Rich Internet Clients – First Steps with AJAX

By Thomas Maul, General Manager, 4D Germany.

TN 06-05

Abstract

このテクニカルノートでは、Ajax の基本的な背景情報を解説し、4D で Ajax ベースのデザインを導入する際に必要な最初のステップを説明しています。Ajax では XML を多用する必要がありますが、4D には XML 関連機能が豊富に用意されているので、Ajax ベースソリューションを短期間で開発することができます。

Introduction

多くのインターネットアプリケーションがユーザインターフェースを簡単なものにとどめている中、最近、新世代の Web アプリケーションが登場しました。そのような Web アプリケーションは、通常のデスクトップアプリケーションに匹敵する高級なインタフェースを有しているのが特徴です。一例として、Google（Google Mail、Google Map、Google Suggest）や Flickr（http://flickr.com/）を挙げることができます。

そのような Web ページでは、タイプアヘッド、ドラッグ＆ドロップ、アニメーションなどが Web サーバとの非同期通信と一緒に使用されています。"普通の"Web ページとは異なり、リクエストを送信してページ全体をリロードするようなことはありません。ページの一部だけが更新され、高速でユーザフレンドリなクライアントを実現しています。

このテクニックは、Asynchronous JavaScript and XML（Ajax=非同期 JavaScript+XML）と呼ばれ、既存の規格を幾つか組み合わせることで成り立っています。実際に使用している Web サイトは比較的少数ですが、決して新しい技術ではありません。ニューオーリンズで開催された 4D Summit 2004 では、XMLHTTPRequest という主題で Dashboard 用の Ajax セミナーも開かれました。テクニカルノートで Ajax を取り上げるのは、今回が初めてになります。

What is Ajax?

英文の Wikipedia（http://en.wikipedia.org/wiki/Wikipedia）には、次のような Ajax の定義が掲載されています：

Ajax(アジャックス、エイジャックス)とは、Asynchronous JavaScript and XML の略で、次のものを使用し、インタラクティブな Web アプリケーションを作成するための Web 構築技術である：

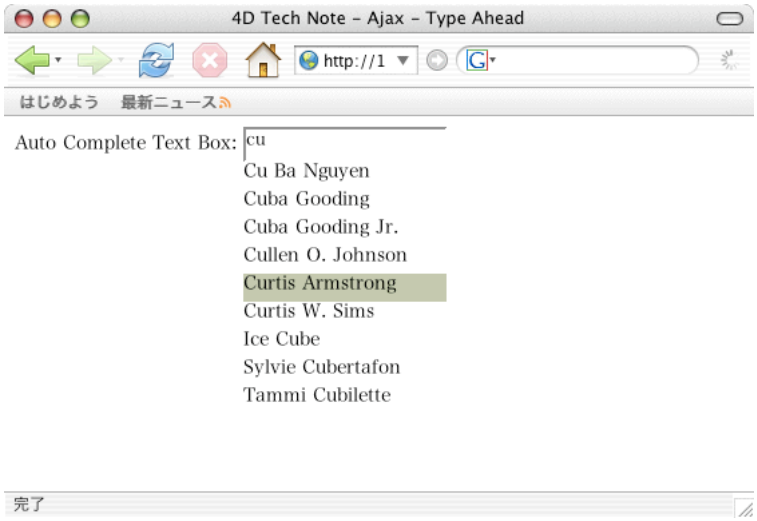
- マークアップおよびスタイル情報の XHTML（または HTML）と CSS。（標準テキスト、JSON、EBML、フォーマット済 HTML など形式は不問だが、一般的には XML が使用される。）
- JavaScript で制御され、情報に応じて動的またインタラクティブにページの一部を書き換える Document Object Model。
- Web サーバと非同期通信を実現するための XMLHttpRequest HTTP（通信を行うための JavaScript 組み込みクラス)オブジェクト。一部の Ajax では、Web サーバと通信するために XMLHttpRequest の代わりに IFrame オブジェクトが使用されている。

Some examples...

