Efikasno pisanje skriptova iz konzole u 'oblak'

PowerShell
praktična automatizacija

Matthew Dowst

kompjuter biblioteka

MANNING

PowerShell

praktična automatizacija

Matthew Dowst

PowerShell jezik za pisanje skriptova je multiplikator snage, koji obezbeđuje programsku kontrolu nad celim centrom podataka. Pomoću te moćne alatke možete da kreirate automatizaciju za višekratnu upotrebu, koja radikalno poboljšava konzistentnost i produktivnost vašeg operativnog tima. U ovoj knjizi je pokazano kako možete da dizajnirate, napišete, organizujete i primenite skriptove za automatizaciju operacija na sistemima svih veličina, od lokalnih servera, do klastera preduzeća u „oblaku".

U knjizi „**PowerShell, praktična automatizacija: efikasno pisanje skriptova iz konzole u 'oblak'**" pokazano je kako da kreirate PowerShell automatizaciju za lokalne sisteme i sisteme u „oblaku". U njoj ćete pronaći savete za identifikaciju zadataka koji se mogu automatizovati, tehnike za strukturiranje skriptova i upravljanje njima i mnoštvo dobro objašnjenih primera koda. Čak ćete naučiti kako da prilagodite postojeće skriptove novim slučajevima korišćenja i osnažite korisnike koji nemaju tehničke veštine pomoću lako razumljivih SharePoint front-endova.

Šta ćete pronaći u knjizi:

- strukturiranje PowerShell koda za deljenje i ponovnu upotrebu
- čuvanje i obezbeđivanje vaše automatizacije
- izvršenje automatizacije pomoću Azure Automationa, Jenkinsa, Task Schedulera i Crona
- čuvanje i preuzimanje podataka, akreditiva i promenljivih
- upotreba rešenja za kontrolu izvora za održavanje i testiranje promena koda

Za sistemske administratore i IT profesionalce koji upravljaju backend sistemima

Matthew Dowst ima više od 15 godina iskustva u IT menadžmentu i konsaltingu.

„Na svako pitanje koje sam imao o automatizaciji u PowerShellu ova knjiga je imala odgovor! Toplo je preporučujem."

Eric Dickey,
Raytheon

„Briljantan uvod u PowerShell za zadatke iz stvarnog sveta. Pristup i stil čine čitanje knjige užitkom."

Nik Rimington,
Spindo

„Prevaziđite isečke koda kopirane sa Interneta i naučite kako da izvršite automatizaciju."

Wayne Boaz,
Providence Health Plan

„Ova knjiga se izdvaja iz gomile. Uči vas kako da dizajnirate i pišete izvanredne PowerShell skriptove."

Roman Levchenko,
Microsoft MVP

ISBN: 978-86-7310-590-1

PowerShell
praktična automatizacija

MATTHEW DOWST

MANNING

kompjuter biblioteka

Izdavač:

kompjuter biblioteka

Obalskih radnika 4a
Beograd, Srbija

Tel: 011/2520272

e-pošta: kombib@gmail.com

veb-sajt: www.kombib.rs

Za izdavača:
Mihailo J. Šolajić, direktor

Autor:
Matthew Dowst

Prevod: Biljana Tešić
Lektura: Miloš Jevtović
Slog: Zvonko Aleksić

Znak Kompjuter biblioteke:
Miloš Milosavljević

Štampa: „Pekograf“, Zemun
Tiraž: 500
Godina izdanja: 2023.

Broj knjige: 567
Izdanje: Prvo
ISBN: 978-86-7310-590-1

Naslov originala:

Practical Automation with PowerShell
MATTHEW DOWST
ISBN 9781617299551

PowerShell, praktična automatizacija

Autorizovani prevod sa engleskog jezika.

CIP - Каталогизација у публикацији
Народна библиотека Србије, Београд

004.438PowerShell
004.7

ДАУСТ, Мерју
PowerShell: praktična automatizacija / Matthew Dowst ; [prevod Biljana Tešić]. - Izd. 1. - Beograd: Kompjuter Biblioteka, 2023 (Zemun: Pekograf). - XVII, 393 str.: ilustr.; 24 cm. - (Kompjuter biblioteka ; br. knj. 567)

Prevod dela: Practical Automation with PowerShell. - Tiraž 500. -
O autoru: str.
[XVIII]. - Registar.

ISBN 978-86-7310-590-1

a) Програмски језик „PowerShell“
b) Рачунарске мреже -- Администрирање

COBISS.SR-ID 121336329

KRATAK SADRŽAJ

DEO 1

DEO 2

SADRŽAJ

POGLAVLJE 2

DEO 2

POGLAVLJE 3

POGLAVLJE 4

POGLAVLJE 9

POGLAVLJE 10

DEO 3

POGLAVLJE 11

POGLAVLJE 14

DODATAK

PREDGOVOR

Iako većina ljudi poznaje PowerShell kao alatku komandne linije, on je mnogo više od toga. Ako pogledate „Microsoftov“ opis PowerShella, videćete da je u njemu navedeno da je PowerShell alatka/radni okvir za automatizaciju i konfiguraciju. PowerShell je napisan kao jezik čistog teksta koji je lako naučiti i lako je započeti rad pomoću njega, ali je i veoma moćna alatka koju možete koristiti za automatizaciju oromnog broja zadataka u svom okruženju i svakodnevnom životu. Međutim, nisam ovde da vam prodam PowerShell. Činjenica da čitate ovu knjigu pokazuje da znate za šta je PowerShell sposoban. Ova knjiga je osmišljena tako da vam pomogne da učite iz mog više-decenijskog iskustva u kreiranju automatizacije zasnovane na PowerShellu i da primenite ono što naučite za svoje potrebe automatizacije.

Baš kao mnogi ljudi u oblasti informacionih tehnologija, započeo sam svoju karijeru u tehničkoj podršci i prešao na ulogu sistemskog administratora. Bez obzira na kojoj sam poziciji bio, ako je trebalo da izvršim neki zadatak koji se ponavlja, skriptovao sam ga – prvo u VBS-u, a na kraju u PowerShellu. Bio sam u jedinstvenoj poziciji, jer sam se bavio infrastrukturom, ali sam na kraju došao u kompaniju koja se bavi razvojem prilagođenih aplikacija. Naučio sam mnogeo veština od ljudi sa kojima sam radio usput i koji su mi pomogli da kreiram veće i bolje automatizacije.

Kada sam radio kao konsultant, više puta sam uvideo da se kompanije plaše automatizacije – ne plaše se nužno da će ostati bez posla zbog automatizacije, već da ne ostanu „dužne“ automatizaciji. Bezbroj puta sam čuo da se neki proces ne može promeniti, jer niko ne zna kako da ažurira neku „ezoteričnu“ automatizaciju koju je neko kreirao pre mnogo godina.

Moj cilj je da u ovoj knjizi pomognem drugima da izbegnu pomenutu situaciju kreiranjem robusnih automatizacija koje se lako održavaju i koje će biti podržane u godinama koje dolaze.

PRIZNANJA

Proveo sam mnoge večeri i vikende u pisanju ove knjige, pa se, pre svega, zahvaljujem mojoj porodici. Zahvaljujem se mojoj supruzi Leslie, čija me ljubav prema čitanju zaista inspirisala da krenem ovim putem, da ne spominjem njenu beskrajnu podršku na tom putu, i mojoj deci Jasonu i Abigailu, koji su proveli mnoge subote i nedelje u čekanju da im tata izađe iz kancelarije i da se igra sa njima.

Takođe odajem priznanje Cameronu Fuleru, čije su mentorstvo i podrška bili najvažniji da me dovedu tu gde sam danas, i ostalim mojim kolegama u kompaniji „Quisitive“, koji su me inspirisali i podržavali tokom ovog procesa. To uključuje Grega Tatea i Davida Steina i druge, koji su obezbedili neprocenjive povratne informacije tokom MEAP procesa.

Osim toga, ova knjiga ne bi bila moguća bez pomoći mojih urednika Conora O'Briena i Michaela Lunda. Hvala vam, Conore, što ste radili sa mnom i naučili me kako da prenesem svoju poruku drugima. Mislio sam da sam ranije znao mnogo o pisanju, ali vaše strpljenje i posvećenost mojoj viziji pomogli su mi da učinim knjigu još boljom nego što sam ikada zamišljao. Takođe hvala Michaelu na njegovim tehničkim povratnim informacijama i smernicama, koje su mi izuzetno pomogle tokom procesa pisanja.

Zahvaljujem se i recenzentima i onima koji su obezbedili povratne informacije pomoću MEAP-a. Njihove povratne informacije su mi pomogle da napišem ovu knjigu za širu publiku.

Moju zahvalnost zaslužili su svi recenzenti: Aleksandar Nikolic, Alice Chang, Andreas Schabus, Anne Epstein, Anton Herzog, Bruno Sonnino, Charles Mike Shelton, Chuck Coon, Eric Dickey, Fredric Ragnar, Giuliano Latini, Glen Thompson, Glenn Swonk, Gonzalo Huerta Canepa, Håvard Wall, Jan Vinterberg, Jeremiah Griswold, Jérôme Bezet-Torres, Jiri Pik, Kent Spillner, Mike Haller, Milan Sarenac, Muralidharan T R, Mustafa Özçetin, Nik Rimington, Orlando Méndez Morales, Przemysław Chmielecki, Ranjit S. Sahai, Roman Levchenko, Sander Zegveld, Satej Kumar Sahu, Shawn Bolan, Sylvain Martel, Wayne A Boaz, Werner Nindl i Zoheb Ainapore – njihove povratne informacije su mi pomogle da napišem ovu knjigu za širu publiku.

Na kraju, zahvaljujem se PowerShell timu iz „Microsofta“ i, posebno, široj PowerShell zajednici. Ova knjiga ne bi bila moguća bez njihovog rada.

O OVOJ KNJIZI

Iako su lekcije u ovoj knjizi napisane pomoću PowerShella, koncepti kojima vas učim mogu se primeniti na bilo koji jezik ili platformu automatizacije, pa ćete detaljnije saznati kako nešto da uradite i više se oslanjati na pianje zašto treba da nešto uradite. Moj cilj je da vam pomognem da prihvatitete koncepte i da ih primenite direktno za vaše potrebe, tako što ću vam pokazati kako da razmišljate o automatizaciji i šta treba da postignete kako biste mogli da kreirate efikasne automatizacije koje su održive i koje ćete moći da koristite u godinama koje dolaze.

