SweeterSwift

Jason

Growth Session #24 - May 16-17 2019

What is this project about

SweeterSwift is Swift code formatter.

```
if name == "Swift" {
    print("Hello World!")
}
```

Abstract Syntax Tree (AST)

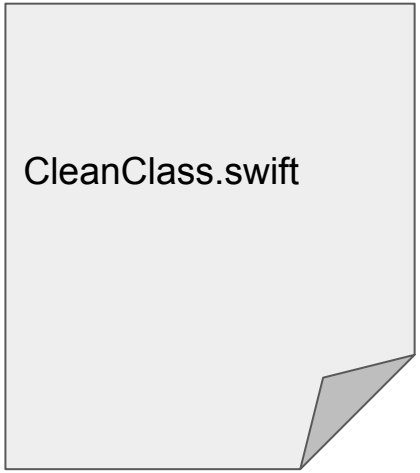
We write code...

Intro

```
// swift-tools-version:4.0
import PackageDescription

let package = Package(
    name: "Flint",
    products: [
        .executable(
            name: "flint",
            targets: ["Flint"]),
    ],
    dependencies: [
        .package(url: "https://github.com/flintbox/Bouncer", from: "0.1.3"),
        .package(url: "https://github.com/flintbox/Motor", from: "0.1.2"),
        .package(url: "https://github.com/flintbox/Work", from: "0.1.1"),
        .package(url: "https://github.com/flintbox/ANSIEscapeCode", from: "0.1.1"),
        .package(url: "https://github.com/jasonnam/PathFinder", .branch("develop")),
        .package(url: "https://github.com/jpsim/Yams.git", from: "1.0.0"),
    ],
    targets: [
        .target(
            name: "Flint",
            dependencies: ["Bouncer", "Motor", "Work", "ANSIEscapeCode", "PathFinder", "Yams"]),
    ]
)
```

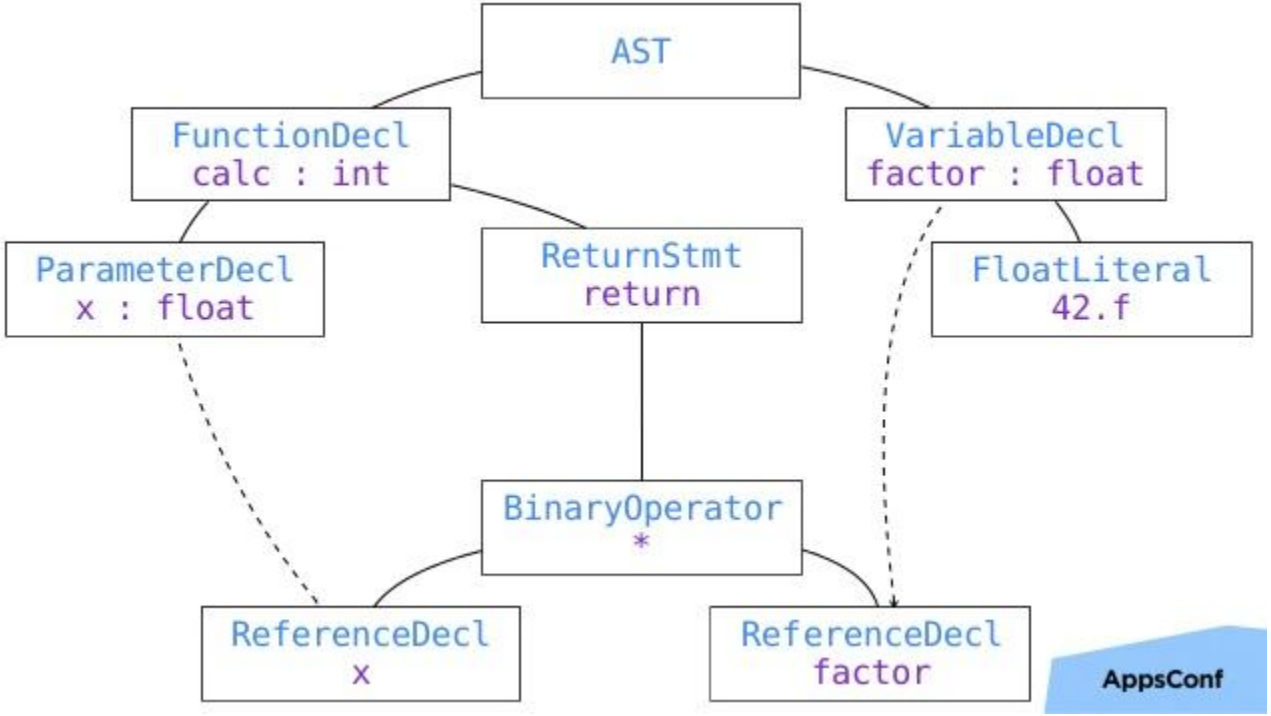
Compile the code



Compiler

Intro

Abstract Syntax Tree



What does code formatter do?

