#WWDC18

Measuring Performance Using Logging

Signposts and Instruments

Session 405

Shane Owara, Darwin Runtime
Chad Woolf, Instruments

Your mission: Improve performance

Introducing Signposts

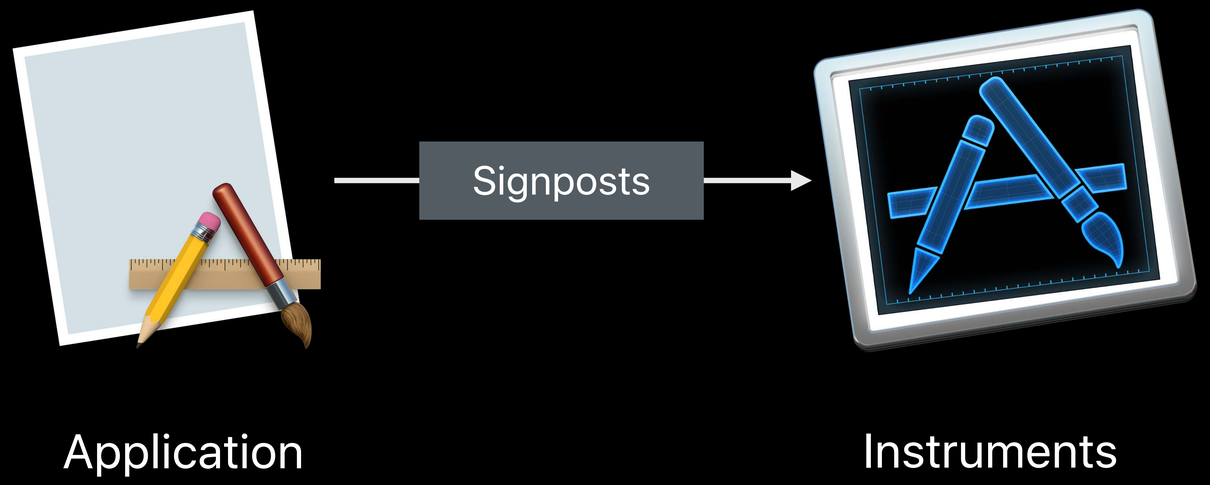
NEW

Signposts

- Part of the os_log family
- Performance-focused time markers

Instruments

- Aggregate and analyze signpost data
- Visualize activity over time



Logging

```
let logHandle = OSLog(subsystem: "com.example.widget", category: "Setup")

os_log(.info, log: logHandle, "Hello, %{public}s!", world)
```

Our new logging system was introduced at WWDC 2016

- Built for debugging with efficiency and privacy in mind

Unified Logging and Activity Tracing WWDC 2016

---

https://developer.apple.com/documentation/os/logging

Logging

```
let logHandle = OSLog(subsystem: "com.example.widget", category: "Setup")

os_log(.info, log: logHandle, "Hello, %{public}s!", world)
```

Our new logging system was introduced at WWDC 2016

- Built for debugging with efficiency and privacy in mind

Signposts created for investigating performance

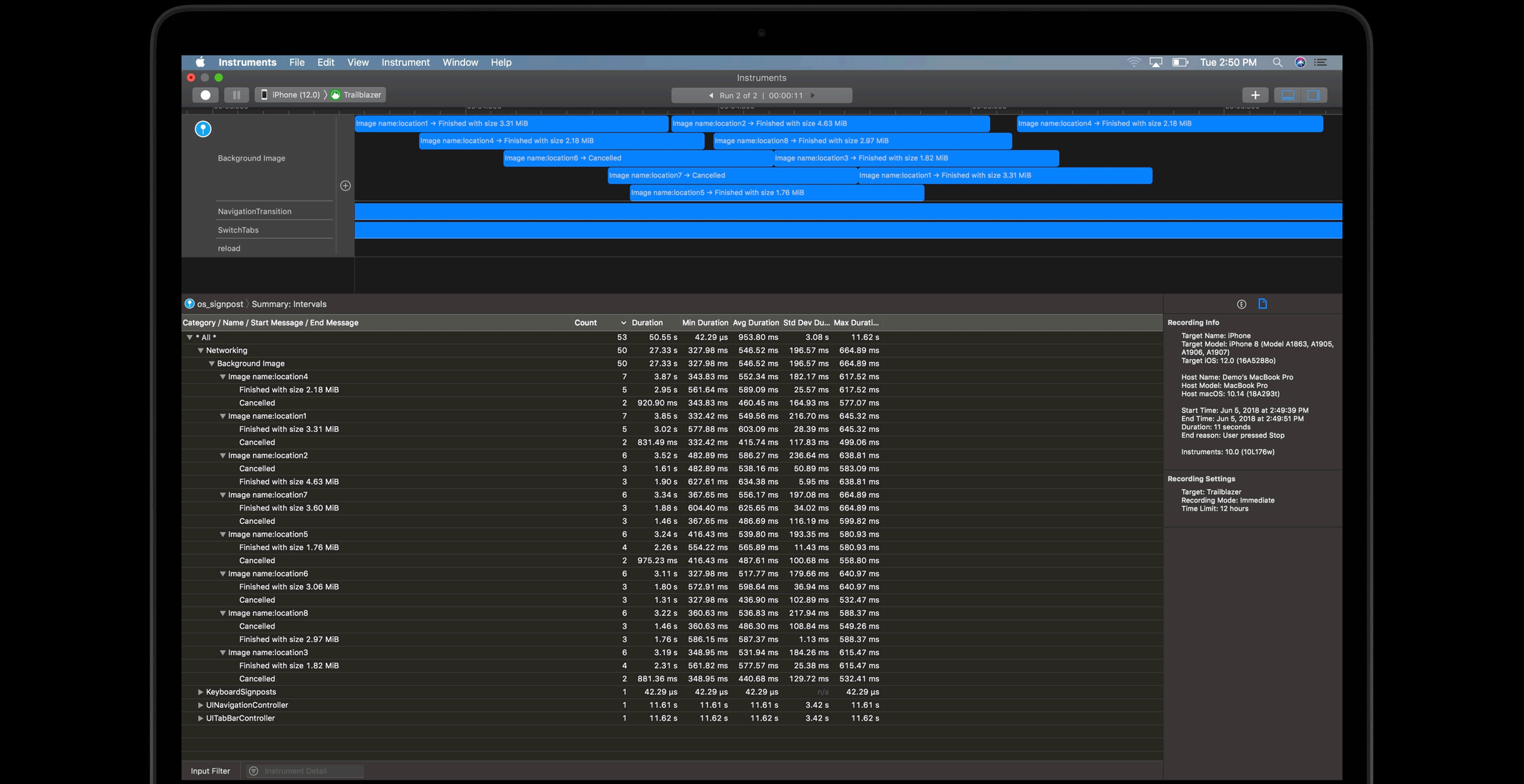
- Built for performance use case and integration with developer tools

---

Unified Logging and Activity Tracing WWDC 2016

---

https://developer.apple.com/documentation/os/logging


Instruments File Edit View Instrument Window Help
Tue 2:50 PM
Instruments
iPhone (12.0) Trailblazer
Run 2 of 2 | 00:00:11
Image name:location1 → Finished with size 3.31 MiB
Image name:location2 → Finished with size 4.63 MiB
Image name:location4 → Finished with size 2.18 MiB
Image name:location4 → Finished with size 2.18 MiB
Image name:location8 → Finished with size 2.97 MiB
Image name:location6 → Cancelled
Image name:location3 → Finished with size 1.82 MiB
Image name:location7 → Cancelled
Image name:location1 → Finished with size 3.31 MiB
Image name:location5 → Finished with size 1.76 MiB
Background Image
NavigationTransition
SwitchTabs
reload
os_signpost 〉 Summary: Intervals
Category / Name / Start Message / End Message
Count
Duration
Min Duration
Avg Duration
Std Dev Du...
Max Durati...
* All * 53 50.55 s 42.29 µs 953.80 ms 3.08 s 11.62 s
Networking 50 27.33 s 327.98 ms 546.52 ms 196.57 ms 664.89 ms
Background Image 50 27.33 s 327.98 ms 546.52 ms 196.57 ms 664.89 ms
Image name:location4 7 3.87 s 343.83 ms 552.34 ms 182.17 ms 617.52 ms
Finished with size 2.18 MiB 5 2.95 s 561.64 ms 589.09 ms 25.57 ms 617.52 ms
Cancelled 2 920.90 ms 343.83 ms 460.45 ms 164.93 ms 577.07 ms
Image name:location1 7 3.85 s 332.42 ms 549.56 ms 216.70 ms 645.32 ms
Finished with size 3.31 MiB 5 3.02 s 577.88 ms 603.09 ms 28.39 ms 645.32 ms
Cancelled 2 831.49 ms 332.42 ms 415.74 ms 117.83 ms 499.06 ms
Image name:location2 6 3.52 s 482.89 ms 586.27 ms 236.64 ms 638.81 ms
Cancelled 3 1.61 s 482.89 ms 538.16 ms 50.89 ms 583.09 ms
Finished with size 4.63 MiB 3 1.90 s 627.61 ms 634.38 ms 5.95 ms 638.81 ms
Image name:location7 6 3.34 s 367.65 ms 556.17 ms 197.08 ms 664.89 ms
Finished with size 3.60 MiB 3 1.88 s 604.40 ms 625.65 ms 34.02 ms 664.89 ms
Cancelled 3 1.46 s 367.65 ms 486.69 ms 116.19 ms 599.82 ms
Image name:location5 6 3.24 s 416.43 ms 539.80 ms 193.35 ms 580.93 ms
Finished with size 1.76 MiB 4 2.26 s 554.22 ms 565.89 ms 11.43 ms 580.93 ms
Cancelled 2 975.23 ms 416.43 ms 487.61 ms 100.68 ms 558.80 ms
Image name:location6 6 3.11 s 327.98 ms 517.77 ms 179.66 ms 640.97 ms
Finished with size 3.06 MiB 3 1.80 s 572.91 ms 598.64 ms 36.94 ms 640.97 ms
Cancelled 3 1.31 s 327.98 ms 436.90 ms 102.89 ms 532.47 ms
Image name:location8 6 3.22 s 360.63 ms 536.83 ms 217.94 ms 588.37 ms
Cancelled 3 1.46 s 360.63 ms 486.30 ms 108.84 ms 549.26 ms
Finished with size 2.97 MiB 3 1.76 s 586.15 ms 587.37 ms 1.13 ms 588.37 ms
Image name:location3 6 3.19 s 348.95 ms 531.94 ms 184.26 ms 615.47 ms
Finished with size 1.82 MiB 4 2.31 s 561.82 ms 577.57 ms 25.38 ms 615.47 ms
Cancelled 2 881.36 ms 348.95 ms 440.68 ms 129.72 ms 532.41 ms
KeyboardSignposts 1 42.29 µs 42.29 µs 42.29 µs n/a 42.29 µs
UINavigationController 1 11.61 s 11.61 s 11.61 s 3.42 s 11.61 s
UITabBarController 1 11.62 s 11.62 s 11.62 s 3.42 s 11.62 s
Recording Info
Target Name: iPhone
Target Model: iPhone 8 (Model A1863, A1905, A1906, A1907)
Target iOS: 12.0 (16A5288o)
Host Name: Demo's MacBook Pro
Host Model: MacBook Pro
Host macOS: 10.14 (18A293t)
Start Time: Jun 5, 2018 at 2:49:39 PM
End Time: Jun 5, 2018 at 2:49:51 PM
Duration: 11 seconds
End reason: User pressed Stop
Instruments: 10.0 (10L176w)
Recording Settings
Target: Trailblazer
Recording Mode: Immediate
Time Limit: 12 hours
Input Filter
Instrument Detail
MacBook Pro

