Siddhartha Rao

PREVOD
DEVETOG IZDANJA

POKRIVA
C++20 i
C++23

NAUČITE SAMOSTALNO C++

jednalekcijaDNEVNO

kompjuter biblioteka

C++

PREVOD DEVETOG IZDANJA

jednalekcijaDNEVNO

SKORO 300 UZORAKA KODA ZA KOMPAJLIRANJE

Učeći jednu lekciju dnevno, možete da steknete sve veštine koje su vam potrebne za početak programiranja u C++ jeziku. Ovaj kompletan vodič će vam pomoći da brzo ovladate osnovama objektno-orijentisanog programiranja, kao i naprednim funkcijama i konceptima jezika C++. Potpuno ažurirana za standard C++20, ova praktična knjiga je dizajnirana da vam pomogne da napišete C++ kod koji je brži, jednostavniji i pouzdaniji.

- Naučite osnove C++ jezika i objektno-orijentisanog programiranja
- Koristite Standard Template Library (STL) da brzo razvijete moćnije i pouzdanije aplikacije
- Naučite moderne funkcije C++20 verzije, kao što su koncepti, opsezi, prikazi, adapteri i moduli
- Saznajte koje bi dokazane akcije trebalo da koristite, a koje ne, da biste iskoristili najbolju praksu i izbegli zamke, od prvog dana
- Testirajte svoje znanje i stručnost vežbama nakon svake lekcije
- Naučite da koristite skoro 300 uzoraka koda za kompajliranje koji su dostupni za preuzimanje besplatno i koji su detaljno objašnjeni u knjizi

Siddhartha Rao je potpredsednik u SAP SE-u, zadužen za Security Response za vodeći svetski trgovinski softver. Evolucija C++-a ubedila ga je da je sada moguće programiranje bržih, jednostavnijih i moćnijih aplikacija nego ikada do sada.

Učite kada imate vremena, tempom koji vama odgovara

- **Nije potrebno iskustvo u programiranju: ubrzo ćete umeti da pišete dobro organizovane, efikasne C++ programe!**
- **Ovladajte objektno-orijentisanim konceptima kao što su klase, nasleđivanje, polimorfizam, enkapsulacija i apstrakcija**
- **Kreirajte pouzdane programe bogate funkcijama pomoću STL klasa, kontejnera i algoritama**
- **Pojednostavite svoj kod pomoću automatskog utvrđivanja tipa i drugih funkcija**
- **Programirajte objekte funkcija pomoću modernih C++ Lambda izraza**
- **Ubrzajte učenje korišćenjem skoro 300 uzoraka koda objašnjenih u knjizi**
- **Iskoristite nove C++20 koncepte, opsege, prikaze, adaptere i module**
- **Pregledajte poboljšanja koja se očekuju u verziji C++23**

Kategorija: Programiranje
Pokriva: C++23, C++20 C++17, C++14, C++11
Nivo: Početni–Srednji

ISBN: 978-86-7310-578-9

kompjuter biblioteka
osnovana 1986.
www.kombib.rs

Pearson

Siddhartha Rao

NAUČITE SAMOSTALNO

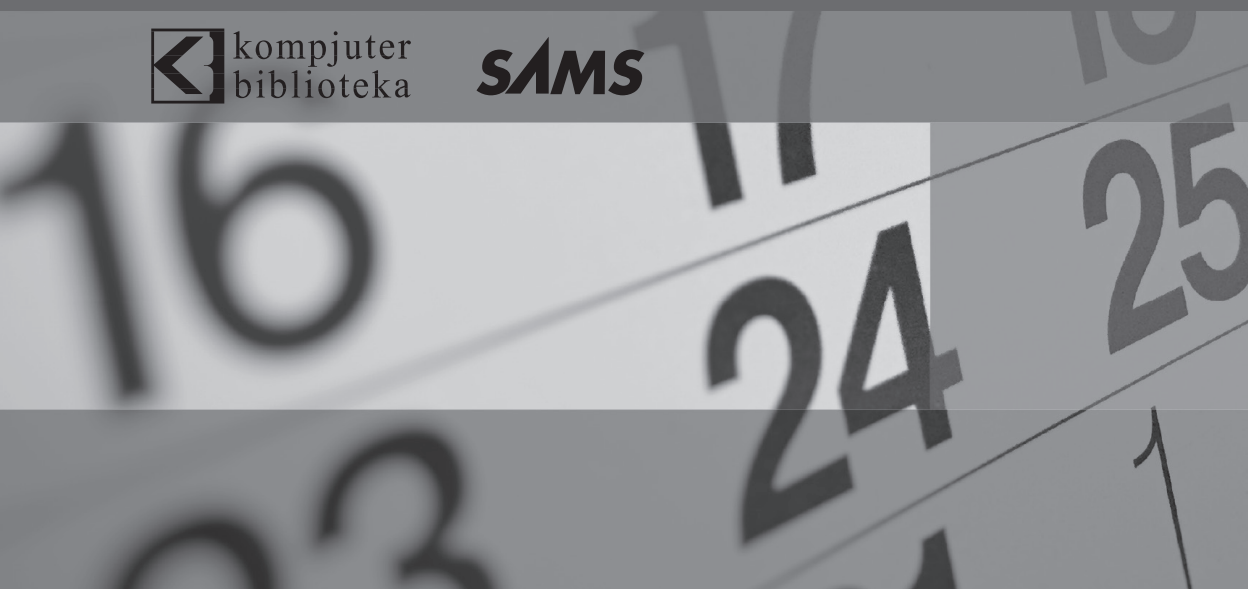
C++

PREVOD DEVETOG IZDANJA

jednalekcijaDNEVNO

kompjuter biblioteka

SAMS



Izdavač:

kompjuter biblioteka

Obalskih radnika 15, Beograd

Tel: 011/2520272

e-mail: kombib@gmail.com

internet: www.kombib.rs

Urednik: Mihailo J. Šolajić

Za izdavača, direktor:
Mihailo J. Šolajić

Autor: Siddhartha Rao

Prevod: Slavica Prudkov

Lektura: Nemanja Lukić

Slog : Zvonko Aleksić

Znak Kompjuter biblioteke:
Miloš Milosavljević

Štampa: „Pekograf“, Zemun

Tiraž: 500

Godina izdanja: 2022.

Broj knjige: 555

Izdanje: Prvod IX izdanja

ISBN: 978-86-7310-578-9

Sams Teach Yourself C++
in One Hour a Day

by Siddhartha Rao
ISBN: 978-0-13-733468-1

O autoru

Siddhartha Rao je potpredsednik SAP SE-a, zadužen za bezbednost proizvoda. SAP SE je najpouzdaniji svetski dobavljač poslovnog softvera i cloud servisa. Softverski inženjer u srcu, Siddhartha je ubeđen da je brzi razvoj programskog jezika C++ pokrenuo ovo doba mašinskog učenja i veštačke inteligencije. Funkcije predstavljene u verziji C++20 omogućavaju da programirate jednostavnije, ali moćnije aplikacije nego ikada ranije. Prilikom pisanja ove knjige Siddharhta se pobrinuo da skoro 300 primera kompajliranja koda bude propraćeno detaljnim analizama načina na koji funkcionišu.

Siddhartha bi želeo da čuje vaše komentare, recenzije i povratne informacije!

Kratak sadržaj

Sadržaj

LEKCIJA 2

Anatomija C++ programa

LEKCIJA 3

Upotreba promenljivih i deklarisanje konstanti ... 27

LEKCIJA 22

LEKCIJA 23

LEKCIJA 24

Prilagodljivi kontejneri: stack i queue 615

LEKCIJA 25

Upotreba bit indikatora pomoću biblioteke STL 633

DEO V: NAPREDNI C++ KONCEPTI

LEKCIJA 26

LEKCIJA 27

LEKCIJA 28

LEKCIJA 29

LEKCIJA 30

DODATAK E

Uvod

Brz napredak programskog jezika C++ pokrenuo je ovo doba mašinskog učenja i veštačke inteligencije. Možete da upotrebite C++20 da biste programirali jednostavnije, ali moćnije aplikacije nego ikada ranije. U ovoj knjizi, u kojoj detaljno analiziramo skoro 300 primera kompajliranja koda, predstavićemo važne funkcije C++20 jezika i opisati njihov interni rad.

Osim što ćemo objasniti osnove C++ programskog jezika, učićete o funkcijama jezika koje su neophodne u profesionalnom C++ programiranju. Između ostalog, učićete o osnovama objektno-orijentisanog programiranja, osnovnim funkcijama i algoritmima Standard Template Library biblioteke i o konceptima, rasponima, prikazima i adapterima verzije C++20. Bez obzira na to da li ste početnik ili profesionalni programer, ova knjiga će vam biti veoma korisna.

Kome je ova knjiga namenjena?

Knjiga počinje opisom osnova C++ jezika. Da biste razumeli kako sve funkcioniše, potrebne su vam želja za učenjem ovog programskog jezika i radoznalost. Postojeće znanje C++ programiranja može biti prednost, ali nije preduslov. Ovo je, takođe, knjiga koju možete da koristite kao referencu ako već znate C++, ali želite da naučite i poboljšanja koja su dodata u taj jezik. Ako ste profesionalni programer, deo III, „Učenje o biblioteci Standard Template Library (STL)“, deo IV, „Lambda izrazi i STL algoritmi“ i deo V, „Napredni C++ koncepti“ pomoći će vam da kreirate bolje, praktičnije C++ aplikacije.

> **NAPOMENA** Posetite veb sajt izdavača i registrujte ovu knjigu na adresi informit.com/register za lakši pristup ažuriranjima, fajlovima za preuzimanje i spiskovima grešaka dostupnim za ovu knjigu.

