F# na .NET platformi

T.Todorov T.Garibović D.Nedeljković M.Vićentijević

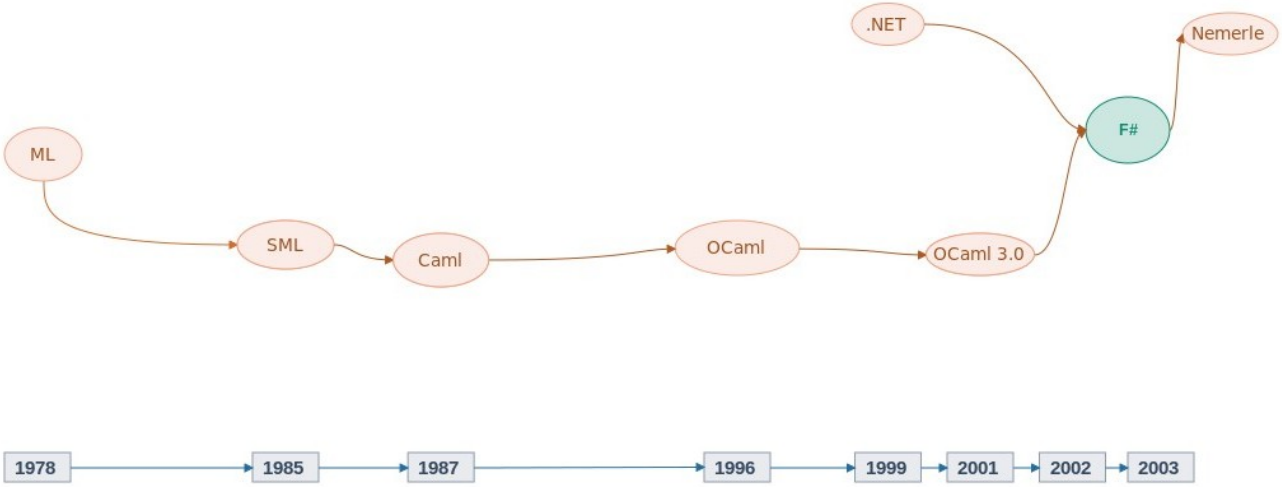
7. maj 2019

Pregled

Uvod

- F# je jezik koji danas ima veoma široku upotrebu
- Nastao je iz ideje da se strogo tipizirani funkcionalni jezik preusmeri na .NET platformu
- Autor jezika je Don Sajm (eng. *Don Syme*)
- Pretežno je funkcionalan jezik, ali podržava još dosta programskih paradigmi
- F# ima jednostavnu sintaksu, a dovoljno moćnu i za složene matematičke probleme
- Cilj rada je izdvajanje nekih specifičnih karakteristika

Poreklo i uticaj drugih jezika



Slika 1: Razvojno stablo

Primena i mogućnosti

- Jednostavna sintaksa omogućava laku čitljivost koda
- Jezik omogućava brzo generisanje prototipova
- Kôd napisan u F#-u lako može da se paralelizuje
- Jezik ima primenu u oblastima: bioinformatika, baze podataka, statistika, finansijsko modelovanje, ...
- Osim funkcionalnog, jezik F#, podržava još neke tipove programiranja: imperativno, objektno-orjentisano, paralelno, distribuirano, asinhrono, veb programiranje, skript programiranje, ...
- Koristi se na dosta operativnih sistema: Linux, MAC, Windows, Android, iOS, ...

Osobine i specifičnosti

Osobine:

- Bezbedan
- Funkcionalan
- Strogo tipiziran
- Automatski zaključuje tipove
- Kompatibilan

Specifičnosti:

- Povratna vrednost if/else
- Opciono - return
- Ključne reči let i mutable
- Pattern matching
- Novi tip option type

Funcionalna paradigma - Pattern matching

Pattern matching

Pattern matching je mehanizam koji koristi dekompoziciju i kontrolu toka podataka za poklapanje obrazaca koriscenjem navedene konstrukcije: **match ... with ...**

```
1 let urlFilter url port =
2  match (url,port) with
3  | "http://www.control.org", 99 -> true
4  | "http://www.kaos.org" , _ -> false
5  | _, 86 -> true
6  | _ -> false
```

