LLVM C API

How to use it with Swift

whoami

- iOS Apps Developer
- Compiler Hobbyist
- Internet User:

  https://twitter.com/1101_debian
  https://github.com/AlexDenisov
  http://lowlevelbits.org

Outline

- Development Environment
- Hello LLVM World
- Code Generation and Execution
- QA

Motivation

Motivation

Show how to:

Motivation

Show how to:

- Use LLVM tooling

Motivation

Show how to:

- Use LLVM tooling
- Deal with common errors

Motivation

Show how to:

- Use LLVM tooling
- Deal with common errors

=> Attract more developers

Demo

Running SwiftKaleidoscope : SwiftKaleidoscope

SwiftKaleidoscope 〉 SwiftKaleidoscope 〉 Driver.swift 〉 runloop()

- SwiftKaleidoscope
  - SwiftKaleidoscope
    - main.swift
    - ASCIICharacters.swift
    - Lexer.swift
    - Parser.swift
    - Driver.swift
    - CodeGen.swift
    - JIT.swift
    - kstdlib.c
  - Products

```
15
16      consumeToken()
17  runloop:
18      while true {
19          switch getCurrentToken() {
20          case .Def: handleDefinition()
21          case .Extern: handleExtern()
22          case .Character(.Semicolon): consumeToken()
23          case .EOF: break runloop
```

SwiftKaleidoscope

Filter

All Output

Running SwiftKaleidoscope : SwiftKaleidoscope

SwiftKaleidoscope 〉 SwiftKaleidoscope 〉 Driver.swift 〉 runloop()

- SwiftKaleidoscope
  - SwiftKaleidoscope
    - main.swift
    - ASCIICharacters.swift
    - Lexer.swift
    - Parser.swift
    - Driver.swift
    - CodeGen.swift
    - JIT.swift
    - kstdlib.c
  - Products

```
15
16      consumeToken()
17  runloop:
18      while true {
19          switch getCurrentToken() {
20          case .Def: handleDefinition()
21          case .Extern: handleExtern()
22          case .Character(.Semicolon): consumeToken()
23          case .EOF: break runloop
```

SwiftKaleidoscope

```
def square(x) x * x;

define double @square(double %x) {
entry:
  %multemp = fmul double %x, %x
  ret double %multemp
}
```

Filter

All Output

Running SwiftKaleidoscope : SwiftKaleidoscope

SwiftKaleidoscope 〉 SwiftKaleidoscope 〉 Driver.swift 〉 runloop()

- SwiftKaleidoscope
  - SwiftKaleidoscope
    - main.swift
    - ASCIICharacters.swift
    - Lexer.swift
    - Parser.swift
    - Driver.swift
    - CodeGen.swift
    - JIT.swift
    - kstdlib.c
  - Products

```
15
16      consumeToken()
17  runloop:
18      while true {
19          switch getCurrentToken() {
20          case .Def: handleDefinition()
21          case .Extern: handleExtern()
22          case .Character(.Semicolon): consumeToken()
23          case .EOF: break runloop
```

SwiftKaleidoscope

```
def square(x) x * x;

define double @square(double %x) {
entry:
  %multemp = fmul double %x, %x
  ret double %multemp
}

square(square(5));

____
625.0
```

Filter

All Output

Running SwiftKaleidoscope : SwiftKaleidoscope

SwiftKaleidoscope 〉 SwiftKaleidoscope 〉 Driver.swift 〉 runloop()

```
15
16      consumeToken()
17  runloop:
18      while true {
19          switch getCurrentToken() {
20          case .Def: handleDefinition()
21          case .Extern: handleExtern()
22          case .Character(.Semicolon): consumeToken()
23          case .EOF: break runloop
```

SwiftKaleidoscope

```
def fib(x) if x < 3 then 1 else fib(x - 1) + fib(x - 2);
```

- SwiftKaleidoscope
  - SwiftKaleidoscope
    - main.swift
    - ASCIICharacters.swift
    - Lexer.swift
    - Parser.swift
    - Driver.swift
    - CodeGen.swift
    - JIT.swift
    - kstdlib.c
  - Products

