

***Flash Communication Components を使おう***  
***Macromedia Flash Communication Server MX***

---

本ドキュメントは、

Using Flash Communication Components Macromedia Flash Communication Server MX を和訳したものです。

誤訳、タイプミスの可能性もあり、内容、スクリプトの参照には、原文を参考にされた方が正確です。

ヒム・カンパニー 永井勝則

## CONTENTS

## CONTENTS

# CHAPTER 1

## コミュニケーション・コンポーネントで始めよう

コンポーネントは、定義されたパラメータを持ったムービー・クリップです。コンポーネントを Macromedia Flash MX オーサリング環境にインストールすることで、それらを使用して Flash ムービーのデザインや開発を行うことができます。コンポーネントに関する一般的な情報は、“*Using Flash MX*” の “*Using Components*” 章 “ を参照してください。

Macromedia Flash Communication Server MX コンポーネントは、コミュニケーション・コンポーネントと呼ばれる、クライアント・サイド部、サーバー・サイド部から成ります。クライアント・サイド部は Flash Player 内で動作する Flash ムービー・クリップです。サーバー・サイド部はコミュニケーション・コンポーネントの基本クラスを拡張した Action Script クラス(function オブジェクト)です。

### 読まれるに当たって

このドキュメントにはコミュニケーション・コンポーネントの使い方が説明されています。Macromedia Flash Communication Server MX 書類を理解し、Flash コミュニケーション・アプリケーション開発に伴う一般的な原則を把握した上でこのドキュメントを読めば、開発プロセスを簡素化したり Flash コミュニケーション・アプリケーション開発にかかる時間を減らしたりできます。

このドキュメントに書かれている手順を追ったり、コミュニケーション・コンポーネントを使ってコミュニケーション・アプリケーションを構築するには、Flash MX がインストールされ、稼働している Flash Communication Server にアクセスできる環境が必要です。

### Flash MX にコミュニケーション・コンポーネントをインストールする

インストールは他の Flash MX コンポーネントと同じ要領でできます。全てのコンポーネントはコンポーネント・パネルに準備されます。Flash MX\First Run\Components ディレクトリ内にコンポーネント・ムービークリップを含む FLA ファイルを置くと、コンポーネント・パネルにインポートしたり作成したコンポーネントが現れます。

### Flash MX 環境にコミュニケーション・コンポーネントをインストールする

サポート・センターからダウンロードした CommunicationComponents.zip または CommunicationComponents.hqx を開き、Communication Components.flx ファイルを取り出します。そしてインストールした Macromedia Flash MX ディレクトリにある FlashMX\First Run\Components ディレクトリにファイルを加えます。

### コミュニケーション・コンポーネントを確認する：

1 Macromedia Flash MX を起動させる

2 ウィンドウ・メニュー->コンポーネントを選択する

コンポーネント・パネルが現れる

3 コンポーネント・ポップアップ・メニューからコミュニケーション・コンポーネントを選択する



**Note:**コンポーネントのリストは、インストールされているコンポーネントによります。

### サーバー・サイド・ライブラリを更新する：

本ドキュメントに書かれているコンポーネントを使用するには、`¥scriptlib¥components` ディレクトリにある ASC ファイルを更新する必要があります。Zip ファイルを解凍して現れた全ての\*.asc ファイルをコピーし、`¥scriptlib¥components` ディレクトリ内のそれぞれのファイルと置き換えます。

**Note:**scriptlib ディレクトリの場所は Application.xml ファイル内の ScriptLibPath タグで特定されています。

### Scriptlib ディレクトリにアクセスする

Flash Communication Server をインストールすると、スクリプト・ライブラリもインストールされます。スクリプト・ライブラリは、重要なユーティリティやヘルパー・ファイルから構成される scriptlib ディレクトリです。これらのファイルはサーバー・サイド・スクリプトであり、アプリケーションのサーバー・サイド・スクリプト・ファイルの中へインクルードしたりロードしたりするものです。コンポーネントを読み込む.asc ファイルの詳しい説明は、11ページの“Your main.asc file”をご覧ください。

このドキュメントに含まれるサンプルを実行するには、サーバーへのインストールや構成が成功していなければなりません。scriptlib ディレクトリは flashcom アプリケーション内の既定の場所か、インストール中に指定した場所にインストールされます。

どちらの場合も、scriptlib ディレクトリの場所は、コンフィギュレーション・ファイル Application.xml の ScriptLibPath タグに記述されています。

## 環境のテスト

Flash MX 内にコミュニケーション・コンポーネントをインストールすると、コミュニケーション・アプリケーション作成の準備は整います。アプリケーション内で実行することの1つは、サーバーへの接続を確立させることです。以下のテストは PeopleList コンポーネントを使用して、アクティブなユーザーのリストを表示します。

### コミュニケーション・コンポーネントが正しくインストールされているかテストする：

1 Flash Communication Server が起動していることを確認する

2 Flash Communication Server MX のアプリケーション・ディレクトリ(通常は`flashcom\applications`)にアプリケーション・ディレクトリを作成、`com_test`と名前をつける。

**Note:** コミュニケーション・アプリケーション開発に関する詳しい情報は、Macromedia Flash Communication Server MX 書類を参照してください。

3 Flash MX で、Communication Components パネルが表示されていない場合は、ウィンドウ>コンポーネントを選択し、コンポーネント・ポップアップ・メニューから Communication Components を選択する。

4 コンポーネント・パネルからステージ上に PeopleList コンポーネントをドラッグする。

5 プロパティ・インスペクタで、インスタンス名 `peopleList_mc` をつける。

6 サーバーに接続するために、タイムラインの1番目のキーフレームを選択し、そのフレームのアクション・パネル内で次のコードを記述する。

//新しいネットワーク・コネクションを作成し、ストップ・アクションを加える

```
nc = new NetConnection();
```

```
stop();
```

//ステータス・ハンドラの作成

```
nc.onStatus = function(info) {trace(info.code);}
```

//リストをクリーン・アップするオプション・コール

```
peopleList_mc.close();
```

//アプリケーションに接続し、ユーザ名を与える

```
nc.connect("rtmp:/com_test","Jane");
```

**Note:** Flash Communication Server が localhost で動作していない場合、このテスト・アプリケーションにはフル URI(Uniform Resource Identifier) が必要となります(例：`rtmp://www.myflashcomdomain.com/com_test`)。NetConnection.connect メソッドについての詳しい内容は、Macromedia Flash Communication Server MX ドキュメントを参照してください。

7 nc NetConnection インスタンスを使って、PeopleList\_mc ムービー・クリップをサーバーへ接続させるには、タイムラインの1番目のキーフレームのアクション・パネルに以下のコードを加える。

```
PeopleList_mc.connect(nc);
```

8 保存した後、制御>ムービープレビューを選択しテストする。

以下のは onStatus メッセージを示している。これはサーバーへの接続が成功した場合出力ウィンドウに表示される。



この時、ユーザ名はリストに表示されていない。PeopleList コンポーネントはサーバーからユーザのリストを取得するだけで、サーバー・サイドの結果を取得する命令はまだ加わっていない。サーバー・サイド・スクリプトに何行か書き加えることで、コンポーネントはサーバーからユーザ名を取得することができる。

9 アプリケーション・ディレクトリに main.asc という新しいファイルを作成し、components.asc ファイルをロードする次のコードを記述する。

```
load("components.asc");
```

**Note:**サーバーは scriptlib ディレクトリにアクセスし、components.asc ファイルをロードする必要があります。

components.asc ファイルのロードに関する詳しい説明は、11ページの"Your main.asc.file"を参照してください。

10 クライアントからのユーザ名を受け取り、接続を許可する onConnect ハンドラを、以下のように作成する。

//newUserName はクライアント・サイドの nc.connect call からのパラメータ

```
application.onConnect = function(newClient,newUserName)
{
    //このファンクションを通るユーザ名でグローバルなユーザ名を設定する
    gFrameworkFC.getClientGlobals(newClient).username = newUserName;
    //ユーザからの接続を許可する
    application.acceptConnection(newClient);
}
```

11 main.asc ファイルを保存する。

**Note:**アプリケーション起動後 ASC ファイルを書き変えた時は、アプリケーションを再ロードさせる必要があります。

再ロードは Communication App Inspector で行います。Flash MX で(SWF ウィンドウでなく Flash MX のウィンドウが開かれていることを確認してください)、ウィンドウ>Communication App Inspector を選択します。ログイン後 Active Apps インスタンスウィンドウから com\_test¥\_definst\_を選択します。View Detail をクリック後、Reload App をクリックします。

1 2 Flash MX オーサリング環境で、制御>ムービープレビューを選択する。以下は接続した時の値。



## SimpleConnect コンポーネントを使う

コミュニケーション・コンポーネントを使ってアプリケーションを作成する時は、通常いくつかのコンポーネントを加えて作成しますが、それぞれがサーバーへの接続確立を必要とします。サーバーに接続するアプリケーション内で、同じネットワーク接続インスタンスを使用し、全てのムービー・クリップを接続させることができます。SimpleConnect コンポーネントは、あらゆるコミュニケーション・オブジェクトの接続を制御します。加えて、SimpleConnect は作成したアプリケーションで、ユーザがログインするインターフェイスも提供します。

次のサンプルは com\_test アプリケーションを使用し、SimpleConnect の働きを示すものです。PeopleList コンポーネントは NetConnection オブジェクトを使用せず、SimpleConnect の接続を使用してサーバーとやりとりします。SimpleConnect にサーバーへの接続を受け持たせることで、コードを能率的に機能させることができ、書かなくてはならない ActionScript を大幅に減らしてくれます。サンプルでは、SimpleConnect はサーバーへの接続を制御し、ActionScript を書くことなく PeopleList をその接続に結合させます。nc.connect コールの中にユーザ名を入れるという大変なコードを書かないで、実行させることができます。

### SimpleConnect コンポーネントを使って接続するには：

- 1 Flash Communication Server が起動していることを確認する
- 2 Flash Communication Server MX のアプリケーション・ディレクトリ(通常は`flashcom\applications`)にアプリケーション・ディレクトリを作成し、com\_test\_simcon の名前をつける。
- 3 com\_test\_simcon 内に main.asc の名前で新しいファイルを作成する。components.asc ファイルをロードする以下のコードを記述する。

```
load("components.asc");
```

**Note:**main.asc ファイルを *appName.asc* の名前にすることも可能です。appName は作成したアプリケーション名で、appName はサーバーのアプリケーション・ディレクトリ名を意味します。

- 4 Flash MX で Communication Components パネルが表示されていない場合は、ウィンドウ・メニュー>コンポーネントを選択し、コンポーネント・ポップアップ・メニューから Communication Components を選択する。
- 5 コンポーネント・パネルからステージ上に peopleList コンポーネントをドラッグする。
- 6 プロパティ・インスペクタで、peopleList コンポーネントのインスタンス名に peopleList\_mc をつける。
- 7 Flash Communication Server へ接続させるために、ステージ上に SimpleConnect コンポーネントをドラッグする。

8 プロパティ・インスペクタで、以下のパラメータを与える。

Application Directory テキスト・ボックスに、rtmp:/com\_test\_simcon と入力する。URI の com\_test\_simcon 部分は、アプリケーション名で、ステップ2 で作成したディレクトリ名に一致させる。

Communication Components テキスト・ボックスをダブル・クリックし、現れた値ダイアログ・ボックスで、+印をクリックし、peopleList\_mc と入力、OK をクリックする。

この値を与えることによって、SimpleConnect コンポーネントは PeopleList コンポーネントの接続を制御する。

SimpleConnect コンポーネントがサーバー接続に成功すれば、PeopleList コンポーネントは自動的にサーバーに接続する。

9 このテストファイルを com\_test\_simcon.fla 名で保存、パブリッシュする。

10 SWF ファイルを開くか、Flash MX オーサリング環境で制御>ムービープレビューを選択する。適当なユーザ名でログインする。

入力したユーザ名が、以下のように PeopleList コンポーネント内に表示される。



SimpleConnect コンポーネントはログイン時のインターフェイスを提供します。アクティブなユーザのリストを増やすには、このアプリケーションの別のインスタンスを開き、それぞれのインスタンスに新しいユーザ名を入れます。

ただ PeopleList コンポーネントをステージ上にドラッグし、peopleList\_mc と名づけただけなので、com\_test のサンプルに SimpleConnect を加えるのは大げさな気もしますが、以下のコードを加えます。

```
nc = new NetConnection();
nc.onStatus = function(info) { trace(info.code);}
peopleList_mc.close();//オプション、クリーンアップ・コール
nc.connect("rtmp:/com_test", "Jane");
peopleList_mc.connect(nc);
```

しかし SimpleConnect は、作成するアプリケーションが大きくなればなるほど、使い勝手がよくなっていきます。新たなコミュニケーション・コンポーネントを加えるごとに、SimpleConnect コンポーネントのプロパティ・インスペクタで、Communication Component テキスト・ボックスをダブル・クリックして開き、新しいコンポーネントのインスタンス名を加えるだけでよいのです。

## Your main.asc file

コミュニケーション・コンポーネントを使用する時、いくつかのサーバ・サイド・スクリプトを使う必要があります。最小限のコール含む main.asc ファイルを作成した後、コミュニケーション・コンポーネントを使うアプリケーション・ディレクトリにこのファイルをコピーして使い回すことができます。

**Note:** サーバーは components.asc ファイルをロードするために scriplib ディレクトリにアクセスする必要があります。components.asc ファイルについての詳しい情報は、6 ページの "scriplib ディレクトリにアクセスする" をご覧ください。

main.asc ファイルの大きな役目は、scriplib ディレクトリに置かれている components.asc ファイルをロードすることです。このファイルをロードすることで、コンポーネント・サーバ・サイド・スクリプトの全ての機能を、作成するアプリケーションに持たせることができます。それはただ、作成するアプリケーションの main.asc ファイルの冒頭に次のコードを書くだけで可能になります。

```
load("components.asc");
```

## SimpleConnect コンポーネントを使用しない場合

SimpleConnect コンポーネントを使用しないなら、コミュニケーション・コンポーネントを正しく機能させるためには、サーバ・サイドの application.onConnect メソッドに以下のコードを加え、サーバーにユーザ名を登録しなければなりません。

```
GFrameworkdFC.getClientGlobals(newClient).username = newUserName;
```

サーバ・サイド・コミュニケーション ActionScript 辞書に記述されているように、onConnect メソッドを書いた時は常に新しいユーザの接続を明確に許可しなければなりません。

SimpleConnect コンポーネントを使用しない場合は、main.asc ファイルは以下のコードを含んでいる必要があります。

```
load("components.asc");

application.onConnect = function(newClient,newUserName)

{

    gFrameworkdFC.getClientGlobals(newClient).username = newUserName;

    application.acceptConnection(newClient);

}
```

## SimpleConnect コンポーネントを使用する場合

SimpleConnect コンポーネントを使用する場合は、以下のコードを main.asc ファイルに書きます。

```
load("components.asc");
```

## コンポーネントのリスキニング

Flash ドキュメントにコンポーネントを加えると、ライブラリ・パネルに Commucation Components ディレクトリが加わります。このディレクトリには FlashCom UI Componets というサブディレクトリが含まれています。このサブディレクトリの中には、デザインを修正できる、UI 要素を含むそれぞれのコンポーネント用のスキン・サブディレクトリがあります。本ドキュメントにあるそれぞれのコンポーネントの記述には、コンポーネント・スキンの場所を示した“リスキニング”セクションも含まれています。スキンを編集しコンポーネントの見た目を変えることができます。ステージ上でコンポーネントを選択したり、右クリックしたり、「同じ位置で編集」を選択することで、コンポーネントのインスタンスのスキンを変更できます。またライブラリ・パレットのディレクトリ内でダブルクリックすることで、コンポーネント・スキンを編集することもできます。

## オリジナル・コンポーネントの開発

上級開発者は Flash コミュニケーション・ムービー・クリップとサーバー・サイド・コードを作成することで、コミュニケーション・コンポーネントを作成することができます。クライアント・サイド部は Flash ムービー・クリップとして実行され、Flash クライアント環境で動作します。サーバー・サイド部は、コミュニケーション・コンポーネント基本クラスである FComponent を拡張した ActionScript クラス(ファンクション・オブジェクト)です。コンポーネントの作成方法に関する情報は、Macromedia Flash MX web サイトを、コミュニケーション・コンポーネントの作成方法に関する情報は、Macromedia Designer & Developer Center

([http://www.macromedia.com/go/fcs\\_components](http://www.macromedia.com/go/fcs_components))をご覧ください。