Ko bi trebalo da čita ovu knjigu?

Ova knjiga je za sve one koji su upoznali PowerShell, a koji bi želeli da kreiraju automatizaciju spremnu za preduzeća. Iako se koncepti ove knjige primenjuju na sve, od početnika, do stručnjaka, da biste izvukli maksimum iz ove knjige, trebalo bi da malo upoznate PowerShell. Trebalo bi da znate kako da instalirate module, da razumete osnove kreiranja PowerShell skripta (.ps1) i da znate neke osnove jezika, kao što su uslovni iskazi `if/else`, grupisanje (splatting) i petlje.

Kako je ova knjiga organizovana

Ova knjiga se sastoji od 14 poglavlja, koja su podeljena na tri dela. Svakim delom je obuhvaćen osnovni koncept procesa automatizacije.

Delom 1 je obuhvaćen početak vašeg „puta“ automatizacije:

- U Poglavlju 1 je razmotrena upotreba PowerShella sa tačke gledišta automatizacije i kako da osigurate da koristite odgovarajuće alatke za posao.
- U Poglavlju 2 je pokazano kako da organizujete vaše skriptove i module da biste napravili alatke za višekratnu upotrebu.

Deo 2 je „srce“ knjige i sadrži mnogo različitih koncepata automatizacije:

- Poglavljem 3 je obuhvaćena automatizacija planiranja i objašnjeno je kako da razmišljate o vašem kodu kada je pokrenut po rasporedu.
- U Poglavlju 4 je pokazano kako da rukujete bezbednim podacima u vašim automatizacijama, uključujući i upotrebu skladišta lozinki.
- U Poglavlju 5 je demonstrirano više načina na koje možete koristiti PowerShell za daljinsko izvršavanje i kako ih primeniti na situacije u stvarnom svetu.
- Poglavlje 6 počinje tako što pokazujem kako da koristite logiku u vašem kodu da biste automatizaciju učinili prilagodljivom. Zatim, taj koncept ide korak dalje, pokazujući kako da koristite eksterne podatke za kontrolu izvršavanja skripta za automatizaciju.

- U Poglavlju 7 je detaljno razmotreno korišćenje PowerShella sa backend bazom podataka, bez Excel i CSV datoteka, koje mnogobrojne kompanije koriste za skladištenje važnih podataka.
- U Poglavlju 8 je pokazano kako da koristite Azure za upravljanje i izvršavanje vaših automatizacija kombinovanjem mnogih koncepata iz prethodnih poglavlja u jednu platformu.
- U Poglavlju 9 je pokazano kako možete da koristite PowerShell za interakciju sa različitim rešenjima. Time su obuhvaćeni generisanje Word dokumenta iz PowerShella, komunikacija sa veb API-em, pa čak i pozivanje Pythona i prosleđivanje podataka između dva skripta.
- Poglavljem 10 su obuhvaćene neke najbolje tehnike za pisanje PowerShella, posebno za svrhe automatizacije.

U Delu 3 je pokazano kako možete da delite i održavate vaše skriptove za automatizaciju:

- Poglavljem 11 je obuhvaćen način na koji možete da koristite SharePoint kao front-end za PowerShell skript i kako da dizajnirate skriptove koje je potrebno pokrenuti na uređajima krajnjih korisnika.
- U Poglavlju 12 je pokazano kako da koristite GitHub za kontrolu izvora i za deljenje skriptova sa vašim kolegama.
- U Poglavlju 13 ćete naučiti osnove korišćenja Pestera za kreiranje jediničnih testova i testova integrisanosti, koji će vam pomoći da osigurate da vaši skriptovi budu usklađeni sa svim scenarijama za koje ste ih dizajnirali.
- U Poglavlju 14 je pokazano kako da se vratite na prethodni skript i izvršite izmene u njemu. Time su obuhvaćeni ono što treba da razmotrite unapred i uključivanje automatizovanog testiranja u vašu kontrolu izvora.

O kodu

Osim ako nije drugačije navedeno, ceo kod u ovoj knjizi je napisan tako da se u njemu koristi PowerShell 7.2 ili novija verzija. Za neke odeljke je i dalje potreban Windows PowerShell 5.1, ali su oni jasno naglašeni. Pokušavajući da napišem ovu knjigu da bude što uverljivija, pokušao sam da svedem na minimum zavisnost od platformi trećih strana. Bilo koja platforma ili eksterna alatka koje se koriste u ovoj knjizi su ili besplatne ili imaju besplatnu probnu verziju dovoljno dugo da možete da završite vežbe. Nema zavisnosti od „stvari“ kao što je Active Directory.

Da bi se zadovoljile potrebe za razmacima u štampanoj knjizi, u celoj knjizi se koristi grupisanje (splatting). Ako ne poznajete grupisanje, treba da znate da je to način za prosleđivanje kolekcije parametara komandi pomoću heš tabele. Grupisanje omogućava da podelite parametre na pojedinačne redove, pri čemu ih činite čitljivijim.

Da bi bila prikazana razlika između komande i izlaza iz komande, uvek kada se prikaže izlaz, kod će biti u posebnom bloku odmah nakon komande i biće uvučen. Osim toga, izlaz se može skratiti da prikaže samo relevantne podatke:

```
Code example
Output example
```

Pomoćni skriptovi su takođe navedeni u nekim poglavljima. Oni se, obično, koriste da vam pomognu da podesite svoje razvojno okruženje da prati lekcije u određenom poglavlju. Njihova upotreba će biti objašnjena u pojedinačnim poglavljima.

Forum za diskusiju liveBook

Kupovinom knjige *„PowerShell, praktična automatizacija“* dobijate besplatan pristup privatnom veb forumu koji vodi Manning Publications, a na kojem možete da komentarišete knjigu, postavljate tehnička pitanja i dobijate pomoć od autora i drugih korisnika. Da biste pristupili forumu, otvorite stranicu https://livebook.manning.com/book/practical-automation-with-powershell/discussion. Možete saznati više o Maningovim forumima i pravilima ponašanja na stranici https://livebook.manning.com/discussion.

Maningova posvećenost čitaocima ogleda se u njegovoj potrebi da bude obezbeđeno mesto na kojem se može održavati smislen dijalog između pojedinaca i između čitalaca i autora. Ovde se ne radi o obavezi autora za učešće u forumu - njihov doprinos forumu ostaje dobrovoljan (i neplaćen). Predlažemo da pokušate da im postavite neka izazovna pitanja! Forum i arhiva prethodnih diskusija biće dostupni sa veb stranice izdavača sve dok je knjiga u štampi.

O AUTORU

MATTHEW DOWST je konsultant upravljanja za kompaniju „Quisitive“ (ranije „Catapult Systems“) i vodeći arhitekta za njen tim upravljanja automatizacijom. U poslednjih 10 godina intenzivno je radio u PowerShellu kako bi pomogao klijentima svih veličina da automatizuju svoja proizvodna radna opterećenja. Matthew je takođe intenzivno uključen u PowerShell zajednicu, piše blogove, kreira module i učestvuje u onlajn forumima. On je takođe kreator PowerShell Weekly biltena, odnosno nedeljnog pregleda vesti o PowerShellu.

O ilustraciji korica

Slika na koricama knjige „PowerShell, praktična automatizacija“ se zove „Habitante de Frascati“ ili „Resident of Frascati“, a preuzeta je iz kolekcije „Jacques Grasset de Saint-Sauveur“, koja je objavljena 1797. godine. Svaka ilustracija je fino nacrtana i obojena rukom.

U to vreme bilo je lako prepoznati gde ljudi žive i koji je njihov zanat ili položaj u životu samo po odeći. Manning „slavi“ inventivnost i pokretanje računarskog poslovanja, tako što korice knjige zasniva na bogatoj raznolikosti regionalne kulture pre nekoliko vekova, oživljenoj iz kolekcija poput one koja se nalazi na korici ove knjige.

Deo 1

Ako odete na bilo koju konferenciju, pročitate bilo koju stručnu publikaciju ili samo razgovarate sa ljudima, shvatićete da je budućnost u automatizaciji, koja je mnogo više od pukog prihvatanja postojećeg ručnog procesa i pisanja nekog koda koji će proces izvršiti automatski. Da biste bili zaista uspešni u svojim poduhvatima u oblasti automatizacije, vaše automatizacije vam moraju uštedeti vreme i novac. Međutim, osim što morate da izračunate vreme koje vam je potrebno da izvršite zadatak u odnosu na mašinu, morate da izračunate i vreme koje je potrebno za kreiranje i održavanje automatizacije.

U ovom odeljku ćete naučiti ne samo kako da izračunate troškove automatizacije, već i kako da minimizirate troškove kreiranja i održavanja. Osim toga, videćete kako će vam pravilno planiranje vaših projekata i kreiranje koda za višekratnu upotrebu uštedeti vreme sada i u budućnosti.

1

POWERSHELL AUTOMATIZACIJA

Ovim poglavljem obuhvaćene su sledeće teme:

- kada da koristite evidentiranje, praćenje ili debagovanje u vašoj veb
  kako da konceptualizujete svoje potrebe automatizacije
- zašto bi trebalo da automatizujete pomoću PowerShella
- kako da znate kada je PowerShell odgovarajuća alatka za posao
- kako danas možete započeti automatizaciju vašeg posla

Svakog dana u svim industrijama IT profesionalci imaju zadatak da urade više pomoću manje sredstava, a najbolji način da se to postigne je automatizacija. Međutim, mnogo kompanija ne vidi IT automatizaciju kao prednost u odnosu na ručni proces. Najčešće automatizaciju kreira neki sistemski administrator u svoje slobodno vreme. To često dovodi do situacija u kojima automatizacija postaje manje efikasna od ručnog izvršavanja zadatka ili, još gore, postaje blokator promena.

Siguran sam da ste u nekom trenutku pokušali da koristite jednu od platformi za automatizaciju bez koda, kao što su IFTTT, Flow ili Zapier. Ako ste poput mene, verovatno smatrate da su funkcije previše osnovne. One su odlične za personalne automatizacije jednokratnog tipa, ali da biste zaista iz njih izvukli ono što vam je potrebno i osigurali da mogu da podrže automatizaciju na nivou preduzeća, potrebno je da budu prilagođene onome što je obezbeđeno izvan njihovih GUI-ova.

