4D v14 R5 - アップグレード

4Dの継続的なデリバリープログラムの最新バージョンとなる、4D v14 R5へようこそ。このマニュアルでは、この新しいリリースの中に含まれる全ての新機能や実装について説明しています。

ランゲージ

- Get database measures
- Get database parameter と SET DATABASE PARAMETER
- OBJECT GET COORDINATES
- Toolbar form windows
- Clickcount
- LISTBOX GET CELL COORDINATES

Get database measures

Get database measures {(options)} -> 戻り値

説明

Get database measures コマンドは4D v14R5 以降、インデックスの使用、クエリ、並べ替えに関する情報を返すようになりました。

(以下の様な)特定の状況において、"table" オブジェクトと"indexes" オブジェクト内に新しい属性が提供されます:

- "tables" の新しい属性: "fields" と "queries"
- "indexes" の新しい属性: "queryCount"、"sortCount"、"insertKeyCount"、"deleteKeyCount"
  これらの情報の項目を使用することにより、データベースの内部の設定を分析し、最適化することができるようになります。例えば、インデックスにおいて、"insertKeyCount" 項目が大量にあり、"queryCount" と "sortCount" 項目が少ない場合、インデックスが不要である可能性を示唆しています。

これに加えて、*options* 引数は"path"プロパティとして文字列の配列を受け取れるようになりました。

テーブルオブジェクトの新しい属性

"tables.*table*" オブジェクトは、テーブル で実行されている実際のクエリと並び替えに応じて、二つまで新しいプロパティを含めることが出来るようになりました:

- "fields"
- "queries"

fields

"fields" はクエリと並び替えに使用されたフィールド数と同じだけの数のフィールド名属性(これもまたサブオブジェクトです)を内包するオブジェクトです。

それぞれのフィールド名オブジェクトには以下のものが含まれます:

- このフィールドを使用してクエリが実行されたことがある場合には、"queryCount" オブジェクト(*options* 引数に応じて履歴が表示または非表示)が含まれます。
- このフィールドを使用して並び替えが実行されたことがある場合には、"sortCount" オブジェクト(*options* 引数に応じて履歴が表示または非表示)が含まれます。

この属性はインデックスの使用には基づいていません。クエリと並び替えであれば全てカウントされます。

例:

データベース起動の瞬間から、CompID、NameとFirstNameフィールドを使用して複数のクエリと並び替えが実行された場合を考えます。返されたオブジェクトには以下の"fields"サブオブジェクトが含まれます(*options* 引数にはパスは含まれますが履歴は含まれません):

```
{     "DB": {         "tables": {             "Employees": {                 "fields": {
"CompID": {                         "queryCount": {                             "value": 3
    }                     },                      "Name": {                         "queryCount": {
                "value": 1                          },                          "sortCount": {
        "value": 3                      }                   },                          "FirstName": {
                "sortCount": {                              "value": 2                          }
    } (...)
```

注: "fields" 属性は、クエリまたは並び替えがテーブル上で実行された場合にのみ作成されます。それ以外の場合にはこの属性は現れません。

queries

"queries" は *table* で実行されたそれぞれのクエリの詳細を含んだオブジェクトの配列です。それぞれの配列の要素は3つの属性を含みます:

- "queryStatement" (文字列): フィールド名を含むクエリ文字列。ただし検索した値は含まれません。
  例:"(Companies.PK_ID != ?)"
- "queryCount" (オブジェクト):
  - "value" (数値): クエリ宣言が実行された回数(検索した値は関係ありません)。
  - "history" (オブジェクト配列) (*options* 引数でリクエストされた場合に限る): "value" と "time" の標準の履歴プロパティ
- "duration" (オブジェクト) (timeの "value" が >0だった場合)
  - "value" (数値): ミリ秒数
  - "history" (オブジェクト配列) (*options* 引数でリクエストされた場合に限る): "value" と "time" の標準の履歴プロパティ

注: "queries" 属性は*table*において少なくとも一つのクエリが実行されいた場合に作成されます。

例:

データベースが起動された瞬間から、Employeesテーブル上ではクエリが一つだけ実行された場合を考えます(*options* 引数にはパスと履歴が含まれます):

```
{     "DB": {          "tables": {              "Employees": {                  "queries": [                      {
                        "queryStatement": "(Employees.Name == ?)",                          "queryCount": {
                        "value": 1,                                 "history": [                                    {
                                    "value": 1,                                             "time": -2022
                    }                               ]                                 },
"duration": {                               "value": 2,                                 "history": [
                {                                       "value": 2,                                             "time": -
2022                                }                                   ]                                   }
    }, (...)
```

indexesオブジェクトの新しい属性

"indexes.*table*.*field"* オブジェクトはデータベースが起動してからのインデックスの使用によって、最大で4つの新しいサブオブジェクトを含むことができるようになりました:

- "insertKeyCount": 新しいインデックスキーが挿入された場合に更新されます。
- "deleteKeyCount": インデックスキーが削除されたときに更新されます。
- "queryCount": インデックスがクエリで使用されたときに更新されます。
- "sortCount": インデックスが並び替えで使用されたときに更新されます。

これらのキーは、それぞれに対応するオペレーションが、データベースの起動以降どこかの時点で実行されていた場合にのみ存在します。

例:

データベースが起動した瞬間から、二つのレコードが作成され、[Companies] テーブルにおいて16レコードが削除されました。このテーブルでは"name"フィールドがインデックスされています。またこのテーブルは、そのフィールドを使用してクエリと並び替えが行われました。返されるオブジェクトは以下のようになります:

```
{     "DB": {          "indexes": {              "Companies": {                  "Name":                   (...)
                "queryCount": {                             "value": 41                         },
"sortCount": {                          "value": 3                      },                      "insertKeyCount": {
                    "value": 2                      },                  "deleteKeyCount": {
            "value": 16                     }             (...)
```

options 引数内のPath配列

options オブジェクトに対し、**文字列配列**を"path"プロパティに渡す事ができるようになりました。このプロパティには、取得したい特定のプロパティへのフルパスの配列を渡します。例えば、["DB.tables.Employee.records.diskWriteBytes", "DB.tables.Employee.records.diskReadCount","DB.dataSegment1.diskReadBytes"]といった配列です。この場合、それらに対応する値のみが"DB"オブジェクトに返されます。

Get database parameter と SET DATABASE PARAMETER

Get database parameter ({aTable ;} selector {; stringValue}) -> 戻り値

SET DATABASE PARAMETER ({aTable ;} selector ; value)

説明

Get database parameter と **SET DATABASE PARAMETER** コマンドにおいて、新しい*selector* 引数が使えるようになりました:

定数	型	値
Use legacy network layer	倍長整数	87

- **スコープ**: ローカルモードの4Dおよび 4D Server
- **2セッション間で設定を保持**: Yes
- **説明**: クライアント/サーバー通信における旧式ネットワークレイヤーの状態を設定、または取得します。旧式ネットワークレイヤーは4D v14 R5以降廃止予定となり、*ServerNet* ネットワークレイヤーへと積極的に置き換えられるべきです。*ServerNet* は今後の4Dリリースにおいて、将来のネットワークの進化の利益を享受できるようにするために必須要項となる予定です。互換性の観点から、既存のアプリケーションのスムーズな移行のために旧式ネットワークレイヤーは引き続きサポートはされます(v14 R5以前の4Dから変換されたアプリケーションにおいてはデフォルトで使用されます)。この引数に1を渡すと、クライアント/サーバー通信において旧式ネットワークレイヤーが使用されます(同時に*ServerNet* は無効化されます)。0を渡すと、旧式ネットワークレイヤーが無効化されます(同時に*ServerNet* が使用されます)。
  このプロパティは、データベース設定の"互換性"ページの"旧式ネットワークレイヤーを使用する"のオプションを使用することによっても設定することができます(**新しいServerNetネットワークレイヤー** を参照して下さい)。この章では、以降の戦略についてのディスカッションを見る事もできます。**新しいServerNetネットワークレイヤー**をなるべく早く有効化することが推奨されます。
  この引数の設定が使用されるためには、アプリケーションの再起動が必要です。この引数は*ServetNet のみ*をサポートするOS X用の4D Server v14 R5 64-bit では使用できません(値は常に0を返します)。
- **取り得る値**: 0 または 1 (0 = 旧式ネットワークレイヤーを使用しない、1 = 旧式ネットワークレイヤーを使用する)
- **デフォルト値**: 4D v14 R5 以降で作成されたデータベースにおいては0、4D v14 R4以前から変換されたデータベースにおいては1。

OBJECT GET COORDINATES

OBJECT GET COORDINATES ({* ;} object ; left ; top ; right ; bottom)

説明

テーマ: オブジェクト(フォーム)

