Custom Swift Formatter

A takeaway from iOS Conf SG 2019

Introduction

Current State of the Art

SwiftLint

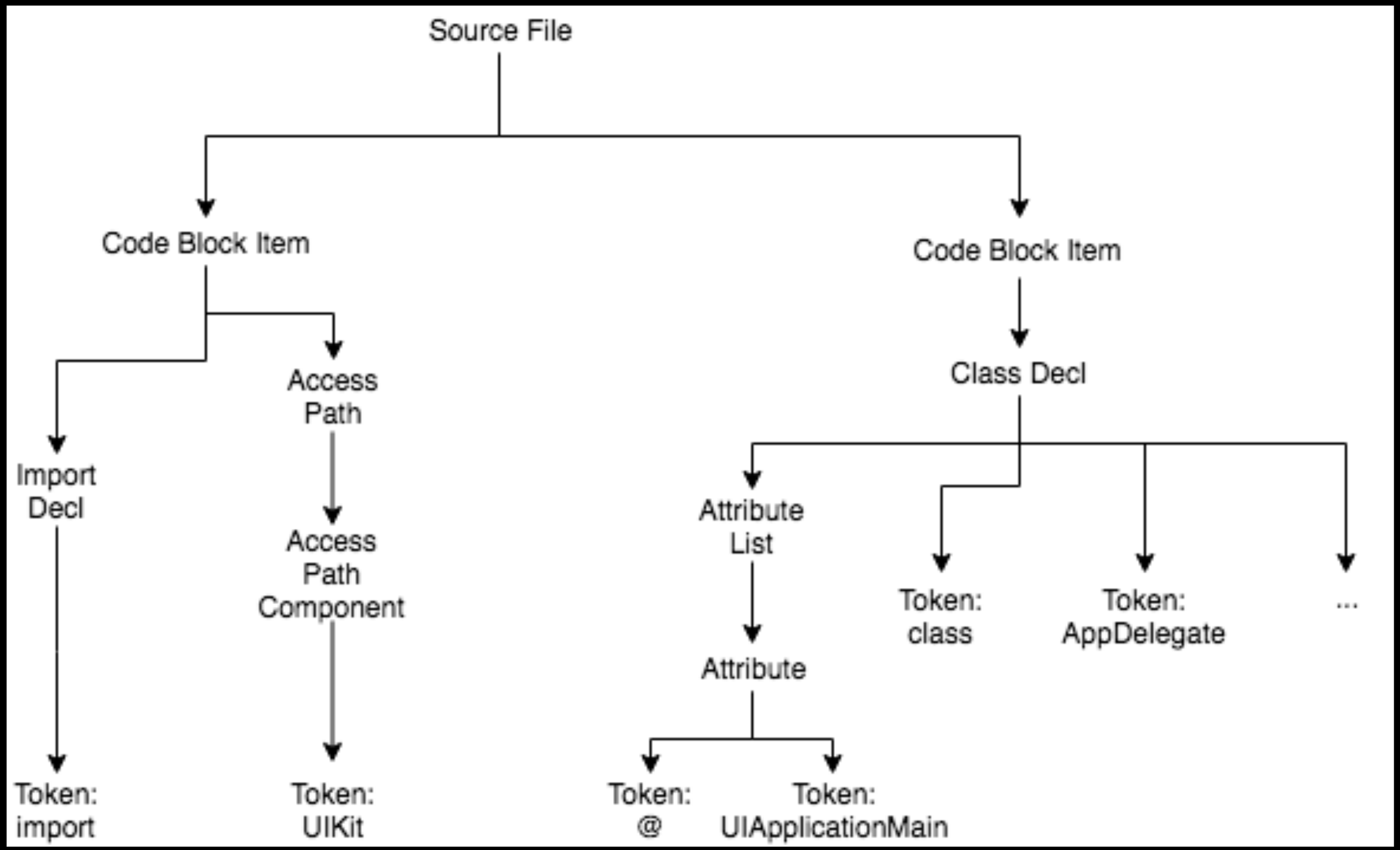
SourceKit

and SourceKitten

lib/Syntax

- Structured editing
- Granular parsing

SwiftSyntax


Source File
Code Block Item
Code Block Item
Import Decl
Access Path
Class Decl
Access Path Component
Attribute List
Token: class
Token: AppDelegate
...
Attribute
Token: import
Token: UIKit
Token: @
Token: UIApplicationMain

Abstract Syntax Tree

SwiftRewriter

Potentials

- Assisting code review
- Pre-review automatic editing
- CI integration

Challenges

Nov 22nd, 2018 | *Comments* | *RSS Feed*

Evaluating SwiftSyntax for use in SwiftLint

tl;dr; Implementing SwiftLint using SwiftSyntax instead of SourceKitten would make it run over 20x slower 😭

Swift Syntax and Structured Editing Library

Welcome to lib/Syntax!

This library implements data structures and algorithms for dealing with Swift syntax, striving to be safe, correct, and intuitive to use. The library emphasizes immutable, thread-safe data structures, full-fidelity representation of source, and facilities for *structured editing*.

What is structured editing? It's an editing strategy that is keenly aware of the *structure* of source code, not necessarily its *representation* (i.e. characters or bytes). This can be achieved at different granularities: replacing an identifier, changing a call to global function to a method call, or indenting and formatting an entire source file based on declarative rules. These kinds of diverse operations are critical to the Swift Migrator, which is the immediate client for this library, now developed in the open. Along with that, the library will also provide infrastructure for a first-class `swift-format` tool.

Eventually, the goal of this library is to represent Swift syntax in all of the compiler. Currently, lib/AST structures don't make a very clear distinction between syntactic and semantic information. Long term, we hope to achieve the following based on work here:

- Adoption throughout the compiler
- Clear separation of syntactic and semantic information
- Greater stability with immutable data structures
- Lower high-water memory use due to reference counting without the need for leak-forever memory contexts
- Incremental re-parsing
- Incremental, lazier re-type-checking, helped by separating syntactic information

This library is a work in progress and should be expected to be in a molten state for some time. Don't integrate this into other areas of the compiler or use it for anything serious just now.

You can read more about the status of the library's implementation at the Syntax Status Page. More information about opportunities to get involved to come.

36 lines (94 sloc) | 4.86 KB

Raw | Blame | History

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.

> Note: SwiftSyntax is still in development, and the API is not guaranteed to be stable. It's subject to change without warning.

Usage

Add this repository to the `Package.swift` manifest of your project:

```
// swift-tools-version:4.2
import PackageDescription

let package = Package(
  name: "MyTool",
  dependencies: [
    .package(url: "https://github.com/apple/swift-syntax.git", .exact("<#Specify Release tag#>")),
```

Let’s discuss!