POGLAVLJE 12

Računarstvo u oblaku: Google App Engine

Našu delatnost je zahvatila plima inovacija i sada je pokreće... jedan glavni fenomen... oblak. I niko ne zna tačno šta je to ni šta znači.
– Stiv Bolmer, oktobar 2010.

U ovom poglavlju...

- Uvod
- Šta je računarstvo u oblaku?
- Kutija s peskom i App Engine SDK
- Biranje radnog okvira za App Engine
- Podrška za Python 2.7
- Kratki primeri (s Pythonovim kodom)
- Poređenje s Djangom?
- Pretvaranje aplikacije „Hello World“ u jednostavan blog
- Dodavanje servisa Memcache
- Statičke datoteke
- Dodavanje servisa za proveru identiteta korisnika
- Remote API Shell

- Slanje trenutnih poruka pomoću protokola XMPP
- Obrada slika
- Redovi za čekanje zadataka (nezakazani zadaci)
- Profilisanje pomoću alatke Appstats
- Servis URLfetch
- Kratki primeri (bez Pythonovog koda)
- Zaključavanje proizvođača
- Resursi

12.1 Uvod

Sledeći razvojni sistem koji ćemo predstaviti jeste Google App Engine. Iako App Engine ne nudi kompletan radni okvir za veb (engl. *full-stack web framework*), kao Django (mada Django možete izvršavati na njemu, što ćemo objasniti kasnije u ovom poglavlju), reč je o razvojnoj platformi, izvorno orijentisanoj ka veb aplikacijama (sadrži sopstveni mikro radni okvir, `webapp`, ili njegovu zamenu, novo okruženje `webapp2`), ali se koristi i za pravljenje aplikacija i servisa uopšte.

Kada kažemo „uopšte“, ne mislimo da se *svaka* aplikacija može napraviti niti prilagoditi pomoću sistema App Engine, već se to odnosi na mrežne aplikacije kojima je dovoljna samo HTTP krajnja tačka da bi bile dostupne. Tu spadaju, između ostalog, i veb aplikacije. Jedna popularna primena van veba jesu pozadinski servisi za mobilne klijente s korisničkim interfejsom. App Engine spada u kategoriju *računarstva u oblak*u (engl. *cloud computing*) fokusiranog na stvaranje platforme za programere za pravljenje i hosting aplikacija ili pozadinskih, to jest pogonskih mehanizama servisa. Pre nego što zađemo u detalje o platformi, moramo da predstavimo ekosferu računarstva u oblaku kako bismo mogli bolje da objasnimo gde se App Engine uklapa u tu sliku.

12.2 Šta je računarstvo u oblaku?

Dok Django, Pyramid ili Turbogears aplikacije isporučuje vaš omiljeni davalac internet usluga ili se čak izvršavaju na vašem računaru, Google App Engine aplikacije ugošćuje Google i deo su veće klase servisa, objedinjene pod kišobranom računarstva u oblaku. Glavna premisa iza tih servisa jeste da njihovi korisnici upotrebljavaju spoljne resurse za deo računarske infrastrukture kompanije (ili pojedinca), bilo da je reč o hardveru, razvoju ili izvršavanju aplikacija, ili smeštanju softvera. Ukoliko koristite računarstvo u oblaku, delegirate računarstvo, hosting, i/ili ponudu vaše aplikacije nekoj drugoj kompaniji.

Takvi servisi su dostupni samo na internetu, a njihova fizička lokacija može, a i ne mora biti poznata. U to spada sve – počev od sirovog hardvera[1] do aplikacija, i svi ostali mogući servisi između ta dva – poput operativnih sistema, baza podataka, datoteka i sirovog prostora na disku, izračunavanja, slanja e-poruka, razmene

1. Reč „hardver“ obuhvata fizičke uređaje (pod koje potpadaju i disk i memorija) s napajanjem, hlađenjem i mrežnom strukturom.

trenutnih poruka, virtuelnih mašina, keširanja (na više nivoa, od opcije Memcached do mreža za isporučivanje sadržaja, engl. *content delivery networks, CDN*) itd. Mnogo je aktivnosti u ovoj branši, a davaoci usluga neprestano izlaze s novim servisima. Usluge se obično plaćaju putem neke vrste pretplate ili modela plaćanja po upotrebi.

