Skriptni programski jezici

## Hrvoje Čičak

Što su skriptni jezici?

Skriptni programski jezici spadaju u kategoriju alata koji omogućuju automatizaciju procesa na računalu korištenjem skripti. Najčešće je riječ o interpretiranim jezicima, odnosno skripte prije pokretanja najčešće nije potrebno kompajlirati u nekakav oblik binarnog koda već se izvršavaju izravno iz tekstualne datoteke.

Izraz „glue language“ veže se uz skriptne jezike zato što oni omogućuju povezivanje gotovih dijelova koda (poznati i kao cmdledts) u svrhu kreiranja novih „programa“. Pojednostavljeno, korištenje skriptnih programa na ovom nivou je poput igranja LEGO kockicama kada iz istog seta kockica možemo napraviti različite oblike. Tako i korištenjem gotovih naredbi u skriptama možemo dobiti različite rezultate.

Prema namjeni prepoznajemo skriptne jezike opće namjene, primjeri kojih su Python, Lua ili JavaScript, zatim skriptne jezike vezane za određeno područje djelovanja “domenu” poput PowerShell-a, Bash-a, ili Rexx-a.

Specifične, usko specijalizirane skriptne jezike nalazimo u periferiji - poput PostScript-a npr. Automatizacija je u ovom slučaju otišla toliko daleko da program (npr CAD, vektorski editor ili bitmap editor) sam kreira skriptu koja se prenosi pisaču koji podržava postscript te je on interpretira u stranicu ispisa. U pozadini i dalje imamo skriptu pisanu u skriptnom jeziku i potpuno čitljivu ljudima, ali samo generiranje skripte više ne radi čovjek, već program.

Još jedno područje u kojem je uporaba skriptnih jezika učestala su i video igre. Većina AAA naslova podržava barem jedan jezik kao skriptni. Time se igri povećava playability budući da igraća zajednica razvija modove i dodatke za igru. Blizzard recimo preferira Lua programski jezik u svojim igrama pa je npr. za WoW prije bilo moguće napisati skripte u Lui koje bi se ponašale kao pravi igrač i grindale iskustvo ili resurse. Naravno da takvo korištenje skripti nije niti etično, niti Blizzard blagonaklono gleda na to te su zbog toga banali ljude sa servera. Međutim kojekakvi add-onovi za WoW koji su perfektno legalni pisani su baš u Lui. Koliko je fina granica između skriptnih jezika i čistih programskih jezika uočljivo je baš u videoigrama. Unity kao glavni skriptni jezik koristi C#, a Unreal C++. S druge strane CCP-ov MMO EVE Online pisan je u stackless Pythonu.

Kratka povijest skriptnih jezika

Prvi skriptni jezici pojavili su se otprilike u isto vrijeme kad su računala postala dovoljno napredna da mogu obavljati više zadaća. Budući da tadašnja računala nisu imala terminala i programirana su putem bušenih kartica, u IBM-u je razvijen Job Control Language (JCL) koji je služio automatizaciji batch procesiranja i unošenja podataka putem kartica. Sam JCL je teško čitljiv i kompleksan zbog kompleksnosti mainframe-a za koji je razvijan (OS/360).

U šezdesetim godinama u masovniju uporabu dolaze ekranski terminali za komunikaciju sa mainframe računalima i razvijaju se prvi CLI interpreteri (Command Line Interface). Taj način komunikacije s računalom ostaje primaran način interakcije sve do popularizacije GUI rješenja (Windows, MacOS). Čak i danas kad spominjemo skriptne jezike prva asocijacija su upravo CLI i komandni redak.

U sedamdesetima jedan OS razvijen u Bell Labsu za potrebe AT&T postao je de facto standardni OS za mainframe računala. Riječ je o UNIX-u koji je kao svoj CLI uveo shell (sh). Zahvaljujući UNIX filozofiji da je sve van kernela dostupno za zamjenu / poboljšanje shell je tijekom generacija doživo nekoliko modifikacija (korn shell, bourne shell).

Boom IBM PC-a u osamdesetima donosi računala u kuće, s njima je u široku uporabu ušao i PC DOS (nekima poznat i kao MS DOS) – Radi se o proširenoj varijanti CP/M OS-a. Naravno i tu je prisutan CLI u obliku command.com shell-a. Meni osobno DOS je bio prvi susret sa skriptama i skriptnim jezicima prvenstveno proučavajući autoexec.bat datoteku. Za one koji ne znaju (ako ima takvih) autoexec.bat je prva skripta koju command.com pokrene prilikom starta računala. Iako command nije podržavao neke naprednije stvari poput user inputa (set /p se pojavio tek sa Windowsima XP) na netu se još mogu pronaći impresivni primjeri batch skripti.