Adopting signposts

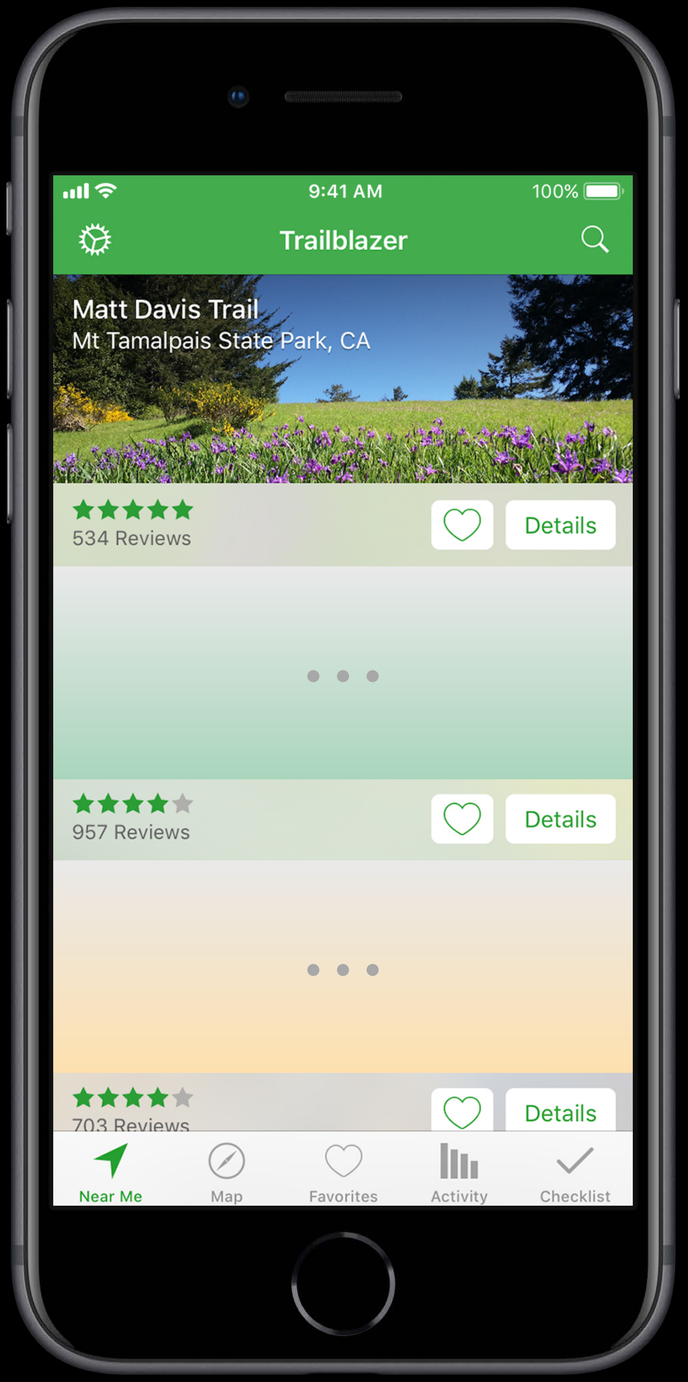
Overlapping operations

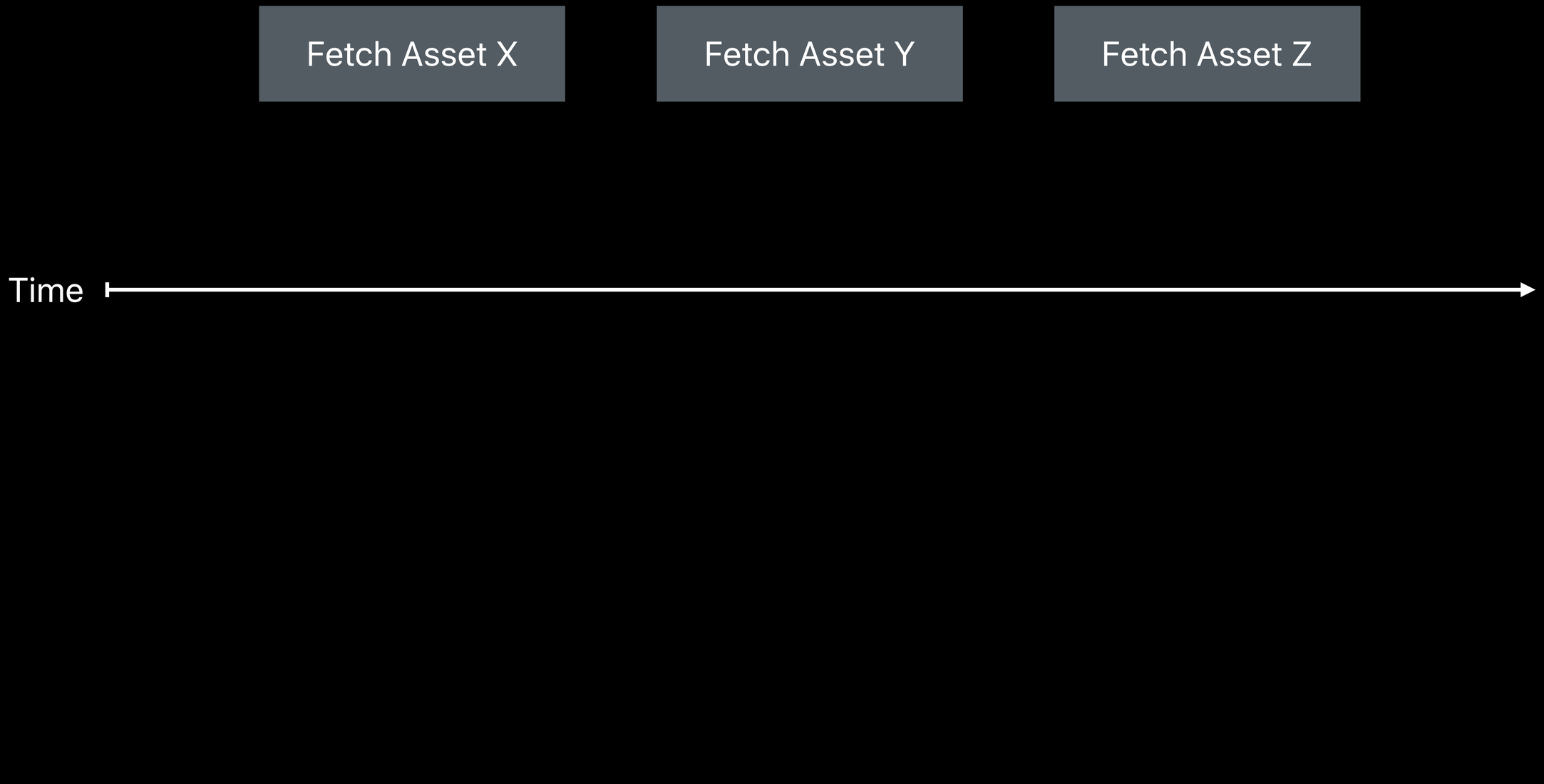
Adding metadata

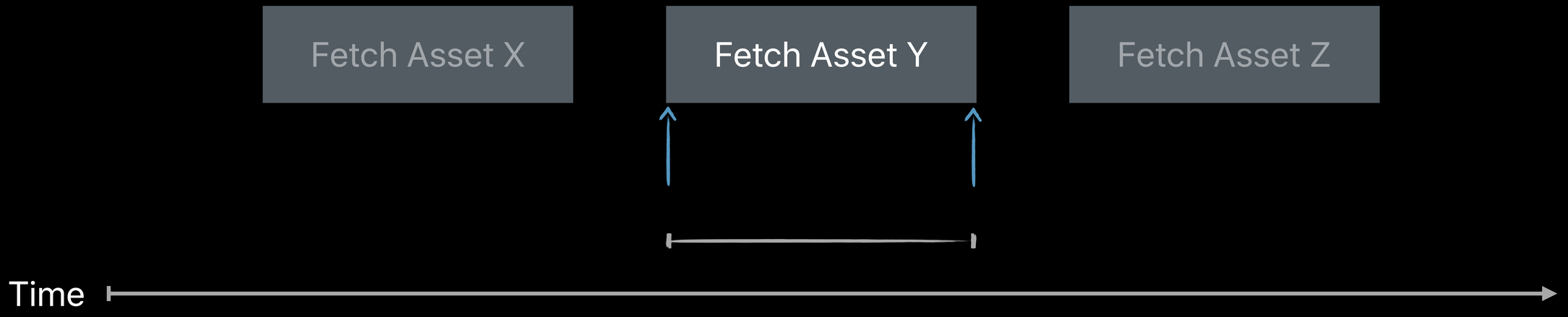
Controlling signposts

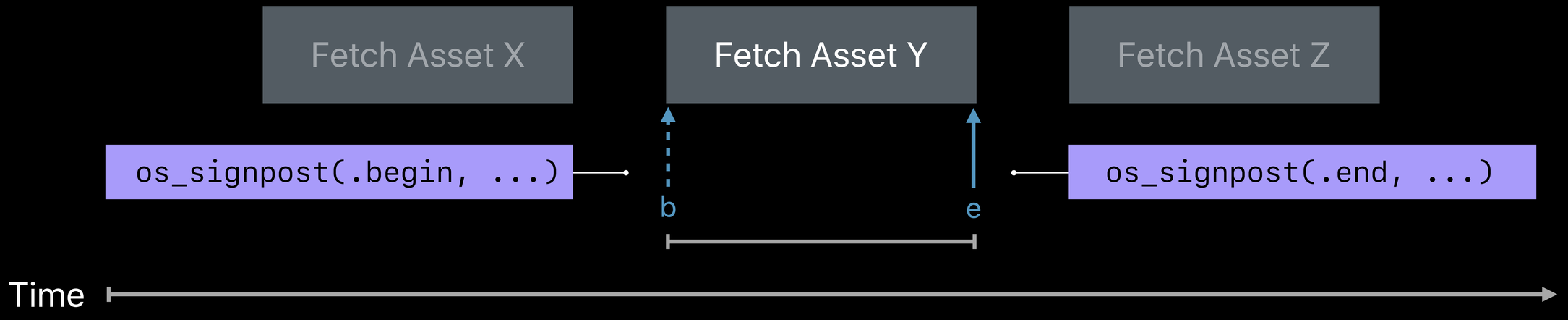
Investigating with Instruments

Measuring Intervals with Signposts


9:41 AM
100%
Trailblazer
Matt Davis Trail
Mt Tamalpais State Park, CA
534 Reviews
Details
957 Reviews
Details
703 Reviews
Details
Near Me
Map
Favorites
Activity
Checklist


Fetch Asset X
Fetch Asset Y
Fetch Asset Z
Time


Fetch Asset X
Fetch Asset Y
Fetch Asset Z
Time


Fetch Asset X
Fetch Asset Y
Fetch Asset Z
os_signpost(.begin, ...)
os_signpost(.end, ...)
b
e
Time

```
for element in panel.elements {

