

今から始める Cocoa プログラミング》

文書ファイルを扱うアプリケーションを作る(19)

パッケージ情報とアバウトの国際化

nib ファイルを言語ごとに用意することや、あるいは文字列の置き換えテーブルを使って、言語ごとに異なる文字列メッセージを出すといったことはすでに説明した。国際化関連の作業の残りとして、パッケージ情報の国際化や、あるいはアバウト画面の国際化についても作業をしておこう。

パッケージ情報の国際化

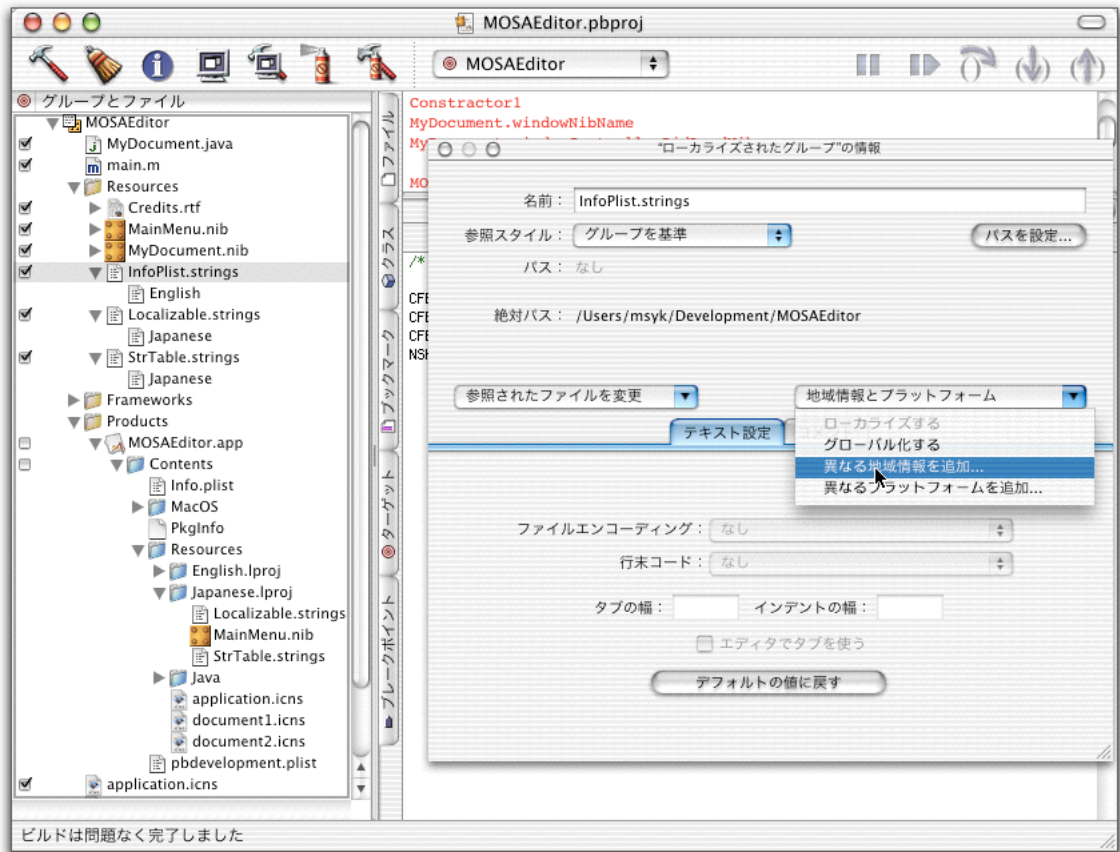
アプリケーションパッケージには、Contents/Info.plist に、プロパティリスト形式、つまり XML 形式で保存された情報があり、システムはここに設定された内容を調べて、さまざまな動作に反映させるようになっている。この Info.plist ファイルで定義されたものを「パッケージ情報」として総称するが、基本は、ターゲットの編集の画面にある「アプリケーション設定」のタブで設定する。ここではテキストフィールドなどに文字を入れるなどすれば設定ができるが、場合によっては、「詳細表示」にして、階層構造リスト形式で表示すれば、Info.plist として作られる結果そのものの編集も可能となる。

この、Info.plist による情報自体を、国際化することができる。つまり、結果的にはパッケージ情報を言語ごとに用意できるということである。そのためには、言語とのリソースとして、InfoPlist.strings というファイルを用意し、そこに言語に特有の情報を記載する。