## lurker モードの理解

コンポーネントの中には、接続した他のユーザからユーザを見えなくするユーザ・モードを提供するものもあります。このモードは *lurker mode* と呼ばれ、例えば司会者や参加者が必要な場面では有用に使えます。lurker モードでは、他のユーザに必ずしも知られず、全機能にアクセスする必要のないユーザが存在します。。例えば Chat コンポーネントは、このルーカ(見ているだけの人)に、他のユーザの文字チャットを見せるが、メッセージを送ることは許可しない lurker mode をサポートしています。別のコンポーネントはまた別の lurker mode 機能を提供します。本ドキュメントに記述されている機能は、作成するアプリケーションが SimpleConnect を使用すると仮定し、ユーザがイグジットせずアプリケーションからログアウトすることを拒否します。コミュニケーション・コンポーネントに関する記述はどれも、lurker mode が可能かどうか、その機能について書かれています。

## CHAPTER 2 コミュニケーション・コンポーネントを使おう

この章では、コンポーネントそれぞれの使い方、もしあればスクリプティングに必要なことが書かれています。コンポーネントはアルファベット順に並んでいますが、Macromedia は何よりもまず 64 ページの”SimpleConnect コンポーネント”を読まれることを推奨します。49 ページの”RoomList コンポーネント”のサンプルを動作させたいと思われるかも知れませんが、これは最も上級で、JavaScript もサーバー・サイド・スクリプティングも必要となります。

コンポーネントを使ったセクションの次は、コンポーネント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) のセクションで、コンポーネントのオブジェクトやメソッドについて記述されています。コミュニケーション・コンポーネントは、Macromedia Flash Player 6 とそれ以降でサポートされています。

### AudioConference コンポーネント

AudioConference コンポーネントを使用すると、多数のクライアントによる同時ストリーミングに対応する、マルチプレックスなアプリケーションが作成できます。このアプリケーションはまた、アプリケーションに今ログインしているユーザをリストアップするユーザ・インターフェイス(UI)を提供します。UI は接続している各ユーザをライトで示し、ユーザが音声を送るとライトは緑色に変わります。ユーザは会話に加わるには Talk ボタンをクリックするか、喋ると自動的に音声を送る Auto チェック・ボックスを選択します。このコンポーネントでは、リストに表示されたり他のユーザと話をするには、まずログインすることが必要となります。

このコンポーネントは、SimpleCpconnect と併用すればスクリプティングの必要はなくなります。

### AudioConference コンポーネントを使う

以下はこのコンポーネントについてのいくつかの使用例です。

パーティ・ライン：ユーザのグループがある話題を同時に討論する。

仮想電話会議：電話をかけてスピーカーフォンで行う会議の代わりに行うオンライン本社会議。このコンポーネントを使って、企業イメージを使ったカスタム・アプリケーションを簡単に作成することができます。

テクニカル・サポート：トレーニング器具や複雑なアプリケーションを扱う場合、問題解決フォーラムでこのコンポーネントを使用する。

## アプリケーションでコンポーネントを使用するには

1 Flash Communication Server が起動していることを確認する

2 Flash Communication Server MX のアプリケーション・ディレクトリ(通常は%flashcom%applications)にアプリケーション・ディレクトリを作成し、例えば audioconf\_test の名前をつける。

3 アプリケーション・ディレクトリ内に main.asc の名前で新しいファイルを作成する。components.asc ファイルをロードする以下のコードを記述する。

```
load("components.asc");
```

**Note:**他のアプリケーション・ディレクトリからこのアプリケーション・ディレクトリへ、main.asc ファイルをコピーして使用することもできます。

4 Flash MX で、ステージ上に AudioConference コンポーネントをドラッグする。

5 プロパティ・インスペクタで、AudioConference コンポーネントのインスタンス名に audioconf\_mc をつける。

6 ステージ上に SimpleConnect コンポーネントをドラッグする。

7 プロパティ・インスペクタで、以下の2つのパラメータを入力する。

Application Directory テキスト・ボックスで、ステップ 2 で作成したアプリケーションの URI、例えば、rtmp:/audioconf\_test を入力する。

Communication Components テキスト・ボックスをダブル・クリックする。現れる値ダイアログ・ボックスで、+ 印をクリックし、AudioConference コンポーネントのインスタンス名 audioconf\_mc を入力し、OK をクリックする。この値を入力することで、SimpleConnect コンポーネントは AVPresence コンポーネントの接続を制御できる。SimpleConnect コンポーネントがサーバーへの接続に成功したら、AudioConference コンポーネントは自動的にサーバーに接続する。

8 myAudioApp のファイル名でアプリケーション・ディレクトリ内に保存、パブリッシュする。

9 アプリケーション・ディレクトリ内にある SWF ファイルを開く(または Flash MX オーサリング環境では、制御>ムービープレビューを選択する)。ログインし Auto チェック・ボックスをクリックする。

オーディオ入力デバイスが動作していれば、話すとライトが緑に変わる。ライトはそれぞれのユーザ名の隣に現れる。話していなければライトはくすんだ色になる。



## AudioConference コンポーネントをリスキニングする

このコンポーネントのアセットは、ライブラリ・パネルの Core Assets-Developer Only\AudioConference Assets ディレクトリにあります。

### 関係するコンポーネント

“SimpleConnect コンポーネント” 64 ページ

## FCAudioConference オブジェクト

FCAudioConference オブジェクトでは、SimpleConnect コンポーネントと併用すればスクリプティングの必要はありません。ユーザは、リストに現れ話すには、ログインしなければなりません。

FCAudioConference API は、サーバーに接続したりクリーンアップするオブジェクトに、クライアント・サイドとサーバー・サイドでの機能を提供します。

### FCAudioConference オブジェクトのメソッド・サマリー

メソッド	記述
FCAudioConference.connect	サーバー上のアプリケーションに接続する
FCAudioConference.setUsername	ユーザ名を設定し、アクティブなユーザのリストに表示する
FCAudioConference.close	クリーンアップし、音声会議からそのユーザを消去する

### FCAudioConference.connect

#### 利用

Flash Player 6

Flash Communication Server MX

#### 使い方

```
myAudioConf_mc.connect(nc)
```

#### パラメータ

nc      アクティブな NetConnection オブジェクト

#### 戻り値

なし

#### 説明

メソッド；サーバー上のアプリケーションに接続する。また、クライアント・サイドやサーバー・サイドでコンポーネントが必要とする全てのアセットをセット・アップし、サーバーが提供する名前でも setUsername メソッドをコールする。

## **FCAudioConference.setUsername**

### **利用**

Flash Player 6

Flash Communication Server MX

### **使い方**

```
myAudioConf_mc.setUsername([newName])
```

### **パラメータ**

*newName* リストに表示するユーザ名を含むストリング。*newName* が null または *undefined* でない場合、Talk オプションと Auto オプションが選択可能になり、ユーザ名がリストに表示される。*newName* が null の場合は何も変化せず、このユーザは lurker mode(見ているだけ)となる。これは他のユーザがこのユーザの存在に気づかないことを意味する。

### **戻り値**

なし

### **説明**

メソッド：アクティブなユーザのリストに表示するためユーザ名を設定する。

## **FCAudioConference.close**

### **利用**

Flash Player 6

Flash Communication Server MX

### **使い方**

```
myAudioConf_mc.close()
```

### **パラメータ**

なし

### **戻り値**

なし

### **説明**

メソッド：クライアントのコンポーネントで使用されていたアセットを解放し、音声会議からユーザを消去することでクリーンアップする。

## AVPresence コンポーネント

AVPresence コンポーネントは、作成したアプリケーションで音声や映像の送受信を可能にします。このコンポーネントは、接続したユーザなら誰でも使用できる、プレゼンター・シートの役目を果たします。ユーザが音声か映像、もしくはその両方をコンポーネントのインスタンスから送信すると、他のユーザは自動的にその音声や映像を見たり聞いたりできます。コンポーネントのインスタンスが使用されていない場合、ユーザはそれをクリックして音声や映像を送信できます。ユーザは音声や映像、その両方を切断するだけでなく、送信も受信もできるコントロールを持ちます。ユーザが、入ってくるストリーム上でこのコントロールを使うと、結果はローカル上に現れます。自分自身の音声や映像のインスタンス上でこのコントロールを使うと、接続している全てのユーザのインターフェイス上にその効果は現れます。このように、見られているか聞かれているかだけでなく、誰を見たり、聞くかをどのユーザも自分でコントロールできるのです。

このコンポーネントは、SimpleConnect コンポーネントと併用するとスクリプティングの必要はなくなります。ステージ上にドラッグするコンポーネント・インスタンスにはどれも唯一のインスタンス名が必要です。AVPresence コンポーネントには変更可能ないくつかのコンポーネント・パラメータがあります。

### AVPresence コンポーネントを使用する：

このコンポーネントは多くの用途に使用できるコミュニケーション・コンポーネントであり、ユーザに仮想空間を提供するために使用できます。以下はこのコンポーネントの使用例です。

パネル・ディスカッション：人々が聴衆の前で討論を交わしたり、ディベート、プレゼンテーションを行う。

ビデオ電話：1つのアプリケーションに2つの AVPresence を使うことで、ビデオ電話アプリケーションを作成できます。

### アプリケーションでコンポーネントを使用するには

1 Flash Communication Server が起動していることを確認する

2 Flash Communication Server MX のアプリケーション・ディレクトリ(通常は`flashcomapplications`)にアプリケーション・ディレクトリを作成し、例えば `avPresence_test` の名前をつける。

3 アプリケーション・ディレクトリ内に `main.asc` の名前で新しいファイルを作成し、以下のコードを記述する。

```
load("components.asc");
```

**Note:**他のアプリケーション・ディレクトリからこのアプリケーション・ディレクトリへ、`main.asc` ファイルをコピーして使用することもできます。

4 Flash MX で、ステージ上に AVPresence コンポーネントをドラッグする。

5 プロパティ・インスペクタで、以下の値を入力する。

コンポーネントのインスタンス名に `avPresence_mc` をつける。

パラメータの定義は、18ページの“AVPresence コンポーネント・プロパティ・インスペクタ”を参照のこと。

6 ステージ上に Flash Communication Server への接続を設定する SimpleConnect コンポーネントをドラッグする。

7 SimpleConnect コンポーネントのプロパティ・インスペクタで、以下の2つのパラメータを入力する。

Application Directory テキスト・ボックスで、ステップ2で作成したアプリケーションのURI、例えば、`rtmp://AVPresence_test`を入力する。

Communication Components テキスト・ボックスをダブル・クリックする。現れる値ダイアログ・ボックスで、+印をクリックし、AVPresence コンポーネントのインスタンス名 `avPresence_mc` を入力し、OK をクリックする。

この値を与えることで、SimpleConnect コンポーネントは AVPresence コンポーネントの接続を制御できる。SimpleConnect コンポーネントがサーバーへの接続に成功したら、AudioConference コンポーネントは自動的にサーバーに接続する。

8 `myAVPresence` のファイル名でアプリケーション・ディレクトリ内に保存、パブリッシュする。

9 アプリケーション・ディレクトリ内にある SWF ファイルを開く(または Flash MX オーサリング環境では、制御>ムービープレビューを選択する)。ログインし Send Audio/Video ボタンをクリックする。

以前に Flash Player のプライバシー設定を行っていない場合は、カメラ、マイクへのアクセスを許可するかどうか聞かれる。

オーディオ・デバイスデバイスが作動している場合、喋ると音声レベルが上下する。ビデオ・デバイスが作動していると、ビデオ画面にその出力結果を見ることができる。マウス・ポインタをビデオ画面上で動かすと、マイクとカメラのUIを見ることができる。これは音声や映像データへの変換を止めるボタンとしても働く。

## AVPresence コンポーネント・プロパティ・インスペクタ

コンポーネント・インスタンスのプロパティ・インスペクタで以下の変数を設定できます。

名前	変数	説明
Presenter SharedObject	<code>soName</code>	ストリング：デフォルト値： <code>av</code> 。動作中コンポーネントを制御するため、サーバー上で使用される共有 オブジェクト名を決定する。
Sync Speed	<code>updateFps</code>	数値：デフォルト値：3。このコンポーネントが毎秒何回メッセージを送るかを設定する。この値は音声、映像クオリティとは独立しており、インスタンスが、音声のアップデート・レートや、送受信する新しいストリームにどれくらい速く応答するかを制御する。
Video Width	<code>vidWidth</code>	数値：デフォルト値：120。ローカルでのカメラ設定で画像の幅をピクセル単位で設定する。 <code>vidWidth</code> は <code>FCSetBandwidth</code> が使用されたら上書きされる。
Video Height	<code>vidHeight</code>	数値：デフォルト値：120。ローカルでのカメラ設定で画像の高さをピクセル単位で設定する。 <code>vidHeight</code> は <code>FCSetBandwidth</code> が使用されたら上書きされる。
Video Bandwidth	<code>vidBandwidth</code>	数値：デフォルト値：12,000。ローカルでのカメラ設定で帯域幅の最大値をビット単位で設定する。 <code>VidBandwidth</code> は <code>FCSetBandwidth</code> が使用されたら上書きされる。
Video Quality	<code>vidQuality</code>	数値：デフォルト値：0。ローカルでのカメラ設定で最大品質を設定する。0は



## FCAVPresence.setUsername

### 利用

Flash Player 6

Flash Communication Server MX

### 使い方

*avPresence\_mc.setUsername(newName)*

### パラメータ

*newName* 新しいユーザ名を記述するオプションのストリング。*NewName* パラメータが null、もしくは undefined でない場合、Flash はそのユーザ用の新しい参照を作り、名前を *newName* に設定する。*NewName* が null の場合は、何も変化せず、ユーザは lurker mode となる。

### 戻り値

なし

### 説明

メソッド； アクティブなユーザのリストで表示するためのユーザ名を取得する

## FCAVPresence.close

### 利用

Flash Player 6

Flash Communication Server MX

### 使い方

*avPresence\_mc.close()*

### パラメータ

なし

### 戻り値

なし

### 説明

メソッド； クライアント上でコンポーネントが使用しているアセットを解放しクリーンアップする。

## Chat コンポーネント

このコンポーネントは一般的な文字チャットのウィンドウを持ち、lurker mode をサポートします。SimpleConnect コンポーネントと一緒に使用すれば、ログインしないユーザがテキスト・ボックスに入力しても Send ボタンを押しても、lurker mode により表示されません。見ているだけのユーザも他の人の文字チャットを見ることはできます。ユーザ名を入力してログインすると(SimpleConnect コンポーネントを使用するか、1 番目のパラメータとして、nc.connect をコールするか)、ユーザは文字を送信できます。UserColor コンポーネントも使用すれば、ユーザの文字が個別に色分けされます。Chat コンポーネントは SimpleConnect コンポーネントを併用すればスクリプティングの必要がなくなります。

このコンポーネントを UserColor コンポーネントと一緒に使用することもできます。ユーザはいつでも新しい色を選択でき、サーバーサイドではグローバルな色として更新され、全てのクライアントでも色は更新されます。

### Chat コンポーネントを使用する

このコンポーネントは、チャットルーム・アプリケーション作成のために使用したり、以下に記すように大きなアプリケーションの内部で動作するコンポーネントとしても使用できます。

一般的なチャットルーム:このコンポーネントを UserColor コンポーネントや PeopleList コンポーネント、SimpleConnect コンポーネントと併用してチャットルームを作成できます。

あらゆるアプリケーションにおいて使いやすい:コミュニケーション・アプリケーションの中に Chat コンポーネントを含めることは有益です。文字のコミュニケーションは扱いやすく、例えばユーザのコンピュータの接続環境が遅い場合など、音声・映像でのコミュニケーションがしづらい場合の代案になります。

### Chat コンポーネントを使うには

1 Flash Communication Server が起動していることを確認する

2 Flash Communication Server MX のアプリケーション・ディレクトリ(通常は`flashcom\applications`)にアプリケーション・ディレクトリを作成し、`chat_test`の名前をつける。

3 アプリケーション・ディレクトリ内に `main.asc` の名前で新しいファイルを作成し、以下のコードを記述する。

```
load("components.asc");
```

**Note:**他のアプリケーション・ディレクトリからこのアプリケーション・ディレクトリへ、`main.asc` ファイルをコピーして使用することもできます。

4 Flash MX 画面で、ステージ上に Chat コンポーネントをドラッグし、プロパティ・インスペクタで、`chat_mc` のインスタンス名をつける。

5 ステージ上に UserColor コンポーネントをドラッグし、プロパティ・インスペクタで、`color_mc` のインスタンス名をつける。

6 ステージ上に SimpleConnect コンポーネントをドラッグする。

7 SimpleCpnnection 用のプロパティ・インスペクタで以下のパラメータを入力する。

Application Directory テキスト・ボックスで、ステップ 2 で作成したアプリケーションの URI、例えば、`rtmp:/chat_test` を入力する。

