

Funkcionális programozás a gyakorlatban

előadás

Iván Szabolcs – Számítástudomány Alapjai Tanszék,
SZTE TTIK

2021 © CC BY NC SA 4.0, 2021.03.03.



flatMap

Bevezetés

Mi a cél?

A funkcionális paradigma, melyben az imperatív paradigmában megszokott változók, ciklusok, mutable state helyett (immutable) értékeket, függvényeket és rekurziót használunk, erős típusozottsággal, mellékhatások nélkül, olyan programok fejlesztését teszi lehetővé, melyek – kis túlzással – “ha lefordulnak, akkor helyesek is lesznek”. Mindezt kombinálva a Scala mint JVM programnyelv teljes hozzáféréssel a Java csomagrendszeréhez, és hogy egy projekten belül aszerint választhatunk az imperatív Java és a funkcionális Scala implementáció közül, hogy az adott feladatra éppen melyik az alkalmasabb, egy hatékony, gyors és helyes fejlesztési ciklust lehetővé tevő környezetet kapunk.

A kurzus egyrészt betekintést ad a funkcionális paradigmába oly módon, hogy az más FP nyelvekben is felhasználható legyen, másrészt a Scala nyelv egyes specifikus elemeit is ismerteti a Java-Scala integrációval együtt, hogy a kurzust elvégzettek képesek legyenek dolgozni adott esetben egy már létező Scala frameworkkel.

Milyen előzetes tudásra épít?

A tananyag létrejöttét indokló kurzus – tekintve, hogy második féléves kötelező tárgy az üzemmérnök informatikus szakon – minimális előfeltételeket támaszt: a nyelvi elemeket és a paradigma alapkoncepcióit is a nulláról igyekszik felépíteni. Ugyanakkor, a rekurzió megértéséhez valamennyi algoritmus-tervezési érzék szükséges lehet, és az imperatív nyelvi elemekkel is összevetjük a funkcionálisakat, így módon egy imperatív programozás kurzus (mint amilyen az előfeltétel Programozás alapjai) anyagának ismerete szükséges, ha nem is egy konkrét programozási nyelv vagy környezeté.

A félév közepe felétől elmozdulva a core FP paradigmától a Scala nyelv elemeire fokozatosan kerül át a hangsúly, így módon a félév során a Java nyelv elemeinek megismerése is szükségessé válik, melyet a kurzus párhuzamos feltételén, a Programozás I-en tanulnak a szak hallgatói.

Milyen referenciákra épül? Miért hasznos egy újabb tananyag?

A szerző elvégezte a Scala nyelvet és a javac fordítót megalkotó Martin Odersky nevével fémjelzett [Functional Programming in Scala specializációt](#) Courserán. A kurzus anyagának egy része – részben azért is, hogy a hallgatók a nemzetközi terminológiát is megismerjék – hasonló ezen kurzuséhoz, azonban attól annyiban mindenképp eltér, hogy olyanok számára is elérhető, akik a Java nyelvvel még csak ismerkednek (a Courserás specializáció bevallottan olyanok számára készült, akik olyan erős Java, OOP és tervezési minta alapokkal rendelkeznek, mely a BProf képzés elejében tartóktól semmiképp nem várható el).

A szerző szerencsésnek tartja, ha a funkcionális paradigmával a hallgatók már a képzés elején megismerkednek, minél később kell “felülírniuk” az imperatív nyelvekben tanult ismereteket és szemléletmódot, annál nehezebbé válik ez a váltás. Ily módon a tananyag – ugyan mélységében természetesen nem kíván versenyezni sem a fentebb említett specializációval, sem egy kifejezetten Scala-specifikus kurzussal – egy olyan “early expose” a hallgatók számára, amilyen az SZTE berkein belül nem létezik (a Funkcionális programozás kurzus állna hozzá legközelebb, de a szerző ismeretei szerint a bolognai átmenet során ez a kurzus 2005 óta lényegében megszűnt, és egy egész más, főként akadémiai szférában alkalmazott programozási nyelven, a Haskellben keresztül ismertette a paradigmát).

Mennyire használható és naprakész tudást ad?