Jedan od glavnih razloga zašto kompanije koriste usluge računarstva u oblaku obično je cena. Međutim, potrebe se dovoljno razlikuju da svaka kompanija mora prvo da sprovede istraživanje kako bi odredila da li je to pravo rešenje za nju. Da li imate „startap" (engl. *startup*) kompaniju i ne možete priuštiti sav taj hardver (a ne želite da iznajmite usluge centra za podatke)? Nema problema, iznajmite jedan računar ili čak hiljadu od Amazona ili koristite veoma veliki disk ljubaznošću Googla. Prošli su dani kada osnivači malih startap kompanija nisu mogli da pokrenu biznis dok ne ulože u ovakvu infrastukturu. Sada se mogu fokusirati na svoje aplikacije i probleme koje one rešavaju.

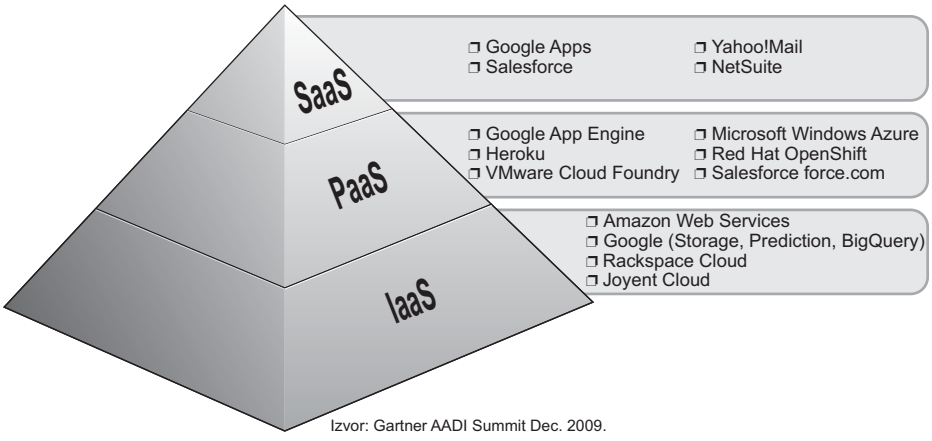
Situacija je nešto drugačija kada je reč o velikim preduzećima ili najprofitabilnijm kompanijama s liste Fortune 500 koje raspolažu dovoljno velikom računarskom snagom, ali uviđaju da se ona ne koristi do granica potencijala. Ne morate razviti posao sa oblakom do nivoa kakav je Amazon dostigao (više o tome u narednom odeljku), ali možete da napravite *privatni oblak* kako biste interno omogućili usluge oblaka, ili da formirate *hibridni oblak* i da interno opslužujete deo svoje infrastrukture (možda onaj sa osetljivim podacima), a druge delove (izračunavanje, aplikacije, skladištenje itd.) da dobavljate spolja od *javnog oblaka* kao što je Google ili Amazon.

Kompanije koje koriste usluge oblaka često brine fizička lokacija, sigurnost, ugovor o davanju usluga (engl. *service-level agreement, SLA*) i usklađenost; zavisno od delatnosti u kojoj su ili jurisdikcije, možda i moraju da budu vrlo oprezni u vezi s tim. Jasno je da kada svoje aplikacije, podatke itd., prepuštaju spoljnim servisima, kompanije žele garancije da je njihova intelektualna svojina bezbedna i sigurna, dostupna s fizičke lokacije sa odobrenjem vlasti te geografske oblasti (ako postoji) i da je pristup tim resursima omogućen u svakom trenutku. Pošto se ispune ti zahtevi, sledeći korak je da odrede koji nivoi računarstva u oblaku su im potrebni.

12.2.1 Nivoi usluga računarstva u oblaku

Računarstvo u oblaku dostupno je na tri nivoa usluga. Na slici 12-1 predstavljen je svaki sloj usluga kao i neki reprezentativni proizvodi na svakom od nivoa. Najniži sloj, poznat kao *infrastruktura kao servis* (engl. *Infrastructure-as-a-Service, IaaS*), pruža računarsku moć u najsvedenijem smislu – na primer, same računare (fizičke ili virtuelne), skladište (obično disk) i računsku obradu. Na nivou IaaS, Amazon Web Services (AWS) nudi usluge Elastic Compute Cloud (EC2) i Simple Storage System (S3). Google takođe pruža IaaS uslugu skladištenja zvanu Cloud Storage.

