

COSC282

BIG DATA ANALYTICS

FALL 2015

LECTURE 2 - SEP 9

HOW WAS YOUR WEEKEND?



1. Read and Post on Piazza
2. Installed JDK & Spark
3. Submit Your Assignment 1 (Due Today 11:59pm, Blackboard)
4. Office hours: Tue 1-2, 6-7:30, Wed 6-7:30PM

[HTTPS://PIAZZA.COM/GEORGETOWN/FALL2015/COSC282/HOME](https://piazza.com/georgetown/fall2015/cosc282/home)

PIAZZA

The screenshot shows a web browser with multiple tabs open, including 'COSC 282', 'https://s3...', 'cs-sys-1.u...', '"Hello Wor...', 'Washingto...', and 'Inbox'. The address bar shows the URL <https://piazza.com/class/idw3rs25awv7fd>. The page header includes the 'piazza' logo and navigation links for 'COSC 282', 'Q & A', 'Resources', 'Statistics', and 'Manage Class'. Below the header, there are tabs for 'polls', 'hw1', 'hw2', 'hw3', 'hw4', 'logistics', and 'other'. The main content area is divided into two columns. The left column contains a sidebar with filters for 'Unread', 'Updated', 'Unresolved', and 'Following'. It also features a 'New Post' button and a search bar. The 'PINNED' section includes a post titled 'Search for Teammates!' dated 8/28/15. The 'TODAY' section lists three posts: 'Scala readings' (4:38PM), 'submission of assignment 1' (1:05PM), and 'Office hours' (12:57PM). The 'YESTERDAY' section is partially visible. The right column features a 'Class at a Glance' summary box with the following status: 'no unread posts' (checked), 'no unanswered questions' (checked), and '1 unresolved followups' (warning icon). Below this is a 'Student Enrollment' box showing '31 enrolled'. At the bottom right, there is a section titled 'Dear Professors:' with text about a new service launched last fall.

Class at a Glance *Updated*

- ✓ no unread posts
- ✓ no unanswered questions
- ! 1 unresolved followups

Student Enrollment

31 enrolled

Dear Professors:

Last Fall, we launched a new service ca
connected students with other students
and with potential employers. Many of t

SCALA CRASH COURSE

- “Stairs” in Italian
- Why Scala?
 - Spark is written in Scala originally
- Quite fun

A comprehensive step-by-step guide

Programming in

Scala

Second Edition



Updated for Scala 2.8

artima

Martin Odersky
Lex Spoon
Bill Venners

WHAT DO YOU KNOW ABOUT **SCALA**?



THINGS ABOUT SCALA

- Object-Oriented
 - classes can be extended
 - every value is an object
- Functional
 - every function is a value
 - so, every function is an object
- Statically typed
 - type inference saves us efforts to write explicit types
- Interoperates with Java
 - can use any Java class and can be called by Java

WHAT DO YOU WANT TO LEARN ABOUT **SCALA**?



SCALA FOR TODAY

- **Syntax**
 - **define variables**
 - **define functions**
 - **closures**
 - **collection**
 - **control structures**
- **Compile using sbt**
- **A show-and-tell**

A comprehensive step-by-step guide

Programming in

Scala

Second Edition



Updated for Scala 2.8

artima

Martin Odersky
Lex Spoon
Bill Venners

LET'S WORK IN SCALA
SHELL

VARIABLES

- `var x: Int = 5`
- `var x = 5 // type inferred`
- `val myState = "free fall" // read-only, final, value cannot be changed`

DATA TYPES

- **Byte** **8 bit signed value. Range from -128 to 127**
- **Short** **16 bit signed value. Range -32768 to 32767**
- **Int** **32 bit signed value. Range -2147483648 to 2147483647**
- **Long** **64 bit signed value. -9223372036854775808 to 9223372036854775807**
- **Float** **32 bit IEEE 754 single-precision float**
- **Double** **64 bit IEEE 754 double-precision float**
- **Char** **16 bit unsigned Unicode character. Range from U+0000 to U+FFFF**
- **String** **A sequence of Chars**
- **Boolean** **Either the literal true or the literal false**

All the data types listed above are objects. There are no primitive types like in Java. This means that you can call methods on an Int, Long, etc.

