Cocoa はやっぱり!
ステータスバーの巻
2002.6.20

■ 今回のテーマ

今回のテーマは、**ステータスバー**です。ステータスバーというのは、メニューバーの右側に表示されるユーザーインターフェイスです。メニューバー上にある時計とか音量調整は Menu Extra と呼ばれるものですが、これに似たものです。アクティブなアプリケーションが切り替わっても、常に表示されて機能を提供できるところが特徴です。ステータスバーを使うと、Cocoa アプリケーションを作るのとほぼ同じ要領で作ることが出来ます。

◎ 推奨環境

この解説は、以下の環境を前提にしていますので、ご確認ください。これを書いている段階では、Developer Tools の最新版がベータ版なのですが、これをそのまま使用しています。

- Mac OS X 10.1.5
- Project Builder Version 2.0 (April 2002 Developer Tools Beta)
- Interface Builder 2.2.1 (v263.2)

■ メニューを持つステータスバーの作成

まずは、メニューを持つステータスバーの作り方から説明していきます。以下のようなものを目指します。

メニューを持つステータスバー

● 下準備

まずは、以下の手順で下準備をしてください。まずは、Project Builder にて、プロジェクトを作成します。

- 「 File → New Project... 」メニューを実行。
- 「 Application → Cocoa Application 」を選択し、「 Next 」ボタンをクリック。
- 「 Project Name : 」に「 StatusBarTest 」と入力し、タブキーを押し、「 Finish 」ボタンをクリック。

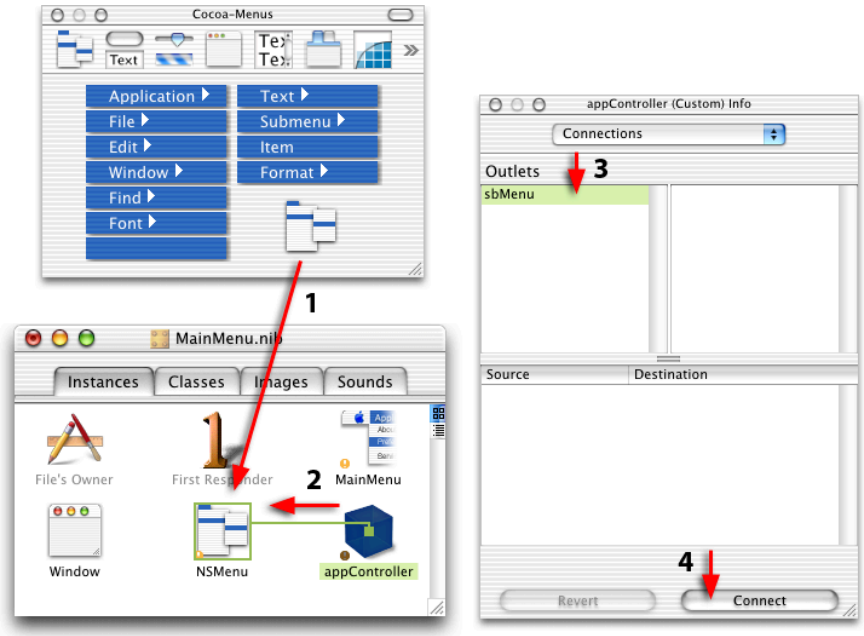
続いて、Interface Builder にてアプリケーションデリゲート用のクラスを作成します。ステータスバーの内容が固定ならば、一般的にアプリケーション起動時にステータスバーの作成を行いますので、アプリケーショ

ンデリゲートのインスタンスが必要になります。

・Project Builder の「 Resources → MainMenu.nib 」をダブルクリック。
・`NSObject` のサブクラス「 `appController` 」を作成。
・`appController` にアウトレット「 `sbMenu` 」を追加。
・`appController` をインスタンス化。
・`appController` のソースコードを出力。
・File's Owner の `delegate` に `appContoller` を接続。