Google App Engine radi na srednjem nivou računarstva u oblaku poznatom kao *platforma kao usluga* (engl. *Platform-as-a-Service, PaaS*). Na ovom nivou, korisnici mogu da koriste izvršnu platformu za svoje aplikacije. Najviši nivo je *softver kao usluga* (engl. *Software-as-a-Service, SaaS*). Na tom nivou, korisnici pristupaju aplikacijama koje su nastale i dostupne su samo na internetu. U primere SaaS usluga spadaju veb servisi za e-poštu kao što su Gmail, Yahoo! Mail i Hotmail.



Slika 12-1 Tri nivoa usluga računarstva u oblaku.

Od ta tri nivoa, najpoznatiji su IaaS i SaaS, dok PaaS ređe dospeva u prvi plan. Međutim, situacija se možda menja, jer je sasvim moguće da je PaaS najmoćniji od njih. IaaS se dobija besplatno uz PaaS, ali PaaS obuhvata mnogo usluga koje su izuzetno skupe i često previše zahtevne da biste ih sami održavali. U to može da spada bilo šta na IaaS ili višem nivou, poput operativnog sistema, baza podataka, licenciranja softvera, umrežavanja i raspoređivanja opterećenja, veb i drugih servera, softverskih zakrpa i nadogradnji, nadzora, upozoravanja, sigurnosnih popravki, administriranja sistema itd. Ključna prednost korišćenja ovog nivoa usluga nad samostalnim održavanjem opreme jeste to što neće biti neiskorišćenih kapaciteta (engl. *idle capacity*), jer ste kupili mnogo više računarske moći nego što vam je trebala, na osnovu veb saobraćaja koji ste isprva predvideli. Ništa ne frustrira kao skupa investicija za koju znate da se ne koristi kako i koliko treba.

Iako koncept računarstva u oblaku postoji već duže vreme – Džon Gejdž iz kompanije Sun Microsystems skovao je upečatljiv slogan, ***Mreža je računar*** (*The Network is the Computer*), 1984. godine – komercijalizovan je tek sredinom prve decenije ovog veka, a naročito početkom 2006, kada je Amazon uveo AWS. Podstrekač je bilo upravo pitanje neiskorišćenih kapaciteta. Amazon je morao da kupi dovoljno računarskih resursa kako bi njegova on-lajn maloprodaja mogla da izdrži veb saobraćaj i potražnju tokom sezone prazničnih kupovina.

Amazon u svom izveštaju korisnicima[2] tvrdi da je „*do 2005. uloženo više od jedne decenije i stotine miliona dolara u izgradnju i održavanje pouzdane i efikasne IT strukture velikog obima koja je pokretala jednu od najvećih svetskih platformi za on-lajn maloprodaju*.“

Međutim, šta mislite da većina tih uređaja, sa svim tim kapacitetom i računarskom moći, radi tokom preostalog dela godine? Iskreno, jedno veliko ništa – zašto onda ne iznajmiti tu dodatnu CPU moć i kapacitete za skladištenje kao servis?

2. http://media.amazonwebservices.com/AWS_Overview.pdf

Amazon je upravo to i uradio. Tom trendu se u međuvremenu pridružilo nekoliko velikih tehnoloških kompanija: Google, Salesforce, Microsoft, RackSpace, Joyent, VMware i mnogi drugi koji su uzleteli među oblake.

Dok se Amazonovi servisi EC2 i S3 nesumnjivo nalaze na infrastrukturnom nivou, počelo je da se stvara novo tržište za one koji žele spoljne usluge za svoje aplikacije – konkretnije, da imaju mogućnost pisanja namenskih softverskih sistema koji koriste prednosti korporativnih Salesforce podataka (za odnose s potrošačima). To je navelo Salesforce da napravi force.com, prvi servis platforme namenjen upravo tome. Naravno, ne žele baš svi Salesforce aplikaciju napisanu na još jednom programskom jeziku, pa je Google razvio opštiji PaaS servis zvani App Engine, koji se pojavio aprila 2008.

12.2.2 Šta je App Engine?