    fetchAsset(for: element)

}
```

```
import os.signpost



for element in panel.elements {

    fetchAsset(for: element)

}
```

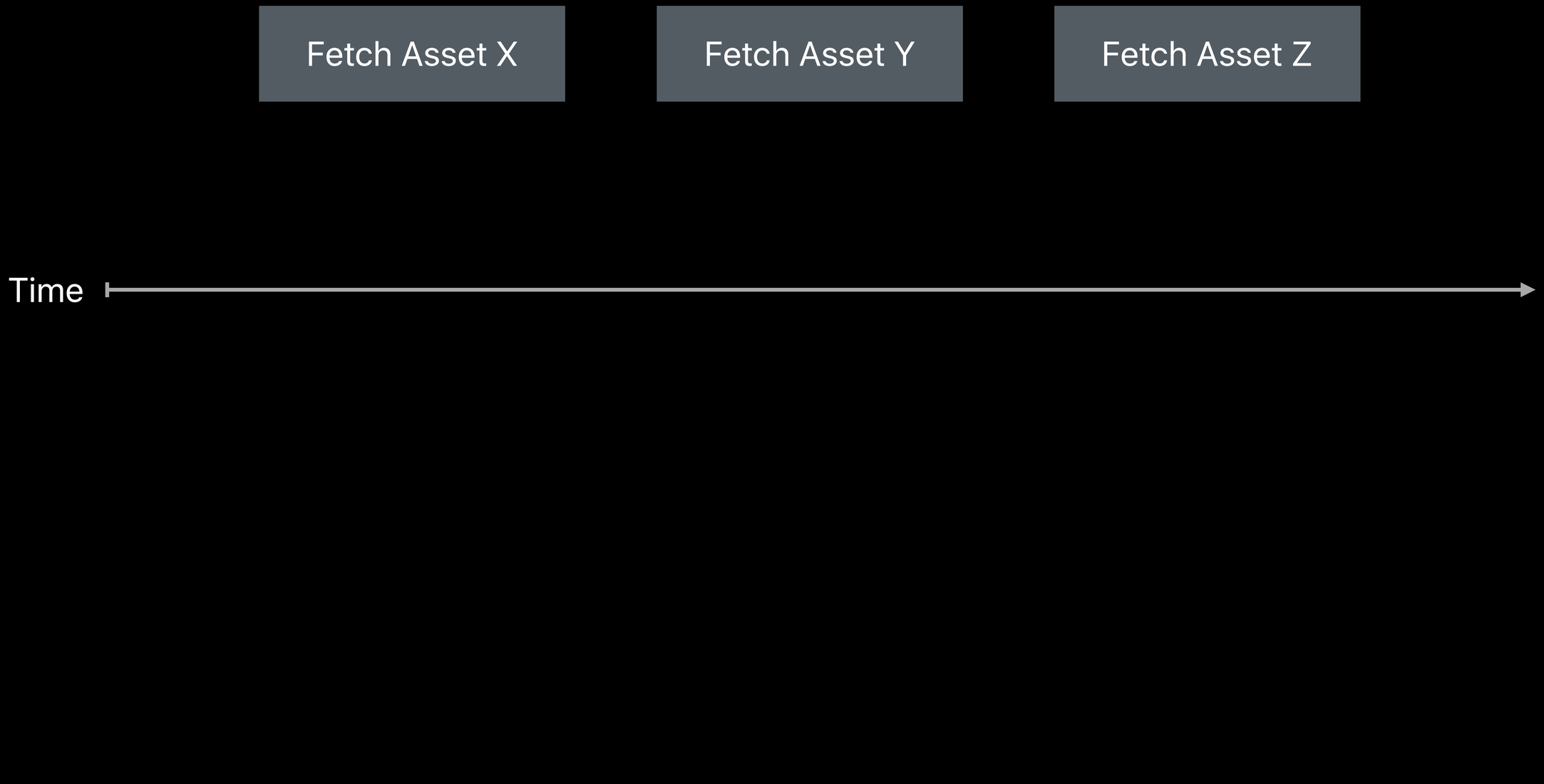
```
import os.signpost

let refreshLog = OSLog(subsystem: "com.example.your-app", category: "RefreshOperations")


for element in panel.elements {

    fetchAsset(for: element)

}
```

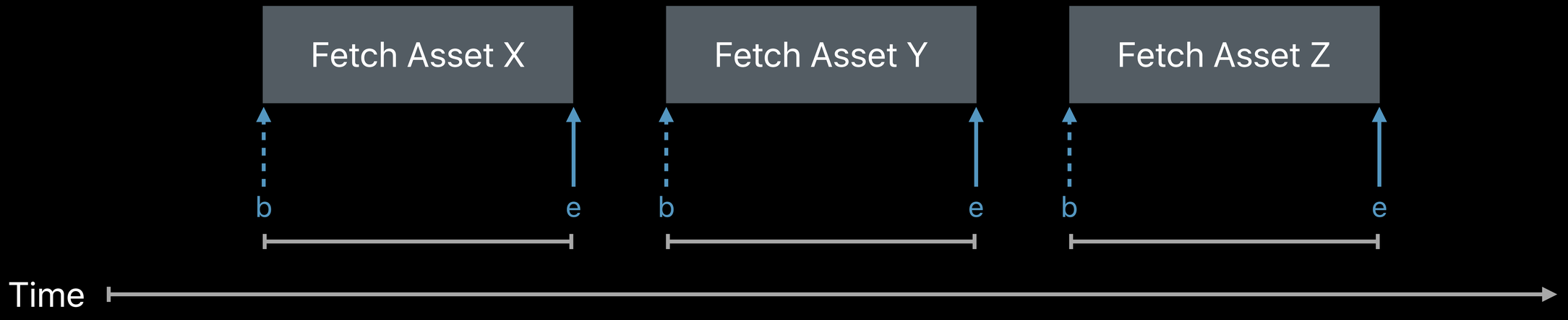
Category: Use for grouping

NEW

```
import os.signpost

let refreshLog = OSLog(subsystem: "com.example.your-app", category: "RefreshOperations")


for element in panel.elements {
    os_signpost(.begin, log: refreshLog, name: "Fetch Asset")
    fetchAsset(for: element)
    os_signpost(.end, log: refreshLog, name: "Fetch Asset")
}
```

