Programske paradigme

— Skript programiranje —

Milena Vujošević Janičić

www.matf.bg.ac.rs/~milena

Matematički fakultet, Univerzitet u Beogradu

Pregled

Pregled

Popularnost skript jezika

- Skript programiranje je važan stil programiranja u velikom broju različitih domena
- Prema trenutnom TIOBE indeksu, trećina najpopularnijih programskih jezika pripadaju skript paradigmi
- Python je skript jezik opšte namene čija popularnost sve više raste

Nastanak skript jezika

- Prvi skript jezik je bio komandni jezik `sh` (Unix shell)
- On je započeo kao kolekcija komandi koje se interpretiraju kao pozivi sistemskih funkcija, npr upravljanje fajlovima i filtriranje fajlova
- Vremenom su na to dodate promenljive, kontrola toka, funkcije i razne druge mogućnosti, i na kraju je to rezultovalo kompletnim programskim jezikom
- U skladu sa time, skript jezici su bili korišćeni da se zapiše u fajlu lista komandi — **skript**, koja treba da se interpretira

Skript jezici

- Skript jezik je programski jezik koji služi za pisanje skriptova: **skript** je niz komandi koje se izvršavaju u odgovarajućem izvršnom okruženju (engl. *run-time environment*)
- Neke skriptove odlikuje da mogu biti izvršeni bez interakcije sa korisnikom (na primer, skriptovi koji se izvršavaju na serveru), dok za neke skriptove važi da se izvršavaju interaktivno u komunikaciji sa korisnikom (na primer, u okviru *REPL* okruženja (engl. *read-eval-print-loop*))
- Skript jezici se najčešće interpretiraju
- Upotreba modernih skript jezika se može lako i brzo savladati, čak i od strane početnika ili ljudi koji po struci nisu programeri. Sintaksa i upotreba nekih starijih skript jezika je veoma kompleksna i nije za početnike.

Skript jezici

- Skript jezici su u ekspanziji: jako puno događanja na polju razvoja programskih jezika u prethodne dve decenije je u okviru razvoja skript jezika
- Skript jezici imaju veliki broj domena primene, a posebna popularnost dolazi iz upotrebe u veb programiranju (PHP, JavaScript, Perl...).
- U prvobitnom obliku pojavljuju se kao komandni jezici operativnih sistema (npr. `bash`), danas imaju najrazličitije primene.
- Skript jezici mogu imati specifičan domen primene, ali mogu biti i jezici opšte namene (npr. Python).
- Pošto se često koriste za povezivanje komponenti, nazivaju se *glue languages*

Povezivanje aplikacija (engl. *glue languages*)

- Tradicionalni programski jezici (C/C++, Java) su namenjeni za razvoj samostalnih aplikacija koje imaju za cilj da prime neku vrstu ulaza i na osnovu nje generišu odgovarajući izlaz.
- Međutim, upotreba računara često zahteva manilpulaciju i koordinaciju različitih programa.
- Ručno sprovođenje tih poslova je naporno i sklono greškama tako da se u takvim situacijama najčešće koriste skript jezici.
- Koordinaciju drugim programima moguće je ostvariti i u tradicionalnim programskim jezicima, ali to nije uvek jednostavno.

Primeri koordinacije rada različitih programa

Primer 1 **Fotografija**. Fotograf treba da skine fotografije sa digitalnog fotoaparata, konvertuje u odgovarajući format, rotira slike po potrebi, napravi manje koji su pogodne za brzo razgledanje, indeksira ih po vremenu, temi, napravi bekap na udaljenoj arhivi i ponovo inicijalizuje memoriju...

Primer 2 **Sistem za obračun plata**. Obračunavanje vremena sa kartica, paprinih izveštaja i unosa sa tastature, manipulacija bazama podataka, poštovanje pravnih i institucionalnih regulativa, priprema poreza, doprinosa i medicinskog osiguranja, pravljenje papirnih evidencija za arhivu...

Primer 3 **Kreiranje dinamičkih veb stranica**. Autentikacija i autorizacija, komunikacija sa udaljenim uređajem, manipulacija sa slikama, komunikacija sa serverom, čitanje i pisanje HTMLa

Tradicionalni vs skript jezici

- Tradicionalni jezici (C/C++, Java) adresiraju
  - efikasnost izvršavanje,
  - lako održavanje,
  - portabilnost i
  - statičko otkrivanje grešaka.

  Sistem tipova je obično izgrađen oko **primitivnih koncepta** kao što su celobrojne vrednosti fiksnih veličina, brojevi u pokretnom zarezu, karakteri i nizovi.

- S druge strane, skript jezici imaju za cilj da adresiraju
  - fleksibilnost razvoja,
  - brz razvoj,
  - lokalnu prilagodiljivost i
  - dinamičke provere.

  Njihovi sistemi tipova, zbog toga teže da podrže **programske koncepte višeg nivoa**, kao što su tabele, katalozi, liste, datoteke...

