

Razvoj programskog jezika Haskell

Seminarski rad u okviru kursa
Dizajn programskih jezika
Matematički fakultet

Marijana Miljić
marijanamiljic17311@gmail.com

19. decembar 2019

Sažetak

U ovom radu prikazuju se osnove razvoja programskog jezika Haskell. Ukratko su opisani jezici koji su najviše uticali na njegov razvoj: Lisp, ML, Miranda i Scheme. Za svaki jezik date su osnovne informacije kao i način na koji je taj jezik uticao na osobine i razvoj jezika Haskell. Prikazano je razvojno stablo koje uključuje ove jezike.

Sadržaj

1	Uvod	2
2	Osnovno o Haskellu	2
3	Razvojno stablo	2
3.1	Miranda	3
3.2	Lisp	4
3.3	ML	5
3.4	Scheme	5
3.5	Scala	5
4	Zaključak	6
	Literatura	6

1 Uvod

Postoji na hiljade programskih jezika, ali nisu svi jednako važni i zastupljeni [3]. Stalno nastaju novi programski jezici, a glavni razlog je poboljšanje određenih koncepata koji postoje u nastalim jezicima. U zavisnosti od toga šta treba isprogramirati, važno je izabrati što efikasniji programski jezik. Važno mesto zauzimaju jezici koji pripadaju funkcionalnoj paradigmi. Osnovni cilj ove paradigme je da oponaša matematičke funkcije. Nastaje 59.godine prošlog veka i najistaknutiji predstavnik je Lisp. Osnovna karakteristika čistih funkcionalnih programskih jezika je transparentnost referenci što kao posledicu ima nepostojanje bočnih efekta. Programski jezik Haskell je jedan od predstavnika funkcionalne paradigme. Za njega se može reći da je standardizovani, tipizirani jezik opšte namene. Logo jezika Haskell prikazan je na slici 1.



Slika 1: Logo programskog jezika Haskell

2 Osnovno o Haskellu

Haskell je standardizovani, funkcionalan, strogo tipizirani programski jezik nastao 1990. godine. Njegovim tvorcem se smatra Haskell Brooks Curry, čiji je rad u oblasti matematičke logike poslužio kao osnova za funkcionalne jezike. Haskell karakterišu sistem tipova sa statičkim tipiziranjem i lenjo izračunavanje. Haskell se zasniva na lambda računu i zbog toga se lambda nalazi na njegovom logou. Jezik ima otvorenu, javnu specifikaciju i postoje mnoge implementacije. Glavna implementacija Haskell, GHC, je istovremeno i intepretator i kompilator koji radi na mnogim platformama.

Haskell je specifično dizajniran za širok spektar upotreba, od numerčkih do simpoličkih. Zbog toga Haskell ima izraženu sintaksu i bogat izvor ugrađenih tipova podataka. Ovaj programski jezik se koristi u edukativne svrhe, ali ga takođe koriste IT odeljenja velikih investicionih banaka. Cryptol jezik i lanac za razvoj i verifikaciju kriptografskih algoritama je implementiran u Haskellu. Takođe, Facebook implementira svoje anti-spam programe u ovom programskom jeziku.

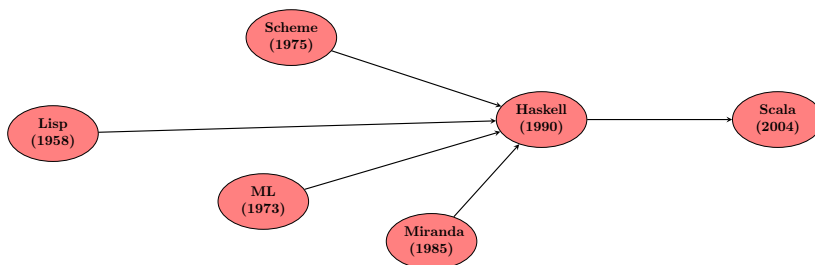
3 Razvojno stablo

Na nastanak jezika Haskell je najviše uticao programski jezik **Miranda**. Haskell je zasnovan na semantici, ali ne i na sintaksi programskog jezika Miranda. Pored Mirande, ogroman uticaj na nastanak Haskell

je imao i programski jezik ML. Tri osnovne karakteristike po kojima se Haskell razlikuje od ML:

- Haskell koristi nestriktnu semantiku,
- funkcije u Haskellu mogu biti opterećene,
- u njemu ne postoje izrazi i naredbe koji imaju bočne efekte.