Asinhrono i paralelno programiranje

- **Asinhrono programiranje**
  - Opisuje programe i operacije koje se izvršavaju u pozadini i završavaju nakon nekog vremena
  - Programe možemo pisati korišćenjem **APM** biblioteke
  - APM deli asinhrone operacije na dve metode: BeginOperation i EndOperation
- **Paralelno programiranje**
  - Na .NET 4.0 platformi se koristi **PFX** biblioteka čija je osnovna struktura **Task objekat**
  - Zbog problema Task objekta sa deljenim podacima uvodi se zaključavanje podataka
  - PFX biblioteka uvodi nove kolekcije System.Collections.Concurrent

Radni okvir - .NET framework

- **.NET platforma** je platforma koja podržava veliki broj različitih kompatabilnih jezika, programskih paradigmi ali i .NET radni okvir (eng. *framework*).
- .NET radni okvir je pojednostavio razvoj RP(*Reactive Programming*) aplikacija ali i podržava veliki broj biblioteka i editora.
- Temelj .NET platforme je zajednička jezička infrastruktura **CLI** (*Common Language Infrastructure).*
- Kodovi se prevode na **MSIL** (*Microsoft Intermediate Language*) asemblerski jezik.

Radni okvir - .NET framework

- Implementacija MSIL-a na CLI kompajleru je brža i ima sledeće prednosti u odnosu na mašinski:
  - kompatibilnost među jezicima
  - mogućnost rada na više platformi
  - mašinska nezavisnost
- Mogućnost automatskog prikupljanja smeća je još jedna prednost.

Još neki radni okviri: **veb** radni okviri(*Suave, Fable, ASP.NET Core...*) i radni okviri za **testiranje veba** (*Web Testing, Frameworks, Unit Testing Libraries...*)

Instalacija i pokretanje

- Alati koji na **Windows-u** podržavaju F# instaliraju se u nekoliko koraka:
  - Visual Studio Code
  - Visual studio
  - JetBrains Rider
- Na **Linux-u** se instalacija vrši na isti način za sledece verzije:
  - Ubuntu
  - Mint
  - Debian

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

Fizz Buzz

```
let (|Fizz|Buzz|FizzBuzz|Other|) n =
    match (n % 3, n % 5) with
    | 0, 0 -> FizzBuzz
    | 0, _ -> Fizz
    | _, 0 -> Buzz
    | _ -> Other n

let fizzBuzz =
    function
    | Fizz -> "Fizz"
    | Buzz -> "Buzz"
    | FizzBuzz -> "FizzBuzz"
    | Other n -> n.ToString()

seq { 1..100 } |> Seq.map fizzBuzz|> Seq.iter (printfn "%s")
```

Jedinica mere

- Pad orbitera poslatog na Mars 1999.
  - Uzrokovan činjenicom da je deo softvera koristio numeričke, a deo softvera engleske jedinice
- Prevencija grešaka na osnovu konteksta primene
  - Numeričkim tipovima se pridružuju metapodaci
  - Kompajler na osnovu metapodataka proverava ispravnost
- Jedinstveno svojstvo jezika F#
- Primer definisanja jedinice mere

```
1  [<Measure>] type cm
2  [<Measure>] type inch
```

Jedinica mere

```
1  [<Measure>] type rsd
2  [<Measure>] type eur
3  [<Measure>] type hour
4  [<Measure>] type week
5  [<Measure>] type year
6
7  let hoursBilledPerWeek = 40.0<hour/week>
8  let weeksWorkedPerYear = 35.0<week/year>
9  let rsdPerHour = 1000.0<rsd/hour>
10 let exchangeRate = 0.008547<eur/rsd>
11
12 let eurPerYear = rsdPerHour * hoursBilledPerWeek *
       weeksWorkedPerYear * exchangeRate
13 let bonus = 500.0<eur/year>
14
15 printfn "%f" (eurPerYear + bonus)
```

Literatura

- Don Syme, The Early History of F#
- Smith Chris, Programing F# O'Reilly (2009)
- Antonio Cisternino, Don Syme, Adam Granicz, Expert F# (2007)
- Jon Harrop, F# for Scientists (2008)
- https://fsharp.org

Hvala na pažnji!