# Monads and why the even matter in Java (and many other languages)

daherb (they/them)

Bornhack 2024 Lightning Talks 2020721

# (Probably) unnhelpful definitions

- a monoid in the category of functors (from values to computations, with monadic composition as a binary operator in the monoid and ‘unit‘ as identity in the monad)
- a box where you can put stuff
- something something haskell IO

# (Hopefully) more helpful

- A design pattern
- A data structure with two functions and three laws governing the functions

# Monads and monadic laws

Monad functions:

- Put something into a monad data structure (bind or unit)
- Apply an operation to something stored within a monad data structure

Monad laws: tl; dr

# The Maybe monad

Motivation

```
Result result = getResult();
if (result != null) {
  int newValue = result.getValue() + 42;
}
```

# Java

```
/*
  Returns an Optional with the specified present
  non-null value.
*/
static <T> Optional<T> of(T value);

/*
  If a value is present, apply the provided
  Optional-bearing mapping function to it, return
  that result, otherwise return an empty Optional.
*/
<U> Optional<U>
  map(Function<? super T,? extends U> mapper);
```

# Examples

```
Optional<Result> result = test.getResult();
Optional<Integer> newValue = result.map((x) -> x.getResult(
```

# Other monads

Haskell: Lists

But: Java lists cannot be modified element-wise

Java: Streams

```
/*
  Returns a sequential Stream containing a single element.
*/
static <T> Stream<T> of(T t);
/*
  Returns a stream consisting of the results of
  replacing each element of this stream with the
  contents of a mapped stream produced by applying
  the provided mapping function to each element.
*/
<R> Stream<R> flatMap(Function<? super T,? extends Stream<
```

# Take away message

- Monads are a design pattern providing a more pipeline style of data processing
- You don't have to know too much about monads to be able to use them
- Alternatives: applicative functor (simpler laws)