Compact Value Witnesses in Swift

Dario Rexin

Value Witnesses

Value Witnesses

```swift
func genericFn<T>(x: T) {
    // ...
}
```

```swift
func genericFn(x: AnyObject) {
    // ...
}
```

Value Witnesses

```swift
func genericFn<T>(x: T) {
    // ...
}
```

```swift
let y: Int = 42
genericFn(x: y)
```

```swift
func genericFn(x: Int) {
    // ...
}
```

Value Witnesses

```swift
func genericFn<T>(x: T) {
    // ...
}
```

```swift
let y: Int = 42
genericFn(x: y)
```

```swift
func genericFn(x: Int) {
    // ...
}
```

Value Witnesses

```
struct ValueWitnessTable {
    T*   (*initializeBufferWithCopyOfBuffer)(B *dest, const B *src, const Metadata *self);
    void (*destroy)(T* object, const Metadata *self);
    T*   (*initializeWithCopy)(T *dest, const T *src, const Metadata *self);
    T*   (*assignWithCopy)(T *dest, const T *src, const Metadata *self);
    T*   (*initializeWithTake)(T *dest, const T *src, const Metadata *self);
    T*   (*assignWithTake)(T *dest, const T *src, const Metadata *self);
    // ...
    size_t size, stride, alignment;
};
```

Value Witnesses

```swift
func genericFn<T>(x: T) {
    // ...
}
```

Value Witnesses

```swift
func genericFn<T>(x: T) {
    genericFn2(x: x)
}
```

```c
void genericFn(T* object, const Metadata *type) {
    size_t size = type->vw_size();
    T* objectCopy = alloca(size);
    type->vw_initializeWithCopy(objectCopy, object);
    genericFn2(objectCopy, type);
}
```

Value Witnesses

```
func genericFn2<T>(x: consuming T) {
    // ...
}
```

```
void genericFn(T* object, const Metadata *type) {
    // ...
    type->vw_destroy(object);
}
```

Value Witnesses

```
struct SomeStruct {
    let x: Int
    let y: SomeClass
}
```

```
void SomeStruct_destroy(SomeStruct *obj, const Metadata *self) {
    swift_release(obj->y);
}
```

Value Witnesses

```
struct SomeStruct {
    let x: Int
    let y: SomeClass
}
```

```
SomeStruct* SomeStruct_initWithCopy(SomeStruct *dest, const SomeStruct *src,
                                    const Metadata *self) {
    dest->x = src->x;
    dest->y = src->y;
    swift_retain(dest->y);
    return dest;
}
```

Value Witnesses

```
struct SomeStruct {
    let x: Int
    let y: SomeClass
}
```

```
SomeStruct* SomeStruct_assignWithCopy(SomeStruct *dest, const SomeStruct *src,
                                      const Metadata *self) {
    swift_release(dest->y);
    dest->x = src->x;
    dest->y = src->y;
    swift_retain(dest->y);
    return dest;
}
```

Value Witnesses

```
struct SomeStruct<T> {
    let x: Int
    let y: T
}
```

```
SomeStruct* SomeStruct_assignWithCopy(SomeStruct *dest, const SomeStruct *src,
                                      const Metadata *self) {
    dest->x = src->x;
    const Metadata *T = self->getGenericArgs()[0];
    T->vw_assignWithCopy(&dest->y, &src->y);
    return dest;
}
```

Code size impact

Code size impact

```swift
struct SomeStruct {
    let x: Int
    let y: SomeClass
}
```

```
_$s4test10SomeStructVwCP:
    stp     x20, x19, [sp, #-32]!
    stp     x29, x30, [sp, #16]
    add     x29, sp, #16
    mov     x19, x0
    ldp     x8, x0, [x1]
    stp     x8, x0, [x19]
    bl      _swift_retain
    mov     x0, x19
    ldp     x29, x30, [sp, #16]
    ldp     x20, x19, [sp], #32
    ret

_$s4test10SomeStructVwxx:
    ldr     x0, [x0, #8]
    b      _swift_release

_$s4test10SomeStructVwcp:
    stp     x20, x19, [sp, #-32]!
    stp     x29, x30, [sp, #16]
    add     x29, sp, #16
    mov     x19, x0
    ldp     x8, x0, [x1]
    stp     x8, x0, [x19]
    bl      _swift_retain
    mov     x0, x19
    ldp     x29, x30, [sp, #16]
    ldp     x20, x19, [sp], #32
    ret
```

```
_$s4test10SomeStructVwca:
    stp     x20, x19, [sp, #-32]!
    stp     x29, x30, [sp, #16]
    add     x29, sp, #16
    mov     x19, x0
    ldr     x8, [x1]
    str     x8, [x0]
    ldr     x0, [x1, #8]
    ldr     x20, [x19, #8]
    str     x0, [x19, #8]
    bl      _swift_retain
    mov     x0, x20
    bl      _swift_release
    mov     x0, x19
    ldp     x29, x30, [sp, #16]
    ldp     x20, x19, [sp], #32
    ret

_$s4test10SomeStructVwta:
    stp     x20, x19, [sp, #-32]!
    stp     x29, x30, [sp, #16]
    add     x29, sp, #16
    mov     x19, x0
    ldp     x8, x9, [x1]
    ldr     x0, [x0, #8]
    stp     x8, x9, [x19]
    bl      _swift_release
    mov     x0, x19
    ldp     x29, x30, [sp, #16]
    ldp     x20, x19, [sp], #32
    ret
```

Code size impact

```swift
struct SomeStruct {
    let x: Int
    let y: SomeClass
}
```

```
_$s4test10SomeStructVwCP:
    stp    x20, x19, [sp, #-32]!
    stp    x29, x30, [sp, #16]
    add    x29, sp, #16
    mov    x19, x0
    ldp    x8, x0, [x1]
    stp    x8, x0, [x19]
    bl     _swift_retain
    mov    x0, x19
    ldp    x29, x30, [sp, #16]
    ldp    x20, x19, [sp], #32
    ret

_$s4test10SomeStructVwxx:
    ldr    x0, [x0, #8]
    b      _swift_release

_$s4test10SomeStructVwcp:
    stp    x20, x19, [sp, #-32]!
    stp    x29, x30, [sp, #16]
    add    x29, sp, #16
    mov    x19, x0
    ldp    x8, x0, [x1]
    stp    x8, x0, [x19]
    bl     _swift_retain
    mov    x0, x19
    ldp    x29, x30, [sp, #16]
    ldp    x20, x19, [sp], #32
    ret
```

```
_$s4test10SomeStructVwca:
    stp    x20, x19, [sp, #-32]!
    stp    x29, x30, [sp, #16]
    add    x29, sp, #16
    mov    x19, x0
    ldr    x8, [x1]
    str    x8, [x0]
    ldr    x0, [x1, #8]
    ldr    x20, [x19, #8]
    str    x0, [x19, #8]
    bl     _swift_retain
    mov    x0, x20
    bl     _swift_release
    mov    x0, x19
    ldp    x29, x30, [sp, #16]
    ldp    x20, x19, [sp], #32
    ret

_$s4test10SomeStructVwta:
    stp    x20, x19, [sp, #-32]!
    stp    x29, x30, [sp, #16]
    add    x29, sp, #16
    mov    x19, x0
    ldp    x8, x9, [x1]
    ldr    x0, [x0, #8]
    stp    x8, x9, [x19]
    bl     _swift_release
    mov    x0, x19
    ldp    x29, x30, [sp, #16]
    ldp    x20, x19, [sp], #32
    ret
```

208 bytes

Code size impact

```
struct SomeStruct {
    let x: Int
    let y: SomeClass
}

struct SomeOtherStruct {
  let x: SomeStruct?
  let y: SomeClass
}
```

Code size impact

```swift
struct SomeStruct {
    let x: Int
    let y: SomeClass
}

struct SomeOtherStruct {
  let x: SomeStruct?
  let y: SomeClass
}
```

328 bytes

Compact Value Witnesses

Compact Value Witnesses

- Compact
- Instantiable
- Fast
- Compatible

Compact Value Witnesses

```
struct LayoutString {
    uint64_t flags;
    size_t opsBytes;
    uint8_t ops[];
} __attribute__((__packed__));
```

Compact Value Witnesses

Native references

- Strong
- Weak
- Unowned

ObjC references

- ObjC
- Block
- Bridge

Unknown references

- Strong
- Weak
- Unowned

Compact Value Witnesses

```swift
struct SomeStruct {
    let x: Int
    let y: SomeClass
}
```

Compact Value Witnesses

```swift
struct SomeStruct {
    let x: Int
    let y: SomeClass
}
```

0x0200000000000008

Compact Value Witnesses

```swift
struct SomeStruct {
    let x: Int
    let y: SomeClass
}
```

```
0x0200000000000008
0x0000000000000000
```

Compact Value Witnesses

```swift
struct SomeStruct {
    let x: Int
    let y: SomeClass
}
```

```
0x0000000000000000
0x0000000000000008
0x0200000000000008
0x0000000000000000
```

Compact Value Witnesses

```
struct SomeStruct {
    let x: Int
    let y: SomeClass
}
```

```
0x0000000000000000
0x0000000000000008
0x0200000000000008
0x0000000000000000
```

32 bytes

Compact Value Witnesses

Enums

- Empty
- No payload
- Singleton
- Single payload
- Multi payload

Compact Value Witnesses

Enums

- ~~Empty~~
- ~~No payload~~
- Singleton
- Single payload
- Multi payload

Compact Value Witnesses

Enums

- ~~Empty~~
- ~~No payload~~
- ~~Singleton~~
- Single payload
- Multi payload

Compact Value Witnesses

Enums

- ~~Empty~~
- No payload
- Singleton
- Single payload
- Multi payload

Compact Value Witnesses

```
unsigned (*getEnumTag)(T *obj, const Metadata *self);

void  (*destructiveInjectEnumTag)(T *obj, unsigned tag, const Metadata *self);
```

