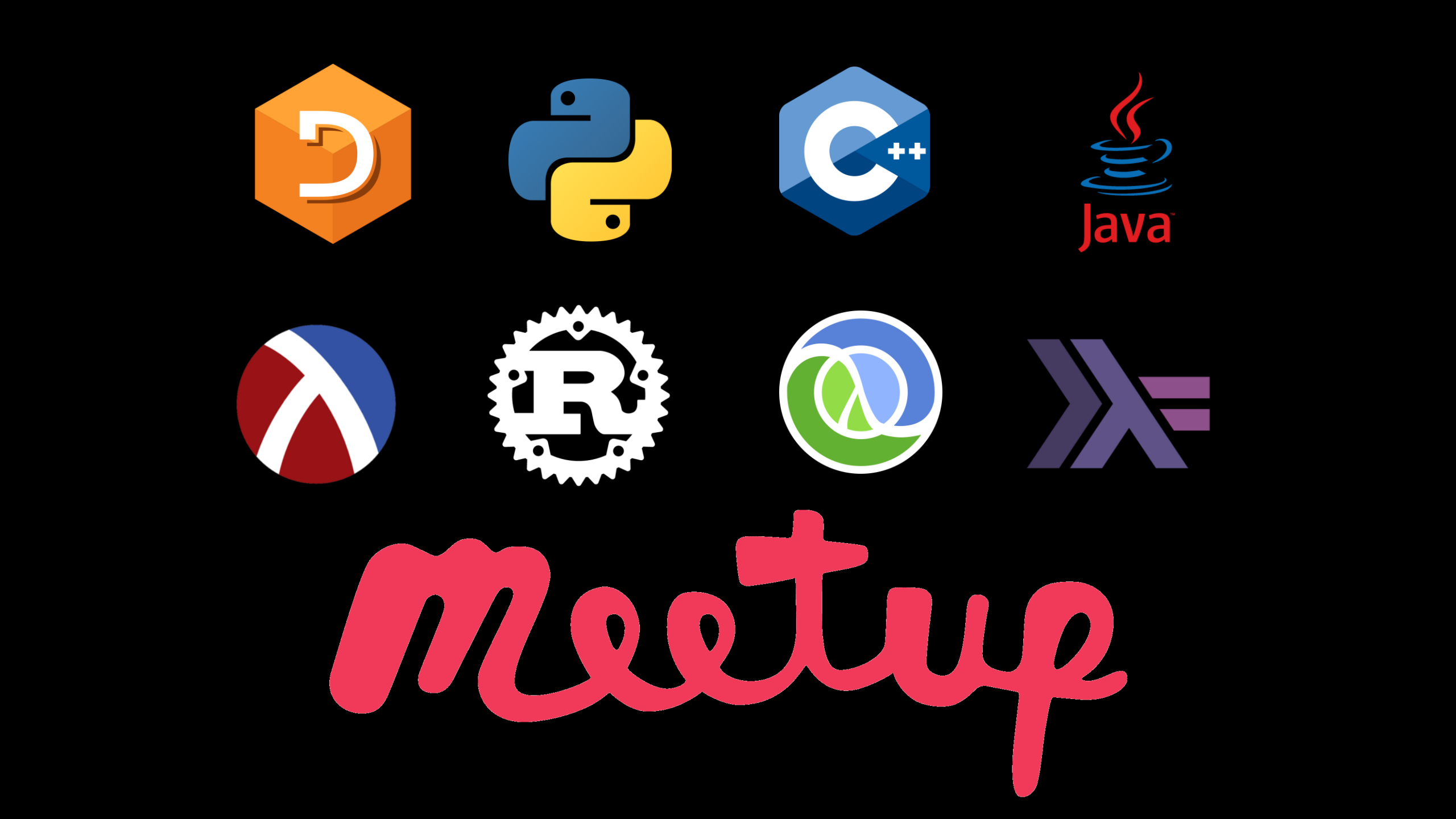
