Przegląd i ewaluacja narzędzi do szybkiego tworzenia interfejsu użytkownika (RAD).

Szybkie tworzenie programów użytkowych (Rapid Application Development – RAD) jest mającą już prawie 20 lat techniką szybkiego konstruowania programów, szczególnie przydatną dla systemów, w których istotną rolę odgrywa interfejs użytkownika, natomiast mniejszą przetwarzanie danych

Programy takie mają zbliżoną do siebie strukturę, wiele czynności przy programowaniu daje się w nich zautomatyzować. Typowe dla RAD jest korzystanie z zaawansowanych środowisk wytwarzania oprogramowania (narzędzi CASE) oraz posługiwanie się programowaniem graficznym (visual programming).

Typowym środowiskiem RAD jest oprogramowanie CASE służące do produkcji kodu dokonującego interakcji z bazą danych i tworzącego rozmaite raporty (współcześnie najczęściej taki kod wyposażony jest w internetowy interfejs użytkownika). Środowisko takie zawiera:

- narzędzia interfejsu z bazą danych (generowanie zapytań na podstawie formularzy)
- generator interfejsu użytkownika (formularzy z guzikami, polami itp.)
- powiązania z aplikacjami biurowymi, takimi jak arkusze kalkulacyjne, do dokonywania podstawowych operacji na danych
- narzędzia generowania raportów

W latach 90-tych opracowana została całościowa metodologia tworzenia oprogramowania RAD:

- Oprogramowanie (kolejne iteracje, kolejne wersje prototypów) są tworzone w przedziałach czasowych o ściśle określonej długości (time boxing)

- Jeśli jakieś wymagane elementy nie dają się zrealizować w zadanym przedziale czasowym są opuszczane

- Oprogramowanie jest tworzone przez mały zespół, który często spotyka się z przedstawicielami klienta, aby przedyskutować funkcjonowanie aktualnych prototypów i ewentualnie korygować wymagania dla kolejnych

- Tworzenie kodu opiera się głównie na predefiniowanych elementach, rozmaitych API, bibliotekach, komponentach, zintegrowanych środowiskach wytwarzania kodu; mało jest ręcznego programowania

Techniki RAD krytykowane są pod wieloma względami:

- "szybkie" oznacza często obniżenie standardów niezawodności, efektywności, jakości oprogramowania

- zależność od zastosowanych komponentów może prowadzić do kłopotów z ewolucją oprogramowania, kosztami użytkowania, czasem do problemów prawnych

- stosowanie gotowych komponentów często prowadzi do

  niekompatybilności wymagań – nie są realizowane żądane cechy, natomiast pojawiają się inne niepotrzebne

- programy stają się zbyt podobne jedne do drugich

Techniki RAD jako techniki prototypowania mogą być wykorzystywane w większych projektach w fazie określania wymagań, a także w procesie projektowania.

Spośród znajdujących się aktualnie na rynku narzędzi wspomagających tworzenie programów użytkowych godne uwagi są systemy oprogramowania:

- Borland Delphi,
- Borland C++ Builder,
- Sybase Power++,
- Microsoft Visual C++,
- Microsoft Visual .NET C#

Mówimy o nich, że są to środowiska programistyczne.

Wszystkie te systemy działają w 32 – bitowym środowisku Windows 95/98/2000/XP NT i wykorzystują jego mechanizmy GUI (Graphic User Interface). Narzędzia tej klasy określa się wspólnym mianem RAD (Rapid Application Developement – błyskawiczne tworzenie aplikacji).

W systemach C++ Builder, Delphi, Power++, C# każdy formularz jest zdefiniowany jako klasa.

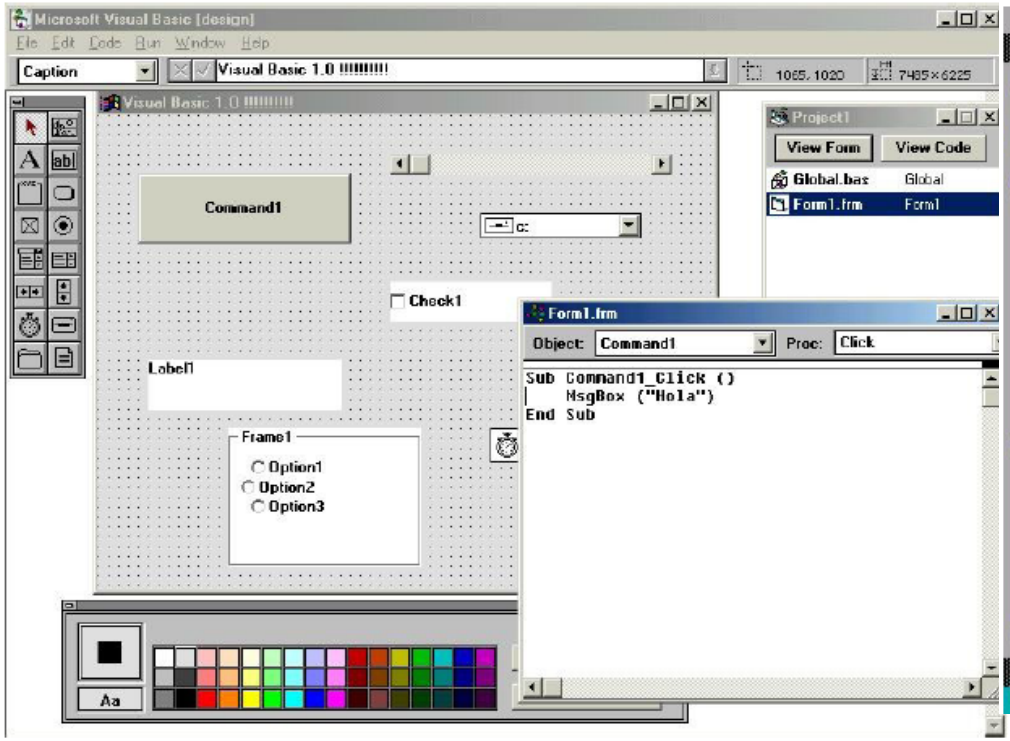
Fakt ten ma następujące konsekwencje:

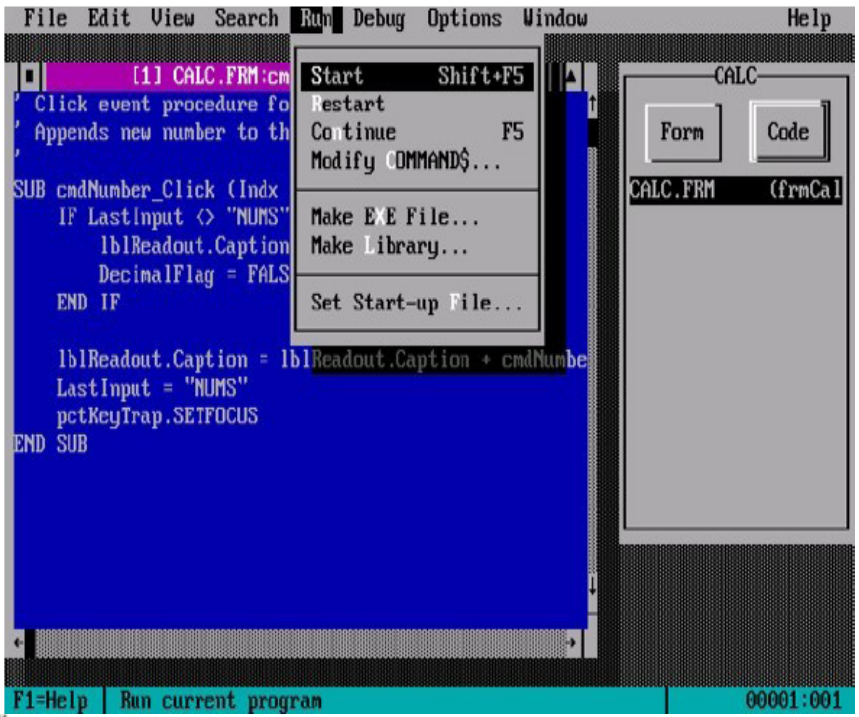
1. Formularz jest typem danych, a nie obiektem danych. Jeden z formularzy składających się na aplikację jest formularzem głównym. Jest on tworzony automatycznie w momencie uruchomienia programu i służy jako okienko inicjacyjne.
2. Można utworzyć wiele obiektów tego samego typu formularza.

3. Każda klasa formularza ma związane ze sobą własności. Jedne z nich dotyczą postaci formularza widocznego na ekranie (kolor, wielkość itp.). inne z nich mają wpływ na zachowanie się – np. określają, czy wielkość formularza może być zmieniana. Projektując formularz określamy wartości początkowe jego własności. Zestaw wartości początkowych jest używany zawsze wtedy, gdy nasz program tworzy nowy obiekt klasy formularz. W czasie wykonywania programu wiele z tych własności można zmienić.

4. Każda klasa formularza ma zbiór związanych z nią metod. Metoda jest funkcją, dzięki której można wykonać akcję z użyciem formularza. Przykładowo, formularz ma metody sprawdzania zmian własności formularza.
5. We wszystkich tych systemach istnieje możliwość dodania własnej metody do klasy formularza. Jest to przydatne wtedy, gdy chcemy zdefiniować procedurę, która może być użyta przez inną funkcję wewnątrz klasy lub gdy chcemy zapewnić kontrolowany dostęp do klasy z obiektów znajdujących się poza klasą.

- Pierwszym popularnym środowiskiem o cechach RAD był *VisualBasic firmy*
- Microsoft (koniec lat 80-tych XX w., oficjalna premiera 1991).
- VB zyskał ogromną popularność i odebrał spory rynek twórcom klasycznych
- kompilatorów i środowisk IDE zorientowanych na kod — m.in. firmie Borland.



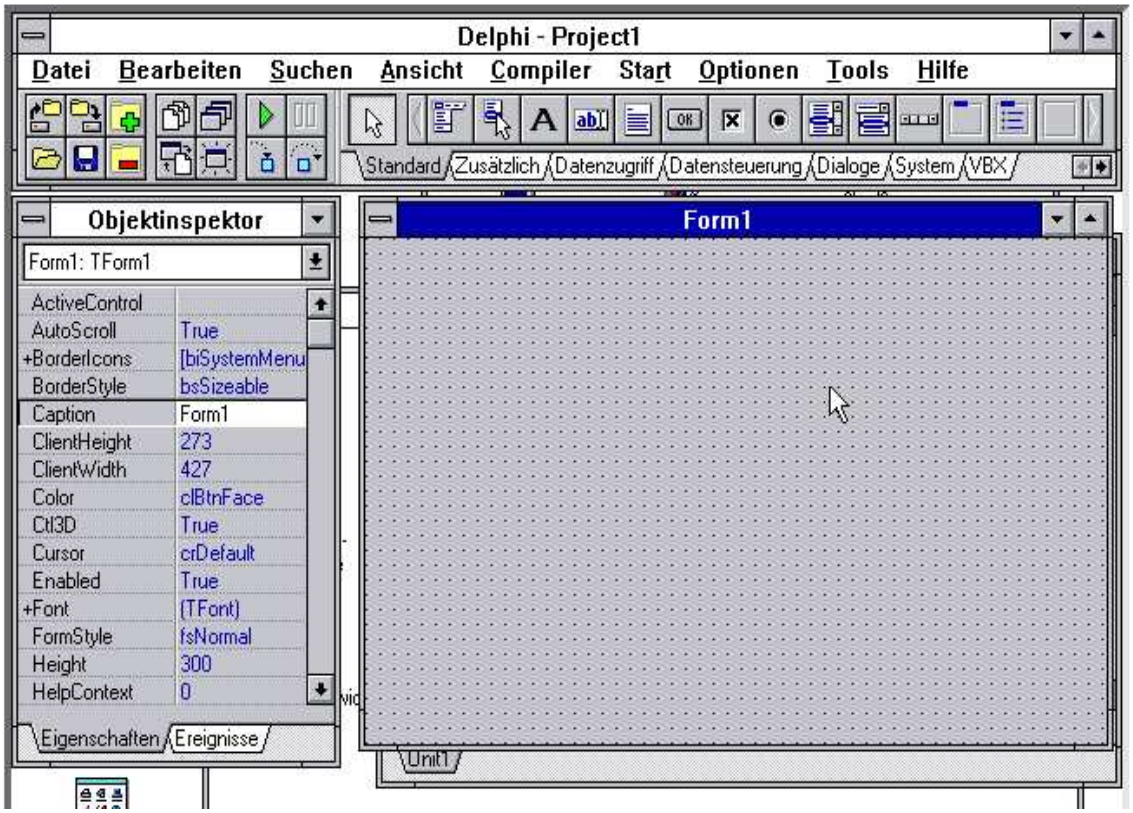


Delphi 1

Pojawienie się Delphi 1 to był prawdziwy przełom. Narzędzie oferowało możliwość wizualnego projektowania aplikacji, jak i wykorzystywania modułów tekstowych, jak również tworzenie wielu niezależnych plików wykonywalnych, wykorzystanie bibliotek DLL, dostęp do mechanizmów obsługi baz danych itp. Był to świetny kompromis między "pisaniem" i "budowaniem".

Efektywność Delphi była tak uderzająca, że na jego określenie (jego i podobnych narzędzi w przyszłości) stworzono powszechny dziś akronim RAD.

