Technical Note 05-26
サブフォーム vs リストボックス

By Kent Wilbur, Manager Information Systems, 4D, Inc.
Technical Note 05-26

（原題: Using ListBoxes as Subforms）

概要

4D 2004 では、リストフォームが完全に作り直され、出力フォーム、サブフォーム、MODIFY SELECTION/DISPLAY SELECTION で表示されるフォームのすべてが改良されました。加えて、グループ化されたスクロールエリアを強化したものともいえる新しいオブジェクト/リストボックスが使用できるようになり、表現力が増し加わっています。中には、サブフォームとリストボックスのどちらを採用するかで迷っているデベロッパもいると聞きます。今回の Tech Note で、その質問に対する答えは出ないかもしれませんが、サブフォームの代替案としてリストボックスを使用する方法をご紹介したいと思います。

サブフォームの長所

従来から存在するオブジェクトなので、ユーザが慣れています。ダブルクリックで詳細フォームに移行するなど、基本的な動作にはコーディングがほとんど要りません。加えてバージョン 2004 ではヘッダにボタンが配置できるようになったのも魅力です。

リストボックスの長所

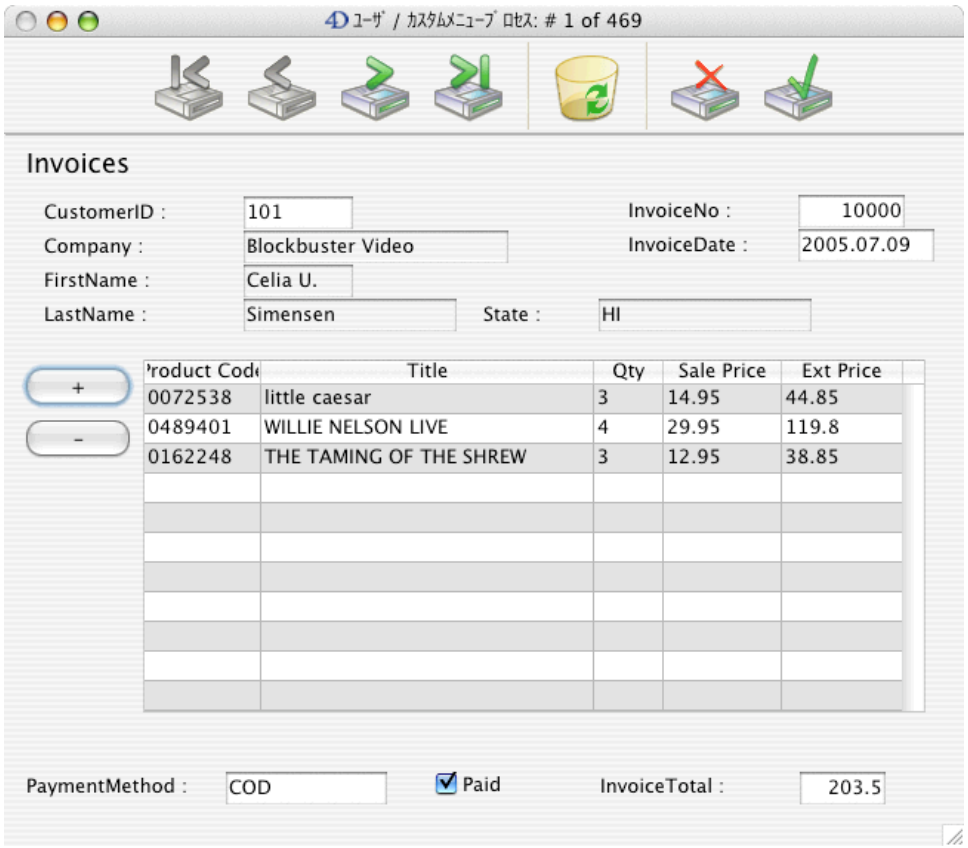
サブフォームは、クライアント/サーバ環境で使用するとネットワーク通信に負荷がかかります。これは 4D Client がレコード（サブレコードではなく、リレートテーブルのレコード）をひとつずつロードするためで、別のレコードを表示するたびに、前のレコードをサーバに登録する動作が発生しているのが理由です。さらにキャンセルの可能性を考えるとトランザクションの開始が必須であり、加えられた変更を一時保管しなくてはならないので、サーバの負担が増えることになります。