Pregled

Karakteristike skript jezika

- Skript paradigma je često specifična kombinacija drugih paradigmi, kao što su:
  - imperativna paradigma (prisutnost promenljivih, naredbi, petlji, if, switch/case, funkcija/procedura),
  - objektno-orijentisana paradigma (prisutnost klasa, objekata, nasleđivanja),
  - funkcionalna paradigma (lambda funkcije, funkcije višeg reda)
  - kroz biblioteke, može biti prisutna podrška i drugim programskim paradigmama
- Nije uvek lako napraviti razliku između skript-jezika i drugih programskih jezika ali ipak imaju određene karakteristike na osnovu kojih se mogu izdvojiti
- Primeri skript jezika: Unix Shell (sh), Bash, PowerShell, JavaScript, TypeScript, PHP, Perl, Python, XSLT, Tcl, VBScript, Lua, Ruby...

Karakteristike skript jezika

- Interaktivno korišćenje/serijska obrada
- Skraćen zapis
- Deklaracije i pravila dosega
- Dinamičko tipiziranje
- Sistemske funkcije
- Manipulacija stringovima i poklapanje obrazaca
- Tipovi podataka visokog nivoa

Interaktivno korišćenje/serijska obrada

- Većina skript jezika omogućava interaktivno korišćenje (kroz REPL), ali ima i jezika koji su planirani za obradu bez komunikacije sa korisnikom (npr skiriptovi na serverskoj strani)
- Većina skirpt jezika je interpretatorskog tipa i omogućava obradu linije za linijom ulaza
  - To znači da je izvršavanje skripta moguće čak i kada postoji neka sintaksna greška u kodu: izvršavanje se onda izvodi sve do linije u kojoj se nalazi ta greška.
  - S druge strane, podsetimo se da kompilirani jezici zahtevaju da je kôd u potpunosti sintaksno ispravan da bi mogao da se prevede i zatim izvrši.
- Neki skript jezici zahtevaju da se sve komande učitaju pre nego što počne obrada, ali takvi su u manjini (npr. Perl)
  - Perl skriptovi se najpre kompiliraju do međukoda, a zatim se taj međukod interpretira
  - Ove dve faze se izvršavaju uvek zajedno, tj međukod se ne čuva u zasebnoj datoteci već se pravi svaki put iznova

Skraćen zapis

- Skraćen zapis zarad brzog razvoja i interaktivnog korišćenja
- Skraćen zapis se može ostvariti na različite načine, ali većina skript jezika izbegava korišćenje deklaracija i opštih delova koda koji su prisutni u tradicionalnim jezicima
- Primer **Perl**

```
print "Hello, world\n"
```

Java:

```
class Hello {
    public static void main(String[] args) {
        System.out.println("Hello, world!");
    }
}
```

Deklaracije i pravila dosega

- Promenljive se obično ne deklarišu
- Postoje jednostavna pravila dosega. Na primer:
  - U nekim jezicima, sve promenljive su globalne (npr. Perl, mada po potrebi se mogu ograničiti dosezi promenljivama),
  - U nekim jezicima, sve promenljive su lokalne (npr. PHP, Tcl, ali se po potrebi mogu eksplicitno napraviti globalne promenljive)
  - Python — svaka promenljiva je lokalna za blok u kojoj joj je dodeljena vrednost
- Pravila dosega za tradicionalne jezike su često veoma kompleksna i ne mogu se opisati u jednoj do dve rečenice (kao što je to za skript jezike). Na primer, pogledajte pravila dosega za C
  `https://www.w3schools.in/c-tutorial/variable-scope/`
  i C++
  `https://en.cppreference.com/w/cpp/language/scope`

Dinamičko određivanje tipova promenljivih

- Većina skript jezika dinamički određuje tipove podataka
- *If it looks like a duck, swims like a duck, and quacks like a duck, then it probably is a duck*
- U nekim jezicima, tip se proverava neposredno pred korišćenje (PHP, Ruby, Python)
- U nekim jezicima, tip će se interpretirati drugačije u različitim kontekstima (Perl, Tcl)

```
$a = "4"
print $a . 3 . "\n"  # '.' je nadovezivanje: 43
print $a + 3 . "\n"  # '+' je sabiranje: 7
```

Dinamičko određivanje tipova

- Dinamičko određivanje tipova je posebno važno kada se jezik koristi kao lepak za druge aplikacije
- U takvom scenariju skript mora da bude u mogućnosti da prihvati i prosledi podatke iz različitih sistema koji mogu biti napisani u različitim programskim jezicima a koji mogu čak imati i nekompatibilne sisteme tipova
- Skriptovi obrađuju različite vrste podataka, počevši od podataka iz formi, baza podataka, tabela (engl. *spreadsheets*) ili veb stranica. U tom smislu, jednostavan statički sistem tipova bi bio nefleksibilan, a napredni statički sistemi tipova koji uključuju parametarski polimorfizam ili klasne šablone bi zakomplikovao i usporio razvoj. Dinamičko određivanje tipova je zato pravo rešenje