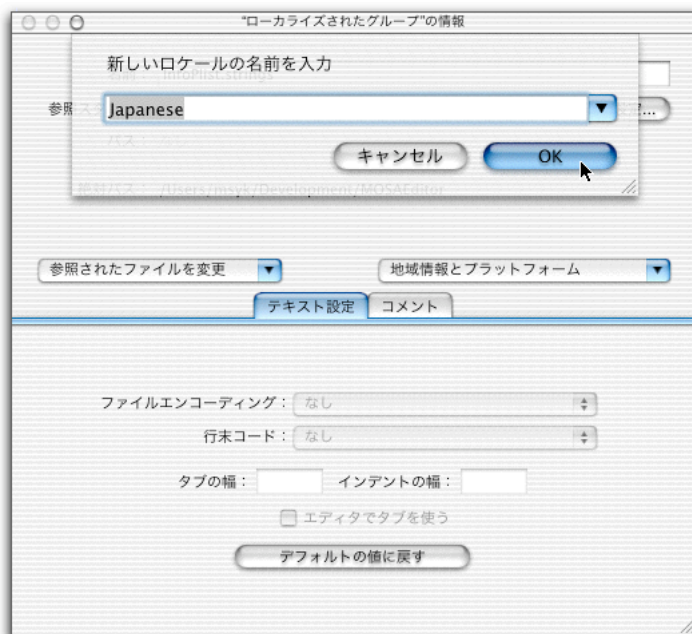
プロジェクトには最初から Resources グループに、InfoPlist.strings が存在している。この項目を選択して、「プロジェクト」メニューの「情報を表示」(Command+I)を選

択し、情報パレットを表示する。ここで「地域情報とプラットフォーム」のポップアップメニューから「異なる地域情報を追加」を選択する。そして、表示されるシートで、追加する言語として Japanese を選択すれば良い。つまり、nib ファイルの日本語向けのものを用意するのと同じ手順でかまわないということである。

InfoPlist.strings に新たな地域を追加する

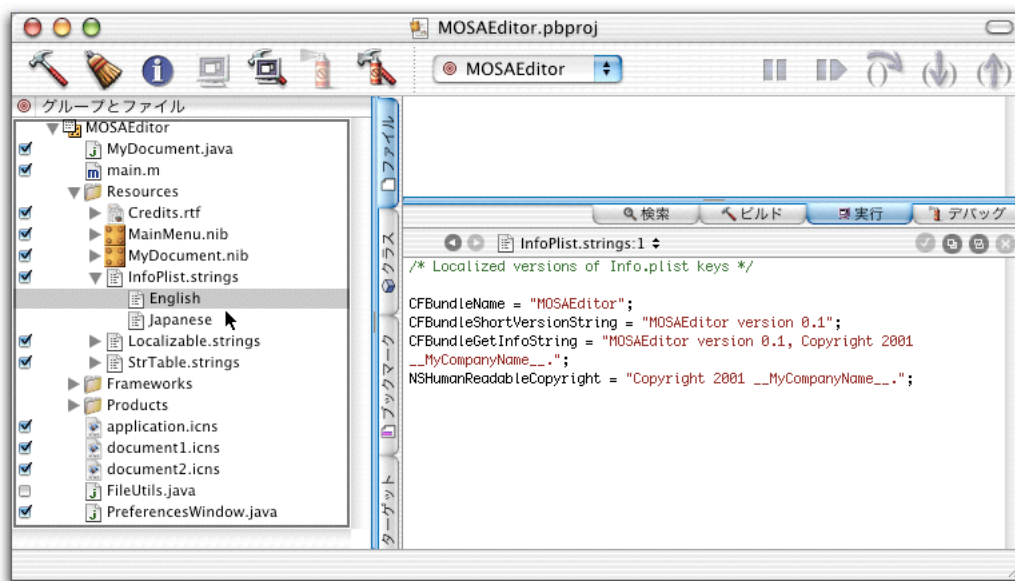


日本語を追加する



以上の作業で、InfoPlist.strings の下位項目が English だけだったのが、Japanese も増えた。English は最初から用意されているが、とりあえずは同じ内容の Japanese という項目も作られている。これらは、nib ファイルと同様で、それぞれ English.lproj フォルダの InfoPlist.strings と、Japanese.lproj フォルダの InfoPlist.strings ファイルとなっている。つまり、Project Builder で見えている階層と、フォルダ階層は逆になっている。