Sada PowerShell može „zablistati“. On je radni okvir za automatizaciju zadataka sa jednostavnim i intuitivnim jezikom za pisanje skriptova. PowerShell uključuje *komande lets* (cmdlets), koje omogućavaju da izvršavate slične zadatke dostupne na administratorskim konzolama i obezbeđuju radni okvir u kojem se zadaci mogu povezati zajedno da bi bio izvršen niz logičkih koraka. Komande cmdlets omogućavaju da automatizujete ceo Microsoft eko-sistem (Azure, Office 365, Microsoft, itd) i druge platforme, kao što su Linux i Amazon Web Services i da upravljate njima. Kada iskorišćava potencijal PowerShella i uči nekoliko osnovnih principa automatizacije, svaki IT stručnjak može postati guru za automatizaciju.

Osim što od IT profesionalaca zahteva da urade više pomoću manje sredstava, IT industrija prelazi na infrastrukturu kao model koda. Imam jedinstveno iskustvo rada u kompaniji specijalizovanoj za infrastrukturni konsalting i razvoj prilagođenih aplikacija. To mi je dalo priliku da učestvujem u projektima automatizacije u okviru obe specijalnosti i naučio sam da svako može postati guru za automatizaciju sa malo poznavanja tih specijalnosti.

Ako ste sistemski administrator ili drugi IT stručnjak, verovatno već poznajete rad u interfejsima komandne linije (CLI-ovima), tako što koristite batch/shell datoteke i pokrećete PowerShell skriptove. Stoga, prelazak na pisanje koda specijalno za automatizaciju nije mnogo značajan. Međutim, možda ne poznajete mnogo neke od pratećih veština koje se odnose na kontrolu izvora i jedinično testiranje, a ova knjiga ima za cilj da vam pomogne da ih upoznate.

U isto vreme neko sa snažnim iskustvom u programiranju možda ne poznaje mnogo sve idiosinkrazije sistemske administracije. Tu „na scenu“ stupa PowerShell, jer se ne oslanja na arhitekturu preduzeća. Možete isto toliko lako pokrenuti skript i na svojoj lokalnoj mašini i na serveru. U ovoj knjizi pokazano je kako možete da iskoristite PowerShell u organizaciji bilo koje veličine da biste kreirali robusne, održive i bezbedne automatizacije.

1.1 Šta ćete naučiti u ovoj knjizi

U ovoj knjizi nije pokazano samo kako da pišete PowerShell skriptove. Već postoje stotine resursa za samo pisanje koda. Umesto toga, cilj je da vam pokažem kako možete da koristite PowerShell kao alatku za automatizaciju, tako što čete naučiti:

- kako možete da iskoristite PowerShell da biste automatizovali zadatke koji se ponavljaju
- kako da izbegnete uobičajene „zamke“ automatizacije pomoću PowerShella
- kako da delite i održavate skriptove za svoj tim
- kako da uključite svoje skriptove za krajnje korisnike

Taj cilj ćemo postići korišćenjem primera iz stvarnog sveta sa kojima se IT profesionalci svakodnevno susreću. Upoznaćete tehničku stranu pisanja koda, konceptualnu stranu, tj. zašto je kod strukturiran na konkretan način i kako te obe strane možete primeniti za svoje potrebe automatizacije.

1.2 Praktična automatizacija

Možemo pretpostaviti da ste se zapitali šta da automatizujete. Dok je odgovor koji najverovatnije želite da čujete „sve“, generički odgovor je „svaki zadatak koji se ponavlja za čiju automatizaciju treba manje vremena nego za izvršenje“. Međutim, baš kao mnoge „stvari“ u IT oblasti, odgovor nije uvek toliko jednostavan. Morate uzeti u obzir više faktora da biste utvrdili da li je nešto vredno automatizacije, a, kao što ćete videti, time možda nećete uvek uštedeti vreme.

Ako vam je potrebno manje vremena da neki zadatak automatizujete nego što je potrebno da ga uradite ručno, lako je reći da je onda vredan automatizacije, ali to nije potpuna „priča“. Morate uzeti u obzir sledeće stavke:

- *vreme* – Koliko vremena je potrebno za izvršenje zadatka?
- *učestalost* – Koliko često se zadatak izvršava?
- *relevantnost* – Koliko će dugo biti potreban zadatak/potrebna automatizacija?
- *implementacija* – Koliko vremena će potrebno za automatizaciju zadatka?
- *održavanje* – Koliko dugotrajnog održavanja i uobičajenog održavanja će biti potrebno?

Prve dve stavke, koliko dugo i koliko često, obično su najjednostavniji brojevi koji se mogu izračunati, zajedno sa poslovnom stranom – na primer, koliko dugo će zadatak biti relevantan. Na primer, ako automatizujete zadatak koji će nestati nakon sledeće nadgradnje sistema, možda nećete nadoknaditi uloženo vreme.

Vreme implementacije i troškovi održavanja mogu biti malo teži za izračunavanje. To su „stvari“ koje ćete početi da dobijate ako što više automatizujete. Samo zapamtite da treba da uzmete u obzir cenu alatki, platformi, licenci, itd. Da biste utvrdili troškove održavanja, potrebno je da uzmete u obzir zadatke održavanja zasnovane na tehnologiji, kao što su održavanje platforme, promene API-a i nadgradnje sistema.

Kada dobijete odgovore na ta pitanja, možete izračunati koliko vremena možete potrošiti na automatizaciju zadatka da biste utvrdili da li je ona vredna vašeg vremena. Možete izračunati trošak množenjem vremena sa učestalošću i relevantnošću. Zatim, dodajte svoju implementaciju i održavanje tokom perioda relevantnosti. Ako vaši trenutni troškovi premašuju vašu cenu automatizacije, onda je zadatak vredan automatizacije.

$$\text{Vreme} \times \text{Učestalost} \times \text{Relevantnost} > \text{Implementacija} + (\text{Održavanje} \times \text{Relevantnost})$$

$$\text{Trenutni troškovi} > \text{Troškovi automatizacije}$$

Na početku vašeg „puta“ automatizacije procena troškova implementacije i održavanja može biti izazovna. Međutim, procenu ćete više naučiti što je budete više izvršavali. Sve dok je na naučite, dobro pravilo je da, ako mislite da možete automatizovati zadatak za duplo kraće vreme, to i uradite, jer će vam biti skoro zagarantovan zadovoljavajući povraćaj vaše investicije.

Osim prednosti pojednostavljivanja zadatka koji se ponavlja, postoje i drugi faktori koje treba uzeti u obzir prilikom određivanja šta bi trebalo da automatizujete. Sve što je sklono visokom

stepenu ljudske greške je odličan „kandidat“ za automatizaciju. Rad sa velikim skupom podataka i transkripcija podataka su odličan primer dva zadatka koja su „zrela“ za automatizaciju. Ljudi čine greške kada kucaju. Te greške se povećavaju kada rade sa mnogo podataka. Ako ste ikada kreirali složenu Excel formulu za manipulaciju nekim podacima, već ste kreirali jednostavnu automatizaciju.

Čak i ako zadatak koji imate ne treba da ponavljate više puta, kreiranje jednokratne automatizacije može vam uštedeti vreme. Osim toga, ako zadržite tu automatizaciju, možete je koristiti kao osnovu ako budete morali da izvršavate sličan zadatak u budućnosti. Odličan primer za to su zadaci manipulacije znakovnim nizovima. Na primer, recimo da imate tekstualnu datoteku sa gomilom loše formatiranog teksta, koji treba da raščlanite u kolone i unesete u unakrsnu tabelu. Međutim, tekst ne sadrži mnogo redova i možete ga transkribovati ili kopirati/nalepiti u roku od nekoliko minuta. Ili možete to iskoristiti kao priliku da usavršite svoje veštine, koristeći regularne izraze, podele, podstringove, indekse, zamene ili bilo koji drugi broj metoda manipulacije znakovnim nizovima. Kada naučite da pravilno koristite te metode, steći ćete neprocenjivu veštinu na vašem „putu“ automatizacije.

Drugo što možete automatizovati su zadaci koje možda nećete morati da izvršavate često, ali koji su složeni i dugotrajni. Ako je zadatak dovoljno složen da kreirate kontrolnu listu, kreirali ste i plan projekta automatizacije. Započnite rad tako što ćete automatizovati jedan od koraka, zatim još jedan, pa još jedan i tako dalje, dok ne dobijete potpuno automatizovan proces. Sledećeg puta kada se pojavi taj zadatak, možete da kliknete na dugme, umesto da koristite kontrolnu listu ili da pokušavate da zapamtite svaki korak procesa.

Najbolji način da počnete rad na vašem „putu“ automatizacije je da pronađete jednostavan zadatak koji stalno izvršavate i automatizujete. To ne mora da bude veliki zadatak ili bilo šta otmeno. Samo razmislite o nečemu što će vam uštedeti vreme.

Takođe možete da koristite automatizaciju da vam pomogne da prevaziđete prepreke ili hendikepe koji vas mogu sprečiti da budete produktivni onoliko koliko biste želeli. Recimo, slobodno ću priznati da nisam najorganizovanija osoba kada je reč o mojem prijemnom sandučetu za e-poštu. Voleo bih da budem organizovan, ali ne mogu uvek sve držati pod kontrolom. Ne volim da koristim Outlook pravila, jer ne želim da propustim neko upozorenje, posebno ako sam daleko od svog stola. Dakle, ono što se na kraju dešava je da brzo pročitam svoje e-poruke, ali ih ne arhiviram uvek odmah. Kao rezultat toga, vremenom dobijem hiljade e-poruka u prijemnom sandučetu. Da bih se izborio sa tim, napisao sam skript koji će arhivirati moje e-poruke, umesto mene. Taj skript premešta poruke u određene fascikle na osnovu adresa e-pošte i ključnih reči. Ne samo da mi ta automatizacija štedi vreme i pomaže da budem produktivniji, već me čini srećnim, a na kraju dana to mi se isplati.

