Big Data Systems Engineering with Scala

Robin Hillyard, M.A (Oxon), Ph.D (Cantab)
Associate Teaching Professor

What makes a good language?

- Versatility
  - Is it a general-purpose language?
  - Or is it designed for a specific niche or platform?
- Design—is it well designed?
  - you will not run into arbitrary limitations;
  - syntax is intuitive;
  - once it's compiled, it should run;
  - it's easy to extend.
- Suitability for concurrency/parallel processing
  - This is extremely important for modern computers.

Language choice—the practicalities

- In-demand
  - If you learn it, can you expect to find a good job?
  - Is it an academic-only language?
  - Does it run everywhere?
  - Is it dead?

Why Scala?

- Scala meets all of the requirements of a good language described in the previous slides.
- However, there are many new and exciting languages competing for your interest.
- Is Scala sufficiently good and interesting?
  - We will go into this question in some detail.
  - For now:
    - 13 programming languages defining the future of coding (Nov, 2015)
    - 14 languages help you block entire classes of bug (Sep, 2016)

About me

- B.A/M.A Engineering Science (Oxford) (1st Class) 1973
- Ph.D. Computer Science (Cambridge) 1978
- Worked in:
  - Computer-aided design (“Solid Modeling”, “Surface Modeling”) (Pascal, Algol68)
  - Artificial Intelligence/Machine Learning/NLP (Lisp, Java, etc.)
  - Object-relational database design (C)
  - Document Management (C, C++, OPL, Perl, Java)
  - Financial (Java)
  - eCommerce (Java, ColdFusion, Javascript, Groovy)
  - Healthcare
    - Privacy, security, crypto, anonymization (Java)
    - Reactive programming (Java, Scala)
    - Big-data analysis with Hadoop/Spark/GraphX/ElasticSearch (Pig, Java, Scala)

About me (contd.)

- Recent activities:
  - Big Data consulting
  - Specializing in Spark
  - This is my eighth time teaching this class (which I created)
- Slack team*: https://csye7200-xxx.slack.com
- Blog: http://scalaprof.blogspot.com
- LinkedIn: https://www.linkedin.com/in/robinhillyard
- Twitter: @Phasm1d

* best way to contact me regarding the class

About me (contd.)

- 1968: wrote my first program
  - solve ***sech(x)=x*** (in Fortran)
  - it worked first time.
- 1969: wrote my first driver (for a plotter) as well as first use of a “personal computer”
- 1972: wrote my first debugger (for Assembly language).
- 1983: wrote my first object-relational database.
- 1984: wrote my first unit-test runner.
- 1994: wrote my first Java program.
- 2012: wrote my first Scala program.

About you…

- Backgrounds?
- Programming classes?
- Programming jobs?
- O-O?
- F-P?
- Java? Java8?
- Big Data?
- Functional Programming?

About the class: *Big Data Systems Engineering with Scala*

- Why *Big Data*?
  - Until fairly recently, ***most*** business-oriented computer software was developed for one of these purposes:
    - personal applications (document preparation, spreadsheets, presentations, email, etc.)
    - database applications to support internal business needs
    - interactive systems for business ("eCommerce")
    - analysis of finite, usually static, datasets ("science")
  - But now, the internet can provide essentially infinite, streaming datasets with a huge potential for data-mining, inference, etc. Collectively, these vast resources are known as "Big Data."

Big Data Systems Engineering with Scala

- Why ***Systems Engineering***?
  - This class aims to provide a ***practical*** approach to dealing with Big Data:
    - performant
    - testable
    - versatile
    - elegant
- Will our solutions always be the shortest? The fastest possible? The most mathematically sound?
  - no—but they will be tested and effective

Big Data Systems Engineering with Scala

- Why ***with Scala***?
  - The Big Data world is increasingly turning to Scala as the language of choice*:
    - Functional Programming ->
    - performance, provability, testability, parallelizable
    - Spark is written in Scala

* four years ago, this was undoubtedly true;
now, this may not be so true, but Scala is still important.

Academic Honesty

- You are *expected* to uphold the highest standards of academic integrity: **do not submit another's work as your own!!**
- Sources of information:
  - My lecture notes, code samples, blog (OK except for mid-term exam)
  - Recommended text, your notes (OK even for exams)
  - The internet (OK for clarification, background, etc. but **not** OK for copying code samples*)
  - plagiarism will never be tolerated

* with the exception of the term project (where all properly attributed code is allowable)

Is this class for YOU?

- Why you ***should not*** attend this class:
  - This is ***not*** an easy class;
  - You cannot just coast through without doing the work;
  - You will be learning some significant new concepts;
  - Scala is not Java (or Python);
  - Unless you have some familiarity with Haskell, Clojure, or other functional language you will probably struggle at first;
  - I give A grades purely on ***merit***: there is never a fixed number of each grade.

Is this class for YOU? (2)

- Why you ***should*** attend this class:
  - You will be challenged;
  - You will learn a lot about ***good programming techniques,*** most of which are applicable to ***any*** language;
  - You will be more employable—even if you can't find a Scala job right away;
  - You will have a lot of fun:
    - programming in a functional way is very satisfying;
    - especially when it comes to the term project