第1章　JBuilder の概要

【学習内容とねらい】

これから、JBuilder を用いた Java 言語プログラミングの学習を始めます。まず、本章では、JBuilder の基本操作の仕方をマスターしましょう。とは言っても現時点では「JBuilder とは何？」と戸惑う人も多いと思います。詳細は、この後すぐに体験してもらうとして、一言で言うと、Java 言語プログラムの**統合開発環境**（ソフト）と言うことになります･･･。やはりこれだけではピンと来ませんね。もう少し補足しておきましょう。皆は、すでにワープロを用いた文書作成を習得しました。具体的には MS Word を用いたと思いますが、このワープロを用いれば、文書の作成・編集そして印刷やファイルへの保存など、文書作成に必要な操作を全て処理できたはずです。あえて言えば、ワープロは文書作成の統合開発環境ソフトと言って良いでしょう。それと同じように、JBuilder は Java 言語プログラム開発に必要な全ての処理が備わったソフトだと捉えて下さい。これで少しはイメージがつかめたでしょうか。

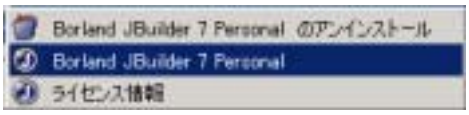
さて、実際の学習に入る前に次の点を頭に入れておきましょう。プログラムは、大まかには「プログラムの記述」→「コンパイル（翻訳）」→「実行」という過程を経て実行結果を得ます。ここで注目して欲しいのは、「**コンパイル（翻訳）**」という過程が必要である点です。Java 言語に限らず、プログラミング言語を用いて記述したプログラムは、コンピュータには理解不能です。そこで、コンピュータが理解できる（処理できる）**機械語**というコードに変換する事が必要となり、その処理をコンパイル（翻訳）と言います。機械語に変換されたコードは所定の実行命令によって実行されます。このような開発過程の大まかな流れを ” 常識 ” として頭に入れておくと、以下の学習がスムーズに行きます。

それでは、本章で学ぶのは今後必要となる基本操作ですので、しっかりと身につけて下さい。なお、1-4 節では、統合開発環境を用いずに、**手作業で** Java 言語プログラムを作成、翻訳・実行するという体験を行ってもらいます。こうすることによって本来必要な作業が JBuilder を用いることによって、いかに軽減されているかが理解できると思います。

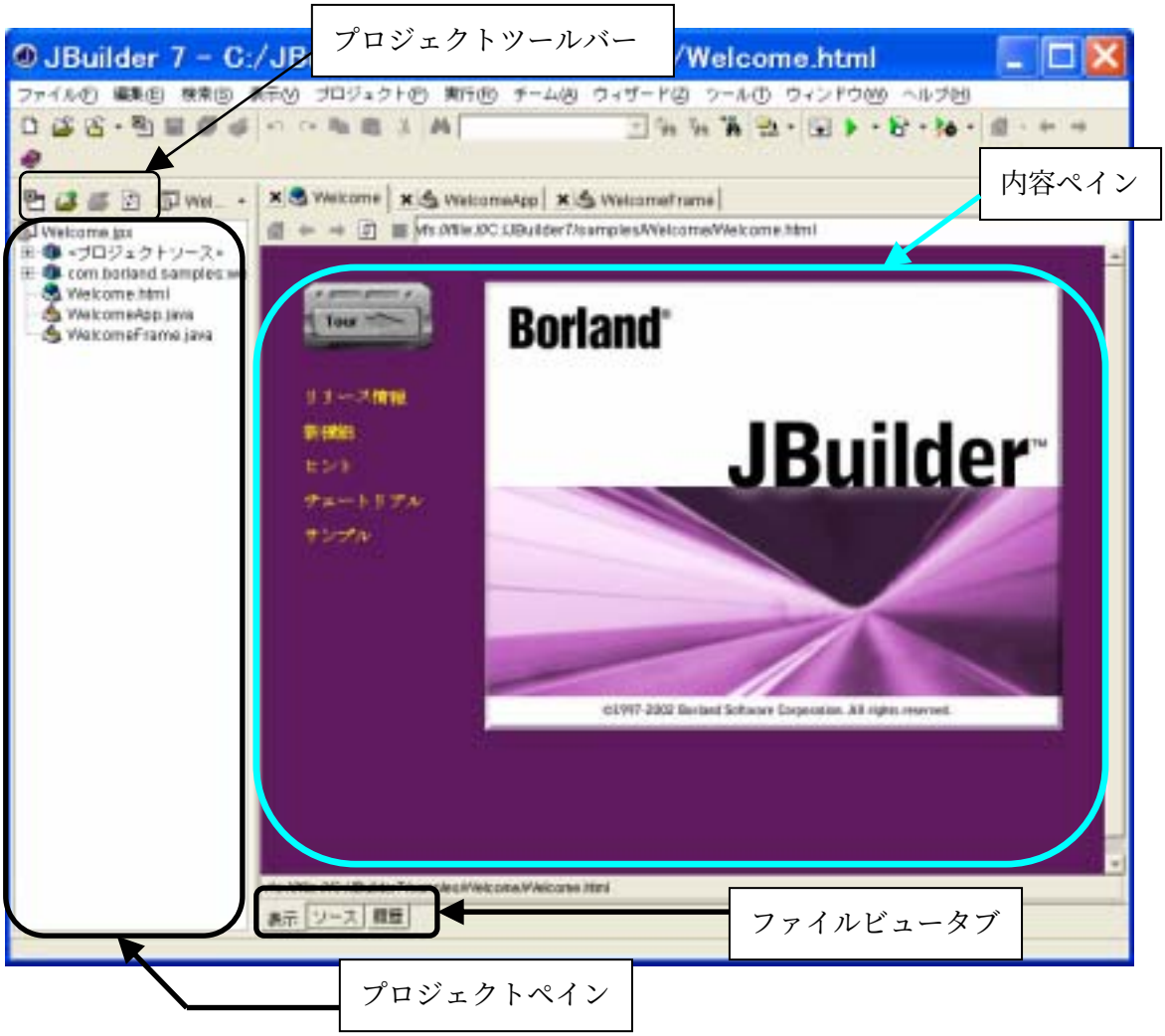
＜第１章の構成＞

1-1 JBuilder の概観

まず、JBuilder を起動しましょう。［スタート］メニューから［プログラム］→［JBuilder 7 Personal］を選択し、現れたメニューから［JBuilder 7 Personal］を選択します。



すると、下の様な JBuilder の起動画面（AppBrowser「**アプリケーションブラウザ**」と呼びます）が現れます。これは初めて JBuilder を起動したときの場合です。一般には前回作成したプログラムの画面が現れます。



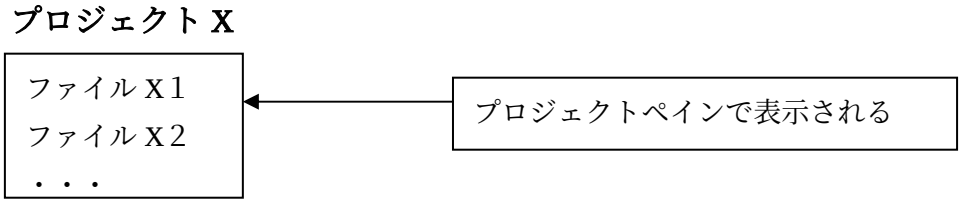
まずは、これからプログラムを作成するに当たって、**アプリケーションブラウザ**の構成を眺めておきましょう。JBuilder では、この一つの画面でプログラムの作成や実行等のす

べての行程を行えるように設計されています。そのために、画面を機能別にいくつかの領域に分割しています。JBuilder に慣れて行くためにそれらを順に見て行きましょう。

まず、画面の中央が「内容ペイン」と「プロジェクトペイン」の二つの領域に分割されているのが分かると思います。ペイン(pane)とは聞き慣れない言葉ですが直訳すると「１区画」という意味になります。JBuilder では作業画面の一区画を「ペイン」と呼んでいます。

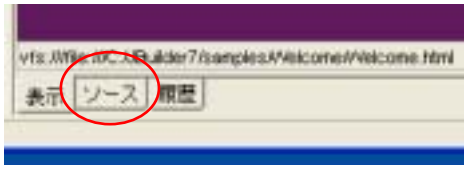
プロジェクトペイン

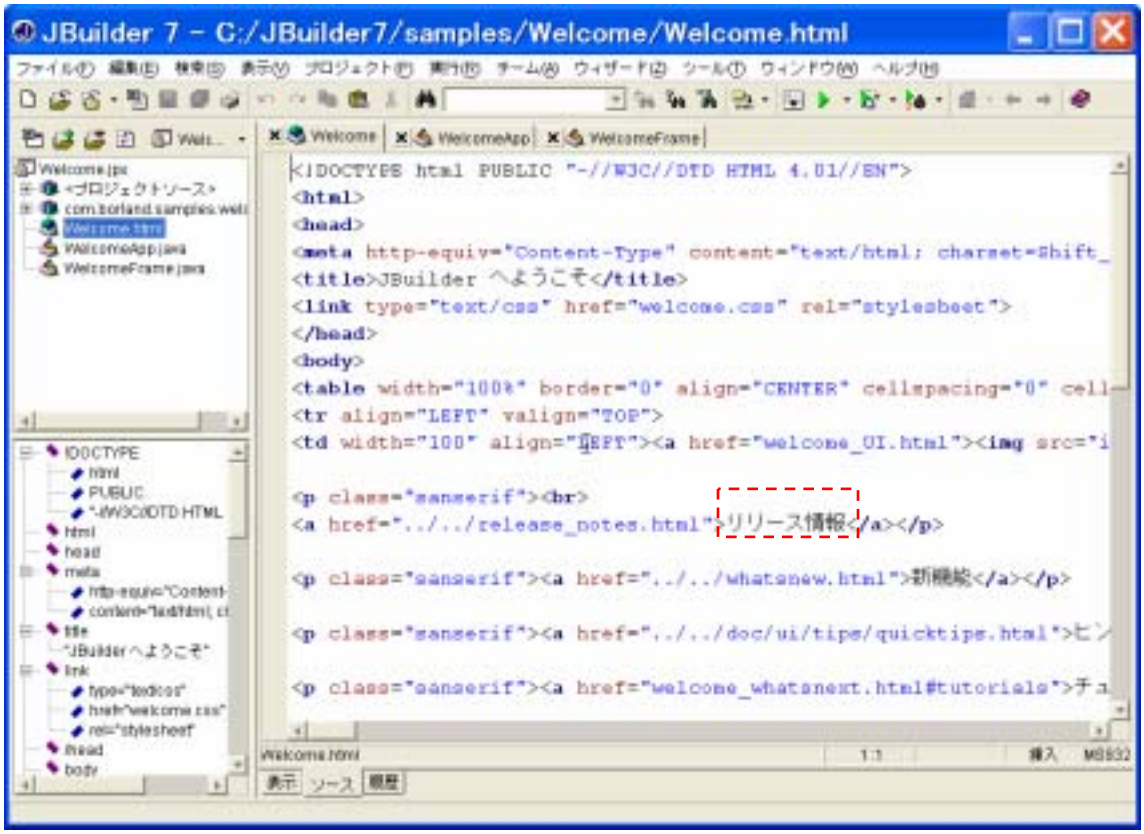
さて、左側の「プロジェクトペイン」を見てみましょう。ここには、現在のプロジェクトに登録されたファイルの一覧が表示されます。「プロジェクト」と言ってもまだピンと来ないと思いますが、ここではとりあえず、プログラムを実行させるのに必要なファイルのグループと捉えておいてください。1-3 節で説明するように、JBuilder では一つのプログラムを動作させるのに複数のファイルを必要とします。そしてそれらを総動員して一つの処理を実行するのです。まさに一つの目的に対してプロジェクトチームを組んで事に当たるのに似ています。各ファイルの詳細は現時点では気にする必要はありません。