Razvojno stablo jezika Haskell može se videti na slici 2.



Slika 2: Razvojno stablo jezika Haskell

3.1 Miranda

U vreme kada je Haskell nastao, Miranda je bio najzreliji i široko rasprostranjen ne strogi funkcionalni jezik [7]. Ona je proizvod kompanije *Research Software Limited*, Dejvida Tajnera, koju je on osnovao 1982 [1]. On je zamišljao da se Miranda koristi za lenjo funkcionalno programiranje u komercijalne svrhe. Zbog svoje dobro podržane implementacije, interaktivnog korisničkog interfejsa i mnogih udžbenika koji su napisani za ovaj programski jezik, ranih 90-ih on je korišćen na 250 univerziteta i 50 kompanija.

Programski jezik Miranda je imao snažan uticaj na Haskellov dizajn. Postoje mnoge sličnosti između ova dva jezika:

- lenjost,
- statičko kucanje,
- sintaksički izgled.

Postoje i vidljive razlike između Mirande i Haskell:

- Haskell ima bogatiju sintaksu za izraze,
- različita sintaksa za deklaraciju tipa podataka,
- Haskell nije usvojio apstraktni tip podataka, umesto toga koristi sistem modula.

Programski jezik Miranda je danas velikim delom zamenjen Haskellom, o čemu nam govori činjenica da je za ovaj programski jezik poslednji udžbenik izašao 1995, dok za Haskell udžbenici izlaze i danas. Ekonomski faktori su bili protiv Mirande, jer je *Research Software* bila mala kompanija koja je tražila novčanu naknadu za korišćenje svog softvera, tako da ni komercijalna ni akademska licenca nisu bile besplatne. Sa druge strane, Haskell je osmišljen i napravljen od strane grupe univerziteta sa javnim fondovima, tako da je bio besplatan. Još jedna činjenica koja ide protiv Mirande je to što ne postoji Windows verzija [1].

U početku Miranda jeste imao bolju implementaciju, ali se vremenom implementacija Haskell-a brže razvijala. Mala kompanija je imala poteškoća da ide u korak sa razvojem Haskell-a. On je dobio brži interaktivan interfejs, sličan Mirandinom, a Murov zakon je učinio Haskellov spor kompajler prihvatljivo brzim. Zbog novih važnih ideja, Haskell ubrzo postaje mnogo praktičniji izbor od Mirande.



Slika 3: Logo programskog jezika Miranda

3.2 Lisp

Lisp je najkorišćeniji funkcionalni programski jezik danas. Lisp je nastao 1958. na univerzitetu MIT, što ga čini jednim od najstarijih programskih jezika jer je od njega stariji jedino Fortran [5]. Razvio ga je Džon Mekarti. Naziv Lisp je nastao od izraza "LISt Processor", zato što su povezane liste jedna od glavnih struktura podataka u ovom programskom jeziku, a uz to je i izvorni kod Lispa sastavljen od listi. Najveću primenu ovaj programski jezik nalazi u oblasti veštačke inteligencije. Razlike između Lispa i Haskell-a:

- Prilikom pisanja programa u Lispu kompajler nije od preterane pomoći, ne pomaže prilikom pisanja koda,
- Lisp ima manji broj ugrađenih funkcija,
- Lisp ima jednostavniju sintaksu.



Slika 4: Logo programskog jezika Lisp

```
fun fac (0 : int) : int = 1
  | fac (n : int) : int = n *
    fac (n - 1)
```

Listing 1: ML

```
fun fac 0 = 1
  | fac n = n * fac (n - 1)
```

Listing 2: ML

Slika 5: Funkcija koja računa faktorijel u jeziku ML

3.3 ML

ML, skraćenica od "Meta Language", je funkcionalan programski jezik za opštu upotrebu [6]. Svoje korene nalazi u Lispu, i često ga nazivaju "Lisp sa tipovima". Razvio ga je Robin Milner 1970-ih godina na Univerzitetu u Edinburgu [9]. Njegova sintaksa je inspirisana sintaksom programskog jezika ISWIM. Za razliku od Haskell, ML nije isključivo funkcionalan jezik, jer on dozvoljava bočne efekte.