Compact Value Witnesses

```
unsigned (*getEnumTag)(T *obj, const Metadata *self);

void  (*destructiveInjectEnumTag)(T *obj, unsigned tag, const Metadata *self);
```

Compact Value Witnesses

```swift
enum SomeEnum {
    case a
    case b
    case c
    case d(Int, SomeClass)
}
```

Compact Value Witnesses

```
enum SomeEnum {
    case a
    case b
    case c
    case d(Int, SomeClass)
}
```

480 bytes

Compact Value Witnesses

```
struct SinglePayloadSimple {
    uint64_t opCodeAndOffset;
    uint64_t byteCountsAndOffset;
    size_t payloadSize;
    uint64_t zeroTagValue;
    size_t numNonPayloadCases;
    size_t opsBytes;
    size_t skip;
    uint8_t payloadOps[];
} __attribute__((__packed__));
```

Compact Value Witnesses

```
struct SinglePayloadSimple {
    uint64_t opCodeAndOffset;
    uint64_t byteCountsAndOffset;
    size_t payloadSize;
    uint64_t firstNonPayloadValue;
    size_t numNonPayloadCases;
    size_t opsBytes;
    size_t skip;
    uint8_t payloadOps[];
} __attribute__((__packed__));
```

Compact Value Witnesses

```
struct SinglePayloadSimple {
    uint64_t opCodeAndOffset;
    uint64_t byteCountsAndOffset;
    size_t payloadSize;
    uint64_t zeroTagVal
    size_t numNonPaylo dCa es
    size_t opsBytes;
    size_t skip;
    uint8_t payloadOps[
} __attribute__((__packed__));
```