A Java Virtual Machine eszközök milliárdjain fut, androidos telefonoktól kezdve minden számítástechnikai eszköz (kivéve talán a mikrokontrollereket, melyeket leginkább natív C-ben programoznak ma is) képes ezáltal Java bytecode-ot futtatni. Mivel a Scala nyelv fordítója egy teljesen JVM-kompatibilis bytecode-ot készít, ez azt jelenti, hogy a Scalában írt programok minden nehézség nélkül (mind fejlesztői, mind felhasználói szemszögből) futtathatóak szinte tetszőleges architektúrán és környezetben. A tananyagot keresztül szerezhető kompetenciáknak része, hogy az olvasó a végére képes lesz felmérni, hogy egy adott feladatot inkább a szokott imperatív módon, vagy funkcionálisan lenne jobb megoldani, és képes lesz eddigi ismereteit és eszköztárát nagyon hatékonyan bővíteni. A Scala nyelv népszerűségét és használhatóságát mutatja, hogy [egy szakmai lista szerint](#) 2021-ben data science és cloud computing (a large scale alkalmazások fejlesztésénél nagyon hasznos, ha a programunk függvényei garantáltan mellékhatás-mentesek, számos hibalehetőség eleve fel sem merül ekkor, az említett két terület pedig a szoftverfejlesztésben nagyon is “hot topic” manapság) célra a top 5 nyelvben szerepel, a linkelt oldal szerint is “Scala is currently considered to be one of the best programming languages for functional programming”.

Milyen formában adja át a tudást? Milyen eszköztárakat használ?

A tananyag egy [önálló weboldalon](#) érhető el. Az előadás anyag kódrészleteket is tartalmaz, gyakori közbeszúrt kérdésekkel, melyekre a (jellemzően rövid) válasz lenyitható, önellenőrzés céljára, a lényegesebb kulcsszavak, új fogalmak kiemelve szerepelnek; számos aloldalhoz tartozik egy-egy maximum 20 perces videósegédlet (melyek az egész tárgy vonatkozásában külön gyűjtve is elérhetőek egy [YouTube playlistben](#), ha valaki egyfajta “podcast”ként szeretné inkább feldolgozni az anyagot, logikailag egymásra épülő sorrendben hallgatva az előadást és a gyakorlatot).

Hogyan érdemes használni a tananyagot?

A tananyag 34 kisebb alpontra oszlik. Egyszerre érdemes egy alpontot feldolgozni, elolvasva az írásos anyagot és (ajánlottan) átnézve a videóanyagot, oly módon, hogy a hallgató megpróbálja meg is oldani a kérdésként feltett szoftverfejlesztési feladatokat, és miután ezt (sikeresen vagy sem) megtette, nézi meg a megoldást. Fontos, hogy a kérdésekre megpróbáljon az Olvasó maga válaszolni, mielőtt megnézi a megoldást.

Az előadás anyag egyes leckéinek megismerése után az ugyanezen weblapon elérhető gyakorlati anyag (mely jellemzően hosszabb, szintetizáló jellegű feladatokat tartalmaz az előadás rövid, "unit test" jellegű kérdéseihöz képest) vonatkozó leckéjében szereplő feladatok tényleges megoldása egy fejlesztőkörnyezetben javasolt, így mélyülhet el és ültetődhet át a gyakorlatba az előadáson elsajátított absztrakt fogalmi tárház. Továbbá, számos esetben van a gyakorlati anyagban olyan visszautalás az előadás anyagára, melyben is az absztrakt fogalmat konkrét szinten implementálva ismerik meg a hallgatók, ezután az előadás további részeiben is – a könnyebb befogadhatóság érdekében – a konkrét szintet is használni fogjuk példakódjainkban és a mini feladatokban, ezért is nagyon ajánlott az előadás és a gyakorlat párhuzamos feldolgozása.

Témakörök

A funkcionális paradigma eszköztára

- típusok és értékek változók helyett
- szelekció (if), függvények és a substitution model
- rekurzió ciklusok helyett
- a call stack és a tail rekurzió
- tail rekurzív függvény készítése
- rekurzív kód helyességigazolása
- a függvény típus
- algebrai adattípusok és a lista
- mintaillesztés
- típusparaméterek
- higher-order műveletek kollektciókon: map, filter, reduce és fold
- a foreach konstrukció, for comprehension
- map, for és yield
- egymásbaágyazott for comprehension: flatMap
- monádok
- ko- és kontravariancia
- a variancia szabályai I és II

A Scala nyelv elemei

- függvény deklarálása másik függvény scope-jában
- case class és tuple: a szorzat típusok
- trait: az összeg típus
- case object: az üres szorzat
- member függvények: az FP-OOP összefonódás
- a beépített scala.collection.immutable.List osztály
- companion objectek és az apply metódus
- a Vector monád
- az Option monád
- a Set monád
- a Map kollekció
- a collect metódus
- a Scala típushierarchiája: Any, AnyRef, AnyVal, Null és Nothing
- a Try monád
- a Future monád

Funkcionális programozás a gyakorlatban

Iván Szabolcs – Számítástudomány Alapjai Tanszék,
SZTE TTIK

2021 © CC BY NC SA 4.0, 2021.03.03.

Jelen tananyag a Szegedi Tudományegyetemen
készült az Európai Unió támogatásával.

Projektazonosító: EFOP-3.4.3-16-2016-00014

SZÉCHENYI 2020



MAGYARORSZÁG
KORMÁNYA

Európai Unió
Európai Szociális
Alap



BEFEKTETÉS A JÖVŐBE