内容ペイン

次に内容ペインを見てみましょう。内容ペインは、プロジェクトペインで選択されたファイルの内容を表示するところです。初めて JBuilder を起動した場合、前ページの図のように、（JBuilder にあらかじめ用意されている）「Welcome」プロジェクトの中の「Welcom.html」ファイルが選択された状態になっています。JBuilder は Web のブラウザの機能を内蔵しているので、このように HTML ファイルを Web ブラウザのように表示することができるのです。ここで、内容ペイン下方の「ファイルビュータブ」から「ソース」タブを選択（クリック）すると、内容ペインは、次ページに示すように HTML ファイルのソース表示に切り替わります。



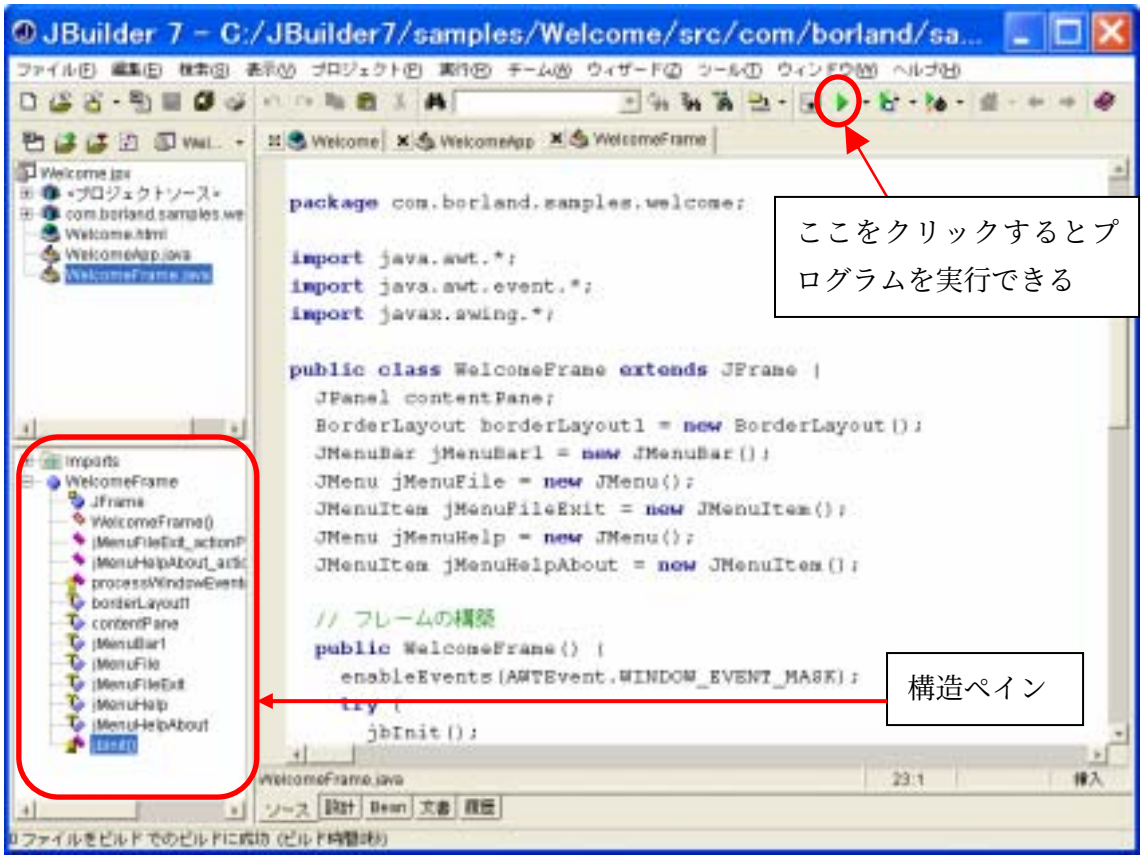


このように、ソースが表示された状態では、内容ペインが「ソースコードエディタ」となり、ソースの編集を行うことができます。試しに、少し”いたずら”してみましょう。上の点線枠で囲んだ部分は「リリース情報」と表示させる部分ですが、ここを「リリース情報だよ」という文に修正してみてください。修正後、ファイルビュータブの「表示」を選んで表示ビューに戻ってみましょう。すると今度は、次のように一番上の項目が「リリース情報だよ」と変わっているはずです。まぁ、何の意味もない変更ですが、ソースを編集できること、そしてその編集結果が反映されていることが確認できたら OK です。

構造ペイン

それでは、せっかくですから、次に Java 言語のソースプログラムを見てみましょう。プロジェクトペインから、「WelcomeFrame.java」を選びダブルクリックしてみてください。

すると、内容ペインに次のようにプログラムのソースリストが表示されます（ソースリストの上方には著作権関係のコメントが書かれているので、下図では若干下方にスクロールさせています）。

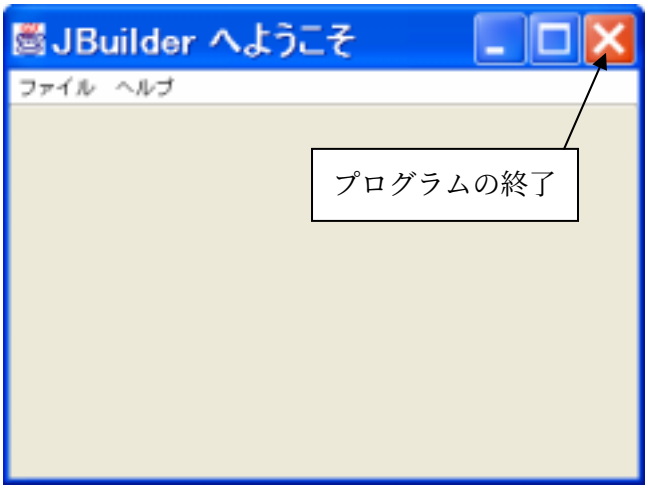


今の時点では、何が書かれているのか全く分からないと思います。でも、これから徐々に分かるようになるので、その点は心配いりません。ここでは、アプリケーションブラウザの機能に注目することにしましょう。先ほど HTML ファイルのソース表示を見た際に気づいた人がいるかもしれませんが、ソース表示になると、上のように「構造ペイン」という新しいペイン（区画）が現れました。構造ペインでは、プログラムの階層構造を視覚的に表示してくれます。また、リストアップされている項目をクリックすると、その該当部分にジャンプするようになっています。例えば構造ペインの「WelcomeFrame()」項目をクリックすると、その定義を記述した部分にジャンプします。現時点では、この程度のことを把握しておけば十分でしょう。

プログラムの実行

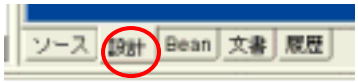
さて、このサンプルプログラムは一体、何を行うプログラムなのでしょうか？とりあえずプログラムを実行してみましょう。実は、（内容は分からなくても）プログラムの実行は簡単です。ソースコードエディタ（内容ペイン）の上方にある緑色の右三角ボタン  を選んでクリックしてください（これは、Java プログラムを翻訳して実行する命令です）。しばらくすると、次のようなウィンドウが現れます。

実は、このプログラムは、ファイルメニューから「終了」を選んでプログラムを終了させるだけの単純なプログラムだったのです。なお、通常のウィンドウアプリケーションの様に、右上隅の［×］をクリックしてもプログラムを終了させることができます。いったんプログラムを終了させてください。