Jačanjem hardvera u devedesetima i pojavom sve kvalitetnijih programskih alata pojavila se potreba za skriptnim jezicima opće namjene koji bi se mogli izvršavati neovisno o sistemu na kojem se nalaze. Uočite da iza iste filozofije stoji i programski jezik java – write once execute on multiple platforms. Najpoznatiji primjeri ovih jezika su Python, Tcl i Lua – svi nastali krajem osamdesetih ili početkom devedesetih. S njima su skriptni jezici napravili zaokret prema programskim jezicima opće namjene. Istodobno javlja se i obrnuti slučaj – javljaju se programi koji kao skriptne jezike koriste C, Object Pascal i Javu.

Glavna pokretačka snaga IT industrije 90-ih i 2000-ih bio je web, i tamo se ukazala potreba za skriptnim jezikom opće namjene koji bi se mogao izvršavati u browseru. U jeku browserskih ratova sredinom devedesetih tvrtka Netscape je u suradnji sa Sun Microsystems radila na uvrštanju java programskog jezika kao skriptnog, međutim projekt nije u potpunosti zaživio, već smo dobili jezik koji je slijedio filozofiju Perla ili Tcl-a, a koristio sintaksu Jave (i C-a). Standardizacijom u 2000-ima jezik mjenja ime iz kolokvijalnog JavaScript koji je Netscape koristio 90-ih sa ciljem da otme Microsoftu dio tržišnog udjela, jer Java je tad bila vrlo in, u ECMAScript. Međutim jezik je i dalje poznatiji pod imenom JavaScript.

Sve većim korištenjem OOP-a kao temeljne filozofije dolazimo do situacije kada i operativni sistemi, prvenstveno Windows, ali i ostali, postaju sve više OOP orijentirani – do te mjere da kao ulaz ili izlaz sistemskog poziva dobivamo objekte. Klasični CLI u tom slučaju nisu baš od neke koristi jer većina njihovih komandi kao izlaz daje tekst. Microsoft je godinama imao problem što njegov CLI na NT sistemima (cmd.exe) ne može direktno pristupiti svim dijelovima OS-a, baš zbog strukturalnog uređenja Windowsa npr. registry ili Active Directory. Kao rješenje za taj problem napravili su PowerShell – projekt Monad, i kad su vidjeli da su napravili dobar posao napravili su ga open source i cross platformskim. PowerShell se može izvršavati svugdje gdje je moguće instalirati .NET Core framework jer ovisi o njemu. Trenutno to znači Windows, MacOS i x86-64 distribucije Linuxa (Ubuntu i CentOS).

Bash

Bash je nastao relativno kasno, krajem osamdesetih, kao zamjena za Bourne shell – komercijalni shell koji se isporučivao sa UNIX system 7. Budući da se standardno isporučuje sa većinom linux distribucija sam shell je stekao ogromnu popularnost i prihvaćen je i na ostalim UNIX like sistemima. Bash podržava:

- pipeline

- controlne operatore (&& i ||)

- ekspanzivne stringove (brace expansion, "$expression" evaluation)

- wildcards

- cjelobrojnu aritmetiku...

#!/bin/bash

# Autor: Hrvoje Čičak - zadatak se nalazi na https://rosettacode.org/wiki/Chinese\_zodiac

# nastala na temelju PowerShell skripe s te stranice

# deklaracije string array-a korištenje obiju vrsti navodnika

declare -a pinyin=('甲' '乙' '丙' '丁' '戊' '己' '庚' '辛' '壬' '癸' '子' '丑' '寅' '卯' '辰' '巳' '午' '未' '申' '酉' '戌' '亥')

declare -a pinyin\_ascii=("jiă" "yĭ" "bĭng" "dīng" "wù" "jĭ" "gēng" "xīn" "rén" "gŭi" "zĭ" "chŏu" "yín" "măo" "chén" "sì" "wŭ" "wèi" "shēn" "yŏu" "xū" "hài")

# varijabla inicijalizirana na vrijednost "4"

base=4

# obične varijable, inicijalizirane na 0

celestial=0

terrestrial=0

# još jedan način deklariranja array-a - ovaj put bez declare

animals=('Štakora' 'Vola' 'Tigra' 'Zeca' 'Zmaja' 'Zmije' 'Konja' 'Koze' 'Majmuna' 'Pijetla' 'Psa' 'Svinje')

elements=('Drvo' 'Vatra' 'Zemlja' 'Metal' 'Voda')

aspects=('yang' 'yin')

# command line arguments - if (args == 0) then... obratiti pažnju na whitespace; $# - broj argumenata; $@ - vrijednosti argumenata

if [ $# -eq 0 ]; then

# postavljamo 0 element niza na vrijednost tekuće godine

asked\_year[0]=`date +%Y`

else

# kopriamo argumente u niz, usput radimo i ekspanziju - zato navodnici

asked\_year=( "$@" )

fi # end if u skripti je "fi" logično, zar ne?

