

Влияние Swift Runtime на производительность приложения

Про меня



Опыт

- 2 года в команде Яндекс Браузера и ПП
- Год в команде производительности Мобильного Банка в Т-Банке



Что люблю

- Копаться в кишках Swift-а
- Улучшать производительность кода
- Переписывать код других команд



Чем горжусь

- Своей командой
- Пониманием Swift Runtime
- Ускорением Мобильного Банка

Что мы делаем



О чем поговорим



Swift Runtime: Что под капотом

И почему так медленно?



Кто и как вызывает этот метод

Как эти методы работают,
потенциальные ловушки



Что мы сделали

И каких результатов добились



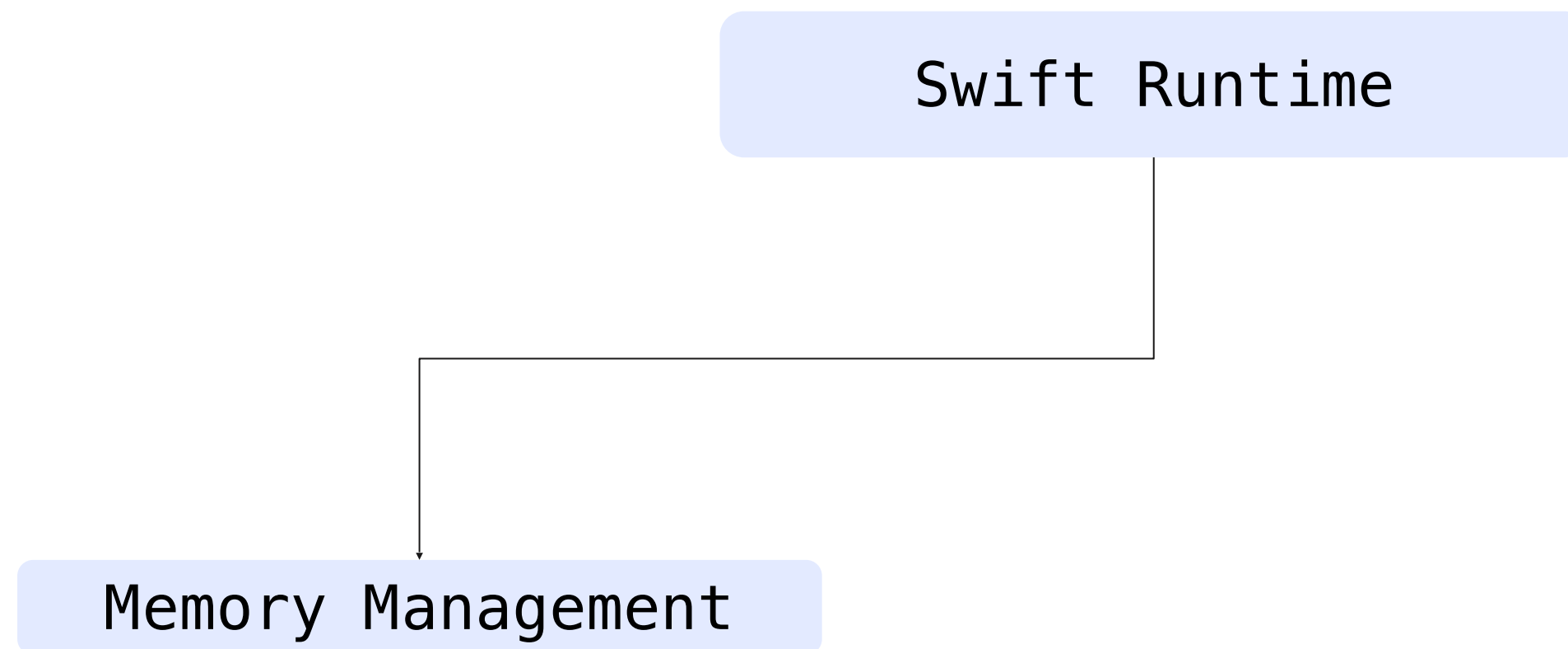
Рекомендации

Как Вы можете малой кровью ускорить
Ваше приложение

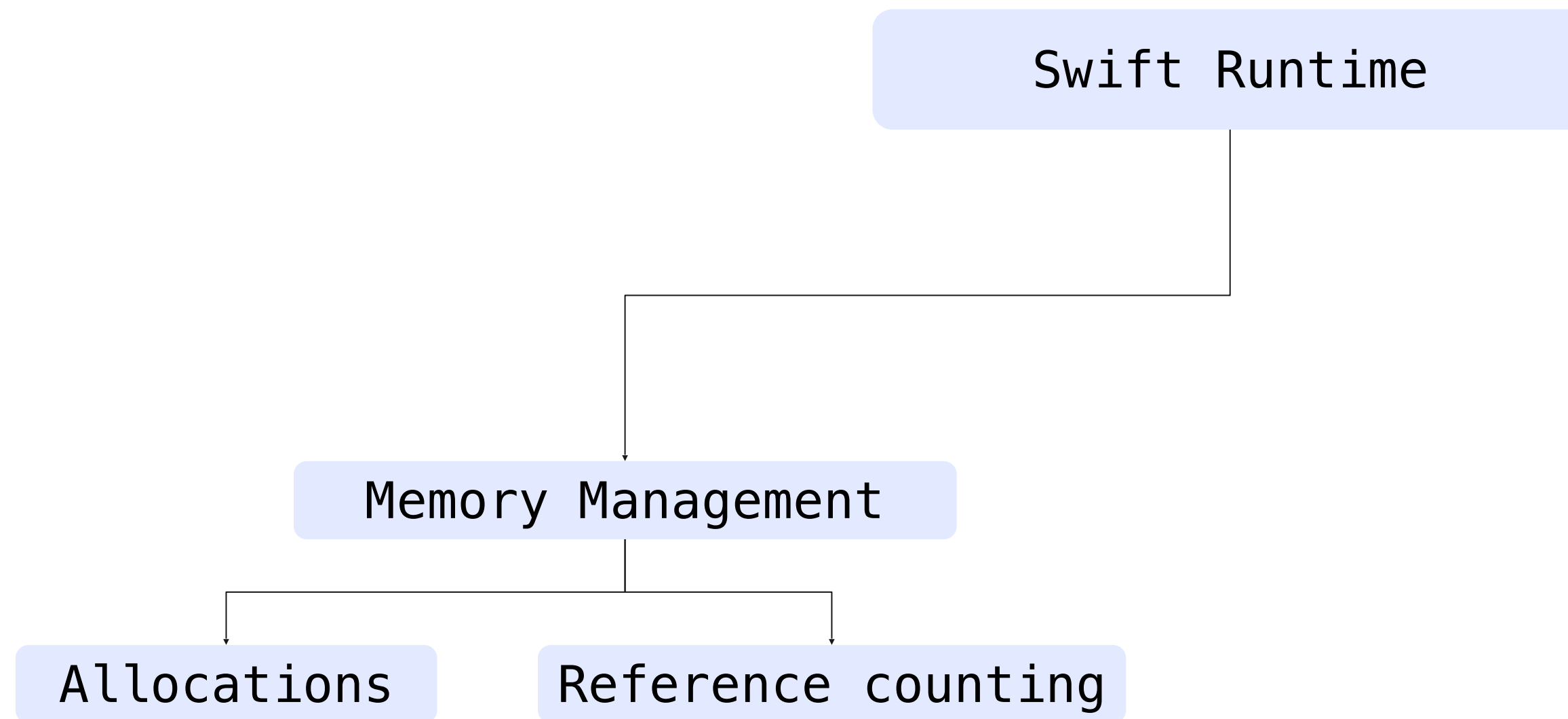
Swift Runtime

Swift Runtime

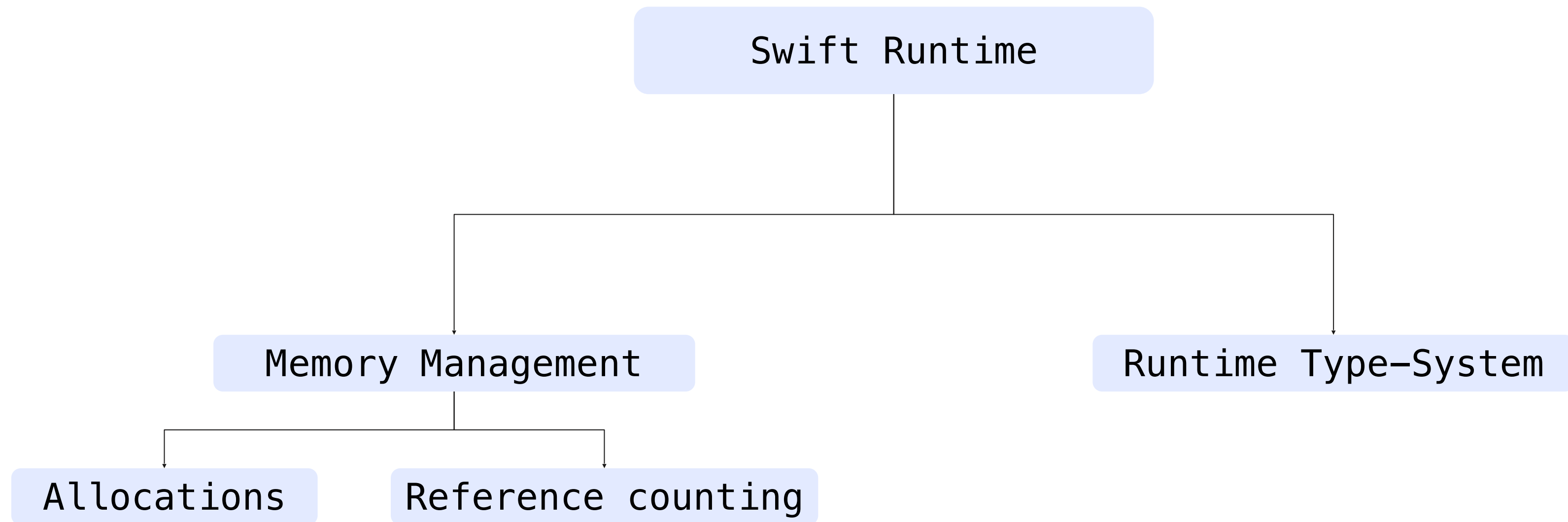
Swift Runtime



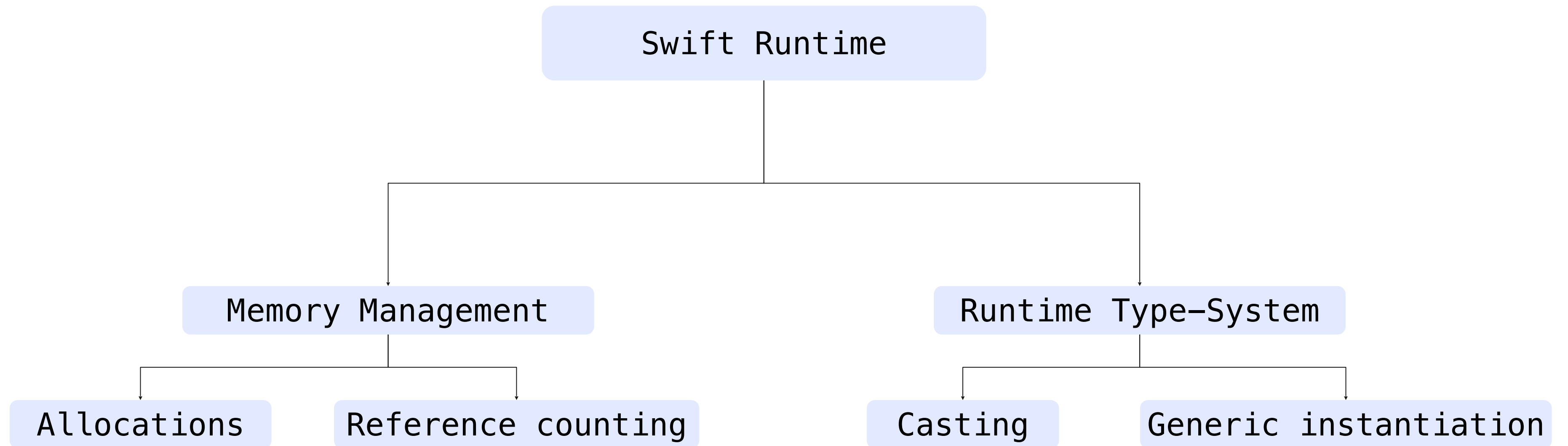
Swift Runtime



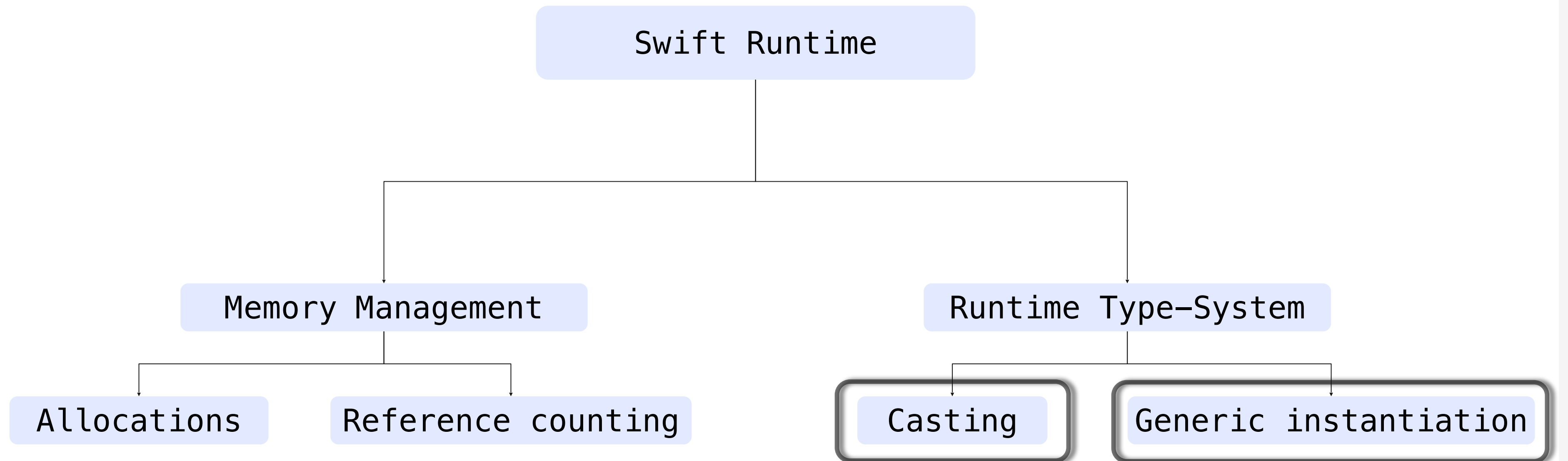
Swift Runtime






Swift Runtime






Swift Runtime



Один метод

| | | | | | |
|-----------|--------|-----------|---|---------------|--|
| 2.37 s | 100.0% | 0 s |  | ▼ Main Thread | 0x17d40b |
| 1.51 s | 63.8% | 1.51 s |  | > | swift_conformsToProtocolMaybeInstantiateSuperclasses |
| 125.00 ms | 5.2% | 125.00 ms |  | > | swift_conformsToProtocolMaybeInstantiateSuperclasses |

Один метод

| | | | | |
|-----------|--------|-----------|---|--|
| 2.37 s | 100.0% | 0 s |  | ▼ Main Thread 0x17d40b |
| 1.51 s | 63.8% | 1.51 s |  | > swift_conformsToProtocolMaybeInstantiateSuperclasses |
| 125.00 ms | 5.2% | 125.00 ms |  | > swift_conformsToProtocolMaybeInstantiateSuperclasses |

Один метод

```
static std::pair<const WitnessTable *, bool>
swift_conformsToProtocolMaybeInstantiateSuperclasses(
    const Metadata *const type,
    const ProtocolDescriptor *protocol,
    bool instantiateSuperclassMetadata
)
```

| | | | | | |
|-----------|--------|-----------|-----|---------------|--|
| 2.37 s | 100.0% | 0 s | | ∨ Main Thread | 0x17d40b |
| 1.51 s | 63.8% | 1.51 s | </> | > | swift_conformsToProtocolMaybeInstantiateSuperclasses |
| 125.00 ms | 5.2% | 125.00 ms | </> | > | swift_conformsToProtocolMaybeInstantiateSuperclasses |

