1996年4月1日(毎年1回1日いちいち発行)

提供・ボーランド オブジェクト指向開発環境
CBマガジン Borland

CB MAGAZINE

APR. 1997
April Fool

探究
C++Builderを選ぶ50の理由

●緊急報告 ActiveXだけじゃない! Delphi97の魅力 ●待望報告 Java開発が変わる!
JBuilderの真髄 ●詳細報告 イントラネット構築の救世主! IntraBuilderの実力
●毎度報告 復刊の挨拶 [CB MAGAZINE SPECIAL] ボーランドが送り出したC/C++の歴史

誌名変更・復刊の挨拶｜開発者の生産性を追及する CB MAGAZINE

2 度あることが 3 度あり、ふたたび誌名を変更し、復刊することになった。今回は扱う製品やページ数も増え、より多くの情報を活用していただけると思う。

私的な話で恐縮だが、昨年もっとも印象的だったのは MMX でも NT 4.0 でもなくメモリ価格の暴落である。Windows 95 がリリースされる頃は、32 ビット化でアプリケーションの消費メモリが増えメモリが高騰すると予想していたパソコン雑誌もあったが、現実には下落の一途をたどった。たとえば、16MB の SIMM は昨年の今ごろは 5〜6 万だったが、いまや 1 万を切るような値段である。私も、先日 32MB EDO RAM が¥17,000 だったのでひと組衝動買いしてしまった。

メモリ価格の下落を招いた理由には Windows95 の発売当初の不振が挙げられているが、皮肉なことにメモリが安くなり Windows95 を快適に使う環境も手に入れやすくなった。直接恩恵を受けたのは、もちろん消費者であるユーザーということになるが、それとてメーカーの宣伝に振り回されているだけだと、パソコンも「ただの箱」と化してしまうことになる。そうしないためには、目的を持った買い物と正しい情報収集が欠かせない。もっとも、実はそのことが大変な苦労なのだが。

メモリ価格の下落は、我々開発ツールベンダーも恩恵を受けている。もともと、開発ツールはツール自身と作成するアプリケーションのためにメモリが必要になる。

また、Delphi が使っている Object Pascal はすっきりしたものだが、C++コンパイラは C++自身が持つ機能が多いため（多重継承やオーバーロード、標準 C++ライブラリなど）、なかなかメモリ食いになってしまうのである。

C++Builder も必要メモリ 24MB、推奨 32MB となっているが、もっとメモリがあればさらによい。とはいえ、1 年前だったら 32MB のメモリを揃えるだけでけっこう高額になり、ひんしゅくものだっただろう（もっとメモリを消費する処理系はあるが）。しかし、今ならたいていのパソコンが 32MB のメモリを標準搭載しているし、そもそもそのくらいなければ Windows95 が快適ではない。それこそ、CPU を速いものに替えるよりメモリを増設するほうが、アプリケーションの快適な利用に影響することもある。Windows95 自身にはまだ 16 ビットコードが大量に残っていて、そんなにパフォーマンスを高められないという話もあるくらいだ。

さて、今年のボーランドは新製品が目白押しである。昨年末に発売された新製品 IntraBuilder や、発売されたばかりの C++Builder をはじめ、今後も Delphi の新バージョンや Java 開発ツール JBuilder が控えている。いずれも、本誌が創刊以来貫いてきた「開発者の生産性を追及する」という主張にかなう製品である。ここに掲載した情報が、読者の生産性を追及するための一助になれば幸いである。

探究

Borland C++Builder

C++Builder を選ぶ 50 の理由

C++普及委員会

昨年の COMDEX/Fall'96 で、突如その姿を見せた Borland C++Builder は、従来の C++プログラミングの認識を大きく改めるものとなりました。最新の ANSI C++に対応した機能をそのままに、ソースコードに依存していた C++開発を、生産性の高いビジュアル開発に移行させたのです。
ここに挙げる 50 の理由は、いずれも迅速なアプリケーション開発を求めるプログラマーが望むものばかりでしょう。是非、この素晴らしい環境を活用してください。

ビジュアル開発環境

1. C++言語とビジュアル開発の統合
   あらためて述べるまでもないでしょう。C++Builder の最大の特長は、C++のパワーとビジュアル開発の生産性を統合したことです。市販のアプリケーションや Windows 自身を含め、多くのアプリケーションが C/C++を使って開発されています。コンポーネントを使った迅速な開発のために、他の言語を覚える必要はありません。
2. ビジュアルとソースコードを連携させる 2Way-Tool
   2Way-Tool とは、ビジュアル開発で設計した内容がソースコードに反映されたり、ソースコードでの編集結果がビジュアル開発に反映されるというものです。ビジュアル開発の機能がいくらすぐれていると言っても、テキストベースの開発がおそろかになっていては、プロフェッショナルの利用に耐えるとは言えません。C++Builder の 2Way-Tool は、あらゆる側面で、ビジュアル開発とソースコード開発の両面を支援します。
3. 機能別に分類されたコンポーネントパレット
   C++Builder には 90 種類以上のコンポーネント（部品）が登録されていますが、これらはコンポーネントパレットに機能別に分類されています。このため、画面の貴重な領域をあまり占有することなく、アプリケーションを設計できます。
4. ビジュアルなフォームの継承
   C++のオブジェクト指向プログラミングを活かし、フォームやデータモジュールを継承して利用できます。単純にコピーして使う場合と違い、継承元の変更は、そのまま継承先にも反映され、仕様統一や既存フォームの再利用にも役立ちます。
5. フォームリンク
   別のフォームやデータモジュール、あるいはそこに配置されているコンポーネントを他のフォームなどから利用できます。設計時に、フォーム間でコンポーネントの相互利用できるため、プログラミングの手間が軽減されます。
6. データモジュール
   データベーステーブルや問い合わせ、コモンダイアログなど実行時に見えないコンポーネントを配置したり、データ構造などを設計するためのものです。ユーザーインターフェースとプログラムロジックを独立させ、モジュール性を高めます。
7. 再利用性を強化するオブジェクトリポジトリ
   オブジェクトリポジトリ（オブジェクトの倉庫）は、フォームやデータモジュールなど、文字通りあらゆるオブジェクトを管理しておける場所です。登録済みのフォームを使ったり、開発中のプロジェクトを登録したりできるので、アプリケーションの保守や拡張などにも役立ちます。

8. **イベントにも対応したオブジェクトインスペクタ**
   オブジェクトインスペクタは、コンポーネントのプロパティだけでなくイベントを設定するためにも利用します。複数のコンポーネントで同一のイベントハンドラを共用することもできます。
9. **好みに応じて操作性を変更できるカスタマイズ性**
   お仕着せの操作性をユーザーに強要するのではなく、ユーザーの使い方に合わせてカスタマイズできるよう、さまざまな機能をカスタマイズできます。C++Builder ではスピードバーに配置するボタンやコンポーネントパレットの表示、エディタの配色やキー操作など、多くの設定を変更できるようになっています。
10. **スピードバーやスピードメニュー**
    メニューバーの左端にあるスピードバーやマウスの右ボタンをクリックして表示されるスピードメニューは、必要なコマンドをすばやく呼び出すために便利なものです。位置合わせやサイズ調整なども簡単に処理できます。
11. **コードエディタ**
    キーボードマクロやインクリメンタルサーチといった実用的な機能を持つエディタが組み込まれています。
12. **プロジェクトやフォームを容易に作成できるウィザード**
    MDI/SDI 対応のアプリケーションプロジェクトやデータベースフォームウィザードなど、アプリケーションの要素を簡単に作成する機能が充実しています。

コンポーネント

13. **90 種類以上の再利用可能なコンポーネント**
    C++Builder には、Standard 版で 90 種類以上、Professional 版や Client/Server Suite 版では 100 種類以上のコンポーネントが登録されています。これらのコンポーネントを使って、広範なアプリケーションをビジュアルに開発できます。
14. **プログラミング作業を軽減する充実したプロパティ**
    スクロールバーの位置や文字列グリッドのセルの高さや幅、リストボックスやツリービューの項目など、さまざまな情報をオブジェクトインスペクタを使って設計時に指定できます。充実したプロパティがビジュアル性を高め、プログラミングの必要性を減らします。
15. **ユニークな拡張コンポーネント**
    イメージ付きボタンやセル内での編集もできる文字列グリッド、マルチメディアプレーヤー、ビジュアルに設計できるポップアップメニューなど便利なコンポーネントを数多く揃えており、オプションの製品を購入しなくても、最新の機能を持つアプリケーションを開発できます。
16. **Windows95 コモンコントロール**
    Standard 版、Professional 版、Client/Server Suite 版のすべてで、書式付きテキスト編集やページコントロールといった Windows95 のコモンコントロールが利用できます。
17. **レポート出力をアプリケーションに組み込む QuickReports**
    完全にアプリケーションに埋め込むことができ、プログラムでレポート出力を制御できるコンポーネントです。
18. **ActiveX コントロールの組み込み**
    サードパーティ製の ActiveX コントロール(OCX)を組み込んで利用することもできます。
19. **Internet Solutions Pack**
    Professional 版以上で提供されるインターネット・イントラネットに対応するための ActiveX コントロールです。
20. **独自コンポーネントの開発**
    C++Builder で利用するコンポーネントを C++Builder 自身で開発できます。既存のコンポーネントの機能を拡張したり、すべての機能を開発者が設計することができます。

