Vysoké učení technické v Brně

Fakulta elektrotechniky a komunikačních technologií

http://www.utee.feec.vutbr.cz/~fialap/vyuka/cpl.html

Ústav teoretické a experimentální elektrotechniky UTEE

Kolejní 2906/4
612 00 Brno

Ovládání Builderu C++ pro kurz BSCP

vytvořila : Ing. Eva Kroutilová, Ph.D.

kontakt : E410, kroutila@feec.vutbr.cz

Borland Builder C++

seznámení s prostředím

Borland C++ Builder je nástrojem pro vývoj programů, určených pro operační systémy Microsoft Windows.

Builder pracuje se zdrojovými kódy, sestavenými v programovacím jazyce C++. Builder využívá principů tzv. **vizuálního programování** (*drag &drop design*).

Psaní aplikace probíhá tak, že programátor sestavuje myší uživatelské rozhraní svého programu, a Builder mu generuje odpovídající zdrojový kód, napsaný v jazyce C++.

Pokud programátor zasáhne do zdrojového kódu, změna se samočinně promítne do vizuálně sestaveného prvku a naopak (tzv. *two-way tool*).

Integrované prostředí nástroje Builderu sestává ze sedmi základních částí :

- **Paleta komponentů** (*component palette*)
- **Formulář** (*form*)
- **Inspektor objektů** *(Object Inspector)*
- **Editor kódu** *(Code Editor)*
- **Urychlující panel** *(Speedbar)*
- **Menu**
- **Ladič** *(Debugger)*

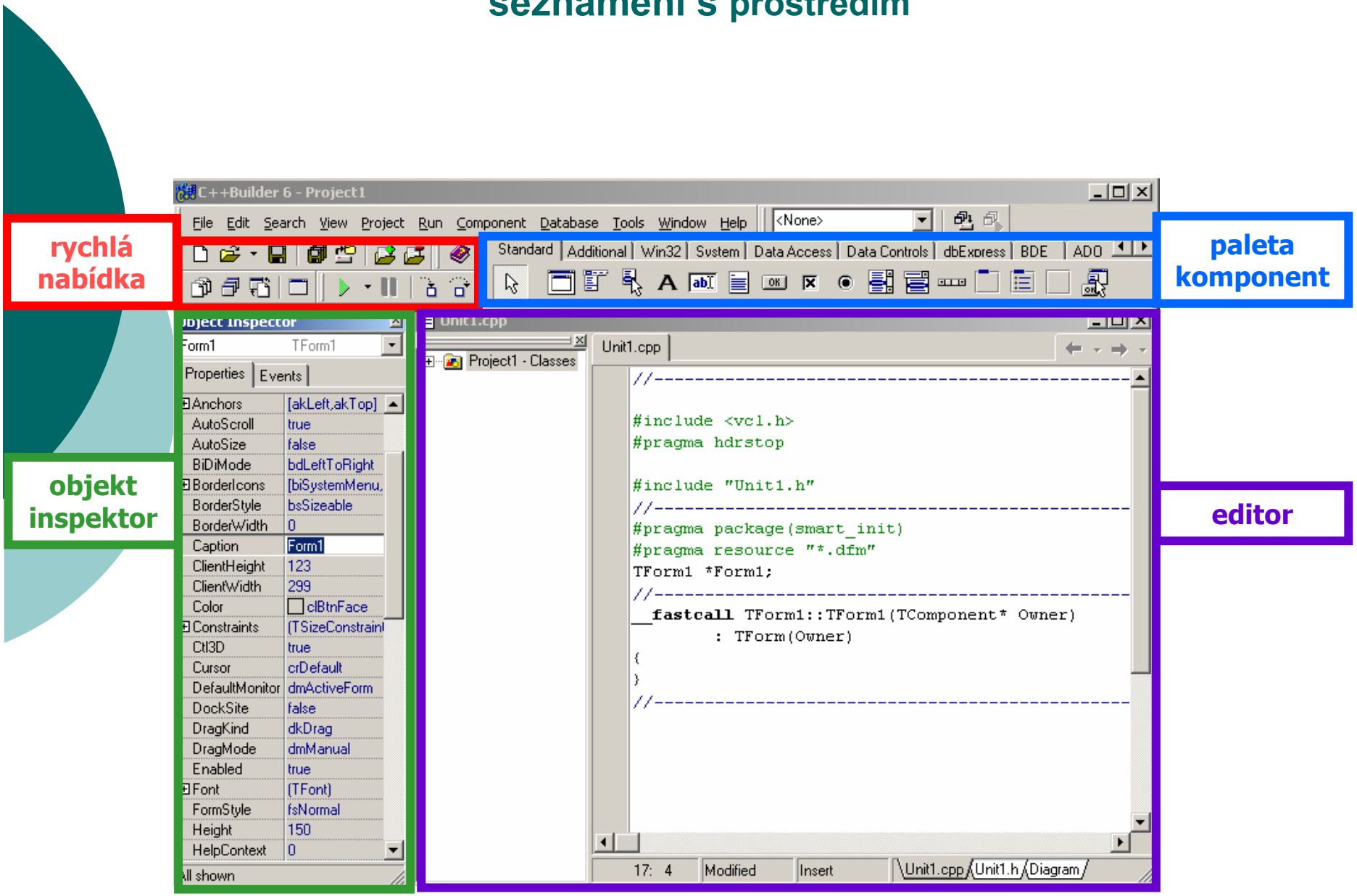
Borland Builder C++

seznámení s prostředím

Nespouštět Borland Builder C++ ze síťového disku (práci ukládat na pevný, pak si překopírovat zálohu)

Borland Builder C++

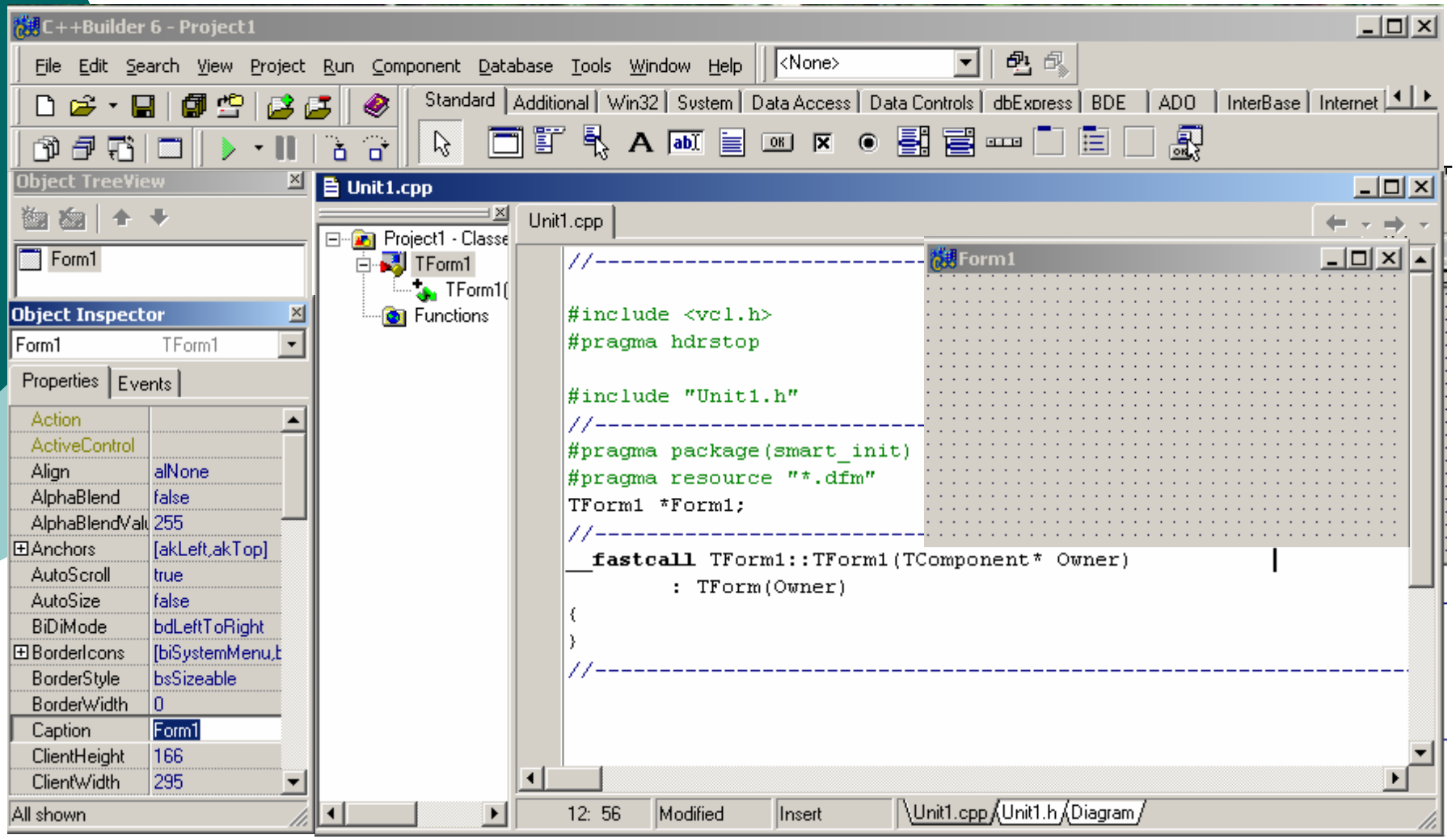
seznámení s prostředím



Borland Builder C++

seznámení s prostředím

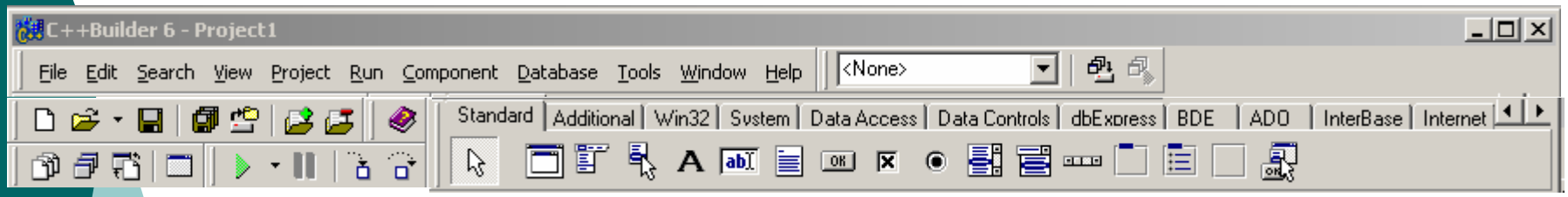
úvodní okno



Borland Builder C++

seznámení s prostředím

Hlavní menu



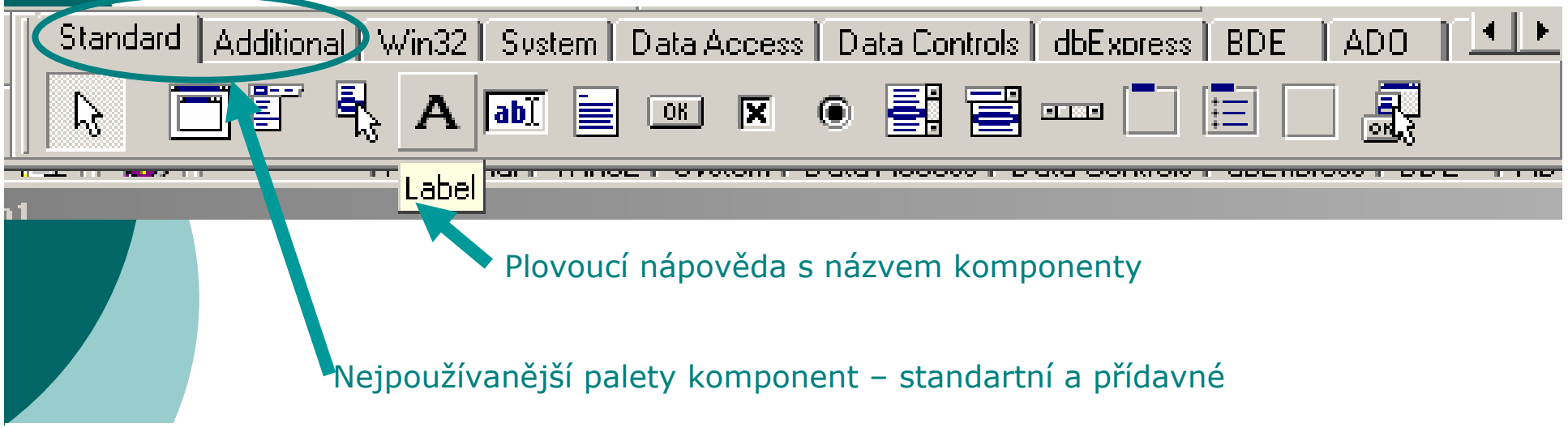
Paleta nástrojů

Palety komponent

Borland Builder C++

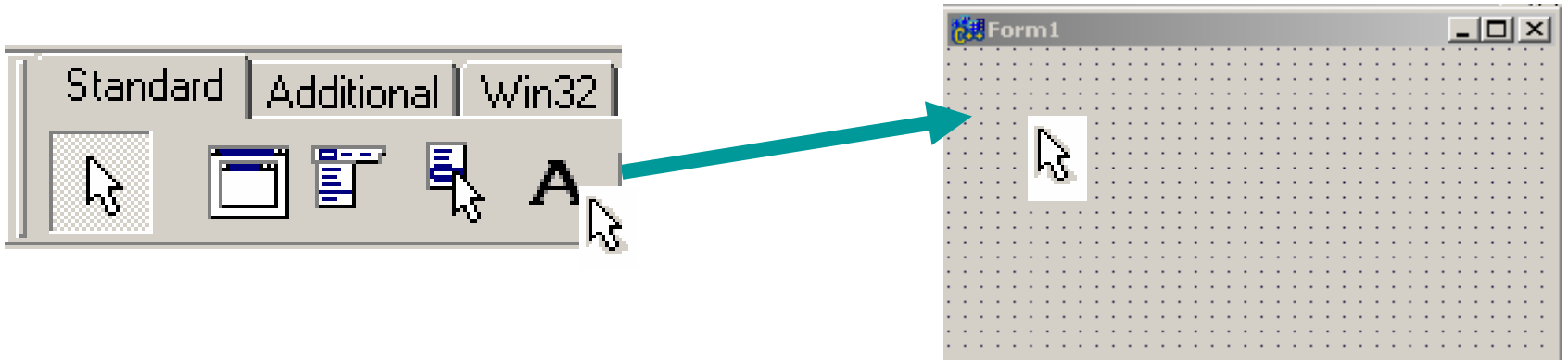
seznámení s prostředím

Palety komponent



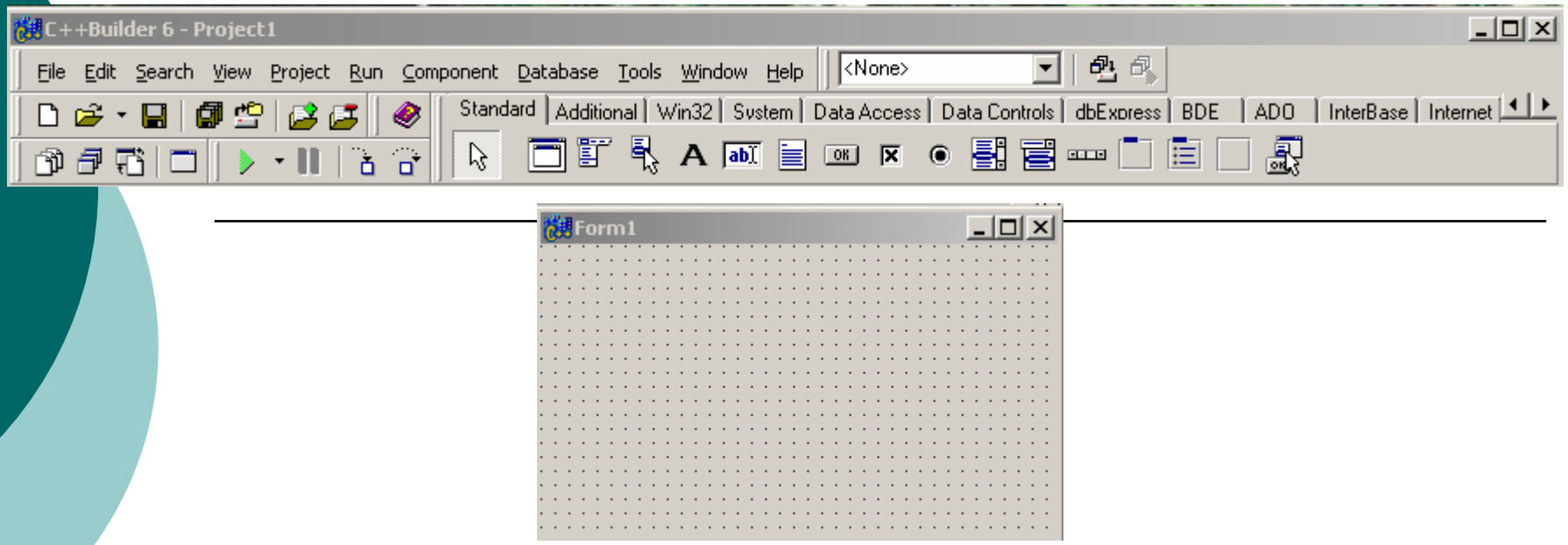
Výběr komponenty

– kliknutí levého tlačítka myši na požadovanou komponentu (zvýrazní se) a pak kliknutí levého tlačítka myši na libovolnou pozici do formuláře



Borland Builder C++

seznámení s prostředím



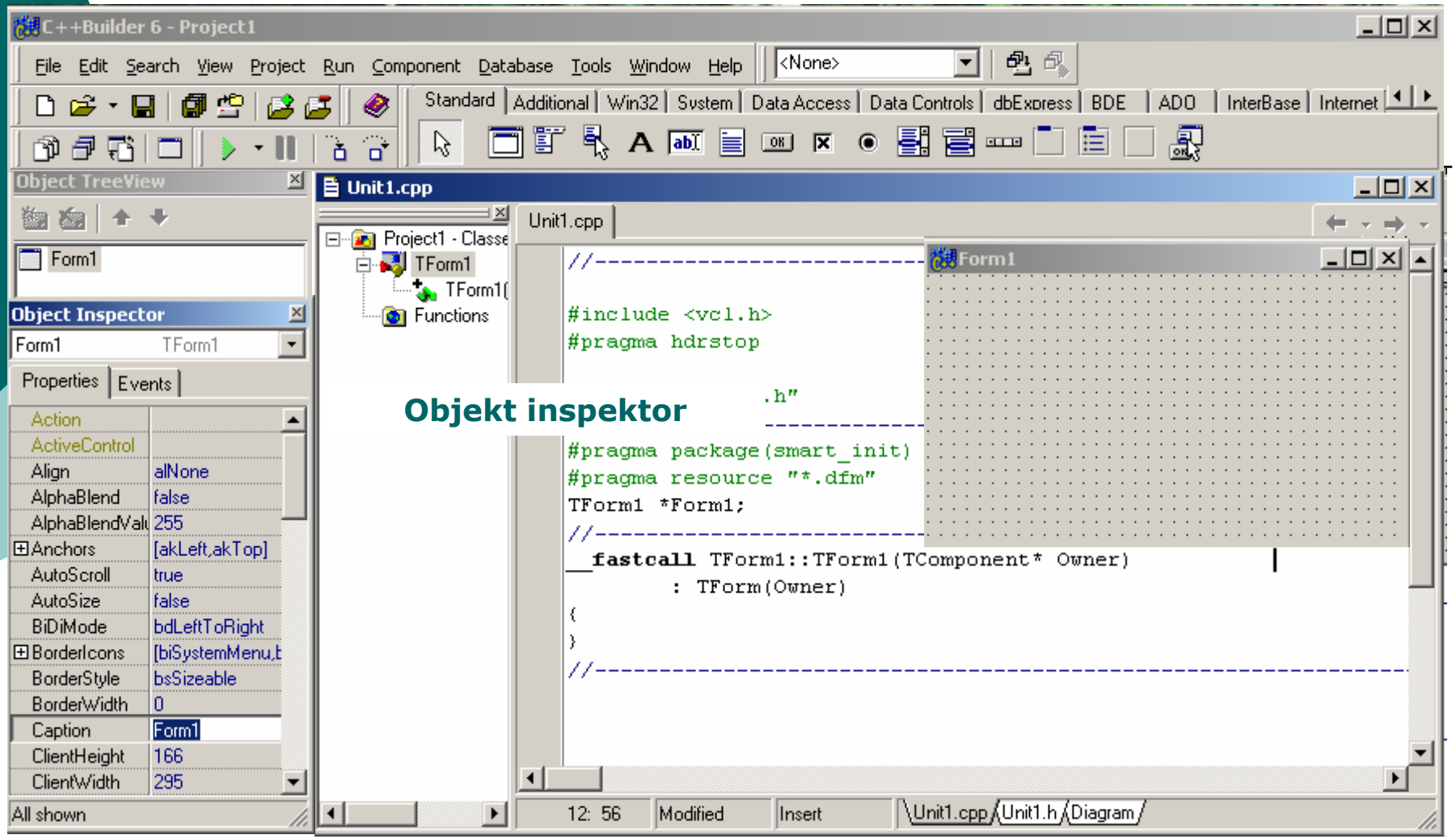
Paleta komponentů (*component palette*) je tvořena sadou záložek, na kterých jsou umístěny komponenty, z nichž lze vizuálně sestavovat okno programu (**formulář**, *form*). Volná místa na záložkách jsou připravena pro umístění vlastních uživatelských komponentů, vytvořených programátorem.

Formulář (*form*) je základní okno vytvářené aplikace s tečkovaným rastrem, do něhož myší umisťujeme jednotlivé komponenty. Jednotlivé parametry (proměnné) formuláře a vkládaných komponentů můžeme zadávat buď pomocí myši (umístění, rozměr), a jednak je můžeme určovat prostřednictvím **inspektoru objektů** (viz dále). Všechny zadané hodnoty proměnných se samočinně promítnou do zdrojového kódu vyvíjené aplikace.