Dzięki kombinacji kompilatora, narzędzi projektowania wizualnego oraz efektywnej obsługi baz danych Delphi było bardzo kusząca propozycją dla entuzjastów Visual Basic'a..



Delphi 2

W 1996 roku Delphi stało się 32-bitowe i mogło pracować z Windows 95 / NT. Oferowało, oprócz bardzo efektywnego i zoptymalizowanego 32-bitowego kompilatora, rozszerzoną bibliotekę komponentów, o wiele lepsze mechanizmy obsługi baz danych, nową obsługę łańcuchów tekstowych, wizualne dziedziczenie formularzy, wykorzystanie OLE i zgodność z 16-bitowymi projektami.

Delphi 3

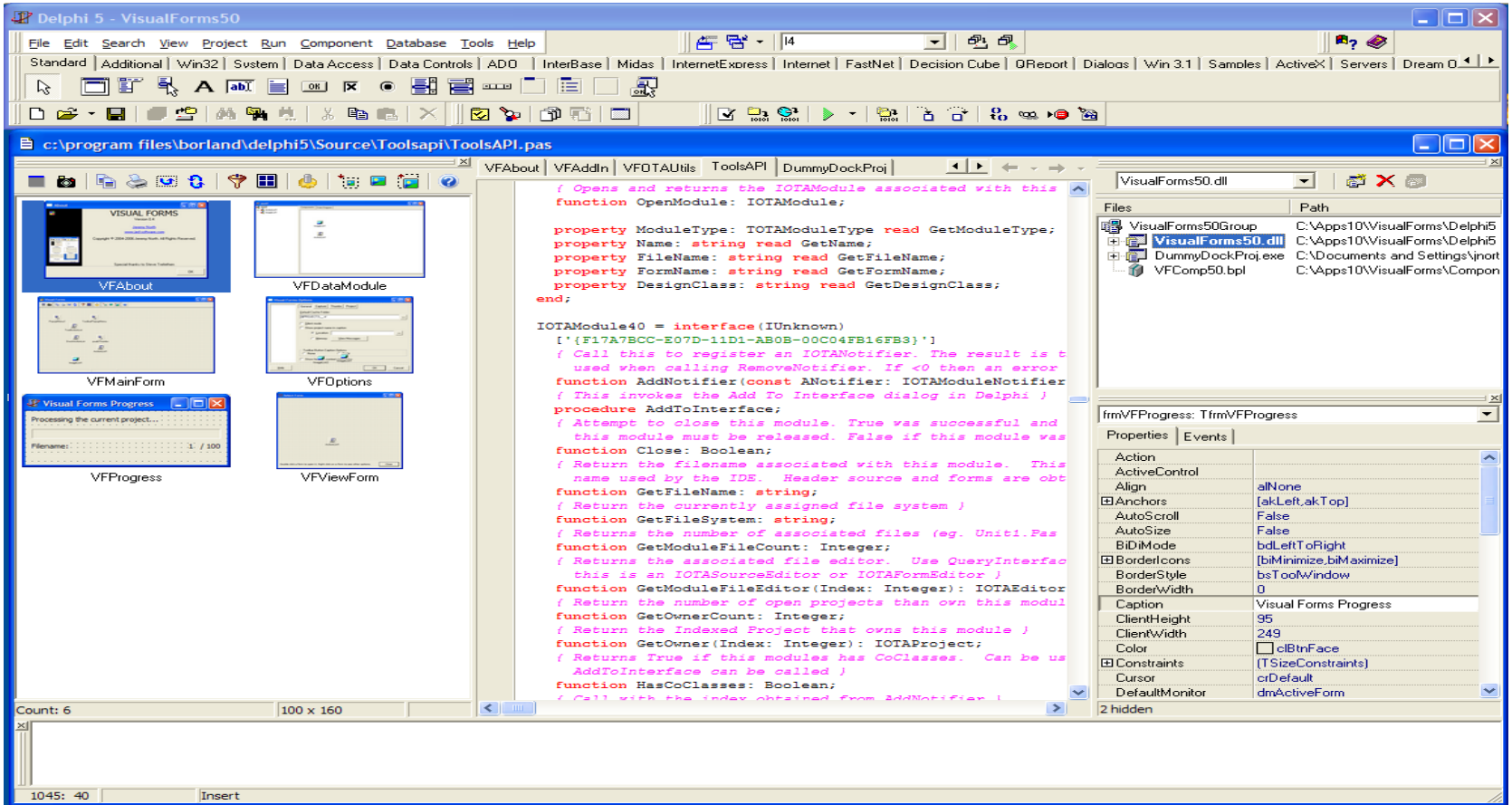
Delphi 3 przyniosło implementację nowych technologii towarzyszących programowaniu w Windows - obiektów COM, kontrolek ActiveX, aplikacji serwerów WWW, wielowarstwowych aplikacji SQL oraz aplikacji "uproszczonego klienta". Poprawiono też środowisko projektowe - implementacji pakietów, konfigurowanie dla poszczególnych projektów palety komponentów, usprawnienia edytora, debuggera

Delphi 4

W tej wersji usprawniono oczywiście środowisko IDE - udoskonalono Eksplorator Modułów (uwypuklający modularno - hierarchiczną strukturę kodu) oraz udoskonalenia edytora w postaci kompletacji definicji klas i usprawnionego nawigowania. Okna i paski narzędzi zyskały własność dokowania (docking), usprawniono też zintegrowany debugger. Programiści baz danych dostali narzędzie do tworzenia wielowarstwowych aplikacji klient - serwer na podstawie takich technologii, jak MIDAS, DCOM, MTS i CORBA..