Communication Components テキスト・ボックスをダブル・クリックする。現れる値ダイアログ・ボックスで、+ 印をクリックし、Chat コンポーネントのインスタンス名 `chat_mc` を入力、さらに+印をクリックし、UserColor コンポーネントのインスタンス名 `color_mc` を入力し、OK をクリックする。

これらの値を与えることで、SimpleConnect コンポーネントは Chat コンポーネントと UserColor コンポーネントの接続を制御できる。SimpleConnect コンポーネントがサーバーへの接続に成功したら、Chat と UserColor コンポーネントは自動的にサーバーに接続する。

8 `chat_test` のファイル名でアプリケーション・ディレクトリ内に保存、パブリッシュする。

9 アプリケーション・ディレクトリ内にある SWF ファイルを開くか、または Flash MX オーサリング環境では、制御 >ムービープレビューを選択しログインする。

多くの SWF ファイル・インスタンスを開くことができます。違うユーザとしてログインしたり、下記に示すように違う文字カラーを選ぶこともできます。



#### Chat コンポーネントのリスキニング

Chat コンポーネントのアセットは、ライブラリ・パネルの Core Assets-Developer Only\Chat Assets ディレクトリにあります。

#### 関係するコンポーネント

“SimpleConnect コンポーネント” 6 4 ページ、“UserColor コンポーネント” 6 5 ページ、“PeopleList コンポーネント” 3 4 ページ。

#### FCChat オブジェクト

FCChat API は、サーバーに接続したりクリーンアップするオブジェクトに、クライアント・サイドでの機能を提供します。またプログラミングで、クライアントからチャットの履歴を消すこともできます。このコンポーネントのサーバー・サイド API については 2 5 P の“FCChat サーバー・サイド・オブジェクト”をご覧ください。

#### FCChat オブジェクトのメソッド・サマリー

メソッド	説明
FCChat.connect	クライアント・サイド、サーバー・サイドで、コンポーネントの必要とするアセットを全てセット・アップし、チャット履歴を取得する。
FCChat.setUsername	アクティブなユーザのリストを現わすためユーザ名を取得する。
FCChat.clearHistory	チャット履歴を消去し、全てのクライアント上で履歴を更新する。
FCChat.close	クリーンアップする。

## **FCChat.connect**

### **利用**

Flash Player 6

Flash Communication Server MX

### **使い方**

*chat\_mc.connect(nc)*

### **パラメータ**

*nc* アクティブな NetConnection オブジェクト

### **戻り値**

なし

### **説明**

メソッド ; クライアント・サイド、サーバー・サイドで、コンポーネントの必要とするアセットを全てセットアップし、チャット履歴を取得する。

## **FCChat.setUsername**

### **利用**

Flash Player 6

Flash Communication Server MX

### **使い方**

*chat\_mc.setUsername(newName)*

### **パラメータ**

*newName* スtring

*newName* パラメータが null か undefined でない場合は入力テキスト・ボックスと Send ボタンを表示する。

*NewName* パラメータが null の場合は、何もしない(lurker mode)

### **戻り値**

なし

### **説明**

メソッド ; アクティブなユーザのリストに表示するためにユーザ名を取得する

## **FCChat.clearHistory**

### **利用**

Flash Player 6

Flash Communication Server MX

### **使い方**

`chat_mc.clearHistory()`

### **パラメータ**

なし

### **戻り値**

なし

### **説明**

メソッド ; チャット履歴を消去し、全てのクライアント上で履歴を更新する。この命令は、クライアントによるクリアを、サーバー・サイドの設定(デフォルトでは実行可能)が許可して初めて成功する。さらに詳しい情報は 27 P の “ FCChat.allowClear ” をご覧ください。

## **FCChat.close**

### **利用**

Flash Player 6

Flash Communication Server MX

### **使い方**

`chat_mc.close()`

### **パラメータ**

なし

### **戻り値**

なし

### **説明**

メソッド ; クライアント上でコンポーネントが使用しているアセットを解放しクリーンアップする。

## **FCChat サーバー・サイド オブジェクト**

FCChat サーバー・サイド API は、履歴プロパティの接続と履歴のクリアを可能にするサーバー・サイドの機能を提供します。

## FCChat サーバー・オブジェクトのメソッド・サマリー

### メソッド 説明

FCChat.clearHistory 履歴をクリアする。

## FCChat サーバー・オブジェクトのプロパティ・サマリー

以下のサーバー・サイド・プロパティは、コンポーネントのふるまいを修正するために設定することができます。

### プロパティ 説明

FCChat.histlen チャット履歴の可能な大きさを定義する。

FCChat.persist チャット履歴がアプリケーションの実行中に持続するかどうかを定義する。

FCChat.allowClear チャット履歴がクライアントからクリアできるかどうかを定義する。

## FCChat.histlen

### 利用

Flash Player 6

Flash Communication Server MX

### 使い方

```
FCChat.prototype.histlen = maxHistory;
```

### 説明

プロパティ； チャット履歴の可能な大きさを定義する。取得、設定できるプロパティ。値 *maxHistory* は、サーバーが保有する文字メッセージの最大長を示す数値。デフォルト値は 250。

## FCChat.persist

### 利用

Flash Player 6

Flash Communication Server MX

### 使い方

```
FCChat.prototype.persist = true;
```

### 説明

プロパティ； チャット履歴がアプリケーションのある間ずっと保持されるかどうかを定義する。履歴はアプリケーションがリスタートしても持続する。これは取得、設定できるプロパティ。値 *true* と *false* は、サーバーが履歴を保存するかどうかを決める Boolean 値。デフォルト値は *true*。

## FCChat.allowClear

### 利用

Flash Player 6

Flash Communication Server MX

### 使い方

```
FCChat.prototype.allowClear = true;
```

### 説明

プロパティ； チャット履歴がクライアントからクリアできるかどうかを定義する。これは取得、設定できるプロパティ。値 true と false はユーザが履歴をクリアできるかどうかを決定する Boolean 値。デフォルト値は true。

## FCChat.clearHistory

### 利用

Flash Player 6

Flash Communication Server MX

### 使い方

```
FCChat.prototype.clearHistory()
```

### パラメータ

なし

### 説明

メソッド； 履歴をクリアする。クライアント・サイドでなくサーバー・サイドでこのメソッドをコールすると成功する。

## ConnectionLight コンポーネント

このコンポーネントは、クライアントの接続状況を視覚的に戻す機能を提供します。ライトは、接続すれば緑、切断すれば赤、接続のレイテンシー(ネットワークを経由してサーバーからデータを送るのにかかる時間)が高すぎると赤になります。ライトは色を変える機能のほか、クリックするとトグル・ボタンとして、接続に関する詳細な情報をボックスの中に表示します。情報とは、レイテンシー・レート、アップロード、ダウンロードのレートです。

ConnectionLight コンポーネントは、SimpleConnection コンポーネントと併用すればスクリプティングの必要はなくなります。

### ConnectionLight コンポーネントを使用する

このシンプルで使い勝手の良いコンポーネントは、どんなアプリケーションにでも使用でき、コミュニケーション・アプリケーションの標準的なコンポーネントといえます。

ConnectionLight コンポーネントの使い方は簡単で、ステージにドラッグしてサーバー・スクリプトで components.asc がロードされているかを確認するだけです。このコンポーネントを SimpleConnection コンポーネントと併用すれば、作業は終了です。SimpleConnection コンポーネントを使わない場合は、以下のクライアント・サイド ActionScript を記述する必要があります。

```
light_mc.connect(nc);
```

この ActionScript の中で、light\_mc はムービー内にドラッグされたライトのインスタンス名で、nc は NetConnection オ

プロジェクトです。

### アプリケーションでコンポーネントを使う：

1 Flash Communication Server が起動していることを確認する。

2 Flash Communication Server MX のアプリケーション・ディレクトリ(通常は`flashcom\applications`)にアプリケーション・ディレクトリを作成し、`connect_test` の名前をつける。

3 `connect_test` 内に `main.asc` の名前で新しいファイルを作成し、以下のコードを記述する。

```
load("components.asc");
```

**Note:**他のアプリケーション・ディレクトリからこのアプリケーション・ディレクトリへ、`main.asc` ファイルをコピーして使用することもできます。

4 Flash MX でコミュニケーション・コンポーネント・パネルが表示されていない場合は、ウィンドウ・メニュー>コンポーネントを選択し、コンポーネント・ポップアップ・メニューからコミュニケーション・コンポーネントを選択する。

5 コンポーネント・パネルから `ConnectionLight` コンポーネントをステージ上にドラッグする。プロパティ・インスペクタで `light_mc` のインスタンス名をつける。他のパラメータについては、29ページの“`ConnectionLight` コンポーネント・プロパティ・インスペクタ”をご覧ください。

6 サーバーに接続するには、タイムラインの一番めのキーフレームを選択し、そのフレームのアクション・パネルで以下のコードを記述する。

```
nc = new NetConnection();
```

```
nc.connect("rtmp://connect_test");
```

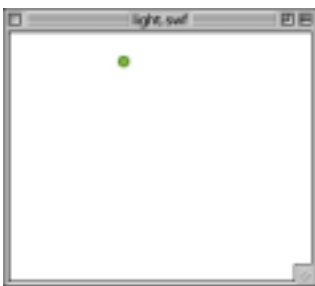
7 サーバーへ `light_mc` ムービー・クリップを接続させるには、以下のように、`nc` `NetConnection` インスタンスを使用する。

```
light_mc.connect(nc);
```

8 このファイルを `connect_test fla` 名で保存、パブリッシュする。

9 アプリケーション・ディレクトリで SWF ファイルを開くか、Flash MX オーサリング環境の場合は、制御>ムービープレビューを選択する。緑のボタンをクリックする。

すると以下のように表示される。



### ConnectionLight コンポーネントをリスキニングする

このコンポーネントのアセットは、ライブラリ・パネルの `Core Assets-Developer Only\ConnectionLight Assets` ディレクトリにあります。

## ConnectionLight コンポーネント・プロパティ・インスペクタ

### プロパティ

### 解説

measurementInterval	このプロパティは、帯域幅とレイテンシーの値をどれくらいの間隔で測定するかを制御する。デフォルト値は2秒。
latencyThreshold	このプロパティは、ライトが黄色に変わるレイテンシーを決定する。デフォルト値は0.1秒。よって、接続レイテンシーが100ミリ秒より大きくなればライトは黄色に変わり、レイテンシーがしきい値以下に落ちるまでそのままのままでいる。

### 関係コンポーネント

“ SimpleConnect コンポーネント ” 64ページ。

## FCConnectionLight オブジェクト

FCConnectionLight API は、サーバーに接続したりクリーンアップするオブジェクトに、クライアント・サイドとサーバー・サイドでの機能を提供します。またサーバーからの送信データを受け取る時間の長さを設定することもできます。

### FCConnectionLight オブジェクトのプロパティ・サマリー

#### プロパティ

#### 解説

FCConnectionLight.measurementInterval	帯域幅とレイテンシーの値をどれくらいの間隔で測定するかを制御する。
FCConnectionLight.latencyThreshold	ライトが黄色に変わるレイテンシーを決定する。

### FCConnectionLight オブジェクトのメソッド・サマリー

#### メソッド

#### 解説

FCConnectionLight.connect	接続状況の監視を開始する。
FCConnectionLight.close	接続の監視をやめ、クリーンアップする。

### FCConnectionLight.measurementInterval

### 利用

Flash Player 6

Flash Communication Server MX

### 使い方

```
connectLight_mc.measurementInterval = maxInterval
```

### 説明

プロパティ；帯域幅とレイテンシーの値をどれくらいの間隔で測定するかを制御する。これは取得、設定できるプロパティ。値 *maxInterval* は、帯域幅とレイテンシーの値をどれくらいの間隔で測定するかを決定する数値。デフォルト値は2秒。

## **FCConnectionLight.latencyThreshold**

### **利用**

Flash Player 6

Flash Communication Server MX

### **使い方**

```
connectLight_mc.latencyThreshold = maxLatency
```

### **説明**

プロパティ；ライトが黄色に変わるレイテンシーを決定する。これは取得、設定できるプロパティ。値 *maxLatency* は、レイテンシーが高くなると黄色に変わるのにかかる時間を決定する。デフォルト値は 0.1 秒。よって、接続レイテンシーが 100 ミリ秒より大きくなればライトは黄色に変わり、レイテンシーがしきい値以下に落ちるまでそのままのまである。

## **FCConnectionLight.connect**

### **利用**

Flash Player 6

Flash Communication Server MX

### **使い方**

```
connectLight_mc.connect(nc)
```

### **パラメータ**

*nc*            アクティブな NetConnection オブジェクト。

### **戻り値**

なし

### **説明**

メソッド；接続状況の監視を開始する。

## **FCConnectionLight.close**

### **利用**

Flash Player 6

Flash Communication Server MX

### **使い方**

```
connectLight_mc.close()
```

### **パラメータ**

なし

### **戻り値**

なし

### **説明**

メソッド；クライアント上でこのコンポーネントが使用しているアセットを解放し、クリーンアップする。その後は接続監視をしない。

## Cursor コンポーネント

このコンポーネントは、アプリケーションに接続しているそれぞれのユーザに、マウス・ポインタ（カーソル）を表示します。コミュニケーション・アプリケーション内でユーザがマウスを動かせば、その動きは他のユーザの画面上に反映されます。ユーザ名がそのユーザのカーソル近くに表示され、ユーザは自分用のポインタの色を選べます。

Cursor コンポーネントは、SimpleConnect コンポーネントと併用すればスクリプティングの必要はありません。

UserColor コンポーネントとも合わせて使用できます。

**Note:**このコンポーネントがコミュニケーション・アプリケーションの一部で、ユーザがステージ上でポインタを初めて動かすと、ポインタは見えなくなります。ユーザがログインすれば、他のログインしているユーザと同様に、そのユーザのポインタも見えるようになります。

### Cursor コンポーネントを使う

このコンポーネントはアバターのシンプルな手法です。アバターとは、複数のユーザが同時操作できる仮想のステージで、ユーザに代わる象徴を意味します。このコンポーネントを使用すると、コミュニケーション・アプリケーション内のステージ上のポインタがそれぞれのユーザの場所を代わって表します。

以下はこのコンポーネントの使用法をいくつか示したものです。

**教育アプリケーション：**このコンポーネントを e-ラーニングのアプリケーションに組み込むことで、プレゼンテーションや討論、質問中に、全てのユーザがグラフや地図、式などを指し示すことができるようになります。

**使い方の研究：**例えば web サイトなどのアプリケーションにこのコンポーネントを追加することで、人々がどのようにこれを使用するか見ることができます。

**新しい双方向的モデルでの視覚的ヘルプ：**多くのユーザが同時操作するアプリケーションで、ステージ上の全てのマウス・ポインタの場所を明らかにすることで、他のユーザの動向をよく知ることができる。

### アプリケーションでコンポーネントを使用する

1 Flash Communication Server が起動していることを確認する

2 Flash Communication Server MX のアプリケーション・ディレクトリ(通常は`flashcomapplications`)にアプリケーション・ディレクトリを作成し、`cursor_test`の名前をつける。

3 アプリケーション・ディレクトリ内に `main.asc` の名前で新しいファイルを作成し、以下のコードを記述する。

```
load("components.asc");
```

**Note:**他のアプリケーション・ディレクトリからこのアプリケーション・ディレクトリへ、`main.asc` ファイルをコピーして使用することもできます。

4 ステージ上に Chat コンポーネントをドラッグする。

5 プロパティ・インスペクタで、`cursor_mc` のインスタンス名をつける。

6 ステージ上に SimpleConnect コンポーネントをドラッグする。

7 SimpleCpnnection 用のプロパティ・インスペクタで以下のパラメータを入力する。

Application Directory テキスト・ボックスで、ステップ 2 で作成したアプリケーションの URI、例えば、`rtmp:/cursor_test` を入力する。

Communication Components テキスト・ボックスをダブル・クリックする。現れる値ダイアログ・ボックスで、+ 印をクリックし、Cursor コンポーネントのインスタンス名 cursor\_mc を入力し、OK をクリックする。