Borland Builder C++

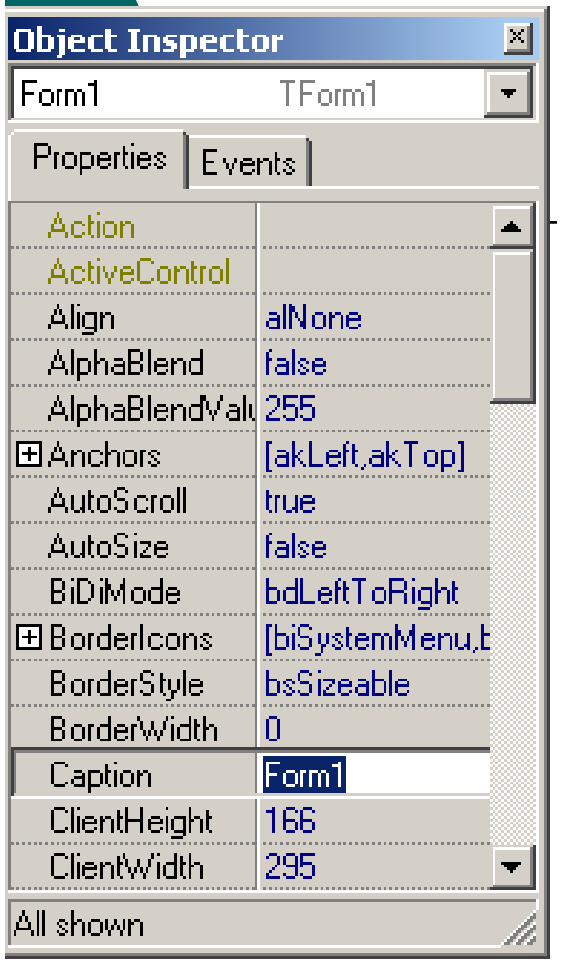
seznámení s prostředím

úvodní okno



Borland Builder C++

seznámení s prostředím



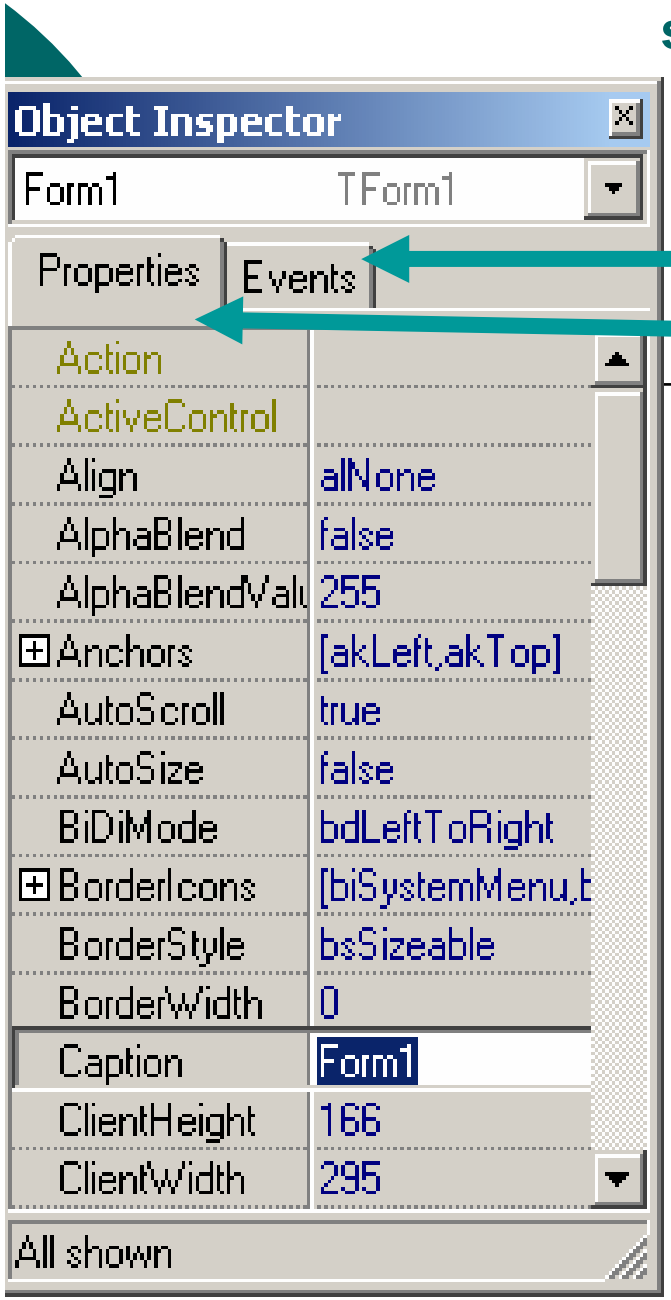
Inspektor objektů (*Object Inspector*) sestává ze dvou záložek – ze záložky s názvem *Vlastnosti* (*Properties*) a ze záložky s názvem *Události* (*Events*).

Properties obsahuje seznam všech parametrů toho komponentu, který je zaostřen (*focused*, programátor na něm kliknul myší). Nastavíme-li např. šířku a výšku komponentu myší, numerické vyjádření nastavených rozměrů komponentu se automaticky objeví v inspektoru objektů vedle proměnných Height a Width. Postupovat lze samozřejmě i obráceně.

Záložka ***Events*** obsahuje seznam událostí objektu, který je právě zaostřen ve formuláři. Událostí rozumíme vše, co může nastat při zaostření daného komponentu. Např. při kliknutí na tlačítko je generována událost OnClick. Má-li kliknutí na tlačítko spustit vykonávání určitého kódu (má být volána určitá funkce), vepíšeme do editačního řádku vedle události jméno funkce a stiskneme klávesu *Enter*. Builder automaticky vygeneruje hlavičku funkce, a nám stačí její tělo (prostor mezi složenými závorkami) vyplnit zdrojovým kódem.

Borland Builder C++

seznámení s prostředím



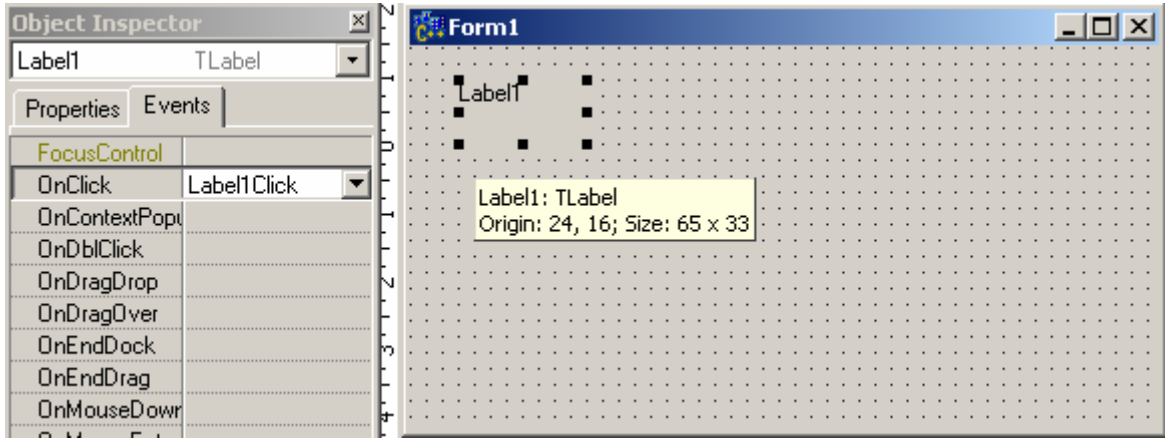
Object inspektor

Nastavení událostí (např. po kliknutí myši)

Nastavení vlastností (např. barva a font písma)

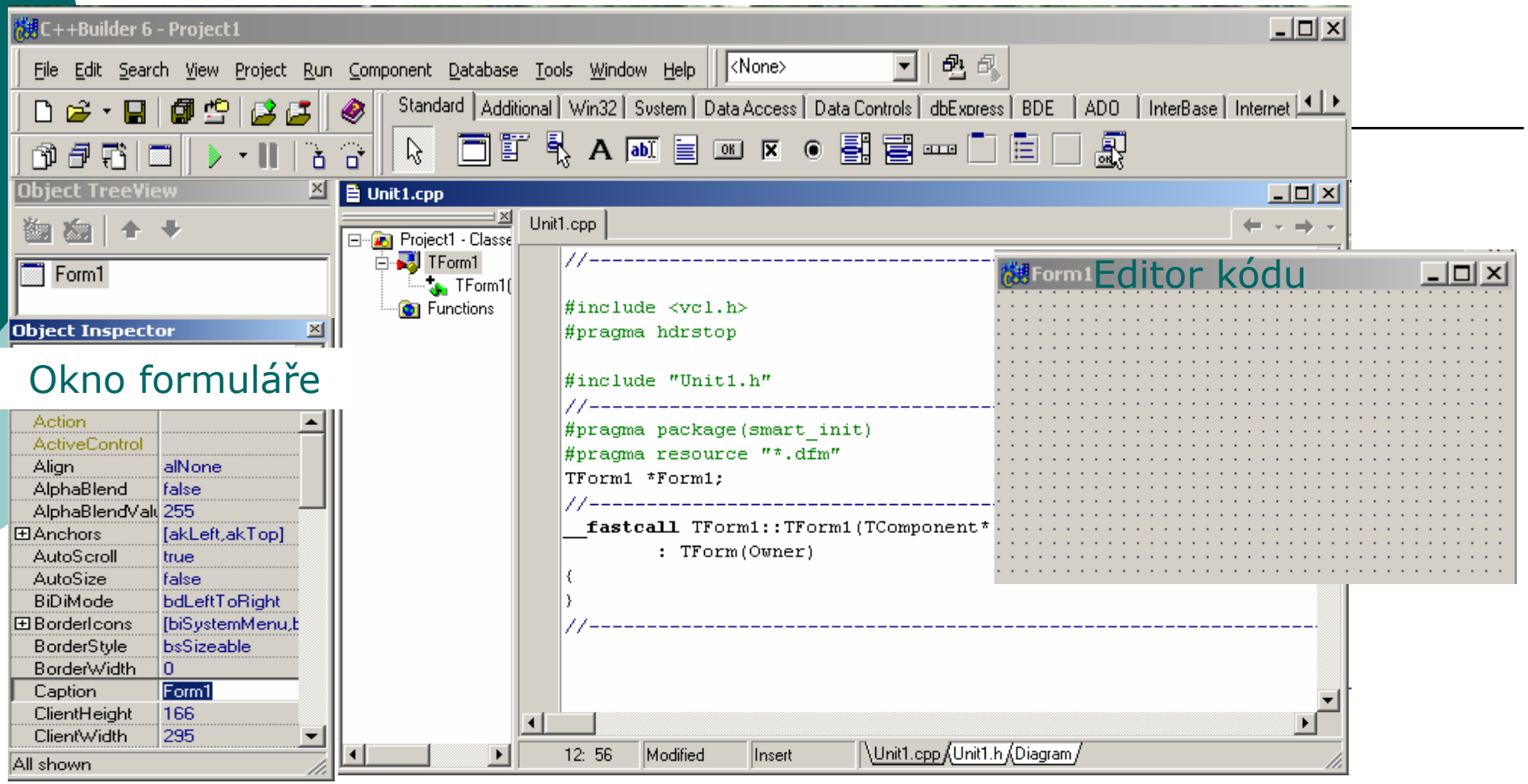
Nastavení

– kliknutí levého tlačítka myši do požadovaného řádku v object inspektoru, pak dvojklik na komponentu – přenese se název komponenty do řádku



Borland Builder C++

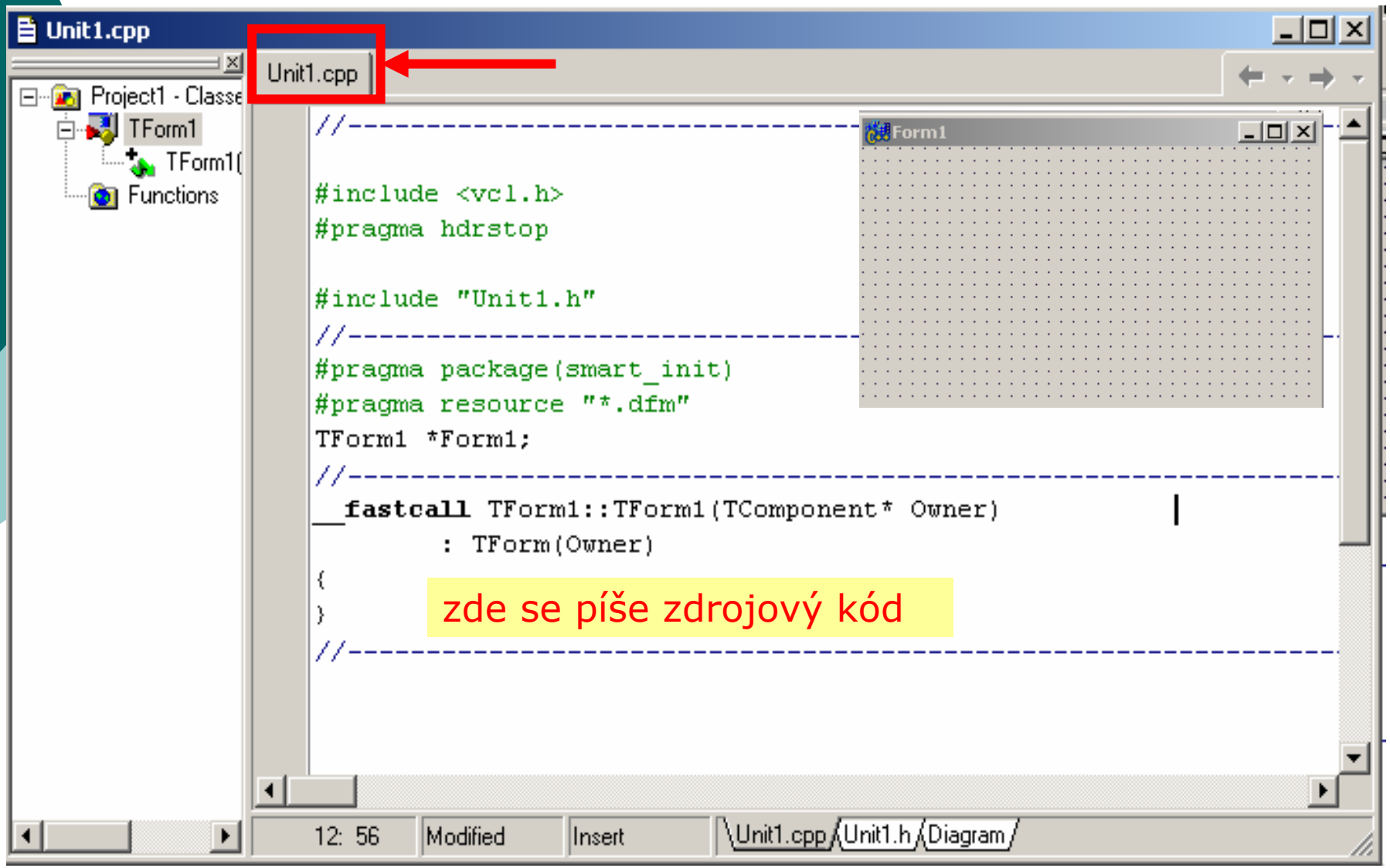
seznámení s prostředím



Borland Builder C++

seznámení s prostředím

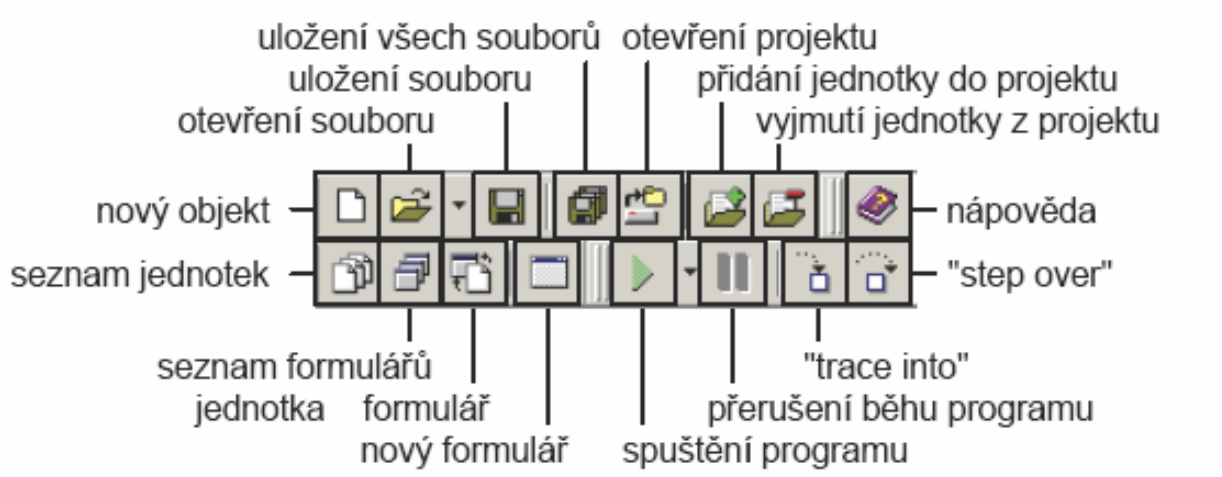
Editor kódu



Borland Builder C++

seznámení s prostředím

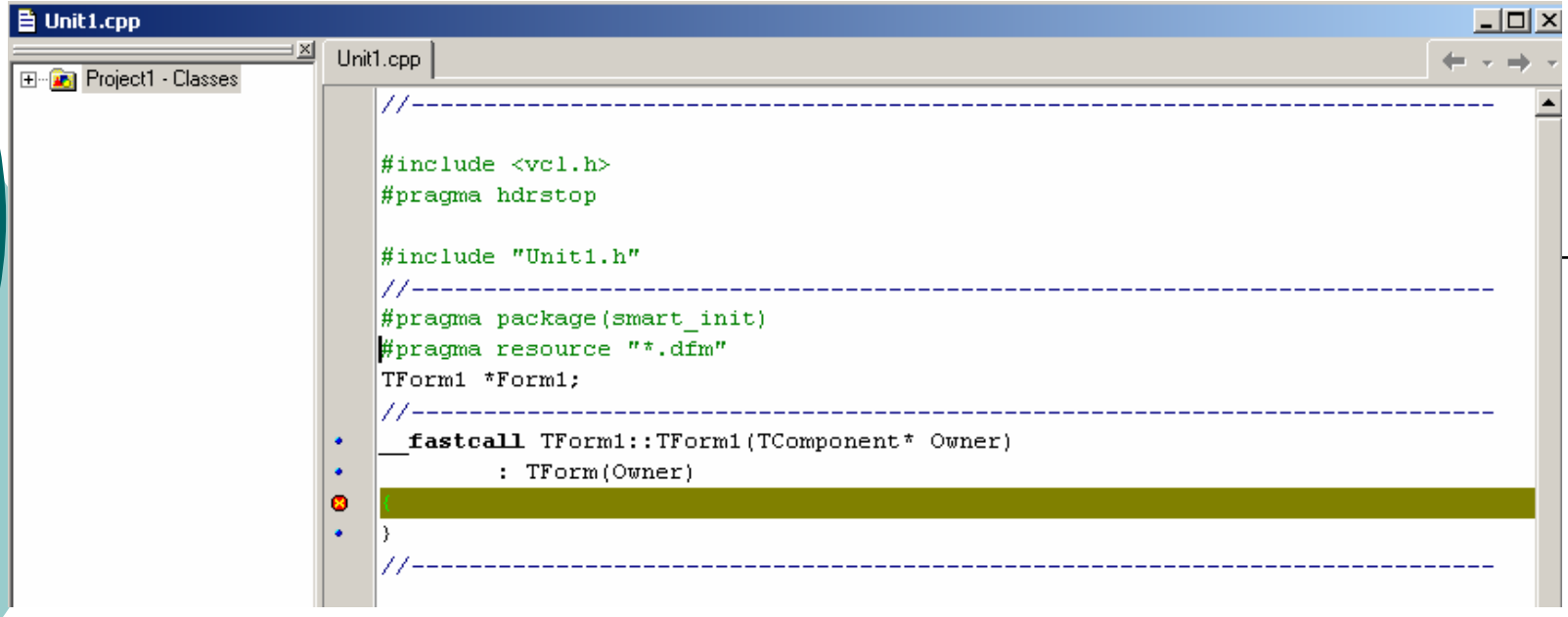
Editor kódu (*Code Editor*) slouží k vytváření zdrojového kódu, napsaného v jazyce C++. Část kódu je generována samočinně jako reakce na vizuální sestavování formuláře, zbytek musí programátor dopsat sám. Každý formulář aplikace je uložen v samostatné programové jednotce (*unit*). Zdrojový text jednotky má příponu ***.cpp**.



Urychlující panel (*Speedbar*) je sestaven z tlačítek, soužících k vyvolání nejčastějších akcí. Detail rychlého panelu a popis funkce jednotlivých tlačítek je zobrazen na obr. Anglický popis tlačítka je zobrazován v „bublinové" nápovědě. Tlačítka *Trace Into* a *Step Over* slouží ke krokování programu (programátor manuálně vykonává instrukci za instrukcí). Při *Trace Into* se vnoříme se do funkce a řádek po řádku vykonáváme jednotlivé její příkazy. Při Step Over se do funkce nevnoříme a vykonáme ji jako jeden jediný příkaz.

Borland Builder C++

seznámení s prostředím



Menu. Vzhledem k obrovskému množství položek, jež jsou do menu zahrnuty, nemá si smysl brát jednu položku po druhé a vysvětlovat jejich význam.

