Sadržaj

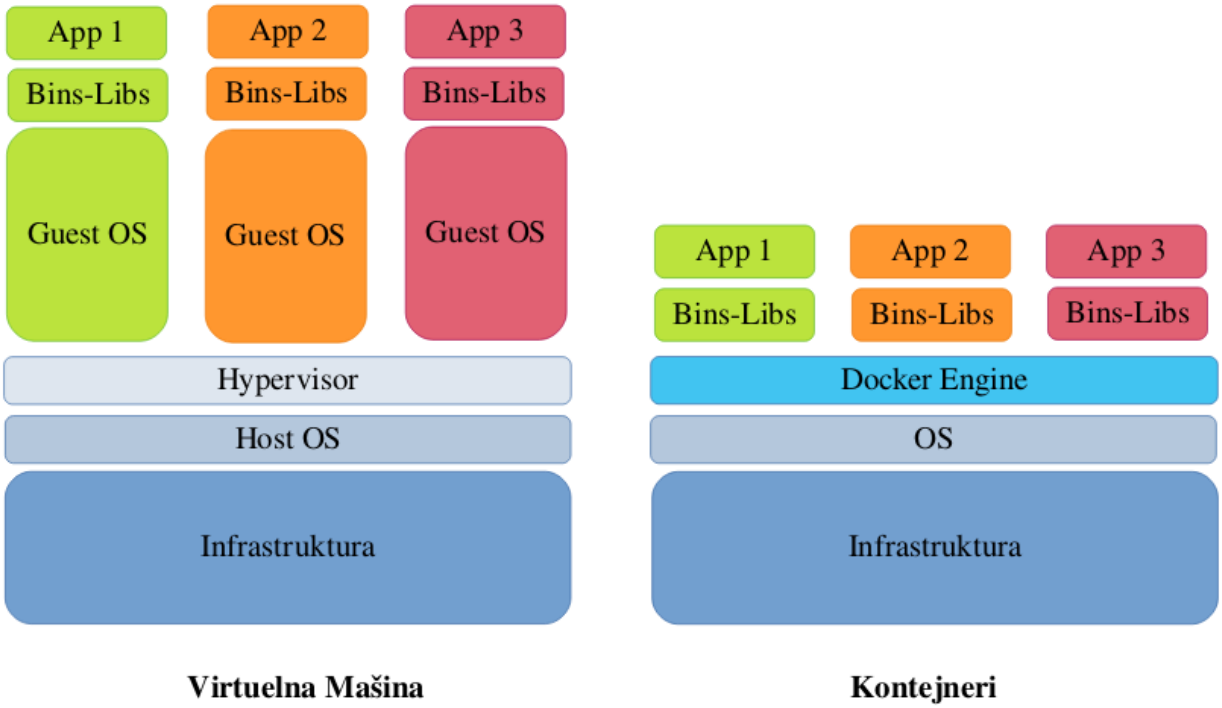
Spisak slika

Skraćenice

API – *Application programming interface*, Interfejs za programiranje aplikacija

1. Uvod

U današnje vreme



Slika 1.1: Razlika između arhitekture Virtuelne Mašine a arhitekture kontejnera

2. Teorijske osnove

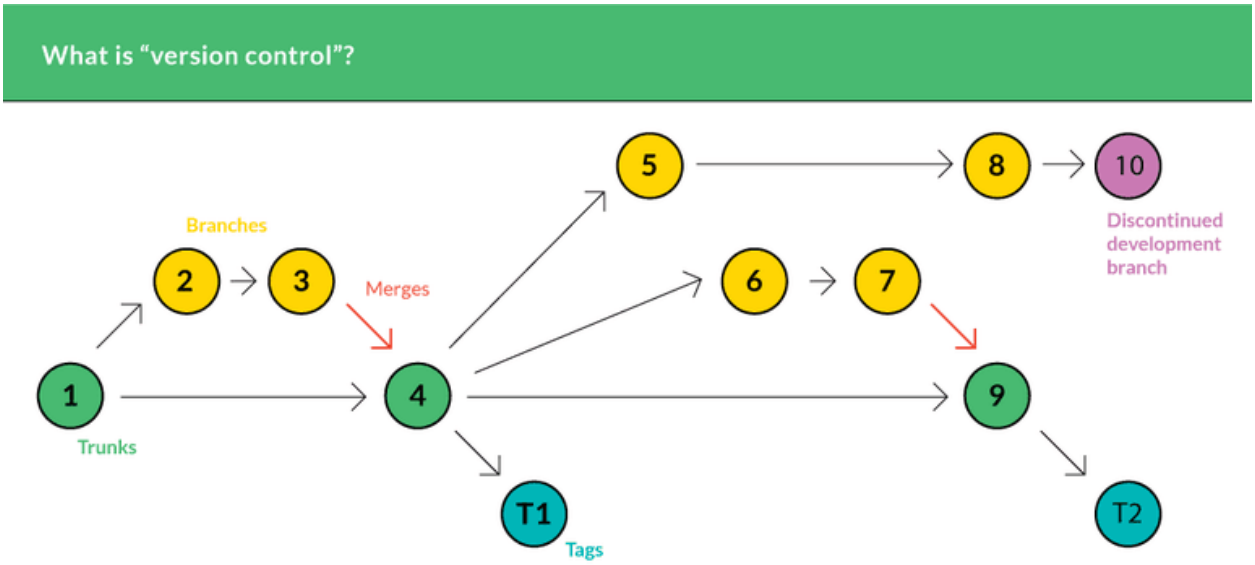
U ovom poglavlju će biti ukratko objašnjeni svi koncepti i alati koji su bili potrebni za izradu projekta o kojem ovaj rad govori. NABROJ

2.1 Open source razvoj

2.2 Sistem za praćenje verzija datoteka

Sistem za praćenje verzija datoteka (eng. *version control system*) je sistem zadužen za praćenje istorije promena u datotekama. Promena koja se pamti se najčešće naziva revizija. Sve promene se uglavnom čuvaju na centralizovanom sistemu za skladištenje podataka i svi korisnici sa potrebnim privilegijama mogu da mu pristupaju, preuzmu najnoviju verziju datoteke ili grupe datoteka, izmene ih i zatim objave novu verziju koja se od tog momenta smatra najnovijom.

Pored najnovije verzije, čuva se i celokupna istorija promena i moguće je vratiti datoteke u bilo koju od prethodnih revizija. U poglegu softvera (eng. *software*), ovo nam omogućava da vratimo u upotrebu bilo koju prethodnu verziju, u slučaju da dođe do greške u nekoj od novijih verzija. Istorija se čuvaju u strukturi tipa grafa:



Slika 2.1: napravi novu sliku