Šta App Engine radi u knjizi o Pythonu? Je li to urođeni deo jezika ili paket drugog proizvođača? Iako nije ni jedno ni drugo, njegovo objavljivanje i postojanje ostavilo je dubok utisak na Pythonovu zajednicu korisnika i tržište; zapravo, toliko jak utisak, da sam s više strana dobio snažnu podršku za odeljak o platformi Google App Engine. (Isto se desilo s knjigom *Python Web Development with Django* koju sam pisao sa svojim cenjenim kolegama, Džefom Forsijerom i Polom Biseom.)

Razni radni okviri za veb imaju očekivane sličnosti i razlike, ali App Engine je zadivljujući pomak od njih po tome što ne samo da je reč o razvojnoj platformi, već se uz njega dobija i usluga hostinga aplikacije, što je glavni razlog zašto biste uopšte *hteli* da pravite aplikacije s njim. Korisnici ne moraju da pišu aplikacije, a potom da traže rešenje za hosting, ili, još gore, da prave sopstvenu infrastrukturu koja će podržati njihovu aplikaciju. Sav taj dodatni posao traži mnogo više od pukog projektovanja, pisanja koda i testiranja aplikacije.

Umesto da se bakću s davaocem internet usluga ili da sami udomljuju aplikaciju, programeri mogu svoje aplikacije da otpreme na Google, koji će se starati o svoj logistici održavanja aplikacija on-lajn. Veb programeri sada koriste iste resurse kao Google, iste centre za podatke i hardver koji pokreće ovog internet giganta. Zapravo, App Engine i ostali Google servisi oblaka predstavljaju javni API za resurse koje i sam Google koristi za svoje potrebe. Tu spadaju App Engine API interfejsi kao što su Datastore (Megastore, Bigtable), Blobstore, Image (Picasa), Email (GMail), Channel (GTalk) itd. Pored toga, programer više ne mora da brine o računarima, umrežavanju, operativnim sistemima, napajanju, hlađenju, raspoređivanju opterećenja itd.

Sve je to lepo, ali kako se Python uklapa u tu sliku?

Kada je App Engine pokrenut 2008, jedino jezičko okruženje koje je podržano bio je Python. Godinu dana kasnije došla je i Java, ali Python ima posebno mesto, jer je to bilo prvo izvršno okruženje s podrškom platforme App Engine. Programeri na Pythonu već znaju da je on kralj lake primene, da podstiče timski rad, omogućava izuzetno brz razvoj i ne traži od korisnika diplomu inženjera računarstva kako bi ga efikasno koristio. Ovakav pristup je dobrodošao kod programera svih mogućih radnih biografija i ubeđenja. Sam tvorac Pythona je inženjer u App Engine timu, da ne pominjem potpisnika ovih redova. Zbog njegove revolucionarne prirode i uskih

veza s Pythonovom zajednicom korisnika, sa ushićenjem sam se latio posla da vam predstavim App Engine!

App Engine ima četiri glavne komponente koje čine ceo sistem: jezička izvršna okruženja, skalabilnu hardversku infrastrukturu, administrativnu konzolu zasnovanu na vebu, i skup alata za razvoj softvera (engl. *software development kit, SDK*) tako da korisnici imaju alatke koje su im potrebne – server za razvoj i pristup App Engine API-jima.

Izvršna jezička okruženja

Kada je reč o jezičkim izvršnim okruženjima (engl. *language runtimes*), preostalo vreme ćemo (jasno) posvetiti Pythonu, ali znajte da su u trenutku pisanja ovih redova dostupni i Java i Go. Takođe, zahvaljujući podršci za Javu, programeri mogu da pišu kôd na jezicima sa odgovarajućim interpreterom koji se može izvršavati u Javinoj virtuelnoj mašini (JVM), kao što su Ruby, PHP, JavaScript i Python, a izvršavaju ga, redom, JRuby, Quercus, Rhino i Jython (dodajmo tome i Scalu i Groovy). Python s Jythonom je najzanimljiviji; neki se pitaju zašto bi korisnici hteli da izvršavaju Jythonovu aplikaciju, kada naprosto mogu da iskoriste matičnu podršku za Python. Reč je prevashodno o korisnicima koji žele da prave nove projekte na Pythonu, ali već imaju Java pakete. Naravno, hoće da iskoriste svoje postojeće pakete, ali ne mogu da priušte ili žele da te biblioteke prilagode Pythonu.

