



Technical Note 04-45

ユーザ定義フォーム

By Sati Hillyer, 4D Evangelist
Technical Note 04-45

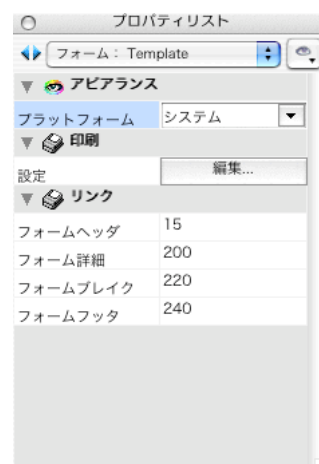
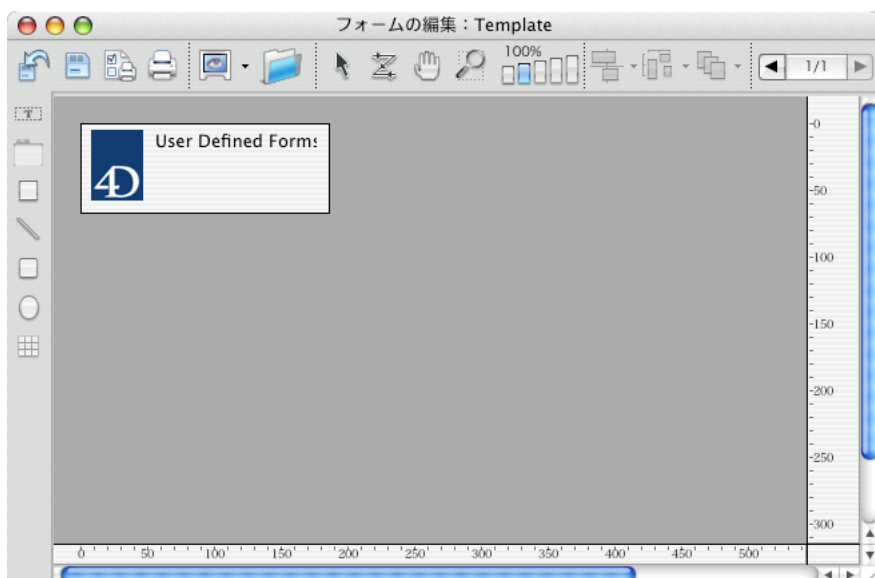
(原題: User Defined Forms)

概要

4D 2004 の新しい機能であるユーザ定義フォームによって、複数のクライアントに対して自ら調整することのできるアプリケーションを配布したり、些細な変更点はユーザに任せたりすることが可能になりました。この機能はコンパイルされたアプリケーションであっても利用することができます。

説明

4D 2004 で作成されたフォームのプロパティには「ユーザによる更新可」という項目が存在します。これが有効にされたフォームは、ソースフォーム、あるいはデベロッパフォームと呼ばれ、ユーザ定義フォームの原型となります。カスタムモードで EDIT FORM コマンドを実行されると、ユーザは簡易版のフォームエディタでこれを編集することができます。編集されたフォームは保存され、本来のフォームの代わりに使用されます。CREATE USER FORM コマンドが実行されると、このフォームをテンプレートとして幾つでもユーザ定義フォームを追加することができます。それぞれのフォームには固有の名称が与えられ、EDIT FORM コマンドで編集することが可能です。INPUT FORM や OUTPUT FORM でユーザフォームを指定することもできます。



簡易版フォームエディタでは、ユーザが追加したオブジェクトについては自由に編集コマンド(カット、ペースト、コピーなど)を使用することができますが、オリジナルのオブジェクトは削除ができないようになっています。ユーザが変更することのできるプロパティはデベロッパによって許可されたものだけです。

変更を取り消したいときには、左上のボタンをクリックすることによってデザインモードで定義された本来のフォームに戻ることができます。EDIT FORM コマンドは、実行時にオブジェクトライブラリをひとつ任意のパラメータとしてとることができるので、ライブラリを経由してオブジェクトをユーザに提供することができます。これらのオブジェクトは、標準アクションや画像などのプロパティも継承しますが、オブジェクトメソッドのあるものは使用することができません。