私がリストボックスに分があると考えているのは特にこの点です。はじめに表示するとき、リストボックス、サブフォームともまとめてデータがクライアントに送られますが、サブフォームとは違い、配列であるリストボックスは、更新を確定するまではネットワーク通信が発生しません。ARRAY TO SELECTION は 4D Server 用に最適化されているので、このアプローチはたいへん実用的です。

トランザクションが不要なこと加えて、デザイン面でも、リストボックスには奇数行背景色、スプリッタ要らずのカラムリサイズ、ドラッグ&ドロップなど、ノンプログラミングで利用することのできる機能が用意されており、オブジェクトの数を最小限に抑えることができるのが魅力です。

サンプルデータベース

入力画面では、サブフォームの代わりにリストボックスを採用しました。実際にはトランザクションを使用していませんが、それでもトランザクションのあるサブフォームに相当する動作を実現しています。



見た目は標準的なサブフォームに似ていますが、特別なコーディングがないにも関わらず、カラムの並び替えやドラッグ&ドロップによるカラム移動が可能です。本論から逸れるような機能はなるべく付加しないつもりでしたが、ユーザによる行の並び替えがレコードを保存した後も残るような工夫を施しました。サブフォームではまったく不可能な動作です。

各部の説明

複数のオブジェクトがまとまって、リストボックスというオブジェクトを構成しています。それぞれの構成部分には特有のプロパティがあり、個別に値を設定することができます。

A リストボックス

リストボックスの本体であるブール値の配列です。列数は 6 ですが、ひとつは非表示です。

複数選択不可、幅を固定、行の移動可、列の並び替え可、加えて On After Sort、On Row Moved フォームイベントで実行されるオブジェクトメソッドが設定されています。

B ヘッダ

ヘッダのプロパティは列ごとに設定することができますが、オブジェクトメソッドというものはありません。今回は特別な設定をしませんでした。

C 列

リストボックスの列はそれぞれが 4D の配列です。オブジェクトメソッドは列ごとに設定することができます。コマンドで列を追加/削除することは可能ですが、コマンドでオブジェクトメソッドを関連づけられるわけではないので、そのような列は個別に作成します。

aLInvioceNo は、非表示に設定されています。atProductCode および aiQty だけが入力可であり、かつオブジェクトメソッドを関連づけられています。利用するフォームイベントは両方とも On Data Change です。

プロパティにはフォント、カラーなどの項目がありますが、*セル単位でプロパティを設定することはできない*、という点に留意してください。リストボックスは Area List Pro、4D View に取って代わるものではありません。しかしながら、行単位での設定は可能です。これにはリストボックスのプロパティである、行スタイルの配列/行のフォントカラーの配列/行の背景色の配列を使用します。

フォームメソッド

入力に関わる処理のほとんどはフォームメソッドに記述されています。リストボックスに関係があるフォームイベントは、On Load、On Unload および On Validate だけです。

On Load では、新規レコードの場合、メソッド GEN_LNextSequence でレコード番号を取得しています。Sequence number の代替案であるこのメソッドは、私が 1998 年 12 月の Tech Note で取り上げましたので、興味がある方はご覧いただければと思います。今回はトランザクションを使用していないので、Sequence number レコードをロックする必要はありません。

レコードが他のプロセスによってロックされている場合は、更新できない項目のボタンは DISABLE にされます。必然的に On Validate イベントは発生しないので、レコードが不意に更新される心配はありません。

最後にリレートレコードがリストボックスの各配列にロードされ、保存された表示位置を示すフィールド[LineItems]Position でソートされます。

On Unload では、キャンセルの判別をしてからリストボックスの変数をクリアしています。

On Validate では、ユーザが実際に値を入力したかを調べ、並び順のローカル配列を作成

しています。並び順は、リストボックスの非表示列としても良かったのですが、行が追加/削除されるたびの処理が煩雑なうえ、結局のところ確定したときの順番だけが問題なので、このような方法を取りました。

データは ARRAY TO SELECTION コマンドで保存されます。新規レコードの場合は、フラグ fNewInvoice がクリアされ、キーフィールドである Invoice number がコミットされます。

リストボックスのオブジェクトメソッド

これは並び順の保存という機能を採用したがために必要になった処理です。レコードが保存される条件は On Validate が発生したとき、つまりレコードが更新されて ACCEPT が発行された場合ですが、単に並び順をいじっただけでは On Validate が起こりません。そこで On Row Moved および On After Sort で [Invoices]Paid にカレントの値を代入することによって強制的に On Validate を起こしています。このフィールドを選んだのは、属性がインデックスなしのブール値で、データベースに及ぼす影響が最小限だからです。

