UVOD

Postoji mnoštvo knjiga za C# koje su namenjene da budu sveobuhvatne reference za C# programski jezik i .NET Framework.

Ova knjiga je drugačija. Ona je koncizna, žustra i zabavna za čitanje i sadrži praktične vežbe koje će vas voditi kroz svaku temu. Širina sveobuhvatnog narativa ima cenu dubine, ali ćete pronaći mnoge putokaze za dalje istraživanje, ako želite. Ova knjiga je istovremeno vodič, korak-po-korak, za učenje modernih, dokazanih tehnika C#-a upotrebom međuplatformskog .NET-a, a takođe predstavlja kratak uvod u glavne tipove aplikacija koje možemo da gradimo. Ona je najbolja za početnike za C# i .NET ili za programere koji su već koristili C# u prošlosti, ali osećaju kao da zaostaju u upotrebi novih funkcija zbog promena u proteklih nekoliko godina. Ako ste već koristili C# 5.0 ili noviji, u prvoj temi Poglavlja 2, „Govorite C# jezikom“, možete da pregledate tabele novih funkcija jezika i da pređete direktno na njih. Ako već imate iskustva u upotrebi .NET Corea 1.0 ili novijeg, u prvoj temi Poglavlja 7, „Razumevanje i pakovanje .NET tipova“, možete da pregledate tabele novih funkcija platforme i da pređete direktno na njih.

Istaći ću sve interesantne delove C#-a i .NET-a da biste mogli da impresionirate svoje kolege i brzo postanete produktivni. Da ne bih dosađivao nekim čitaocima objašnjavanjem svake sitnice, pretpostaviću da ste dovoljno pametni da potražite na „Googleu“ objašnjenje za teme koje su povezane, ali nisu obavezno uključene u vodič za početnike ili programere sa malo iskustva.

Možete da preuzmete rešenja za zadatke, vođene korak po korak, i vežbe iz sledećeg GitHub skladišta. Obezbediću vam i instrukcije kako da to uradite koristeći Visual Studio Code, na kraju Poglavlja 1, „Zdravo C#! Dobrodošao .NET!“.

http://bit.ly/3aB2P40

ŠTA OBUHVATA OVA KNJIGA

Poglavlje 1, „Zdravo C#! Dobrodošao .NET!“, posvećeno je podešavanju razvojnog okruženja i upotrebi Visual Studio Codea za kreiranje najjednostavnijih mogućih aplikacija pomoću C#-a i .NET-a. Naučićete kako da napišete i kompajlirate kod u bilo kojem od podržanih operativnih sistema: Windows, mac OS X, Linux varijantama. Takođe ćete naučiti koja su najbolja mesta da potražite pomoć.

U Poglavlju 2, „Govorite C# jezikom“, predstavićemo verzije C#-a i tabele u kojima su prikazane verzije koje uključuju nove funkcije, a zatim ćemo objasniti gramatiku i rečnik koje ćete koristiti svakodnevno za pisanje izvornog koda za vaše aplikacije. Konkretno, naučićete kako da deklarišete i upotrebite različite vrste promenljivih i učićete o velikoj promeni u C# 8.0 predstavljanjem referentnih tipova koji prihvataju nulu.

U Poglavlju 3, „Kontrolisanje toka i konvertovanje tipova“, biće reči o upotrebi operatora za izvršavanje jednostavnih akcija u promenljivim, uključujući poređenja, pisanje koda za donošenje odluka, ponavljanje bloka iskaza i konvertovanje između tipova. Takođe ćete upoznati odbrambeno pisanje koda za obradu grešaka kada se one neizbežno javljaju.

Poglavlje 4, „Pisanje, ispravljanje grešaka i testiranje funkcija“, posvećeno je praćenju principa **Don’t Repeat Yourself (DRY)** pisanjem ponovo upotrebljivih funkcija. Naučićete kako da upotrebite alatke za ispravljanje grešaka da biste pratili i uklanjali programske greške, da biste nadgledali kod dok se izvršava, da biste dijagnostifikovali probleme i da biste strogo testirali kod, uklonili programske greške i obezbedili stabilnost i pouzdanost pre nego što je kod raspoređen u proizvodnju.

Poglavlje 5, „Kreiranje sopstvenih tipova upotrebom objektno-orijentisanog programiranja“, posvećeno je različitim kategorijama članova koje tipovi mogu imati, uključujući polja za čuvanje podataka i metode za izvršavanje akcija. Upotrebićete OOP koncepte, kao što su agregacija i kapsuliranje. Naučićete i funkcije jezika, kao što su podrška za sintaksu torke i promenljive out, podrazumevani literali i izvedeni nazivi torke.

Poglavlje 6, „Implementiranje interfejsa i nasleđivanje klasa“, govorićemo o izvođenju novih tipova iz postojećih upotrebom objektno-orijentisanog programiranja (OOP). Naučićete kako da definišete operatore i lokalne funkcije, prosleđivanja i događaje, kako da implementirate interfejse o osnovnim i izvedenim klasama, kako da promenite člana tipa, kako da upotrebite polimorfizam, kako da kreirate proširene metode i kako da konvertujete između klasa u hijerarhiji nasleđivanja.

U Poglavlju 7, „Razumevanje i pakovanje .NET tipova“, predstavićemo verzije .NET Corea i uključićemo tabele u kojima su navedene verzije koje uključuju nove funkcije, zatim ćemo predstaviti .NET Core tipove koji su usklađeni sa .NET Standardom i načine na koji su oni povezani sa C# jezikom. Naučićete kako da rasporedite i pakujete sopstvene aplikacije i biblioteke.

Poglavlje 8, „Upotreba uobičajenih .NET tipova“, posvećeno je tipovima koji omogućavaju kodu da izvrši uobičajene praktične zadatke, kao što su manipulisanje brojevima i tekstom, skladištenje stavki u kolekcijama i implementiranje internacionalizacije.

U Poglavlju 9, „Upotreba fajlova, nizova i serijalizacije“, govorićemo o interakciji sa fajl sistemom, čitanju fajlova i tokova podataka i pisanju u njih, šifrovanju teksta i serijalizaciji formata kao što su JSON i XML.

U Poglavlju 10, „Zaštita podataka i aplikacija“, biće reči o zaštiti podataka, odnosno kako da upotrebom enkripcije sprečite zlonamerne korisnike da ih pregledaju i kako da ih sprečite pomoću heširanja i potpisivanja da oštete podatke i manipulišu njima. Takođe ćete učiti o autentifikaciji i autorizaciji da biste zaštitili aplikacije od neautorizovanih korisnika.

U Poglavlju 11, „Upotreba baza podataka pomoću Entity Framework Corea“, objasnićemo čitanje i pisanje u bazu podataka, kao što su Microsoft SQL Server i SQLite, upotrebom tehnologije objektno-relacionog mapiranja (ORM) pod nazivom Entity Framework Core.

U Poglavlju 12, „Slanje upita i manipulisanje podacima upotrebom LINQ-a“, biće reči o Language Integrated Queries (LINQ) – ekstenzijama jezika koje dodaju mogućnost upotrebe sekvenci stavki i filtriranja, sortiranja i projektovanja stavki u različite ispise.

U Poglavlju 13, „Poboljšanje performanse i skalabilnosti korišćenjem višeprogramskog rada“, objasnićemo kako se omogućava izvršenje više akcija istovremeno za poboljšanje performanse, skalabilnosti i produktivnosti korisnika. Učićete o funkciji async Main i o upotrebi tipova u imenskom prostoru System.Diagnostics za nadgledanje koda i merenje performansi i efikasnosti.

U Poglavlju 14, „Praktična primena C#-a i .NET-a“, predstavićemo tipove međuplatformskih aplikacija koje možete da izgradite pomoću C#-a i .NET-a.

U Poglavlju 15, „Izgradnja veb sajtova upotrebom ASP.NET Core Razor Pagesa“, upoznaćete osnove izgradnje veb sajtova pomoću moderne HTTP arhitekture na strani servera upotrebom ASP.NET Corea. Naučićete kako da implementirate ASP.NET Core funkciju, poznatu kao Razor Pages, koja pojednostavljuje kreiranje dinamičkih veb stranica za male veb sajtove.

U Poglavlju 16, „Kreiranje veb sajtova upotrebom Model-View-Controller obrasca“, naučićete kako da izgradite velike, složene veb sajtove tako da jedinično testiranje i upravljanje timovima programera budu olakšani upotrebom ASP.NET Core MVC-a. Učićete o početnoj konfiguraciji, autentifikaciji, rutama, modelima, prikazima i kontrolerima.

U Poglavlju 17, „Izgradnja veb sajtova upotrebom sistema za upravljanje sadržajem“, predstavićemo kako sistem za upravljanje sadržajem (CMS) omogućava programerima da brzo izgrade veb sajtove, koristeći prilagodljive korisničke interfejse za administraciju, koju netehnički korisnici mogu da upotrebe za kreiranje

i upravljanje sopstvenim sadržajem. Na primer, učićete o jednostavnom CMS-u otvorenog koda, koji je zasnovan na .NET Coreu, pod nazivom Piranha CMS.

U Poglavlju 18, „Kreiranje veb servisa i njihova upotreba“, objasnićemo kako se grade backent veb servisi REST arhitekture upotrebom ASP.NET Core Web API-a i kako se pravilno upotrebljavaju korišćenjem fabrički instanciranih HTTP klijenata.

U Poglavlju 19, „Izgradnja inteligentnih aplikacija upotrebom mašinskog učenja“, predstavićemo „Microsoftov“ ML.NET paket algoritama mašinskog učenja, otvorenog koda, koji može da se upotrebi za ugradnju prilagodljive inteligencije u bilo koju međuplatformsku .NET aplikaciju, kao što je veb sajt za online trgovinu, koji obezbeđuje preporuke posetiocima da određene proizvode dodaju u svoje korpe za kupovinu.