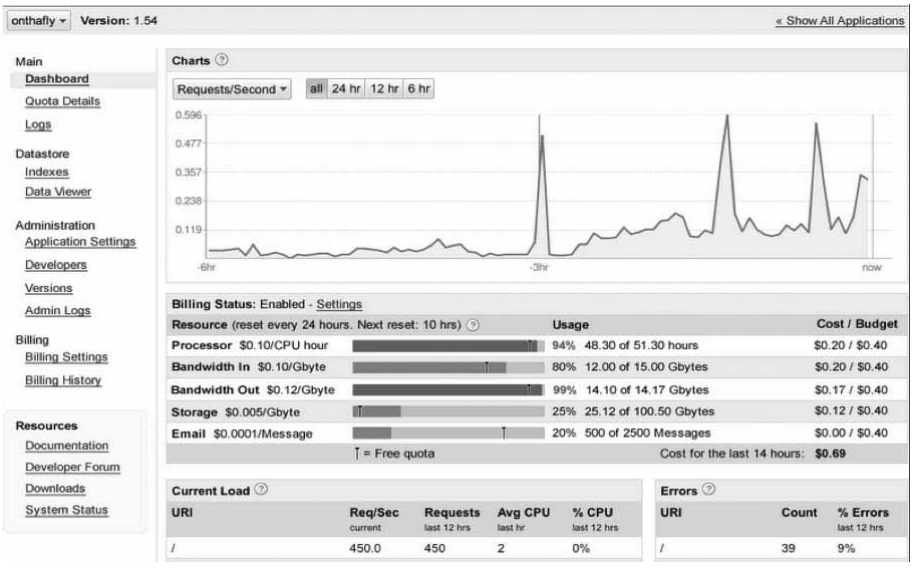
Hardverska infrastruktura

Hardverska struktura je zapravo crna kutija za korisnike: ne znate mnogo o bilo kom delu opreme na kojoj se izvršava vaš kôd. Verovatno ćete doći do zaključka da se koristi Linux u nekoj meri, i da kutije obitavaju u centrima za podatke priključenim na globalnu mrežu. Možda ste čak čuli za *Bigtable*, nerelacioni sistem baza podataka koji App Engine koristi za skladištenje podataka. Za većinu ljudi to je otprilike sve što treba da znaju: računarstvo u oblaku vas upravo oslobađa potrebe da brinete o tome. Izuzetno težak posao i detalji skopčani sa održavanjem takve infrastrukture i njene dostupnosti korisnicima, sklonjeni su u pozadinu.

Administriranje na vebu i stanje sistema

U preostalim odeljcima ovog poglavlja, osvrnućemo se na različite mogućnosti Pythonovog interfejsa za programiranje aplikacija (API). Imajte na umu da u fazi produkcije, vaše aplikacije neće izvršavati pune verzije Pythonovih (ili Javinih) interpretera. Pošto vaša aplikacija deli resurse s drugim korisničkim aplikacijama, sve aplikacije se, iz bezbednosnih razloga, moraju izvršavati u *kutiji s peskom (engl. sandbox)*, to jest, u ograničenom okruženju. Da, izgubićete neki nivo kontrole u zamenu za komponente koje je izuzetno teško napraviti i skalabilnost.

App Engine vam zato daje administrativnu veb konzolu (zvanu kraće *konzola admin*) koja programerima pruža uvid u njihovu aplikaciju, saobraćaj, podatke, prijavljivanje, fakturisanje, podešavanja, korišćenje, kvote itd. Slika 12-2 predstavlja ekran konzole za administriranje jedne aplikacije.



