

// Adaptor.xml のキーを変更
// HostPort を 128.0.0.1:1938 に設定
key = "Adaptor:_defaultRoot_/HostPortList/HostPort";
val = "128.0.0.1:1938";
nc.call("setConfig", new onSetConfig(), key, val, "/");

// Vhost.xml のキーを変更
// RecordAccessLog を true に設定
key = "Adaptor:_defaultRoot_/VirtualHost:_defaultVHost_/RecordAccessLog";
val = "true";
nc.call("setConfig", new onSetConfig(), key, val, "/");

// 管理サーバーにログオンしたときに接続した仮想ホストの
// アプリケーションの Application.xml のキーを変更。
// 直前のサブキーおよび最後の
// パラメータ "/" は必須ではない。
key = "Application:FinanceApp/RecordAppLog";
val = "true";
nc.call("setConfig", new onSetConfig(), key, val);

// 別の仮想ホストの Application.xml のキーを変更。
// アダプタ名、仮想ホスト名、および 2 番目のパラメータは必須。
key = "Adaptor:_defaultRoot_/VirtualHost:www.redpin.com/Application:ChatApp/RecordAppLog";
val = "true";
nc.call("setConfig", new onSetConfig(), key, val, "/");

```

関連項目

[getAdminContext\(\)](#)、[getConfig\(\)](#)

setConfig2()

使用できるバージョン

- Flash Player 6
- Flash Media Server 2

シンタックス

```
setConfig2(key, key_value, .scope)
```

パラメータ

key 値を変更する設定キーを指定する文字列。

キーは、スラッシュ (/) で区切られたサブキーのリストとして指定されます。最初のサブキーは、目的の設定キーを含む XML 設定ファイルを指定します。2 番目以降のサブキーは、XML 設定ファイルに関連したタグに対応しています。したがって、サブキーの階層と名前は、XML ファイルのタグと一致することになります。

Flash Media Server には、Server.xml、Users.xml、Logger.xml、Adaptor.xml、Vhost.xml、および Application.xml という 6 つのサーバー設定ファイルが含まれています。ユーザーに与えられた権限によっては、これらのファイルすべての設定キーの値を変更できます。

key_value 指定した設定キーの設定に使用される値の文字列。指定した値が単純な文字列の場合、指定したタグのタグデータとして設定されます。指定した値が、<foo>bar</foo> のように有効な XML の場合、その XML は指定したタグの子タグとして追加されます。指定した値が空の文字列の場合、指定したタグは削除されます。

scope key パラメータによって指定されたタグに設定されている値を示す文字列。指定した値が単純な文字列の場合、指定したタグのタグデータとして設定されます。指定した値が、<foo>bar</foo> のように有効な XML の場合、その XML は指定したタグの子タグとして追加されます。指定した値が空の文字列の場合、指定したタグは削除されます。

- "/" は、Server.xml を指定します。このファイルの大部分は、サーバー管理者の特権を持つユーザーのみしかアクセスできません。サーバー管理者の権限を必要としないセクションは、呼び出し側が所属する仮想ホストの管理者を含む <VirtualHost> セクションです。
- "Users" は、サーバー管理者用の Users.xml を指定します。
- "Logger" は、Logger.xml を指定します。
- "Adaptor:<adaptor_name>" は、Adaptor.xml を指定します。<adaptor_name> は当該アダプタの名前です。このファイルにアクセスできるのは、サーバー管理者の特権を持つユーザーのみです。呼び出し側が接続しているアダプタの名前が <adaptor_name> でない場合は、呼び出しは失敗します。
- "Adaptor:<adaptor_name>/VHost:<vhost_name>" は、VHost.xml を指定します。<vhost_name> は、当該の仮想ホスト名です。呼び出し側が接続しているアダプタの名前が <adaptor_name> ではない場合や、呼び出し側が接続している仮想ホストの名前が <vhost_name> がない場合は、呼び出しは失敗します。
- "Adaptor:<adaptor_name>/VHost:<vhost_name>/Users" は、仮想ホスト管理者の Users.xml を指定します。
- "Adaptor:<adaptor_name>/VHost:<vhost_name>/App[<app_name>]" は、Application.xml を指定します。<app_name> が指定されていない場合は、デフォルトの Application.xml が使用されます。指定されている場合は、指定したアプリケーション用の Application.xml が使用されます。指定するアプリケーションが設定されていない場合や、そのアプリケーション用の Application.xml がない場合は、呼び出しは失敗します。

ヒント	接続しているアダプタまたは仮想ホストを確認するには、getAdminContext() メソッドを使用します。
-----	---

戻り値

呼び出しが成功すると、サーバーは level プロパティの status および code プロパティの NetConnection.Call.Success を含む情報オブジェクトを応答として送信します。設定キーの値は変更されます。

呼び出しが失敗すると、サーバーは、level プロパティの error、code プロパティの NetConnection.Call.BadValue、および失敗の原因について説明する文字列を含む description プロパティを含む情報オブジェクトを応答として送信します。

指定した設定キーが検出されない場合、または設定キーの値を変更する権限をユーザーが持っていない場合は、呼び出しは失敗します。

説明

指定した設定ファイルの指定した設定キーの値を変更します。Flash Media Server には、Server.xml、Users.xml、Logger.xml、Adaptor.xml、Vhost.xml、および Application.xml という 6 つのサーバー設定ファイルが含まれています。

仮想ホスト管理者は、自分の仮想ホストの Vhost.xml ファイルと Application.xml ファイルの設定キーの値を変更できます。

Server.xml および Adaptor.xml ファイルの大部分の設定キーの値は、サーバー管理者のみが変更できます。

XML 設定ファイルについては、『Flash Media Server 管理ガイド』を参照してください。

ヒント	XML ツリーの同じレベルに同じ名前前の XML タグが共存することも可能です。設定ファイル内では、XML タグの名前属性を使用して、このようなタグを区別する必要があります。たとえば、複数の仮想ホストについては、<VirtualHost name="www.redpin.com"></VirtualHost> という形式を使用します。<VirtualHost name="www.redpin.com"></VirtualHost>.)setConfig コマンドを呼び出して設定のサブキーを指定するとき、タグ名、コロン、正しい名前属性の順に定義することによって、必要なタグを指定することができます。たとえば、以下のよう に指定できます。 Adaptor:_defaultRoot_/VirtualHost:www.redpin.com.
-----	--

例

次の例では、設定キーに新しい値を設定する方法を示します。