Один метод

```
swift_conformsToProtocolMaybeInstantiateSuperclasses
```

Один метод

`swift_conformsToProtocolMaybeInstantiateSuperclasses`

`String(describing:)`



```
graph TD; A[swift_conformsToProtocolMaybeInstantiateSuperclasses] --> B[String(describing:)]
```

Один метод

`swift_conformsToProtocolMaybeInstantiateSuperclasses`

`String(describing:)`

`as? as!`

Один метод

`swift_conformsToProtocolMaybeInstantiateSuperclasses`

`String(describing:)`

`as? as!`

`Base<T: SomeProtocol>`

Один метод

```
swift_conformsToProtocolMaybeInstantiateSuperclasses
```

Один метод

```
swift_conformsToProtocolMaybeInstantiateSuperclasses
```

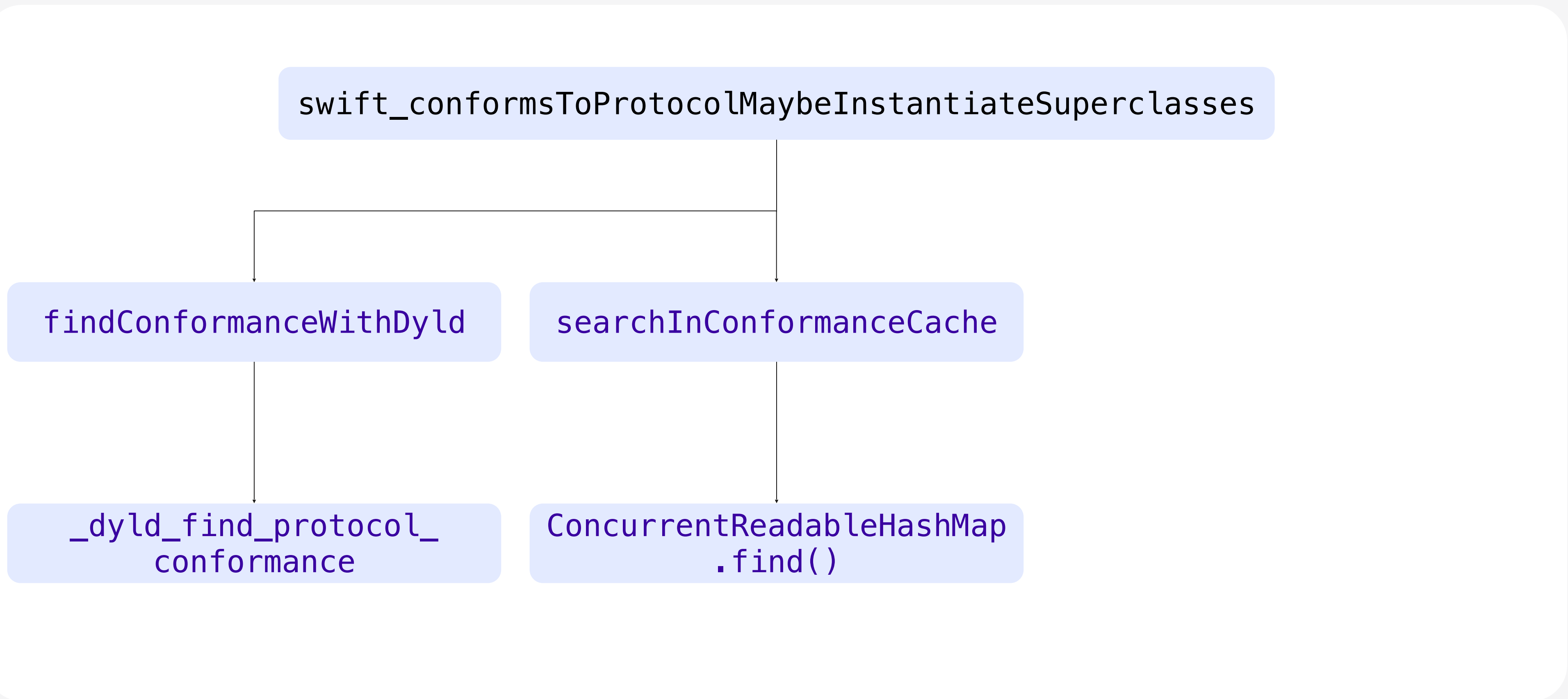
Один метод

`swift_conformsToProtocolMaybeInstantiateSuperclasses`

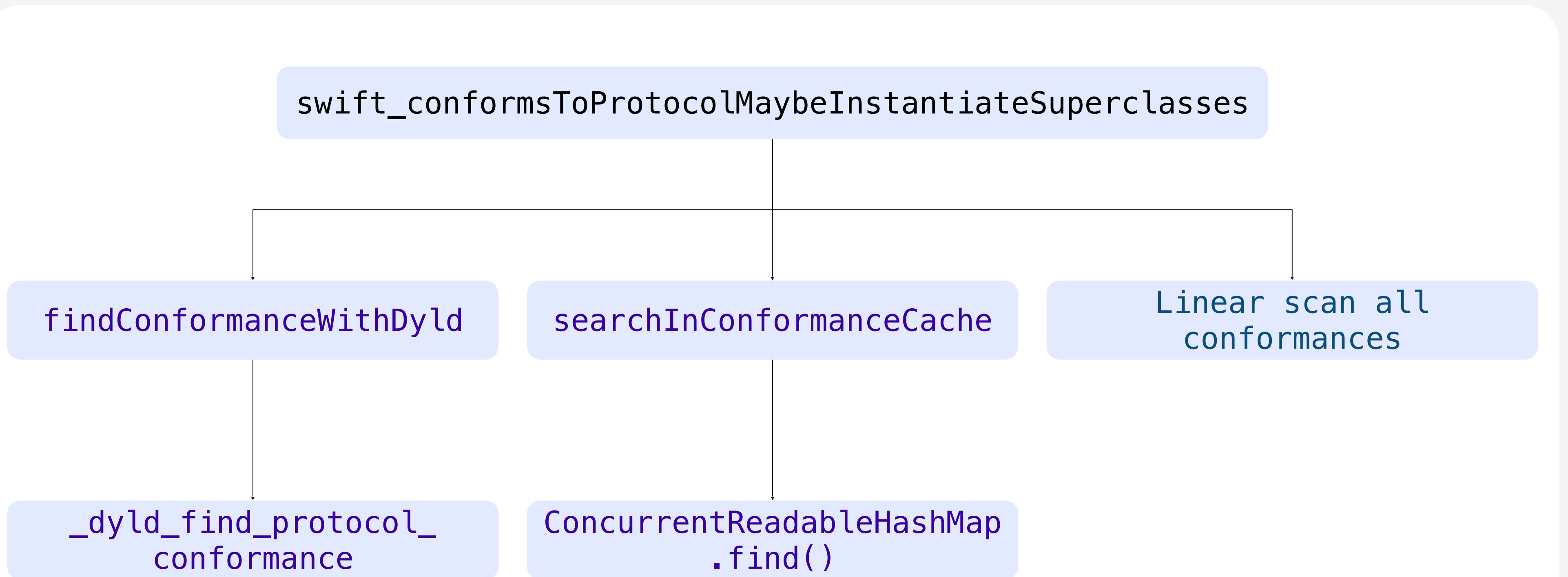
`findConformanceWithDyld`

`_dyld_find_protocol_
conformance`

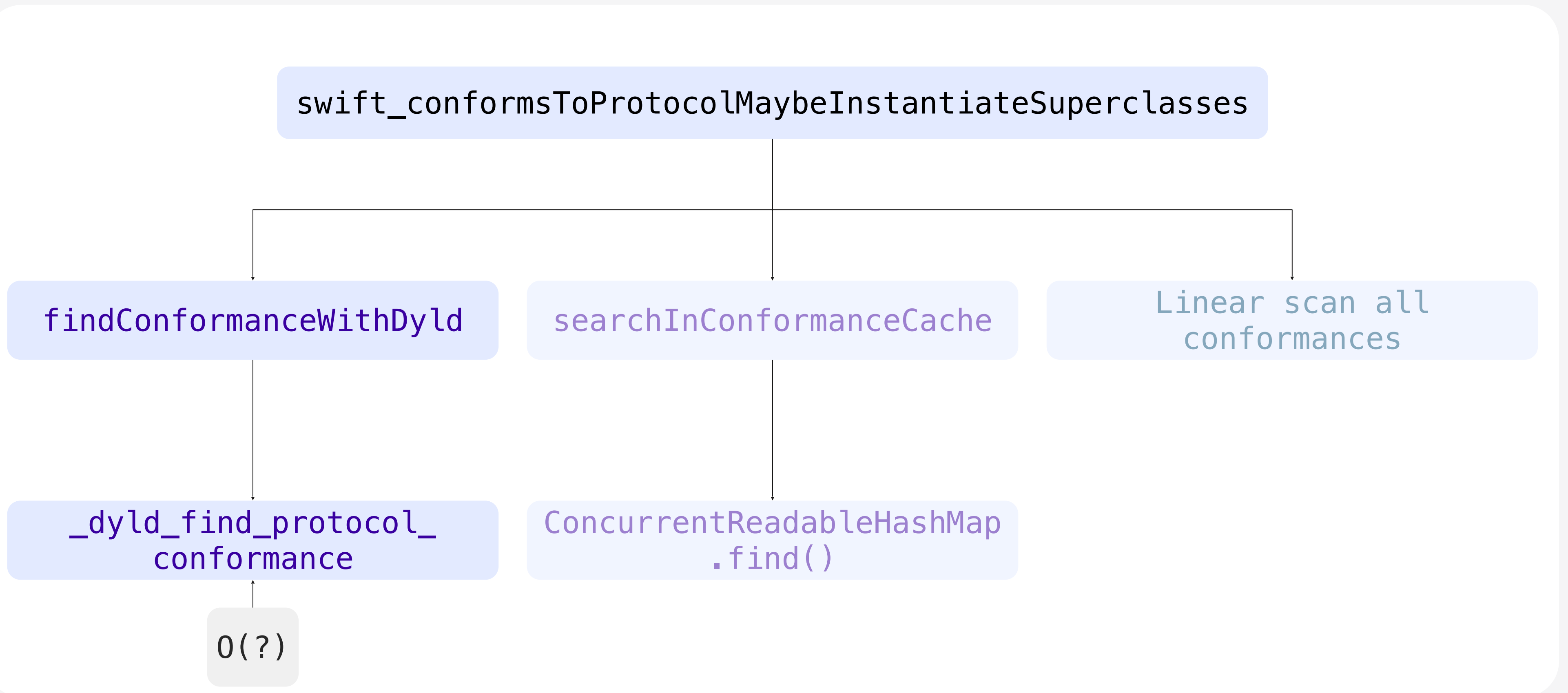
Один метод



Один метод



Один метод



Один метод

`swift_conformsToProtocolMaybeInstantiateSuperclasses`

`findConformanceWithDyld`

`searchInConformanceCache`

Linear scan all
conformances

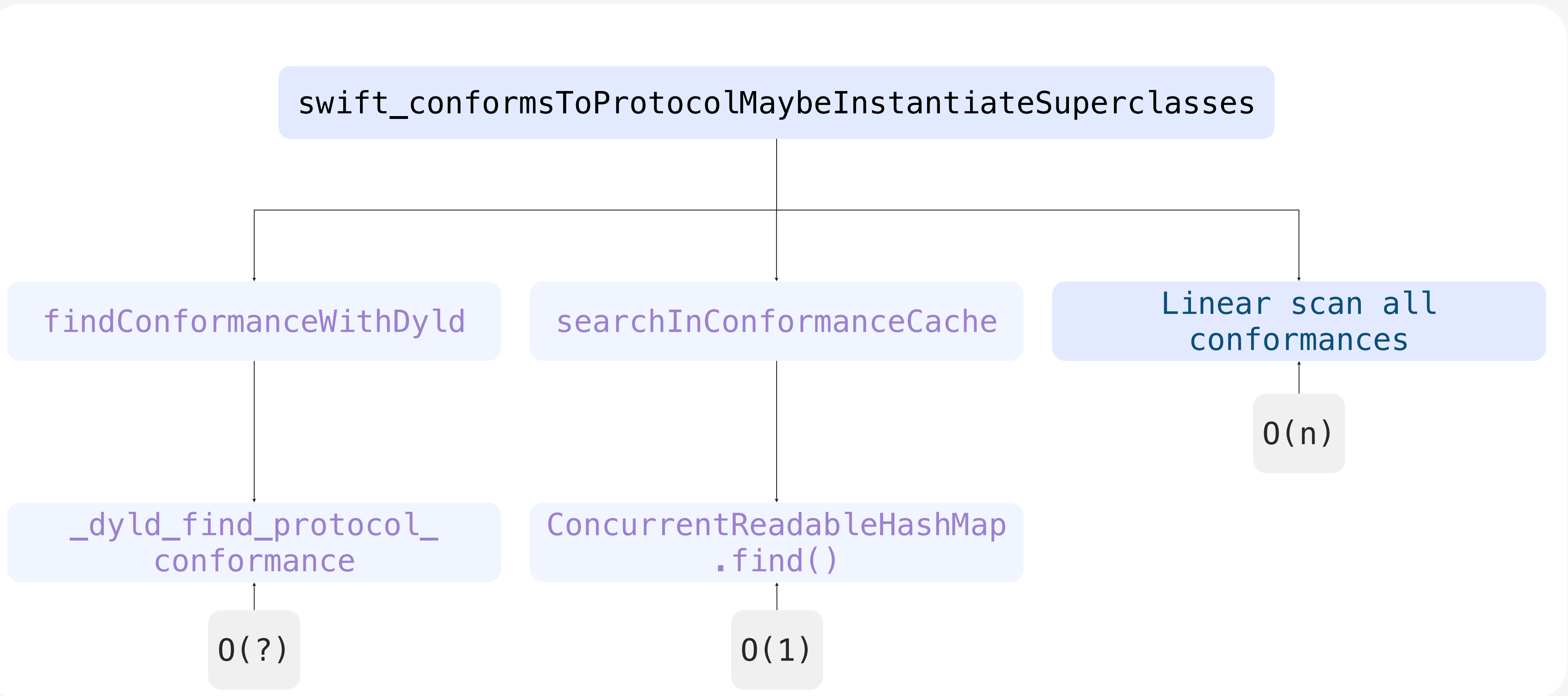
`_dyld_find_protocol_
conformance`

`ConcurrentReadableHashMap
.find()`

$O(?)$

$O(1)$

Один метод



Откуда линейный поиск? .app

 MobileBank.app

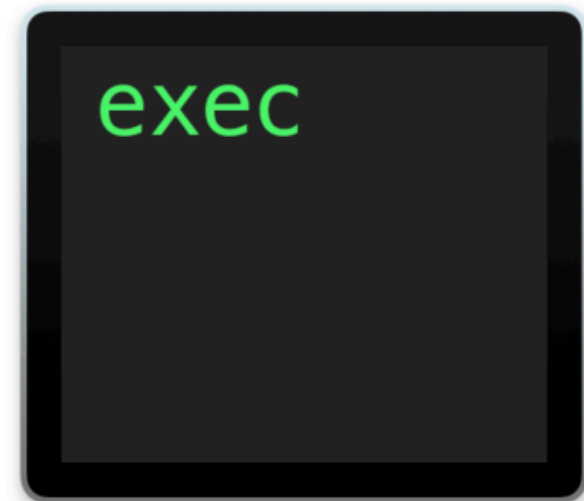
 MobileBank.swiftmodule >

Откуда линейный поиск? .app

MobileBank.app

MobileBank.swiftmodule >

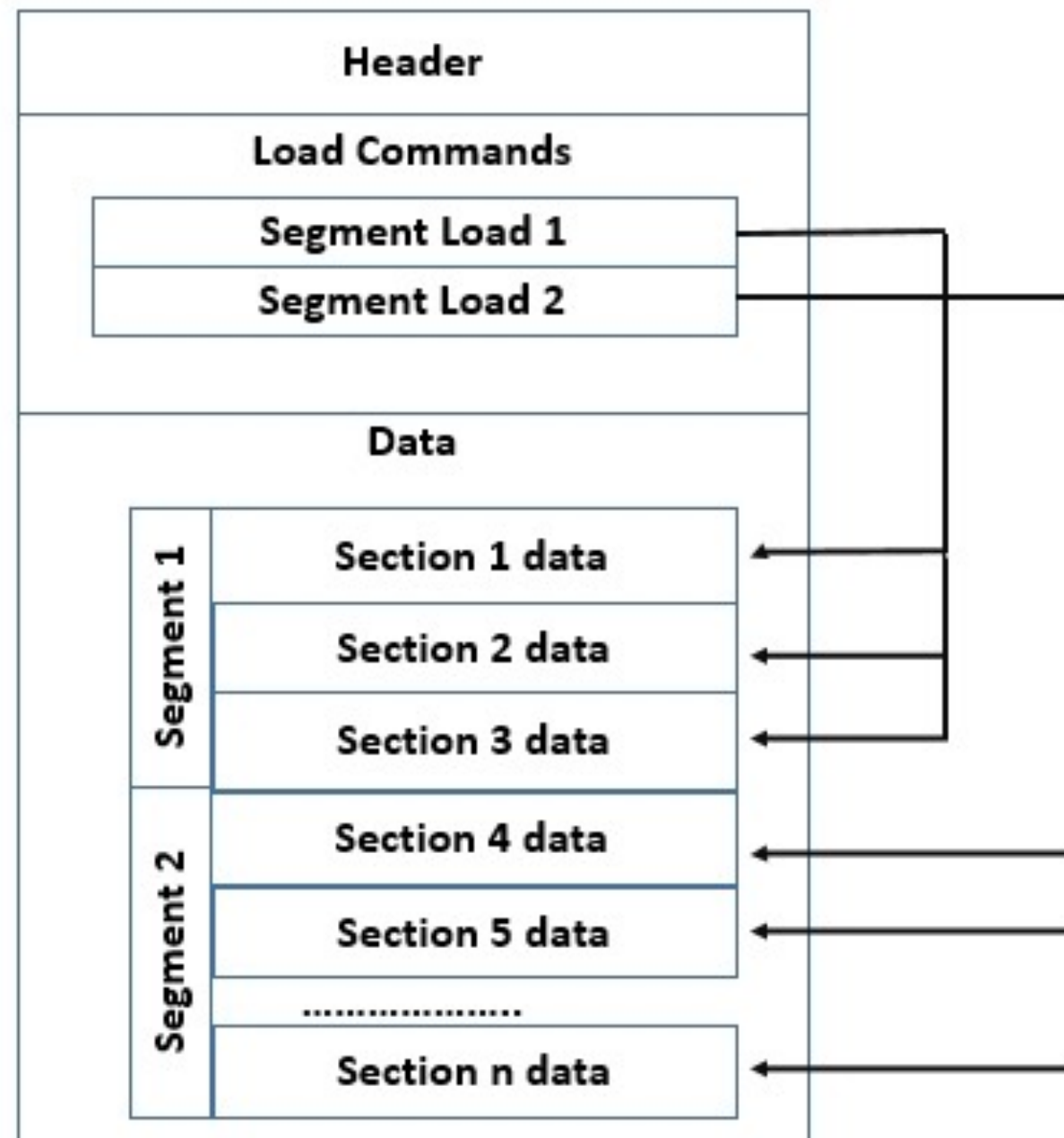
- MBUIConstantsResources.bundle
- MBUIFoundationResources.bundle
- MBUIModulesResources.bundle
- MBWidgetsLocalizationResources.bundle
- MeetingAppointmentResources.bundle
- MetalAccountsResources.bundle
- MobileBank**
- MobileBankCoreResources.bundle
- MobileBankPersistenceResources.bundle
- MobileBankUIResources.bundle
- MobileBankUISharedResources.bundle
- MobileBankUtilsResources.bundle



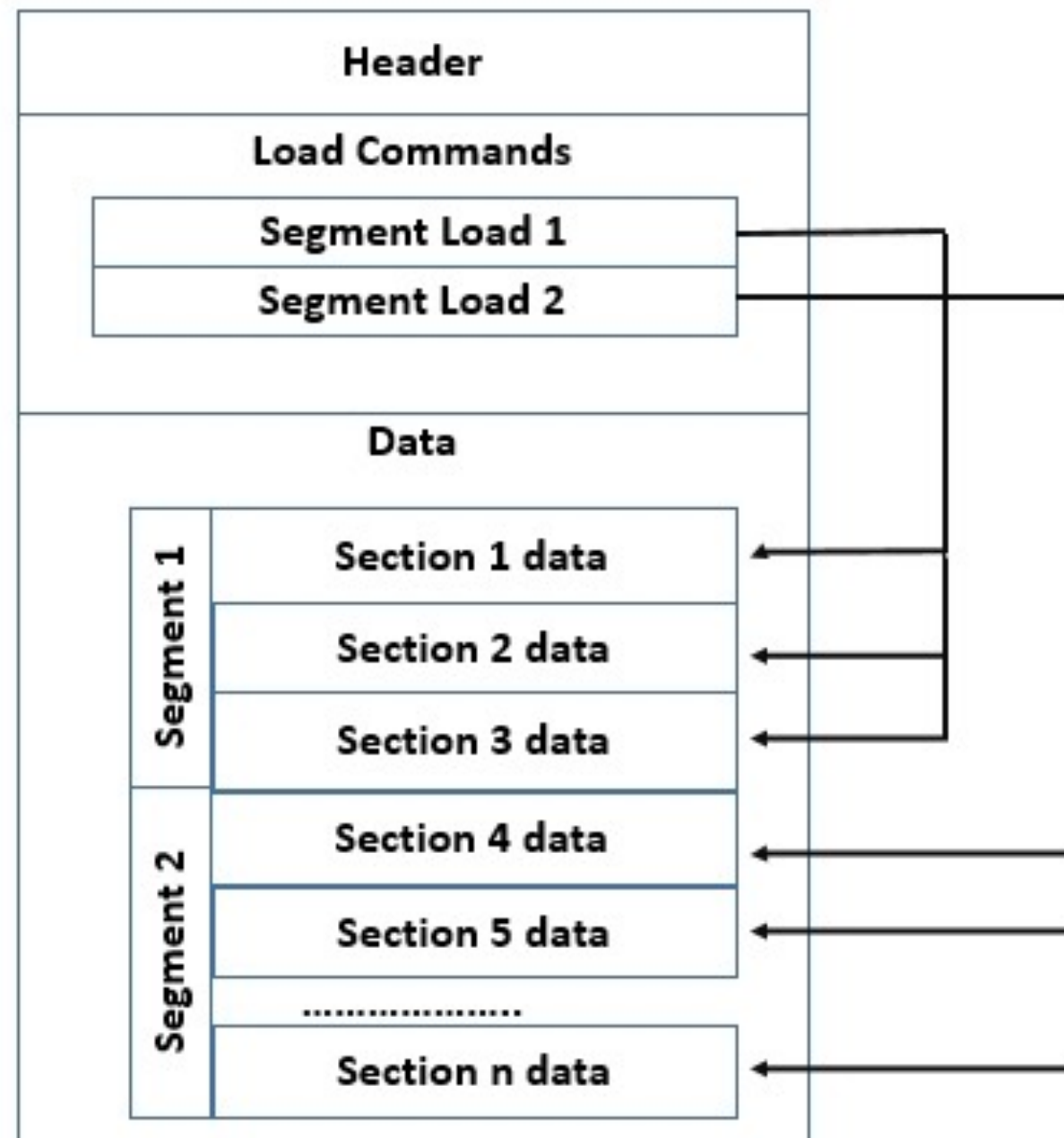
MobileBank

Unix Executable File - 1,19 GB

Откуда линейный поиск? Бинарь



Откуда линейный поиск? Бинарь



Доклад про Mach-O
и не только

Откуда линейный поиск?

```
$ jtool2 -l /Applications/Stocks.app/Contents/MacOS/Stocks
```

```
...
```

| | |
|------------------------------|--|
| Mem: 0x100002ca0-0x100020304 | __TEXT.__text (Normal) |
| Mem: 0x100020304-0x100020a90 | __TEXT.__stubs (Symbol Stubs) |
| Mem: 0x100020a90-0x100021734 | __TEXT.__stub_helper (Normal) |
| Mem: 0x100021740-0x100022c28 | __TEXT.__const |
| Mem: 0x100022c30-0x100024dc5 | __TEXT.__cstring (C-String Literals) |
| Mem: 0x100024dc5-0x100026630 | __TEXT.__objc_methname (C-String Literals) |
| Mem: 0x100026630-0x100026d54 | __TEXT.__swift5_typeref |
| Mem: 0x100026d60-0x100027061 | __TEXT.__swift5_reflstr |
| Mem: 0x100027064-0x1000274cc | __TEXT.__swift5_fieldmd |
| Mem: 0x1000274cc-0x100027608 | __TEXT.__swift5_capture |
| Mem: 0x100027608-0x100027668 | __TEXT.__swift5 ASSOCTY |
| Mem: 0x100027668-0x1000276f8 | __TEXT.__swift5_proto |
| Mem: 0x1000276f8-0x10002776c | __TEXT.__swift5_types |
| Mem: 0x10002776c-0x100027794 | __TEXT.__swift5_builtin |
| Mem: 0x100027794-0x1000277ac | __TEXT.__swift5_protos |