Organizacija ove knjige

U zavisnosti od aktuelnog nivoa stručnosti u C++ jeziku, možete da izaberete odeljak od kojeg želite da počnete učenje. Ova knjiga je podeljena u pet delova:

- Deo I, „*Osnove*“- počinje učenjem pisanja jednostavnih C++ aplikacija. Upoznaćete ključne reči koje se najčešće viđaju u C++ kodu.
- Deo II, „*Osnove objektno-orijentisanog C++ programiranja*“ - učićete o principima objektno-orijentisanog programiranja, kao što sukapsuliranje, apstrakcije, nasleđivanja i polimorfizam. O važnosti programiranja konstruktora copy i upotrebe konstruktora move za optimizaciju performansi učićete u lekciji 9, „Klase i objekti“, a o trosmernom operatoru poređenja, koji nazivamo i spaceship operator zbog njegovog oblika `(<=>)`, učićete u lekciji 12, „Tipovi operatora i preklapanje operatora“. U lekciji 14, „*Predstavljanje makroa i šablona*“, naučićete da pišete moćan generički C++ kod.
- Deo III, „*Biblioteka Standard Template Library (STL)*“ - pomoći će vam da napišete efikasan i praktičan C++ kod, korišćenjem STL klasa i kontejnera. Na primer, u ovom delu saznaćete kako std::string čini operacije spajanja znakovnog niza bezbednim i jednostavnim. U lekcijama 17, “*STL klase dinamičkog niza*” i 18, “*STL list i forward_list*” naučićete da koristite standardizovane dinamičke nizove i povezane liste tako da ne morate da programirate sopstvene. U lekciji 20, “*STL mapa i multimapa*” upoznaćete upotrebu parova ključ/vrednost koji su sačuvani u asocijativnim kontejnerima, kao što su std::map i std::multimap.
- Deo IV, „*Lambda izrazi i STL algoritmi*“ - počinje objašnjenjem kako da programirate objekte funkcije. U lekciji 22, „*Lambda izrazi*“, predstavljena je implementacija neimenovanih objekata funkcije. U lekciji 23 „*STL algoritmi*” učićete o različitim algoritmima koji vam pomažu pri izvršenju operacija u kontejnerima, kao što je pronalaženje, uklanjanje i sortiranje elemenata.
- Deo V, „*Napredni C++ koncepti*“ - počinje objašnjenjem pametnih pokazivača i rukovanja izuzecima, koji značajno doprinose stabilnosti i kvalitetu aplikacije. Objašnjene su ključne funkcije predstavljene u verziji C++20. U lekciji 29, „*C++20 koncepti, opsezi, prikazi i adapteri*” naučićete da potvrdite parametre šablona upotrebom koncepata i strukturnih prikaza elemenata u opsegu upotrebom adaptera. U lekciji 31, „*C++20 moduli i C++23*” učićete o modulima koji će zameniti tradicionalne fajlove zaglavlja, a lekciju završavamo predstavljanjem poboljšanja jezika za koje se očekuje da će biti izdati u sledećoj verziji standardnog C++ jezika, u verziji C++23..

Konvencije upotrebljene u ovoj knjizi

Unutar lekcija pronaći ćete sledeće elemente koji obezbeđuju dodatne informacije:

NAPOMENA Ovi okviri obezbeđuju dodatne informacije u vezi onoga što čitate.

PAŽNJA Ovi okviri upozoravaju na probleme ili sporedne efekte koji se mogu desiti u posebnim situacijama.

SAVET Ovi okviri predstavljaju najbolju praksu u pisanju C++ programa.

URADITE	NE RADITE
Upotrebite „Uradite/Ne radite“ okvire da biste pronašli rezime osnovnih principa lekcije.	Nemojte da previdite korisne informacije koje se nalaze u ovim okvirima.

U ovoj knjizi upotrebljeni su različiti fontovi da biste razlikovali kod i običan tekst. Kod, komande i termini koji se odnose na programiranje prikazani su računarskim fontom.

Primeri koda za ovu knjigu

Primeri koda za ovu knjigu dostupni su besplatno na stranici https://github.com/learncppnow/9E.git.

Želimo da vas čujemo!

Kao čitaoci ove knjige, vi ste nam najvažniji kritičari i komentatori. Mi cenimo vaše mišljenje i želimo da znamo šta smo, po vašim ocenama, uradili dobro, šta smo mogli da uradimo bolje, iz koje oblasti biste želeli da publikujemo sledeće knjige i željni smo da čujemo vaše mudre reči koje želite da nam pošaljete.

Očekujemo vaše komentare. Možete da nam pošaljete e-mail i napišete šta vam se dopalo ili nije dopalo u našoj knjizi i šta možemo da uradimo da bismo poboljšali naše knjige.

Imajte na umu da ne možemo da vam pomognemo u vezi sa tehničkim problemima koji su u vezi sa temom ove knjige.

Kada nam pišete, navedite naslov knjige i ime autora, kao i vaše ime i e-mail adresu. Mi ćemo pažljivo pregledati vaše komentare i podeliti ih sa autorom i urednicima koji su radili na knjizi.

Email: community@informit.com

LEKCIJA 1

Početak rada

Dobrodošli u Sams Teach Yourself C++ za sat vremena dnevno! Spremni ste da postanete vešt C++ programer.

U ovoj lekciji otkrićete:

- zašto je C++ standard u razvoju softvera
- kako da unesete, kompajlirate i povežete prvi radni C++ program
- šta je novo u verziji C++20

Kratka istorija C++ jezika

Svrha programskog jezika je da olakša potrošnju računarskih resursa. C++ nije nov - to je jezik koji je popularan i neprestano se razvija. U vreme pisanja ove knjige, najnovija verzija C++ jezika je C++20, koju je ratifikovao International Organization for Standardization (ISO), a izdata je u decembru 2020. godine. Prethodne izmene jezika su bile označene kao verzija C++17 (izdata 2017. godine), verzija C++14 (izdata 2014. godine) i verzija C++11 (izdata 2011. godine).