Poglavlje 20, „Izgradnja Windows desktop aplikacija“, je prvo od dva poglavlja u kojima govorimo izvan okvira onoga što može da se postigne upotrebom međuplatformskog .NET Corea i Visual Studio Codea. U njemu ćemo predstaviti kako .NET Core 3.0 i njegov Windows Desktop Pack omogućavaju Windows Formsu i WPF-u da iskoriste prednosti pokretanja na .NET Coreu. Zatim ćete učiti o osnovama XAML-a koji možete da upotrebite za definisanje korisničkog interfejsa za grafičke aplikacije za Windows Presentation Foundation (WPF) ili Universal Windows Platform (UWP). Primenićete principe i funkcije Fluent Designa da biste poboljšali UWP aplikaciju. Aplikacije za ovo poglavlje moraju biti izgrađene upotrebom Visual Studioa 2019 na Windows 10 mašini.

U Poglavlju 21, „Izgradnja međuplatformskih aplikacija za mobilne uređaje upotrebom Xamarin.Formsa“, opisaćemo upotrebu C# jezika za mobilne uređaje, tako što ćemo izgraditi međuplatformsku aplikaciju za iOS i Android. Aplikacija za ovo poglavlje će biti izgrađena upotrebom Visual Studioa 2019 za Mac na macOS mašini.

Dodatak „Odgovori na pitanja za testiranje znanja“ sadrži odgovore na test pitanja na kraju svakog poglavlja. Možete da pročitate ovaj dodatak na stranici https://static.packt-cdn.com/downloads/9781788478120_Appendix_Answers_to_the_Test_Your_Knowledge_Questions.pdf.

ŠTA VAM JE POTREBNO ZA OVU KNJIGU?

Možete da razvijate i raspoređujete C# i .NET Core aplikacije upotrebom Visual Studio Codea na mnogim platformama, uključujući Windows, macOS i mnoge različite distribucije Linuxa. Operativni sistem koji podržava Visual Studio Code i internet konekciju je sve što vam je potrebno da završite vežbe iz poglavlja od 1 do 19.

Potreban vam je Windows 10 da biste izgradili aplikacije iz Poglavlja 20, „Izgradnja Windows desktop aplikacija“.

Potreban vam je macOS da biste izgradili aplikacije iz Poglavlja 21, „Izgradnja međuplatformskih aplikacija za mobilne uređaje upotrebom Xamarin.Formsa“, jer treba da imate macOS i Xcode da biste kompajlirali iOS aplikacije.

KONVENCIJE

U ovoj knjizi pronaći ćete više različitih stilova za tekst koje sam upotrebio za različite vrste informacija. Evo nekih primera ovih stilova i objašnjenja njihovog značenja.

Reči koda u tekstu, nazivi tabela baze podataka, nazivi direktorijuma, nazivi fajlova, ekstenzije fajla, nazivi putanja, kratki URL-ovi, korisnički unos i Twitter statusi su prikazani na sledeći način: „Direktorijumi `Controllers`, `Models` i `Views` sadrže ASP.NET `Core` klase i `.cshtml` fajlove za izvršenje na serveru“. Blok koda je postavljen na sledeći način:

```
// storing items at index positions
names[0] = "Kate";
names[1] = "Jack";
names[2] = "Rebecca";
names[3] = "Tom";
```

Da bismo usmerili vašu pažnju na određeni deo bloka koda, relevantne linije ili stavke smo ispisali podebljanim slovima:

```
// storing items at index positions
names[0] = "Kate";
names[1] = "Jack";
names[2] = "Rebecca";
names[3] = "Tom";
```

Svi unosi ili ispisi komandne linije napisani su na sledeći način:

```
dotnet new console
```

Novi termini i važne reči su napisani **podebljanim** slovima. Reči koje vidite na ekranu - na primer, u menijima ili okvirima za dijalog, biće prikazane u tekstu na sledeći način: „Kliknite na dugme **Next** da biste se prebacili na sledeći ekran“.\



Više informacija: linkovi ka eksternim izvorima za dalje čitanje biće prikazani u okviru kao što je ovaj.



Dobra praksa: Preporuke kako da programirate kao stručnjak prikazane su ovako.

POVRATNE INFORMACIJE OD ČITALACA

Povratne informacije od naših čitalaca su uvek dobrodošle. Obavestite nas šta mislite o ovoj knjizi – šta vam se dopalo ili šta vam se možda nije dopalo. Povratne informacije čitalaca su nam važne, jer nam pomažu da objavljujemo naslove od kojih ćete dobiti maksimum.

Da biste nam poslali povratne informacije, jednostavno nam pošaljite e-mail na adresu feedback@packtpub.com i u naslovu poruke napišite naslov knjige.

Ako postoji tema za koju ste specijalizovani i zainteresovani ste da pišete ili sarađujete na nekoj od knjiga, pogledajte vodič za autore na adresi www.packtpub.com/authors.

KORISNIČKA PODRŠKA

Sada, kada ste ponosni vlasnik „Packt" knjige, imamo mnogo štošta da vam ponudimo da biste dobili maksimum iz nje.

Preuzimanje primera koda

Kod možete da preuzmete sa našeg sajta:

http://bit.ly/2NMr3P2

Kada su fajlovi preuzeti, raspakujte ili ekstrahujte direktorijum, koristeći najnoviju verziju:

- **WinRAR** / 7-Zip za Windows
- **Zipeg** / iZip / UnRarX za Mac
- **7-Zip** / PeaZip za Linux

Kod za ovu knjigu se nalazi i u GitHub skladištu, na adresi:

http://bit.ly/2RJSSc9

Takođe imamo druge pakete koda iz našeg bogatog kataloga knjiga i video-snimaka, koji su dostupni na adresi https://github.com/PacktPublishing/.

Preuzimanje kolornih slika za ovu knjigu

Takođe smo vam obezbedili PDF fajl koji sadrži kolorne slike ekrana/dijagrama koji su upotrebljeni u ovoj knjizi. Te slike će vam pomoći da bolje razumete promene u ispisu.

Ovaj fajl možete da preuzmete sa adrese:

http://bit.ly/2RhzML9

1

Zdravo C#! Dobrodošao .NET!

Ciljevi u Poglavlju 1 su podešavanje razvojnog okruženja, razumevanje sličnosti i razlika između .NET Corea, .NET Frameworka i .NET Standarda, a zatim kreiranje najjednostavnije moguće aplikacije pomoću C# jezika i .NET Corea upotrebom „Microsoftovog" Visual Studio Codea.

Nakon ovog poglavlja, knjiga može da se podeli na tri dela: prvi sadrži gramatiku i rečnik C# jezika, drugi tipove dostupne u .NET Coreu za izgradnju funkcija aplikacije, a treći primere uobičajenih međuplatformskih aplikacija koje možete da izgradite pomoću C#-a i .NET-a. Poslednja dva poglavlja sadrže dva tipa aplikacija koje mogu da se izgrade pomoću C# jezika, ali prva nije međuplatformska, a druga ne koristi .NET Core, pa zbog toga ova poglavlja treba da se tretiraju kao bonus poglavlja.

Većina korisnika uči komplikovane teme imitacijom i ponavljanjem, umesto da pročita detaljno objašnjenje teorije; prema tome, ja neću objašnjavati detaljno svaki korak u ovoj knjizi. Cilj je da vas naučim da napišete kod, da izgradite aplikaciju iz tog koda, a zatim da aplikaciju vidite pokrenutu.

Ne treba odmah da znate sve detalje. To je nešto što ćete saznavati vremenom dok budete gradili svoje aplikacije i budete izašli izvan okvira onoga što bilo koja knjiga može da vas nauči.

Sudeći po knjizi „English dictionary iz 1755", čiji je autor Samuel Johnson, verovatno sam napravio nekoliko kardinalnih grešaka i napisao nekoliko apsurdnih izraza, od kojih ni jedan posao ovakvog opsega nije oslobođen. Preuzimam potpunu odgovornost za greške i nadam se da ćete ceniti izazov mog pokušaja da „podignem vetar" pisanjem ove knjige o tehnologijama koje se veoma brzo razvijaju, kao što su C# i .NET Core, i o aplikacijama koje pomoću njih možete da izgradite.

Ovo poglavlje obuhvata sledeće teme:

- podešavanje razvojnog okruženja
- razumevanje .NET-a
- izgradnja konzolne aplikacije upotrebom Visual Studio Codea
- preuzimanje koda rešenja iz GitHub skladišta
- traženje pomoći

PODEŠAVANJE RAZVOJNOG OKRUŽENJA

Pre nego što započnete da programirate, potreban vam je editor koda za C#. „Microsoft" ima familiju editora koda i **integrisana razvojna okruženja (IDE)**, koja uključuju:

- Visual Studio Code
- Visual Studio 2019
- Visual Studio 2019 for Mac

Upotreba Visual Studio Codea za međuplatformski razvoj

Najmoderniji i najjednostavniji editor koda koji možete da izaberete i jedini iz „Microsofta" koji je međuplatformski je Microsoft Visual Studio Code. On može da se pokreće na svim uobičajenim operativnim sistemima, uključujući Windows, macOS i mnoge distribucije Linuxa, uključujući **Red Hat Enterprise Linux (RHEL)** i Ubuntu.

Visual Studio Code je dobar izbor za moderan međuplatformski razvoj, jer ima obiman i rastući skup ekstenzija za podršku mnogih jezika pored C#-a, a pošto je međuplatformski i jednostavan, može da se instalira na svim platformama na kojima će aplikacija biti upotrebljena za brzo ispravljanje programskih grešaka i izvršavanje nekih drugih zadataka.