Ne morate da automatizujete ceo proces s kraja na kraj. Možete vrlo dobro izračunati da bi cena automatizacije zadatka bila viša od cene ručnog izvršavanja. Međutim, možda ćete moći da automatizujete određene delove da biste uštedeli vreme i dobili pozitivan povraćaj ulaganja. Savršen primer za to su bar-kodovi, koji omogućavaju kasirima i magacionerima da brzo skeniraju artikle, umesto da ručno unose njihove kodove. RFID čipovi su još brži, ali cena njihove implementacije je do sada bila viša od cene skeniranja bar-koda.

Što više iskustva steknete u automatizaciji, moći ćete uspešnije da odredite šta je vredno, a šta nije vredno automatizovati. Osim toga, kao što ćete videti u sledećem odeljku, korišćenjem faznog pristupa sa gradivnim blokovima za višekratnu upotrebu u vašim procesima automatizacije možete se pripremiti za dalje kreiranje većih i boljih automatizacija.

Da bismo vam pomogli da započnete rad, ističemo četiri ključna faktora koja treba da uzmete u obzir prilikom dizajniranja automatizacije:

- cilj automatizacije
- okidače automatizacije
- akcije
- održivost

Cilj automatizacije je ono što ona treba da postigne. *Okidač* je ono što pokreće akcije automatizacije. *Akcije* su koraci preduzeti tokom automatizacije. Na kraju, *održivost* je ono što je potrebno da se ta automatizacija održi u celini i kao pojedinačni gradivni blokovi.

Možemo koristiti uobičajeni primer iz stvarnog sveta da bismo ilustrovali svaki deo „anatomije“ automatizacije. Na primer, zamislite da imate veb server koji nastavlja da funkcioniše oflajn, jer evidencije popunjavaju diskove. Te evidencije se ne mogu izbrisati, pošto su neophodne za praćenje sigurnosnih događaja. Dakle, otprilike jednom nedeljno morate potražiti datoteke starije od 30 dana, komprimovati te stare evidencije i premestiti ih u dugotrajno skladište.

1.2.1 Cilj automatizacije

Cilj automatizacije je ono što pokušavate da postignete pomoću specifične automatizacije. Iako se čini da ga je lako definisati, morate da se pobrinete da uzmete u obzir sve aspekte automatizacije.

U našem primeru čišćenja datoteke evidencije naš očigledan cilj je da sprečimo da se diskovi na veb serveru popune, ali time ćemo samo dotaći vrh „ledenog brega“. Da nam je to jedini cilj, mogli bismo jednostavno da izbrišemo stare evidencije. Međutim, te evidencije su potrebne za praćenje sigurnosnih događaja. Dakle, naš cilj je da kreiramo proces automatizacije koji će sprečiti popunjavanje diskova, istovremeno obezbeđujući da se podaci ne izgube i da budu dostupni u retkim prilikama kada su potrebni. To obezbeđuje pregled automatizacije i može se koristiti za kreiranje kontrolne liste prilikom dizajniranja vaših akcija.

Na primer, ako promenimo naš cilj tako da uključimo redovan pristup podacima, to bi moglo da izmeni naše akcije. U tom slučaju komprimovanje datoteka i njihovo premeštanje u dugotrajno skladište ne bi bila najbolja opcija. Umesto toga, možemo da premestimo datoteke u veći niz za skladištenje. Time bismo olakšali pristup podacima, a istovremeno sprečili popunjavanje diskova. Sada, kada znate šta želite svojom automatizacijom, možete početi da planirate korake koje su potrebni da to postignete.

1.2.2 Okidači

Okidači pokreću automatizaciju. Uopšteno govoreći, postoje dve vrste okidača: okidači prozivanja (polling) i okidači zasnovani na događajima. Okidači prozivanja se prijavljuju pomoću krajnjih tačaka, a okidače događaja inicira neki spoljni događaj. Razumevanje razlike između ta dva tipa okidača i načina na koji oni funkcionišu značajno će uticati na vaš „put“ automatizacije.

Okidači prozivanja rutinski proveravaju da li neki sistem ima specifične uslove. Dve tipične implementacije, odnosno one koje ćemo koristiti u ovoj knjizi, su *monitori* i *rasporedi*.

Monitor se prijavljuje i čeka određeno stanje. Stanje može biti bilo šta, od praćenja FTP sajta za otpremanje datoteka, do nagledanja prijemnog sandučeta za e-poštu ili potvrde da je neka usluga pokrenuta. Monitori mogu da rade neprekidno ili u intervalima koji se ponavljaju.

Izbor korišćenja kontinuiranog ili intervalnog monitora zavisiće od ravnoteže između potreba i troškova automatizacije. Na primer, recimo da nadgledate deljenje datoteke da bi ona bila upisana. Ako znate da se datoteka upisuje samo jednom na svaki sat, provera automatizacije na svakih 60 sekundi bi bila gubitak resursa.

Dok monitor može da radi po rasporedu koji se redovno ponavlja, planirana automatizacija se razlikuje po tome što sam okidač ne proverava stanje pre nego što pokrene sledeće korake. Umesto toga, koraci će biti pokrenuti uvek kada je pokretanje planirano. Uobičajeni primeri koraka uključuju čišćenje datoteka, sinhronizaciju podataka i zadatke rutinskog održavanja. Baš kao kada koristite monitor, morate pažljivo razmotriti potrebe vaše automatizacije kada postavljate raspored.

Okidač događaja se pokreće kada spoljni događaj inicira automatizaciju. Na primer, uobičajeni okidač događaja je http zahtev, kao što je webhook. Okidači događaja takođe mogu sadržati pozive iz drugih automatizacija, a većina alatki usluge Service Desk ima mehanizam toka posla koji može da pokrene automatizaciju kada bude primljen određeni zahtev. Ovo je samo nekoliko primera automatizovanih okidača događaja, ali svaka spoljna interakcija se može smatrati okidačem događaja.

Jednostavan pritisak na neko dugme ili izvršavanje komandnog shella može biti okidač događaja. Ključno je da okidače događaja inicira bilo koji spoljni događaj, dok okidači prozivanja dosežu do krajnje tačke.

Vratimo se na primer čišćenja evidencija veb servera. Morate da shvatite koji okidač bi bilo najbolje koristiti, okidač prozivanja ili okidač događaja. U ovom primeru okidač prozivanja ima smisla koristiti, jer veb server nema načina da dopre do krajnje tačke. Sada morate da odredite da li treba da koristite monitor ili raspored. Obično se monitor koristi za probleme koji zahtevaju trenutne akcije ili akcije u bliskoj budućnosti – na primer, kada je usluga zaustavljena ili je veza sa mrežom prekinuta. Pošto je čišćenje evidencije veb servera zadatak održavanja, logično bi bilo da koristite raspored. Zatim, morate odrediti interval ponavljanja.

Već znate da te evidencije morate čistiti najmanje jednom nedeljno. Logično, najbolje bi bilo da koristite okidač sa intervalom ponavljanja kraćim od nedelju dana. Takođe znate da se nova datoteka evidencije kreira nakon određenog broja redova. Dnevno se generišu oko tri ili četiri evidencije. Prema tome, posao koji se izvršava jednom dnevno bio bi dobra opcija, jer bi sve manje od toga bilo previše, a sve više bi dovelo do rizika da evidencije postanu prevelike. Kada odredite koji ćete okidač koristiti, vreme je da pređete na osnovni deo vaše automatizacije, odnosno na akcije.

1.2.3 Akcije

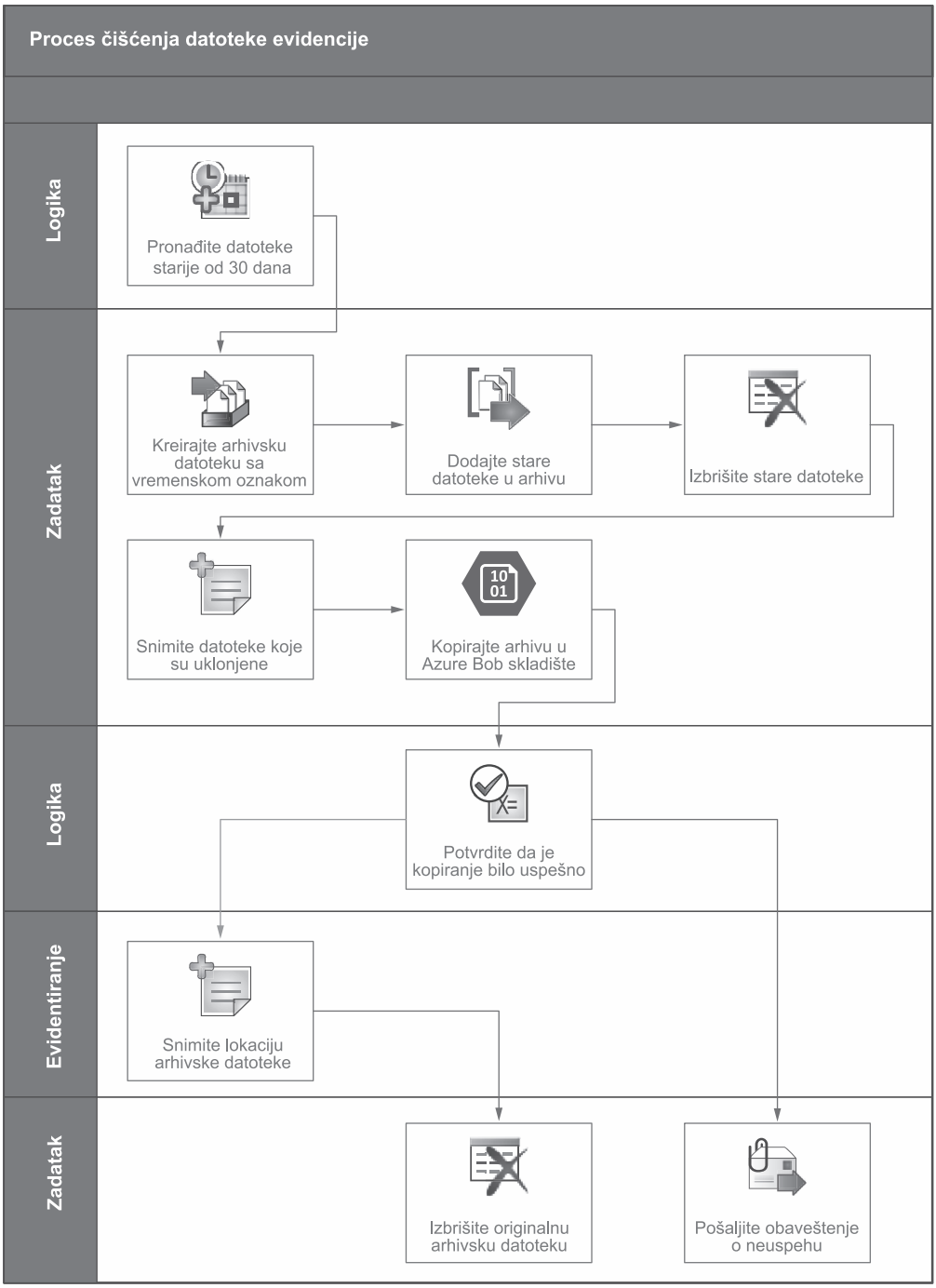
Akcije su ono na šta većina ljudi pomisli kada razmišlja o automatizaciji. Akcije su operacije koje vaša automatizacija izvršava da bi bio postignut njen cilj. Automatizacije se mogu sastojati od više različitih akcija, koje se ponekad nazivaju *koraci*. Akcije možete klasifikovati u tri glavne kategorije: *logika*, *zadaci* i *evidentiranje*. Na slici 1.1 su prikazani koraci za automatizaciju čišćenja evidencije.