Povezanost sa C jezikom

Bjarne Stroustrup u Bell Labs-u je razvio C++ 1979. godine. C++ jezik je projektovan da bude naslednik jezika C, od koga se razlikuje po činjenici da je dizajniran da bude objektno-orijentisan jezik, koji implementira koncepte, kao što su nasleđivanje, apstrakcija, polimorfizam i kapsuliranje. C++ uključuje klase koje se koriste za skladištenje podataka člana i metoda člana. Ovi metodi člana su funkcije koje izvršavaju operacije na podacima člana. Ova organizacija pomaže programeru da modeluje podatke i akcije koje želi da izvrši na podacima. Popularni C++ kompajleri nastavili su da podržavaju i programiranje C jezikom i da pružaju jednostavnu integraciju za oba sveta, što je rezultiralo odličnom kompatibilnošću sa starijim verzijama.

NAPOMENA Poznavanje C programiranja nije preduslov za učenje jezika C++. Ako je vaš cilj da naučite objektno-orijentisan programski jezik, kao što je C++, nije potrebno da počinjete od učenja proceduralnog jezika, kao što je C, već možete početi direktno da koristite ovu knjigu.

Prednosti C++ jezika

C++ smatramo programskim jezikom srednjeg nivoa. To je svestran jezik i možete da ga koristite za programiranje aplikacija visokog nivoa koje ne treba da budu svesne specifičnosti hardvera na kom se pokreću. C++ takođe možete da koristite za programiranje biblioteka niskog nivoa, koje funkcionišu bliže hardveru, kao što su drajveri uređaja. Prema tome, C++ pruža optimalnu srednju putanju kao pomoć programerima da razviju kompleksne aplikacije koje ne zahtevaju nikakve kompromise u pogledu performansi ili upravljanja resursima.

Uprkos postojanju programskih jezika kao što su Java, C# i drugi jezici, C++ je i dalje veoma popularan i još uvek se razvija. Ovi drugi jezici su *interpretirani* izvršnom komponentom koja upravlja resursima za programera. Izvršenje olakšava programiranje i apstrahuje sistemske resurse do te mere da su jezici neprikladni za mnoga izračunavanja visoke performanse. Stoga, C++ ostaje izabrani jezik u slučajevima gde je potrebna potpuna kontrola nad upotrebom memorije i nad performansama. Slojevita arhitektura, gde veb server programiran C++ jezikom služi druge komponente programirane u jezicima HTML, Java, JavaScript ili .NET, je uobičajena. C++ je izabrani jezik za veštačku inteligenciju i mašinsko učenje, drajvere uređaja, baze podataka, operativne sisteme, servise, pa čak i za kompajlere i interpretere drugih programskih jezika.

Evolucija C++ standarda

Zbog svoje popularnosti C++ jezik je prihvaćen i usvojen na mnogim različitim platformama (operativnim sistemima) koje koriste sopstvene C++ kompajlere. Njegov razvoj je izazvao odstupanja specifična za kompajlere i, samim tim, probleme interoperabilnosti i probleme promene platforme. Stoga se pojavila potreba za standardizacijom jezika i proizvođačima kompajlera su obezbeđene standardne specifikacije jezika koje bi trebalo da koriste.

ISO Committee je 1998. godine ratifikovao prvu standardnu verziju jezika C++ ISO/IEC 14882:1998. Od tada je taj standard pretrpeo velike promene kojima je poboljšana upotrebljivost jezika i proširena podrška standardne biblioteke. Kao što je ranije pomenuto, u vreme pisanja ove knjige aktuelna ratifikovana verzija standarda je ISO/IEC 14882:2020, popularno nazvana C++20.

NAPOMENA Aktuelni standard možda nisu u potpunosti podržali svi popularni kompajleri. Stoga ćete u ovoj knjizi učiti o najnovijim dodacima jeziku C++20 i objasnićemo dobro podržane funkcije jezika koje vam pomažu u pisanju dobrih, funkcionalnih C++ aplikacija.

Ko koristi programe napisane C++ jezikom?

Lista aplikacija, operativnih sistema, interpretera, veb servisa, baza podataka i poslovnog softvera programiranih u jeziku C++ je veoma duga. Bez obzira na to šta ste i šta radite na računaru, postoji mogućnost da već koristite softver programiran u jeziku C++. Na primer, V8 JavaScript Engine od Google-a je programiran u jeziku C++. On je sastavni deo popularnih pretraživača i serverske tehnologije, kao što je Node.js.

C++ je izabrani jezik za istraživačke radove fizičara, matematičara i istraživača podataka. Uspon veštačke inteligencije upotrebom algoritama mašinskog učenja podstaknut je računarstvom visokih performansi koje omogućava C++.