| Mem: 0x1000277ac-0x100027bb0 | __TEXT.__unwind_info |
| Mem: 0x100027bb0-0x100027ff8 | __TEXT.__eh_frame |

Откуда линейный поиск?

```
$ jtool2 -l /Applications/Stocks.app/Contents/MacOS/Stocks
```

```
...
```

```
Mem: 0x100002ca0-0x100020304    __TEXT.__text      (Normal)
Mem: 0x100020304-0x100020a90    __TEXT.__stubs     (Symbol Stubs)
Mem: 0x100020a90-0x100021734    __TEXT.__stub_helper (Normal)
Mem: 0x100021740-0x100022c28    __TEXT.__const
Mem: 0x100022c30-0x100024dc5    __TEXT.__cstring   (C-String Literals)
Mem: 0x100024dc5-0x100026630    __TEXT.__objc_methname (C-String Literals)
Mem: 0x100026630-0x100026d54    __TEXT.__swift5_typereref
Mem: 0x100026d60-0x100027061    __TEXT.__swift5_reflstr
Mem: 0x100027064-0x1000274cc    __TEXT.__swift5_fieldmd
Mem: 0x1000274cc-0x100027608    __TEXT.__swift5_capture
Mem: 0x100027608-0x100027668    __TEXT.__swift5 ASSOCTY
Mem: 0x100027668-0x1000276f8    __TEXT.__swift5_proto
Mem: 0x1000276f8-0x10002776c    __TEXT.__swift5_types
Mem: 0x10002776c-0x100027794    __TEXT.__swift5_builtin
Mem: 0x100027794-0x1000277ac    __TEXT.__swift5_protos
Mem: 0x1000277ac-0x100027bb0    __TEXT.__unwind_info
Mem: 0x100027bb0-0x100027ff8    __TEXT.__eh_frame
```

```
struct ProtocolConformanceDescriptor {
    let ProtocolDescriptor: Int32
    let NominalTypeDescriptor: Int32
    let ProtocolWitnessTable: Int32
    let ConformanceFlags: UInt32
}
```

Один метод

Linear scan all
conformances

Конформансы в бинаре

Один метод

Linear scan all
conformances

Конформансы в бинаре

Конформансы системным протоколам: Hashable, Equatable ...

Один метод

Linear scan all
conformances

Конформансы в бинаре

Конформансы системным протоколам: Hashable, Equatable ...


Конформансы вашим протоколам

С увеличением
кодовой базы
производительность
будет деградировать

В Мобильном Банке сейчас
около 130 тысяч конформансов
без учета динамических библиотек

```
>$ otool -l MobileBank.app/MobileBank |  
grep swift5_proto$ -A 10  
> ... size 0x00000000000007d34c # = 128211
```

String(describing:)



| | | | | |
|-----------|-------|-----------|--|--|
| 254.00 ms | 67.7% | 0 s |  | ∨ Main Thread 0x17d40b |
| 254.00 ms | 67.7% | 254.00 ms |  | > String.init<A>(describing:) libswiftCore.dylib |

String(describing:)

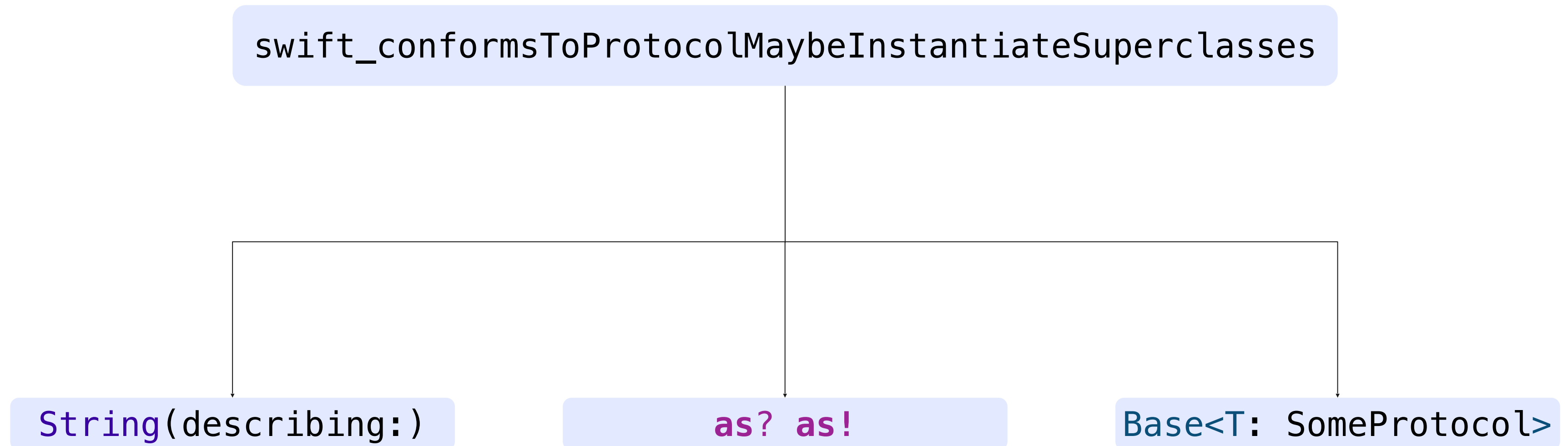
| | | | | | |
|-----------|-------|-----------|---|-----------------------------|--------------------|
| 254.00 ms | 67.7% | 0 s |  | Main Thread | 0x17d40b |
| 254.00 ms | 67.7% | 254.00 ms |  | String.init<A>(describing:) | libswiftCore.dylib |

String(describing:)

```
public init<Subject>(describing instance: Subject) {  
    self.init()  
    _print_unlocked(instance, &self)  
}
```

| | | | | | |
|-----------|-------|-----------|---|-------------------------------|--------------------|
| 254.00 ms | 67.7% | 0 s |  | Main Thread | 0x17d40b |
| 254.00 ms | 67.7% | 254.00 ms |  | > String.init<A>(describing:) | libswiftCore.dylib |

String(describing:)



String(describing:)

```
String(describing:)
```

String(describing:)

```
String(describing:)
```

String(describing:)