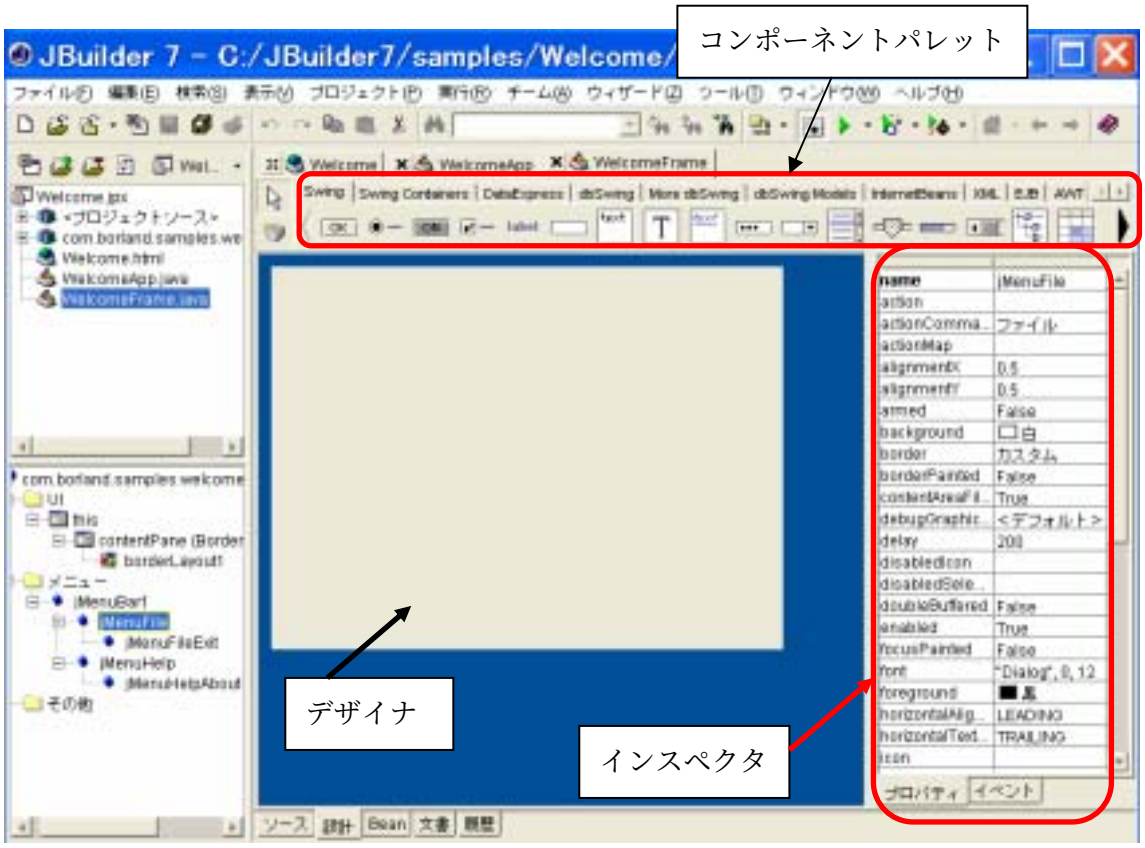


設計ビュー

もう少し、このサンプルプログラムを使って、JBuilder のアプリケーションブラウザの機能を眺めることにしましょう。今度は、ファイルビュータブから「設計」タブを選択します。すると、次ページのような画面に切り替わる、つまり**設計ビュー**に切り替わります。



この設計ビューを眺めながら簡単に説明しましょう。設計ビューでは、内容ペインに**デザイナ**と呼ばれる矩形領域が配置されています。この上に、ボタンやメニューなど Windows アプリケーションに必要な ” 部品 ” を配置して行くのです。Windows アプリケーションを作成する場合、ボタンやテキスト入力欄等の **GUI**（Graphical User Interface）は不可欠です。JBuilder では、これら GUI の各部品のことを**コンポーネント**と呼びます。デザイナは、GUI コンポーネントを配置する、つまりデザインする作業台の様なものなのです。そして利用可能なコンポーネントは内容ペイン上方の**コンポーネントパレット**に納められています。具体的な使い方は次節以降で学習します。ただ、一点だけ先取りして、内容ペインの右側にある**インスペクタ**について簡単にふれておきましょう。ボタンなどのコンポーネントはその色やボタン上に記される表記文字など、いくつかの性質を持っています。これらは**プロパティ**と呼ばれますが、各コンポーネントについて定義可能なプロパティがこのイ



ンスペクタに納められています。そしてインスペクタ内の各欄の値を変更するだけでコンポーネントの該当するプロパティを変えられるようになっています。インスペクタを眺めると、プロパティの他に「**イベント**」というタブがありますね。これについては、第３章で詳しく説明する事にしましょう。

以上の様に、設計ビューには、コンポーネントを貼り絵のように配置することで Windows アプリケーションを設計できるよう機能が備わっています。具体的な操作は第２章で学習します。なお、このような**デザイン機能**は JBuilder 独自の機能です。”独自 ”というのは、Java 言語とは（基本的には）関係のない部分であるという意味です。Java 言語を用いたプログラムの記述はソースビューで行うことになります。したがって、皆はこれから以下の２点を学習することになります（きわめて大まかに言うと、ということですが・・・）。

① 設計ビューによる GUI のデザイン

② ソースビューによる Java 言語プログラミング

まず、本章と次章で①の操作に慣れた上で、第３章以降で②の Java 言語の本格的な学習に入ります。今後の学習の流れがイメージできたでしょうか？

ヘルプ

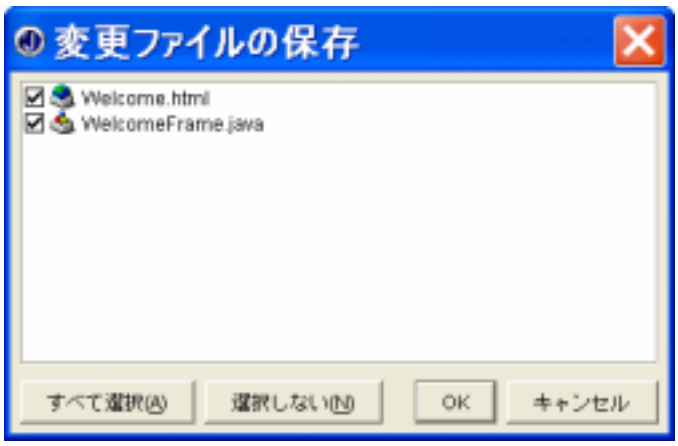
早く実際のプログラミングに入りたくてウズウズしているかもしれませんが、この節の最後に、JBuilder のヘルプメニューについて触れておきたいと思います。次の図は、ヘルプメニューを開いたところです。

分からない用語に行き当たったり、文法事項等で腑に落ちない事があったりしたら、ヘルプメニューから検索して該当事項を調べることを勧めます。最初は、求める情報になかなか行き当たらなかったり、またあるいは該当する記述を見つけても、意味がよく分からなかったりと、苦労する事が多いと思いますが、ヘルプで調べる習慣を身につけると、急速にプログラミング力がつきます。どうか、この講義が終わる頃には、ヘルプで疑問点を調べることができるようになることを目標に置いてください。ところで、ヘルプ活用に関して次のような一学生（皆の先輩）の例があります。彼は、プログラムは得意な方ではなかったのですが、卒業研究でソフトウェア制作をテーマに選んだため、必要に迫られてヘルプメニューで疑問点を調べるようになりました。数ヶ月するとその効果が現れ、大体のことは、ヘルプメニューおよび Web 上の情報で技術的な問題をクリアーできるようになりました。その後、みるみる力をつけた彼は在学中に制作した２つのソフトウェアがフリーソフトの Web サイトに掲載されるまでに成長しました。どうやら、ヘルプを活用できるようになってその才能が開花したようです。

なお、調べたい項目（ソースビューにおける Java 言語の命令や、設計ビューにおけるコンポーネント等）にカーソルを合わせ［**F1**］キーを押しても、該当するヘルプメニューが現れます。大変便利なのですが、コンポーネントやプロパティに関するヘルプは英語のままになっています。これが活用できるようになると立派なのですが・・。ともかく、「ヘルプメニューで問題を解決できるようになったら一人前」であると考えて結構です。皆も一

人前を目指して欲しいものです。

さて、ここで、JBuilder の開発環境の概観をいったん終えましょう。JBuilder のアプリケーションブラウザを閉じてください（右上隅の×をクリックすれば良いです）。このとき、プログラムの保存の有無を確認する、以下のようなダイアログボックスが現れます。



ここでは、変更を保存する必要がないので、［選択しない］をクリックした後［OK］ボタンをクリックして、JBuilder を終了してください。

1-2 アプリケーションの作成･実行･保存

【準備】 フォルダ「ProgJava」の作成

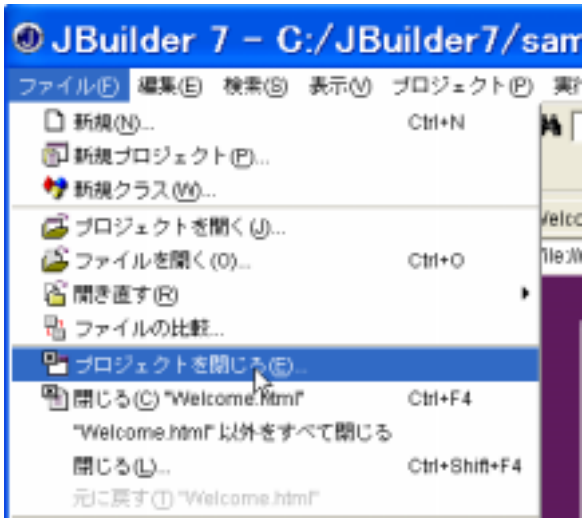
本節では、新たにプログラム（アプリケーション）を作成する際の手順を学習します。その際に、作成したプログラム（今後**アプリケーション**と呼びます）を保存するためのフォルダを作成しておく必要があります。各自、自分の PC の適当な場所（例えばマイドキュメントの中）に「**ProgJava**」というフォルダを作成してください。ここには、今後プログラミングで作成する課題プログラムが保存されることになります。以下の説明では、プログラムは全てこの「ProgJava」内に保存するものとして説明を進めます。

【基礎課題 1-2-1】

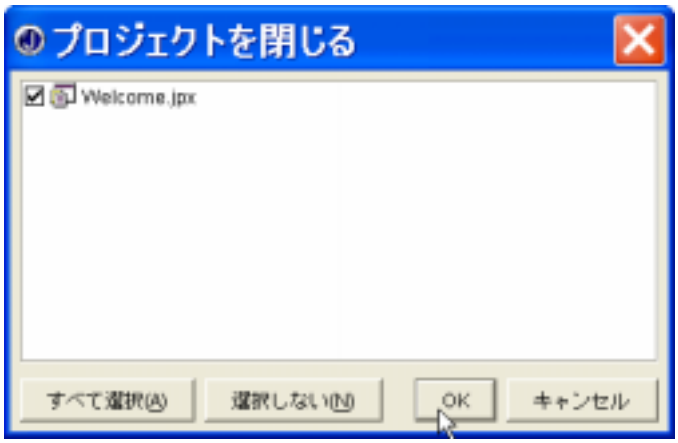
以下の指示に従って、アプリケーションの作成・実行・保存を行ってください。

Ⅰ. 既存のプロジェクトを閉じる

JBuilder を起動してください。再び（前回作業した）「Welcome」プロジェクトが開きますが、「ファイル」メニューから「プロジェクトを閉じる」を選択し、それを閉じましょう。