Osnovna namena ovog programskog jezika je metaprogramiranje, odnosno pisanje programa koji manipulišu drugim programima. Pored metaprogramiranja, ML se koristi i za:

- naučna istraživanja,
- dokazivanje teorema,
- finansijske sisteme i bioinformatiku,
- genealoške baze podataka.

3.4 Scheme

Scheme je programski jezik koji podržava više paradigmi, uključujući funkcionalno i imperativno programiranje [2]. Ovaj programski jezik nastao je tokom 1970-tih godina u MIT-ovoj laboratoriju za veštačku inteligenciju. U početku je bio namenjen za istraživanje i edukaciju. Objavili su ga dvojica programera, Guy L. Steele i Gerald Jay Sussma kroz niz beleški koji su sada poznate kao **Lambda papiri**. Za Scheme se može reći da je jedan od glavnih dijalekata programskog jezika Lisp koji je zahtevao samu implementaciju kako bi se izvela eliminacija repne rekurzije, dajući podršku funkcionalnom programiranju. Dizajniran je da ima izuzetno jasnu i jednostavnu semantiku i nekoliko različitih načina za formiranje izraza. Ovaj jezik takođe karakterišu: sakupljači otpada, statički doseg identifikatora, homoikoničnost i jaka tipiziranost.

Iako je ovo jezik opšte namene, vremenom su nastale brojne nadogradnje i implementacije jezika što je dovelo do nekompaktibilnosti kodova pisanih u različitim implementacijama. Zbog toga se on često naziva "**najnekompaktibilnijim programskim jezikom na svetu**".

3.5 Scala

Scala je moderan programski jezik koji spaja više paradigmi. Nastao je 2004. godine i njegova sintaksa podseća na sintaksu programskog jezika C. Dizajniran je da izrazi česte programske šablone kroz precizan, elegantan i tipski bezbedan način. Sve biblioteke koje su podržane u Javi, mogu se koristiti i u Scali [8]. Scala ima mnogo karakteristika funkcionalnih programskih jezika kao što su Scheme, Standard ML i Haskell, uključujući

podrazumevane tipove podataka, nepromenljivosti, lenje evaluacije, poklapanje obrazaca. Ime Scala je nastala od dve reči: skalabilnosti i jezika, što znači da je dizajnirana da napreduje sa zahtevima svojih korisnika. Jedna od većih primena ovog programskog jezika je za razvoj mobilnih aplikacija.



Slika 6: Logo programskog jezika Scala

4 Zaključak

U ovom tekstu ukratko su predstavljene osnove razvoja programskog jezika Haskell. Prikazani su jezici koji su najviše uticali na njegov nastanak i razvoj, kao i jezik na koji je on uticao i prikazano je njegovo elementarno razvojno stablo. Za dodatne informacije i praćenje daljeg razvoja, kao početnu odrednicu najbolje je koristiti zvaničnu stranu programskog jezika Haskell [4].

Literatura

- [1] Colin Myers Ellen Poon Chris Clack. *Programming with Miranda*. Prentice Hall, 1995.
- [2] R. Kent Dybvig. *The Scheme Programming Language*. the MIT Press, 1987.
- [3] Matematički fakultet. Dizajn programskih jezika. <http://www.programskijezici.matf.bg.ac.rs/DizajnProgramskihJezika.html>.
- [4] Haskell. Zvanična stranica programskog jezika Haskell. <https://www.haskell.org/>.
- [5] Lisp. Zvanična stranica programskog jezika Lisp. <https://lisp-lang.org/>.
- [6] Robin Milner. *A theory of type polymorphism in programming*. Journal of Computer and System Sciences, 1978.
- [7] Miranda. Sajt programskog jezika Miranda. <http://miranda.org.uk/>.
- [8] Scala. Zvanična stranica programskog jezika Scala. <https://www.scala-lang.org/>.
- [9] Jeffrey D. Ullman. *Elements of ML Programming*. Prentice-Hall, 1998.