Upotreba Visual Studio Codea podrazumeva da programer može da koristi međuplatformski editor koda za razvoj međuplatformskih aplikacija. Prema tome, ja sam izabrao Visual Studio Code za sva poglavlja ove knjige, osim za poslednja dva, jer ona zahtevaju specijalne funkcije koje nisu dostupne u Visual Studio Codeu za izgradnju Windows i mobilnih aplikacija.



Više informacija: Možete da pročitate o planovima „Microsofta" za Visual Studio Code na linku https://github.com/Microsoft/vscode/wiki/Roadmap.

Ako više volite da se koristite Visual Studio Code 2019 ili Visual Studio for Mac, umesto Visual Studio Codea, naravno da možete da ih upotrebite, ali pretpostavljam da vam za to nisu potrebne instrukcije korak-po-korak, pa ih neću navesti u ovoj knjizi.



Više informacija: Možete da pročitate poređenje Visual Studio Codea i Visual Studioa 2019 na linku https://www.itworld.com/article/3403683/visual-studio-code-stepping-on-visual-studios-toes.html.

Upotreba Visual Studioa 2019 za razvoj Windows aplikacija

Microsoft Visual Studio 2019 se pokreće samo na Windows verziji 7 SP1 ili novijoj. Treba da pokrenete Windows 10 da biste kreirali **Universal Windows Platform (UWP)** aplikacije. To je jedina „Microsoftova" razvojna alatka koja može da kreira Windows aplikacije, pa ćemo je koristiti u Poglavlju 20, „Izgradnja Windows desktop aplikacija".

Upotreba Visual Studioa za Mac za razvoj mobilnih aplikacija

Da biste kreirali aplikacije za Apple operativni sistem, kao što je iOS, za pokretanje uređaja kao što su iPhone i iPad treba da imate Xcode, ali se ta alatka pokreće samo na macOS sistemu. Iako možete da upotrebite Visual Studio 2019 na Windowsu sa njegovim Xamarin ekstenzijama za pisanje međuplatformske aplikacije za mobilne uređaje, i dalje su vam potrebni macOS i Xcode da biste kompajlirali aplikaciju.

Upotrebićemo Visual Studio 2019 for Mac na macOS sistemu u Poglavlju 21, „Izgradnja međuplatformske aplikacije za mobilne uređaje pomoću Xamarin.Formsa".

Preporučene alatke za poglavlja

Da bismo vam pomogli da podesite najbolje okruženje za upotrebu u ovoj knjizi, u sledećoj tabeli rezimirali smo alatke i operativne sisteme koje preporučujem da upotrebite za svako od poglavlja ove knjige:

POGLAVLJA	ALATKA	OPERATIVNI SISTEMI
Poglavlja od 1 do 19	Visual Studio Code	Windows, macOS, Linux
Poglavlje 20	Visual Studio 2019	Windows 10
Poglavlje 21	Visual Studio 2019 for Mac	macOS

Za pisanje ove knjige ja sam upotrebio MacBook Pro i sledeći softver:

- Visual Studio Code na macOS sistemu kao primarni editor koda
- Visual Studio Code na Windows 10 sistemu u virtuelnoj mašini za testiranje specifičnih ponašanja operativnog sistema, kao što je upotreba fajl sistema

- Visual Studio 2019 na Windows 10 sistemu u virtuelnoj mašini za izgradnju Windows aplikacija
- Visual Studio 2019 for Mac na macOS sistemu za izgradnju aplikacija za mobilne uređaje



Više informacija: Google i Amazon podržavaju Visual Studio Code, kao što možete pročitati na linku https://www.cnbc.com/2018/12/20/microsoft-cmo-capossela-says-google-employees-use-visual-studio-code.html.

Međuplatformsko raspoređivanje

Vaš izbor editora koda i operativnog sistema za razvoj neće ograničiti mesto gde će kod biti raspoređen.

.NET Core 3.0 podržava sledeće platforme za raspoređivanje:

- **Windows**: Windows 7 SP1, ili noviji, Windows 10 verziju 1607, ili noviju, Windows Server 2012 R2 SP1, ili noviji, Nano Server verziju 1803 ili noviju
- **Mac**: macOS High Sierra (verzija 10.13), ili noviju
- **Linux**: RHEL 6, ili noviju, RHEL, CentOS, Oracle Linux verziju 7 ili noviju, Ubuntu 16.04, ili noviju, Fedora 29, ili noviju, Debian 9, ili noviju, openSUSE 15, ili noviju.



Više informacija: Zvaničnu listu podržanih operativnih sistema možete da pročitate na linku https://github.com/dotnet/core/blob/master/release-notes/3.0/3.0-supported-os.md.

Razumevanje verzija Microsoft Visual Studio Codea

„Microsoft“ izdaje novu verziju Visual Studio Codea (skoro) svakog meseca, a ispravke programskih grešaka još češće - na primer:

- Verzija 1.38, glavno izdanje, u avgustu 2019. godine
- Verzija 1.38.1, ispravke grešaka, izdanje u avgustu 2019. godine



Više informacija: Možete da pročitate više o najnovijim verzijama na linku https://code.visualstudio.com/updates.

Verzija upotrebljena u ovoj knjizi je 1.38.1, izdata 11. septembra 2019. godine, ali je verzija Microsoft Visual Studio Codea manje važna od verzije **C# for Visual Studio Code** ekstenzije, koju ćete kasnije instalirati.

Iako C# ekstenzija nije potrebna, obezbeđuje IntelliSense dok kucate, navigaciju koda i funkcije za ispravljanje grešaka, pa je, stoga, veoma korisno da je instalirate. Za podršku C# 8.0 treba da instalirate C# ekstenziju verziju 1.21.3 ili noviju.

U ovoj knjizi ja ću vam pokazati prečice na tastaturi i snimke ekrana Visual Studio Codea upotrebom macOS verzije. Verzije Visual Studio Code na Windowsu i varijantama Linuxa su praktično identične, mada se prečice na tastaturi verovatno razlikuju.

Neke od uobičajenih prečica na tastaturi koje ćemo upotrebiti prikazane su u sledećoj tabeli:

AKCIJA	MACOS	WINDOWS
prikaz komandne palete	Cmd + Shift + P, F1	Ctrl + Shift + P, F1
prelaz na definiciju	F12	F12
vraćanje nazad	Ctrl + -	Alt + ⌧
napred	Ctrl + Shift + -	Alt + ⌧
prikaz terminala	Ctrl + , (backtick)	Ctrl + , (quote)
novi terminal	Ctrl + Shift + , (backtick)	Ctrl + Shift + , (quote)
prekopčavanje komentara linije	Ctrl + /	Ctrl + /
prekopčavanje komentara bloka	Shift + Option + A	Shift + Alt + A

Preporučujem da preuzmete PDF prečica na tastaturi za vaš operativni sistem sa sledeće liste:

- Windows: https://code.visualstudio.com/shortcuts/keyboardshortcuts-windows.pdf
- macOS: https://code.visualstudio.com/shortcuts/keyboardshortcuts-macos.pdf
- Linux: https://code.visualstudio.com/shortcuts/keyboardshortcuts-linux.pdf



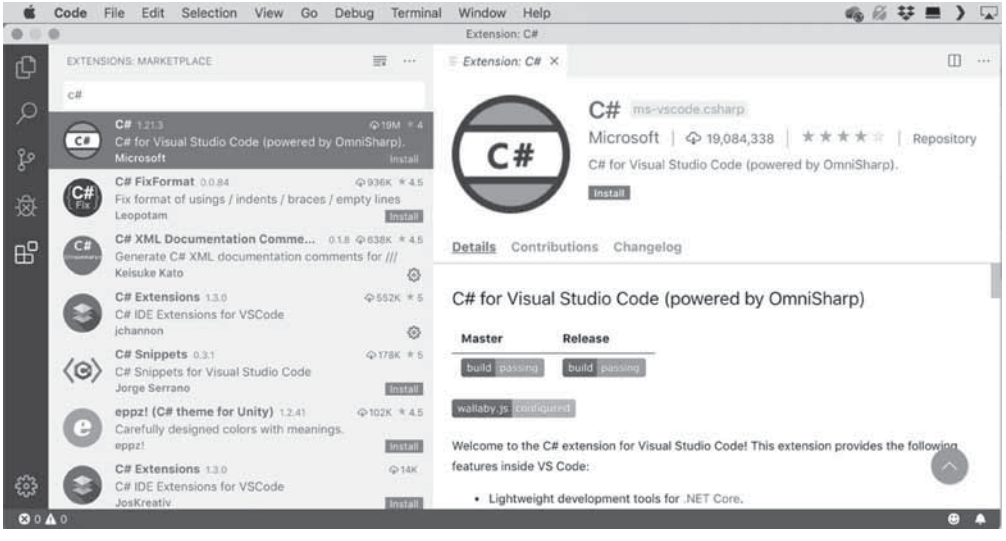
Više informacija: Možete da naučite više o podrazumevanim prečicama za Visual Studio Code i kako da ih prilagodite na linku https://code.visualstudio.com/docs/getstarted/keybindings.

Visual Studio Code je brzo poboljšan tokom poslednjih nekoliko godina i prijatno je iznenadio „Microsoft“ svojom popularnošću. Ako ste hrabri i volite da živite „na ivici“, upotrebite Insiders izdanje.

Preuzimanje i instaliranje Visual Studio Codea

Sada ste spremni za preuzimanje i instaliranje Visual Studio Codea, njegove C# ekstenzije i .NET Core 3.0 SDK-a.