この定義では複雑で難解に思えるかもしれません。Ajax は、実際に動かして試してみたほうが理解できるものです。Mac あるいは Windows 版の 4D 2004 で、サンプルデータベースを起動してみてください。このサンプルはバージョン 2004 標準デモアプリケーションの Video Library を改造したもので、データベースには約 1500 本の映画と 13000 名の俳優の名前が登録されています。

Launch the Browser ボタンをクリックして Web ブラウザを起動し、ローカル 4D Web サーバに接続してみてください。（あるいは手動で http://127.0.0.1:8080 を開いても構いません。）4D Web Server 2004 のライセンスがない場合、サーバはデモモードで動作します。

Type Ahead: Actor Suggest リンクをクリックしてみてください。ActorSuggest.html ページが読み込まれます：



好みの役者の姓または名をタイプ入力してみてください。名前をタイプするに従って、画面上のテキストボックスには、検索によって絞り込まれた名前が表示されてゆきます。注記：ブラウザは JavaScript をサポートしている必要があります。JavaScript が有効にされていない場合、このタイプアヘッドは作動しません。タイプアヘッドがかなり高速であることに注目することができます。入力とほぼ同時に候補のリストが変化します。このシステムでは、最初の名を検索し、候補を 10 件まで表示しています。名の候補が 10 件に満たない場合、姓も検索の対象に含められます。例えば、"cu"あるいは"cur"とタイプしてみてください。全体で 13000 件の人名がデータベースに登録されていることを考えると、この速度は特筆に値します。

注記：最高のパフォーマンスを得られるのは、4D がインデックスをキャッシュに読み込んだ後です。したがって、速度を上げるため、最初に数件の検索を実行する必要があります。

次にキーボードの上下キーで選択項目をナビゲートしてみてください。Enter キーを押すと選択したフルネームがフィールドに入力されます。マウスで項目をナビゲートすることもできます。

Firefox と一緒に Live HTTP Headers エクステンションがインストールされているのであれば、Live HTTP Headers ダイアログを表示し、Auto Complete Text Box の内容に合わせてヘッダが変化する様子を確認してみてください。注記：Live HTTP Headers は、次の URL から自由にダウンロードすることができます：