OBJECT GET COORDINATES コマンドは特定の親のリストボックスオブジェクトだけではなく、リストボックスのパーツ(カラム、ヘッダーまたはフッター)の座標を返す事ができるようになりました。

以前のリリースでは、このコマンドは適用されたリストボックスのパーツに関わらず、常に親のリストボックスの座標を返していました。4D v14 R5以降、*object* 引数で参照しているオブジェクトがリストボックスヘッダー、カラム、またはフッターサブオブジェクトの場合、返される座標は指定されたそれらのサブオブジェクトのものとなります。

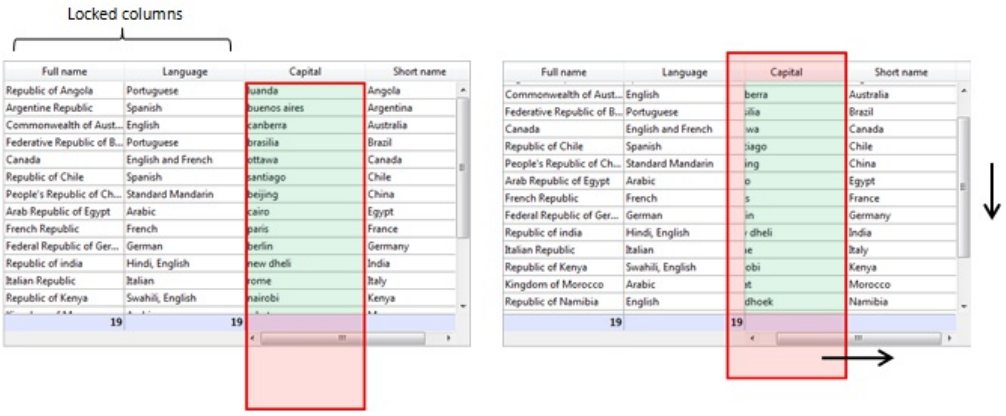
この新機能を使用して、例えばリストボックスのヘッダーにマウスオーバーしたときにそこに小さなアイコンを表示させ、ユーザーがそれをクリックするとコンテキストメニューが表示される、といったようなこともできるようになります。

互換性に関する注意: 既存のアプリケーションにおいてこのコマンドをリストボックスヘッダー、フッター、またはカラムに対して使用している場合、4D v14 R5以降のバージョンに変換した際には返される座標が異なります。リストボックス全体の座標を取得したい場合には、コードを修正してコマンドがサブオブジェクトに対してではなく、リストボックスオブジェクト自身に対して呼び出されているようにしなければなりません。

統一性のために、対象のオブジェクトがリストボックスサブオブジェクトであってもリストボックスオブジェクトであっても、使用される参照フレームは同じとなっています。原点はそのオブジェクトを含むフォームの左上端となっています。リストボックスサブオブジェクトに対しては、返される値は理論値となり、クリッピングが起きない限りリストボックスのスクロールの状態を考慮に入れます。結果として、サブオブジェクトはその座標において(一部または全部が)見えないこともあり、これらの座標がフォームの端より外側にあることもありえます(または、負の値を返す事も有り得ます)。サブオブジェクト(の一部または全部が)見えているかどうかを調べるためには、リストボックスの座標と比較する必要があります。その際、以下のルールが適用されます:

- 全てのサブオブジェクトは親のリストボックスの座標(**OBJECT GET COORDINATES** がリストボックスに対して返す値)に沿って切り取られます。
- ヘッダーとフッターのサブオブジェクトはカラムの中身の上に表示されます。カラムの座標とヘッダーまたはフッターの座標が交錯した場合、その重なった部分においてはカラムは表示されません。
- ロックされたカラムの要素はスクロール可能なカラムの要素の上に表示されます。スクロール可能なカラムの要素がロックされたカラムの座標と交錯した場合、スクロール可能なカラムの要素はこの重なった部分では表示されません。

具体例として、以下の画像において、*Capital* のカラムに注目して下さい(このカラムを囲っている赤い四角がこのカラムの座標を表しています):

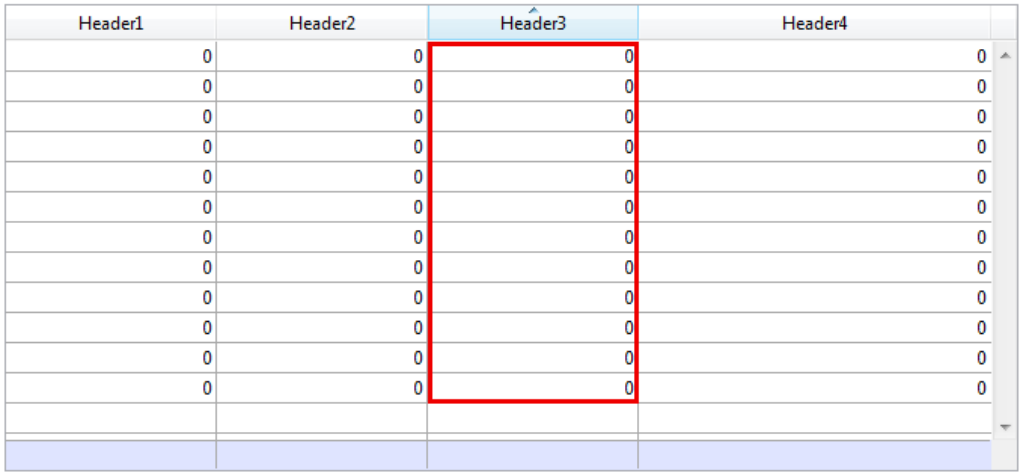


1枚目の画像にあるように、カラムはリストボックスより大きいので、カラムの座標はリストボックス(とフッター)の下限より下まで広がっています。2枚目の画像では、リストボックスはスクロールされており、その結果このカラムは*Language* カラムとヘッダーエリアの"下"へと移動します。どちらの場合においても、実際に表示されている部分(緑のエリア)を計算する

ためには、赤いエリアを計算に入れる必要があります。

例題

インターフェースのために、クリックされたエリアを赤い四角で囲いたい場合を考えます:



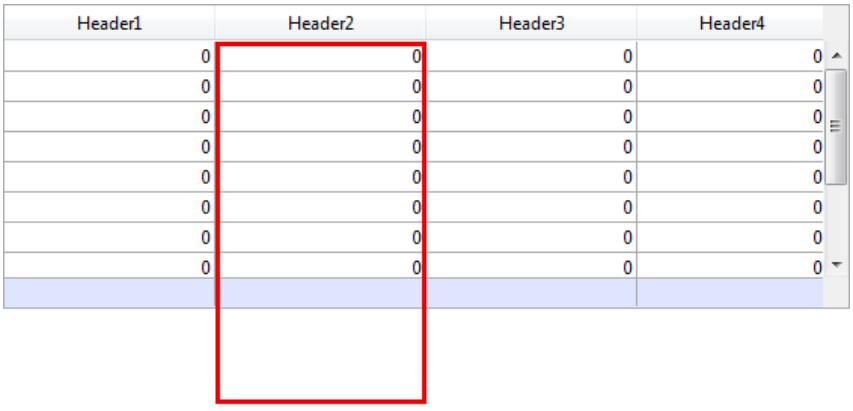
Header1	Header2	Header3	Header4
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0

リストボックスのオブジェクトメソッドにおいて、以下の様に記述します:

```
OBJECT SET VISIBLE(*;"rectangleInfo";False) //赤い四角を初期化
$ptr:=OBJECT Get pointer(Object current)
OBJECT GET COORDINATES($ptr->;$x1;$y1;$x2;$y2)
OBJECT SET VISIBLE(*;"RedRect";True)
OBJECT SET COORDINATES(*;"RedRect";$x1;$y1;$x2;$y2)

OBJECT GET COORDINATES(*;"LB1";$lbx1;$lby1;$lbx2;$lby2)
If($lby1>$y1)|($lby2<$y2) // クリックされたエリアがリストボックスの外側にある場合、
   OBJECT SET VISIBLE(*;"Alert";True) //警告を表示します
Else
   OBJECT SET VISIBLE(*;"Alert";False)
End if
```

メソッドは座標の理論値を返します。リストボックスがリサイズされた場合、どの部分が表示されているのかを調べるためにクリッピングを再計算が必要になる場合もあります:



Header1	Header2	Header3	Header4
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0

Toolbar form windows

4D v14 R5にはデベロッパがカスタムのツールバーをデザインし、管理するのを手助けするための様々な新機能が実装されています。ツールバーとは、自身の位置とサイズを管理する特定のプロパティを持ったウィンドウの事です。

以下のコマンドはツールバーの作成と管理をサポートします:

- **Open form window**: 新しい定数タイプをToolbar form window 受け付けます。
- **Tool bar height**: カスタムのツールバーの高さを返します。
- **HIDE TOOL BAR** と **SHOW TOOL BAR**: 以前は廃止予定でしたが、これらのコマンドはカスタムのツールバーを管理するために再度有効化されました。

Open form window

テーマ: ウィンドウ

```
Open form window ( {aTable ;} formName {; type {; hPos {; vPos {; *}}}} ) -> 戻り値
```

Open form window コマンドは、**Toolbar** タイプのフォームウィンドウを作成できるようになりました。

"Open Form Window" テーマに新しい定数が追加され、*type* 引数に対して使用できるようになりました:

定数	型	値
Toolbar form window	倍長整数	35

Toolbar form window 定数が渡された場合、ウィンドウはツールバーの位置、サイズ、グラフィカルプロパティで作成されます。具体的には以下の様な特性を持ちます:

- ウィンドウはメニューバー直下に常に表示されます。
- ウィンドウの横幅(水平方向の幅)は、デスクトップ上(OS X)または4Dのメインウィンドウ内(Windows)で使用可能な横幅を全て埋めるように調整されます。ウィンドウの縦方向(垂直方向の幅)は、他の全てのフォームウィンドウタイプ同様、フォームプロパティに基づいて決められます。
- ウィンドウには境界線はなく、手動で移動・リサイズはできません。表示されている間は*hPos*、*vPos* そして * 引数は無視されます。

二つの異なるツールバーウィンドウを同時に作成することはできません。ツールバー型のウィンドウが既に存在している状態で、**Open form window** コマンドを *type* 引数にToolbar form window 定数を渡して呼び出した場合、エラー -10613 ("ツールバー型のフォームウィンドウを二つ作成することはできません")が生成されます。

OS Xのフルスクリーンモードとツールバーフォームウィンドウに関する注意: アプリケーションがツールバーウィンドウと、フルスクリーンモードをサポート(Has full screen mode Mac オプション)する標準のウィンドウの両方を表示しているとき、インターフェースのルールに基づき、標準のウィンドウがフルスクリーンモードになった際にはツールバーが非表示にならなければなりません。ウィンドウがフルスクリーンモードになったかどうかを調べるためには、縦幅(垂直方向のサイズ)がスクリーンの高さと完全に一致するかどうかをテストします(以下参照)。

Tool bar height

テーマ: ウィンドウズ(このコマンドは"ユーザーインターフェース"から移動になりました)

Tool bar height -> 戻り値

引数	型		詳細
戻り値	倍長整数	<-	ツールバーの高さ(ピクセル単位)、またはツールバーが非表示の場合には0

4D v14 R5以降、このコマンドは、**Open form window** コマンドに Toolbar form window 型の定数を使用して作成されたツールバーに対しても使用できるようになりました。

このコマンドはカレントの表示されているツールバーの高さを、ピクセル単位で返します。ツールバーは4D デザインモードツールバーでも**Open form window** コマンドで作成されたツールバーでも、コンテキストに応じてどちらでも可能です(デザインモードのツールバーは**Open form window** コマンドで作成されたカスタムのツールバーが表示された際には自動的に非表示になります)。

ツールバーが非表示の場合、コマンドは0を返します。

HIDE TOOL BAR と SHOW TOOL BAR

テーマ: ウィンドウ(このコマンドは"ユーザーインターフェース"から移動になりました)

SHOW TOOL BAR
このコマンドは引数を必要としません。

HIDE TOOL BAR
このコマンドは引数を必要としません。

これらのコマンドは4D v14 R5以降廃止予定が取り消されます。これらのコマンドはカレントプロセスにおいて **Open form window** を使用して作成されたカスタムのツールバーフォームウィンドウを管理するために使用されます。

- **SHOW TOOL BAR**: ツールバーウィンドウが(**Open form window** コマンドの Toolbar form window オプションを使用して)開かれている場合、このコマンドはそのウィンドウを表示状態にします。ツールバーウィンドウが既に表示状態になっている、または存在しない場合には、このコマンドは何もしません。
- **HIDE TOOL BAR**: ツールバーウィンドウが(**Open form window** コマンドの Toolbar form window オプションを使用して)開かれている場合、このコマンドはそのウィンドウを非表示にします。ツールバーウィンドウが既に非表示になっている、または存在しない場合には、このコマンドは何もしません。

例題

OS Xにおいて、カスタムのツールバーフォームウィンドウとHas full screen mode Mac オプションを持った標準のウィンドウを定義したとします。ツールバーウィンドウが表示されているときに標準のウィンドウがユーザーによって最大化された場合、最大化されたウィンドウにツールバーがかぶさるのは避けたいところです。

これを避けるために、標準ウィンドウの"On Resize"フォームイベントにおいてウィンドウがフルスクリーンモードに切り替わったことを検知し、**HIDE TOOL BAR** コマンドを呼び出す必要があります:

```
Case of
   :(Form event=On Resize)
      GET WINDOW RECT($left;$top;$right;$bottom)
      If(Screen height=($bottom-$top))
         HIDE TOOL BAR
      Else
         SHOW TOOL BAR
      End if
End case
```

Has too l bar button Mac 定数は廃止予定に

Has toolbar button Mac 定数は今後廃止予定になりました。これに対応するオプションはAppleによってOS X 10.6から廃止予定となってきました。

この定数は"Open Form Window" と "Open Window" の両方の定数テーマにおいて使用可能でしたが、4D v14 R5以降、_O_Has toolbar button Mac と改名されました。

Clickcount

Clickcount -> 戻り値

引数	型		説明
戻り値	倍長整数		連続したクリックの回数

説明

テーマ: フォームイベント

新しい**Clickcount** コマンドはマウスクリックイベントのコンテキストにおいて、ユーザーが同じマウスボタンを短時間に連続してクリックした回数を返します。通常、このコマンドはダブルクリックを表す2を返します。

このコマンドを使用することによって、リストボックスのヘッダー・フッターでのダブルクリックを検知したり、トリプルクリックなどのそれ以上の連続したクリックを扱えるようになります。

全てのマウスボタンのクリックは個別のクリックイベントを生成します。例えばユーザーがダブルクリックをした場合、最初のクリックに対しイベントが送信され、**Clickcount** は1を返します。二つ目のクリックに対してももう一つイベントが発生し、**Clickcount** は2を返します。

このコマンドはOn Clicked、On Header Click または On Footer Click フォームイベントのコンテキストにおいてのみ使用すべきものです。そのため、デザインモードを調べ、フォームプロパティや特定のオブジェクトのプロパティにおいて適切なイベントが正常に選択されているかどうかをチェックする必要があります。

On Clicked と On Double Clicked イベントが両方有効化されている場合、**Clickcount** コマンドは以下の組み合わせを返します:

- On Clicked イベントに1
- On Double Clicked イベントに2
- On Clicked イベントに2+n

例題 1

以下のコードストラクチャーをリストボックスヘッダー内に配置すると、シングルクリックとダブルクリックを管理することが出来るようになります:

```
Case of
   :(Form event=On Header Click)
      Case of
         :(Clickcount=1)
            ... //シングルクリックアクション
         :(Clickcount=2)
            ... //ダブルクリックアクション
      End case
End case
```

例題 2

ラベルは入力可能ではないですが、トリプルクリックをすると入力可能になります。ユーザーにラベルの編集を許可したい場合、以下の様なコードをオブジェクトメソッドに記載します:

```
If(Form event=On Clicked)
   Case of
      :(Clickcount=3)
         OBJECT SET ENTERABLE(*;"Label";True)
         EDIT ITEM(*;"Label")
   End case
End if
```

LISTBOX GET CELL COORDINATES

LISTBOX GET CELL COORDINATES ({* ;} object ; column ; row ; left ; top ; right ; bottom)

引数	型		説明
*	演算子	➡	指定時, Objectはオブジェクト名 (文字列) 省略時, Objectはフィールドまたは変数
object	フォームオブジェクト	➡	オブジェクト名 (* 指定時), または フィールドまたは変数 (* 省略時)
column	倍長整数	➡	列番号
row	倍長整数	➡	行番号
left	倍長整数	⬅	オブジェクトの左座標
top	倍長整数	⬅	オブジェクトの上座標
right	倍長整数	⬅	オブジェクトの右座標
bottom	倍長整数	⬅	オブジェクトの下座標

説明

LISTBOX GET CELL COORDINATES コマンドは引数 * および*object*によって指定されたリストボックス内の、*column* と *row* 引数で指定したセルの*left*、*top*、*right* および *bottom*にそれぞれ左端、上端、右端、下端の座標を(ポイント単位で)返します。

