

# Programski jezik F#

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

Tijana Todorov, Tamara Garibović,  
David Nedeljković, Mihajlo Vićentijević  
tijana.todorov710@gmail.com, t.garibovic1995@gmail.com,  
dneljkovic710@gmail.com, mihajlovicent@gmail.com

30. mart 2019

## Sažetak

Dodati na kraju sažetak.

## Sadržaj

<b>1</b>	<b>Uvod</b>	<b>2</b>
<b>2</b>	<b>Poreklo programskog jezika F#</b>	<b>2</b>
2.1	Zašto je nastao jezik F#?	2
2.2	Koji programski jezici su najviše uticali na razvoj jezika F#?	3
<b>3</b>	<b>Primena i mogućnosti</b>	<b>3</b>
<b>4</b>	<b>Radni okviri</b>	<b>4</b>
<b>5</b>	<b>Uputstvo za instalaciju</b>	<b>4</b>
5.1	Uputstvo za Windows	4
5.1.1	Visual studio	5
5.1.2	Visual Studio Code	5
5.1.3	JetBrains Rider	5
5.2	Uputstvo za Linux	5
5.2.1	Ubuntu/Mint/Debian	5
<b>6</b>	<b>Zaključak</b>	<b>5</b>
	<b>Literatura</b>	<b>5</b>

# 1 Uvod

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

## 2 Poreklo programskog jezika F#

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

Istorija programskog jezika F# datira još od 1970. godine pa sve do danas. U ranim 70-im godinama na Univerzitetu u Edinburgu Robin Milner sa još nekoliko svojih kolega razvija jezik ML (Meta-Language) baziran na programskom jeziku LISP. Njegova osnovna namena bila je pragmatičnog karaktera. Osmišljen 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 među njima i F#.

Sve do danas ključne ideje ove familije jezika ostaju osnova jezika F# koji se nadograđivao 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 takođe pojavljivali novi jezici i programske paradigme. U tom periodu i Microsoft doživljava veliku ekspanziju kao kompanija koja razvija operativne sisteme i aplikacije. Međutim, u ovom periodu, a najviše u kasnim 80-im godinama pojavljuje se novi, objektno-orjentisani talas razmišljanja u programiranju koji veoma utiče na razvoj softvera.

Početkom 90-ih godina, dok je Microsoft težio da održi monopol na tržištu i dok je u fokusu i dalje bila objektno-orjentisana paradigma, strogo tipizirani funkcionalni jezici bili su manje aktivni i okrenuti su ka razviku na drugim poljima. Njihova primena uglavnom je imala ulogu u pronalaženju bagova, održavanju sistema i verifikaciji hardvera i softvera. Primeri jezika koji su se koristili za izgradnju ovakvih sistema su: Edinburgh ML, Standard ML, Ocaml, Caml Light, LISP. Takođe, neki funkcionalni jezici razvijeni su u cilju istraživanja u okviru paralelnog programiranja, kao što su Parallel ML i vrsta paralelnog Haskell-a [12].

### 2.1 Zašto je nastao jezik F#?

Tokom 90-ih godina Microsoft razvija .NET [7, 12] platformu za razvoj softvera. Ideja je bila da se omogući međusobna kompatibilnost više programskih jezika, odnosno da svaki jezik podržan na ovoj platformi može da koristi kôd nekog drugog jezika platforme. Glavni cilj Microsoft-a bio je da se na ovoj platformi podrži što veći broj jezika iz različitih paradigmi.