Dinamičko određivanje tipova

- Dinamičko određivanje tipova omogućava da se preskoče deklaracije tipova, ali odsustvo informacije o tipovima čini skript jezike manje čitljivim
- Dinamičko određivanje tipova preskače potrebu za komplikovanim sistemom tipova, ali greške u radu sa tipovima mogu da se detektuju samo u fazi izvršavanja, a neke greške mogu da ostanu nedetektovane jako dugo ili zauvek (npr. ukoliko ne dovode do pada pri radu, već samo do blago izmenjenog ponašanja)
- Podsetimo se da statičko određivanje tipova omogućava kompajleru da utvrdi da se neke greške u tipovima nikada neće desiti u fazi izvršavanja, kao i da utvrdi da će se neke greške možda desiti

Dinamičko određivanje tipova

- Neki skriptovi se napišu da se koriste samo jednom, ali mnogi skriptovi se koriste puno puta.
- Ako skript treba da se koristi puno puta, veoma je važno da bude lak za održavanje, kao što su to programi pisani u tradicionalnim jezcima
- Da bi bio lak za održavanje, potrebno je da bude čitljiv, dobro dokumentvan, dizajniran tako da se može lako menjati i da sadrži što manje grešaka
- Dizajneri modernih skript jezika, kao što je to npr Python, razumeju važnost održavanja, ali i dalje nije lako pomiriti ciljeve koji su međusobno konfliktni: fleksibilnost u implementaciji i eliminisanje grešaka tipova u fazi izvršavanja

Sistemske funkcije

- U skript jezicma obično je omogućen lak pristup funkcionalnostima operativnog sistema
  - funkcije za ulaz/izlaz,
  - manipulacija fajlovima i direktorijumima,
  - upravljanje procesima,
  - pristup bazama podataka,
  - pristup soketima,
  - interprocesna komunikacija i sinhronizacija,
  - zaštita i autorizacija,
  - datum i sat,
  - komunikacija preko mreže
- I u drugim programskim jezicima je to moguće, ali ne na tako jednostavan način

Manipulacija stringovima i poklapanje obrazaca

- Skript jezici imaju svoje pretke u jezicima za procesiranje teksta i za generisanje izveštaja.
- Zbog toga imaju bogatu podršku za rad sa stringovima, za poklapanje obrazaca, pretragu i slično.
- Ovo se obično bazira na proširenim regularnim izrazima.
- Odrediti pravo ime funkciji osobina (tj naći koju osobinu broja n ispituje naredna funkcija u Python-u)

```
def osobina(n):
    return not re.match(r'^.?$|^(..+?)\1+$', '1'*n)
```

Tipovi podataka visokog nivoa

- Koriste se i osnovni tipovi podataka ali u okviru same sintakse i semantike skript jezika postoji i direktna podrška za više tipove podataka:
  - skupove,
  - multiskupove,
  - rečnike,
  - liste,
  - torke...
- U tradicionalnim jezicima potrebno je implementirati podršku za napredne tipove podataka i oni se ne mogu koristiti ravnopravno sa osnovnim tipovima podataka (osim u jeziku C++ koji omogućava predefinisanje osnovnih operatora)
- Skript jezici koriste sakupljač otpadaka

Pregled

Domeni upotrebe skript jezika

- Komandni jezici
- Procesiranje teksta
- Matematika i statistika
- Jezici proširenja
- Jezici za www
- Jezici opšte namene

Komandni jezici

- Komandni jezici (engl. *shell languages*): `sh`, `csh`, `ksh`, `bash`, PowerShell — omogućavaju komunikaciju sa operativnim sistemom i manipulaciju sa fajlovima, argumentima i komandama, kao i spajanje različitih aplikacija.
- Komande u shell jezicima generalno uzimaju formu niza reči, od kojih je prva ime komande koja treba da se izvrši. Većina komandi su programi koji se mogu naći u okviru direktorijuma koji se nalazi u putanji shell-a, ali postoji i veliki broj ugrađenih komandi koje shell prepoznaje i samostalno izvršava umesto pretrage za spoljnim programima. Neke ugrađene komande su duplirane, odnosno koriste se ugrađene iako postoje dustupni programu u okviru operativnog sistema
- Primeri komandi sa kojima ste se susretali, a mogu se koristiti u okviru unix-olikog shell-a: `cd`, `ls`, `cp`, `mv`, `mkdir`...

Komandni jezici

- Petlje (`for`), uslovi (`if`), funkcije — slično imperativnim jezicima
- Ekspanzija imena fajlova i varijabli (npr. `*.pdf` odgovara svim fajlovima sa ekstenzihom `.pdf`)
- Osim *wildcard* karaktera `*`, postoje i alternative, na primer:
  - `?` označava bilo koji karakter, npr `fig?.pdf` odgovara bilo kojoj datoteci kojoj ime počinje sa `fig`, zatim sledi bilo koji karakter, pa nakon toga `.pdf`
  - na primer, `fig[0-9].pdf` zahteva cifru nakon `fig` a pre `.pdf`,
  - na primer, `fig3.{eps,pdf}` se poklapa sa `fig3.eps` ili sa `fig3.pdf`

Komandni jezici

- Za sve fajlove u .eps formatu pokreni program konverzije eps-a u pdf

```
for fig in *.eps
do
   ps2pdf $fig
done
```

