Creating Client/Server Applications: Managing clients, Plug-ins, tips and tricks

By Chiheb NASR, QA Engineer, 4D S.A.

TN 06-21

Introduction

4D 2004 では、クライアントの自動アップグレードに対応したクライアント/サーバ用のカスタムアプリケーションジェネレータが導入されました。この新しいメカニズムは、デザインモードのアプリケーションビルドメニュー、あるいは XML プロジェクトファイルを作成してコマンドを実行することによって利用でき、後者の場合、より細かい制御ができます。このテクニカルノートでは、新しいアプリケーションジェネレータの仕組みと使用方法を解説しています。

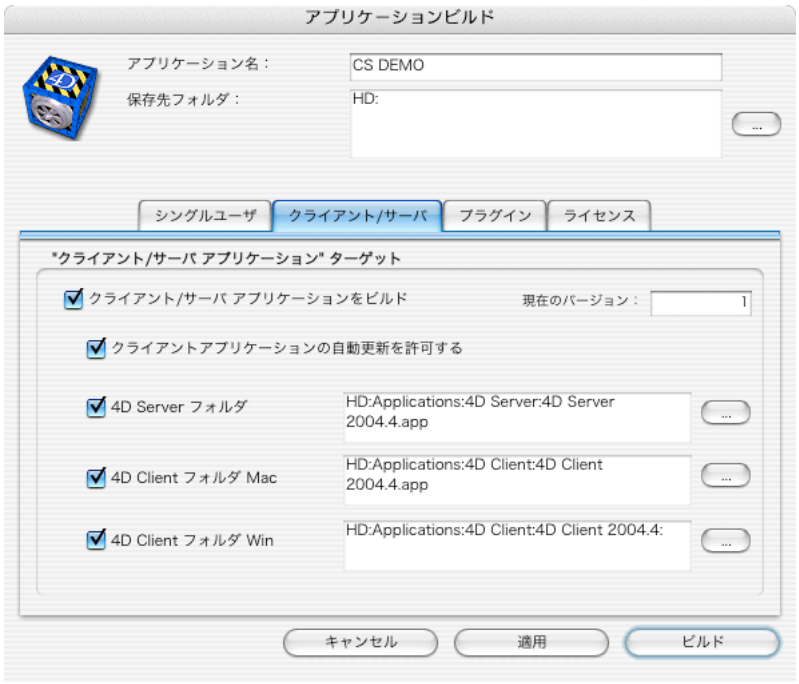
Prerequisites:

事前にアプリケーションビルドのドキュメントおよびクライアント自動アップグレードに関するテクニカルノート 04-32 の内容を確認することが勧められています：

http://tech.4djpn.jp/TechNote/329

Building multiplatform client applications

アプリケーションビルドダイアログの「クライアント/サーバ」タブには、クライアント自動アップグレードの際に配信される Macintosh および Windows のクライアントエンジンが置かれている場所を指定するフィールドがあります。Windows 用のパスは、4D Client アプリケーションが含まれているフォルダのパスを指定するのに対し、Macintosh 用のパスは 4D Client が含まれているフォルダではなく 4D Client パッケージ（4D Client.app）のパスを指定するという点に注意してください。サーバが Windows または Macintosh のどちらであっても 4D Client パスの指定方法は変わりません。例えば次の画面のように設定します：



ダブルクリックで起動可能なビルドアプリケーションのプラットフォームは、ビルドを実行するプラットフォームに依存しています。Windows では Windows 用のクライアント、Macintosh では Macintosh 用のクライアントがビルドできます。Windows および Macintosh それぞれのプラットフォームで有効な Developer Edition ライセンスがなければ、両プラットフォーム用のクライアントアプリケーションはビルドできない点に注意してください。

ダブルクリックで起動可能なクライアントアプリケーションをダイアログでビルドした場合、必要に応じて「EginedServer.xml」ファイルを作成することができます。このファイルをクライアントの「4D Extensions」フォルダにあらかじめ含めておけば、サーバの IP アドレスとポート番号を入力しなくても自動的にサーバへ接続できるようになるからです。アプリケーションビルドのダイアログには、サーバの IP アドレスとポート番号を指定するための場所がありませんが、「EginedServer.xml」ファイルを作成することにより、サーバへの接続を自動化することができます。下記はファイル内容の例です：>

