
i_494


a_160


n_704


g_832


f_124


e_415


m_452

BAD

Pi!Po!Pa!


h_894

Creator
さうなまん（モイモラ）

同じ場所にいる人たちと1つの画面を共有する

Flash Liteによるイベントツール。
ポイントは、携帯電話からの操作でいかにリアルタイムなレスポンスを実現するか。


e_642


m_808

Title
Pi!Po!Pa!

Sample URL
ステージ画面（PC用）：http://vgzh.dtdns.net/z/monitor.html
操作画面（携帯電話用）：http://vgzh.dtdns.net/z/

Archive
pipopa.zip

File
06

スクリプト
ActionScript 1.0

対応プレーヤー
Flash Player 8以上

制作アプリケーション
Flash 8

OOD

発案～デザイン
携帯電話による参加型コンテンツ

「Pi!Po!Pa!」はセミナーやカンファレンスなど多くの来場者が集まる場において活躍するプレゼンテーションツールです。

参加者はインターネット端末としてもっとも身近なツールである携帯電話を使ってステージ画面上に表示される自分のアイコンを操作します。進行するプレゼン内容に従って自分のアイコンを○、もしくは×のエリアに移動し、自身の意見を伝えることができます。おもしろかったら○であり、つまらないと思ったら×という意味付けです。これにより、スピーカーはリアルタイムにプレゼンテーションの反応を知ることができます。誰でも参加でき、同じ場所にいる人たちと 1 つの画面を共有する体験が「Pi!Po!Pa!」の楽しさです。

この章では Flash Lite1.1 による携帯用 Flash の作成から XML-Socket によるサーバサイド Flash の実装までを解説します。

○ Q-JAM

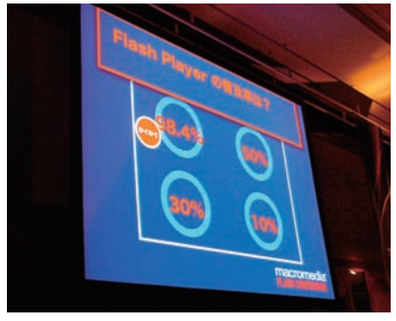
2003 年 6 月 20 日に東京都内で「Macromedia Flash Conference」というイベントが開催されました。その基調講演で紹介された「Q-JAM」というソリューションは、会場にいた私に大きな衝撃を与えてくれました。

「Q-JAM」は、スクリーンに表示されたクイズ問題に対してブラウザ付き携帯電話を利用して答えると、正解や順位などの結果がリアルタイムで集計され、スクリーンに表示することができるという仕組みで、より高機能なイベントツールとしてすでに完成されたものです。

URL台 **Q-JAM**
http://www.indexweb.co.jp/

会場には事前に Q-JAM アプリ搭載の携帯端末をわたされた参加者が何名かおり、彼等がスクリーン上に映し出されるクイズに回答していく形で講演は進みました。クイズの出題から始まって、参加者が回答するまでのスクリーン画面上の展開は、リアルタイムにユーザーのレスポンスを反映している事実を証明するに十分なインタラクションでした。また、正誤の集計や成績結果などもリッチなインターフェース上でアニメーションで表示され、端末を持たず、見るだけだった自分にも十分に満足のいく内容でした。もし端末をわたされ、参加していたら一体どれだけ楽しい時間になっていたことでしょうか。

「Macromedia Flash Conference」での Q-JAM を使った基調講演（会場のスクリーン）



時は過ぎ、2005 年 6 月 14 日、私は「マクロメディア集中ゼミ」というイベントで講演することになりました。一体感が生まれるセッションにしたいといろいろ考えているとき、「Q-JAM」を使ったあの日の体験を思い出しました。私のスピーチに聴衆の興味を持続させる力など当然なく、そのために自分の携帯電話をいじる人が続出するだろう、であればはじめから携帯電話による参加型コンテンツを提供して 75 分の長丁場を乗り切ろう！　…という企みです。