英語に対応した InfoPlist.strings ファイル

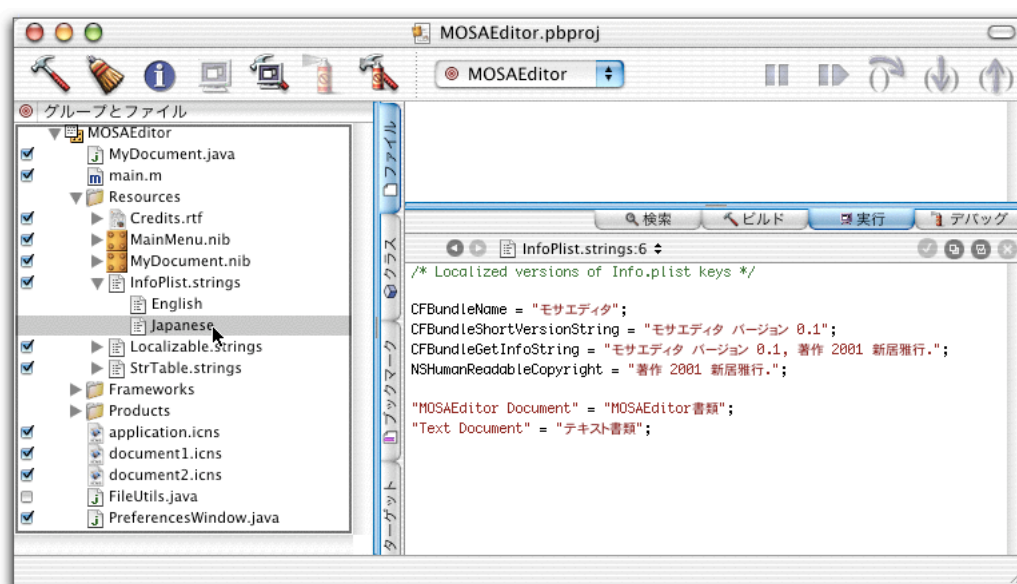


この InfoPlist.strings ファイルでは、Info.plist ファイルの XML タグで言えば <key> ~ </key> で指定するキーを左に書いて、イコールで結び、その右側に、キーの値を文

字列として記述するというのが 1 つの形態として取りうることができる。たとえば、CFBundleName というキーに対する値は MOSAEditor という文字列だが、とりあえずはこのキーに対する情報は、Info.plist には含まれていない。したがって、英語のときには、システムは English.lproj の InfoPlist.strings ファイルを参照して、このキーに対する値を取り出す。

一方、日本語に対応した InfoPlist.strings ファイルを次のように書き換えてみた。あまりにベタなカタカナが続くが、どこが変化するかを分かりやすくするために、わざとこうしてみた。CFBunldeName というキーに対する値は「モサエディタ」としている。

日本語の InfoPlist.strings ファイルを編集した



日本語の InfoPlist.strings ファイルには、イコールの左右に文字列を記載している行を追加している。これは、プロジェクトの設定にある「アプリケーション設定」で「書類の情報」として設定した文字列を思い出してほしい。そこでは、拡張子とファイルの種類の対応を指定したが、拡張子 medit に対して「MOSAEditor Document」という名前を指定した。この定義は、Info.plist ファイルに作られている。Products グループの下にあるアプリケーションのパッケージを開いていってみれば分かるはずだ。Info.plist ファイルで定義した内容の置き換えを、InfoPlist.strings ファイルで定義できるのである。つまり、言語が英語の場合は InfoPlist.strings に特に設定はない。だから、拡張子 medit に対する文書の種類を示す名前は MOSAEditor Document である。だが、このように記述すると、日本語環境では拡張子 medit に対する文書の種類名は「MOSAEditor 書類」となる。ただし、プログラム中は、言語に関わらず「MOSAEditor

Document」で参照できる。

実際にアプリケーションを起動してみよう。アプリケーションメニューの項目は、CFBundleName で指定したカタカナの名前が見えているが、メニュー項目はたぶん、MOSAEditor のままだろう。これは、メニュー項目自体は nib ファイルを修正しないといけないのである。そして、ファイルを保存するとき、ファイルフォーマットのポップアップメニューを見ると、文書の種類は InfoPlist.strings で指定した文字列になっている。