Akcije logike kontrolišu tok vaše automatizacije. One uključuju uslovne konstrukcije (vaše uobičajene if/else uslove), petlje, čekanja, „hvatanje"/obradu grešaka i rukovanje promenljivim ili drugim podacima tokom izvršavanja. Zadaci su akcije koje se izvode u odnosu na krajnje tačke. Drugim rečima, ako akcija nije logika ili evidentiranje, to je zadatak. Najbolji način da razmišljate o zadatku je da su akcije logike kao mozak, a zadaci kao ruke.

Evidentiranje je, kao što sam naziv ukazuje, snimanje akcija. Vaše evidentiranje se može sastojati od izlaza akcija logike i zadataka. Dok se akcije evidentiranja mogu smatrati zadacima, ja radije razmišljam o njima odvojeno, jer nisu direktno uključene u ispunjavanje cilja automatizacije. Međutim, one će biti direktno uključene u kreiranje uspešnih i održivih automatizacija.

Kada pogledamo naš primer čišćenja datoteka evidencije, možemo identifikovati akcije koje treba da preduzmemo i tipove akcija:

1. Pronađite evidencije starije od 30 dana (logika).
2. Kreirajte arhivsku datoteku sa nazivom vremenske oznake (zadatak).
3. Dodajte stare datoteke u arhivu (zadatak).
4. Uklonite stare datoteke sa diska (zadatak).
5. Snimite datoteke koje su uklonjene i naziv arhivske datoteke (evidencija).
6. Kopirajte arhivske datoteke u Azure Blob Storage za dugotrajno skladištenje (zadatak).
7. Potvrdite da je kopiranje bilo uspešno (logika). Ako nije bilo uspešno, zaustavite proces i pošaljite obaveštenje.
8. Snimite lokaciju nove datoteke (evidencija).
9. Uklonite originalnu arhivsku datoteku (zadatak).



Slika 1.1 Koraci za proces automatizacije čišćenja datoteka odvojeni prema logici, zadacima i evidentiranju

1.2.4 Održivost

Pre nekoliko godina pomogao sam jednom klijentu da automatizuje svoje procese dodeljivanja privilegija korisnicima. Tokom faze otkrivanja rečeno mi je da korisnici moraju da budu kreirani u određenom direktorijumu, ostavljeni u tom direktorijumu jedan sat, a zatim premešteni u odgovarajući direktorijum. Rečeno mi je da će to omogućiti korisnicima da se sinhronizuju sa internom aplikacijom. Ispostavilo se da je osoba čiji je posao bio da dodaje i uklanja korisnike iz te aplikacije odlučila da će automatizovati proces. U to vreme svi korisnici su se nalazili u istom direktorijumu. Prema tome, kreirana je automatizacija koja je kompaniji uštedela 30 do 60 minuta rada nedeljno. Međutim, vremenom su se okolnosti promenile.

Kompanija se proširila i morala je da obezbedi različita pravila različitim korisnicima, tako da je kreirala različite direktorijume. Onda je primećeno da određeni korisnici nisu kreirani u tom internom sistemu. U to vreme je osoba koja je napisala automatizaciju otišla iz kompanije i niko drugi nije razumeo kako ona funkcioniše. Dakle, zaposleni u kompaniji bi dodali korisnike u direktorijum, sačekali dok se ne pokrene sinhronizacija po satu, a zatim bi ih premestili u odgovarajući direktorijum. Ono što je u početku jednoj osobi uštedelo 60 minuta rada nedeljno, sada je druge koštalo nekoliko dodatnih minuta za svakog korisnika kojeg su kreirali, što znači da ih je ta automatizacija dugoročno koštala više nego što je ikada uštedela. To je klasičan primer neplaniranja budućnosti.

Niko ne može da predvidi budućnost, ali je svakako možete planirati. Bez obzira na kojem koraku procesa automatizacije radite, morate se zapitati koliko će nešto biti teško održavati. Nakon toga, razmislite o svom iskustvu i razmislite kako se zahtevi mogu promeniti tokom vremena.

Rekli smo da je u našem scenariju čišćenja evidencije prva akcija bila da pronađemo datoteke evidencije starije od 30 dana. Jedna od prvih „stvari“ o kojima treba da razmislite je šta se dešava ako disk počne da se popunjava brže i morate da čistite evidencije na svakih 14 dana. Koliko bi bilo teško izvršiti tu promenu? To uopšte ne bi bilo teško ako biste kreirali broj dana kao promenljivu. Međutim, ako ste kodirali broj dana u svojim skriptovima, moraćete da se vratite na skript i da izvršite više promena.

Drugi scenario koji možda nije toliko jednostavan je ako se smatra da je neophodna druga fascikla evidencije. Za početak, morate da se zapitate koliko je verovatan taj scenario. Ako je verovatan, trebalo bi da razmislite da li je vredno pisati automatizaciju za obradu više putanja direktorijuma ili biste mogli da uradite nešto jednostavno, kao što je pokretanje dva puta - po jednom za svaku putanju.

Još jedan aspekt koji bi trebalo uzeti u obzir je da li morate da promenite čišćenje evidencije sa jednog dnevno na čišćenje svakog sata. Ponavljam: zapitajte se da li je to verovatan scenario. Ako jeste, odredite šta bi bilo potrebno da se automatizacija promeni na svaki sat. To može izgledati kao jednostavan odgovor – na primer, da promenite filter za dane u sate, ali takođe morate da pogledate kako bi to moglo da utiče na druge akcije. Na primer, kada kreirate arhivsku datoteku, da li dodajete vremensku oznaku nazivu? Ako dodajete vremensku oznaku, da li ona uključuje sate? Ako ne uključuje sate, možete stvoriti situaciju u kojoj ćete slučajno zameniti podatke. Odgovori na bilo koje od tih pitanja će zavisiti od vaših jedinstvenih zahteva. Naravno, nećete moći da predvidite svaku moguću situaciju, ali ako imate ta pitanja na umu i

znate kako da odgovorite na njih, koristeći PowerShell, bićete spremniji kada promene postanu neophodne.

Takođe treba da pazite da se ne „zaglavite“. Ako ste primetili, moj prvi odgovor na bilo koje pitanje je „kontrapitanje“ koliko je verovatan taj scenario. Možete se toliko „zaglaviti“ u razmatranju različitih scenarija da nikada nećete ništa postići ili ćete svoju logiku učiniti toliko složenom da je niko drugi neće razumeti. To je delikatan čin ravnoteže, kojeg ćemo se neprestano doticati u ovoj knjizi.

1.3 Proces automatizacije

Kada pogledate projekat automatizacije, videćete da je lako biti preopterećen. Ljudi će vam reći da koristite princip KISS (neka bude kratko i jednostavno). Iako je to lako reći, nije uvek lako koristiti ovaj princip u praksi. Može biti skoro nemoguće koristiti ga kada imate više sistema koji međusobno komuniciraju, složenu logiku i zahteve koji se stalno menjaju. Ovde stupaju „na scenu“ koncepti *gradivnih blokova* i *faza.* Korišćenjem gradivnih blokova i faza možete da podelite svoje složene zadatke na male, jednostavne korake.

1.3.1 Gradivni blokovi

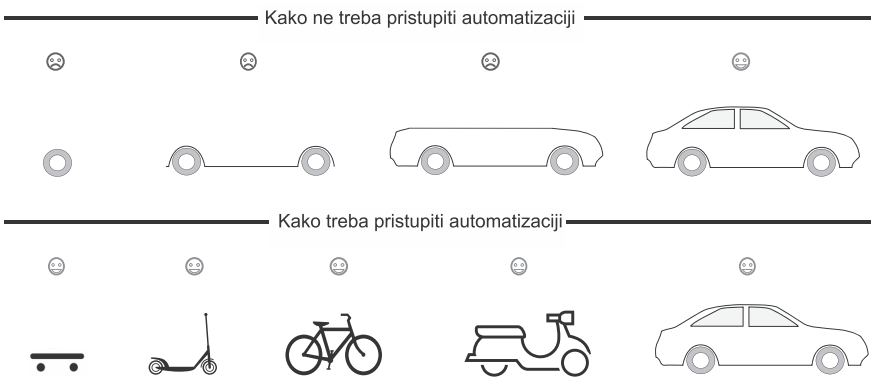
Bez obzira koliko je složena automatizacija, uvek se može podeliti na manje pojednostavljene korake ili blokove. Deljenjem zadataka na takve blokove možete sprečiti da postanete preopterećeni i da obezbedite jasne ciljeve koje možete redovno ispunjavati. Osim toga, taj koncept će vam omogućiti da koristite delove automatizacije već prvog dana rada i obezbediće vam radni okvir za proširenje vašeg ranijeg rada. Veći deo ove knjige će vam pomoći da kreirate te različite gradivne blokove koje možete koristiti u svim automatizacijama.

