Programski jezik SWIFT

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

Anđelković Dragica, Nikolić Igor,
Pejović Petar, Mandić Igor
andjelkovic.dragica96@gmail.com, igor.nikolic032@hotmail.com,
petar.pejovic8@gmail.com, igormandic996@gmail.com

2. april 2019

Sažetak

U ovom tekstu je ukratko prikazana osnovna forma seminarskog rada

Sadržaj

1 Uvod **2**

2 Nastanak i istorijski razvoj, uticaji drugih programskih jezika **2**
2.1 Swift 1 . . . 3
2.2 Swift 2 . . . 3
2.3 Swift 3 . . . 3
2.4 Swift 4 . . . 3

3 Osnovna namena, svrha i mogućnosti **3**

4 Osnovne osobine ovog programskog jezika, podržane paradigme i koncepti **4**

5 Najpoznatija okruženja (framework) za korišćenje ovog jezika i njihove karakteristike **5**

6 Instalacija i uputstvo za pokretanje na Linux/Windows operativnim sistemima **5**

7 Primer jednostavnog koda i njegovo objašnjene **5**

8 Sve ono što je specifično i važno za sam taj programski jezik **6**

9 Zaključak **6**

Literatura **6**

A Dodatak **6**

1 Uvod

Swift je novi programski jezik opšte namene razvijen od strane kompanije Apple za iOS, macOS, watchOS, tvOS, Linux i z/OS. Dizajniran je da radi u Apple radnim okruženjima, Cocoa i Cocoa Touch i postojećeg Objectiv-C koda pisanog za Apple proizvode. Podržava imperativni, objektno-orjantisani i funkcionalni način programiranja. Napravljen je upotrebom LLVM programskog prevodioca otvorenog koda i ukljucen je u Xcode, počev od verzije 6. Swift koristi izvršno okruženje programskog jezika Objective-C, što omogućava izvršavanje C, C++, Objective-C i Swift koda u okviru jednog programa. Namera kompanije Apple je bila da Swift podrži mnoge ključne koncepte povezane sa programskim jezikom Objectiv-C.

2 Nastanak i istorijski razvoj, uticaji drugih programskih jezika

Razvoj programskog jezika Swift je započeo 2010. godine Chirs Lattner, koji je implementirao veći deo osnovne strukture jezika, za čije je postojanje znala samo nekolicina ljudi. Tek su krajem 2011. godine i drugi programeri počeli da sarađuju na projektu Swift, a u julu 2013 godine on je posato glavni fokus grupe Apple Developer Tools.

Swift je predstavljen na međunarodnoj konferenciji programera (WWDC-Worldwide Developers Conference) 2014. godine, uz integrisano razvojno okruženje Xcode 6 i OS 8. U decembru 2015. godine Apple je zvanično izdao Swift kao projekat otvorenog koda i pokrenuo je veb sajt `http://swift.org`, koji je posvećen zajednici Swift. Swift skladište se nalazi na GitHub stranici kompanije Apple (`http://github.com/apple`). Swift razvojno skladište (`https://github.com/apple/swift-evolution`) prati napredak Swifta, dokumentujući predložene promene. U razvojnom skladištu se može pronaći lista predloženih promena koje su prihvaćene i onih koje su odbijene. Swift 3 sadrži nekoliko poboljšanja koje je preporučila zajednica programera. Na razvoj Swifta uticali su mnogi programski jezici, od kojih su najznačajniji: Objective-C, Ruby, Haskell, C#, Python. U tabeli 1 se nalaze sve do sada izbačene verzije programskog jezika Swift, u hronološkom redosledu.

Tabela 1: Istorijski razvoj programskog jezika Swift.

Datum	Verzija
2014-09-09	Swift 1.0
2014-10-22	Swift 1.1
2015-04-08	Swift 1.2
2015-09-21	Swift 2.0
2016-09-13	Swift 3.0
2017-09-19	Swift 4.0
2018-03-29	Swift 4.1
2018-09-17	Swift 4.2
2019-02-28	Swift 4.3
...	Swift 5.0

2.1 Swift 1

Prvu verziju karakteriše REPL alat koji omogućava izvršavanje manjih fragmenata Swift koda i njegovo testiranje sa komandne linije. U Swift 1.2 verziji uvedena je nova struktura podataka - skup. Ova verzija je donela poboljšanja performansi kompajlera i smanjila vreme potrebno za kompajliranje Swift programa. Prilikom pokretanja projekta, kompajliraju se samo fajlovi kod kojih je detektovana izmena, što je posebno značajno kod većih projekata.

2.2 Swift 2

Glavne funkcionalnosti koje su ugrađene u programski jezik su:

- programiranje orjentisano na protokole,
- model za obradu grešaka,
- odlaganje izvršavanja naredbe pomoću ključne reči defer,
- provera da li je funkcija dostupna na trenutnoj verziji uređaja i platforme koja pokreće našu aplikaciju pomoću ključne reči available.

U drugoj verziji, kao deo novog projekta, predstvljen je Swift paket menadžer za upravljanje Swift bibliotekama. Kao priprema za naredne verzije dodata je provera verzije izvršnog okruženja.

2.3 Swift 3

Treća verzija sadrži osnovne promene u samom jeziku i biiblioteci Swift standarda, zbog toga nije kompatibilan sa prethodnim verzijama Swift jezika. Jedan od osnovnih ciljeva autora Swifta 3 je da on bude kompatibilan na više platformi, tako da kod koji se napiše za jednu platformu bude kompatibilan na svim drugim platformama. To znači da će kod koji se napise za MAC OS funkcionisati i na Linuxu.