```
tSocket = new NetConnection();
tSocket.connect("rtmp://localhost/admin", "user", "password");

// Server.xml にタグデータを設定
key = "Server/LicenseInfo";
val = "SFD150-XXXXX-XXXXX-XXXXX";
tSocket.call("setConfig2", new onSetConfig(), key, val, "/");
```

```

// Adaptor.xml にタグデータを設定
key = "HostPortList/HostPort";
val = ":1935, 80, 443";
scope = "Adaptor:_defaultRoot_";
tSocket.call("setConfig2", new onSetConfig(), key, val, scope);

// Vhost.xml にタグデータを設定
key = "AppsDir";
val = "c:\\applications";
scope = "Adaptor:_defaultRoot_/VHost:_defaultVHost_";
tSocket.call("setConfig2", new onSetConfig(), key, val, scope);

// アプリケーション "foo" の Application.xml にタグデータを設定
key = "Process/Scope";
val = "inst";
scope = "Adaptor:_defaultRoot_/VHost:_defaultVHost_/App:foo";
tSocket.call("setConfig2", new onSetConfig(), key, val, scope);

// デフォルトの Application.xml のタグに子タグを追加
key = "Process";
val = "<Scope>inst</Scope>";
scope = "Adaptor:_defaultRoot_/VHost:_defaultVHost_/App";
tSocket.call("setConfig2", new onSetConfig(), key, val, scope);