Filter

All Output

Running SwiftKaleidoscope : SwiftKaleidoscope

SwiftKaleidoscope 〉 SwiftKaleidoscope 〉 Driver.swift 〉 runloop()

```
15
16      consumeToken()
17  runloop:
18      while true {
19          switch getCurrentToken() {
20          case .Def: handleDefinition()
21          case .Extern: handleExtern()
22          case .Character(.Semicolon): consumeToken()
23          case .EOF: break runloop
```

- SwiftKaleidoscope
  - SwiftKaleidoscope
    - main.swift
    - ASCIICharacters.swift
    - Lexer.swift
    - Parser.swift
    - Driver.swift
    - CodeGen.swift
    - JIT.swift
    - kstdlib.c
  - Products

SwiftKaleidoscope

```
def fib(x) if x < 3 then 1 else fib(x - 1) + fib(x - 2);

define double @fib(double %x) {
entry:
  %cmptemp = fcmp ult double %x, 3.000000e+00
  br i1 %cmptemp, label %continue, label %else

else:                                             ; preds = %entry
  %subtemp = fadd double %x, -1.000000e+00
  %calltmp = call double @fib(double %subtemp)
  %subtemp1 = fadd double %x, -2.000000e+00
  %calltmp2 = call double @fib(double %subtemp1)
  %addtemp = fadd double %calltmp, %calltmp2
  br label %continue

continue:                                         ; preds = %entry,
%else
  %iftmp = phi double [ %addtemp, %else ], [ 1.000000e+00, %entry ]
  ret double %iftmp
}
```

Filter

All Output

Running SwiftKaleidoscope : SwiftKaleidoscope

SwiftKaleidoscope 〉 SwiftKaleidoscope 〉 Driver.swift 〉 runloop()

- SwiftKaleidoscope
  - SwiftKaleidoscope
    - main.swift
    - ASCIICharacters.swift
    - Lexer.swift
    - Parser.swift
    - Driver.swift
    - CodeGen.swift
    - JIT.swift
    - kstdlib.c
  - Products

```
15
16      consumeToken()
17  runloop:
18      while true {
19          switch getCurrentToken() {
20          case .Def: handleDefinition()
21          case .Extern: handleExtern()
22          case .Character(.Semicolon): consumeToken()
23          case .EOF: break runloop
```

SwiftKaleidoscope

```
fib(15);

____
610.0
```

Filter

All Output

Development Environment

Development Environment

- git
- CMake
- GNU/Make
- Modern C++ Compiler

Development Environment

```
$ export LLVM_SOURCE_DIR=$HOME/LLVM
$ export LLVM_BUILD_DIR=$HOME/LLVMBuild
$ export CPU_NUM=`sysctl -n hw.ncpu`
```

Development Environment

```
$ git clone \
      http://llvm.org/git/llvm.git \
      $LLVM_SOURCE_DIR
$ mkdir $LLVM_BUILD_DIR
```

Development Environment

```
$ git clone \
      http://llvm.org/git/llvm.git \
      $LLVM_SOURCE_DIR
$ mkdir $LLVM_BUILD_DIR
$ cd $LLVM_BUILD_DIR
$ cmake $LLVM_SOURCE_DIR
```

Development Environment

```
$ git clone \
      http://llvm.org/git/llvm.git \
      $LLVM_SOURCE_DIR
$ mkdir $LLVM_BUILD_DIR
$ cd $LLVM_BUILD_DIR
$ cmake $LLVM_SOURCE_DIR
$ make LLVMCore -j $CPU_NUM
```

Hello World!