この値を与えることで、SimpleConnect コンポーネントは Cursor コンポーネントの接続を制御できる。SimpleConnect コンポーネントがサーバーへの接続に成功したら、Cursor コンポーネントは自動的にサーバーに接続する。

8 アプリケーション・ディレクトリ内に保存、パブリッシュする。

9 アプリケーション・ディレクトリ内にある SWF ファイルを開くか、または Flash MX オーサリング環境では、制御 >ムービープレビューを選択しログインする。Auto チェック・ボックスをクリックする。

下記のようにポインタとログイン名が表示される。さらにアプリケーションのインスタンスを開くと、またポインタが現れる。



#### Cursor コンポーネントのリスキニング

このコンポーネントのアセットは、ライブラリ・パネルの Core Assets-Developer Only\Cursor Assets ディレクトリにあります。作成するアプリケーションに合わせて、ポインタの矢印やラベルの文字を編集できます。

#### 関係するコンポーネント

“SimpleConnect コンポーネント” 6 4 p、“UseColor コンポーネント” 6 5 P。

#### FCCursor オブジェクト

FCCursor API は、FCConectionLight API は、サーバーに接続したりスクリーンアップするオブジェクトにクライアント・サイドとサーバーサイドでの機能を提供します。またプログラミングでユーザのポインタの色を設定できます。

## FCCursor オブジェクトのメソッド・サマリー

メソッド	説明
FCCursor.connect	クライアント・サイドとサーバー・サイドでコンポーネントが必要とするアセットを全てセットアップし、サーバーが提供する名前ですetUsernameをコールする。
FCCursor.setUsername	表示し、サーバーへ渡すユーザ名を設定する。
FCCursor.close	クリーン・アップし、Mouse.showメソッドをコールしてマウス・ポインタを通常の状態に戻す。
FCCursor.setColor	ユーザのポインタの色を変更する。

### FCCursor.connect

#### 利用

Flash Player 6

Flash Communication Server MX

#### 使い方

*cursor\_mc..connect(nc)*

#### パラメータ

*nc* アクティブな NetConnection オブジェクト

#### 戻り値

なし

#### 説明

メソッド；クライアント・サイドとサーバー・サイドでコンポーネントが必要とするアセットを全てセットアップしサーバーが提供する名前ですetUsernameをコールする。

### FCCursor.setUsername

#### 利用

Flash Player 6

Flash Communication Server MX

#### 使い方

*cursor\_mc..setUsername(newName)*

#### パラメータ

*newName* 新しいユーザ名を特定するオプションのストリング。*newName* パラメータが null または undesine でない場合、このメソッドはユーザへの新しい参照を作成し、*newName* への参照名を設定、Mouse.hideメソッドをコールすることで、通常のマウス・ポインタを隠す。*newName* が null の場合は、何もせずユーザは lurker mode に入る。

#### 戻り値

なし

#### 説明

メソッド；表示、サーバーへ渡すユーザ名を設定する。

## **FCCursor.close**

### **利用**

Flash Player 6

Flash Communication Server MX

### **使い方**

*cursor\_mc.close()*

### **パラメータ**

なし

### **戻り値**

なし

### **説明**

メソッド ; クリーン・アップし、このクライアント上でコンポーネントが必要としたアセットを解放し、`Mouse.show` メソッドをコールしてマウス・ポインタを前の状態に戻す。

## **FCCursor.setColor**

### **利用**

Flash Player 6

Flash Communication Server MX

### **使い方**

*cursor\_mc.setColor(newColor)*

### **パラメータ**

*newColor* 16進法を示すストリング(例: 0xFC00F3)

### **戻り値**

なし

### **説明**

メソッド ; ユーザのポインタの色を特定の色に変更する。`Cursor` コンポーネントは、`UserColor` コンポーネントと合わせて使用することもでき、`gFlashCom.usercolor` の値の変更を監視するように作られている。この値への変更はコールされる `setColor` によるものであり、全クライアントへの色も更新される。

## **PeopleList コンポーネント**

このコンポーネントは、コミュニケーション・アプリケーションに接続しているユーザのリストを表示します。ユーザ名を入力したユーザだけがリストに表示されます。

このコンポーネントは、`SimpleConnect` コンポーネントと併用すれば、スクリプティングの必要はなくなります。

## PeopleList コンポーネントを使用する

このコンポーネントはコミュニケーション・アプリケーションの中で標準的なもので、現在ログインしているユーザのリストを提供します。

以下はこのコンポーネントの使用法を2つ紹介したものです。

チャット；PeopleList コンポーネントと Chat、SimpleConnect コンポーネントを合わせて使用することで、チャットルーム・アプリケーションの基本部分を作成できます。

仮想会議；このコンポーネントを使用して、仮想会議部屋を作成し、現在会議に参加している全ユーザを表示できます。

## PeopleList コンポーネントのリスキニング

このコンポーネントのアセットは、ライブラリ・パネルの Core Assets-Developer Only\PeopleList Assets ディレクトリにあります。作成するアプリケーションに合わせて、ポインタ矢印やラベルの文字を編集できます。

## 関係するコンポーネント

“SimpleConnect コンポーネント”64 p、“Chat コンポーネント”20 p。

## FCPeopleList オブジェクト

FCPeopleList API は、サーバーに接続したりクリーンアップするオブジェクトに、クライアント・サイドとサーバー・サイドでの機能を提供します。またプログラミングを施せば lurker mode を設定することもできます。

### 変数

名前	変数	説明
Lurkers	lurkers	整数；新しいユーザが接続したり切断すると必ず更新される変数。アプリケーションに接続して  いるユーザ数と等しいが、ユーザ名は供給しない。この変数は取得できるが、設定はできない。
Uers	users	整数；lurkers と類似した変数。アプリケーションに接続しているユーザ数と等しく、ユーザ名を供給する値。この変数は取得できるが、設定はできない。

## FCPeopleList オブジェクトのメソッド・サマリー

メソッド	説明
FCPeopleList.connect	クライアント・サイドとサーバー・サイドでコンポーネントが必要とする全てのアセットをセットアップし、サーバーが提供する名前でも setUsername フังก์ションをコールする。
FCPeopleList.setUsername	アクティブなユーザのリストに表示するユーザ名を設定する。
FCPeopleList.close	クリーンアップする。

## **FCPeopleList.connect**

### **利用**

Flash Player 6

Flash Communication Server MX

### **使い方**

*peopleList\_mc.connect(nc)*

### **パラメータ**

*nc* アクティブな NetConnection オブジェクト

### **戻り値**

なし

### **説明 ;**

メソッド ; クライアント・サイドとサーバー・サイドでコンポーネントが必要とする全てのアセットをセットアップし、サーバーが提供する名前ですetUsername フังก์ションをコールする。

## **FCPeopleList.setUsername**

### **利用**

Flash Player 6

Flash Communication Server MX

### **使い方**

*peopleList\_mc.setUsername(newName)*

### **パラメータ**

*newName* 新しいユーザ名を記述するオプションのストリング。*newName* が null または undefined でない場合、*newName* はユーザのリストに表示される。*newName* が null の場合は何も変更されず、ユーザは lurker mode に入る。

### **戻り値**

なし

### **説明 ;**

メソッド ; アクティブなユーザのリストに表示されるユーザ名を設定する。

## FCPeopleList.close

### 利用

Flash Player 6

Flash Communication Server MX

### 使い方

```
peopleList_mc.close()
```

### パラメータ

なし

### 戻り値

なし

### 説明；

メソッド；クライアント上のこのコンポーネントが使用しているアセットを解放、クライアントのユーザ・リストからユーザを消去し、クリーン・アップする。

## PresentationSWF コンポーネント

PresentationSWF コンポーネントは、共有する別の SWF ファイルを見せるプレゼンテーションを可能にします。

このコンポーネントは2つのモードで動作が可能です。*speaker mode*ではユーザがプレゼンターとなり、SWF ファイルを制御すると同時に、全てのユーザが同じフレームを見ます。*default mode*では、ユーザは視聴者となり、Next ボタンと Back ボタンを使うことで SWF ファイルを行き来し、非同期にファイルを見ることができます。しかし、プレゼンターがまだ行っていないプレゼンテーションを先に行って見ることはできません。

PresentationSWF コンポーネントは、SimpleConnection コンポーネントと合わせて使用すれば、スクリプティングの必要はなくなります。

### presentationSWF ファイルを設定する

このコンポーネントでは、プレゼンテーション用 SWF としてロードする別の SWF ファイルが必要になります。以下のサンプルでは、Flash Communication Server MX をインストールした flashcom¥applications¥sample\_broadcast ディレクトリにある、simple\_preso.swf ファイルをコピーして使用します。

このコンポーネントに表示するプレゼンテーションを作成するときは、SWF ファイルを適切に設定する必要があります。

ファイルは、メインのタイムライン上にそれぞれのフレームがひとつのプレゼンテーション・スライドとしてある構成になっていなくてはなりません。

1 番目のフレームに stop アクションが書かれていることを確認します。

プレゼンテーション SWF ファイルには ActionScript を含まないようにしてください。メインのタイムラインを行き来するようなアクションは、コンポーネントに含まれています。

### PresentationSWF コンポーネントを使用する

以下はこのコンポーネントの2通りの使用方法を記したものです。

セールス・プレゼンテーション：チャートや他の情報を示しながら、オンラインでプレゼンテーションを行います。

ミーティング・アジェンダ：会議中ユーザの注意を引きつけ、別のアジェンダに導きます。

以下のサンプルでは、2つのクライアント・ファイルを作成します。presentSWF\_test\_speaker はスライドを制御するプレゼンター用として、presentsSWF\_test は、プレゼンターではない参加視聴者用として作成します。どちらも presentSWF\_test という同じアプリケーションに注目させますが、speaker mode にあるクライアントだけがプレゼンターになることができます。

### 話者アプリケーションでこのコンポーネントを使用する

1 Flash Communication Server が起動していることを確認する

2 Flash Communication Server MX のアプリケーション・ディレクトリ(通常は%flashcom%applications)にアプリケーション・ディレクトリを作成し、presentsSWF\_test の名前をつける。

3 アプリケーション・ディレクトリ内に main.asc の名前で新しいファイルを作成し、以下のコードを記述する。

```
load("components.asc");
```

**Note:**他のアプリケーション・ディレクトリからこのアプリケーション・ディレクトリへ、main.asc ファイルをコピーして使用することもできます。

4 Flash MX で、ステージ上に presentationSWF コンポーネントをドラッグし、プロパティ・インスペクタで、presentSWF\_mc のインスタンス名をつける。

5 プロパティ・インスペクタで、デフォルトの presentationSWF ファイル名、simple\_preso.awf は変更しない。他のパラメータについては"PresentationSWF コンポーネント・プロパティ・インスペクタ"をご覧ください。

6 ステージ上に SimpleConnect コンポーネントをドラッグする。

7 SimpleCpnnection 用のプロパティ・インスペクタで以下のパラメータを入力する。

Application Directory テキスト・ボックスで、ステップ 2 で作成したアプリケーションの URI、例えば、rtmp:/presentSWF\_test を入力する。

Communication Components テキスト・ボックスをダブル・クリックする。現れる値ダイアログ・ボックスで、+ 印をクリックし、PresentationSWF コンポーネントのインスタンス名 presentSWF\_mc を入力し、OK をクリックする。

この値を与えることで、SimpleConnect コンポーネントは PresentationSWF コンポーネントの接続を制御できる。SimpleConnect コンポーネントがサーバーへの接続に成功したら、PresentationSWF コンポーネントは自動的にサーバーに接続する。

8 アクション・パネルの 1 番めのキーフレームにムービーの進行を止め、speaker mode に設定する以下のコードを記述する。

```
stop();
```

```
_global.speakerMode = true;
```

9 アプリケーション・ディレクトリ内に presentSWF\_test\_speaker 名で保存、パブリッシュする。

### 視聴者アプリケーションで PresentationSWF コンポーネントを使用する

1 presentSWF\_test\_speaker.fla をコピーし、presentSWF\_test.fla 名で保存する。

2 presentSWF\_test.fla 内で、アクション・パネルの 1 番めのキーフレームに、ユーザがプレゼンターにならないよう、stop コールの後に以下のコードを記述する。

```
_global.speakerMode = false;
```

8 presentSWF\_test.fla を保存する。

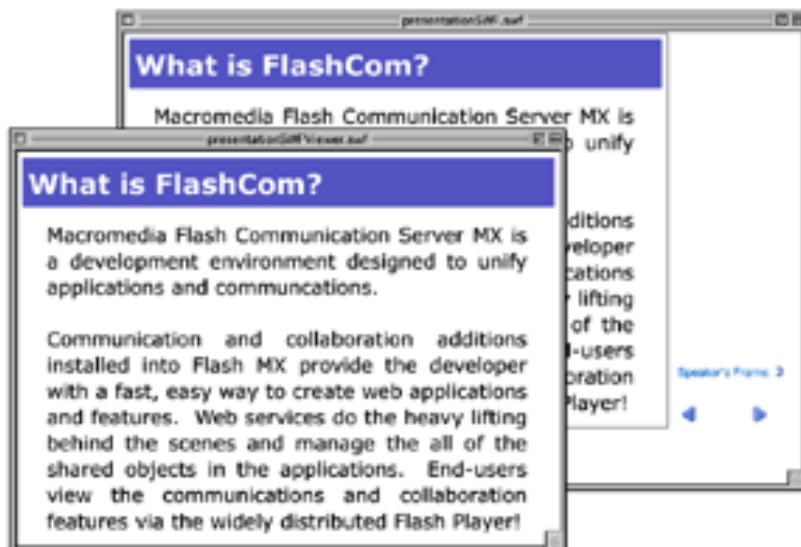
### 両方のモードで PresentationSWF コンポーネントをテストする

1 アプリケーション・ディレクトリで presentSWF\_test\_speaker.swf ファイルを開くか、Flash MX オーサーリング環境の場合は、制御>ムービプレビューを選択し、ログインする。

2 アプリケーション・ディレクトリで presentSWF\_test.swf ファイルを開くか、Flash MX オーサーリング環境の場合は、制御>ムービプレビューを選択し、別のユーザとしてログインする。

presentSWF\_test\_speaker.swf では、ユーザはプレゼンターとなる。新しいスライドへ移動し、プレゼンテーション中先へ進んだり、戻ったりできる。presentSWF\_test.swf では、ユーザは視聴者となる。スライドを見ることはできるが、プレゼンターが進んだところまでしか移動することはできない。

以下は、後ろにあるのがプレゼンター・クライアント・アプリケーションで、前にあるのが視聴者プレゼンテーション。



## PresentationSWF コンポーネント・プロパティ・インスペクタ

名前	変数	説明
PresentationSWF	swFile	ストリング; デフォルト値は simple_preso.swf。プレゼンテーションで  ロードするファイルを特定する。LoadSWF メソッドを使って動作中このファイルを変更することもできる。
Viewer Button Enabled	viewerButtons	Boolean 値; デフォルト値は true。true なら speaker mode でないユーザもプレゼンテーション・ファイルを行き来できる。

## PresentationSWF コンポーネントのリスキニング

このコンポーネントのアセットは、ライブラリ・パネルの Core Assets-Developer Only PresentationSWF Assets ディレクトリにあります。

## 関係するコンポーネント

“SimpleConnect コンポーネント” 6 4 p

## FCPresentation オブジェクト

FCPresentation API は、サーバーに接続したりクリーンアップするオブジェクトに、クライアント・サイドとサーバー・サイドでの機能を提供します。またプログラミングでフレームを操作することもできます。

## FCPresentationSWF オブジェクト・メソッド・サマリー

メソッド	説明
FCPresentationSWF.connect	クライアント・サイドとサーバー・サイドでコンポーネントが必要とする全てのアセットをセットアップします。
FCPresentationSWF.close	クリーン・アップし接続を切断します。
FCPresentationSWF.loadSWF	プレゼンテーションで使用する新たな SWF ファイルを設定するためにプレゼンターがオプションで使用します。
FCPresentationSWF.next	1 フレームプレゼンテーション SWF を進めます。
FCPresentationSWF.back	プレゼンテーション SWF 内で 1 フレーム戻ります。
FCPresentationSWF.sync	視聴者のフレームとプレゼンターのフレームを再同調させます。

## 変数

### グローバル変数

変数	説明
<code>_global.speakerMode</code>	Boolean 値; デフォルト値は true。true の場合、他のユーザが見ている今のスライドを変更できます。false なら、プレゼンターが見せるスライドを見るだけです。この変数をプログラミングで設定するには、 <code>_global.sepeakerMode =[true または false]</code> に設定します。