Ladič (*Debugger*) není na obrázku vidět, protože je do Builderu integrován. Kliknutím na levý okraj řádku v editoru kódu vložíme na tento řádek tzv. **přerušovací bod** (*Breakpoint*). Spuštěný program se na daném řádku zastaví, takže programátor zde může kontrolovat a měnit obsah proměnných, může program krokovat nebo jej může znovu spustit, aby pokračoval ve svém chodu. Opětovným kliknutím na totéž místo přerušovací bod odstraníme.

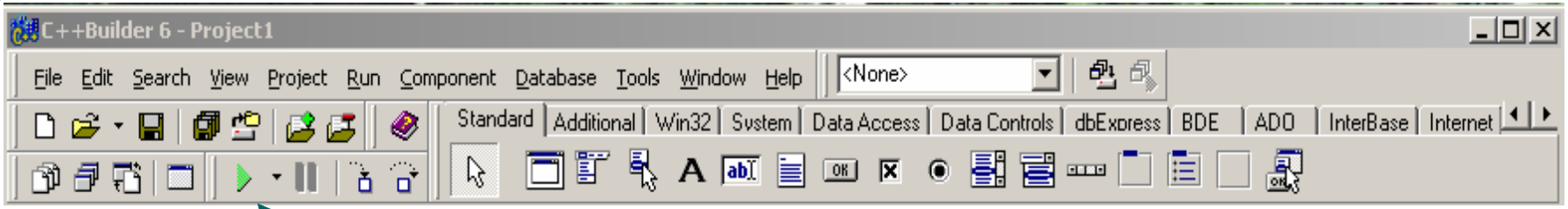
Borland Builder C++

seznámení s prostředím

Spuštění Builderu

Po spuštění obsahuje Builder prázdný formulář a prázdné editační okno, které odpovídá programové jednotce tohoto formuláře. Před zahájením práce je vhodné tuto „prázdnou" aplikaci uložit. Stisknutím čtvrté horní ikony rychlého panelu (*Save All*) otevřeme standardní dialog pro ukládání do souboru. Builder nám v řádku Název souboru nabídne standardní jméno jednotky formuláře (**unit1.cpp**). My toto jméno změníme na **add_form.cpp**, aby se nám v souborech lépe orientovalo. Po tisku tlačítka Uložit nám Builder nabídne standardní jméno projektu (**project1.bpr**), přičemž my toto jméno změníme na **addition.bpr**.

Pokud prázdný, právě uložený projekt spustíme (pátá dolní ikona, *Run*), objeví se prázdné okno se standardní ikonou a s názvem *Form1*. Ve stavové liště Windows je běžící aplikace reprezentována tlačítkem se stejnou ikonou a se jménem *jmeno* (jméno odpovídá zvolenému pojmenování projektu).

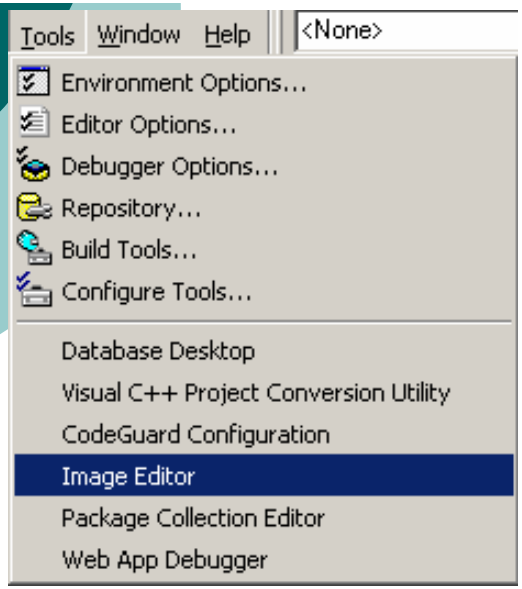


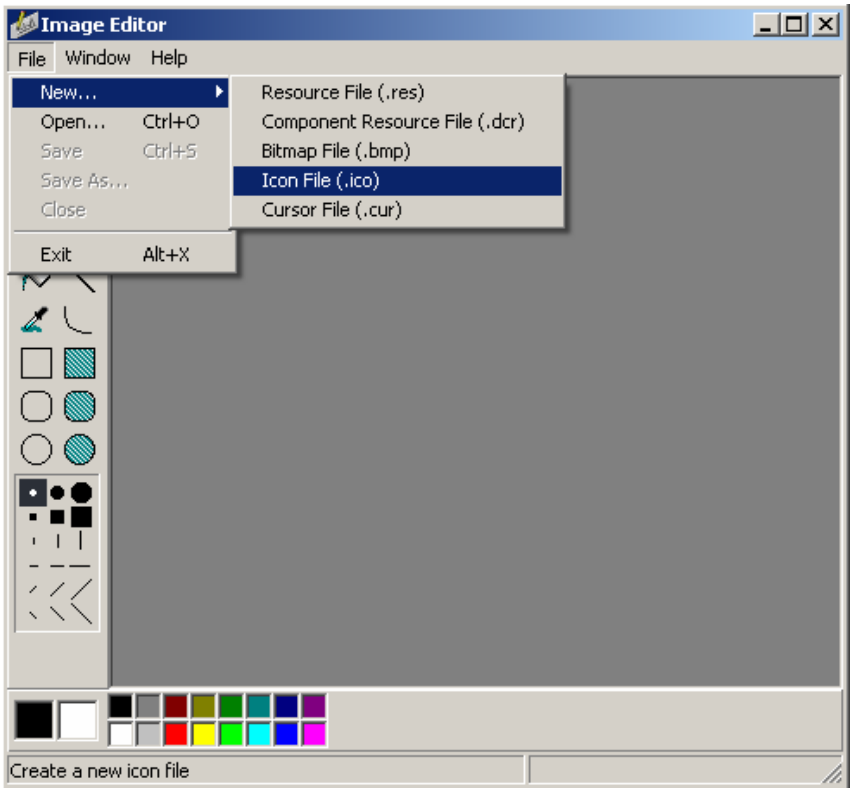
Borland Builder C++

seznámení s prostředím

Základní nastavení

Každému programu bývá většinou přiřazena vlastní ikona. Ikonu vytvoříme pomocí editoru, který v Builderu spustíme prostřednictvím položky menu *Tools* → *Image editor* (editor obrázků). Vybereme-li z menu editoru obrázků položku *File* → *New…* → *Icon file* (a potvrdíme- li standardní parametry ikony 32×32 bodů, 16 barev), spustíme jednoduchý grafický editor, v němž můžeme bod po bodu ikonu sestavit. Výběrem položky menu *File* → *Save* uložíme ikonu do adresáře k ostatním souborům našeho programu (soubor **add_icon.ico**).

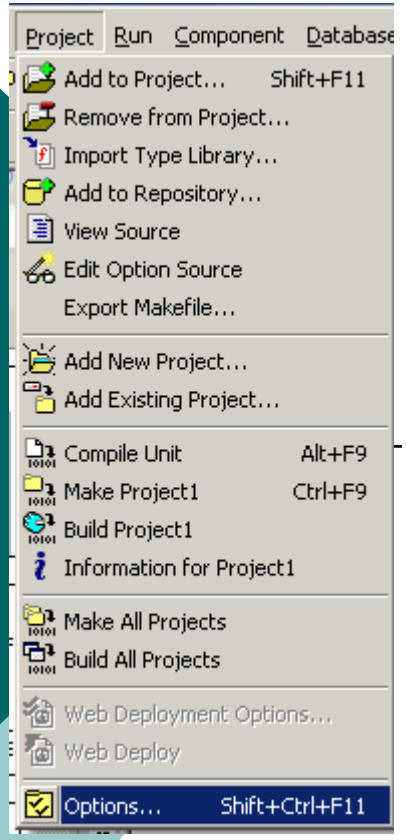


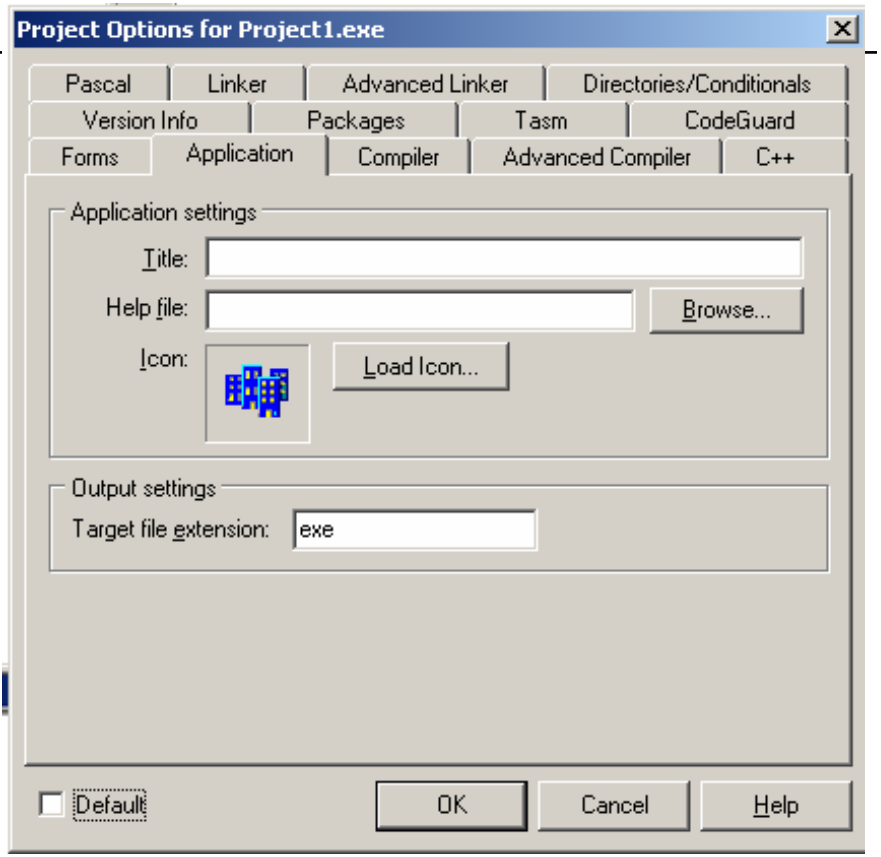


Borland Builder C++

seznámení s prostředím

Základní nastavení



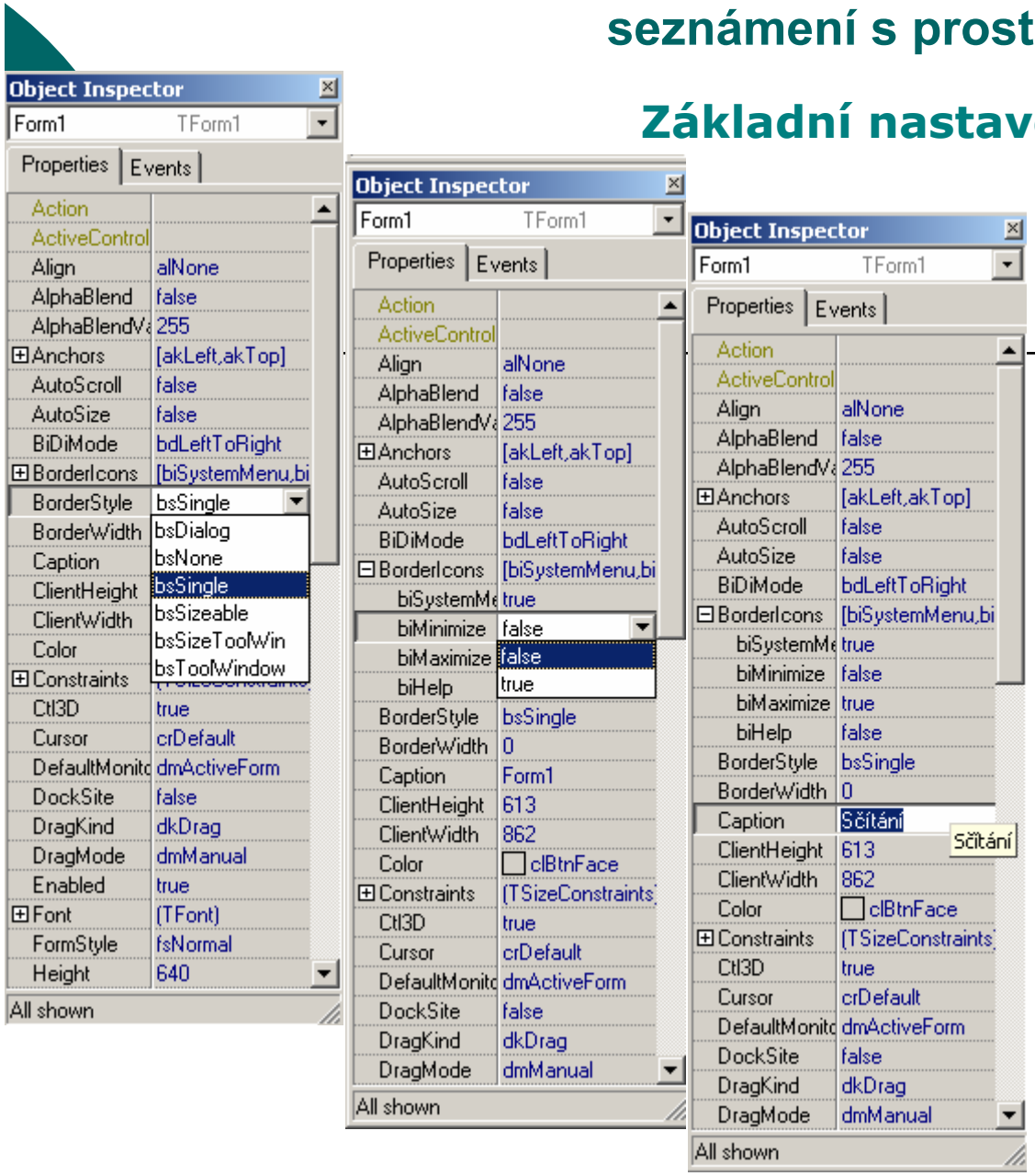


Vytvořenou ikonu přiřadíme naší aplikaci prostřednictvím položky menu Builderu *Project* → *Options*. Otevřeme tím dialog s třemi řadami záložek. Vybereme záložku *Application* a stiskem tlačítka *Load Icon* načteme námi vytvořenou ikonu. Do řádku *Title* vepíšeme řetězec *Sčítání*. Stiskem tlačítka *OK* dialog uzavřeme. Spustíme-li naši aplikaci znovu, jak okno tak tlačítko ve stavové liště Windows budou mít naši ikonu, a navíc, tlačítko v liště bude obsahovat český název *Sčítání* místo původního *Addition*.

Borland Builder C++

seznámení s prostředím

Základní nastavení



Po základním nastavení aplikace se zaměříme na nastavení parametrů formuláře:

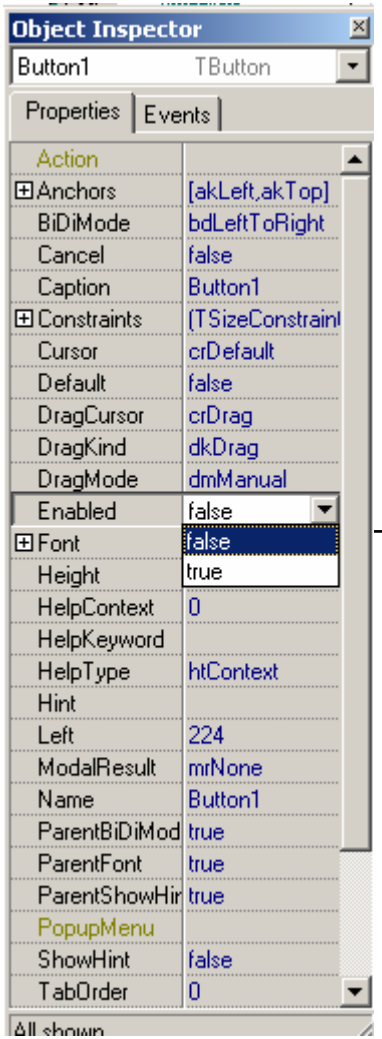
1.Pevné rozměry okna. Myší nastavíme rozměr formuláře (v inspektoru se automaticky mění obsah proměnných Height a Width). V inspektoru nastavíme ***BorderStyle*** na ***bsSingle*** (okno nepůjde roztáhnout myší taháním za okraje) a v ***BorderIcons*** nastavíme ***biMaximize*** na false (zablokujeme ikonu pro roztažení okna přes celou obrazovku).

2. Popis okna. V inspektorovi naplníme parametr Caption = Sčítání (dosud parametr obsahoval řetězec Form1). Tím jsou základní nastavení dokončena.

Borland Builder C++

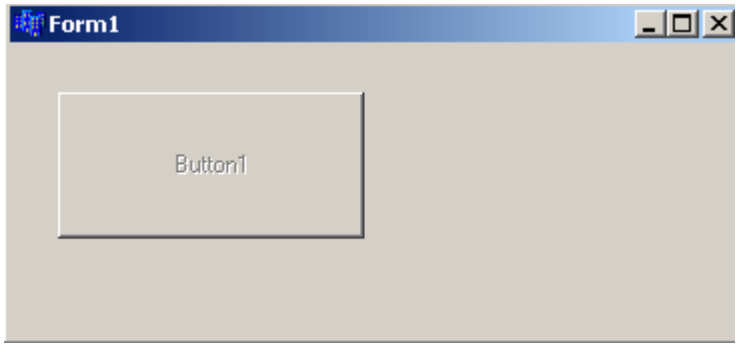
seznámení s prostředím

Základní nastavení

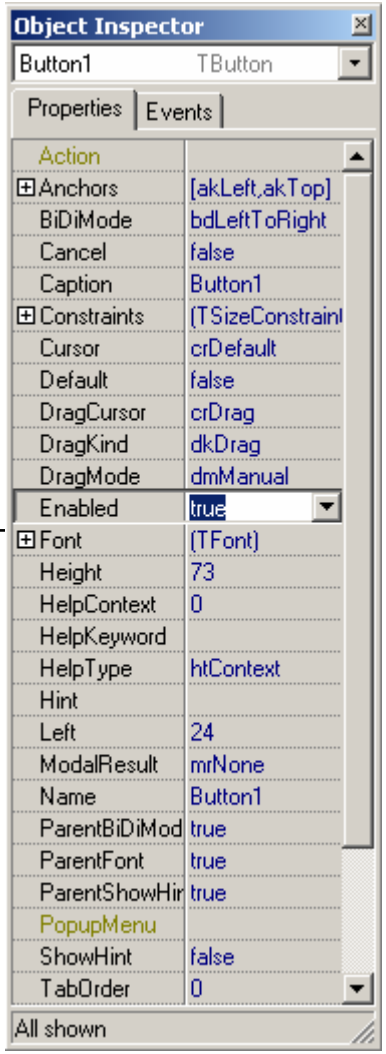


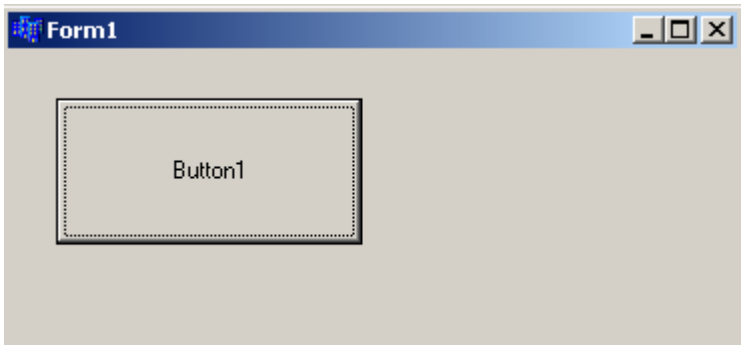
Odstavení kurzoru myši (**Enabled→false**)
po spuštění nelze na tlačítko kliknout (zšedne)

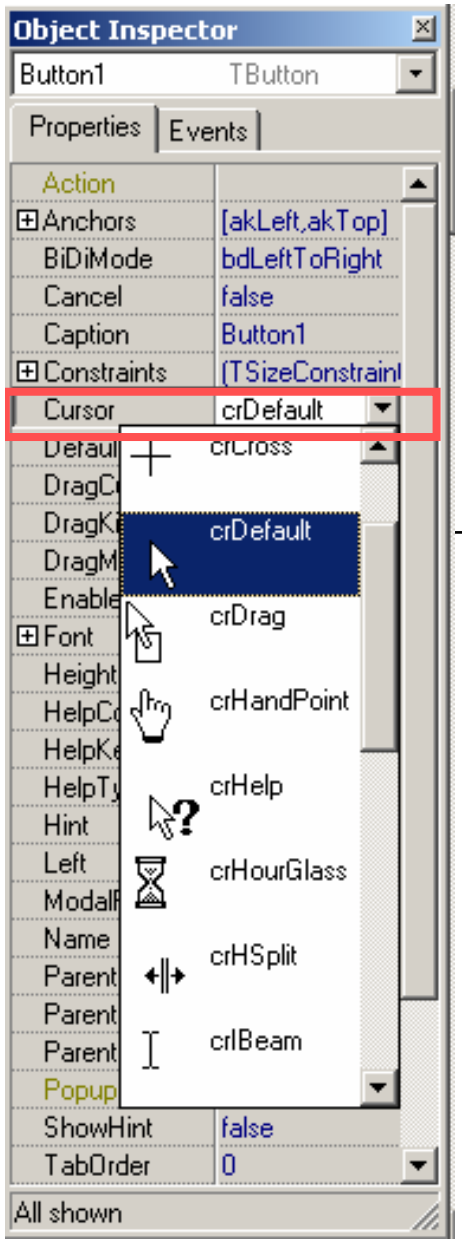
false



true







Borland Builder C++

seznámení s prostředím

Základní nastavení

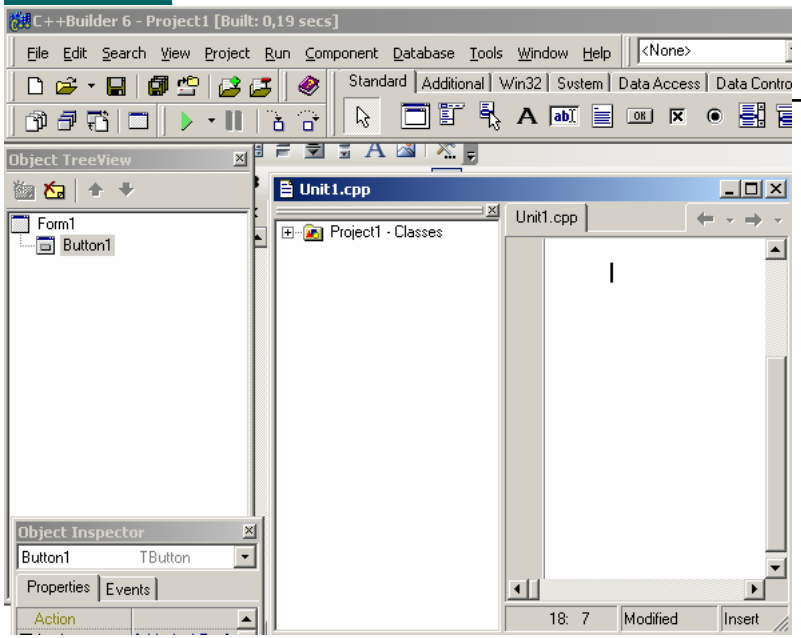
Tvar kurzoru myši (**Cursor →vyberu, tvar co se mi líbí**)

