David Matoušek

C++

BEZ PŘEDCHOZÍCH ZNALOSTÍ

Datové typy, operátory, řídicí struktury

Funkce, ukazatele, odvozené datové typy

Pole, řetězce, vstup a výstup programu

Práce se soubory, obsluha výjimek

computer press

David Matoušek

C++ bez předchozích znalostí

Computer Press
Brno
2016

C++ bez předchozích znalostí

David Matoušek

Obálka: Martin Sodomka
Odpovědný redaktor: Martin Herodek
Technický redaktor: Jiří Matoušek

Objednávky knih:
http://knihy.cpress.cz
www.albatrosmedia.cz
eshop@albatrosmedia.cz
bezplatná linka 800 555 513
ISBN 978-80-251-4640-8

Vydalo nakladatelství Computer Press v Brně roku 2016 ve společnosti Albatros Media a. s.
se sídlem Na Pankráci 30, Praha 4. Číslo publikace 23 292.

1. vydání

ALBATROS MEDIA a.s.

Obsah

Předmluva

Kniha, kterou právě držíte v ruce, je určena zájemcům, kteří se chtějí naučit programovat v jazyce C++. Provádí čtenáře od začátků programování až k pokročilým záležitostem, jako je objektově orientované programování a práce se soubory. Text je doplněn četnými příklady.

Prakticky se pracuje s vývojovým prostředím **Dev-C++**, které bylo zvoleno především s ohledem na nízké nároky při instalaci, dobrou lokalizaci do češtiny a relativně snadné ovládání. Prostředí je volně k dispozici, je poskytováno na základě GNU licence.

V úvodní kapitole jsou vysvětleny základní pojmy programování, projdeme instalaci vývojového prostředí (IDE) Dev-C++ a sestavíme první program. Nakonec se seznámíme se základním dělením datových typů.

Kapitoly 2 a 3 jsou věnovány datovým typům pro celá a reálná čísla. Jsou také vysvětleny základní vstupně/výstupní operace (načítání čísel z klávesnice a jejich výpis na obrazovku) a operátory. Pozornost je věnována i problematice tzv. konverzí (převodů hodnot různých datových typů).

Kapitoly 4 a 5 jsou věnovány větvení programu pomocí příkazů `if` a `switch` a konstrukci cyklů typu `while`, `do...while` a `for`. Použita jsou i související klíčová slova `break` a `continue`.

Kapitola 6 vysvětluje pojem pole, základní operace s poli a směřuje výklad k používání funkcí.

Kapitola 7 ukazuje používání funkcí (tedy podprogramů) za účelem přehlednější tvorby programů v jazyce C++. Je vysvětleno předávání parametrů hodnotou a návratová hodnota. Poté je vytvořena skupina funkcí pro práci s poli a jsou uvedeny základní knihovní funkce (tedy již hotové funkce). Závěr je věnován otázce globálních a lokálních proměnných.

Kapitoly 8 a 9 jsou věnovány ukazatelům. Jsou vysvětleny pojmy reference a dereference, dynamická alokace paměti (operátory `new` a `delete`), předávání parametrů přes ukazatel a předávání parametrů odkazem. Jsou také zařazeny pokročilejší záležitosti, jako jsou ukazatelová aritmetika a souvislost ukazatele s polem.

Kapitoly 10 a 11 se zaměřují na znaky a řetězce. Je vysvětlen způsob reprezentace znaků a řetězců, jsou předvedeny základní knihovní funkce pro práci s těmito typy dat společně se vstupně/výstupními operacemi.

Kapitola 12 popisuje odvozené datové typy. Kromě úvodního rozdělení datových typů jsou postupně probírány nové datové typy: výčet, struktura, sjednocení a bitové pole. Datový typ třída je popsán samostatně až v kapitole 16.

Kapitola 13 je zařazena jako shrnutí operátorů. Doplněny jsou informace k dalším operátorům, jako jsou ternární operátor, bitové operátory, operátory přiřazení, operátor čárka a na závěr je uvedena přehledová tabulka priority a asociativity všech operátorů.

Kapitola 14 představuje modulární programování, tedy tvorbu programů založenou na více zdrojových souborech (používá se pro tvorbu rozsáhlejších programů). Souběžně s tím jsou vysvětleny direktivy překladu a paměťové třídy proměnných.

