Appendix B: Java/Scala[2]

Feature	Java	Scala
Public Class	public class	class
Loops	for(Type it : c){...}	c.foreach {...}
Lists	List list = asList(1,2,3);	val list = List(1,2,3)
Maps	Map m = ...; m.put(x,y);	val m = Map(x -> y)
Function Def.	void method(Type t) {}	def method(t: Type) = {}
Mutable Value	Type t	var t: Type
Immutable Value	final Type t	val t: Type
Null safety	(x == null ? null : x.y)	for (a <- Option(x)) yield a.y
Null replacement	(x == null ? "y" : x)	Option(x) getOrElse "y"
Sort	Collections.sort(list)	list.sort(_ < _)
Wildcard import	import java.util.*;	import scala.collection._
Var-args	(String... args)	(args: String*)
Type parameters	Class<T>	Class[T]
Concurrency	Fork/Join	Akka

No Java Analog

Feature	Scala
Default closure arg.	_ *(underscore is positionally matched)*
Default value	def method(t:String = "yes")
Add method to object	*use* Trait
Auto-delegate	*use* Trait
Extension methods	implicit class
Rename import	import scala.collection.{Vector => Vect}

[2]Version 1.3 of this cheat sheet.

Null, Nil, Etc.

Type	Description
Null	A `Trait` with one instance, null, similar to Java's null.
Nil	Represents an empty `List` of zero length.
Nothing	A `Trait` that is a subtype of everything. There are no instances of it.
None	`None` signifies no result. Option has two subclasses: `Some` and `None`.
Unit	Type to use on a method that does not return a value