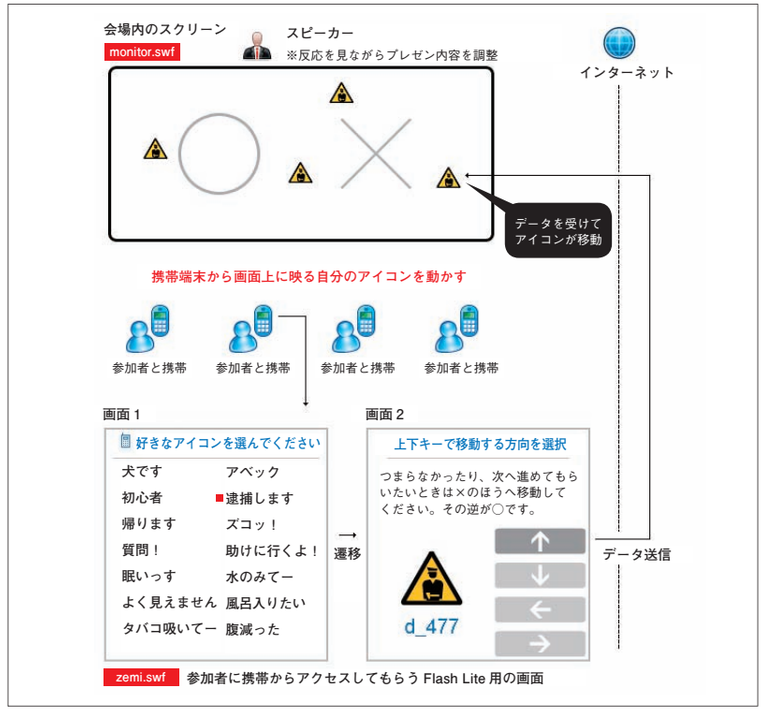
当日「Q-JAM」を使用するという手段があったかもしれません。しかし無料のはずはありませんし、何より「この手のコンテンツの体験を飛び越えて、実際に制作したい！」という思いが強かったからです。

電話のプッシュホンによる擬音語を組み合わせて「Pi!Po!Pa!」と命名し、「Q-JAM」と同じような仕組みを実現するための技術的課題の検討に入りました。

○ 全体像の整理

携帯電話と会場のスクリーンを接続し、来場者が 1 つの画面を共有し操作できるようにする、「Pi!Po!Pa!」の全体像は次のイメージです。

Pi!Po!Pa! の全体像



来場者には必ず、持参しているであろう自分の携帯電話から特定の URL へアクセスしてもらいます。アクセスすると、Flash Lite 用に書き出した zemi.swf が表示されます。これは上図の全体像でいう、「画面 1」と「画面 2」にあたります。「画面 1」では自分がこれから操作するアイコンを選び、「画面 2」ではそのアイコンを上下左右に移動するためのボタンをクリックします。このボタンクリックによりデータの送信が発生し、会場のスクリーン上にアイコンが現れ、指定した方向へリアルタイムに移動します。

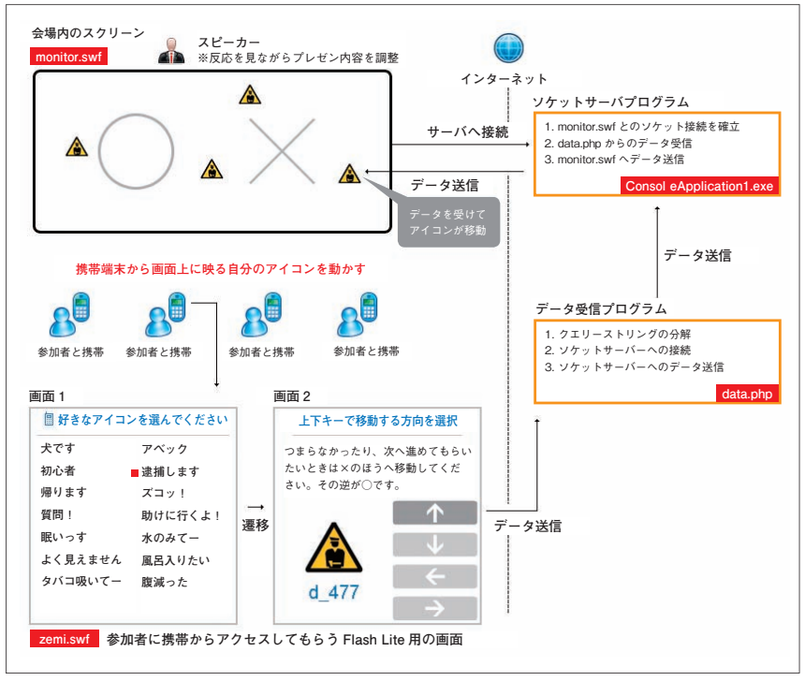
○ サーバサイドの仕様

肝心のサーバサイドの仕様についてはいろいろと模索しました。携帯端末で閲覧している zemi.swf が会場内のスクリーン（monitor.swf）と常時接続できれば、データの送信がシンプルになり一番よいのですが、携帯電話用 Flash からは Flash Communication Server などに接続することはできません。サーバサイドへデータ送信する術として使用できるのは Flash Lite 1.1 から新機能として追加された loadVariables メソッドだけです。