Programiranje C++ aplikacije

Kada pokrenemo omiljen pretraživač ili aplikaciju za obradu reči, mi u stvari ukazujemo procesoru da pokrene izvršni fajl konkretnog programa. Izvršni fajl je završeni proizvod koji bi trebalo da izvrši ono što je programer želeo da postigne.

Koraci za izgradnju izvršnog fajla

Programiranje korišćenjem jezika C++ je prvi korak ka kreiranju izvršnog fajla koji se pokreće na operativnom sistemu. Osnovni koraci u kreiranju aplikacija u C++ jeziku su sledeći:

1. Unesite C++ kod pomoću editora teksta. Editor teksta je obično editor koda ili integrisano razvojno okruženje (IDE).
2. Kompajlirajte kod pomoću C++ *kompajlera* koji kreira verziju mašinskog jezika koji se nalazi u fajlovima objekta.
3. Povežite fajlove objekta pomoću *povezivača* da biste dobili izvršni fajl (na primer, .exe u Windowsu)

Kompajliranje je korak u kojem je kod u C++ jeziku, koji je sadržan u tekstualnim fajlovima i ima ekstenziju .cpp, konvertovan u kod bajtova koji procesor može da izvrši. Kompajler konvertuje po jedan fajl koda generisanjem fajla objekta koji ima ekstenziju .o ili .obj i ignoriše zavisnosti koda u drugom fajlu.

> **SAVET** Popularni kompajleri uključuju `g++` od GNU Projecta, `clang++` od LLVM-a i Microsoft Visual C++ (MSVC) kompajler. `g++` i `clang++` su uglavnom u upotrebi u Linux i macOS okruženjima, dok je MSVC izabrani kompajler za Windows.
>
> U vreme pisanja ovog teksta nijedan kompajler ne obećava punu podršku za C++20, ali `g++` i MSVC su bolji od ostalih.

Povezivač (eng. linker) je alat koji rešava istaknute zavisnosti između `.obj` fajlova. Prilikom uspešnog povezivanja alat kreira izvršni fajl koji programer može da izvrši ili distribuira.

Proces uspešnog kompajliranja i povezivanja nazivamo *izgradnja* izvršnog fajla.

> **SAVET** Upotreba online kompajlera koji je dostupan putem pretraživača može da bude najbrži način za početak editovanja, kompajliranja i izvršavanja jednostavnih C++ aplikacija. Potražite „online C++ compiler" i isprobajte opcije koje pronađete. Za bilo koji kompajler, prilikom kompajliranja novih funkcija jezika predstavljenih u ovoj knjizi, obratite pažnju da podržava C++20.

Analiziranje i ispravljanje grešaka

Aplikacije se retko kompajliraju i izvršavaju pri prvom pokretanju, kao što bi trebalo. Velika ili složena aplikacija programirana u bilo kom jeziku (uključujući i C++) treba da se pokrene više puta, što je deo testiranja i identifikovanja grešaka u kodu, koje nazivamo *programske greške*. Nakon što su programske greške ispravljene, izvršni fajl je ponovo izgrađen i proces testiranja se nastavlja. Stoga, pored tri koraka - programiranja, kompajliranja i povezivanja - razvoj softvera uključuje i korak koji nazivamo *otklanjanje grešaka*, u kojem programer analizira greške u kodu i ispravlja ih. Dobra razvojna okruženja obezbeđuju alatke i funkcije koje pomažu pri otklanjanju grešaka.

Integrisana razvojna okruženja

Mnogi programeri često koriste integrisano razvojno okruženje (IDE) u kojem su koraci programiranja, kompajliranja, povezivanja i otklanjanja grešaka integrisani unutar jedinstvenog korisničkog interfejsa.

Ako želite da programirate u C++ jeziku korišćenjem IDE-a, možete da instalirate jedan od mnogih besplatno dostupnih C++ IDE-a i kompajlera da biste započeli učenje. Neki od popularnih IDE-a su Eclipse i Code::Blocks za Linux sisteme, Xcode za macOS sisteme i Microsoft Visual Studio za Windows sisteme.

URADITE	NE RADITE
Snimite C++ fajlove korišćenjem ekstenzije `.cpp`.	Nemojte da upotrebite ekstenziju `.c` za C++ fajlove, zato što će neki kompajleri prevesti te fajlove kao C programe, umesto kao C++.
Za pisanje koda upotrebite jednostavan editor teksta, editor koda ili integrisano razvojno okruženje.	Za pisanje koda nemojte da koristite programe za obradu teksta, jer funkcije za automatsko ispravljanje i formatiranje teksta mogu da dovedu do grešaka kompajliranja.

Programiranje prve C++ aplikacije

Sada, kada poznajete alatke i korake, vreme je da programirate prvu C++ aplikaciju koja prati tradiciju i prikazuje „Hello World!“ na ekranu.

Ako programirate na Linux ili macOS sistemu, upotrebite editor teksta koji vam je poznat (ja sam upotrebio `gedit` na Ubuntu sistemu) za kreiranje fajla koji sadrži kod prikazan u programskom kodu 1.1. Zatim snimite fajl čiji se naziv završava ekstenzijom `.cpp` - na primer, `Hello.cpp`.