`sbMenu` というアウトレットを作りましたが、これにはステータスバー上で使うメニューを繋ぎます。

・Cocoa-Menus から `NSMenu` を Instances タブにドロップ。
・`appController` の `sbMenu` にこの `NSMenu` を接続。



ステータスバーで使用するメニューをアウトレットに接続

後、ウィンドウが表示されないように、Window の Attributes の「 Visible at launch time 」のチェックを外しておきます。削除しても構いませんが、今回は後程使いますので残しておきます。これで下準備は完了です。

● 起動時の処理をコーディング

では、アプリケーション起動時に呼ばれるデリゲートの applicationDidFinishLaunching : メソッドを実装していきます。

appController.m > applicationDidFinishLaunching :

```
- (void) applicationDidFinishLaunching : (NSNotification *) aNote {

  // ステータスバー作成
  NSStatusBar *bar = [ NSStatusBar systemStatusBar ];

  { // ステータスアイテム作成

    NSStatusItem *sbItem = [ bar statusItemWithLength : NSVariableStatusItemLength ];
    [ sbItem retain ];

    [ sbItem setTitle          : @"Test"   ]; // タイトルをセット
    [ sbItem setToolTip        : @"Hello!" ]; // ツールチップをセット
    [ sbItem setHighlightMode : YES        ]; // クリック時にハイライト表示

    [ sbItem           setMenu : sbMenu    ]; // メニューをセット

  }

}
```

最初の 1 行でステータスバーのインスタンスを作成しています。「 **systemStatusBar** 」メソッドで空のステータスバーが出来ます。

```
  NSStatusBar *bar = [ NSStatusBar systemStatusBar ];
```

NSStatusBar : ステータスバーのインスタンスを生成

書式
```
  + (NSStatusBar *) systemStatusBar
```
出力
返り値 : ステータスバーのインスタンス

このステータスバーに「 **ステータスアイテム** 」を追加していきます。**1 つのステータスバーには複数のステータスアイテムが存在できる**ようになっています。つまり、ステータスバーは、ステータスアイテム群を管理するもので、ステータスアイテムはそこに乗っているボタンのようなものです。

```
    NSStatusItem *sbItem = [ bar statusItemWithLength : NSVariableStatusItemLength ];
    [ sbItem retain ];
```

ステータスアイテムを作成してステータスバーに登録するには、「 **statusItemWithLength :** 」メソッドを使います。パラメータは横幅です。幅が決まっているものはドット数を指定しますが、今回のようにタイトルに文字列が含まれている場合は「 **NSVariableStatusItemLength** 」を指定することで**自動計算**させることができます。また、statusItemWithLength : メソッドで生成されたインスタンスは、そのままだと破棄されてしまうので retain で保持しておきます。

NSStatusBar : ステータスアイテムのインスタンスの生成と登録

書式

```
- (NSStatusItem *) statusItemWithLength : (float) length
```

入力

```
length : ステータスアイテムの幅 ( ドット数 )
          or NSVariableStatusItemLength – 自動計算する
          or NSSquareStatusItemLength   – 正方形
```

出力

```
返り値  : ステータスアイテムのインスタンス
```

この後は、このステータスアイテムの属性を指定していきます。

```
[ sbItem setTitle          : @"Test"   ]; // タイトルをセット
[ sbItem setToolTip        : @"Hello!" ]; // ツールチップをセット
[ sbItem setHighlightMode : YES        ]; // クリック時にハイライト表示

[ sbItem           setMenu : sbMenu    ]; // メニューをセット
```

「 **setTitle :** 」メソッドでタイトルを、「 **setToolTip :** 」でツールチップ (簡易ヘルプ) をセットすることが出来ます。「 **setHighlightMode :** 」で YES を指定しておくと、クリックされたときに自動的にハイライト表示されるようになります。

ここまでで、メニューバー上にステータスアイテムが表示されるようになります。さらに「 **setMenu :** 」メソッドを使うことで、ステータスアイテムにメニューをぶら下げることが出来ます。これで実行すると、メニューバー上にステータスバーが表示され、メニューがプルダウンできるようになります。他のアプリケーションに切り替えても消えないことも確認してください。

