Literate Functional Java

Slides and Code Available

The slides and code are available at
https://github.com/sethkraut/presentation-functional-literate-java

Or find it at http://tech.sethkraut.com

About Me

Literate Functional

Use Java’s functional capabilities to create easy to read code

What is a Function

Accepts a value and returns a different one

HashMap is like a function

Specifying a Function

Lambdas

```
    (n) -> n + 1
```

Method references

```
    System.out::println
```

Functional Composition

```
<V> Function<T,V> andThen(Function<? super R,? extends V> after)
```

Functional Interfaces

`Function<T,R>`	`R apply(T t)`
`Consumer<V>`	`void accept(V v)`
`Supplier<T>`	`T get()`
`Runnable`	`void run()`
`Predicate<T>`	`Boolean test(T t)`

Magic

You can use any method that matches the pattern.

System.out::println is a Consumer

Math::abs is a Function

BigDecimal::negate is a Function

Function Method References

Math::abs is defined as `static double abs(double)`

BigDecimal::negate is defined as `BigDecimal negate()`

The compiler accepts both as a function.

Where do we use them

Streams

Multiple element structure

`filter`

`map`

`flatMap`

`collect`

Optional

Single value or nothing

`map`

`flatMap`

`orElse`

Simple Examples

Higher Order Functions

Functions that return functions

```
Function<String, String> append(String suffix) {
    return s -> s + suffix;
}
```

Literate Functional

We can combine all of this to create very readable code.

Use only method references and higher order functions.

Toolbox

There are some really useful method references.

```
System.out::println
```

```
Objects::nonNull
```

```
String.class::cast
```

```
String.class::isInstance
```

Primitive Streams

IntStream

DoubleStream

Primitive Functions

Operator = Function with same types

IntUnaryOperator = Function<int,int>

Demo

Guidelines

Prioritize readability

Prefer Method References and Higher Order Functions

Collect them into

- Domain objects
- DSL classes

Custom Interfaces

Try it

Try this on some code sometime

Always ask if it could be better