任意の * 引数を指定した場合、*object*はオブジェクト名です(文字列)。任意の * 引数を省略すると、*object* はフィールドまたは変数です。この場合、文字列ではなくフィールドまたは変数参照(フィールドまたは変数のみ)を指定します。

OBJECT GET COORDINATES コマンドとの統一性のため、原点はセルを含むフォームの左上端になります。また返される座標は理論値となります。この値は、クリッピングが起こるまではスクロールを考慮に入れます。結果として、そのセルは表示されていない(または一部しか表示されていない)こともあり、座標の位置はフォームの範囲を超えている(負の数値が返される)こともあります。セルが表示されているか(また、表示されているならどの部分が表示されているか)を調べるためには、返された座標と、リストボックスの座標を比較する必要があります。その際、以下の点に注意する必要があります:

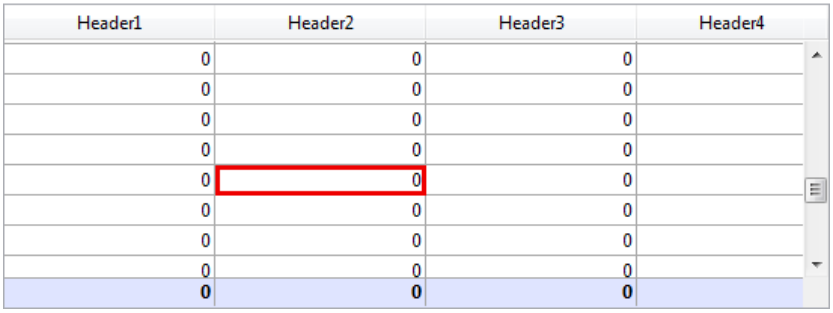
- 全てのセルは、親のリストボックスの座標(リストボックスに対しての**OBJECT GET COORDINATES** の値)に沿ってクリップされています。
- ヘッダー・フッターのサブオブジェクトは、列のコンテンツの上に表示されています。セルの座標がヘッダーやフッターの線と交差する場合には、その部分のセルは表示されません。
- ロックされた列の要素は、スクロール可能な列の要素の上に表示されます。スクロール可能な列の要素がロックされた列の要素と交差するとき、スクロール可能な列の要素は表示されません。

より詳細な情報に関しては、**OBJECT GET COORDINATES** コマンドの詳細を参照して下さい。

例題

リストボックス内の選択されたセルの周りに赤い長方形を描画する場合を考えます:

```
OBJECT SET VISIBLE(*;"rectangleInfo";False) //赤い長方形を初期化
 //長方形はフォーム内のどこかに既に定義済み
LISTBOX GET CELL POSITION(*;"LB1";$col;$raw)
LISTBOX GET CELL COORDINATES(*;"LB1";$col;$raw;$x1;$y1;$x2;$y2)
OBJECT SET VISIBLE(*;"RedRect";True)
OBJECT SET COORDINATES(*;"RedRect";$x1;$y1;$x2;$y2)
```



デザインモード

フォームオブジェクトにより長い名前が使用可能に

フォームオブジェクトにより長い名前が使用可能に

フォームオブジェクト名には、**255 バイト** までの名前を付ける事ができるようになりました。以前のリリースでは、これらの名前は31バイトまでに制限されていました。

より長い名前をフォームオブジェクト名に使用することができるようになったおかげで、"xxxx_Button" や "xxx_Mac"など、特定の命名ルールに則った名前を定義・適用しやすくなりました。

4D Server

新しいServerNetネットワークレイヤー
OS X用4D Serverの64ビット版について(最終リリース)

新しいServerNetネットワークレイヤー

v14 R5以降、4Dアプリケーションには*ServerNet* という名前の新しいネットワークレイヤーが追加されました。これは4D Serverとリモート4Dマシン(クライアント)間の通信を管理するためのものです。*ServerNet* は現代的で強固なAPIに基づいており、維持が簡単で、最新のネットワークテクノロジーを簡単に導入できる一方、高いレベルのパフォーマンスを発揮することができます。

現在の"旧式"のネットワークレイヤーは廃止予定となりますが、互換性のためにサポートはされます。*ServerNet* は新規に作成されたデータベース内では自動的に採用されます。

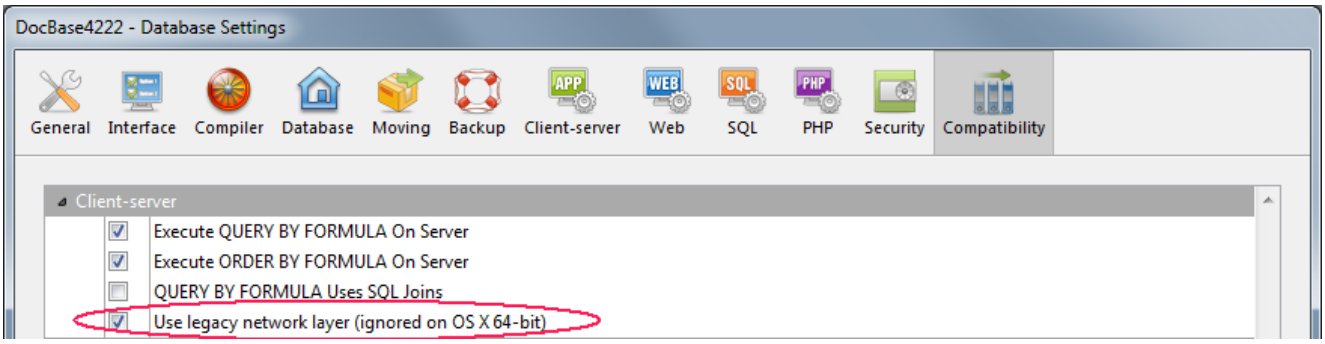
新しいオプションを使用して*ServerNet* を有効化/無効化することができます。ご自分のアプリケーションが将来のネットワークの進化の恩恵を享受できるように、全てのデータベースにおいてなるべく早く*ServerNet* を有効化することが強く推奨されます。

旧式ネットワークレイヤーの有効化または無効化

新しい互換性オプションによって、4D Serverにおける旧式ネットワークレイヤーをいつでも有効化または無効化することができます。以下のどちらかの方法を使用して下さい:

- データベース設定ダイアログボックス内で**旧式ネットワークレイヤーを使用**オプションをチェックする(以下参照)。
- SET DATABASE PARAMETER コマンドに対してUse legacy network layer 定数を使用する(これについては**Get database parameter と SET DATABASE PARAMETER** の章で説明があります)。

新しい互換性オプションは、互換性のページ内にあります:



注: このオプションはOS X用の4D Server v14 R5 64-bit版では無視されます。このプラットフォームでは*ServerNet* のみがご利用になれます。

デフォルトでは、このオプションは以下の様に設定されています:

- 4D v14 R5以降で作成されたデータベースにおいてはチェックがされていません(この場合*ServerNet* レイヤーが使用されます)。
- 変換された既存のデータベースにおいてはチェックがされています(この場合旧式のネットワークレイヤーが使用されます)。

このオプションは必要に応じてチェックをしたり外したりすることができます。例えば、クライアントアプリケーションの移行作業の途中などの場合に有効でしょう(以下を参照してください)。

設定を変更した際には、その設定を有効にするためにはアプリケーションの再起動が必要になるという点に注意して下さい。また、接続してたクライアントアプリケーションも、新しいレイヤー設定で接続するためには全て再起動する必要があります(*ServerNet* を使用するための必要最低限のクライアントのバージョンは4D v14 R4です。以下を参照して下さい)。

組み込み4Dクライアントを移行する

サーバーアプリケーションにおいて*ServerNet* レイヤーを有効化した場合、適合する4Dクライアントアプリケーションのみが接続することができます:

- 4D v14 R4 以降のバージョンのクライアントは何も変更しないまま接続することができます。

- それ以前のバージョン(v14.x とR4より前のv14'R'リリース)のクライアントはサーバーに接続する前にアップグレードをしなければなりません。

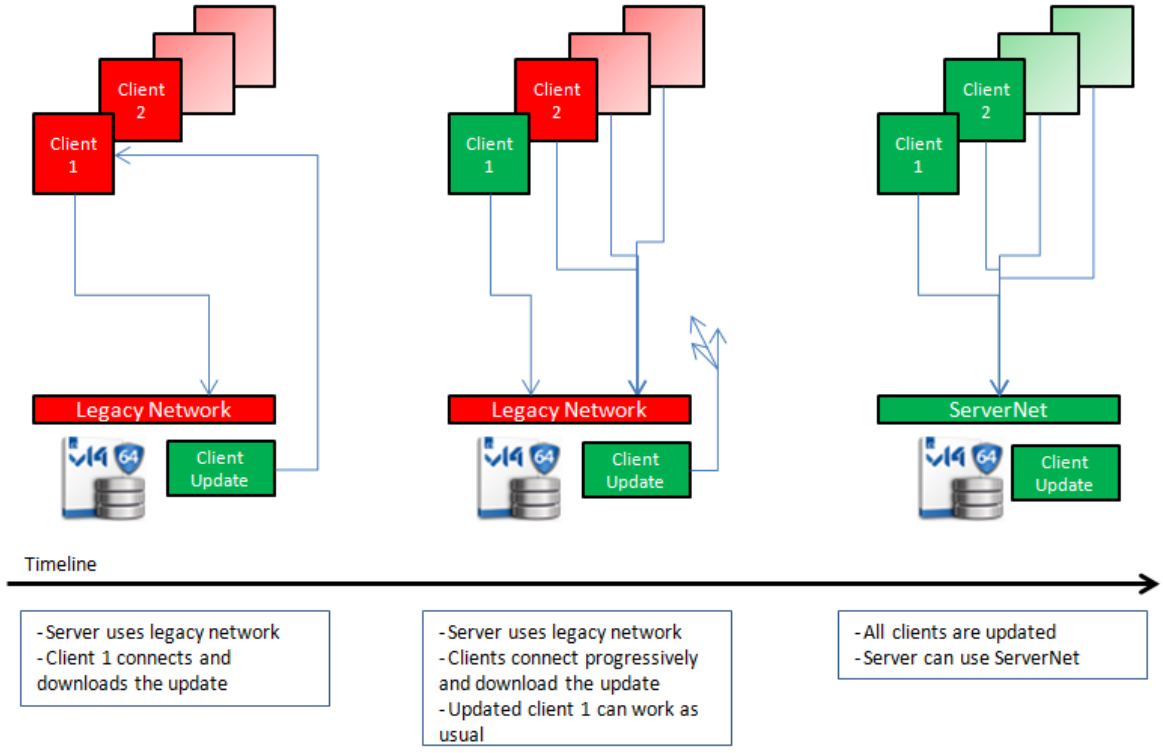
ご自分のアプリケーションがv14 R4以前のバージョンの組み込みクライアントで動いていて、4D Serverの自動機構を使用してアップデートされたクライアントアプリケーションをネットワーク越しに配付したい場合、移行戦略を練る必要があります。この戦略は以下の原則に則って練る必要があります:

- 互換性のないクライアントは旧式ネットワークレイヤーを使用する4D Server にしか接続することができません。
- アップデートされたクライアントはプロトコルを動的に適応させることができるので、サーバーが使用しているネットワークレイヤーに関わらず4D Server v14 R5以降に接続することができます。

移行戦略は、以下の様な段階を踏む必要があります:

1. 4D v14 R5以降を使用した、アップデートされたクライアントアプリケーションをビルドします。
2. 4D Server v14 R5 を、"旧式ネットワークレイヤーを使用"ネットワーク引数を有効化して実行します。
   この設定により、全てのクライアントが接続することができます。
   注: OS X用4D Server v14 R5 64-bit 版はこのオプションをサポートしていないことに注意して下さい。
3. 全てのクライアントが接続し、新しいバージョンをダウンロードし終わるまで一定時間待ちます。
   これには1日、1週間、あるいはそれ以上の時間がかかる可能性があります。この移行期間中、以前のバージョンのクライアントも、アップデートされたクライアントも、旧式のネットワークサーバーに接続することができます。
4. 全てのクライアントのアップデートが完了したら、旧式のネットワークレイヤーを無効化し、4D Serverを*ServerNet*へと切り替えることができます。

この戦略を図に表すと、以下の様になります:



クライアントのリクエストのログを取る

移行プロセスの間、"Diagnostic log recording"ファイルを有効化することが推奨されます。このファイルが有効化されると、4D Serverはそれぞれのクライアントのアップデートリクエストをこのファイルに記録するので、プロセスをモニターすることが出来るようになります。このログファイルはデフォルトでは有効化されていません。**SET DATABASE PARAMETER** コマンドを、 Diagnostic log recording 定数を1に設定して呼び出す必要があります。

それぞれのアップデートリクエストに対して、以下の情報が記録されます:

- クライアントのIPアドレス
- クライアントのバージョン
- "Update client" イベント

ログファイルをモニタリングすることは、サーバーを*ServerNet* ネットワークレイヤーに切り替えた後も、全てのクライアントが適切にアップデートされたかどうかを確認するために有用です。互換性のないクライアントが接続しようとした場合、サーバーは以下の情報を記録します:

- クライアントのIPアドレス
- クライアントのバージョン
- "Fail to connect" イベント

この場合、例えばクライアントを手動でアップデートするかどうか等を自分で判断することができます。

OS X用4D Serverの64ビット版について(最終リリース)

4D v14 R3 以降、4DはOS X用の64 ビット版の4D Serverの、使用可能なプレビューバージョンを提供してきました。このバージョンについては4D v14 R3 - Upgrade (PDF) マニュアル内にて詳細な説明があります。

このバージョンの開発は継続的なステップバイステップ方式で行われてきており、以前は利用不可だった機能についてもOS X用の64 ビット版の4D ServerのR5プレビューバージョンにおいて有効化されています。

機能の状況について

この一覧は、最初のプレビュー版(v14 R3)ではご利用いただけなかった機能についての現在のOS X用4D Server 64-bit版での状況をまとめたものです。

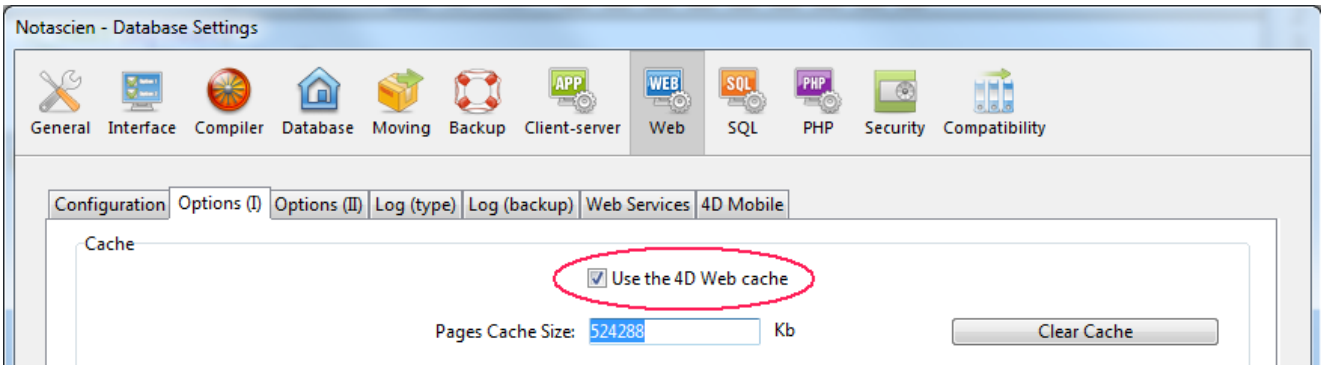
機能/テクノロジー	現在の状況
シリアルポート通信	利用不可
読み込み/書き出しダイアログボックス	利用不可
ラベルエディター	利用不可
サーバー上でグラフを生成	R4以降のバージョンで利用可能
HTTP クライアント(クライアント証明書の管理)	R4以降のバージョンで利用可能
4D Internet Commands (プラグイン)	R4以降のバージョンで利用可能
4D Pack (プラグイン)	**R5で利用可能** (**OS X用64ビット版の4D Packについて** を参照して下さい)
4D ODBC Pro (プラグイン)	利用不可
4D For OCI (プラグイン)	利用不可

Web サーバー

4D Webキャッシュがデフォルトで有効に

4D Webキャッシュがデフォルトで有効に

4D v14 R5以降、新規作成されたデータベースにおいて、スタティックリソースの4D Webキャッシュがデフォルトで有効化されます。データベース設定ダイアログボックス内において、**4D Webキャッシュを使用する**オプションは新規作成されたデータベースにおいては自動的にチェックされています:



以前のリリースから変換されたデータベースにおいては、このオプションの値は変更されずにそのままになっています。

4D Internet Commands

SMTP_QuickSend

SMTP_QuickSend

SMTP_QuickSend (hostName ; msgFrom ; msgTo ; subject ; message {; sessionParam}{; port}{; userName ; password}) -> Function result