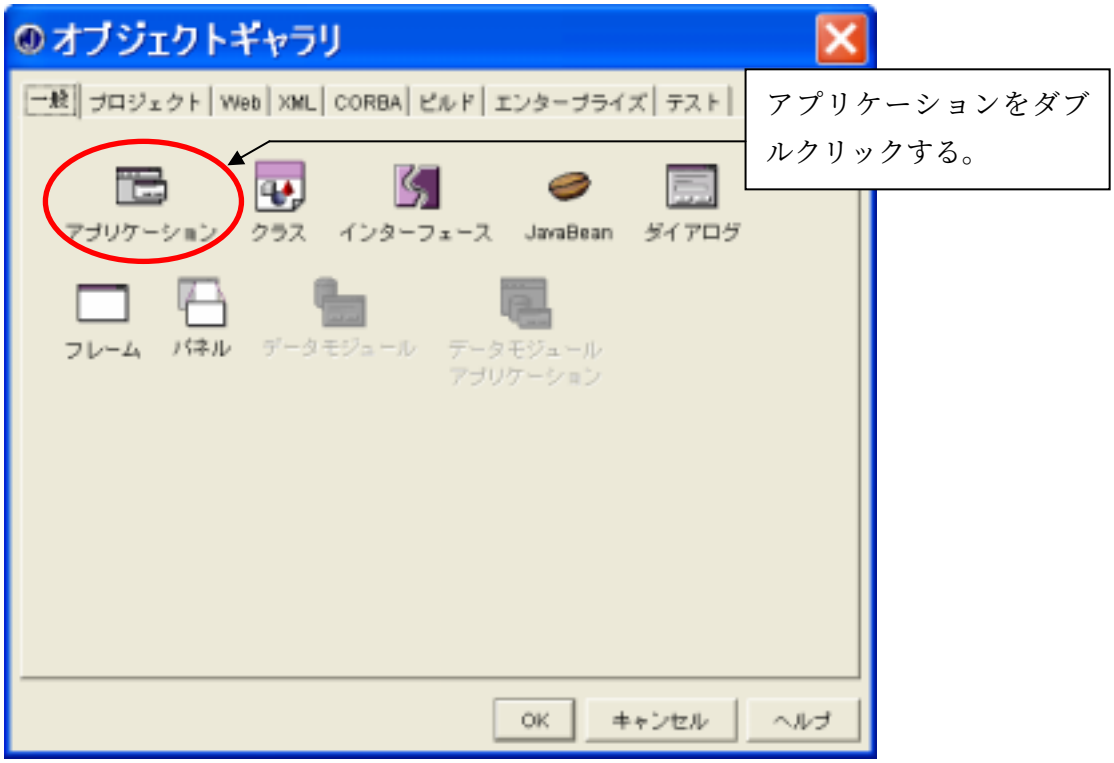
その際、閉じるプロジェクトを確認するウィンドウ（ダイアログボックス）が現れますが、ここで全てのプロジェクトがチェックされている事を確認して［OK］ボタンをクリックします。



すると、プロジェクトが閉じられ、アプリケーションブラウザには何も表示されていない状態になります。新たにプロジェクトを作成する際には、いつもこのように既存のプロジェクトを閉じてから、作業を行うようにすると良いです。

II. オブジェクトギャラリからアプリケーションを選択する

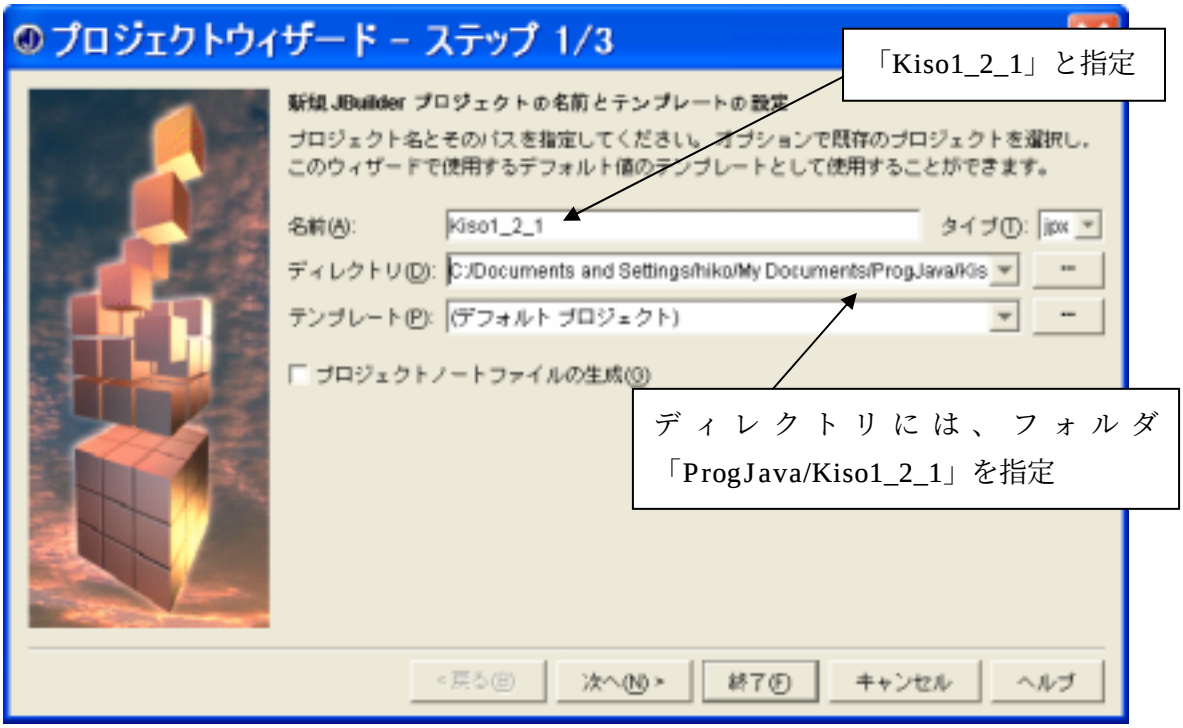
次に、 [ファイル]メニューから［新規］を選択してください。すると、次のような「オブジェクトギャラリ」が現れます。ここには、アプリケーションやプロジェクトなどを作成する際の各種ウィザードが用意されており、メニューに答えるだけで、作成に必要な環境を生成してくれます。ここでは、「アプリケーション」を選択し、ダブルクリックしてください。



III. プロジェクトウィザードによるプロジェクト情報の設定

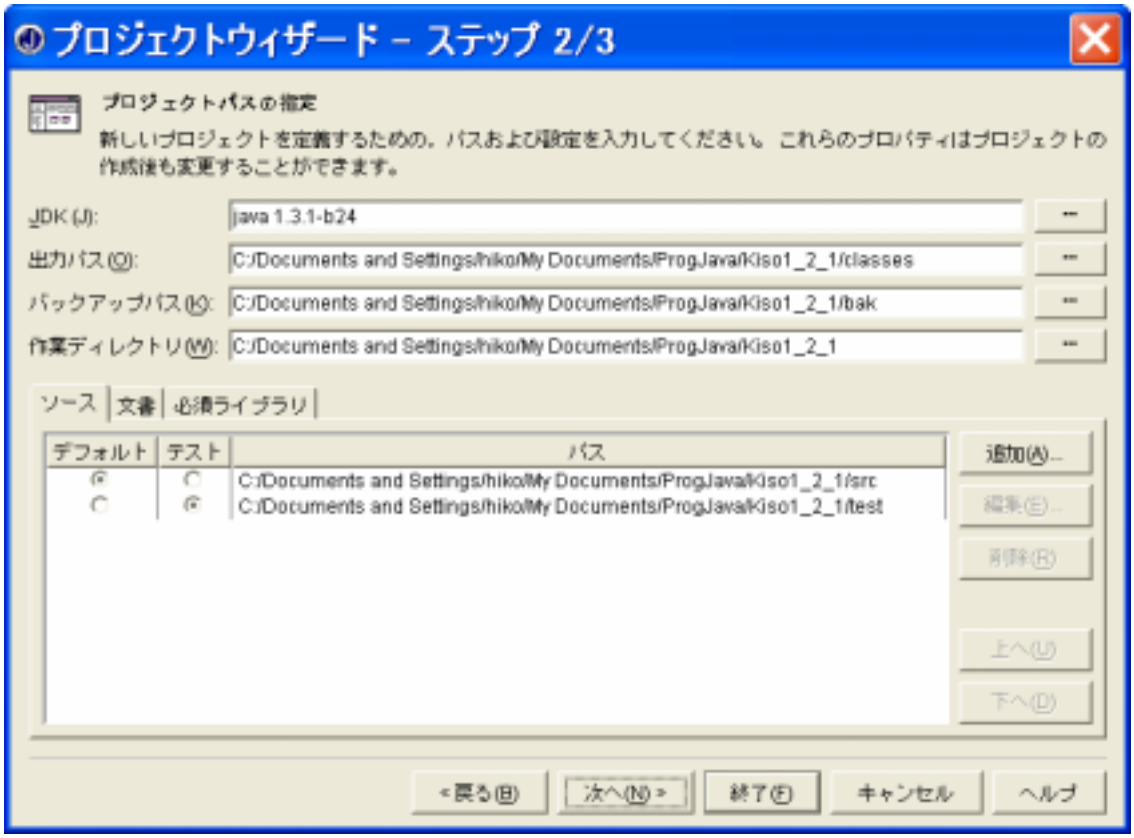
すると、次のようなプロジェクトウィザードが起動し１ページ目が現れます。ここでは、プロジェクトの名前を指定するのですが、基礎課題 1-2-1 に対応して「Kiso1_2_1」としておきましょう。今後、プログラムを作成する際には、基礎課題の番号に合わせて命名するようにしてください。

そしてディレクトリには、プログラムを保存する場所を指定します。つまり、先ほど作成した「ProgJava」を指定します。右端の［…］ボタンをクリックすると、フォルダ「ProgJava」フォルダの場所を探索する事ができます。その際、「ProgJava」を選択しさえすれば、「…/ProgJava/Kiso1_2_1」という様に、ProgJava 内に自動的にプロジェクトの名前が付いたフォルダが作成されるようになっています。