最初に考えたのは、この loadVariables を使って携帯電話からの情報をサーバに GET し、PHP プログラムが受信したデータをデータベースへ格納する方法です。そして、会場内の monitor.swf がデータベースを常時ポーリングしており、新しいデータのみを画面上に反映します。比較的簡単に実現できる方法ですが、これはデータのやり取りとしてあまりに下品すぎるため、ボツとしました。

最終的には、以下のようなイメージになりました。

サーバサイドの仕様



まず携帯電話からのデータを受け取る data.php を用意します。この PHP プログラムは、携帯電話からの情報をサーバ上で起動しているソケットサーバプログラムへわたし、処理を終了します。

ソケットサーバプログラムである ConsoleApplication1.exe は会場内のスクリーンである monitor.swf とソケット接続により、常時つながっています。data.php からデータを受信すると、接続している monitor.swf へデータをキャストします。monitor.swf は受け取ったデータの内容に従って、指定されたアイコンを移動させます。

この仕組みにより、携帯電話のボタンクリックでリアルタイムにアイコンが移動するレスポンスを提供できます。

スクリプト
携帯電話と Flash の通信機能の実装

前述したように、プログラムは 4 つのファイルになります。

- 携帯電話で表示する Flash Lite 用の zemi.swf
- 携帯電話からのデータを受け取る data.php
- ソケットサーバプログラム ConsoleApplication1.exe
- 会場内のスクリーンで表示する monitor.swf

各ファイルごとに説明していきます。

○ 携帯電話で表示するムービー

残念なことに、携帯電話用 Flash の作成にあたっては大昔の ActionScript を使用しなければなりません。しかし、やることは以下のように非常にシンプルです。

❶ アイコンの選択（フレーム 1 と 2）

「タバコ吸いてー」といったようなアイコンのラベルのみを表示したボタンを配置しています。各ボタンには以下のように記述し、選択されたタイミングで au という変数にアイコンを識別するための文字列（a ～ n）をセットしています。これは会場内のスクリーンに表示するアイコンで、ユーザーにとってはどれが自分か識別するためのものになります。

SOURCE 変数 au のセット

```
on (press) {
   /:au = "n";
   gotoAndPlay("ready");
}
```

❷ ユニーク ID の生成（フレーム 3）

会場内のスクリーンに表示される monitor.swf がどのアイコンを移動させればよいかを判別できるように、ユニーク ID を生成します。具体的には yname という変数に au の値（1 で作った変数）とランダムに生成した数値を結合した文字列をセットしています。

❸ アイコンの表示（フレーム 3）

au の値を判定し、選択したアイコンを端末上に表示します。

❹ 上下左右のボタン（フレーム３以降）

上下左右に向いた矢印のボタンを４つ配置し、クリックで data.php を叩くための記述を加えます。変数名 cmd には移動する方向を示す u（up）、d（down）、l（left）、r（right）のいずれかの値がセットされます。

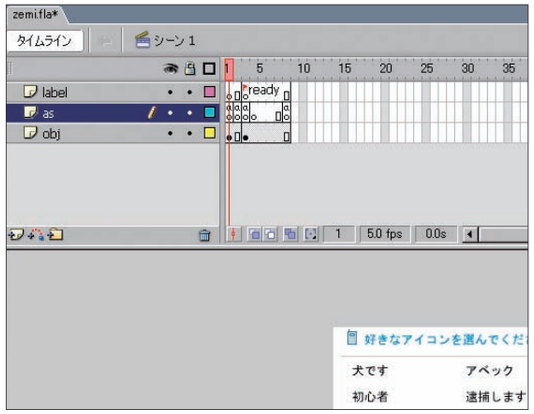
SOURCE 上下左右ボタンの記述

```
on (press, keyPress "1") {
    /:cmd = "u";
    loadVariablesNum("data.php", "", "GET");
}
```

各ボタンは矢印の方向に合わせて十字キーのように配置したかったのですが、携帯電話は通常、ブラウジングにあたって端末の左キーを押すと前の画面に移動する機能を持っています。