Graf se najčešće sastoji od glavne grane(eng. *branch*), na kojoj se nalaze glavne izmene, i pomoćnih grana na kojima se nalaze izmene za koje još nije odlučeno da li će pripadati glavnoj grani. Ako sve izmene na pomoćnim granama budu odgovarajuće, one se spajaju (eng. *merge*) sa glavnom granom i postaju deo njenog sadržaja. U slučaju praćenja verzija softvera, ovakve grane najčešće predstavljaju nove funkcionalnosti (eng. *feature*) koje se dodaju na glavnu granu projekta ili ispravljanje greške (eng. *bugfix*) koja je nastala u nekoj od prethodnih verzija.

Važne izmene se mogu dodatno naglasiti dodavanjem oznake (eng. *tag*). One uglavnom predstavljaju objavljivanje nove verzije softvera (eng. *release*), dodavanje nove funkcionalnosti ili ispravku greške.

2.3 Git

Git [1] je besplatan softver otvorenog koda (eng. *open-source*) za praćenje verzija datoteka u distribuiranom okruženju. Primarno se koristi za razvoj softvera, ali se može koristiti i za praćenje bilo kojih drugih datoteka. Razvoj je započeo 2005. godine kreator Linux-a, Linus Torvalds, za potrebe razvoja jezgra Linux operativnog sistema. Distribuiran je pod GNU General Public License Version 2 licencom. Primarno je napravljen za Linux operativni sistem, ali je naknadno prilagođen i za macOS, BSD, Solaris i Windows.

Za razliku od drugih sistema za praćenje verzija datoteka u distribuiranom okruženju, Git nije softver klijent-server arhitekture. Svaki Git repozitorijum na svakom računaru čuva kompletnu istoriju promena i verzija i nije zavisan od pristupa mreži ili centralizovanom serveru. Kao takav, Git je osnova za razvoj mnogih drugih nezavisnih alata za praćenje verzija datoteka.