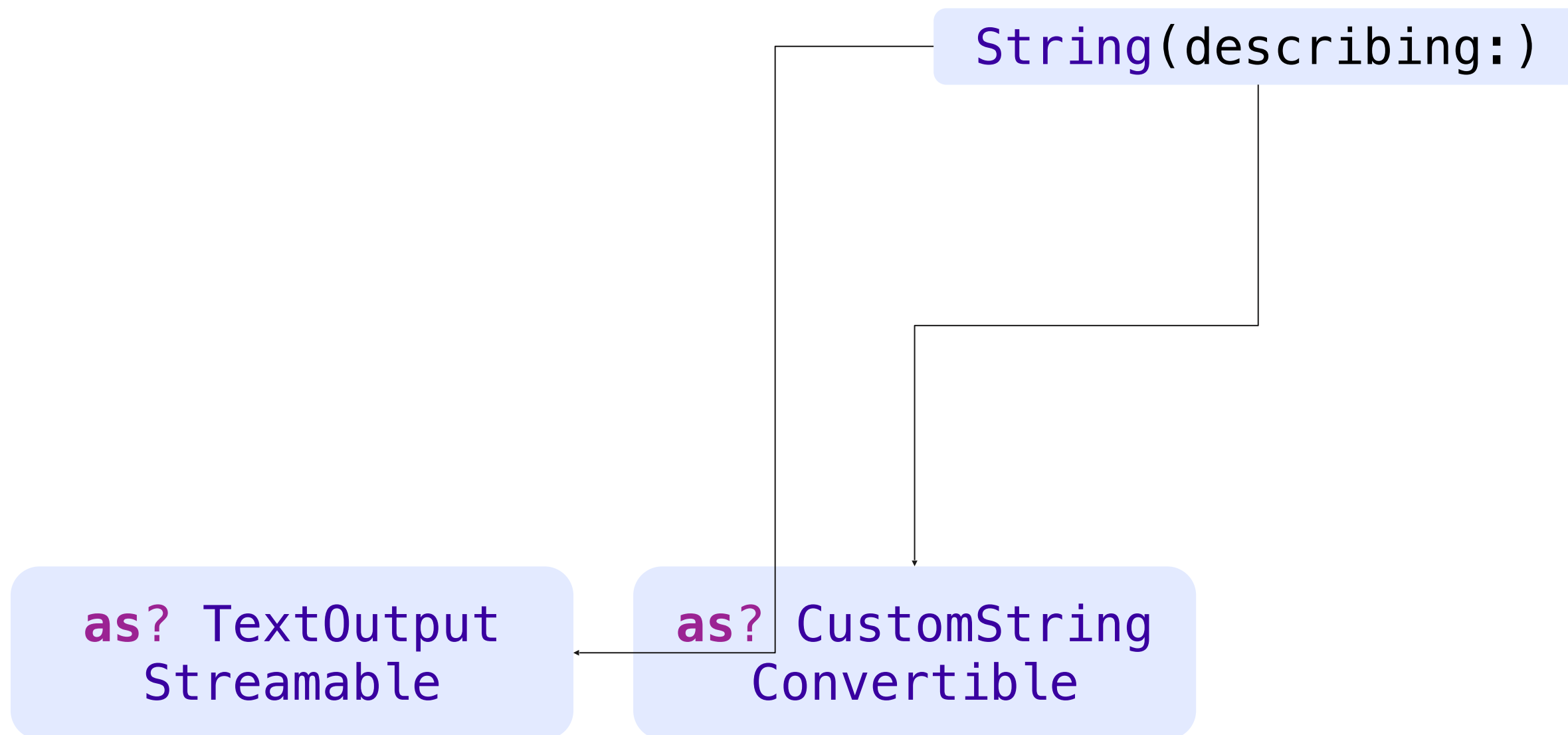
```
String(describing:)
```

String(describing:)

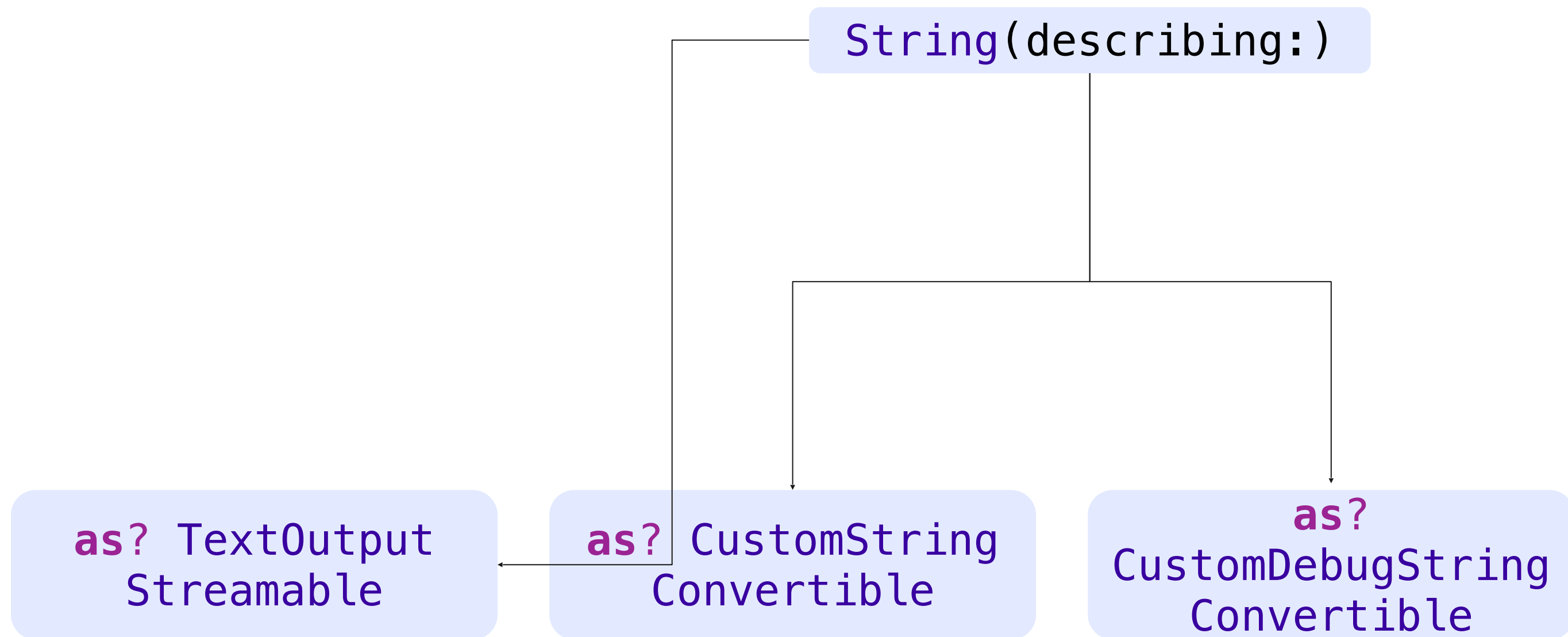
String(describing:)

as? TextOutput
Streamable

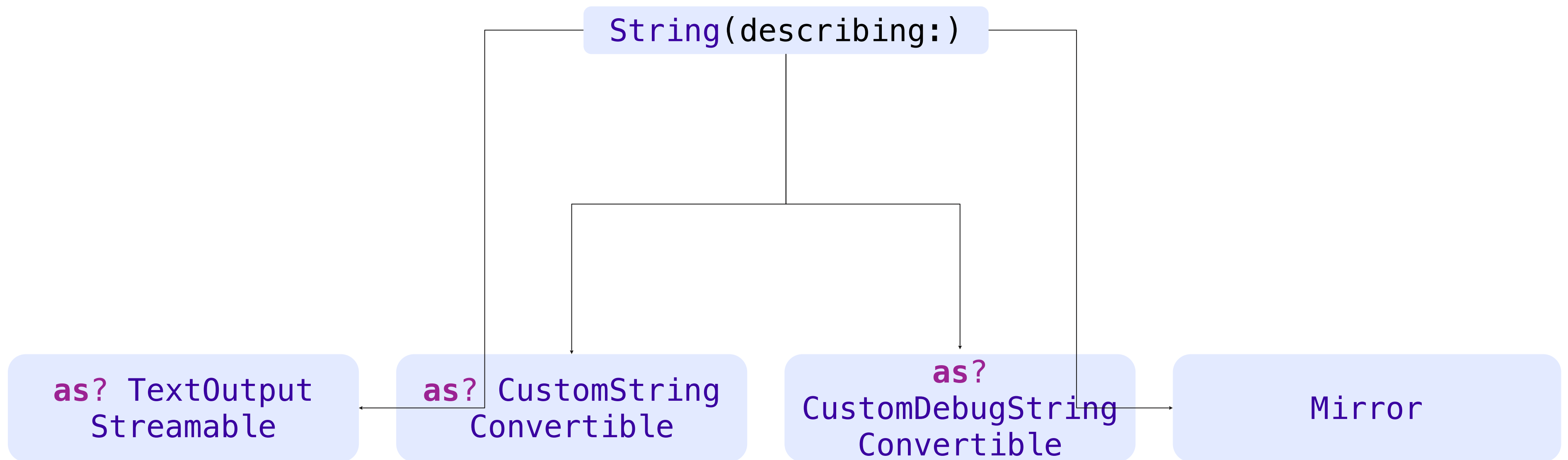
String(describing:)



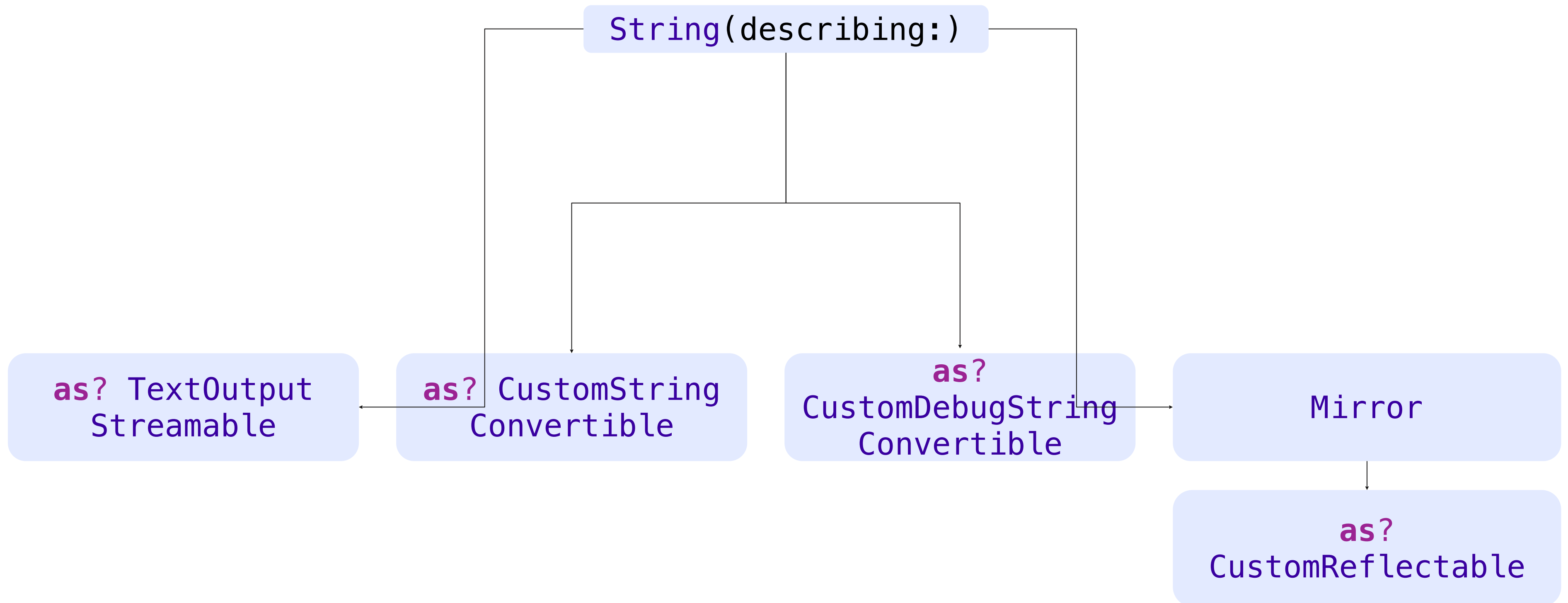
String(describing:)



String(describing:)



String(describing:)



String(describing:)



```
struct A {}  
  
// сколько раз вызовется swift_dynamicCast?  
let description = String(describing: A())
```

String(describing:)

```
struct A {}  
  
// сколько раз вызовется swift_dynamicCast?  
let description = String(describing: A())
```

4

String(describing:)



```
struct A {}  
  
// сколько раз вызовется swift_dynamicCast?  
let description = String(describing: A())
```

4



```
struct A {  
    let a: Int  
}  
  
// сколько раз вызовется swift_dynamicCast?  
let description = String(describing: A(...))
```

String(describing:)



```
struct A {}  
  
// сколько раз вызовется swift_dynamicCast?  
let description = String(describing: A())
```

4



```
struct A {  
    let a: Int  
}  
  
// сколько раз вызовется swift_dynamicCast?  
let description = String(describing: A(...))
```

6

String(describing:)



```
struct A {}  
  
// сколько раз вызовется swift_dynamicCast?  
let description = String(describing: A())
```

4



```
struct A {  
    let a: Int  
}  
  
// сколько раз вызовется swift_dynamicCast?  
let description = String(describing: A(...))
```

6

as? TextOutput
Streamable



as? CustomString
Convertible

String(describing:)



```
struct A {}  
  
// сколько раз вызовется swift_dynamicCast?  
let description = String(describing: A())
```

4



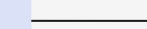
```
struct A {  
    let a: Int  
}  
  
// сколько раз вызовется swift_dynamicCast?  
let description = String(describing: A(...))
```

6



```
struct A {  
    let a: Int  
    let b: Int  
}  
  
// сколько раз вызовется swift_dynamicCast?  
let description = String(describing: A(...))
```

as? TextOutput
Streamable



as? CustomString
Convertible

String(describing:)



```
struct A {}  
  
// сколько раз вызовется swift_dynamicCast?  
let description = String(describing: A())
```

4



```
struct A {  
    let a: Int  
}  
  
// сколько раз вызовется swift_dynamicCast?  
let description = String(describing: A(...))
```

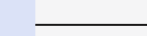
6



```
struct A {  
    let a: Int  
    let b: Int  
}  
  
// сколько раз вызовется swift_dynamicCast?  
let description = String(describing: A(...))
```

8

as? TextOutput
Streamable



as? CustomString
Convertible

String(describing:)

```
struct A {  
    let a: Int  
    let b: Int  
    let c: B  
}  
  
struct B {  
    let c: Int  
}  
  
// сколько раз вызовется swift_dynamicCast?  
let description = String(describing: A(...))
```

String(describing:)

```
struct A {  
    let a: Int  
    let b: Int  
    let c: B  
}  
  
struct B {  
    let c: Int  
}  
  
// сколько раз вызовется swift_dynamicCast?  
let description = String(describing: A(...))
```

14

String(describing:)

```
struct A {  
    let a: Int  
    let b: Int  
    let c: B  
}  
  
struct B {  
    let c: Int  
}  
  
// сколько раз вызовется swift_dynamicCast?  
let description = String(describing: A(...))
```

14

```
struct A {  
    var a: Int { 1 }  
}  
  
// сколько раз вызовется swift_dynamicCast?  
let description = String(describing: A(...))
```

String(describing:)



```
struct A {  
    let a: Int  
    let b: Int  
    let c: B  
}  
  
struct B {  
    let c: Int  
}  
  
// сколько раз вызовется swift_dynamicCast?  
let description = String(describing: A(...))
```

14



```
struct A {  
    var a: Int { 1 }  
}  
  
// сколько раз вызовется swift_dynamicCast?  
let description = String(describing: A(...))
```

4

as? as!



as? as!

| | | | |
|--------|--------|--------|--|
| 1.35 s | 100.0% | 0 s | ∨ MobileBank (4775) |
| 1.03 s | 75.9% | 0 s | ⏏ ∨ Main Thread 0x17d40b |
| 1.03 s | 75.9% | 1.03 s | </> > swift_dynamicCast libswiftCore.dylib → |

as? as!

| | | | |
|--------|--------|--------|--|
| 1.35 s | 100.0% | 0 s | ∨ MobileBank (4775) |
| 1.03 s | 75.9% | 0 s | ⏏ ∨ Main Thread 0x17d40b |
| 1.03 s | 75.9% | 1.03 s | </> > swift_dynamicCast libswiftCore.dylib → |

>60% времени, уходящего на swift_conformsToProtocol

as? as!

swift_conformsToProtocolMaybeInstantiateSuperclasses

String(describing:)

as? as!

Base<T: SomeProtocol>

as? as!

as? as!