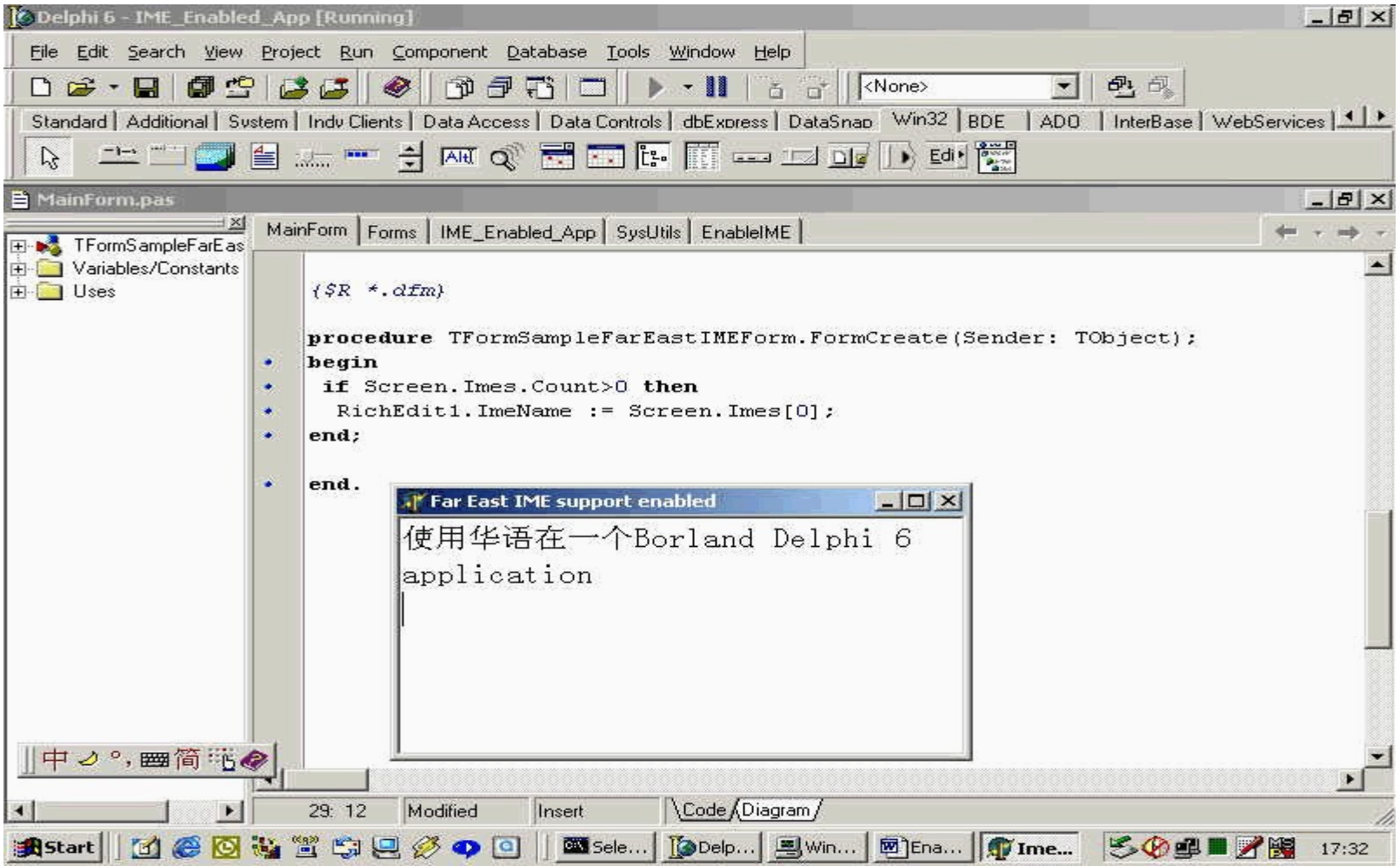
Delphi 5

W Delphi 5 kontynuowano rozpoczęte w Delphi 4 zabiegi upraszczające korzystanie z IDE i z debuggera. Wprowadzono też wiele elementów mających uprościć tworzenie aplikacji związanych z Internetem, takich jak np. Active Serwer Object Wizard, komponenty InternetExpress z obługą XML i nowe elementy MIDAS-a. Poświęcono też dużo czasu i wysiłku, żeby wyposażyć nowe Delphi w stabilność



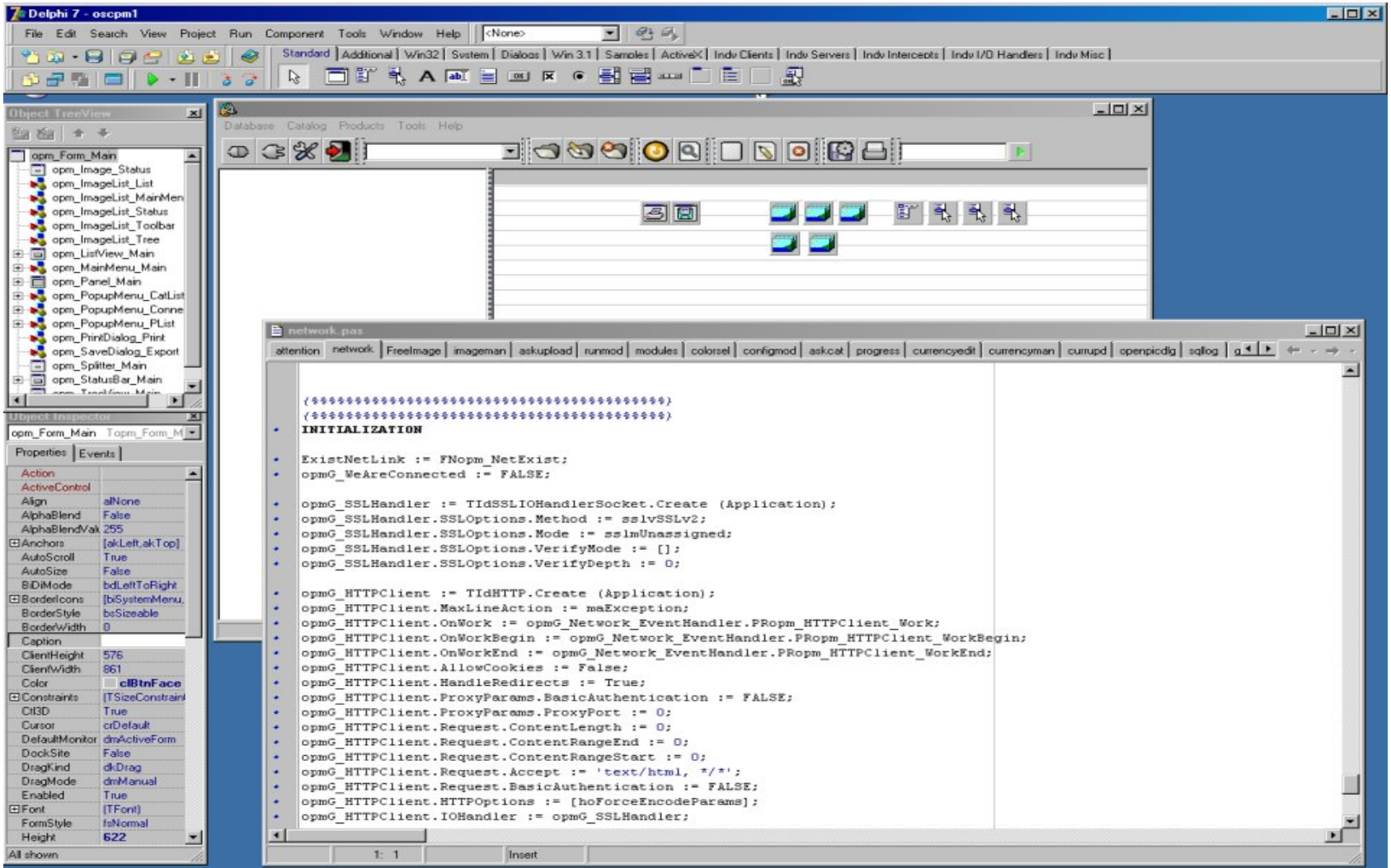
Delphi 6

- Najważniejszą nowością Delphi 6 jest jego zgodność z Kyliksem - narzędziem RAD dla Linuksa. W związku z tym Borland opracował komponenty CLX, czyli komponenty międzyplatformowe. Dzięki temu aplikację można przenosić pomiędzy Windowsem i Linuksem.