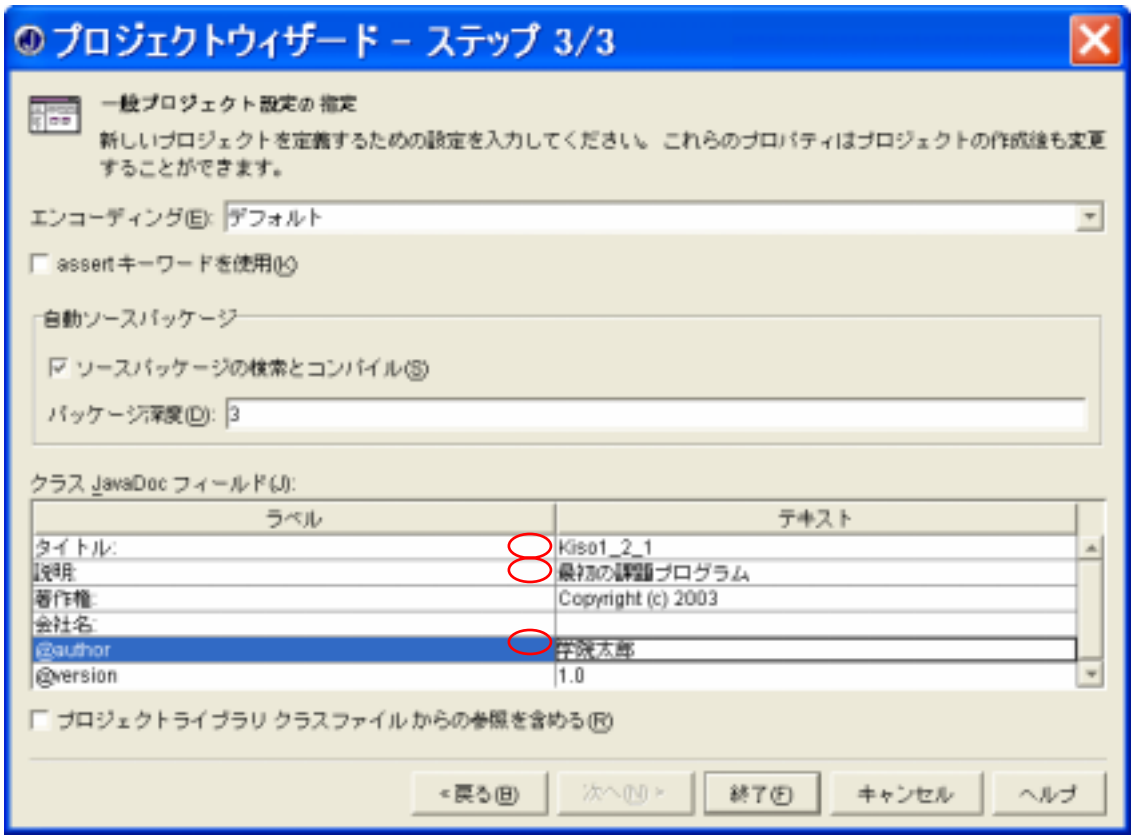


上のように指定後、［次へ］ボタンをクリックします。

すると、２ページ目の画面が現れますが、ここは自動的に生成された情報を変える必要はありませんので、［次へ］ボタンをクリックします。



すると、プロジェクトウィザードの最終ページが現れます。ここでも、何も記入しなくても結構なのですが、下のように、「タイトル」、「説明」そして「@author（制作者）」の欄に記入しておくと、後のために便利です。後から分かりますが、ここで記入した情報はコメントの形で、プログラムファイルに書き込まれます。適当に記入後［終了］ボタンをクリックしてください。



III. アプリケーションウィザードによるアプリケーションのひな形の作成

続いて、次のようなアプリケーションウィザードが起動します。これは、アプリケーションのひな形を作るためのウィザードです。１ページ目では、「ヘッダーコメントの生成」欄をチェックしておきましょう。その他は特に変更する必要はありません。［次へ］ボタンをクリックしてください。

2ページ目は、アプリケーションのウィンドウ（Java では**フレーム**と呼びます）に関する設定を行います。ここでは、フレームのタイトルを「Kiso1_2_1」としておきましょう。フレームタイトルは、アプリケーションのウィンドウ最上部のタイトルバーに表示されるメッセージです。ですから各自の判断で適当な内容に変更しても構いません。

［次へ］ボタンをクリックすると、下の最終画面が現れます。ここでは何も記入する必要がないので、［終了］ボタンをクリックしてください。これで、アプリケーションのひな形が作成されました。

Ⅳ. アプリケーション･プログラムの編集･実行

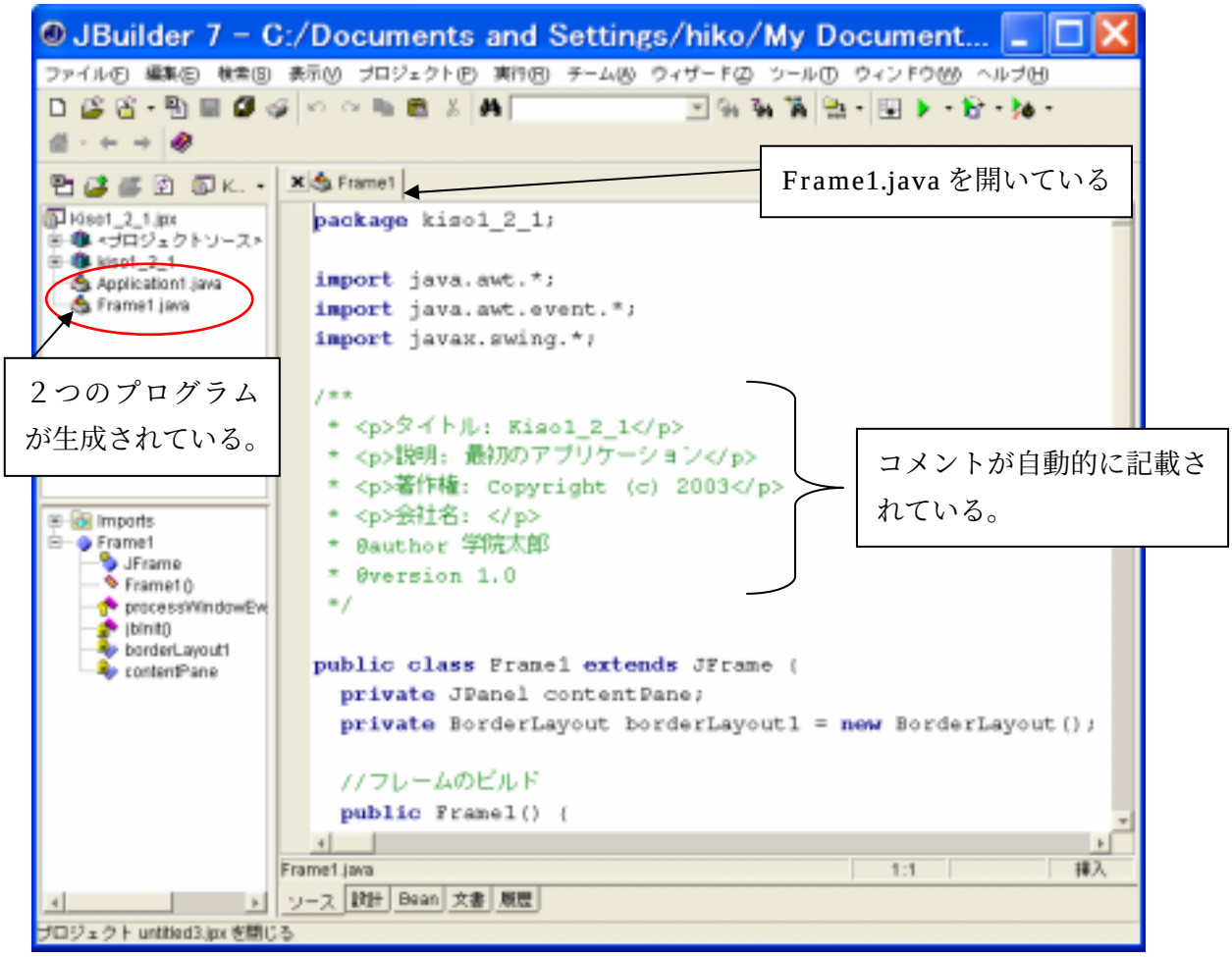
ウィザードによる設定終了後、アプリケーションブラウザには、次ページのように「Frame1.java」のプログラムが表示されているはずです。先ほどプロジェクトウィザードで指定した、アプリケーションのタイトルや説明等が、コメントとして挿入されている事が分かるでしょう（コメントの記述の仕方については、第４章で説明します）。

ここで、左上方のプロジェクトペインを見てください。このプロジェクトには「Application1.java」、「Frame1.java」という**二つのプログラム**がある事が分かります。これは、アプリケーションウィザードによって自動生成されたプログラムです。なぜ二つあるのか疑問に思うかもしれませんが、現時点では二つのファイルの意味を次のように捉えておいてください。

★　Frame1.java：アプリケーションで使用するフレーム（ウィンドウのことです）の設計および（必要な）処理内容を記述するプログラム

★　Application1.java：Frame1.java で定義したフレームを生成するプログラム

通常、皆が作成・編集を行うのは、Frame1.java のプログラムの方になります。詳細は徐々に分かってきますので、とにかく、慣れることにしましょう。



さて、ここで、ファイルビュータブから「設計」を選択しましょう。すると、前節のサンプルプログラムでみた通り、フレームのデザイナが現れます。まだ、何もコンポーネントを配置していないので、まっさらなフレームのままです。

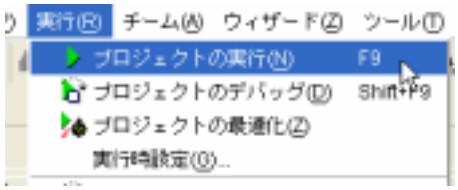
この状態でプログラムを実行してみましょう。実行は前節で説明した通り、内容ペイン上方の緑色の右三角ボタンをクリックすればよいのです。すると、次のようなウィンドウが現れます。これは、ただウィンドウを出現させるだけのプログラムですが、それでも立派なアプリケーション・プログラムです。