Kapitola 15 vysvětluje pojmy přetížení funkcí a implicitní parametry funkcí.

Kapitola 16 je úvodem do objektově orientovaného programování (OOP). Vysvětluje jeho výhody a základní rysy na postupně budovaném příkladu třídy `TClovek`. Jsou tak postupně vysvětleny pojmy: atribut, metoda, zapouzdření, přístupové úrovně, konstruktor, destruktor. Rovněž se seznámíme s pokročilejšími termíny, jako je mělká a hluboká kopie nebo statické a konstantní členy třídy. Na závěr je pak vysvětlen pojem dědičnost.

Kapitola 17 ukazuje možnosti přetěžování operátorů (možnost definovat vlastní význam operace pro danou třídu) a použití výjimek (řeší havarijní stavy programu).

Kapitola 18 popisuje další možnosti proudové knihovny a ukazuje použití proudů pro ovládání souborů. Kromě textových souborů je ukázána i práce s binárními soubory.

Kapitola 19 je věnována třídě `string`, která zjednodušuje práci s řetězci.

Kapitola 20 se zaměřuje na „napojení" programu na zbytek operačního systému. Řeší parametry příkazové řádky programu a návratovou hodnotu programu. Vše je předvedeno na příkladu programu pro kopírování souborů. Příklad používá pokročilejší práci se soubory včetně datového bufferu.

Dále jsou zařazeny dvě přílohy. Příloha A vysvětluje číselné soustavy a reprezentaci čísel v počítači. Jedná se o vysvětlení základních a násobných jednotek informace, převody mezi číselnými soustavami, reprezentaci celých čísel bez znaménka a se znaménkem a reprezentaci čísel v plovoucí řádové čárce. Dále jsou vysvětleny základní logické operace. Příloha B je věnována stručnému popisu vývojového prostředí Dev-C++ (popis položek nabídky) a také ukazuje ladění programu na krátkém příkladu.

Zpětná vazba od čtenářů

Nakladatelství a vydavatelství Computer Press, které pro vás tuto knihu připravilo, stojí o zpětnou vazbu a bude na vaše podněty a dotazy reagovat. Můžete se obrátit na následující adresy:

Computer Press
Albatros Media a.s., pobočka Brno
IBC
Příkop 4
602 00 Brno

nebo

sefredaktor.pc@albatrosmedia.cz

Computer Press neposkytuje rady ani jakýkoli servis pro aplikace třetích stran. Pokud budete mít dotaz k programu, obraťte se prosím na jeho tvůrce.

Zdrojové kódy ke knize

Z adresy *http://knihy.cpress.cz/K2237* si po klepnutí na odkaz Soubory ke stažení můžete přímo stáhnout archiv s ukázkovými kódy.

Errata

Přestože jsme udělali maximum pro to, abychom zajistili přesnost a správnost obsahu, chybám se úplně vyhnout nelze. Pokud v některé z našich knih najdete chybu, ať už chybu v textu nebo v kódu, budeme rádi, pokud nám ji oznámíte. Ostatní uživatele tak můžete ušetřit frustrace a pomoci nám zlepšit následující vydání této knihy.

Veškerá existující errata zobrazíte na adrese *http://knihy.cpress.cz/K2237* po klepnutí na odkaz Soubory ke stažení.

KAPITOLA 1

Úvod do programování v jazyce C++

V této kapitole:
- Základní pojmy
- IDE – integrované vývojové prostředí
- První program
- Pár zajímavostí
- Rozdělení základních datových typů

V této úvodní kapitole se seznámíme s úplnými základy programování, které jsou společné nejen pro programovací jazyk C++, ale pro všechny programovací jazyky. Budou vysvětleny základní pojmy (termíny), které budeme v dalším textu používat. Vzhledem k obsáhlosti celé problematiky budeme výklad pojmů „dávkovat" a s dalšími (složitějšími) termíny se budeme seznamovat postupně v následujících kapitolách.

Základní pojmy

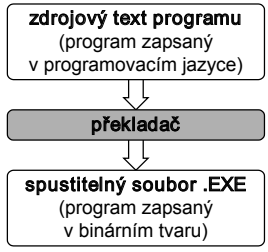
Než začneme psát jakýkoli program, musíme si provést rozbor řešeného problému. Pokud tuto fázi vynecháme („To je přeci otrava – přemýšlet, lepší je začít psát program!"), může se nám lehce stát, že tvorbou programu strávíme mnohem více času, než jsme původně předpokládali.