コンパイラ

21. **32 ビット最適化コンパイラ**
    Pentium スケジューリングをはじめ、強力な最適化機能を持つ 32 ビットコンパイラを搭載しています。
22. **高速コンパイラとインクリメンタルリンカ**
    高速なコンパイラ、プリコンパイルヘッダ、そして新しいインクリメンタルリンカによって、短時間でプロジェクトを構築でき、ビジュアル開発の生産性を妨げることはありません。
23. **ANSI C++のサポートと拡張構文**
    名前空間、テンプレート、演算子オーバーロード、標準 C++ ライブラリなど、すべての ANSI C++の機能を利用できます。さらに、プロパティを定義するための__property やオートメーションに対応した__automated など、ビジュアル開発に役立つ拡張構文をサポートしています。
24. **プログラミングを効率化するクラスや関数**
    異なる型を保持できオートメーションオブジェクトも制御できるバリアントクラス、文字列操作を容易にする文字列クラス、日付時間クラス、通貨型クラス、集合クラスなどが定義されています。C++の特長を活かし、多くの演算子オーバーロードが定義されています
25. **すべての Windows システムの利用**
    他の C/C++処理系と同様、マルチスレッドなど Windows 95/NT が持つすべての機能を利用できます。コールバック関数やウィンドウハンドルなど、Windows API の利用にも制約はありません。また、エクスポート関数を持つ DLL を作成したり、利用することもできます。また、C++を使っているため新しい開発キットへの対応も容易です。

デバッガ

26. **ソースレベル統合デバッガ**
    トレース実行やステップ実行をサポートした、ソースレベルの統合デバッガが組み込まれており、アプリケーションの問題を検出するために役立ちます。マルチスレッドにも対応しています。
27. **式の指定や表示形式を指定できる監視式**
    プログラムの停止中に値を監視したり、即座に式を評価するために、任意の C++式を指定できます。表示形式を指定したり、デバッグ情報の付いた関数を呼び出すこともできます。
28. **条件式やパスカウントを指定できるブレークポイント**
    ブレークポイントでは、条件を満たしたり指定した回数通過した場合だけ停止させるということを設定できます。
29. **CPU ビュー**
    逆アセンブルリストやレジスタを表示できる CPU ビューによって、機械語レベルでのデバッグもサポートします。
30. **例外の発生によるデバッガの呼び出し**
    ビジュアル開発環境でプログラムを発生中に例外が発生すると、自動的に開発環境に戻って停止させられます。

データベース処理

31. コンポーネントベースのデータベース開発
C++Builder に組み込まれている 20 種類以上のコンポーネントを使って、データベース対応のアプリケーションをビジュアルに開発できます。テーブルや問い合わせ、ストアドプロシージャのそれぞれについて専用のコンポーネントがあり、項目エディタを使えばドラッグ&ドロップを使ってフォームを設計することもできます。また、レコード操作やイメージ表示など多彩な表示用コンポーネントが提供されます。

32. 選択リストを指定できるデータベースグリッド
データソースの内容をグリッド形式で表示するデータベースグリッドでは、項目ごとに選択リストを指定しておき、ユーザーがコンボボックスを使って項目を選べます。

33. マルチオブジェクトグリッド
Professional 版以上では、複数のレコードを表示するために、他のデータベース対応コントロールを配置して、複数のレコードを表示させるマルチオブジェクトグリッドが提供されています。

34. テーブルの結合や複製を行なうバッチムーブ
テーブルを結合したり複製するためにバッチムーブコンポーネントが使えます。オブジェクトインスペクタやスピードメニューを使えば、設計時にも結合や複製を実行できます。

35. 設計画面でのデータ表示
C++Builder では、コンポーネントを使って作成したデータベース対応のフォームに、実際のテーブルや問い合わせの内容を表示できます。このため、わざわざプログラムを実行させることなく、設計時にコンポーネントや個々の項目の大きさを細かく調整し、見やすく使いやすいアプリケーションを作成できます。

36. Borland Database Engine
C++Builder に組み込まれている Borland Database Engine (BDE)によって、dBASE/Paradox/テキスト形式のデータを処理できます。また、Professional 版では ODBC 接続に対応し、さらに Client/Server Suite 版では SQL Link を使って直接 SQL データベースに接続できます。これらのデータ処理は、BDE によって共通化されています。

37. キャッシュアップデート
データベースに対する編集をデータベースエンジンでキャッシングすることで、まとめて適用したり、編集を取り消すことができます。とくに、クライアント/サーバーアプリケーションの開発においては、問い合わせの処理が局所化されパフォーマンスが向上します。

38. データベースセッション
マルチスレッドを使ったバックグラウンドでの問い合わせなど、独立したデータベースセッションが必要な場合も、コンポーネントを使ってビジュアルに開発できます。

39. データベースエクスプローラ
テーブルやエリアスを参照・編集できるユーティリティです。テーブルや項目をフォーム上にドラッグ&ドロップして、データベース対応のフォームを設計することもできます。Client/Server Suite 版に付属している SQL エクスプローラでは、ストアドプロシージャやトリガーのようなメタデータを参照することもできます。

40. データディクショナリ
Professional 版以上でサポートされているデータディクショナリを使うと、複数のアプリケーションで共通に使いたいデータベース固有の属性を設定しておけます。

41. データベースデスクトップ
新しいテーブルや QBE による問い合わせを作成したり、テーブルの編集やコピーをサポートしたユーティリティです。

42. Local InterBase Server&InterBase Server
Professional 版以上で提供される Local InterBase Server を使うと、スタンドアロン環境でクライアント/サーバースタイルのアプリケーションを開発できます。また、Client /Server Suite 版では、開発とテストの目的で 4 ユーザー版の InterBase Server が提供されます。

43. SQL Link および再配布権
Client/Server Suite 版で提供される InterBase、Oracle、MS-SQL Server、Sybase、Informix、DB2 に直接接続し、高速にデータを処理するためのミドルウェアです。開発したアプリケーションとともに再配布でき、追加のロイヤリティは不要です。

44. SQL モニタ
Client/Server Suite 版にバンドルされている、SQL のパフォーマンスを調査するためのツールです。

45. Visual Query Builder
Client/Server Suite 版にバンドルされている、問い合わせコンポーネントから呼び出し、ビジュアルに問い合わせを設計するためのツールです。

46. Data Migration Wizard
Client/Server Suite 版にバンドルされている、ローカルデータから SQL データベースへのテーブルをアップサイズするためのツールです。

その他

47. インストーラ作成ツール InstallShield Express
Professional 版以上に添付される InstallShield Express は、アンインストール対応のインストーラをダイアログボックスを使った簡単な設定で作成できます。

48. Windows メッセージを追跡する WinSight32
Professional 版以上に添付される WinSight32 を使えば、システム上のすべてのウィンドウに関する情報や Windows メッセージを追跡でき、アプリケーションの問題をシステムレベルで調査できます。

49. VCL ソースコード
Professional 版以上では、コンポーネントライブラリのソースコードが提供されます。コンポーネントの動作をリバースエンジニアリングをせずに、ソースコードで確認できます。VCL は、Object Pascal で書かれています。

50. PVCS バージョニングエンジン
Client/Server Suite 版では、チーム開発とバージョン管理を支援する PVCS バージョニングエンジンが組み込まれています。

ここに挙げた以外にも、ビットマップやアイコンを編集するためのイメージエディタや開発環境に独自の機能を組み込む Open Tools API などユニークな特長は数多くあります。もちろん、特長の数が多ければよいというものではないでしょう。C++Builder の真価は、これらがすぐれたアプリケーション開発のために使いやすく統合されているところにあります。

詳細報告

イントラネットの救世主！ IntraBuilder の実力

イントラネット普及委員会

IntraBuilder

イントラネットといえば、ひところは企業内システムのあり方が変わると騒がれたものだが、実用化は決して容易なものではなかった。そんな中、昨年末に発売され、イントラネット構築のためのユニークなツールとして注目を集めている IntraBuilder について、実力を探ってみることにする。

クライアントサーバー vs. イントラネット

企業内でデータを共有する手段として、TCP/IP ベースの WWW 技術を応用したイントラネットがクローズアップされて久しい。そして、あらゆる開発ツールが HTML 対応やイントラネット対応を打ち出し、まるでクライアント/サーバーシステムの時代が終焉を向かえ、今後はすべての業務システムがイントラネット上に構築されるような騒ぎである。

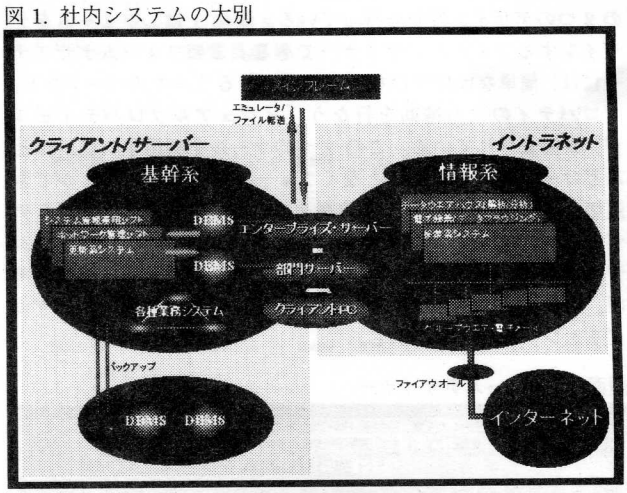
しかし、日常の業務の多くはクライアント/サーバーシステムで稼動しており、また、今後も多くのクライアント/サーバーシステムが構築されるはずである。

では、Web ブラウザーをクライアントプラットホームとしたイントラネットは、どのように利用されているのか。

答えは情報系システムである。

図 1 は、企業内で利用される業務システムを基幹系と情報系に大別したものである。基幹系ではメインフレームや DBMS との結び付きが強く、情報系では基幹系のデータを参照するシステムが多い。また、現在では基幹系も情報系もクライアント/サーバーシステムが中心である。ボーランドでは、DBMS との結び付きの強い基幹系はクライアント/サーバーシステム、データ参照の多い情報系をイントラネットシステムで構築するソリューションを提案している。また、クライアント/サーバー開発ツールとしては、Delphi、C++Builder、そしてイントラルット開発ツールとしては IntraBuilder を提供している。

図 1. 社内システムの大別



情報系システムとは

情報系システムの代表格は、部署間でのデータ共用である。たとえば、部門単位で発信する情報を社内に公開しすることである。部門サーバーでは、次のようなデータベースを構築していることが多いのではないだろうか。