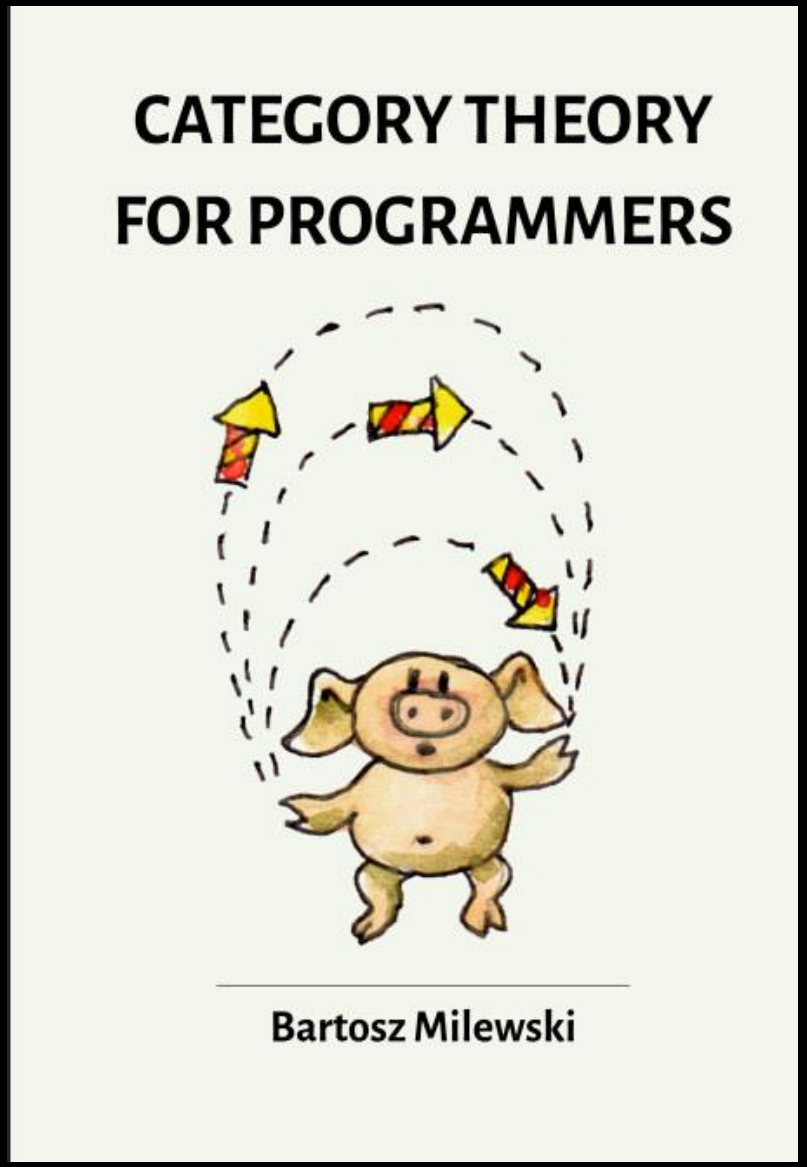
Friendly Environment Policy