Následně pak dojdeme k tomu, jaké činnosti má počítač provádět a v jakém pořadí. Přesný návod, jak vyřešit daný typ úlohy, se nazývá **algoritmus**.

Pro zápis algoritmu do počítače používáme speciální **programovací jazyk**. Programovací jazyk musí být sestaven tak, aby bylo možno snadno vyjadřovat algoritmy a aby byl blízký lidskému uvažování.

Počítačový **program** vznikne tak, že algoritmus vyjádřený v programovacím jazyce převedeme do tvaru spustitelného na počítači. Přechod od algoritmu zapsaného v programovacím jazyce k programu spustitelnému na počítači se nazývá **překlad** a provádí jej speciální program označovaný jako **překladač** (kompilátor). Překladem se zdrojový text (je zapsaný v programovacím jazyce) převede do tzv. binární formy, což je vlastně sled instrukcí, které bude provádět procesor počítače.

Činnost, kterou počítač provádí při vykonávání programu, označujeme jako **proces**. Označení proces je na místě, přetváří vstupní údaje na výstupní.



Obrázek 1.1. Průběh překladu (zjednodušeno)

Pro správný zápis programu pomocí programovacího jazyka musíme dodržet syntaxi (věcnou správnost) a sémantiku (význam konstrukcí):

- **Syntaxe** popisuje, z čeho se může skládat zápis programu. Určuje například tzv. *klíčová slova*. Nemůžeme tedy použít slova, která daný programovací jazyk nezná. Syntaktická kontrola pak znamená, že se kontroluje dodržení korektnosti zdrojového textu (tedy „dodržení pravopisu").
- **Sémantika** přiřazuje jednotlivým konstrukcím jazyka význam. Je jistě možné napsat syntakticky správný program, který však nemá správný smysl.

Syntaxe + sémantika = správně fungující program

Proměnné a konstanty

Předměty, se kterými program pracuje (čte jejich hodnoty nebo je mění), nazýváme **data**. Může se jednat o čísla, znaky apod.

Většina dat obvykle během provádění programu mění svůj obsah, proto je označujeme jako **proměnné**. Data, jejichž hodnota se v průběhu provádění programu nemění, označujeme jako **konstanty**.

Typy příkazů

Zápis programu v programovacím jazyce se skládá z popisu použitých dat (tzv. deklarace) a z jednotlivých příkazů. Nejčastějšími variantami příkazů jsou:

- **deklarace proměnné** – nahlášení datového typu a názvu proměnné, kterou budeme používat (překladač pak může sledovat, jestli s proměnnými provádíme správné operace, a kontroluje, zda má dost paměti),
- **příkaz vstupu** – příkaz, který zajistí načtení dat například z klávesnice nebo jiného vstupního zařízení (například ze souboru),

- **příkaz výstupu** – příkaz, který vypíše data například na obrazovce nebo je uloží do jiného výstupního zařízení (například do souboru),
- **přiřazovací příkaz** – příkaz, který přiřadí do proměnné novou hodnotu,
- **větvení** – příkaz, který podle určité podmínky rozdělí další postup do více cest,
- **volání podprogramu** – příkaz, který provede dříve vytvořenou část programu; takto můžeme například provést výpočet funkce sin(x), pokud je tento podprogram v daném jazyce k dispozici; rovněž můžeme vytvářet vlastní podprogramy (funkce).

IDE – integrované vývojové prostředí

V souvislosti s programováním se zkratka IDE objevuje s anglickým souslovím Integrated Development Environment. Tedy doslovně přeloženo jako „integrované vývojové prostředí".

IDE znamená pro programátora komfort, kdy v rámci jediného programu (přesněji aplikace) zapíše zdrojový text, nastaví (pokud je to nutné) parametry překladu, provede překlad, testuje a ladí činnost hotového programu.

Existuje řada překladačů jazyka C++, které však vyžadují další znalosti uživatele, což je pro začátečníky nevýhodné. Totiž, pokud nemáme k dispozici IDE, musíme napsat zdrojový text například v poznámkovém bloku a poté spustit překlad pomocí příkazové řádky obvykle s použitím speciálního dávkového souboru. Možnosti ladění jsou v takovém případě velmi omezené.