プロジェクトに画像ファイルを登録しておけば、その画像をアイコンとして表示させることも出来ます。「 **setImage :** 」メソッドで画像をステータスアイテムにセットできます。setTitle : でヌルストリングをセットすればアイコンだけのステータスアイテムに出来ます。

```
[ sbItem setTitle : @""   ];                                  // タイトルを消す
[ sbItem setImage : [ NSImage imageNamed : @"Kero" ] ]; // 画像をセット
```

画像だけのステータスアイテム

メニューが選択されたときに何かを実行させるようにするのは、今までと全く同じで**アクションの接続をする**

だけです。例えば、メニューから File's Owner の `hideOtherApplications :` に接続するなどしてテストしてみてください。

また、先程ステータスバーには、複数のステータスアイテムを登録できると書きました。ステータスアイテムの作成する部分をコピー&ペーストして 2 回実行するようにしてみてください。以下のようになるはずです。後から登録した方が左側に位置します。

1 つのステータスバーに複数のステータスアイテムを登録した場合

では、ここまでで出てきたメソッドのおさらいと関連メソッドの紹介をしておきます。

● タイトル関連

`NSStatusItem` : ステータスアイテムのタイトルを変更する

書式
```
- (void) setTitle : (NSString *) title
```
入力
`title` : ステータスアイテムのタイトル文字列

`NSStatusItem` : ステータスアイテムのタイトルを取得する

書式
```
- (NSString*) title
```
出力
返り値 : ステータスアイテムのタイトル文字列

ステータスアイテムの文字列は、修飾属性付きにもできます。

`NSStatusItem` : ステータスアイテムのタイトルをリッチテキストで変更する

書式
```
- (void) setAttributedTitle : (NSAttributedString *) title
```
入力
`title` : ステータスアイテムのタイトル文字列

`NSStatusItem` : ステータスアイテムのタイトルをリッチテキストで取得する

書式
```
- (NSAttributedString *) attributedTitle
```
出力
返り値 : ステータスアイテムのタイトル文字列

● ツールチップ関連

NSStatusItem : ステータスアイテムのツールチップを変更する

書式
```
- (void) setToolTip : (NSString *) toolTip
```
入力
```
toolTip : ツールチップの文字列
```

NSStatusItem : ステータスアイテムのツールチップを取得する

書式
```
- (NSString *) toolTip
```
出力
```
返り値 : ツールチップの文字列
```

● ハイライトモード関連

NSStatusItem : ステータスアイテムのハイライトモードを変更する

書式
```
- (void) setHighlightMode : (BOOL) highlightMode
```
入力
```
highlightMode : ハイライトモード
                YES — クリックされたらハイライト表示する。
                 NO — クリックされてもハイライト表示しない。
```

NSStatusItem : ステータスアイテムのハイライトモードを取得する

書式
```
- (BOOL) highlightMode
```
出力
```
返り値 : ハイライトモード
         YES — クリックされたらハイライト表示する。
          NO — クリックされてもハイライト表示しない。
```

● 画像関連

NSStatusItem : ステータスアイテムの画像を変更する

書式
```
- (void) setImage : (NSImage *) image
```
入力
```
image : ステータスアイテムの画像。
```

NSStatusItem : ステータスアイテムの画像を取得する

書式
```
- (NSImage *) image
```
出力
返り値 : ステータスアイテムの画像

● メニュー関連

NSStatusItem : ステータスアイテムのメニューを変更する

書式
```
- (void) setMenu : (NSMenu *) menu
```
入力
menu: ステータスアイテムのメニュー。

NSStatusItem : ステータスアイテムのメニューを取得する

書式
```
- (NSMenu *) menu
```
出力
返り値 : ステータスアイテムのメニュー。

■ メニューバーのクリックだけで実行するステータスバーの作成

続いては、メニューバーにクリックしただけで機能を実行するようなものを作ってみます。基本的には先程と同じですが、setMenu : でメニューを登録する代わりに、「 **setAction :** 」メソッドで実行するアクションと「 **setTarget :** 」でアクションメッセージの送信先を指定します。以下は、ステータスアイテムを作るところのみを抜き出してあります。

```
NSStatusItem *sbItem = [ bar statusItemWithLength : NSVariableStatusItemLength ];
[ sbItem retain ];

[ sbItem setTitle         : @"Click" ];
[ sbItem setHighlightMode : YES      ];

[ sbItem setAction : @selector( terminate : ) ]; // アクションをセット
[ sbItem setTarget : NSApp                    ]; // ターゲットをセット
```