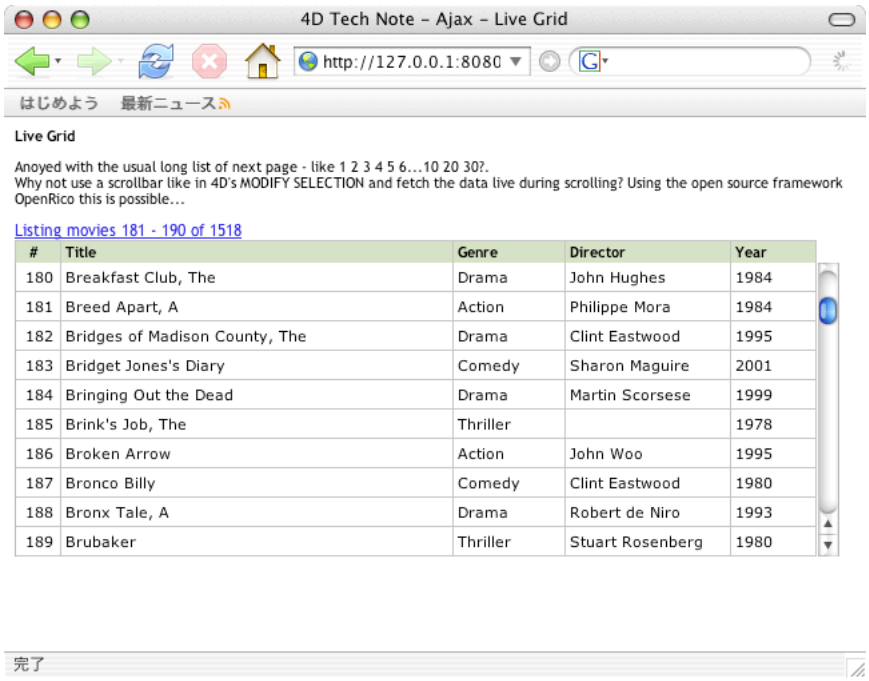
http://livehttpheaders.mozdev.org

これは Web プロジェクトをデバッグする上では非常に重宝するエクステンションで、Ajax 開発には必須のアイテムといっても過言ではありません。

テキストがタイプされると、裏ではブラウザと 4D Web サーバの間で通信が交わされ、テキストボックスの現在の値と 4D の返り値がやり取りされている点に注目してください。クライアントはキャッシュを使用しており、すでに把握しているデータについては、再度サーバにリクエストを送信しないようになっています。この動作を確認するために、"alc"とタイプしてみてください。この条件に合致するレコードは 4 件しか存在しません。"alca"または"alco"と入力を継続したとしても、結果を絞りこむだけなので、サーバにリクエストを送信する必要はなく、実際、送信されません。

なかなか優秀ではないでしょうか。それでは次のサンプルをみてみましょう。ブラウザの戻るボタンを使用してください。一度のクリックでホームページに戻ることができます。タイプアヘッドで実施された検索は Web ページの URL を変更しないため、何度繰り返したとしても、ブラウザはページが移動していないものとみなします。

Live Grid リンクをクリックしてみてください：



右側の縦スクロールバーに注目してください。

多くの Web ページでは、データ数が多い場合、複数のページに対するリンクを作成します：

<Previous page> 1 2 3 4 5 6 7 8 9 10 20 30 40 <next page>

この方法は非常に一般的で、あまり深く考えられずに使用されている場合もあるようです。

これに対し。4D デベロッパは 20 年以上前から使用されてきた MODIFY SELECTION のコンセプトに慣れています。4D Server 1.0（1992）以降の 4D Client インタフェースでは、画面に表示されているレコードだけがネットワークを介して送信され、ユーザがスクロールバーを操作してセレクションの残りをブラウズすると同時に必要なデータが送られる仕組みになっています。

Ajax を使用すれば､そのような動作を Web アプリケーションを実現することができるのです。

Live Grid ページのスクロールバーを操作してみてください。クライアントマシンのスピード次第では即座、あるいは少し遅れてページが再描画されます。スクロールバーが動かされると、ブラウザは URL を変更することなく動的にリクエストを 4D の Web サーバに送信します。スクロールバーの上下ボタンについても同様です。

HTTP Live Headers を使用してクライアント-サーバの通信を監視すると、それが確かに最適化されていることが分かります。最初、画面には 10 レコードが表示されますが、ブラウザは 70 レコード分のデータをリクエストしています。スクロールバーを下に操作すると、次の 70 レコードがリクエストされます。1 秒間、何の操作も起きなければ、ブラウザは前の 70 レコードをリクエストし、上方向へのスクロールに備えてキャッシュします。

Live Grid ページでは、ヘッダクリックによる並び替えも可能です。Year または Director をクリックしてみてください。もう一度クリックすると、並び替えが実行されます。

How does Ajax work?

Ajax は、幾つかの技術の組み合わせによって実現しています。
まずもっとも重要なのが JavaScript です。JavaScript はクライアント（Web ブラウザ）のアクションをすべて制御するために使用されます。Ajax デザインコンセプトは JavaScript を抜きにして語ることはできません。同時にこれは最大の弱点でもあります。JavaScript が有効にされていなければ、まったく作動しないからです。JavaScript はセキュリティ上の理由から無効にされている場合もあるため、代替ソリューションとして非 Ajax インタフェースも用意しなくてはならないかもしれません。

それぞれのオブジェクト、例えばテーブルのセル（LiveGrid サンプルでは 10 行のテーブルを使用しており、疑似スクロールを表現するために JavaScript がその内容を書き換えています）などの制御を可能にしているのが Document Object Model（DOM）です。

Cascading Style Sheets （CSS）はアピアランスを制御し、オブジェクトの作成、移動、リサイズを随時実行するために使用されます。サンプルではタイプアヘッドに CSS が使用されていました。

