Sadržaj

Uvod **1**

Prvi deo

Osnove **5**

1. Internet 7
   - Internet mreža 7
   - Intranet 11
   - Adrese računara 12
     - Privatne adrese 13
   - Domenska imena 14
   - Portovi 15
   - Identifikatori resursa (URI) 16
     - URI parametri 18
     - URI fragmenti 18
   - HTTP protokol 19
     - HTTP metode 22
     - HTTP statusi 25
   - Sigurnost 26
     - Serverski sertifikati 28
     - Ključevi 29
     - Šifrovanje saobraćaja 30
     - Preuzimanje identiteta 31
   - Zaključak 32

2. HTML i struktura informacija 33
   - HTML jezik 33
   - Tekstualne strukture 34
   - Kreiranje veb-strana 35
   - Atributi HTML elemenata 37
   - Hipertekst 38
   - Složene strukture podataka 40
   - Multimedijalni sadržaji 42
   - HTML formulari 43
   - Blokovi 45
   - Zaključak 46

3. Stilizovanje strana pomoću CSS pravila 47
CSS pravila 47
Stilovi 49
CSS selektori 50
Raspored elemenata 52
Proširenja CSS jezika (Less i SaSS) 56
Stilizovanje Twitter Bootstrap bibliotekom 58
Stilovi 59
Raspored elemenata 60
Komponente 61
Zaključak 63

4. JavaScript 65
Osnove JavaScript jezika 67
Model dokumenta (DOM) 70
Obrada događaja 71
JavaScript biblioteke 73
Moduli – savremene JavaScript biblioteke 75
Komponente 76
Tabele (JQuery DataTable) 78
Stabla (jsTree) 79
Dijalozi (Bootstrap, JQuery UI) 80
Zaključak 83

Drugi deo

Serverske veb-aplikacije 85

5. Serverske veb-aplikacije 87
Razvoj veb-aplikacija 88
Veb-serveri 91
Node.js veb-server 92
ASP.Net veb-server 95
Serveri statičkih veb-strana 96
Nginx server 98
Node.js server statičkih strana 99
ASP.NET server statičkih strana 100
Serviranje dinamičkog sadržaja 101
Rutiranje HTTP zahteva 102
Podaci u veb-aplikacijama 105
Šablonski prikazi 116

CMS aplikacije 121
Pristup bazama podataka 124
Dapper.NET 126
Entity frejmvork 128
Arhitekture serverskih veb-aplikacija 129
Troslojna arhitektura veb-aplikacija 129
Kontroler prikaza 132
Dizajn zasnovan na komponentama 134
Dizajn zasnovan na modelima 146
Keširanje podataka 155
Zaključak 158

6. Veb-servisi 161
Implementacija veb-servisa 163
OData 167
GraphQL 169
Opisivanje interfejsa veb-servisa 170
Sigurnost veb-servisa 172

7. Veb-strane sa prezentacionom logikom 173
Implementacija naprednih JavaScript funkcionalnosti 173
Šablonski prikazi u JavaScript kodu 175
Objektni model dokumenta strane (DOM) 176
Vue 177
React 180
Angular 181
Povezivanje podataka 182
JavaScript prezentacione komponente 186
Eksterne biblioteke 187
Vue komponente 189
React komponente 190
Angular komponente 193
Asinhrona obrada zahteva 195
Obećanja 196
Posmatrači 197
Asinhrono učitavanje podataka (AJAX) 198
Veb-utičnice (WS) 202
Implementacija veb-utičnica u .NET tehnologiji (SignalR) 204

Treći deo

Implementacija klijentskih veb-aplikacija 209

8. Klijentske JavaScript aplikacije (SPA) 211
Projektno okruženje 212
Upravljanje komponentama 214
Proširenja komandnog interfejsa 215
Arhitektura klijentskih aplikacija 218
Angular komponentna arhitektura 219
Komponente 220
Servisi 221
Arhitekture zasnovane na stanju 223
Linearni tok kontrole (Flux frejmvork) 224
Moderne arhitekture zasnovane na stanju (Redux, NgRx) 228
Rutiranje 231
Zaključak 233

9. Aplikacije učitane u pregledaču 237
Web Assembly (WASM) 237
Projekti 239
Blazor 241
Zaključak 244

10. Naredni koraci 245

Uvod

Želite li da se bavite veb-programiranjem? Ako biste upitali nekoga kako da naučite veb-programiranje, odgovor bi verovatno bio: „Pa, prvo nauči HTML – W3Schools sajt je odličan za to. Posle toga pogledaj i CSS i JavaScript, W3Schools je takođe odlično mesto za to." Ovo nije pogrešan savet. Bez poznavanja jezika kao što su HTML, CSS i JavaScript se ne može zamisliti veb-programiranje. Međutim, postoji zamka u koju mnogi upadnu. Postoji mogućnost da ćete početi da učite što je moguće više HTML elemenata i CSS pravila, ali vam možda neće biti jasno zašto to radite i da li je to stvarno pravi put.