このため十字キーになるよう配置するとまぎらわしいので、４つのボタンを上から下に縦並びさせています。

ムービーの構造



○携帯電話からのデータを受け取る PHP プログラム

data.php の全コードは以下になります。

SOURCE data.php

```
<?
$strTx = "";
if(isset($_GET['yname']) && isset($_GET['cmd'])){
    $cmd = $_GET['cmd'];
    $yname = $_GET['yname'];
    //b_460-d といったような形式で届く
    if($cmd =="u" || $cmd =="d" || $cmd =="l" || $cmd =="r"){
        //$fp = stream_socket_client("tcp://127.0.0.1:12000", $errno, $errstr);
        $fp = fsockopen("tcp://127.0.0.1", 12000);
        $strTx = $yname . "-" . $cmd . "/";
        fwrite($fp,$strTx);
        fclose($fp);
    }
}
```

次のページに続く


```
exit;
?>
```

まず、最初の条件文で携帯電話用 Flash（zemi.swf）から送られてきた yname と cmd を取り出します。次に、fsockopen 関数でローカルサーバ内の 12000 番ポートを指定してソケット接続を開始します。ソケット接続とは、アドレスとポート番号を指定することでサーバとのデータの送受信を可能にしてくれる、TCP/IP を使った通信方法です。ここではソケットを開いて関数 fwrite() で情報を送信しているだけの、もっともシンプルな使い方をしています。ちなみにコメントしてある箇所の stream_socket_client は PHP5 から使用できる関数です。

○ソケット接続を行うプログラム

● 開発環境の用意

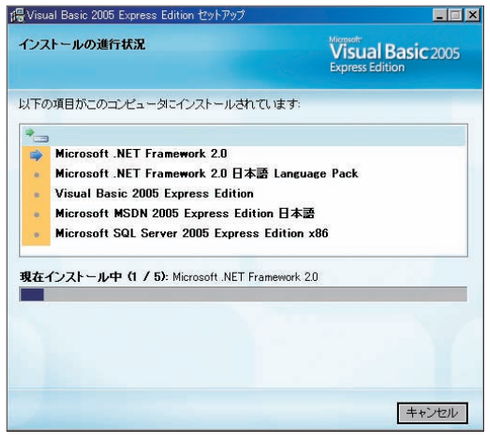
ConsoleApplication1.exe の開発にあたっては、「Visual Basic 2005 Express Edition 日本語版」を使用しました。これは、１年間にわたって無償で使うことができる評価版です。以下の URL の「CD イメージのダウンロード」から「vb.iso」ファイルをダウンロードしてください。

URL㊟ Visual Basic 2005 Express Edition 日本語版
http://www.microsoft.com/japan/msdn/vstudio/express/vbasic/

インストーラーはイメージ形式による提供のため、ダウンロードした vb.iso をマウントする必要があります。マウント用ツールがない場合は別途用意してください。私が愛用している「DAEMON Tools」の場合、以下の URL からダウンロードできます。

URL㊟ DAEMON Tools
http://www.daemon-tools.cc/dtcc/download.php?mode=ViewCategory&catid=5

インストール画面



vb.iso をマウントし、インストールを開始します。すべてのインストールが完了するまでに 15 分ほどの時間がかかります。ConsoleApplication1.exe を実行するために必要になる .NET Framework 2.0 も同時にインストールされます。

なお、「Visual Basic 2005 Express Edition 日本語版」の具体的なインストール方法については、以下のページに詳しく記載されていますので、問題があった場合には参照してください。

URL㊟ Visual Studio 2005 Express Edition —CD イメージからのインストール方法
http://www.microsoft.com/japan/msdn/vstudio/express/ maninstall/#installlocal

● ConsoleApplication1.exe での処理

¥ConsoleApplication1 以下にある ConsoleApplication1.sln を選択し、Visual Basic 2005 を起動します。そして画面右にある「ソリューションエクスプローラー」から Module1.vb を開いてください。ConsoleApplication1.exe の全処理はこの Module1.vb 内に記述しています。以下、重要な箇所をブロック単位で説明していきます。

❶ ソケットの定義

StateObject クラスでは、非同期通信を行うためのソケットの内容を定義しています。これは data.php（Module1.vb 内でクライアントと呼んでいるもの）からデータを受信した際や monitor.swf（Module1.vb 内でサーバと呼んでいるもの）へデータを送信する際に毎回生成されるオブジェクトになります。

❷ リスナーオブジェクトの作成

関数 StartClientListening() 内ではクライアントである data.php との接続方法をパラメータとして取り決めています。具体的には localhost からの接続のみで 12000 番のポートを使ってやり取りを行う宣言です。これは前述の data.php 内にあった以下の記述と対応する部分になります。