- Može se zapisati sve u jednom redu, upotrebom ; za razdvajanje

```
for fig in *.eps; do ps2pdf $fig; done
```

Komandni jezici

- Upotreba jednostrukih i dvostrukih navodnika

```
foo=bar
single='$foo'
double="$foo"
echo $single $double
```

Biće odštampano

```
$foo bar
```

- I jednostruki i dvostruki navodnici dozvoljavaju upotrebu belina u okviru niske
- Jednostruki navodnici neomogućavaju ekspanziju
- Dvostruki navodnici omogućavaju ekspanziju

Komandni jezici

- Cevi (|)
  `for fig in *; do echo ${fig%.*}; done | sort -u | wc -l`
  Štampanje svih fajlova sa uklonjenom ekstenzijom, njihovo sortiranje radi uklanjanja duplikata, i na kraju prebrojavanje linija
- Redirekcija (> i <)
  `for fig in *; do echo ${fig%.*}; done | sort -u > all_figs`
- `#!` *shebang* konvencija: ukoliko su prva dva bajta fajla `0x2321`, tada kernel pretpostavlja da je u pitanju skript i čita naredne karaktere kako bi ustanovio koji interpreter treba da koristi, npr. `#!/bin/bash`

Procesiranje teksta i generisanje izveštaja

- Jezici za procesiranje teksta i generisanje izveštaja (domenski specifični skript jezici) u čijoj se osnovi nalazi rad sa (proširenim) regularnim izrazima
- Prvi jezik **sed** (*stream editor*) a ubrzo zatim i jezik **awk**

Procesiranje teksta i generisanje izveštaja

- **sed** — dizajniran za primenu akcija skripta na svaku liniju željenog teksta (ili na preciziran opseg linija u tekstu) iz nekog fajla ili grupe fajlova.
- Ovo je mali jezik sa uskim domenom primene koji ima nečitljivu sintaksu i skriptovi nisu pogodni za ponovnu upotrebu.
- Podržava rad sa regularnim izrazima.
- `$ echo "Hello, sed" | sed 's/sed/world/'`
  `Hello, world`
  U prethodnom primeru, `sed` se pokreće iz terminala. Ulaz u `sed` je izlaz komande `echo`. `sed` nad niskom `Hello, sed` vrši naredbu `s` (substitucija, tj zamena) kojom se reč `sed` menja sa rečju `world`
- Vidimo da se `sed` koristi u kombinaciji sa šelom

Procesiranje teksta i generisanje izveštaja

- **awk** — ime po autorima: Alfred Aho, Peter Weinberger, Brian Kernighan
- Nastao je sa cljem olakšavanja formatiranja/generisanja izveštaja, kao i ispravljanja propusta koji su postojali u sed-u
- Osnovna ideja je program koji se sastoji od obrazaca i akcija koje se sprovode kada se obrasci poklope.
- Obrasci podržavaju rad sa proširenim regularnim izrazima, a akcije su slične jeziku C.

Procesiranje teksta i generisanje izveštaja

- Dva primera za Hello world

```
$ echo 'this line of data is ignored' > test
$ awk '{ print "Hello, world" }' test
Hello, world

$ awk 'BEGIN { print "Hello, world" }'
Hello, world
```

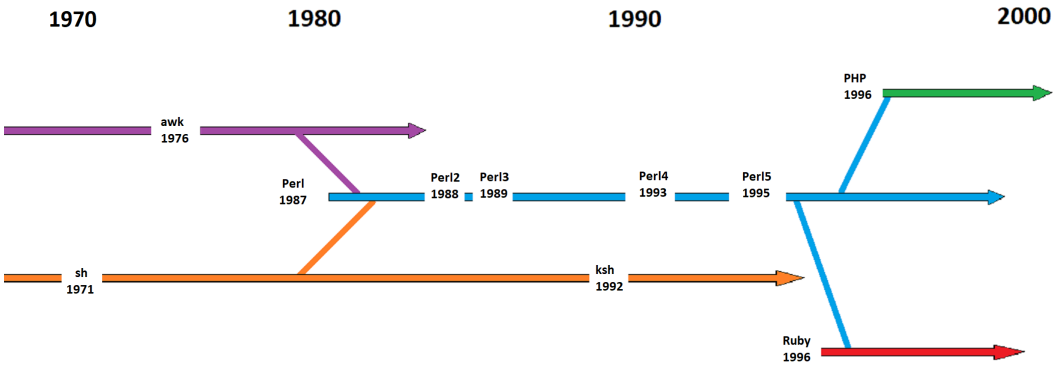
- U prvom primeru se najpre pravi datoteka test u koju se upisuje jedna linija teksta. Zatim se nad tom datotekom pokreće `awk`, sa komandom koja kaže da se za svaku liniju ulaza, uradi štampanje `Hello, world`
- U drugom primeru se koristi komanda BEGIN koja omogućava da se pre čitanja ulaza uradi nešto (tj štampanje `Hello, world`)