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<Connection>
    <IPAdress>192.68.91.10,19815</IPAdress>
</Connection>
```

4D Japan 注記：Address のスペルは上記で問題ありません。
2004.4 バージョンでは、ポート番号のデリミターはカンマ、コロンのどちらでも動作します。

ポート番号は任意であり、省略した場合、デフォルトのポート番号 19813 が使用されます。

アプリケーションビルドダイアログで「EnginedServer.xml」が生成されないのは、ビルドを実行したマシンの IP アドレスとデフォルトポート番号で公開されているサーバについては、ファイルがなくても自動的に接続ができるからです。通常、ダイアログでビルドされたアプリケーションは、最初の起動で「EnginedServer.xml」ファイルを参照し、ファイルが存在しない場合、ビルドを実行したマシンの IP アドレスにあるサーバへの接続を試みます。接続に成功すれば「EnginedServer.xml」が自動的に生成され、失敗すれば手動でサーバを選択するための標準ダイアログが表示されます。接続に成功すれば、新しい設定で「EnginedServer.xml」が生成されます。XML プロジェクトファイルを使用して BUILD APPLICATION コマンドでアプリケーションをビルドするのであれば、専用の XML キーを使用して「EnginedServer.xml」ファイルの内容を指定することができます。必要な情報は、下記のようにプロジェクトファイル(BuilApp.XML)に含めておきます：

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<Preferences4D>
   <BuildApp>
    <CS>
     <BuildServerApplication>True</BuildServerApplication>
     <BuildCSUpgradeable>True</BuildCSUpgradeable>
     <CurrentVers>2</CurrentVers>
     <HardLink>Mabase_CS_Engine</HardLink>
     <IPAddress>192.68.10.10</IPAddress>
     <PortNumber>19815</ PortNumber >
    </CS>
   </BuildApp>
</Preferences4D>
```

「4D Client フォルダ Mac」、「4D Client フォルダ Win」オプションが選択されている場合、ビルドされたサーバアプリケーションのパッケージ/フォルダ内には「Upgrade4DClient」というフォルダが作成され、クライアント自動アップグレードで配信されるクライアントアプリケーションのアーカイブ(archive.mac および archive.win)が収められます。自動アップグレード用のファイルは、ビルドを実行するアプリケーションのプラットフォームには関係なく、両プラットフォーム用のものが作成されます。典型的な Upgrade4DClient フォルダは、次のようなファイル構成になっています。

Note:
ビルドされたクライアント/サーバカスタムアプリケーションは、コンパイルモード専用の実行環境です。したがってここまで考慮してきたメカニズムは、インタプリタモードやクライアント/サーバのデザインモードでは動作しません。

Using the XML keys to customize your Client/Server applications

XML キーを使用してコマンドでアプリケーションをビルドすれば、デザインモードのダイアログを使用するよりも細かいカスタマイズを実施することができます。

クライアント/サーバアプリケーションをビルドするには、所定の様式に合わせて XML プロジェクトファイルを作成し、そのファイルのパスを BUILD APPLICATION コマンドに渡す必要があります。ストラクチャがコンパイルされていなければ、はじめにコンパイルが自動的に実行されます。

このテクニカルノートでは、次の設定をするための XML キーを取り上げています：

- サーバの IP アドレス
- サーバのポート番号(19813 以外の場合)›
- サーバ、Windows および Macintosh 用クライアントのアイコン
- バージョン管理(接続を許可する最低/最新バージョンおよびカレントバージョン)
- プラグインの管理

Note:
バージョン 2004.0 と 2004.1 以降では、XML キーの名称が一部変更になっています。意味を明確にするため、「...FolderIsValid」という文字列はすべて「...IncludeIt」になりました。

/Preferences4D/BuildApp/SourcesFiles/RuntimeVL/RuntimeVLFolderIsValid
/Preferences4D/BuildApp/SourcesFiles/CS/ServerFolderIsValid
/Preferences4D/BuildApp/SourcesFiles/CS/ClientWinFolderIsValid
/Preferences4D/BuildApp/SourcesFiles/CS/ClientMacFolderIsValid