Hello World

```
import LLVM_C
```

Hello World

```
import LLVM_C

let name = "Hello World”

let module = LLVMModuleCreateWithName(name)
```

Hello World

```
import LLVM_C

let name = "Hello World”

let module = LLVMModuleCreateWithName(name)

LLVMDumpModule(module)
```

Hello World

```
import LLVM_C

let name = "Hello World"

let module = LLVMModuleCreateWithName(name)

LLVMDumpModule(module)

LLVMDisposeModule(module)
```

Hello World

```
$ xcrun -sdk macosx \
    swiftc HelloWorld.swift
```

Hello World

```
$ xcrun -sdk macosx \
    swiftc HelloWorld.swift

HelloWorld.swift:1:8: error: no such module
'LLVM_C'
import LLVM_C
       ^
```

Hello World

```
$ xcrun -sdk macosx \
    swiftc HelloWorld.swift \
    -I $LLVM_SOURCE_DIR/include/ \
    -I $LLVM_BUILD_DIR/include/
```

Hello World

```
include/llvm-c/Types.h:17:10: note: while building module
'LLVM_Support_DataTypes' imported from include/llvm-c/Types.h:
17:
#include "llvm/Support/DataTypes.h"
         ^
<module-includes>:1:9: note: in file included from <module-
includes>:1:
#import "Support/DataTypes.h"
        ^
include/llvm/Support/DataTypes.h:57:3: error: "Must #define
__STDC_LIMIT_MACROS before #including Support/DataTypes.h"
# error "Must #define __STDC_LIMIT_MACROS before #including
Support/DataTypes.h"
```

Hello World

```
$ xcrun -sdk macosx \
    swiftc HelloWorld.swift \
    -I $LLVM_SOURCE_DIR/include/ \
    -I $LLVM_BUILD_DIR/include/ \
    -Xcc -D__STDC_CONSTANT_MACROS \
    -Xcc -D__STDC_LIMIT_MACROS
```

Hello World

```
Undefined symbols for architecture x86_64:
  "_LLVMAddFunction", referenced from:
      _main in HelloWorld-f367c0.o
      __TF10HelloWorld14runSumFunctionFTSiSi_Si in HelloWorld-f367c0.o
  "_LLVMAppendBasicBlock", referenced from:
      _main in HelloWorld-f367c0.o
      __TF10HelloWorld14runSumFunctionFTSiSi_Si in HelloWorld-f367c0.o
  "_LLVMBuildAdd", referenced from:
      _main in HelloWorld-f367c0.o
  "_LLVMBuildCall", referenced from:
      __TF10HelloWorld14runSumFunctionFTSiSi_Si in HelloWorld-f367c0.o
  "_LLVMBuildRet", referenced from:
      _main in HelloWorld-f367c0.o
      __TF10HelloWorld14runSumFunctionFTSiSi_Si in HelloWorld-f367c0.o
  "_LLVMConstInt", referenced from:
      __TF10HelloWorld14runSumFunctionFTSiSi_Si in HelloWorld-f367c0.o
  "_LLVMCreateBuilder", referenced from:
      _main in HelloWorld-f367c0.o
      __TF10HelloWorld14runSumFunctionFTSiSi_Si in HelloWorld-f367c0.o
  "_LLVMCreateExecutionEngineForModule", referenced from:
      __TF10HelloWorld14runSumFunctionFTSiSi_Si in HelloWorld-f367c0.o
  "_LLVMDeleteFunction", referenced from:
```

Hello World

```
$ xcrun -sdk macosx \
    swiftc HelloWorld.swift \
    -I $LLVM_SOURCE_DIR/include/ \
    -I $LLVM_BUILD_DIR/include/ \
    -Xcc -D__STDC_CONSTANT_MACROS \
    -Xcc -D__STDC_LIMIT_MACROS \
    -lLLVMCore -lLLVMSupport \
    -L $LLVM_BUILD_DIR/lib
```

Hello World

```
$ xcrun -sdk macosx \
    swiftc HelloWorld.swift \
    -I $LLVM_SOURCE_DIR/include/ \
    -I $LLVM_BUILD_DIR/include/ \
    -Xcc -D__STDC_CONSTANT_MACROS \
    -Xcc -D__STDC_LIMIT_MACROS \
    -lLLVMCore -lLLVMSupport \
    -L $LLVM_BUILD_DIR/lib \
    -lc++ -lncurses
```

Hello World

```
$ ./HelloWorld
```

Hello World

```
$ ./HelloWorld
; ModuleID = 'Hello World'
```

Code Generation

Code Generation

```
int sum(int a, int b) {
    int result = a + b;

    return result;

}
```