営業部門	売上予測金額と売上実績金額の報告
技術部門	製品開発スケジュール
製造部門	製品在庫数の報告

そして、これらの情報に関する問い合わせは、予想以上に多いものである。部門サーバー上に小規模 Web を構築し、部門単位でイントラネットサーバーを用意すれば、部署間でのデータ共用は著しく向上するはずである。

なお、部門サーバーで運用する小規模 Web であれば、Windows NT Server 4.0+Internet Information Server と、IntraBuilder Professional の組み合わせで構築できる。

IntraBuilder の仕組み

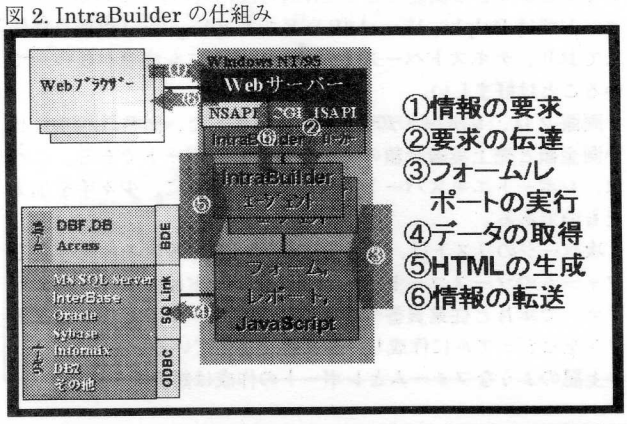
情報系イントラネットを構築する IntraBuilder は、作り置きの HTML を生成する HTML エディターではない。サーバーPC 内で Web サーバーと共に動作する IntraBuilder サーバーは、DBMS と連携して最新データを反映した動的な HTML を随時生成する Web データベース開発システムである。常に、最新データをクライアントプラットホームの Web ブラウザーから閲覧でき、HTML を更新(メンテナンス)する手間は不要である。

IntraBuilder の仕組みは、図 2 のようになる。

1. Web ブラウザーが Web(HTTP)サーバーに情報を要求する。
2. Web ブラウザーが受け取った要求を、IntraBuilder ブローカーが IntraBuilder エージェントに伝達する。
3. JavaScript->HTML 変換エンジンである IntraBuilder エージェントで、IntraBuilder デザイナーで作成したフォームかレポートを実行する。
4. フォームとレポートは、データベースとリンクしているので、Borland Database Engine を通じて最新データを取得する。
5. IntraBuilder エージェントが、最新データを含んだフォームスタイルかレポートスタイルの HTML を生成する
6. 生成された HTML は、IntraBuilder ブローカーと Web サーバーを経由して、Web ブラウザーに返される。

この仕組みによって、最新データを反映した動的な Web パブリッシングが実現できるのである。

図 2. IntraBuilder の仕組み



拡張 JavaScript を採用

IntraBuilder の最大の特徴はカスタマイズ言語に JavaScript を利用することだろう。JavaScript は、動的なホームページを作成するためにネットスケープが開発したスクリプト言語であり、Web ブラウザー上でフォーム内のデータ処理やページウィンドウの表示などを制御でき、Netscape Navigator 2.0 以降と、Microsoft Internet Exproler 3.0 以降で利用することができる。

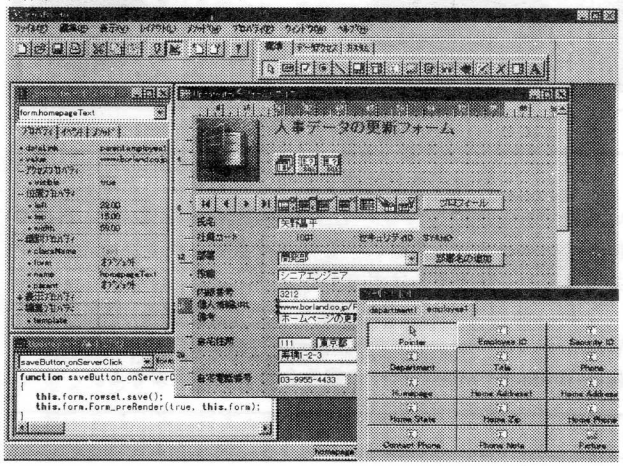
IntraBuilder では、IntraBuilder 用に拡張した JavaScript を搭載し、カスタマイズ用のスクリプト言語として利用できるのである。拡張 JavaScript は、データ処理やプログラミングの効率を向上させるため、データベースを管理するクラスとオブジェクトの継承やサブクラス化の言語要素が追加されている。

また、テーブルのデータを Web ブラウザーに表示する画面、データを更新する画面、クエリー(検索)を行う画面を作成する場合に利用するビジュアル開発ツールは、JavaScript コードを自動生成する。これらのカスタマイズも同一の拡張 JavaScript が利用することができる。

ビジュアル開発ツール

先に紹介したビジュアル開発ツールは、フォームとレポートの 2 つのデザイン機能を持っている。画面 1 は、フォームをデザインするフォームデザイナーである。このフォームデザイナーには、簡単な操作でひな型を作成できる「エキスパート」と、プロパティの入力補助を行なう「ビジュアルプロパティビルダ」が用意されている。この 2 つのツールを利用すれば、ほとんどコードを記述することなくフォームを完成することができ、適切な JavaScript コードが自動生成される。また、フォームデザイナーには 2Way-Tool 機能が搭載されているので、フォームデザイナーが生成したコードにエディタで変更を加えた後、再び、フォームデザイナーに読み込みなおしてビジュアルにプログラミングできるのである。

画面 1. フォームデザイナー



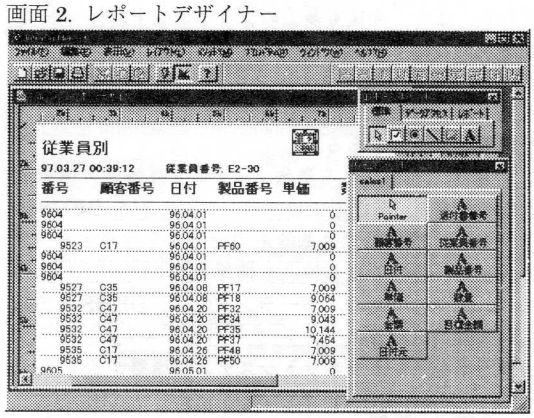
2Way-Tool 機能は、ビジュアルツールによる易しい操作性を実現しつつ、エディタによるプログラミングの柔軟性を損なわないものである。幅広い開発者が快適に利用できるインタフェースであり、筆者は、プログラミングの生産性を最大限に発揮することができる機能として 2Way-Tool に注目している。ボーランドでは Delphi、Visual dBASE でも 2Way-Tool 機能を提供しており、テキストベースでの使いやすさにも注意が払われていることは好ましい。

画面 2 は、レポートデザイナーで作成した、従業員別の売上予測金額と売上実績金額の差を把握するレポートである。これは、レポートエキスパートで作成したひな型に、少々手を加えたものである。

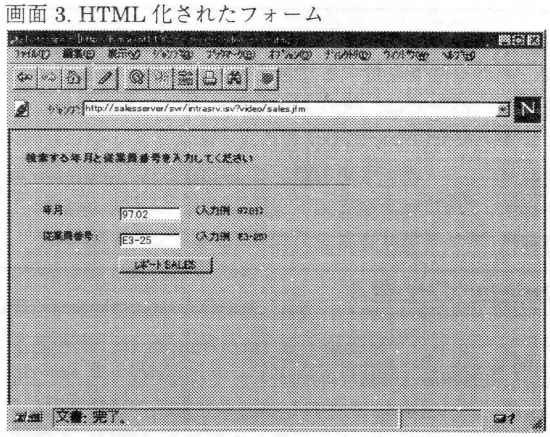
次ページのリストは、画面 2 のレポートに検索条件を与えるフォームのソースコードである。このコードは、フォームデザイナーで年月と従業員番号を入力するフィールドとプッシュボタンをビジュアルに作成し、少々手を加えている。

上記のようなフォームとレポートの作成は短時間で終了する。フォームとレポートを部門サーバーの Web サーバーPC にコピーし、Web サーバーソフトと IntraBuilder エージェントを起動すると、画面 3、画面 4 のように Web ブラウザーから HTML 化されたフォームやレポートを利用できるようになる。

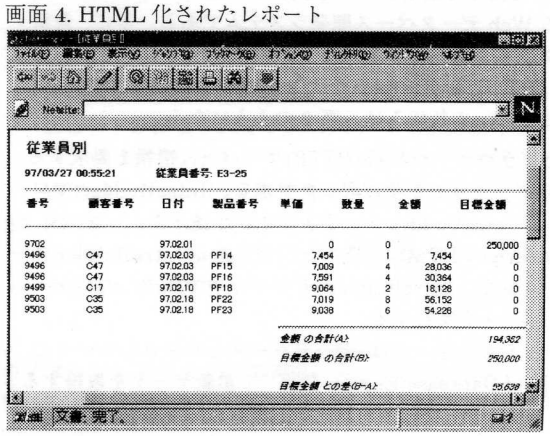
画面 2. レポートデザイナー



画面 3. HTML 化されたフォーム



画面 4. HTML 化されたレポート



構築できるWebデータベースの規模

IntraBuilder には、IntraBuilder(標準パッケージ)、IntraBuilder Professional、IntraBuilder Client/Server の3種類の形態があり、構築するWebデータベースの規模により、利用するパッケージが異なる。パッケージ毎の想定規模は、表1の通りである。

表1 パッケージ毎の想定規模

IntraBuilder	想定サーバ機種と台数	想定同時接続数
(標準パッケージ)	Win95/NT× 1台	～ 5ユーザー程度
Professional	WinNT × 1台	～20ユーザー程度
Client/Server	WinNT × 2台	～50ユーザー程度