もうひとつの技術が XMLHTTPRequest オブジェクトです。これは Microsoft Internet Explorer 5.5 および Safari 1.2 で導入されたもので、Web ブラウザから非同期リクエストを送信し、ユーザやネットワークに負担を強いることなく、バックグラウンドで処理することを可能にしている主要素です。

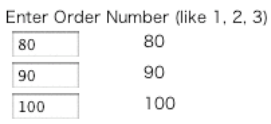
このテクニカルノートでは JavaScript のごく基本的な内容を取り上げていますが、JavaScript、DOM、CSS について詳細な説明はしていません。こうしたトピックについては多数のトレーニング本や Web サイトが存在するので、そちらを参照してください。

JavaScript

JacaScript や DOM を扱ったことがあれば、この部分は読み飛ばしてください。

JavaScript はシンプルなスクリプト言語です。シンタックスが部分的に対応している以外 Java とは関係がありません。詳しくは Wikipedia を参照してください：
http://en.wikipedia.org/wiki/JavaScript
上記のページには JavaScript 習得のための基本的なステップ、ガイド、チョートリアルなどに対するリンクが掲載されています。

サンプルデータベースには簡単な JavaScript のページが含まれています。ブラウザの戻るボタンをクリック（あるいは http://127.0.0.1:8080 を再読込）して Simple HTML をクリックしてみてください：



テキストボックスに数字（または文字）を入力し、そのボックスからフォーカスを移すと、入力された値が隣の（入力不可）テキストボックスにコピーされます。4D の On Data Change フォームイベントによく似た動作です。

ブラウザでソースを表示（あるいは WebFolder の中にある SimpleHTML.html をテキストエディタで表示）してみてください。HTML の BODY パートにはフォームがあり、そのフォームにはテーブルがあります。テーブルにはヘッダが 1 行とテキストボックスが 3 行の合計 4 行が存在します。各行にはセルが 2 個ずつ存在します。セルの HTML は次のとおりです：

```
<td width="103" colspan="2" class="in_label" nowrap>
    <input name="artNew1" type="text" class="inp" style="width: 50px;
        margin-left: 5px;" ID="Input1" onChange="setTextValue('Text1', this)">
</td>
<td width="450" class="in_label" ID="Text1">
</td>
```

それぞれのセルに ID が定義されている点に注目してください。この ID が今回の鍵となっている要素であり、JavaScript でページ内の任意のオブジェクトに直接アクセスすることを可能にしています。

最初のセルの HTML には、スタイル情報の他に次のようなコードも含まれています：

```
onChange="setTextValue('Text1',  this)"
```

4D に置き換えてみるならば、これは On Data Change イベントの中で setTextValue メソッドをコールしていると考えることができます：

```
$event:=Form event
Case of
    ¥ ($event:=On Data Change )
        setTextValue(“Text1”, Self)
End case
```

JavaScript メソッドが受け取るパラメータは 2 個、つまり"Text1"という内容のテキスト定数とカレントオブジェクトに対するポインタ（4D では Self、JavaScript では this）です。

最初のパラメータ"Text1"は更新するセルの ID です。JavaScript メソッドは HTML ページのヘッダで定義されています：

```
<script type="text/javascript">
function setTextValue(id, obj)
{
    document.getElementById(id).innerHTML=obj.value;
}
</script>
```

関数の本体は一行だけで、「ID で document から element を get する」となっています。ユニークな ID があれば、この方法で任意のオブジェクトにアクセスすることができます。属性の

innerHTML はオブジェクトに値を代入するために使用されています。もうひとつのパラメータ obj は一種のポインタで、新しい値を含んでいる呼出元オブジェクトを指しています。

この先は JavaScript の基本的な知識があることを前提に書かれています。読み進めるうちに困難を覚えるようであれば、JavaScript の入門書などを参照するとよいでしょう。

