

let swift: Race?

Jack Nutting
@jacknutting
jacknutting@mac.com



thoughtbot

let swift: Race?

```
struct Sun {
  let swift: Race?
  let strong: Battle?
  let wise: Bread?
  let intelligent: Riches?
  let knowledgeable: Favor?

  init() {
    swift = TimeAndChance() ? Race() : .None
    strong = TimeAndChance() ? Battle() : .None
    wise = TimeAndChance() ? Bread() : .None
    intelligent = TimeAndChance() ? Riches() : .None
    knowledgeable = TimeAndChance() ? Favor() : .None
  }
}
```

let swift: Race?



`[[somePeople like:ObjectiveC] evenIn:2015]`

```
[[[[[receiver method] method] method]  
    method] method] method];
```

```
receiver.method().method().method()  
    .method().method().method()
```

Static

**fixed, stable, steady,
consistent**

**motionless, frozen,
inert, lifeless**

enum

```
enum Button {  
    case Rectangular(bounds: CGRect, title: String, font: UIFont)  
    case RoundedCorners(bounds: CGRect, title: String, font: UIFont,  
        radius: CGFloat)  
}
```

```
extension Button {
  func draw() {
    switch self {
    case let .Rectangular(bounds, title, font):
      drawRectangleButton(bounds, title, font)
    case let .RoundedCorners(bounds, title, font, radius):
      drawRoundedRectangleButton(bounds, title, font, radius)
    }
  }
}
```

```
extension Button {  
    func buttonOfPreferredSize() -> CGSize {  
        switch self {  
        case let .Rectangular(bounds, title, font):  
            return preferredSizeForRectangleButton(bounds,  
                title, font)  
        case let .RoundedCorners(bounds, title, font, radius):  
            return preferredSizeForRoundedRectangleButton(bounds,  
                title, font, radius)  
        }  
    }  
}
```

- Boilerplate
- Eternal Vigilance
- My Way or the Highway

If a huge chunk of your enum's code is just switching on the type, maybe you should just use a set of classes.

Using `struct` instead of `class`

Marking classes as `final`

Marking methods as `final`

Future developers will want the flexibility to extend your system in ways you can't foresee.

“Premature optimization is the
root of all evil.”

–Sir Tony Hoare

C++

“All of C++ is premature optimization.”

–Jack Nutting

Functional Programming is not a
silver bullet

[http://chris.eidhof.nl/
posts/lenses-in-swift.html](http://chris.eidhof.nl/posts/lenses-in-swift.html)

[https://www.destroyallsoftware.com/
talks/boundaries](https://www.destroyallsoftware.com/talks/boundaries)

Conclusions

Sometimes enums aren't the answer.

Static has drawbacks.

All of C++ is premature optimization.

FP is not a silver bullet.

let swift: Race?

Jack Nutting
@jacknutting
jacknutting@mac.com

