```swift
import Foundation
import _Concurrency
import _StringProcessing
import _SwiftConcurrencyShims

extension OSLogEntry {

    /**
     * @enum OSLogEntryStoreCategory
     *
     * @abstract
     * A classification of how the entry was to be stored and
     * rotated at the point when it was created.
     *
     * @discussion
     * The unified logging system keeps entries in one of two
     * places: a ring buffer in memory and a persisted data store.
     * Entries are rotated out of both places to free up resources.
     * This rotation is not strictly aligned with entries'
     * timestamps --- in particular, they can be rotated in bulk,
     * and they are rotated according to a series of heuristics that
     * take into account space, time, and how the entries were
     * classified.
     *
     * @constant
OSLogEntryStoreCategoryMetadata
     * This entry was generated as
information about the other
     * entries or about the sequence of
entries as a whole.
     *
     * @constant
OSLogEntryStoreCategoryShortTerm
     * This entry was not intended to be
long-lived and was captured
     * in the ring buffer.
     *
     * @constant
OSLogEntryStoreCategoryLongTermAuto
     * The entry was intended to be
persisted in a filesystem-backed
     * data store and kept mainly based
on the amount of space
     * available.
     *
     * @constant
OSLogEntryStoreCategoryLongTerm1
     * @constant
OSLogEntryStoreCategoryLongTerm3
     * @constant
OSLogEntryStoreCategoryLongTerm7
     * @constant
OSLogEntryStoreCategoryLongTerm14
     * @constant
OSLogEntryStoreCategoryLongTerm30
     * The entry was tagged with a hint
indicating that the system
```

```
     * should try to preserve it for a
certain amount of time. It
     * was persisted in the filesystem-
backed data store, and
     * rotation of these entries was
based on both time and space
     * considerations.
     */
    @available(macOS 10.15, *)
    public enum StoreCategory : Int,
@unchecked Sendable {

        case undefined = 0

        case metadata = 1

        case shortTerm = 2

        case longTermAuto = 3

        case longTerm1 = 4

        case longTerm3 = 5

        case longTerm7 = 6

        case longTerm14 = 7

        case longTerm30 = 8
    }
}

/**
```

```
 * @class OSLogEntry
 *
 * @abstract
 * A single entry from the unified
logging system.
 */
@available(macOS 10.15, *)
open class OSLogEntry : NSObject {

    /**
     * @property composedMessage
     *
     * @abstract
     * The fully formatted message for
the entry.
     */
    @available(macOS 10.15, *)
    open var composedMessage: String {
get }

    /**
     * @property date
     *
     * @abstract
     * The timestamp of the entry.
     */
    @available(macOS 10.15, *)
    open var date: Date { get }

    /**
     * @property storeCategory
     *
     * @abstract
     * This entry's storage tag. See
OSLogEntryStoreCategory.
     */
    @available(macOS 10.15, *)
    open var storeCategory:
OSLogEntry.StoreCategory { get }
}

/**
 * @protocol OSLogEntryFromProcess
 *
 * @abstract
 * Entry subclasses conforming to this
protocol represent data
 * that are generated from a process;
they have metadata about
 * the originator.
 */
@available(macOS 10.15, *)
public protocol OSLogEntryFromProcess {

    /**
     * @property activityIdentifier
     *
     * @abstract
     * The activity ID associated with
the entry.
     */
    @available(macOS 10.15, *)
    var activityIdentifier:
os_activity_id_t { get }

    /**
```

```
     * @property process
     *
     * @abstract
     * The name of the process that made
the entry.
     */
    @available(macOS 10.15, *)
    var process: String { get }

    /**
     * @property processIdentifier
     *
     * @abstract
     * The pid of the process that made
the entry.
     */
    @available(macOS 10.15, *)
    var processIdentifier: pid_t { get }

    /**
     * @property sender
     *
     * @abstract
     * The name of the binary image that
made the entry.
     */
    @available(macOS 10.15, *)
    var sender: String { get }

    /**
     * @property threadIdentifier
     *
     * @abstract
     * The tid of the thread that made the entry.
     */
    @available(macOS 10.15, *)
    var threadIdentifier: UInt64 { get }
}

/**
 * @protocol OSLogEntryWithPayload
 *
 * @abstract
 * Entry subclasses conforming to this protocol represent
 * entries that were made using a handle and a format string.
 */
@available(macOS 10.15, *)
public protocol OSLogEntryWithPayload {

    /**
     * @property category
     *
     * @abstract
     * The category from the os_log_t handle used.
     */
    @available(macOS 10.15, *)
    var category: String { get }

    /**
     * @property components
     *
     * @abstract
     * An array of the various parts of
the composed message.