# for petlja sintaksa slična c-u "(( ))" označavaju aritmetičke operacije

for ((i=0; i<${#asked\_year[@]}; i++)); do

cycle\_year=$(( asked\_year[i] - base ))

stem\_number=$(( cycle\_year % 10 ))

celestial=$stem\_number

terrestrial=$(( branch\_number + 10 ))

# trik - nema floating point aritmetike već cijelobrojno djeljenje, ponaša se kao floor funkcija (2/3=0)

element\_number=$(( stem\_number / 2 ))

branch\_number=$(( cycle\_year % 12 ))

aspect\_number=$(( cycle\_year % 2 ))

index=$(( cycle\_year % 60 + 1 ))

echo "${asked\_year[i]} je godina ${animals[branch\_number]}."

echo "Element je: ${elements[element\_number]}."

echo "Aspekt je: ${aspects[aspect\_number]}."

echo "To je $index godina u ciklusu."

echo "Kineska oznaka za to je: ${pinyin[celestial]}${pinyin[terrestrial]} odnosno: ${pinyin\_ascii[celestial]}-${pinyin\_ascii[terrestrial]}."

echo "------------------------------------------"

done

PowerShell

Svoj život počeo je 2002 kao projekt Monad, Microsoftova zamjena za CMD koji se sve teže nosio sa taskovima unutar windowsa. Zbog inovativnog pristupa – rezultat svake interne naredbe je objekt shell je omogućio korištenje windows resursa koji inače nisu bili mogući iz CMD.exe bez uporabe 3rd party programa. Budući da je PowerShell u uskoj sprezi sa .NET frameworkom moguće je pozivanje metoda iz .NET objekata.

# Autor: Hrvoje Čičak - zadatak se nalazi na https://rosettacode.org/wiki/Chinese\_zodiac

# nastala na temelju PowerShell skripe s te stranice

# primjer eksplicitnie sintakse za definiciju niza "@("- sa ekspandirajućim navodnicima (") i literalnima (')

$pinyin = @('甲','乙','丙','丁','戊','己','庚','辛','壬','癸','子','丑','寅','卯','辰','巳','午','未','申','酉','戌','亥')

$pinyin\_ascii = @("jiă","yĭ","bĭng","dīng","wù","jĭ","gēng","xīn","rén","gŭi","zĭ","chŏu","yín","măo","chén","sì","wŭ","wèi","shēn","yŏu","xū","hài")

# varijabla inicijalizirana na vrijednost "4"

$base = 4

# obične varijable, inicijalizirane na 0

$celestial = 0

$terrestrial = 0

# još jedan način deklariranja array-a - ovaj put samo nabrajanjem elemenata

$animals = 'Štakora', 'Vola', 'Tigra', 'Zeca', 'Zmaja', 'Zmije', 'Konja', 'Koze', 'Majmuna', 'Pijetla', 'Psa', 'Svinje'

$elements= 'Drvo', 'Vatra', 'Zemlja', 'Metal', 'Voda'

$aspects = 'yang', 'yin'

# command line arguments - if (args == 0) then... obratiti pažnju na whitespace; $# - broj argumenata; $@ - vrijednosti argumenata

if ( $args.Count -eq 0 ) {

# postavljamo vrijednost tekuće godine

$asked\_year = (Get-Date).Year # i get-date -UFormat %Y radi

}

else {

# kopriamo argumente u niz, usput radimo i ekspanziju - zato navodnici

$asked\_year = $args

}

# foreach iteracija kroz array

foreach ($year in $asked\_year) {

$cycle\_year = $year - $base

$stem\_number = $cycle\_year % 10

$celestial = $stem\_number

$terrestrial =$branch\_number + 10

# trik - koristimo .NET math lib za floor funkciju (2/3=0)

$element\_number = [math]::Floor($stem\_number / 2 )

$branch\_number = $cycle\_year % 12

$aspect\_number= $cycle\_year % 2

$index= $cycle\_year % 60 + 1

echo "$year je godina $($animals[$branch\_number])."

echo "Element je: $($elements[$element\_number])."

echo "Aspekt je: $($aspects[$aspect\_number])."

echo "To je $index godina u ciklusu."

echo "Kineska oznaka za to je: $($pinyin[$celestial])$($pinyin[$terrestrial]) odnosno: $($pinyin\_ascii[$celestial])-$($pinyin\_ascii[$terrestrial])."

echo "------------------------------------------"

}