Gradivni blokovi takođe omogućavaju da izgradite svoje veštine tokom vremena. Kako budete sve više automatizovali, vaše veštine će nastaviti da se povećavaju. Naučićete nove tehnike, ne samo u svom kodiranju, već i u celokupnom procesu. Možda ćete pronaći bolji način da izvršite zadatak, koristeći PowerShell. Ako ste koristili gradivne blokove, možete se vratiti nazad i ažurirati sve svoje prethodne automatizacije brzo i lako.

1.3.2 Faze

Izreka „Morate da naučite da hodate pre nego što možete da trčite“ savršeno se odnosi na „svet“ automatizacije. Vaših prvih nekoliko automatizacija koje kreirate verovatno neće biti lepe – baš kao prva slika koju ste ikada nacrtali ili prvi literarni rad koji ste napisali u školi. Potrebni su vreme i iskustvo da biste usavršili svoje veštine. Međutim, to ne znači da ne možete odmah početi da uživate u prednostima automatizacije.

Deljenjem automatizacije na faze možete kreirati inkrementalne prednosti. Zamislite da treba da stignete od tačke A do tačke B. Naravno, automobilom je možda najbrže stići, ali nemate pojma kako da napravite automobil, a čak nemate ni resurse. Dakle, započnite rad malim koracima i napredujte. Prvo napravite skejtbord. Zatim ga nadgradite na skuter, bicikl i motocikl i, na kraju, napravite potrebni automobil. Na slici 1.2 su prikazane koristi faznog pristupa automatizaciji.



Slika 1.2 Način na koji će fazni pristup omogućiti da ranije počnete da ostvarujete koristi

Svakim korakom na „putu“ ćete nastaviti da poboljšavate svoj proces. Osim toga, videćete koristi od samog početka, koje ne biste videli da ste krenuli da pravite automobil „od nule“ - u toj situaciji hodali biste sve vreme dok konačno ne biste napravili automobil.

Tokom svake faze najverovatnije ćete kreirati nekoliko gradivnih blokova. Štaviše, te gradivne blokove koje kreirate često ćete koristiti u različitim fazama i poboljšavaćete ih od jedne do druge faze. Na primer, na slici 1.2 naučili ste da pravite točak u fazi 1. Zatim ste u fazi 2 poboljšali to znanje i napravili još bolji točak.

Faze takođe omogućavaju da se usput prilagodite automatizaciji. Možete dobiti povratne informacije nakon svake faze od ljudi koji je koriste. Možda ćete otkriti da postoje „stvari“ koje niste uzeli u obzir. U scenariju na slici 1.2, nakon što ste napravili skejtbord, ljudi su vam rekli da je odličan za neki deo puta, ali ne i za neke njegove blatnjave delove. Možete uzeti u obzir tu povratnu informaciju i prilagoditi fazu 2 da uključuje veće točkove. Uporedite taj pristup sa pristupom pravljenja automobila odmah bez faza, a zatim otkrijte da se taj automobil zaglavio u blatu. Da niste napravili amortizer i otvore za točkove tako da odgovaraju većim točkovima, trebalo bi da izvršite mnogo dorade.

1.3.3 Kombinovanje gradivnih blokova i faza

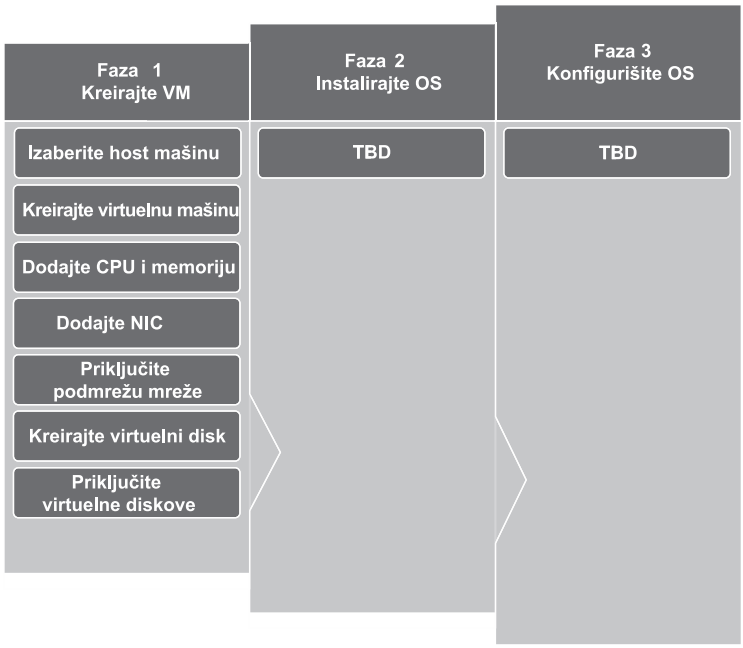
Da biste demonstrirali koncept gradivnih blokova i faza na način koji se više fokusira na IT, možete pogledati uobičajeni scenario automatizacije obezbeđivanja virtuelne mašine. Iako u tom procesu može biti mnogo čega, možete ga podeliti u nekoliko faza:

1. Kreirajte virtuelnu mašinu.
2. Instalirajte operativni sistem.
3. Konfigurišite operativni sistem.

Iako bi bilo sjajno da se pozabavite svim tim fazama odjednom, to bi bio ogroman poduhvat i ne biste videli nikakve koristi do samog kraja. Umesto toga, možete se pozabaviti jednom po jednom fazom, obezbeđujući sebi dodatne pogodnosti na tom „putu“. Počnite rad od faze 1, tako što ćete kreirati virtuelnu mašinu. Gradivni blokovi za to bi se mogli sastojati od:

1. odabira host mašine
2. kreiranja prazne virtuelne mašine
3. dodeljivanja CPU-a i memorije
4. priključivanja kartice mrežnog interfejsa na odgovarajuću podmrežu
5. kreiranja i priključivanja virtuelnih čvrstih diskova

Kada završite fazu 1 (kreiranje virtuelne mašine, koje prikazano na slici 1.3), možete da pređete na fazu 2 dok već ubirete „plodove“ faze 1.



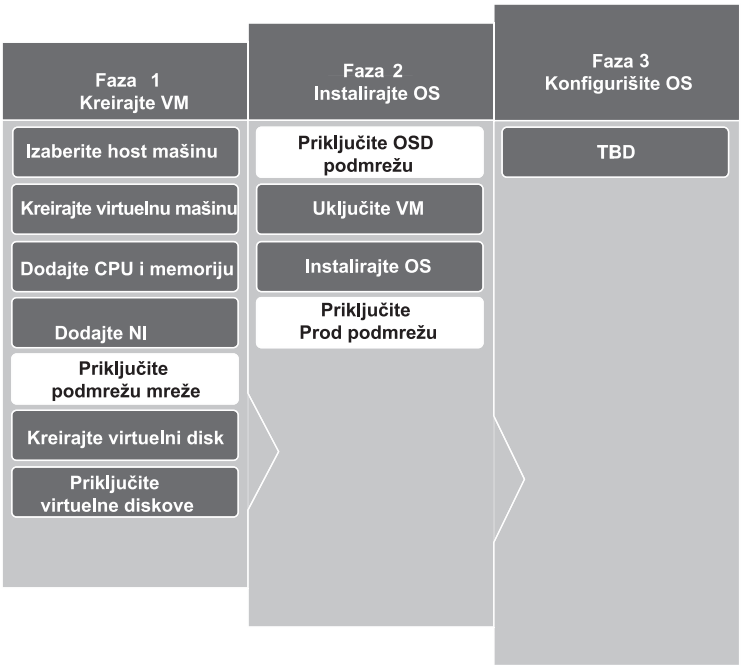
Slika 1.3 Fazni pristup obezbeđivanju virtuelne mašine (faza 1)

U fazi 2 ćete instalirati operativni sistem. Ovde imate nekoliko opcija. Možete kreirati šablon virtuelnog hard diska sa već instaliranim operativnim sistemom. Međutim, to bi značilo da morate održavati šablon i primeniti „zakrpe“. Ako imate više hostova u različitim regionima, moglo bi biti teško da se pobrinete da svi budu sinhronizovani. Umesto toga, odlučili ste da koristite svoje alatke za upravljanje konfiguracijom da biste instalirali operativni sistem, pa je vaša slika dosledna u vašem okruženju i uvek ažurna.

Kada počnete da „gradite“ taj deo automatizacije, shvatate da vaša virtuelna mašina mora da bude na određenoj podmreži da bi primila sliku. Vaši gradivni blokovi mogu biti slični sledećim gradivnim blokovima:

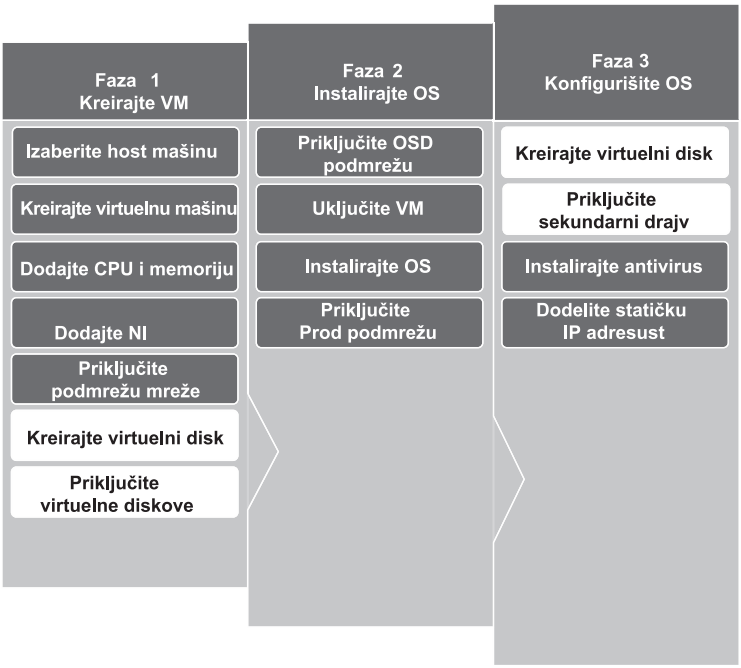
1. Priključite podmrežu za implementaciju operativnog sistema.
2. Uključite virtuelnu mašinu.
3. Sačekajte da se operativni sistem instalira.
4. Priključite proizvodnu podmrežu.

