What’s New in Clang and LLVM

Jessica Paquette, Compiler Engineer
JF Bastien, Compiler Engineer
Devin Coughlin, Program Analysis Engineer
Agenda
Agenda

New platform support
Agenda

- New platform support
- Low-level code size optimizations
Agenda

New platform support

Low-level code size optimizations

Language-level code size optimizations
Agenda

New platform support
Low-level code size optimizations
Language-level code size optimizations
Diagnostics
Agenda

- New platform support
- Low-level code size optimizations
- Language-level code size optimizations
- Diagnostics
- Static Analyzer checks
New Platform Support
Bitcode
What is LLVM Bitcode?

Source Code

doggo.c
snek.m
birb.swift
pupper.cc
What is LLVM Bitcode?

Source Code

doggo.c
snek.m
birb.swift
pupper.cc
What is LLVM Bitcode?

Source Code

- doggo.c
- snek.m
- birb.swift
- pupper.cc

Compiler
What is LLVM Bitcode?

Source Code
- doggo.c
- snek.m
- birb.swift
- pupper.cc

Compiler

LLVM Bitcode
- my_program.bc
What is LLVM Bitcode?

LLVM Bitcode

Serialization of internal compiler state

my_program.bc
Producing Apps for Two Chips with Bitcode

LLVM Bitcode

my_program.bc
Producing Apps for Two Chips with Bitcode

LLVM Bitcode

my_program.bc

App for 32-bit Chip

App for 64-bit Chip
Producing Apps for Two Chips with Bitcode

LLVM Bitcode

my_program.bc

App for 32-bit Chip

App for 64-bit Chip
Target-Tailored Bitcode = More Optimization

Bitcode for 32-bit Chip

my_program.bc → App for 32-bit Chip

Bitcode for 64-bit Chip

my_program.bc → App for 64-bit Chip
Code Size Improvements

Low-level code size optimizations
Why code size?
Jessica is Canadian, so she says “zed”
How does a compiler work?
int collatz(int Num) {
    if (Num % 2 == 0)
        return Num / 2;
    return Num * 3 + 1;
}
define i32 @collatz(i32) {
    if (Num % 2 == 0)
        return Num / 2;
    return Num * 3 + 1;
}
define i32 @collatz(i32) {
    %2 = and i32 %0, 1
    if (Num % 2 == 0)
        return Num / 2;
    return Num * 3 + 1;
}
define i32 @collatz(i32) {
    %2 = and i32 %0, 1
    %3 = icmp eq i32 %2, 0
    %4 = sdiv i32 %0, 2
    return Num * 3 + 1;
}
define i32 @collatz(i32) {
    %2 = and i32 %0, 1
    %3 = icmp eq i32 %2, 0
    %4 = sdiv i32 %0, 2
    %5 = mul nsw i32 %0, 3
    %6 = add nsw i32 %5, 1
    %7 = select i1 %3, i32 %4, i32 %6
    ret i32 %7
}
define i32 @collatz(i32) {
    %2 = and i32 %0, 1
    %3 = icmp eq i32 %2, 0
    %4 = sdiv i32 %0, 2
    %5 = mul nsw i32 %0, 3
    %6 = add nsw i32 %5, 1
    %7 = select i1 %3, i32 %4, i32 %6
    ret i32 %7
}
name: collatz

body:

bb.0:
%2 = and i32 %0, 1
%3 = icmp eq i32 %2, 0
%4 = sdiv i32 %0, 2
%5 = mul nsw i32 %0, 3
%6 = add nsw i32 %5, 1
%7 = select i1 %3, i32 %4, i32 %6
ret i32 %7

}
name:               collatz
body:             |
bb.0:
  %0: gpr32 = COPY $w0
  %3 = icmp eq i32 %2, 0
  %4 = sdiv i32 %0, 2
  %5 = mul nsw i32 %0, 3
  %6 = add nsw i32 %5, 1
  %7 = select i1 %3, i32 %4, i32 %6
ret i32 %7
}
name: collatz
body:

```
bb.0:

  %0:gpr32 = COPY $w0
  %5:gpr32 = MOVi32imm 2
  %6:gpr32 = SDIVWr %0, %5
  %5 = mul nsw i32 %0, 3
  %6 = add nsw i32 %5, 1
  %7 = select i1 %3, i32 %4, i32 %6
  ret i32 %7
```
name: collatz

body:

bb.0:

%0:gpr32 = COPY $w0
%5:gpr32 = MOVi32imm 2
%6:gpr32 = SDIVWr %0, %5
%7:gpr32 = MOVi32imm 3
%8:gpr32common = MADDWrr %0, %7, $wzr
%7 = select i1 %3, i32 %4, i32 %6
ret i32 %7

}
name: collatz

body:

bb.0:

%0:gpr32 = COPY $w0
%5:gpr32 = MOVi32imm 2
%6:gpr32 = SDIVWr %0, %5
%7:gpr32 = MOVi32imm 3
%8:gpr32common = MADDWrrr %0, %7, $wzr
%10:gpr32 = CSELWr %6, %9, 1, implicit $nzcv
ret i32 %7

}
name: collatz
body:

bb.0:

%0:gpr32 = COPY $w0
%5:gpr32 = MOVi32imm 2
%6:gpr32 = SDIVWr %0, %5
%7:gpr32 = MOVi32imm 3
%8:gpr32common = MADDWr %0, %7, $wzr
%10:gpr32 = CSELWr %6, %9, 1, implicit $nzcv
RET_ReallyLR implicit $w0
name:            collatz
body:             |

bb.0:
    %0:gpr32 = COPY $w0
    %5:gpr32 = MOVi32imm 2
    %6:gpr32 = SDIVWr %0, %5
    %7:gpr32 = MOVi32imm 3
    %8:gpr32common = MADDWrrr %0, %7, $wzr
    %10:gpr32 = CSELWr %6, %9, 1, implicit $nzcv
    RET_ReallyLR implicit $w0
name: collatz
body:

bb.0:

$w8 = \text{MOV}i32\text{imm} \ 2$

%6:gpr32 = \text{SDIV}W\text{r} \ %0, %5

%7:gpr32 = \text{MOV}i32\text{imm} \ 3

%8:gpr32\text{common} = \text{MADD}W\text{rrr} \ %0, %7, $wzr

%10:gpr32 = \text{CSEL}W\text{r} \ %6, %9, 1, \text{implicit} \ $nzcv

\text{RET\_ReallyLR} \ \text{implicit} \ $w0
name: collatz
body:

bb.0:

\$w8 = MOVi32imm 2
\$w9 = MOVi32imm 3
%6:gpr32 = SDIVwr %0, %5
%8:gpr32common = MADDwrrr %0, %7, \$wzr
%10:gpr32 = CSELwr %6, %9, 1, implicit \$nzcv
RET_ReallyLR implicit \$w0
name: collatz
body: |

bb.0:

$w8 = MOVi32imm 2
$w9 = MOVi32imm 3
$w8 = SDIVWr $w0, $w8
%8:gpr32common = MADDWrrr %0, %7, $wzr
%10:gpr32 = CSELWr %6, %9, 1, implicit $nzcv
RET_ReallyLR implicit $w0
name: collatz

body:

bb.0:
$w8 = MOVi32imm 2
$w9 = MOVi32imm 3
$w8 = SDIVWr $w0, $w8
$w9 = MADDWrrr $w0, $w9, $wzr
%10:gpr32 = CSELWr %6, %9, 1, implicit $nzcv
RET_ReallyLR implicit $w0
name: collatz
body:

```
bb.0:
  $w8 = MOVi32imm 2
  $w9 = MOVi32imm 3
  $w8 = SDIVWr $w0, $w8
  $w9 = MADDWrrr $w0, $w9, $wzr
  $w9 = ADDWri $w9, 1, 0
  $w0 = CSELWr $w8, $w9, 1, $nzcv
  RET_ReallyLR implicit $w0
```
name: collatz

body:

bb.0:

$w8 = MOVi32imm 2
$w9 = MOVi32imm 3
$w8 = SDIVWr $w0, $w8
$w9 = MADDWrrr $w0, $w9, $wzr
$w9 = ADDWri $w9, 1, 0
$w0 = CSELWr $w8, $w9, 1, $nzcv
RET_ReallyLR $w0
name:    collatz
body:    
bb.0:

$w8 = MOVZWi 2, 0
$w9 = MOVi32imm 3
$w8 = SDIVWr $w0, $w8
$w9 = MADDWrrr $w0, $w9, $wzr
$w9 = ADDWri $w9, 1, 0
$w0 = CSELWr $w8, $w9, 1, $nzcv
RET_ReallyLR $w0
name: collatz

body:

bb.0:

$w8 = MOVZWi 2, 0
$w9 = MOVZWi 3, 0
$w8 = SDIVWr $w0, $w8
$w9 = MADDWrrr $w0, $w9, $wzr
$w9 = ADDWri $w9, 1, 0
$w0 = CSELWr $w8, $w9, 1, $nzcv
RET_ReallyLR $w0
name:            collatz
body:             |
bb.0:
  $w8 = MOVZwi 2, 0
  $w9 = MOVZwi 3, 0
  $w8 = SDIVWr $w0, $w8
  $w9 = MADDWrrr $w0, $w9, $wzr
  $w9 = ADDWri $w9, 1, 0
  $w0 = CSELWr $w8, $w9, 1, $nzcv
  RET $lr, $w0
name: collatz

body: |

bb.0:

  $w8 = MOVZwi 2, 0
  $w9 = MOVZwi 3, 0
  $w8 = SDIVwr $w0, $w8
  $w9 = MADDWrr $w0, $w9, $wzr
  $w9 = ADDWri $w9, 1, 0
  $w0 = CSELWr $w8, $w9, 1, $nzcv
  RET $lr, $w0

collatz:

  mov   w8, #2
  mov   w9, #3
  sdiv  w8, w0, w8
  mul   w9, w0, w9
  add   w9, w9, #1
  csel  w0, w8, w9, ne
  ret
Function Outlining
An -Oz code size optimization
// These two functions share some instructions

hasse:

...  
      ldr w0, [sp, #16]  
      mul w0, w1, w2  
      add sp, sp, #16  
      ret

kakutani:

...  
      ldr w0, [sp, #16]  
      mul w0, w1, w2  
      add sp, sp, #16  
      ret
// Create a new function using the shared instructions...

hasse:
    ...
    ldr w0, [sp, #16]
    mul w0, w1, w2
    add sp, sp, #16
    ret

kakutani:
    ...
    ldr w0, [sp, #16]
    mul w0, w1, w2
    add sp, sp, #16
    ret

OUTLINED_FUNCTION_0:
    ldr w0, [sp, #16]
    mul w0, w1, w2
    add sp, sp, #16
    ret
// Replace the repeated sequences with calls to the new function

hasse:
  ...
  b OUTLINED_FUNCTION_0

kakutani:
  ...
  b OUTLINED_FUNCTION_0

OUTLINED_FUNCTION_0:
  ldr  w0, [sp, #16]
  mul  w0, w1, w2
  add  sp, sp, #16
  ret
Where do the savings come from? 🤔
// What does the compiler have to do to represent this function in assembly?

int ulam(int Num, int NumIters) {
    // TODO: Does this always converge?
    while (Num != 1) {
        Num = collatz(Num);
        ++NumIters;
    }
    return NumIters;
}
// What does the compiler have to do to represent this function in assembly?

```c
int ulam(int Num, int NumIters) {
    // TODO: Does this always converge?
    while (Num != 1) {
        Num = collatz(Num);
        ++NumIters;
    }
    return NumIters;
}

ulam:
    stp    x20, x19, [sp, #-32]!
    stp    x29, x30, [sp, #16]
    add    x29, sp, #16
    b      LBB1_2

LBB1_1:
    bl     collatz
    add    w19, w19, #1

LBB1_2:
    cmp    w0, #1
    b.ne   LBB1_1
    mov    x0, x19
    ldp    x29, x30, [sp, #16]
    ldp    x20, x19, [sp], #32
    ret
```
What does the compiler have to do to represent this function in assembly?

Prologue

```
ulam:
    stp    x20, x19, [sp, #-32]!
    stp    x29, x30, [sp, #16]
    add    x29, sp, #16
    b      LBB1_2
LBB1_1:
    bl     collatz
    add    w19, w19, #1
LBB1_2:
    cmp    w0, #1
    b.ne   LBB1_1
    mov    x0, x19
```

Epilogue

```
    ldp    x29, x30, [sp, #16]
    ldp    x20, x19, [sp], #32
    ret
```
// Compiler introduces similarity into code

collatz:

```assembly
stp    x20, x19, [sp, #-32]!
stp    x29, x30, [sp, #16]
```

...  

```assembly
ldp    x29, x30, [sp, #16]
ldp    x20, x19, [sp], #32
ret
```

kakutani:

```assembly
stp    x20, x19, [sp, #-32]!
stp    x29, x30, [sp, #16]
```

...  

```assembly
ldp    x29, x30, [sp, #16]
ldp    x20, x19, [sp], #32
ret
```

ulam:

```assembly
stp    x20, x19, [sp, #-32]!
stp    x29, x30, [sp, #16]
add    x29, sp, #16
b      LBB1_2
LBB1_1:
bl     collatz
add    w19, w19, #1
LBB1_2:
cmp    w0, #1
b.ne   LBB1_1
mov    x0, x19
ldp    x29, x30, [sp, #16]
ldp    x20, x19, [sp], #32
ret
```
// Compiler introduces similarity into code

```assembly
ulam:
    stp    x20, x19, [sp, #-32]!
    stp    x29, x30, [sp, #16]
    add    x29, sp, #16
    b      LBB1_2
LBB1_1:
    bl     collatz
    add    w19, w19, #1
LBB1_2:
    cmp    w0, #1
    b.ne   LBB1_1
    mov    x0, x19
    ldp    x29, x30, [sp, #16]
    ldp    x20, x19, [sp], #32
    ret
```

collatz:
    stp    x20, x19, [sp, #-32]!
    stp    x29, x30, [sp, #16]
    ...
    ldp    x29, x30, [sp, #16]
    ldp    x20, x19, [sp], #32
    ret

kakutani:
    stp    x20, x19, [sp, #-32]!
    stp    x29, x30, [sp, #16]
    ...
    ldp    x29, x30, [sp, #16]
    ldp    x20, x19, [sp], #32
    ret
Gotchas
Outlining Changes Control Flow

```
ulam:
  ...
  bl collatz
  add w19, w19, #1
  ...
  ret
```