// デフォルトの Application.xml からタグを削除
key = "Process";
val = "";
scope = "Adaptor:_defaultRoot_/VHost:_defaultVHost_/App";
tSocket.call("setConfig2", new onSetConfig(), key, val, scope);

```

関連項目

[getAdminContext\(\)](#)、[getConfig2\(\)](#)

startServer()

使用できるバージョン

- Flash Player 6
- Flash Communication Server MX 1.0

シンタックス

```
startServer([start_type])
```

パラメータ

start_type オプションのパラメータ。サーバーの再起動方法を指定する文字列です。

実行していないサーバーを起動する場合は、このパラメータを削除します。実行中のサーバーを終了して再起動する場合は、このパラメータに「restart」と指定します。サーバーが実行中である場合にこのパラメータを削除すると、コマンドは動作しません。

戻り値

呼び出しが成功すると、サーバーは level プロパティの status および code プロパティの NetConnection.Call.Success を含む情報オブジェクトを応答として送信します。

呼び出しが失敗すると、サーバーは、level プロパティの error および code プロパティの NetConnection.Admin.Command.Failed、またはより具体的な値 (存在する場合) を含む情報オブジェクトを応答として送信します。失敗の原因について説明する文字列を含む description プロパティが含まれているオブジェクトもあります。

説明

Flash Media Server サービスを起動、または終了して再起動します。

この操作を実行できるのは、サーバー管理者のみです。

関連項目

[stopServer\(\)](#)

startVHost()

使用できるバージョン

- Flash Player 6
- Flash Communication Server MX 1.0

シンタックス

```
startVHost(vhost_name)
```

パラメータ

vhost_name 起動する仮想ホスト名または新規に有効にする仮想ホスト名 (*[adaptor]/vhost* の形式)。

接続しているアダプタ上で新規に仮想ホストを有効にする場合は、*adaptor/* の部分を削除します。

戻り値

呼び出しが成功すると、サーバーは level プロパティの status および code プロパティの NetConnection.Call.Success を含む情報オブジェクトを応答として送信します。

呼び出しが失敗すると、サーバーは、level プロパティの error および code プロパティの NetConnection.Admin.Command.Failed、またはより具体的な値 (存在する場合) を含む情報オブジェクトを応答として送信します。失敗の原因について説明する文字列を含む description プロパティが含まれているオブジェクトもあります。

説明

停止している仮想ホストを起動、または、新しい仮想ホストを有効にします。

このコマンドを使用すると、サーバーを再起動せずに、ランタイムに新しい仮想ホストを有効にできます。startVHost コマンドを使用して、ディレクトリや設定ファイルを作成したりコピーすることはできません。したがって、サーバーの conf ディレクトリには、新しい仮想ホストのディレクトリと設定ファイルを事前に作成しておく必要があります。仮想ホストの追加方法については、『Flash Media Server 管理ガイド』を参照してください。

startVHost() コマンドを使用できるのは、サーバー管理者のみです。

例

次の例では、現在接続しているアダプタ上で仮想ホスト diamond.world.com を起動する方法、およびアダプタ gem 上で仮想ホスト diamond.world.com を起動する方法を示します。

```
/* 仮想ホスト diamond.world.com を起動 */
nc.call("startVHost", new onStartVHost(), "diamond.world.com");
```

```
/* アダプタ gem 上で仮想ホスト diamond.world.com を起動 */
nc.call("startVHost", new onStartVHost(), "gem/diamond.world.com");
```

関連項目

[restartVHost\(\)](#)

stopServer()

使用できるバージョン

- Flash Player 6
- Flash Communication Server MX 1.0

シンタックス