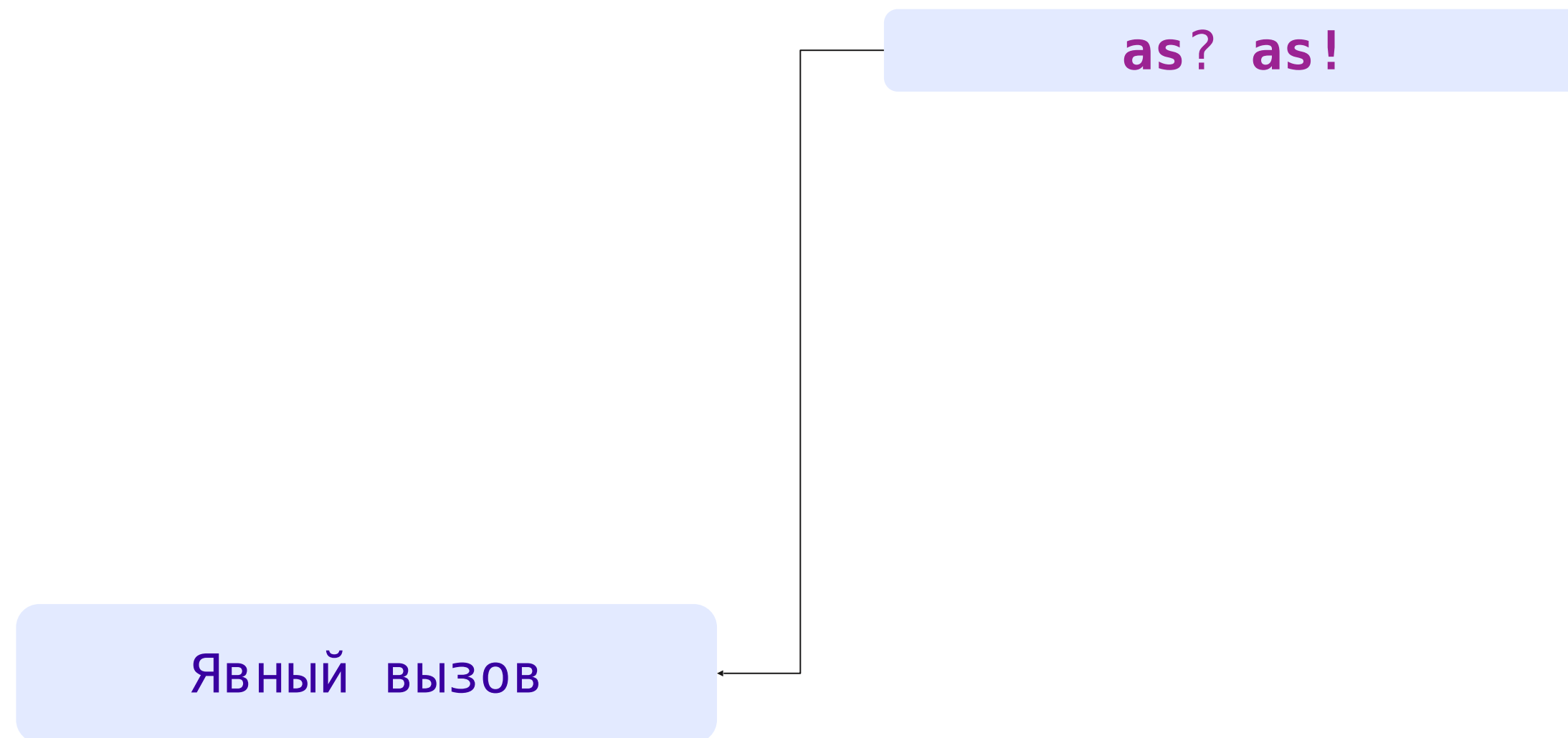
as? as!

as? as!

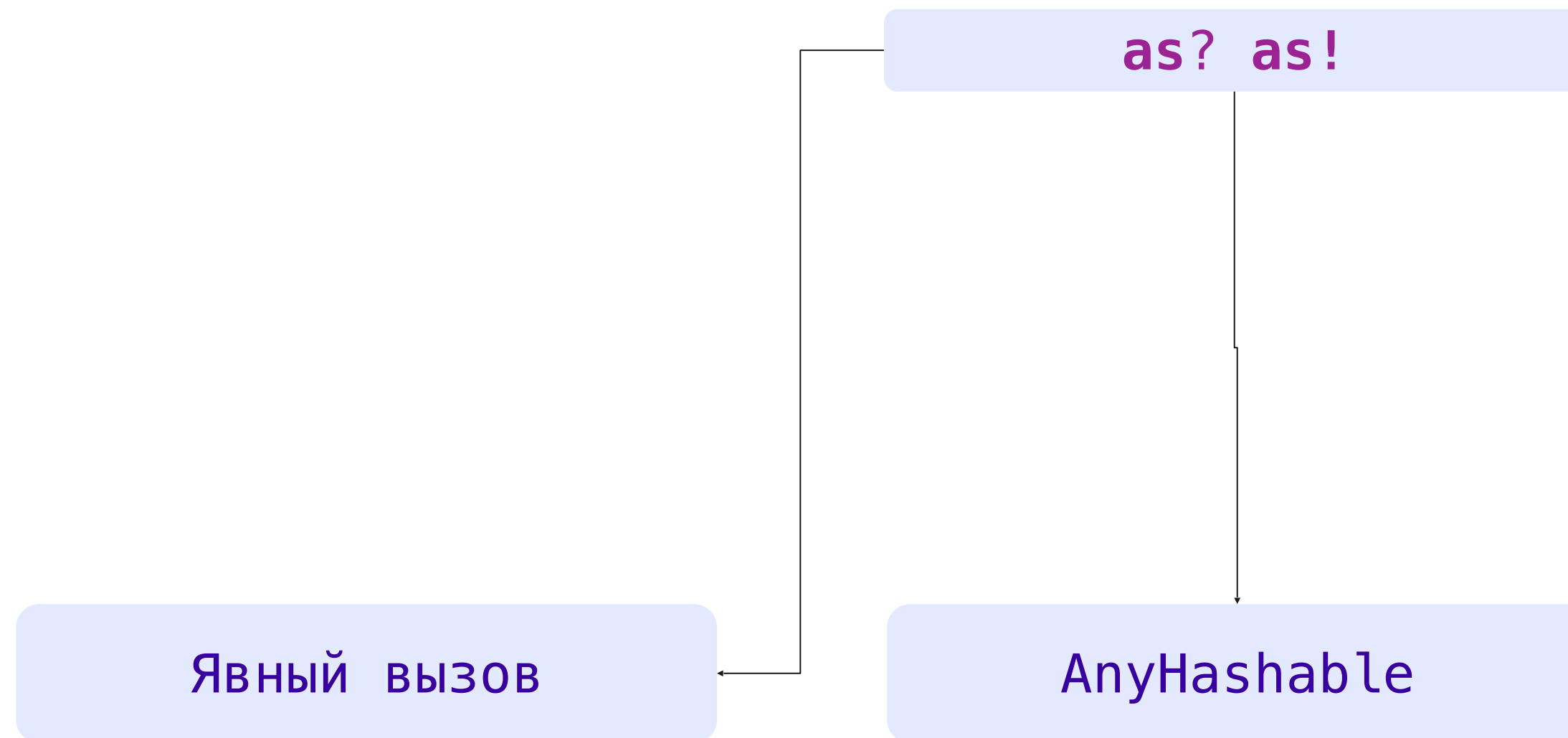
as? as!

as? as!

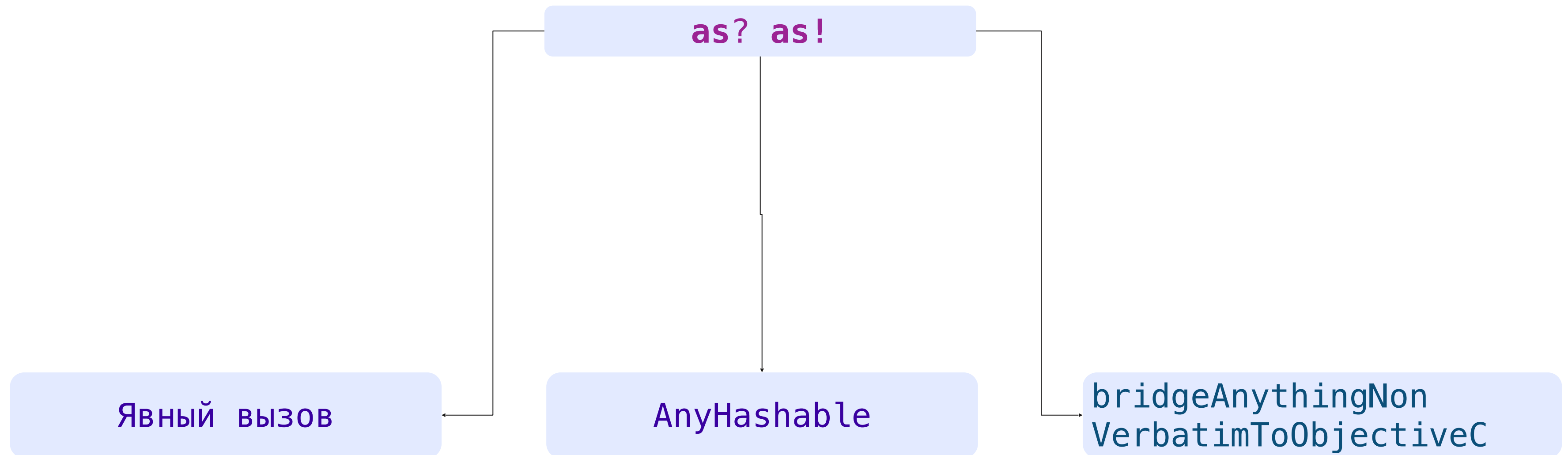
as? as!



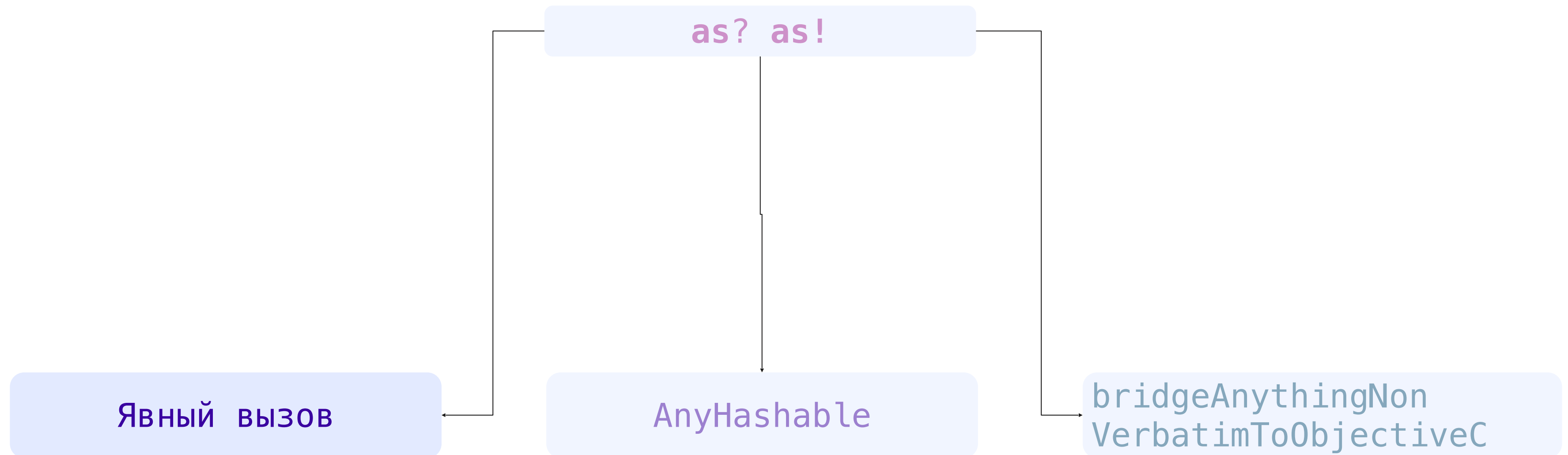
as? as!



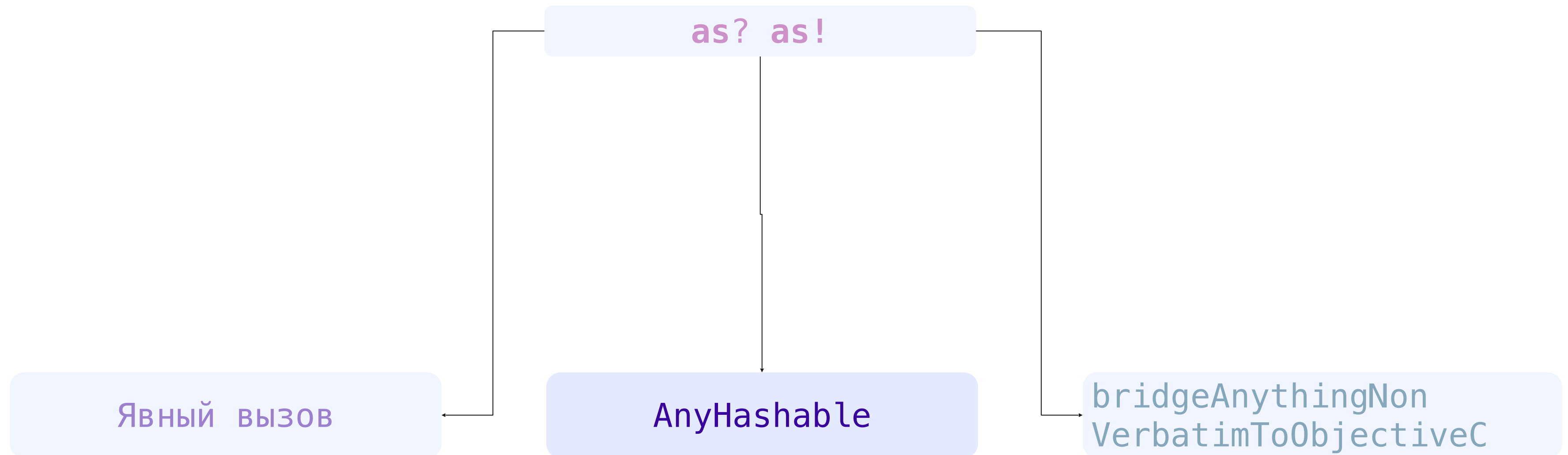
as? as!



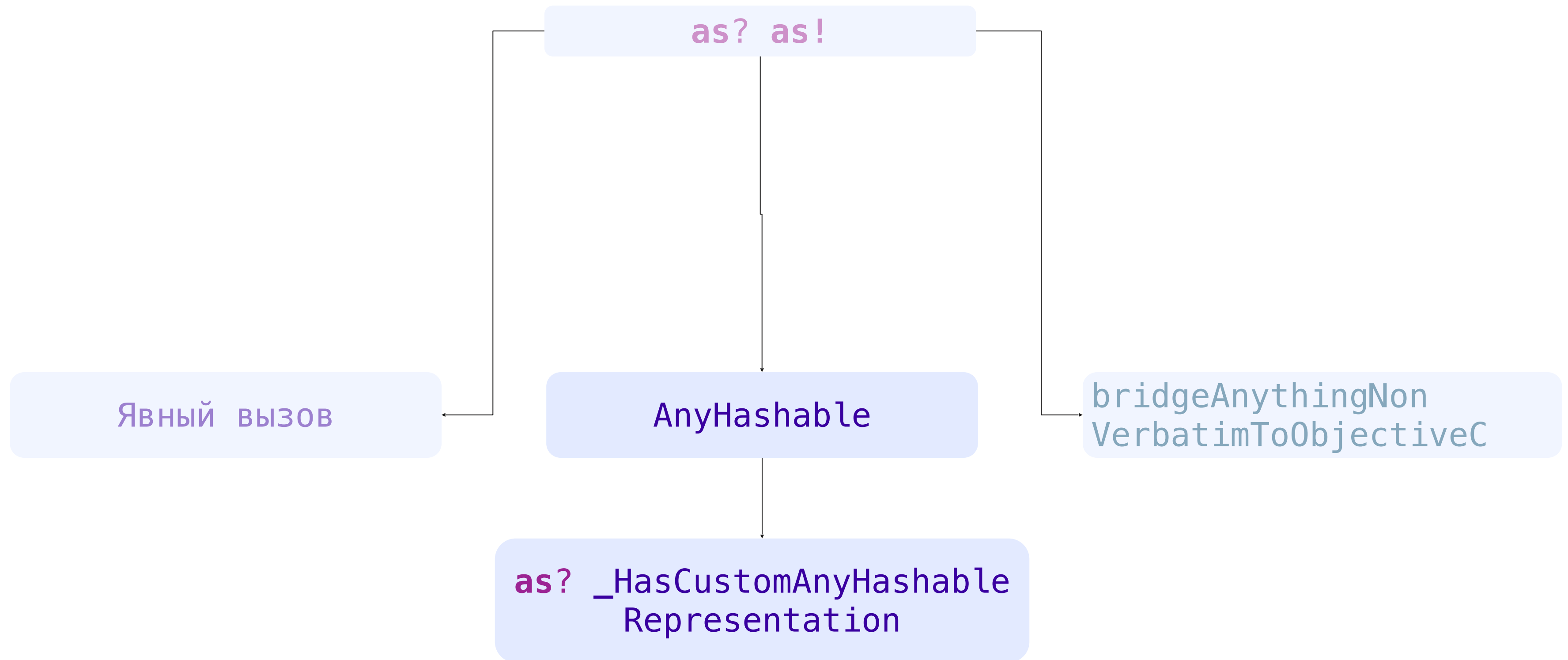
as? as!