Berlin Code of Conduct

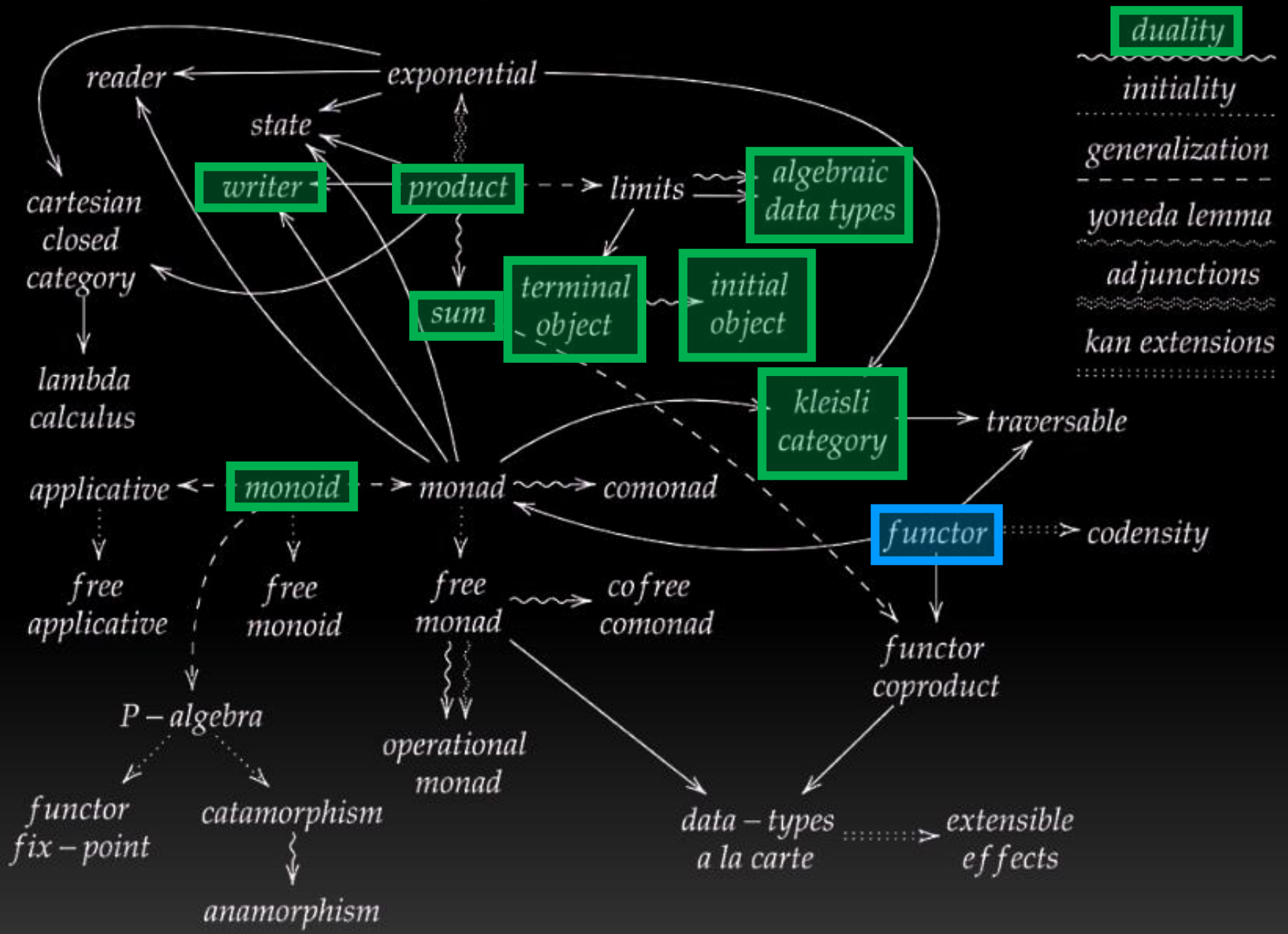

Java
meetup



Category Theory for Programmers

Chapter 7: Functors

The Tools for Thought



At the risk of sounding like a broken record, I will say this about functors: A functor is a very simple but powerful idea. Category theory is just full of those simple but powerful ideas. A functor is a mapping between categories. Given two categories, $\mathbf{C}$ and $\mathbf{D}$, a functor F maps objects in $\mathbf{C}$ to objects in $\mathbf{D}$ — it's a function on objects.

```haskell
class Functor f where
    fmap :: (a -> b) -> f a -> f b

instance Functor f where ...
```

```
fmap :: (a -> b) -> [a] -> [b]
fmap = map
```

```haskell
fmap :: (a -> b) -> Maybe a -> Maybe b
fmap _ Nothing  = Nothing
fmap f (Just x) = Just (f x)
```

```
fmap :: (a -> b) -> (r -> a) -> (r -> b)
fmap = (.)
```

```haskell
class Functor f where
    fmap :: (a -> b) -> f a -> f b

instance Functor f where ...

fmap :: (a -> b) -> Maybe a -> Maybe b
fmap _ Nothing  = Nothing
fmap f (Just x) = Just (f x)

fmap :: (a -> b) -> [a] -> [b]
fmap = map

fmap :: (a -> b) -> (r -> a) -> (r -> b)
fmap = (.)
```

January 22, 2018

Functors Done Quick!