Slika 12-2 Konzola za administriranje aplikacija platforme Google App Engine. (Slika objavljena uz dozvolu kompanije Google)

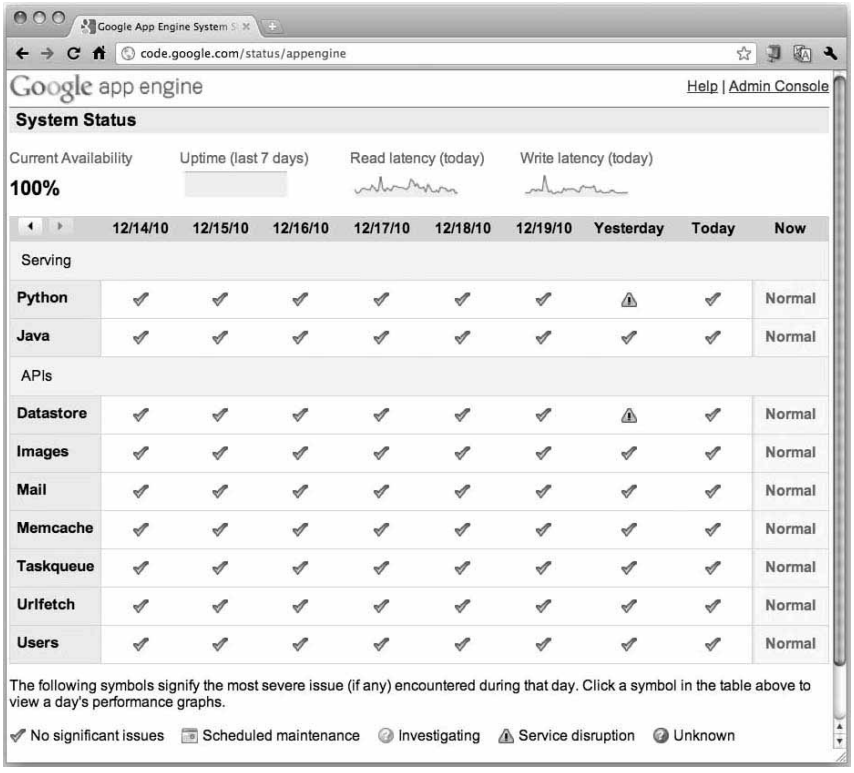
Postoji i stranica o stanju na nivou sistema (slika 12-3) na kojoj možete da pratite kako App Engine funkcioniše u celini za sve aplikacije.

Imajte na umu da „za sve aplikacije" znači upravo to. Od zime 2010, Google App Engine opslužuje više od milijardu veb strana dnevno. Pošto napravite i postavite svoju aplikaciju, bićete dodati tom zbiru. Iako je to uzbudljiva pomisao, ne zaboravite sledeće: pošto App Engine dele mnogi programeri, morate da naučite da živite u kutiji s peskom. To nije tako loše kao što zvuči, jer App Engine pruža mnoge usluge i API interfejse za programere.

12.3 Kutija s peskom i App Engine SDK

Jasno je da programeri ne bi voleli da druge aplikacije imaju pristup izvornom kodu njihovih aplikacija niti podacima, te je jedino fer poštovati druge aplikacije i njima pridružene podatke. U kutiji s peskom postoje izvesna ograničenja koja se ne mogu premostiti. (S vremena na vreme, Google ukida takva ograničenja, i samo kada je to bezbedno učiniti.) U zabranjene aktivnosti, između ostalog, spadaju:

- Ne možete napraviti datoteku na lokalnom disku, ali *možete* da napravite distribuiranu datoteku koristeći API Files.
- Ne možete da otvorite dolaznu konekciju mrežne utičnice.
- Ne možete da granate nove procese.
- Nisu vam dozvoljeni pozivi (operativnog) sistema.
- Ne možete da otpremate izvorni kôd koji nije na Pythonu.



Slika 12-3 Strana sa stanjem sistema Google App Engine aplikacija. (Slika objavljena uz dozvolu kompanije Google)

Zbog ovih ograničenja, App Engine SDK sadrži API interfejse višeg nivoa, pomoću kojih se mogu nadoknaditi eventualni gubici funkcionalnosti usled ograničenja.

Povrh toga, pošto je verzija Pythona koju App Engine izvršava (trenutno su to verzije 2.5 i 2.7) podskup pune distribucije, nemate pristup svim Pythonovim alatkama, naročito onim koje se kompajliraju na C-u. Neki od tih Pythonovih modula i paketa koji se kompajliraju na C-u dostupni su. Verzija 2.7 podržava znatno više C biblioteka, među njima i neke od dobro poznatih spoljnih paketa kao što su NumPy, lxml i PIL. Zapravo, dok verzija 2.5 podržava C biblioteke u vidu „bele liste“, s verzijom 2.7 ih je dostupno mnogo više – toliko da za ovu verziju postoji „crna lista“.