Calls

```
collatz

ulam
```
Outlining Changes Control Flow

ulam:
... bl OUTLINED_FUNCTION_314 ...
ret

collatz

OUTLINED_FUNCTION_314

 Calls

Calls

ulam
Outlining Changes Control Flow

ulam:
...
bl OUTLINED_FUNCTION_314
...
ret

Calls

OUTLINED_FUNCTION_314

Calls

collatz

ulam
Crashing with Outlined Code

- `ulam`
- `collatz`
- `OUTLINED_FUNCTION_314`
- `calls`
Crashing with Outlined Code
Outlining Can Impact Backtraces

Original Backtrace

* frame #0: 0x0000BEEF collatz
* frame #1: 0x0000DEAD ulam
* frame #2: 0x0000FEED main

Outlined Backtrace

* frame #0: 0x0000CAFE collatz
* frame #1: 0x0000BEEF OUTLINED_FUNCTION_314
* frame #2: 0x0000DEAD ulam
* frame #3: 0x0000FEED main
Outlining Can Increase Execution Time
Outlining Can Increase Execution Time

Calls can have execution time overhead
Outlining Can Increase Execution Time

Calls can have execution time overhead

-Oz prioritizes size at all costs
Don’t compile performance-sensitive code with -Oz!
Use Instruments!
Smaller Faster

Prioritize Speed

Prioritize Size

No Optimization
Prioritize Size

Prioritize Speed

Smaller

Faster

No Optimization
Prioritize Size

Prioritize Speed

-O3

-O2

-Os

-Oz

No Optimization

Faster

Smaller
Prioritize Speed
Prioritize Size
Faster
No Optimization
Smaller
Extra Optimizations
Extra Optimizations

PGO — Profile-guided optimization
Extra Optimizations

PGO — Profile-guided optimization

LTO — Link-time optimization
Combining Optimizations

LTO + PGO + -O3

Wait until link-time to optimize
Use profiling information
Prioritize execution time
Combining Optimizations

LTO + PGO + -O3

- Wait until link-time to optimize
- Use profiling information
- Prioritize execution time

What’s New in LLVM

WWDC 2016
Enable `-Oz` in Your Project’s Build Settings
C, C++, and Objective-C Projects

- None [-O0]
- Fast [-O, O1]
- Faster [-O2]
- Fastest [-O3]
- Fastest, Smallest [-Os]
- Fastest, Aggressive Optimizations [-Ofast]
- Smallest, Aggressive Size Optimizations [-Oz]
Enable `-Oz` on Specific Files

C, C++, and Objective-C Files

Build Phases > Compile Sources > Compiler Flags

<table>
<thead>
<tr>
<th>Name</th>
<th>Compiler Flags</th>
</tr>
</thead>
<tbody>
<tr>
<td>ViewController.m</td>
<td>-O2</td>
</tr>
<tr>
<td>main.m</td>
<td>-Oz</td>
</tr>
<tr>
<td>AppDelegate.m</td>
<td>-Os</td>
</tr>
</tbody>
</table>
How does this impact app code size?
Executable binary code size information
Executable binary code size information

⚠ Not the total size of the app
# Example: Finding Code Size Info Using size

$ size ~/Library/Developer/Xcode/.../Sniffo.app/Contents/MacOS/Sniffo
# Example: Finding Code Size Info Using size

```
$ size ~/Library/Developer/Xcode/.../Sniffo.app/Contents/MacOS/Sniffo
```

<table>
<thead>
<tr>
<th>Segment name</th>
<th>__TEXT</th>
<th>__DATA</th>
<th>__OBJC</th>
<th>others</th>
<th>dec</th>
<th>hex</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>63221760</td>
<td>3289088</td>
<td>0</td>
<td>4313575424</td>
<td>4380086272</td>
<td>10512d000</td>
</tr>
</tbody>
</table>
# Example: Finding Code Size Info Using size

```bash
$ size ~/Library/Developer/Xcode/…/Sniffo.app/Contents/MacOS/Sniffo
```

```
__TEXT
63221760
```
# Example: Finding Code Size Info Using `size`

```
$ size ~/Library/Developer/Xcode/.../Sniffo.app/Contents/MacOS/Sniffo
```

<table>
<thead>
<tr>
<th>Category</th>
<th>Size (bytes)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Executable</td>
<td>63221760</td>
</tr>
<tr>
<td>C strings</td>
<td>3289088</td>
</tr>
<tr>
<td>Unwind info</td>
<td>0</td>
</tr>
<tr>
<td>Constants</td>
<td>4313575424</td>
</tr>
<tr>
<td>Stubs</td>
<td>4380086272</td>
</tr>
<tr>
<td>Stub helper</td>
<td>10512d000</td>
</tr>
</tbody>
</table>
# Example: Finding Code Size Info Using size

```
$ size ~/Library/Developer/Xcode/.../Sniffo.app/Contents/MacOS/Sniffo

__TEXT   __DATA   __OBJC  others      dec         hex
63221760  3289088  0       4313575424  4380086272  10512d000
```

Executable instructions

- 63221760

C strings

- __TEXT

Unwind info

- Constants

Stubs

- Stub helper
# Example: Detailed Code Size Info Using size

```shell
$ size -l -m ~/Library/Developer/Xcode/.../Sniffo.app/Contents/MacOS/Sniffo
```
# Example: Detailed Code Size Info Using size

```
$ size -l -m ~/Library/Developer/Xcode/.../Sniffo.app/Contents/MacOS/Sniffo
```

- **Size of segment**
  - Segment `__TEXT`: 63221760 (vmaddr 0x100000000 fileoff 0)
  - Section `__text`: 55384851 (addr 0x100002350 offset 9040)

- **Size of section**
  - Section `__text`: 55384851 (addr 0x100002350 offset 9040)

- **Section starting address**
Code Size Improvements

Language-level optimizations

JF Bastien, Compiler Engineer
Merge redundant block helpers and metadata
// Merge redundant block helpers and metadata
- (void)neutron:(id)particle {
    [self fuseWithCallbackBlock:^ (id nuclei) {
        [nuclei collide:particle];
        [self upDownDown];
    }];
}

- (void)proton:(id)particle chargeQuantity:(double)charge {
    [self fuseWithCallbackBlock:^ (id nuclei) {
        [self checkCoulombForce:charge];
        [particle collide:nuclei];
    }];
}
// Merge redundant block helpers and metadata

struct Metadata {
    unsigned long reserved, block_size;
    void *copy_helper;
    void *destroy_helper;
    const char *block_method_signature;
    uintptr_t block_layout_info;
};

- (void)neutron:(id)particle {
    [self fuseWithCallbackBlock:^ (id nuclei) {
        [nuclei collide:particle];
        [self upDownDown];
    }];
}