IntraBuilder(標準パッケージ)は、「IntraBuilder の仕組み」で触れたIntraBuilderエージェントがシングルインスタンスのため、大量のデータ処理には不向きである。しかし、IntraBuilder Professional と IntraBuilder Client/Server の IntraBuilder エージェントはマルチインスタンスで、WindowsNT 上に複数起動できるので、スループットを維持したまま大量のデータ処理を行うことができる。また、IntraBuilder Client/Server の IntraBuilder ブローカーは、複数台の Windows NT マシンに対するディスパッチ機能を持ち、サーバーPC 1台あたりの負荷を分散するとともに、同時接続ユーザー数を拡大することができるのである。

リスト1 検索フォームのソースコード

```
// {End Header} ｺﾒﾝﾄは削除できません//
// 生成 97.03.26
//
var f = new salesForm();
f.open();
class salesForm extends Form {
   with (this) {
      height = 10;
      left = 0;
      top = 0;
      width = 59.4286;
      title = "";
   }


   with (this.rule1 = new Rule(this)){
      left = 1;
      top = 2;
      size = 2;
      right = 56;
   }


   with (this.html1 = new HTML(this)){
      height = 1.2;
      left = 1;
      top = 0.5;
      width = 55;
      color = "blue";
      fontName = "ＭＳ Ｐゴシック";
      text = "<H3>検索する年月と従業員番号を入力してください</H3>";
   }


   with (this.html2 = new HTML(this)){
      height = 0.95;
      left = 4;
      top = 3;
      width = 12;
      color = "blue";
      fontName = "ＭＳ Ｐゴシック";
      text = "年月 :";
   }


   with (this.html3 = new HTML(this)){
      height = 0.95;
      left = 4;
      top = 4.5;
      width = 12;
      color = "blue";
      fontName = "ＭＳ Ｐゴシック";
      text = "従業員番号 :";
   }


   with (this.text1 = new Text(this)){
      left = 17;
      top = 3;
      width = 12;
   }


   with (this.text2 = new Text(this)){
      left = 17;
      top = 4.5;
      width = 12;
   }


   with (this.button1 = new Button(this)){
      onServerClick = {;var
searchMonth=form.text1.value;var
searchNo=form.text2.value;_sys.reports.run("SALE
S.JRP",1,1,searchMonth,searchNo)};
      left = 17;
      top = 6;
      width = 17;
      text = "ﾚﾎﾟｰﾄ SALES";
   }


   with (this.html4 = new HTML(this)){
      height = 0.95;
      left = 30;
      top = 3;
      width = 20;
      color = "black";
      fontName = "ＭＳ Ｐゴシック";
      text = "(入力例 : 97.01)";
   }


   with (this.html5 = new HTML(this)){
      height = 0.95;
      left = 30;
      top = 4.5;
      width = 20;
      color = "black";
      fontName = "ＭＳ Ｐゴシック";
      text = "(入力例 : E3-25)";
   }
}
```

緊急報告

Delphi 3

ActiveX だけじゃない！ Delphi 3 の魅力

Delphi 普及委員会

Delphi が「次世代のビジュアル開発ツール」というふれこみで輝かしい登場をとげてから、早くも2年が経つ。今では、主要な開発ツールとしての地位を築き、ネイティブコードとビジュアル開発の統合を「新しいスタイル」と呼ぶことさえ、古めかしい気がするほどである。しかし、Delphi の進化が止まることはない。プログラマー垂涎の新機能について紹介しよう。

はじめに

まずお断りしておかなければならないのは、Delphi の次バージョンは表紙に書かれている「Delphi 97」ではなく「Delphi 3」になりそうだということである。もちろん、正式なアナウンスは何もないが、現在のベータ版の表記からは、そのような製品名に決まる可能性が高い。そして、Delphi 1.0、Delphi 2.0 が「ホップ」、「ステップ」ならば、Delphi 3.0 はまさに「ジャンプ」に相当するバージョンだと言えよう。

正直なところ、Delphi 1.0 でユーザーから寄せられた要望のほとんどは Delphi 2.0 で実現されていたと思う。特に、（性能はともかく）他のツールで実現されていたフォームの継承やドラッグ&ドロップによるデータベースフォームの設計などが、実用的な形で実装された点は大きいだろう。

また、Delphi 1.0 で他のフォームを利用するためにプログラムコードが必要だった点も、フォームどうしを関連づけるフォームリンクによって、よりスマートに解決された。今では、ただ別のフォームを利用しやすくなっただけでなく、そのフォームにあるコンポーネントまで参照したり、データモジュールによって非ビジュアルコンポーネントなどを組み合わせたアプリケーションロジックを抽象化するための仕組みさえある。

逆に、いまなお Delphi の機能に追いついていないビジュアル開発ツールは多い。たとえば、設計画面で実際のデータを確認するというごく基本的なことでさえ、できないツールがほとんどである。たしかに、ちょっと設計して、ちょっと実行してみるというのはビジュアル開発ツールの魅力である。だから実行してみればよいわけだが、アプリケーションが大規模化する状況において、大きなプロジェクトになってから同じことをするのは、なかなか骨の折れる仕事である。Delphi なら設計フォームで確認できるので、そんな手間は不要である。このことひとつだけでも、Delphi がどれだけアプリケーション開発の設計を簡単にしているかを想像していただけるだろう。

そして、Delphi 3 である。

Delphi 3 でも、もちろん「こんな機能が欲しかった」という新機能はあるだろう。しかし、予想さえしなかったもので「なるほど、これは便利だ」と思わせる機能も、数多く見つけられるとはずである。いずれも、ユーザーニーズに応えるだけでなく、新しいソリューションを提案するボーランドならではの機能と言ってもよい。Delphi 3 を手にしたとき、もはや手放せないツールになること請け合いである。

とりあえず ActiveX

「ActiveX だけじゃない！」とはいえ、やはり ActiveX を紹介しないわけにはいかないだろう。

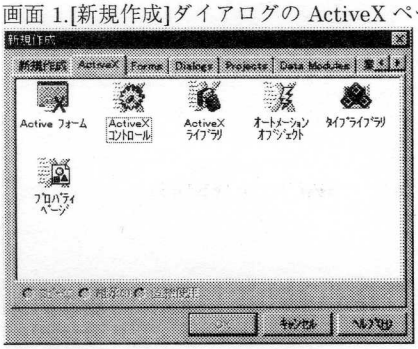
Delphi 3 は ActiveX コントロールの作成、Active フォームの作成、Active ドキュメントをサポートする。この中で Active ドキュメントは、従来の OLEContainer にインターネットを経由したオブジェクトを扱える機能を追加したものである。

ActiveX コントロールは、かつて OCX と呼ばれていた汎用的な部品の仕様である。Delphi は、通常 Delphi 専用の部品（コンポーネント）を使う。これは、Delphi 自身で作成されており、実行形式ファイルに直接組み込まれるので処理速度も速い。しかし、Visual Basic をはじめとする他のツールでは利用できなかった。せっかく作ったのだから他でも使えるようにしたいと思っても、ActiveX コントロールを作るのは、かなり高度な知識が必要であり、開発者も限られていた。

それこそ、市販されるものでなければ、Delphi のコンポーネントの方が作りやすいので、ActiveX コントロール(OCX)よりも多く出回っていると言われることもあるくらいである（インターネットを利用できる人は、http://www.delphi32.com/を参照してみてほしい。英語版ではあるが、山のようなコンポーネントに出会えることだろう）。

Delphi 3 の ActiveX コントロールの作成機能は、Delphi 用のコンポーネントを ActiveX コントロールに変換してしまうものである。ウィンドウハンドルを持っていなければならないなど、コンポーネントを作成するときの上位クラスにいくらか制約はあるものの、とにかく作ったコンポーネントをマウス操作だけで、ActiveX コントロールに変換できてしまうのである。

画面 1.[新規作成]ダイアログの ActiveX ページ



Active フォームは、インターネットエクスプローラ 3.0 のようなブラウザで表示できる ActiveX コントロールと考えるとわかりやすいだろう。だが、その名の通りコントロールではなくフォームである。つまり、Delphi のコンポーネントではなく、Delphi のフォームデザイナを使って設計したものが、IE 3.0 で表示できるわけである。もちろん、WINTEL(Windows +Intel)に依存することになるが、その自由度ははかりしれない。

Delphi 3 での ActiveX の対応は、かなりハイレベルなものである。最先端の機能を提供するだけでなく、使いやすい形で実現するというボーランドの方針に偽りはない。

ただ、一般論として ActiveX に依存し過ぎるのは禁物である。特に最近話題になっているが、インターネットでのセキュリティは完璧ではないからである。

なお、ActiveX コントロール/フォームの作成は、Professional 版以上でサポートされる。

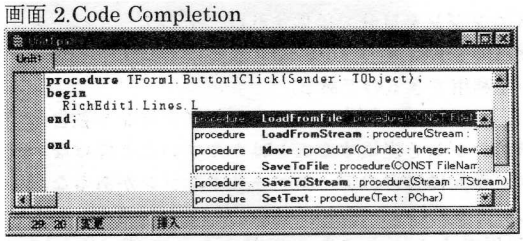
コードエディタの入力支援機能

Delphi のエディタは、他のツールのエディタに比べてはるかに強力な機能を持っている。キーボードマクロ、矩形ブロック、構文強調表示、大規模なテキストの編集、複数ウィンドウでの表示など、プロフェッショナルの要求に耐えうるものだと言える。これは、ボーランドが 10 年以上にわたる統合開発環境の歴史において、ソースコード開発の効率に大きく影響するエディタの操作性を重視してきたためでもある。

Delphi 3 は、さらに翔び抜けた機能を提供している。

画面 2 に示しているのは、Code Completion という機能である。これは、コンポーネント名を入力してピリオド (.) を入力したときなど、オブジェクト名の後ろにピリオドを入力すると、自動的にそこで使えるプロパティやメソッドの一覧を表示してくれるというものである。