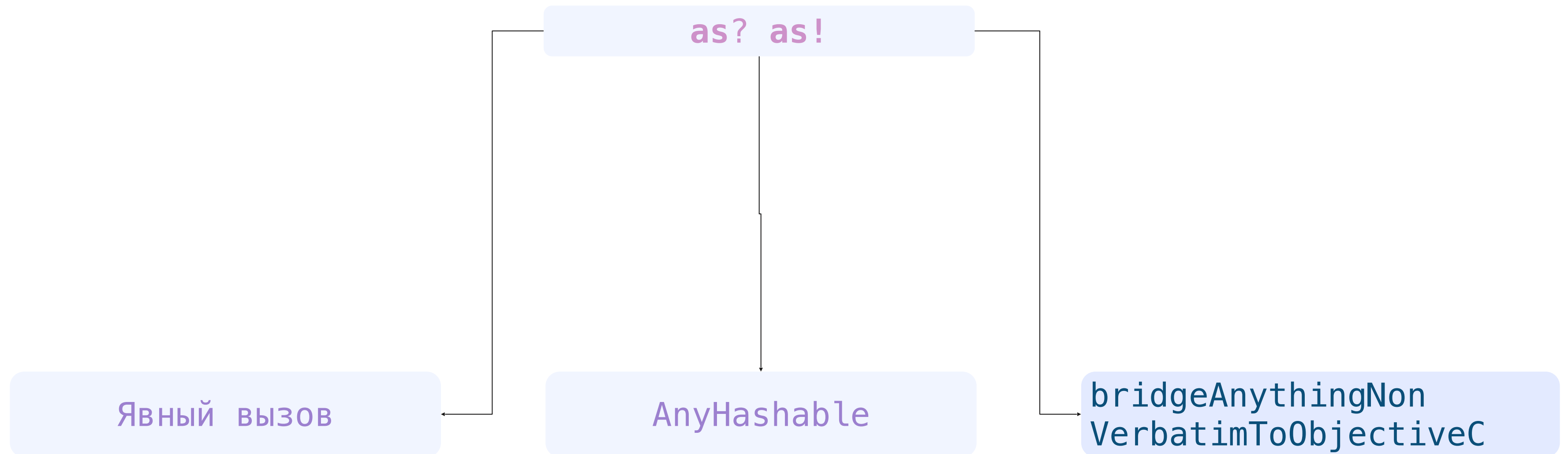
as? as!



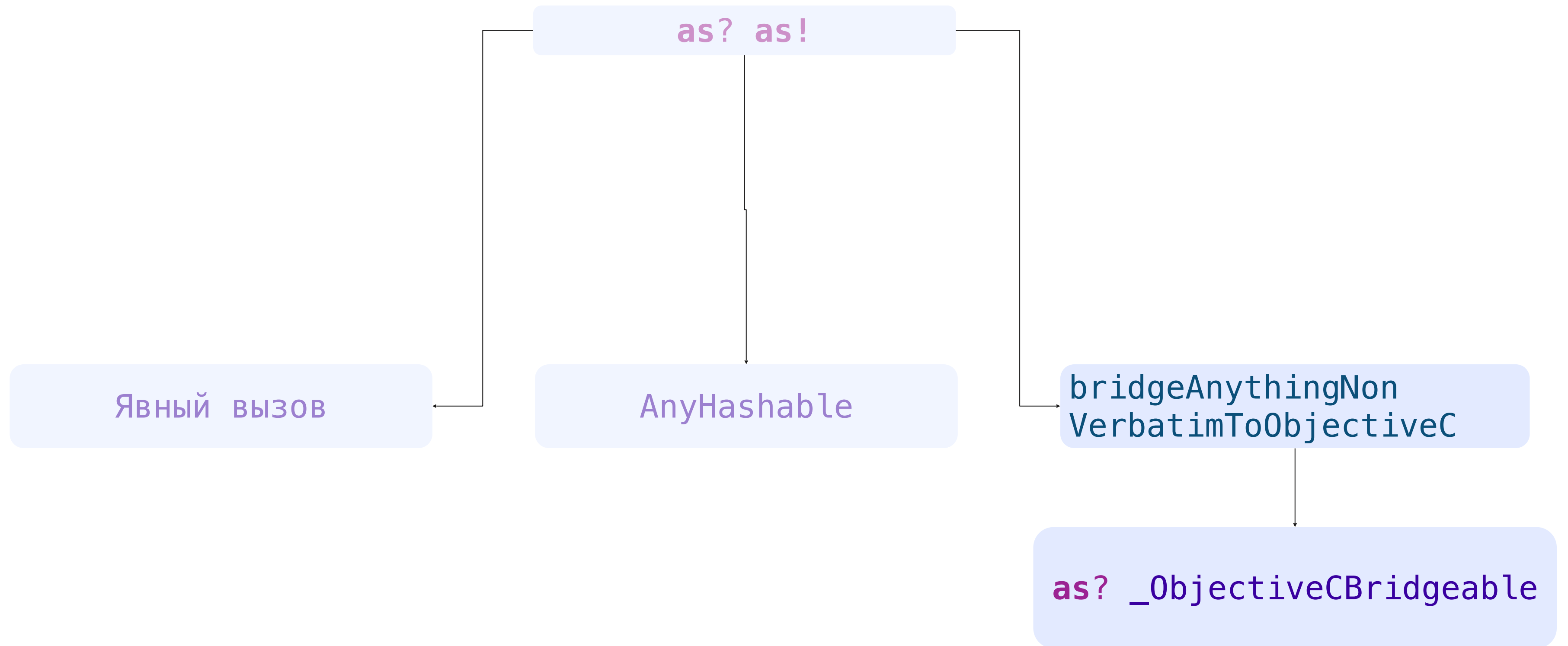
as? as!



as? as!



as? as!



as? as!

as? as!

as? as!

as? as!

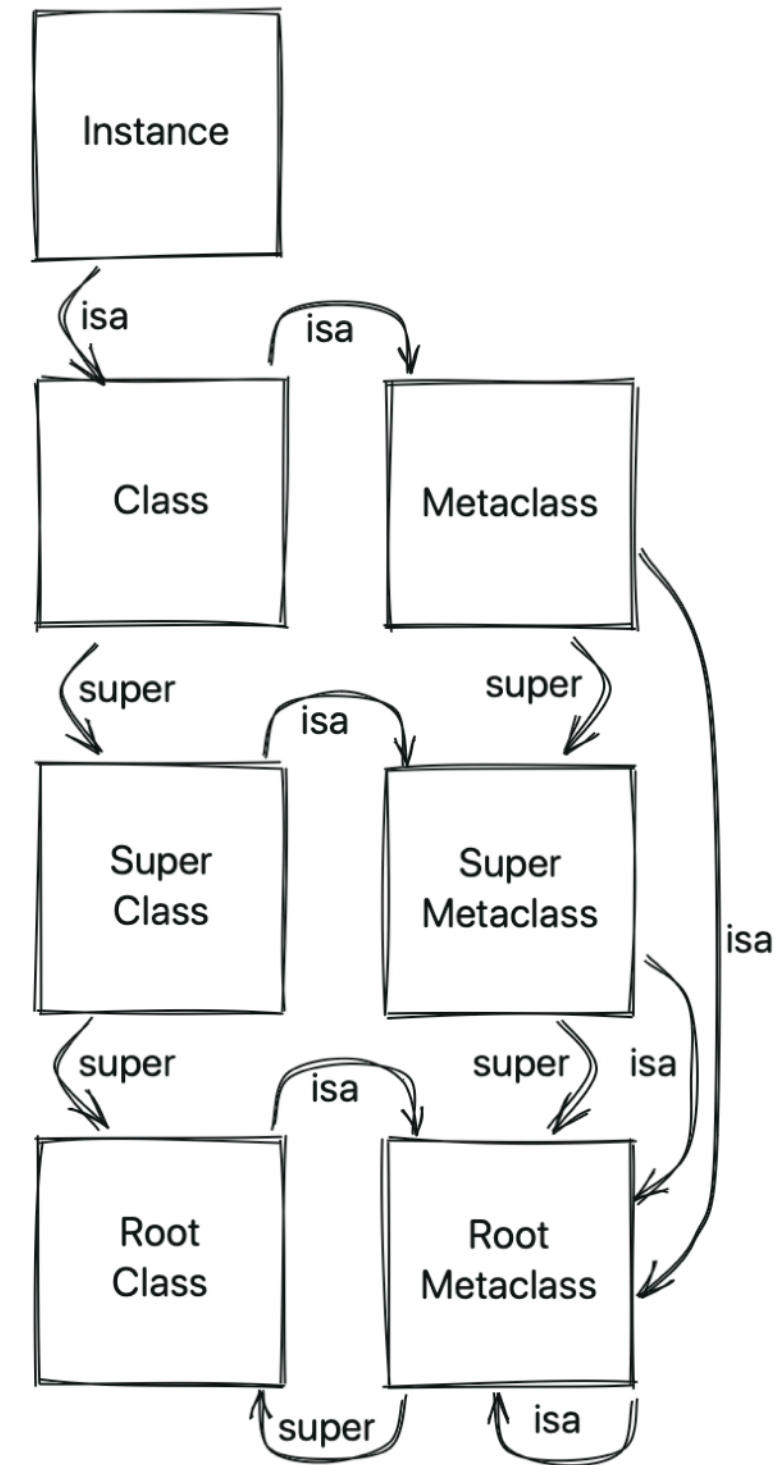
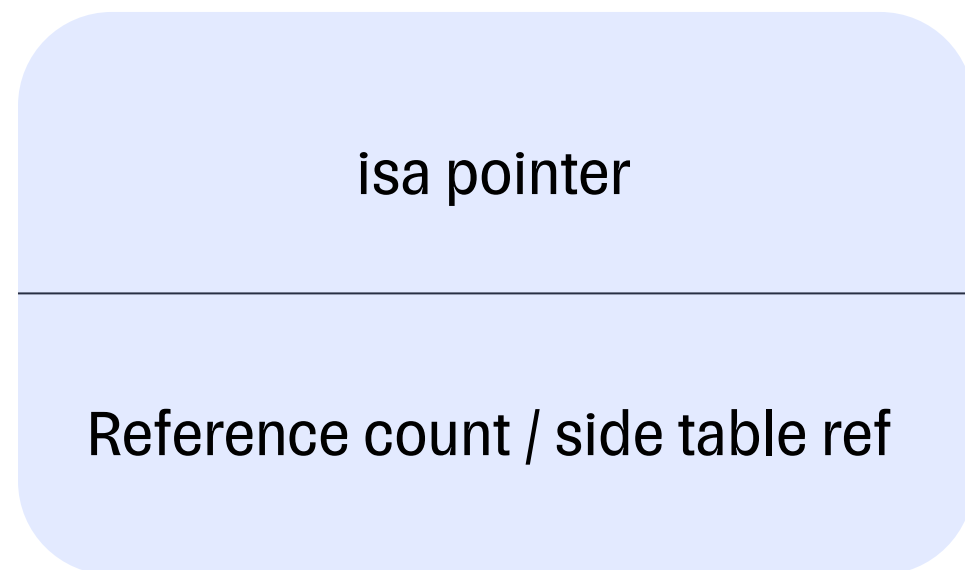
final class