- (void)proton:(id)particle chargeQuantity:(double)charge {
    [self fuseWithCallbackBlock:^ (id nuclei) {
        [self checkCoulombForce:charge];
        [particle collide:nuclei];
    }];
}
// Generated block metadata – as of Xcode 11

static const char *__block_method_signature_v16_0_8 = "v16@?0@8";
static const struct { // neutron block metadata
    unsigned long reserved = 0, block_size = 48;
    void *copy_helper = ___copy_helper_block_ea8_32s40s;
    void *destroy_helper = ___destroy_helper_block_ea8_32s40s;
    const char *block_method_signature = __block_method_signature_v16_0_8;
    uintptr_t block_layout_info = 512;
} ___block_descriptor_48_ea8_32s40s_e8_v16?08l;
static const struct { // proton block metadata
    unsigned long reserved = 0, block_size = 52;
    void *copy_helper = ___copy_helper_block_ea8_32s40s;
    void *destroy_helper = ___destroy_helper_block_ea8_32s40s;
    const char *block_method_signature = __block_method_signature_v16_0_8;
    uintptr_t block_layout_info = 512;
} ___block_descriptor_52_ea8_32s40s_e8_v16?08l;
// Generated block metadata — as of Xcode 11

static const char *__block_method_signature_v16_0_8 = "v16@?0@8";

static const struct { // neutron block metadata
    unsigned long reserved = 0, block_size = 48;
    void *copy_helper = ___copy_helper_block_ea8_32s40s;
    void *destroy_helper = ___destroy_helper_block_ea8_32s40s;
    const char *block_method_signature = __block_method_signature_v16_0_8;
    uintptr_t block_layout_info = 512;
} ___block_descriptor_48_ea8_32s40s_e8_v16?08l;

static const struct { // proton block metadata
    unsigned long reserved = 0, block_size = 52;
    void *copy_helper = ___copy_helper_block_ea8_32s40s;
    void *destroy_helper = ___destroy_helper_block_ea8_32s40s;
    const char *block_method_signature = __block_method_signature_v16_0_8;
    uintptr_t block_layout_info = 512;
} ___block_descriptor_52_ea8_32s40s_e8_v16?08l;

/* 48 ≠ 52 :: can't merge */
static const char *__block_method_signature_v16_0_8 = "v16@?0@8";
static const struct { // neutron block metadata
    unsigned long reserved = 0, block_size = 48;
    void *copy_helper = ___copy_helper_block_ea8_32s40s;
    void *destroy_helper = ___destroy_helper_block_ea8_32s40s;
    const char *block_method_signature = __block_method_signature_v16_0_8;
    uintptr_t block_layout_info = 512;
} ___block_descriptor_48_ea8_32s40s_e8_v16?08l;
static const struct { // proton block metadata
    unsigned long reserved = 0, block_size = 52;
    void *copy_helper = ___copy_helper_block_ea8_32s40s;
    void *destroy_helper = ___destroy_helper_block_ea8_32s40s;
    const char *block_method_signature = __block_method_signature_v16_0_8;
    uintptr_t block_layout_info = 512;
} ___block_descriptor_52_ea8_32s40s_e8_v16?08l;
// Generated block metadata — as of Xcode 11

static const char *__block_method_signature_v16_0_8 = "v16@?0@8";

static const struct // neutron block metadata
{
    unsigned long reserved = 0, block_size = 48;
    void *copy_helper = ___copy_helper_block_ea8_32s40s;
    void *destroy_helper = ___destroy_helper_block_ea8_32s40s;
    const char *block_method_signature = __block_method_signature_v16_0_8;
    uintptr_t block_layout_info = 512;
} ___block_descriptor_48_ea8_32s40s_e8_v16?08l;

static const struct // proton block metadata
{
    unsigned long reserved = 0, block_size = 52;
    void *copy_helper = ___copy_helper_block_ea8_32s40s;
    void *destroy_helper = ___destroy_helper_block_ea8_32s40s;
    const char *block_method_signature = __block_method_signature_v16_0_8;
    uintptr_t block_layout_info = 512;
} ___block_descriptor_52_ea8_32s40s_e8_v16?08l;
// Generated block metadata – as of Xcode 11

static const char *__block_method_signature_v16_0_8 = "v16@?0@8";
static const struct { // neutron block metadata
    unsigned long reserved = 0, block_size = 48;
    void *copy_helper = ___copy_helper_block_ea8_32s40s;
    void *destroy_helper = ___destroy_helper_block_ea8_32s40s;
    const char *block_method_signature = __block_method_signature_v16_0_8;
    uintptr_t block_layout_info = 512;
} ___block_descriptor_48_ea8_32s40s_e8_v16?08l;

static const struct { // proton block metadata
    unsigned long reserved = 0, block_size = 52;
    void *copy_helper = ___copy_helper_block_ea8_32s40s;
    void *destroy_helper = ___destroy_helper_block_ea8_32s40s;
    const char *block_method_signature = __block_method_signature_v16_0_8;
    uintptr_t block_layout_info = 512;
} ___block_descriptor_52_ea8_32s40s_e8_v16?08l;
// Generated block helpers – as of Xcode 11

static void ___copy_helper_block_ea8_32s40s(void *block) {

}

static void ___destroy_helper_block_ea8_32s40s(void *block) {

}
// Generated block helpers – as of Xcode 11

static void ___copy_helper_block_ea8_32s40s(void *block) {
    objc_retain(*(id*)((char*)block) + 32));
    objc_retain(*(id*)((char*)block) + 40));
}

static void ___destroy_helper_block_ea8_32s40s(void *block) {
    objc_release(*(id*)((char*)block) + 32));
    objc_release(*(id*)((char*)block) + 40));
}
2–7%

Typical code size reduction
Instance Variables of Direct Subclasses of NSObject

Offsets can be constant in the implementation
// Card.h – a direct subclass of NSObject

@interface Card : NSObject

@property (copy) NSString *name;
@property (copy) NSString *type;
@property (copy) NSDictionary *manaCost;
@property (copy) NSString *abilitiesText;
@property (copy) NSString *flavourText;
@property (copy) NSString *expansion;
@property int power;
@property int toughness;
@property UIImage *art;
@property (copy) NSString *artist;

- (instancetype)initWithName:(NSString *)name;

@end
// Card.h — a direct subclass of NSObject

@interface Card : NSObject

@property (copy) NSString *name;
@property (copy) NSString *type;
@property (copy) NSDictionary *manaCost;
@property (copy) NSString *abilitiesText;
@property (copy) NSString *flavourText;
@property (copy) NSString *expansion;
@property int power;
@property int toughness;
@property UIImage *art;
@property (copy) NSString *artist;

-(instancetype)initWithName:(NSString *)name;
@end
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>NSObject</td>
</tr>
<tr>
<td>8</td>
<td>power</td>
</tr>
<tr>
<td>12</td>
<td>toughness</td>
</tr>
<tr>
<td>16</td>
<td>name</td>
</tr>
<tr>
<td>24</td>
<td>type</td>
</tr>
<tr>
<td>32</td>
<td>manaCost</td>
</tr>
<tr>
<td>40</td>
<td>abilitiesText</td>
</tr>
<tr>
<td>48</td>
<td>flavourText</td>
</tr>
<tr>
<td>56</td>
<td>expansion</td>
</tr>
<tr>
<td>64</td>
<td>art</td>
</tr>
<tr>
<td>72</td>
<td>artist</td>
</tr>
</tbody>
</table>
// Card.mm — @implementation knows all offsets

#import "Card.h"

@implementation Card

- (instancetype)initWithName:(NSString *)name {
    if ((self = [super init])) {
        self.name = name;
        // ...
        // ☀💧💀🔥🌳
        // ...
    }
    return self;
}
@end
// Card.mm — @implementation knows all offsets

#import "Card.h"