## **\_global.speakerMode**

### **利用**

Flash Player 6

Flash Communication Server MX

### **使い方**

```
_global.speakerMode = true;
```

### **説明；**

グローバル変数； speaker mode を制御する Boolean 値。true(デフォルト値)ならユーザは他のユーザが見ている現在のスライドを変更できる。False ならユーザは、プレゼンターが選んで見せるスライドのみ見ることができる。この変数を設定するには、\_global.speakerMode を true か false に設定する。

## **FCPresentationSWF.connect**

### **利用**

Flash Player 6

Flash Communication Server MX

### **使い方**

```
presentSWF_mc.connect(nc)
```

### **パラメータ**

*nc*            アクティブな NetConnection オブジェクト

### **戻り値**

なし

### **説明；**

メソッド；クライアント・サイドとサーバー・サイドでコンポーネントが必要とする全てのアセットをセットアップする。

## **FCPresentationSWF.close**

### **利用**

Flash Player 6

Flash Communication Server MX

### **使い方**

```
presentSWF_mc.close()
```

### **パラメータ**

なし

### **戻り値**

なし

### **説明；**

メソッド；クライアント上でこのコンポーネントが使用したアセットを解放し、クリーンアップする。

## FCPresentationSWF.loadSWF

### 利用

Flash Player 6

Flash Communication Server MX

### 使い方

*presentSWF\_mc.loadSWF(swfFile)*

### パラメータ

*swfFile* SWF ファイルへの相対または絶対パスを指定するストリング。

### 戻り値

なし

### 説明；

メソッド； プレゼンテーションで使用する新たな SWF ファイルを設定するためにプレゼンターがオプションで使用する。このメソッドは、視聴者がファイルをロードするためには使用できない。SWF ファイルはプレゼンターが新しい  
フ

ァイルをロードすると自動的にロードされる。

## FCPresentationSWF.next

### 利用

Flash Player 6

Flash Communication Server MX

### 使い方

*presentSWF\_mc.next()*

### パラメータ

なし

### 戻り値

なし

### 説明；

メソッド； 1 フレームだけプレゼンテーション SWF を進める。speaker mode であれば、フレームは全ての視聴者と  
同調して進む。speaker mode でなく視聴者のボタンが実行可能の場合は、視聴者はローカルで（他のユーザには影響を  
与えず）フレームを進めることができるが、プレゼンターの現在のフレームを超えて先を見ることはできない。

## FCPresentationSWF.back

### 利用

Flash Player 6

Flash Communication Server MX

### 使い方

*presentSWF\_mc.back()*

### パラメータ

なし

## 戻り値

なし

## 説明；

メソッド； プレゼンテーション SWF 内で 1 フレーム戻る。ユーザが speaker mode であれば、フレームは全ての視聴者と同調して戻る。ユーザが speaker mode でなく視聴者のボタンが実行可能な場合は、ローカルで変更でき、他の視聴者には影響しない。

FCPresentationSWF.sync

## 利用

Flash Player 6

Flash Communication Server MX

## 使い方

`presentSWF_mc.sync()`

## パラメータ

なし

## 戻り値

なし

## 説明；

メソッド； 視聴者のフレームと話者のフレームを再同調させる。

## PresentationText コンポーネント

このコンポーネントは、共有した文字でのプレゼンテーションを可能にします。PresentationSWF のように、PresentationText コンポーネントは、speaker mode または default mode で動作します。speaker mode では、ユーザはリアルタイムでスライドの編集や、現在のスライドの変更ができ、さらにスライドを加えたり消去したりできます。default mode では、ユーザはプレゼンテーションを見られますが、プレゼンターが達していないスライドへ進むことはできません。

PresentationText コンポーネントは、SimpleConnect コンポーネントと合わせて使えば、スクリプティングの必要はありません。

### PresentationText コンポーネントを使う

このコンポーネントは、様々なプレゼンテーション・アプリケーションに追加して使用することができます。

以下はこのコンポーネントの 2 つの使用例です。

トレーニング・セミナー：トレーニング・アプリケーションで、メモをとったり、サンプルの概略説明に用います。

ミーティング・ノート：ミーティング・アプリケーションで、ユーザを現在の討議事項に注目させます。

以下の例では、2 つのクライアント・ファイルを作成します。PresentText\_test\_speaker はプレゼンテーションのスライドや概略を制御するプレゼンター用のもので、presentText\_test は、プレゼンターとまらない参加者が見るためのものです。これらのクライアントは両方とも同じアプリケーション、presentText\_test に向かいますが、speaker mode のクライアントだけがプレゼンターになることができます。

### 話者アプリケーションで PresentationText コンポーネントを使用する

1 Flash Communication Server が起動していることを確認する

2 Flash Communication Server MX のアプリケーション・ディレクトリ(通常は`flashcom`applications)にアプリケーション・ディレクトリを作成し、`presentText_test`の名前をつける。

3 アプリケーション・ディレクトリ内に `main.asc` の名前で新しいファイルを作成し、以下のコードを記述する。

```
load("components.asc");
```

**Note:**他のアプリケーション・ディレクトリからこのアプリケーション・ディレクトリへ、`main.asc` ファイルをコピーして使用することもできます。

4 Flash MX で、ステージ上に `presentationText` コンポーネントをドラッグし、プロパティ・インスペクタで、`presentText_mc` のインスタンス名をつける。

**Note:**プロパティ・インスペクタでの唯一のプロパティは、`speaker mode`。デフォルトでの設定は `true`。これによりクライアントはプレゼンターになることができる。

5 ステージ上に `SimpleConnect` コンポーネントをドラッグする。

6 `SimpleCpnnection` 用のプロパティ・インスペクタで以下のパラメータを入力する。

Application Directory テキスト・ボックスで、ステップ 2 で作成したアプリケーションの URI、例えば、`rtmp:/presentText_test` を入力する。

Communication Components テキスト・ボックスをダブル・クリックする。現れる値ダイアログ・ボックスで、`+` 印をクリックし、`PresentationText` コンポーネントのインスタンス名 `presentText_mc` を入力し、`OK` をクリックする。

この値を与えることで、`SimpleConnect` コンポーネントは `PresentationText` コンポーネントの接続を制御できる。`SimpleConnect` コンポーネントがサーバーへの接続に成功したら、`PresentationText` コンポーネントは自動的にサーバーに接続する。

7 アプリケーション・ディレクトリ内に `presentText_test_speaker fla` として保存、パブリッシュする。

### 視聴者アプリケーションで Presentation コンポーネントを使用する

1 `PresentText_test_speaker fla` をコピーしファイル名を `presentText fla` にする。

2 `presentText fla` 内で、`PresentationText` コンポーネントを選択し、プロパティ・インスペクタで `speaker mode` のパラメータを `false` に変更する。

3 `presentText_test fla` を保存する。

### 両方のモードで PresentationText コンポーネントをテストする

1 アプリケーション・ディレクトリで `presentText_test_speaker.swf` を開くか、Flash MX オーサリング環境では、制御>ムービープレビューを選択し、ログインする。

2 アプリケーション・ディレクトリで presentText\_test.swf を開くか、Flash MX オーサリング環境では、制御>ムービープレビューを選択し、別のユーザ名でログインする。

presentText\_test\_speaker.swf では、ユーザはプレゼンターとなり、プレゼンテーション中新しいスライドを作成したり、先へ進んだり戻ったりできる。またプレゼンテーションのアウトライン情報も見ることができる。

presentText\_test.swf では、ユーザは視聴者となる。スライドを見ることはできるが、プレゼンターがスライドを進めるようにしか見ることはできない。

以下は、後ろにあるのがプレゼンター・クライアント・アプリケーションで、前にあるのが視聴者プレゼンテーション。



#### **PresentationText コンポーネントのリスキニング**

このコンポーネントの資産は、ライブラリ・パネルの Core Assets-Developer Only¥PresentationText Assets ディレクトリにあります。

#### **関係するコンポーネント**

“SimpleConnect コンポーネント” 6 4 p

#### **FCPresentationText オブジェクト**

FCPresentationText API は、サーバーに接続したりクリーンアップしたりするオブジェクトにクライアント・サイドとサーバーサイドでの機能を提供します。またプログラミングでスライドを操作することもできます。

## FCPresentationText オブジェクトのメソッド・サマリー

### メソッド

### 説明

FCPresentationText.connect	クライアント・サイドとサーバー・サイドでコンポーネントが必要とする全てのアセットをセットアップします。
FCPresentationText.setUsername	表示しサーバーに渡すユーザ名を設定します。
FCPresentationText.close	クリーン・アップし接続を切断します。
FCPresentationText.nextSlide	リストの次のスライドへ移動します。
FCPresentationText.backSlide	リストの1つ前のスライドへ戻ります。
FCPresentationText.newSlide	新しいスライドを作り、スライドのリストに加えます。
FCPresentationText.deleteSlide	現在のスライドを消去します。

### FCPresentationText.connect

#### 利用

Flash Player 6

Flash Communication Server MX

#### 使い方

*presentText\_mc.connect(nc)*

#### パラメータ

*nc* アクティブな NetConnection オブジェクト

#### 戻り値

なし

#### 説明 ;

メソッド ; クライアント・サイドとサーバー・サイドでコンポーネントが必要とする全てのアセットをセットアップします。

### FCPresentationText.setUsername

#### 利用

Flash Player 6

Flash Communication Server MX

#### 使い方

*presentText\_mc.setUsername(newName)*

#### パラメータ

*newName* 新しいユーザ名を特定するオプションのストリング。*newName* が null または undefined でない場合、ユーザはコンポーネントの全ての機能を使用できる。*newName* が null の場合は何も変更されず、ユーザは lurker mode に入る。

#### 戻り値

なし

説明 ; メソッド ; 表示しサーバーに渡すユーザ名を設定する。

## **FCPresentationText.close**

### **利用**

Flash Player 6

Flash Communication Server MX

### **使い方**

*presentText\_mc.close()*

### **パラメータ**

なし

### **戻り値**

なし

### **説明；**

メソッド；このクライアントが使用していたアセットを解放し、このコンポーネントの接続を切断することで、クリーンアップする。

## **FCPresentationText.nextSlide**

### **利用**

Flash Player 6

Flash Communication Server MX

### **使い方**

*presentText\_mc.nextSlide()*

### **パラメータ**

なし

### **戻り値**

なし

### **説明；**

メソッド；リスト内の次のスライドへ移動する。

## **FCPresentationText.backSlide**

### **利用**

Flash Player 6

Flash Communication Server MX

### **使い方**

*presentText\_mc*.backSlide()

### **パラメータ**

なし

### **戻り値**

なし

### **説明；**

メソッド；リスト内の前のスライドへ移動する。

## **FCPresentationText.newSlide**

### **利用**

Flash Player 6

Flash Communication Server MX

### **使い方**

*presentText\_mc*.newSlide()

### **パラメータ**

なし

### **戻り値**

なし

### **説明；**

メソッド；新しいスライドを作成しスライドのリストに加える。

## FCPresentationText.deleteSlide

### 利用

Flash Player 6

Flash Communication Server MX

### 使い方

```
presentText_mc.deleteSlide()
```

### パラメータ

なし

### 戻り値

なし

### 説明；

メソッド；現在のスライドを削除する。

## RoomList コンポーネント

RoomList コンポーネントは、チャット・アプリケーションのように、ユーザが部屋を作成、参加し、削除する上級のコンポーネントです。このコンポーネントは、他のアプリケーションへのユーザ・アクセスを管理する使い勝手の良いロビー・アプリケーションに不可欠です。

Create Room をクリックすると、部屋名と説明のフィールドのあるポップ・アップ・メニューが現れます。部屋を選択し、Join をクリックすることで、新しいブラウザ・ウィンドウが開き、選択した部屋、つまり別のアプリケーション・インスタンスが現れます。誰もいなければユーザはリストから部屋を消去できます。コンポーネントの全機能を使うには、ユーザはユーザ名を入力してサーバーに接続する必要があります。

### RoomList コンポーネントを使用する

このコンポーネントではロビーと部屋の2つのアプリケーションを作成します。ロビー・アプリケーションはユーザのスタート地点となり、部屋アプリケーションは、ユーザの行き先となります。

*Note:* ユーザはローカル、リモート環境に関わらず、web ブラウザを使用してこのアプリケーションにアクセスします。例えば、典型的な開発環境では、<http://localhost> でアプリケーションをテストします。

RoomList コンポーネントは、SimpleConnect コンポーネントを使用すれば、クライアント・サイドでのスクリプティングの必要はなくなります。ロビー・アプリケーションには、components.asc ファイルをロードする main.asc ファイルを作成する必要があります。

部屋アプリケーションでは、components.asc ファイルをロードする main.asc ファイルの中で、サーバー・サイド・スクリプトを記述する必要があります。それによってアプリケーションは、ユーザ名と部屋名を得、部屋数とユーザ数を管理します。この main.asc ファイルには、前後にパラメータを渡すサーバー・サイドのコールを含みますが、このコールにはユーザが実際に部屋アプリケーションに接続しているかどうかを制御するコールは含みません。サーバー・サイドの機能は、ただロビー・アプリケーションの部屋で表示されている人数を更新し、アプリケーションにアプリケー

シ

ョン名を渡すだけです。このことによってそれぞれにインスタンスにおいて、ダイナミックに部屋名が表示されるので

१.

RoomList コンポーネントから開いた部屋アプリケーションは、送信する username と appInstance パラメータを使用する必要がありますが、SimpleConnect アプリケーションを使用すれば、この過程は自動的に処理されます。例えば、Flash Communication Server MX とともにインストールされる sample\_room アプリケーションをご覧ください。

ロビー・アプリケーションでは、以下の手順を踏みます。

- 1 SimpleConnect コンポーネントを加える
- 2 RoomList コンポーネントを加える
- 3 ロビー・アプリケーションを拡張する components.asc ファイルをロードする。

この後、部屋アプリケーションでは、以下の手順を踏みます。

- 4 以前作成したチャット・ルーム・アプリケーションに手を加え、行き先にします。
- 5 ロビー・アプリケーションからチャット・ルームに値を渡すため、HTML を変更します。

以下は、このコンポーネントを使用した2つの例です。

チャット・ロビー：ユーザが集まる中央ロビー、会議場所を作成します。その後参加するには部屋を選択します。

アプリケーション・インスタンス・ランチャー：仮想センターを作成し、そこからユーザはアプリケーションの多様なインスタンスを生み出します。

*Note:*以下のサンプルでは、2 1 ページの”Chat コンポーネントを使用する”のサンプルを作成したものと仮定しています。

#### ロビー・アプリケーションでコンポーネントを使用する

- 1 Flash Communication Server が起動していることを確認する
- 2 Flash Communication Server MX のアプリケーション・ディレクトリ(通常は`flashcomapplications`)にアプリケーション・ディレクトリを作成し、lobby\_com\_test の名前をつける。
- 3 Flash MX で、ステージ上に SimpleConnect コンポーネントをドラッグする。
- 4 SimpleCpnnection 用のプロパティ・インスペクタで以下のパラメータを入力する。  
Application Directory テキスト・ボックスで、ステップ 2 で作成したアプリケーションの URI、例えば、`rtmp:/lobby_com_test` を入力する。  
Communication Components テキスト・ボックスをダブル・クリックする。現れる値ダイアログ・ボックスで、+ 印をクリックし、RoomList コンポーネントのインスタンス名 `roomList_mc` を入力し、OK をクリックする。
- 5 RoomList コンポーネントをステージ上にドラッグする。
- 6 RoomList コンポーネント用のプロパティ・インスペクタで、下記の値を入力する。  
インスタンス名に `room_list_mc` をつける。この値は部屋アプリケーションの main.asc ファイル内で使用される。  
Room Application Path テキスト・ボックスで、部屋アプリケーションの相対位置を入力する。例えば、`./chat_room_test/chat_room_test.html` を入力する。これは `chat_room_test.fla` をパブリッシュしたときに生成される部屋アプリケーションの HTML ファイル名。
- 7 lobby\_com\_test.fla として、web サーバーのアプリケーション・ディレクトリに保存し、パブリッシュする。

8 lobby\_com\_test アプリケーション・ディレクトリで、main.ascを作成し、以下のコードを記述する。

```
load("components.asc");
```

#### ロビー・アプリケーションの行き先として働く部屋アプリケーションを作成する

1 2.1ページの"チャットコンポーネントを使う"で作成した chat\_test アプリケーションをコピーする。Web サーバーのアプリケーション・ディレクトリ名を chat\_room\_test に変え、コピーした FLA ファイル名を chat\_room\_test fla に変える。