Dok budete na ovaj način učili veb-programiranje, videćete da drugi počinju da rade sa nekim aplikacijama kao što je WordPress i prave odlične sajtove na kojima zarađuju bez muka kroz koje vi prolazite – to vas može navesti i da odustanete od saveta prijatelja. Čak i ako naučite većinu HTML elemenata i CSS pravila i prođete razne vežbe, možete primetiti da i dalje ne možete da napravite dobar veb-sajt kao neko ko koristi WordPress, a možda ne zna veliki broj elementa koje ste vi naučili. Posle toga možete da pogledate šta rade pravi veb-programeri i shvatite da se često koristi nešto što se zove Bootstrap, koji omogućava da naprave kvalitetnije strane bez poznavanja desetina CSS pravila koje ste naučili. Da li to znači da je učenje CSS-a bio loš savet?

Savladali ste JavaScript i onda videli da svi koriste JQuery (mada većina kaže da je i on zastareo). Zašto niste odmah učili JQuery? Posle nekog vremena ćete naći odgovore na ova pitanja i shvatiti šta treba da učite od ovih tehnologija. Međutim, problem je u tome što i dalje sve to nije dovoljno za veb-programiranje. Tada shvatite da posle svih tih HTML, CSS, JavaScript jezika i biblioteka kao Bootstrap ili JQuery morate da naučite nešto kao što je PHP, ASP.NET ili Node.js kako biste napravili veb-aplikaciju. Ali ni to nije kraj – ubrzo shvatate da se moderno veb-programiranje svodi na programiranje bogatih klijentskih aplikacija pomoću JavaScript-a i biblioteka ili frejmvorka kao što su Angular, React ili Vue. Da li su onda PHP i ASP.NET bili pogrešan pravac?

Kada bi mene pitali kako se uči veb-programiranje, rekao bih: „Najbolji način da NE naučite veb-programiranje je da naučite što je moguće više HTML, CSS i ostalih preporučenih veb-tehnologija i jezika."

Veb-aplikacije su distribuirani i heterogeni sistemi koji se sastoje od velikog broja komponenti i tehnologija implementiranih na različitim jezicima na pregledaču i veb-serveru, a često uključuju i baze podataka. Ako počinjete da učite programiranje, nije dobra ideja da učite samo jedan ili dva jezika ili tehnologije i pokušavate da naučite sve detalje o njima. Na početku učenja veb-programiranja je potrebno da se upoznate sa što je moguće više različitih koncepata kako biste shvatili gde kasnije želite da se specijalizujete.

Nekima će odgovarati da se specijalizuju u HTML i CSS jezicima i postanu veb-dizajneri bez potrebe da znaju programske jezike. Drugi opet više žele da samo programiraju bez ulaženja u detalje prikazivanja strane. Neko će želeti da implementira funkcionalnosti na serverskoj strani (engl. *backend*) pomoću neke tehnologije kao što je ASP.NET ili Node.js i možda će imati samo minimalno znanje HTML jezika. Neko će želeti da radi isključivo sa ASP.NET ili Node.js programskim kodom, a neko će se baviti i pisanjem upita u bazi podataka. Neki od vas će implementirati aplikacije samo u pregledaču (engl. *frontend*) i morati dobro da shvate JavaScript, ali i HTML i CSS, mada ih neće zanimati previše kako se implementira serverski kod a pogotovo ne baze podataka. Biće i nekih koji će želeti da rade sve od pregledača do baze podataka (engl. *full-stack*), ali opet neće naučiti sve tehnologije u svim slojevima aplikacije do najsitnijih detalja.

Kao što vidite, veb-programiranje može da bude veoma složeno sa različitim ulogama i oblastima (veb-dizajn, *frontend*, *backend*, *full-stack*) i ne bi trebalo da pitate kako da ga naučite. Pravo pitanje treba da bude:

Kako da shvatim šta je veb-programiranje i koja uloga mi najviše odgovara?

Jednom kada shvatite šta sve obuhvata veb-programiranje bez ulaženja u najsitnije detalje svakog jezika, shvatićete koja uloga vam najviše odgovara. Tada ćete se lakše odlučiti koji su to jezici kojima ćete produbiti svoje znanje i u kojima ćete se specijalizovati.

Ova knjiga vam upravo to i omogućava. U ovoj knjizi ćete naučiti ono što je najbitnije iz svih oblasti veb-programiranja. Naučićete šta se dešava kada pregledač pošalje zahtev ka nekom veb-serveru, kako veb-server obrađuje zahtev i kako se prikazuju podaci u pregledaču. U knjizi ćemo proći kroz sve oblasti koje su potrebne

u veb-programiranju, ali bez ulaženja u detalje. U svakoj oblasti ćete naučiti samo ono što je neophodno kako biste shvatili zašto vam je bitna.