@implementation Card
- (instancetype)initWithName:(NSString *)name {
    if ((self = [super init])) {
        self.name = name;
        // ...
        // ☀💧💀🔥🌳
        // ...
    }
    return self;
}
@end
// Card.mm — @implementation knows all offsets

#import "Card.h"

@implementation Card

- (instancetype)initWithName:(NSString *)name {
    if ((self = [super init])) {
        self.name = name;
        //   ...
        //   ...
        //  
☀
💧
💀
🔥
🌳 
        //   ...
    }
    return self;
}
@end
# Card.h

@import "Card.h"

@implementation Card

- (instancetype)initWithName:(NSString *)name {
    if ((self = [super init])) {
        self.name = name;
        //   ...
        //   ...
        //   ...
        //   ...
        return self;
    }
}
@end

// Card.mm — @implementation knows all offsets

#import "Card.h"

@implementation Card

- ...

    self.name = name;
    //   ...
    //   ...
    //   ...
    //   ...

    return self;

@end
2%

Typical code size reduction
Improved Debuggability of C++ Types

And associated code size wins
// Debugging around standard library code — print.cc

#include <cstdlib>
#include <iostream>
#include <string>
#include <vector>

int main(int argc, char** argv) {
    std::vector<std::string> args(argv + 1, argv + argc);
    std::vector<int> numbers;
    numbers.reserve(args.size());
    for (std::string const& arg : args) {
        int n = std::atoi(arg.c_str());
        numbers.push_back(n);
    }
    for (int i : numbers)
        std::cout << i << '
';
}
// Debugging around standard library code – print.cc

#include <cstdlib>
#include <iostream>
#include <string>
#include <vector>

int main(int argc, char** argv) {
    std::vector<std::string> args(argv + 1, argv + argc);
    std::vector<int> numbers;
    numbers.reserve(args.size());
    for (std::string const& arg : args) {
        int n = std::atoi(arg.c_str());
        numbers.push_back(n); // Let's set a breakpoint at line 12.
    }
    for (int i : numbers)
        std::cout << i << '
';
}
$ lldb -- ./print 1 1 2 3 5
(lldb)
$ lldb -- ./print 1 1 2 3 5
(lldb)
$ lldb -- ./print 1 1 2 3 5
(lldb) b 12
Breakpoint 1: where = print`main + 420 at print.cc:12:17, address = 0x0000000000c0ffee
(lldb)
$ llldb -- ../print 1 1 2 3 5  
(lldb) b 12  
Breakpoint 1: where = print\`main + 420 at print.cc:12:17, address = 0x0000000000c0ffee  
(lldb) r
$ lldb -- ./print 1 1 2 3 5
(lldb) b 12
Breakpoint 1: where = print`main + 420 at print.cc:12:17, address = 0x0000000000c0ffee
(llldb) r
Process 1337 launched: '/Users/j_appleseed/print' (x86_64)
Process 1337 stopped
* thread #1, queue = 'com.apple.main-thread', stop reason = breakpoint 1.1
  frame #0: 0x0000000000c0ffee print`main(argc=6, argv=0x0000c0defefe0000) at print.cc:12:17
  9         numbers.reserve(args.size());
  10        for (std::string const& arg : args) {
  11            int n = std::atoi(arg.c_str());
->  12            numbers.push_back(n);
  13        }
  14        for (int i : numbers)
  15            std::cout << i << '\n';
Target 0: (print) stopped.
$ lldb -- ./print 1 1 2 3 5
(lldb) b 12
Breakpoint 1: where = print`main + 420 at print.cc:12:17, address = 0x0000000000c0ffee
(lldb) r
Process 1337 launched: '/Users/j_appleseed/print' (x86_64)
Process 1337 stopped
* thread #1, queue = 'com.apple.main-thread', stop reason = breakpoint 1.1
  frame #0: 0x0000000000c0ffee print`main(argc=6, argv=0x0000c0defefe0000) at print.cc:12:17
  9        numbers.reserve(args.size());
  10       for (std::string const& arg : args) {
  11           int n = std::atoi(arg.c_str());
  12           numbers.push_back(n);
  13       }
  14       for (int i : numbers)
  15           std::cout << i << '\n';
Target 0: (print) stopped.
† in release configuration of codebases that heavily use the C++ Standard Library
Code size reduction† and better debugging!

† in release configuration of codebases that heavily use the C++ Standard Library
C++ Static Destructor Suppression
class Logger {
public:
    template <typename... Ts>
    void log(Ts&&... message);
    void flush();

private:
    std::vector<std::string> mBuffer;
};

extern Logger logger;

Logger logger;
// Global with a destructor — logger.h

class Logger {
public:
    template<typename... Ts>
    void log(Ts&&... message);
    void flush();

private:
    std::vector<std::string> mBuffer;
};

extern Logger logger;

// Implementation file — logger.cc

Logger logger;
// Global with a destructor — logger.h

class Logger {
  public:
    template<typename... Ts>
      void log(Ts&&... message);
    void flush();
  
  private:
    std::vector<std::string> mBuffer;
  }

extern Logger logger;

// Implementation file — logger.cc

Logger logger;

// Another file — Game.cc

#include "logger.h"

class Game {
  public:
    Game();
    ~Game();

  private:
    Game game;

};

Game::~Game() {
    Game::~Game() {
}
// Global with a destructor — logger.h

class Logger {
public:
    template <typename... Ts>
    void log(Ts&&... message);
    void flush();

private:
    std::vector<std::string> mBuffer;
};

extern Logger logger;

// Implementation file — logger.cc

Logger logger;

// Another file — Game.cc

#include "logger.h"

class Game {
public:
    Game();
    ~Game();
    // ...
    // ...
};

Game game;

Game::~Game() {}
// Global with a destructor – logger.h

class Logger {
public:
    template <typename... Ts>
        void log(Ts&&... message);
    void flush();

private:
    std::vector<std::string> mBuffer;
};

extern Logger logger;

// Implementation file – logger.cc

Logger logger;

// Another file – Game.cc

#include "logger.h"

class Game {
public:
    Game();
    ~Game();
    // ...
    // ...
    Game game;

Game::~Game() {
    logger.log("Thank you for playing ",
            game_name, ",!"));
}
// Global with a destructor – logger.h

class Logger {
public:
    template <typename... Ts>
    void log(Ts&&... message);
    void flush();

private:
    std::vector<std::string> mBuffer;

};

extern Logger logger;

// Another file – Game.cc
#include "logger.h"

class Game {
public:
    Game();
    ~Game();
    // ...
    // ...