新しい名称は次のとおりです：

/Preferences4D/BuildApp/SourcesFiles/RuntimeVL/**RuntimeVLIncludeIt**

/Preferences4D/BuildApp/SourcesFiles/CS/**ServerIncludeIt**

/Preferences4D/BuildApp/SourcesFiles/CS/**ClientWinIncludeIt**

/Preferences4D/BuildApp/SourcesFiles/CS/**ClientMacIncludeIt**

Note:

プロジェクトファイルは名前を変更し、デフォルト (/Preferences/BuildApp)以外の場所に保存しておくことが勧められています。そうしておかないと、ビルドアプリケーションダイアログによってファイルが上書きされてしまうからです。

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<Preferences4D>
 <BuildApp>
  <BuildCompiled>False</BuildCompiled>
  <BuildApplicationSerialized>False</BuildApplicationSerialized>
  <BuildApplicationLight>False</BuildApplicationLight>
  <SourcesFiles>
   <RuntimeVL>
    <RuntimeVLIncludeIt>False</RuntimeVLIncludeIt>
   </RuntimeVL>
   <CS>
    <ServerIncludeIt>True</ServerIncludeIt>
    <ClientWinIncludeIt>True</ClientWinIncludeIt>
    <ClientMacIncludeIt>True</ClientMacIncludeIt>
    <ServerWinFolder>F:¥PC¥4D Server¥</ServerWinFolder>
    <ClientWinFolderToWin>F:¥PC¥4D Client¥</ClientWinFolderToWin>
    <ClientMacFolderToWin>F:¥Mac¥4D Client.app¥</ClientMacFolderToWin>
    <ServerIconWinPath>F:¥PC¥pc¥Server.ico</ServerIconWinPath>
    <ClientWinIconForWinPath>F:¥PC¥pc¥Client.ico</ClientWinIconForWinPath>
    <ClientMacIconForWinPath>F:¥PC¥Mac¥client.icns</ClientMacIconForWinPath>
  </CS>
 </SourcesFiles>
  <BuildApplicationName>Mabase_CS_Engine</BuildApplicationName>
  <BuildWinDestFolder>F:¥TEST_CS_ENGINE¥Application_Cible¥</BuildWinDestFolder>
  <CS>
   <BuildServerApplication>True</BuildServerApplication>
   <BuildCSUpgradeable>True</BuildCSUpgradeable>
   <RangeVersMin>1</RangeVersMin>
   <RangeVersMax>5</RangeVersMax>
   <CurrentVers>2</CurrentVers>
   <HardLink> Mabase_CS_Engine </HardLink>
   <IPAddress>192.68.10.10</IPAddress>
   <PortNumber>19815</ PortNumber >
  </CS>
   <ArrayExcludedPluginName>
```