2.4 Swift 4

Četvrta verzija je kompatibilna sa trećom. Nove karakteristike koje je podržala četvrta verzija su:

- parsiranje fajlova u json i xml formatu,
- jednostrano definisani opsezi,
- poboljšanje funkcionalnosti struktura podataka: rečnik i skup,
- kombinovanje klasa i protokola,
- pisanje generičkih podskripti.

3 Osnovna namena, svrha i mogućnosti

Pomoću Swift programskog jezika moguće je razviti bilo koji tip iOS i macOS aplikacija. Cilj Swift projekta je da stvori najbolji raspoloživi jezik za upotrebu od programiranja sistema, preko razvoja mobilnih i desktop aplikacija do cloud usluga. Takođe, ubrzan je proces razvoja proizvoda, poboljšane su performanse i povećana sigurnost aplikacija.

Jedna od glavnih namena Swift programskog jezika je kreiranje mobilnih aplikacija za iPhone i iPad uređaje. Swift je moguće izvršavati na

Linux operativnom sistemu i Raspberry Pi. Članovi zajednice rade na stvaranju Swift aplikacija koje će se izvršavati i na Android platformama.

Osim što je Swift poznat po razvoju aplikacija za Apple platforme, koristi se i u modernim server aplikacijama. Swift je odličan izbor za server aplikacije koje zahtevaju visoke performanse kompajlera, nizak stepen korišćenja memorije i visok nivo bezbednosti.

Swift je sve popularniji programski jezik za razvoj IoT aplikacija. Kako bi kompanija Apple postala lider u primeni IoT aplikacija, razvijene su biblioteke i razvojni okviri koje rade najveći deo posla, dok se programeri mogu fokusirati na funcionalnosti IoT aplikacija.

Neka od mogućnosti koje pruža pomoću svojih funkcija:

- Automatsko utvrđivanje tipova - Swift može automatski da utvrdi tip promenljive ili konstante na osnovu inicijalne vrednosti.
- Generički tipovi - generički tipovi omogućavaju da se piše kod jednom za izvršenje identičnih zadataka za različite tipove objekata dok se zadržava bezbednost tipa.
- Promenljivost kolekcije - Swift nema posebne objekte za promenljive ili nepromenljive kontejnere. Umesto toga, možete da definišete promenljivost definisanjem kontejnera kao konstante ili kao promenljive.
- Sintaksa zatvorenog izraza - zatvoreni izrazi su samostalni blokovi funkcionalnosti koji mogu da se proslede i upotrebe u kodu.
- Pseudoklase - pseudoklasa definiše promenljivu koja možda nema vrednost.
- Switch iskaz - Switch iskaz je drastično poboljšan funkcijama, kao što su poklapanje šablona i zaštitni uslovi; zahvaljujući njima, izbegnute su automatske greške.
- Višestruki povratni tipovi - funkcije mogu da imaju višestruke povratne tipove upotrebom torki.
- Preklapanje operatora - klase mogu da obezbede sopstvenu implementaciju postojećih operatora.
- Nabrajanja sa pratećim vrednostima - u Swiftu može da se uradi mnogo više od jednostavnog definisanja grupe srodnih vrednosti pomoću nabrajanja.

Postoji još jedna funkcija, koja tehnički nije funkcija Swifta, već Xcodea i kompajlera. To je Mix and match. Ona omogućava kreiranje aplikacija koje sadrže Objective-C i Swift fajlove. To omogućava da se sistematski ažuriraju aktuelne Objective-C aplikacije pomoću Swift klasa i upotrebu Objective-C biblioteka/radnih okvira u Swift aplikacijama.

4 Osnovne osobine ovog programskog jezika, podržane paradigme i koncepti

Moj deo
Moj deo
Moj deo
Moj deo
Moj deo
Moj deo
Moj deo

5 Najpoznatija okruženja (framework) za korišćenje ovog jezika i njihove karakteristike

Igorov deo
Igorov deo
Igorov deo
Igorov deo
Igorov deo
Igorov deo
Igorov deo

6 Instalacija i uputstvo za pokretanje na Linux/Windows operativnim sistemima

Igorov deo
Igorov deo
Igorov deo
Igorov deo
Igorov deo
Igorov deo
Igorov deo

7 Primer jednostavnog koda i njegovo objašnjene

Pekijev deo
Pekijev deo
Pekijev deo
Pekijev deo
Pekijev deo
Pekijev deo
Pekijev deo
Pekijev deo

Sablon za pisanje koda u latehu.

```
1000 # This program adds up integers in the command line
     import sys
1002 try:
         total = sum(int(arg) for arg in sys.argv[1:])
1004     print 'sum =', total
     except ValueError:
1006     print 'Please supply integer arguments'
```

Listing 1: Primer ubacivanja koda u tekst

8 Sve ono što je specifično i važno za sam taj programski jezik

Moj deo
Moj deo
Moj deo
Moj deo
Moj deo
Moj deo
Moj deo

9 Zaključak

Ovde pišem zaključak. Ovde pišem zaključak. Ovde pišem zaključak. Ovde pišem zaključak. Ovde pišem zaključak. Ovde pišem zaključak. Ovde pišem zaključak. Ovde pišem zaključak. Ovde pišem zaključak. Ovde pišem zaključak. Ovde pišem zaključak. Ovde pišem zaključak.

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.