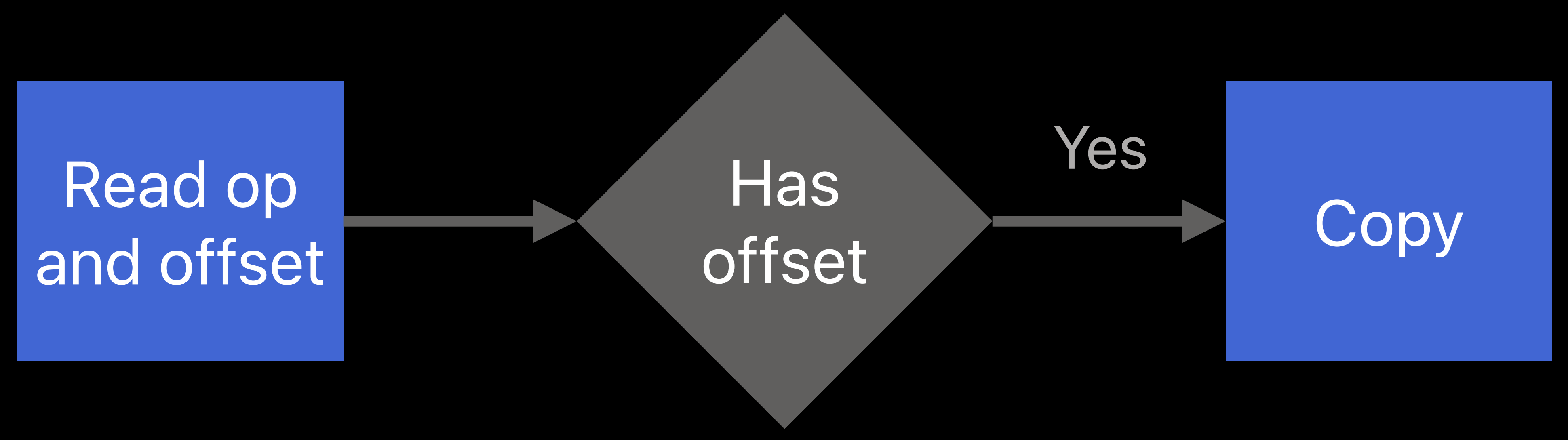
88 bytes

Runtime

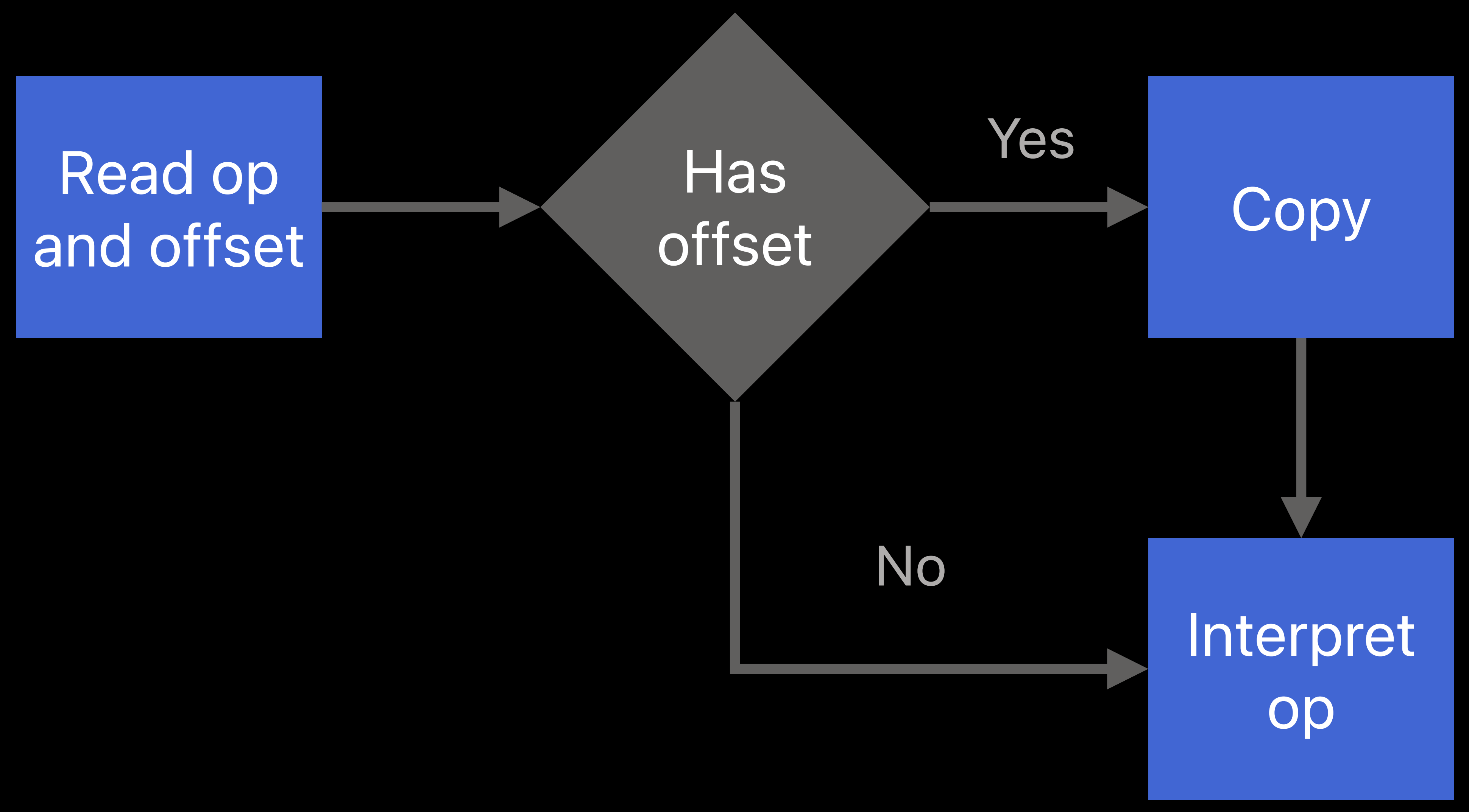
Runtime

Read op and offset

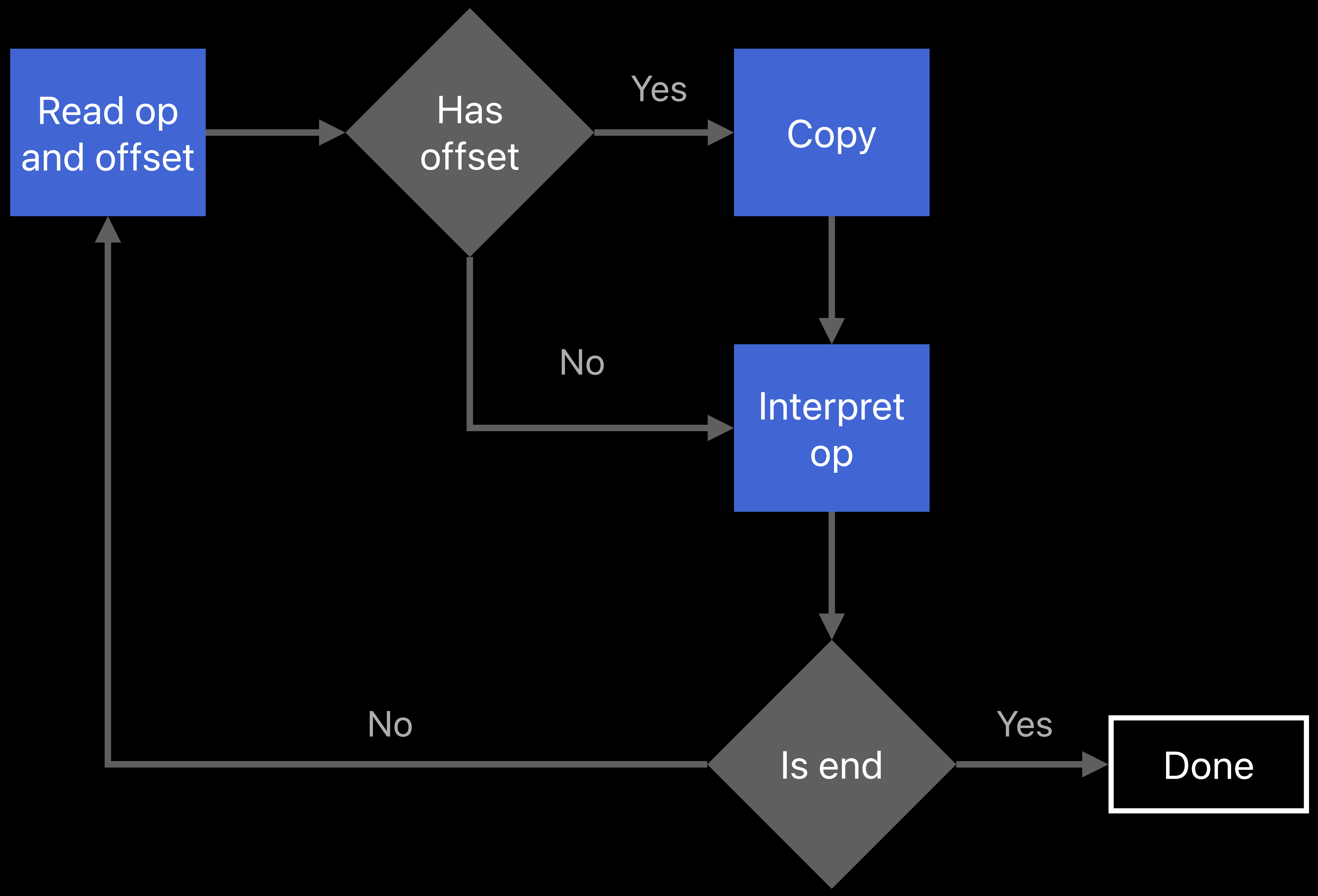
Runtime



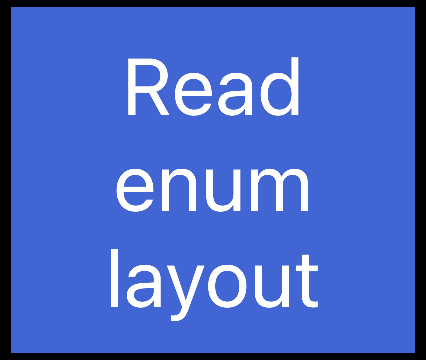
Runtime



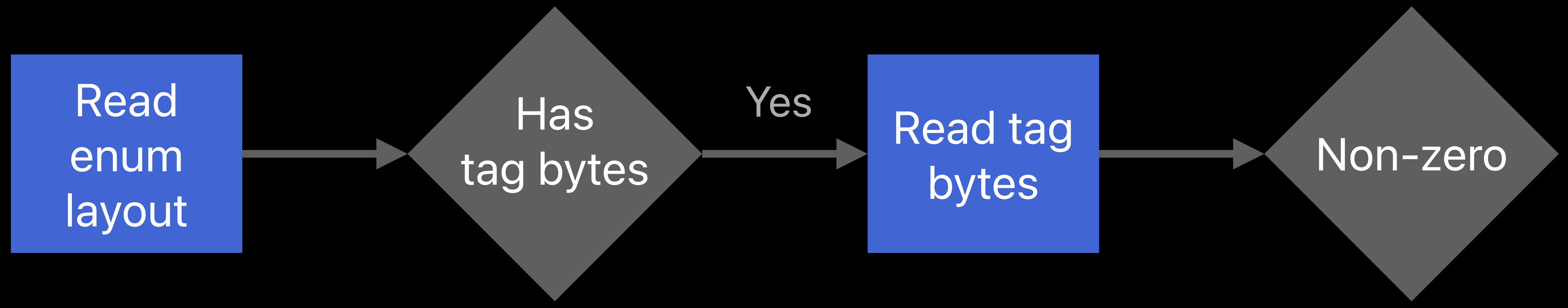
Runtime


Read op and offset