IDE Dev-C++

Vývojových prostředí existuje celá řada. K nejznámějším patří **Microsoft Visual Studio** od Microsoftu nebo **C++ Builder** od firmy Embarcadero (původně Borland).

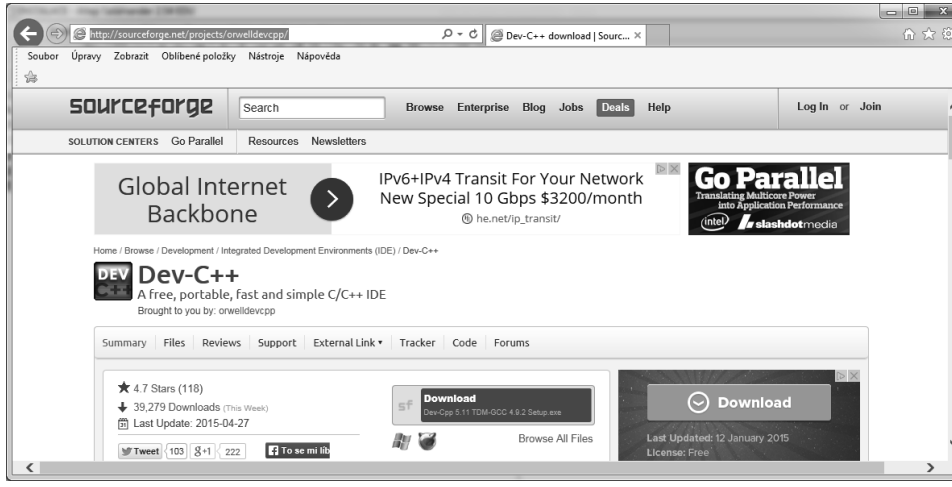
My jsme pro tuto knihu vybrali vývojové prostředí **Dev-C++**, které je vyvíjeno na základě licence GNU pod hlavičkou Bloodshed Software. Důvody byly: dobrá znalost tohoto prostředí, časově neomezená licence pro nekomerční použití, nízké nároky při instalaci a částečná lokalizace prostředí do češtiny (většina příkazů hlavní nabídky je v češtině, ale například chyby překladu jsou hlášeny v angličtině).

Vývojové prostředí Dev-C++ lze stáhnout ze stránek *http://www.bloodshed.net/dev/devcpp.html*.

Instalační soubor verze 5.9.2 z února 2015 měl velikost okolo 45 MB (tedy mnohem méně než ostatní výše uváděné produkty).

Stažení a instalace Dev-C++

Odkaz pro stažení instalace Dev-C++ je možné najít na výše uvedených stránkách (*http://sourceforge.net/projects/orwelldevcpp/*) nebo lze použít přímý odkaz (jak také ukazuje obrázek 1.2).



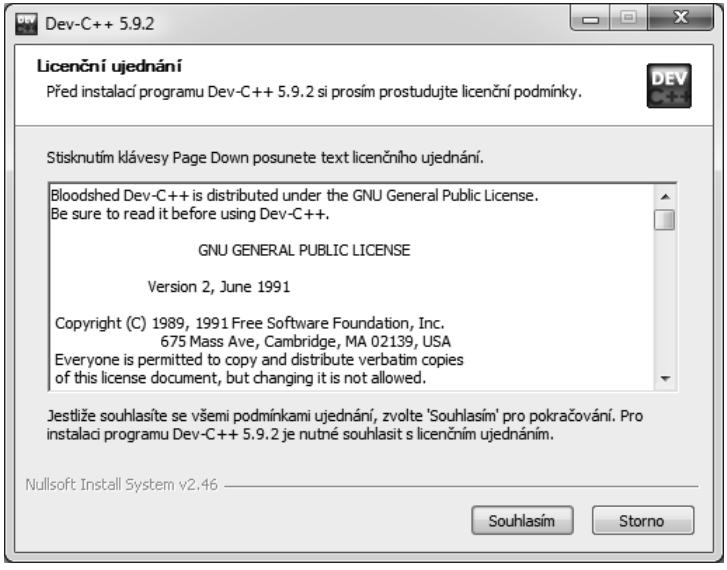
Obrázek 1.2. Stažení instalačního programu ze stránek SourceForge

Instalační soubor stáhneme pomocí tlačítka **Download**.