Borland Builder C++

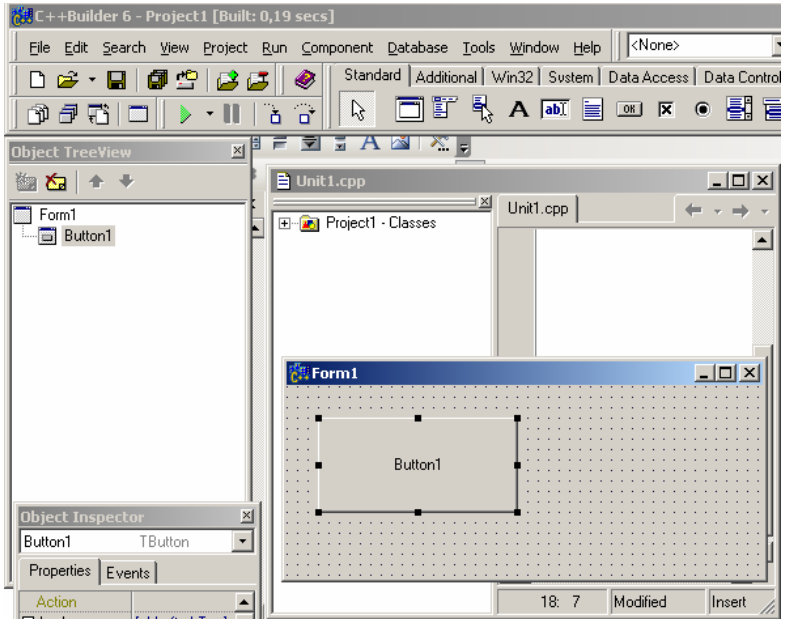
seznámení s prostředím

Základní nastavení

Vyvolání formuláře – rychlý přechod **F12**



F12→

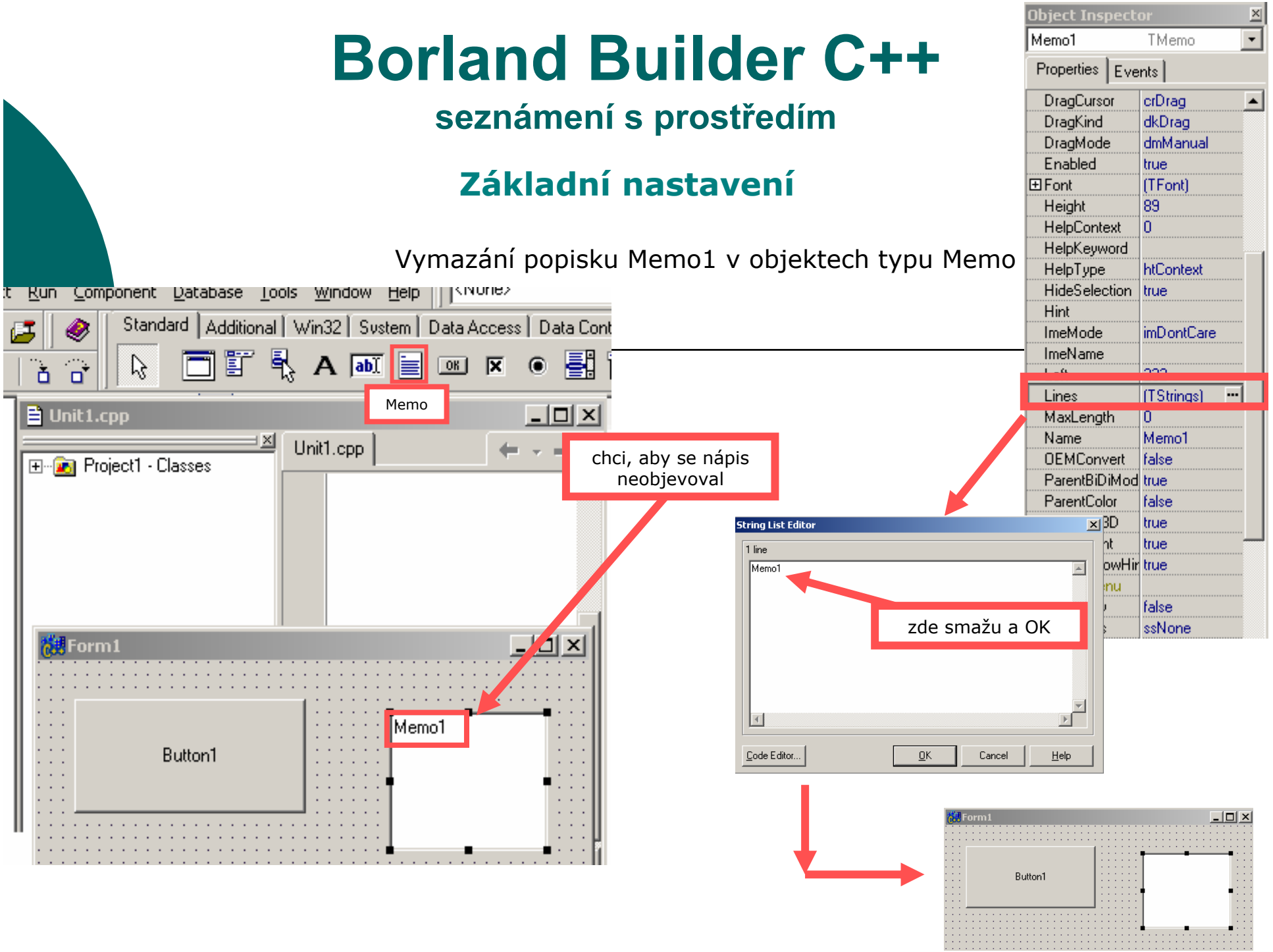


Borland Builder C++

seznámení s prostředím

Základní nastavení

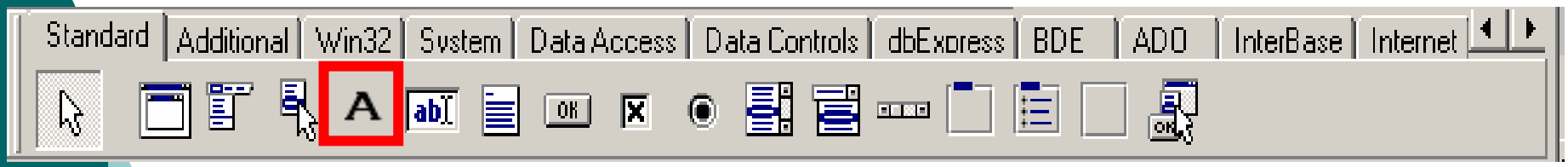
Vymazání popisku Memo1 v objektech typu Memo



Borland Builder C++

seznámení s prostředím

Sestavení okna



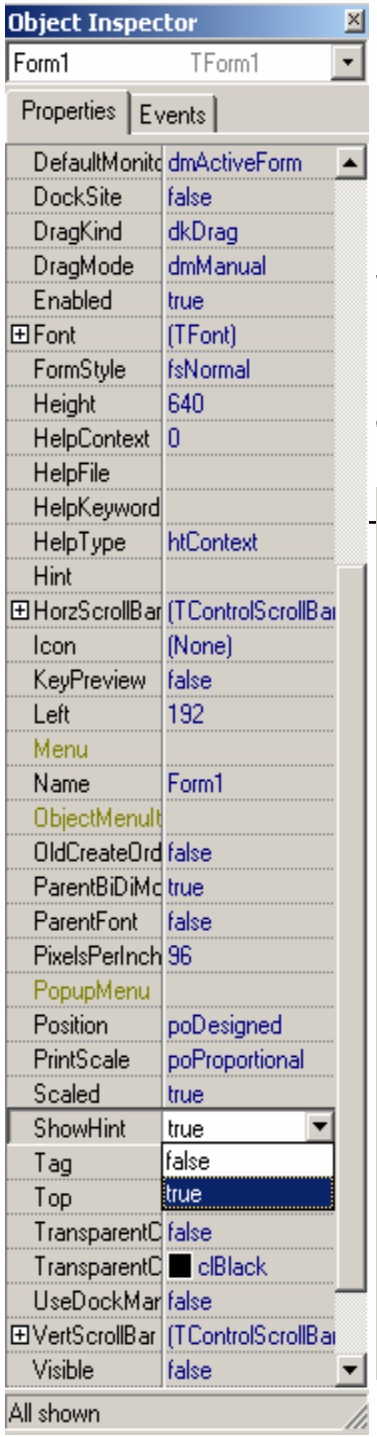
V dalším kroku postupně umístíme dovnitř formuláře komponenty z palety. Pracujeme zde v ukázce se třemi komponenty (všechny tři se v paletě nacházejí na záložce *Standard*).

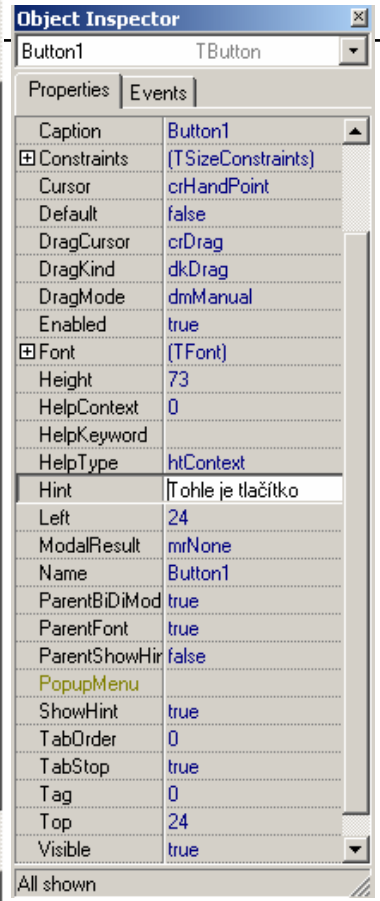
Návěští (*Label*). Jedná se o text, kterým ve formuláři popisujeme další objekty (v našem případě editační řádky). Návěští můžeme rovněž použít jako textový výstup (v našem případě pro vypsání součtu). V inspektorovi vyplňujeme u návěští proměnné Caption (řetězec, který se objeví ve formuláři), Font (otevře se standardní dialog pro výběr parametrů písma) a Name (jméno návěští). Pro snadnější orientaci ve zdrojovém kódu je vhodné přepisovat standardní jména generovaná Builderem jmény vlastními.

Borland Builder C++

seznámení s prostředím

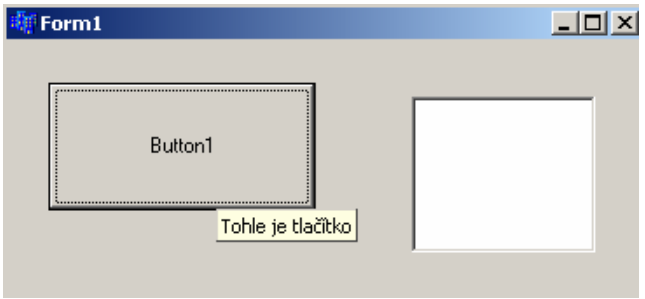
Sestavení okna





Editační řádek (***Edit***). Jedná se o jednoduchý jednořádkový editor, který můžeme využít jako textový vstup programu (v našem případě pro načítání sčítanců). V inspektoru vyplňujeme u editačního řádku ***Text*** (obsah editačního řádku; v našem případě prázdný řetězec – tedy nic), ***Font*** a v případě potřeby jméno řádku Name.

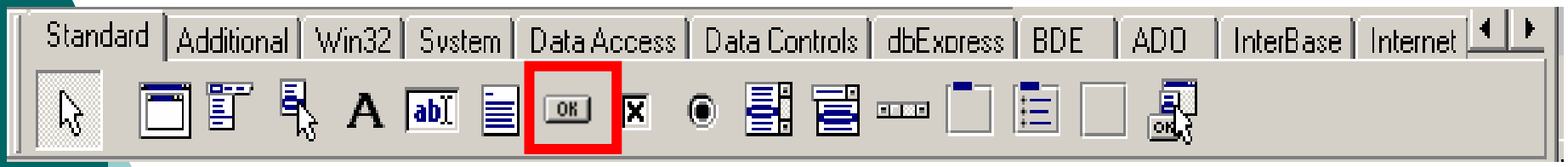
„Bublinkovou" nápovědu nastavíme v inspektoru ***ShowHint*** = **true** a do proměnné ***Hint*** vepíšeme obsah „bublinky" (v našem případě vepíšeme upozornění, že lze vepsat pouze celé číslo).



Borland Builder C++

seznámení s prostředím

Sestavení okna



Tlačítko (*Button*). Základním úkolem tlačítka je dát pokyn k provedení nějaké akce.

Klikne-li uživatel na tlačítko nebo ho stiskne prostřednictvím klávesnice, vždy je generována událost tlačítka *OnClick*.

Tuto událost nalezneme v inspektoru na záložce *Events*.

Vepíšeme-li do editačního řádku vedle události jméno funkce, Builder tuto funkci deklaruje v editoru kódu (jako prázdnou) a její volání pevně sváže s danou událostí.

V našem případě nazveme obslužnou funkci addition a její tělo si popíšeme v další podkapitole.

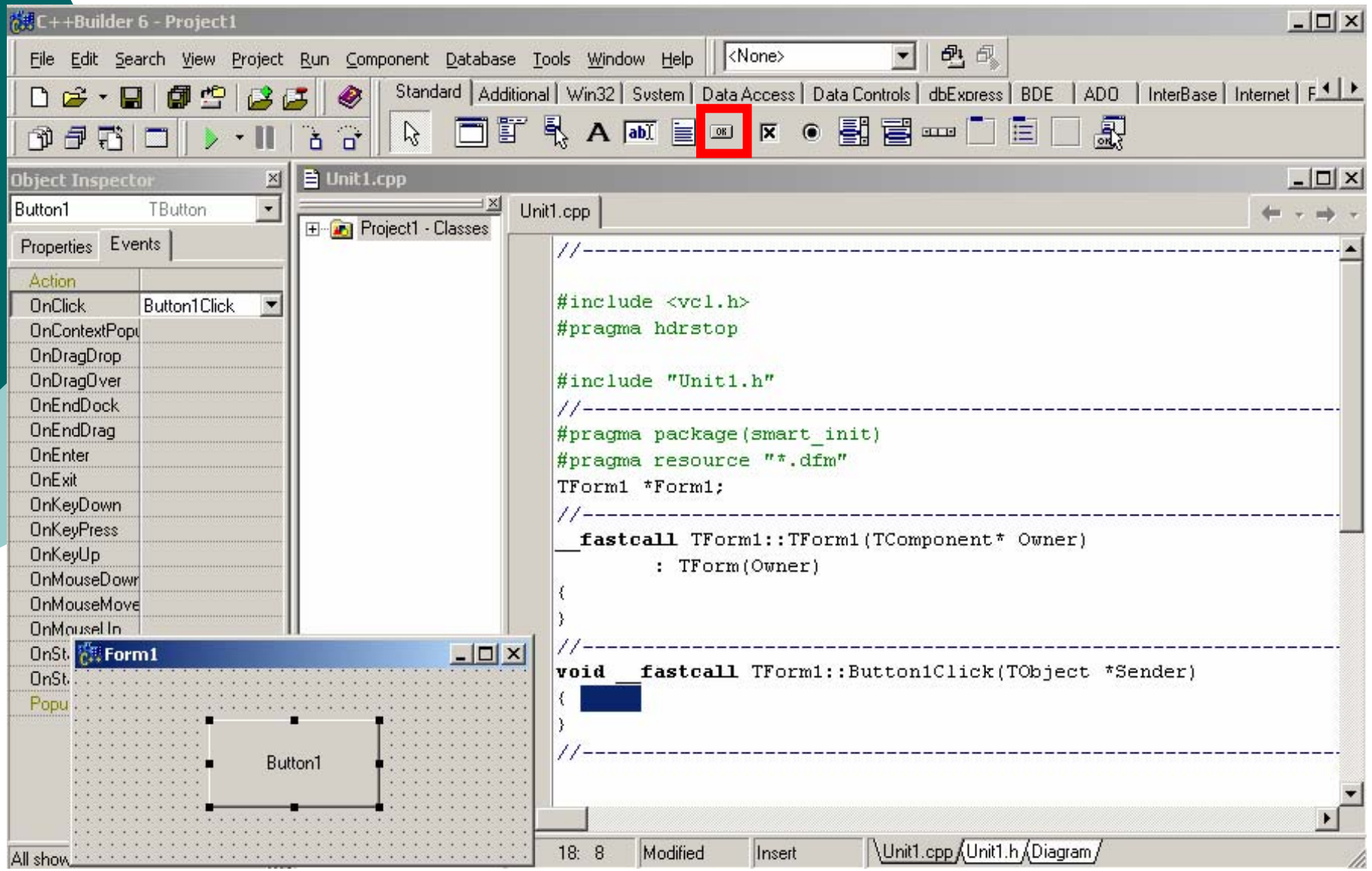
Na záložce *Properties* inspektora zadáme text uvnitř tlačítka Caption = &Počítej (znak & způsobí podtržení následujícího písmene; při stisku klávesové kombinace *Alt+P* dojde ke stlačení tlačítka).

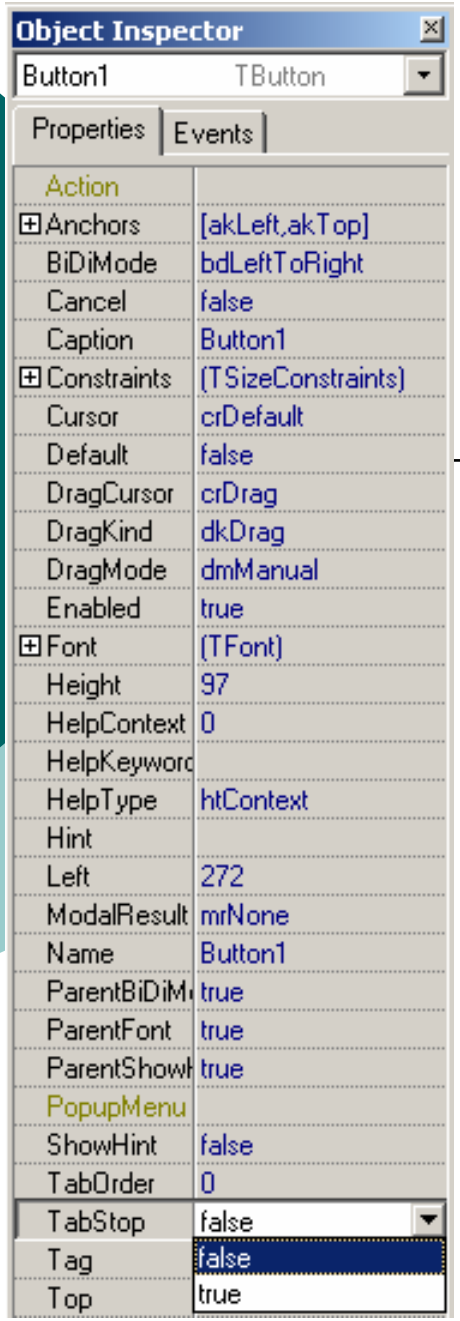
Opět můžeme změnit jméno objektu (Name) a parametry písma uvnitř objektu (Font).

Borland Builder C++

seznámení s prostředím

Sestavení okna





Borland Builder C++

seznámení s prostředím

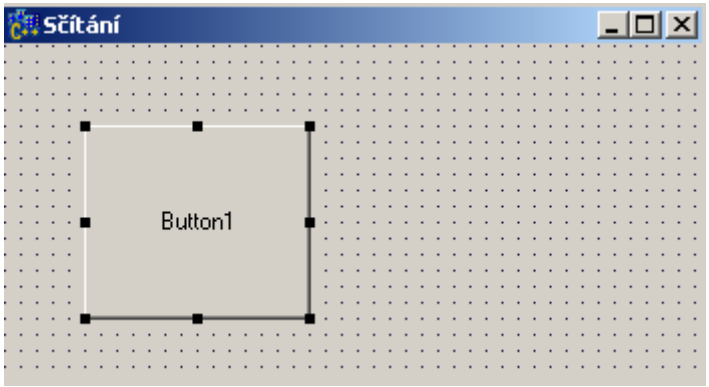
Sestavení okna

Při postupném mačkání tabelační klávesy postupně zaostřujeme jednotlivé komponenty v okně.

Pořadí zaostřování přitom odpovídá pořadí, v němž byly komponenty do okna vkládány.