Has offset
Yes
Copy
No
Interpret op
Is end
No
Yes
Done

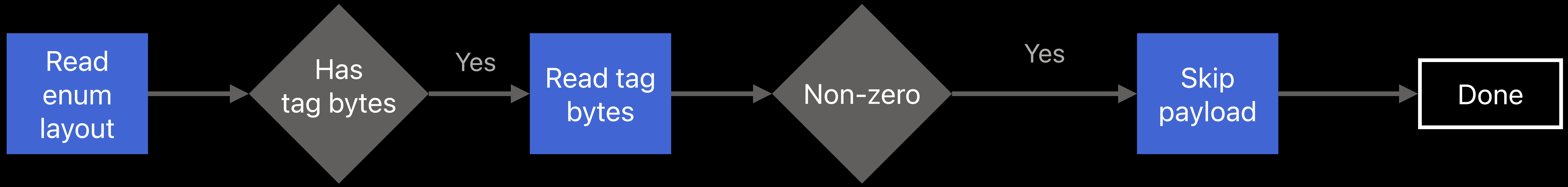
Runtime



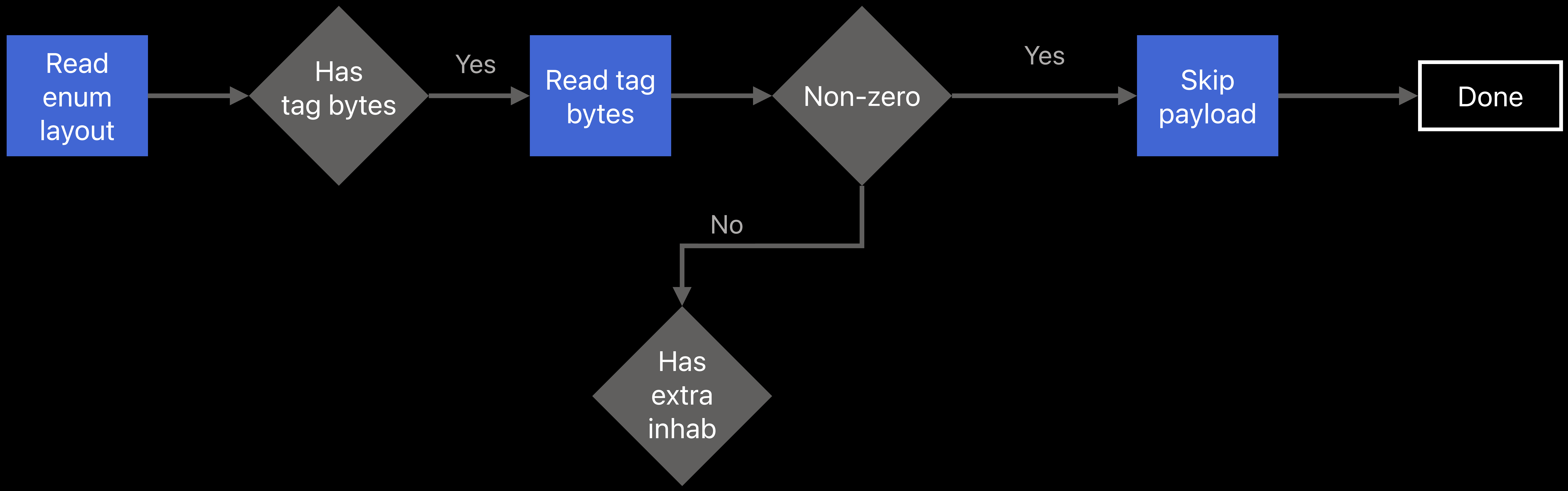
Runtime



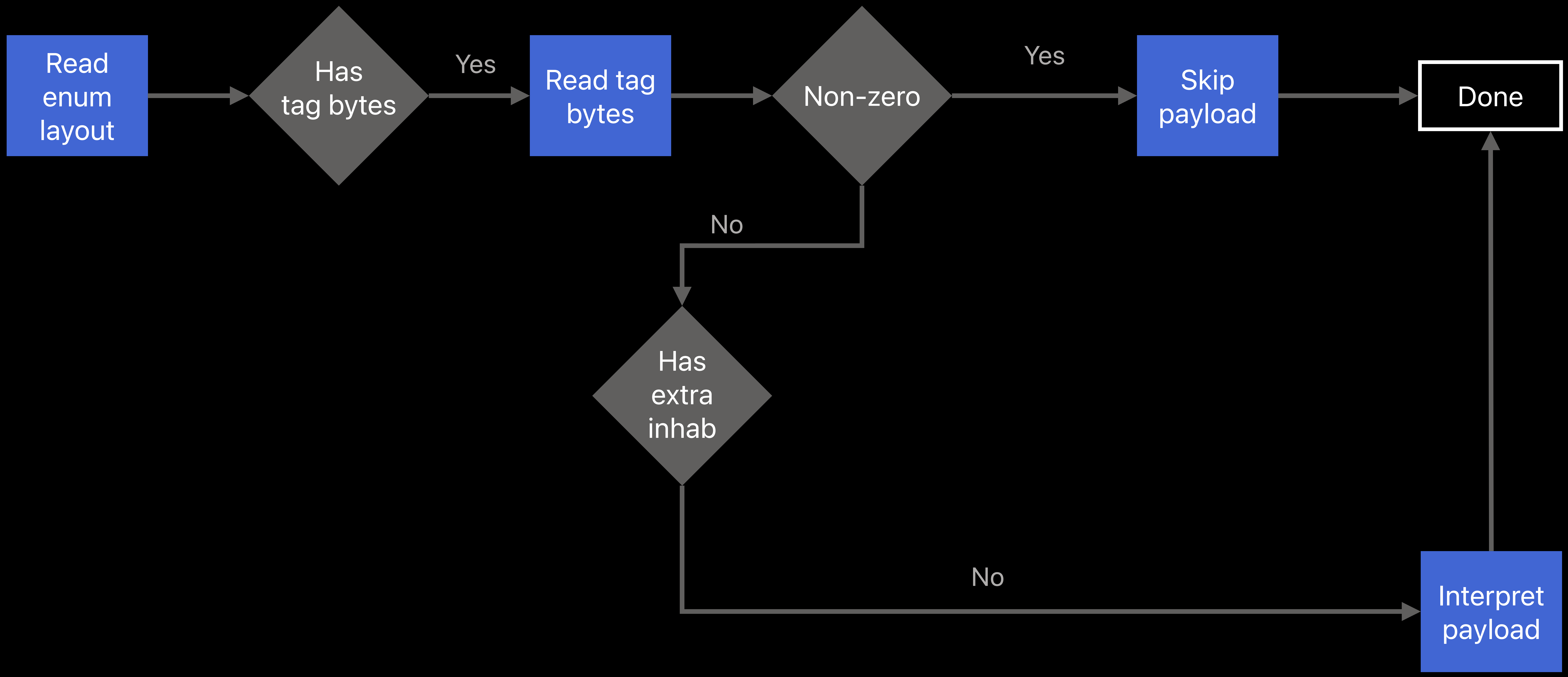
Runtime


Read enum layout
Has tag bytes
Yes
Read tag bytes
Non-zero
Yes
Skip payload
Done