JavaScript のデバッグは大変な作業になる場合があります。例えば、ブラウザが何も表示しないことがあります。プログラムエラーが存在する場合、単純に実行を中止してしまうからです。Firefox には実行エラーのリストを表示する JavaScript Console というツールがあり、開発の大きな助けになります。そのほかのツールとしては JavaScript をトレースし、変数の読み書きができる Venkman JavaScript Debugger というフリーのデバッガがあります：
http://www.mozilla.org/projects/venkman/

The XMLHTTPRequest Object

XMLHTTPRequest は Microsoft 社によって開発され、Internet Explorer 5.0 では JScript や VBScript など、ブラウザがサポートするスクリプト言語でアクセスできる ActiveX オブジェクトとして導入されました。Mozilla チームは、互換性のあるネイティブバージョンを Mozilla 1.0 に実装し、続いて Apple 社が Safari 1.2、Opera Software は Opera 8.0 で実装させました。このオブジェクトはバックグラウンド動作で Web サーバからデータを取得するために使用されます。データは一般に XML でやり取りされますが、実際、どんなテキストデータでも使用することができます。

XMLHTTPRequest については、4D Summit 2004 セッションノート 377-381 ページで詳しく論じられています。

もっとも単純な XMLHTTPRequest セッションはこのように記述されます：

```
var oXMLHTTP = new ActiveXObject("Microsoft.XMLHTTP")
var sURL = "/csutomerIDcheck.xml?username="+custid;
oXMLHTTP.open("POST", sURL, false);
oXMLHTTP.send();
alert(oXMLHTTP.responseText);
```

Microsoft Object はすべてのブラウザがサポートしているわけではありません。

Mozilla は別のオブジェクトを使用しているので、両方のブラウザに対応するためには、利用できるオブジェクトを特定する必要があります：

```
if (window.XMLHttpRequest){
oXMLHTTP =new XMLHttpRequest();
} else if (window.ActiveXObject){
oXMLHTTP =new ActiveXObject("Microsoft.XMLHTTP");
}
```

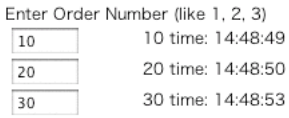
XMLHTTPRequest.open メソッドの 3 番目のパラメータは async フラグで、サーバがリクエストに応答するまで待機するのかをブラウザに指示します。非同期モードで動作するようにするためには、サーバからのレスポンスを処理するための追加コーディングが必要です。(この点については後述します。)

非同期モードにおける例外処理はなかなか骨の折れる作業です。それでも、例題が豊富に存在するというのは朗報かもしれません。単純に Google で XMLHttpRequest を検索してみてください。Apple 社の Web サイトには、Windows 版の Microsoft Internet Explorer にも対応した例題が公開されています：
http://developer.apple.com/internet/webcontent/xmlhttpreq.html
高度にダイナミックな Web アプリケーションを開発するのであれば、JavaScript のフレームワークを利用するとよいでしょう。Prototype や Rico などの JavaScript フレームワークには、XMLHttpRequest オブジェクトを処理するためのコードが含まれています。Type Ahead および Live Grid サンプルを参照してください。

The Rich Internet Client

このステップでは JavaScript、XMLHTTPRequest そして 4D を組み合わせて使用します。

Simple Ajax のページを開いてみてください。



表面的には Simple HTML サンプルに似ていますが、4D と通信するために XMLHTTPRequest が使用されています。

テキストボックスに適当な数字を入力してみてください。タブキーなどを使用して別のエリアに移ると、入力した値と現在の時刻が隣に表示されます。日付のデータは 4D から送信されたものです。

Webfolder フォルダの中にある SimpleAjax.html ファイルをテキストエディタなどで開き、SimpleHTML.html と比較してみてください。HTML の BODY 部分は同じですが、ヘッダだけが異なっています。

Ajax を使用しないほうのページである SimpleHTML.html では、次のように setTextValue 関数が定義されています：

```
function setTextValue(id, obj)
	{
	document.getElementById(id).innerHTML=obj.value;
	}
```