Perl logo https://www.perl.org/

Procesiranje teksta i generisanje izveštaja

- Perl je nastao sa idejom da kombinuje sed, awk i sh, ali je izrastao u mnogo više od toga, i pritom je uticao na razvoj modernih skript jezika (PHP, Ruby, Python, JavaScript...)
- Perl je u svojoj prvoj verziji isporučivao i programe a2p i s2p za konvertovanje awk skriptova i sed skriptova u Perl, sa ciljem da se u potpunosti potisnu awk i sed
- Perl je jedan od najranijih skript jezika nove generacije, sa podrškom za proširene regularne izraze, sa pristupom sistemskim pozivima, podržava klase i rad sa objektima, funkcionalne koncepte i omogućava razne vrste proširivanja

Perl: deo razvojnog stabla https://www.perl.org/



Perl

- Perl je nastao 1987. godine, dizajnirao ga je Leri Vol (eng. *Larry Wall*).
- Poslednja verzija 5.32.1 je izdata 23. januara 2021
- Ime Perl je sastavljeno kao pearl (na engleskom *biser*) ali bez slova a
- Iako nastao kao jezik za procesiranje teksta, Perl danas spada u jezike opšte namene, sa veoma širokim mogućnostima primene, sa posebnim primenama na vebu

Hello, world

- Perl skriptovi imaju ekstenziju `.pl`
- Neka je sadržaj datoteke `hello.pl` naredna linija

```
print "Hello, world\n"
```

- Perl se pokreće iz komandne linije naredbom `perl`.

```
$ perl hello.pl
Hello, world
```

Promenljive u Perl-u

- Interesantno je pomenuti promenljive i tipove u Perlu
- Promenljive u Perlu su statički tipizirane (što ga značajno razlikuje od drugih skript jezika!) i implicitno deklarisane, slabo tipiziran jezik
- Postoje tri različita prostora imena za promenljive, koja se označavaju prvim krakterom imena promenljive
  - Skalarne promenljive (koje uključuju i stringove i brojeve) počinju sa znakom dolar $
  - Nizovne promenljive počinju sa znakom @
  - Asocijativni nizovi, koji se nazivaju heševi (strukture podataka koje se indeksiraju stringovima i implicitno se kontrolišu heš tabelama) počinju sa znakom %
- Ova konvencija čini imena varijabli čitljivijim u odnosu na druge skript jezike

Raku logo https://www.raku.org/

Perl se može shvatiti i kao ime za familiju jezika. Odnosi se na Perl 5, ali se od 2000. do 2019. odnosilo i na Perl 6, koji se paralelno razvijao i koji je zatim preimenovan u jezik Raku https://www.raku.org/

```
print "Hello, world\n"

say "Hello, world"
```

Raku skriptovi mogu da imaju različite ekstenzije, npr .raku ili .p6

Procesiranje teksta i generisanje izveštaja

- Moderni skript jezici nasleđuju razne ideje iz jezika sed, awk i Perl, pre svega u implementaciji i sintaksi regularnih izraza. Regularni izrazi su duboko integrisani u skript jezike i obuhvataju specijalnu sintaksu i ugrađene operatore (sed, awk, Perl, PHP, Ruby, JavaScript...)
- Većina jezika koji imaju podršku za rad sa regularnim izrazima se mogu svrstati u dve glavne grupe
  (i) Prva grupa uključuje awk, grep, regex biblioteku za C i starije verzije jezika Tcl. Oni implementiraju regularne izraze kao što je to definisano POSIX standardom
  (ii) Jezici u drugoj grupi prate Perl, koji obezbeđuje širok skup ekstenzija (napredni/prošireni regularni izrazi). U ovu grupu spadaju PHP, Python, Ruby, JavaScript, Emacs Lisp, Java, C#, i novije verzije Tcl

Matematika i statistika

- Ovi jezici pripadaju i grupi domenski specifičnih jezika
- Matematika
  - Moderni naslednici jezika APL[1]: Maple, Wolfram Mathematica, Mathlab
  - Podrška numeričkim metodama, simboličkoj matematici, vizuelizaciji podataka, matematičkom modelovanju. Primene u najrazličitijim oblastima, pre svega za rešavanje konkretnih problema i za izradu simulacija i prototipova.
- `Hello, world` program u Matlabu:

  ```
  disp('Hello world')
  ```

  Matlab skriptovi imaju ekstenziju `.m`

[1] Jezik nastao '60tih godina prošlog veka, orginalno osmišljen kao notacija za podučavanje primenjene matematike, kao programski jezik zadržao je akcenat na koncinznom i elegantnom izražavanju matematičkih algoritama. Karakteriše ga veoma nebična i nečitljiva sintaksa.