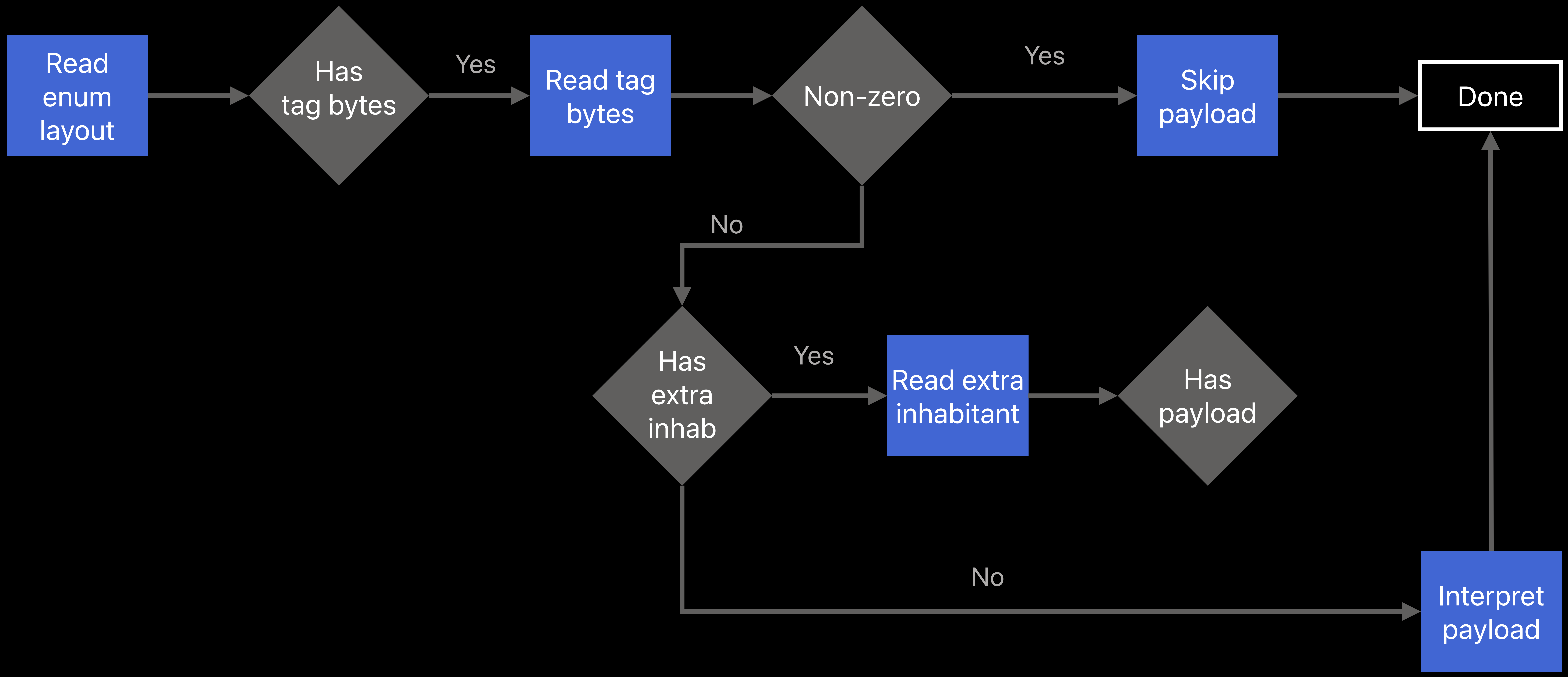
Runtime


Read enum layout
Has tag bytes
Yes
Read tag bytes
Non-zero
Yes
Skip payload
Done
No
Has extra inhab

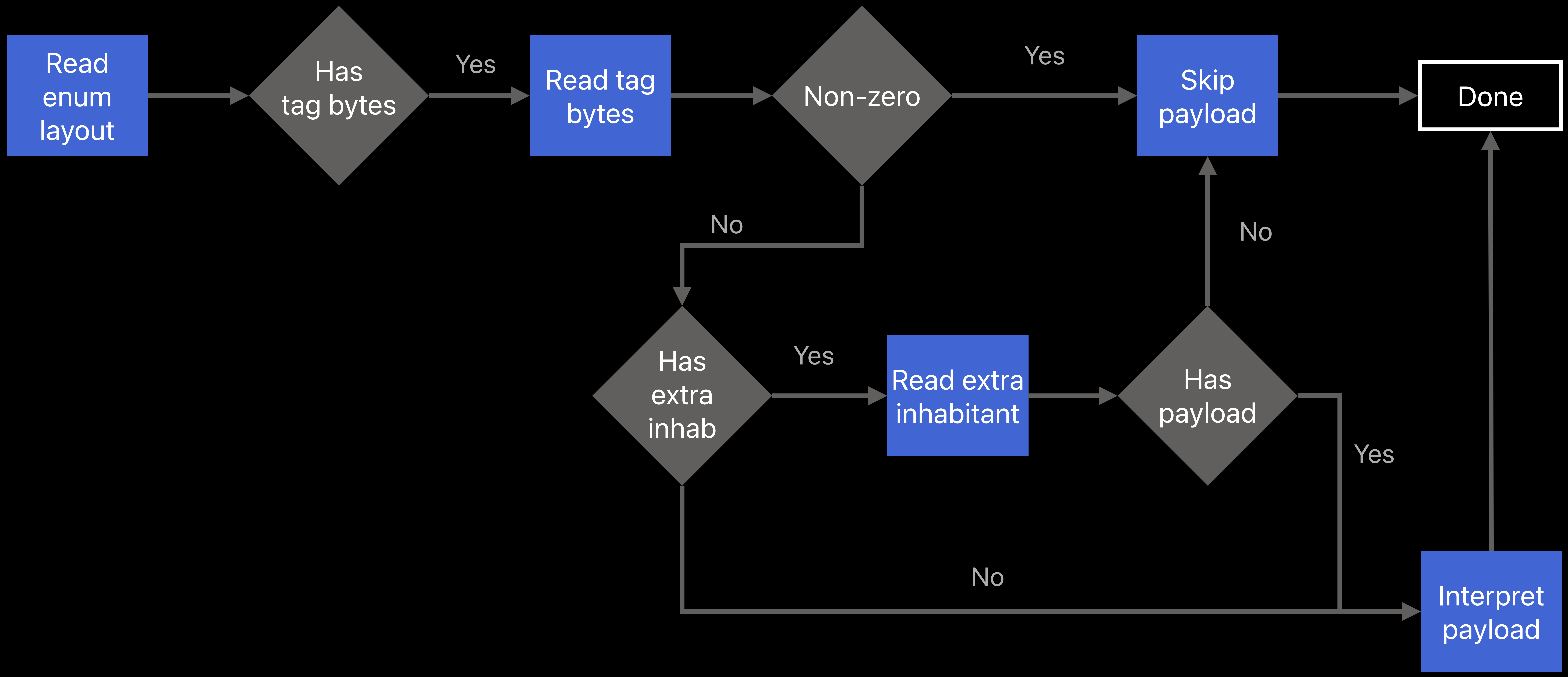
Runtime


Read enum layout
Has tag bytes
Yes
Read tag bytes
Non-zero
Yes
Skip payload
Done
No
Has extra inhab
No
Interpret payload