実行できることを確認したらウィンドウを閉じてプログラムを終了させてください。

なお、プログラムを実行させるには以下の３つの方法があります。

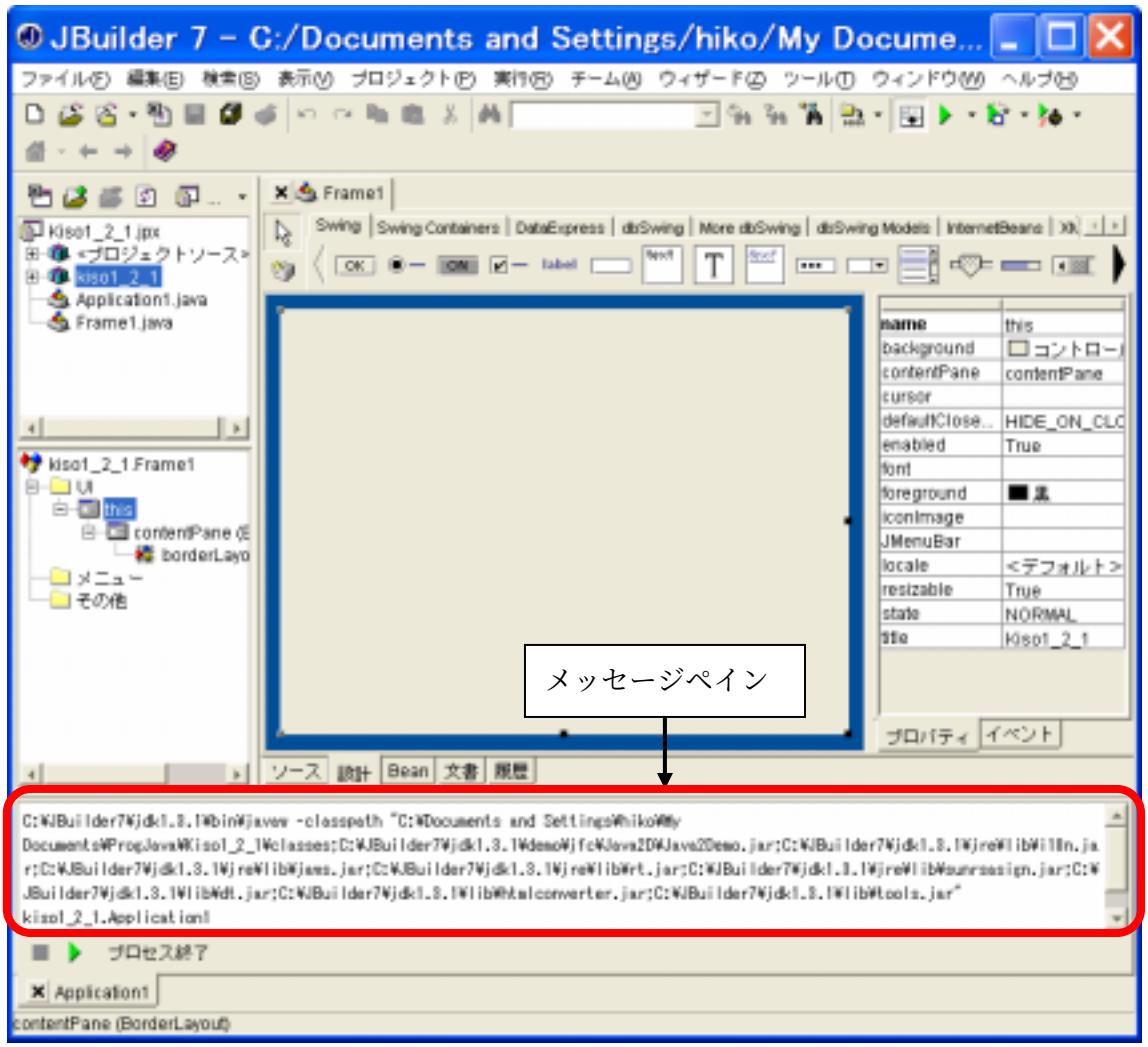
① ボタンをクリックする。

② ［実行］メニューから「プロジェクトの実行」を選択する。



③ ［F9］キーを押す。

さて、プログラムを実行すると、次のように、内容ペインの下方にメッセージペインが現れました。

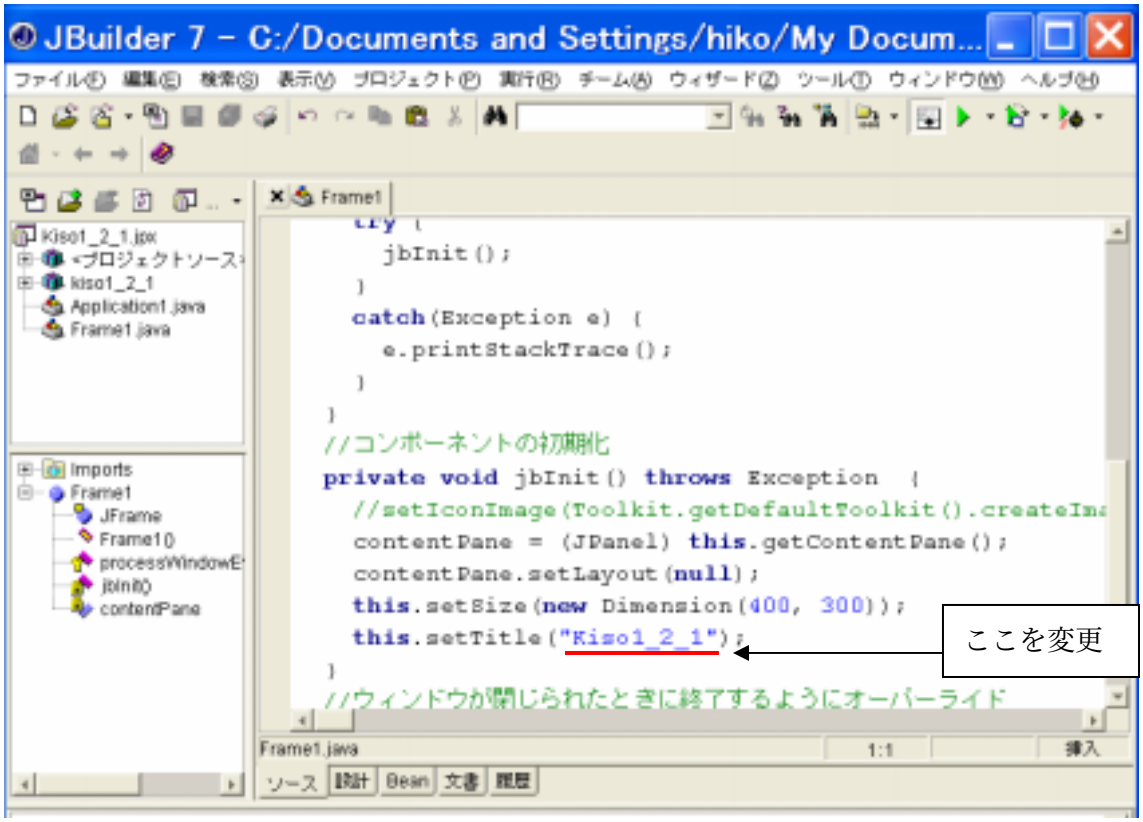


ここには、アプリケーション実行時の様々な情報が表示されます。ソースプログラムにエラーが含まれている場合には、エラーメッセージが表示されます。それについては、今後必要に応じて説明して行きます。

ここで、ちょっとだけプログラムを編集してみましょう。内容ペインを設計ビューからソースビューに変更し、下方にスクロールさせて

```
this.setTitle("Kiso1_2_1");
```

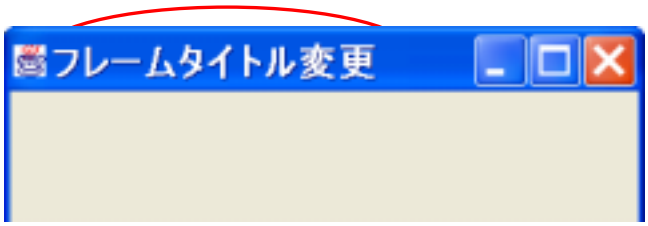
という部分を見つけてください。



ここを以下のように変更してみましょう。

```
this.setTitle("フレームタイトル変更");
```

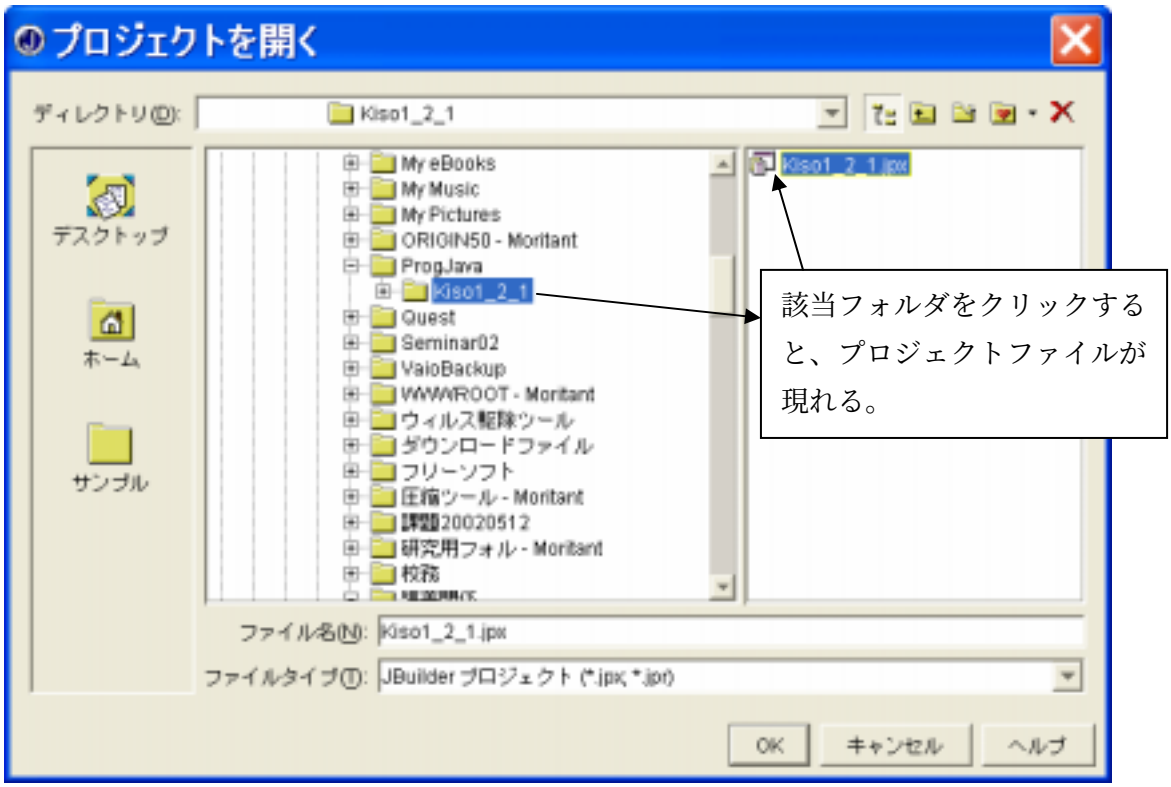
予想がつくと思いますが、ここは、フレーム（ウィンドウ）のタイトルを指定しているところです。修正後実行すると、以下のようにタイトルバーのタイトルが変更されているはずです。