Ova knjiga je osnova za vašu karijeru. Nakon što se upoznate sa svim aspektima veb-programiranja i odaberete svoju buduću ulogu, lako ćete odabrati prave materijale koji će vam pomoći da postanete veb-dizajner, *frontend*, *backend* ili *full-stack* inženjer. U ovoj knjizi ćete steći fundamentalna znanja koja su preduslov za donošenje ovakve odluke:

1. U prvoj glavi ćete naučiti osnove interneta i shvatiti kako se vrši interakcija između pregledača i servera, kako izgleda HTTP protokol, koje podatke šalje pregledač, a koje informacije server vraća. Pored toga ćete se upoznati sa jezicima koji se koriste kako bi se u pregledaču prikazale veb-strane, što uključuje:
    a. Osnove HTML jezika gde ćete se upoznati sa najbitnijim elementima HTML jezika koji vam omogućavaju da definišete pasuse, zaglavlja, tabele, liste, slike, multimedijalni sadržaj i druge često korišćene elemente koji se prikazuju na veb-stranama.
    b. CSS stilovi će vam pokazati kako se stilizuju i formatiraju veb-strane.
    c. Osnove JavaScript jezika, sa kojim ćete naučiti kako da reagujete na događaje koje izazove korisnik u pregledaču i implementirate interakciju sa korisnikom. Ovo je osnova za implementaciju bilo koje funkcionalnosti na modernim veb-stranama i nešto što morate naučiti ako se bavite veb-programiranjem.
2. U drugoj glavi ćete naučiti kako se implementiraju veb-aplikacije koje na serverskoj strani generišu HTML, CSS i JavaScript kod koji će biti učitan u pregledač. U ovoj glavi ćete naučiti kako da napravite veb-aplikacije pomoću ASP.NET i Node.js koje mogu da prihvate zahtev koji dolazi od pregledača i vrate odgovor koji će pregledač prikazati. Pored klasičnih veb-aplikacija generisanih na serverskoj strani, naučićete šta su veb-servisi, kao i kako se implementiraju veb-aplikacije sa naprednim korisničkim interfejsom na klijentskoj strani koje su dalje generisane na serverskoj strani.
3. U trećoj glavi ćete naučiti kako se implementiraju veb-aplikacije koje se učitavaju u pregledač i samo po potrebi kontaktiraju serversku stranu. Ovde ćete naučiti kao rade moderne JavaScript biblioteke kao što su Angular, React i Vue, ali i neki jezici kao Rust/WASM i klijentski C#/Blazor, koji nisu inicijalno bili namenjeni razvoju korisničkog interfejsa u pregledaču, ali vam

omogućavaju da napravite napredne korisničke interfejse veb-aplikacija bez upotrebe „klasičnih jezika interneta".

Iako će se u ovim poglavljima govoriti o „osnovama", nemojte se zavaravati i pomisliti da su to samo neke jednostavne stvari za početnike. Informacije koje ćete pronaći u ovoj knjizi su verovatno nešto što možete očekivati na intervjuima za posao bez obzira na to za koju ulogu se odlučite pošto su ovo neka opšta znanja o veb-programiranju koja morate imati. Pored toga, opšte poznavanje različitih oblasti veb-programiranja vam je neophodno kako biste razumeli šta vaše kolege u drugim ulogama tačno rade i, što je još bitnije, da vidite da li biste možda želeli u nekom trenutku da pređete iz svoje uloge u neku drugu. Da biste imali uspešnu karijeru, morate uvek da učite i o stvarima koje nisu u vašem užem domenu kako biste bili spremni na promene koje će na vas uticati, a uzrokovane su promenom tehnologije i načina rada drugih timova.

Kada jednom razumete osnove, lako ćete naći detaljnije materijale kojima ćete produbiti znanje. Na internetu već postoji ogromna količina materijala pomoću kojih možete produbiti znanje o ovim oblastima. Ova knjiga će vam pružiti ono što je najbitnije za početak – pravce u kojima se možete kretati i oblasti koje bi trebalo istraživati.

Prvi deo

Osnove

U prvom delu knjige ćemo se upoznati sa osnovama koje morate da znate kako biste se bavili razvojem veb-aplikacija. Osnove predstavljaju principe rada internet mreže, HTML, CSS i JavaScript jezika koji će vam biti neophodni bez obzira na to čime se bavite. Ovo su verovatno informacije koje će vam bilo ko preporučiti kao prvi korak koji morate da učinite kako biste počeli da se bavite razvojem veb-aplikacija. Međutim, ovo nisu opširni i detaljni materijali koje biste mesecima čitali dok ne postanete eksperti za HTML, CSS ili JavaScript. Svrha ovog poglavlja je da kroz desetak strana o internetu, HTML, CSS i JavaScript jeziku naučite šta je najbitnije u ovim tehnologijama i jezicima kako biste znali u kojem pravcu dalje treba da se usavršavate.