1. Preuzmite i instalirajte Stable verziju ili Insiders izdanje Visual Studio Codea sa linka https://code.visualstudio.com/.
2. Preuzmite i instalirajte .NET Core SDK sa linka https://www.microsoft.com/net/download.
3. Da biste instalirali C# ekstenziju, prvo treba da pokrenete Visual Studio Code aplikaciju.
4. U Visual Studio Codeu kliknite na Extensions ikonicu ili na View | Extensions.
5. C# je jedna od najpopularnijih dostupnih ekstenzija, pa bi trebalo da je vidite na vrhu liste, ili možete da unesete C# u polje za pretragu, kao što je prikazano na sledećoj slici:



6. Kliknite na Install i sačekajte da paketi podrške budu preuzeti i instalirani.



Više informacija: Možete da pročitate više o podršci Visual Studio Codea za C# na linku https://code.visualstudio.com/docs/languages/csharp.

Instaliranje drugih ekstenzija

U narednim poglavljima ove knjige upotrebićemo više ekstenzija. Ako želite da ih instalirate sada, sve ekstenzije koje ćemo upotrebiti prikazane su u sledećoj tabeli:

EKSTENZIJA	OPIS
C# for Visual Studio Code (powered by OmniSharp) ms-vscode.csharp	Podrška za C# editovanje, uključujući isticanje sintakse, IntelliSense, Go to Definition, Find All References, podršku ispravljanja grešaka za .NET Core (CoreCLR) i podršku za projekte project.json i csproj na Windows, macOS i Linux sistemima
C# XML Documentation Comments k--kato.docomment	Generiše komentare XML dokumentacije za Visual Studio Code.
C# Extensions jchannon.csharpextensions	Dodaje C# klasu, C# interfejs, polja i svojstva iz konstruktora i konstruktor iz svojstava.
REST Client humao.rest-client	Šalje HTTP zahtev i prikazuje odgovor direktno u Visual Studio Codeu.
ILSpy .NET Decompiler icsharpcode.ilspy-vscode	Dekompajlira MSIL sklopove (podrška za .NET Framework, .NET Core i .NET Standard).
SharpPad jmazouri.sharppad	Lako ispituje rezultate koda. Funkcioniše slično kao samostalne alatke, kao što su LinqPad i RoslynPad.

RAZUMEVANJE .NET-A

.NET Framework, .NET Core, Xamarin i .NET Standard su slične platforme koje programeri koriste za razvoj aplikacija i servisa. U nastavku ćemo predstaviti svaki od ovih .NET koncepata.

Razumevanje .NET Frameworka

.NET Framework je razvojna platforma koja uključuje **Common Language Runtime (CLR)** za upravljanje izvršenjem koda i **Base Class Library (BCL)**, bogatu biblioteku klasa za izgradnju aplikacija. „Microsoft" je prvobitno dizajnirao .NET Framework tako da ima mogućnost da bude međuplatformska platforma, ali se „Microsoft" potrudio i da ova platforma najbolje funkcioniše na Windows sistemima.

Zbog toga je .NET Framework 4.5.2 zvanična komponenta Windows operativnog sistema. .NET Framework je instaliran na više od milijardu računara, pa se mora menjati što manje. Ne ažurira se često, jer, čak, i ispravke grešaka mogu izazvati probleme.

Sve aplikacije na računaru napisane za .NET Framework dele istu verziju CLR-a i biblioteke sačuvane u Global Assembly Cacheu (GAC-u), što može da dovede do problema ako neki od njih zahteva specifičnu verziju za kompatibilnost.



Dobra praksa: Praktično, .NET Framework je zastarela platforma samo za Windows sisteme. Nemojte kreirati nove aplikacije upotrebom ove platforme.

Razumevanje Mono i Xamarin projekata

Nezavisni programeri su razvili .NET Framework implementaciju pod nazivom Mono projekat. Mono je međuplatformski projekat, ali znatno zaostaje za zvaničnom implementacijom .NET Frameworka.



Više informacija: Možete da pročitate više o ovom projektu na linku http://www.mono-project.com/.

Mono projekat je odlična osnova za Xamarin mobilnu platformu i za platforme za razvoj međuplatformskih igara, kao što je Unity.



Više informacija: Možete da pročitate više o Unityju na linku https://docs.unity3d.com/.

„Microsoft“ je kupio „Xamarin“ 2016 .godine i sada nudi ono što je nekada bila skupa Xamarin ekstenzija, besplatno sa Visual Studio 2019 softverom. „Microsoft“ je promenio naziv razvojne alatke **Xamarin Studio**, koja je mogla da se koristi samo za razvoj mobilnih aplikacija, u **Visual Studio for Mac** i omogućio je kreiranje drugih tipova aplikacija. Koristeći Visual Studio 2019 for Mac, „Microsoft“ je zamenio delove Xamarin Studio editora delovima iz Visual Studioa za Windows da bi obezbedio slično iskustvo i performanse.

Razumevanje .NET Corea

Mi živimo danas u pravom međuplatformskom svetu u kojem je zbog razvoja modernih mobilnih i cloud platformi Windows mnogo manje važna operativni sistem. Zbog toga se „Microsoft“ trudio da razdvoji usku vezu .NET-a od Windows operativnog sistema. Dok je menjao .NET Framework, tako da bude stvarno međuplatformski, iskoristio je mogućnost da ga refaktoriše i ukloni glavne delove koji se više ne smatraju važnim.

Ovaj novi proizvod je nazvan .**NET Core**, a uključuje međuplatformsku implementaciju CLR-a, poznatiju kao **CoreCLR**, i modernizovanu biblioteku klasa, koja je poznata kao **CoreFX**.

Scott Hunter, direktor programa za Microsoft partnere za .NET, ističe: „Oko četrdeset procenata naših .NET Core kupaca su novi programeri na ovoj platformi, što upravo i želimo za .NET Core. Želimo da uvedemo nove ljude".

NET Core se brzo razvija i pošto može da bude raspoređen „rame uz rame" sa aplikacijom, može da se menja često, a te promene neće uticati na druge .NET Core aplikacije na istoj mašini. Poboljšanja koja „Microsoft" može da izvrši u .NET Coreu ne mogu da se dodaju u .NET Framework.



Više informacija: Možete da pročitate više o „Microsoftovom" pozicioniranju .NET Corea i .NET Frameworka na linku https://devblogs.microsoft.com/dotnet/updateon-net-core-3-0-and-net-framework-4-8/.

Razumevanje budućih verzija .NET-a

Na Microsoft Build programerskoj konferenciji, održanoj u maju 2019. godine, .NET tim je objavio da će, nakon izdavanja .NET Corea 3.0, .NET Core biti preimenovan u .NET i da će u glavnim brojevima biti izostavljen broj 4 da bi se izbegla konfuzija sa .NET Frameworkom 4.x. Prema tome, sledeća verzija .NET Corea će biti .NET 5.0, koji čije izdavanje je planirano za novembr 2020. godine. „Microsoft" planira, nakon toga, godišnja izdavanja glavne verzije svakog novembra, a ne kao što je „Apple" izdavao glavne verzije iOS-a svake druge nedelje u septembru.



Više informacija: Možete da pročitate više o „Microsoftovim" planovima za .NET 5.0 na linku https://devblogs.microsoft.com/dotnet/introducing-net-5/.

U sledećoj tabeli prikazano je kada su izdate ključne verzije .NET Corea, za kada „Microsoft" planira sledeća glavna izdanja i koja verzija je upotrebljena za različita izdanja ove knjige:

VERZIJA	DATUM IZDANJA	IZDANJE KNJIGE	PUBLIKOVANA
.NET Core RC1	u novembru 2015. godine	prvo	u martu 2016. godine
.NET Core 1.0	u junu 2016. godine		
.NET Core 1.1	u novembru 2016. godine		
.NET Core 1.0.4 and .NET Core 1.1.1	u martu 2017. godine	drugo	u martu 2017. godine

VERZIJA	DATUM IZDANJA	IZDANJE KNJIGE	PUBLIKOVANA
.NET Core 2.0	u avgustu 2017. godine		
.NET Core for UWP in Windows 10 Fall Creators Update	u oktobru 2017. godine	treće	u novembru 2017. godine
.NET Core 2.1	u maju 2018. godine		
.NET Core 2.2	u decembru 2018. godine		
.NET Core 3.0 (Current)	u septembru 2019. godine	četvrto	u oktobru 2019. godine
.NET Core 3.1 (LTS)	u novembru 2019. godine		
.NET 5.0	u novembru 2020. godine		
.NET 6.0	u novembru 2021. godine		

Ne mogu da obećam 5. i 6. izdanje ove knjige, koja bi se poklapala sa budućim izdanjima .NET-a.

Razumevanje .NET Core podrške

.NET Core verzije su ili Long-Term Support (LTS) ili Current, kao što je opisano u sledećoj listi:

- **LTS** izdanja su stabilna i zahtevaju manje ažuriranja u svom „životnom veku". Ovo je dobar izbor za aplikacije koje ne nameravate da ažurirate često. LTS izdanja će biti podržana tri godine nakon što aplikacije postanu dostupne. .NET Core 3.1 će biti LTS izdanje.

.NET Core 1.0 i 1.1 došli su do kraja svog „životnog veka" i više ne uključuju podršku od juna 2019. godine, tri godine nakon prvog izdavanja .NET Corea 1.0.