Pošto ste kreirali blok za dodeljivanje virtuelne mašine podmreži u fazi 1, možete ponovo koristiti taj kod za blokove 1 i 4 u toj fazi. Primetićete da sam podmreži priključio poseban blok, zato što sam ranije automatizovao upravo ovaj scenario i više puta sam naišao na tu situaciju. Ako kombinujete sve resurse u jedan blok, tj. ako dodelite CPU i memoriju, priključite mrežu i dodelite virtuelni čvrsti disk, ne možete ponovo koristiti taj blok. Ako želite da se povežete sa drugom mrežom, možete ponovo da dodelite CPU i memoriju, ali dodeljivanje drugog virtuelnog čvrstog diska može izazvati značajne probleme. Ako uradite nešto ovako, ne brinite. Zamislite to kao iskustvo na osnovu učenja. I dalje to stalno radim. Osim toga, pošto ćete morati da kreirate gradivni blok da biste dodelili podmrežu za tu fazu, nema razloga zbog kojeg se ne možete vratiti i ažurirati blokove u prethodnoj fazi. Na slici 1.4 je prikazan taj razvoj u fazi 2.



Slika 1.4 Fazni pristup obezbeđivanju virtuelne mašine (faza 2) pomoću deljenih komponenata

Sada imate dve faze u proizvodnji i korisnici počinju da vide stvarne prednosti. Osim toga, u sledećoj fazi, prikazanoj na slici 1.5, naučite šta bi im koristilo. Možete razgovarati sa ljudima koji koriste automatizaciju i otkriti šta bi želeli da vide u fazi 3. To može biti dodeljivanje statičke IP adrese, kreiranje sekundarnih diskova podataka ili bilo koji drugi broj „stvari“ koje možda niste razmatrali. Takođe, ne morate da stanete nakon faze 3. Možete dodati fazu 4 da biste automatski instalirali aplikacije.



Slika 1.5 Fazni pristup obezbeđivanju virtuelne mašine (faza 3) pomoću deljenih komponenata

Najznačajnija korist kombinovanja koncepta gradivnih blokova i faza je fleksibilnost – ne samo tokom procesa kreiranja, već i kasnije. Ako se vaši zahtevi ili resursi promene, potrebno je samo da zamenite gradivne blokove specifične za tu promenu. Sam proces i ostali gradivni blokovi će ostati nepromenjeni.

Zamislite da je vaša kompanija odlučila da pređe na novi hipervizor ili na „oblak“. U tim slučajevima trebalo bi da ponovite fazu 1. U fazi 2 jednostavno morate da zamenite blokove mrežnog dodeljivanja novim blokovima koje ste izgradili. Ostatak faze 2 ostaje isti. Alternativno, recimo da je vaša kompanija odlučila da pređe na drugi operativni sistem. Bilo bi malo ili nimalo promena u fazi 1, a možda i nekih manjih promena u fazi 2. Sve promene bi se fokusirale na fazu 3. Ako ste koristili fazni pristup, bez obzira u kojoj situaciji se nađete, moći ćete da se brzo prilagodite.

1.4 Odabir odgovarajuće alatke za posao

Jedna od najvećih grešaka koje možete da načinite kada nastojite da automatizujete zadatak je pokušaj da primorate alatku da uradi nešto za šta nije dizajnirana. Stoga, pre nego što započnete bilo koji PowerShell projekat automatizacije, morate da utvrdite koja alatka je najbolja za posao.

Na primer, ne bih preporučio korišćenje Pythona ako postavljate resurse u Azureu, ali ne zato što je on loša alatka (daleko od toga), već zato što nema istu izvornu podršku za Azure resurse. Pozvaćete Azure CLI pomoću Pythona, što može izazvati drugi skup problema. Sada vaš Python skript zavisi od instaliranja Azure CLI-a. Pošto je Azure CLI samostalna aplikacija, a ne paket za Python, moraćete da ugradite posebne provere u vaš skript da biste bili sigurni da su datoteke koje su vam potrebne dostupne. Osim toga, vaš skript sada zavisi od platforme koja podržava i Python i Azure CLI. To dramatično povećava složenost vaše automatizacije i čini je mnogo manje prenosivom.

Sada, ako odaberete PowerShell za taj zadatak, možete za izvršavanje vaših zadataka koristiti Azure PowerShell module koje je kreirao i održava „Microsoft“. Sve funkcije za proveru i rešavanje problema zavisnosti su ugrađene u PowerShell. Pomoću dve ili tri linije koda vaš skript možete učiniti potpuno prenosivim na bilo koji drugi sistem koji koristi PowerShell.

Ne kažem da je PowerShell jedini izbor, ali za određene poslove jednostavno ga je logično koristiti. Zahvaljujući PowerShell Coreu, broj zadataka koje možete da automatizujete pomoću PowerShella postaje sve veći, iako još uvek nisu obuhvaćeni svi zadaci. Ako treba da izvršite tehničku analizu kao deo vaše automatizacije, kao što su izračunavanje i crtanje statističkih grafikona, ne bih preporučio PowerShell. U tom slučaju biblioteka pana-das u Pythonu je mnogo bolja od onoga što je dostupno u PowerShellu.

1.4.1 Stablo odlučivanja o automatizaciji

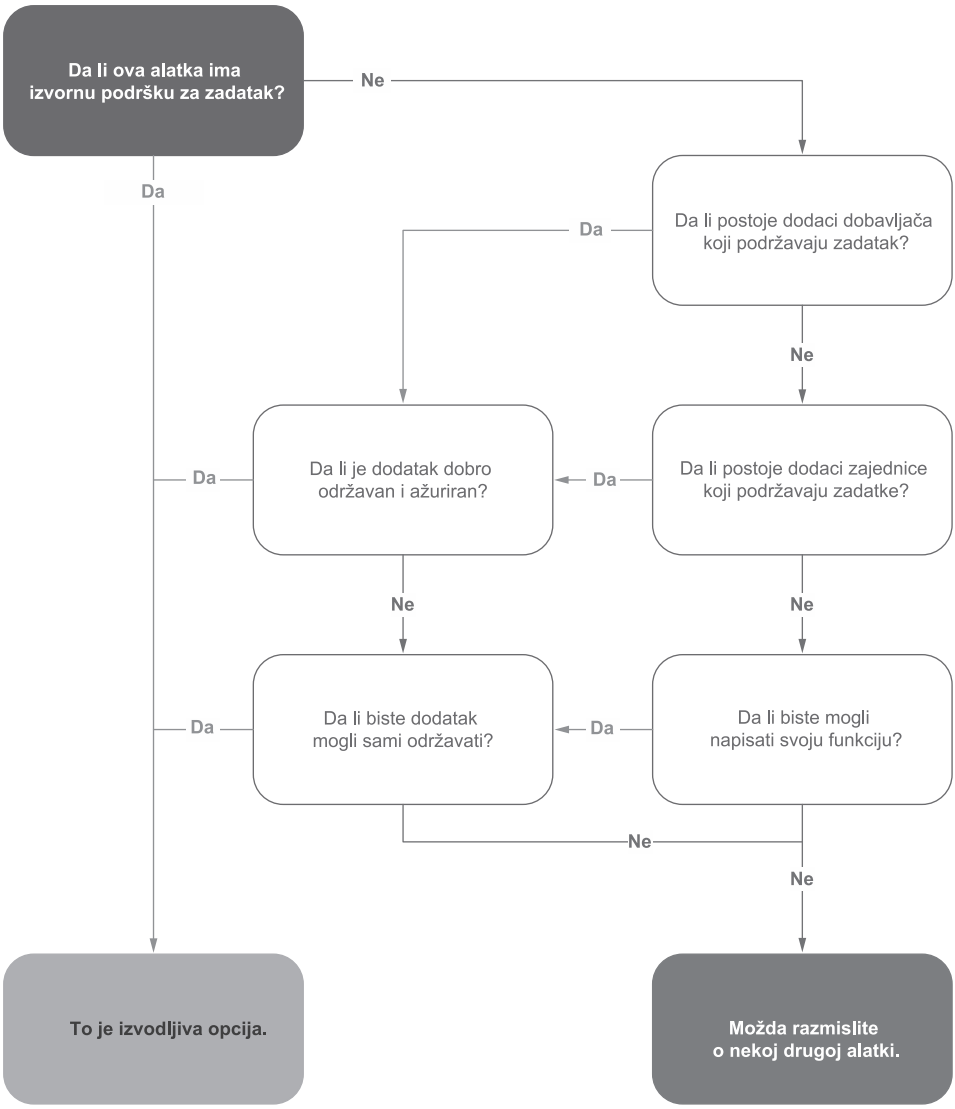
Kako utvrditi da li je PowerShell odgovarajuća alatka za posao? Jedan od načina je korišćenje stabla odlučivanja, prikazanog na slici 1.6.

Kada koristite stablo odlučivanja, morate da pogledate sve aspekte procesa automatizacije koji kreirate. Na primer, vratimo se na naš prethodni primer arhiviranja starih datoteka evidencije i dodajmo zahtev za njihovo otpremanje u Azure Blob skladište. Prva akcija je bila pronalaženje datoteka starijih od 30 dana. Ako sve akcije tog primera primenimo na stablo odlučivanja, to bi izgledalo otprilike ovako:

- Da li ta alatka ima izvornu podršku za sve zadatke koje treba da izvršim? *Da, PowerShell ima ugrađenu funkcionalnost za rad sa sistemima datoteka.*

Nema potrebe da nastavljamo postavljanje drugih pitanja, jer je odgovor na prvo pitanje definitivno *potvrdan*. Sledećih nekoliko akcija u procesu će biti slične. Na primer, kada kreirate arhivsku datoteku, zapitajte se:

- Da li ta alatka ima izvornu podršku za sve zadatke koje treba da izvršim*? Da,* `Compress-Archive cmdlet` *je izvorni za PowerShell.*



Slika 1.6 PowerShell stablo odlučivanja možete koristiti da biste utvrdili da li je PowerShell odgovarajuća alatka za posao

Međutim, neće sve akcije biti toliko jednostavne. Uzmite kao primer akciju kopiranja datoteka u Azure Blob skladište:

- Da li ta alatka ima izvornu podršku za sve zadatke koje treba da izvršim? *Ne.*
- Da li postoje moduli/dodaci kompanije koji mogu da izvrše zadatke? *Da, „Microsoft“ ima oficijalni modul Azure Blob Storage.*