Code Generation

```
let int32 = LLVMInt32Type()
```

Code Generation

```
let int32 = LLVMInt32Type()

let returnType = int32
```

Code Generation

```
let int32 = LLVMInt32Type()

let returnType = int32

var paramTypes = UnsafeMutablePointer.alloc(2)

paramTypes.initializeFrom([int32, int32])
```

Code Generation

```
let int32 = LLVMInt32Type()

let returnType = int32

var paramTypes = UnsafeMutablePointer.alloc(2)

paramTypes.initializeFrom([int32, int32])

let functionType =

  LLVMFunctionType(returnType, paramTypes, 2, 0)
```

Code Generation

```
let functionType =
  LLVMFunctionType(returnType, paramTypes, 2, 0)

let sumFunction =

  LLVMAddFunction(module, "sum", functionType)
```

Code Generation

```
$ ./HelloWorld
```

Code Generation

```
$ ./HelloWorld
; ModuleID = 'Hello World’

declare i32 @sum(i32, i32)
```

Code Generation

```
let builder = LLVMCreateBuilder()
```

Code Generation

```
let builder = LLVMCreateBuilder()
let entryBlock =
    LLVMAppendBasicBlock(sumFunction, "entry")
LLVMPositionBuilderAtEnd(builder, entryBlock)
```

Code Generation

```
let a = LLVMGetParam(sumFunction, 0)
let b = LLVMGetParam(sumFunction, 1)
```

Code Generation

```
let a = LLVMGetParam(sumFunction, 0)

let b = LLVMGetParam(sumFunction, 1)

let result = LLVMBuildAdd(builder, a, b, "entry")
```

Code Generation

```
let a = LLVMGetParam(sumFunction, 0)

let b = LLVMGetParam(sumFunction, 1)

let result = LLVMBuildAdd(builder, a, b, "entry")

LLVMBuildRet(builder, result)
```

Code Generation

```
$ ./HelloWorld
```

Code Generation

```
$ ./HelloWorld
; ModuleID = 'Hello World’
```

```
define i32 @sum(i32, i32) {
entry:
  %result = add i32 %0, %1
  ret i32 %result
}
```

Code Execution

Code Execution

```
int sum(int a, int b) {

    int result = a + b;

    return result;

}
```

Code Execution

```
int sum(int a, int b) {

   int result = a + b;

   return result;

}

int main() {

   return sum(5, 6);

}
```

Code Execution

```
func runSumFunction(a: Int, _ b: Int) -> Int {
  return 0
}
runSumFunction(5, 6)
```

Code Execution

```
func runSumFunction(a: Int, _ b: Int) -> Int {

  let returnType = int32

  let functionType =

      LLVMFunctionType(returnType, nil, 0, 0)

  let wrapperFunction =

      LLVMAddFunction(module, "", functionType)

  // …
```

Code Execution

```
func runSumFunction(a: Int, _ b: Int) -> Int {
  // …
  let entryBlock =
          LLVMAppendBasicBlock(wrapperFunction,
                               "entry")
  LLVMPositionBuilderAtEnd(builder, entryBlock)
  // …
```

Code Execution

```
func runSumFunction(a: Int, _ b: Int) -> Int {
  // …
  let argumentsSize = strideof(LLVMValueRef) * 2
  let arguments = UnsafeMutablePointer.alloc(argumentsSize)
  let argA = LLVMConstInt(int32, UInt64(a), 0)
  let argB = LLVMConstInt(int32, UInt64(b), 0)
  arguments.initializeFrom([argA, argB])
  // …
```

Code Execution

```
func runSumFunction(a: Int, _ b: Int) -> Int {
  // …
  let callTemp =
        LLVMBuildCall(builder, sumFunction,
                      arguments, 2, "sum_temp")
  LLVMBuildRet(builder, callTemp)
  return 0
```

Code Execution

```
$ ./HelloWorld
```

Code Execution

```
$ ./HelloWorld
; ModuleID = 'Hello World’

define i32 @sum(i32, i32) {
entry:
  %result = add i32 %0, %1
  ret i32 %result
}

define i32 @0() {
entry:
  %sum_temp = call i32 @sum(i32 5, i32 6)
  ret i32 %sum_temp
}
```