これで、少しプログラムをいじったことになります。そこで、プログラム編集を終えたものとしてプロジェクトを閉じましょう。［ファイル］→［プロジェクトを閉じる］を選択してプロジェクトを閉じてください。

1-3 JBuilder の生成するファイル

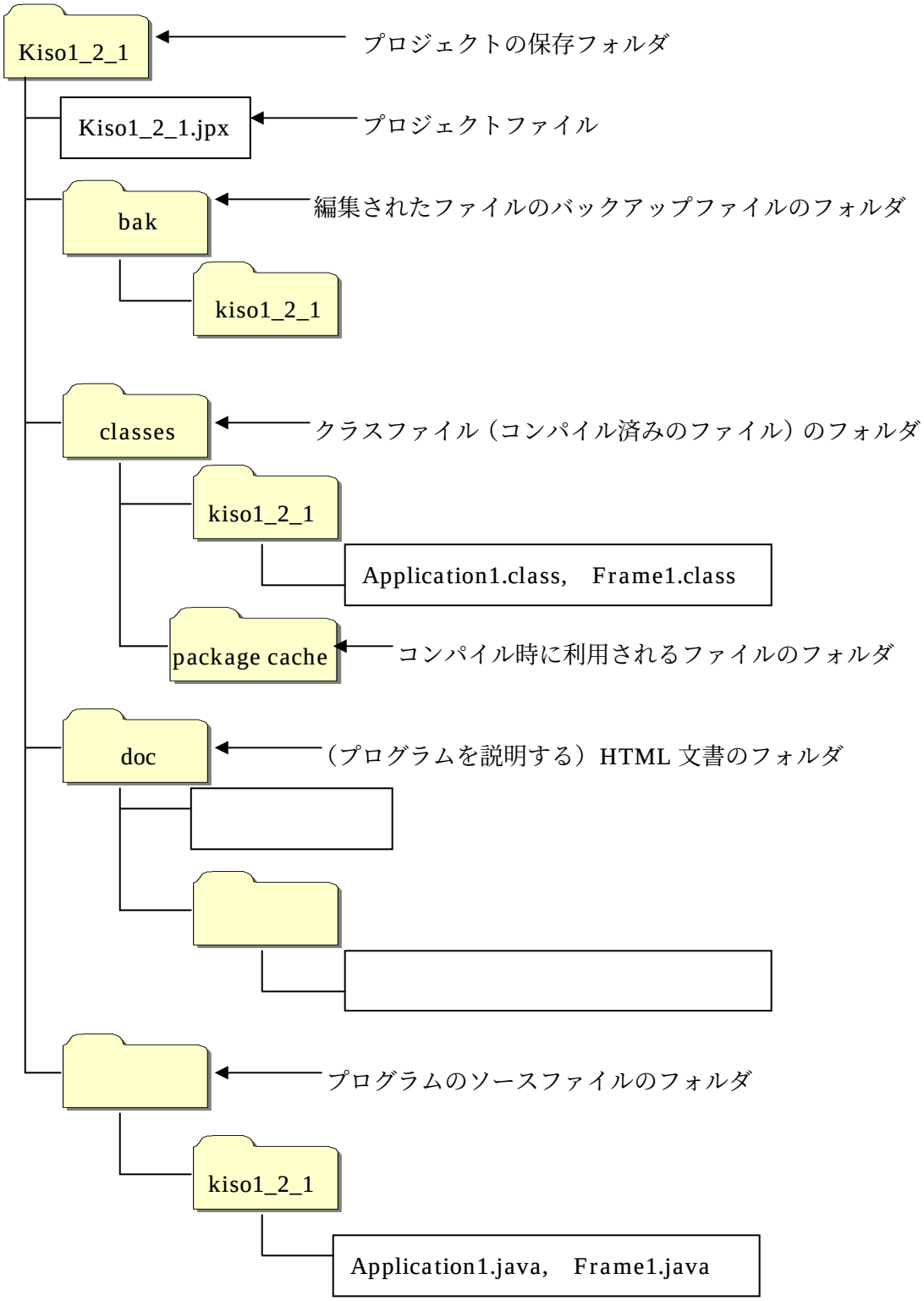
一度作成したプロジェクトファイルは、［ファイル］→［プロジェクトを開く］によって、開くことができます。試しに、前節で作成したプロジェクト「Kiso1_2_1」をもう一度開いてみましょう。［プロジェクトを開く］を選択すると、「プロジェクトを開く」画面が現れます。ここで、以下のようにフォルダ「Kiso1_2_1」を探してクリックしてください。



すると、右側の画面に「Kiso1_2_1.jpx」というファイルが現れます。これがプロジェクトファイルです。このファイルを選択し［ＯＫ］ボタンをクリックすると、再びプロジェクト「Kiso1_2_1」のプログラムが表示されます。

ここで、JBuilder が生成するファイルを眺めておきましょう。プロジェクト「Kiso1_2_1」のファイル構成は次のようになります。この中で重要なファイルは、上のプロジェクトファイルに加えて、クラスファイル（Application1.class と Frame1.class）とソースコード（Application1.java と Frame1.java）です。ソースコードは Java 言語のプログラムですから、当然必要だとしても、クラスファイルとは何でしょうか？次ページの図中では、「コンパイル済みのファイル」と記していますが、それだけではよく分かりませんね。そこで、その点について少し次節でふれておきましょう。

＜ JBuilder のプロジェクトのファイル構成 (Kiso1_2_1 の場合) ＞

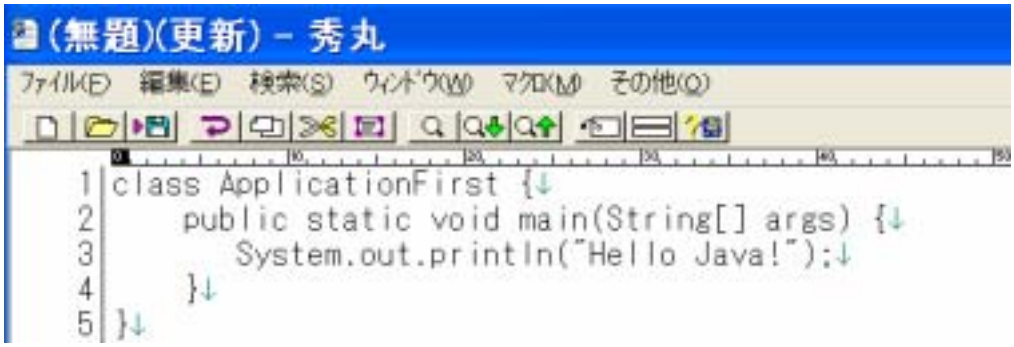


1-4 Java 言語プログラムの翻訳･実行（コンソールバージョン）

JBuilder はプログラミングの手間を省くために実は背後で様々な作業を行ってくれています。そのため、このようなアプリケーション・プログラム開発環境を **RAD**（Rapid Application Development）と呼んでいます。プログラムを迅速に（Rapid）開発できる環境という訳です。これは大変便利な開発環境なのですが、Java 言語プログラミング開発の流れあるいは仕組みを理解するためには、本来どのような作業が必要なのかを理解しておく必要があります。そこで、本節では、JBuilder を用いずに ”**手作業**” で簡単なプログラムを作ってみましょう。JBuilder などの RAD が開発される以前は全てのプログラマが行っていた作業を体験してみるわけです。以下の手順にしたがって、プログラムの作成・実行を行い、”プログラマ” の気分を味わってください。そうすれば、JBuilder の便利さ、有り難さも分かってくるでしょう。

Ⅰ. プログラムの作成

秀丸エディタやメモ帳などのエディタを起動させ、以下のプログラムを記述してください。



プログラム入力は全て半角英数文字で、空白の位置などは大体上の通りであれば結構です。

何も意味が分からなくては釈然としないでしょうから簡単に解説しておきます。Java 言語では、class（クラス）というものを定義し、その中に特定の処理を記述します。つまり、

Java 言語プログラムはクラスの集まり

なのです。そして**クラス**は次の形で定義します。

```
class クラス名 {
  クラスの定義部分
}
```

上の例の場合は、

◆ クラス名が「ApplicationFirst」である。

◆ 「ApplicationFirst」クラスの定義部分には、「**main**」という名前の”**処理**”がある。

というようになっています。実は、この時点でオブジェクト指向という点について解説すると包括的な理解に到達できるのですが、それは別の機会に学習する事にして、今は全体の流れをできるだけ早くつかむことに重点を置き、先に進むことにします。

当面は以下の点を頭に入れておけば十分です。

◆ 上の”**処理**”を**メソッド**と呼ぶ。

◆ 一つのアプリケーション・プログラムは一般に複数のクラスからなる。

◆ その中のどれか一つのクラスには **main** メソッドがなければならない。

◆ main メソッドには、「public static void」という修飾子がつく（今の段階では、そういう約束だと思っておいてください）。

さて、最後に main メソッドの中身を見てみましょう。命令らしきものとして、

```
System.out.println("Hello Java");
```