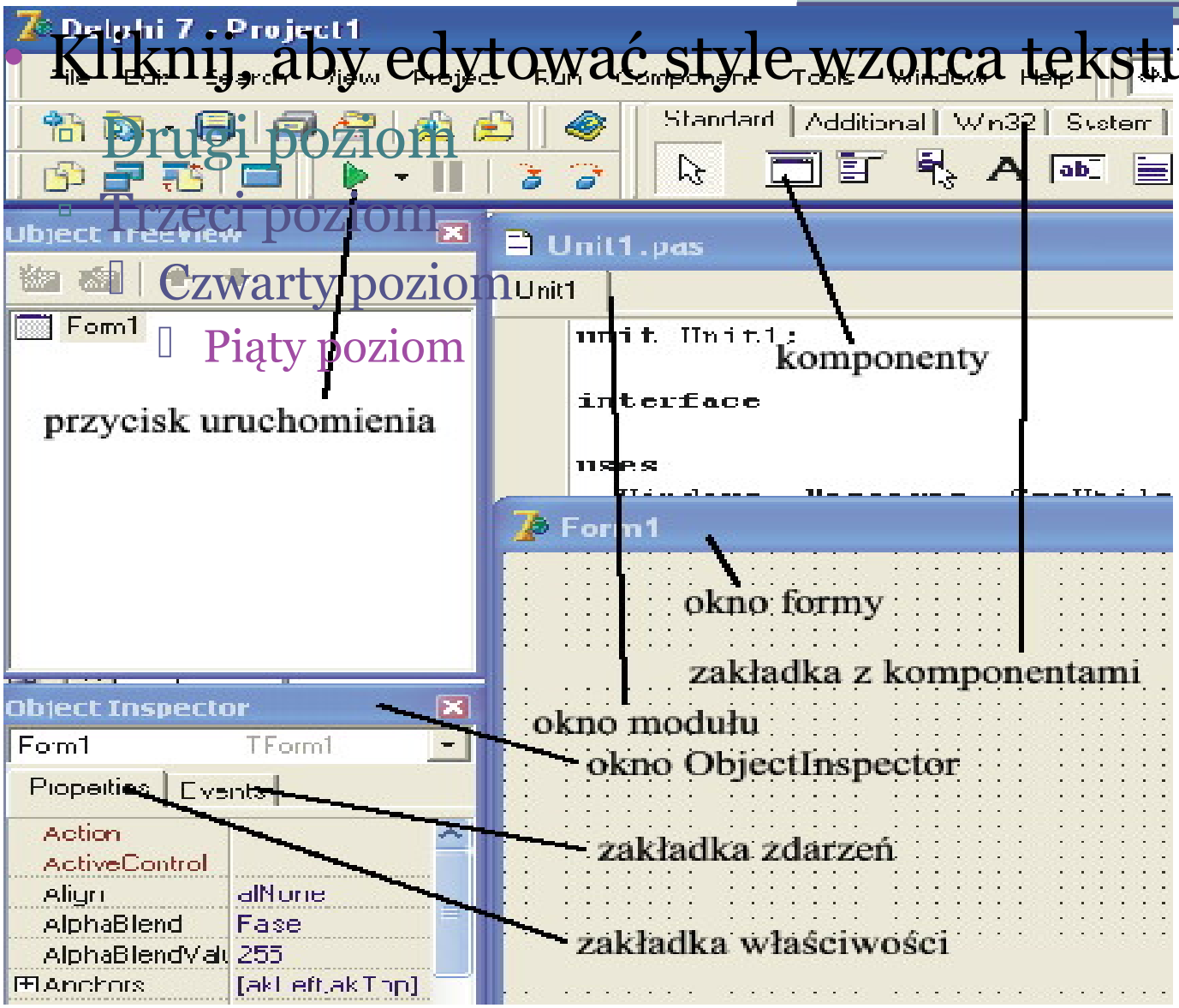


Delphi 7

- Dzisiaj najbardziej popularna wersja Delphi. Dodano nowe komponenty pozwalające na używanie stylu Microsoft Windows Xp. Była także ostatnią wersją ze starym układem okien.



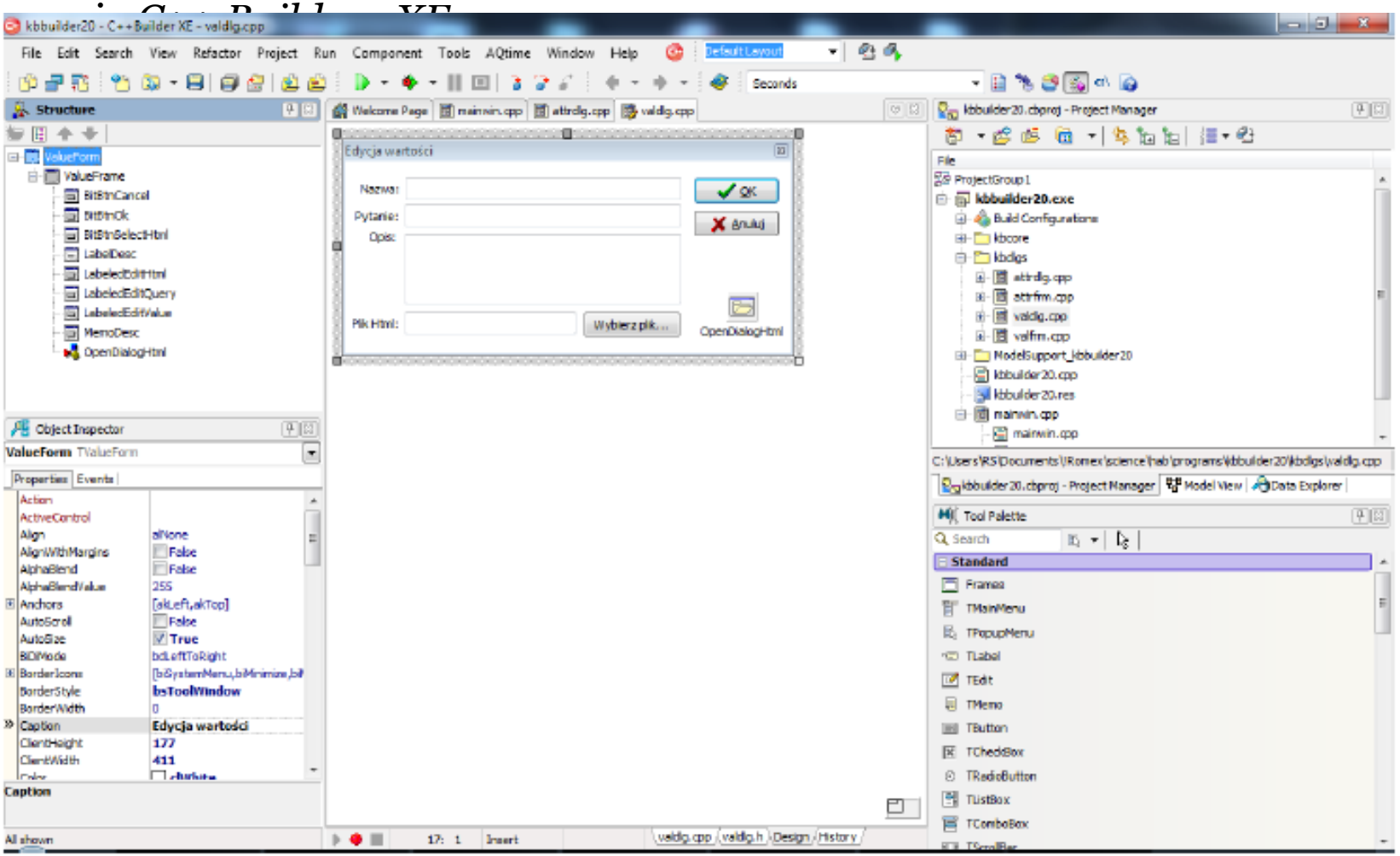
- Kliknij, aby edytować style wzorca tekstu
  - Drugi poziom
    - Trzeci poziom
      - Czwarty poziom
        - Piąty poziom