ユーザフォームの仕組み

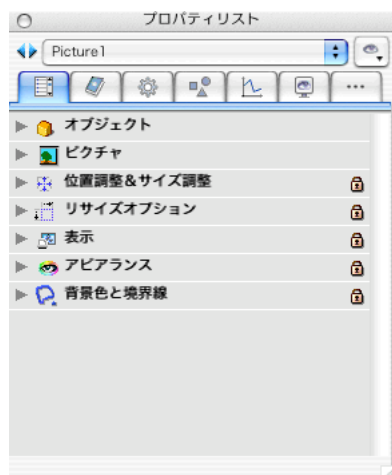
ユーザフォームは元のフォームを複製したものをユーザストラクチャファイルに保存することによって動作しています。.4DA という拡張子をもつこのファイルはバックアップの対象に含めるべき重要なファイルとなります。



ひとたびユーザによる更新可に設定されたフォームは、デザインモードで右上に南京錠のアイコンが表示されてロックされます。



ユーザによる更新可のフォームは、プロパティごとに更新を許可するかどうかを示す南京錠のアイコンが存在します。



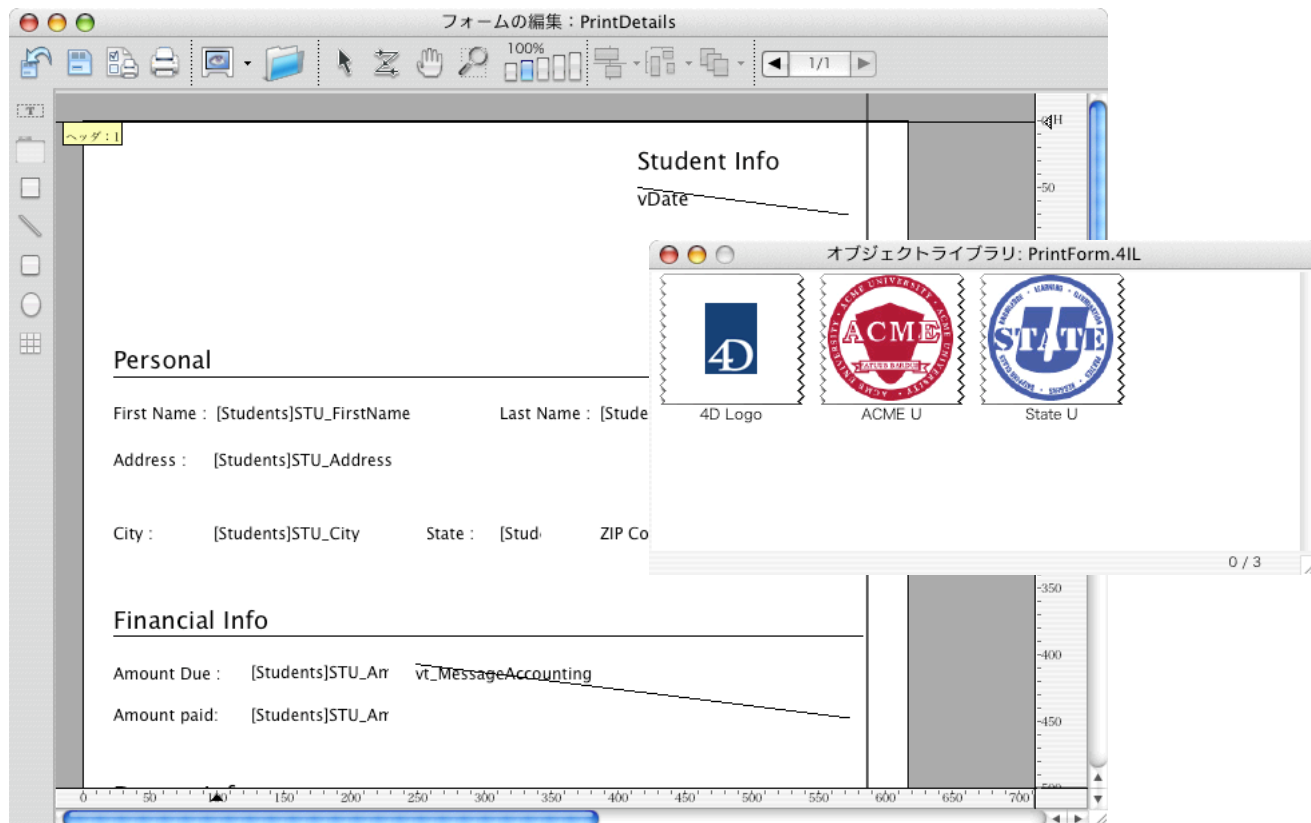
デベロッパがユーザによる更新可のフォームを保存すると、内部的なバージョン情報が加算されてストラクチャファイルに保存されます。EDIT FORM でユーザが編集するフォームは、このバージョン情報を一緒にユーザストラクチャファイルに保存しています。フォームをロードするとき、4D は両方のバージョン情報を比較して新しいほうのフォームを採用するので、デザインモードで元のフォームが後から変更された場合には、ユーザフォームは使用されなくなります。フォームが削除されるわけではないので、ユーザストラクチャファイルのサイズは変わりません。したがって両者のバージョン情報が一致するバックアップにおいては、ユーザフォームを使用することができます。いずれにしてもユーザフォームを残しておきたい場合は、デザインモードで元のフォームを変更しないというのが鉄則です。