Ako koristite macOS sa Xcode IDE-om, pratite sledeće korake da biste kreirali novi C++ projekat:

1. Otvorite New Project Wizard pomoću opcije menija File, New, Project.
2. Selektujte Command Line Tool i kliknite Next.
3. Izaberite naziv proizvoda, kao što je Hello. Za jezik izaberite C++ i kliknite Next.
4. Zamenite automatski generisan sadržaj u fajlu main.cpp isečkom koda koji je prikazan u programskom kodu 1.1.

Ako koristite Windows sa Microsoft Visual Studio IDE-om, pratite sledeće korake da biste kreirali nov C++ projekat:

1. Otvorite New Project Wizard pomoću opcije menija File, New, Project.
2. Selektujte C++, za tip izaberite Console App i kliknite Next.
3. Dodelite naziv projektu, kao što je Hello. Kliknite Create.
4. Zamenite automatski generisan sadržaj u fajlu Hello.cpp isečkom koda koji je prikazan u programskom kodu 1.1.

PROGRAMSKI KOD 1.1 Hello.cpp, program Hello World

```
1: #include<iostream>
2:
3: int main()
4: {
5:     std::cout << "Hello World!" << std::endl;
6:     return 0;
7: }
```

Ova jednostavna aplikacija samo prikazuje „Hello World!“ na ekranu, korišćenjem linije koda `std::cout`. Linija koda `std::endl` daje instrukcije za `cout` da završi konkretnu liniju unosom novog reda, a aplikacija se zatvara i operativnom sistemu vraća `0`.

SAVET Možda vidite varijante linije 5 u programskom kodu 1.1, koja je sledeća:

```
5:      std::cout << "Hello World!\n";
```

Izlaz programa se neće promeniti. `"Hello World!\n"` uključuje nov red u formi `\n` i nije dodatno unet upotrebom linije `cout::endl`.

Neki primeri koda u ovoj knjizi koriste `'\n'` kao pomoć da se linija koda uklopi u liniju na štampanoj stranici.

NAPOMENA Da biste čitali program, može biti korisno da znate kako da izgovarate specijalne karaktere i ključne reči.

Na primer, `#include` možete da čitate kao „hash-include“. Ostale verzije su „sharp-include“ ili „pound-include“, u zavisnosti od toga odakle ste.

Slično tome, možete da čitate `std::cout` kao „standard-c-out“, a `endl` kao „end-line“.

PAŽNJA Važni su detalji, što znači da bi trebalo da kucate kod na potpuno isti način kao što je prikazano u programskom kodu. Kompajleri su striktni i ako slučajno na kraj iskaza unesete :, a potrebno je da unesete ;, možete da očekujete grešku kompajliranja i izveštaj o grešci! Iz istog razloga ne bi trebalo da koristite program za obradu teksta za editovanje koda.

SAVET Editori koda, kao što je Visual Studio Code, dostupni su besplatno za Linux, macOS i Windows sisteme. Ako ne koristite IDE, sledeća najbolja opcija je da upotrebite Visual Studio Code za editovanje koda.

Izgradnja i izvršavanje prve C++ aplikacije

Ako koristite g++ kompajler od GCC-a, otvorite terminal, kliknite na direktorijum koji sadrži hello.cpp i otvorite kompajler i program za povezivanje, korišćenjem komandne linije:

```
g++ -o hello Hello.cpp -std=c++20
```

Alternativno, ako koristite clang++ kompajler na macOS sistemu, pratite prethodno navedene korake i izvršite sledeću komandnu liniju:

```
clang++ -o hello Hello.cpp
```

Ove komande daju instrukcije kompajleru g++ i clang++ da kreiraju izvršni fajl pod nazivom hello, kompajliranjem C++ fajla Hello.cpp.

OMOGUĆITE C++20 FUNKCIJE KADA KORISTITE KOMPAJLER G++ ILI CLANG++

Da biste kompajlirali kod specifičan za verziju C++20 korišćenjem kompajlera g++ ili clang++ dodajte parametar -std=c++20 u komandnu liniju:

```
g++ -o hello Hello.cpp -std=c++20
```

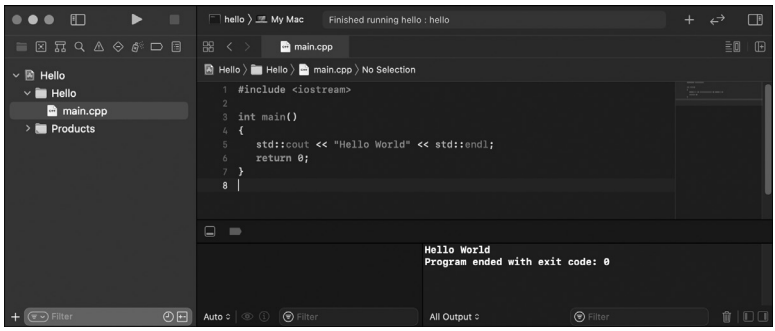
Ili

```
clang++ -o hello Hello.cpp -std=c++20
```

Parametar će vam biti potreban kada kompajlirate kod koji koristi funkcije verzije C++20.

Ako koristite macOS sistem sa Xcode IDE-om, možete da izgradite i pokrenete aplikaciju tako što ćete selektovati opciju Product, Run.