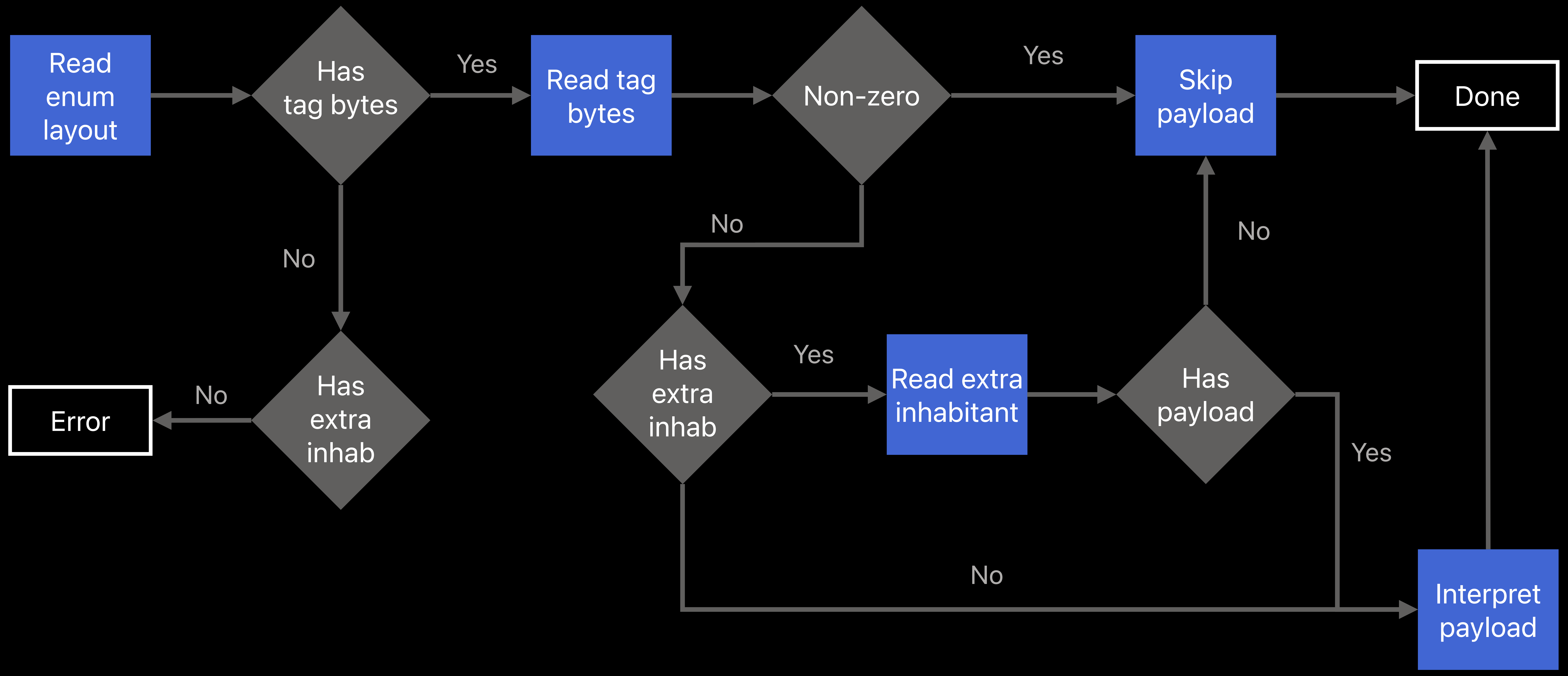
Runtime


Read enum layout
Has tag bytes
Yes
Read tag bytes
Non-zero
Yes
Skip payload
Done
No
Has extra inhab
Yes
Read extra inhabitant
Has payload
No
Interpret payload

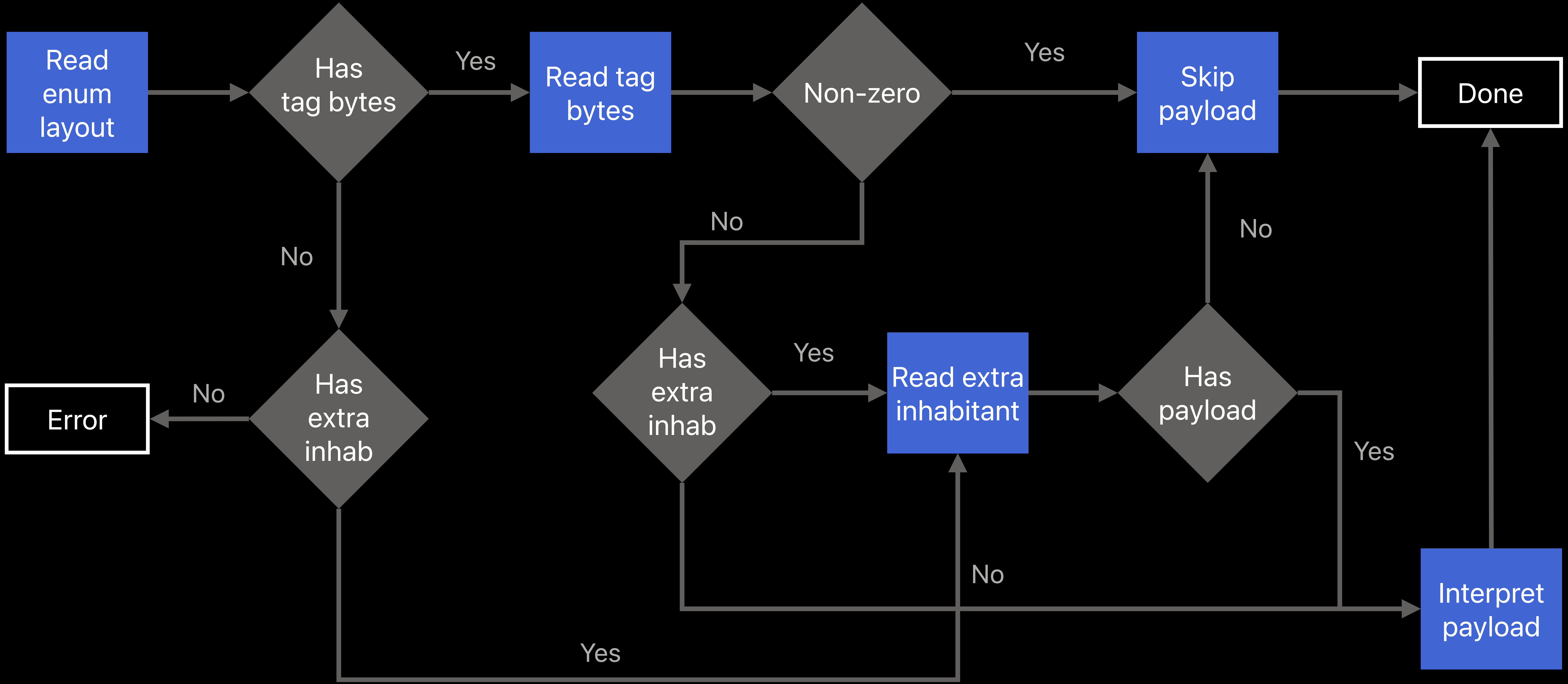
Runtime


Read enum layout
Has tag bytes
Yes
Read tag bytes
Non-zero
Yes
Skip payload
Done
No
No
Has extra inhab
Yes
Read extra inhabitant
Has payload
Yes
No
Interpret payload

Runtime


Read enum layout
Has tag bytes
Yes
Read tag bytes
Non-zero
Yes
Skip payload
Done
No
No
No
Error
No
Has extra inhab
Has extra inhab
Yes
Read extra inhabitant
Has payload
Yes
No
Interpret payload

Runtime

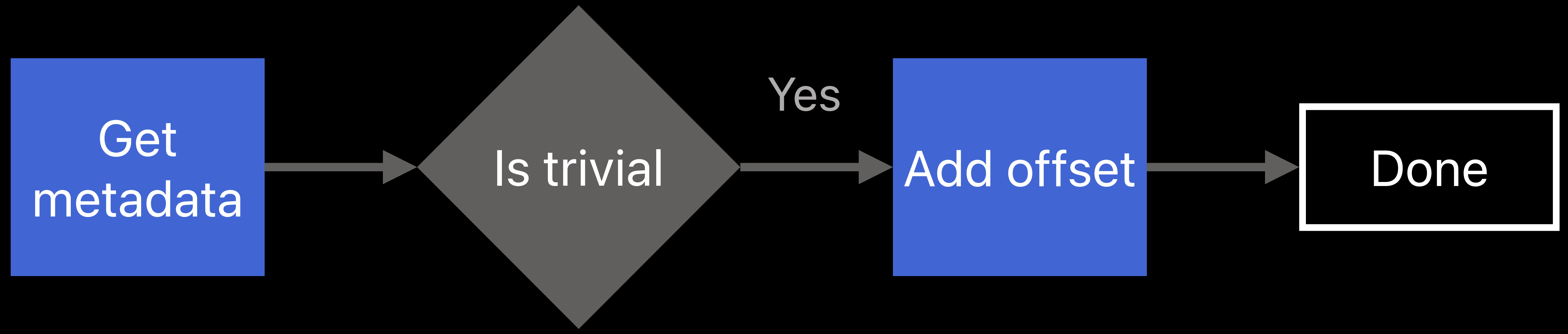

Read enum layout
Has tag bytes
Yes
Read tag bytes
Non-zero
Yes
Skip payload
Done
No
No
No
Error
No
Has extra inhab
Has extra inhab
Yes
Read extra inhabitant
Has payload
Yes
No
Yes
Interpret payload