Signpost name: A string literal that identifies interval

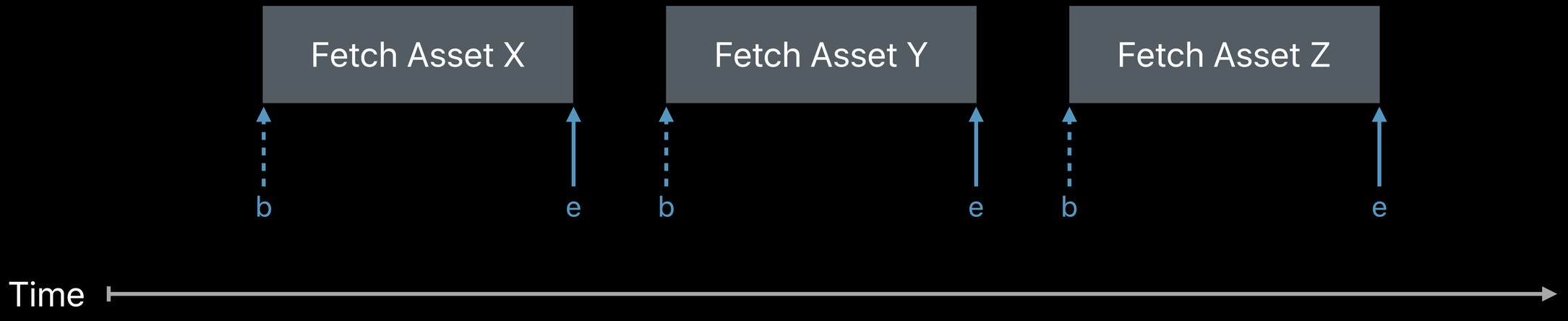

Fetch Asset X
Fetch Asset Y
Fetch Asset Z
Time

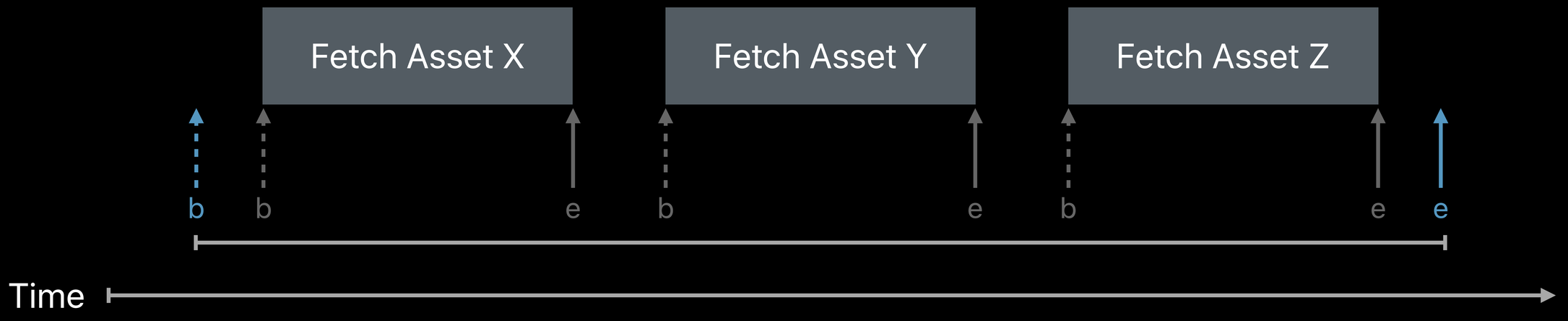

Fetch Asset X
Fetch Asset Y
Fetch Asset Z
b
e
b
e
b
e
Time

```
import os.signpost

let refreshLog = OSLog(subsystem: "com.example.your-app", category: "RefreshOperations")


for element in panel.elements {
    os_signpost(.begin, log: refreshLog, name: "Fetch Asset")
    fetchAsset(for: element)
    os_signpost(.end, log: refreshLog, name: "Fetch Asset")
}
```

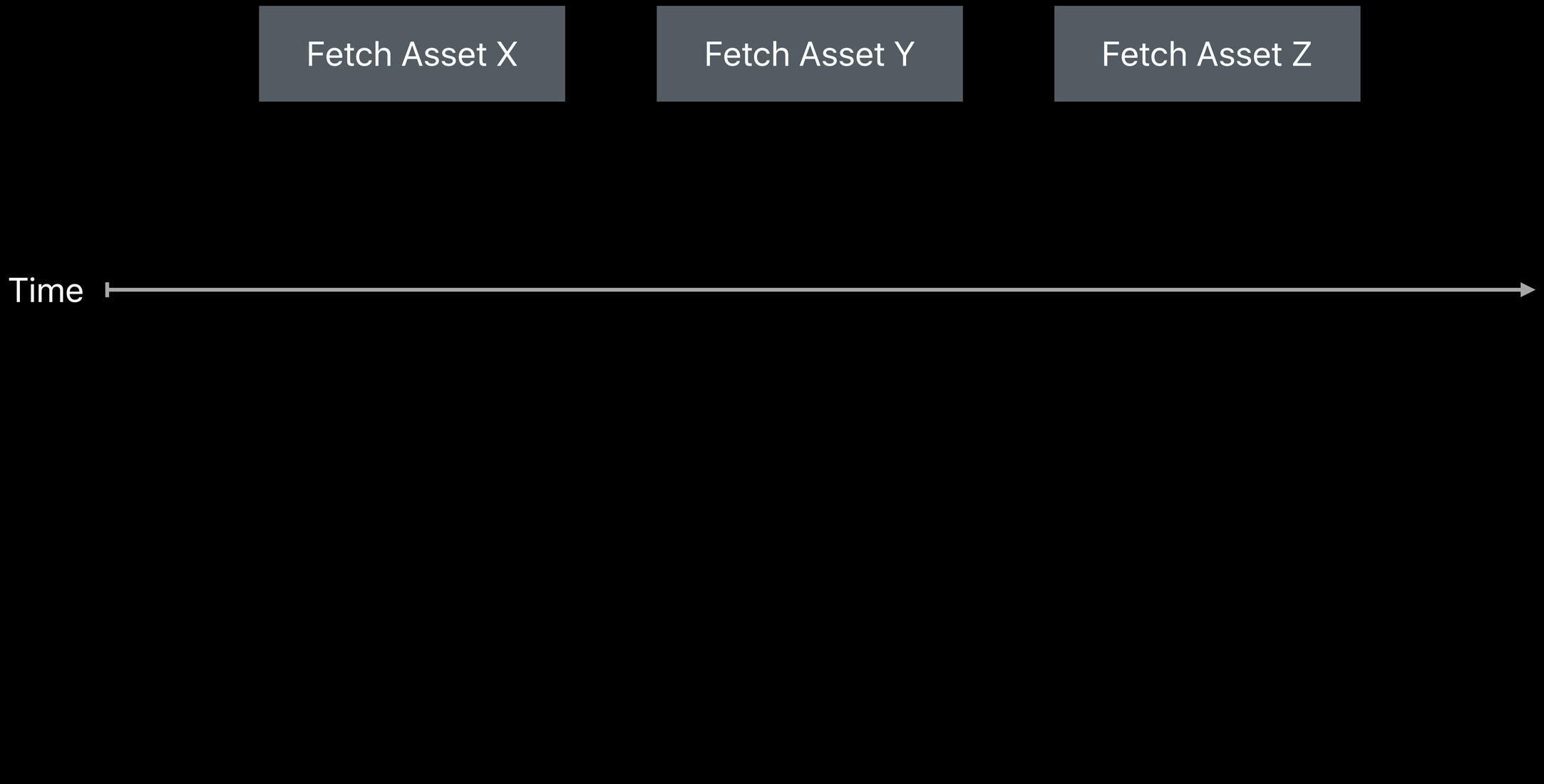
```
import os.signpost

let refreshLog = OSLog(subsystem: "com.example.your-app", category: "RefreshOperations")

os_signpost(.begin, log: refreshLog, name: "Refresh Panel")
for element in panel.elements {
    os_signpost(.begin, log: refreshLog, name: "Fetch Asset")
    fetchAsset(for: element)
    os_signpost(.end, log: refreshLog, name: "Fetch Asset")
}
os_signpost(.end, log: refreshLog, name: "Refresh Panel")
```