文書の種類を示す文字列が日本語化されている



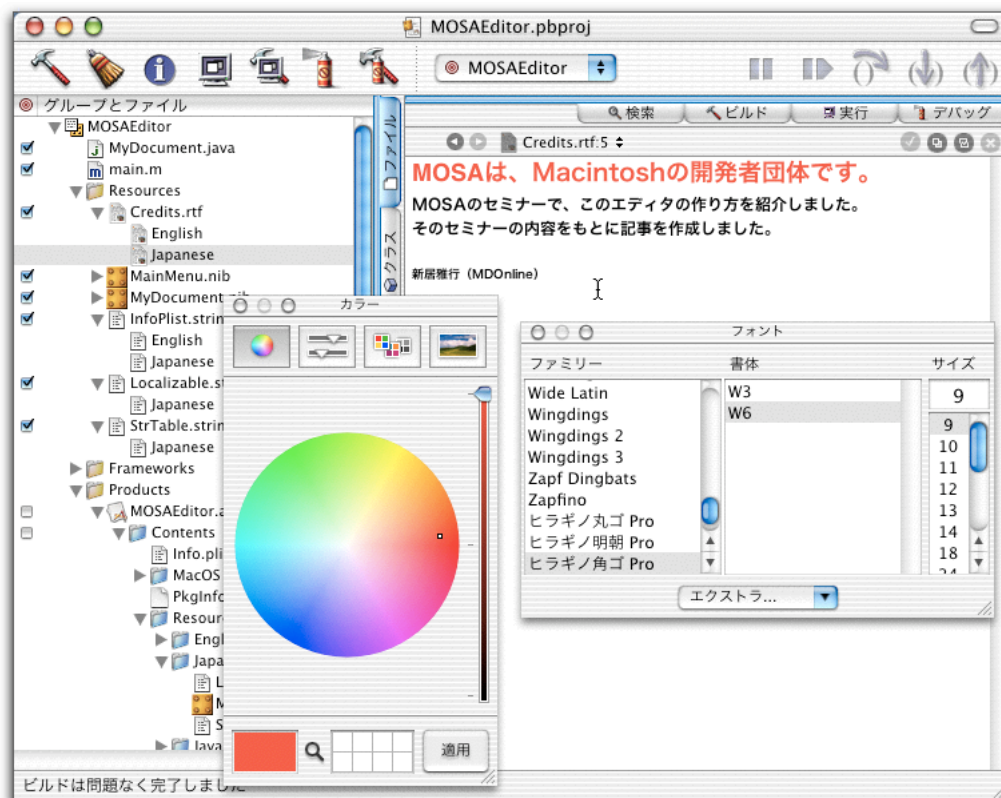
以前に説明した Localizable.strings ファイルで指定するような文字列の置き換えを InfoPlist.strings ファイルで指定するとともに、XML ファイルのキーに対する値も、ちょうど Java のプロパティファイルの形式のような記述で付け加えることができるというわけである。

アバウト画面の国際化

Cocoa では、Credits.rtf という名前でリッチテキストフォーマットのファイルを用意しておくことで、アプリケーションメニューの「○○について」を選択したときに、そのリッチテキストの内容などを持つアバウト表示ウインドウを自動的に表示する。もちろん、自分でアバウトのウインドウを凝りに凝って作ってもいいのだが、簡単な手段が用意されているので、とりあえずは手軽なところから対処しておくのがいいだろう。この Credit.rtf についても、nib ファイルや InfoPlist.strings ファイルと同じように

国際化ができる。これまでに説明してきたのと同じように、日本語の Credits.rtf を用意する。もちろん、Japanese.lproj というフォルダに Credits.rtf という名前でファイルは保存される。

日本語の Credits.rtf を編集した



こうして作成された Credits.rtf に日本語を入力するなど自由に編集してよい。もちろん、ワープロなどを使っても良いが、Project Builder でもスタイル付きでのテキスト編集が可能だ。左側で、Credits.rtf の Japanese の項目を選択し、右側で編集作業を行えばよい。つまり、ソースと同じということである。このとき、「形式」メニューの「フォント」や「カラーパネルを表示」を選択して、カラーパネルやフォントパネルを出しておいて、色や文字書式の設定ができる。

実際にアプリケーションを起動して、「MOSAEditor について」を選択すると、次のようなアバウト画面が出てくる。ウィンドウサイズは変更できないのであるが、これ以上凝った形式にしたいのなら、やはり自分でウィンドウを定義しないといけないだろう。

日本語環境で表示させたアバウト画面



ここで中心のスクロール部分には、Credits.rtf で作成した文書がそのまま見えているのが分かる。また、トップの画像は、アプリケーションのアイコンが見えている。そして、CFBundleName で指定した名前、CFBundleShortVersionString で指定したバージョン名が見えている。ここで (v0.2) と見えているのは、「アプリケーション設定」のタブで設定するバージョン番号で、キーは CFBundleVersion として Info.plist ファイルに定義されている情報だ。ウィンドウの下部には、NSHumanReadableCopyright で指定したコピーライト表示が見えている。

~~~~~この項、以上~~~~~[新居雅行]~~~~~