```
stopServer(stop_type)
```

パラメータ

stop_type 文字列。可能な値は、normal または abort となります。

値 normal を使用する場合は、サーバーは終了し、実行中のアプリケーションは正常に終了します。

値 `abort` を使用する場合は、サーバーは即座に終了し、実行中のアプリケーションは正常に終了しません。

`abort` は、緊急時、または `normal` を指定しても正常に動作しない場合に限って使用してください。

戻り値

呼び出しが成功すると、サーバーは `level` プロパティの `status` および `code` プロパティの `NetConnection.Call.Success` を含む情報オブジェクトを応答として送信します。

呼び出しが失敗すると、サーバーは、`level` プロパティの `error` および `code` プロパティの `NetConnection.Admin.Command.Failed`、またはより具体的な値 (存在する場合) を含む情報オブジェクトを応答として送信します。失敗の原因について説明する文字列を含む `description` プロパティが含まれているオブジェクトもあります。

説明

Flash Media Server を終了します。ユーザーが接続しているときにこのコマンドを使用する場合は、アプリケーションが即座に終了する旨をユーザーに通知し、ユーザーの作業が失われないように配慮してください。

この操作を実行できるのは、サーバー管理者のみです。

関連項目

[startServer\(\)](#)

stopVHost()

使用できるバージョン

- Flash Player 6
- Flash Communication Server MX 1.0

シンタックス

```
stopVHost([vhost_name])
```

パラメータ

vhost_name オプションのパラメータ。停止する仮想ホストを含む文字列です。接続している仮想ホストを停止するには、このパラメータを削除します。

別の仮想ホストを停止するには *vhost_name* を *adaptor_name/vhost_name* の形式で指定します。

自分が接続している仮想ホスト以外の仮想サーバーを停止できるのは、サーバー管理者のみです。

戻り値

呼び出しが成功すると、サーバーは `level` プロパティの `status` および `code` プロパティの `NetConnection.Call.Success` を含む情報オブジェクトを応答として送信します。

呼び出しが失敗すると、サーバーは、`level` プロパティの `error` および `code` プロパティの `NetConnection.Admin.Command.Failed`、またはより具体的な値 (存在する場合) を含む情報オブジェクトを応答として送信します。失敗の原因について説明する文字列を含む `description` プロパティが含まれているオブジェクトもあります。

説明

実行中の仮想ホストを停止します。仮想ホストの停止後、すべてのアプリケーションはアンロードされ、すべてのユーザーとの接続が切断されます。`startVHost` コマンドを使用して仮想ホストを再起動するか、サーバーが再起動するまでは、仮想ホストは要求を一切受け付けません。

例

次の例では、仮想ホスト `tree.oak.com` を停止する方法、およびデフォルトのアダプタ上の仮想ホスト `tree.oak.com` を停止する方法を示します。

```
/* 仮想ホスト tree.oak.com を停止 */
nc.call("stopVHost", new onStopVHost(), "tree.oak.com");
```

```
/* デフォルトのアダプタ上の仮想ホスト tree.oak.com を停止 */
nc.call("stopVHost", new onStopVHost(), "_defaultRoot_/tree.oak.com");
```

関連項目

[restartVHost\(\)](#)、[startVHost\(\)](#)

unloadApp()

使用できるバージョン

- Flash Player 6
- Flash Communication Server MX 1.0

シンタックス

```
unloadApp(app_instance)
```

パラメータ

app_instance アプリケーションまたはアプリケーションインスタンスの名前を含む文字列 (`application_name[/instance_name]` の形式)。

戻り値

呼び出しが成功すると、サーバーは `level` プロパティの `status` および `code` プロパティの `NetConnection.Call.Success` を含む情報オブジェクトを応答として送信します。

呼び出しが失敗すると、サーバーは、`level` プロパティの `error` および `code` プロパティの `NetConnection.Admin.Command.Failed`、またはより具体的な値 (存在する場合) を含む情報オブジェクトを応答として送信します。失敗の原因について説明する文字列を含む `description` プロパティが含まれているオブジェクトもあります。

説明

指定したアプリケーションのすべてのインスタンス、または指定したアプリケーションインスタンスを終了します。アプリケーション名を指定すると、このアプリケーションのすべてのインスタンスは終了し、アプリケーションインスタンスへのユーザー接続はすべて即座に切断されます。アプリケーションインスタンスを指定する場合は、指定したインスタンスのみが終了し、このインスタンスへのユーザー接続すべてが即座に切断されます。

メモ

このコマンドを使用する前に、アプリケーションが即座に終了する旨をユーザーに通知してください。

関連項目

[reloadApp\(\)](#)