NSApp に terminate : を送っているので、ステータスアイテムをクリックするとアプリケーションが終了するはずです。

● アクション関連

NSStatusItem : ステータスアイテムのアクションを変更する

書式
```
- (void) setAction : (SEL) action
```
入力
action : ステータスアイテムのアクション。

NSStatusItem : ステータスアイテムのアクションを取得する

書式
```
- (SEL) action
```
出力
返り値 : ステータスアイテムのアクション。

NSStatusItem : ステータスアイテムのターゲットを変更する

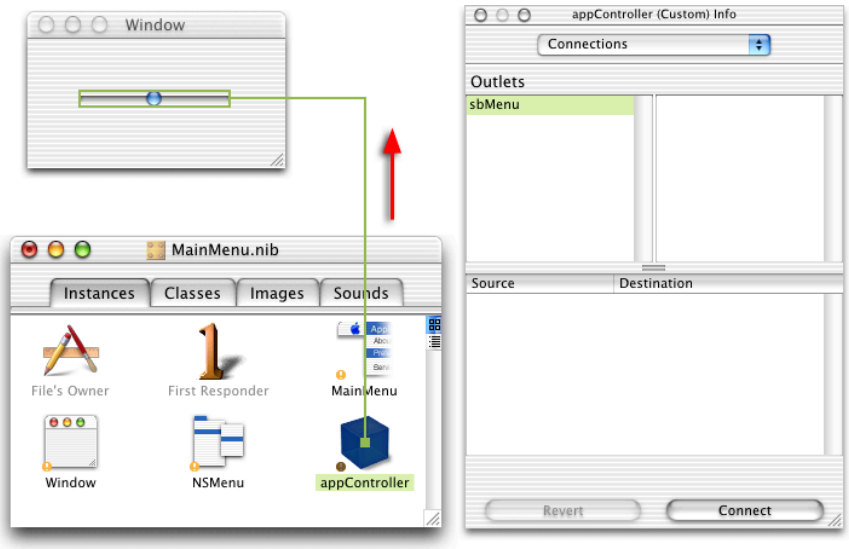
書式
```
- (void) setTarget : (id) target
```
入力
menu: ステータスアイテムのターゲット。

NSStatusItem : ステータスアイテムのターゲットを取得する

書式
```
- (id) target
```
出力
返り値 : ステータスアイテムのターゲット。

■ ビューを持つステータスバーの作成

もっと複雑なものをメニューバー上に起きたい場合は、「 **setView :** 」メソッドでビューを置くことも出来ます。例えば、Interface Builder 上でウィンドウにスライダーを配置して sbMenu に繋ぎかえます。



sbMenu にスライダーを繋ぎかえる

そして、以下のようにコードを書き換えます。

```
NSStatusItem *sbItem = [ bar statusItemWithLength : [ sbMenu bounds ].size.width ];
[ sbItem retain ];

[ sbItem setView : sbMenu ]; // ビューをセット
```

statusItemWithLength : のパラメータにはビューの幅を渡すことになります。bounds メソッドで矩形が分かりますので「 .size.width 」で幅を取得します。そして、「 **setView :** 」メソッドでビューをセットします。実行すると以下のようになります。