画面 2.Code Completion



あまり知られていないように思うが、Delphi のエディタはコンポーネントの名前からコンポーネントの型を判別する機能がある。たとえば、Edit を配置していたら、それが Hello という名前であろうと Goodbye という名前であろうと、プログラム上の Hello や Goodbye のところで[F1]を押すと正しく Edit のヘルプが表示される。

Code Completion は、プログラム中に入力されたオブジェクトの名前から、どのオブジェクト型が対象であるかを判別し、自動的に名前をリストアップしてくれるのである。メソッドの名前は分かっていても、引数の順序がわからないとか、プロパティの正確な綴りが思い出せないといったときにも、いちいちオンラインヘルプを呼び出す必要がなくなる。

Code Completion は、VCL に組み込まれているコンポーネントやライブラリだけでなく、今プログラム中に入力しているフォームやクラス、レコードにまで対応している。わざわざ、カスタマイズの手間を書けなくても、自動的にすべての名前をリストアップしてくれるのである。

また、Code Parameters という機能は、関数名やメソッド名の後ろに左カッコ (() を入力すると、自動的に引数の型を表示してくれるというものである。

Code Templates は、あらかじめ定義されている短縮入力を呼び出す機能である。たとえば、forb と入力して[CTRL]+[J]を押すと、次のプログラムがエディタに埋め込まれ、for の直後にカーソルが移動する。

```
for  := to do
begin


end;
```

for だけを入力すれば、該当するものが複数あるので、リストが表示され、そこから選べるようになっている。また、自分で短縮入力したい文字列を登録することもできる。

これらの機能は、プログラム入力時のスペルミスを減らし、信頼性の高い開発に役立つだろう。

新しいコンポーネント

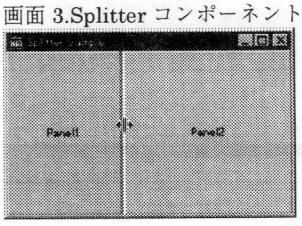
Delphi 3 には、130 個以上のコンポーネントが提供されている。だが、問題は数ではない。数を増やすだけなら、つまらないものをいくつか用意しておけば済むからだ。Delphi の凄いところは、これらのコンポーネントが、どれもビジュアル性に富み、ユニークな機能を提供していることにある。

別のセクションで詳しく紹介するものもあるが、新しいコンポーネントをいくつか取り上げてみよう。

- Splitter（[Additional]ページ）

フォームをパネルで分割したいが、「分割した領域の大きさを実行時に変更したい」という要望はこれまでにもあった。これまでは、マウスのイベントを処理したり、分割する領域のためにパネルを使うといった工夫が必要であった。Splitter は、まさにこの分割線の機能をサポートするものである。

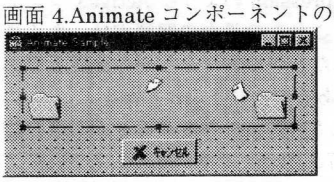
画面 3.Splitter コンポーネントの例



- Animate（[Win32]ページ）

エクスプローラでファイルを削除するときなど、ファイルをごみ箱に放り込むアニメーションが表示される。これをアプリケーションで使うためのコンポーネントである。また、これも Delphi の特長的な機能であるが、ちゃんと設計時にどんなアニメーションが表示されるかを確認できる。

画面 4.Animate コンポーネントの例



- ToolBar/CoolBar（[Win32]ページ）

インターネットエクスプローラ 3.0 を使われたことがある方なら、そこで使われているツールバーが簡単に設計できると考えてもらうとよい。ToolBar は、配置されているボタンなどのコンポーネントの位置や大きさを自動的に調整し、マウスカーソルが移動するとボタンを浮き出させるということもできる。

Coolbar では、ツールバーのような他のコントロールを配置しておき、左隅のグリップを使って全体を移動できる。

画面 5.ToolBar/CoolBar を組み合わせた例

- その他

複数のチェックボックスをまとめて扱える CheckListBox、ウィンドウハンドルを持つテキスト StaticText、日付と時刻を設定するための DateTimePicker、イメージプレビュー付きの OpenPictureDialog/SavePictureDialog などがある。

パッケージ

ネイティブコードと言いながら、ランタイム DLL を必要としているものもある中で、Delphi がネイティブコードを生成し、単独で実行できるアプリケーションを作成できるのは大きな特長である。しかし、この特長のおかげで実行形式のファイルにすべてのコンポーネントが埋め込まれるため、最低でも 200KB 程度の大きさになってしまうというのが残念という声があった。

Delphi 3 は、この問題をスマートに解決した。それがパッケージである。

パッケージとは、コンポーネントライブラリを DLL として提供するものである。もちろん、ランタイムインタープリタ DLL のように実行速度が遅いものではない。ちょうど、Visual C++ や Borland C++が MFC や OWL の DLL を提供しているようなものである。

32 ビットでは、DLL 内の関数の呼び出しにおけるオーバーヘッドはほとんどないので、コンポーネントライブラリを DLL として利用する場合にもほとんど速度は低下しない。この機能は、特に ActiveX コントロールのように複数の部品を配布する場合や、Active フォームのようにネットワークのトラフィックに気を使わなければならない場合に大きな意味を持つ。

もちろん、通常のアプリケーション開発でも役立つだろう。ごく簡単なプログラムの場合なら、パッケージを使って 10KB 程度のサイズになるのだ。

マルチデータベースエンジンのサポート

Delphi でデータベースアプリケーションを作製する場合、Borland Database Engine(BDE)でサポートされている範囲の処理であれば通常は問題ない。製品形態にもよるが、BDE は dBASE/Paradox/テキスト、32 ビット ODBC、InterBase/Oracle /MS SQL Server/Sybase/Informix/DB2 へ直接接続する SQL Link など幅広いデータに対応するスケーラビリティがある。

だが、独自のデータ形式を扱うにはどうすればよいか。従来、Delphi のコンポーネント群は、BDE に依存する仕組みを持っていたので、データアクセスとデータ表示コントロールの両方のコンポーネントを作製する必要があった。

しかし、Delphi 3 ではデータ処理部分から BDE に依存する部分を取り除き、抽象化している。つまり、BDE を使わないデータベースエンジンを自分で提供すれば、そのデータベースエンジンを使って DBGrid や DBEdit のようなビジュアルなコンポーネントを利用できるのである。

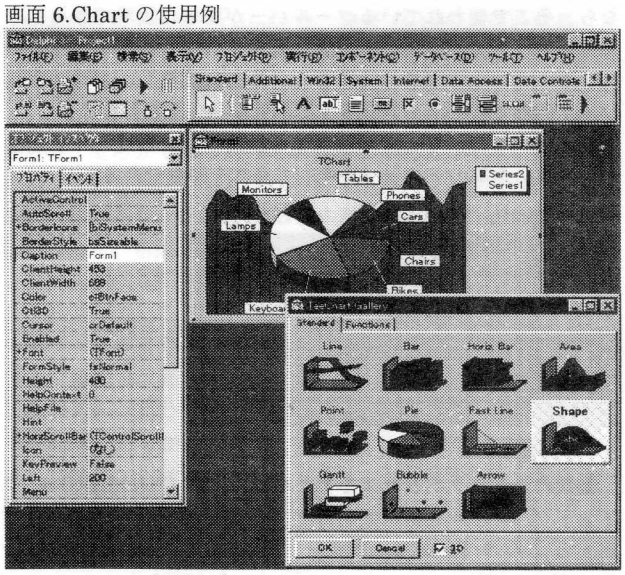
もちろん、レコードの編集やキャッシングを行なう独自のデータベースエンジンを提供することはたやすいことではないだろう。だが、手元に独自のデータベースエンジンがあるなら、それに少し加えるだけで、Delphi の強力なビジュアルコンポーネントが利用できるようになるのだ。たとえば、Delphi 3 Client/Server Suite では、この機能を使って BDE でなくリモートサーバーからデータを受け取る抽象的なデータセットを実現している。これにより、3 層（多層）アプリケーションの開発にもデータベースコンポーネントが使えるのだ。

TeeChart と QuickReports 2.0

いままで、Delphi に組み込まれていたチャート（グラフ）の機能は、ChartFX や FirstImpression のような ActiveX コントロール（OCX）だけだった。これらの最大の問題点は、あくまでもサンプルとして提供されたものであり、しかも英語版であるため日本語の利用に制約があったということである。

だが、Delphi 3 には新たに TeeChart というチャート用のコンポーネントが追加された。TeeChart では、棒グラフや円グラフなど 11 種類の表示形式を使い、任意のグラフを重ね合わせたり、立体化や見出しの位置など詳細なスタイルを設定できる。

画面 6.Chart の使用例



TeeChart には、データベース対応の DBChart やレポート出力用の QRChart があり、幅広い用途に利用できる。TeeChart は、Professional 版以上で提供される機能である。

また、レポート出力を設計するための QuickReports は、バージョン 2.0 になり全面的な改良が施された。QuickReports は、もともとコンポーネントとして実装されているため、イベントハンドラなどを活用して、レポート出力を完全に制御できる。

新しい QuickReports では、レポート出力の核となる QuickRep コンポーネントは、フォーム上にグリッドを表示するビジュアルコンポーネントになった。この上に、レポート出力用のコンポーネントを配置でき、従来よりもずっとわかりやすくなっている。また、詳細なレポートを設計するためのコンポーネントや書式付きテキストを扱うコンポーネントなど、かなり拡張されている。