Rysunek 1.1. Środowisko Delphi 7.0 Studio

RAD Studio

- W 2010 dostępne jest *RAD Studio, integrujące Delphi i C++ Buildera XE.*
- Na początku 2011 pojawia się *Delphi i C++ Builder Starter — uproszczona*



Jądro RAD Studio: VCL — Visual Component Library

Visual Component Library — wersja podstawowa

- Oferuje dużą liczbę wizualnych i niewizualnych komponentów pozwalajacych na budowanie natywnych interfejsów użytkownika dla środowiska Windows.
- Zawiera zbiór standardowych elementów sterujących interfejsu (przyciski, menu, listy itd.) jak również zestaw komponentów rozszerzonych nie występujących bezpośrednio w zestawie elementów sterujących.
- Oferuje obsługę *akcji, pozwalających centralizować przetwarzanie w aplikacji*.
- Elementy sterujące wrażliwe na dane — przeznaczone do realizacji aplikacji
- wykorzystujących bazy danych.
- W ramach biblioteki VCL dostępna jest duża liczba klas niewizualnych, służących m.in. do zarządzania kolekcjami obiektów.

C++ Builder 6

C++ Builder 6 jest środowiskiem programistycznym umożliwiającym szybkie tworzenie wydajnych aplikacji konsolowych i okienkowych dla systemów Microsoft Windows 98, 2000, 2003, XP, Vista.

Zaimplementowanym językiem jest C++.

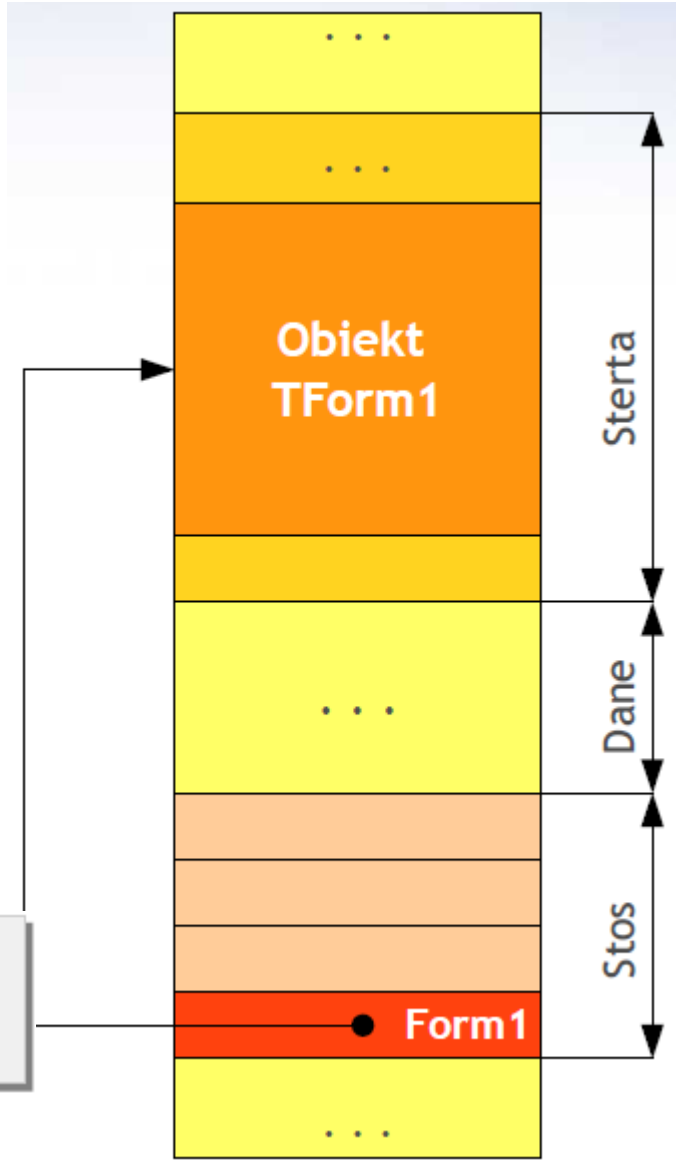
Produkt zawiera kilkadziesiąt gotowych do wykorzystania komponentów znacznie ułatwiających pisanie programów okienkowych. Ponadto w C++ Builder Personal można tworzyć i instalować własne komponenty.