Program sastavljen u Xcode IDE-u bi trebalo da izgleda slično onom koji je ilustrovan na slici 1.1.

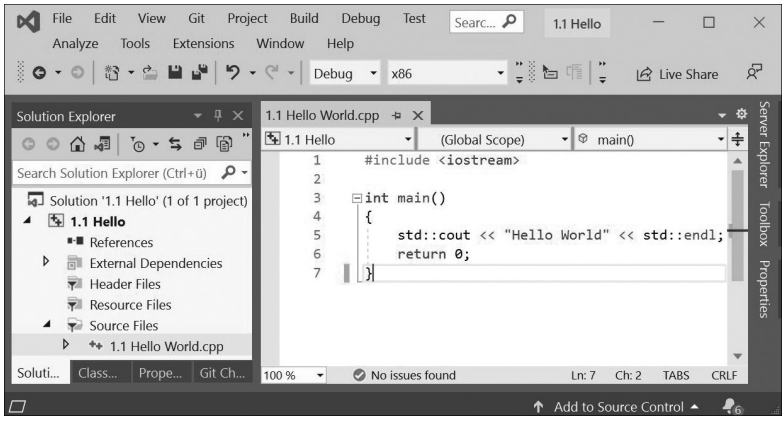


Slika 1.1 Jednostavan „Hello World“ C++ program editovan u Xcode 12.5 IDE-u na macOS sistemu.

SAVET Da biste kompajlirali C++20 kod u Xcode IDE-u potrebno je da ga eksplicitno omogućite podešavanjem opcije C++ Language Dialect na C++20. Ova opcija se nalazi u meniju Build Settings.

Ako koristite Microsoft Visual Studio na Windows sistemu, pritisnite tastere Ctrl+F5 da biste pokrenuli program direktno pomoću IDE-a. Ova prečica će kompajlirati, povezati i izvršiti aplikaciju.

Program kreiran u Microsoft Visual Studio IDE-u bi trebalo da izgleda slično kao što je ilustrovano na slici 1.2.



Slika 1.2 Jednostavan „Hello World“ C++ program editovan u Microsoft Visual Studio 2019 IDE-u.

SAVET Da biste kompajlirali C++20 kod u Microsoft Visual Studio IDE-u potrebno je da ga eksplicitno omogućite podešavanjem opcije C++ Language Standard na ISO C++20 Standard. Ova opcija se nalazi u meniju Project, Properties.

Izvršavanje fajla `./hello` ili fajla `Hello.exe` vraća sledeći ispis:

```
Hello World!
```

Čestitam! Započeli ste učenje jednog od najpopularnijih i najmoćnijih programskih jezika svih vremena!

ZNAČAJ C++ ISO STANDARDA

Programski kod 1.1 prikazuje da usklađenost sa C++ ISO standardom pomaže pri razvoju koda koji može da bude kompajliran i izvršen na više operativnih sistema.

ISO standard za C++ omogućava konzistentnu podršku za kompajler i prenosivost koda na različite platforme. Zato, kao programer možete da doprete do više korisnika, bez potrebe za programom koji je specifičan za svaki operativni sistem koji oni koriste.

Razumevanje grešaka kompajlera

Kompajleri su veoma precizni u svojim zahtevima, ali samo dobri kompajleri se trude da ukažu gde se nalaze greške u kodu. Da biste to videli na delu, možete namerno da unesete grešku u programski kod 1.1 tako što ćete izbrisati tačku-zarez (;) na kraju linije 5 ispred `return` u liniji 6. Sa tačke gledišta kompajlera, započeli ste nov iskaz u liniji 6 bez završavanja prethodnog iskaza u liniji 5 - što je protiv pravila. Ako sada pokušate da kompajlirate kod, dobićete grešku, kao što je sledeća:

```
1.1 Hello World.cpp(6,2): error C2143: syntax error: missing ';' before 'return'
```

Ova poruka o grešci iz Visual C++ kompajlera počinje nazivom fajla koji sadrži grešku, brojem linije u kojoj se suočio sa novim iskazom pre zaključenja prethodnog (u ovom slučaju 6) i opisom greške i brojem greške (u ovom slučaju `C2143`). Možete da ispravite ovu grešku dodavanjem znaka tačka-zarez na početak šeste linije i videćete da će program biti kompajliran bez problema!

NAPOMENA Upotreba znaka tačka-zarez (;) je validan način za završavanje iskaza u C++ jeziku. U C++ jeziku prelom linije neće automatski završiti iskaze kao u nekim drugim jezicima, kao što je VBScript.

U C++ jeziku iskazi mogu da se protežu kroz više linija. Takođe je moguće da imate više iskaza u jednoj liniji, u kojoj je svaki iskaz završen znakom tačka-zarez.

Šta je novo u verziji C++20?

Revizije C++ standarda dovele su do jezika koji je jednostavniji za upotrebu za programiranje bez kompromitovanja mogućnosti pisanja aplikacija visokih performansi.