2 chat\_room\_test fla ファイルで、ステージ上の Simple Connect コンポーネントを選択し、Application Directory の設定を rtmp:/chat\_room\_test に変更する。

3 SimpleConnect 用のプロパティ・インスペクタで、コンポーネント・インスタンス名を connector\_mc にする。この値は行き先の部屋の main.asc ファイルからコールされる。

4 ファイルを chat\_room\_test アプリケーション・ディレクトリに保存しパブリッシュする。

5 HTML エディタで chat\_room\_test.html を開き、<BODY>...</BODY>タグ内を全て中に収まるように置き換える。

```
<BODY bgcolor="#FFFFFF">
<script language=JavaScript1.1>
<!--
    var appURL = String(document.location);
    if (appURL.indexOf("?") != -1){
        var appParams = appURL.substr(appURL.indexOf("?"));
    }else{
        var appParams = "";
    }
    document.write('<OBJECT classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000">');
    document.write('codebase="http://download.macromedia.com/pub/shockwave/cabs/flash/swflash.cab#version=6,0,0,0" ');
    document.write(' WIDTH="550" HEIGHT="400" id="chat_room_test" ALIGN="">');
    document.write('<PARAM NAME=movie VALUE="chat_room_test.swf" + appParams + ">');
    document.write('<PARAM NAME=quality VALUE=high> <PARAM NAME=bgcolor VALUE=#FFFFFF>');
    document.write('<EMBED src="chat_room_test.swf" + appParams + " quality=high bgcolor=#FFFFFF');
    document.write(' swLiveConnect=FALSE WIDTH="550" HEIGHT="300" NAME="chat_room_test"
    ALIGB=""');
    document.write('TYPE="application/x-shockwave-flash"
    PLUGINSPPAGE="http://www.macromedia.com/go/getflashplayer">');
    document.write('</embed>');
    document.write('</object>');
    //-->
</script></BODY>
```

*Note:*HTML は URL からパラメータを引き出し、部屋アプリケーションに渡す。これは web サーバーから、例えば <http://localhost> で、ページを見たときのみ機能する。またエレメントの値は行き先の部屋アプリケーションと、SWF ファイル名に合っていないとてはならない。前の HTML では、例えば chat\_room\_test と chat\_room\_test.swf という風に。Flash MX からまたムービーをパブリッシュしたら、この HTML ファイルは上書きされてしまう。

6 次に、chat\_room\_test アプリケーションの main.asc ファイルに、ロビー・アプリケーションに行き先へ向かわせるコードを加える。

*Note:*chat\_room\_test アプリケーション・ディレクトリの main.asc ファイルは、すでに以下のサーバー・サイド ActionScript を含んでいる。

```
load("components.asc");
```

chat\_room\_test アプリケーションの main.asc ファイル内に、ユーザをクライアントのグローバル・リストに加え、ユーザの接続を許可し、部屋インスタンス名を取得、部屋数を更新する onConnect ハンドラを記述する。

```
application.onConnect = function(newClient,username,password){  
    gFrameworkFC.getClientGlobals(newClient).username = username;  
    application.acceptConnection(newClient);  
    if(this.name.indexOf("/") != -1){  
        newClient.room = this.name.substr(this.name.lastIndexOf("/")+1);  
        roomConnect(newClient);  
    }  
}
```

7 roomConnect コールの結果を受け取り、結果を SimpleConnect コンポーネントに渡す roomResult ハンドラを記述する。

```
function roomResult(newClient){  
    this.onResult = function(roomName){  
        newClient.call("FCSimpleConnect/connector_mc/roomName",null,roomName);  
    }  
}
```

**8** ロビーに接続し、ユーザが選択した部屋名を取得、戻り値で roomResult をコールする。

```
function roomConnect(newClient,room){
    lobby_nc = new NetConnection();
    lobby_nc.onStatus = function(infoStatus){
        if(infoStatus.code == "NetConnection.Connect.Success"){
            lobby_nc.call("FCRoomList/roomlist_mc/roomConnect",new
                roomResult(newClient),newClient.room);
        }
    };
    lobby_nc.connect("rtmp://localhost/lobby_com_test");
}
```

**9** ロビー・アプリケーションに通知する onDisconnect ハンドラを記述する。

```
application.onDisconnect = function(client){
    if(client.room != null){
        roomDisconnect(client.room);
    }
}
```

**10** 部屋数を合わせ、表示からユーザを消す。

```
function roomDisconnect(room){
    lobby_nc = new NetConnection();
    lobby_nc.onStatus = function(infoStatus){
        if(infoStatus.code == "NetConnection.Connect.Success"){
            lobby_nc.call("FCRoomList/roomlist_mc/roomDisconnect",null,room);
        }
    }
    lobby_nc.connect("rtmp://localhost/lobby_com_test");
}
```

**11** main.asc を保存し、Communication App Inspector からアプリケーションをリロードする

### RoomList コンポーネントをテストする

1 web ブラウザで、lobby\_com\_test.html を開く。

例えば、[http://localhost/flashcom/applications/lobby\\_com\\_test/lobby\\_com\\_test.html](http://localhost/flashcom/applications/lobby_com_test/lobby_com_test.html)になる。ログインする。

2 Create Room を選択し、部屋リストに部屋を加える。

3 作成した部屋をクリックし、Join Room をクリックする。

4 チャットができ、文字色も変更できる。

5 lobby\_com\_test アプリケーションの別のインスタンスを開き、大勢のユーザでチャットしているかのように、別の名前でログインする。

### RoomList コンポーネント・プロパティ・インスペクタ

名前	変数	説明
----	----	----

Room Application Path	roomPath	ストリング；部屋に加わったとき出すアプリケーションへのパスを与える。パスは絶対でも相対でもかまわない。
-----------------------	----------	---

### RoomList コンポーネントのリスキニング

このコンポーネントのアセットは、ライブラリ・パネルの Core Assets-Developer Only\RoomList Assets ディレクトリにあります。

### 関係するコンポーネント

“SimpleConnect コンポーネント” 6 4 p

### FCRoomList オブジェクト

FCRoomList API は、サーバーに接続したりクリーンアップしたりするオブジェクトに、クライアント・サイドとサーバー・サイドの機能を提供します。またプログラミングで、部屋を操作することもできます。

## FCRoomList オブジェクトのメソッド・サマリー

### メソッド

### 説明

FCRoomList.connect	クライアント・サイドとサーバー・サイドでコンポーネントが必要とするアセットを全てセットアップし、サーバーが与える名前で setUsername メソッドをコールします。
FCRoomList.setUsername	表示し、サーバーに渡す名前を設定します。
FCRoomList.close	クリーン・アップしコンポーネントを切断します。
FCRoomList.createRoom	Create Room ダイアログ・ボックスを開き、一時的に Join、Create、Delete の各ボタンを使用不可にします。
FCRoomList.deleteRoom	現在選択されている部屋を消去するよう、サーバーにコールします。
FCRoomList.joinRoom	新しいブラウザ・ウィンドウを出し、選択された部屋のインスタンスを開きます。

### FCRoomList.connect

### 利用

Flash Player 6

Flash Communication Server MX

### 使い方

`roomList_mc.connect(nc)`

### パラメータ

`nc` アクティブな NetConnection オブジェクト

### 戻り値

なし

### 説明 ;

メソッド ; クライアント・サイドとサーバー・サイドでコンポーネントが必要とするアセットを全てセットアップし、サーバーが与える名前で setUsername メソッドをコールする。

## **FCRoomList.setUsername**

### **利用**

Flash Player 6

Flash Communication Server MX

### **使い方**

*roomList\_mc.setUsername(newName)*

### **パラメータ**

*newName* 新しいユーザを特定するオプションのストリング。*NewName* が null または undefined でない場合、ユーザは部屋のリストを見、参加、作成、消去はできる。*newName* が null の場合は、何も変化せずユーザは lurker mode に入る。

### **戻り値**

なし

### **説明；**

メソッド；表示し、サーバーに渡すユーザ名を設定する。

## **FCRoomList.close**

### **利用**

Flash Player 6

Flash Communication Server MX

### **使い方**

*roomList\_mc.close()*

### **パラメータ**

なし

### **戻り値**

なし

### **説明；**

メソッド；クライアント上でこのコンポーネントが使用していたアセットを解放し、サーバーからコンポーネントを切断することで、クリーンアップする。

## **FCRoomList.createRoom**

### **利用**

Flash Player 6

Flash Communication Server MX

### **使い方**

`roomList_mc.createRoom()`

### **パラメータ**

なし

### **戻り値**

なし

### **説明；**

メソッド； Create Room ダイアログ・ボックスを開き、一時的に Join、Create、Delete の各ボタンを使用不可にする。

## **FCRoomList.deleteRoom**

### **利用**

Flash Player 6

Flash Communication Server MX

### **使い方**

`roomList_mc.deleteRoom()`

### **パラメータ**

なし

### **戻り値**

なし

### **説明；**

メソッド； 現在選択されている部屋を消去するよう、サーバーにコールする。

## FCRoomList.joinRoom

### 利用

Flash Player 6

Flash Communication Server MX

### 使い方

```
roomList_mc.joinRoom()
```

### パラメータ

なし

### 戻り値

なし

### 説明；

メソッド； 新しいブラウザ・ウィンドウを出し、選択された部屋のインスタンスを開く。アプリケーション・インスタンスとユーザ名は、パラメータとして新しいウィンドウに送られる。

## SetBandwidth コンポーネント

このコンポーネントにより、ユーザはアップロードとダウンロードの帯域幅を特定させることができます。この新しい設定が、コンポーネントが操作するマイクやカメラの品質を左右します。これにより複数のマイクやカメラを制御することができ、帯域幅に優先順位をつけて割り振られます。

ユーザはアップロード、ダウンロードで Modem、DSL、LAN、Custom を選択できます。Custom を選択すると、現れるダイアログ・ボックスで、ユーザは固定された帯域幅数を選ぶか、帯域幅数を入力できます。帯域幅は対称か非対称かどちらかに設定されます。ユーザが非対称オプションを選んだ場合は、アップロードとダウンロードの数字は等しくなりません。

SetBandwidth コンポーネントは、ユーザが選択したアップロードとダウンロードの帯域幅を制限する、サーバー・サイドの複製を使用します。帯域幅の制限により、接続に可能な帯域幅を Flash Communication Server は認知し、一般的に音声と映像の品質を向上させます。

*Optimal settings* でのパブリッシュが望まれます。これは、接続者が受信するより高いレートにはしない、という設定です。例えば、オンラインのプレゼンターは高スピードで接続していますが、視聴者はモデムや DSL 接続で見

ているといった場合、モデムのスピードより速い設定でパブリッシュすべきではありません。なぜならばモデムのスピードは結果として、品質を著しく落としかねないからです。プレゼンターは DSL 設定でパブリッシュしたいかも知れ

ませんが、モデムユーザの見るインターフェイスでは、動きの品質がはなはだ損なわれることとなります。

SetBandwidth コンポーネントは、SimpleConnect コンポーネントと併用すれば、スクリプティングに必要ななくなります。

### SetBandwidth コンポーネントを使用する

シンプルでパワフルなこのコンポーネントは、可能なアップロード帯域幅でパブリッシュされた、マイクやカメラの品質に自動的に合わせます。このコンポーネントは、ライブ音声や映像をパブリッシュするあらゆるアプリケーションに使用できます。

以下はこのコンポーネントの2つの使用例です。

映像会議：FCVideoConference は、FCSetBandwidth と併用できるように作成されている AVPresence コンポーネントを使用するので、マイクやカメラの品質は自動的に調整されます。

プレゼンテーション：プレゼンテーション・アプリケーションのコンセプトに、1人のプレゼンターに多くの視聴者というのがあります。そういったアプリケーションで、AVPresence コンポーネントは、ビデオ・プレゼンテーションを提供します。こうしたアプリケーションで SetBandwidth コンポーネントと共用することは効果的です。

### アプリケーションでコンポーネントを使用する

1 Flash Communication Server が起動していることを確認する

2 Flash Communication Server MX のアプリケーション・ディレクトリ(通常は`flashcom/applications`)にアプリケーション・ディレクトリを作成し、`setBand_test` の名前をつける。

3 アプリケーション・ディレクトリ内に `main.asc` の名前で新しいファイルを作成し、以下のコードを記述する。

```
load("components.asc");
```

**Note:**他のアプリケーション・ディレクトリからこのアプリケーション・ディレクトリへ、`main.asc` ファイルをコピーして使用することもできます。

4 Flash MX で、ステージ上に SetBandwidth コンポーネントをドラッグする。

5 プロパティ・インスペクタで、インスタンス名 `setBand_mc` をつける。他のパラメータについては、61ページの FCSetBandwidth オブジェクト・プロパティ・サマリーをご覧ください。

6 ステージ上に SimpleConnect コンポーネントをドラッグする。

7 SimpleCpnnection 用のプロパティ・インスペクタで以下のパラメータを入力する。

Application Directory テキスト・ボックスで、ステップ2で作成したアプリケーションの URI、例えば、`rtmp:/setBand_test` を入力する。

Communication Components テキスト・ボックスをダブル・クリックする。現れる値ダイアログ・ボックスで、+ 印をクリックし、SetBandwidth コンポーネントのインスタンス名 `setBand_mc` を入力し、OK をクリックする。

この値を与えることで、SimpleConnect コンポーネントは SetBandwidth コンポーネントの接続を制御できる。SimpleConnect コンポーネントがサーバーへの接続に成功したら、SetBandwidth コンポーネントは自動的にサーバーに接続する。

8 アプリケーション・ディレクトリ内に `SetBand_tst` 名で保存、パブリッシュする。

9 アプリケーション・ディレクトリ内にある SWF ファイルを開くか、または Flash MX オーサリング環境では、制御 >ムービープレビューを選択しログインする。

下記のように、違った帯域幅を設定できる。



### SetBandwidth コンポーネントのリスキニング

このコンポーネントのアセットは、ライブラリ・パネルの Core Assets-Developer Only\SetBandwidth Assets ディレクトリにあります。

### 関係するコンポーネント

“AVPresence” コンポーネント 17 P、“SimpleConnect コンポーネント” 64 p

### FCSetBandwidth オブジェクト

FCSetBandwidth オブジェクトは、グローバルな Quality Manager(gFlashCom.quality)を設定し、管理メソッドを供給することで、カメラやマイクの品質を管理します。管理を必要とするマイクとカメラ・オブジェクトは、manage、unmanage メソッドをコールできます。

以下のサンプルは、マイクとカメラ 2 セットが登録されることを示しています。manage コールの最後のパラメータは、帯域幅の共有量を決定します。サンプルでは、mic1/cam1 が 33 パーセント、mic2/cam2 が 66 パーセント得ます。

このパラメータは、一定の共有にすることで特定しないことも可能です。

```
GflashCom.quality.manage(mic1,cam1,100);
```

```
GflashCom.quality.manage(mic2,cam2,200);
```

マイクとカメラの管理を止めるには、以下のように記述します。Quality Manager が自動的に品質を調整します。

```
GflashCom.quality.unmanage(mic1,cam1);
```

このステップは、AVPresence コンポーネントによらないアプリケーションで、SetBandwidth コンポーネントを使用するときのみ必要になります。

## FCSetBandwidth オブジェクト・メソッド・サマリー

### メソッド

### 説明

FCSetBandwidth.connect  
現在選択されている帯域幅にしたがって、特定の NetConnection オブジェクトの帯域幅制限を設定する。

FCSetBandwidth.close  
クリーン・アップする

## FCSetBandwidth オブジェクト・プロパティ・サマリー

FCSetBandwidth.modemUp  
Modem オプションを選択したとき、アップロードの帯域幅を決定する。デフォルトは 3 3 Kbps アップ。

FCSetBandwidth.modemDown  
Modem オプションを選択したとき、ダウンロードの帯域幅を決定する。デフォルトは 3 3 Kbps ダウン。

FCSetBandwidth.dslUp  
DSL オプションを選択したとき、アップロードの帯域幅を決定する。デフォルトは 1 2 8 Kbps アップ。

FCSetBandwidth.dslDown  
DSL オプションを選択したとき、ダウンロードの帯域幅を決定する。デフォルトは 2 5 6 Kbps ダウン。

FCSetBandwidth.lanUp  
LAN オプションを選択したとき、アップロードの帯域幅を決定する。デフォルトは 1 0 0 0 Kbps アップ。

FCSetBandwidth.lanDown  
LAN オプションを選択したとき、ダウンロードの帯域幅を決定する。デフォルトは 1 0 0 0 Kbps ダウン。