Read and parse code

Transverse AST and apply rules

Write output to the same or different file. If needed just print to standard output

What are the benefits?

We can stop wasting time with things does not matter much.

New Lines

```
import UIKit

@UIApplicationMain
class AppDelegate: UIResponder, UIApplicationDelegate {
    var window: UIWindow?
}
```

New Lines

```
import UIKit

@UIApplicationMain
class AppDelegate: UIResponder, UIApplicationDelegate {

    var window: UIWindow?
}
```

New Lines

```
import UIKit

@UIApplicationMain
class AppDelegate: UIResponder, UIApplicationDelegate {

    var window: UIWindow?

}
```

Intro

Snapshot Testing

```
<a
  className="normal"
  href="http://www.facebook.com"
  onMouseEnter={[Function]}
  onMouseLeave={[Function]}
>
  Facebook
</a>
```

=

```
<a onMouseEnter={[Function]}
onMouseLeave={[Function]}
className="normal"
  href="http://www.facebook.com"
  >Facebook</a>
```

SweeterSwift is a Swift code formatter.

1. Uses SwiftSyntax
2. Good documentation
3. Production ready interfaces

SwiftSyntax

SwiftSyntax is a set of Swift bindings for the libSyntax library. It allows for Swift tools to parse, inspect, generate, and transform Swift source code.

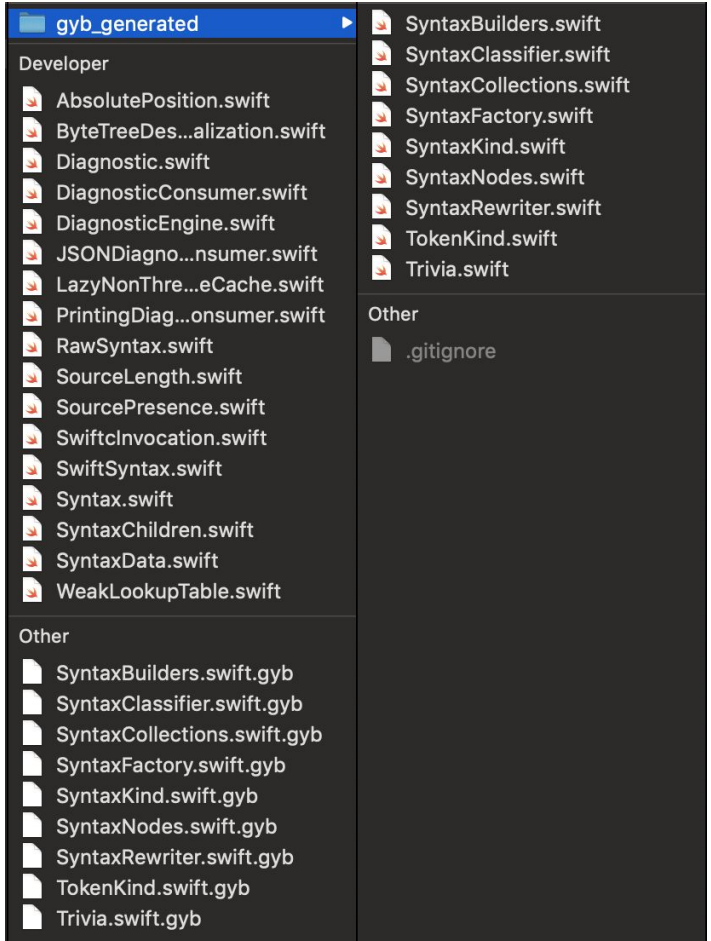
Integrating SwiftSyntax

```
// swift-tools-version:5.0

import PackageDescription

let package = Package(
    name: "Sweeter",
    products: [
        .library(
            name: "SweeterKit",
            targets: ["SweeterKit"]),
    ],
    dependencies: [
        .package(url: "https://github.com/apple/swift-syntax.git", .exact("0.50000.0")),
    ],
    targets: [
        .target(
            name: "SweeterKit",
            dependencies: ["SwiftSyntax"]),
        .testTarget(
            name: "SweeterKitTests",
            dependencies: ["SweeterKit"]),
    ]
)
```