ISO je objavio verziju C++20 decembra 2020. godine, što predstavlja ogroman korak u modernizaciji jezika. Predstavljeni su trosmerni operator poređenja (takođe ga nazivamo i spaceship operator), validacija parametara šablona, biblioteka novih raspona koja uključuje prikaze i adaptere koji olakšavaju moćne operacije u kolekcijama, dalja standardizacija višenitnog rada, podrška za sinhronizaciju pomoću korutina i poboljšani lambda izrazi. Takođe, u verziji C++20 predstavljeni su *moduli* - dugo očekivana funkcija koja prevazilazi nedostatke uključivanja fajlova zaglavlja (`.h`) i znatno ubrzava proces kompajliranja u velikim projektima. Izmene u verziji C++20 su toliko drastične da u vreme pisanja ovog teksta nijedan kompajler ne obećava podršku za C++20 standard, mada ovde pomenuti kompajleri odlično napreduju ka tom cilju.

C++ je jezik koji se i dalje razvija. Funkcije koje očekujemo u sledećoj reviziji, koju očekujemo 2023. godine, predstavljene su na kraju knjige, u lekciji 31, „C++20 moduli i C++23“.

Rezime

U ovoj lekciji naučili ste da programirate, kompajlirate, povezujete i izvršavate prvi C++ program. U njoj je, takođe, predstavljen kratak pregled evolucije C++ jezika i demonstrirana je efikasnost standarda prikazom da isti program može da bude kompajliran korišćenjem različitih kompajlera na različitim operativnim sistemima.

Pitanja i odgovori

P **Mogu li da ignorišem poruke upozorenja kompajlera?**

O U određenim slučajevima kompajler će prikazati poruke upozorenja. Upozorenja se razlikuju od grešaka po tome što je konkretna linija sintaktički tačna i može da se kompajlira. Međutim, verovatno postoji bolji način da se konkretna linija napiše, a dobri kompajleri će prikazati upozorenje, zajedno sa preporukom za ispravku.

Trebalo bi da obratite pažnju na ova upozorenja i da, u skladu sa njima, poboljšate aplikaciju. Nemojte ignorisati poruke upozorenja osim ako ste sigurni da su one lažno pozitivne.

P Po čemu se razlikuje interpretirani od kompajliranog jezika?

O Jezici kao što su JavaScript, Ruby i Python su interpretirani. Interpretirani jezik koristi interpreter, koji čita (interpretira) kod u fajlu skripta i izvršava željene akcije. Kompajlirani jezici, kao što je C++, koriste korak izgradnje za kreiranje izvršnih fajlova koji sadrže instrukcije za izvršenje na procesoru.

P Šta su greške pri izvršenju i po čemu se one razlikuju od grešaka u vreme kompajliranja?

O Greške koje se dešavaju kada izvršavate aplikaciju nazivamo *greške pri izvršenju*. Možda ste već iskusili poznatu grešku „Access Violation“ na Windows sistemu - to je greška pri izvršenju koju uzrokuje izvršni fajl koji ima grešku. Greške u vreme kompajliranja prekidaju proces izgradnje. Ukazuju na sintaktičke probleme u kodu i potrebno je da ih programer ispravi pre nego što može da nastavi kompajliranje.

P Zašto je potrebno eksplicitno omogućiti C++20 funkcije pri upotrebi g++, clang++ ili MSVC kompajlera? Zašto nisu podrazumevano uključene?

O Izmene u jeziku C++ uključuju nove funkcije i poboljšanja postojećih. Kompajleri treba da obezbede kompatibilnost sa starijim verzijama i stabilnost izvršenja postojećeg koda. Oni to rade predstavljanjem novih funkcija (neke su još u eksperimentalnoj fazi) koje podržavaju C++ standard korišćenjem novih parametara komandne linije da postojeći skriptovi izgradnje ne bi bili ugroženi.

Radionica

Radionica sadrži kviz da biste proverili koliko dobro razumete ono što ste naučili u ovoj lekciji i vežbe za sticanje iskustva u upotrebi onoga što ste naučili. Pokušajte da odgovorite na pitanja u kvizu i obavite vežbe pre nego što proverite odgovore u Dodatku E, i uverite se da razumete odgovore pre nego što pređete na sledeću lekciju.

Kviz

1. Ignorišući sintaksu, koja je razlika između kompajliranog jezika, kao što je C++ i interpretiranog jezika, kao što je JavaScript?
2. Šta radi program za povezivanje?
3. Koji su koraci u normalnom ciklusu razvoja programa?

Vežbe

1. Pogledajte sledeći program i pokušajte da pogodite šta on izvršava, a da ga ne pokrenete:

```
 1: #include<iostream>
 2: int main()
 3: {
 4:    int x = 8;
 5:    int y = 6;
 6:    std::cout << std::endl;
 7:    std::cout << x - y << " " << x * y << " " << x + y;
 8:    std::cout << std::endl;
 9:    return 0;
10: }
```

2. Unesite program iz vežbe 1, a zatim ga kompajlirajte i povežite. Šta izvršava ovaj program? Izvršava li ono što ste mislili?

3. Šta mislite u čemu je greška u sledećem programu:

```
1: include<iostream>
2: int main()
3: {
4:      std::cout << "Hello Buggy World " << std::endl;
5:      return 0;
6: }
```

4. Ispravite grešku u programu iz vežbe 3, kompajlirajte ga, povežite i pokrenite. Šta izvršava ovaj program?