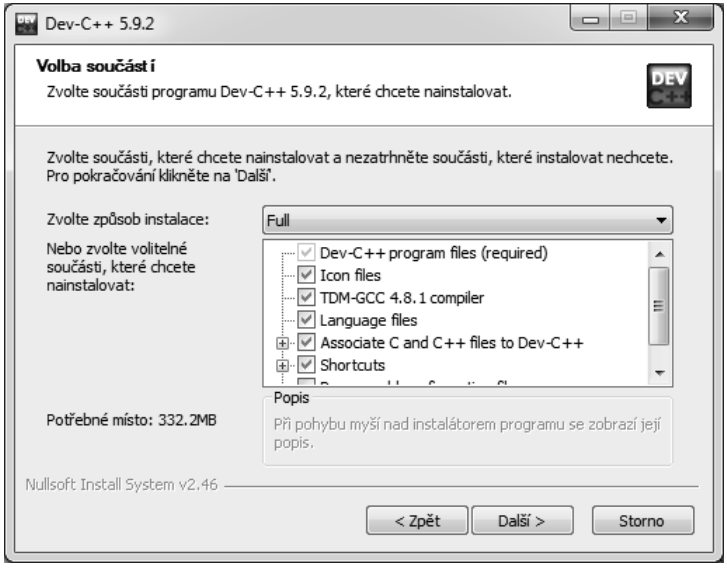
Spustíme instalaci pomocí souboru s názvem (název může být upraven při stažení novější verze): **Dev-Cpp 5.9.2 TDM-GCC 4.8.1 Setup.exe**. Nejdříve je provedena dekomprese a následně se zobrazí dialog pro volbu jazyka instalace dle obrázku 1.3. Vybereme **Cesky** a potvrdíme stiskem tlačítka **OK**.

Obrázek 1.3. Volba jazyka instalace

Následuje dialog s licenčním ujednáním dle obrázku 1.4, který je nutné potvrdit stiskem tlačítka **Souhlasím**.



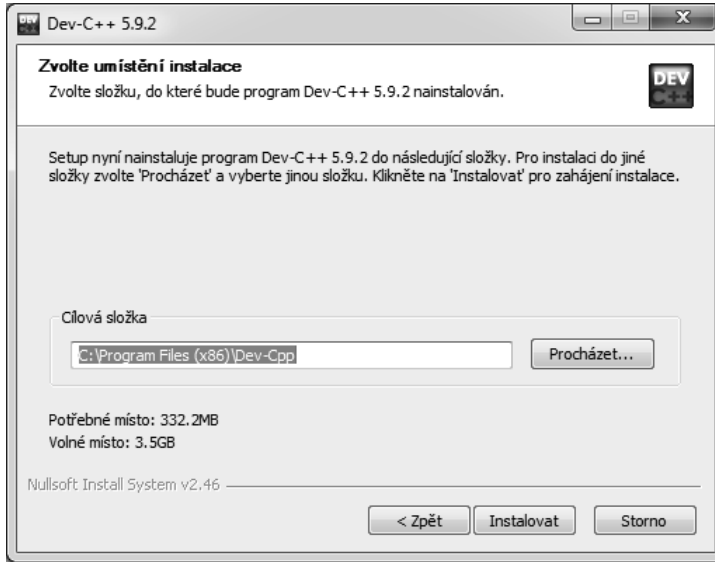
Obrázek 1.4. Dialog s licenčním ujednáním