- **Current** izdanja uključuju funkcije koje se mogu menjati na osnovu povratnih informacija. Ova izdanja su dobar izbor za aplikacije koje trenutno razvijate, jer obezbeđuju pristup najnovijim poboljšanjima. Nakon tromesečnog održavanja, prethodna manja verzija više neće uključivati podršku. Na primer, nakon 1.2 izdanja sistemi koji pokreću verziju 1.1 treba u roku od tri meseca da se ažuriraju na verziju 1.2 da bi zadržali podršku. .NET Core 3.0 je aktuelno izdanje, pa ako je .NET Core 3.1 izdat u novembru 2019. godine, kao što je planirano, biće potrebno da ga nadgradite do februara 2020. godine.

Oba izdanja se ažuriraju važnim ispravkama u svom „životnom veku" zbog bezbednosti i pouzdanosti. Morate da poznajete najnovije „zakrpe" da biste dobili podršku. Na primer, ako sistem pokreće 1.0 verziju, a izdata je verzija 1.0.1, treba da bude instalirana verzija 1.0.1.

Šta se razlikuje u .NET Coreu?

.NET Core je mnogo manji od aktuelne verzije .NET Frameworka, zbog činjenice da su zastarele tehnologije uklonjene. Na primer, **Windows Forms** i **Windows Presentation Foundation (WPF)** mogu da se upotrebe za izgradnju aplikacija grafičkog korisničkog interfejsa (GUI), ali su u uskoj vezi sa Windows operativnim sistemom, pa su, zato, uklonjeni iz .NET Corea na macOS i Linux sistemima.

Jedna od novih funkcija .NET Corea 3.0 je podrška za pokretanje starih Windows Forms i WPF aplikacija upotrebom **Windows Desktop Packa**, koji je uključen u Windows verziju .NET Corea 3.0, zbog čega je i veći od SDK-a za macOS i Linux. Možete da izvršite neke male promene u starim Windows aplikacijama ako je potrebno, a zatim da ih ponovo izgradite za .NET Core da biste iskoristili nove funkcije i poboljšanja performanse. Učićete o podršci za izgradnju ovih tipova Windows aplikacija u Poglavlju 20, „Izgradnja Windows desktop aplikacija".

Najnovija tehnologija koja se koristi za izgradnju Windows aplikacija je **Universal Windows** platforma (UWP), koja je građena na prilagođenoj verziji .NET Corea. UWP nije deo .NET Corea, jer nije međuplatformska alatka.

ASP.NET Web Forms i **Windows Communication Foundation (WCF)** su stare tehnologije za veb aplikacije i servise, koje sve manje programera koristi u novim razvojnim projektima, pa su takođe uklonjene iz .NET Corea. Umesto toga, programeri koriste ASP.NET MVC i ASP.NET Web API. Ove dve tehnologije su refaktorisane i kombinovane u novi proizvod koji se pokreće na .NET Coreu, pod nazivom **ASP.NET Core**. Učićete o ovim tehnologijama u Poglavlju 15, „Izgradnja veb sajtova pomoću ASP.NET Core Razor Pagesa", u Poglavlju 16 „Izgradnja veb sajtova upotrebom model-view-controller obrasca" , i u Poglavlju 18, „Izgradnja veb servisa i njihova upotreba".



Više informacija: Neki .NET Framework programeri su uznemireni što ASP.NET Web Forms, WCF i Windows Workflow (WF) nedostaju iz .NET Corea i voleli bi kada bi se „Microsoft" predomislio. Postoje projekti otvorenog koda koji omogućavaju uključivanje WCF-a i WF-a u .NET Core. Možete da pročitate više o ovome na linku https://devblogs.microsoft.com/dotnet/supporting-the-community-with-wf-and-wcf-ossprojects/.

Entity Framework (EF) 6 je tehnologija objektno-relacionog mapiranja koja je projektovana za upotrebu podataka sačuvanih u relacionim bazama podataka, kao što su Oracle i Microsoft SQL Server. Tokom godina je ova tehnologija „dobila na težini", pa je, stoga, međuplatformski API skraćen i dobija podršku za nerelacione baze podataka, kao što je Microsoft Azure Cosmos DB, i nosi naziv **Entity Framework Core**. O ovoj verziji ćete naučiti više u Poglavlju 11, „Upotreba baza podataka pomoću Entity Framework Corea".

Ako imate postojeće aplikacije koje koriste stari EF, znate da je verzija 6.3 podržana u .NET Coreu 3.0.

Osim uklanjanja velikih delova iz .NET Frameworka za kreiranje .NET Corea, „Microsoft" je „smestio" .NET Core u male pakete funkcionalnosti NuGet, koji mogu da budu raspoređeni nezavisno.

„Microsoftov" glavni cilj nije da.NET Core bude manji od .NET Frameworka, već da bude razdvojen u komponente .NET Core za podršku modernim tehnologijama i da ima manje zavisnosti, pa da raspoređivanje zahteva samo one pakete koji su potrebni za aplikaciju.

Razumevanje .NET Standarda

Postoje tri grane .NET platforme koje kontroliše „Microsoft", kao što je prikazano u sledećoj listi:

- .NET Core: za međuplatformske i nove aplikacije
- .NET Framework: za zastarele aplikacije
- Xamarin: za aplikacije za mobilne uređaje

Sve tri ove platforme imaju svoje vrline i mane, jer su dizajnirane za različite scenarije. To je dovelo do problema da programer mora da nauči tri platforme, od kojih svaka ima neke dosadne „začkoljice" i ograničenja. Zbog toga, „Microsoft" je definisao .NET Standard: specifikaciju za skup API-a koje svaka .NET platforma može da implementira da bi ukazala koji nivo kompatibilnosti ima. Na primer, osnovna podrška ukazuje da je platforma usklađena sa .NET Standardom 1.4.

Za .NET Standard 2.0 i noviji podrazumeva se veći broj nedostajućih API-a u koje programeri treba da prenesu stari kod napisan za .NET Framework u međuplatformski .NET Core. Neki API-i su implementirani, ali podižu izuzetak (generišu grešku), što ukazuje programeru da ne bi trebalo da bude upotrebljen. To se dešava obično zbog razlika u operativnom sistemu na kojem pokrećete .NET Core. U Poglavlju 2, „Govorite C# jezikom", naučićete kako da rukujete ovim izuzecima.

Važno je da razumete da je .NET Standard samo standard. Ne možete da ga instalirate isto kao što ne možete da instalirate HTML5. Da biste upotrebili HTML5, treba da instalirate veb pretraživač koji implemenitira HTML5 standard.

Da biste upotrebili .NET Standard, treba da instalirate .NET platformu koja implementira .NET Standard specifikaciju. .NET Standard 2.0 je implementiran u najnovijim verzijama .NET Frameworka, .NET Corea i Xamarina.

Najnoviji .NET Standard 2.1 je implementiran samo u .NET Coreu 3.0, Monou i Xamarinu. Neke funkcije jezika C# 8.0 zahtevaju .NET Standard 2.1, koji nije implementiran u .NET Framework 4.8, pa bi trebalo da se .NET Framework tretira kao zastareo.



Više informacija: .NET Standard verzije i .NET platforme koje ih podržavaju izlistani su na linku https://github.com/dotnet/standard/blob/master/docs/versions.md.

.NET platforme i alatke upotrebljene u izdanjima ove knjige

Za prvo izdanje ove knjige, koje se iz štampe pojavilo u martu 2016. godine, fokusirao sam se na .NET Core funkcionalnosti, ali sam upotrebio .NET Framework kada u .NET Core još nisu bile implementirane važne i korisne funkcije, jer je to bilo pre finalnog izdanja .NET Core 1.0 verzije. Visual Studio 2015 je upotrebljen za većinu primera, a Visual Studio Code je samo ukratko predstavljen.

Drugo izdanje je bilo skoro potpuno očišćeno od svih primera .NET Framework koda, pa su čitaoci mogli da se fokusiraju na .NET Core primere koji se zaista pokreću međuplatformski. Treće izdanje, u kojem je završeno prebacivanje, ponovo je napisano tako da je ceo kod čist .NET Core. Međutim, instrukcije korak po korak za više alatki dodale su nepotrebnu složenost ovom izdanju.

U četvrtom izdanju nastavljamo trend prikazivanjem primera kodiranja samo upotrebom Visual Studio Codea za skoro sva poglavlja ove knjige (osim za poslednja dva). U Poglavlju 20, „Izgradnja Windows desktop aplikacija“, treba da upotrebite Visual Studio 2019, pokrenut na Windows 10 sistemu, a u Poglavlju 21, „Izgradnja međuplatformskih aplikacija za mobilne uređaje pomoću Xamarin.Formsa“, treba da upotrebite Visual Studio 2019 za Mac.

Razumevanje posredničkog jezika

C# kompajler (pod nazivom Roslyn) koji koristi CLI alatka dotnet konvertuje C# izvorni kod u kod posredničkog jezika (IL) i skladišti IL u sklopu (DLL ili EXE fajlu). Iskazi IL koda su kao instrukcije asemblerskog jezika, koje izvršava virtuelna mašina .NET Corea, poznata kao CoreCLR.

U vreme izvršenja IL kod iz programskog sklopa učitava CoreCLR, pravovremeni (JIT) kompajler ga prevodi u izvorne CPU instrukcije, a zatim ga CPU izvršava na vašoj mašini. Prednost ovog procesa kompajliranja, koji se sastoji od tri koraka, proističe od činjenice da „Microsoft“ može da kreira CLR-ove za Linux i macOS,

kao i za Windows sisteme. Isti IL kod se pokreće svuda zbog drugog procesa kompajliranja, koji generiše kod za izvorni operativni sistem i skup CPU instrukcija.

Bez obzira u kojem jeziku je napisan izvorni kod - na primer, u jezicima C#, Visual Basic ili F#, sve .NET aplikacije koriste IL kod za svoje instrukcije sačuvane u programskom sklopu. „Microsoft" i drugi proizvođači obezbeđuju disasembler alatke koje mogu da otvore programski sklop i otkriju IL kod, kao što je ekstenzija ILSpy .NET Decompiler.