ビューを持つステータスアイテム

● ビュー関連

NSStatusItem : ステータスアイテムのビューを変更する

書式

```
- (void) setView : (NSView *) view
```

入力

action : ステータスアイテムのビュー。

NSStatusItem : ステータスアイテムのビューを取得する

書式

```
- (NSView *) view
```

出力

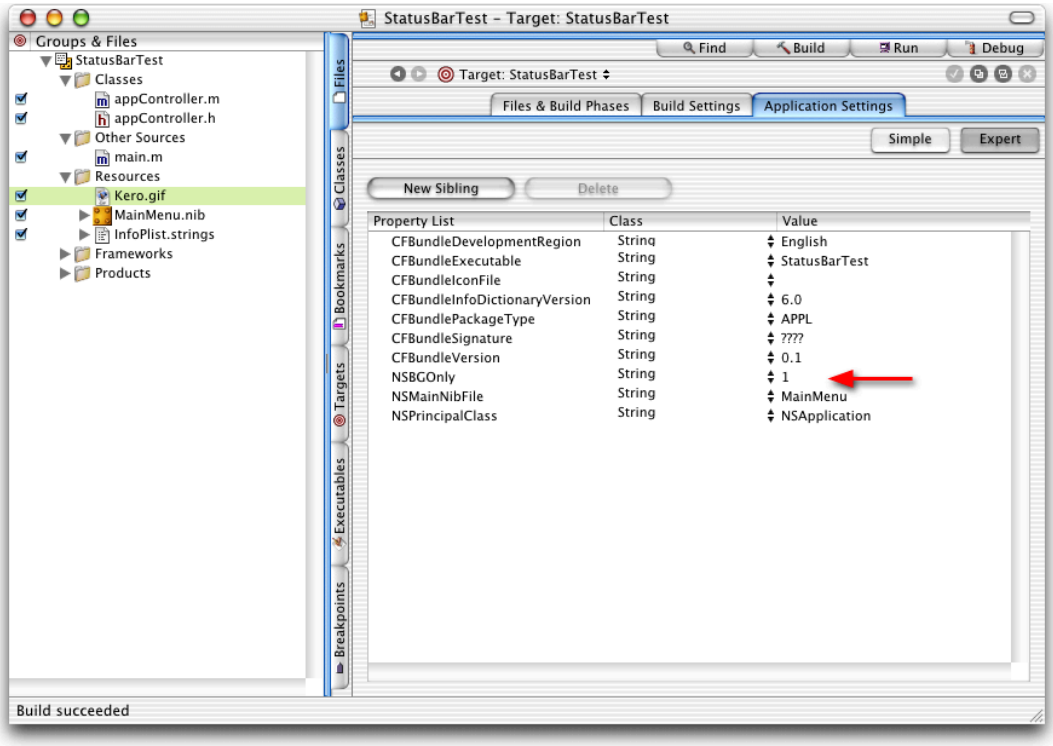
返り値 : ステータスアイテムのビュー。

■ アプリケーションを見えなくする

実際に動かしてみると分かりますが、作ったアプリケーションは起動するとドックに表示されますし、ステータスバーをクリックするとアプリケーションがアクティブになります。Menu Extra のようにメニューバーだけに存在させて、アプリケーションが切り替わらないようにしたいことが多いと思いますので、この方法について説明します。

Project Builder のプロジェクトウィンドウの Target タブを開いて、Application Settings の Expert まで移動します。そして、以下の手順を実行します。

- 「 New Sibling 」ボタンをクリックして項目を追加。
- 「 Property List 」に「 NSBGOnly 」と入力。
- 「 Class 」を「 String 」に変更。
- 「 Value 」に「 1 」と入力。



Application Settings を変更してバックグラウンドアプリケーションへ

「 BGOnly 」というのは「 BackGround Only 」のことです。この設定だけでアプリケーションはステータスバーのみに存在できるようになります。起動してもドックには現れませんし、ステータスバーをクリックし

てもアプリケーションは切り替わりません。ただし、こうすると、「 File → Quit 」メニューも画面に出なくなりますので、何かしらの終了手段を用意する必要があります。アプリケーションを終了させるには、以下の 1 行を実行します。

```
[ NSApp terminate : self ];
```

また、「 hideOtherApplication : 」など、いくつかのアクション無効になるようですので注意が必要です。