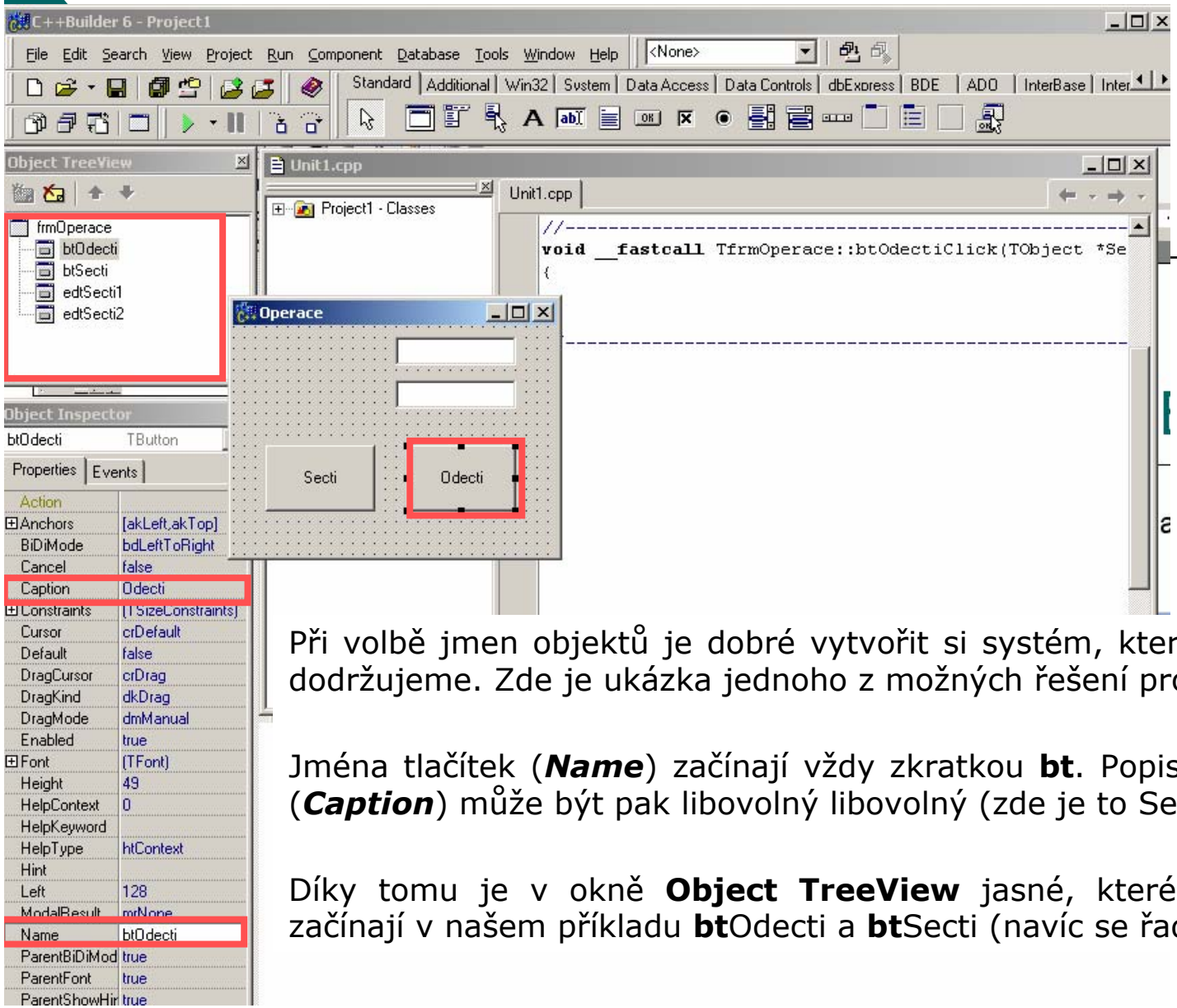
Chceme-li pořadí zaostřování změnit, učiníme tak v inspektoru prostřednictvím parametru *TabOrder.*

Nechceme-li, aby se tabelátor na určitém komponentu zastavil, nastavíme pro něj v inspektoru *TabStop = false.*



Borland Builder C++

seznámení s prostředím



Sestavení okna

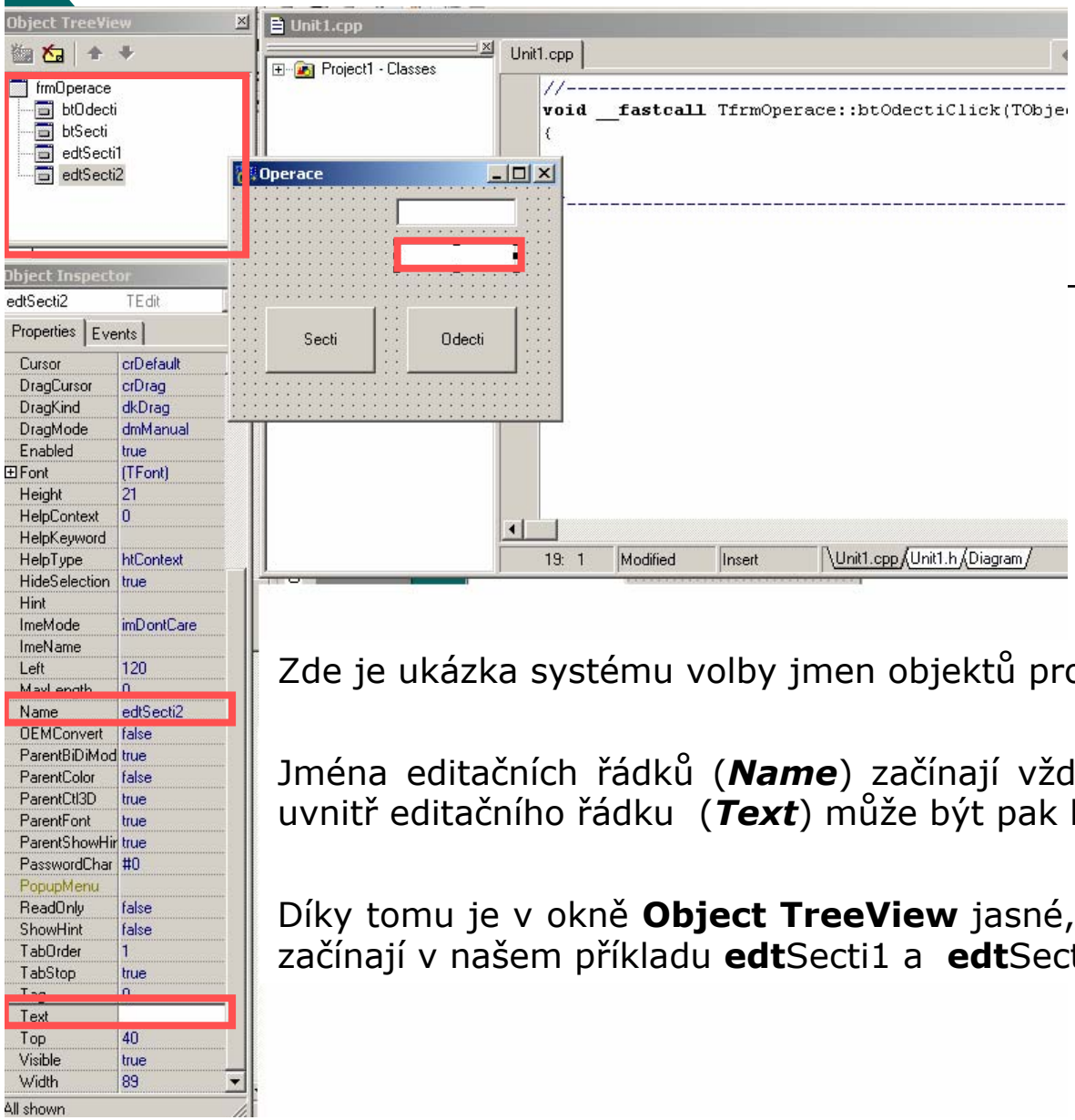
Při volbě jmen objektů je dobré vytvořit si systém, který pak v celém programu dodržujeme. Zde je ukázka jednoho z možných řešení pro tlačítka:

Jména tlačítek (***Name***) začínají vždy zkratkou **bt**. Popis zobrazovaný na tlačítku (***Caption***) může být pak libovolný libovolný (zde je to Secti a Odecti).

Díky tomu je v okně **Object TreeView** jasné, které objekty jsou tlačítka – začínají v našem příkladu **bt**Odecti a **bt**Secti (navíc se řadí abecedně za sebe).

Borland Builder C++

seznámení s prostředím

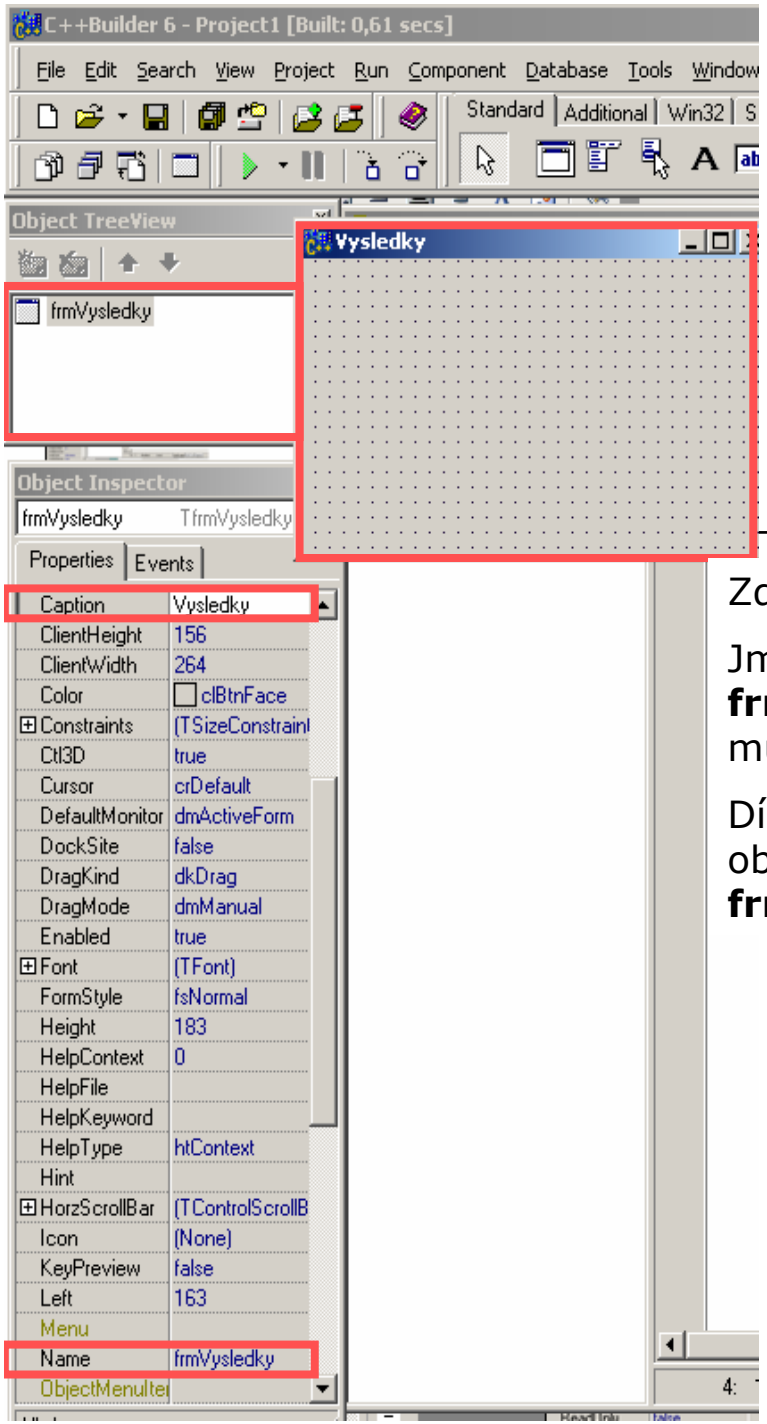


Sestavení okna

Zde je ukázka systému volby jmen objektů pro editační řádky :

Jména editačních řádků (***Name***) začínají vždy zkratkou **edt**. Popis zobrazovaný uvnitř editačního řádku (***Text***) může být pak libovolný (zde popis není žádný).

Díky tomu je v okně **Object TreeView** jasné, které objekty jsou editační řádky – začínají v našem příkladu **edt**Secti1 a **edt**Secti2 (řadí abecedně za sebe).



Borland Builder C++

seznámení s prostředím

Sestavení okna

Zde je ukázka systému volby jmen objektů pro formuláře:

Jména editačních řádků (***Name***) začínají vždy zkratkou **frm**. Popis zobrazovaný uvnitř editačního řádku (***Caption***) může být pak libovolný (zde je to Vysledky).

Díky tomu je v okně **Object TreeView** jasné, které objekty jsou editační řádky – začínají v našem příkladu **frm**Vysledky (řadí abecedně za sebe).

Borland Builder C++

seznámení s prostředím

Ošetření událostí

Jako příklad si vytvoříme jednoduchý program pro sčítání dvou čísel. V našem programu budeme pracovat s jedinou událostí, a to se stiskem tlačítka *Počítej*.

Jakmile uživatel programu toto tlačítko stiskne (objeví se událost tlačítka OnClick), zavoláme funkci addition.

Abychom funkci svázali s uvedenou událostí tlačítka, tlačítko zaostříme (klikneme na něj myší) a v inspektoru vepíšeme na záložce *Events* řetězec addition vedle události OnClick.

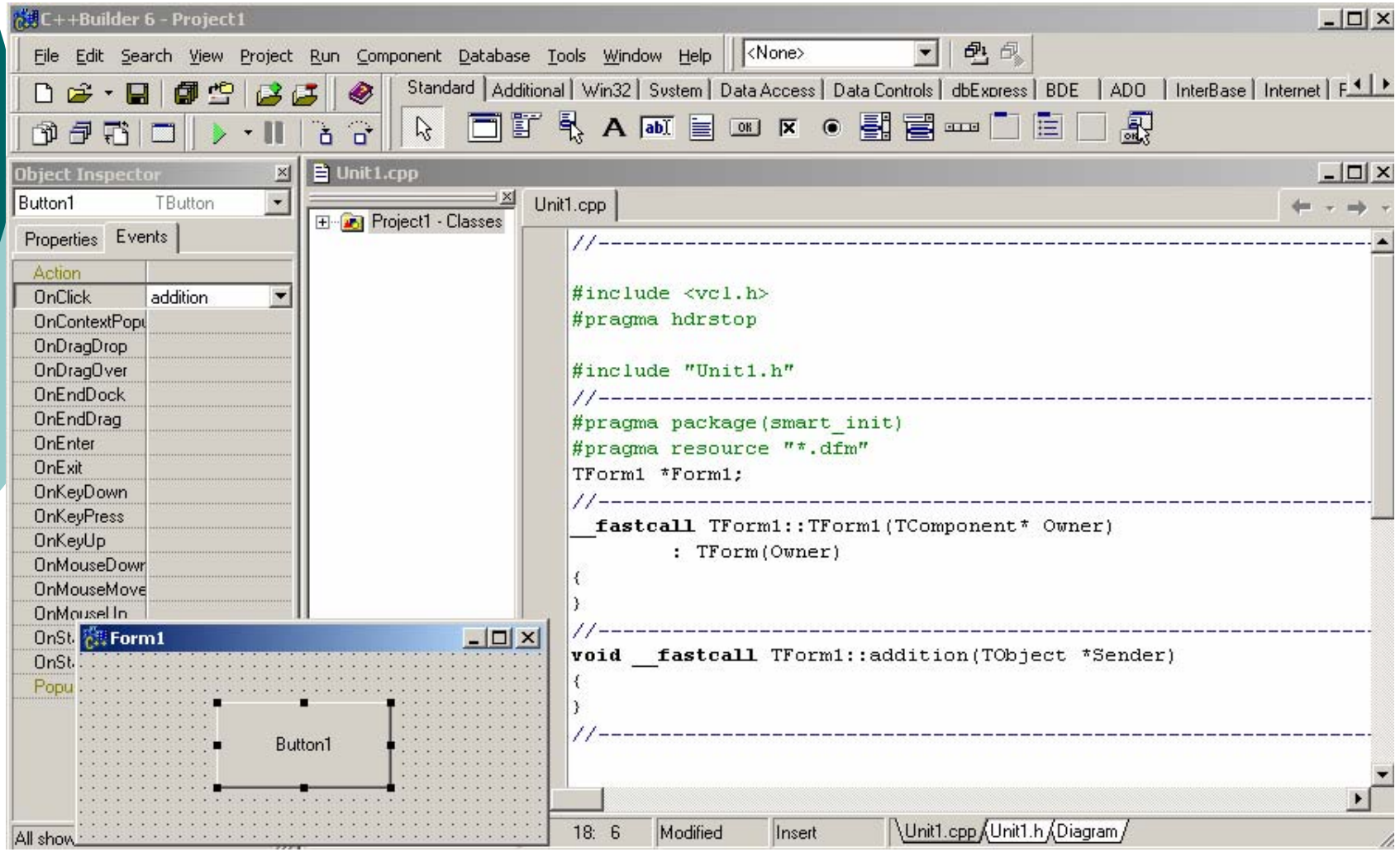
Potvrdíme-li svou volbu stiskem klávesy *Enter*, Builder vygeneruje deklaraci této funkce:

```
void __fastcall TForm1::addition( TObject *Sender)
{
}
```

Borland Builder C++

seznámení s prostředím

Ošetření událostí



Borland Builder C++

seznámení s prostředím

Ošetření událostí

```
void __fastcall TForm1::addition( TObject *Sender)
{
}
```

Funkce je prázdná (neobsahuje žádnou instrukci, nic nedělá).

Reakci na stisk tlačítka musíme mezi složené závorky napsat sami jako posloupnost vhodných instrukcí.

void označuje funkci, která nevrací žádnou hodnotu, žádnou hodnotu neočekáváme.

Zatímco např. **sin(0.5)** vrací sinus 0.5 radiánu.

__fastcall udává způsob, jakým má být funkce volána. Funkce pro ošetření událostí musejí být volány takto.

TForm1::addition říká, že funkce addition je pevně svázána s hlavním formulářem našeho programu (na ploše tohoto formuláře naše tlačítko leží).

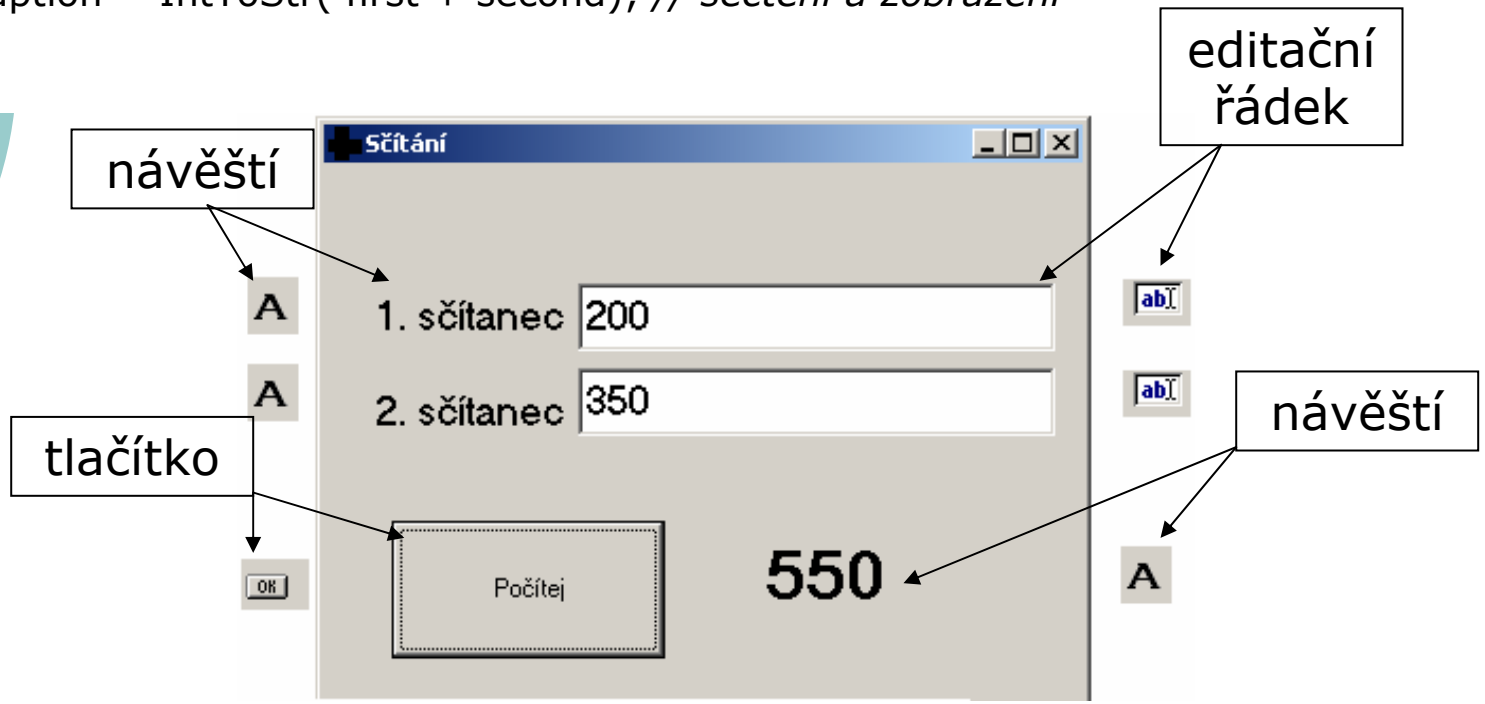
Hlavnímu formuláři jsme ponechali standardní název Form1 (položka *Name* na záložce inspektora *Properties*), a proto má naše funkce tzv. předponu TForm1.