クライアント/サーバの場合、.4DA ファイルはサーバ側でストラクチャと同じ階層に置かれます。オブジェクトライブラリは、クライアント側に用意することになります。簡易版フォームエディタはプロセスにとって最初のウィンドウである場合にのみ起動します。したがって、通常は New process でこのコマンドを実行することになります。オブジェクトライブラリは開かれている間はロックされているので、同時に別の簡易版フォームエディタで開くことはできません。

サンプルデータベース

Scenario1

大学の学生名簿を想定しています。デフォルトの印刷フォームには学校名や紋章が含まれていない代わりに、ユーザ定義フォームとして編集することができるようになっています。画像は、オブジェクトライブラリとして提供されます。



Scenario2

学部ごとに学生管理のインタフェースが微妙に異なる場合を想定しています。フォームのページ 2 以降は白紙でなので、ライブラリから適当なものを貼付けることができます。

フォームの編集: Accounting

100%

1/3

Student

at_Choices

Student Information

Student ID : [Students] Department : [Students]STU_SchoolDepartment

First Name : [Students]STU_FirstName Last Name : [Students]STU_LastName

Address : [Students]STU_Address

City : [Students]STU_City State : [Stud] ZIP Code : [Students]STU_ZI

< > Print Cancel Save

フォームの編集: Accounting

100%

2/3

Student

at_Choices

オブジェクトライブラリ: student.4il

Financial Financial Comments Degree Information Degree Comments

0 / 4

< > Print Cancel Save

Scenario3

デベロッパが元のフォームを変更してしまうと、すべてのユーザフォームが使用できなくなるという点については既に触れました。継承フォームを使用することによってこれを回避することができます。

はじめにデベロッパは親のフォームを作成し、それを継承する子フォームをユーザフォームの原型として定義します。デベロッパが継承フォームを編集しても、ユーザフォームの原型は未修整なので、ユーザフォーム群は引き続き有効なバージョンとみなされます。

応用として、親フォームにもデベロッパだけが編集できるユーザフォームを用意すれば、再コンパイルをすることなく、各ユーザフォームに対して活かされるフォームの編集を施すことも考えられます。

Scenario4

ユーザフォームに対応しているコマンドは他にあります。たとえばラベルエディタの利用するフォームには、ユーザフォームが含まれています。

- ADD RECORD
- MODIFY RECORD
- DISPLAY RECORD
- MODIFY SELECTION
- DISPLAY SELECTION
- PRINT SELECTION
- PRINT LABEL
- PRINT RECORD
- QUERY BY EXAMPLE
- DIALOG