Suppose we're writing some code to deal with bank accounts. Most of our code will refer to these using a proper data type. But less refined parts of our code might use a tuple with the same information instead. We would want a conversion function to go between them. Here's a simple example:

```
data BankAccount = BankAccount
  { bankName :: String
  , ownerName :: String
  , accountBalance :: Double
  }
```

```haskell
instance Functor Vector where
  fmap = Data.Vector.map

instance Functor Set where
  fmap = Data.Set.map

instance Functor (Either a) where
  fmap _ (Left a) = Left a
  fmap f (Right x) = Right (f x)
```

3. Implement the reader functor in your second favorite language (the first being Haskell, of course).

```
auto reader_fmap = [](auto f, auto g) {
    return [&] (auto r) { return g(f(r)); };
};
```

```cpp
auto reader_fmap = [](auto f, auto g) {
    return [&] (auto r) { return g(f(r)); };
};

auto string_to_float = [](auto s) { return std::stof(s); };
auto float_to_int    = [](auto f) { return static_cast<int>(f); };
auto string_to_int   = reader_fmap(string_to_float, float_to_int);
```

```cpp
auto reader_fmap = [](auto f, auto g) {
    return [&] (auto r) { return g(f(r)); };
};

auto string_to_float = [](auto s) { return std::stof(s); };
auto float_to_int    = [](auto f) { return static_cast<int>(f); };
auto string_to_int   = reader_fmap(string_to_float, float_to_int);

for (auto s : { "1.23", "42.42", "17.29"}) {
    fmt::print("{}\n", string_to_float(s)); // 1.23, 42.42, 17.29
    fmt::print("{}\n", string_to_int(s));   // 1     42     17
}
```

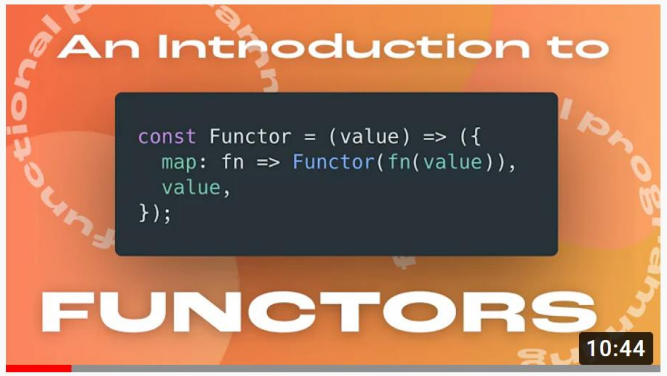
```cpp
template <typename A, typename B, typename R>
struct reader {

    using AtoB = B(A);
    using RtoB = B(R);
    using RtoA = A(R);

    auto fmap(RtoA f, AtoB g) {
        return [=] (R r) { return g(f(r)); };
    }

};
```

```
auto string_to_int   =
    reader<float, int, std::string>{}
        .fmap(string_to_float, float_to_int);
```

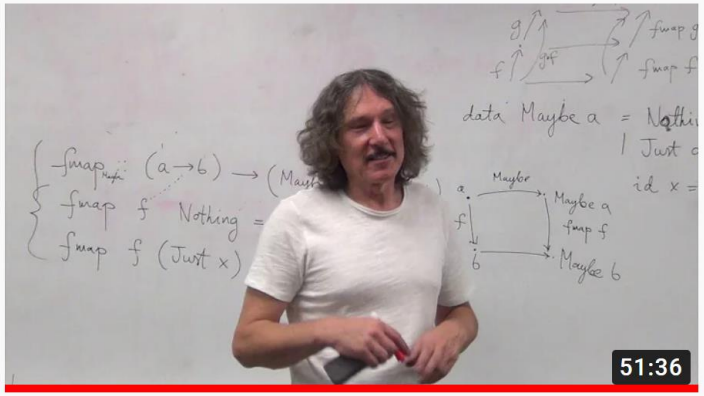


An Introduction to Functors in JavaScript: The Basics in 9 Minutes with Examples

355 views • 7 months ago

Ijemma Onwuzulike

A **functor** is a data structure that can be mapped over using a custom mapping interface. In order words, a **functor** asks as a ...