HeapObject

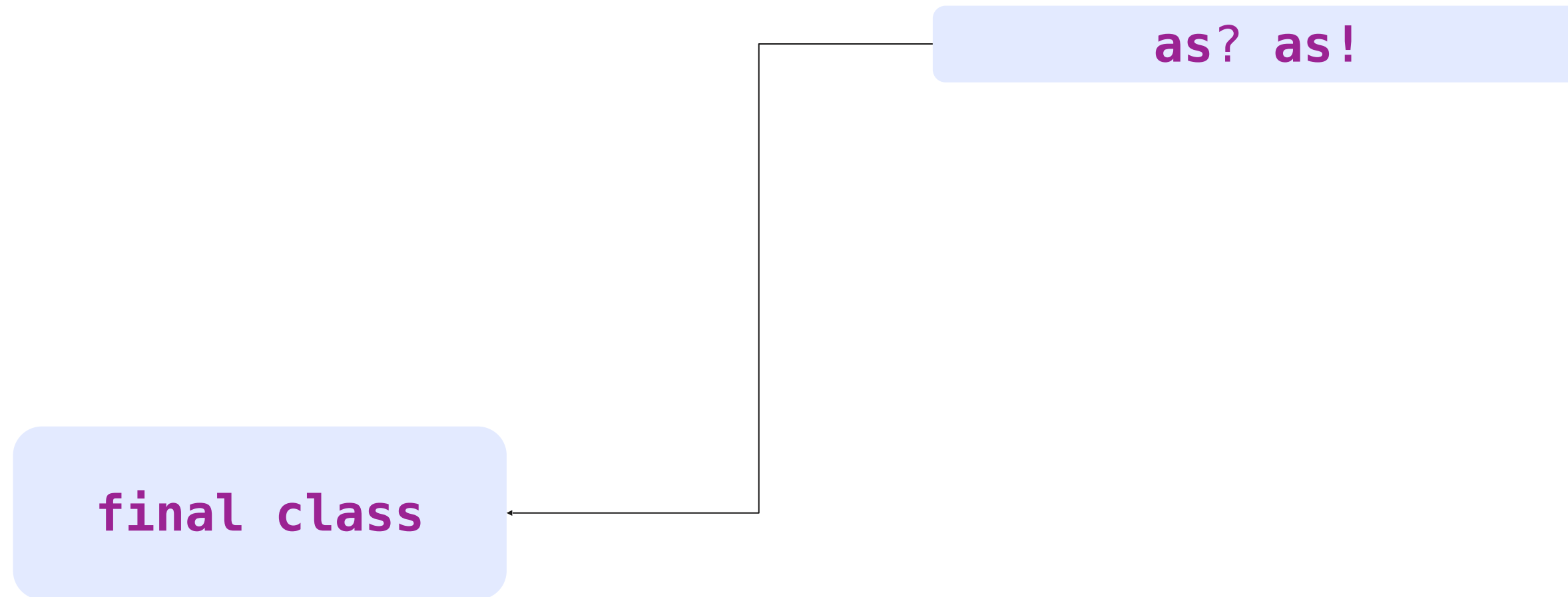
isa pointer

Reference count / side table ref

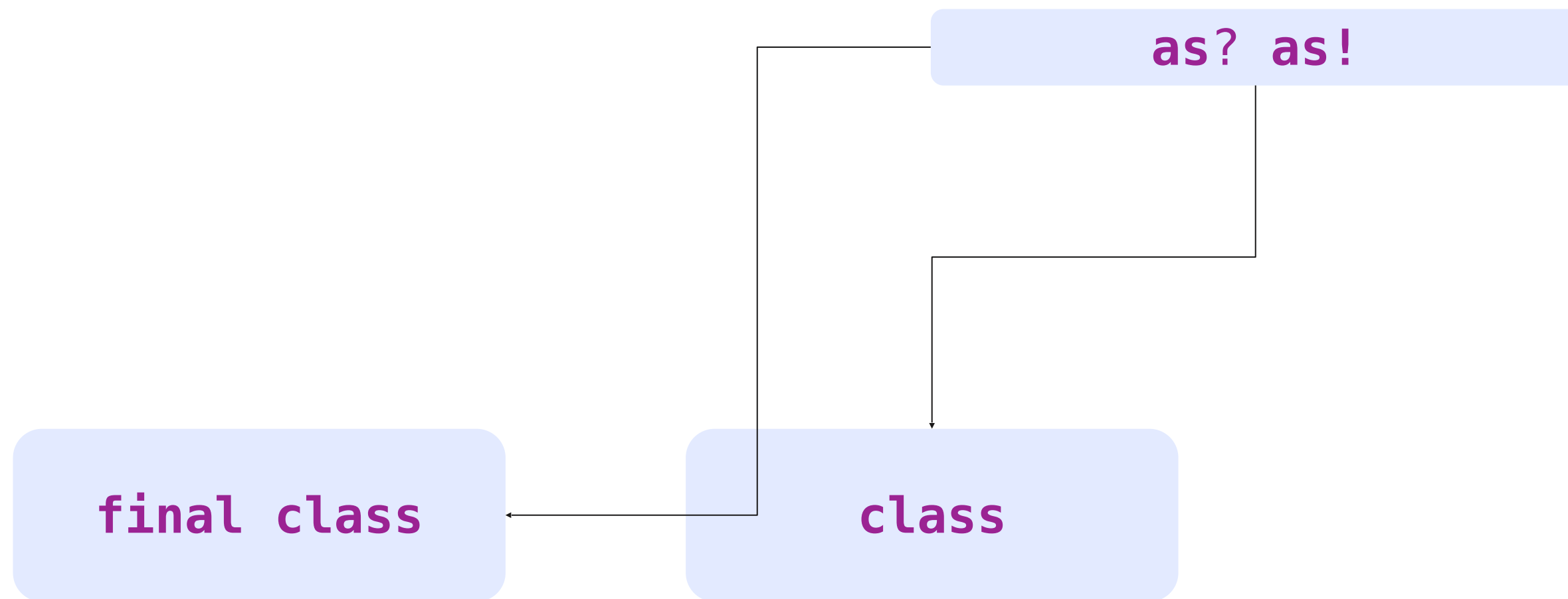
HeapObject



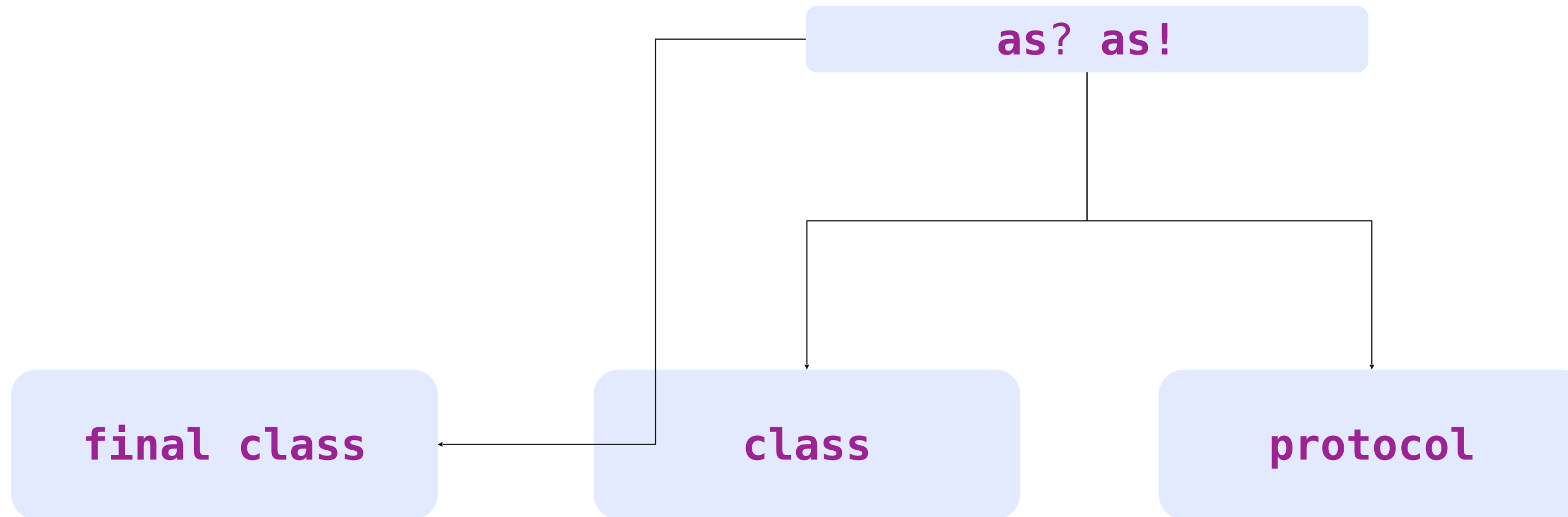
as? as!



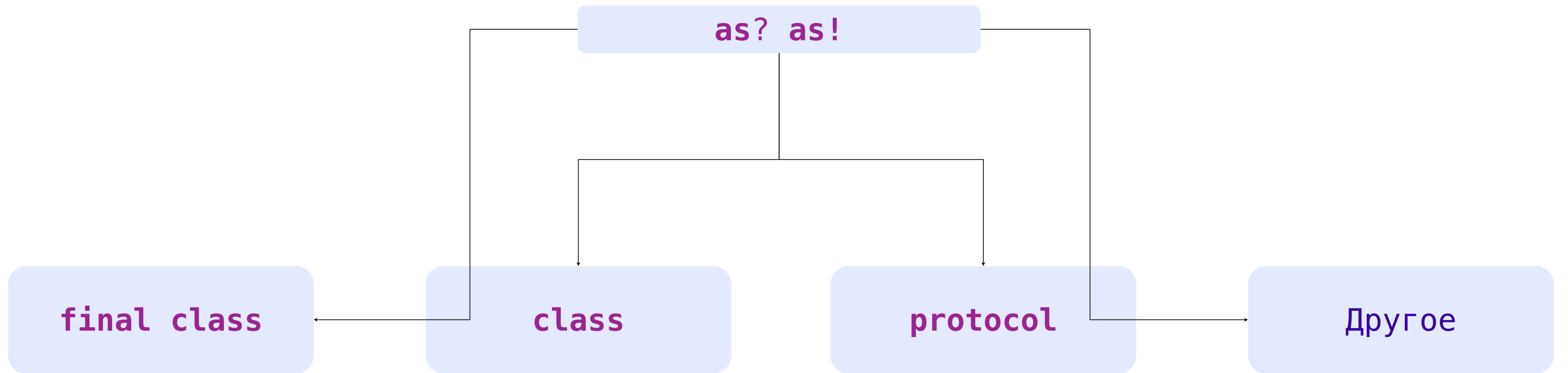
as? as!



as? as!



as? as!



T: SomeProtocol



T: SomeProtocol

| | | | |
|-----------|--------|-----|---|
| 653.00 ms | 100.0% | 0 s | ∨ MobileBank (4775) |
| 520.00 ms | 79.6% | 0 s | ∨ Main Thread 0x17d40b |
| 520.00 ms | 79.6% | 0 s | > swift::_checkGenericRequirements(__swift::__runtime::llvm:: |

T: SomeProtocol

| | | | |
|-----------|--------|-----|--|
| 653.00 ms | 100.0% | 0 s | ∨ MobileBank (4775) |
| 520.00 ms | 79.6% | 0 s | ∨ Main Thread 0x17d40b |
| 520.00 ms | 79.6% | 0 s | </> > swift::_checkGenericRequirements(__swift::__runtime::llvm::) |

~30% времени, уходящего на swift_conformsToProtocol

T: SomeProtocol

`swift_conformsToProtocolMaybeInstantiateSuperclasses`

`String(describing:)`

`as? as!`

`Base<T: SomeProtocol>`

T: SomeProtocol

```
Base<T: SomeProtocol>
```

T: SomeProtocol

```
Base<T: SomeProtocol>
```

T: SomeProtocol

```
Base<T: SomeProtocol>
```

T: SomeProtocol

Base<T: SomeProtocol>

↓

```
class Test<T: Decodable> {}  
let _ = Test<Int>.self
```

T: SomeProtocol

Base<T: SomeProtocol>



```
class Test<T: Decodable> {}  
let _ = Test<Int>.self
```

T: SomeProtocol

Base<T: SomeProtocol>

```
class Test<T: Decodable> {}  
let _ = Test<Int>.self
```

Test class type Metadata

T: SomeProtocol

Base<T: SomeProtocol>

```
class Test<T: Decodable> {}  
let _ = Test<Int>.self
```

Test class type Metadata

```
struct GenericParameterVector {  
    TypeMetadata *T;  
    DecodableWitnessTable *T_Decodable;  
};
```


T: SomeProtocol

Base<T: SomeProtocol>

```
class Test<T: Decodable> {}  
let _ = Test<Int>.self
```

Test class type Metadata

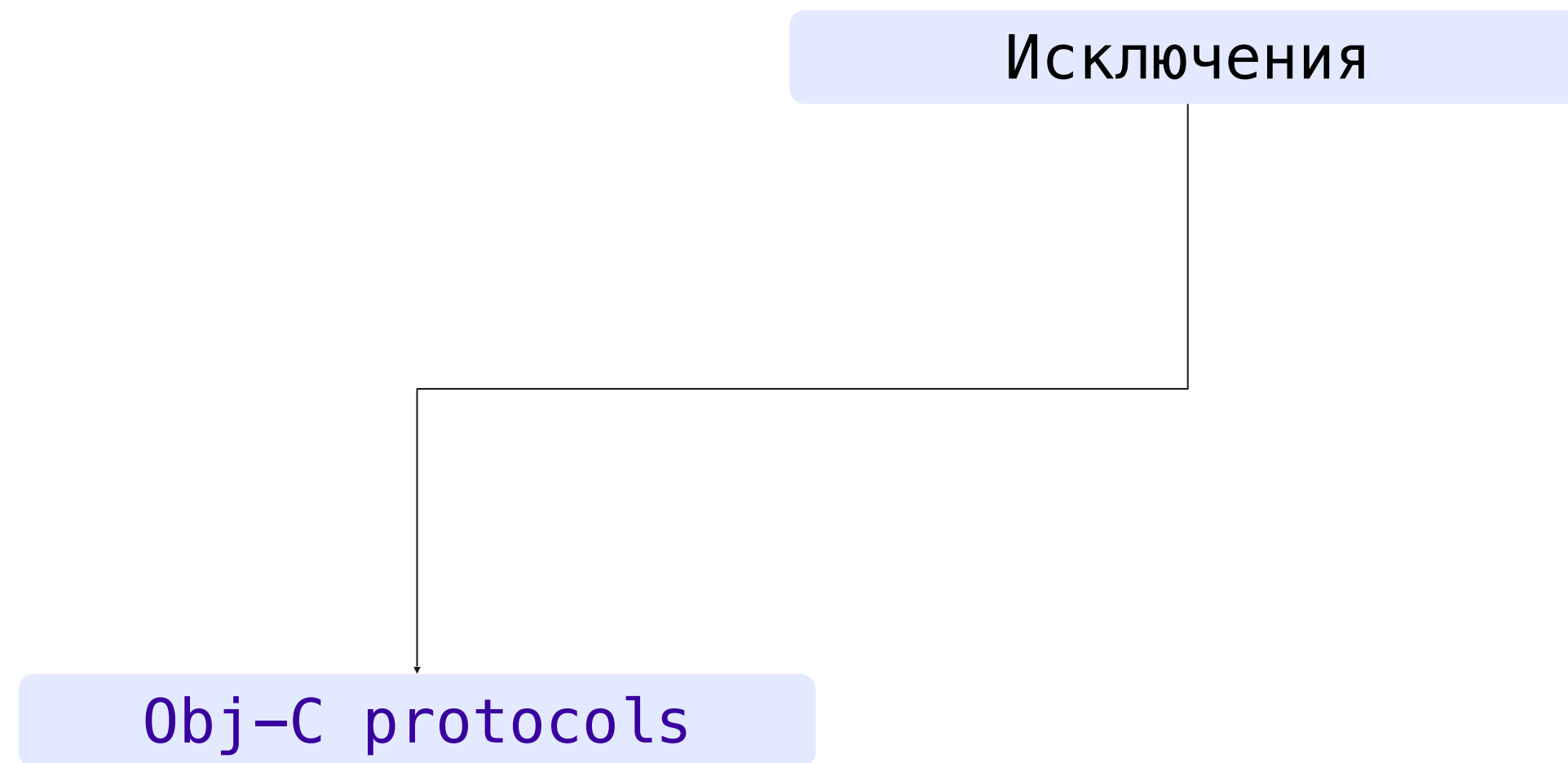
```
struct GenericParameterVector {  
    TypeMetadata *T;  
    DecodableWitnessTable *T_Decodable;  
};
```

swift_conformsToProtocol

T: SomeProtocol

Исключения

T: SomeProtocol



T: SomeProtocol



T: SomeProtocol. Исключения



T: SomeProtocol. Исключения



```
struct A<T: Decodable> {}  
  
let a = A<Int>()  
// нет вызова  
// swift_instantiateConcreteTypeFromMangledName
```

T: SomeProtocol. Исключения



```
struct A<T: Decodable> {}

let a = A<Int>()
// нет вызова
// swift_instantiateConcreteTypeFromMangledName
```



```
class A<T: Decodable> {}

func make<T: Decodable>() → A<T> {
    A<T>()
}

let a: A<Int> = make()
```

Все плохо...?



Отказ от `String(describing:)`



-5% старта приложения

Или более 100 мс в абсолютных

Отказ от

`String(describing:)`



-5% старта приложения

Или более 100 мс в абсолютных



Или более 130 мс в абсолютных

Отказ от

`String(describing:)`



-5% старта приложения

Или более 100 мс в абсолютах



Или более 130 мс в абсолютах



Около 100 мс в абсолютах

Отказ от

`String(describing:)`



-5% старта приложения

Или более 100 мс в абсолютных



-12% подготовки главной

Или более 130 мс в абсолютных



-7% загрузки главной

Около 100 мс в абсолютных



-30% инициализации экрана платежей

Около 35мс в абсолютных

Оптимизация DI

Заменяли касты к протоколам на
касты к классам;
Не храним структуры в DI



-20% старта приложения

Или более 400 мс в абсолютных

Оптимизация DI

Заменяли касты к протоколам на
касты к классам;
Не храним структуры в DI



-20% старта приложения

Или более 400 мс в абсолютных



Или более 420 мс в абсолютных

Оптимизация DI

Заменяли касты к протоколам на
касты к классам;
Не храним структуры в DI



-20% старта приложения

Или более 400 мс в абсолютных



-40% подготовки главной

Или более 420 мс в абсолютных



Ускорения открытия всех экранов

При открытии 15 экранов суммарно
экономим более 200 мс

Отказ
от as? as!



Отказ
от as? as!



Оптимизация DelegatesList

OptimizedDelegatesList

```
final public class DelegatesList<T>: Sequence {
    private let delegatesHashTable = NSHashTable<AnyObject>.weakObjects()
    ...
    public func makeIterator() → Array<T>.Iterator {
        let delegates = delegatesHashTable.allObjects.compactMap { $0 as? T }
        return delegates.makeIterator()
    }
}
```

OptimizedDelegatesList

```
final public class DelegatesList<T>: Sequence {  
    private let delegatesHashTable = NSHashTable<AnyObject>.weakObjects()  
    ...  
    public func makeIterator() → Array<T>.Iterator {  
        let delegates = delegatesHashTable.allObjects.compactMap { $0 as? T }  
        return delegates.makeIterator()  
    }  
}
```

OptimizedDelegatesList

```
final public class DelegatesList<T>: Sequence {  
    private let delegatesHashTable = NSHashTable<AnyObject>.weakObjects()  
    ...  
    public func makeIterator() → Array<T>.Iterator {  
        let delegates = delegatesHashTable.allObjects.compactMap { $0 as? T }  
        return delegates.makeIterator()  
    }  
}
```

```
final public class OptimizedDelegatesList<T>: Sequence {  
    private var delegates = Atomic<[() → T?]>([])  
  
    public func makeIterator() → Array<T>.Iterator {  
        delegates.value.compactMap { $0() }.makeIterator()  
    }  
}
```

Отказ от as? as!



Оптимизация DelegatesList

Отказ от as? as!



Оптимизация DelegatesList

10% подготовки главного экрана и около

10% загрузки главного экрана

Отказ от as? as!



Оптимизация DelegatesList

10% подготовки главного экрана и около
10% загрузки главного экрана



Оптимизация SnapKit

Оптимизации в SnapKit



Оптимизации в SnapKit

```
public final class ConstraintItem {  
    ...  
    internal var layoutConstraintItem: LayoutConstraintItem? {  
        return self.target as? LayoutConstraintItem  
    }  
}
```

Оптимизации в SnapKit

```
public final class ConstraintItem {  
    ...  
    internal var layoutConstraintItem: LayoutConstraintItem? {  
        return self.target as? LayoutConstraintItem  
    }  
}
```

```
extension LayoutConstraintItem {  
    internal func prepare() {  
        if let view = self as? ConstraintView {  
            view.translatesAutoresizingMaskIntoConstraints = false  
        }  
    }  
    internal var superview: ConstraintView? {  
        if let view = self as? ConstraintView {  
            return view.superview  
        }  
    }  
}
```

Оптимизации в SnapKit

```
public final class ConstraintItem {  
    ...  
    internal var layoutConstraintItem: LayoutConstraintItem? {  
        return self.target as? LayoutConstraintItem  
    }  
}
```

```
extension LayoutConstraintItem {  
    internal func prepare() {  
        if let view = self as? ConstraintView {  
            view.translatesAutoresizingMaskIntoConstraints = false  
        }  
    }  
    internal var superview: ConstraintView? {  
        if let view = self as? ConstraintView {  
            return view.superview  
        }  
    }  
}
```

```
public class ConstraintMakerRelatable {  
    internal func relatedTo(...) {  
        else if let other = other as? ConstraintView {  
            ...  
        } else if let other = other as? ConstraintConstantTarget {  
            ...  
        }  
    }  
}
```

Отказ от as? as!