引数	型		詳細
hostName	文字列	->	ホスト名またはIPアドレス
msgFrom	テキスト	->	メールアドレスまたはアドレスリスト
msgTo	テキスト	->	メールアドレスまたはアドレスリスト
subject	テキスト	->	メッセージの件名(デフォルトではUTF-8)
message	テキスト	->	メッセージ(デフォルトではUTF-8)
sessionParam	倍長整数	->	0 または 省略時 = SSLは使用しないがSSLへの変更は許可、1 = SSLを使用、2 = SSLを決して使用しない(SSLへの変更も許可しない)、4 = HTMLテキストをSSLを使用せずに送信する、5 = HTML テキストをSSLを用いて送信する、*8 = Mime HTML をSSL/TLS*を使用せずに送信する、*9 = Mime HTML をSSL/TLS*を使用して送信する
port	倍長整数	->	使用するポート番号
userName	テキスト	->	認証に使用するユーザー名
password	テキスト	->	認証に使用するパスワード
戻り値	整数	<-	エラーコード

説明

SMTP_QuickSend コマンドはMime HTML フォーマットのメッセージを送信することができるようになりました(SSL/TLS プロトコルの有無は選択できます)。Mime HTML (ファイル拡張子は.mht または .mhtml)とはHTMLコードやほかの外部リソース(画像など)を単一のドキュメントへとマージすることのできるWebページアーカイブフォーマットのことです。これは複数のブラウザと、それに加えてMSワードなどでもサポートされているフォーマットです。このフォーマットが4D Write

Pro エリアでサポートされていることから、4D Write Proエリアを、そのリソースも含めて保存したりメールで送信したりすることが簡単にできるようになりました。

sessionParam 引数に8を渡すと、Mime HTMLフォーマットのメッセージを標準モードで送信します。
sessionParam 引数に9を渡すと、Mime HTMLフォーマットのメッセージをSSL/TLSモードで送信します。

これらの値は通常の組み合わせと対応しますが、*sessionParam* 引数は実際はビットフィールドであり、bitwise 演算子を使用するとどんなカスタムの組み合わせも使用できます:

ビット数	bitが1の際に使用されるフォーマット
0	SSLまたはデフォルトの仕様を使用、クリアに接続、可能であればSSLへとアップグレードする
1	決してアップグレードをしない、アップグレードが可能な場合でも非-暗号化モードにとどまる。SSL(bit-0)が選択されている際にはビットは無視されます。
2	メッセージ本文はHTMLで、ヘッダーもそれに応じて設定されます。
3	MHTML メッセージを使用し、bit-2 (HTML) は無視されます。ユーザーは"To"、"From"、"Date"、そして"Subject"を除き、他の全ての設定を自分で設定する必要があります。

注: このコマンドは"非-Unicode"モードで実行されている変換されたデータベースをサポートしません。

例題

ディスク上に保存した .mht ドキュメントを、Eメールで送信する場合を考えます。この場合、以下の様な記述で送信できます:

```
$Message:=Document to text("c:\\documents\\invitation.mht")
$Host:="smtp.gmail.com"
$ToAddress:="john@4d.com"
$FromAddress:="harry@gmail.com"
$Subject:="Let's party"
$Param:=9 //SSLを賜与空いて MHTML で送信
$Port:=465 //gmail のSSLポート
$User:="harry@gmail.com"
$Password:="xyz&@!&@"
$Error:=SMTP_QuickSend($Host;$FromAddress;$ToAddress;$Subject;$Message;$Param;$Port;$User;$Password)
```

4D SVG

WindowsでDirect2Dを使用したフィルターが使用可能に

WindowsでDirect2Dを使用したフィルターが使用可能に

SVG_Filter_Blend、**SVG_Filter_Blur** そして **SVG_Filter_Offset** コマンドはDirect2Dが有効化されているWindowsのグラフィックソフトウェアのコンテキストにおいてもサポートされるようになりました(4D v14のデフォルトコンテキスト。詳細は**SET DATABASE PARAMETER**コマンドの Direct2D Software オプションを参照して下さい)。

以前の4Dのリリースでは、これらのコマンドはWindows上ではDirect2Dが無効化されている必要がありました。

4D Pack

OS X用64ビット版の4D Packについて
廃止予定コマンドの改名

OS X用64ビット版の4D Packについて

v14 R5のリリースをもって、4DはOS X用64-bit版の4D Packをリリースします。このバージョンによって、4D v14 R3でリリースされたOS X用の4D Serverの64-bit版で、4D Packのプラグインを使用できるようになります。

OS X用64-bit版の4D Packのインストールと使用は既存のバージョンに近いですが、以下のようないくつかの制約がつきます。

サポートされないコマンド

既存のバージョンで利用できた4D Pack コマンドは、以下のものを除き、全てOS X用の64-bit版で引き続きサポートされます:

- **_o_AP Save BMP 8 bits_o_AP Save BMP 8 bits**
  このコマンドは廃止予定のテクノロジーに基づいており、既に以前の4D Packのリリースで廃止予定となっていました。このコマンドはOS X用64-bit版の4D Packではサポートされません。

廃止予定コマンドの改名

いくつかの4D Pack コマンドは長い間廃止予定となってきました。これらのコマンドは今後の新しいプロジェクト内においては使用しないことが強く推奨されます。廃止予定である という事を明確にするために、4D Pack v14 R5以降、これらのコマンドには"_o_"の接頭辞がつけられました:

テーマ	以前の名前	新しい名前	状態
4D_Pack: ANSI streams	AP FCLOSE	**_o_AP FCLOSE**	廃止予定
	AP fopen	**_o_AP fopen**	廃止予定
	AP FPRINT	**_o_AP FPRINT**	廃止予定
	AP fread	**_o_AP fread**	廃止予定
4D Pack: Picture files	AP Save BMP 8 bits	**_o_AP Save BMP 8 bits**	廃止予定、OS X用64-bit版ではサポートされません
4D Pack: Utilities	AP Add table and fields	**_o_AP Add table and fields**	廃止予定、ドキュメントから削除
	AP Create relation	**_o_AP Create relation**	廃止予定、ドキュメントから削除
	AP Get file MD5 digest	**_o_AP Get file MD5 digest**	廃止予定、**Generate digest** に置き換え
	AP ShellExecute	**_o_AP ShellExecute**	廃止予定、**LAUNCH EXTERNAL PROCESS** に置き換え

4D Write Pro (テクニカルプレビュー)

4D Write Pro とは 4D Write plug-in の後継版です。

4D Write Pro は、4Dにとって大きな発展です。ですから、R-リリースの利点を利用してそれを徐々に配付しています。4D v14 R5は4D Write Pro の第一歩としてテクニカルプレビューを含んだバージョンになります。この最初のステップではドキュメントの互換性とHTMLフォーマットのEメールに焦点を当てています。今後の各ステップにおいて、使用できる機能とプログラミングのしやすさが改善していく予定です。

以下の点にご注意ください:

- 4D Write Pro は4D Writeと同じライセンスを使用しています。
- 4D Write Pro はプラグインではなくなり、4D自身に完全に統合されました。その結果、配布と管理がしやすくなりました。

4D Write Pro のドキュメントについて

新しいランゲージコマンドを含めた4D Write Pro の機能は、新しい**4D Write Proリファレンス** マニュアル内に全て記載されています。

このテクニカルプレビューについて

4D Write Pro テクニカルプレビュー はどれだけの機能が製品に実装されているかをお見せするためだけにリリースされたわけではありません。実際に皆様にお使いいただき、そのフィードバックをいただくためにリリースされました。もちろん、一部の機能は完全に開発が完了したわけではなく、不安定な部分もあるかもしれません。
以下の一覧は、主な機能の現在の状況をまとめたものです:

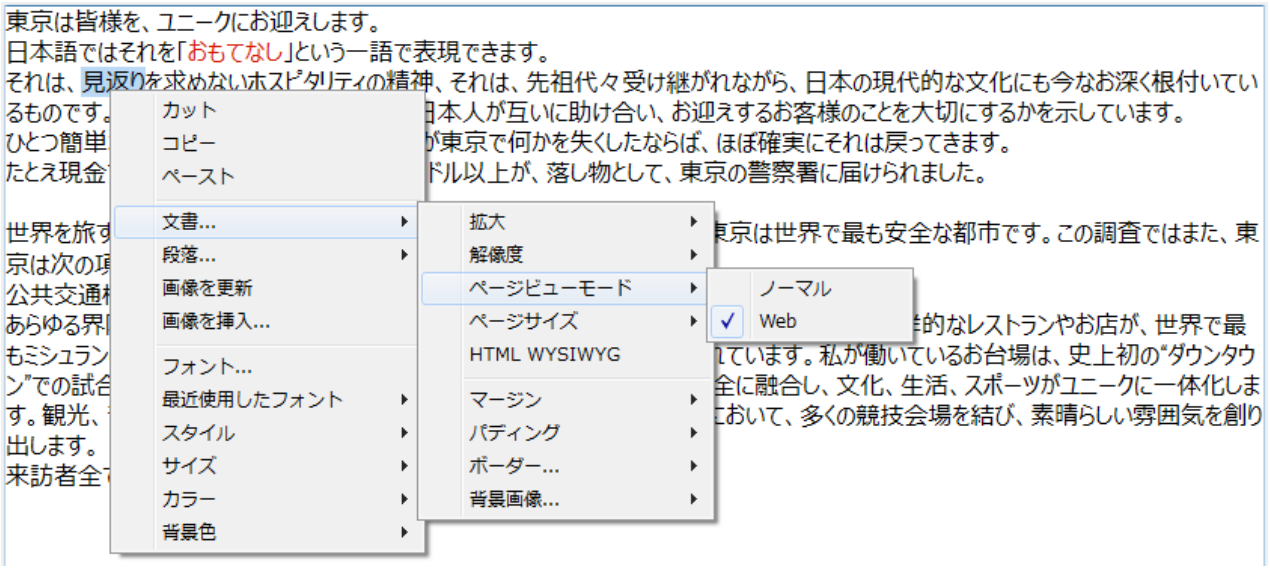
機能	現状(R5 テクニカルプレビュー時)	補足
フォーム内に4D Write Pro エリアを描画する	使用可能	
新規4D Write Pro ドキュメントを作成する	使用可能	**WP New**
4D Write ドキュメントを4D Write Pro オブジェクト内に読み込む	使用可能	**WP Import document** または **WP New**
4D Write Pro オブジェクトをドキュメントまたは変数へと書き出す	使用可能	**WP EXPORT DOCUMENT** または **WP EXPORT VARIABLE**
ドキュメントをコンテキストポップアップメニューを使用して編集	使用可能	
式(データベースフィールド、4Dメソッドの結果)を挿入する	使用可能	ST INSERT EXPRESSION コマンドを使用して下さい
ドキュメントの中身をプログラムを使用して編集する	一部利用可能	"ST" と "OBJECT" 4D コマンドのみ使用可能です
4D Write Pro ドキュメントを印刷する	利用不可	
ページ付けの管理	利用不可	
プログラムを使用してドキュメントのプロパティにアクセス	利用不可	
4D Write Proドキュメントをデータベースフィールド内に保存する	利用不可	

4D Write Pro インターフェース

4D Write Pro コマンドの状況
4D Write ドキュメントの読み込み

4D Write Pro インターフェース

4D Write Pro エリアにおいて**コンテキストメニュー**プロパティがチェックされている場合(**4D Write Pro エリアの作成**を参照して下さい)、アプリケーションモードにおいてユーザーは包括的なコンテキストメニューを使用することができます:



このメニューでは現在利用可能な全ての4D Write Proの機能を提供します(最終リリースでは、全ての機能とプロパティがプログラミングを通じてもご利用になれます)。

現時点ではコンテキストメニューの構成も中身も最終決定はしていない段階のため、このドキュメントではここの機能について詳細に記述することはしていません。
様々なサブメニューの中を見ることによって、4D Write Proでどんなことが出来るのかということを、直接感じ取っていただくことをお勧めします。

注: このテクニカルプレビューにおいては、4D Write Proドキュメントはデフォルトで**Web**ページビューモードで表示されています。このモードにおいては、テキストは自動的に折り返され、水平スクロールバーは(設定されていても)無効化されます。水平スクロールバーを使い、テキストを("ページサイズ"プロパティで定義されている)固定長に設定したい場合には、**ノーマル**ページビューモードへと切り替える必要があります。

4D Write Pro コマンドの状況

4D Write Pro v14 R5において、いくつかの新しいコマンドが4D Write Proエリアを管理するために"4D Write Pro"テーマに追加されました。これらのコマンドは4D Write Proリファレンスマニュアルの**4D Write Proランゲージ** の章内に詳細が書かれています。

以下の一覧は、4D v14 R5のテクニカルプレビューにおける4D Write Proコマンドの実装状況をまとめたものです:

コマンド名	現時点での状況(R5 テクニカルプレビュー時点)
WP EXPORT VARIABLE	現時点では書き出しのフォーマットは二つのみご利用いただけます。
WP EXPORT DOCUMENT	*format* 引数を渡さない場合、".htm" または ".html"ファイル拡張子を使う必要がありますが、ご利用いただける書き出しフォーマットは現在一つだけです。
WP Import document	4D Write Pro オブジェクト内で現在サポートされている4D Write 機能の詳細な一覧については、**4D Write ドキュメントの読み込み** の章を参照して下さい。
WP New	4D Write Pro オブジェクト内で現在サポートされている4D Write 機能の詳細な一覧については、**4D Write ドキュメントの読み込み** の章を参照して下さい。

注: 4D Write Pro エリアでは"オブジェクト(フォーム)" と "スタイル付テキスト" テーマコマンドのみがご利用いただけるという点にご注意ください(**オブジェクト(フォーム)テーマのコマンドの使用** と **スタイル付テキストテーマのコマンドの使用** の章を参照して下さい)。

4D Write ドキュメントの読み込み

新しい4D Write Proオブジェクトの主要な機能の一つとして、既存の4D Writeドキュメントの読み込みと変換機能が挙げられます。これによって4D Write プラグインに依存しているアプリケーションを移行させることができます。

互換性に関する注意: サポートされるのは、4DWite ドキュメントのうち最後の世代("4D Write v7")のドキュメントのみに限られます。

4D Write ドキュメントを読み込むためには

このテクニカルプレビューでは、4D Write Proオブジェクトは4D Writeドキュメントを読み込むための方法を二つご用意しています:

- ディスク上にに保存されいている4D Writeに対しては、**WP Import document** コマンドを使用して下さい。
- BLOBフィールドに4D Writeエリアに対しては、**WP New** コマンドを使用して下さい。

より詳細な情報に関しては、それぞれのコマンドの詳細を参照して下さい。

4D Write のプロパティで復元されるもの

4D Write プラグインから4D Write Proへの移行を簡単にするために、4D Write の機能のうち、出来る限り多くのものが4D Write Proオブジェクトでサポートされようとしています。

以下の段落では、4D Write プラグインのプロパティのうち、 **WP Import document** または **WP New** コマンドを使用して4D Write Proエリアへと読み込んだときに復元されるプロパティについてまとめています。

しかしながら一部の機能において、バグとはみなされない小さな差異が生じることがあります。これらは例えば、4D Write Proで使用されるデフォルトの行頭の記号や、下線のタイプの小さな変化などによるものです。

ドキュメント情報

4D Write プラグイン	4D Write Pro
作成日時	v14 R5で利用可能Available in v14 R5
修正日時	v14 R5で利用可能
ロック	利用不可(読み込み専用オブジェクトプロパティを使用して下さい)
タイトル	v14 R5で利用可能
件名	v14 R5で利用可能(標準テキストのみ)
作者	v14 R5で利用可能
会社名	v14 R5で利用可能
注記	v14 R5で利用可能

ドキュメントの表示の引数

4D Writeプラグイン	4D Write Pro
ページモード	読み込まれません(コンテキストメニューの、ドキュメント/ページモードを使用して下さい)
ルーラー	利用不可
枠	利用不可
ヘッダー	利用不可
フッター	利用不可
最初のページのヘッダー	利用不可
最初のページのフッター	利用不可
画像	利用不可
縦スクロールバー	読み込まれません(オブジェクトプロパティの縦スクロールバーを使用して下さい)
横スクロールバー	読み込まれません(オブジェクトプロパティの横スクロールバーを使用して下さい)
非表示文字	利用不可
参照	利用不可(**ST SET OPTIONS**を参照して下さい)
カラム分割	利用不可
縦スプリッター	利用不可
横スプリッター	利用不可
Wysiwyg	利用不可
ズーム	読み込まれません(コンテキストメニューの、ドキュメント/ズームを使用して下さい)

ドキュメントの引数

4D Write プラグイン	4D Write Pro
単位	利用不可
言語	利用不可
カラム数	利用不可
行間隔	利用不可
ウィンドウ&オーファン	利用不可
デフォルトのタブ	v14 R5で利用可能
先頭のタブ	利用不可
URLカラー	利用不可
アクセス済みのURLカラー	利用不可

ドキュメントのページ付の引数

4D Writeプラグイン	4D Write Pro
ページ幅	利用不可
ページの高さ	利用不可
最初のページ番号	利用不可
最初のページのヘッダーとフッターを別にする	利用不可
左右のページでヘッダーとフッターが異なる	利用不可
ページバインディング	利用不可
反対側のページ	利用不可
ページマージン	v14 R5 にて利用可能(一時的な実装)
ヘッダーの上マージン	利用不可
ヘッダーの下マージン	利用不可
フッターの上マージン	利用不可
フッターの下マージン	利用不可
最初のページの上マージン	利用不可
最初のページの下マージン	利用不可
最初のページのヘッダーの上マージン	利用不可
最初のページのヘッダーの下マージン	利用不可
最初のページのフッターの上マージン	利用不可
最初のページのフッターの下マージン	利用不可
最初のページを右側にする	利用不可

