```
Link = class:
    fields first, rest
    length():
        1 + rest.length()
    end
end
```

"Length" interface

Dynamic Dispatch

```
Empty = class:
    length(): 0 end
end
lst = Link(4, Link(7, Empty()))
lst.length()
```

```
fun Link(first, rest):
    [dict:
      "length": lam():
        1 + rest.get("length")()
      end]
end
```

```
fun Empty():
    [dict: "length",
        lam(): 0 end]
end
```

o.fld

PROTOTYPE INHERITANCE

SELF

JS

```
lst = { length: lam() ... end }
lst2 = { super: lst,
         fold: lam(f, base): ... end }
lst3 = { super: lst2,
         map: lam(f): ... end }
lst2.fold(...)
lst2.length()
lst3.length()
```

```
fun lookup(o, x):
  cases(option) o.get(x):
    | some(v) => v
    | none =>
      cases(option) o.get("super"):
        | none => error
        | some(s) =>
          lookup(s, x)
      end
  end
end
lookup(lst3, "length")
```

```
LinkWFold = class extends Link:
    fields first, rest
    fold(f, base):
        f(first, rest.fold(f, base))
    end
end
```

```
AddFold = fun ___(super):
    fun LinkWFold(first, rest):
        merge(Link(first, rest), super(first, rest),
              [dict:
                  "fold",
                  lam(f, base): ...end])
    end
end
```

Mixin : Class → Class

```
LinkWFold = AddFold(Link)
AddMap(AddFold(Link))
```

```
EmptyWFold = class extends Empty:
    fold(f, base): base end
end
```

```
fun ___(super):
    fun EmptyWFold():
        merge(Empty(), super(),
              [dict:
                  "fold",
                  lam(f, base):
                      base
                  end])
```

CLASS EXTENSION INHERITANCE