Code Execution

```
let engineSize =
        strideof(LLVMExecutionEngineRef)
let engine =
        UnsafeMutablePointer.alloc(engineSize)
let errorSize =
        strideof(UnsafeMutablePointer<Int8>)
let error = UnsafeMutablePointer.alloc(errorSize)
```

Code Execution

```
let res =
    LLVMCreateExecutionEngineForModule(engine,
                                       module, error)
if res != 0 {
  let msg = String.fromCString(error.memory)
  print("\(msg)")
  exit(1)
}
```

Code Execution

```
let value = LLVMRunFunction(engine.memory,
                            wrapperFunction,
                            0, nil)
let result = LLVMGenericValueToInt(value, 0)
return Int(result)
```

Code Execution

```
LLVMLinkInMCJIT()
```

Code Execution

```
LLVMLinkInMCJIT()

LLVMInitializeNativeTarget()

LLVMInitializeNativeAsmPrinter()
```

Code Execution

```
$ cd $LLVM_BUILD_DIR
$ make llvm-config -j $CPU_NUM
```

Code Execution

```
$ $LLVM_BUILD_DIR/bin/llvm-config \
          --libs mcjit native
```

Code Execution

```
$ $LLVM_BUILD_DIR/bin/llvm-config \
          --libs mcjit native
-lLLVMX86Disassembler -lLLVMX86AsmParser -lLLVMX86CodeGen
-lLLVMSelectionDAG -lLLVMAsmPrinter -lLLVMCodeGen
-lLLVMScalarOpts -lLLVMInstCombine -lLLVMInstrumentation
-lLLVMProfileData -lLLVMTransformUtils -lLLVMBitWriter
-lLLVMX86Desc -lLLVMMCDisassembler -lLLVMX86Info
-lLLVMX86AsmPrinter -lLLVMX86Utils -lLLVMMCJIT
-lLLVMExecutionEngine -lLLVMTarget -lLLVMAnalysis
-lLLVMRuntimeDyld -lLLVMObject -lLLVMMCParser
-lLLVMBitReader -lLLVMMC -lLLVMCore -lLLVMSupport
```

Code Execution

```
$ $LLVM_BUILD_DIR/bin/llvm-config \
    --libs mcjit native | \
  sed "s/-l//g" | \
  xargs make -j $CPU_NUM
```

Code Execution

```
$ xcrun -sdk macosx \
    swiftc HelloWorld.swift \
    -I $LLVM_SOURCE_DIR/include/ \
    -I $LLVM_BUILD_DIR/include/ \
    -Xcc -D__STDC_CONSTANT_MACROS \
    -Xcc -D__STDC_LIMIT_MACROS \
    -lLLVMCore -lLLVMSupport \
    -L $LLVM_BUILD_DIR/lib \
    -lc++ -lncurses
```

Code Execution

```
$ xcrun -sdk macosx \
    swiftc HelloWorld.swift \
    -I $LLVM_SOURCE_DIR/include/ \
    -I $LLVM_BUILD_DIR/include/ \
    -Xcc -D__STDC_CONSTANT_MACROS \
    -Xcc -D__STDC_LIMIT_MACROS \
    `$LLVM_BUILD_DIR/bin/llvm-config \
        –libs mcjit native` \
    -L $LLVM_BUILD_DIR/lib \
    -lc++ -lncurses
```

Code Execution

```
func runSumFunction(a: Int, _ b: Int) -> Int {
    // …
}

let sum1 = runSumFunction(5, 6)

let sum2 = runSumFunction(10, 42)
```

Code Execution

```
$ ./HelloWorld
```

Code Execution

```
$ ./HelloWorld
11
52
```

What’s next?

What’s next?

- http://lowlevelbits.org/how-to-use-llvm-api-with-swift/
- http://github.com/AlexDenisov/swift_llvm

- http://llvm.org/docs/tutorial/index.html
- http://github.com/AlexDenisov/SwiftKaleidoscope

Thank You!

AlexDenisov,
http://lowlevelbits.org