SOURCE 12000 番のポートの使用を宣言

```
$fp = fsockopen("tcp://127.0.0.1", 12000);
```

そのあと data.php からの接続を待つリスナーオブジェクトを作成し、頻発する接続要求をキューとして保持する最大数を100に設定しています。これにより、仮に100台の携帯電話から同時にリクエストがきた場合でも順次処理できるようになっています。クライアントからデータ受信を待つ意味で「Waiting for a Client connection...」の文字列をコンソール上に書き出しつつ、このリスナーオブジェクトと受信に際して発動するコールバック関数の AcceptClientCallback をひも付けています。

❸ クライアントからのデータ抽出とサーバへの送信

実際に data.php からわたされたデータを抽出し、monitor.swf へデータ送信している部分が ReadClientCallback になります。この中では、0 バイト以上のデータであることを確認した上で、受信データの末尾に付与されている区切り文字の「/」をチェックしています。「/」が存在する場合は、vbNullChar をデータ末尾に付けて monitor.swf へ送信します。これは、Flash の XML-Socket が 0 バイト文字を XML オブジェクトの終端文字として解釈するためです。

● ConsoleApplication1.exe のビルド

ConsoleApplication1.exe を書き出すには、メニューから「デバッグ」→「デバッグの開始」をクリックします。ビルドする場合は、メニューから「ビルド」→「ConsoleApplication1 のビルド」を実行してください。ConsoleApplication1.exe は以下のフォルダに格納されます。この実行形式ファイルは、サーバ内であれば場所を問わず動作します。

```
¥ConsoleApplication1¥ConsoleApplication1¥bin¥Debug¥
```


○ 会場内のスクリーンに表示するムービー

最後は、会場内から起動し、参加者が共有する画面となる monitor.swf です。大きく、4 つの構成になっています。

❶ ConsoleApplication1.exe への接続

以下のように XMLSocket オブジェクトを作成し、アドレスとポート番号を引数に関数 connect() を実行します。onConnect は接続時に、onClose はサーバとの切断時に発生するハンドラです。そして接続した XML オブジェクト経由でデータ送受信が発生するたびに実行されるのが、onXML です。

SOURCE 接続処理

```
socket = new XMLSocket();
socket.connect(server, port);
socket.onConnect = onSockConnect;
socket.onClose = onSockClose;
socket.onXML = onSockXML;
```

❷ 受信データの分解

XML-Socket といっても、XML ではなく普通のストリングを受信することができます。関数 onSockXML() 内では受信したデータ（data.php で送信した情報そのまま）を分解し、配列にセットしています。その後に指定のアイコンを移動するための関数 mvIcon() へ引き継ぎます。

SOURCE 受信データの分解

```
//--------------------------------------------
// データ受信
//--------------------------------------------
function onSockXML(stringxml) {
    var rawStr = String(stringxml);
    //trace("データ受信  = "+rawStr);
    if (rawStr != null && rawStr != "" && rawStr != undefined) {
              // 動作待ちの情報を保持する配列
              wArray = new Array();
              var tmpArray = rawStr.split("/");
              for (var i = 0; i<tmpArray.length; i++) {
                   if (tmpArray[i] != null && tmpArray[i] != "" && tmpArray[i] != undefined)
    {
                        wArray.push(tmpArray[i]);
                   }
              }
              mvIcon();
    }
}
```

❸ アイコンのムービークリップへ指示を送る

mvIcon 関数では、まず「b_460-d」といったような形式で配列内に保持されている値を取り出します。「b_460」の部分が data.php でいう $yname にあたり、これは携帯電話用 Flash の zemi.swf にてユニーク ID として生成した値です。その後に続く「d」の箇所が data.php の $cmd に該当します。ステージ上に $yname の値と同名

のMCが存在する場合には、そのMC内のonEnterFrameに移動方向（$cmd変数内の値）をわたします。ステージ上に存在しない場合はattachMovie()を実行し、その際に$ynameの値を生成する新規MCの名前として使用します。

❹ アイコンの移動

アイコンのMC内では、前述のmvIcon関数から受け取った移動方向へ拡大しながらコミカルに座標を変えていくonEnterFrameが定義されています。

単純な座標移動だとおもしろみに欠け、複数人で画面を共有していると自分のアイコンが何処にいるのか判別がむずかしかったため、カエルが飛び跳ねるような動きにしました。また、画面右上に非常口のボタンを設置し、アイコンがぶつかるとステージ中央に飛ばされる遊びを裏技的に入れました。