さらに、レポート出力は、ただ印刷するだけでなく、テキスト形式や HTML 形式でエクスポートできるようになった。

QuickReport ウィザードを使えば、テーブルを参照するレポートフォームを簡単に作成できる。

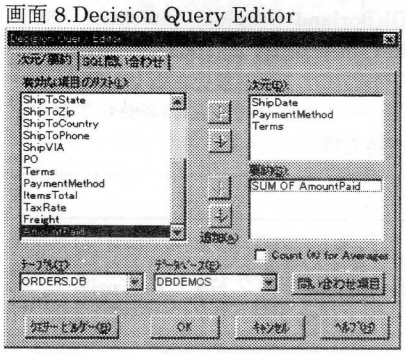
画面 7.QuickReports 2.0 の使用例

Decision Cube

Decision Cube は、Delphi 3 Client/Server Suite で提供されるコンポーネント群で、多次元のデータ解析を実行し、特定の目的に対応した調査に利用したり、重要な意思決定を求められるユーザーに高度な情報を提供する。Decision Cube のクロスタブは、コンポーネントで提供されていることで、アプリケーション固有のデータ解析のためにカスタマイズでき、前述の Active フォームでも利用できるため、Web ブラウザでダイナミックに変換する情報を表示させたい場合にも利用できる。

Decision Cube を使うのは、非常にやさしい。まず、問い合わせのための DecisionQuery というコンポーネントをフォームに配置し、ダブルクリックして Decision Query Editor を呼び出す（画面 8）。

画面 8.Decision Query Editor



ここで、エリアスやテーブルを選び、解析したい項目を選ぶ。このコンポーネントは、Decision Cube Editor を呼び出すという以外は、Query コンポーネントとたいして違いはなく、自分で SQL 文を記述したり、Visual Query Builder を使ってもよい。

設定が終わったら DecisionCube を配置して、DataSet プロパティに DecisionQuery1 を設定する。

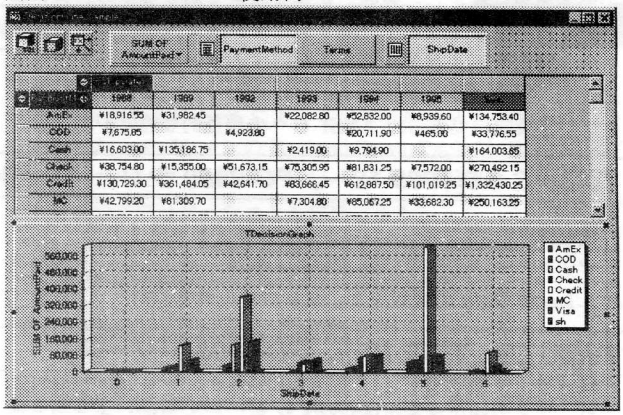
さらに、DecisionSource を配置し、DecisionCube プロパティに DecisionCube1 を設定する。これは、データ処理における DataSource のようなもので、DecisionCube と表示用コントロール（DecisionPivot、DecisionGrid、DecisionChart）の仲介役となっている。

DecisionCube では、解析のために許されるメモリや個々の次元や要約に関する情報を設定できる。たとえば、四半期ごとの解析といった設定が簡単にできる。

フォームに表示用コントロールを配置し、DecisionSource プロパティに DecisionSource1 を選ぶと、指定された解析が解析され、それぞれの形式で表示される。

DecisionGrid と DecisionChart は、それぞれ解析した結果をグリッドやチャートで表示する。DecisionPivot は、解析する条件を変更するためのコントロールで、条件を変更したり、ピボット（軸）を入れ替えたりするために使う。もっとも、軸の変更や条件の切り替えは、グリッドの左上にある見出しをドラッグ＆ドロップしたり、(+)や(−)の印をクリックしてもよい。

画面 9.Decision Cube の使用例



Decision Cube の果たす役割は、限られた紙面で紹介できるほど小さなものではない。しかし、高価だった従来の多次元解析ツールに匹敵する機能を、作成するアプリケーションに埋め込んで利用できるというのは、何にも変え難い特長といってよいだろう。いままで同様の機能を取り込もうとしていた開発者に申し訳ないくらいである。

Decision Cube は、Delphi がエンタープライズ開発での圧倒的地位を占める重要な鍵となるだろう。

Web 対応アプリケーションの開発

Delphi 3 の Professional 版以上には、従来同様 Internet Solutions Pack がバンドルされる。Delphi 3 Client/Server Suite には、Web サーバー側のアプリケーションを開発するための充実した機能が提供される。

画面 10.コンポーネントパレットの Internet ページ

コンポーネントパレットの Internet ページの左側にあるものは、Internet Solutions Pack の ActiveX コントロール群であるが、その右には Delphi 固有のコンポーネントが用意されている。

たとえば、WebDispatcher はデータモジュールを Web モジュールに変換し、HTTP の要求メッセージに対応するアプリケーションを開発するためのものである。

また、QueryTableProducer や DataSetTableProducer は、指定された問い合わせやデータセットに対応する HTML テキストを生成するためのものである。どのような HTML テキストを生成するかは、プレビュー画面付きのダイアログボックスで設定できる。もちろん、ここでもリモートサーバーからデータを参照できる。

おわりに

Delphi 3 の新機能は、この他にも COM サポートの組み込み（ネイティブ COM）、Borland Database Engine による Microsoft Access のデータベース処理、DBRichEdit など新しいコンポーネントの拡張やプロパティの追加、オブジェクトリポジトリの共有、データベースデスクトップのサポートなどがある。また、本文でも触れたが Delphi 3 Client/Server Suite では NT 4.0 などで提供されている DCOM を使って、リモートサーバーで提供されるデータベースを処理することもできる。

もちろん、フォームの継承、オブジェクトインスペクタによるイベントハンドラの共有、ドラッグ＆ドロップでフォームを設計できるデータベースエクスプローラや項目エディタといった従来からの機能は、そのまま引き継がれている。

Delphi 3 の強力な新機能により、今後ますます Delphi 利用者が増えるだろう。

CB MAGAZINE SPECIAL

ボーランドが送り出したC/C++の歴史

CB MAGAZINE 編集部

米ボーランドがはじめてCの処理系をリリースしたのは1987年、ちょうど10年前のことである。当時は、ちょうどC言語への期待が非常に高まっていた時期でもある。すでにTurbo Pascalで安価かつ高性能な処理系を提供することで知られていたボーランドが、満を持してリリースしたTurbo C 1.0はそうした期待に応えるものだったと言えよう。C処理系が乱立する中での、Turbo Cの高速性や機能性は他を圧倒し、不動の地位を築いたのであった。今また、C++Builderが新たなC++プログラミングの時代を導くために登場した。
温故知新。ここで、C++Builder以前のボーランドのC/C++製品を振り返ってみよう。

Turbo C 1.5[1988.2～、PC-9800シリーズ]

エディタ、コンパイラ、リンカ、オンラインヘルプをひとつの開発環境に統合した製品。安価な設定でありながら、メニューによる容易な操作とキー割り当てをカスタマイズできるエディタ、言語やライブラリ関数に対応するオンラインヘルプなどにより、プログラミング作業が容易になった。また、コンパイラは規格策定中であったANSI Cへの積極的な対応や充実したエラー・警告メッセージ、1パスコンパイルによる高速処理、6つのメモリモデル、BGI(Borland Graphics Interface)、BIOSや画面制御ライブラリなどが標準添付された。当時は珍しかったライブラリソースコードのオプション販売も行なわれた。

画期的なコストパフォーマンスにより、初心者からプロフェッショナルまで幅広い支持を得た。

Turbo C 2.0[1988.12～、PC-9800シリーズ、FM-R、AX]

統合開発環境にソースレベルデバッガが組み込まれた。また、エディタバッファとしてのEMSの利用、エミュレータの高速化、Turbo Helpのサポート、コンパイル速度の向上、ANSI C対応の80bit long doubleなどが追加された。

Turbo Assembler&DebuggerをバンドルしたTurbo C 2.0 Professionalも発売された（1989.4）。

Turbo C++[1990.12～、PC-9800シリーズ]

ボーランドにとって、はじめてのC++対応製品。全面的な改良が施され、Turbo C++と命名された（バージョンは1.0）。ANSI Cに加え、多重継承や関数の自由なオーバーロードを含むAT&T C++ 2.0に対応した。当時、比較的多かったトランスレータ形式のものに比べ、完全なC++ネイティブコンパイラであり、ソースコードデバッグなどにも制約を受けることがなかった。また、Programmer's Platformと呼ばれる新しい統合開発環境では、大規模ファイルやアンドゥ・リドゥに対応したエディタや自動依存情報によって効率化されたプロジェクト機能をサポートした。また、EMS/XMS対応の効率的なオーバーレイ技術VROOMM(Virtual Runtime Object Oriented Memory Manager)が採用された。

Turbo Debugger&Toolsをバンドルした Turbo C++ Professionalも発売された。

後に、添付マニュアルの一部を別売とし価格を改訂したTurbo C++ 2nd Editionが発売された（1991.4）。

Borland C++ 2.0[1991.8～、PC-9800シリーズ]

ボーランドではじめてWindows 3.0対応アプリケーションを開発できるようになったC/C++コンパイラ。統合開発環境を使ってDOS/Windowsに対応するアプリケーションを構築できる。また、プロテクトモード対応のコンパイラや統合開発環境によって、大規模なアプリケーションも構築できた。また、すべてのWindowsリソースをビジュアルに編集できるResource WorkshopやTurbo Debugger/Profiler/Assemblerなどプロフェッショナル向けの機能も標準添付された。

Borland C++ & Application Frameworks 2.0[1991.12～、PC-9800シリーズ]

Borland C++ 2.0に、Windows用クラスライブラリObjectWindowsとDOS用クラスライブラリTurbo Vision、およびライブラリソースコードを標準添付した製品。ObjectWindowsは、Borland C++で拡張されたDDVT(Dynamic Dispatch Virtual Table)という仕組みを使うことで、膨大なWindowsメッセージを効率的に処理することができた。Turbo Visionは、Turbo Pascal 6.0に開発されたものをC++に移植したもので、メニュー、マルチウィンドウ、ダイアログボックスなどを使ったキャラクタベースのユーザーインターフェースを効率的に構築するためのクラスライブラリである。

Borland C++ & Application Frameworks 3.0[1992.4～、PC-9800シリーズ、DOS/V]