## FCSetBandwidth.modemUp

### 利用

Flash Player 6

Flash Communication Server MX

### 使い方

*setBand\_mc.modemUp = maxSpeed*

### 説明 ;

プロパティ。Modem オプションを選択したとき、アップロードの帯域幅を決定する。これは設定、取得できるプロパティ。変数 *maxSpeed* は Kbps 単位のアップロードのスピード。デフォルトは 3 3 Kbps。

## FCSetBandwidth.modemDown

### 利用

Flash Player 6

Flash Communication Server MX

### 使い方

*setBand\_mc.modemDown = maxSpeed*

**説明；**

プロパティ。Modem オプションを選択したとき、ダウンロードの帯域幅を決定する。これは設定、取得できるプロパティ。変数 *maxSpeed* は Kpbs 単位のダウンロードのスピード。デフォルトは 3 3 Kbps。

### **FCSetBandwidth.dslUp**

#### **利用**

Flash Player 6

Flash Communication Server MX

#### **使い方**

*setBand\_mc.dslUp = maxSpeed*

#### **説明；**

プロパティ。DSL オプションを選択したとき、アップロードの帯域幅を決定する。これは設定、取得できるプロパティ。変数 *maxSpeed* は Kpbs 単位のアップロードのスピード。デフォルトは 1 2 8 Kbps。

### **FCSetBandwidth.dslDown**

#### **利用**

Flash Player 6

Flash Communication Server MX

#### **使い方**

*setBand\_mc.dslDown = maxSpeed*

#### **説明；**

プロパティ。DSL オプションを選択したとき、ダウンロードの帯域幅を決定する。これは設定、取得できるプロパティ。変数 *maxSpeed* は Kpbs 単位のダウンロードのスピード。デフォルトは 2 5 6 Kbps。

### **FCSetBandwidth.lanUp**

#### **利用**

Flash Player 6

Flash Communication Server MX

#### **使い方**

*setBand\_mc.lanUp = maxSpeed*

#### **説明；**

プロパティ。LAN オプションを選択したとき、アップロードの帯域幅を決定する。これは設定、取得できるプロパティ。変数 *maxSpeed* は Kpbs 単位のアップロードのスピード。デフォルトは 1 0 0 0 Kbps。

### **FCSetBandwidth.lanDown**

#### **利用**

Flash Player 6

Flash Communication Server MX

#### **使い方**

*setBand\_mc.lanDown = maxSpeed*

#### **説明；**

プロパティ。LAN オプションを選択したとき、ダウンロードの帯域幅を決定する。これは設定、取得できるプロパティ。変数 *maxSpeed* は Kpbs 単位のダウンロードのスピード。デフォルトは 1 0 0 0 Kbps。

## FCSetBandwidth.connect

### 利用

Flash Player 6

Flash Communication Server MX

### 使い方

`setBand_mc.connect(nc)`

### パラメータ

`nc` アクティブな NetConnection オブジェクト。

### 戻り値

なし

### 説明 ;

メソッド。現在選択されている帯域幅にしたがって、`nc` パラメータの帯域幅制限を設定する。

## FCSetBandwidth.close

### 利用

Flash Player 6

Flash Communication Server MX

### 使い方

`setBand_mc.close()`

### パラメータ

なし

### 戻り値

なし

### 説明 ;

このクライアント上でこのコンポーネントが使用するアセットを解放し、クリーン・アップする。

## SetBandwidth コンポーネントとコンポーネントを合わせて使用する

カメラやマイクの品質が変わったとき、サーバーにそれを知らせるイベントをコンポーネントに書き込むことができます。これらのイベントに常に注視するには、下記のステップを踏みます。

1 以下のように `ocChange` メソッドを与える。

```
listener.onChange = function(microphone, camera)
```

```
{
```

```
    //セッティングの再調整など
```

```
}
```

2 以下のように Quality Manager にリスナーを登録する。

```
GFlashCom.quality.addListener(listener);
```

これで、SetBandwidth コンポーネントがマイクやカメラの設定を調整すると常に `onChange` メソッドがコールされます。

## SimpleConnect コンポーネント

このコンポーネントは、Macromedia Flash Communication Server に接続したユーザや、サーバー・アプリケーションに接続したある特定のコンポーネントを制御します。またユーザ名を表示したり、ユーザが名前を変えるユーザ・インターフェイスも持っています。

SimpleConnect コンポーネントは、ユーザが初めてログインしたときに作成されるローカルの共有オブジェクトからデータを取得することで、ユーザを記録します。ローカルの共有オブジェクトを使用することで、接続と接続とを関連させてつなぎ合わせたり、ユーザがエグジットすることなくアプリケーションからログアウトすることを避けることができます。

このコンポーネントを使用するには、ムービーにドラッグし、プロパティ・インスペクタに表示されるコンポーネント・パラメータを編集します。rtml:/test といったアプリケーション・ディレクトリを準備する必要があり、自動的に接続される他の全てのコミュニケーション・コンポーネントのリストを入力します。

このコンポーネントは、他の全てのコミュニケーション・コンポーネントと併用できるよう設計されており、NetConnection オブジェクトの管理を簡単なものになっています。ActionScript を少し書き加えた、もしくは全く書き加えないコミュニケーション・コンポーネントを使用して、新しいコミュニケーション・アプリケーションを構築することができます。

### SimpleConnect コンポーネントを使用する

以下はこのコンポーネントの使用例です。

映像会議：このコンポーネントと VideoConference コンポーネントを併用して、多くの機能を持ったアプリケーションを簡単に構築できます。

カスタム・コミュニケーション・コンポーネント：connect、close、setUsername などのメソッドを使用し、コミュニケーション・コンポーネントを作成します。SimpleConnect コンポーネントを使用して、ネットワーク接続を管理できます。

### プロパティ・インスペクタ・パラメータ

名前	説明
Application Directory	ストリング；サーバーに接続する際使用するアプリケーション・ディレクトリの場所を特定するコンポーネント・パラメータ。
Communication Components	Array；FCSimpleConnect が接続する他のコミュニケーション・コンポーネントのリストを含むコンポーネント・パラメータ。

### SetBandwidth コンポーネントのリスキニング

このコンポーネントのアセットは、ライブラリ・パネルの Core Assets-Developer Only¥Assets ディレクトリにあります。

## **UserColor コンポーネント**

このコンポーネントによって、ユーザは Flash Communication Server アプリケーション内にあるコンポーネントの選択した色を変更できます。ユーザがメニューから色を選択すると、例えば Chat コンポーネントのように、色を表示する他のコンポーネントが、自動的に色を変更します。

UserColor コンポーネントは、ユーザのコンピュータ上にあるローカルの共有オブジェクトに選択した色を保管します。もし、初めての訪問などで、ローカルの共有オブジェクトが見つからない場合は、リストからランダムに色が選ばれ表示されます。

UserColor コンポーネントは、SimpleConnect コンポーネントと併用すれば、スクリプティングの必要はなくなります。

## **UserColor コンポーネントを使用する**

このシンプルで視覚的にパワフルなコンポーネントは、集中管理的ツールで、他のコンポーネントに色を加えます。

以下はこのコンポーネントの2つの使用例を示したものです。

読みやすいチャット：UserColor コンポーネントなしに Chat コンポーネントを使用すると、全ユーザの文字色は黒になります。UserColor コンポーネントを加えることで、チャットの文字は読みやすくなり、ユーザにカスタマイズ  
の機会を与えることができます。

ユーザに仮想アイデンティティが必要な場合はいつでも、ユーザが特定した色はそのユーザ名のように働きます。

UserColor コンポーネントにより、多くのコーディングをする苦勞なしに、アイデンティティが拡張された感覚を味わわせることができます。

SimpleConnect、UserColor、Corsor、Chat コンポーネントを合わせて使用してみてください。ユーザは自分の名前や色、場所を表示する文字とカーソルをステージ上に見ることができます。ユーザの色はより豊かなユーザ体験をもたらします。

アプリケーションでこのコンポーネントを使用するには、20ページの“Chat コンポーネント”をご覧ください。

## **コンポーネントのリスキニング**

このコンポーネントの色を変えたい場合は、ライブラリ・パネルでダブル・クリックし、ColorCombo ボックスをクリックして、プロパティ・インスペクタを開き、Data array 内で値を変更します。カラー・パレットは自動的に更新されます。Label array と Data array の要素数は同じにしておくよう注意してください。

## **関係するコンポーネント**

“Chat コンポーネント”20ページ。

## FCUserColor オブジェクト

FCUserColor API は、サーバーに接続したりクリーンアップしたりするオブジェクトに、クライアント・サイドとサーバー・サイドでの機能を提供します。

### FCUserColor オブジェクト・メソッド・サマリー

メソッド	説明
FCUserColor.connect	保存された FCUserColor のローカル共有オブジェクトを探し、ユーザの最後の色を取得します。
FCUserColor.close	クリーン・アップします。

### FCUserColor.connect

#### 利用

Flash Player 6

Flash Communication Server MX

#### 使い方

`userColor_mc.connect(nc)`

#### パラメータ

`nc` アクティブな NetConnection オブジェクト

#### 戻り値

なし

#### 説明；

メソッド；保存された FCUserColor のローカル共有オブジェクトを探し、ユーザの最後の色を取得する。ローカル共有

オブジェクトが存在する場合、色は選択される。そうでない場合は、そのユーザの色がランダムに選択され、ローカル共有オブジェクト内に保存される。

### FCUserColor.close

#### 利用

Flash Player 6

Flash Communication Server MX

#### 使い方

`userColor_mc.close()`

#### パラメータ

なし

#### 戻り値

なし

#### 説明；

メソッド；このクライアント上でこのコンポーネントが使用したアセットを解放し、クリーン・アップする。

## UserColor コンポーネントとコンポーネントを併用する

コンポーネントを書き、UserColor コンポーネントと併用させたいときは、以下のステップで見ている人の色を変更します。

1 下記のように、ocColorChange メソッドを記述します。

```
MyCompnentClass.prototype.onColorChange = function()
{
    this.setColor(gFlashCom.userprefs.color);
}
```

gFlashCom.userprefs グローバル・オブジェクトは、新しい色を保持した FCUserColor コンポーネントによって更新

されます。

コンポーネントの this.setColor メソッドは、色の変更を処理します。onColorChange と setColor メソッドを論理的に

分けるのはよいことです。そうすることでonColorChange を呼ぶUserColor コンポーネントを使用しなくても、setColor メソッドを使用できます。

2 色も含む userprefs オブジェクトが変更されたことを知らせるため、リスナーのリストにコンポーネント・クラスを

加えます。コンポーネントの#endinitclip ステートメントの後に、以下を入力します。

```
GFlashCom.userprefs.addListener(this);
```

これらのステップで、色の変更に FCUserColor を使用すればいつでも、onColorChange メソッドがコールされます。

## VideoConference コンポーネント

このコンポーネントは、AVPresence コンポーネントを用いて、多数のユーザが互いに音声や映像を使用することを可能にします。VideoConference コンポーネントは SimpleConnection コンポーネントと併用すれば、スクリプティングの必要はなくなります。

### VideoConference コンポーネントを使用する

アプリケーション内にこの多機能なコミュニケーション・コンポーネントを使用することで、ビデオ会議スペースを作成

することができます。

VideoConference、SimpleConnect、SetBandwidth、ConnectionLight の各コンポーネントをステージ上にドラッグして

ビデオ会議アプリケーションを作成します。

### アプリケーションにコンポーネントを使用する

1 Flash Communication Server が起動していることを確認する

2 Flash Communication Server MX のアプリケーション・ディレクトリ(通常は%flashcom%applications)にアプリケーション・ディレクトリを作成し、video\_test の名前をつける。

3 Flash MX で、ステージ上に videoConference コンポーネントをドラッグする。

4 プロパティ・インスペクタで、インスタンス名 `videoconf_mc` をつける。他のパラメータについては、68ページの `videoConference` コンポーネント・プロパティ・インスペクタをご覧ください。

5 ステージ上に `SimpleConnect` コンポーネントをドラッグする。

6 `SimpleCpnnection` 用のプロパティ・インスペクタで以下のパラメータを入力する。

Application Directory テキスト・ボックスで、ステップ 2 で作成したアプリケーションの URI、例えば、rtmp://video\_test を入力する。

Communication Components テキスト・ボックスをダブル・クリックする。現れる値ダイアログ・ボックスで、+ 印をクリックし、videoConference コンポーネントのインスタンス名 videoconf\_mc を入力し、OK をクリックする。この値を与えることで、SimpleConnect コンポーネントは videoConference コンポーネントの接続を制御できる。SimpleConnect コンポーネントがサーバーへの接続に成功したら、videoConference コンポーネントは自動的にサーバーに接続する

7 アプリケーション・ディレクトリ内に video\_test 名で保存、パブリッシュする。

8 このアプリケーション・ディレクトリ内に main.asc の名前で作成し、以下のコードを記述する。

```
load("components.asc");
```

**Note:**他のアプリケーション・ディレクトリからこのアプリケーション・ディレクトリへ、main.asc ファイルをコピーして使用することもできます。

9 アプリケーション・ディレクトリ内にある SWF ファイルを開くか、または Flash MX オーサリング環境では、制御>ムービープレビューを選択しログインし、Auto チェック・ボックスをクリックする。

以前に Flash Player のプライバシー設定を行っていない場合は、カメラ、マイクへのアクセスを許可するかどうか聞かれる。

オーディオ・デバイスが作動している場合、喋ると音声レベルが上下する。ビデオ・デバイスが作動していると、ビデオ画面にその出力結果を見ることができる。マウス・ポインタをビデオ画面上で動かすと、マイクとカメラの UI を見ることができる。これは音声や映像データへの変換を止めるボタンとしても働く。

dragSharing プロパティが true に設定されていると、ビデオ画面を動かすことができる。

#### VideoConference コンポーネント・プロパティ・インスペクタ

名前	変数	説明
Show Boundary	showBoundary	Boolean 値；デフォルト値は false。ビデオ会議エリアの境界線を示すかどうかを決定する。
Show Background	showBackground	Boolean 値；デフォルト値は false。ビデオ会議に背景を設定するか、透明にするかを決定する。
Clip Mask	clipMask	Boolean 値；デフォルト値は false。ビデオ・ウィンドウがビデオ会議エリアに切り取られるかどうかを決定する。
Drag Sharing	dragSharing	Boolean 値；デフォルト値は true。ビデオ・ウィンドウの位置を共有するかどうかを決定する。共有することによりビデオウィンドウは離れたユーザがドラッグすると同じように移動する。どのようにドラッグしても他のユーザに見える。

#### VideoConference コンポーネントのリスキニング

このコンポーネントの資産は、ライブラリ・パネルの Core Assets-Developer Only¥VideoConference Assets ディレクトリにあります。

## 関係するコンポーネント

“SimpleConnect”コンポーネント 64 ページ、“AVPresence”コンポーネント 17 ページ、“SetBandwidth”コンポーネント 58 ページ。

## FCVideoConference オブジェクト

FCVideoConference API は、オブジェクトがサーバーに接続したりクリーンアップしたりする、クライアント・サイドやサーバー・サイドの機能を提供します。

### FCVideoConference オブジェクト・メソッド・サマリー

#### メソッド

#### 説明

FCVideoConference.connect

ビデオ会議を開始する。

FCVideoConference.setUsername

*newName* パラメータが null または undefined でない場合、メソッドがユーザのアイデンティティを変更する。そうでなければカメラとマイク

ク

のパブリッシュを止める。

FCVideoConference.close

ビデオ会議への参加を止める。

#### FCVideoConference.connect

#### 利用

Flash Player 6

Flash Communication Server MX

#### 使い方

*myVideoConf\_mc*.connect(nc)

#### パラメータ

*nc* アクティブな NetConnection オブジェクト

#### 戻り値

なし

**説明** ; メソッド ; ビデオ会議を開始する。ユーザ名が設定されていれば、このメソッドはカメラとマイクをパブリッシュ

する。

#### FCVideoConference.setUsername

#### 利用

Flash Player 6

Flash Communication Server MX

#### 使い方

*myVideoConf\_mc*.setUsername(*newName*)

#### パラメータ

*newName* スtring

#### 戻り値

なし

#### 説明；

メソッド；*newName* が null または undefind でなければ、このメソッドはユーザ・アイデンティティを変更する。

*NewName* が null または undefined の場合は、カメラとマイクをパブリッシュするストリームは止まる。

#### FCVideoConference.close

#### 利用

Flash Player 6

Flash Communication Server MX

#### 使い方

```
myVideoConf_mc.close()
```

#### パラメータ

なし

#### 戻り値

なし

#### 説明；

メソッド；このクライアント上でこのコンポーネントが使用しているアセットを解放し、ビデオ会議内でのこのクライアントの参加を終わらせることで、クリーン・アップする。

### VideoPlayback コンポーネント

VideoPlayback コンポーネントによって、バッファされた音声映像ストリームの再生が可能になります。ムービーを一時停止、再生させたり、再生ヘッドをドラッグして好きな場所に飛んだり、また音声レベル設定もできます。