C++ Builder 6

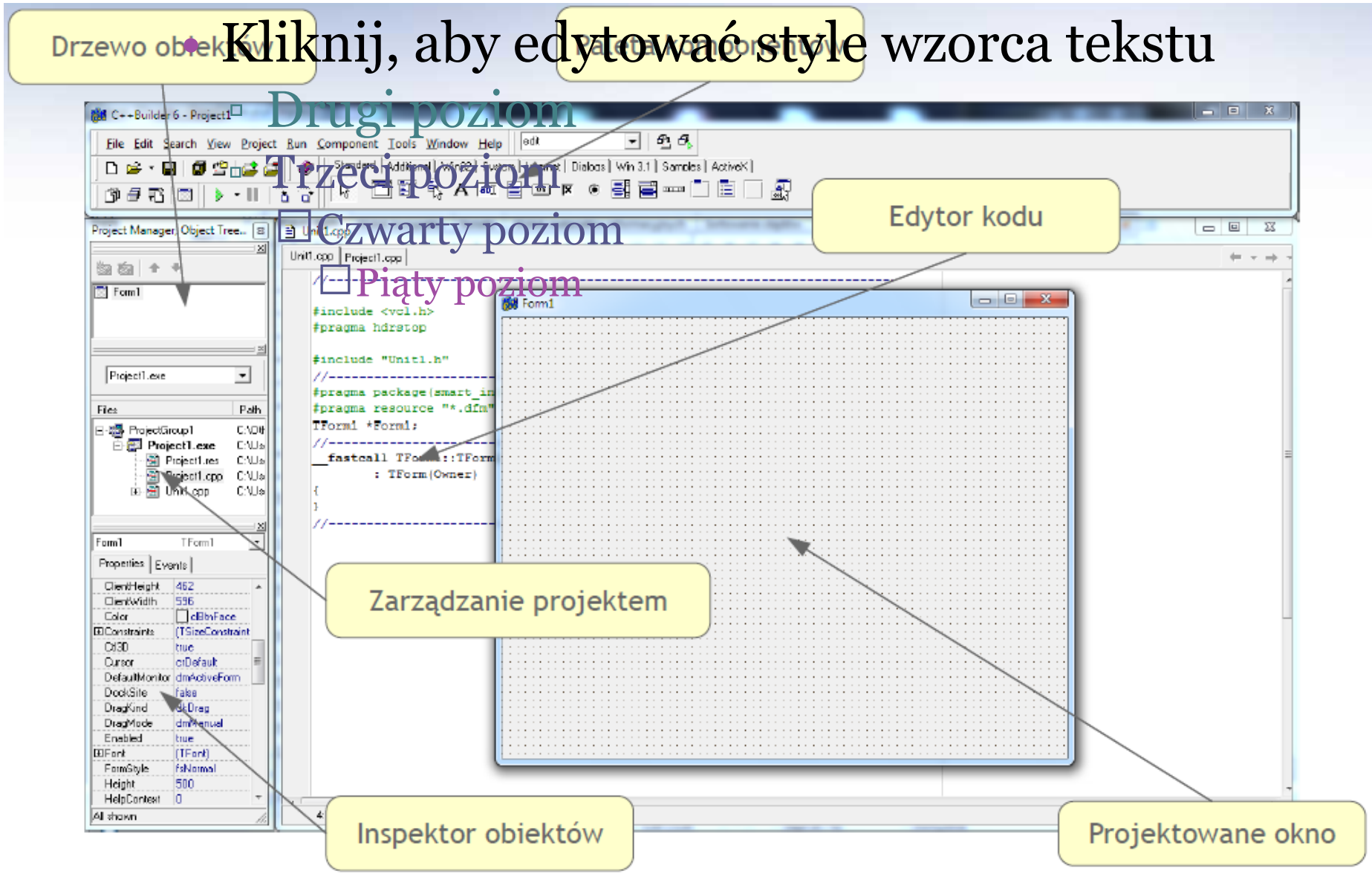
W systemie C++ Builder komponenty „obsługiwane” przez środowisko przyjmują postać obiektów dynamicznych, lokowanych na stercie programu i dostępnych za pośrednictwem wskaźników.

Zwykle to kod generowany przez środowisko jest odpowiedzialny za tworzenie i likwidowanie obiektów takich, programista w typowych sytuacjach nie musi się tym zajmować.

```
. . .
Application->CreateForm(__classid(TForm1), &Form1);
. . .
```