Ako nemate iskustva sa razvojem veb-aplikacija, ovaj deo knjige je za vas verovatno najbolji početak. U slučaju da već znate ove koncepte, možda ćete biti u iskušenju da preskočite ovaj deo knjige ili bar da brže pređete preko ovih poglavlja. Međutim, imajte na umu da je, bez obzira na prethodno znanje, veoma bitno da pažljivo prođete ova poglavlja kako biste bili sigurni da ne postoji nešto od osnovnih koncepata koje ste propustili, a koji će vam biti bitni kasnije, kada budete učili o serverskim i klijentskim aplikacijama.

Prvo ćemo proći kroz neophodne koncepte internet komunikacije. Cilj ovog poglavlja nije da postanete „mrežni administratori", već da se upoznate sa konceptima koji će vas pratiti dok razvijate veb-aplikacije. Potom ćemo se upoznati sa formatima i jezicima koji se koriste kako bi se prikazale informacije u pregledačima, što uključuje:

1. HTML sadržaj koji definiše informacije koje treba prikazivati na strani, kao i njihovu strukturu.
2. CSS pravila koja definišu kako se stilizuju, raspoređuju i prikazuju elementi na stranama.
3. Multimedijalni sadržaj (slike, audio i video materijali) koji se prikazuje na stranama.
4. JavaScript programski kod koji obezbeđuje funkcionalnosti na strani koja je učitana u pregledaču.

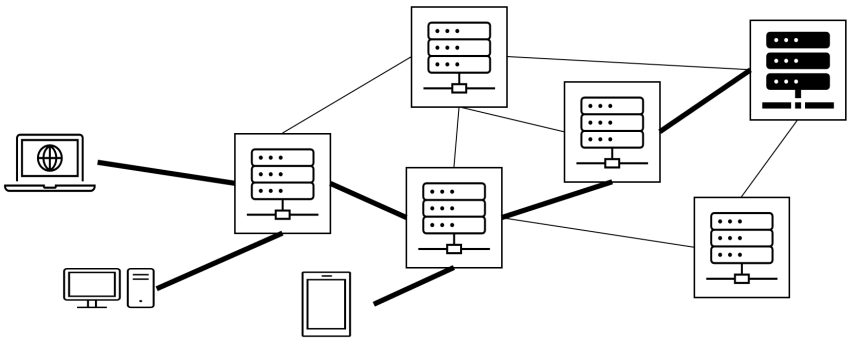
Jezici i tehnologije sa kojima ćemo se upoznati u ovom delu knjige su gradivni elementi kojima se može napraviti bilo koji napredni veb-sajt sa organizovanom strukturom informacija, stilizovanim elementima, i funkcionalnostima koje vrše interakciju sa korisnikom.

1. Internet

Ovu knjigu ćemo početi sa pričom o internetu. Poznavanje osnovnih koncepta internet infrastrukture je neophodno kako biste mogli da rešavate probleme koji će vam se dešavati u komunikaciji između pregledača i veb-aplikacija. Često ćete morati da utvrdite uzroke problema u vašim veb-aplikacijama i biće mnogo pitanja na koje treba naći odgovor koji leži u detaljima internet protokola. Zašto se veb-sajt učitava sporo samo ljudima iz Amerike? Šta to znači da je neko presreo zahtev koji je pregledač poslao? Zašto pregledač preuzima 100 MB sadržaja umesto 5 MB kompresovane verzije? Kako to da neko ne može da pristupi mom sajtu, a ja mogu? Da biste odgovorili na ova pitanja, potrebno je da znate malo više od osnova interneta. U ovom poglavlju ćete naučiti najbitnije i neophodne koncepte kako biste razumeli sve što vam je potrebno da naučite kako rade veb-aplikacije.

Internet mreža

Internet predstavlja osnovnu infrastrukturu u kojoj se implementiraju distribuirane veb-aplikacije. Internet je velika informaciona mreža koja omogućava računarima da razmenjuju informacije.



Slika 1.1. Internet – informaciona mreža koje povezuje geografski udaljene računare i uspostavlja putanje među njima.

Internet obezbeđuje konekcije kojima se prenose podaci, metode za lokaciju računara na mreži i komunikacione protokole za prenos podataka među njima. Svrha interneta je da identifikuje rutu između računara koji razmenjuju podatke i da omogući komunikaciju među njima tako što računari na ruti prosleđuju dalje informacije koje se šalju.