```
      <ItemsCount>1</ItemsCount>
      <Item1>4D View</Item1>
    </ArrayExcludedPluginName>
    <ArrayExcludedPluginID>
      <ItemsCount>1</ItemsCount>
      <Item1>13000</Item1>
    </ArrayExcludedPluginID>
      <Licenses>
      <ArrayLicenseWin>
        <ItemsCount>1</ItemsCount>
        <Item1>E:¥Documents and Settings¥All
        Users¥ApplicationData¥4D¥Licenses¥4SDE80....html</Item1>
        </ArrayLicenseWin>
    <ArrayLicenseTarget>
        <ItemsCount>1</ItemsCount>
        <Item1>1</Item1>
    </ArrayLicenseTarget>
    </Licenses>
  </BuildApp>
</Preferences4D>
```

4D Client Update

ビルドアプリケーションの「クライアントアプリケーションの自動アップグレードを許可する」オプションが有効にされている場合、サーバのバージョンが更新された後の最初の接続でクライアントアプリケーションのアップグレードを実行するメカニズムが起動します。この動作は、XML キーの<BuildCSUpgradeable>で有効(True)または無効(False)にすることができます。

例えば、次の XML キーはオプションを有効にします：

<BuildCSUpgradeable>True</BuildCSUpgradeable>.

クライアント自動アップグレードは、4D Client と 4D Server のバージョンが一致することを保証し、各クライアントマシンを手作業でアップグレードする手間を省くものとなっています。自動アップグレード中、サーバは圧縮されたアップグレードをクライアントに送信します。送信が完了すると、クライアントは新しいアプリケーションを展開し、自らを削除してサーバへの接続を確立します。

トラブルを回避するためにも、自動アップグレードのメカニズムは正確に把握しておくことが望ましいといえるでしょう。アップグレードの前半は古いクライアントが実行し、その後を引き継ぐのはスクリプト(Windows では upgclnt.bat、Macintosh では upclnt.sh)です。いずれのスクリプトも新しい 4D Client の 4D Extensions フォルダの中に含められています。

これはクラアント自動アップグレードの流れを表にまとめたものです：

ステップ	説明	注記
1	旧クライアントは、接続に必要な最低/最高バージョン、カレントバージョンなどの情報をサーバに問い合わせます。	
2	旧クライアントは、バージョン情報に基づき、接続の可否を判断します。	
3	バージョンが対応すれば接続し、しなければ確認ダイアログを表示します。	
4	OK の場合、アーカイブのダウンロードを開始します。キャンセルの場合、4D Client を終了します。	アーカイブダウンロードの仕組みは、リソースをダウンロードする場合と同じです。
5	アーカイブは 4D Client と同じ階層に作成される一時フォルダ$temp4dclient に展開されます。	アーカイブを展開するのは旧クライアントの役目です。
	スクリプト upgclnt.bat または upgclnt.sh がカレントセッションのテンポラリフォルダにコピーされます。オリジナルのスクリプトは新しいクライアントの 4D Extensions フォルダに入っています。	コピー後、スクリプトは削除されます。
6	旧クライアントがスクリプトを起動します。	スクリプトには 4D Client の終了を検出するためのコードが含まれています。
7	旧クライアントが終了します。	
8	Macintosh のスクリプトは、カレントセッションで 4D Client パッケージが実行できるようにします。(chmod コマンド)	Mac OS のみ。
9	スクリプトは$temp4dclient フォルダから新しい 4D Client を取り出し、旧 4D Client と同じ場所に配置します。	