Ponavljam: ti odgovori su prilično jasni, jer znamo da „Microsoft“ kreira oficijelne PowerShell module za podršku svih Azure funkcionalnosti. Međutim, biće slučajeva, čak i unutar Microsoft eko-sistema, kada odgovor možda neće biti toliko jasan. Na primer, recimo da za akciju evidentiranja datoteka koje su uklonjene morate da upišete te datoteke u SQL tabelu:

1. Da li ta alatka ima izvornu podršku za sve zadatke koje treba da izvršim? *Ne.*
2. Da li postoje moduli/dodaci kompanije koji mogu da izvrše te zadatke? *Postoji „Microsoftov“ modul SqlServer, ali on ne podržava sve zadatke koje želim da automatizujem.*
3. Ako ne postoje moduli/dodaci kompanije, da li postoje moduli/dodaci zajednice koji mogu da izvrše zadatke? *Da. Modul dbatools je dostupan u PowerShell galeriji.*
    a. Da li se modul održava i ažurira? *GitHub spremište ima više od 15.000 izvršavanja i 200 saradnika i redovno se ažurira.*
4. Koliko bi bilo teško napisati prilagođenu funkciju? *Moguće je postaviti SQL upit direktno iz PowerShella, koristeći klasu System.Data.SqlClient, koja je izvorna u .NET-u.*
    a. Da li će biti teško održavati tu funkciju? *Možda postoje razlike između .NET-a i .NET Corea za klasu SqlClient.*

Kao što vidite, postoji mnogo načina na koje možete izvršiti zadatak. Vaš posao će biti da donesete odluku na osnovu informacija koje alatke su najpogodnije za zadatak koji imate. Naravno, možda ćete otkriti da ni jedna alatka ne može zadovoljiti sve vaše potrebe, što je takođe u redu. Kada koristite PowerShell, možete lako da se „prebacujete“ između različitih rešenja da biste postigli svoje ciljeve. Nakon što pročitate knjigu, moći ćete da identifikujete zadatke za koje možete da koristite PowerShell.

1.4.2 Nema potrebe da se ponovo „izmišlja topla voda“

Ono što je sjajno u vezi sa PowerShellom je velika zajednica koja voli da deli svoje znanje. U vreme pisanja ovog teksta više od 6.400 različitih PowerShell modula je dostupno u zvaničnoj PowerShell galeriji. Postoje i brojne veb stranice, forumi i blogovi posvećeni PowerShellu. Dakle, ako postoji nešto što pokušavate da uradite pomoću PowerShella, velika je verovatnoća da je neko već uradio to ili nešto slično.

Nema potrebe da pišete svaku liniju koda u svojim skriptovima „od nule“. Podstičem vas da istražite šta su drugi ljudi uradili. Učite iz njihovih grešaka i iskustava. Bezbroj puta sam video blok koda koji je kreirao neko i pomislio u sebi zašto je uradio na taj način. Onda to napišem na drugi način i naiđem na problem, pa shvatim da je zato drugi skript to uradio.

U isto vreme nemojte samo da kopirate i nalepite kod pomoću GitHuba ili StackOverflowa u svoj skript i da očekujete da sve funkcioniše. Umesto toga, pogledajte kod. Otkrijte šta tačno on radi i kako izvršava svoj zadatak. Zatim, možete da implementirate kod u svoj skript sa uverenjem da će funkcionisati i, što je najvažnije, da ćete moći da ga održavate.

1.4.3 Dodatne alatke

Iako PowerShell može da uradi mnogo štošta, postoji nešto što ne može da uradi. Na primer, nema frontend koji može da obezbedi obrasce koje korisnici mogu da popune. Takođe nije planer poslova i nema ugrađene okidače, kao što su webhooksi. Iako kreiranje nekih od tih funkcionalnosti pomoću PowerShella nije tehnički nemoguće, možda nije praktično. Postoje i druge alatke koje su izrađene posebno za te zadatke, a mnoge od njih podržavaju PowerShell.

Međutim, kao što ćete videti u ovoj knjizi, nema razloga zbog kojeg ne možete kombinovati više alatki. Na primer, u Poglavlju 3 ćete naučiti kako da koristite više alatki za planiranje poslova koji se izvršavaju, a u Poglavlju 11 ćete videti kako da koristite alatku SharePoint za kreiranje frontend obrazaca za vaše automatizacije.

PLANER POSLOVA

PowerShell nema ugrađeni planer poslova. Možda vam je poznata funkcija cmdlet `Register-ScheduledJob` , ali ona je samo kreirala PowerShell poslove u Windows Task Scheduleru. Da bi se postigla odgovarajuća podrška za više platformi pomoću PowerShell Corea, ta funkcija je uklonjena iz verzije 6.0 i novijih verzija. Naravno, još uvek možete da koristite Task Scheduler da biste planirali i pokrenuli svoje PowerShell skriptove u Windowsu, baš kao što možete da koristite Cron u Linuxu, ali postoje i druge alatke koje su namenski izrađene za rukovanje poslovima automatizacije.

Ako već koristite alatke, kao što su Jenkins, Ansible ili Control-M, možete da koristite PowerShell unutar tih platformi da biste ispunili svoje zahteve za automatizaciju. Najbolje je što vaše automatizacije onda neće zavisiti od platformi. Na primer, ako ste uložili svoje napore u rešenja, kao što su IFTTT ili System Center Orchestrator, sada ste „zaključani“ na tim platformama. Ako je taj softver zastareo, promenio svoju licencu ili uklonio funkcionalnost, vaš jedini smer akcije je da ponovo kreirate celu automatizaciju. Međutim, ako izradite automatizaciju pomoću PowerShella u Jenkinsu i vaša kompanija odluči da pređe na Ansible, lako možete preneti svoje skriptove za automatizaciju sa jedne platforme na drugu, uz minimalan napor.

FRONTEND

Sve što je prethodno pomenuto se može reći i za frontend obrasce. Frontend je samo način za prikupljanje informacija za vašu automatizaciju. Tehnički, možete da kreirate obrasce u PowerShellu i postoje slučajevi u kojima ima smisla to učiniti, ali postoji mnogo upozorenja. Baš kao u planeru poslova, dostupne su brojne alatke koje čine kreiranje i predstavljanje obrazaca jednostavnim i lakim.

Možete da kreirate sve akcije za svoje automatizacije u PowerShellu, a zatim da ih povežete na bilo koji način. Na primer, možete da kreirate SharePoint listu da biste prikupili potrebne informacije za vašu automatizaciju u roku od nekoliko minuta. Sve što treba da uradite je da kreirate jednostavan okidač koji šalje potrebne informacije vašoj automatizaciji. Ako želite da pređete na ServiceNow, nema problema. Jednostavno, preslikajte svoj okidač sa SharePointa na ServiceNow i vaša automatizacija će nastaviti da funkcioniše kao i ranije.

1.5 Šta vam je potrebno da započnete rad danas

Iako je PowerShell Core međuplatformska alatka, većina primera u ovoj knjizi biće pokrenuta u Windows okruženju. Preporučujem korišćenje Windowsa 11 ili Windows Servera 2022, ali trebalo bi da pratite primere u knjizi korišćenjem bilo koje verzije Windowsa koja podržava Windows PowerShell 5.1 i PowerShell 7. Osim ako nije drugačije naznačeno, možete pretpostaviti da je sve u ovoj knjizi napisano za PowerShell 7.

Takođe će vam biti potrebno integrisano razvojno okruženje za pisanje koda. Iako je ugrađeni PowerShell ISE već dugi niz godina glavni izbor, on ne podržava PowerShell 7. Ako to već niste uradili, preporučujem da pređete na Visual Studio Code (VS Code). Za razliku od tradicionalnog Visual Studioa, VS Code je besplatan i lak uređivač koda koji je otvorenog koda, međuplatformski i vođen zajednicom. Osim toga, VS Code podržava većinu uobičajenih jezika za programiranje i pisanje skriptova, kao što su Windows PowerShell i PowerShell, koje možete da koristite zajedno.

Ono što PowerShell čini toliko svestranim je da se može koristiti na velikom broju platformi, uključujući i Windows, Linux, macOS, servere, kontejnere, platforme nezavisnih proizvođača i mnoge platforme u „oblaku“. Ne samo da se PowerShell može pokrenuti na tim platformama, već se može koristiti i za automatizaciju upravljanja njima. U vreme pisanja ove knjige velika „stvar“ u softverskoj industriji su kontejneri. Do sledećeg meseca ili sledeće godine, ko zna šta će biti. Zbog toga je većina primera u ovoj knjizi dizajnirana tako da možete koristiti vaše lokalne resurse.

Budući da većina usluga u „oblaku“ ili PaaS-u ima različite protokole za autentikaciju ili manje razlike u načinu na koji upravljaju skriptovima, bilo bi nemoguće opisati svaku potencijalnu uslugu. Umesto toga, u ovoj knjizi ćete naučiti osnove koje će ostati iste, bez obzira koju platformu koristite ili kojom platformom upravljate. Naučićete kako da razmišljate, identifikujete i radite pomoću platforme koju izaberete.

Iako se u primerima u ovoj knjizi koriste nezavisne platforme ili rešenja u „oblaku“, sve platforme su ili besplatne ili imaju besplatnu probnu verziju koju možete da koristite. To uključuje Jenkins, Azure, SharePoint i GitHub. Preciznije detalje o okruženjima i alatkama koji se koriste možete pogledati u „Dodatku“ ove knjige.

Rezime

- PowerShell je moćan jezik visokog nivoa dizajniran sa IT automatizacijom, koji je lako preuzeti i početi koristiti.
- Možete da koristite PowerShell za kreiranje blokova za višekratnu upotrebu koji se mogu deliti između automatizacija i među članovima vašeg tima.
- Da biste kreirali uspešnu automatizaciju, morate da konceptualizujete proces i planirate budućnost.
- PowerShell je proširiva i prenosiva alatka, što ga čini dobrim za većinu potreba automatizacije.
- PowerShell može da funkcioniše zajedno sa drugim alatkama i platformama kako bi brzo i lako zadovoljio većinu vaših potreba.
- PowerShell ima veliku zajednicu i podržava ga jedna od najvećih tehnoloških kompanija na svetu.