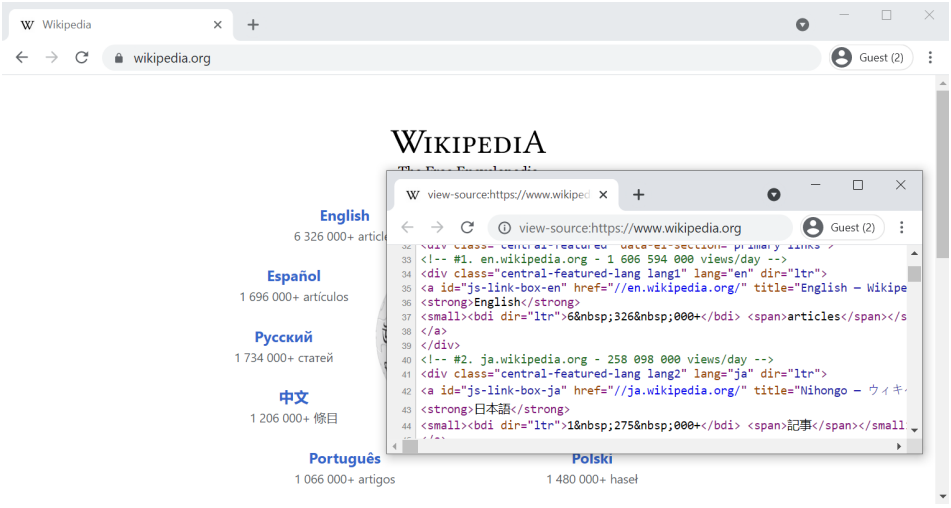
Računari putem interneta traže neke informacije koji treba prikazati. Na primer, korisnici laptopova pristupaju veb-stranama pomoću pregledača kao što su Chrome,

Edge ili Firefox. Korisnik u suštini traži od nekog veb-sajta sadržaj strane koju će pregledač da prikaže. S druge strane, postoje računari koji imaju te informacije i koji će ih poslati kada ih neko zatraži. Internet omogućava programima koji se nalaze na ovim udaljenim računarima da razmenjuju te informacije. Računari na internet mreži se mogu logički podeliti u sledeće grupe:

- **Klijenti** su računari na kojima se nalaze programi koji traže neke informacije. Ovo mogu da budu pregledači, imejl klijenti i slično.
- **Serveri** su računari na kojima se nalaze programi ili servisi[1] koji obezbeđuju informacije koje traže klijenti. Informacije mogu biti strane veb-sajtova, fajlovi, imejl poruke i slično.

Treba imati na umu da mnogi računari ne moraju da budu isključivo klijenti ili serveri i da na njima mogu da se nalaze neki klijentski programi, ali i neki koji serviraju informacije koje traže drugi klijenti.

Internet omogućava da se programi na klijentima i serverima međusobno povežu i razmene informacije. Na primer, pregledačem možete da pristupite veb-serveru koji predstavlja sajt Vikipedija, i on će vratiti neki sadržaj koji predstavlja neku Vikipedijinu stranu. Tekst koji vraća veb-server možete lako da vidite korišćenjem opcije pretraživača kojom se prikazuje izvorni tekst koji dolazi sa servera (engl. *view source*, odnosno *view page source* na nekim pregledačima):



Slika 1.2. Podaci koje vraća sajt Vikipedija kada se otvori glavna strana.

[1] Servis je program ili aplikacija koja se ne pokreće eksplicitno, nego zajedno sa operativnim sistemom. Obično postoje neki mehanizmi koji ponovo automatski pokreću servis u slučaju da se isključio.

Na ovaj način možete da vidite informacije koje vraća server kada mu klijent zatraži podatke. Ovo je takozvani HTML (engl. *HyperText Markup Language*), sa kojim ćemo se upoznati u sledećem poglavlju.

Iako su najčešće klijenti pregledači, ništa vas ne sprečava da napravite program koji koristi internet protokol koji traži neke informacije od servera. Program cURL (engl. *client URL*) je jedan jednostavan program koji se može koristiti kako bi se poslao zahtev, recimo, sajtu Vikipedija i prikazao odgovor. U osnovi, cURL radi isto što i pregledač – šalje zahtev ka zadatoj adresi uz prateće informacije koje se mogu zadati. Na ovaj način možete lako da simulirate pregledač i da vidite šta udaljeni veb-server vraća kao odgovor.