Оптимизация DelegatesList

10% подготовки главного экрана
и около 10% загрузки главного экрана



Оптимизация SnapKit

Отказ от as? as!



Оптимизация DelegatesList

10% подготовки главного экрана
и около 10% загрузки главного экрана



Оптимизация SnapKit

Сэкономили более 100 мс при отрисовке
и скролле главной и не сломали ABI!

Отказ от as? as!



Оптимизация DelegatesList

10% подготовки главного экрана
и около 10% загрузки главного экрана



Оптимизация SnapKit

Сэкономили более 100 мс при отрисовке
и скролле главной и не сломали ABI!



Неявная PWT

Замена `as?` на неявную PWT

```
private fun collectParameters<WidgetInput: IHomeWidgetInput>(
    from widgets: [WidgetInput]
) → [String] {
    widgets.compactMap { widget → String? in
        if let provider = widget as? IHomeAnalyticsProvidableWidgetInput,
            !provider.analyticsParameters.isEmpty {...}
        ...
    }
}
```

Замена **as?** на неявную PWT

```
private fun collectParameters<WidgetInput: IHomeWidgetInput>(
    from widgets: [WidgetInput]
) → [String] {
    widgets.compactMap { widget → String? in
        if let provider = widget as? IHomeAnalyticsProvidableWidgetInput,
            !provider.analyticsParameters.isEmpty {...}
        ...
    }
}
```


Замена **as?** на неявную PWT

```
private fun collectParameters<WidgetInput: IHomeWidgetInput>(
    from widgets: [WidgetInput]
) → [String] {
    widgets.compactMap { widget → String? in
        if let provider = widget as? IHomeAnalyticsProvidableWidgetInput,
            !provider.analyticsParameters.isEmpty {...}
        ...
    }
}
```

```
public struct HomeAnalyticsProviding {
    public var analyticsParameters: () → [String]
}

public protocol IHomeWidgetInput: AnyObject {
    ...
    var analyticsProviding: HomeAnalyticsProviding? { get }
    ...
}
```

Замена `as?` на неявную PWT

```
private fun collectParameters<WidgetInput: IHomeWidgetInput>(
    from widgets: [WidgetInput]
) → [String] {
    widgets.compactMap { widget → String? in
        if let provider = widget as? IHomeAnalyticsProvidableWidgetInput,
            !provider.analyticsParameters.isEmpty { ... }
        ...
    }
}
```

```
public struct HomeAnalyticsProviding {
    public var analyticsParameters: () → [String]
}

public protocol IHomeWidgetInput: AnyObject {
    ...
    var analyticsProviding: HomeAnalyticsProviding? { get }
    ...
}
```

```
public extension IHomeWidgetInput {
    var analyticsProviding: HomeAnalyticsProviding? { nil }
}
```

Замена `as?` на неявную PWT

```
private fun collectParameters<WidgetInput: IHomeWidgetInput>(
    from widgets: [WidgetInput]
) → [String] {
    widgets.compactMap { widget → String? in
        if let provider = widget as? IHomeAnalyticsProvidableWidgetInput,
            !provider.analyticsParameters.isEmpty { ... }
        ...
    }
}
```

```
public struct HomeAnalyticsProviding {
    public var analyticsParameters: () → [String]
}

public protocol IHomeWidgetInput: AnyObject {
    ...
    var analyticsProviding: HomeAnalyticsProviding? { get }
    ...
}
```

```
public extension IHomeWidgetInput {
    var analyticsProviding: HomeAnalyticsProviding? { nil }
}
```

```
public extension IHomeWidgetInput
    where Self: IHomeAnalyticsProvidableWidgetInput {
    var analyticsProviding: HomeAnalyticsProviding? {
        HomeAnalyticsProviding(
            analyticsParameters: { self.analyticsParameters }
        )
    }
}
```

Замена `as?` на неявную PWT

```
private fun collectParameters<WidgetInput: IHomeWidgetInput>(
    from widgets: [WidgetInput]
) → [String] {
    widgets.compactMap { widget → String? in
        if let provider = widget as? IHomeAnalyticsProvidableWidgetInput,
            !provider.analyticsParameters.isEmpty { ... }
        ...
    }
}
```

```
public struct HomeAnalyticsProviding {
    public var analyticsParameters: () → [String]
}

public protocol IHomeWidgetInput: AnyObject {
    ...
    var analyticsProviding: HomeAnalyticsProviding? { get }
    ...
}
```

```
public extension IHomeWidgetInput {
    var analyticsProviding: HomeAnalyticsProviding? { nil }
}
```

```
public extension IHomeWidgetInput
    where Self: IHomeAnalyticsProvidableWidgetInput {
    var analyticsProviding: HomeAnalyticsProviding? {
        HomeAnalyticsProviding(
            analyticsParameters: { self.analyticsParameters })
    }
}
```

Замена `as?` на неявную PWT

```
private fun collectParameters<WidgetInput: IHomeWidgetInput>(
    from widgets: [WidgetInput]
) → [String] {
    widgets.compactMap { widget → String? in
        if let provider = widget as? IHomeAnalyticsProvidableWidgetInput,
            !provider.analyticsParameters.isEmpty { ... }
        ...
    }
}
```

```
private fun collectParameters<WidgetInput: IHomeWidgetInput>(
    from widgets: [WidgetInput]
) → [String] {
    widgets.compactMap { widget → String? in
        if let parameters = widget.analyticsProviding?.analyticsParameters(),
            !parameters.isEmpty { ... }
        ...
    }
}
```

```
public struct HomeAnalyticsProviding {
    public var analyticsParameters: () → [String]
}

public protocol IHomeWidgetInput: AnyObject {
    ...
    var analyticsProviding: HomeAnalyticsProviding? { get }
    ...
}
```

```
public extension IHomeWidgetInput {
    var analyticsProviding: HomeAnalyticsProviding? { nil }
}
```

```
public extension IHomeWidgetInput
    where Self: IHomeAnalyticsProvidableWidgetInput {
    var analyticsProviding: HomeAnalyticsProviding? {
        HomeAnalyticsProviding(
            analyticsParameters: { self.analyticsParameters }
        )
    }
}
```

Отказ от as? as!



Оптимизация DelegatesList

10% подготовки главного экрана
и около 10% загрузки главного экрана



Оптимизация SnapKit

Сэкономили более 100 мс при отрисовке
и скролле главной и не сломали ABI!



Неявная PWT

Отказ от as? as!



Оптимизация DelegatesList

10% подготовки главного экрана
и около 10% загрузки главного экрана



Оптимизация SnapKit

Сэкономили более 100 мс при отрисовке
и скролле главной и не сломали ABI!



Неявная PWT

Сэкономили более 30 мс в момент загрузки
счетов и 2% подготовки главного экрана

Отказ type- generic- constraints



-11% старта

Или около 220 мс в абсолютах

Отказ type- generic- constraints



-11% старта

Или около 220 мс в абсолютных



-10% подготовки главной

Или около 95 мс в абсолютных

Type-generic-constraints

```
final class DefaultsStorage<Item: Codable>: Storage {
    init(key: StorageKey) {...}

    func get() → Item? {
        ...
        if let object = storage.object(forKey: key) as? Data {
            let decoder = JSONDecoder()
            if let decoded = try? decoder.decode(Item.self, from: object) {
                return decoded
            }
        }
    }

    func set(_ item: Item?) {
        ...
        let encoder = JSONEncoder()
        if let encoded = try? encoder.encode(item) {
            storage.set(encoded, forKey: key)
        }
    }
}
```

Type-generic-constraints



```
final class DefaultsStorage<Item: Codable>: Storage {
    init(key: StorageKey) {...}

    func get() → Item? {
        ...
        if let object = storage.object(forKey: key) as? Data {
            let decoder = JSONDecoder()
            if let decoded = try? decoder.decode(Item.self, from: object) {
                return decoded
            }
        }
    }

    func set(_ item: Item?) {
        ...
        let encoder = JSONEncoder()
        if let encoded = try? encoder.encode(item) {
            storage.set(encoded, forKey: key)
        }
    }
}
```

Type-generic-constraints



```
final class DefaultsStorage<Item>: Storage {
    init(key: StorageKey,
        decode: @escaping (JSONDecoder, Data) throws → Item,
        encode: @escaping (JSONEncoder, Item?) throws → Data) {...}

    func get() → Item? {
        ...
        if let object = storage.object(forKey: key) as? Data {
            let decoder = JSONDecoder()
            if let decoded = try? decode(decoder, object) {
                return decoded
            }
        }
    }

    func set(_ item: Item?) {
        ...
        let encoder = JSONEncoder()
        if let encoded = try? encode(encoder, item) {
            storage.set(encoded, forKey: key)
        }
    }
}
```

Type-generic-constraints

```
extension DefaultsStorage where Item: Codable {
  convenience init(
    key: StorageKey,
  ) {
    self.init(
      key: key,
      decode: { decoder, data in
        try decoder.decode(Item.self, from: data)
      },
      encode: { encoder, item in try encoder.encode(item) }
    )
  }
}
```

Type-generic-constraints



```
public typealias TCSCollectionCellContainedView = TCSConfigurableView &  
    TCSDisableable &  
    TCSHighlightable &  
    TCSReusableView &  
    TCSSelectable  
  
public final class TCSCollectionViewContainerCellBase<  
    T: UIView & TCSCollectionCellContainedView  
>
```

Type-generic-constraints

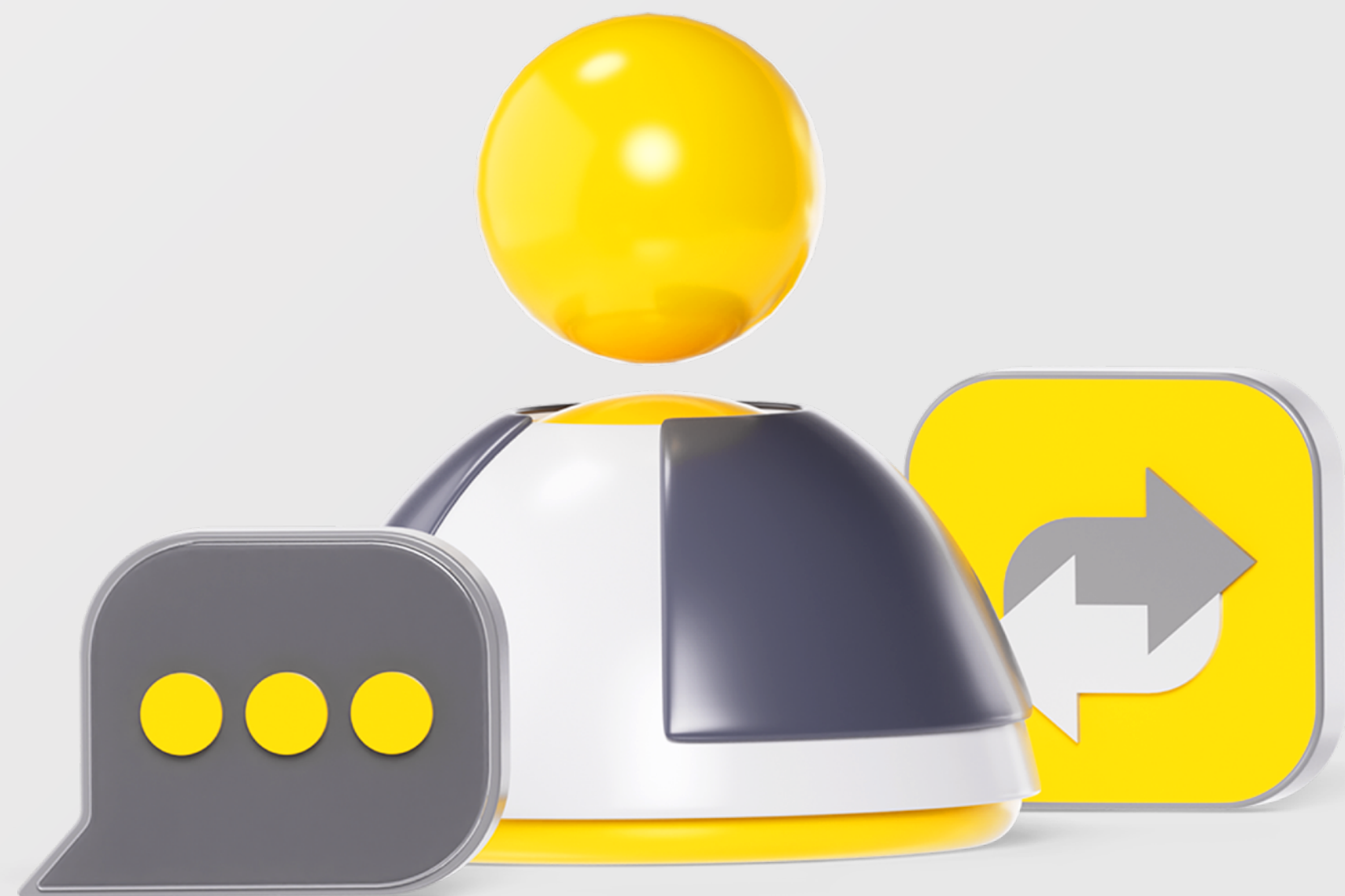
Type-generic-constraints



```
public protocol TCSCollectionCellContainedView: TCSConfigurableView,  
    TCSDisableable,  
    TCSHighlightable,  
    TCSReusableView,  
    TCSSelectable {}
```

```
public final class TCSCollectionViewContainerCellBase<  
    T: UIView & TCSCollectionCellContainedView  
>
```


Рекомендации



Откажитесь от `String` (describing:)

Откажитесь от `String(describing:)`

Строковое название типа

```
_typeName(T.self, qualified: false)  
// или  
"\(T.self)"
```

Откажитесь от `String(describing:)`

Строковое название типа

```
_typeName(T.self, qualified: false)  
// или  
"\(T.self)"
```

Идентификатор типа

```
ObjectIdentifier(T.self)
```

Откажитесь от `String(describing:)`

Строковое название типа

```
_typeName(T.self, qualified: false)  
// или  
"\(T.self)"
```

Идентификатор типа

```
ObjectIdentifier(T.self)
```

Используйте
`CustomStringConvertible`

```
"\(value)"
```

Проблема не только в производительности

```
enum Code: Int {  
    case unknown = 0  
    case prime = 1  
}  
struct X {  
    var a: Int  
}  
print(String(describing: BaseErrorCode.prime))  
print(String(describing: X(a: 5)))
```

Проблема не только в производительности



```
enum Code: Int {  
    case unknown = 0  
    case prime = 1  
}  
struct X {  
    var a: Int  
}  
print(String(describing: BaseErrorCode.prime))  
print(String(describing: X(a: 5)))
```

Проблема не только в производительности

```
enum Code: Int {
  case unknown = 0
  case prime = 1
}
struct X {
  var a: Int
}
print(String(describing: BaseErrorCode.prime))
print(String(describing: X(a: 5)))
```

SWIFT_REFLECTION_METADATA_LEVEL

All

> prime
> X(a: 5)

None

> Code(rawValue: 1)
> X()

Сведите к минимуму as? as!

Старайтесь проектировать Ваш код без dynamicCast-ов

- Если же без них никак, в случае с generic-методами Вы можете воспользоваться трюком с неявной PWT в самом протоколе.
- Помните про накладные расходы на конвертацию value-типов в AnyObject.
- При создании AnyHashable тоже есть проверка на конформанс протоколу
- Первый каст – самый тяжелый.

DI

В большинстве приложений большая часть dynamicCast-ов приходится на DI-фреймворк.

- Старайтесь спроектировать Ваш код так, чтобы в DI-фреймворке, например, Swinject-е, регистрировались зависимости с ключом мета-типом класса, а не протокола.
- Не регистрируйте структуры и не class-constrained протоколы в DI-контейнере.

Type- generic- constrains

Избегайте type-generic-constraint-ов
в переиспользуемых компонентах

Используйте явный инжект PWT в конструктор в связке
с convenience init вместо type-generic-constraint-ов.

Если отказаться от type-generic-constraint-ов не получается,
сведите к минимуму количество протоколов, которым
должен конформить generic-тип.

Список для чтения



Устройство бинаря
приложения



Блог
EmergeTools



Репозиторий
Swift

Спасибо!