C++のテンプレート機能をはじめとするAT&T C++ 3.0を実現し、グローバルな最適化機能が組み込まれた。また、アプリケーションの構築時間を大幅に短縮できるプリコンパイルヘッダが採用された。Windows用の統合開発環境（Turbo C++ for Windows）が標準添付され、Windows上でアプリケーションを開発できるようになった。さらに、Windows上でDOSコンソールをエミュレーションするEasyWinなどライブラリも強化された。

Borland C++ 3.0[1992.6～、PC-9800シリーズ、DOS/V]

Borland C++&App. Frameworks 3.0からObjectWindows、Turbo Vision、ライブラリソースコードを省いた製品。

Turbo C++ for Windows 3.0[1992.6〜、PC-9800 シリーズ、DOS/V]

Windows アプリケーションの開発のみをサポートする C/C++コンパイラ。Windows 上で動作する統合開発環境、ObjectWindows、テンプレートもサポートする C++コンパイラと ANSI C 対応コンパイラが組み込まれている。 また、Turbo Debugger for Windows が標準添付されている。グローバルな最適化機能はない。

Borland C++ & Application Frameworks 3.1
Borland C++ 3.1[1993.6〜、PC-9800 シリーズ、DOS/V]

Windows 3.1 に対応した Borland C++。最適化を含む Borland C++のコンパイル機能を利用できる Windows 用と DOS 用の統合開発環境を持ち、カラー構文強調表示、386 コード生成や効率的な C++の呼び出し形式などがサポートされた。

Turbo C++ for Windows 3.1[1993.6〜、PC-9800 シリーズ、DOS/V]

Windows 3.1 に対応した Turbo C++。統合開発環境のエディタは、カラー構文強調表示をサポート。Windows 3.1 のためのインポートライブラリやヘッダファイルが提供され、一般保護違反をトラップする WinSpector が標準添付された。

Borland C++ 4.0J[1994.9〜、PC-9800 シリーズ&DOS/V 両対応]

DOS/Windows 3.x/Windows NT 対応の 16/32 ビットアプリケーションを開発できる C/C++開発システム。Windows 3.1 用の統合開発環境から、単一のプロジェクトでこれらのすべてに対応するアプリケーションを開発できる。コンパイラは、例外処理や実行時型情報、テンプレートをサポートしている。また、標準的な C++によって全面的に書換えられた ObjectWindows 2.0 は、Doc/View やツールバーなどを 16/32 ビットの両方で利用でき、ソースコードの共通化を実現している。さらに、AppExpert/ClassExpert によるビジュアルプログラミングもサポートした。オプションの Borland PowerPack for DOS[16/32] と組み合わせれば、16 ビット/32 ビットの DOS プロテクトモード対応のアプリケーションも作成できる。

後に、Borland Database Engine 2.0 付きの Borland C++ 4.0J with Database Engine が発売された(1995.9)。

*Turbo C++ 4.0J for DOS[1995.4〜、PC-9800 シリーズ&DOS/V 両対応](*発売中)*

プロテクトモードで動作し、充実した機能を持つ統合開発環境をサポート。テンプレートや例外処理、実行時型情報などの C++機能、オプションの Borland PowerPack for DOS16 と組み合わせてプロテクトモードアプリケーションの開発にも対応している。PC-9800 シリーズと DOS/V の両方で動作するライブラリや両機種用の Turbo Debugger が標準添付されている。

Borland C++ 4.5J
Borland C++ 4.5J with Database Engine[1995.12〜、PC-9800 シリーズ&DOS/V 両対応]

DOS/Win16/Win32 対応の 16/32 ビットアプリケーションを開発できる C/C++開発システム。OLE2 プログラミングを効率化する ObjectComponents Framework(OCF)がサポートされ、OLE サーバー、OLE コンテナ、OLE オートメーションなどのプログラミングが容易になり、ObjectWindows 2.5 は OCF によって OLE 対応が拡張された。また、VBX を 32 ビットアプリケーションで利用する機能や ObjectWindows を使ったサンプルゲームが追加された。 さらに、Windows 95 対応のヘッダファイル・インポートライブラリが提供された。

Borland C++ 4.5J with Database Engine は、Paradox や dBASE で使われている 16 ビット版の Borland Database Engine(BDE)の開発キットをバンドルした製品。ユーザーアプリケーションに BDE を組み込むことで、Paradox や dBASE データを容易にアクセスできる。

Borland C++ 5.0J
*Borland C++ Development Suite 5.0J[1996.8〜、Windows 95&NT 3.51](*発売中)*

Windows 95/NT に対応した 32 ビット対応の統合開発環境を持ち、Windows 95/NT/3.1/DOS 対応の 32/16 ビットアプリケーションを開発できる製品。AppExpert/ClassExpert によるビジュアルプログラミング、強力なカスタマイズ機能、マルチスレッド対応のデバッガ、リソースエディタ、バックグラウンドコンパイルなどの機能を備える。コンパイラは、例外処理、名前空間、explicit/mutable、標準 C++ライブラリなどの新しい C++機能をサポートする。Windows 95 の機能に対応した ObjectWindows が添付され、 一部の Win95 コントロールは Windows 3.1 でも使えるようエミュレーションされる。32 ビットでの VBX のサポートや OCX コンテナの 32/16 ビットでのサポートを含め、広い範囲で 32/16 ビットアプリケーションの並行開発を実現している。また、Visual Database Tools により簡単なマウス操作でデータベースアプリケーションも作成できる。

Borland C++ Development Suite 5.0J では Borland C++ 5.0J に加え、メモリのバグをすばやく見つける CodeGuard 32/16、インストーラ作成ユーティリティ InstallShield Express、バージョン管理を支援する PVCS などが含まれている。

*Turbo C++ 5.0J for Windows 95&Windows NT[1996.9〜、Windows 95&NT 3.51](*発売中)*

Windows 95/NT に対応した 32 ビット対応の統合開発環境を持つ製品。Borland C++ 5.0J から、16 ビット対応、グローバルな最適化、コマンドラインツール、ライブラリソースコードを除いたもの。

(*発売中)と明記されている製品以外は、現在は販売しておりません。

待望報告

Java 開発が変わる！ JBuilder の真髄

Java Coffee 普及委員会

JBuilder

大ブームを巻き起こした Java も、今は落ち着いた状態に見える。理由のひとつは決定的なツールの欠如だろう。ブームに乗らんとばかりにリリースされたツールは、本来開発ツールにあるべき使いやすさや標準化など重要なポイントが押さえ切れていなかった。だが、JBuilder は違う。長期に渡る検討が繰り返され、ほんものの開発ツールとしての実力が備わっている。

JBuilder の特長

Java にとって、これまで開発ツールに恵まれなかったのはアンラッキーだったというしかない。ボーランドも、Latte というコードネームで早くから Java 対応ツールのリリースをアナウンスしていたが、これまで実物を手にする機会がなかった。

だが、ボーランドは長い時間を無駄に過ごしていたわけではない。JavaBeans（Java コンポーネント）への協力をはじめ、実用的な開発ツールとしての体力を着々と身に付けてきたのである。JBuilder と名付けられた Java 開発ツールは、Web 上で配布されるクロスプラットフォーム対応のアプリケーションやコンポーネント、さらに多層開発に対応するアプリケーションをすばやく構築できるツールである。

JBuilder には、次のような特長がある。

- コンポーネントを使ったビジュアル開発の実現。
- 主要な業界標準に対応することで、最大限の柔軟性を提供。
- オープンかつ拡張性の高い環境。
- デスクトップからエンタープライズまで対応するスケーラビリティの高いデータアクセスツール DataDirector。
- プロフェッショナルの使用に耐える開発ツールと Java や Web に最適化されたコンパイラ。

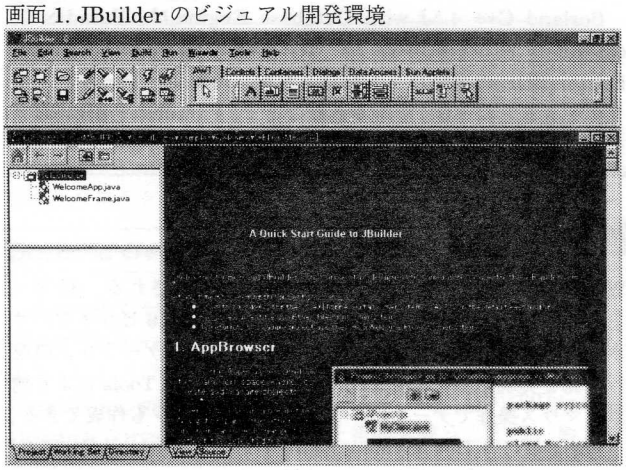
こうした、既存のツールに見られなかった特長により、信頼性やパフォーマンスの高い Java アプリケーション／アプレットを開発できるようになるだろう。

ビジュアル開発

JBuilder は、数々の賞を受賞したボーランドの Delphi の開発環境をモデルとしており、さらにすぐれた性能を持つアプレットやアプリケーションを開発するために最適化されたツール群が組み込まれている。

画面 1 は JBuilder を起動した直後のようすである。Delphi のようなスピードバーやコンポーネントパレットが用意されていることがおわかりいただけるだろう。しかし、ただ Delphi そっくりの環境に終始しているわけではない。ここには、純粋な Java コードとビジュアル開発を両立させる能力が統合されているのである。

画面 1. JBuilder のビジュアル開発環境



プロジェクトウィザードを呼び出せば、空のアプリケーションプロジェクトが作られる。アプリケーションやアプレットを作成するウィザードも用意されている。また、コンポーネントをドラッグ＆ドロップして、アプレットやアプリケーションをビジュアルに設計できる。コンポーネントは定義済みのものも使えるし、自分で拡張することもできる。オブジェクトインスペクタで、コンポーネントのプロパティを設定したり、イベントハンドラに Java コードを割り当てた後、コンパイルする。これで、PC でも、Macintosh でも、UNIX でも、ネットワークコンピュータでも Java をサポートしている環境ならどこでも実行できるプログラムができあがる。