A different signpost name for this different interval

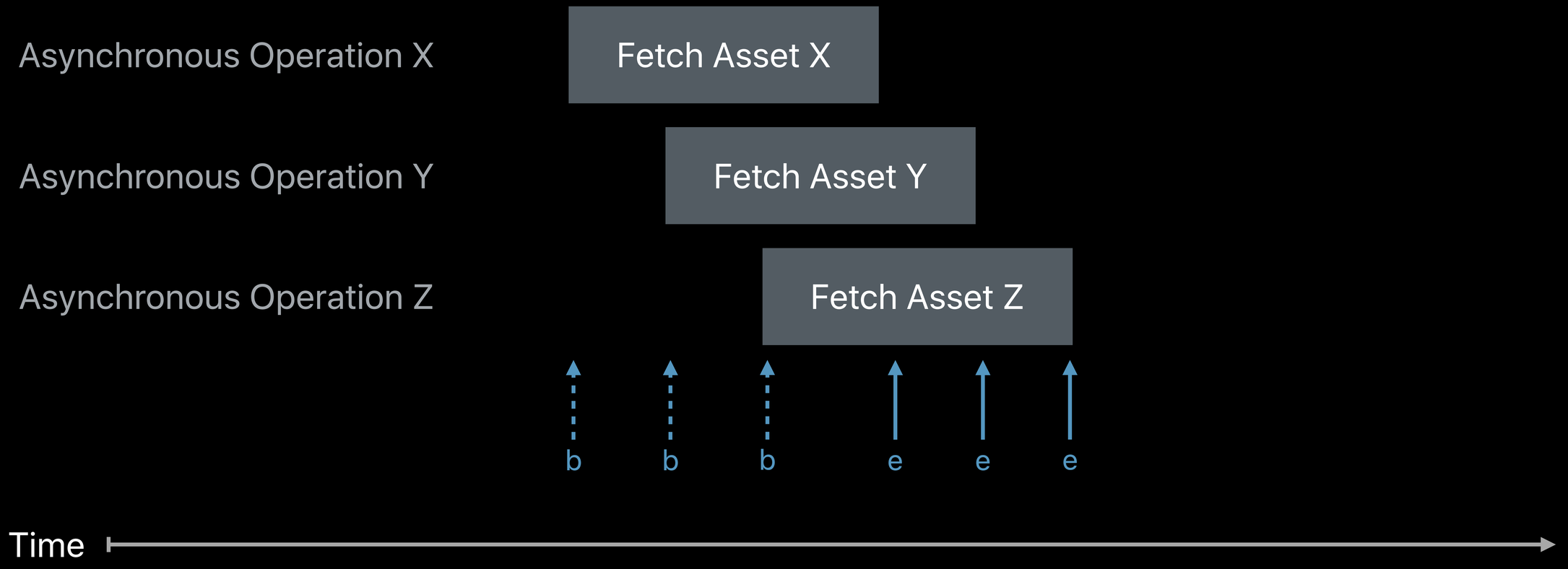

Fetch Asset X
Fetch Asset Y
Fetch Asset Z
b
e
b
e
b
e
Time


Fetch Asset X
Fetch Asset Y
Fetch Asset Z
b
b
e
b
e
b
e
e
Time

Measuring Asynchronous Intervals


Fetch Asset X
Fetch Asset Y
Fetch Asset Z
Time


Asynchronous Operation X
Fetch Asset X
Asynchronous Operation Y
Fetch Asset Y
Asynchronous Operation Z
Fetch Asset Z
Time


Asynchronous Operation X
Fetch Asset X
Asynchronous Operation Y
Fetch Asset Y
Asynchronous Operation Z
Fetch Asset Z
b
b
b
e
e
e
Time

Signpost Names

```
os_signpost(.begin, log: refreshLog, name: "Fetch Asset")
```

```
os_signpost(.end, log: refreshLog, name: "Fetch Asset")
```

The string literal identifies signpost intervals

The name must match at `.begin` and `.end`

Signpost IDs

```
let spid = OSSignpostID(log: refreshLog)
os_signpost(.begin, log: refreshLog, name: "Fetch Asset", signpostID: spid)
```

```
os_signpost(.end, log: refreshLog, name: "Fetch Asset", signpostID: spid)
```

Use signpost IDs to tell overlapping operations apart

While running, use the same IDs for each pair of `.begin` and `.end`

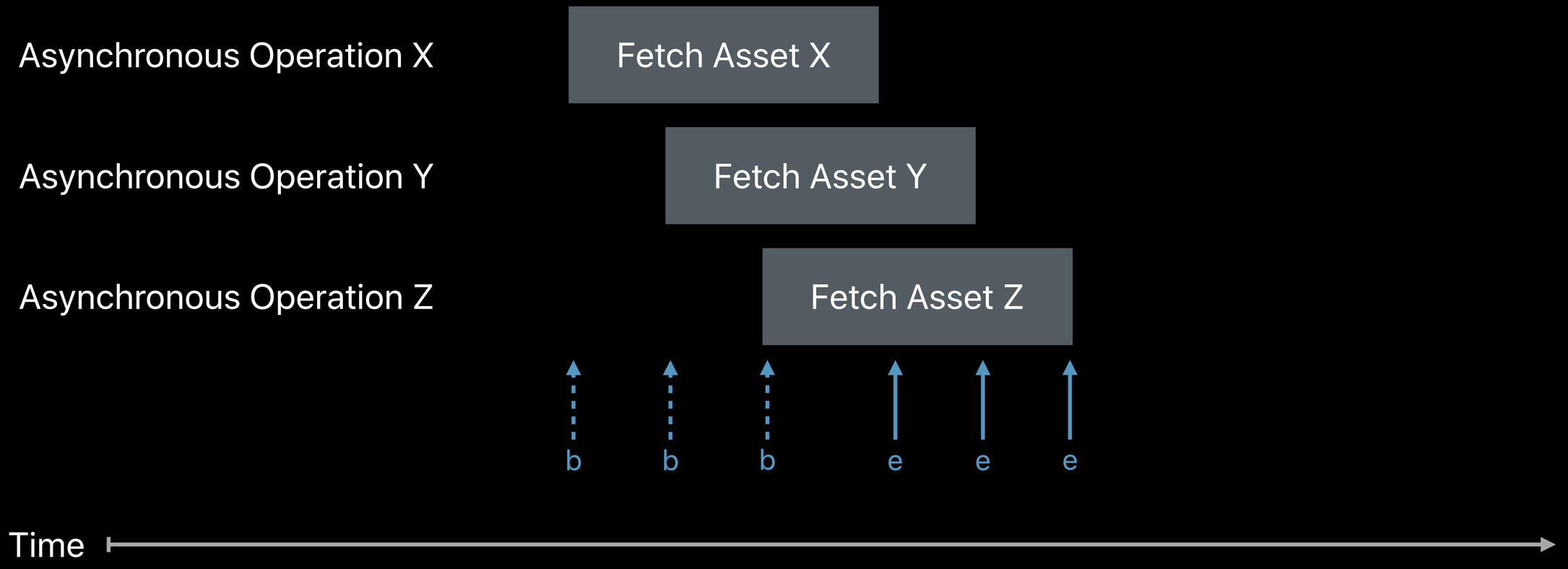
Making Signpost IDs

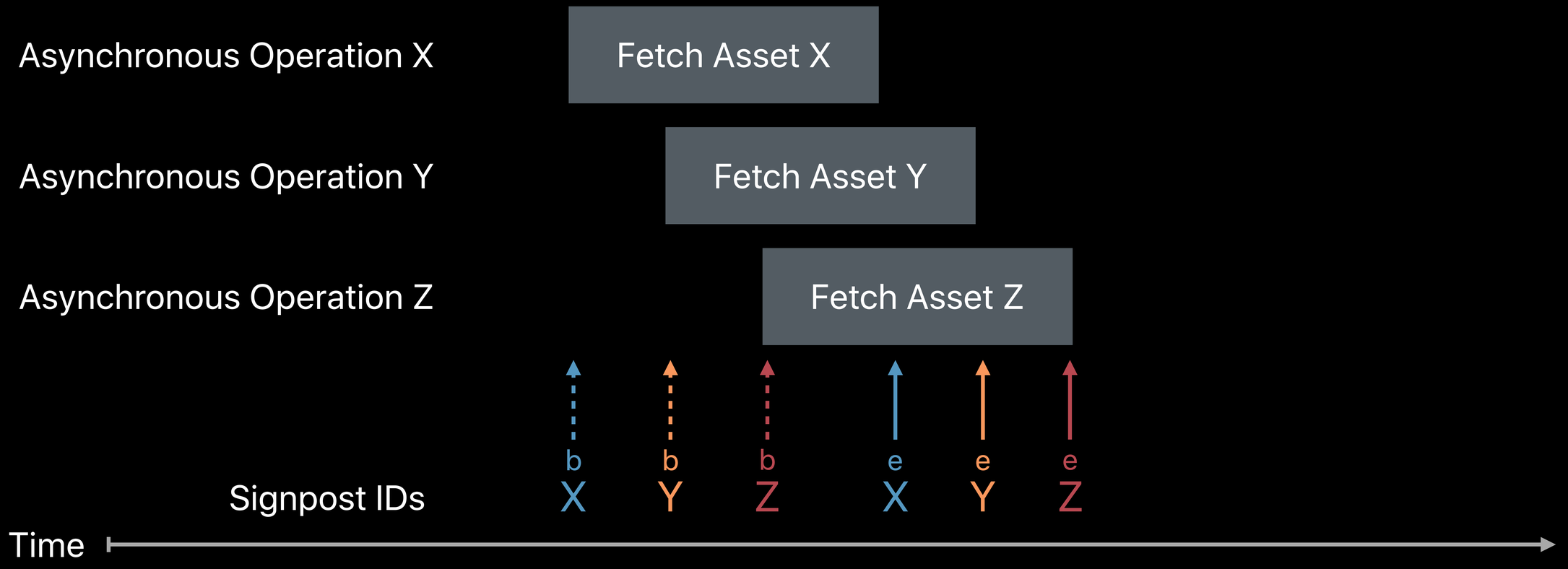
```
let spid = OSSignpostID(log: refreshLog)
```