2.4 Github

2.5 Github akcije - mozda spojiti sa prethodnim

2.6 Python

Python [2] je programski jezik opšte namene, spada u grupu programskih jezika visokog nivoa i interpretira se. Dizajniran je tako da bude izrazito čitljiv, što se postiže velikom upotrebom belina (eng. *whitespace character*). Pogodan je za razvoj različitih projekata pošto podržava više programskih paradigmi:

1. Proceduralnu
2. Objektno orijentisanu
3. Funkcionalnu
4. Strukturalnu paradigmu

Takođe, prilagođen je za korišćenje u radličitim okruženjima na različitim operativnim sistemima.

Python se prvi put spominje krajem 1980-tih kao naslednik ABC programskih jezika. Dizajnirao ga je Guido van Rossum i objavio prvu verziju 1991. godine. Python 2.0, koji je izdat 2000. godine je znatno unapređenje prve verzije, koje se do skoro koristilo. Poslednja velika revizija urađena je 2008. godine kada je nastao Python 3.0. Zbog toga što Python 3 nije u potpunosti kompatibilan sa prethodnim verzijama, podrška za Python prestala je u januru 2020. godine.

Umesto da poseduje svu svoju funkcionalnost u jezgru OS-a. Python je dizajniran tako da bude veoma proširiv, što ga je učinilo veoma pogodnim za dodavanje programabilnih interfejsa postojećim aplikacijama.

2.7 Python paketi, PyPI

Python paketi su imenski prostori (eng. *namespaces*) koji u sebi sadrže druge pakete i module - delove softvera koji obavljaju konkretnu funkcionalnost. Svaki paket je direktorijum koji u sebi mora imati *__init__.py* datoteku. Ova datoteka može biti prazna, što označava da je trenutni

direktorijum Python paket i da može biti korišćen kao zaseban modul. Inače, *__init__.py* datoteka čuva kod koji služi za inicijaliciju tog paketa.

Da bi se koristile funkcionalnosti koje neki paket nudi, potrebno ga je uvezati (eng. *import*) u projekat korišćenjem rezervisane reči `import`. Da bi se uvezao pojedinačni modul iz nekog paketa, treba naznačiti koji modul se iz kojeg paketa uvezuje korišćenjem sledećeg Python koda:

- `from <package> import <module>`

Glavna uloga Python paketa je distribuiranje biblioteka za Python programski jezik. Paket može da kreira bilo ko i da ga potom podeli sa ostatkom Python zajednice. Da bi paket bio dostupan, potrebno je kreirati arhivu sa svim potrebnim konfiguracijama za njega i objaviti je na nekom repozitorijumu za Python pakete.

Repozitorijumi su softverski alati za čuvanje i distribuiranje paketa. Postoje javni repozitorijumi, koji omogućavaju razmenu paketa sa bilo kim, i privatni, koji omogućavaju razmenu paketa u ograničenoj grupi korisnika koji imaju specijalne privilegije.

Najčešće korišćen javni repozitorijum je PyPI [3] - The Python Package Index.

2.8 Poetry

Poetry [4] je alat otvorenog koda za kreiranje Python paketa i upravljanje paketima koje Python projekat koristi(eng. *dependency management*). Pomaže korisnicima tako što umesto njih instalira i menja verzije bibliotekama od kojih projekat zavisi, a koje korisnik mora pretkodno da specificira. Poetry je takođe znatno olakšao kreiranje, pakovanje i objavljivanje paketa na PyPI repozitorijumu i time omogućio lakše deljenje Python projekata sa drugim korisnicima Python programskog jezika.

Prva verzija alata objavljena je 28.02.2018. godine. Verzije programskog jezika Python koje su podržane su 2.7 i 3.4+. Ideja je da Poetry radi podjednako dobro na različitim platformama, između ostalog na Linux-u, Windows-u i OSX-u.