Ajax 版では次のようにふたつの関数を定義しています：

```
function setTextValue(id, obj)
	{
	var url = "/4DAction/Ajax_SimpleRequest/"+obj.value;
	var loader=new net.ContentLoader(url,   ActionCompleted);
	loader.returntarget = id;
	}
function ActionCompleted()
	{
	document.getElementById(this.returntarget).innerHTML=
	this.req.responseText;
	}
```

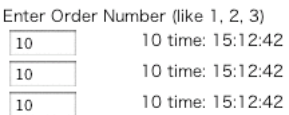
最初の関数 setTextValue は、変数を作成して URL を割り当てています。この URL は 4DAction を使用して 4D のメソッド Ajax_SimpleRequest をコールしており、パラメータとしてはテキストボックスの中身を渡しています。

2 行目は Ajax 通信を処理するオブジェクトを指定しています。このオブジェクトの定義は scripts/ContentLoader.js ファイルにあります。この行ではオブジェクトを作成し、URL とサーバからのレスポンスを受け取った場合にコールする関数の名前（ActionCompleted）を渡しています。

最後の行は、リクエストを送信し、更新されるオブジェクトの ID を残しておくためのものです。

Web サーバからのレスポンスを受け取ると、すぐに ActionCompleted 関数がコールされ、データが this.req.responseText に代入されます。

このサンプルには潜在的な問題があります。1 分間、待って同じ値を入力してみると、時間が更新されないことに気づきます：



これは Web ブラウザが 4D Web サーバにリクエストを送信しないでキャッシュを使用したた

めです。この点は Live HTTP Headers で確認することができます。サーバから返される値が一定の場合（例：製品名）、ネットワークの混雑が抑えられて便利ですが、変動するようなデータの場合（例：在庫数）これは大きな問題です。この対策については後述します。

4D のメソッド Ajax_SimpleRequest の中身は次のようなものです：

```
$request:=Substring($1;2) ` parameter contains entered text, starting with " / "
$answer:=$request+" time: "+String(Current time) ` calculate the answer
SEND HTML TEXT($answer) ` send the answer
```

1 行目にブレークポイントを設定し、ブラウザから新しいデータを送信してみてください。反応の遅い Web サーバを再現するためです。行目でデバッガが起動し、先へ進むまでレスポンスが返されません。ブラウザに表示される時刻は更新されませんが、それでも引き続きブラウザを使用することができます。このようにレスポンスが遅い（あるいはない）場合でも、アプリケーションが止まらないというのが非同期の強みです。

4D に戻り、デバッガのトレースなしボタン（緑色の三角アイコン）をクリックしてください。（デバッガはリクエスト毎に起動します）ブラウザでは、すぐに結果が正しいオブジェクトの隣に表示されます。レスポンスに対応するオブジェクトは loader.returntarget 変数で管理されているためです。

The Web Browser Cache

Web ブラウザのキャッシュが非同期通信による更新の障害になることについては、すでに言及しました。これを回避するにはいろいろな方法があります。Google で Ajax browser cache を検索すれば、同じトピックに関する様々な考察をみつけることができます。

一番単純な解決策は、POST コマンドを使用して HTML FORM フォームを送信するというものです。この場合、ブラウザはキャッシュを使用せずに必ずリクエストをサーバに送ります。

Simple Ajax – using form はそのようなサンプルです。リクスエトを送信し、別の時間に同じ値を入力してみてください。キャッシュが使用されないため、表示中の時刻が更新されます。

この場合、4D のメソッドは次のように書き換えることになります：

```
ARRAY TEXT($name;0)
ARRAY TEXT($values;0)
GET WEB FORM VARIABLES($name;$values)
If (Size of array($values)>0)
	$request:=$values{1}
```

```
        $answer:=$request+" time: "+String(Current time)
        SEND HTML TEXT($answer)
Else
        $answer:="Server communication error - no parameter passed"
        SEND HTML TEXT($answer)
End if
```

パラメータを取得するために GET WEB FORM VARIABLES コマンドが使用され、それに現在の時刻を追加するようになっています。

Web ブラウザのキャッシュ問題を回避する別の方法は、HTTP ヘッダを書き換えて有効期限を変えてしまうというものです。この方法は、Actor Suggest および Live Grid ページで使用されています。

Examples In Depth...