10	スクリプトは$temp4dclient フォルダを削除します。	
11	スクリプトは旧 4D Client を削除します。	Windows では、一時フォルダ$Trash4D に旧 4D Client を移動した後、フォルダをごみ箱に入れます。
12	スクリプトが新しい 4D Client を起動し、4D Client は自動的にサーバに接続します。	

How are the plug-ins generated in Client/Server?

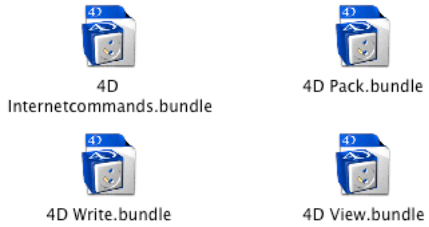
アプリケーションビルドダイアログでは、アプリケーションに含めたいプラグインをチェックボックスで選択することができます。デフォルトの設定では、スタンドアロンの開発環境にロードされているプラグインはすべてビルドに含まれます。デベロッパは、アプリケーションビルドダイアログの第 3 ページ、または XML キーの<ArrayExcludedPluginName>および<ArrayExcludedPluginID>を使用して、ビルドに含めるプラグインを明示的に指定しなければなりません。

クライアント/サーバアプリケーションにプラグインを組み込む場合、プラグインは次のいずれかの場所にあらかじめ格納しておけば良いということです：

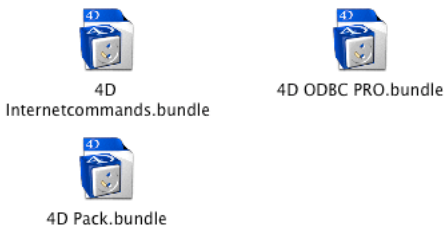
- コンパイルを実行する 4D の「Plugins」フォルダ
- データベースの「Plugins」、「Mac4DX」、「Win4DX」フォルダ
- 4D Server の「Plugins」フォルダ

Note:
同じプラグインの異なるバージョンが複数の場所に格納されている場合、一番優先されるのは 4D Server の「Plugins」フォルダです。

組み込みメカニズムの動作を例証するために仮のシナリオを考慮してみましょう。クライアント/サーバアプリケーションのビルドを実行するスタンドアロン版 4th Dimension の「Plugins」フォルダには、次のようなプラグイン(A)が置かれていたとします：

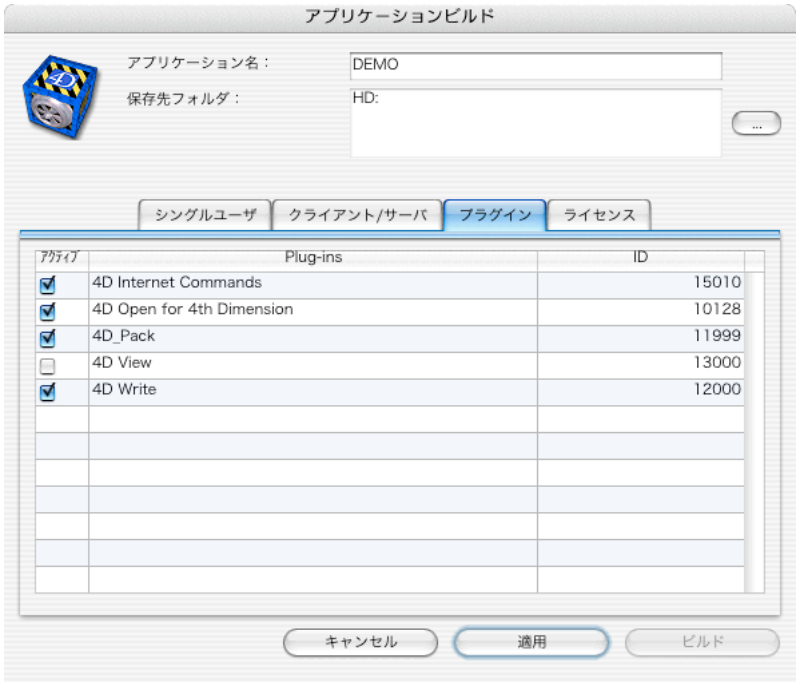


ビルドに使用する 4D Server の「Plugins」フォルダには、次のようなプラグイン(B)が置かれていたとします：

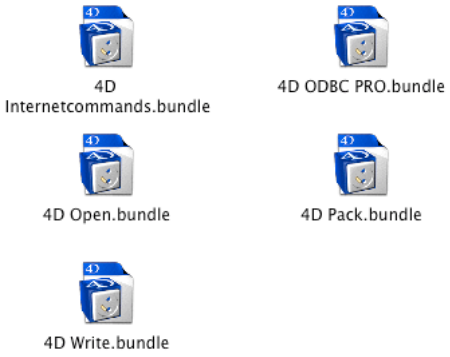


最後に、データベースストラクチャの「Plugins」フォルダには、4D Open プラグイン(C)が置かれていたとします：

ビルドを実行するスタンドアロン版の 4th Diemsion が読み込んでいるのは A と C だけなので、ダイアログは次のような構成になります：



当初の状態では 4D view のチェックボックスも有効にされていましたが、ここではあえてチェックボックスを無効にしてビルドに含まれないようにしました。この設定でアプリケーションビルドを実行すると、4D Server の「Plugins」フォルダは次のようになります：



明示的に除外した 4D View 以外は、A、B、C すべてのプラグインが含まれている点に注目してください。前述した優先順位が効いているので、Internet Commands と 4D pack は 4D Server の「Plugins」フォルダからコピーされました。この場合、プロジェクトファイルの XML キーは、4D View を除外したため次のようになっています：

```
<ArrayExcludedPluginName>
   <ItemsCount>1</ItemsCount>
   <Item1>4D View</Item1>
</ArrayExcludedPluginName>

<ArrayExcludedPluginID>
   <ItemsCount>1</ItemsCount>
   <Item1>13000</Item1>
</ArrayExcludedPluginID>
```

Note:
名称を変更したとしても、4D Server 起動時にプラグイン ID に基づいて衝突が検出されます。そのような場合、同じプラグインが複数の場所にインストールされているというメッセージを表示した後、4D Server は終了します。

Conclusion

このテクニカルノートでは、クライアント/サーバアプリケーションのビルドに関わる注意点、ビルド時におけるプラグインの処理、およびクライアント自動アップグレードのメカニズムについて取り上げました。