FUNCTIONS

“def” starts a function definition

function name

parameter list in parentheses

function’s result type

equals sign

```
def max(x: Int, y: Int): Int = {  
    if (x > y)  
        x  
    else  
        y  
}
```

function body
in curly braces

FUNCTIONS

first letter in function name needs to be lower case

- `def square(x: Int): Int = x*x`
- `def square(x: Int): Int = { x*x }`
- `def announce(text: String) = { println(text) }`
- `def addTwo(x: Int): Int = x + 2`

CLOSURES

- a function, whose return value depends on the value of one or more variables declared outside this function
 - `var factor = 3`
 - `def multiplier = (i:Int) => i * factor` // factor is the variable outside this function

we could also say

`var multiplier = (i:Int) => i * factor`

- What will be the output for
 - `multiplier(1)`
 - `multiplier(2)`

CLOSURES

- `multiplier(1) // 3`
- `multiplier(2) // 6`

CONTROL STRUCTURES

```
var x = 30;
```

```
if (x<20) {
```

```
    println ("free fall");
```

```
} else{
```

```
    println ("parachute");
```

```
}
```

← Semicolon is optional

CONTROL STRUCTURES

```
var x = 30;
```

```
var myState = "free fall";
```

```
while (x>0) {
```

```
    if (x< 15) { myState = "parachute"} ;
```

```
    println (myState);
```

```
    x = x - 1;
```

```
}
```

CONTROL STRUCTURES

- As such there is no built-in `break` nor `continue` statements available in Scala
- well, for the later versions of Scala 2.8, there are objects defined for the purpose.

COLLECTIONS IN SCALA

- Scala collections have **mutable** and **immutable** collections.
- A mutable collection can be updated or extended in place.
 - This means you can change, add, or remove elements of a collection
- Immutable collections, by contrast, never change.

COMMON COLLECTIONS

- Mutable
 - Map, HashMap, ListMap, MutableList, LinkedList, Seq
- Immutable
 - List, Array, Vector, Set, String, Seq

PROCESSING COLLECTIONS

- `val list = List(1, 2, 3)`
- `list.foreach(x => println(x))` // prints 1, 2, 3
- `list.foreach(println)` // same
- `list.map(x => x + 2)` // returns a new `List(3, 4, 5)`
- `list.map(_ + 2)` // same
- `list.filter(x => x % 2 == 1)` // returns a new `List(1, 3)`
- `list.filter(_ % 2 == 1)` // same



**KEEP
CALM
AND
EXERCISE**

WHAT DO YOU GET?

```
> import scala.collection.mutable
> val map = mutable.Map.empty[String, Int]
> map("hello") = 1
> map("there") = 2
> map
> map.foreach(println)
> map("hello")
> map.filter(map("hello")==1)
> map.filter(_==Pair("hello",1))
> map.filter(_==Pair("there",2))
> map.filter(_==Pair("there",1))
```

```
> import scala.collection.mutable
> val map = mutable.Map.empty[String, Int]
> map("hello") = 1
> map("there") = 2
> map
> map.foreach(println)

> map("hello")
// res25: Int = 1

> map.filter(map("hello")==1)
// <console>:14: error: type mismatch;
// found    : Boolean
// required: ((String, Int)) => Boolean
// map.filter(map("hello")==1)

> map.filter(_==Pair("hello",1))
// res27: scala.collection.mutable.Map[String,Int] = Map(hello -> 1)

> map.filter(_==Pair("there",2))
// res29: scala.collection.mutable.Map[String,Int] = Map(there -> 2)

> map.filter(_==Pair("there",1))
// res30: scala.collection.mutable.Map[String,Int] = Map()
```

PROCESSING COLLECTIONS

- `map(f: T => U): Seq[U]` // Each element is result of `f`
- `flatMap(f: T => Seq[U]): Seq[U]` // One to many map
- `filter(f: T => Boolean): Seq[T]` // Keep elements passing `f`
- `exists(f: T => Boolean): Boolean` // True if one element passes `f`
- `forall(f: T => Boolean): Boolean` // True if all elements pass

LET'S WORK IN SCRIPTS
- USING SBT TO COMPILE

STEP 1: SETUP SBT

From the directory that you copy from the spark thumb drive

- Go to `spark_disk/sbt`
 - NOT `spark_disk/spark/sbt`
- `chmod a+x sbt`
- `mkdir -p src/main/scala`

STEP 2: WRITE YOUR HELLOWORLD.SCALA

- Create a file called `HelloWorld.scala` using your text editor
 - in Mac, you could use *emacs*, *vim* or *nano*; You might want to open another Terminal window to work on the editor while keep the `./sbt` directory active in one Terminal
 - in Windows, you could use *NotePad* or *WordPad* as the text editor
- Put the following line in your file

```
object HelloWorld {  
  def main(args: Array [String]) = println ("Hi, cosc 282!")  
}
```
- `mv HelloWorld.scala src/main/scala/.`
 - Note: Make sure there is only one `.scala` file in `src/main/scala/`. We will talk about how to build a package later. As for now, just compile one file

STEP 3: COMPILE AND RUN

- go back to the sbt directory
 - `cd ./spark_disk/sbt`
- type `./sbt`
- from the sbt prompt, type "run"