Razumevanje .NET Nativea

Još jedna .NET inicijativa, koja se naziva.NET Native, kompajlira C# kod u originalne CPU **instrukcije unapred (AoT)**, umesto da koristi CLR za kompajliranje IL koda pravovremeno (JIT) u originalni kod kasnije. .NET Native poboljšava brzinu izvršenja i smanjuje upotrebu memorije za aplikacije, jer je originalni kod generisan u vreme izgradnje, a zatim je raspoređen, umesto IL koda.



Više informacija: Možete da pročitate više o .NET Nativeu na linku https://github.com/dotnet/corert/blob/master/Documentation/intro-to-corert.md.

Upoređivanje .NET tehnologija

Možemo da rezimiramo i uporedimo .NET tehnologije u 2019. godini, što je prikazano u sledećoj tabeli:

TEHNOLOGIJA	OPIS	HOST OPERATIVNI SISTEMI
.NET Core	moderan skup funkcija, potpuna C# 8.0 podrška, postojanje porta i kreiranje novih Windows i veb aplikacija i servisa	Windows, macOS, Linux
.NET Framework	stari skup funkcija, ograničena C# 8.0 podrška, održavanje postojećih aplikacija	Samo Windows
Xamarin	samo za aplikacije za mobilne uređaje	Android, iOS, macOS

„Microsoft" je obećao da će krajem 2020. godine postojati jedna .NET platforma, umesto tri. Planirano je da će .NET 5.0 imati jedan BCL i dva izvršenja: jedno optimizovano za server ili desktop scenarija, kao što su veb sajtovi i Windows desktop aplikacije koje su zasnovane na .NET Core izvršenju, i jedno optimizovano za mobilne aplikacije, koje je zasnovano na Xamarin izvršenju.

IZGRADNJA KONZOLNIH APLIKACIJA POMOĆU VISUAL STUDIO CODEA

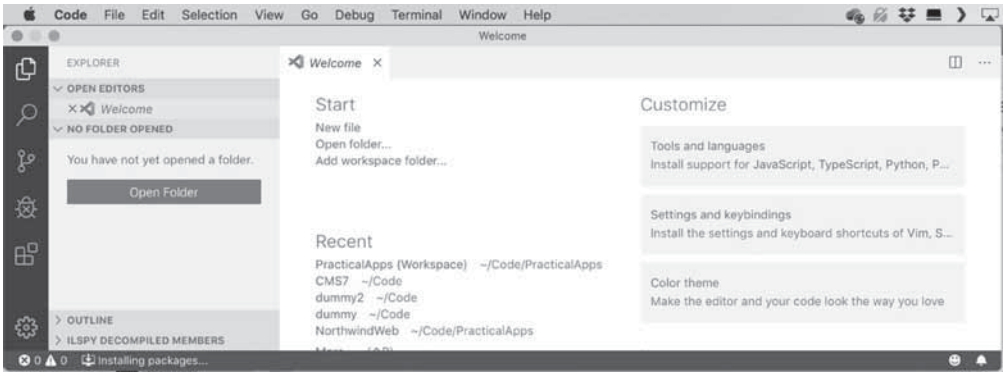
Naš cilj u ovom odeljku je da pokažemo kako da izgradite konzolnu aplikaciju upotrebom Visual Studio Codea. I instrukcije i snimci ekrana u ovom odeljku su za macOS, ali iste akcije će funkcionisati i u Visual Studio Codeu na Windows sistemu i varijantama Linuxa.

Glavne razlike će biti originalne akcije komandne linije, kao što je brisanje fajla: komanda i putanja će verovatno biti drugačije na Windowsu ili macOS i Linux sistemima. Srećom, dotnet alatka komandne linije će biti identična na svim platformama.

Pisanje koda pomoću Visual Studio Codea

Započnite sada pisanje koda:

1. Pokrenite Visual Studio Code.
2. Na macOS sistemu kliknite na **File | Open.**.. ili pritisnite Cmd + O. Na Windows sistemu kliknite na **File | Open Folder**… ili pritisnite Ctrl + K Ctrl + O. U oba Osea možete da kliknete na dugme **Open Folder** u **EXPLORER** panelu ili kliknite na **Open Folder…** link na Welcome stranici, kao što je prikazano na sledećoj slici:



3. U okviru za dijalog otvorite vaš korisnički direktorijum na macOS-u (moj direktorijum je markjprice), selektujte direktorijum Documents na Windowsu ili bilo koji direktorijum ili drajv na koji želite da snimite projekte.
4. Kliknite na dugme New Folder i direktorijumu dodelite naziv Code.
5. U direktorijumu Code kreirajte novi direktorijum, pod nazivom Chapter01.
6. U direktorijumu Chapter01 kreirajte novi direktorijum, pod nazivom HelloCS.

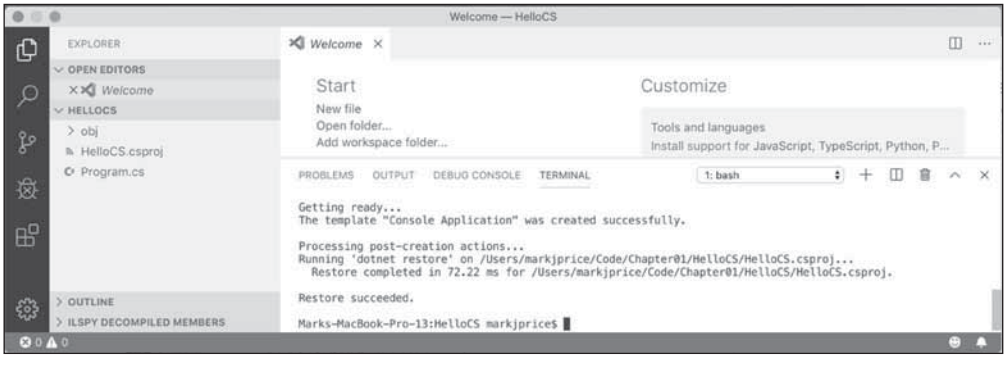
7. Selektujte HelloCS direktorijum na macOS-u i kliknite na **Open** ili na Windowsu kliknite na **Select Folder**.

8. Kliknite na **View | Terminal** ili na macOS-u pritisnite Ctrl + ` (obrnuti jednostruki navodnik), a na Windowsu pritisnite Ctrl + , (jednostruki navodnik). Zbunjujuće je što na Windows sistemu kombinacija tastera Ctrl + , (obrnuti jednostruki navodnik) razdvaja aktuelni prozor!

9. U TERMINAL unesite sledeću komandu:

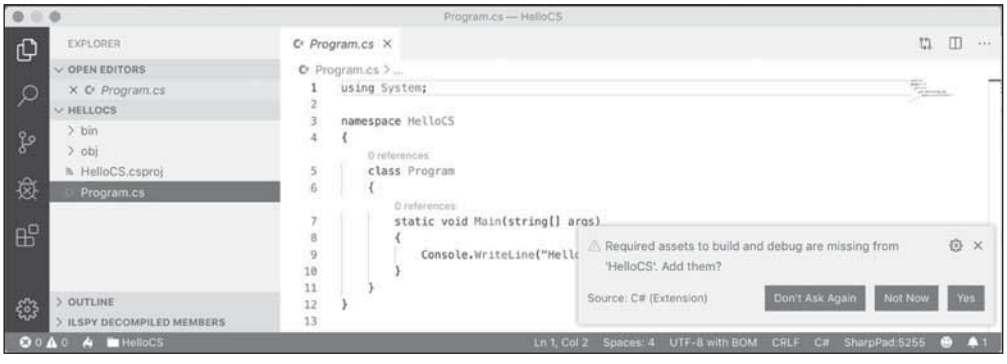
```
dotnet new console
```

10. Videćete da alatka komandne linije dotnet kreira novi projekat **Console Application** u aktuelnom direktorijumu, a prozor **EXPLORER** prikazuje dva kreirana fajla HelloCS.proj i Program.cs, kao što je prikazano na sledećoj slici:



11. U EXPLORER-u kliknite na fajl pod nazivom Program.cs da biste ga otvorili u prozoru editora. Kada to prvi put budete radili, možda će biti potrebno da Visual Studio Code preuzme i instalira C# zavisnosti, kao što su OmniSharp, Razor Language Server i .NET Core debugger, ako to nije uradio kada ste instalirali C# ekstenziju.

12. Ako vidite upozorenje da nedostaju potrebni elementi, kliknite na **Yes**, kao što je prikazano na sledećoj slici:

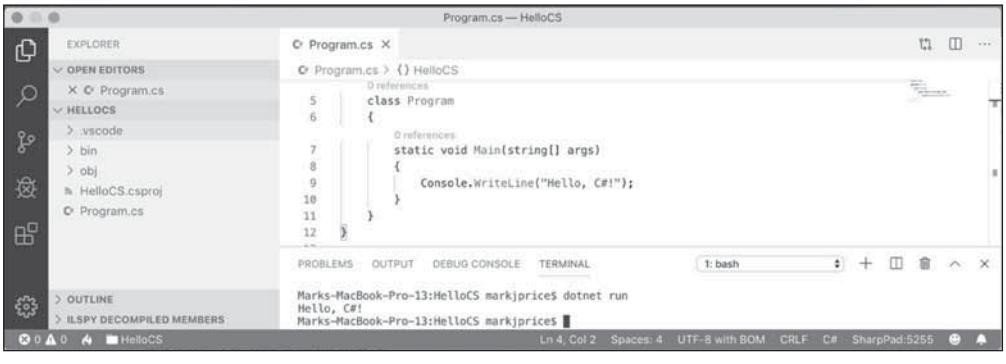


13. Nakon nekoliko sekundi, direktorijum pod nazivom .vscode će biti prikazan panelu EXPLORER-a. Ovaj direktorijum se koristi u toku ispravljanja grešaka, kao što ćete naučiti u Poglavlju 4, „Pisanje, ispravljanje grešaka i testiranje funkcija“.
14. U fajlu Program.cs modifikujte liniju 9 da tekst napisan u konzoli glasi Hello, C#!
15. Kliknite na **File | Auto Save**. Ova opcija će vam pomoći, jer ne morate pamtiti da treba da snimate snimite fajl pre nego što ponovo izgradite aplikaciju.