VideoPlayback コンポーネントは、SimpleConnect コンポーネントと併用すればスクリプティングの必要はなくなり、VideoPlayback コンポーネントは、streamName 変数内に特定したストリームをロードし、playStream メソッドによって再生できます。

#### VideoPlayback コンポーネントを使用する

このコンポーネントは、ストリームを記録したビデオを再生するツールとして使用できます。以下はこのコンポーネントの2つの使用例です。

ムービー再生：これまでに記録したコンテンツを再生します。

ビデオ・ノート：先に訪れたユーザが残したメッセージを再生します。

アプリケーション内でこのコンポーネントを使用するには、72ページの”VideoRecord コンポーネントを使う”をご覧ください。

#### VideoPlayback コンポーネント・プロパティ・インスペクタ

名前	変数	説明
Default Stream Name	streamName	ストリング；デフォルト値は video。ストリーム名を特定してロードし再生する。
Buffer Time	bufferTime	数値；デフォルト値は 2。再生時にバッファするビデオ・

データの毎秒の長さを特定する。

### VideoConference コンポーネントのリスキニング

このコンポーネントのアセットは、ライブラリ・パネルの Core Assets-Developer Only¥VideoPlayback Assets ディレクトリにあります。

### 関係するコンポーネント

“SimpleConnect”コンポーネント 64 ページ、 “VideoRecord”コンポーネント 72 ページ

### FCVideoPlayback オブジェクト

FCVideoPlayback オブジェクトは、サーバーに接続したりクリーンアップしたりするオブジェクトに、クライアント・サイドやサーバー・サイドでの機能を提供します。

### FCVideoPlayback オブジェクトのメソッド・サマリー

メソッド	説明
FCVideoPlayback.connect	クライアント・サイドとサーバ・サイドで VideoPlayback コンポーネントが必要とするアセットをセットアップする。
FCVideoPlayback.close	クリーン・アップし、通常のマウス・ポインタに戻す(Mouse.show)

### FCVideoPlayback.connect

#### 利用

Flash Player 6

Flash Communication Server MX

#### 使い方

`vidPlayback_mc.connect(nc)`

#### パラメータ

`nc` アクティブな NetConenction オブジェクト

#### 戻り値

なし

#### 説明

メソッド ; クライアント・サイドとサーバ・サイドで VideoPlayback コンポーネントが必要とするアセットをセットアップする。

### FCVideoPlayback.close

#### 利用

Flash Player 6

Flash Communication Server MX

#### 使い方

`vidPlayback_mc.close()`

#### パラメータ

なし

## 戻り値

なし

## 説明

メソッド; このクライアント上でこのコンポーネントが使用していたアセットを解放することでコンポーネントを閉じ、クリーン・アップする。

## VideoRecord コンポーネント

VideoRecord コンポーネントは、バッファされたストリームをサーバーに記録します。カメラとマイクの出力結果が記録されます。

### VideoRecord コンポーネントを使用する

このコンポーネントは、サーバーにストリームを保存するビデオ・レコーダーとして使用できます。以下はこのコンポーネントを使用した2つの例です。

ムービー・レコーダー: このコンポーネントを使ってサーバーにビデオをキャプチャーし、後で再生する。

ビデオ・ノート: 他のユーザがメッセージを記録し、後で見る。

### アプリケーションで VideoRecord コンポーネントを使用する

1 Flash Communication Server が起動していることを確認する

2 Flash Communication Server MX のアプリケーション・ディレクトリ(通常は`flashcom/applications`)にアプリケーション・ディレクトリを作成し、`recplay_test`の名前をつける。

3 ステージ上に `videoRecord` コンポーネントをドラッグし、インスタンス名 `record_mc` をつける。他のパラメータについては、74ページの `videoRecord` コンポーネント・プロパティ・インスペクタをご覧ください。

4 ステージ上に `videoPlayback` コンポーネントをドラッグし、インスタンス名 `playback_mc` をつける。他のパラメータについては、70ページの `videoPlayback` コンポーネント・プロパティ・インスペクタをご覧ください。

5 ステージ上に `SimpleConnect` コンポーネントをドラッグする。

6 `SimpleCpnnection` 用のプロパティ・インスペクタで以下のパラメータを入力する。

`Application Directory` テキスト・ボックスで、ステップ2で作成したアプリケーションのURI、例えば、`rtmp:/recplay_test`を入力する。

`Communication Components` テキスト・ボックスをダブル・クリックする。現れる値ダイアログ・ボックスで、+印をクリックし、再生と記録のコンポーネントのインスタンス名にそれぞれ `playback_mc`、`record_mc` と入力し、OKをクリックする。

これらの値を与えることで、`SimpleConnect` コンポーネントは `videoPlayback` コンポーネントと `videoRecord` コンポーネントの接続を制御できる。`SimpleConnect` コンポーネントがサーバーへの接続に成功したら、他のコンポーネントは自動的にサーバーに接続する

7 アプリケーション・ディレクトリ内に `recplay_test` 名で保存、パブリッシュする。

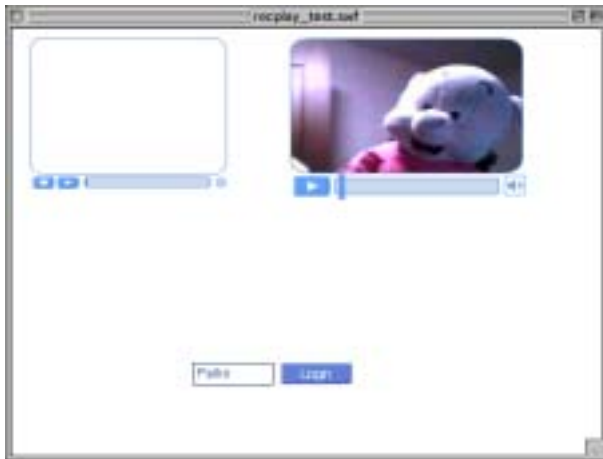
8 このアプリケーション・ディレクトリ内に `main.asc` の名前で新しいファイルを作成し、以下のコードを記述する。

```
load("components.asc");
```

**Note:**他のアプリケーション・ディレクトリからこのアプリケーション・ディレクトリへ、`main.asc` ファイルをコピーして使用することもできます。

9 アプリケーション・ディレクトリ内にある SWF ファイルを開くか、または Flash MX オーサリング環境では、制御>ムービープレビューを選択しログインする。

VideoRecord コンポーネントの Record ボタンをクリックし、コンポーネントの Default Stream Name プロパティで定義されたデフォルトのストリーム名 video を記録する。カメラとマイクの出力は、下記のように VideoRecord コンポーネント上の右矢印の Record ボタンがクリックされたら記録される。



再生する video ストリームは、playback コンポーネントの 1 番目のパラメータで定義される。記録を止めた後、playback コンポーネント上の右矢印の Playback ボタンをクリックすると、下記のように記録されたビデオ・ストリームが再生される。



## VideoRecord コンポーネント・プロパティ・インスペクタ

名前	変数	説明
Default Stream Name	streamName	ストリング；デフォルト値は video。最初にロードし再生するストリーム名を特定する。
Default Stettings	[setLow	ストリング；デフォルト値は setHigh。コンポーネントが初期化されるとき、使用するカメラの設定を指定する。SetLow、setMed、setHigh の 3 種類の設定がある。
Buffer Time	bufferTime	数値；デフォルト値は 10。録画でバッファするビデオ・データの毎秒の長さを指定する。
Low Quality Setting	setLow	オブジェクト；カメラ設定で使用する以下の値を持つ。 幅、水平値 160 ピクセル 高さ、垂直値 120 ピクセル fps、毎秒 10 ビデオ・フレーム 帯域幅、使用する最大の帯域幅、0(0は無制限を示す) 品質、ビデオの圧縮品質、50(0は無制限を示す) キー、キーフレームの間隔、10キーフレーム レート、キロヘルツ単位のオーディオ品質、11KHz
Medium Quality Setting	setMed	オブジェクト；カメラ設定で使用する以下の値を持つ。 幅、水平値 240 ピクセル 高さ、垂直値 180 ピクセル fps、毎秒 12 ビデオ・フレーム 帯域幅、使用する最大の帯域幅、0(0は無制限を示す) 品質、ビデオの圧縮品質、80(0は無制限を示す) キー、キーフレームの間隔、12キーフレーム レート、キロヘルツ単位のオーディオ品質、22KHz
High Quality Setting	setHigh	オブジェクト；カメラ設定で使用する以下の値を持つ。 幅、水平値 320 ピクセル 高さ、垂直値 240 ピクセル fps、毎秒 15 ビデオ・フレーム 帯域幅、使用する最大の帯域幅、0(0は無制限を示す) 品質、ビデオの圧縮品質、90(0は無制限を示す) キー、キーフレームの間隔、5キーフレーム レート、キロヘルツ単位のオーディオ品質、44KHz

## VideoRecord コンポーネントのリスキニング

このコンポーネントのアセットは、ライブラリ・パネルの Core Assets-Developer Only¥VideoRecord Assets ディレクトリにあります。

## 関係するコンポーネント

“SimpleConnect”コンポーネント 64 ページ、“VideoPlayback”コンポーネント 70 ページ

## FCVideoRecord オブジェクト

VideoRecord コンポーネントは、SimpleConnect コンポーネントと併用すればスクリプティングの必要はなくなります。

streamName プロパティを設定しない場合は、コンポーネントはデフォルトのストリーム名で録画します。

## FCVideoRecord オブジェクトのメソッド・サマリー

メソッド	説明
FCVideoRecord.connect	クライアント・サイドとサーバー・サイドで VideoRecord コンポーネントが必要とするアセットをセットアップする。
FCVideoRecord.close	クリーン・アップし、通常のマウス・ポインタに戻す(Mouse.show)

## FCVideoRecord.connect

### 利用

Flash Player 6

Flash Communication Server MX

### 使い方

`vidRecord_mc.connect(nc)`

### パラメータ

`nc` アクティブな NetConnection オブジェクト

### 戻り値

なし

### 説明

メソッド；クライアント・サイドとサーバー・サイドで VideoRecord コンポーネントが必要とするアセットをセットアップする。

## FCVideoRecord.close

### 利用

Flash Player 6

Flash Communication Server MX

### 使い方

`vidRecord_mc.close()`

### パラメータ

なし

### 戻り値

なし

### 説明

メソッド；このクライアント上のこのコンポーネントが使用していたアセットを解放することで、コンポーネントを閉じ、クリーン・アップする。

## Whiteboard コンポーネント

このコンポーネントでは、リアルタイムの共有環境で、テキストやボックス、ラインの編集ができます。シェイプを作るにはツールを選択し、ホワイトボード領域上でクリックします。またシェイプを新しい場所にドラッグしたりテキストを編集したりできます。デリート・キーを押すと、選択されたアクティブなアイテムが削除されます。

Whiteboard コンポーネントは、SimpleConnect コンポーネントと併用するとスクリプトを書く必要がなくなります。

### Whiteboard コンポーネントを使用する

このコンポーネントを使って、アイデア開発などにおける共有環境を作ることができます。以下はこのコンポーネントの3つの使用例です。

共有ホワイトボード：他のユーザにアイデアを示す

組織図：会社のおおまかな組織構成。

ワークフロー：プロジェクトの進捗状況を示す。次々要素を追加できるので、ホワイトボード上でそれを確認でき、全てのユーザが同時に見ることができる。

### アプリケーションでコンポーネントを使用する

1 Flash Communication Server が起動していることを確認する

2 Flash Communication Server MX のアプリケーション・ディレクトリ(通常は`flashcomapplications`)にアプリケーション・ディレクトリを作成し、`whiteboard_test` の名前をつける。

3 ステージ上に Whiteboard コンポーネントをドラッグする。

4 プロパティ・インスペクタでインスタンス名 `whiteboard_mc`

5 ステージ上に SimpleConnect コンポーネントをドラッグする。

6 SimpleCpnnection 用のプロパティ・インスペクタで以下のパラメータを入力する。

Application Directory テキスト・ボックスで、ステップ 2 で作成したアプリケーションの URI、例えば、`rtmp:/whiteboard_test` を入力する。

Communication Components テキスト・ボックスをダブル・クリックする。現れる値ダイアログ・ボックスで、+ 印をクリックし、`whiteboard_m` と入力、OK をクリックする。

7 アプリケーション・ディレクトリ内に `white_test` 名で保存、パブリッシュする。

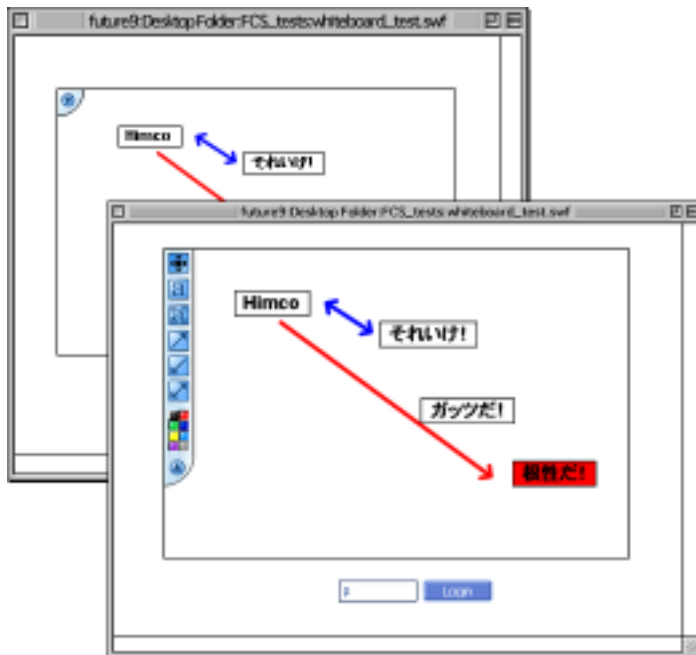
8 このアプリケーション・ディレクトリ内に `main.asc` の名前で新しいファイルを作成し、以下のコードを記述する。

```
load("components.asc");
```

**Note:**他のアプリケーション・ディレクトリからこのアプリケーション・ディレクトリへ、`main.asc` ファイルをコピーして使用することもできます。

9 アプリケーション・ディレクトリ内にある SWF ファイルを開くか、または Flash MX オーサリング環境では、制御>ムービープレビューを選択しログインする。

下図のように、左にいくつかのツールが並ぶホワイトボードが現れる。さらにアプリケーションのインスタンスを開けば、それぞれのインスタンスは他のインスタンスのシェイプやテキストを表示する。オブジェクトを選択、移動させたり、文字を打ったり、テキスト・ボックスや矢印を作成したり、色を選択したりできる。



#### Whiteboard コンポーネントのリスキニング

このコンポーネントの資産は、ライブラリ・パネルの Core Assets-Developer Only\Whiteboard Assets ディレクトリにあります。

#### 関係するコンポーネント

“SimpleConnect”コンポーネント 64 ページ

#### FCWhiteboard オブジェクト

FCWhiteboard API は、サーバーに接続したりクリーンアップするオブジェクトに、クライアント・サイドとサーバー・サイドの機能を提供します。

## FCWhiteboard オブジェクト・メソッド・サマリー

### メソッド

### 説明

FCWhiteboard.connect	クライアント・サイドとサーバーサイドでコンポーネントが必要とする全てのアセットをセット・アップする。
FCWhiteboard.close	このクライアント上でこのコンポーネントが使用していたアセットを解放し、クリーンアップする。

## FCWhiteboard.connect

### 利用

- Flash Player 6
- Flash Communication Server MX

### 使い方

`wb_mc.connect(nc)`

### パラメータ

`nc` アクティブな NetConnection オブジェクト

### 戻り値

なし

### 説明

メソッド；クライアント・サイドとサーバー・サイドでコンポーネントが必要とするアセットをセットアップする。

## FCWhiteboard.close

### 利用

- Flash Player 6
- Flash Communication Server MX

### 使い方

`wb_mc.close()`

### パラメータ

なし

### 戻り値

なし

### 説明

メソッド；このクライアント上のこのコンポーネントが使用していたアセットを解放することで、クリーン・アップする。