Borland Builder C++

seznámení s prostředím

Ošetření událostí

```
void __fastcall TForm1::addition( TObject *Sender)
{
int first, second; // pomocné proměnné, existující jen uvnitř funkce, typu int (celé číslo)
first = StrToInt( Add1Edit->Text); // první edit.řádek na číslo
second = StrToInt( Add2Edit->Text); // druhý edit.řádek na číslo
Result->Caption = IntToStr( first + second); // sečtení a zobrazení
}
```



Borland Builder C++

seznámení s prostředím

Ošetření událostí

```
void __fastcall TForm1::addition( TObject *Sender)
{
int first, second; // pomocné proměnné, existující jen uvnitř funkce, typu int (celé číslo)
first = StrToInt( Add1Edit->Text); // první edit.řádek na číslo
second = StrToInt( Add2Edit->Text); // druhý edit.řádek na číslo
Result->Caption = IntToStr( first + second); // sečtení a zobrazení
}
```

V závorce za jménem funkce - seznam vstupních parametrů (proměnných, jejichž hodnoty funkci předáváme).

Zde se jedná o adresu proměnné **Sender** (hvězdička - nepředáváme funkci číselný obsah proměnné, ale její adresu). **Sender** je typu **TObject**.

Add1Edit->Text - pracujeme s textem, který uživatel vepíše do 1. editačního řádku (*Name* – Add1Edit). Vepsaný text je uložen ve formě řetězce v proměnné editačního řádku Text.

K provedení operace sčítání musíme převést řetězec na celé číslo.

Konverze – např. standardní funkce **StrToInt, StrToFloat** a reverzní **IntToStr, FloatToStr**. Vstupním parametr-řetězec **Add1Edit->Text**, výstupním parametrem je celé číslo, které uložíme do pomocné proměnné **first**.

Borland Builder C++

seznámení s prostředím

Ošetření událostí

```
void __fastcall TForm1::addition( TObject *Sender)
{
int first, second; // první a druhý sčítanec
first = StrToInt( Add1Edit->Text); // první edit.řádek na číslo
second = StrToInt( Add2Edit->Text); // druhý edit.řádek na číslo
Result->Caption = IntToStr( first + second); // sečtení a zobrazení
}
```

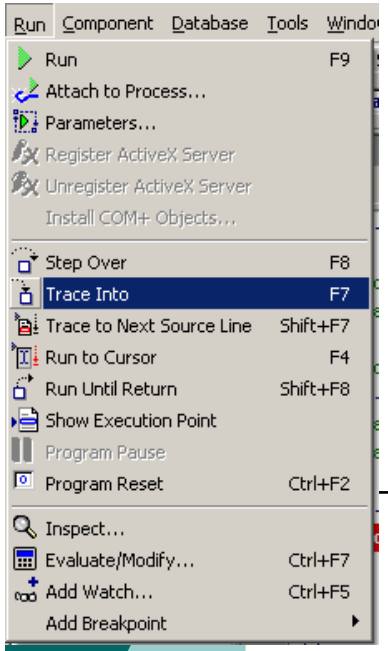
S obsahem druhého editačního řádku a s jeho převodem na druhý sčítanec je to obdobné.

Výsledkem je druhý řetězec převedený na celé číslo **second**.

Na posledním řádku obě čísla sečteme (first+second) a součet převedeme z celočíselné formy na řetězec (IntToStr, *Integer To String*).

K zobrazení získaného řetězce využijeme modrého návěští vedle tlačítka *Počítej* (*Name* – Result). Text návěští je uložen v jeho proměnné Caption. Konstrukce **Result->Caption = IntToStr(first + second);** říká, že řetězec s ukládáme do proměnné **Caption**, která patří návěští Result.

Komentář - libovolný řetězec, umístěný za dvojité lomítko. Text za dvojitým lomítkem je překladačem ignorován.

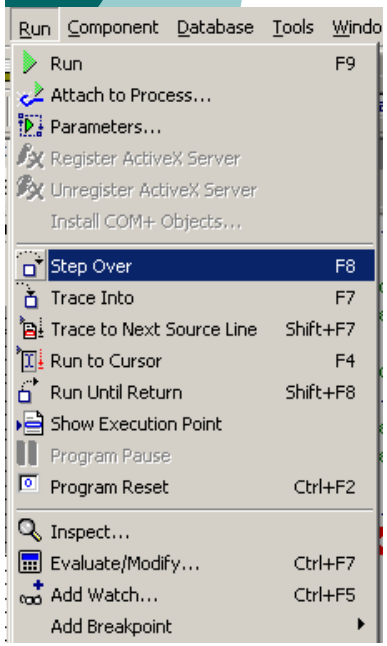


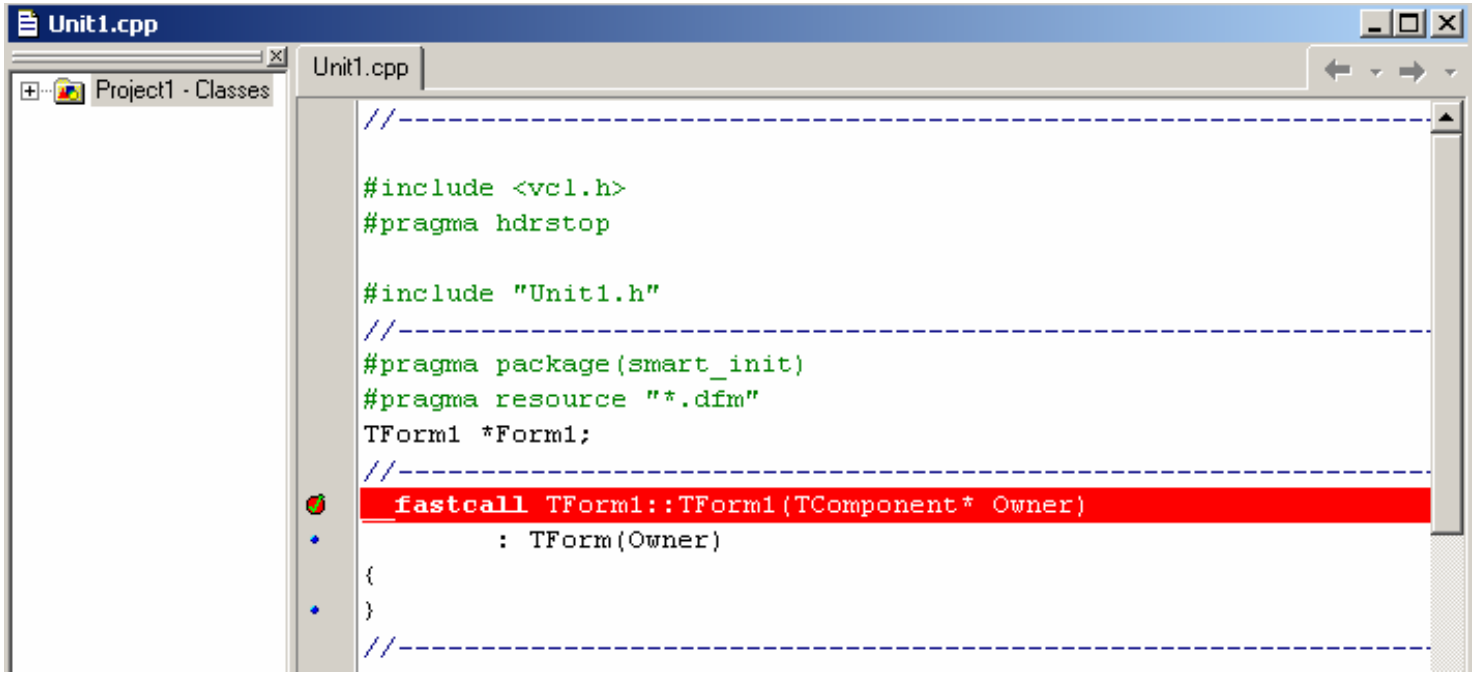
Borland Builder C++

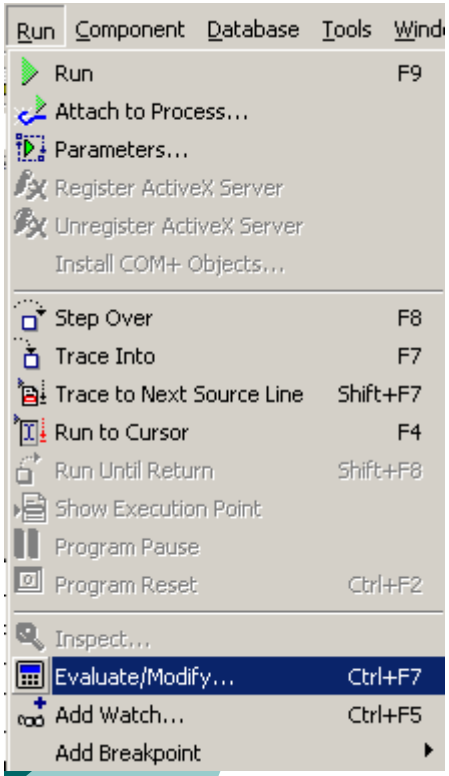
seznámení s prostředím

Ladění

Při ladění zastavíme program na začátku bloku, v němž předpokládáme chybu (na dpovídající řádek programu vložíme přerušovací bod, ***breakpoint***.
Poté kritický blok krokujeme pomocí ***Trace Into*** nebo ***Step Over***. V jednotlivých krocích prohlížíme obsah proměnných a ověřujeme správnost jejich obsahu.
Pokud zjistíme nesprávnou hodnotu, můžeme ji pro další ladění nahradit hodnotou korektní.





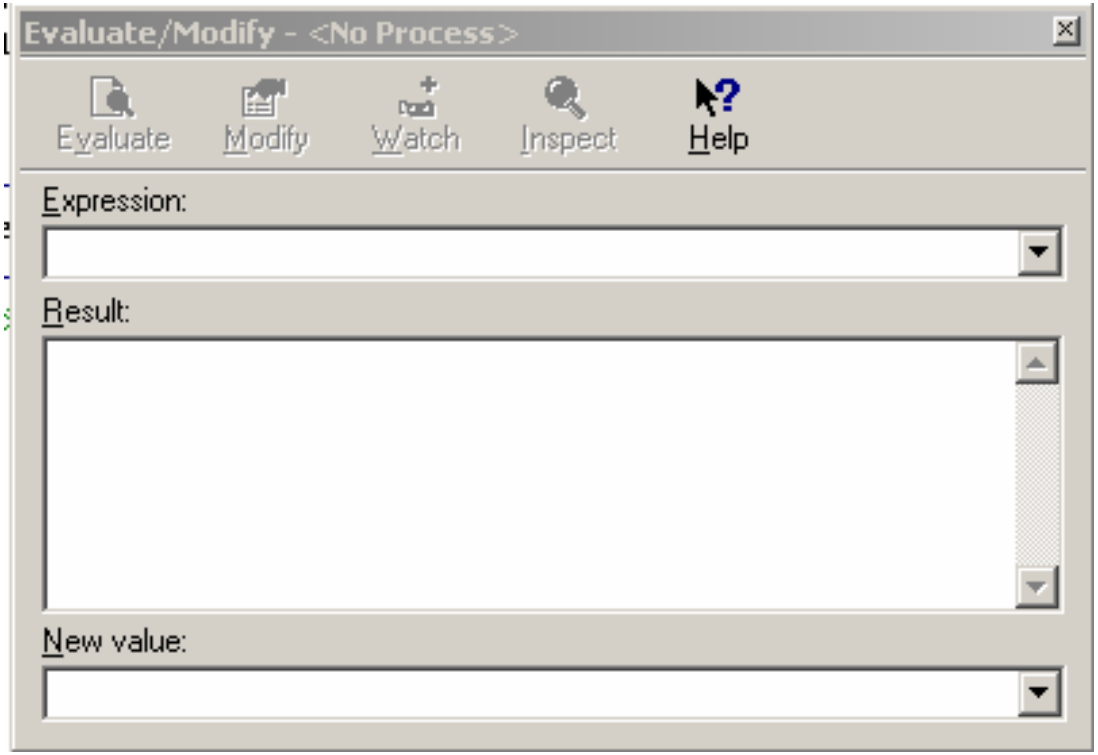


Borland Builder C++

seznámení s prostředím

Ladění

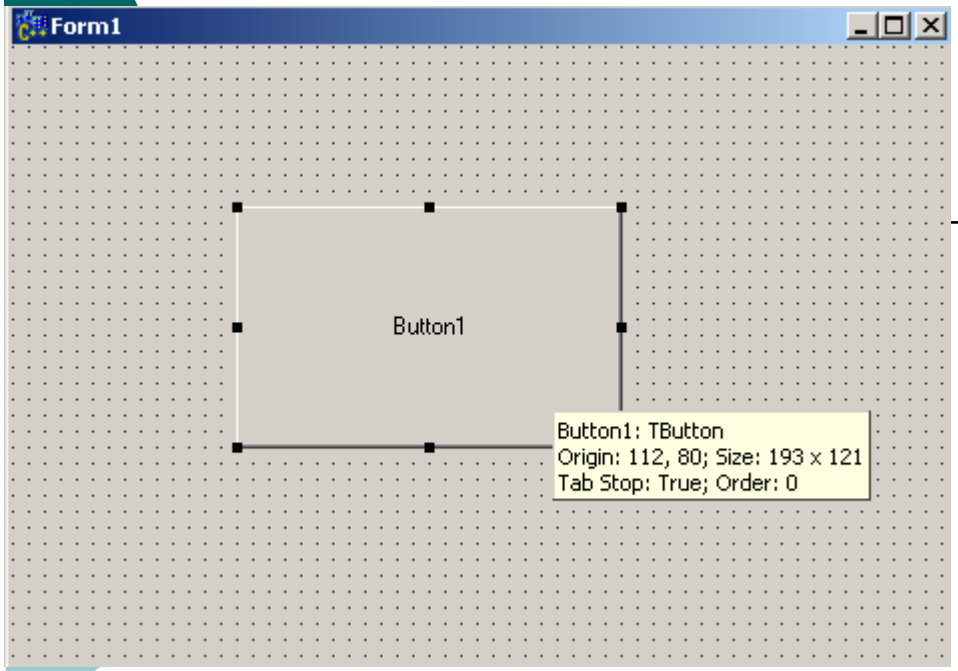
Ke kontrole obsahu proměnných a jejich změně slouží položka menu **Run** → **Evaluate/ Modify**. Výběrem této položky otevřeme okno. Do řádku **Expression** vepíšeme název proměnné, stiskneme **Evaluate** a v editačním poli **Result** objeví její obsah. Chceme-li obsah proměnné změnit, vepíšeme do řádku **New value** novou hodnotu proměnné a stiskneme Modify. Do řádku **Expression** lze psát i celé výrazy (např. i+j).



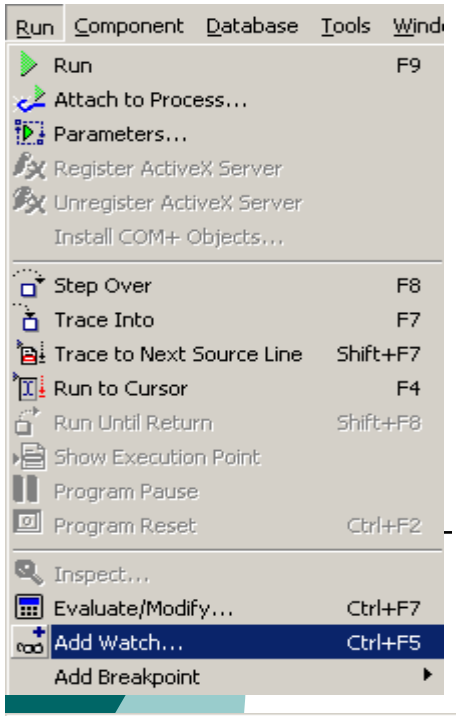
Borland Builder C++

seznámení s prostředím

Ladění



K prostému prohlížení obsahu proměnných slouží jednak „bublinová" nápověda editačního okna (zastavíme-li kurzor myši na jménu proměnné, objeví se barevný obdélník s textovým vyjádřením obsahu proměnné).



Borland Builder C++

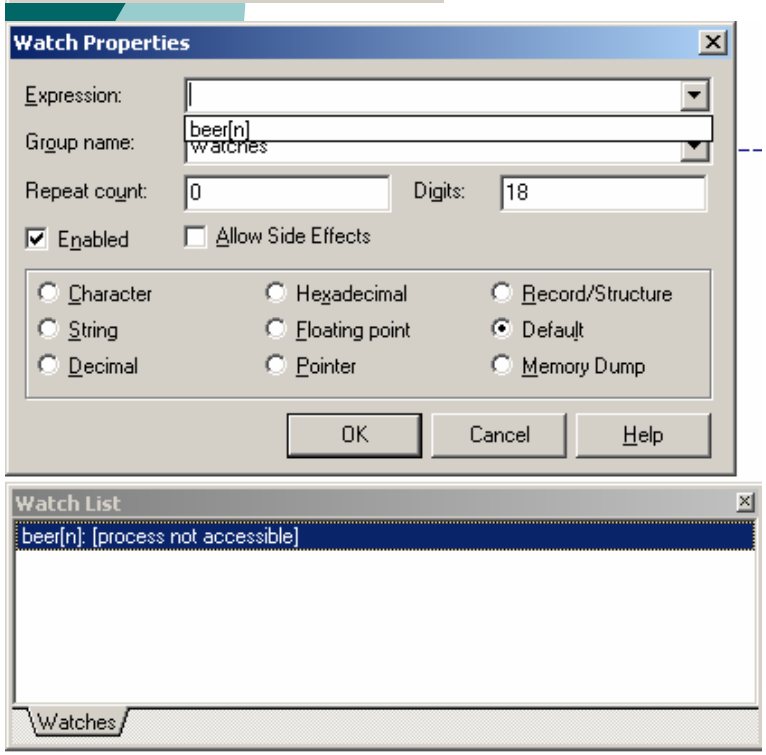
seznámení s prostředím

Ladění

K prostému prohlížení obsahu proměnných slouží i tzv. **Watch List** = okno (otevřeme ho v menu Run → **Add watch**), zobrazující seznam vložených proměnných a jejich obsah.

Proměnné vkládáme stisknutím klávesy **Insert** a mažeme klávesou **Delete**.

Proměnnou (nebo výraz) do okna Watch List je možné rovněž přenést z řádku **Expression** okna **Evaluate/Modify** stiskem **Watch**.



Ladicí nástroje Builderu jsou velmi efektivní a pohodlné.

Přesto je lepší dobře si promyslet.

Nakreslit algoritmus sestavovaného programu, abychom se nedopouštěli zbytečných logických omylů.

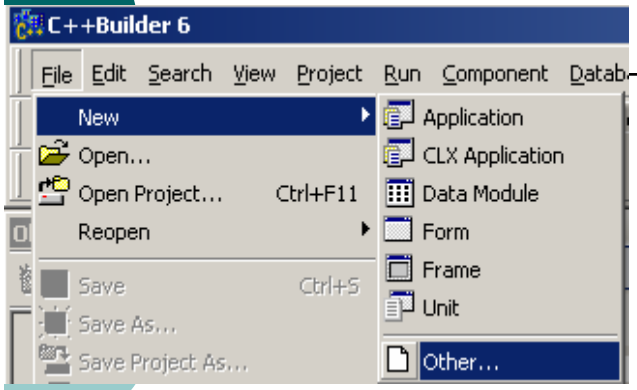
Nutné je se naučit syntaxi programovacího jazyka, abychom vyvarovali zbytečných omylů syntaktických.