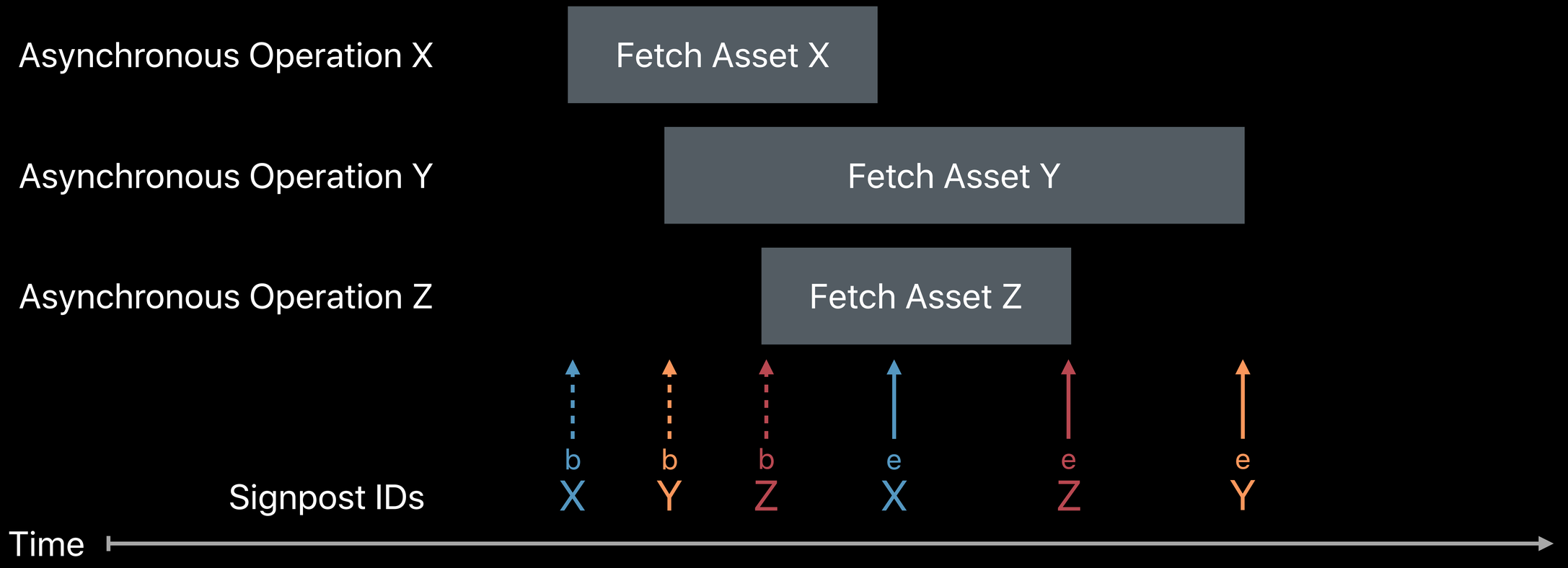
```
let spid = OSSignpostID(log: refreshLog, object: element)
```

Signpost IDs are process-scoped

Making from object is convenient if you have the same object at `.begin` and `.end`


Asynchronous Operation X
Fetch Asset X
Asynchronous Operation Y
Fetch Asset Y
Asynchronous Operation Z
Fetch Asset Z
b
b
b
e
e
e
Time


Asynchronous Operation X
Fetch Asset X
Asynchronous Operation Y
Fetch Asset Y
Asynchronous Operation Z
Fetch Asset Z
b
b
b
e
e
e
Signpost IDs
X
Y
Z
X
Y
Z
Time


Asynchronous Operation X
Fetch Asset X
Asynchronous Operation Y
Fetch Asset Y
Asynchronous Operation Z
Fetch Asset Z
b
X
b
Y
b
Z
e
X
e
Z
e
Y
Signpost IDs
Time

```
let refreshLog = OSLog(subsystem: "com.example.your-app", category: "RefreshOperations")


os_signpost(.begin, log: refreshLog, name: "Refresh Panel")

for element in panel.elements {

    os_signpost(.begin, log: refreshLog, name: "Fetch Asset")
    fetchAsset(for: element)
    os_signpost(.end, log: refreshLog, name: "Fetch Asset")

}

os_signpost(.end, log: refreshLog, name: "Refresh Panel")
```

```
let refreshLog = OSLog(subsystem: "com.example.your-app", category: "RefreshOperations")


os_signpost(.begin, log: refreshLog, name: "Refresh Panel")

for element in panel.elements {

    os_signpost(.begin, log: refreshLog, name: "Fetch Asset")
    fetchAssetAsync(for: element) {
        os_signpost(.end, log: refreshLog, name: "Fetch Asset")
    }
}
notifyWhenDone {
    os_signpost(.end, log: refreshLog, name: "Refresh Panel")
}
```

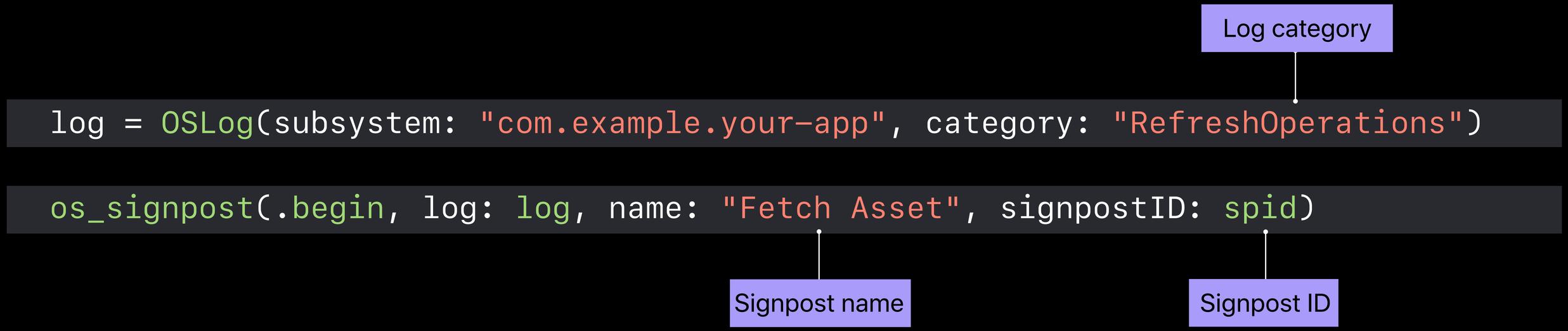
Completion handler for one asset

Completion handler for all assets

```
let refreshLog = OSLog(subsystem: "com.example.your-app", category: "RefreshOperations")

let spidForRefresh = OSSignpostID(log: refreshLog)
os_signpost(.begin, log: refreshLog, name: "Refresh Panel")

for element in panel.elements {
    let spid = OSSignpostID(log: refreshLog, object: element)
    os_signpost(.begin, log: refreshLog, name: "Fetch Asset")
    fetchAssetAsync(for: element) {
        os_signpost(.end, log: refreshLog, name: "Fetch Asset")
    }
}
notifyWhenDone {
    os_signpost(.end, log: refreshLog, name: "Refresh Panel")
}
```

```
let refreshLog = OSLog(subsystem: "com.example.your-app", category: "RefreshOperations")

let spidForRefresh = OSSignpostID(log: refreshLog)
os_signpost(.begin, log: refreshLog, name: "Refresh Panel", signpostID: spidForRefresh)

for element in panel.elements {
    let spid = OSSignpostID(log: refreshLog, object: element)
    os_signpost(.begin, log: refreshLog, name: "Fetch Asset", signpostID: spid)
    fetchAssetAsync(for: element) {
        os_signpost(.end, log: refreshLog, name: "Fetch Asset", signpostID: spid)
    }
}
notifyWhenDone {
    os_signpost(.end, log: refreshLog, name: "Refresh Panel", signpostID: spidForRefresh)
}
```

Organizing Signposts: A Hierarchy



	Example	Represents
Log category	`"RefreshOperations"`	Related operations
Signpost name	`"Fetch Asset"`	An operation to measure
Signpost ID	`spid`	Single interval

Adding Metadata to Signposts

Custom Metadata in Signpost Arguments

```
os_signpost(.begin, log: log, name: "Compute Physics")
```

Custom Metadata in Signpost Arguments

```
os_signpost(.begin, log: log, name: "Compute Physics",
    "for particle")
```

Add context to the `.begin` and `.end`

Custom Metadata in Signpost Arguments

```
os_signpost(.begin, log: log, name: "Compute Physics",
    "%d  %d  %d  %d",
    x1, y1, x2, y2)
```

Add context to the `.begin` and `.end`

Pass arguments with os_log format string literal

Custom Metadata in Signpost Arguments

```
os_signpost(.begin, log: log, name: "Compute Physics",
    "%.1f  %.1f  %.2f  %.1f  %.1f",
    x1, y1, m, x2, y2)
```

Add context to the `.begin` and `.end`

Pass arguments with os_log format string literal

Pass many arguments with different types

Custom Metadata in Signpost Arguments

```
os_signpost(.begin, log: log, name: "Compute Physics",
    "%{public}s  %.1f  %.1f  %.2f  %.1f  %.1f",
    description, x1, y1, m, x2, y2)
```