標準アクション（自動アクション）の動作に近づけるため、削除ボタンは行が選択されない限り DISABLE にしました。この判別は On Selection Change で行なっています。

行の追加と削除

リストボックスを使用する以上、標準アクション（自動アクション）ボタンは利用できないので、追加/削除ボタンを個別に作成しなくてはなりません。これには良い面もあって、自動ボタンの「サブフォームが選択されていないときは DISABLE」という動作（しばしばエンドユーザを当惑させている）に縛られない追加/削除ボタンを用意することが可能です。

DELETE LISTBOX ROW コマンドは、非表示の列も含め、各配列から特定の要素をまとめて削除します。グループ化された配列の場合、これにはループ処理が必要でした。

削除後には、若干の後始末が必要です。合計値を再計算するために Invoice_CalcTotals メソッドをコールし、選択行がなくなったので削除ボタンを DISABLE にしています。

行を追加するときには、リストボックスを構成する各配列に要素を追加します。キーフィールドである aLInvoiceNo は、On Validate で ARRAY TO SELECTION が実行されるときに必ず値が必要なので、ここで値を代入しています。
（「サブフォームにリレート値を自動代入する」に相当）

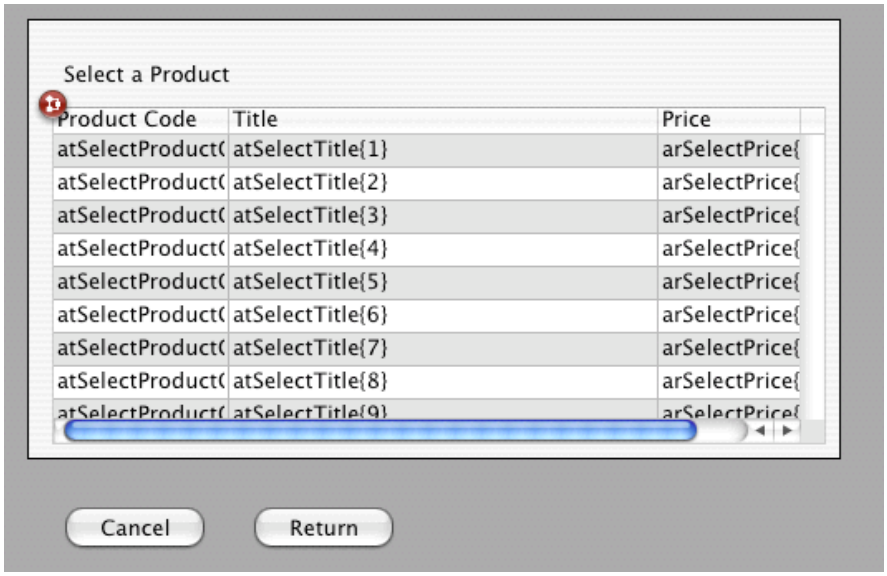
次に、新しい行が表示範囲外である可能性もあるので念のためスクロールし、もっと重要なこととして、すぐデータ入力が始められるように EDIT ITEM を実行しています。

列のオブジェクトメソッド

リストボックスを使用する以上、自動ワイルドカードが利用できないので、同等のものを自作しなくてはなりません。メソッドでは、入力確定時に@を追加してクエリを実行しています。結果のセレクションが 1 レコードならば、それが入力データとなります。0 のときはエラーメッセージが表示されます。問題は、セレクションが複数レコードの場合です。通常でしたら、ここは選択リストが表示される場面なので、同様のものを自作することにします。もちろん、ウィンドウの外観や位置は自由にカスタマイズすることができます。

項目は SELECTION TO ARRAY で配列に納めます。ウィンドウを開いているのはメサオッド WIN_LNewWindow です。Open form window の代替案であるこのメソッドは、私が 2004 年 2 月の Tech Note で取り上げましたので、興味がある方はご覧いただければと思います。

ちなみに、カスタム選択リストのフォームにもリストボックスが使用されています。



ここには非表示の列も、入力可のアイテムもありません。ダブルクリック、Return、enter で項目選択、Esc キーでキャンセルの処理をするだけの表示用フォームです。コードはすべてリストボックスのオブジェクトメソッドにまとめられています。選択時には ACCEPT が発行され、実際の処理は親フォームのメソッドが引き継いでいます。

リストボックスの制限

ひとつ重要な制限事項があります。リストボックスは画面表示用に設計されたオブジェクトであり、印刷することはできません。印刷用にはフォームを別に用意し、サブフォームあるいはグループ化された配列を利用する必要があります。