Matematika i statistika

- Ovi jezici pripadaju i grupi domenski specifičnih jezika
- Statistika
  - Jezici S i R (otvorenog koda)
  - Za statističke obrade podataka u svim oblastima: podrška višedimenzionalnim nizovima, listama, mogu se proširivati infiksnim operatorima, funkcionalno programiranje
- `Hello, world` program u jeziku R (komande u okviru interpretera)

```
> print("Hello, world")
[1] "Hello, world"
> # Quotes can be suppressed in the output
> print("Hello, world", quote = FALSE)
[1] Hello, world
```

Skriptovi se mogu pisati u zasebnim datotekama, obično sa ekstenzijom `.r`

Jezici proširenja

- Većina aplikacija prihvata neke vrste komandi koje im govore šta treba da urade
- Nekada se te komande zadaju tekstualno, ali češće se pokreću putem događaja koji se iniciraju korišćenjem korisničkog interfejsa na primer putem klika mišem na odgovarajući izbor u meniju
  - Na primer, komande programa za crtanje mogu da budu da se sačuva ili učita crtež, da se izabere, umetne, obriše ili izmeni neki deo crteža, da se izabere drugačija stil linije, debljina ili boja, da se zumira ili rotira prikaz...
- Jezici proširenja proširuju korisnost neke aplikacije dozvoljavajući korisniku da dodaje nove komande koristeći postojeće komande kao gradivne blokove
  - Na primer, ukoliko je za seriju crteža potrebno da se učita, umetne izabrani logo i da se sačuva crtež sa novim imenom — jezikom proširanja mogu se spojiti i automatizovati ove operacije kako se ne bi morale da izvode ručno

Jezici proširenja

- Da bi se omogućile izmene putem jezika proširenja, alat mora da
  - Sadrži ili da komunicira sa interpreterom nekog skript jezika.
  - Obezedi način da skript može da poziva postojeće komande alata
  - Dozvoli korisniku da poveže novodefinisanu komandu sa događajima korisničkog interfejsa

Jezici proširenja — primeri

- Adobe grafički skup - Illustrator, Photoshop... dopuštaju dopune različitim skript jezicima: JavaScript, Visual Basic ili AppleScript.
- AutoCAD i Flash imaju svoje skript jezike za proširenje.
- Skript jezik Lua se često koristi u razvoju skriptova za igrice.
- Majkrosoftovi alati obično koriste Visual Basic.
- GIMP (GNU Image Manipulation Program), može da korsiti različite skript jezike, npr Scheme, Tcl, Python i Perl, ali se najčešće koristi Scheme.

Jezici za www

- Najznačajnija primena skript jezika je u programiranju veb aplikacija
- Skritpovi na strani klijenta koji se pokreću u okviru veb razgledača — JavaScript
- Skriptovi na strani servera koji omogućavaju implementaciju dinamičkih veb-stana i veb aplikacija — PHP, Perl, Ruby, Python, JavaScript

JavaScript https://www.javascript.com/

- Upotreba veba je eksplodirala sredinom 90tih godina sa pojavom prvog grafičkog brauzera
- JavaScript je orginalno razvijen u kompaniji Netscape, dizajner jezika je bio Brendan Eich. Originalno ime je bilo Mocha, da bi kasnije bio preimenovan u LiveScript
- Krajem 1995, LiveScript je postao zajednički projekat Netscape-a i Sun Microsystems i zbog toga je ime promenjeno u JavaScript
- Poslednja stabilna verzija: jun 2020.

JavaScript https://www.javascript.com/

- Sem imena, JavaScript nema mnogo toga zajedničkog sa jezikom Java
- JavaScript se intenzivno razvijao i doživeo puno promena i dopuna u odnosu na originalni jezik
- Na razvoj JavaScript-a uticali su brojni programski jezici, uključujući awk, C, HyperTalk, Java, Lua, Perl, Python, Scheme i Self

JavaScript

- Moderan JavaScript je zajedno sa HTMLom i CSSom osnovna tehnologija za razvoj veb aplikacija
- Po nekim merenjima, JavaScript je najpopularniji i najkorišćeniji programski jezik već duži niz godina:
  `https://redmonk.com/sogrady/2020/02/28/language-rankings-1-20/`
  `https://www.benfrederickson.com/ranking-programming-languages-by-github-users/`
- JavaScript omogućava interaktivne veb stranice i većina veb stranica ga koristi za definisanje ponašanja na strani klijenta, a svi najbitniji veb razgledači implementiraju odgovarajuće JavaScript okruženje za izvršavanje JavaScript skriptova
- JavaScript se može koristiti i za razvoj skriptova na strani servera

JavaScript

- JavaScript je multiparadigmatski jezik, podržava programiranje vođeno događajima, funkcionalno i imperativno programiranje
- Interpretiran, slabo tipiziran jezik
- Razvoj JavaScript aplikacija podržan je od strane različitih razvojnih okvira. Neka od najpopularnijih razvojnih okvira su (za *front-end*):
  - Angular
  - React
  - Vue
  - Ember
- Postoje, takođe i razvojni okviri za razvoj skriptova na strani servera, npr Express JS, Next JS, Gatsby JS, Nuxt JS, Node JS

Hello, world

JavaScript komanda `alert` (u okviru `html` dokumenta):