Double Combo Box

Simple Ajax は極めて簡単なサンプルですが、それでも Ajax デザインの基本的なコンセプトが実践されており、体感的にそのポテンシャルを理解することができると思います。

次のサンプル Double Combo Box では、より多くの JavaScript コードが記述されており、汎用的なコードが活用されています。

このサンプルを作る上で参考にした Ajax in Action という書籍の第 9 章はフリーでダウンロードすることができます：
http://www.manning.com/books/crane/chapters
この書籍では、1 番目のコンボボックスの内容に応じて 2 番目のコンボボックスの内容が変動するインタフェースの作り方が説明されています。たとえば、レンタルビデオ店のオンライン予約システムで、最初のコンボボックスでジャンルを選択すると、次のコンボボックスにはサーバから返された貸し出し可能なアイテムだけが表示されるようにできるかもしれません。

ちなみにこのサンプルには潜在的な問題点があります。ジャンルを Action に変更した直後に 2 番目のコンボボックスをクリックすると、前のジャンルのアイテムが残っています。数秒後には Action のアイテムになりますが、この遅延は Action ジャンルの映画が 500 本以上も存在し、それだけのコンボボックスを JavaScript が構築するまでに時間がかかることに起因しています。これはコンボボックスの悪い使用例です。項目数はもっと少なくしなければいけません。とはいえ、非同期通信の可能性を示す上ではなかなか優れた見本なのではないでしょうか。

4D のコードは前のサンプルによく似ており、非常に簡単です。ここでも GET WEB FORM VARIABLES コマンドでパラメータを取得しています。レスポンスには多くのデータが含まれるので、XML で送信します。Ajax の最後の x は XML の X ですが、実際には XML 以外の形式、つまり標準テキスト、エンコードされたピクチャ、HTML など、どんなテキストデータでも構いません。XML の構築には 4D の DOM コマンドを使用すると便利です。

Double Combo Box の HTML は次のようになっています：

```
<head>
      <script type="text/javascript"   src="ricoscripts/prototype.js"></script>
      <script type="text/javascript"   src="ricoscripts/rico.js"></script>
      <script type="text/javascript"   src="ricoscripts/doubleCombo.js">
      </script>
      <script type="text/javascript" >
      function injectComponentBehaviors()
      {
      var doubleComboOptions = { };
      new DoubleCombo( 'Group',
      'Movies',
      '/4DAction/Ajax_DoubleCombo',
      doubleComboOptions );
      }
      </script>
</head>

<body onload="injectComponentBehaviors()">
 ...
<select name="Group" ID="Group">
      <option value="-1">Pick a categorie</option>
       ...
</select>
<select name="Movies" ID="Movies" style="width:200px">
 ...
</body>
```

JavaScript が前述のフレームワークを使用している点に注目してください。

コンボボックス（<select>タグ）の中には JavaScript が何も記述されていないので、簡単に移植することができます。機能的な部分は HTML のヘッダが受け持っています。主な処理をしているのは、DoubleCombo オブジェクト（injectComponentBehaviors で定義）です。最初のパラメータがメインのコンボボックスを設定し、2 番目のパラメータで 2 番目のコンボボックスを設定しています。オブジェクトの判別にはタグの ID が使用されています。3 番目のパラメータがコールする URL です。4 番目のパラメータは、今回未使用ですが、任意のオプションを渡すために利用することができます。

このような仕組みになっているので、とても簡単に既存 Web ページに組み込むことができます。JavaScript のコードについて詳細が知りたければ、Ajax in Action の第 9 章を参照してください。なお、Safari に対応するため、DoubleCombo.js には若干、手が加えられています。