Add context to the `.begin` and `.end`

Pass arguments with os_log format string literal

Pass many arguments with different types

Pass dynamic strings

Custom Metadata in Signpost Arguments

```
os_signpost(.begin, log: log, name: "Compute Physics",
    "for %{public}s at (%.1f, %.1f) with mass %.2f and velocity (%.1f, %.1f)",
    description, x1, y1, m, x2, y2)
```

Add context to the `.begin` and `.end`

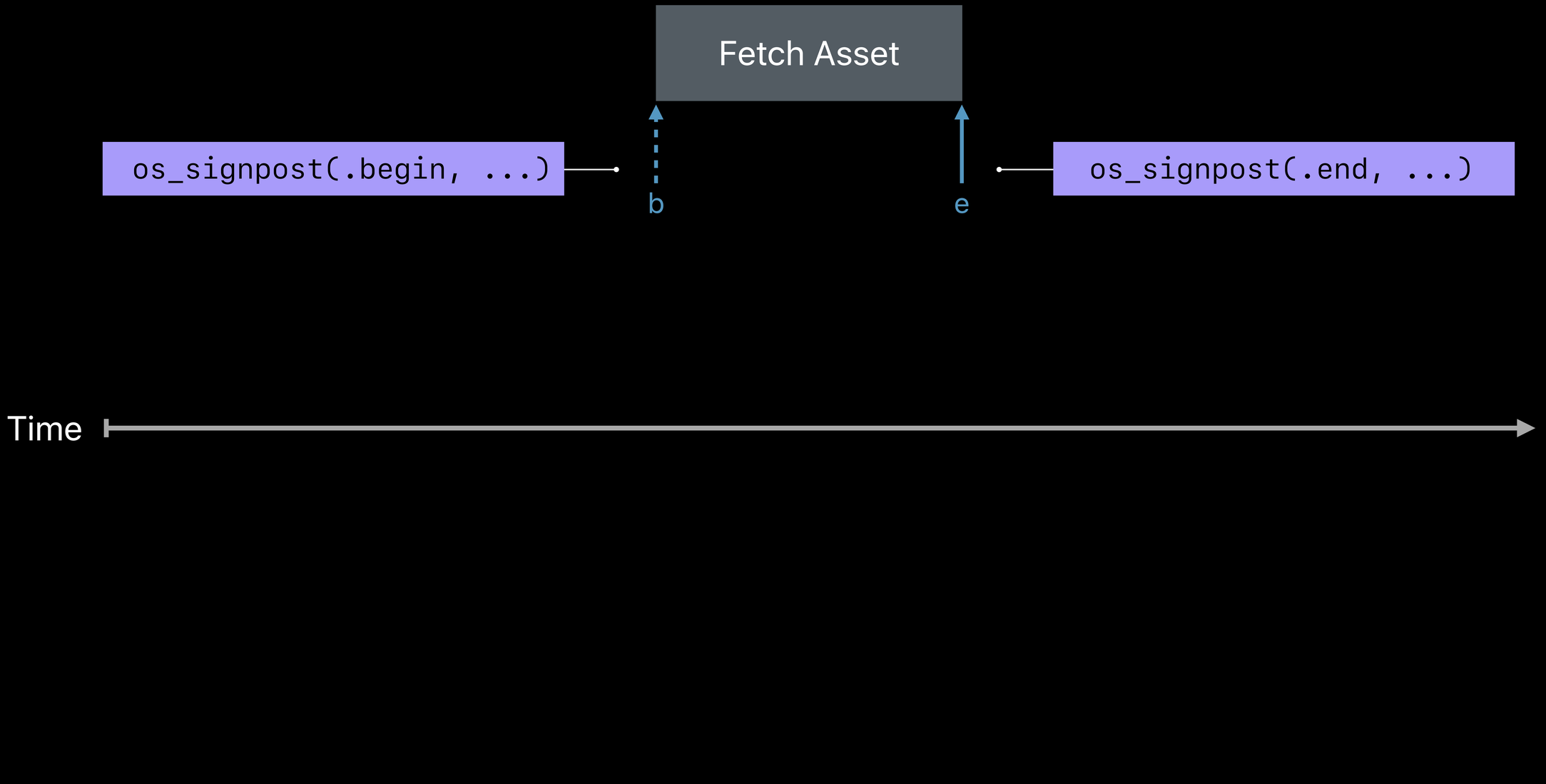
Pass arguments with os_log format string literal

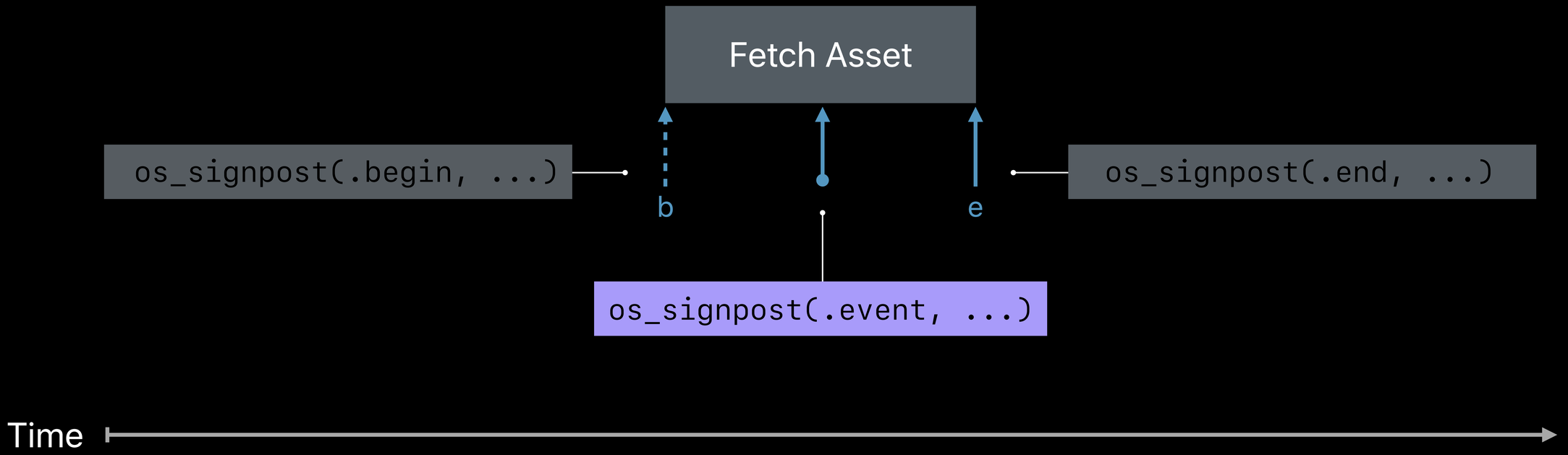
Pass many arguments with different types

Pass dynamic strings

The format string is a fixed cost, so feel free to be descriptive!

Adding Independent Events


Fetch Asset
os_signpost(.begin, ...)
os_signpost(.end, ...)
b
e
Time

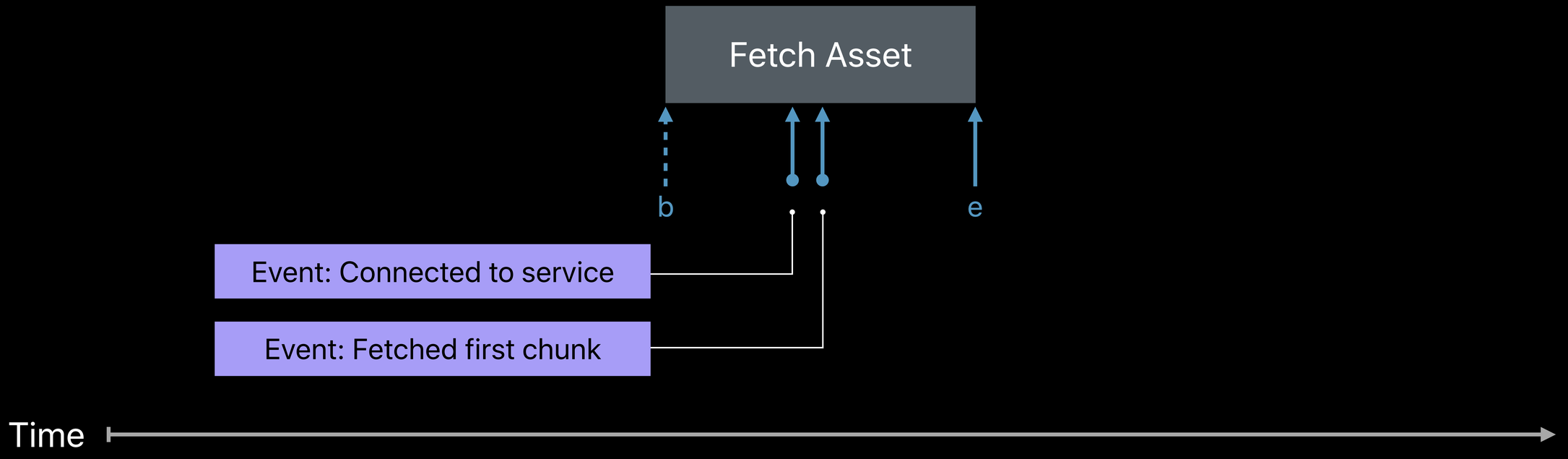

Fetch Asset
os_signpost(.begin, ...)
os_signpost(.end, ...)
b
e
os_signpost(.event, ...)
Time

