Programski jezik F#

Seminarski rad u okviru kursa
Metodologija stručnog i naučnog rada
Matematički fakultet

Tijana Todorov, Tamara Garibovic,
David Nedeljkovic, Mihajlo Vicentijevic
tijana.todorov710@gmail.com, t.garibovic1995@gmail.com,
dnedeljkovic710@gmail.com, mihajlovicent@gmail.com

28. mart 2019

Sažetak

Dodati na kraju sazetak.

Sadržaj

1 Uvod

Dodati na kraju uvod u temu i obavezno izmeniti trenutni radni naslov.

2 U 2018. godini F# opisan je u dokumentaciji kao "funkcionalni programski jezik koji se pokrece na .NET platformi"[12], ali odakle je on potekao?

Istorija programskog jezika F# datira jos od 1970. godine pa sve do danas. U ranim 70-im godinama na Univerzitetu u Edinburgu Robin Milner sa jos nekoliko svojih kolega razvija jezik ML (Meta-Language) baziran na programskom jeziku LISP. Njegova onsnovna namena bila je pragmaticnog karaktera. Osmisljen je da pomogne u dokazivanju LCF (Logical computable functions) [10] teorema. ML jezik je koren svih jezika koji pripadaju familiji strogo tipiziranih funkcionalnih jezika, kao npr: Miranda, Haskell, Standard ML, Ocaml, EdinburghML, ReasonML, PureScript, a medju njima i F#.

Sve do danas kljucne ideje ove familije jezika ostaju osnova jezika F# koji se nadogradjivao kasnije iz dana u dan.

Tokom 80-ih godina dolazi do velike ekspanzije u kompjuterskoj industriji. Kako su se brzo razvijali softverski alati tako su se takodje pojavljivali novi jezici i programske paradigme. U tom periodu i Microsoft dozivljava veliku ekspanziju kao kompanija koja razvija operativne sisteme i aplikacije. Medjutim, u ovom periodu, a najvise u kasnim 80-im godinama pojavljuje se novi, objektno-orjentisani talas razmisljanja u programiranju koji veoma utice na razvoj softvera.

Pocetkom 90-ih godina dok je Microsoft tezio da odrzi monopol na trzistu i dok je u fokusu i dalje bila objektno-orjentisana paradigma, strogo tipizirani funkcionalni jezici bili su manje aktivni i okrenuti su ka razvitku na drugim poljima. Njihova primena uglavnom je imala ulogu u pronalazenju bagova, odrzavanju sistema i verifikaciji hardvera i softvera. Primeri jezika koji su se koristili za izgradnju oviakvih sistema su: Edinburgh ML, Standard ML, Ocaml, Caml Light, LISP. Takodje su neki funkcionalni jezci razvijeni u cilju istrazivanja u okviru paralelnog programiranja, kao sto su Parallel ML i vrsta paralelnog Haskell-a [12].

2.1 Zasto je nastao jezik F#?

Tokom 90-ih godina Microsoft razvija .NET [7, 12] platformu za razvoj softvera. Ideja je bila da se omoguci medjusobna kompatibilnost vise programskih jezika, odnosno da svaki jezik podrzan na ovoj platformi moze koristiti kôd nekog drugog jezika platforme. Glavni cilj Microsofta bio je da se na ovoj platformi podrzi sto veci broj jezika iz razlicitih paradigmi.

U okviru ovoga Don Sym razija projekat SML.NET koji je za ideju imao preusmeravanje Standard ML-a na .NET. Sistem je imao visok kvalitet, ali nije bio usvojen od strane programera. Pocetkom 2000-ih platforma .NET je vec uveliko zazivela, ali za jezik SML.NET nije bilo veceg interesovanja. Autor je imao veliku zelju da implementira strogo tipiziran funkcionalni jezik na nacin koji bi zainteresovao veliki broj programera. U

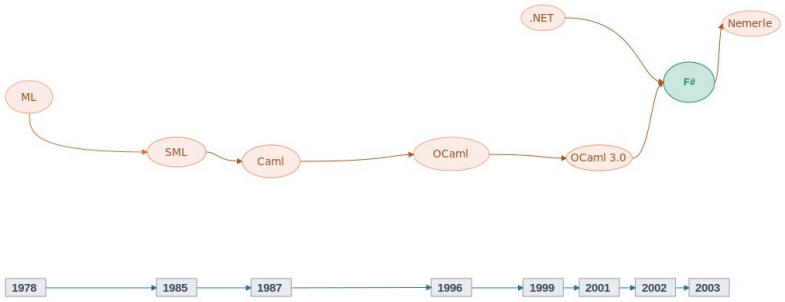
razmatranje ulazi i jezik OCaml, ali prethodno dolazi do pokusaja implementacije Haskell-a za .NET. Ovaj pokusaj bio je samo delimicno uspesan i primenjen na malim programima. Rad na daljoj implementaciji je zaustavljen.