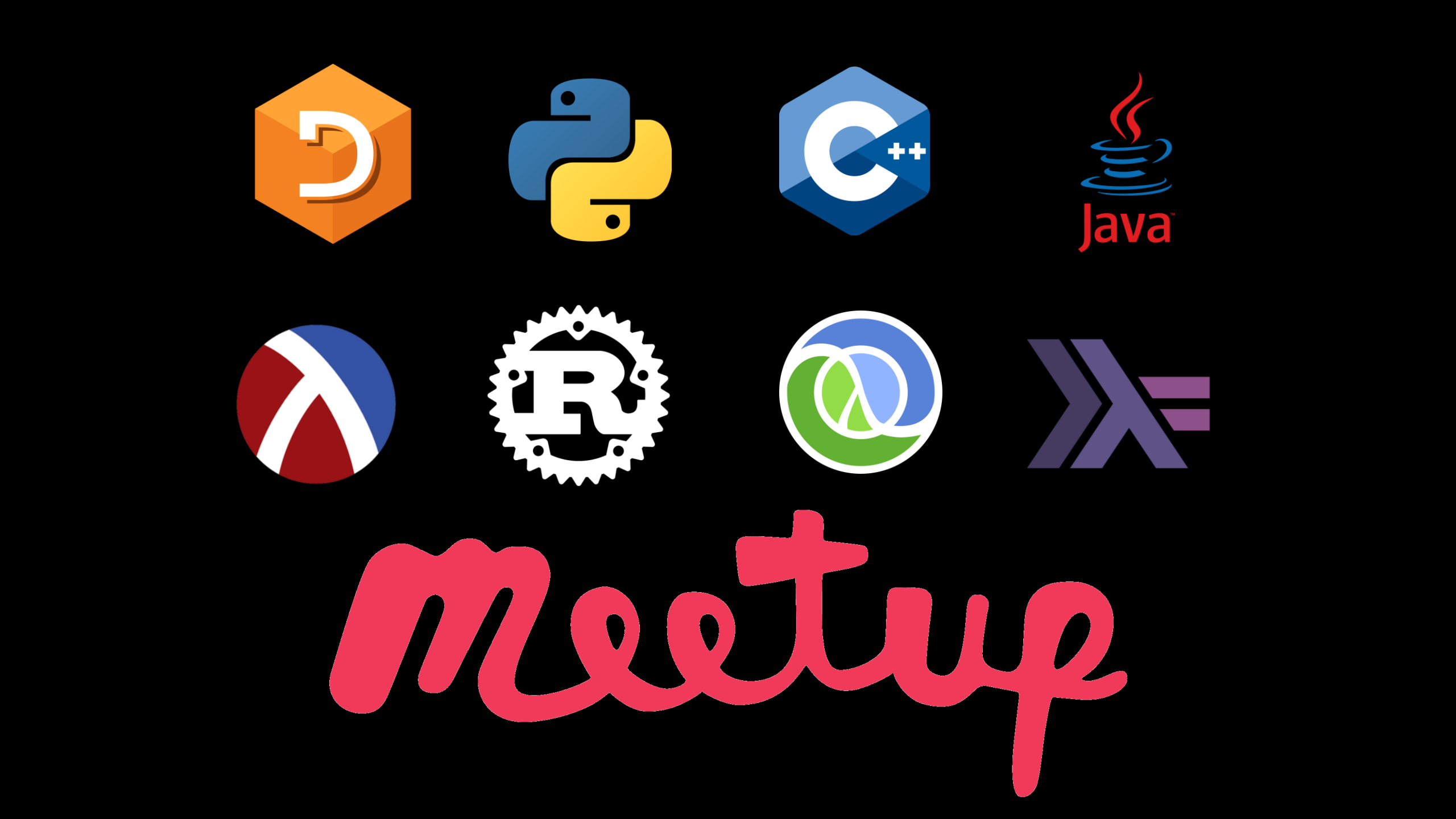


Category Theory 6.2: Functors in programming

27K views • 4 years ago

Bartosz Milewski

Functors in programming.


Java
meetup