Functional Programming with Scala

John Nestor
47 Degrees, Persist Software

uw@persist.com

Oct 17, 2016

Setup

- Zoom
- Canvas
- Anyone not on a Mac?
- Anyone using Eclipse (rather than Intellij)?
- USB Flash Drive
  - copy /uw/week1 to your computer
  - Also available on Canvas

Intros and Attendance

Outline Today

- Hour 1. Intro to Scala, SBT, and Intellij
- Hour 2. Martin Odersky, Scala the Simple Parts
- Hour 3. Simple Scala with the REPL
  Reading and Programming Assignment

Review Syllabus (NYA)

- Textbook (print and/or ebook)
- Lectures
- Reading assignments
- Programming assignments
- Grading
- Participation and working together
- Must do each programming assignment by yourself

Textbook

- Odersky. Programming in Scala (3rd Ed)
- Wampler. Programming in Scala (2nd Ed)

Grading

- Pass-Fail
- 100 points
    - 10 for attending each of 10 classes
    - 10 each for the 9 programming assignments

Grading Assignments

- In order (most important at top)
  - Turn in on time (upload to Canvas)
  - Passes test / meets requirements
  - Easy to understand
  - Use immutable data structures and functional programming when possible
  - Uses good Scala style

Outline Future Classes

- Hour 1.
  - Show and review programming assignment solutions
  - Look at other code
- Hour 2.
  - Interactive exercises
  - Lecture
- Hour 3.
  - More Lecture
  - Discuss new programming assignment

Setup

- Check https://github.com/47deg/scala-setup
- Install java version 8
  - **`java -version`**
- Install SBT
  - **`sbt`**
  - **`run`**
  - **`test`**
  - **`exit`**
- Install Intellij (or Eclipse) on your own

Office Hours

- At 47 Degrees, 321 3rd Ave (opposite King St Station)
- Online via Zoom
- Time by request
- Send me an email in advance if you want to meet at
  - uw@persist.com

Scala Language

Why Scala?

- Strong typing
- Concise elegant syntax
- Runs on JVM (Java Virtual Machine)
- Supports both object-oriented and functional
- Small simple programs through large parallel distributed systems
- Easy to cleanly extend with new libraries and DSL’s
- Ideal for concurrent and distributed systems

Scala: Strong Typing and Concise Syntax

- Strong typing like Java
- Compile time checks
  - Better modularity via strongly typed interfaces
  - Easier maintenance: types make code easier to understand
- Concise syntax like Python
  - Type inference. Compiler infers most types that had to be explicit in Java
  - Powerful syntax that avoid much of the boilerplate of Java code (see next slide)
- Best of both worlds: safety of strong typing with conciseness (like Python)

Scala Case Class

- **Java version**

```
class User {
   private String name;
   private Int age;
   public User(String name, Int age) {
      this.name = name; this.age = age;
   }
   public getAge() { return age; }
   public setAge(Int age) { this.age = age;}
}
User joe = new User(“Joe”, 30);
```

- **Scala version**

```
case class User(name:String, var age:Int)
val joe = User(“Joe”, 30)
```

Functional Scala

- Anonymous functions.
  `(a:Int,b:Int) => a+b`
- Functions that take and return other functions
- Rarely need variables or loops
- Immutable collections: Seq[T], Map[K,V], …
  - Works well with concurrent or distributed systems
  - Natural for functional programming
- Functional collection operations (a small sample)
  - map, flatMap, reduce, …
  - filter, groupBy, sortBy, take, drop, …

Scala on the JVM

- Can use any of the rich set of Java libraries
- Can use use Java tools
- Can write code that is a mix of Java and Scala (for example when moving from Java to Scala)

Typesafe Scala Components

- Scala Compiler (includes REPL)
- Scala Standard Libraries
- SBT - Scala Build Tool
- Play - scaleable web applications
- Scala JS - compiles Scala to JavaScript
- Akka - for parallel and distributed computation
- Akka HTTP - high performance asynchronous TCP/ HTTP library
- Spark - Typesafe also supports Spark
- Slick - for SQL database access
- ConductR - Scala deployment/devops tool

Reactive Scala

- Approach to building more robust fault-tolerant systems that can handle vast amounts of data reliably
- Typesafe and Scala are at the center of the reactive movement
- Reactive Manifesto
  - **Responsive**: responds in timely manner if possible
  - **Resilient**: stays responsive in face of failure
  - **Elastic**: can scale up and down in response to load
  - **Message driven**: key to architecture

Scala Availability and Support

- Open Source
- Language promoted by Scala Center
- Lightbend provides support. Founded my Martin Odersky who designed Scala and Jonas Bonar who designed Akka
- IDEs: Intellij IDEA and Eclipse
- Libraries: lots now and more every day
- Major Scala users: LinkedIn, Twitter, Coursera, Angies List, Goldman Sachs, IBM, Verizon, Xerox, Sony, FourSquare
- In Seattle: Maana, Socrata, Whitepages, Allen Institute, Starbucks, Microsoft, Amazon
- Major systems written in Scala: Spark, Kafka
- Hiring: Scala jobs attract top developers and offer high salaries

Scala Programming Models

- Object-oriented
- **Functional**
- Concurrent and Distributed (Akka)

Scala References

- API, view Scala library
- Book: Odersky: Programming Scala 3rd Ed
- Tools and Libraries
- Coursera: Functional Programming in Scala
- Scala Exercises

Seattle Scala Meetup

- http://www.meetup.com/Seattle-Scala-User-Group/
- 2nd Tuesday of each month doors and food 6 talk 6:30

SBT

Scala SBT Commands and Demo

- Command completion, History
- help, tasks, settings
- clean
- reload (whenever build.sbt is changed)
- compile, ~compile
- run, test, ~test, runOnly, testOnly
- doc
- package, assembly
- exit (^D)
- console, ~console (Scala REPL)

SBT Console Scala REPL and Demo

- Tab completion
- History
- :help
- :quit (^D)
- :javap

ivy2

- finder:Go To Folder: ~/.ivy2
- jars: code, source, doc
- build.sbt
  - Dependencies (libraryDependencies)
  - Resolvers
    - local
    - maven
    - typesafe

Intellij

Intellij Demo

- Project Structure
- Completion
- Add import
- Find type
- Examine Source
- Format
- Refactor
- Debugger

Break

Scala the Simple Parts

- Scala the Simple Parts

Break

Simple Scala with the REPL

Assignment 1

Reading 1

- Odersky Chapters 1,2,3 or
- Wampler Chapters 1,2,3

Tar Gzip

- Unpack files from Canvas
  - `gunzip filename.tar.gz`
  - `tar -xvf filename.tar`
- Pack src directory to submit solution to Canvas
  - `tar -cvzf src.tar.gz src`
- **`Important: Send only the src directory!!!`**

Assign 1: Easter

- Goals
  - start using SBT and IDE
  - write a simple Scala program
  - run the unit test to make sure it works
- USNO Easter: History, Rules to compute
- Look at code skeleton and unit test
- ??? throws exception, replace with your code
- must pass unit test
  - specs2 Unit Specification

End