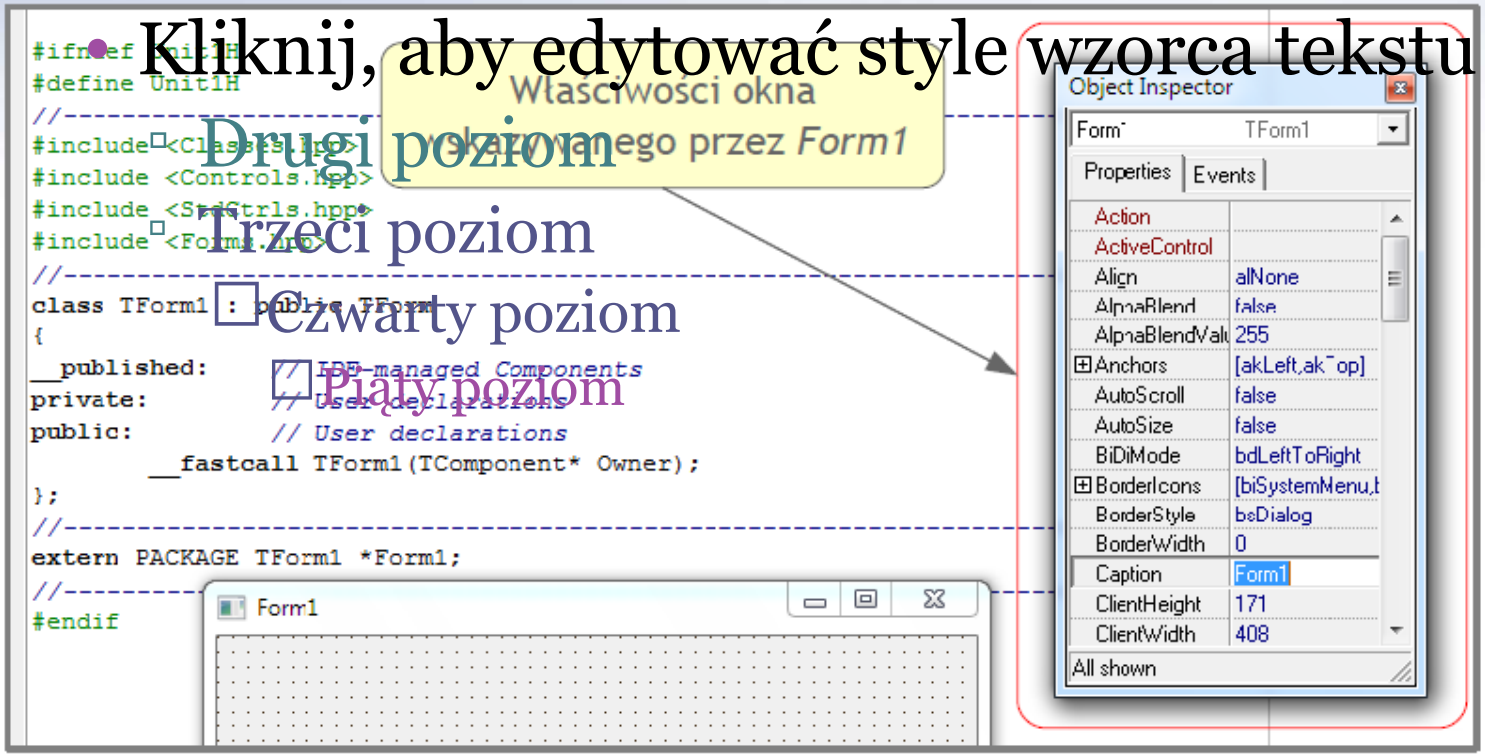
C++ Builder 6


Drzewo obiektów
Paleta komponentów
Kliknij, aby edytować style wzorca tekstu
Drugi poziom
Trzeci poziom
Czwarty poziom
Piąty poziom
Edytor kodu
Zarządzanie projektem
Inspektor obiektów
Projektowane okno

C++ Builder 6

Inspektor obiektów

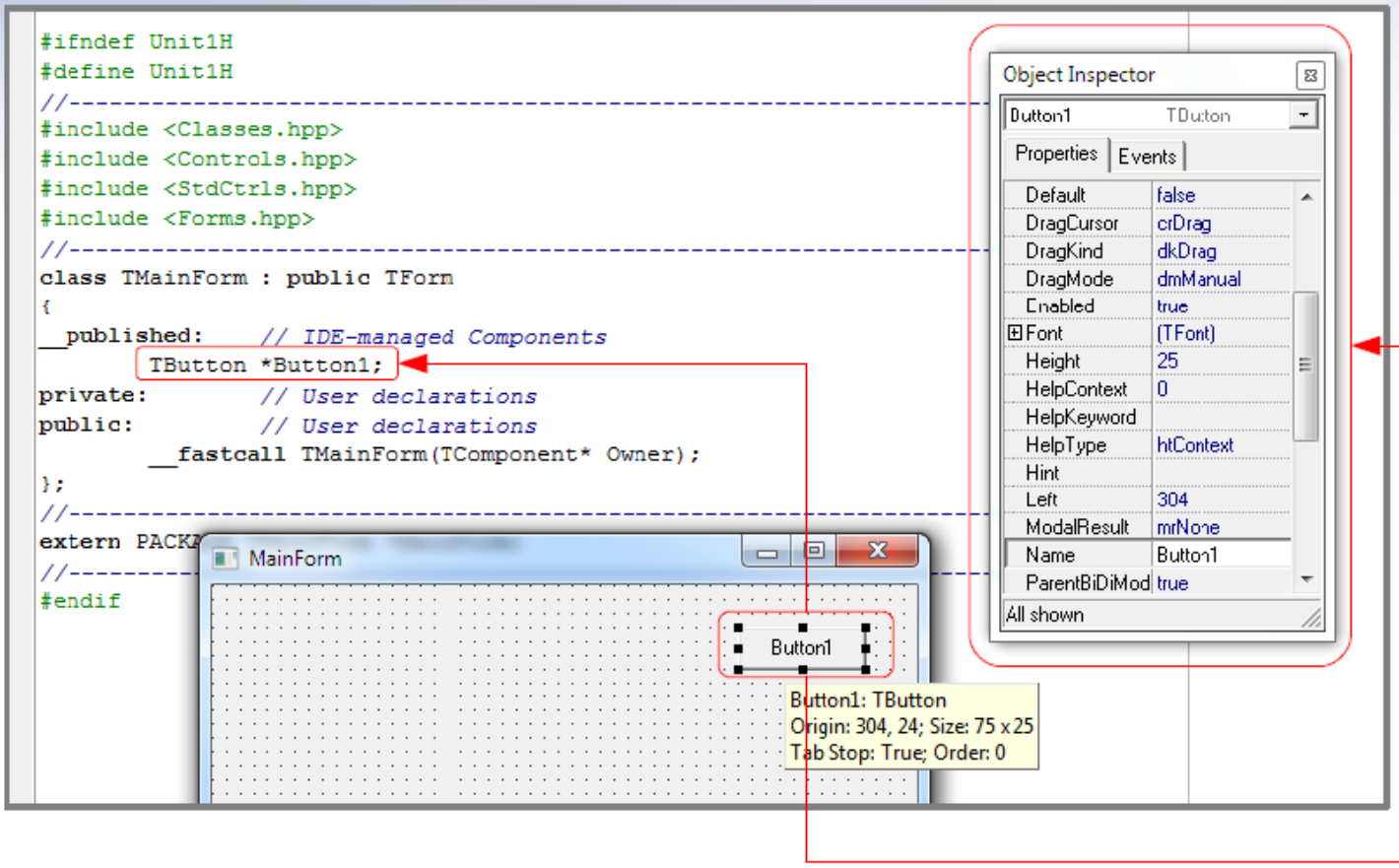
- Kliknij, aby edytować style wzorca tekstu
  - Drugi poziom
    - Trzeci poziom
      - Czwarty poziom
        - Piąty poziom



Inspektor obiektów to podstawowe narzędzie pozwalające na sterowanie właściwościami oraz zdarzeniami komponentów. Tego typu element występuje we większości środowisk typu RAD.

C++ Builder 6

Komponenty wstawiane do okna a ich reprezentacja w kodzie



Komponenty wstawiane na formę reprezentowane są w kodzie za pośrednictwem wskaźników, tworzeniem i usuwaniem obiektów zajmuje się kod biblioteki.

Visual C++

Podstawowe narzędzia i biblioteki C++

- Możliwość szybkiego tworzenia rozbudowanych aplikacji przy zastosowaniu profesjonalnego kompilatora języka C++.

- Pisanie, kompilowanie i debugowanie kodu C++ zgodnego ze standardami ISO.

-Tworzenie programów dla platformy .NET Framework z wykorzystaniem nowej składni C++/CLI.

Łatwiejsze programowanie

- Łatwe tworzenie nowych projektów przy wykorzystaniu istniejących źródeł.

- Filtrowanie i wyszukiwanie klas oraz identyfikatorów przy użyciu ulepszonego widoku Class View.

- Możliwość analizy kodu źródłowego podczas przeglądania go w czasie rzeczywistym.

Pełny dostęp do platformy .NET Framework

-Możliwość tworzenia aplikacji i bibliotek Windows przy użyciu .NET Framework.

- Możliwość wykorzystania funkcji platformy .NET Framework, takich jak moduł odzyskiwania pamięci i programowanie generyczne.

- Możliwość stosowania w .NET Framework szablonów C++ i finalizacji deterministycznej.

WYKONAŁ:

PIOTR MAMALA