がありますが、この命令には次の４つの文法事項が含まれています。

① `System.out`：標準出力（「Ⅲ. プログラムの実行」で用いるコマンドプロンプトの黒い画面のことです）。

② `println( )`：(　)内を表示するという命令。

③ " "で囲まれた部分は文字列となる。

④ 一つの文（命令）の終わりには区切り記号「；」をつける。

したがって、上の命令文は

標準出力に対して"　"内の文字列を出力（表示）せよ。

という指示を送った事になります。ですから、プログラムを実行すると画面に「Hello Java!」という文字が表示されるはずです。

Ⅱ. プログラムの保存

それでは、プログラムを保存しましょう。このプログラムは「ApplicationFirst.java」という名前で保存する必要があります。このように Java では、main メソッドを持つクラスの名前をアプリケーション・プログラムのファイル名とする事になっています。

保存場所は、Java のシステムがインストールされているフォルダ内に行うのが良いでしょう（そうすると、プログラムのファイルがどの場所にあるかを実行時に指定しなくても良いからです）。その場所は、通常の JBuilder のインストールを行った場合

```
C:¥JBuilder7¥jdk1.3.1¥bin
```

（C ドライブの［JBuilder7］→［jdk1.3.1］→［bin］とたどります）

となります。エディタのメニューから［ファイル］→［名前を付けて保存］を選び、上記

の場所に「ApplicationFirst.java」という名前で保存してください。このとき、ファイルの種類をテキストとしたままだと、「.txt」が自動的につけられることがあります。そこで、次のように、ファイルの種類を「すべてのファイル」としてからファイル名を記入するようにしましょう。

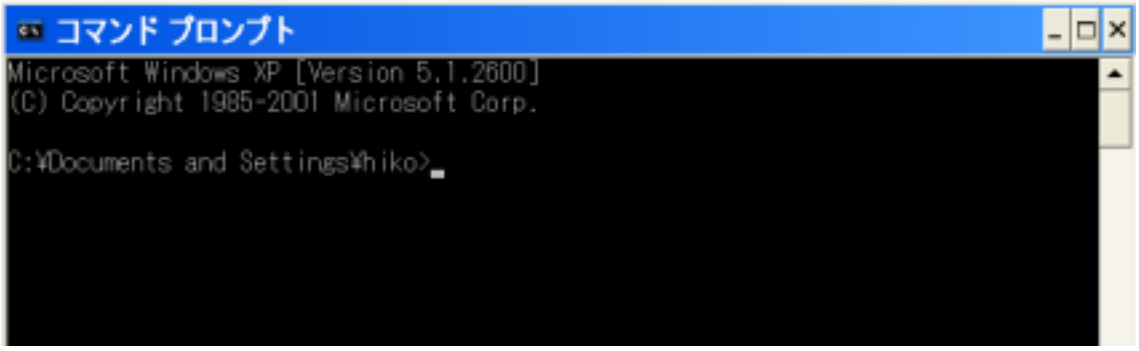
III. プログラムの実行

プログラムを実行しましょう。Java 言語プログラムの実行には

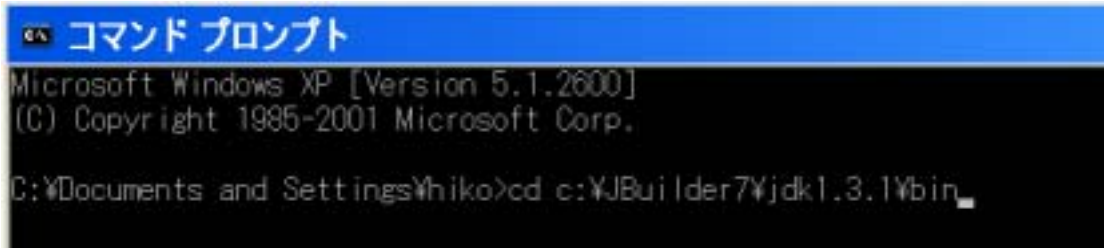
ソースコード（*.java）の翻訳（コンパイル）　→　翻訳済みファイルの生成

→　翻訳済みファイルの実行

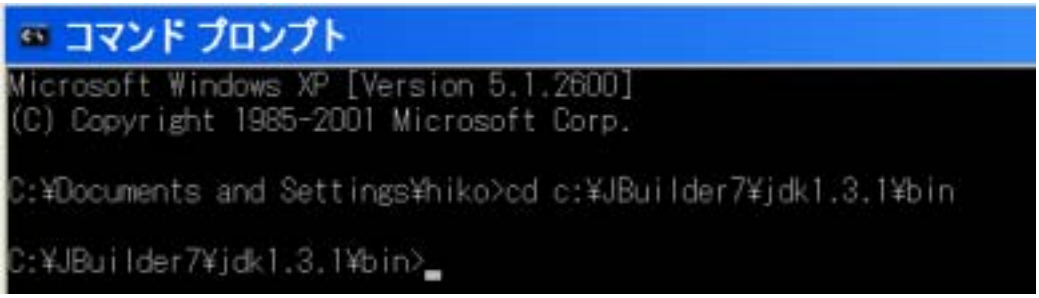
という手順を踏まねばなりません。そのためには、翻訳あるいは実行する命令をコマンド形式で入力しなければなりません。その命令を入力するためには、「コマンドプロンプト」画面を用います。［スタート］→［プログラム］→［アクセサリ］→[コマンドプロンプト］を選択してください。すると、次のような黒いウィンドウが現れるはずです。表示は使っている機種と OS によって若干異なります。



上の例で言うと「C:¥Document…>」の部分を**プロンプト**と言います。コマンド入力を行う際にはプロンプトの右側に入力します。さて、ここで、以下のコマンドを入力して［Enter］キーを押してください（下線部が入力)。コマンド入力の際は、コマンドを記述した後、［Enter］キーを押すことで初めてコマンドがシステム側に受け付けられます。



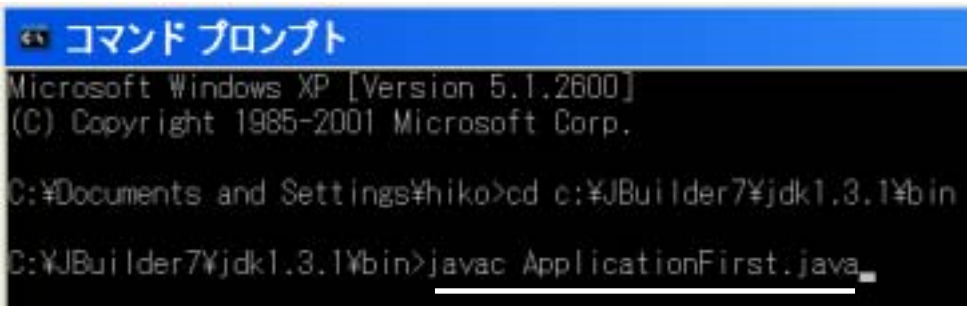
すると、プロンプトが次のように変わったはずです。



プロンプトは、現在のディレクトリ、つまり作業を行うフォルダを表しています。この場所に、Java プログラムに対するコマンドがインストールされています。

★　第一段階：ソースコードの翻訳（クラスファイルの生成）

続いて、次のようにコマンド（下線部）を入力してください（入力後、［Enter］キーを押すことを忘れないように）。



もし、正常に翻訳されれば、カーソルが次のプロンプトに移ります。何らかのエラーメッセージが表示された場合は、ソースコードのタイプミスが原因と思われます。エディタを用いて、もう一度タイプミスがないか確認し、修正後再び保存した上で、上の「javac」命令を実行して下さい。

実は、javac 命令により、「*.java」というソースコードから「*.class」というクラスファイルが生成されます。これは**バイトコード**とも呼ばれ、OS が Windows 以外の Linux や MacOS でも共通です。つまり、Java システムがインストールされてさえいれば、どの OS（若干の例外はありますが）でも処理することができます。

★　第二段階：クラスファイルの実行

さて、上の第一段階により、「ApplicationFirst.class」というクラスファイルが生成されました。これを実行するには、次のように「**java**」命令を用います。この場合、下のように「.class」をつける必要がないことに注意してください。コマンド入力後[Enter]キーを押すと、「Hello Java!」と表示されます。これが今のプログラムの実行結果です。

ここに、「java」命令はクラスファイルの内容を解読し、実行するという機能を持っています。先ほどふれたように、クラスファイルがどのような OS で作られたものでも「java」命令はそれを解読可能です。このような仕組みによって、Java 言語は、異種 OS 間でも互換性の良いプログラム開発環境を実現しているのです。

さて、以上が、通常の Java 言語アプリケーション開発の過程です。行程を振り返ってみると、

◆ エディタを用いてのソースコードの作成・保存

◆ コマンドプロンプトを用いてのソースコードの翻訳・実行（**２段階**）

という手順を踏まねばならないことが分かります。

一方 JBuilder では、アプリケーションブラウザ内でそれを全て作業できるようになっています。このような意味で、JBuilder のような開発ツールを**統合開発環境**と呼びます。さらに、JBuilder における「プロジェクト実行」命令は上の２段階の命令を一度に行ってくれていたことが分かるでしょう。さらにソースコードやクラスファイルなど、一つのアプリケーションに関するファイルを自動的にフォルダ内に納め、それをディスク上のどの場所に保存しても、「プロジェクトの実行」命令だけで簡単に翻訳・実行できるよう、プロジェクトファイルが管理してくれているのです。このように、作業を効率化あるいは高速化できるという意味で **RAD（Rapid Application Development）環境**とも呼ばれます。これは本節の冒頭で述べたことです。実は、JBuilder にはこの他に、

① 以下のような、定型部分（変更の必要のない、決まり切った部分）を自動生成してくれる。

② Windows アプリケーション作成の際に不可欠になる GUI のデザインを(アプリケーションブラウザの内容ペインにおける設計ビューにより)視覚的に行うことができる。

などの、作業の効率化に関する各種の機能が用意されています。それらの詳細については、次章以降の学習で実際に体験することにしましょう。

コマンドプロンプトを閉じてください。コマンドプロンプト・ウィンドウ右上隅の×をクリックすれば閉じます。