};

Game game;

Game::~Game() {
    logger.log("Thank you for playing ",
               game_name, "!");
}

// Implementation file – logger.cc
Logger logger;
// Global with a destructor — logger.h

class Logger {
public:
    template <typename... Ts>
    void log(Ts&&... message);
    void flush();

private:
    std::vector<std::string> mBuffer;
};

extern Logger logger;

// Implementation file — logger.cc

Logger logger;

// Another file — Game.cc
#include "logger.h"

class Game {
public:
    Game();
    ~Game();
    // ...
    // ...
};

Game game;

Game::~Game() {
    // ...
}
C++ Static Destructor Suppression

Application lifecycle
C++ Static Destructor Suppression
Application lifecycle

Not running
- Background
  - Background
  - Background
- Suspended
  - Inactive
    - Active
    - Foreground
@interface CowClicker : NSObject <UIApplicationDelegate>

- (void)applicationWillResignActive:(UIApplication *)application;
- (void)applicationDidBecomeActive:(UIApplication *)application;
- (void)applicationWillEnterForeground:(UIApplication *)application;
- (void)applicationDidEnterBackground:(UIApplication *)application;
- (void)applicationWillTerminate:(UIApplication *)application;

@end
class Logger {
public:
    template<typename... Ts>
    void log(Ts&&... message);
    void flush();

private:
    std::vector<std::string> mBuffer;
};

extern Logger logger;

// Implementation file - logger.cc

Logger logger;
// Global with a destructor — logger.h

class Logger {
public:
    template <typename... Ts>
    void log(Ts&&... message);
    void flush();

private:
    std::vector<std::string> mBuffer;
};
 extern Logger logger;

// Implementation file — logger.cc

[[clang::no_destroy]]
Logger logger;
Typical code size reduction

1%
Diagnostics
// -Wcall-to-pure-virtual-from-ctor-dtor
class Table {

};

// -Wcall-to-pure-virtual-from-ctor-dtor
Grail* find(Knight*);
Shrubbery* find(Knight*);

class Table {
    virtual Knight* galahad() = 0;
    virtual ~Table() { find(galahad()); }
};
Grail* find(Knight*);
Shrubbery* find(Knight*);

class Table {
    virtual Knight* galahad() = 0;
    virtual ~Table() { find(galahad()); } // warning: call to pure virtual member function 'galahad' has undefined behaviour;
                                       // overrides of 'galahad' in subclasses are not available in the destructor of 'Table'
Grail* find(Knight*);
Shrubbery* find(Knight*);

class Table {
    virtual Knight* galahad() = 0;
    virtual ~Table() {}  
};
Grail* find(Knight*);
Shrubbery* find(Knight*);

class Table {
    virtual Knight* galahad() = 0;
    virtual ~Table() { /* nothing! */ } 
};

class RoundTable : public Table {
    Knight* galahad() final { /* */ }
    ~RoundTable() override { find(galahad()); } 
};
// -Wmemset-transposed-args
// -Wmemset-transposed-args

struct Inbox {
    int emails[1024];
};
struct Inbox {
    int emails[1024];
};

void me_after_vacation(struct Inbox* inbox) {
}
struct Inbox { int emails[1024]; }; 

void me_after_vacation(struct Inbox* inbox) { 
  memset(inbox, sizeof(struct Inbox), 0); 
}

warning: 'size' argument to memset is '0';
  did you mean to transpose the last two arguments?
struct Inbox {  int emails[1024];  
};

void me_after_vacation(struct Inbox* inbox) {
  memset(inbox, 0, sizeof(struct Inbox));
}
struct Inbox { int emails[1024]; };  
void me_after_vacation(struct Inbox* inbox) {
    memset(inbox, 0, sizeof(struct Inbox));
}
/ *-Wmemset-transposed-args

```c
struct Inbox { int emails[1024]; }
void me_after_vacation(struct Inbox* inbox) {
    memset(inbox, 0, sizeof(struct Inbox));
}
```

Avoid
memset

Recommended
<algorithm> → std::fill or std::fill_n
struct Inbox {
    int 1024 emails;
};

void me_after_vacation(struct Inbox* inbox) {
}

Avoid
memset

Recommended
<algorithm> → std::fill or std::fill_n
struct Inbox { std::array<int, 1024> emails; };  
void me_after_vacation(struct Inbox* inbox) {
    std::fill(inbox->emails.begin(), inbox->emails.end(), 0);
}
// -Wreturn-std-move
// -Wreturn-std-move

struct Lion { /* 🦁 */ };  
struct Goat { std::vector<int> v; };  
struct Snake { /* 🐍 */ };
// -Wreturn-std-move

struct Lion {
   /* 🦁 */
};

struct Goat {
   std::vector<int> v;
};

struct Snake {
   /* 🐍 */
};

struct Chimæra : Lion, Goat, Snake {
   /* 🔥 🦁 🐐 🐍 */
};
struct Lion { /* 🦁 */};
struct Goat { std::vector<int> v; }; struct Snake { /* 🐍 */};
struct Chimæra : Lion, Goat, Snake { /* 🔥 🦁 🐐 🐍 */};

template <typename Creature> void slay(Creature&);

Goat bellerophon(Chimæra chimæra) {
    slay(chimæra);
    return chimæra;
}

warning: local variable 'chimæra' will be copied despite being returned by name
note: call 'std::move' explicitly to avoid copying
// -Wreturn-std-move

struct Lion { /* 🦁 */ };  
struct Goat { std::vector<int> v; };  
struct Snake { /* 🐍 */ };  

struct Chimæra : Lion, Goat, Snake { /* 🔥 🦁 🐐 🐍 */ };  

template <typename Creature> void slay(Creature&);

Goat bellerophon(Chimæra chimæra) {
  slay(chimæra);
  return chimæra;
}

warning: local variable 'chimæra' will be copied despite being returned by name  
note: call 'std::move' explicitly to avoid copying
struct Lion {
    /* 🦁 */
};

struct Goat {
    std::vector<int> v;
};

struct Snake {
    /* 🐍 */
};

struct Chimæra : Lion, Goat, Snake {
    /* 🔥 🔆 🐐 🐍 */
};

template <typename Creature> void slay(Creature&);

Goat bellerophon(Chimæra chimæra) {
    slay(chimæra);
    return chimæra;
}
/-*return-std-move*

```cpp
struct Lion {       /*🦁*/      }
struct Goat { std::vector<int> v; };      
struct Snake {       /*🐍*/      };

struct Chimæra : Lion, Goat, Snake { /*🔥🦁🐐🐍*/ };

template <typename Creature> void slay(Creature&);

Goat bellerophon(Chimæra chimæra) {
    slay(chimæra);
    return std::move(chimæra);
}
```
// -Wreturn-std-move

struct Lion { /* 🦁 */ };  
struct Goat { std::vector<int> v; };  
struct Snake { /* 🐍 */ }; 

struct Chimæra : Lion, Goat, Snake { /* 🔥 🦁 🐐 🐍 */ }; 

template <typename Creature> void slay(Creature&); 

bellerophon(Chimæra chimæra) {
    slay(chimæra);
    return chimæra;
}
/-*W*return-std-move*

```cpp
struct Lion {       /* 🦁 */      };
struct Goat { std::vector<int> v; };  
struct Snake {       /* 🐍 */      };

struct Chimæra : Lion, Goat, Snake { /* 🔥 🦁 🐐 🐍 */ };

template <typename Creature> void slay(Creature&);

Chimæra bellerophon(Chimæra chimæra) {
    slay(chimæra);
    return chimæra;
}
```
struct Lion  {
    /* 🦁 */
};

struct Goat  {
    std::vector<int> v;
};

struct Snake {
    /* 🐍 */
};

struct Chimæra : Lion, Goat, Snake {
    /* 🔥 🦁 🐐 🐍 */
};

template <typename Creature> void slay(Creature&);

bellerophon(Chimæra chimæra) {
    slay(chimæra);
    return chimæra;
}
// -Wreturn-std-move

struct Lion { /* 🦁 */};
struct Goat { std::vector<int> v; };  
struct Snake { /* 🐍 */};

struct Chimæra : Lion, Goat, Snake { /* 🔥 🦁 🐐 🐍 */};

template <typename Creature> void slay(Creature&);

std::optional<Chimæra> bellerophon(Chimæra chimæra) {
    slay(chimæra);
    return chimæra;
}
/ -Wsizeof-pointer-div

void all_work() {
    int no_play[10];
    size_t array_elts = sizeof(no_play) / sizeof(no_play[0]);
    // ... }
}
// -Wsizeof-pointer-div

void (int [10]) {
    size_t array_elts = sizeof(int) / sizeof(int[0]);
    // ... ...
}

void pigeon(int array[10]) {
    size_t array_elts = sizeof(array) / sizeof(array[0]); // Is this a 10?
    // ... 🦋 ...
    // ...
}

warning: 'sizeof(array)' will return the size of the pointer, not the array itself
void pigeon(int array[10]) {
    size_t array_elts = sizeof(array) / sizeof(array[0]);  // Is this a 10? 🦋
    // ... 🐦 ...
    // ...
}

warning: 'sizeof(array)' will return the size of the pointer, not the array itself