U decembru 2001. godine Don Syme u razmatranje vraca jezik OCaml u cilju da se implementira za .NET platformu i razvija projekat Caml.NET koji ce se kasnije preimenovati u F#.

Inicijalna ideja F# programskog jezika bila je jednostavna. Trebalo je da poveze prednosti OCaml programskog jezika sa prednostima .NET platforme. 2002. godine pojavljuje se prva verzija F# 0.5 koja bila je slabo primecena. Don Syme 2004. godine nastavlja intenzivno da razvija ovaj jezik, a pocetkom 2005. godine izbaciuje prvu potpunu verziju ovog programskog jezika [12]. Poslednja aktuelna verzija ovog jezika je F# 4.1 [5].

2.2 Koji programski jezici su najvise uticali na razvoj jezika F#?

Najveci znacaj za razvoj jezika F# imaju jezici SML (Standard ML) i CAML (Categorical Abstarct Machine Language) porodica jezika koja je razvijana na Nacionalnom institutu za istrazivanje informatike i automatizacije u Francuskoj 1994.godine [6]. U okviru ovog projekta pod nazivom "Cristal project" razvija se jezik Objective Caml koji ima veoma visoke performanse i naucnici ga koriste na Linux i MAC OS X platformama. Kao sto je vec ranije opisano, ovaj jezik i .NET platforma imali su najveci uticaj na razvoj jezika F#. Na slici 1 moze se videti deo razvojnog stabla koji ovo prikazuje.



Slika 1: Razvojno stablo

3 Primena i mogucnosti

Snaga F# programskog jezika lezi u svedenoj sintaksi koja omogućava laku čitljivost kôda, kao i efikasan razvoj programa koji zahtevaju primenu složenih matematičkih algoritama. Jezik omogućava brzo generisanje prototipova i njihovu brzu transformaciju u produkcioni kôd. Kôd napisan u F#-u lako se može paralelizovati, što je posebno značajno danas kada svi novi računari imaju više jezgara. F# danas ima siroku primenu u obradi baza podataka, finansijskog modelovanja, statistici i bioinformatici. Takodje domen primene iz dana u dan raste.

F# je jezik koji ima pretezno funkcionalne karakteristike, ali on je svoju primenu pronasao u jos mnogo vrsta programiranja. Neki od njih su: imperativno, objektno-orjentisano, paralelno, distribuirano, asinhrono, meta programiranje, veb programiranje, skript programiranje, analiticko programiranje, i sl. Danas se on moze koristiti na velikom broju sistema, kao sto su: Linux, MAC, Windows, Android, iOS, i sl. O tome ce biti vise reci u nastavku.

4 Radni okviri

Najpoznatiji i najznačajniji radni okvir(framework) za ovaj programski jezik je .NET. Ovaj radni okvir je potpuno besplatan i na njemu je moguće kreirati veliki broj različitih aplikacija. On podržava više programskih jezika, programskih paradigmi, editora i biblioteka za igradnju mobilnih i veb aplikacija.

Temelj .NET platforme je zajednička jezička infrastruktura CLI (Common Language Infrastructure). Pokretanje F# koda na ovoj platformi se vrši tako što na samom početku kompajler prevodi kod u binarni fajl koji je preveden na asemblerski jezik višeg nivoa koji se zove MSIL (Microsoft Intermediate Language). Implementacija na CLI kompajleru ovog asemblerskog koda u toku izvršavanja je mnogo brža nego da je kod samo interpretiran i ova kompilacija se izvršava u trenutku[3]. Kod koji je preveden u MSIL i izvršen u trenutku, predstavlja kod za upravljanje za razliku od kodova pisanih na progamskim jezicima kao što su C ili C++ koji su sirovi kodovi.

Postoje neke prednosti za prolazak kroz CLI ili upravljani kod a ne direktnog kompiliranja na mašinski nivo:

- kompatibilnost medju jezicima
- mogućnost rada na više platformi
- mašinska nezavisnost

.NET platforma ima jos jednu prednost a to je prikupljanje smeća sto omogućava programeru da ne razmislja mnogo o alociranju memorije i oslobadjanju, već da se samo skoncentriše na pisanje koda. Iako ne mora da obraća pažnju na smeće i rad sa memorijom poželjno je da programer zna na koji način taj sakupljač smeća radi, kako on oslobadja memoriju i rešava taj problem.