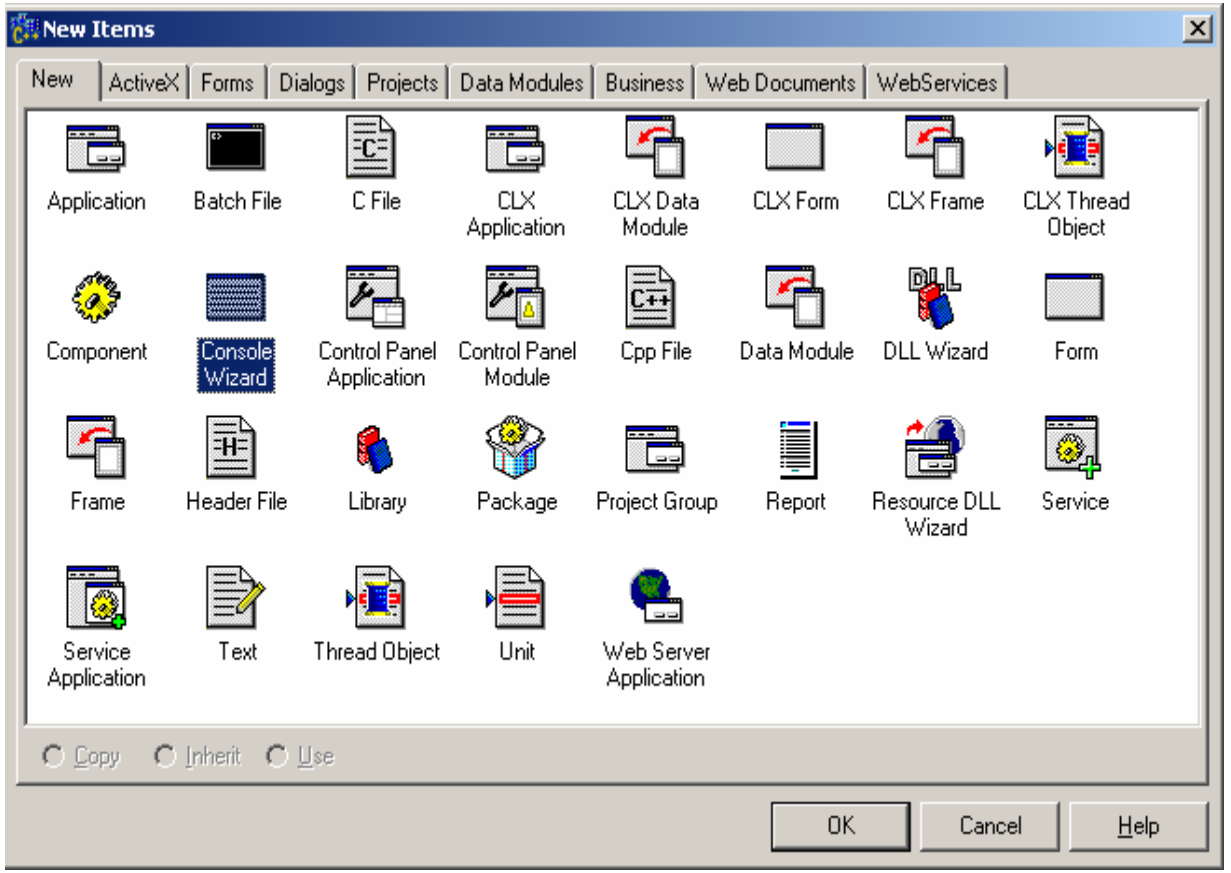
Ctrl+F5

Borland Builder C++

seznámení s prostředím

Spuštění konzolové aplikace

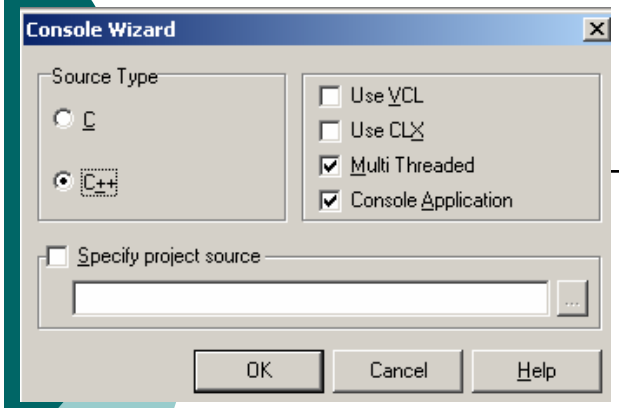


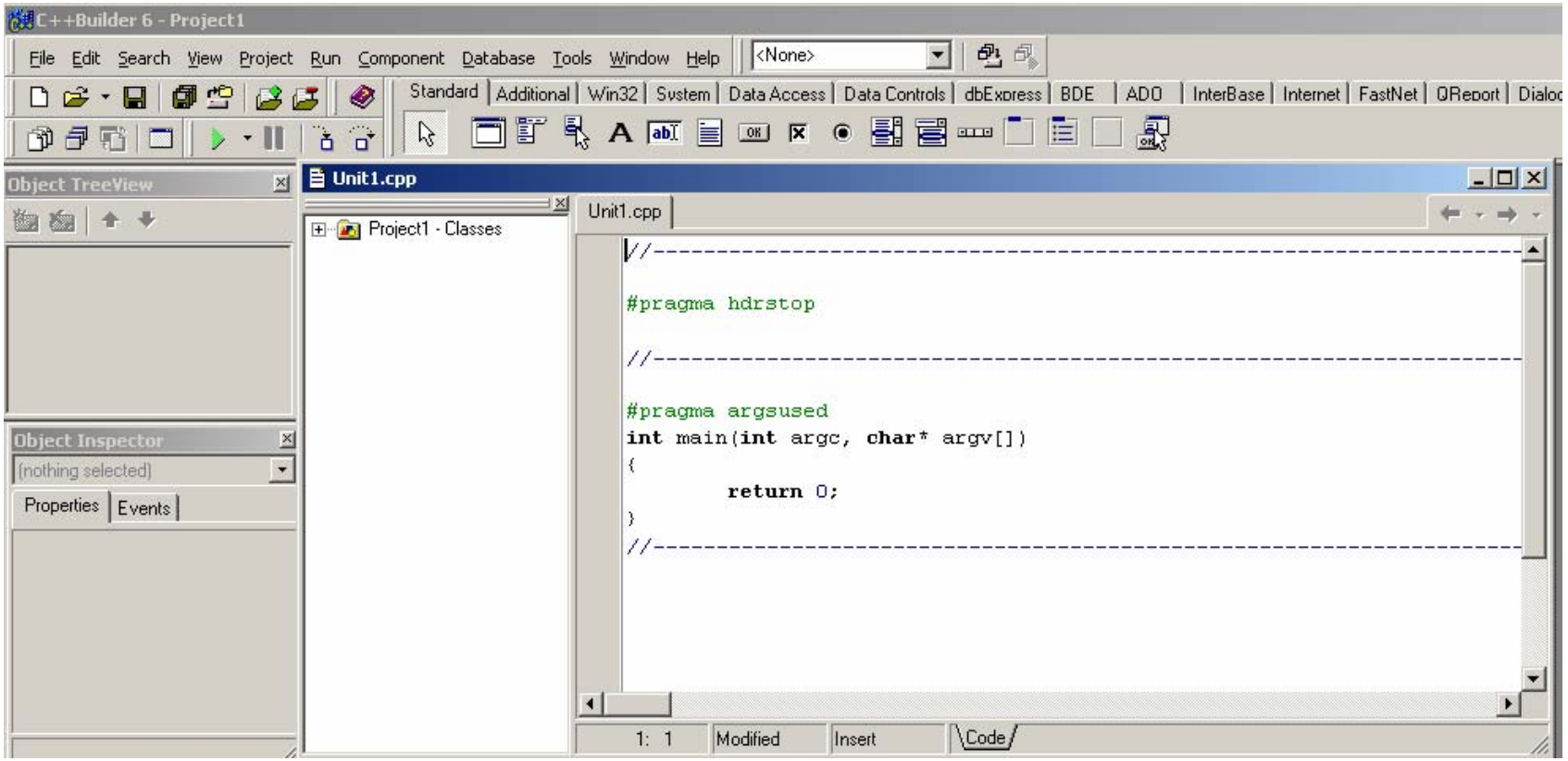


Borland Builder C++

seznámení s prostředím

Spuštění konzolové aplikace



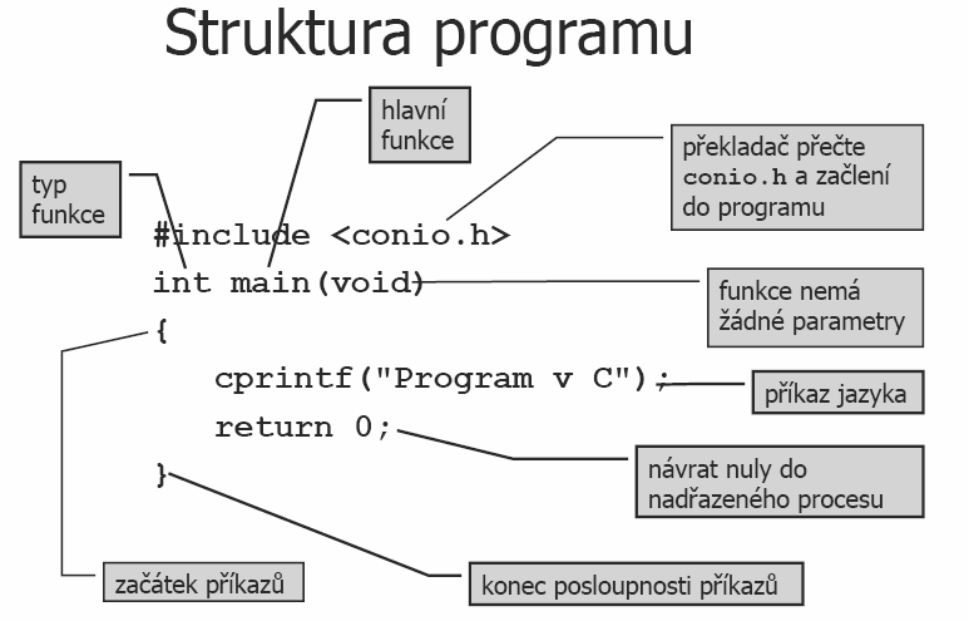


Borland Builder C++

seznámení s prostředím

Hlavičkové soubory

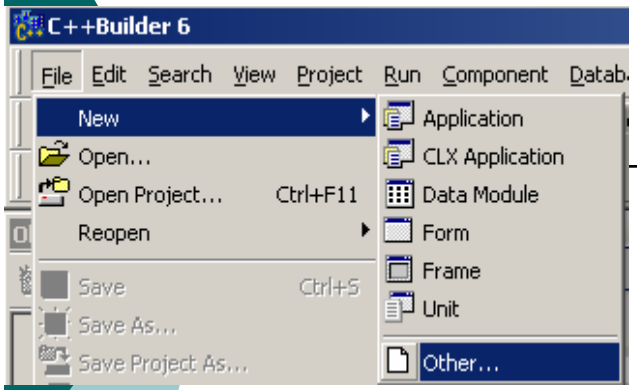
- Obsahují informace o knihovních funkcích.
- Uloženo s příponou **.h**.
- Přidávají se k programu pomocí direktivy preprocesoru **#include**.
- Direktiva říká preprocesoru, aby přečetl jiný soubor začlenil jej do programu.

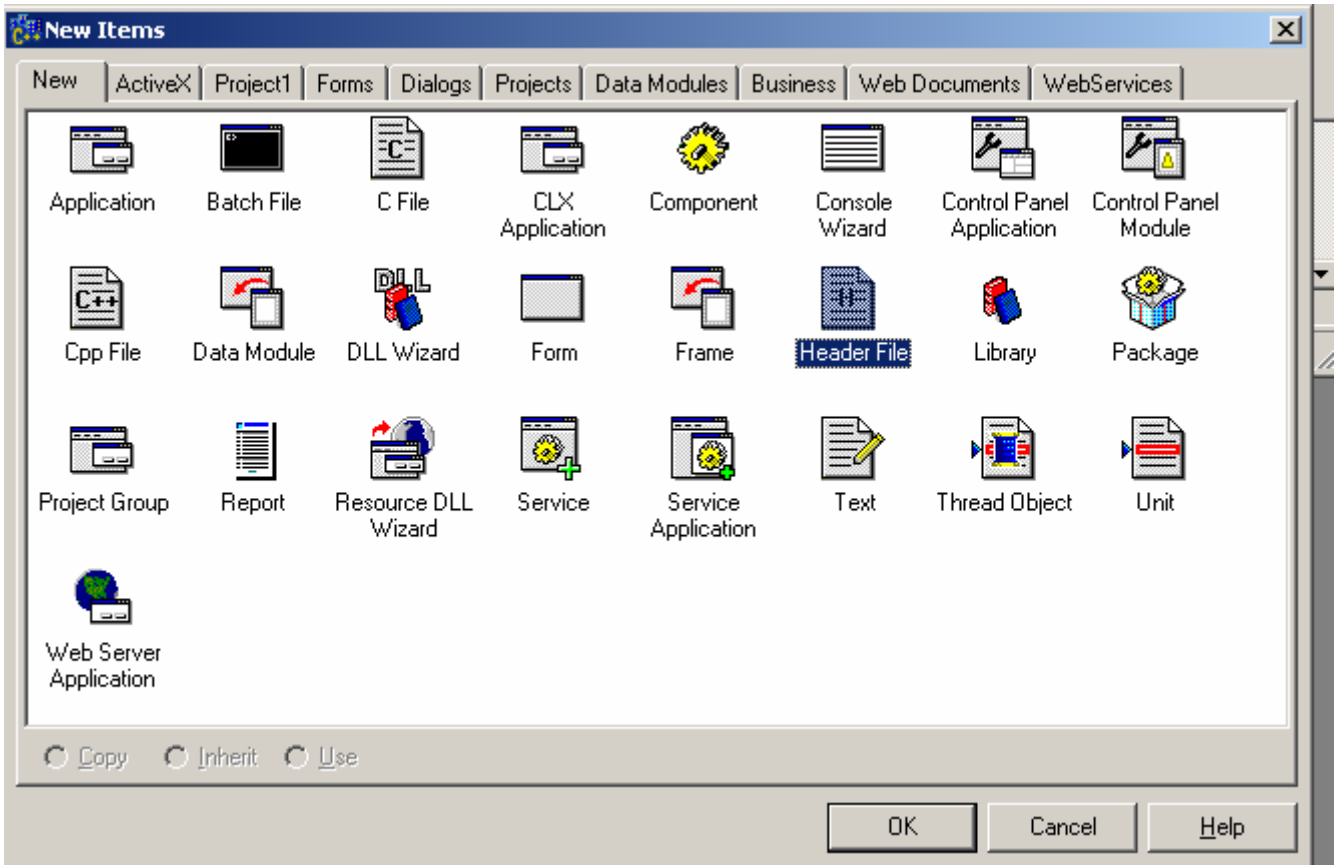


Borland Builder C++

seznámení s prostředím

Otevření hlavičkového souboru

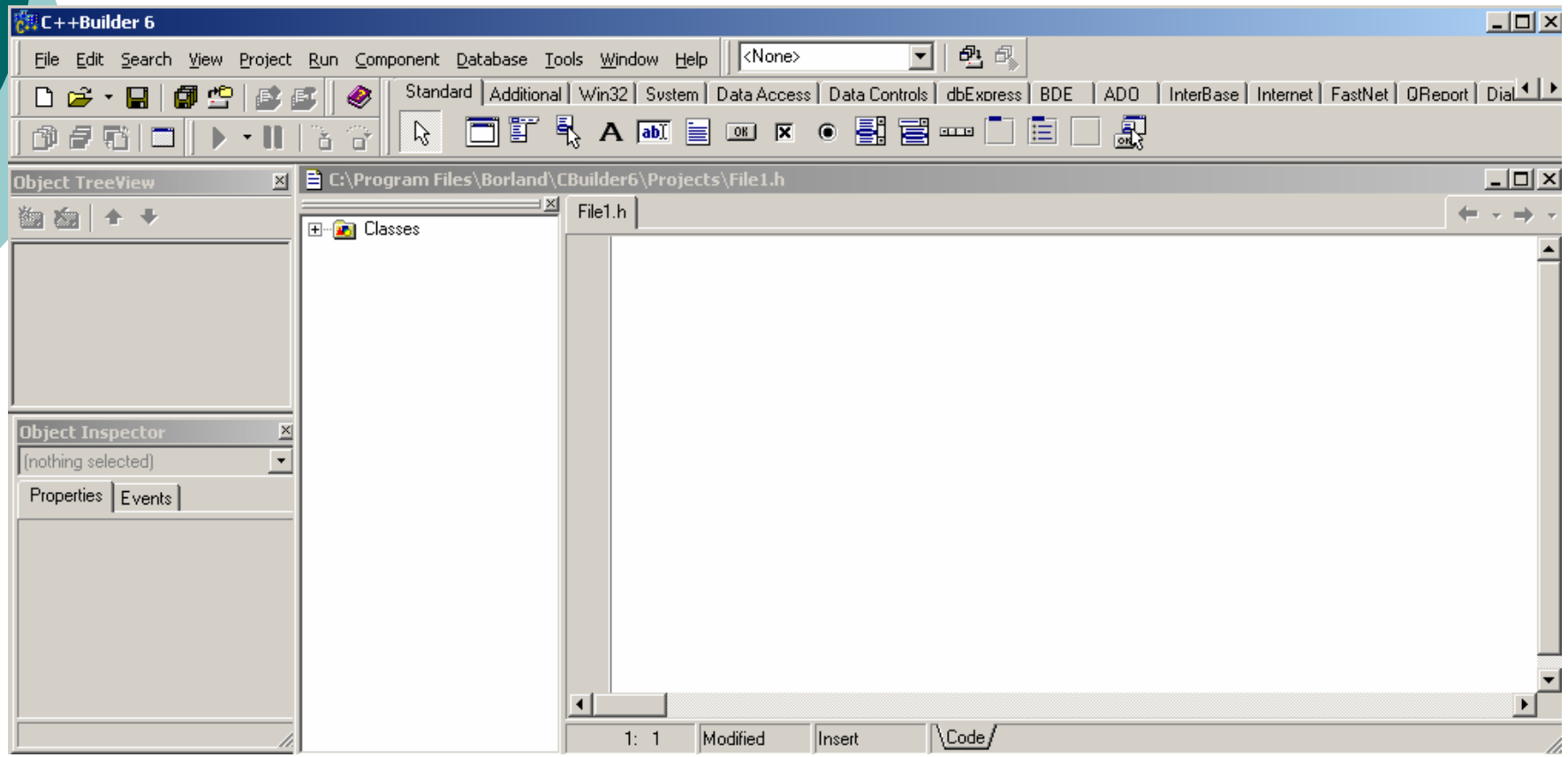




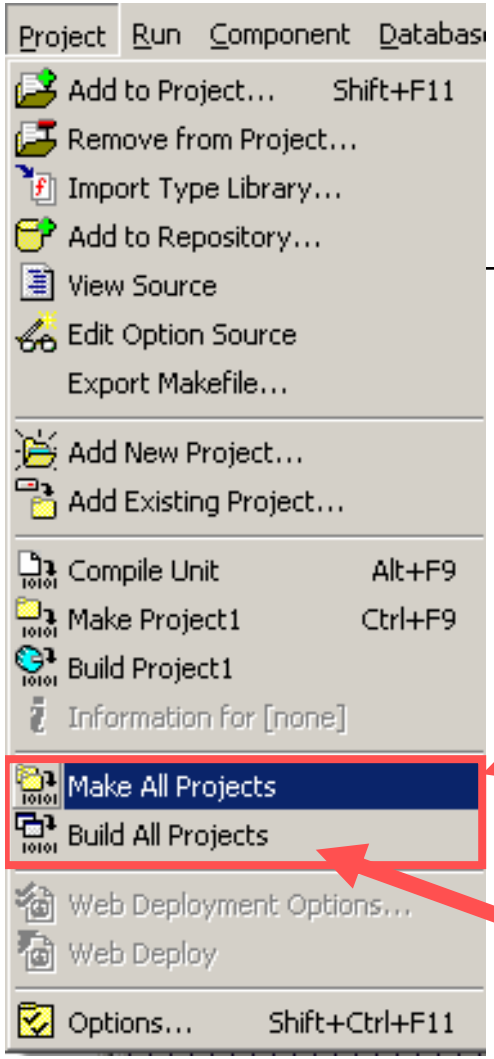
Borland Builder C++

seznámení s prostředím

Otevření hlavičkového souboru



Vytvoření projektu (*.exe souboru) v Builderu



Pro zdar akce je nutné předem v Builderu provést přilinkování knihoven (viz další snímky).

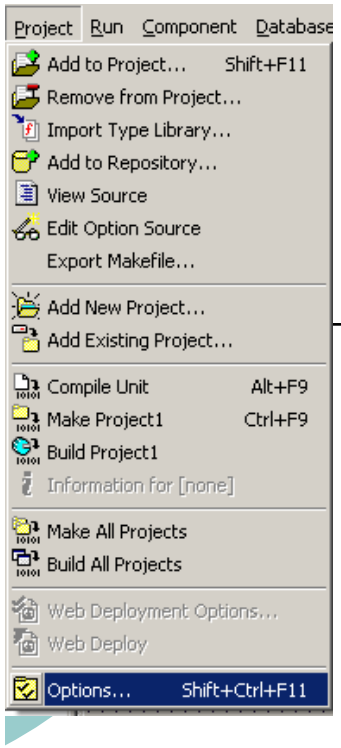
zkompiluje projekt, vytvoří pomocné soubory a *.exe soubor