Obrázek 1.5. Volba součástí pro instalaci

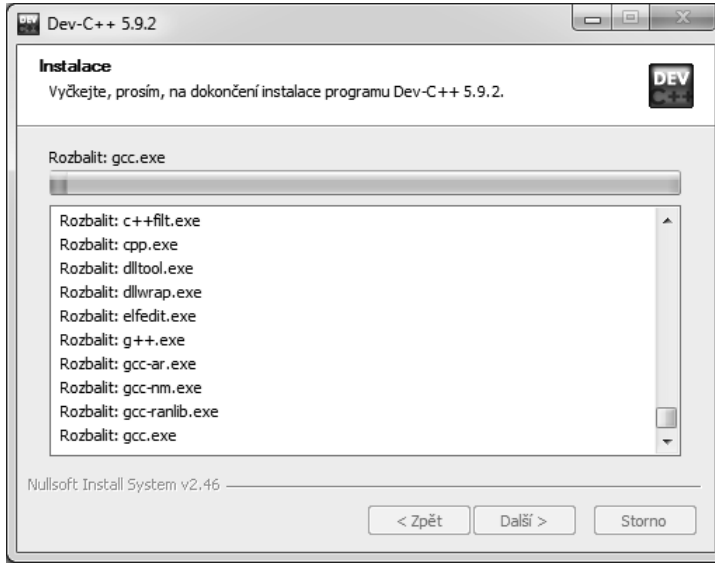
Následuje dotaz na výběr součástí pro instalaci dle obrázku 1.5. Zde kromě jiného dochází k asociaci s koncovkami souborů, které se používají v jazyce C++. Doporučujeme ponechat výchozí nastavení (vše vybrané) a poté pokračovat stiskem tlačítka **Další**. V této verzi je pro instalaci třeba asi 330 MB diskového prostoru.

Nakonec je třeba vybrat adresář, do kterého bude instalace provedena, viz obrázek 1.6. Doporučujeme ponechat výchozí volbu a pokračovat stiskem tlačítka **Instalovat**.



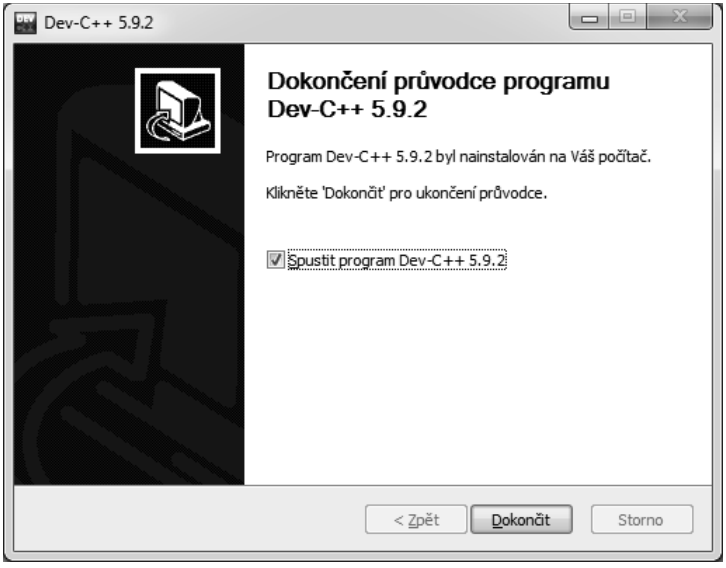
Obrázek 1.6. Volba adresáře pro provedení instalace

Poté se již rozbíhá instalace, jak dokumentuje obrázek 1.7.



Obrázek 1.7. Průběh vlastní instalace

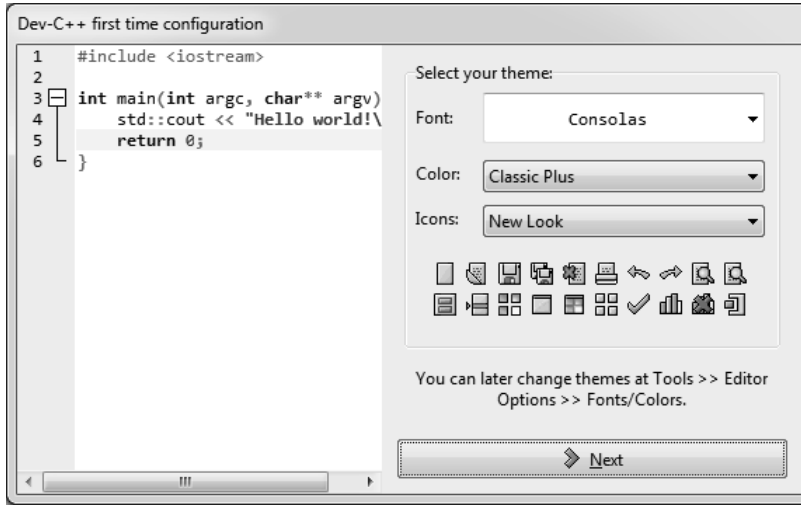
Na závěr se zobrazí dialog dle obrázku 1.8, který umožňuje přímé spuštění vývojového prostředí. Volbu **Spustit program Dev-C++** necháme aktivní a stiskneme tlačítko **Dokončit**.