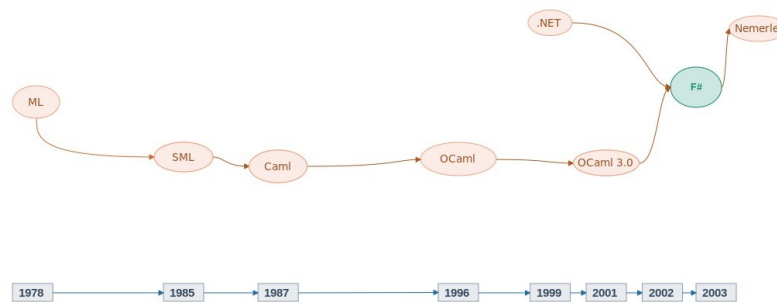
U okviru ovoga Don Syme razvija projekat SML.NET koji je za ideju imao preusmeravanje Standard ML-a na .NET. Sistem je imao visok kvalitet, ali nije bio usvojen od strane programera. Početkom 2000-ih platforma .NET je već uveliko zaživela, ali za jezik SML.NET nije bilo većeg interesovanja. Autor je imao veliku želju da implementira strogo tipiziran funkcionalni jezik na način koji bi zainteresovao veliki broj programera. U razmatranje ulazi i jezik OCaml, ali prethodno dolazi do pokušaja implementacije Haskell-a za .NET. Ovaj pokušaj bio je samo delimično uspešan i primenjen na malim programima. Rad na daljoj implementaciji je završen.

U decembru 2001. godine Don Syme u razmatranje vraća jezik OCaml i razvija projekat Caml.NET koji će se kasnije preimenovati u F#.

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

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

Najveći značaj za razvoj jezika F# imaju jezik SML (Standard ML) i CAML (Categorical Abstract Machine Language) porodica jezika koja je razvijana na Nacionalnom institutu za istraživanje 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 naučnici ga koriste na Linux i MAC OS X platformama. Kao što je već ranije opisano, jezik OCaml i .NET platforma imali su najveći uticaj na razvoj jezika F#. Na slici 1 može se videti deo razvojnog stabla koji ovo prikazuje.



Slika 1: Razvojno stablo

## 3 Primena i mogućnosti

Snaga F# programskog jezika leži 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 produkcionu 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 široku primenu u obradi baza podataka, finansijskog modelovanja, statistici i bioinformatici. Takođe, domen primene svakodnevno raste.

F# je jezik koji ima pretežno funkcionalne karakteristike, ali on je svoju primenu pronašao u još mnogo vrsta programiranja. Neki od njih su: imperativno, objektno-orjentisano, paralelno, distribuirano, asinhrono, meta programiranje, veb programiranje, skript programiranje, analitičko programiranje, i sl. Danas se on može koristiti na velikom broju sistema, kao

što su: Linux, MAC, Windows, Android, iOS, i sl. O ovome će biti više reči u nastavku.

## 4 Radni okviri

Najpoznatiji i najznačajniji radni okvir (framework) za ovaj programski jezik je .NET koji je ranije već pomeut. Ovaj radni okvir je potpuno besplatan i na njemu je moguće kreirati veliki broj različitih aplikacija. On podržava veliki broj programskih jezika iz različitih 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 programskim 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 razmislija 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 programski 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 Uputstvo za instalaciju

U ovom poglavlju biće opisano uputstvo za instalaciju programskog jezika F# na Windows i Linux operativnim sistemima.

### 5.1 Uputstvo za Windows

Na ovom operativnom sistemu postoje tri načina za instalaciju F#-a. 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). Međutim, ukoliko imate već instaliran Visual Studio 2012/13/15 Professional ili above možete ga takođe 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, a među njima i F#. On je podržan od strane Ionide[1].

**1.korak:** Instalirati Visual studio Code za Windows

**2.korak:** Pritisnuti **Ctrl+Shift+p** i ukucati sledeću naredbu za instalaciju Ionide paketa za Visual Studio Code:

```
1000 ext install Ionide-fsharp
```

### 5.1.3 JetBrains Rider

JetBrains Rider je platforma .NET IDE izgrađena je pomoću IntelliJ 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 .NET Core SDK

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

## 5.2 Uputstvo za 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 će istovremeno povući novu verziju Mono-a ukoliko je to potrebno.

```
1000 sudo apt-get update
sudo apt-get install fsharp
```

## 6 Zaključak

Sta je zaključak 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>.