`> run`

- keep typing "run", the program will be compiled and run again

`> run`

YOU SHOULD GET SOMETHING LIKE THIS

```
cs-ad-d9fy11:sbt gh243$ ./sbt
[info] Set current project to sbt (in build file:/Users/gh243/Desktop/Teaching/cosc282/spark_disk/sbt/)
> run
[info] Running HelloWorld
Hi, cosc 282!
[success] Total time: 0 s, completed Sep 9, 2015 1:55:01 PM
> run
[info] Running HelloWorld
Hi, cosc 282!
[success] Total time: 0 s, completed Sep 9, 2015 1:55:03 PM
> run
[info] Running HelloWorld
Hi, cosc 282!
[success] Total time: 0 s, completed Sep 9, 2015 1:55:05 PM
> █
```

FUN TIME

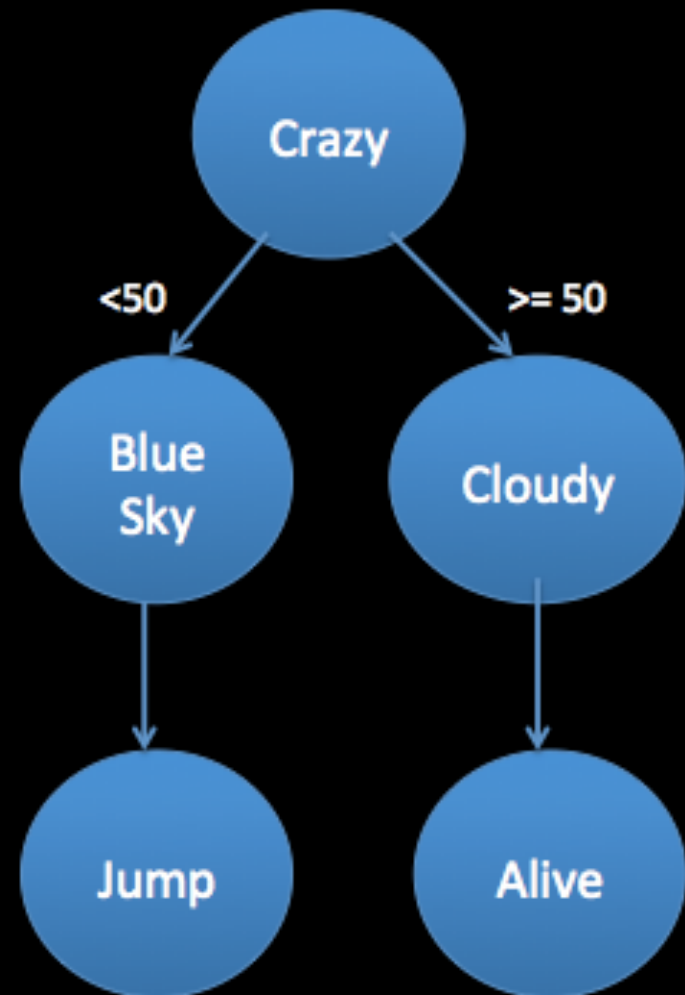


SHOW-AND-TELL

```
val states = List ("blue sky", "crazy", "jump", "free fall", "parachute", "alive", "dead",  
"cloudy")  
var myState = ""  
println ("I have a friend " )  
println ("Sometimes she is " + states(1) )  
print ("When it is ")
```

```
val r = scala.util.Random  
var chance = r.nextInt(100)
```

```
if (chance >= 50) {  
    myState = states(7)  
    println(myState)  
    println ("She is " + states(5) )  
}  
else {  
    myState = states(0)  
    println(myState)  
    println("She " + states(2))  
}
```