Type Ahead

このサンプルのデザインについてはすでに言及しました。ここでは 4D のコードについて説明します。

4D のメソッド AjaxActorSuggest は、バックエンド部分を処理します。コードはコンボボックスのサンプルとよく似ており、パラメータを GET WEB FORM VARIABLES で受け取って結果を XML ドキュメントで返しています。

JavaScript のコードもコンボボックスのサンプルと同じコンセプトを採用しており、汎用的な関数が HTML に機能性を追加しています。ActorSuggest.html と DoubleCombo.shtml を比較してみると、非常に似ていることが分かります。

Live Grid

今回のテクニカルノートで紹介する中では、もっとも複雑なサンプルであり、OpenRico フレームワークを存分に使用しています。ちなみに OpenRico は Prototype フレームワークを使用しています。

OpenRico フレームワークの詳細については、次の URL を参照してください：
http://ww.openrico.org
アニメーションなど、グラフィカルな機能は興味をひくことでしょう。LiveGrid について説明した PDF も入手できます。

基本的な手順は次のとおりです：

- テーブルを特定できるように ID を使用して HTML テーブルを作成します。テーブルは <DIV>タグを使用して構造化しておきます。（ヘッダ、ボディなど）テーブルには実際のデータを入れられますが、空のままでも構いません。テーブルの行数は少なくても 1 行は実際に表示される数よりも多くなければなりません。
- Onload でテーブルの ID、行数、バックエンドの URL を渡して Rico.LiveGrid をコールします。Rico.LiveGrid はテーブルの再描画、サーバに対するリクエスト、内部的なキャッシュの構築、スクロールバーの作成、並び替えなど、ほとんどのことをやってくれます。

- バックエンドとなる 4D メソッドを作成します。

Rico.LiveGrid を使用すれば、スクロール毎にメソッドをコールすることができ、このときページ上の他のオブジェクトも更新することができます。Live Grid ページでは、この点を例証するためテーブルのヘッダに現在のスクロール位置を表示させています。このヘッダ自体、現在のスクロール位置と並び替え順を含んだブックマーク URL になっています。

バックエンドの 4D メソッドは Ajax_LiveGrid です。このメソッドも、パラメータを受け取って XML を構築するという点は、コンボボックスやタイプアヘッドのサンプルと同じです。

Live Grid に返される XML の内容を知りたければ、次の URL をブラウザに入力してください：
http://127.0.0.1:8080/4DAction/Ajax_LiveGrid?id=data_grid&page_size=70&offset=400&_
XML の中にフォーマット済の HTML コードが含まれており、これが HTML テーブルの中に挿入されます。

Ajax Resources

Ajax は一般的な技術を組み合わせたものですが、実際、Ajax に特化された内容の書籍はあまり多く出回っていないようです。Ajax について取り上げた Web サイトは多数、存在しますが、Ajax 自体が比較的新しく、すぐに URL が変わるかもしれません。そのため、ここでそのような URL を掲載するようなことはしません。

IBM 社の Web サイトには Ajax についての紹介があり、他の記事へのリンクもあります：
http://www-128.ibm.com/developerworks/web/library/wa-ajaxintro1.html

Wikipedia にもチュートリアルがあります：
http://en.wikipedia.org/wiki/Ajax_%28programming%29

今回のテクニカルノートでは Dave Crane 著の書籍 Ajax in Action(ISBN 1932394613)の第 9 章を参考にしました：
http://www.manning.com/books/crane/chapters

同じページから JavaScript のソースコードもダウンロードすることができます。

同書には JavaScript、CSS、DOM についての説明もあるので、これから Ajax を始めようという人にはとても役に立ちます。特定の機能を果たすための方法だけでなく、汎用性のあるオブ

ジェクトの作り方も説明されています。Web サイトでは、書籍の内容とサンプルを紹介した 18 分のムービーも見ることができます。

Conclusion

このテクニカルノートでは、Ajax の基本的な背景情報を解説し、4D で Ajax ベースのデザインを導入する際に必要な最初のステップを説明しました。実際に作動する Ajax のサンプルデータベースも用意しました。