Kompajliranje i pokretanje koda pomoću dotnet CLI alatke

Sledeći zadatak je da kompajlirate i pokrenete kod.

1. Kliknite na View | Terminal i unesite sledeću komandu:
2. Rezultat u prozoru TERMINAL-a će prikazati rezultat pokretanja aplikacije, kao što je prikazano na sledećoj slici:



PREUZIMANJE KODA REŠENJA IZ GITHUB SKLADIŠTA

Git je sistem za upravljanje izvornim kodom koji se najčešće koristi. GitHub je kompanija, veb sajt i desktop aplikacija koja olakšava upravljanje Gitom. „Microsoft“ je nedavno kupio GitHub, pa će biti nastavljena bliža integracija sa Microsoft alatkama.

Ja sam upotrebio GitHub za skladištenje rešenja za sve praktične vežbe koje se nalaze na kraju svakog poglavlja na URL-u https://github.com/markjprice/cs8dotnetcore3.

Upotreba Gita sa Visual Studio Codeom

Visual Studio Code ima podršku za Git, ali će upotrebiti Git instalaciju vašeg OS-a, pa prvo morate da instalirate Git 2.0 ili noviju verziju, pre nego što dobijete ove funkcije. Možete da instalirate Git sa linka https://git-scm.com/download.

Ako želite da koristite GUI, možete da preuzmete GitHub Desktop sa linka https://desktop.github.com.

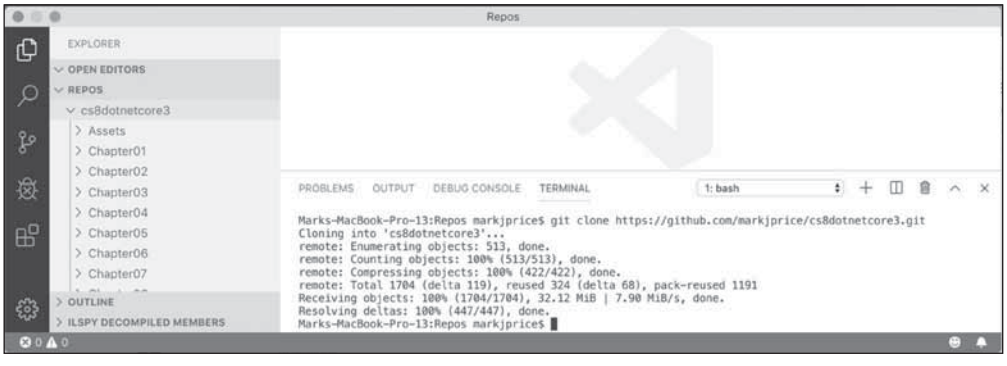
Kloniranje skladišta koda rešenja ove knjige

Sada ćemo klonirati skladište koda rešenja ove knjige.

1. Kreirajte direktorijum, pod nazivom Repos, u vašem korisničkom direktorijumu, ili direktorijum Documents ili u bilo kojem direktorijumu u kojem želite da sačuvate Git skladišta.
2. U Visual Studio Codeu otvorite direktorijum Repos.
3. Kliknite na **View | Terminal** i unesite komandu

   ```
   git clone https://github.com/markjprice/cs8dotnetcore3.git.
   ```

4. Kloniranje svih rešenja za sva poglavlja će potrajati nekoliko minuta, kao što je prikazano na sledećoj slici:





Više informacija: Za više informacija o upravljanju verzijama izvornog koda pomoću Visual Studio Codea pogledajte link https://code.visualstudio.com/Docs/editor/versioncontrol

POTRAŽITE POMOĆ

U ovom odeljku pronaći ćete savete kako da pronađete kvalitetne informacije o programiranju na Vebu.

Čitanje Microsoft dokumentacije

Definitivni resurs za dobijanje pomoći za Microsoft programerske alatke i platforme nekada je bio **Microsoft Developer Network (MSDN)**. Sada je to **Microsoft Docs**, koji možete da pronađete na linku https://docs.microsoft.com/.

Dobijanje pomoći za dotnet alatku

U komandnoj liniji možete da zatražite pomoć za opis komandi dotnet alatke.

1. Da biste otvorili zvaničnu dokumentaciju u prozoru pretraživača za dotnet new komandu, unesite u komandnu liniju ili u Visual Studio Code Terminal sledeće:

   ```
   dotnet help new
   ```

2. Da biste dobili pomoćni ispis u komandnoj liniji, upotrebite -h ili --help oznaku, kao što je prikazano u sledećoj komandi:

   ```
   dotnet new console -h
   ```

3. Videćete sledeći delimičan ispis:

   ```
   Console Application (C#)
   Author: Microsoft
   Description: A project for creating a command-line application
   that can run on .NET Core on Windows, Linux and macOS
   Options:
     --langVersion  Sets langVersion in the created project file
                    text - Optional
     --no-restore   If specified, skips the automatic restore of
   the project on create.
                    bool - Optional
                    Default: false / (*) true
   * Indicates the value used if the switch is provided without
   a value.
   ```

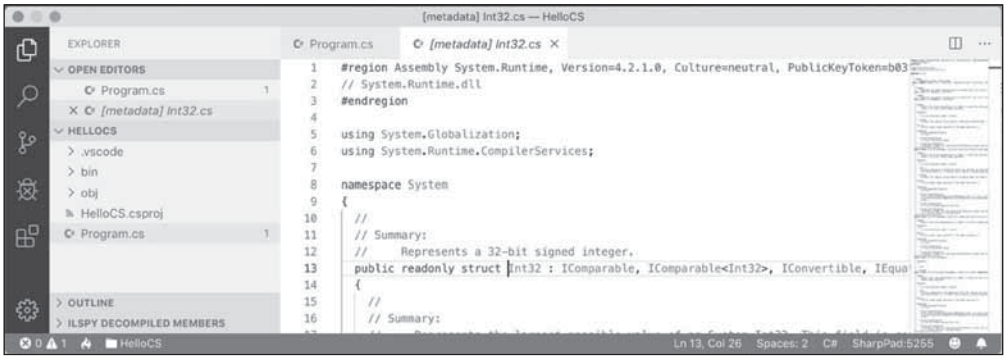
Preuzimanje definicija tipova i njihovih članova

Jedna od najkorisnijih prečica sa tastature u Visual Studio Codeu je F12 za Go To **Definition**. Ona prikazuje čitanjem metapodataka u kompajliranom programskom sklopu kako izgleda javna definicija tipa ili člana. Neke alatke, kao što je ILSpy .NET Decompiler, izvršiće za vas čak povratnu analizu iz metapodataka i IL koda nazad u C#.

1. U Visual Studio Codeu otvorite direktorijum HelloCS.
2. Da biste deklarisali promenljivu celog broja pod nazivom z:, u fajl Program.cs, unutar metoda Main, unesite sledeći iskaz:

   ```
   int z;
   ```

3. Kliknite unutar deklaracije int, a zatim pritisnite F12, ili kliknite desnim tasterom miša i izaberite **Go To Definition**. U novom prozoru koda koji će biti prikazan moći ćete da vidite kako je definisan tip podataka int, kao što je prikazano na sledećoj slici:



Možete da vidite da je int:

- definisan upotrebom ključne reči struct
- da se nalazi u programskom sklopu System.Runtime
- da se nalazi u imenskom prostoru System
- da mu je naziv Int32
- da je, prema tome, alijas za System.Int32 tip
- da implementira interfejse, kao što je IComparable
- da ima konstantne vrednosti za svoju maksimalnu i minimalnu vrednost
- da ima metode, kao što je Parse

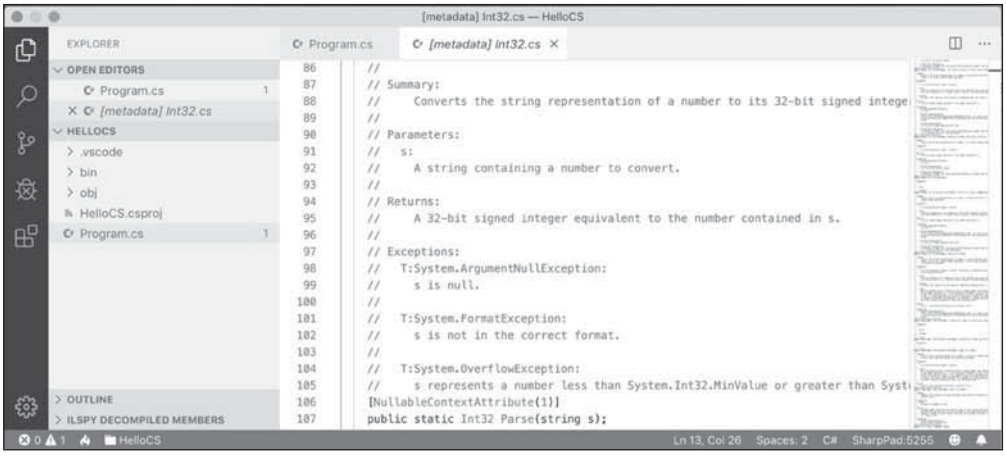


Dobra praksa: Kada pokušate da upotrebite opciju Go To Definition, ponekad ćete videti grešku No definition found. Razlog je činjenica da C# ekstenzija ne poznaje aktuelni projekat. Kliknite na View | Command Palette, unesite i selektujte OmniSharp: Select Project, a zatim selektujte odgovarajući projekat koji želite da upotrebite.

