

JACEK KUNICKI

@rucek



TYPE CLASSES

FROM ZERO TO HERO

POLYMORPHISM

POLYMORPHISM

- ▶ task: add a `show()` method to any type
- ▶ OOP approach with inheritance - limited
- ▶ what if we wanted to add to any type?



LIBRARIES

- ▶ cats/Scalaz
- ▶ simulacrum - less boilerplate with @typeclass

REAL LIFE

SCALA COLLECTIONS

```
def sum[B >: A](implicit num: Numeric[B]): B =  
  foldLeft(num.zero)(num.plus)
```

JSON

```
def toJson(implicit writer: JsonWriter[T]): JsValue =  
  writer.write(any)
```

```
def toJson[T](o: T)(implicit tjs: Writes[T]): JsValue
```


FOR COMPREHENSIONS

```
def addOptions[T: Numeric](a: Option[T], b: Option[T]): Option[T] =  
  for {  
    x <- a  
    y <- b  
  } yield x + y
```

```
def addFutures[T: Numeric](a: Future[T], b: Future[T]): Future[T] =  
  for {  
    x <- a  
    y <- b  
  } yield x + y
```

FOR COMPREHENSIONS

```
def add[T: Numeric, FM[_] : FlatMap](a: FM[T], b: FM[T]): FM[T] =  
  for {  
    x <- a  
    y <- b  
  } yield x + y
```

```
add(1.some, 3.some)  
add(Future.successful(1), Future(3))
```

<https://typelevel.org/cats/typeclasses.html>

DOTTY (AKA SCALA 3.0)

- ▶ goal: reduce boilerplate
- ▶ simulacrum annotation macros won't be supported
- ▶ new `scala.TypeClass` marker trait
- ▶ <https://github.com/lampepfl/dotty/pull/4153>

TAKEAWAYS

- ▶ ad-hoc polymorphism, also for existing types
- ▶ powerful DSLs when combined with implicits
- ▶ context bounds vs. implicit parameter



<http://scalatimes.com>

THANK YOU!



[rucek/type-classes-from-zero-to-hero](https://github.com/rucek/type-classes-from-zero-to-hero)



@rucek

