

# Functional programming from first principles in Scala

張瑋修 Walter Chang

@weihsiu / [weihsiu@gmail.com](mailto:weihsiu@gmail.com)

<https://github.com/weihsiu/fpffp>



# Scala Taiwan

[Scala Taiwan Discord server](#)

[Scala Taiwan FB group](#)

[Scala Taiwan Meetup](#)

# Agenda

- Why functional programming?
- Constraints
- Refresher
- Functor
- Applicative
- Traverse
- Monad
- IO
- Q&A

# Why functional programming?

- Multi-core CPU / multi-thread programming
- Immutable (less bugs)
- Composable

# Constraints

- No `var`, just `val`
- No `scala.collection.mutable.*`
- No Exception
- No functions returning `Unit`

# Refresher

- `List` is a type constructor
- `List[Int]` is a type
- `trait List[A]` is a generic trait which takes a type( `A` ) as its type parameter
- `trait Functor[F[_]]` is a generic trait which takes a type constructor( `F[_]` ) as its type parameter

# Functor

```
trait Functor[F[_]]:  
  def map[A, B](fa: F[A])(f: A => B): F[B]
```

- Transform(map) something in a context to something else in the same context
- Composable
- Examples
  - Map over `List[A]`
  - Map over `Option[A]`
  - Map over `List[Option[A]]`

# Applicative

```
trait Applicative[F[_]] extends Functor[F]:  
  def pure[A](x: A): F[A]  
  def ap[A, B](ff: F[A => B])(fa: F[A]): F[B]
```

- Lift(transform) functions to a new context
- Composable
- Examples
  - Reuse an already-written function in a new context
  - Enable parallel computation



# Traverse

- Swap nested contexts
- Examples
  - Turn a `List[Option[A]]` to `Option[List[A]]`
    - `List(Some(1), Some(2), Some(3)) => Some(List(1, 2, 3))`
    - `List(Some(1), None, Some(3)) => None`

# Monad

```
trait Monad[F[_]] extends Applicative[F]:  
  def flatMap[A, B](fa: F[A])(f: A => F[B]): F[B]
```

- Sequential computation
- Happy path programming
- Examples
  - Sequence of calls to functions that may fail

# IO

- Computation effects
- Examples
  - User interactions

## Q&A

That's all and thank you for your attention

<https://github.com/weihsiu/fpffp>