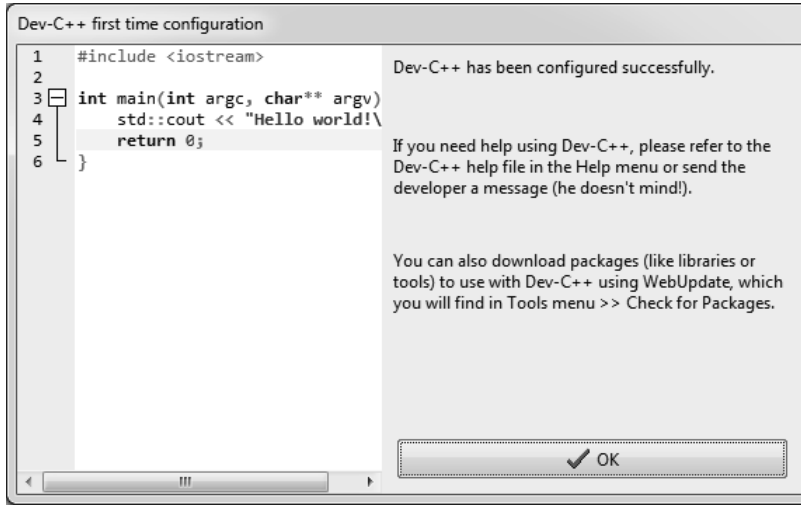
Obrázek 1.8. Dokončení instalace

Následně se objeví tři velmi podobné dialogy dle obrázků 1.9 až 1.11, které slouží pro konfiguraci prostředí. Jedná se o možnost změny jazyka instalace, schématu a nástrojů. Poslední dialog potvrzuje dokončení konfigurace.

Obrázek 1.9. Volba jazyka



Obrázek 1.10. Volba schématu



Obrázek 1.11. Dokončení konfigurace

Následně již nabíhá prostředí.

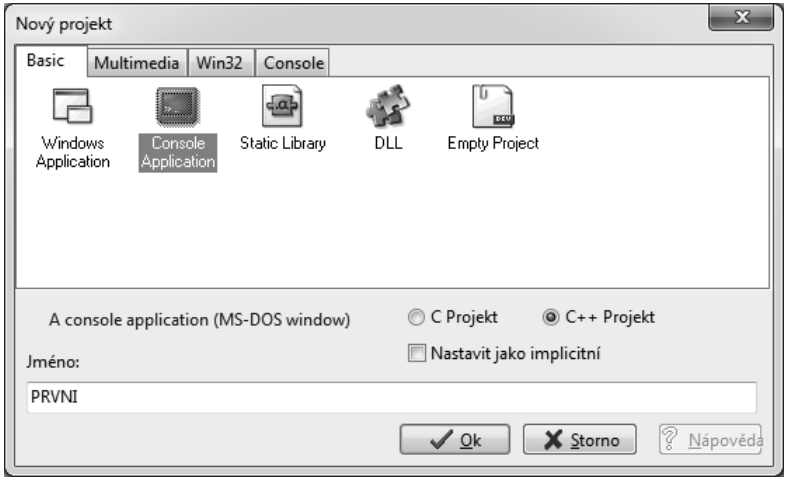
První program

Nyní tedy můžeme vytvořit svůj první program v jazyce C++.

Spustíme vývojové prostředí Dev-C++ (pokud již není spuštěno) a pomocí příkazu nabídky **Soubor → Nový → Projekt** (viz obrázek 1.12) vyvoláme dialog volby typu projektu dle obrázku 1.13.

Obrázek 1.12. Založení nového projektu

V dialogu volby typu projektu dle obrázku 1.13 se přepneme na kartu **Basic** a jako typ projektu zvolíme **Console Application**. Konzolová aplikace je typ aplikace pro operační systém Windows, která pracuje v textovém režimu (jako MS-DOS okno). Dále vybereme projekt typu **C++ Projekt**. V textovém poli **Jméno** zadáme název projektu **PRVNI**. Pokračujeme stiskem tlačítka **Ok**.