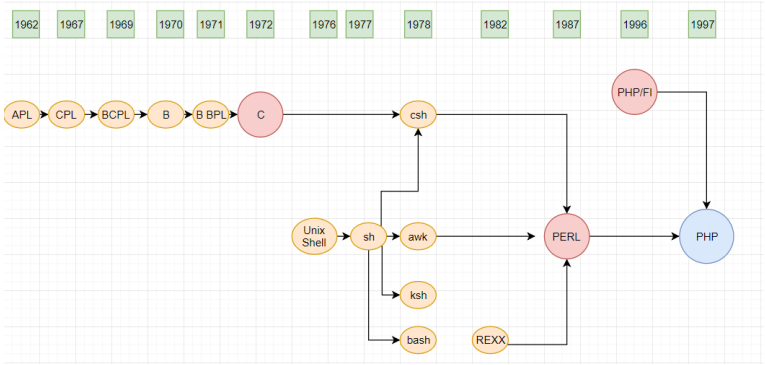
```
<!DOCTYPE HTML>
<html>
<body>
  <p>Before the script...</p>
  <script>
    alert( 'Hello, world!' );
  </script>
  <p>...After the script.</p>
</body>
</html>
```

Može se koristiti i `document.write('Hello, World!');`
JavaScript skriptovi ne moraju da budu u okviru html dokumenata, već mogu da budu u zasebnim fajlovima sa ekstenzijom `.js`

PHP — Hypertext preprocessor

- Originalno, razvoj jezika je započeo 1994. godine, dizajner jezika: Rasmus Lerdorf
- Jezik je napravljen za potrebe razvoja privatnog vebsajta: za praćenje broja poseta. "PHP Tools" — 1995. niz programa u C-u koji su nazvani "Personal Home Page Tools"
- 1997. jezik je kompletno redizajniran i preimenovan — Hypertext preprocessor
- Poslednja stabilna verzija 8.0.1, 7. januar 2021.
- Po nekim statistikama, više od 75% servera koriste PHP
  `https://w3techs.com/technologies/details/pl-php`

PHP: deo razvojnog stabla https://www.php.net/



PHP

- PHP je jezik koji se koristi za razvoj skriptova na strani servera
- Skript jezik, uključuje svojstva imperativne, funkcionalne i objektno-orijentisane paradigma
- Interpretiran, dinamički, slabo tipiziran jezik
- Za razvoj PHP skriptova značajni su razvojni okviri
  - Laravel
  - Codelgniter
  - Symfony
  - CakePHP
  - Yii
  - Zend

PHP Hello, world https://www.w3schools.com/php/

Datoteka hello.php na serveru, levo. Kada joj se pristupi, ukoliko je sve u redu konfigurisano, generiše se izlaz, desno

```
<html>
 <head>
  <title>PHP Test</title>
 </head>
 <body>
 <?php echo '<p>Hello World</p>'; ?>
 </body>
</html>
```

```
<html>
 <head>
  <title>PHP Test</title>
 </head>
 <body>
 <p>Hello World</p>
 </body>
</html>
```

Pregled

Skript jezici opšte namene

- Skript jezici opšte namene koriste se u različitim domenima primene
- U njih spadaju
  - Perl,
  - Python,
  - Ruby,
  - Lua,
  - ...

Python https://www.python.org/

- Python je skript jezik otovorenog koda koji ima veliki broj primena. Primene uključuju
  - veb programiranje (skriptovi na strani servera),
  - nauku o podacima,
  - veštačku inteligenciju,
  - naučne aplikacije...
- Python se može korisiti za brzu izradu prototipa, ali i za razvoj softvera koji se koristi u produkciji
- Korišćen je i intenzivno se koristi u industriji
- Po TIOBE indeksu za 2020. godinu, rangiran je treći po popularnosti, odmah nakon jezika C i Java

Python

- Python omogućava programerima da se fokusiraju na rešavanje problema umesto da se fokusiraju na sintaksu jezika
- Python ima jednostavnu sintaksu koja se lako uči i predstavlja jezik koji se preporučuje kao prvi jezik za učenje programiranja (već od osnovne škole)
- Python prva verzija: 1991. godine, Guido van Rossum
- Python trenutna verzija 3.9.1, decembar 2020, verzija 2.7 je zvanično više nema podršku

Python

- Python uspešno integriše ideje iz različitih jezika, kao što su Perl, Haskell i OO jezici
- Python pripada skript paradigmi, ali ima i imperativne, OO i funkcionalne karakteristike
- Python je kompaktan jezik koji se oslanja na bogate biblioteke koje obezbeđuju najveći deo njegovih funkcionalnosti
- Na primer, regularni izrazi u Pythonu su prisutni putem modula RE, tj nisu ugrađeni u sam jezik
- Bogat izbor biblioteka utiče na popularnost i primenjivost jezika u različitim domenima
- Velika dostupna podrška za učenje i rešavanje najrazličitijih problema preko veba

