Scala collections

1. Review
2. Work environment (2)
3. Collections in Scala
4. `filter`

Review

- values are of a specific *type* or *class*
- use the keyword `val` to *assign* (name) a value
- *functions* produce (new) values
- some functions take *parameters* (or *arguments*)
- use the keyword `import` to bring material in

Example

```
val hmt = "HMT project"
val expanded = hmt.replaceAll("HMT", "Homer Multitext")
```

Work environment

- experiment in `sbt console`
- paste/save in a `.sc` file
- `:load` files from console

Collections

Group of objects of the *same type*

Families

- `Sets`: unordered; each item unique
- `Sequences`: ordered; values may not be unique.

Ordered collections

- `Array` (identical in Java)
- `Vector` (useful general-purpose collection)

Individual elements

Numeric index, starting with 0

```
val words = Vector("Now", "is", "the", "time")
val firstWord = words(0)
```

Good functions

- `size`
- `distinct`
- `contains`
- `isEmpty`, `nonEmpty`

Examples

```
val sententia = "Now is the time for all good people to come to the
val words = sententia.split(" ")
val uniqueWords = words.distinct
```

filter

`filter`

- select *part* of a collection
- one parameter: a boolean test
- result: a *new* collection of the same type
- special shorthand symbol: _ (“fill in the blank”) stands for value of each element in collection

Example

Test if a collection is empty:

```
val sententia = "Now is the time for all good people to come to the
val words = sententia.split(" ")
val longWords = words.filter(_.size > 5)
longWords.size
longWords.nonEmpty
_
```