Type Classes in Scala

What? Why? How?

Daniel Reigada

IST - Programming Languages

August 15, 2018

Questions I would like answer

1. What is type class?
2. Why should you use type classes?
3. How to use type class in Scala?

What is a type class?

From Wikipedia:
“A type system construct that supports ad hoc polymorphism.”

What is a type class?

From Wikipedia:
“A type system construct that supports ad hoc polymorphism.”

But classes in object oriented languages are just that!

What is a type class?

From Wikipedia:
“A type system construct that supports ad hoc polymorphism.”

But classes in object oriented languages are just that!

What’s so different about type classes?

What is a type class?

```haskell
class Writable a where
  write :: a -> String
```

```haskell
data Name = Name String String

instance Writable Name where
  write (Name firstName lastName) =
    firstName ++ " " ++ lastName
```

```java
interface Writable {
  public String write();
}
```

```java
class Name implements Writable {
  String firstName, lastName;

  public String write(){
    return firstName + " " + lastName;
  }
}
```

Why type classes? - Modularity

Why should a list be concerned in implementing RandomAccess, Cloneable and Serializable?

```
public class ArrayList<E> extends AbstractList<E>
    implements List<E>, RandomAccess, Cloneable, java.io.Serializable {
    /* ... */
  }
```

Why type classes? - Extending existing types

```haskell
data MyType = MyType String Int

-- somewhere else, far from the definition:
instance Show MyType where
  show (MyType s i) = s ++ ", " ++ (show i)
```

Why type classes? - Generic code

We would like add to only add numbers of the same type:

```
interface Num {
  Num add (Num i);
}

class Int implements Num {
  public Int add(Num i) { /*...*/ }

  // does not override add:
  // public Int add(Int i) { /*...*/ }
}
```

Why type classes? - Generic code

Solution in java:

```
interface Num<Impl> {
  Impl add (Impl i);
}

class Int implements Num<Int> {
  public Int add(Int i) { /*...*/ }
}
```

Why type classes? - Generic code

Solution in Haskell:

```
class Numero t where
  add :: t -> t -> t

instance Num Int where
  add a b = a + b
```

How to use type classes? (in Scala)

A simple type class:

```scala
trait Show[A] {
  def show(a: A): String
}

val intInstance = new Show[Int] {
  override def show(a: Int) = s"Int: " + a
}

intInstance.show(123)
// res0: String = Int: 123
```

Aside - Scala implicits

```scala
def method(str: String)(implicit strImplicit: String) =
  str + strImplicit

method("parameter1")
// error: could not find implicit value for parameter
// strImplicit: String
// method("parameter1")
// ^
```

Aside - Scala implicits

```scala
def method(str: String)(implicit strImplicit: String) =
  str + strImplicit

implicit val string = "implicitString"
method("parameter1")
// res0: String = parameter1implicitString
```

How to use type classes? (in Scala)

```scala
trait Show[A] {
  def show(a: A): String
}

object Show {
  def show[A](a: A)(implicit sh: Show[A]) = sh.show(a)
}

implicit val intInstance = new Show[Int] {
  override def show(a: Int) = s"Int: " + a
}

Show.show(123)
// res0: String = Int: 123
```

How to use type classes? (in Scala)

Using implicit classes:

```
implicit class ShowOps[A](a: A)(implicit sh: Show[A]) {
  def show: String = sh.show(a)
}

123.show
// res0: String = Int: 123
```

How to use type classes? (in Scala)

Live demo/examples