Python Hello, world

- Python skriptovi imaju ekstenziju `.py`
- Na primer, neka je `hello.py`

  ```
  print("Hello, world")
  ```
- Tada se sa

  ```
  python hello.py
  ```

  na izlazu dobija

  ```
  Hello, world
  ```

Osobine Python-a

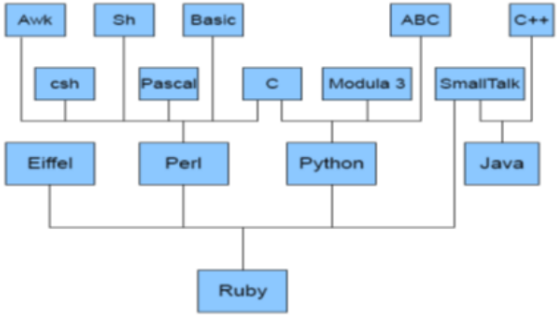
- Lak za učenje, čitanje i pisanje
- Izražajan
- Slobodan i otvorenog koda
- Jezik visokog nivoa
- Portabilan
- Objektno orijentisan
- Proširiv
- Ugradiv
- Interpretiran
- Velika standardna biblioteka
- Podrška za programiranje GUI-a
- Dinamički tipiziran, strogo tipiziran

Ruby https://www.ruby-lang.org/

Ruby

- Ruby je jezik opšte namene
- Ruby je dinamički tipiziran jezik, otovorenog koda, sa fokusom na jednostavnost i produktivnost. Ima elegantnu sintaksu, prirodnu za čitanje i pisanje
- Ruby se pojavio 1995. godine, Yukihiro "Matz" Matsumoto (Japan)
- Verzija 3.0 objavljena je 25. decembra 2020.

Ruby https://www.ruby-lang.org/



Ruby

- Na razvoj jezika Ruby uticao je veliki broj programskih jezika, a najviše su uticali Eiffel, Perl, Python, Smalltalk, Ada, Basic i Lisp
- Ruby je skript jezik koji spaja objektno orijentisano (Eiffel, Smalltalk), funkcionalno (Lisp) i imperativno (Perl, Python, Ada, Basic) programiranje
- Ruby je dinamički, jako tipiziran, interpretiran jezik
- Osnovna ideja dizajna jezika je da se adresiraju ljudske potrebe (a ne potrebe računara), odnosno da programiranje učini programere zadovoljnim, produktivnim i srećnim

Ruby

- Ekstenzija Ruby skriptova je `.rb`
- `hello.rb`

  ```
  puts "Hello, world"
  ```
- Pokretanje skripta

  ```
  ruby hello.rb
  ```
- Izlaz

  ```
  Hello, world
  ```

Ruby vs Ruby on rails

Ruby je svoju popularnost stekao sa razvojnim okvirom *Ruby on Rails*

- Ruby je programski jezik
- Ruby je jezik opšte namene, može se koristiti za razvoj desktop aplikacija
- Ruby je implementiran u programskom jeziku C
- Ruby je inspirisan najviše jezicima Perl i Smalltalk

- Ruby on rails je razvojni okvir
- Ruby on rails je okvir za razvoj veb aplikacija zasnovanih na bazama podataka
- Ruby on rails je implementiran u programskom jeziku Ruby
- Ruby on rails je insiprisan okvirima Django (Python) i Laravel (PHP)

Lua http://www.lua.org/

Lua http://www.lua.org/

- Lua je nastala 1993. godine, u Brazilu, Roberto Ierusalimschy, Waldemar Celes, Luiz Henrique de Figueiredo
- Poslednja stablina verzija: decembar 2020.
- Lua znači *mesec* na portugalskom
- Lua podržava proceduralno, OO i funkcionalno programiranje, kao i programiranje vođeno podacima
- Jezici koji su uticali na njen razvoj: C++, CLU, Modula, Scheme, SNOBOL
- Lua je dinamički tipiziran jezik, bajtkod se interpretira na virtuelnoj mašini koja je napisana u programskom jeziku C, ima automatsko upravljanje memorijom (inkrementalni sakupljač otpadaka)
- Lua se koristi za pisanje skriptova, konfiguracija i brzo pravljenje prototipova

Lua

- Lua je robustan jezik koji se koristi u industrijskim aplikacijama, a svoju popularnost je stekla najviše u industriji igara (npr, World of Warcraft i Angry Birds)
- Programi pisani u Lui imaju dobre performanse. Lua se smatra najbržim skript jezikom
- Lua je portabilna i može se koristiti svuda gde postoji kompajler za standardni C
- Lua može lako da se ugrađuje u druge programske jezike, posebno u C/C++, ali možei u Javu, C#, Fortran, Smalltalk, Ada, Erlang kao i druge skript jezike, npr Perl i Ruby
- Lua je mali i jednostavan jezik, ali sa velikim mogućnostima
- Lua je besplatan softver otvorenog koda

Lua

- Ekstenzija Lua skriptova je `.lua`
- `hello.lua`

  ```
  print ("Hello, world")
  ```
- Pokretanje skripta

  ```
  lua hello.lua
  ```
- Izlaz

  ```
  Hello, world
  ```

Pregled

Literatura

- Concepts of programming languages, Robert W. Sebesta
- Programming language pragmatics, Michael L. Scott, fourth edition, Elsevier.
- Programming language design concepts, David A. Watt, William Findlay.
- Veb stranice jezika (koje su navođene uz svaki jezik)