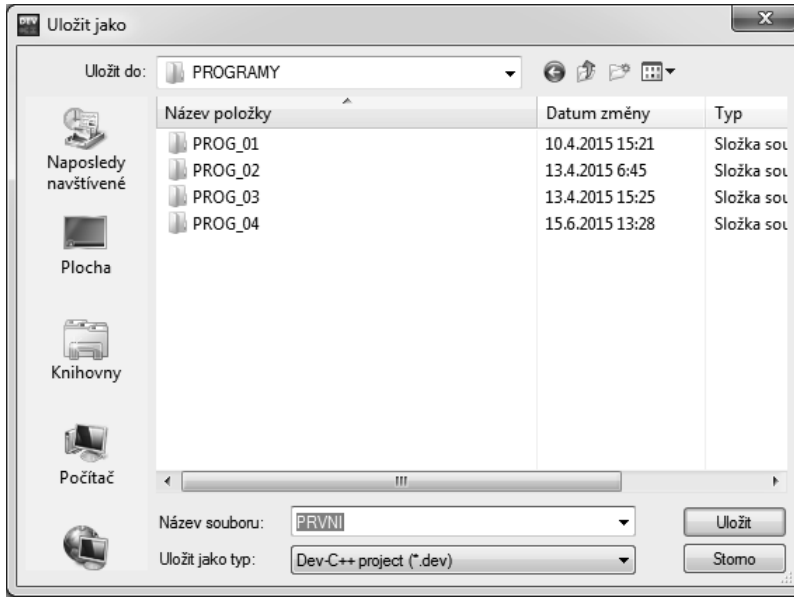


Obrázek 1.13. Volba typu a názvu projektu

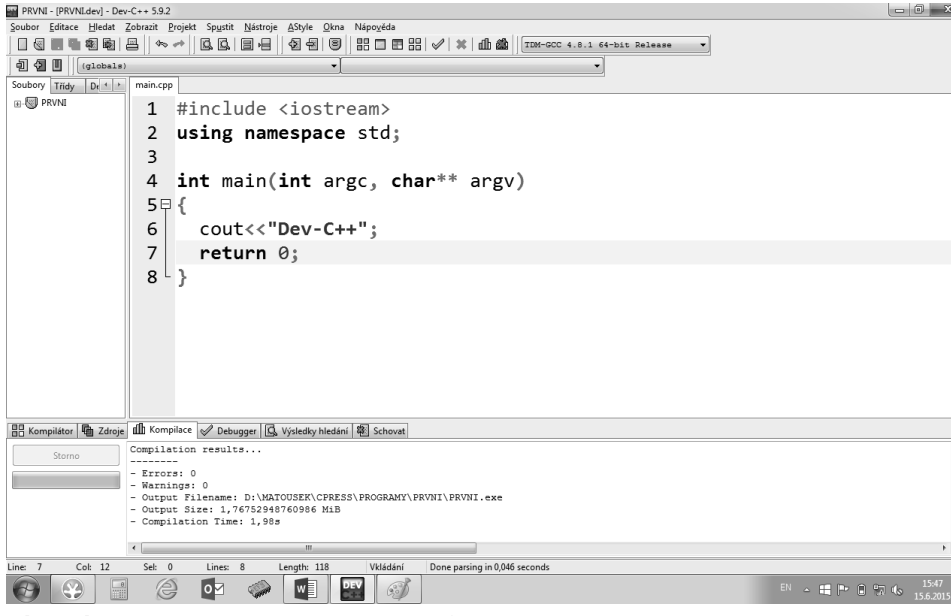
Následně se zobrazí dialog dle obrázku 1.14, což je klasický dialog volby adresáře pro uložení projektu. Zvolíme příslušnou složku a pokračujeme tlačítkem **Uložit**.

Zápis našeho prvního programu dokumentuje obrázek 1.15. Soubor byl standardně pojmenován jako **MAIN.CPP** (main = hlavní program, CPP značí zkratku pro „cé plus plus“). Všimněte si, že textový editor rozlišuje syntaxi pomocí barev a stylů písma. Příkazy preprocesoru (povíme si o nich podrobněji v průběhu našeho seznamování s jazykem C++) se značí zeleně,

tučně jsou označena klíčová slova jazyka C++, červeně speciální znaky, modře textové řetězce a fialově zápisy čísel.



Obrázek 1.14. Volba adresáře pro uložení souborů projektu



Obrázek 1.15. Zápis programu v textovém editoru