Runtime instantiation

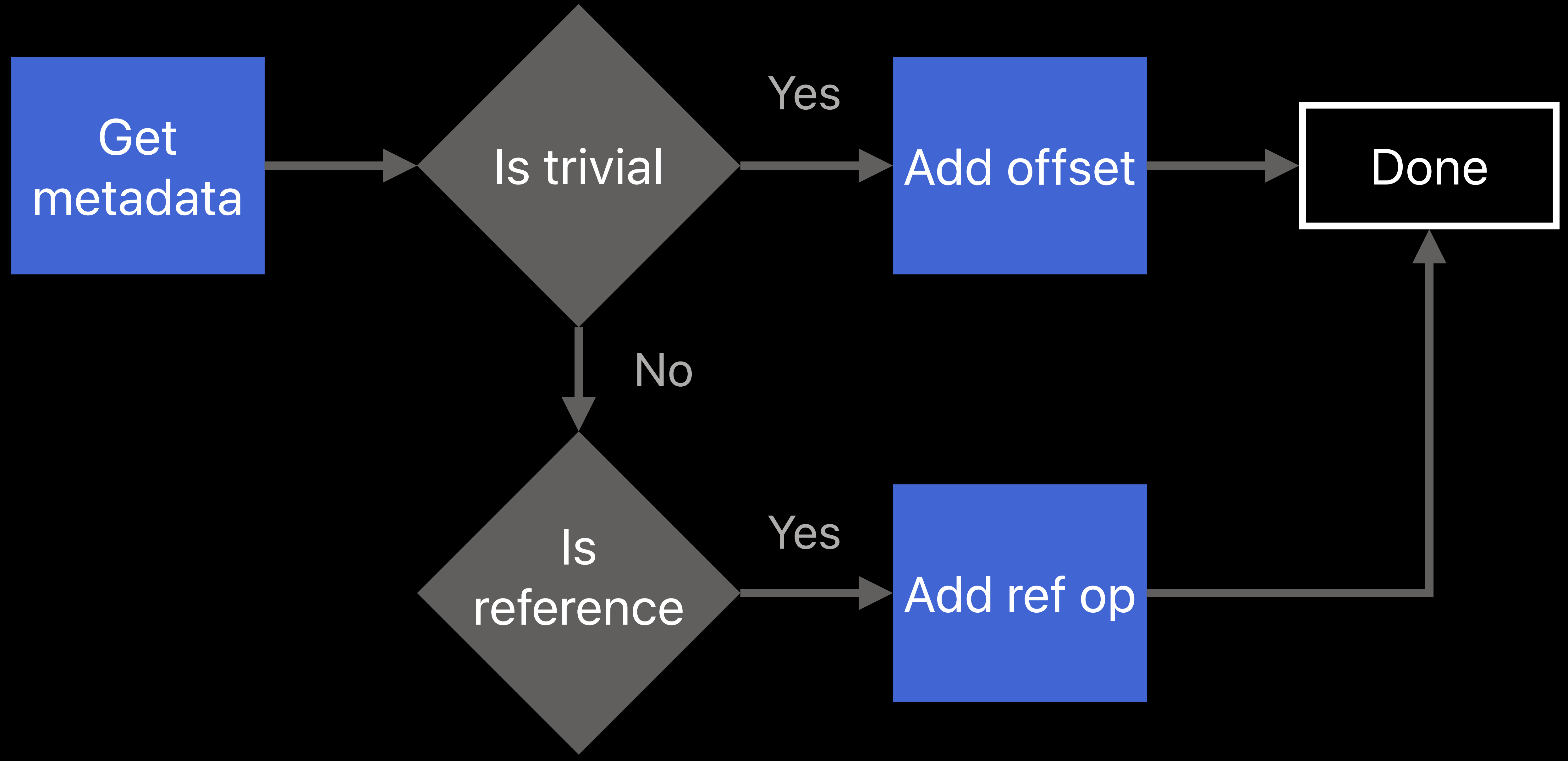
Runtime instantiation

Get metadata

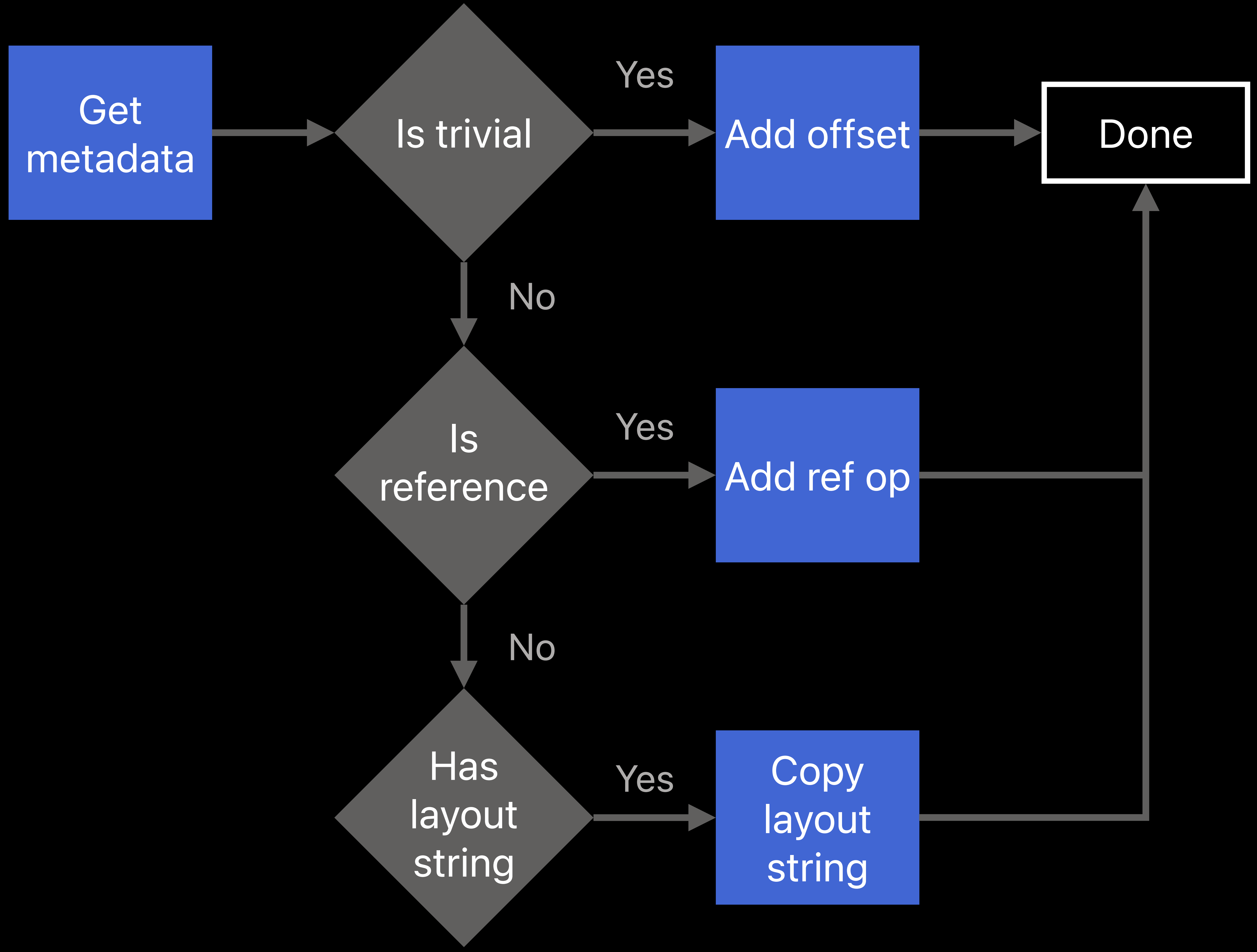
Runtime instantiation



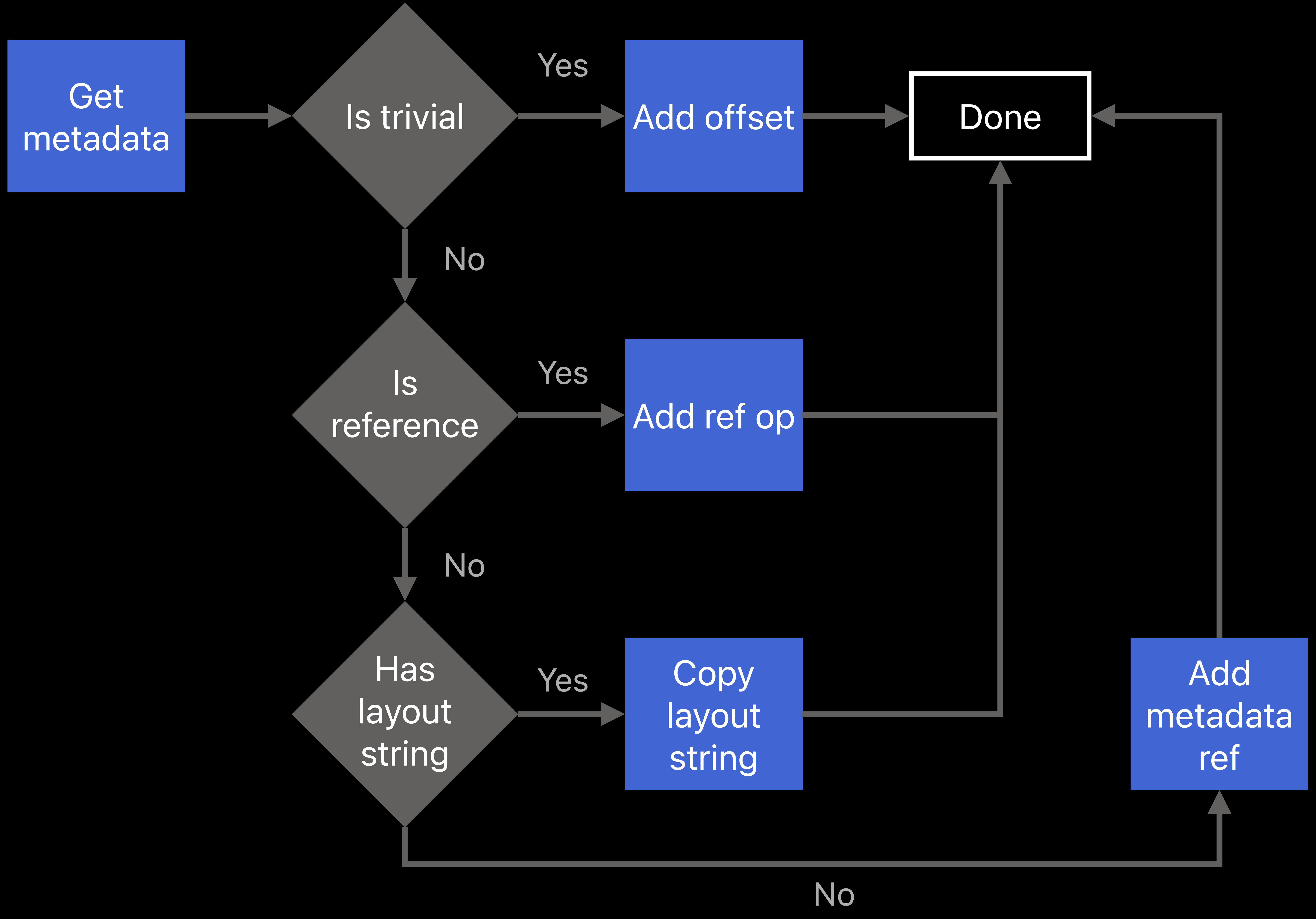
Runtime instantiation



Runtime instantiation



Runtime instantiation



Compatibility

Compatibility

```
ValueWitnessTable VWT {
    .destroy = &SomeStruct_destroy,
    .initializeWithCopy = &SomeStruct_initWithCopy,
    .initializeWithTake = &SomeStruct_initWithTake,
    .assignWithCopy = &SomeStruct_assignWithCopy,
    .assignWithTake = &SomeStruct_assignWithTake,
    // ...
};
```

Compatibility

```
ValueWitnessTable VWT {
    .destroy = &swift_generic_destroy,
    .initializeWithCopy = &swift_generic_initWithCopy,
    .initializeWithTake = &swift_generic_initWithTake,
    .assignWithCopy = &swift_generic_assignWithCopy,
    .assignWithTake = &swift_generic_assignWithTake,
    // ...
};
```

Compatibility

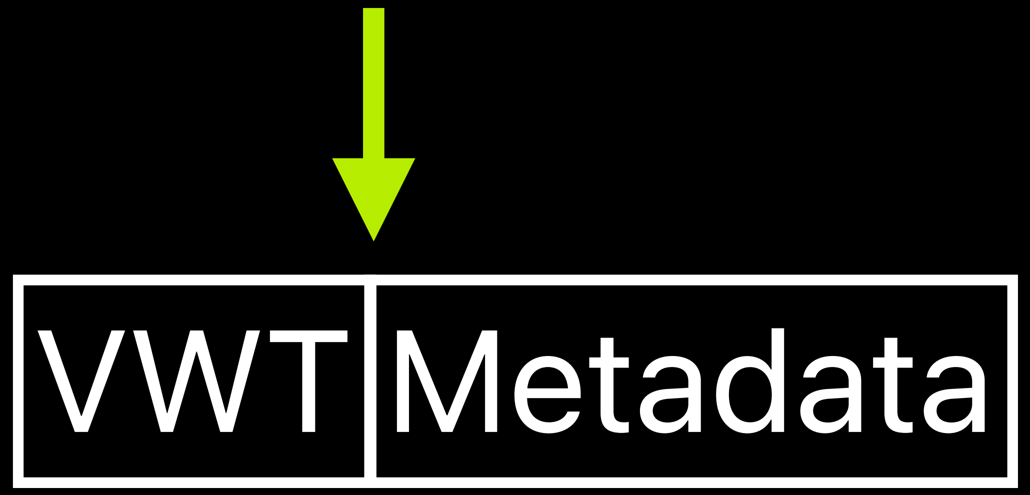
```
struct TypeMetadataHeader {

    const ValueWitnessTable *ValueWitnesses;
};
```

Compatibility

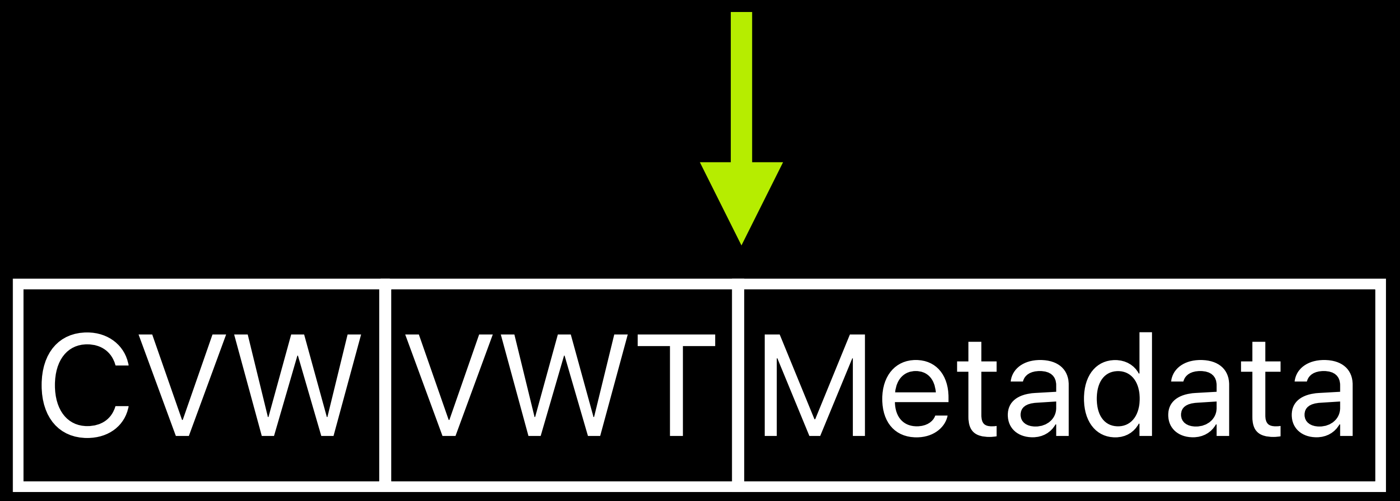
```
struct TypeMetadataHeader {

    const ValueWitnessTable *ValueWitnesses;
};
```



Compatibility

```
struct TypeMetadataHeader {
    const uint8_t *layoutString;
    const ValueWitnessTable *ValueWitnesses;
};
```