> *cURL je jednostavna aplikacija koja se pokreće u komandnoj liniji operativnog sistema. Iako je jednostavna i lako ju je napraviti, Daniel Stenberg je za ovu aplikaciju 2017. dobio nagradu Polhem za tehnološku inovaciju Švedskog udruženja inženjera.*

cURL se pokreće iz komandne linije operativnog sistema i zadaje mu se adresa ka kojoj je potrebno poslati HTTP zahtev i sa koje je potrebno preuzeti podatke. Primer kojim bi se cURL komandom poslao zahtev ka sajtu Vikipedija i kojim bi se prikazo tekst koji udaljeni sajt vraća kao odgovor je prikazan u sledećem primeru:

```
C:\>curl https://www.wikipedia.org
<!DOCTYPE html>
<html lang="mul" class="no-js">
<head>
<meta charset="utf-8">
<title>Wikipedia</title>
<meta name="description" content="Wikipedia is a free online encyclopedia,
created and edited by volunteers around the world and hosted by the Wikimedia
Foundation.">
<script>
document.documentElement.className =
document.documentElement.className.replace( /(^|\s)no-js(\s|$)/, "$1js-
enabled$2" );
</script>
<meta name="viewport" content="initial-scale=1,user-scalable=yes">
```

Ovo se tehnički ne razlikuje od principa rada pregledača u koji na isti način možete da unesete URI nekog veb-sajta i vidite sadržaj. cURL će prikazati odgovor kao tekstualni rezultat, dok će pregledač interpretirati ovaj odgovor koji je predstavljen u HTML[2] jeziku i prikazati sadržaj veb-strane.

---

[2] O HTML jeziku će biti više reči u sledećem poglavlju.

Internet nije samo apstrakcija fizičke mreže nego i protokol koji opisuje kako će se prenositi informacije sa jednog na drugi računar. U internet mreži se informacije dele na tzv. pakete (engl. *networking packet*). Paketi su blokovi podataka koji sadrže informacije o tome kome treba proslediti paket, kao i sadržaj paketa koji predstavlja informacije koje se šalju između klijenta i servera. U slučaju da je potrebno preneti veliku količinu informacija (npr. video-fajl), može se poslati ogroman broj internet paketa koji sadrže delove fajla. Pošiljalac i primalac informacija nisu svesni činjenice da se informacije dele i šalju u paketima. I klijent i server imaju privid da su se sve informacije prenele kao jedna nedeljiva celina.

U nekim slučajevima će vam biti potrebno da utvrdite da li programi na jednom računaru mogu da kontaktiraju neki udaljeni računar. Zamislimo da je potrebno da utvrdimo možemo li da pristupimo nekom računaru na kojem se nalazi neki veb-sajt sa računara na kojem se nalazimo. Možemo da koristimo različite programe kao što su ***ping*** ili ***tnc*** kako bismo utvrdili da li postoji veza između dva računara na internetu. Rezultat ***ping*** naredbe koja proverava da li računar na kojem se nalazimo može da pristupi nekom računaru na kojem je sajt Vikipedija prikazan je u sledećem primeru:

```
C:\>ping www.wikipedia.com

Pinging ncredir-lb.wikimedia.org [91.198.174.194] with 32 bytes of
data:
Reply from 91.198.174.194: bytes=32 time=44ms TTL=58
Reply from 91.198.174.194: bytes=32 time=43ms TTL=58
Reply from 91.198.174.194: bytes=32 time=43ms TTL=58
Reply from 91.198.174.194: bytes=32 time=44ms TTL=58

Ping statistics for 91.198.174.194:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 43ms, Maximum = 44ms, Average = 43ms
```

Naredba ***ping*** će prikazati statistike o paketima koje je poslala zadatom računaru i vremenu koje je proteklo u komunikaciji. Zanimljiva informacija koju možemo videti je tzv. IPv4 adresa 91.198.174.194 sajta Vikipedija. U nastavku ćemo videti da svaki računar na internet mreži ima jedinstvenu adresu u ovakvom formatu na osnovu koje je identifikovan.

U slučaju da vidite ovakve informacije, verovatno možete da pristupite željenom računaru.

Komanda ping nije uvek pouzdana. Nekada možete da dobijete informacije da klijent ne može da dođe do servera, ali ne zato što ne postoji veza među njima, već zato što je udaljeni računar konfigurisan tako da ne odgovara na ping zahteve. Ovo je česta sigurnosna mera koja se podešava na računarima.

Informacije koje se razmenjuju među računarima se mogu prenositi na različite načine – putem bežične (WiFi) konekcije, optičkih kablova, satelitske komunikacije i slično. Samim tim, konekcija među računarima nije direktna, nego se informacije prenose putem velikog broja posrednih uređaja (modemi, ruteri, sateliti). Međutim, fizička infrastruktura mreže koja povezuje dva računara nije bitna za komunikaciju. Ona samo utiče na količinu podataka koja protiče (engl. *throughput*) i vreme potrebno da se prime informacije (engl. *latency*).