Avoid

sizeof(array) / sizeof(array[0])
```c
void pigeon(int array[10]) {
    size_t array_elts = sizeof(array) / sizeof(array[0]);  // Is this a 10? 🦋
    // ... 🦑 ...
    // ...
}

warning: 'sizeof(array)' will return the size of the pointer, not the array itself

Avoid

```

```c
sizeof(array) / sizeof(array[0])
```

Recommended

```
C++17 <iterator> → std::size
```
/-Wsizeof-pointer-div

void () {
    int [10]
    size_t array_elts =
    // ... ...
}
// -Wsizeof-pointer-div

void dull_boi() {
    int no_play[10];
    size_t array_elts = std::size(no_play);
    // ... ...
}

// -Wdefaulted-function-deleted
// -Wdefaulted-function-deleted

```cpp
struct Aberration {
    float& eyestalks;
    int eye;
    int mouth;
    Aberration() = default;
};
```

warning: explicitly defaulted default constructor is implicitly deleted
note:    default constructor of 'Aberration' is implicitly deleted because field 'eyestalks' of reference type 'float &' would not be initialized
// -Wdefaulted-function-deleted

```c
struct Aberration {
    float& eyestalks;
    int eye;
    int mouth;
    Aberration();
};
```

```cpp
struct Aberration {
    float& eyestalks;
    int eye;
    int mouth;
    Aberration(float& eyestalks) : eyestalks(eyestalks) { }
};
```
```c
// -Wdefaulted-function-deleted

struct Aberration {
    float eyestalks;
    int eye;
    int mouth;
    Aberration()
};
```
struct Aberration {
    float eyestalks;
    int eye;
    int mouth;
    Aberration() = default;
};
New Static Analyzer Checks

Devin Coughlin, Program Analysis Engineer
Finds Deep Bugs
Great at catching hard-to-produce, edge-case bugs
Three New C++ Checks

Use after move bugs

Dangling C string pointers with `std::string`

Reference-counting bugs in DriverKit and IOKit
Use After Move in C++
C++ Moves Avoid Unwanted Copies

Book myNovel("It was the best of times...");  
publish(std::move(myNovel));
C++ Moves Avoid Unwanted Copies

```cpp
Book myNovel("It was the best of times..."两岸);
publish(std::move(myNovel));
```
C++ Moves Avoid Unwanted Copies

```
Book myNovel("It was the best of times...");
publish(std::move(myNovel));
```
C++ Moves Avoid Unwanted Copies

```cpp
Book myNovel("It was the best of times..."$
);

publish(std::move(myNovel));
```

Moves from source variable rather than copying
C++ Moves Avoid Unwanted Copies

```
Book myNovel("It was the best of times..."Porno);
publish(std::move(myNovel));
```

Moves from source variable rather than copying
Can enforce unique ownership semantics
C++ Moves Avoid Unwanted Copies

- Moves from source variable rather than copying
- Can enforce unique ownership semantics
- Leaves source in unspecified state

```cpp
Book myNovel("It was the best of times...");
publish(std::move(myNovel));
```
Do Not Use Variable after Moving

```cpp
Book myNovel("It was the best of times...");

publish(std::move(myNovel));
```
Do Not Use Variable after Moving

Book myNovel("It was the best of times...");

publish(std::move(myNovel));

myNovel.spellCheck();
Do Not Use Variable after Moving

May have unexpected results or even crash!
Do Not Use Variable after Moving

Book myNovel("It was the best of times...");

publish(std::move(myNovel));

myNovel.spellCheck();

Method called on moved-from object

May have unexpected results or even crash!
Do Not Use Variable after Moving

Book myNovel("It was the best of times...");

publish(std::move(myNovel));

myNovel.spellCheck(); Method called on moved-from object

May have unexpected results or even crash!

Common fix is to reorder code
Do Not Use Variable after Moving

```cpp
Book myNovel("It was the best of times...");

myNovel.spellCheck();

publish(std::move(myNovel));
```

May have unexpected results or even crash!

Common fix is to reorder code
Dangling Pointers from `std::string`
Mixing C++ and C Strings Can Be Tricky!

```c++
const char *generateGreeting(const char *name) {
    std::string greeting = "Hello ";
    greeting.append(name);
    return greeting.c_str();
}

printf("%s from WWDC!", generateGreeting("World"));
```
Mixing C++ and C Strings Can Be Tricky!

```c++
const char *generateGreeting(const char *name) {
    std::string greeting = "Hello ";
    greeting.append(name);
    return greeting.c_str();
}

printf("%s from WWDC!", generateGreeting("World"));
```
Mixing C++ and C Strings Can Be Tricky!

```cpp
class Example {
public:
    Example() {
        const char *generateGreeting(const char *name) {
            std::string greeting = "Hello ";
greeting.append(name);
            return greeting.c_str();
        }

        printf("%s from WWDC!", generateGreeting("World"));
    }
};
```
Mixing C++ and C Strings Can Be Tricky!

```cpp
const char *generateGreeting(const char *name) {
    std::string greeting = "Hello ";
    greeting.append(name);
    return greeting.c_str();
}

printf("%s from WWDC!", generateGreeting("World"));
```
Mixing C++ and C Strings Can Be Tricky!

```cpp
const char *generateGreeting(const char *name) {
    std::string greeting = "Hello ";
    greeting.append(name);
    return greeting.c_str();
}

printf("%s from WWDC!", generateGreeting("World"));
```

c_str() returns inner pointer to buffer inside std::string
Mixing C++ and C Strings Can Be Tricky!

```cpp
const char *generateGreeting(const char *name) {
    std::string greeting = "Hello ";
greeting.append(name);
    return greeting.c_str();
}

printf("%s from WWDC!", generateGreeting("World"));
```

c\_str() returns inner pointer to buffer inside std::string
Buffer deallocated when std::string goes out of scope
Mixing C++ and C Strings Can Be Tricky!

```cpp
const char *generateGreeting(const char *name) {
    std::string greeting = "Hello ";
greeting.append(name);
    return greeting.c_str();
}

printf("%s from WWDC!", generateGreeting("World"));
```

c\_str\() returns inner pointer to buffer inside std::string

Buffer deallocated when std::string goes out of scope
Mixing C++ and C Strings Can Be Tricky!

```cpp
const char *generateGreeting(const char *name) {
    std::string greeting = "Hello ";
    greeting.append(name);
    return greeting.c_str();
}

printf("%s from WWDC!", generateGreeting("World"));
```