Document printing parameters

4D Write プラグイン	4D Write Pro
用紙の種類	利用不可
横向きに印刷	利用不可
幅	利用不可
高さ	利用不可
ユーザー設定マージン	利用不可
倍率	利用不可
X 解像度	利用不可
Y 解像度	利用不可

画像

注: 4D Write Pro はページ内での画像の絶対位置に関してはまだサポートしていません。このテクニカルプレビューにおいてはインライン画像のみがサポートされ、読み込まれます。

4D Write プラグイン	4D Write Pro
X (左)	(& position :absolute) (ページ内の画像のみ)
Y (上)	(& position :absolute) (ページ内の画像のみ)
幅	v14 R5にてご利用可能
高さ	v14 R5にてご利用可能
ページ番号	利用不可
背景画像	利用不可
最初のページには含めない	利用不可
ビューポートモード(縮小して画像を入れる)	v14 R5にてご利用可能
式である	利用不可
サイズを保持	利用不可

文字プロパティ

4D Write プラグイン	4D Write Pro (span プロパティ)
イタリック	v14 R5でご利用可能
ボールド	v14 R5でご利用可能
取り消し線	v14 R5でご利用可能
下線	v14 R5でご利用可能
シャドウ	v14 R5でご利用可能
指数(上付きまたは下付き)	v14 R5でご利用可能
大文字(英大文字または小型英大文字)	v14 R5でご利用可能
フォントファミリー	v14 R5でご利用可能
フォントサイズ	v14 R5でご利用可能
テキストカラー	v14 R5でご利用可能
テキスト背景カラー	v14 R5でご利用可能
下線カラー	v14 R5でご利用可能
取り消し線カラー	v14 R5でご利用可能
シャドウカラー	v14 R5でご利用可能
ユーザープロパティ	利用不可
スペルチェック(シンタックス&グラマー onまたはoff)	利用不可
表示	利用不可
スタイルシート	読み込まれません(スタイルは読み込まれますが、スタイルシートは利用不可です)

段落プロパティ

4D Write プラグイン	4D Write Pro
均等位置	v14 R5でご利用可能
Interline	v14 R5でご利用可能
箇条書き	v14 R5でご利用可能
左マージン	v14 R5でご利用可能
右マージン	v14 R5でご利用可能
テキストインデント	v14 R5でご利用可能
境界線スタイル	v14 R5でご利用可能
境界線カラー	v14 R5でご利用可能
境界線の背景カラー	v14 R5でご利用可能
左境界線	v14 R5でご利用可能
右境界線	v14 R5でご利用可能
上境界線&上の内側の境界線	v14 R5でご利用可能
下の境界線&下の内側の境界線	v14 R5でご利用可能
境界線間隔	v14 R5でご利用可能
スタイルシート	v14 R5でご利用可能
タブ	v14 R5でご利用可能

ハイパーリンク

4D Write プラグイン	4D Write Pro
URL リンク	v14 R5でご利用可能
4D メソッドリンク	利用不可
ドキュメントを開くリンク	利用不可

4D 式

4D Write プラグイン	4D Write Pro
4D 式	v14 R5で利用可能
日付と時間	v14 R5で利用可能
HTML 式	利用不可
RTF 式	利用不可

テキストデータ

4D Write プラグイン	4D Write Pro
メインテキストデータ	v14 R5で利用可能
ヘッダーテキストデータ	利用不可
フッターテキストデータ	利用不可

最適化

SQL Select の Group by と Order by 宣言
ネットワークセキュリティ

SQL Select の Group by と Order by 宣言

4D v14 R5以降、SQL SELECT宣言のGROUP BY と ORDER BY節が異なる設定において最適化されました:

- SELECT FROM GROUP BY または SELECT FROM ORDER BY が単一のテーブルに適用されたとき
- SELECT FROM GROUP BY または SELECT FROM ORDER BY がinner joinsを使用して複数のテーブルに適用されたとき

この最適化が関係するのは、単純な列の参照に関してのみです(式には適用されません)。
4D v14 R5以降では、一般的に、以下の場合において実行速度が速くなります:

- **Order by の例**

```
Begin SQL
   DROP TABLE IF EXISTS T1;
   DROP TABLE IF EXISTS T2;
   CREATE TABLE T1 (C1 INT);
   CREATE TABLE T2 (C2 INT);
   INSERT INTO T1(C1) VALUES (1);
   INSERT INTO T1(C1) VALUES (2);
   INSERT INTO T1(C1) VALUES (3);
   INSERT INTO T2(C2) VALUES (2);
   INSERT INTO T2(C2) VALUES (3);
End SQL

ARRAY LONGINT($result;0)
ARRAY LONGINT($result1;0)
ARRAY LONGINT($result2;0)

 // T1 と T2 を組み合わせたシンプルなORDER BYの例
 // $result には [2, 3]が格納されます
Begin SQL
     SELECT C1
       FROM T1, T2
       WHERE C1=C2
       ORDER BY C1
       INTO :$result
End SQL

 // ORDER BY は複数の列が使用されているときにも使用できます。
 // $result1 と $result2 にはどちらも[2, 3]が格納されます。
Begin SQL
     SELECT C1, C2
       FROM T1, T2
       WHERE C1=C2
       ORDER BY C1, C2
       INTO :$result1, :$result2
End SQL

 // 最適化されるのは単純な列の参照のみです。
 // 以下の様に、(C1 + 1) のような式がセレクションに使用されている場合、
 // 実行速度は速くなりません。$result には[3, 4]が格納されます(ただし値は正確です)。
Begin SQL
     SELECT C1 + 1
       FROM T1, T2
       WHERE C1=C2
       ORDER BY C1
       INTO :$result
End SQL

 // インデックスを使用してセレクションの内部を参照することもできます。
```

```
 // 既に述べられている通り、実行速度が速くなるのはセレクションに入っているのが単純な列の参照のみのときです。
 // $result には[ 2, 3 ]が格納されます。
Begin SQL
     SELECT C1
       FROM T1, T2
       WHERE C1=C2
       ORDER BY 1
       INTO :$result
End SQL
```

- **Group by の例**

```
Begin SQL
   DROP TABLE IF EXISTS T1;
   DROP TABLE IF EXISTS T2;
   CREATE TABLE T1 (C1 INT, C3 INT);
   CREATE TABLE T2 (C2 INT);
   INSERT INTO T1(C1, C3) VALUES (3, 1);
   INSERT INTO T1(C1, C3) VALUES (1, 1);
   INSERT INTO T1(C1, C3) VALUES (2, 1);
   INSERT INTO T1(C1, C3) VALUES (3, 1);
   INSERT INTO T1(C1, C3) VALUES (2, 1);
   INSERT INTO T1(C1, C3) VALUES (3, 1);
   INSERT INTO T2(C2) VALUES (2);
   INSERT INTO T2(C2) VALUES (3);
End SQL

ARRAY LONGINT($result;0)
ARRAY LONGINT($result1;0)
ARRAY LONGINT($result2;0)

 //T1 と T2 を組み合わせたシンプルなGROUP BY の例
 //$result には[2, 3]が格納されます。必ずしもグループが昇順で返されるとは限らない、
 //という点に注意して下さい。つまり、$result の中身は[3, 2]の可能性もあるという事です。
Begin SQL
   SELECT C1
       FROM T1, T2
       WHERE C1=C2
       GROUP BY C1
       INTO :$result
End SQL

 // 以下は合計をリクエストするシンプルな例です。
 // C3はそれぞれのグループに対して常に1なので(C1の固有値)、
 // SUM(C3) が繰り返しの回数という事になります。
 // $result1 には[2, 3]が格納されます。
 // $result2 にも [2, 3]が格納されます。
Begin SQL
   SELECT C1, SUM(C3)
       FROM T1, T2
       WHERE C1=C2
       GROUP BY C1
       INTO :$result1, :$result2
End SQL
```

ネットワークセキュリティ

ネットワークセキュリティ機能は4D v14 R5において最適化され、4Dアプリケーションの保護を強化できるようにしました:

- 脆弱な暗号リストの組み合わせは削除されました。
- デフォルトの4D証明書キーの長さは、2048bitへと増加されました。
- 安全な通信のために、自分で設定した暗号化キーを使用できるようになりました。

これらの変更が関係するのは、以下のセキュアな接続です:

- クライアント/サーバー間
- SQL サーバー
- HTTP クライアント