U slučaju da vas zanima kako putuju informacije od jednog računara ka drugom, možete pratiti rutu kojom će se kretati informacije do ciljnog računara. Na Windows operativnom sistemu možete da koristite program ***tracert***, kojem možete da prosledite adresu nekog veb-sajta koji želite da otvorite kako biste videli kuda sve prolaze zahtevi i koliko im vremena treba da prođu. Rezultat koji bi možda vratio ovaj program kada bismo mu zadali adresu sajta Vikipedija bi mogao da izgleda kao u sledećem primeru:

```
C:\>tracert www.wikipedia.com
Tracing route to ncredir-lb.wikimedia.org [91.198.174.194]
over a maximum of 30 hops:

 1    28 ms    34 ms     6 ms  csp1.zte.com.cn [192.168.1.1]
 2    23 ms    19 ms    19 ms  109.93.11.1
 3    19 ms    17 ms    16 ms  212.200.179.137
 4    25 ms    26 ms    28 ms  212.200.5.32
 5    61 ms     *       46 ms  176.209.249.80.in-addr.arpa [80.249.209.176]
 6    44 ms    45 ms    47 ms  ncredir-lb.esams.wikimedia.org [91.198.174.194]
```

U svakoj liniji možete da vidite sledeći računar na ruti koji će prenositi informacije, kao i koliko je vremena potrebno da informacije dođu. Poslednja informacija u svakom redu predstavlja računar u internet mreži preko kojeg su prešle informacije na putu do sajta Vikipedija. Ova informacija može biti IP adresa ili ime računara.

Intranet

Internet je javna mreža. Međutim, ništa nas ne sprečava da napravimo manje privatne internet mreže u kojima će se nalaziti uređaji koji nisu javno dostupni. Ovakve privatne internet mreže se nazivaju intranet mreže. U pogledu funkcije ne postoji razlika između internet mreže i intranet mreža. Dva uređaja koja komuniciraju

međusobno internet protokolom često neće primetiti razliku da li se nalaze u javnoj internet mreži ili u nekoj privatnoj intranet mreži.

Iako u teoriji nema razlike između interneta i intraneta, u praksi možete primetiti drugačije ponašanje prilikom komunikacije u intranet okruženju. Intranet mreže održavaju i konfigurišu administratori mreže, koji često zabranjuju pristup nekim adresama van mreže. U ovim slučajevima možete primetiti da neki sajtovi, računari ili servisi nisu dostupni iz intranet mreže iako možete da im pristupite van mreže.

Naravno, računari mogu biti istovremeno i u internet i intranet mreži, pa čak i povezivati računare u intranetu sa računarima u javnoj mreži.

Privatne mreže se obično zamišljaju kao izolovane mreže u okviru firmi, zgrada koje su spojene sa ostatkom sveta putem nekoliko računara koji predstavljaju konekciju između intranet i internet mreža. Međutim, privatne mreže se mogu sastojati od logički izolovanih, udaljenih računara. Na primer, mnogi računarski oblaci nam omogućavaju da određeni skup računara u oblaku grupišemo u logičku grupu koja će biti izolovana od ostatka interneta. Iako su računari u toj „privatnoj mreži" javno dostupni računari u računarskom oblaku, računarski oblak garantuje da niko ne može da im pristupi, tako da su logički identični privatnoj mreži koja je fizički izolovana od ostatka sveta. Primer ovakvih mreža su Azure VNet mreže, u kojima možete grupisati računare (odnosno Azure virtuelne mašine) koje se nalaze u Azure oblaku.

Adrese računara

Svaki računar na internetu ima svoj jedinstveni broj, odnosno IP adresu (engl. *Internet protocol address*). Kada se računar priključi na internet, dodeljuje mu se ova jedinstvena adresa na osnovu koje se razlikuje od ostalih računara u internet mreži.

U prethodnim primerima u kojima smo videli kako rade ***ping*** i ***tracert*** programi videli smo i da su prikazane informacije o računarima koje se sastoje od četiri broja u opsegu od 0 do 255, odvojena tačkom. Ovo su takozvane IPv4 adrese, koje su dodeljene nekim računarima na internetu.

U većini slučajeva, računar koji se priključuje na internet dobija od računara sa kojima je u grupi neku adresu. Ova adresa je najčešće dinamički dodeljena i u najvećem broju slučajeva se ne može garantovati da će računar dobiti istu adresu kada se sledeći put konektuje. Postoje skupovi adresa koji se mogu dodeliti računarima koji se povezuju internetom:

Opseg vrednosti adresa		Oznaka opsega	Broj adresa
212.200.0.0	212.200.255.255	212.200.0.0/16	65,536
212.200.224.0	212.200.224.255	212.200.224.0/24	256
212.200.225.0	212.200.225.255	212.200.225.0/24	256
178.222.0.0	178.222.255.255	178.222.0.0/16	65,536

Na primer, ako biste se uključili u internet mrežu preko svog provajdera i ako bi vam dodelio neku od adresa iz opsega 212.200.0.0/16, moguće je da biste dobili adresu 212.200.32.108. Ako biste se ponovo priključili na internet preko istog provajdera i iste grupe adresa, dobili biste neku drugu adresu, na primer, 212.200.82.48. U nekim slučajevima postoji način i da se zatraži stalna (statička) adresa.