Sada funkcija Go To Definition nije mnogo korisna, jer još ne znate šta ovi termini znače.

Do kraja prvog dela ove knjige, u kojem ćete učiti o C# jeziku, imaćete mnoštvo informacija, pa će vam ova funkcija postati veoma korisna.

4. U prozoru editora koda skrolujte nadole da biste pronašli Parse metod sa jednim string parametrom, koji počinje u liniji 86, kao što je prikazano na sledećoj slici:



U komentaru ćete videti da je „Microsoft“ dokumentovao koji izuzeci se mogu javiti ako pozovete ovaj metod, uključujući ArgumentNullException, FormatException i OverflowException. Sada znate da treba da obuhvatite poziv za ovaj metod u try iskaz i koje izuzetke treba da hvatate.

Pretpostavljam da ste nestrpljivi da naučite šta sve ovo znači!

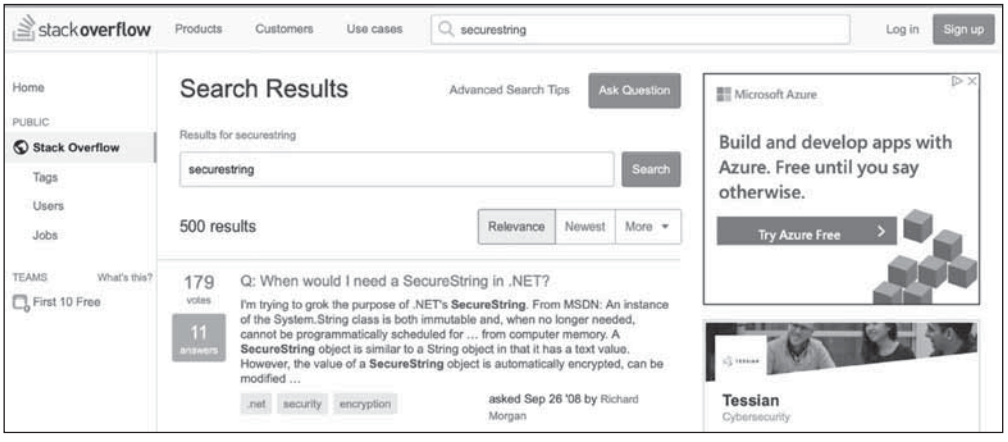
Budite još malo strpljivi. Skoro ste na kraju ovog poglavlja, a u sledećem ćete detaljnije upoznati C# jezik. Međutim, prvo pogledajte gde još možete potražiti pomoć.

Traženje odgovora u Stack Overflowu

Stack Overflow je najpopularniji nezavisni veb sajt za dobijanje odgovora na teška programerska pitanja. Toliko je popularan da pretraživači, kao što je **DuckDuckGo**, imaju specijalan način za pisanje upita za njegovu pretragu.

1. Otvorite vaš omiljeni veb pretraživač.
2. Otvorite stranicu DuckDuckGo.com, unesite sledeći upit i vidite rezultate pretrage, koji su prikazani na sledećoj slici:

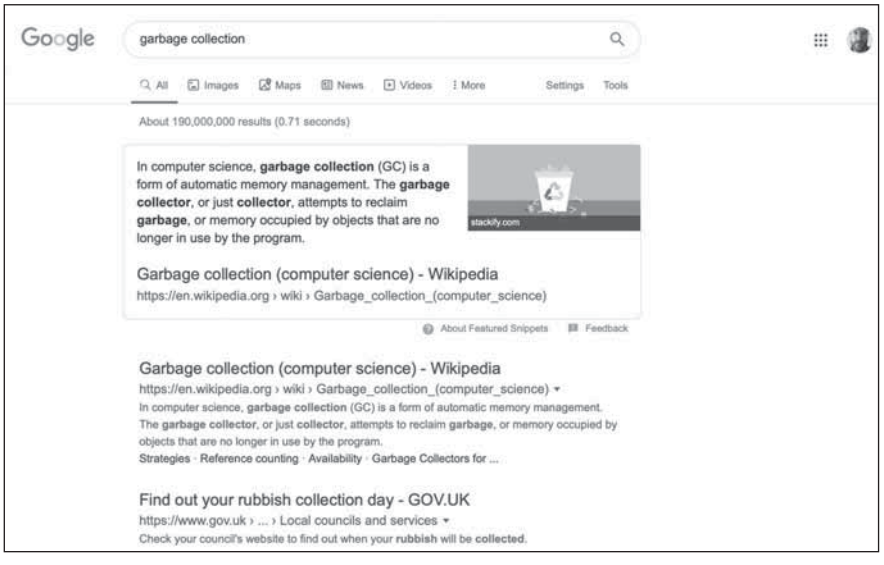
```
!so securestring
```



Pretraživanje odgovora upotrebom Googlea

Možete da pretražujete **Google**, koristeći napredne opcije pretrage, da biste povećali verovatnoću pronalaženja onoga što tražite.

1. Otvorite Google.
2. Potražite informacije za garbage collection, koristeći jednostavan Google upit, i videćete verovatno definiciju na Wikipedia sajtu za garbage collection u računarskoj nauci, a zatim listu garbage collection servisa u vašoj lokalnoj oblasti, kao što je prikazano na sledećoj slici:



3. Poboljšajte pretragu, tako što ćete je ograničiti na koristan sajt, kao što je Stack Overflow, i tako što ćete ukloniti jezike koji vas ne zanimaju, kao što su C++, Rust i Python, ili dodati eksplicitni jezik C# i .NET platforme, kao što je prikazano u sledećem upitu:

```
garbage collection site:stackoverflow.com +C# -Java
```

Pretplata na zvanični .NET blog

Da biste pratili zbivanja u vezi .NET-a, odličan blog na koji možete da se prijavite je zvanični .**NET Blog**, koji su pisali timovi .NET inženjera, a možete da ga pronađete na linku https://blogs.msdn.microsoft.com/dotnet/.

VEŽBANJE I ISTRAŽIVANJE

Testirajte svoje znanje, tako što ćete odgovoriti na neka pitanja, uraditi praktične vežbe i istražiti detaljno teme koje su opisane u ovom poglavlju.

Vežba 1.1 – Testirajte svoje znanje

Pokušajte da odgovorite na sledeća pitanja (većinu odgovora možete da pronađete u ovom poglavlju, a za druge odgovore će biti potrebni istraživanje na Internetu ili pisanje koda):

1. Zašto programer može da upotrebi različite jezike (na primer, C# i F#) za pisanje aplikacija koje se pokreću na .NET Coreu?
2. Šta treba da ukucate u odzivnik da biste kreirali konzolnu aplikaciju?
3. Šta treba da ukucate u odzivnik da biste izgradili i izvršili C# izvorni kod?
4. Koja je prečica na tastaturi Visual Studio Codea za prikaz Terminala?
5. Da li je Visual Studio 2019 bolji od Visual Studio Codea?
6. Da li je .NET Core bolji od .NET Frameworka?
7. Šta je .NET Standard i zašto je važan?
8. Koji je naziv metoda za unos .NET konzolne aplikacije i kako bi trebalo da bude deklarisan?
9. Gde ćete potražiti pomoć o C# ključnim rečima?
10. Gde ćete potražiti rešenja za uobičajene programerske probleme?

Vežba 1.2 – Vežbajte C# svuda

Nisu vam potrebni Visual Studio Code, ili, čak, Visual Studio 2019 ili Visual Studio 2019 for Mac da biste pisali C# kod. Možete da otvorite stranicu .NET Fiddle https://dotnetfiddle.net/ i da započnete kodiranje online.

„Microsoft" je takođe pripremio online verziju Visual Studio Codea, koja može da se pokreće u bilo kom pretraživaču, ali je trenutno dostupna samo za privatni pregled (radi testiranja). Uskoro će biti dostupna svima na linku https://online.visualstudio.com/

Vežba 1.3 – Istražite teme

Da biste pročitali više detalja o temama koje su opisane u ovom poglavlju, upotrebite sledeće linkove:

- **Welcome to .NET Core**: http://dotnet.github.io
- .**NET Core Command-Line Interface (CLI) tool: https**://aka.ms/dotnetcli-docs
- .**NET Core runtime, CoreCLR**: https://github.com/dotnet/coreclr/
- .**NET Core Roadmap**: https://github.com/dotnet/core/blob/master/roadmap.md
- .**NET Standard FAQ**: https://github.com/dotnet/standard/blob/master/docs/faq.md
- **Stack Overflow**: http://stackoverflow.com/
- **Google Advanced Search**: http://www.google.com/advanced_search
- **Microsoft Virtual Academy**: https://mva.microsoft.com/
- **Microsoft Channel 9**: Developer Videos: https://channel9.msdn.com/

REZIME

U ovom poglavlju podesili smo radno okruženje, opisali smo razlike između .NET Frameworka, .NET Corea, Xamarina i .NET Standarda, upotrebili smo Visual Studio Code i .NET Core SDK za kreiranje jednostavne konzolne aplikacije i naučili ste kako da preuzmete kod rešenja za ovu knjigu iz GitHub skladišta i, što je najvažnije, kako da pronađete pomoć.

U sledećem poglavlju ćete naučiti da govorite C# jezikom.