JBuilder のビジュアル開発環境を使えば、信頼性の高いアプリケーションをすばやく開発できる。JBuilder に組み込まれた 2Way-Tool は、ビジュアルな設計とそれに対応する純粋な Java コードとを連携させる。このような処理のためにも、Java に特殊な拡張は施されているわけではなく、常に純粋な Java コードが生成される。

JBuilder の対話型開発ツールによって、データベースとの接続も簡単になる。JBuilder は、ビジュアル開発でもソースコード開発でも制約はない。JBuilder は、アプリケーションのプロトタイプをすばやく作成できるというだけでなく、製品品質に移行させるための機能を持っている。

また、オープンで拡張性のある開発環境には、新しいアドイン、エキスパート、あるいはコンポーネントを組み込める。プログラマーは、必要に応じて自由に開発環境をカスタマイズできるのである。また、ビジュアルコンポーネントデザイナによって、独自のコンポーネントを作成することもできる。

主要な業界標準のサポート

ボーランドは、JBuilder に最大限の柔軟性を与えるため、最新の標準仕様を盛り込む予定である。JBuilder の OpenVM(TM) アーキテクチャは、Microsoft Internet Explorer や Netscape Navigator(TM)、さらに Sun のプラットフォームで使えるアプリケーションの開発とデバッグを実現する。

また、JBuilder は信頼性の増した JDK 1.1 をサポートし、さらに Java のコンポーネントモデルである JavaBeans については、開発レベルで JavaSoft に協力している。

JBuilder はユニークな特長を持っているが、それは独自拡張に基づくものではない。Java 自身の大きな特長である移植性を損なわないように配慮された上で、開発者に幅広い選択肢を与えているのである。

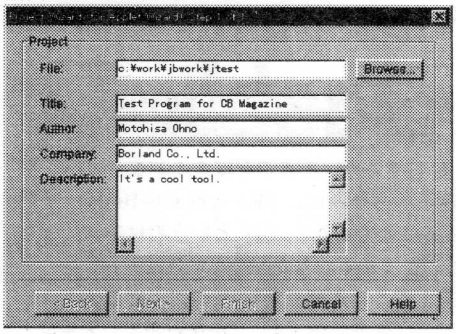
AppBrowser

AppBrowser は、プロジェクト管理やオブジェクトの操作を単純化する。画面 1 でプロジェクトや個々のオブジェクト、ソースコード、フォームデザイナを統合しているのが AppBrowser である。数多く特長を持つ JBuilder であるが、AppBrowser によってウィンドウが乱雑にオープンされるのを防ぎ、すっきりとしたユーザーインターフェースを実現している。もちろん、エディタやブラウザは必要に応じてカスタマイズできるようになっている。JBuilder は、複数のプロジェクトのオープンや、複雑なアプリケーションプロジェクトを容易に管理できるように設計されている。

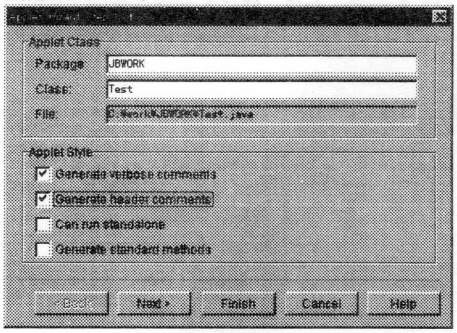
実際の開発

アプリケーションやアプレットを作成するためには、専用のウィザードツールを利用すればよい。プロジェクトがない状態で、アプレットウィザードを呼び出せば、まずプロジェクトウィザード（画面 2）を使って空のプロジェクトを作成し、アプレットウィザード（画面 3）で順に必要な項目を設定すれば、基本的な枠組みが生成される。

画面 2. プロジェクトウィザード

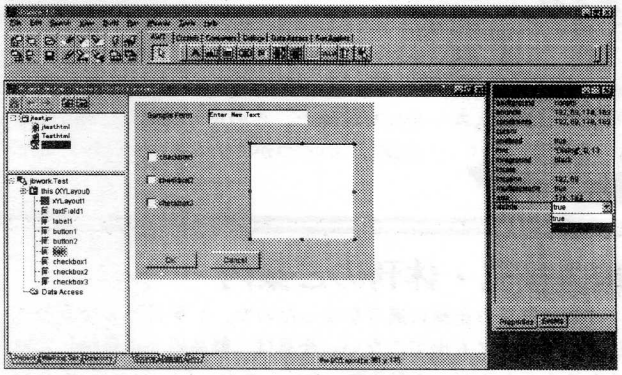


画面 3. アプレットウィザード



JBuilder の実際の設計画面を画面 4 に示す。フォームには、コンポーネントパレットから選んだボタンやエディットなどのコンポーネントが配置されており、オブジェクトインスペクタでコンポーネントのプロパティを設定できる。

画面 4. フォームの設計とオブジェクトインスペクタ



フォームにコンポーネントを配置すると、即座にソースコードに反映される。たとえば、画面 4 のフォームに対しては、次のようなプログラムコードが生成される。

```
public class Test extends Applet {
  public XYLayout xYLayout1 = new XYLayout();
  public boolean isStandalone = false;
  TextField textField1 = new TextField();
  Label label1 = new Label();
  Button button1 = new Button();
  Button button2 = new Button();
  List list1 = new List();
  Checkbox checkbox1 = new Checkbox();
  Checkbox checkbox2 = new Checkbox();
  Checkbox checkbox3 = new Checkbox();
    ...
```

もちろん、オブジェクトインスペクタを使えば、コンポーネントのイベントハンドラのためのメソッドを自動生成し、Java でプログラミングできる。

DataDirector

JBuilder の DataDirector アーキテクチャを使うことで、堅牢でスケーラビリティの高いクライアント/サーバーアプリケーションを迅速に開発できる。DataDirector は、強力なデータベースコンポーネントとツールの総称であり、データベース対応のアプリケーション構築を手助けする。DataDirector は、JDBC や ODBC 互換のデータソースへのアクセスをサポートしている。データソースへの接続は、Oracle、Sybase、Informix、Interbase などのネイティブドライバを使って実現されている。これには、単独のマシンで ANSI 92 互換の SQL エンジンを使える InterBase サーバーも含まれる

Java 最適化ツール群

JBuilder を使って Web 上で配布されるクロスプラットフォームのアプリケーションを迅速に開発するため、JBuilder は純粋な Java アプリケーションを生成する。JBuilder は、クロスプラットフォームのアプリケーション、ヘルプ、コンポーネント、接続性をサポートし、アプリケーションの移植性が維持される。

さらに、マルチプラットフォーム上でのジャストインタイムコンパイラが、アプリケーションの実行を高速化する。インターネット上でのリモートデバッグは、マルチプラットフォームでの開発をさらに容易にする。

行き届いたサポートツール

強力で使いやすい統合デバッガによって、プログラム上の問題点をすばやく特定し修正できる。高速なJavaコンパイラと自動依存解析によって、コンパイル時間にいらいらさせられることもない。また、充実したサンプルは、Javaプログラミングの学習に役立つだろう。また、オブジェクトギャラリーには、プロジェクト、コンテナ、エキスパート、コンポーネント、クラス、ソースファイル、コードの一部などあらゆる情報を再利用のために保存しておける。

C++の標準テンプレートライブラリ(STL)をモデルにしたJava Generic Library(JGL)を使えば、信頼性が高いデータ構造をアプリケーションに組み込むこともできる。また、パフォーマンスの高いオブジェクト指向データベースであるObject Design社のObjectStore PSE for Javaによって、永続的な記憶をアプリケーションに組み込むこともできる。

まとめ

最後に、JBuilderを使ってみたいという人に朗報がある。米ボーランドは、先月末からホームページでJBuilderの評価用リリースの申し込みを受付を始めた。この申し込みは、Borland Onlineのメンバーに登録し、いくつかのアンケートに答えるだけである（ただし、申し込み多数の場合は抽選となる）。ほんもののJavaツールを探している人は、是非トライしてみて欲しい。

（参照先：http://www.borland.com/）

編集後記・休刊のご案内

当初は単発の企画に過ぎなかったので、3年目ともなるとさほどのアイデアも出てこない。今回は、製品紹介に終始しているところもあるが、ご容赦願いたい。また、JBuilderとDelphi 3については、発売前の製品であるので、実際に発売されるときの機能や名称は変更される可能性がある。この点も、あらかじめご了承いただきたい。

さて、2度あることは3度あるのか、3度目の正直なのか、ともかくも第3号が発行された。いつものことだが、C MAGAZINE編集部の寛容さに感謝するばかりである。

カタログなどに比べさほどの数を配布していないので、経緯をご存じない方も多いと思うが、第1号が「BC MAGAZINE」、第2号が「D MAGAZINE」であった。去年の今ごろは「このペースだと、来年はEにするか、Jまでとばすか」などという話をしていたものである。ちなみに、「E」とはC++Builderの開発コード名Ebonyのことであり、「J」というのは本文でも紹介したJBuilderのことである。しかし、C++Builderという名前で製品が出た以上、ここは逆戻りさせて「CB」としようということになったのだ。この先には「A」しかないので、来年はアセンブラの特集でもしようかと思う今日この頃である。

もっとも4度目があるかどうかは、誰にもわからない。(O)

CB MAGAZINE　97年4月1日号
平成9年4月1日発行
制作・著作　ボーランド株式会社

※注意　本誌の内容について、弊社インフォメーションセンターまたはテクニカルサポートでのお問い合わせはご遠慮ください。
本誌で紹介された製品のカタログについては、以下までご連絡ください。
〒151　東京都渋谷区笹塚1-64-8　笹塚サウスビル　ボーランド株式会社
TEL 03-5350-9380　FAX 03-5350-9369

価格 ¥0[5%の消費税込み]