Immutable and Mutable Fields

Lesson Objectives

- After completing this lesson, you should be able to:
  - Describe the difference between mutable and immutable fields
  - Create fields in Scala classes
  - Describe the difference between class parameters and fields
  - Outline how to promote class parameters to fields

What is a Field?

- A value encapsulated within an instance of a class
  - Represents the state of an instance, and therefore of an application at a given time
  - Is accessible to the outside world, unless specified otherwise

Fields versus Parameters

- Parameters are passed to a class and are only visible within a class
- Fields exist in the body of the class, and are accessible to outsiders

Immutable Fields

```
scala> class Hello {
     |   val message: String = "Hello"
     | }
defined class Hello

scala> (new Hello).message
res0: String = Hello
```

Typesafe

Mutable Fields

```
scala> class Hello {
     |   var message: String = "Hello"
     | }
defined class Hello

scala> val hello = new Hello
hello: Hello = Hello@3617a35c

scala> hello.message = "Hello, world!"
hello.message: String = Hello, world!
```

Typesafe

Immutable or Mutable?

- Immutable fields cannot be changed and are therefore “threadsafe” in a multithreaded environment, such as the JVM
- Mutable fields can be useful, but require diligence to ensure that multiple threads cannot update the field at the same time

Use Immutable By Default

- It is easier to reason about immutable fields and classes that only contain immutable fields
- Scala makes all class parameters immutable by default

Specify Types

- Scala has “type inference”
- It is a good habit to be specific about types anyway

```
scala> class Hello {
     |   val message = "Hello"
     | }
defined class Hello

scala> (new Hello).message
res0: String = Hello
```

Typesafe

Promoting Class Parameters

- If you want to make a parameter passed to a class constructor into a publicly visible field, add the val keyword in front of it
- The Scala compiler will generate an accessor method for you, and other class instances can now ask for the current state of the promoted field

Promoting Class Parameters

```
scala> class Hello(val message: String)
defined class Hello

scala> val hello = new Hello("Hello, world!")
hello: Hello = Hello@59d941d7

scala> hello.message
res0: String = Hello, world!
```

Typesafe

Lesson Summary

- Having completing this lesson, you should be able to:
  - Describe the difference between mutable and immutable fields
  - Create fields in Scala classes
  - Describe the difference between class parameters and fields
  - Outline how to promote class parameters to fields