Neke adrese su dodeljene od strane provajdera, ali neke adrese imaju predefinisano značenje. Na primer, IPv4 adresa 127.0.0.1 i IPv6 adresa ::1 predstavljaju adresu lokalnog računara (engl. *localhost*). U slučaju da neki program na računaru želi da dobije informacije od programa koji se nalazi na istom računaru, koristiće ovu adresu kako bi referencirao samog sebe. Svaki računar će ovu adresu poslati samom sebi i neće ni pokušati da pronađe neki računar na internetu.

Privatne adrese

U prethodnoj sekciji smo se upoznali sa privatnim intranet mrežama, koje predstavljaju grupe računara koji možda nisu javno dostupni. Kao što računari u javnoj internet mreži imaju svoje IP adrese, tako i računari u privatnim intranet mrežama dobijaju svoje privatne IP adrese.

Postoji skup adresa koje se mogu koristiti isključivo za privatne mreže:

Oznaka opsega	Broj adresa	Opseg vrednosti adresa
10.0.0.0/8	16777216	10.0.0.0 – 10.255.255.255
172.16.0.0/12	1048576	172.16.0.0 – 172.31.255.255
192.168.0.0/16	65536	192.168.0.0 – 192.168.255.255

Kao što svaki računar na internetu dobija jedinstvenu javnu IP adresu, tako i svaki računar u privatnoj mreži dobija jedinstvenu privatnu IP adresu. Najbitnija razlika između javnih i privatnih IP adresa je u činjenici da ne mogu postojati dva računara koja imaju istu javnu IP adresu, dok mogu postojati računari u različitim privatnim intranet mrežama koji imaju iste privatne adrese. Kada neki računar pokuša da pristupi računaru putem privatne IP adrese, pristupiće računaru u njegovoj privatnoj

mreži. Ne postoji mogućnost da računar koji se nalazi u privatnoj mreži pristupi računaru u drugoj privatnoj mreži na osnovu privatne IP adrese, sem ako te privatne mreže nisu na neki način spojene.

Domenska imena

Svaki računar na internetu ima svoju jedinstvenu IP adresu, ali njen format nije lak za pamćenje. Verovatno nikada ne biste unosili adrese strana koje izgledaju kao:

http://91.198.174.194/Internet

Ako bismo morali da pamtimo veb-sajtove i računare na internetu na osnovu IP adresa, bilo bi izuzetno teško da na ovakav način pristupimo veb-sajtovima koje želimo da otvorimo u pregledaču. Zbog toga su uvedena domenska imena sajtova koja predstavljaju nazive koji odgovaraju IP adresama računara. Na primer, lakše je zapamtiti da je potrebno otići na sajtove www.google.com, www.wikipedia.com, www.bing.com umesto zapamtiti njihove adrese u vidu četiri broja. Ovakva imena se nazivaju domenska imena i predstavljaju zamene za IP adrese. Jedino što je potrebno u ovom slučaju je mehanizam koji će mapirati ova domenska imena na odgovarajuće IP adrese.

U većini operativnih sistema postoji sistemski fajl koji sadrži mapiranje domenskih imena i IP adresa. Na primer, ovakav fajl se često zove ***hosts*** i njegov sadržaj može da bude:

```
www.wikipedia.com 91.198.174.194
www.bing.com 13.107.21.200
www.google.com 172.217.20.4
```

U slučaju da pomoću pregledača pokušate da pristupite nekom od sajtova sa ovim imenima domena, pregledač će u stvari otići na IPv4 adresu upisanu sa desne strane.

Upisivati imena domena i njihove IPv4 adrese u ***hosts*** fajl nije praktično i koristi se samo u specijalnim slučajevima. Najveći problem u ovakvom pristupu leži u činjenici da računari na internetu i sajtovi menjaju svoje IP adrese, tako da bi bilo potrebno ažurirati sve ***hosts*** fajlove na računarima klijenata.

U većem broju slučajeva postoje posebni računari ili servisi koji imaju informacije o imenima domena i njihovim adresama. Ovakvi servisi se nazivaju servisi domenskih imena ili DNS (engl. *Domain Name Service*). Kada se računar priključi na internet preko nekog internet provajdera, on često dobija IP adresu DNS računara koji će obezbediti informacije o IP adresi za neko traženo ime domena. U slučaju da je računar u privatnoj mreži, administrator mreže može da konfiguriše adresu