2.8.1 Podešavanje Poetry-ja

Za razliku od drugih Python alata za upravljanje paketima od kojih projekat zavisi, umesto pip-a - instalera za Python pakete, Poetry koristi svoj specifičan način za instalaciju. Instalator doda Poetry u korisnički direktorijum, tako da može da se koristi sa bilo kojom verzijom Python-a. Omogućena je i instalacija Poetry-ja korišćenjem pip-a, ali se ne proporučuje iz dva razloga:

1. može dovesti do konflikta sa drugim sistemskim fajlovima

2. otežava održavanje konzistentnosti pre korišćenju različitih verzija Python-a i različitih virtuelnih okruženja (eng. *virtual environments*)

2.8.2 Kreiranje Python projekta koji koristi Poetry

Nakon isntalacije alata, moguće je kreirati projekat koji koristi Poetry za upravljanje paketima ili dodati Poetry u postojeći projekat i time kreirati virtuelno okruženje u kojem će se izvršavati naredne akcije. U oba slučaja će se kreirati *pyproject.toml* datoteka koja čuva sve bitne informacije o projektu. U početku ova datoteka sadrži samo osnovne informacije o projektu:

```
[tool.poetry]
name = "poetry-demo"
version = "0.1.0"
description = ""
authors = ["Name Surname <email>"]

[tool.poetry.dependencies]
python = "*"

[tool.poetry.dev-dependencies]
pytest = "^3.4"
```

Informacije o samom projektu, njegovom autoru, verziji, licenci, opisu, dokumentaciji, itd. nalaze se u prvom segmentu - `[tool.poetry]`. Informacije o paketima koje projekat koristi nalaze se u naredne dve sekcije - `[tool.poetry.dependencies]` i `[tool.poetry.dev-dependencies]`, gde se u prvoj od ove dve sekcije nalaze informacije o paketima koji se koriste u produkcionom okruženju, a u drogoj informacije o paketima koji se koriste u toku razvoja projekta. Pri instalaciji novog paketa, njegov naziv i verzija će se dopisati u jednu od ove dve sekcije, ili u obe ako je to potrebno.

Ovu datoteku je moguće ručno menjati, ali se sa njom najčešće interaguje pozivanjem poetry komandi u konzoli. Neke od tih komandi su:

- `poetry add` - dodaje novi paket u *pyproject.toml* datoteku
- `poetry remove` - briše paket iz projekta

Komande koje takođe interaguju sa *pyproject.toml* datotekom, ali je ne menjaju su:

- `poetry install` - instalira pakete definisane u *pyproject.toml* datoteci
- `poetry update` - ažurira pakete u skladu sa verzijama koje pišu u *pyproject.toml* datoteci
- `poetry lock` - zaključava pakete u prijektu
- `poetry check` - proverava ispravnost *pyproject.toml* datoteke

Za pokretanje komandi u virtuelnom okruženju opsisanom u *pyproject.toml* datoteci koristi se komanda:

- `poetry run`

Za sve ostale detalje najbolje je pogledati dokumentaciju pomoću komande:

- `poetry help`

2.8.3 Kreiranje, pakovanje i objavljivanje Python paketa

Da bi se Python projekat mogao objaviti, potrebno je kreirati arhivu sa svim potrebnim konfiguracijama za njega. To se postiže upotrebom komande:

- `poetry build`

Nakon toga, paket je spreman za objavljivanje. U projektu se pojavi novi direktorijum sa nazivom `dist` i u njemu se nalaze dve datoteke:

1. `poetry-demo-0.1.0-py2.py3-none-any.whl`
2. `poetry-demo-0.1.0.tar.gz`

Repozitorijum na kojem se paket objavljuje se podešava pomoću komande:

- `poetry config`

Paket se objavljuje pozivanjem komande:

- `poetry publish`

U tom momentu, paket postaje dostupan na izabranom repozitorijumu i svako ko ima pristup repozitorijumu može da instalira paket. Najčešće se paketi objavljuju na PyPI repozitorijumu.

2.9 Bash

3. Specifikacija i implementacija projekta

3.1 pjisp-assignment-template

3.2 pjisp-template-name

3.2.1 način imenovanja repo-a

3.3 pjisp-diff

3.4 poetry-publish action

Koristi se za publishovanje pjisp-diff i pjisp-template-name

3.5 Error code u smoke-test-u - zasto je potreban?

4. Primeri korišćenja

5. Diskusija i zakljuci

5.1 Da li su github akcije spremne za produkciju?

6. Literatura

[1] Git. *Git*. URL: `https://git-scm.com/`.

[2] Python Software Foundation. *Python documentation*. URL: `https://www.python.org`.

[3] Python Software Foundation. *The Python Package Index*. URL: `https://pypi.org`.

[4] Poetry. *Poetry documentation*. URL: `https://python-poetry.org`.