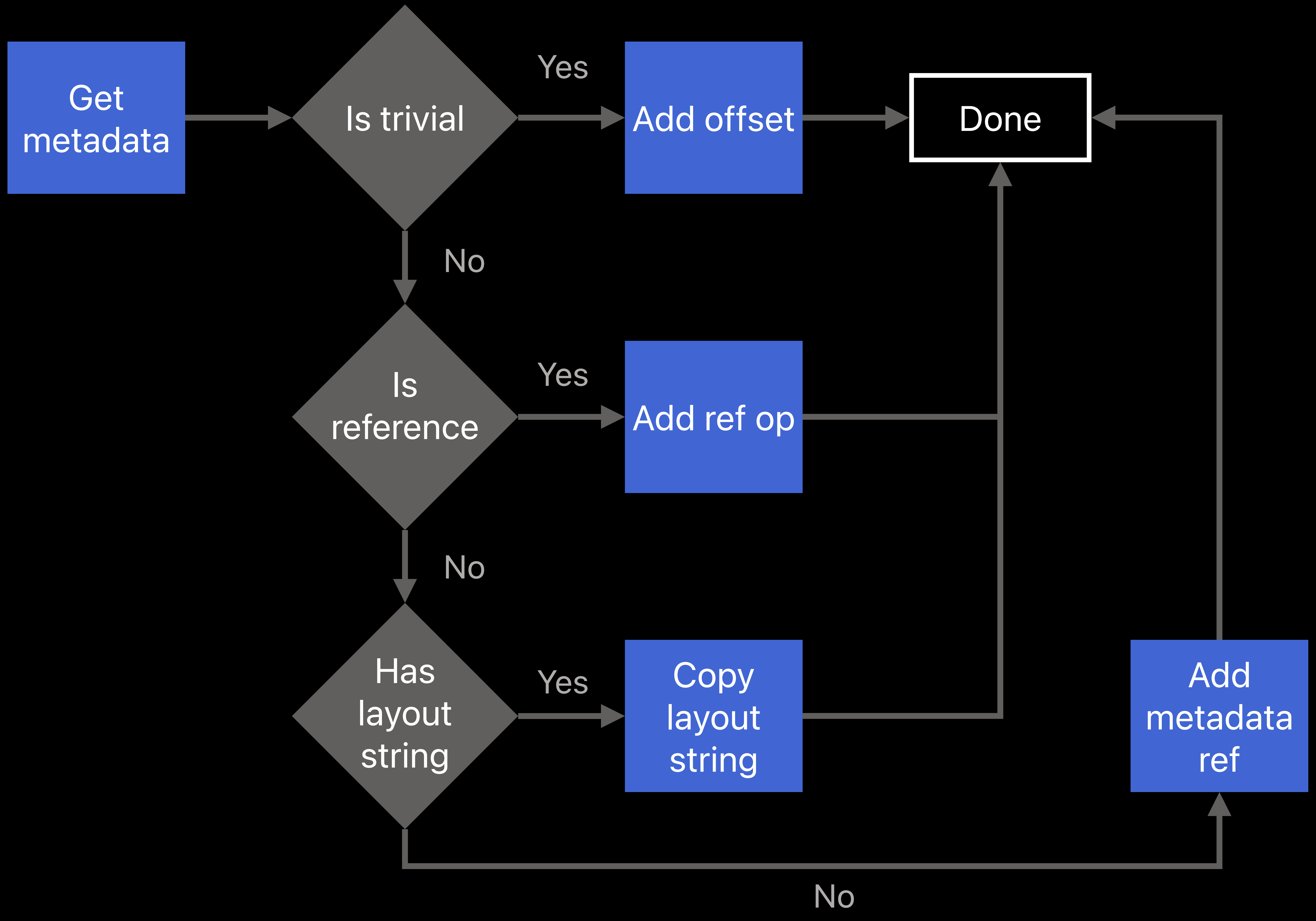


Compatibility

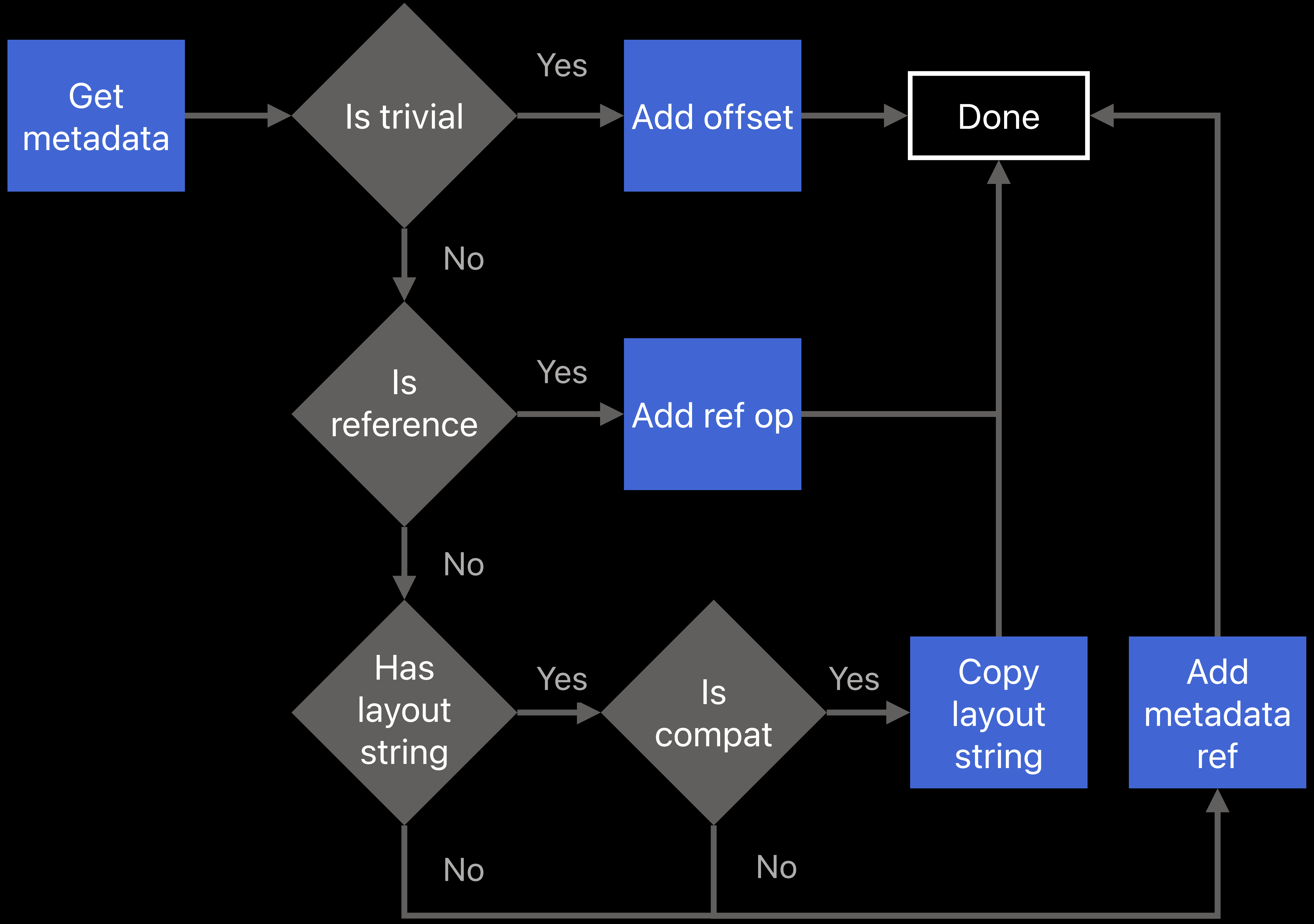
```
ValueWitnessTable VWT {
    .destroy = &swift_generic_destroy,
    .initializeWithCopy = &swift_generic_initWithCopy,
    .initializeWithTake = &swift_generic_initWithTake,
    .assignWithCopy = &swift_generic_assignWithCopy,
    .assignWithTake = &swift_generic_assignWithTake,
    // ...
};
```

```
ValueWitnessTable VWT {
    .destroy = &swift_generic_destroy_v2,
    .initializeWithCopy = &swift_generic_initWithCopy_v2,
    .initializeWithTake = &swift_generic_initWithTake_v2,
    .assignWithCopy = &swift_generic_assignWithCopy_v2,
    .assignWithTake = &swift_generic_assignWithTake_v2,
    // ...
};
```

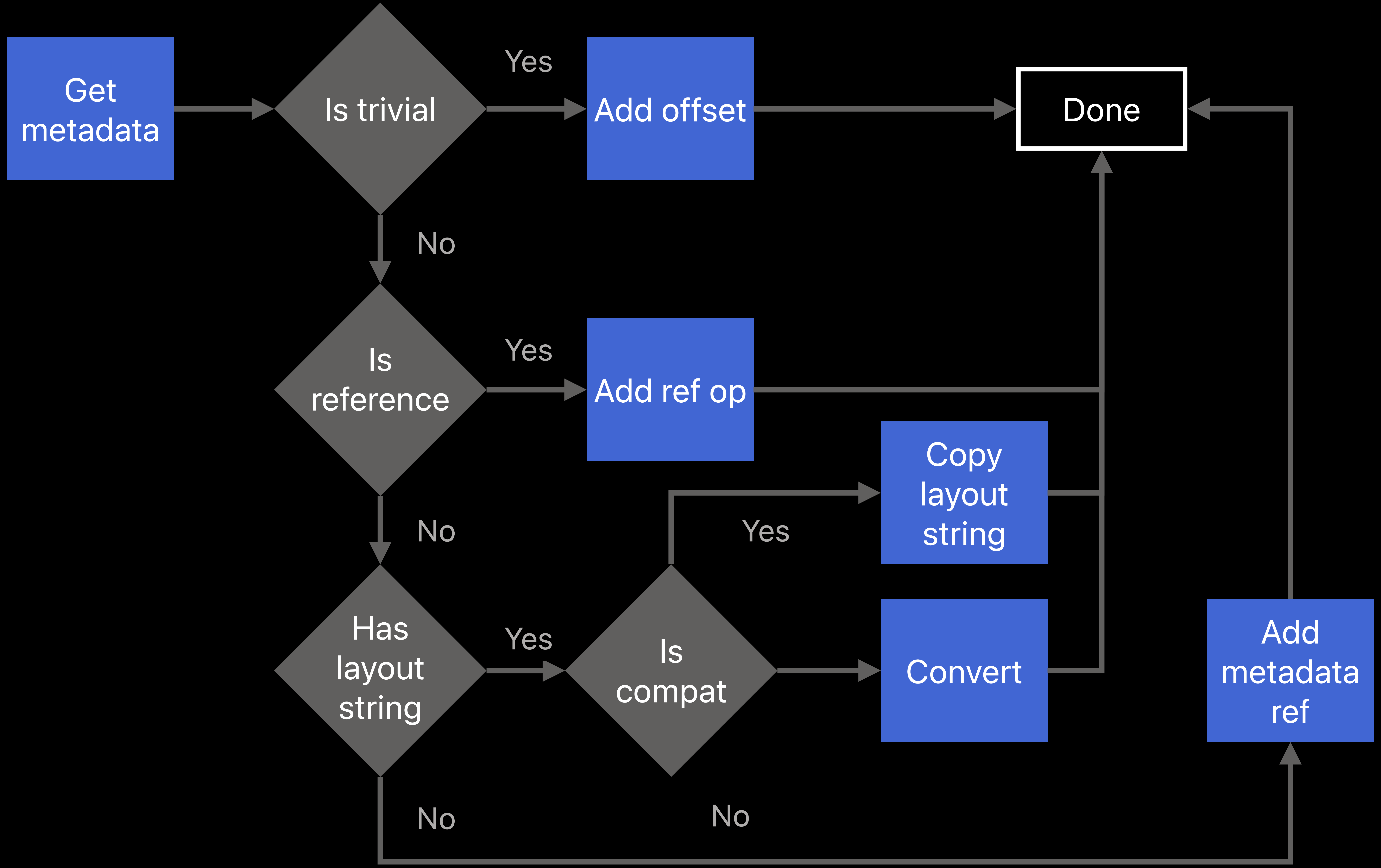
Compatibility



Compatibility



Compatibility



Performance impact

Performance impact

```swift
struct SomeStruct {
    let x: Int
    let y: SomeClass
}
```

~15% perf hit

Performance impact

```
struct SomeStruct {
    let x: Int
    let y: SomeClass
}

public struct GenericStruct<T> {
    let x: Int
    let y: T
}

let x: GenericStruct<GenericStruct<SomeStruct>>
```

~10% perf improvement

Performance impact

10% code size reductions

5-10% lower application startup time

Negligible real world performance difference

Special thanks