C biblioteke koje je Python 2.5 dozvolio, tj. stavio na belu listu, a Python 2.7 zabranio, tj. stavio na crnu listu, navedene su na adresi http://developers.google.com/appengine/docs/python/tools (slična lista postoji i za Javine klase). Međutim, ukoliko želite da koristite bilo koji od Pythonovih paketa drugih proizvođača, slobodno ih dodajte izvornom kodu, dok god je reč o čistom Pythonu (na primer, nema izvršnih datoteka, `.so` niti `.dll` datoteka itd.), i ti moduli/paketi jesu na beloj listi.

Važno je da znate da je ukupan broj datoteka koje možete da otpremite ograničen (trenutno na 10.000), s tim što je ograničena ukupna veličina tih datoteka (150 MB) – time su obuhvaćene datoteke aplikacije ili statičke prateće datoteke kao što su

HTML, CSS, JavaScript datoteke, itd. – kao i maksimalna veličina po datoteci (za sada je to 32 MB). Listu tekućih ograničenja u veličini možete pogledati na adresi http://developers.google.com/appengine/docs/python/runtime.#Quotas_and_Limits, a tim pokušava da ih pomeri kad god i gde god je to moguće. Međutim, postoji nekoliko načina da ublažite boli ovih ograničenja.

Ako vaša aplikacija koristi multimedijalne datoteke koje prevazilaze maksimalnu veličinu po datoteci, možete ih sačuvati pomoću servisa App Engine Blobstore (videti tabelu 12-1). Ovaj servis dozvoljava da čuvate datoteke proizvoljne veličine – to jest, ne postoji ograničenje veličine pojedinačnih datoteka (blobova). Ako vas brine ukupan broj `.py` datoteka, zapakujte ih u .zip datoteku i otpremite taj paket. Bez obzira na to koliko `.py` datoteka ste arhivirali, plaćate samo za jednu .zip datoteku. Naravno, i ta .zip datoteka ne sme da bude veća od maksimalne veličine po datoteci, ali barem ne morate da brinete o *broju* datoteka. Više o korišćenju .zip datoteka saznaćete u članku na adresi http://docs.djangoproject.com/en/dev/ref/settings (obratite pažnju na napomenu pri vrhu članka).

Na stranu ograničenja za datoteke, šta je s pomenutim ograničenjima u izvršavanju (bez utičnica, datoteka, procesa ili sistemskih poziva). Bez tih gradivnih blokova, čini se da ono što preostaje nije odviše korisna aplikacija. Ne očajavajte, pomoć je dostupna!

12.3.1 Servisi i API-ji

Kako bi vam pomogao da obavite posao, Google vam nudi sve više gradivnih blokova koje možete upotrebiti da premostite ova ključna ograničenja. Na primer, *zašto* biste hteli da otvarate mrežnu utičnicu? Želite li da komunicirate s drugim serverima? U tom slučaju, upotrebite API URLfetch. Šta je sa slanjem ili primanjem e-pošte? API Email je napravljen baš radi toga. Takođe, upotrebite API XMPP (eXtensible Messaging and Presence Protocol, ili, jednostavnije, Jabber) za slanje i primanje trenutnih (IM) poruka. Slična rešenja postoje za pristupanje sekundardnom mrežnom kešu (Memcache), primenu obrnutih AJAX tehnika ili guranja na čitač veba (Channel), pristupanje bazi podataka (Datastore) itd. U tabeli 12-1 prikazani su svi servisi i interfejsi dostupni App Engine programerima u trenutku pisanja ovih redova.

Tabela 12-1 Google App Engine servisi i API-ji (neki su eksperimentalni)

Servis/API	Opis
App Identity	Koristite kada aplikacija sadrži kôd koji mora da identifikuje sebe ili druge API-je koji traže takvu informaciju.
Appstats	Okruženje upravljano događajima koje vam pomaže da merite performanse vaše aplikacije.
Backends	Ako standardni rokovi za zahtev/odgovor ili red zadataka nisu dovoljno dugi, Backends omogućava da se App Engine izvršava bez vremenskog ograničenja.