`c_str()` returns inner pointer to buffer inside `std::string`. Buffer deallocated when `std::string` goes out of scope. Using deallocated memory may crash!
Mixing C++ and C Strings Can Be Tricky!

```cpp
const char *generateGreeting(const char *name) {
    std::string greeting = "Hello ";
greeting.append(name);
    return greeting.c_str();
}
```

```cpp
printf("%s from WWDC!", generateGreeting("World"));
```

- `c_str()` returns inner pointer to buffer inside `std::string`
- Buffer deallocated when `std::string` goes out of scope
- Using deallocated memory may crash!
std::string generateGreeting(const char *name) {
    std::string greeting = "Hello ";
    greeting.append(name);
    return greeting;
}

std::string greeting = generateGreeting("World");
printf("%s from WWDC!", greeting.c_str());
std::string generateGreeting(const char *name) {
    std::string greeting = "Hello ";
    greeting.append(name);
    return greeting;
}

std::string greeting = generateGreeting("World");
printf("%s from WWDC!", greeting.c_str());
std::string generateGreeting(const char *name) {
    std::string greeting = "Hello ";
greeting.append(name);
    return greeting;
}

std::string greeting = generateGreeting("World");
printf("%s from WWDC!", greeting.c_str());
std::string generateGreeting(const char *name) {
    std::string greeting = "Hello ";
greeting.append(name);
    return greeting;
}

std::string greeting = generateGreeting("World");
printf("%s from WWDC!", greeting.c_str());
std::string generateGreeting(const char *name) {
    std::string greeting = "Hello ";
    greeting.append(name);
    return greeting;
}
std::string greeting = generateGreeting("World");
printf("%s from WWDC!", greeting.c_str());

Change scope of `std::string` to last as long as C string is used
Match Lifetimes of C++ and C Strings

```cpp
std::string generateGreeting(const char *name) {
    std::string greeting = "Hello ";
greeting.append(name);
    return greeting;
}
std::string greeting = generateGreeting("World");
printf("%s from WWDC!", greeting.c_str());
```

Change scope of `std::string` to last as long as C string is used

Often easier to stay within C++ world
Reference Counting in DriverKit and IOKit
Drivers Use Manual Retain/Release
Drivers Use Manual Retain/Release

OSObject uses retain/release for memory management

- Similar to CoreFoundation or Objective-C without ARC
Drivers Use Manual Retain/Release

**OSObject** uses retain/release for memory management

- Similar to CoreFoundation or Objective-C without ARC

Easy to over-release

- Memory used after deallocated
Drivers Use Manual Retain/Release

Objective uses retain/release for memory management
- Similar to CoreFoundation or Objective-C without ARC

Easy to over-release
- Memory used after deallocated

Easy to under-release
- Memory leaked
Allocated Objects Must Be Released

OSArray *devices = OSArray::withCapacity(2);
fillInDevices(devices);
setUpDevices(devices);

return true;
Allocated Objects Must Be Released

```
OSArray *devices = OSArray::withCapacity(2);
fillInDevices(devices);
setUpDevices(devices);

return true;
```
Allocated Objects Must Be Released

OSArray *devices = OSArray::withCapacity(2);
fillInDevices(devices);
setUpDevices(devices);

return true;

OSArray::withCapacity() returns retained

Array allocated and returned retained
Allocated Objects Must Be Released

```cpp
OSArray *devices = OSArray::withCapacity(2);
fillInDevices(devices);
setUpDevices(devices);
return true;
```

Array allocated and returned retained

```
OSArray::withCapacity() returns retained

Array will leak if not released
```
Allocated Objects Must Be Released

OSArray *devices = OSArray::withCapacity(2);
fillInDevices(devices);
setUpDevices(devices);
return true;

OSArray::withCapacity() returns retained

Array will leak if not released
Allocated Objects Must Be Released

```cpp
OSArray *devices = OSArray::withCapacity(2);
fillInDevices(devices);
setUpDevices(devices);
OSObjectRelease(devices);
return true;
```

- `OSArray::withCapacity()` returns retained
- Array allocated and returned retained
- Array will leak if not released
Convention for Memory Management
Convention for Memory Management

Similar in spirit to rules for CoreFoundation and ObjC Manual Retain/Release
Convention for Memory Management

Similar in spirit to rules for CoreFoundation and ObjC Manual Retain/Release

Default convention is to return retained (+1)

- Clients must call `OSObjectRelease()`
Convention for Memory Management

Similar in spirit to rules for CoreFoundation and ObjC Manual Retain/Release

Default convention is to return retained (+1)
- Clients must call `OSObjectRelease()`

Except getters return unretained (+0)
- Clients must not call `OSObjectRelease()`
Differing from Convention

```c
OSObject *findFirstDevice() {
    OSObject *result = devices->getObject(0);

    return result;
}
```
Differing from Convention

```c
OSObject *findFirstDevice() {
    OSObject *result = devices->getObject(0);

    return result;
}
```
Differing from Convention

```c
OSObject *findFirstDevice() {
    OSObject *result = devices->getObject(0);
    return result;
}
```
Differing from Convention

```c
OSObject *findFirstDevice() {
    OSObject *result = devices->getObject(0);
    return result;
}
```

Object with +0 retain count returned where +1 is expected
Differing from Convention

```c
OSObject *findFirstDevice() {
    OSObject *result = devices->getObject(0);

    return result;
}
```

Change behavior to follow convention
Differing from Convention

Change behavior to follow convention

```c
OSObject *findFirstDevice() {
    OSObject *result = devices->getObject(0);
    OSObjectRetain(result);
    return result;
}
```
OSObject *findFirstDevice() {
    OSObject *result = devices->getObject(0);

    return result;
}
Change behavior to follow convention

Rename method to follow convention

```c
OSObject *getFirstDevice() {  
    OSObject *result = devices->getObject(0);  

    return result;  
}
```
Differing from Convention

```c
OSObject *findFirstDevice() {
    OSObject *result = devices->getObject(0);

    return result;
}
```

Change behavior to follow convention

Rename method to follow convention

Add annotation to tell readers and analyzer convention not followed
Differing from Convention

```c
OSObject *findFirstDevice() DRIVERKIT_RETURNS_NOT_RETAINED;

OSObject *findFirstDevice() {
    OSObject *result = devices->getObject(0);

    return result;
}
```

Change behavior to follow convention

Rename method to follow convention

Add annotation to tell readers and analyzer convention not followed
Run Analyzer on Your Code!
Supports Objective-C, C, C++
Run Analyzer on Your Code!
Supports Objective-C, C, C++

Analyze During Build
Summary

LLVM bitcode enabled seamless 64-bit transition for watchOS

Reduce code size with new compiler optimizations

Run the static analyzer on your code
More Information

developer.apple.com/wwdc19/409

<table>
<thead>
<tr>
<th>Lab</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>LLVM Compiler, Objective-C, C++, and Linking Lab</td>
<td>Thursday, 9:00</td>
</tr>
<tr>
<td>Performance, Power, Crashes, and Debugging Lab</td>
<td>Thursday, 12:00</td>
</tr>
</tbody>
</table>