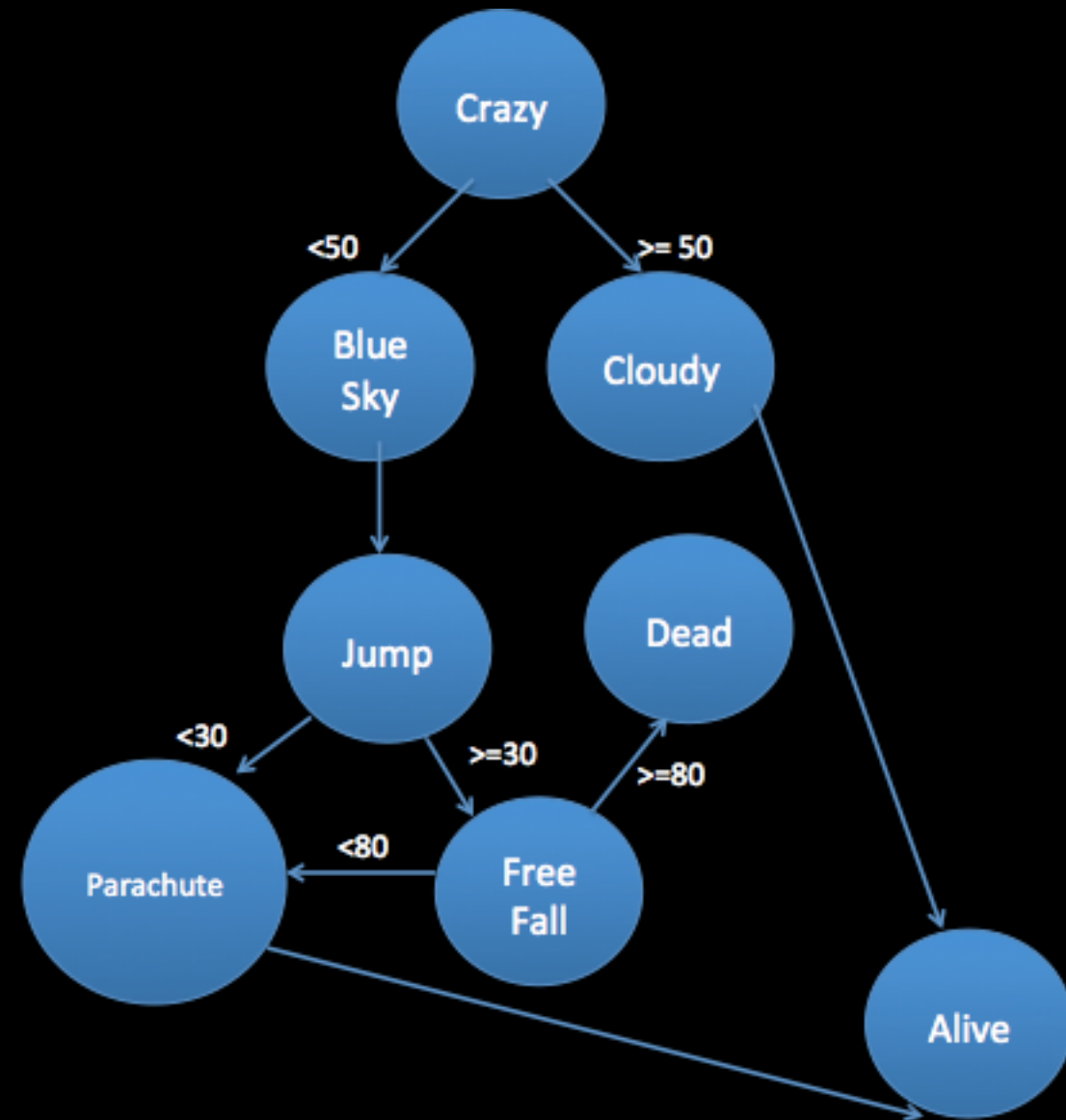


THE STORY LOOKS LIKE:

```
cs-ad-d9fy11:sbt gh243$ ./sbt
[info] Set current project to sbt (in build file:/Users/gh243/Desktop/Teaching/cosc282/spark_disk/sbt/)
> run
[info] Compiling 1 Scala source to /Users/gh243/Desktop/Teaching/cosc282/spark_disk/sbt/target/scala-2.10/classes...
[info] Running HelloWorld
I have a friend
Sometimes she is crazy
When it is cloudy
She is alive
[success] Total time: 2 s, completed Sep 9, 2015 2:08:24 PM
> run
[info] Running HelloWorld
I have a friend
Sometimes she is crazy
When it is cloudy
She is alive
[success] Total time: 0 s, completed Sep 9, 2015 2:08:29 PM
> run
[info] Running HelloWorld
I have a friend
Sometimes she is crazy
When it is cloudy
She is alive
[success] Total time: 0 s, completed Sep 9, 2015 2:08:30 PM
> run
[info] Running HelloWorld
I have a friend
Sometimes she is crazy
When it is blue sky
She jump
[success] Total time: 0 s, completed Sep 9, 2015 2:08:31 PM
> run
[info] Running HelloWorld
I have a friend
Sometimes she is crazy
When it is blue sky
She jump
[success] Total time: 0 s, completed Sep 9, 2015 2:08:33 PM
> run
```

ASSIGNMENT 2 - FINISH THE STORY

- **Using control structures**
- **(Bonus) Using processes for collections**
- **What to submit:**
 - **your codes**
 - **screencapture of at least 4 random runs of results**
 - **Due: Next Wed 9/16, 11:59pm**



HERE COMES THE REAL
SHOW-AND-TELL

COURSE VIDEOS

- a few videos are put on piazza. they are the demos and assignment related procedures that we have shown in class. Please check them out