○ Pi!Po!Pa! の実行のまえに

Pi!Po!Pa!を動作させるためには、最初にサーバ上でConsoleApplication1.exeを実行します。次に、monitor.swfを起動しConsoleApplication1.exeとの接続を確立します。あとは携帯電話からzemi.swfへアクセスし、上下左右のボタンを押すことでmonitor.swf上にてリアルタイムなレスポンスが確認できます。

monitor.swfからは11000番のポートを使用してConsoleApplication1.exeと通信します。そのため、ルータの設定にて11000番のポートを空ける作業を忘れずに行ってください。

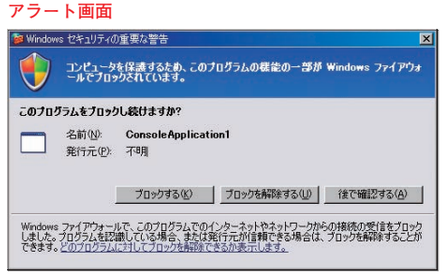
● .NET Framework 2.0 のインストール

ConsoleApplication1.exeの実行にあたっては、実行環境に「.NET Framework 2.0」がインストールされている必要があります。インストールされていない場合には、事前に以下のURLから「Microsoft .NET Framework Version 2.0 再頒布可能パッケージ」をダウンロードし、インストールを行ってください。

URL㊟ **http://www.microsoft.com/downloads/details.aspx?FamilyId=0856EACB-4362-4B0D-8EDD-AAB15C5E04F5&displaylang=ja**

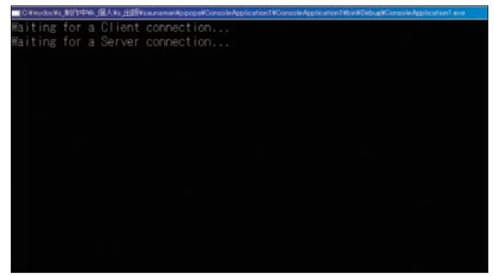
ConsoleApplication1.exeはビルドすると、「¥ConsoleApplication1¥ConsoleApplication1¥bin ¥Debug¥」以下に配置され、Windows XP環境にて実行すると以下のようなアラートが出る場合があります。これは、ファイアウォールを飛び越えてソケット通信を行うプログラムの起動の際に表示される警告です。ブロックを解除して、動作を確認してください。

アラート画面



起動するとウィンドウ内にメッセージが表示されたコンソール画面が開きます（右画面）。

コンソール画面



このあとmonitor.swfを実行すると、画面内にクライアント接続があったことを示す「Waiting for a Server connection... 」の1行が現れます。そして、data.phpからデータを受信するたびに「Waiting for a Client connection... 」の1行が追加されていきます。これは、ConsoleApplication1.exeを間に挟んで、携帯電話と会場内スクリーンがつながっていることを証明する1文となります。

○ 今後の課題

先の講演の結果をいえば、「Pi!Po!Pa!」を使った私の講演は大失敗に終わりました。この作品に問題があったかどうか以前に、私がスピーチの内容をまったく整理せずに本番を迎えたためです。いいわけをすれば、本番直前まで開発を行ったため、スピーチの内容までブラッシュアップしておく時間がありませんでした（それほど開発に没頭していた原因は、「同じ画面を複数人でリアルタイム共有するインタラクションをどうしても体験してほしい！」という思いにつきます）。

それはさておき、今後の課題としては

- **理解を求めるためのマニュアルの必要性**
- **対応機種**

という点があります。私の講演では、「眠いプレゼンを楽しく過ごすためのコンテンツ」と題した簡単な操作マニュアル（howto_outline.swf）を会場入り口にて配布しました。

howto_outline.swf

また対応機種については、Flash Lite1.1に対応する携帯電話が対象となるため、auの3G携帯とNTTドコモの最新機種に動作が限られてしまいます。

私は、渋谷や新宿などの大きな交差点に立つと、目の前の巨大なスクリーンを信号待ちするみんなで共有したくなります。「Q：今一緒にいる人が実は嫌い？」などといった質問をスクリーン上に表示し、携帯電話を使ったリアルタイム集計を取るだけで、信号が青に変わるのを楽しく待てそうです。

課題としてあげた部分が克服できれば、一体感を生む参加型コンテンツが場所を問わず巷にあふれるかもしれません。これから登場するFlash Lite 2.0に期待しつつ、そんな妄想を抱きます。