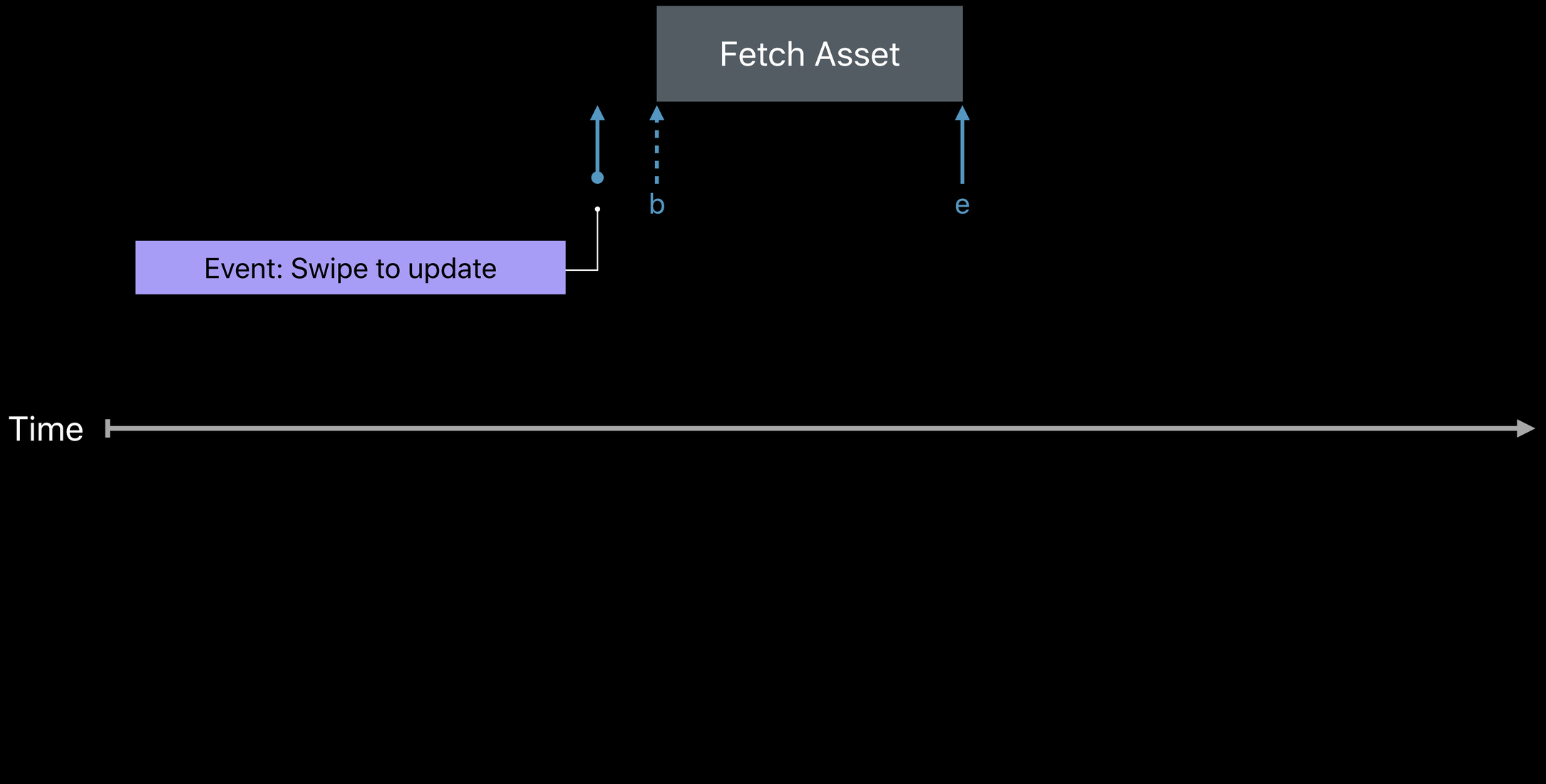
Signpost Events

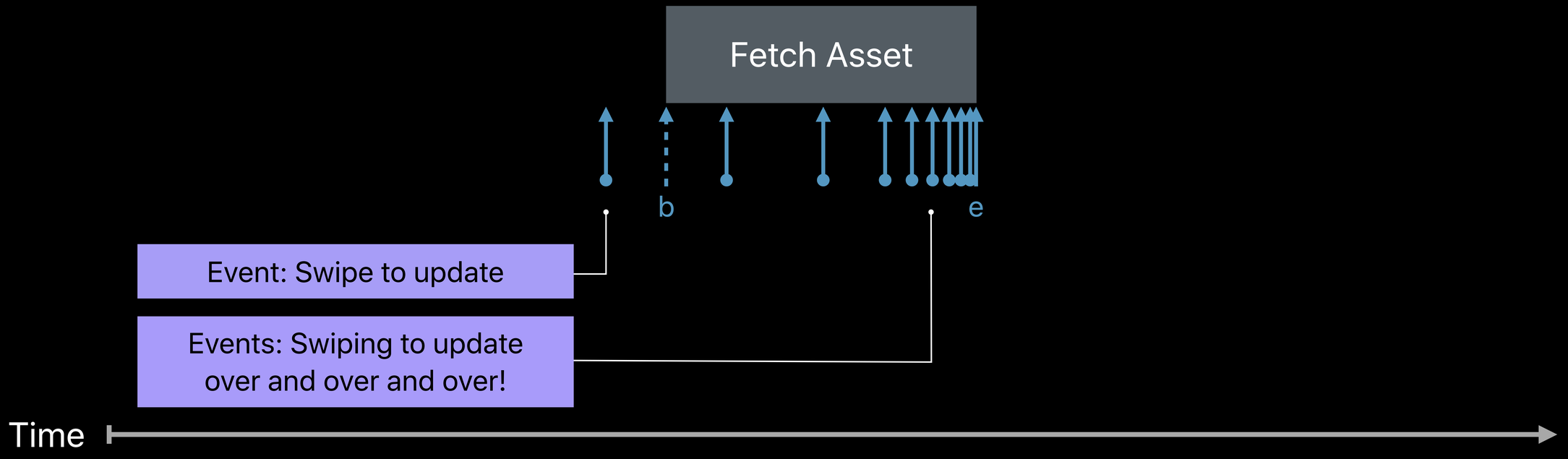
```
os_signpost(.event, log: log, name: "Fetch Asset",
    "Fetched first chunk, size %u", size)
```

```
os_signpost(.event, log: log, name: "Swipe",
    "For action 0x%x", actionCode)
```

Marking a single point in time


Fetch Asset
b
e
Event: Connected to service
Event: Fetched first chunk
Time


Fetch Asset
b
e
Event: Swipe to update
Time


Fetch Asset
b
e
Event: Swipe to update
Events: Swiping to update over and over and over!
Time

Conditionally Enabling Signposts

Signposts Are Lightweight

Built to minimize observer effect

Built for fine-grained measurement in a short time span

Enabling and Disabling Signpost Categories

```
OSLog.disabled
```

Take advantage of special log handle

Just change the handle—can leave calling sites alone

```
let refreshLog = OSLog(subsystem: "com.example.your-app", category: "RefreshOperations")
```

```
os_signpost(.begin, log: refreshLog, name: "Refresh Panel")
for element in panel.elements {
    os_signpost(.begin, log: refreshLog, name: "Fetch Asset")
    fetchAsset(for: element)
    os_signpost(.end, log: refreshLog, name: "Fetch Asset")
}
os_signpost(.end, log: refreshLog, name: "Refresh Panel")
```

```
let refreshLog: OSLog
if ProcessInfo.processInfo.environment.keys.contains("SIGNPOSTS_FOR_REFRESH") {
    refreshLog = OSLog(subsystem: "com.example.your-app", category: "RefreshOperations")
} else {
    refreshLog = .disabled
}
```

```
os_signpost(.begin, log: refreshLog, name: "Refresh Panel")
for element in panel.elements {
    os_signpost(.begin, log: refreshLog, name: "Fetch Asset")
    fetchAsset(for: element)
    os_signpost(.end, log: refreshLog, name: "Fetch Asset")
}
os_signpost(.end, log: refreshLog, name: "Refresh Panel")
```

Instrumentation-Specific Code

```
if refreshLog.signpostsEnabled {
    let information = copyDescription()
    os_signpost(..., information)
}
```

For additional expensive code that is only useful for the signpost

Signposts in C

Signposts in C

Swift	C
`import os.signpost`	`#include <os/signpost.h>`
`OSLog`	`os_log_t, os_log_create()`
`.disabled`	`OS_LOG_DISABLED`
`os_signpost(.begin, ...)`	`os_signpost_interval_begin()`
`os_signpost(.end, ...)`	`os_signpost_interval_end()`
`os_signpost(.event, ...)`	`os_signpost_event_emit()`
`OSSignpostID`	`os_signpost_id_t`

Instruments

Chad Woolf, Instruments

Instruments 10

NEW

Instruments 10

NEW

os_signpost

Instruments 10

NEW

os_signpost

Points of Interest

Instruments 10

NEW

os_signpost

Points of Interest

Custom instruments

Demo

Visualizing signpost data

Summary

Annotate code with signposts

- Easily mark intervals
- Capture metadata of interest

Use Instruments to view signpost data

- Visualize where time is spent
- Understand what program is doing

More Information

https://developer.apple.com/wwdc18/405

Creating Custom Instruments Lab	Technology Lab 8	Wednesday 3:00PM
Creating Custom Instruments	Hall 1	Thursday 11:00AM


WWDC18