Achievement and Progress



SwiftSyntax API

SyntaxTreeParser

SyntaxRewriter

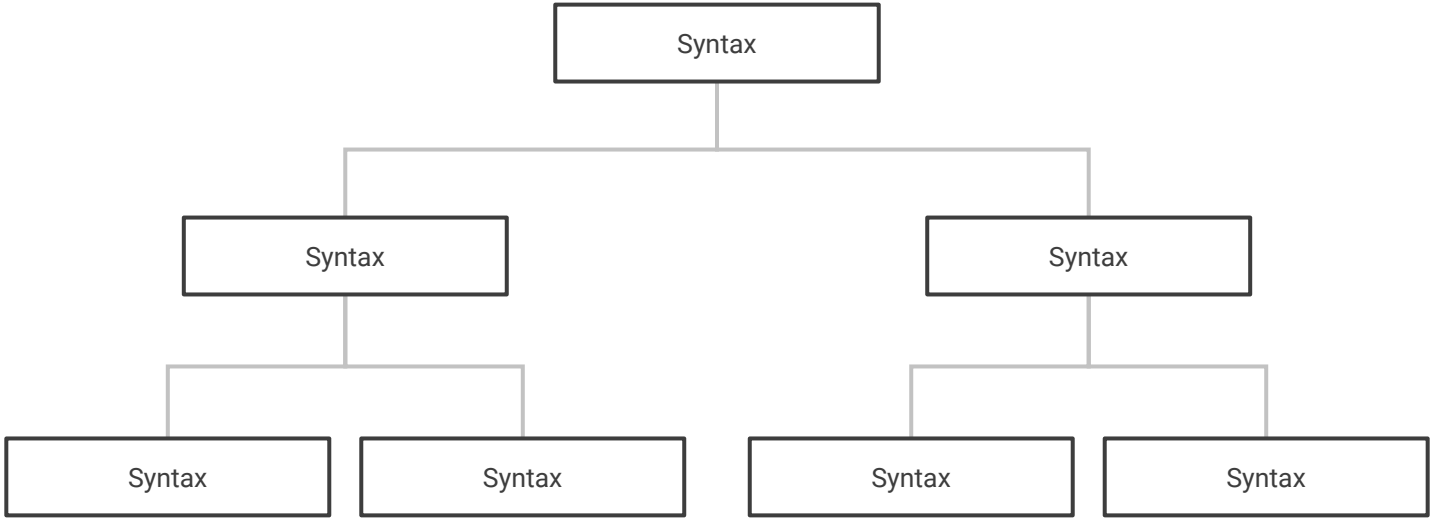
Syntax.write

SyntaxTreeParser

Has one static function

```
public static func parse(_ url: URL, swiftcURL: URL? = nil) throws
```

A composition of syntax objects



SyntaxRewriter

Visitor

```
override func visit(_ token: TokenSyntax) -> Syntax

open func visit(_ node: UnknownStmtSyntax) -> StmtSyntax
```

main.swift

```swift
import Foundation
import SwiftSyntax

class AddOneToIntegerLiterals: SyntaxRewriter {

    override func visit(_ token: TokenSyntax) -> Syntax {
        // Only transform integer literals.
        guard case .integerLiteral(let text) = token.tokenKind else {
            return token
        }

        // Remove underscores from the original text.
        let integerText = String(text.filter { ("0"..."9").contains($0) })

        // Parse out the integer.
        let int = Int(integerText)!

        // Return a new integer literal token with `int + 1` as its text.
        return token.withKind(.integerLiteral("\(int + 1)"))
    }
}

let syntaxTree = try! SyntaxTreeParser.parse(URL(string: "Code.swift")!)
let formattedCode = AddOneToIntegerLiterals().visit(syntaxTree)
print(formattedCode)
```

Integer literal with underscore

```
1   let x = 2
2   let y = 3_000
3
```

Integer literal with underscore

```
➜  cli ./sweeter
let x = 3
let y = 3001

➜  cli
```

Next Steps

What you plan to do next

Planning for features

Architecturing
- Frameworks to use
- SweeterKit

What kind of rules should SweeterSwift have?

Implementation

Write code and unit tests

CI/CD

Release

README

Documentation

Package managers
- Brew
- Mint
- Swift Package Manager

Writing software is like doing an artwork. There are a lot of personal preferences reflected in the code.

Thanks!

Contact Nimble

nimblehq.co

hello@nimblehq.co

Bangkok

399 Interchange 21 Sukhumvit Road, Unit #2402-03, Klong Toei, Wattana, Bangkok 10110, Thailand

Singapore

28C Stanley St, Singapore 068737

Hong Kong

20th Floor, Central Tower

28 Queen's Road, Central, Hong Kong

nimble