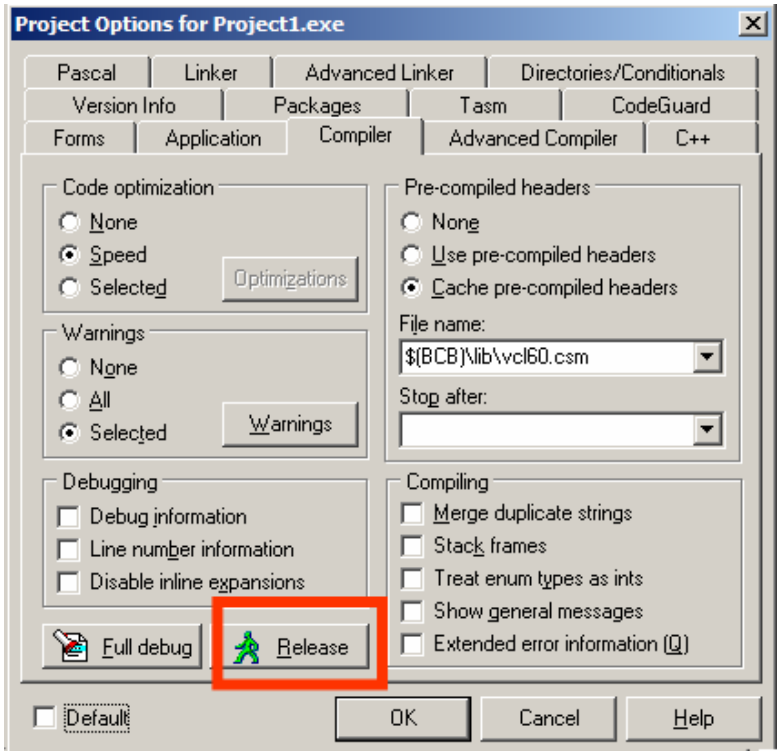
vytváří jen změněné části projektu

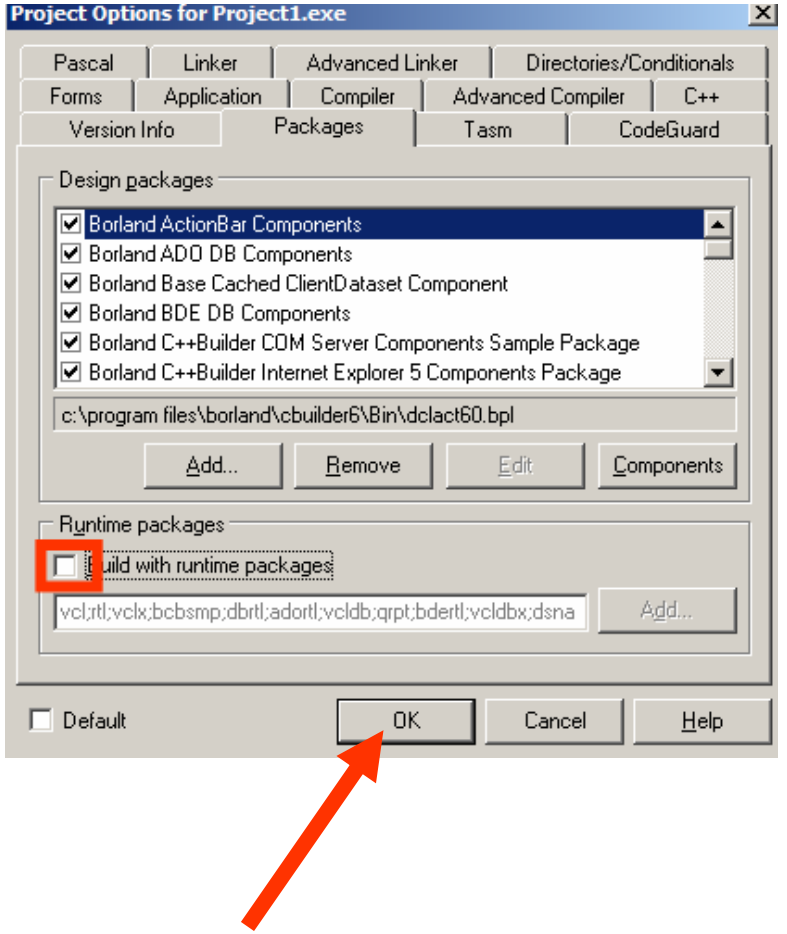
zkompiluje projekt, vytvoří pomocné soubory a *.exe soubor

vytváří celý projekt i nezměněné části

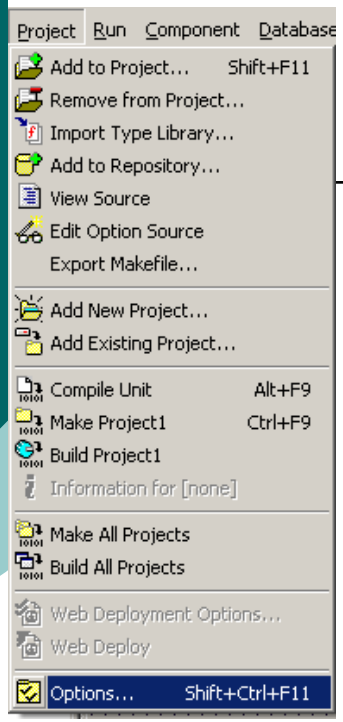
Přilinkování knihoven v Builderu

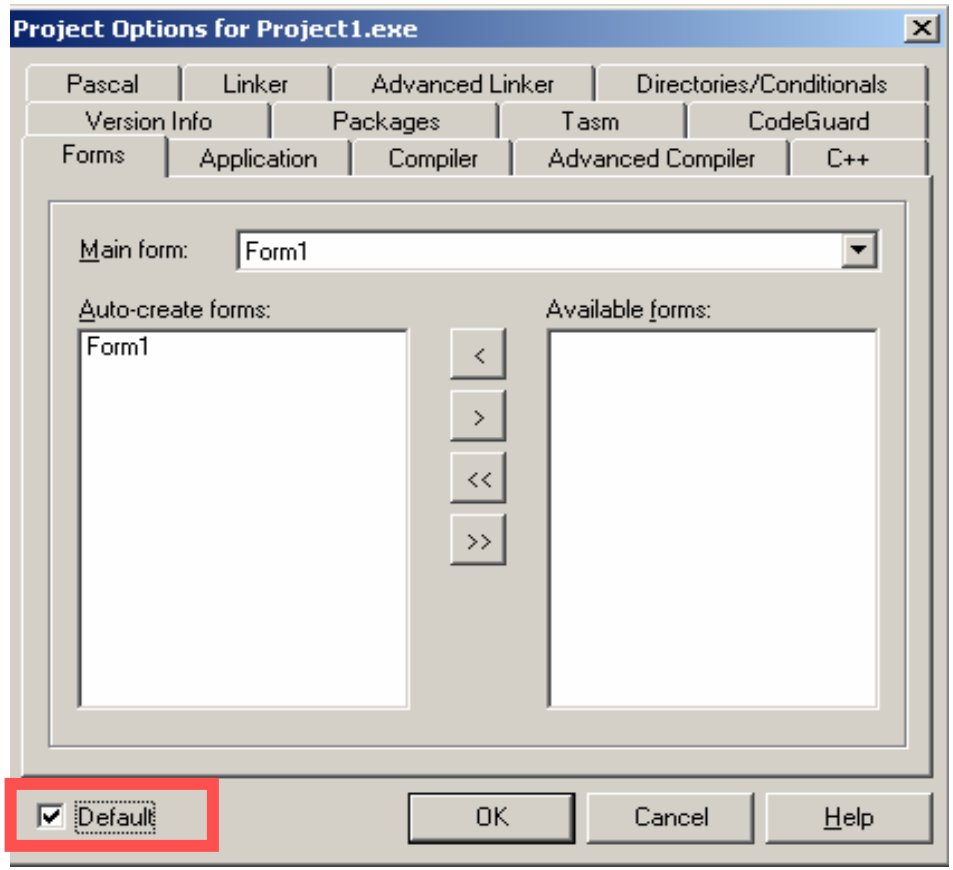




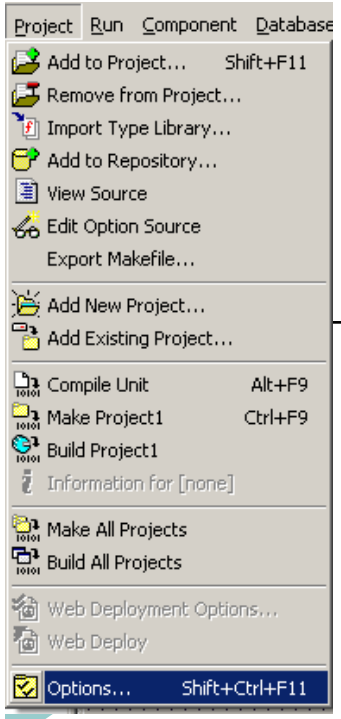


Možná nastavení Builderu



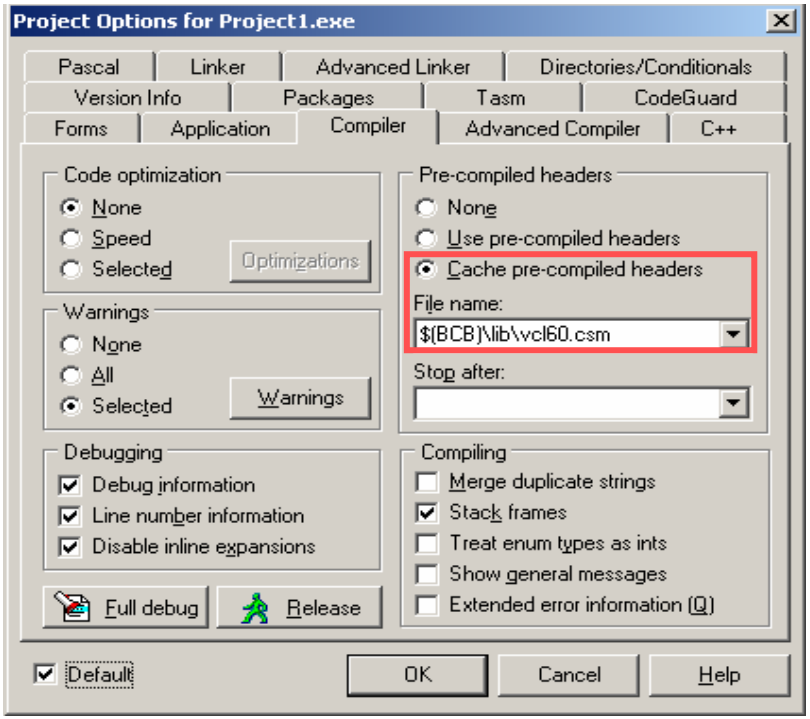


Default – znamená to, že nastavení, která dále provedeme budou společné pro všecky nové projekty.

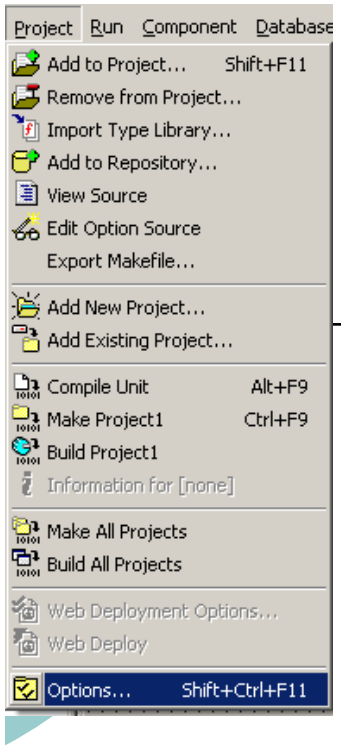


Možná nastavení Builderu

Volba ***Cache pre-compiled headers*** (hlavičkové soubory se kompilují podle potřeby). První kompilace vaší aplikace trvá delší dobu (musí se kompilovat víc jak stotisíc řádků v různých souborech), ale další proběhnou velmi rychle – nezměněné soubory se již nekompilují.

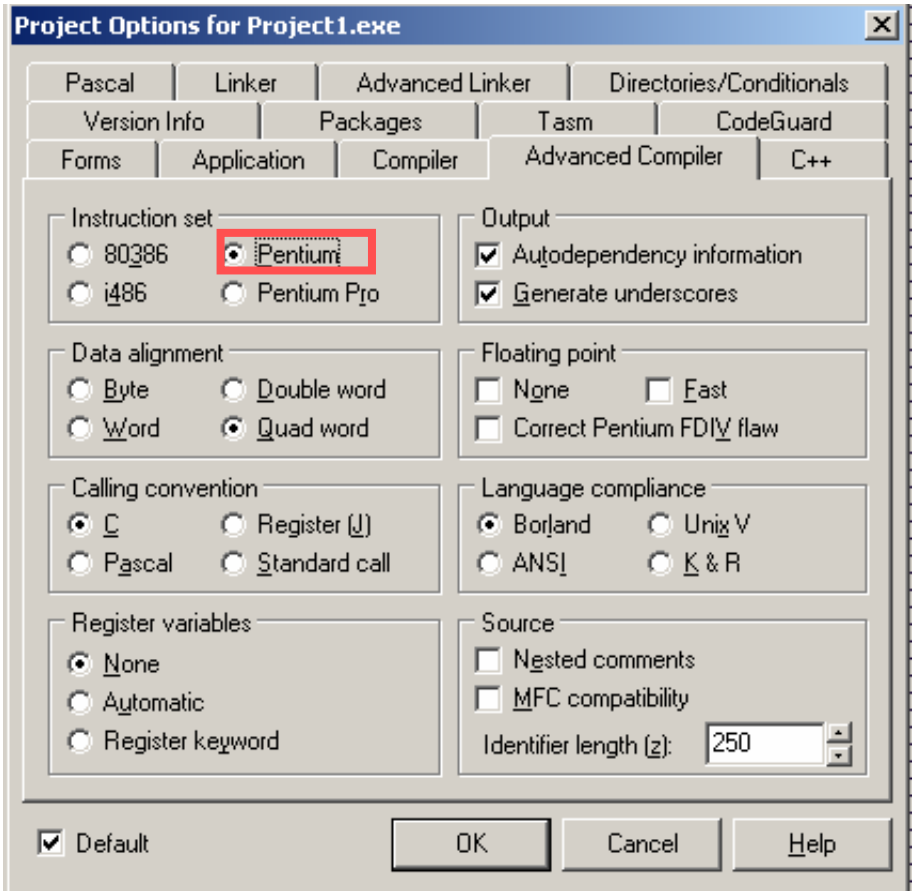


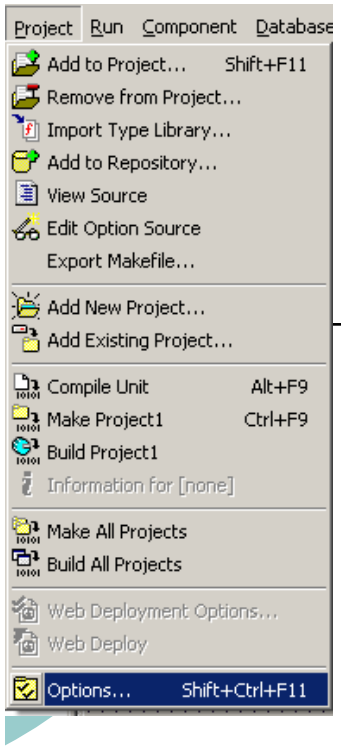
Do položky ***File name*** uvedeme adresář a soubor, do kterého se zkompilované hlavičkové soubory uloží (Pozn.: adresář $(BCB) prostředí C++Builderu pochopí jako adresář, v kterém je C++Builder nainstalovaný).



Možná nastavení Builderu

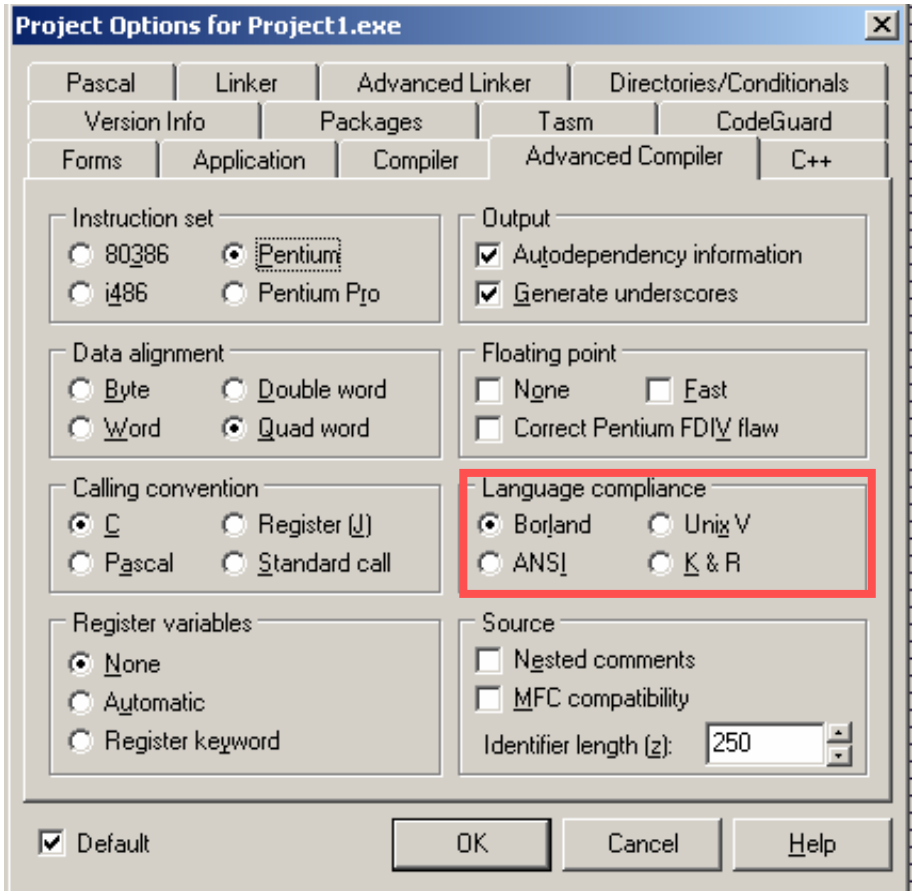
Zapnutím vlastnosti Pentium – kompilátor zkompiluje vaši aplikaci tak, aby optimálně běžela na počítačích s procesorem Pentium.



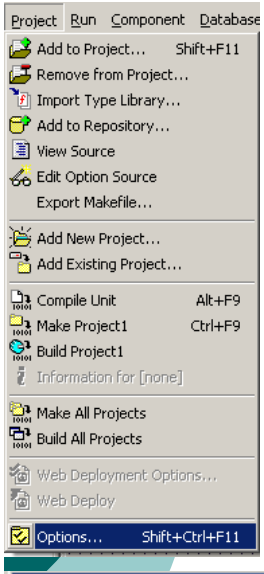


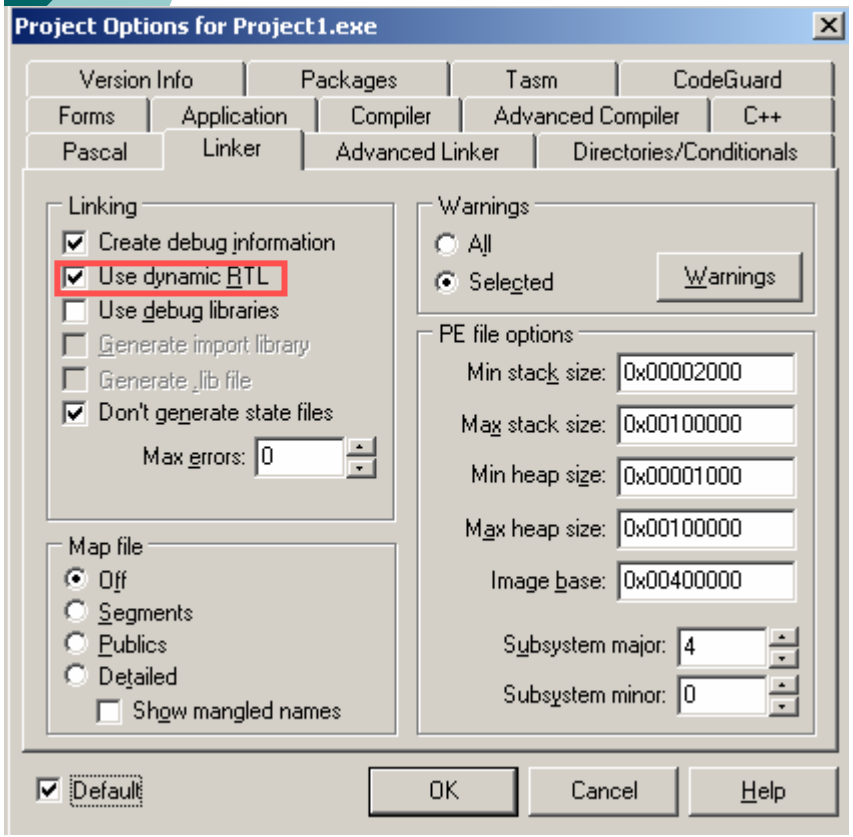
Možná nastavení Builderu

Možnost nastavení normy překladače.



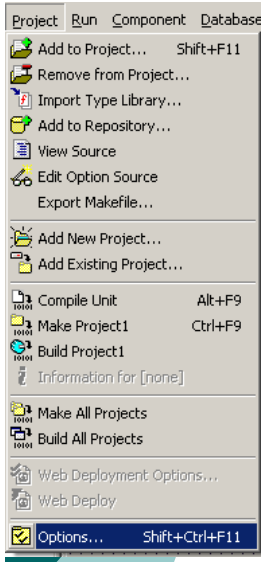
Možná nastavení Builderu





Jestliže vaši aplikaci vyvíjíte, zapněte položku ***Use dynamic RTL*** (použít dynamické knihovny). Zkompilovaná aplikace - exe soubor bude malý.

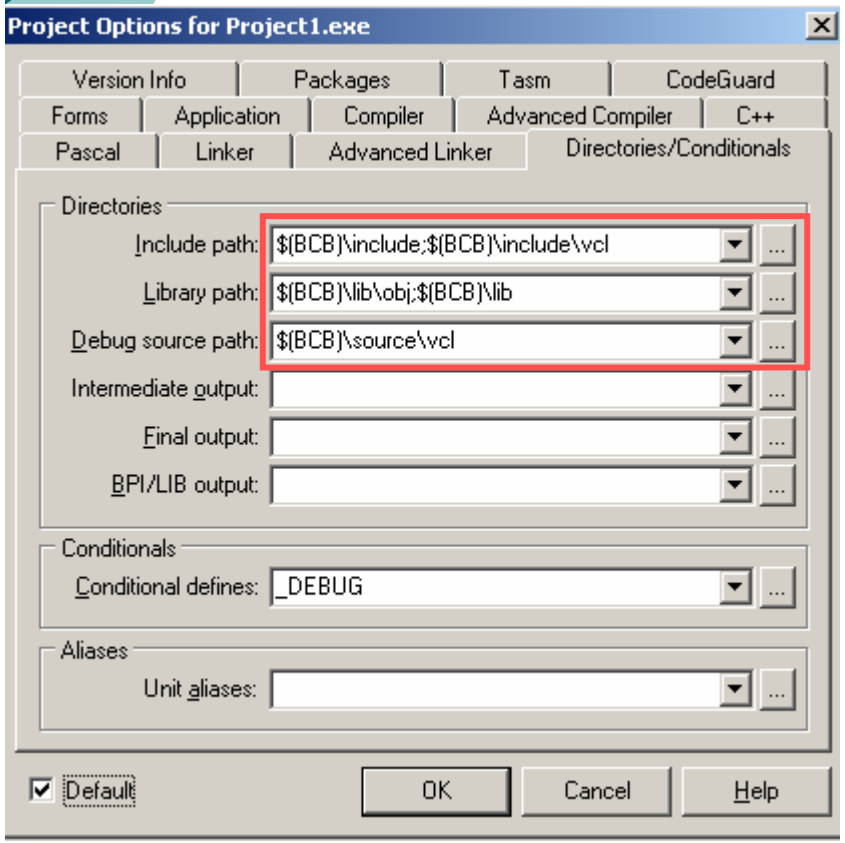
Jestliže ale chcete takovýto soubor zpouštět na počítačích, které nemají nainstalovaný C++Builder raději tuto vlastnost vypněte a aplikaci znovu zkompilujte.



Možná nastavení Builderu

Příslušné položky by měli obsahovat alespoň ty adresáře, které vidíte na obrázku.

Stisknutím OK se vše uloží.



Literatura

1. Raida, Z., Fiala, P. Počítače a programování 2. Brno: FEKT VUT v Brně, 2002.
2. http://www.urel.feec.vutbr.cz/~raida/bpc2/bpc2_ver1.pdf
3. http://www.dbme.feec.vutbr.cz/ubmi/courses/pc2/private/p01.pdf
4. http://www.dbme.feec.vutbr.cz/~provaznik/page.php?lang=cz&page=pc2pr.php
5. http://user.edi.fmph.uniba.sk/salanci/C/index.html