Naravno da ova platforma nije jedina koja se koristi za ovaj progamski jezik, pored nje postoje: web radni okviri (Suave, Fable, ASP.NET Core, Giraffe, WebSharper, Freya, NancyFx, Serving Requests with IHttpHandler, Serving Requests with Azure, Functions, Pure F# Web API 2.0, SignalR, ServiceStack, ASP.NET Blazor) i radni okviri za testiranje weba (Web Testing, Frameworks, Canopy for Client-side Testing, Unit Testing Libraries)[4].

5 Instalacija

Uputstvo za instalaciju F#-a na Windowsu i Linuxu.

5.1 Windows

Na ovom operativnom sistemu postoje 3 načina za instalaciju F#-a i to su: Visual studio[8], Visual Studio Code [9] i JetBrains Rider [2].

5.1.1 Visual studio

Na Windows operativnim sistemima se uglavnom koristi Visual Studio alat.Visual studio 2017 je alat koji dolazi sa podrškom za F# u svim verzijama: Professional, Enterprise i Community (verzija je potpuno besplatna). Medjutim ukoliko imate već instaliran Visual Studio 2012/13/15 Professional ili above možete ga takodje koristiti zato što i on podržava alate za F# iako nije toliko napredan kao Visual Studio 2017.

5.1.2 Visual Studio Code

Visual Studio Code je besplatna platforma koja podržava mnogo jezika pa i F# . On je podržan od strane Ionide[1].

1.korak: Instalirati Visual studio Code za Windows

2.korak: pritisnuti Ctrl+Shift+p i sledeću naredbu za instalaciju Ionide paketa za Visual Strudio Code.

```
ext install Ionide-fsharp
```

5.1.3 JetBrains Rider

JetBrains Rider je platforma .NET IDE izgradjena je pomoću InteliJ i ReSharper tehnologije. JetBrains nudi podršku .NET i .NET Core aplikacijama na svim platformama.

1.korak: Instalirati JetBrains za Windows

2.korak(opciono): instalirati poslednju .NED Core SDK

Takodje vam je potrebno da instalirate ceo Visual Studio ili F# kompajler.

5.2 Linux

Instalacija F# na Linuxu se izvršava na potpuno identičan način za Ubuntu, Mint i Debian verziju.

5.2.1 Ubuntu/Mint/Debian

1.korak: Dodajte Mono[11] repozitorijum vašem menadžeru paketa

2.korak: Instalirati F# koji ce istovremeno povući novu verziju Mono-a ukoliko je to potrebno.

```
sudo apt-get update
sudo apt-get install fsharp
```

6 Zaključak

Sta je zakljucak celog rada.

Literatura

[1] Ionide. on-line at: http://ionide.io/.

[2] JetBrains s.r.o. Developed with drive and IntelliJ IDEA. Jet Brains rider, 2000-2019. on-line at: https://www.jetbrains.com/rider/.

[3] Smith Chris. *Programing F#*. O'Reilly Media, 1005 Gravenstein Highway North, 2009.

[4] F# Software Foundation and individual contributors. List of frameworks, 2012-2018. on-line at: https://fsharp.org/guides/web/.

[5] F# Software Foundation and individual contributors. The F# Language specification, 2012-2018. on-line at: https://fsharp.org/specs/language-spec/.

[6] Jon Harrop. *F# for Scientists*. Wiley-Interscience, New York, NY, USA, 2008.

[7] Microsoft. .NET framework. on-line at: https://dotnet.microsoft.com/learn/dotnet/what-is-dotnet.

[8] Microsoft. Visual Studio, 2019. on-line at: https://visualstudio.microsoft.com/.

[9] Microsoft. Visual Studio Code, 2019. on-line at: https://code.visualstudio.com/.

[10] Robin Milner. Logic for computable functions: Description of a machine implementation. Technical report, Stanford, CA, USA, 1972.

[11] Mono Project. Mono, 2019. on-line at: https://www.mono-project.com/.

[12] Don Syme. The Early History of F#. on-line at: https://fsharp.org/history/hopl-draft-1.pdf.

A Dodatak

Ovde pišem dodatne stvari, ukoliko za time ima potrebe. Ovde pišem dodatne stvari, ukoliko za time ima potrebe. Ovde pišem dodatne stvari, ukoliko za time ima potrebe. Ovde pišem dodatne stvari, ukoliko za time ima potrebe. Ovde pišem dodatne stvari, ukoliko za time ima potrebe.