     */
    @available(macOS 10.15, *)
    var components:
[OSLogMessageComponent] { get }

    /**
     * @property formatString
     *
     * @abstract
     * The format string used.
     */
    @available(macOS 10.15, *)
    var formatString: String { get }

    /**
     * @property subsystem
     *
     * @abstract
     * The subsystem of the os_log_t
handle used.
     */
    @available(macOS 10.15, *)
    var subsystem: String { get }
}

@available(macOS 12.0, *)
extension OSLogEntry : NSSecureCoding {
}

/**
 * @class OSLogEntryActivity
 *
 * @abstract
 * An entry generated by an activity
event.
 */
@available(macOS 10.15, *)
open class OSLogEntryActivity :
OSLogEntry, OSLogEntryFromProcess {

    /**
     * @property parentActivityIdentifier
     *
     * @abstract
     * This parent activity's activity
ID.
     */
    @available(macOS 10.15, *)
    open var parentActivityIdentifier:
os_activity_id_t { get }
}

/**
 * @class OSLogEntryBoundary
 *
 * @abstract
 * This entry represents metadata that
partitions sequences of
 * other entries.
 *
 * @discussion
 * For example, this kind of entry is
used for boot boundaries.
 * The data here are currently
```

```
informational and carried in the
 * composedMessage property.
 */
@available(macOS 10.15, *)
open class OSLogEntryBoundary :
OSLogEntry {
}

extension OSLogEntryLog {

    /**
     * @enum OSLogEntryLogLevel
     *
     * @abstract
     * The level that this entry was
generated at.
     */
    @available(macOS 10.15, *)
    public enum Level : Int, @unchecked
Sendable {

        case undefined = 0

        case debug = 1

        case info = 2

        case notice = 3

        case error = 4

        case fault = 5
    }
```

```
}

/**
 * @class OSLogEntryLog
 *
 * @abstract
 * Entries made by the os_log API.
 */
@available(macOS 10.15, *)
open class OSLogEntryLog : OSLogEntry,
OSLogEntryFromProcess,
OSLogEntryWithPayload {

    /**
     * @property level
     *
     * @abstract
     * The level of the entry, e.g.,
info, debug.
     */
    @available(macOS 10.15, *)
    open var level: OSLogEntryLog.Level {
get }
}

extension OSLogEntrySignpost {

    /**
     * @enum OSLogEntrySignpostType
     *
     * @discussion
     * The kind of of signpost emitted.
     */
    @available(macOS 10.15, *)
    public enum SignpostType : Int,
@unchecked Sendable {

        case undefined = 0

        case intervalBegin = 1

        case intervalEnd = 2

        case event = 3
    }
}

/**
 * @class OSLogEntrySignpost
 *
 * @abstract
 * Entries made by the os_signpost API.
 */
@available(macOS 10.15, *)
open class OSLogEntrySignpost :
OSLogEntry, OSLogEntryFromProcess,
OSLogEntryWithPayload {

    /**
     * @property signpostIdentifier
     *
     * @abstract
     * The signpost ID associated with
this entry.
     */
    @available(macOS 10.15, *)
    open var signpostIdentifier:
os_signpost_id_t { get }

    /**
     * @property signpostName
     *
     * @abstract
     * The signpost name associated with
this entry.
     */
    @available(macOS 10.15, *)
    open var signpostName: String { get }

    /**
     * @property signpostType
     *
     * @abstract
     * The signpost type associated with
this entry.
     */
    @available(macOS 10.15, *)
    open var signpostType:
OSLogEntrySignpost.SignpostType { get }
}

extension OSLogEnumerator {

    /**
     * @enum OSLogEnumeratorOptions
     *
     * @abstract
     * Control the direction of the
iteration.
```

```
     *
     * @constant OSLogEnumeratorReverse
     * Iterate backward in time. If no
starting position is specified,
     * start at the latest entry.
     */
    @available(macOS 10.15, *)
    public struct Options : OptionSet,
@unchecked Sendable {

        public init(rawValue: UInt)

        public static var reverse:
OSLogEnumerator.Options { get }
    }
}

/**
 * @class OSLogEnumerator
 *
 * @abstract
 * An enumerator that views entries in
the unified logging system.
 */
@available(macOS 10.15, *)
open class OSLogEnumerator : NSEnumerator
{
}

extension OSLogMessageComponent {

    /**
     * @enum
```

```
OSLogMessageComponentArgumentCategory
     *
     * @abstract
     * The kind of data corresponding to
an argument in a message
     * payload, like the number
associated with a "%d" placeholder.
     * This value can be undefined if the
argument data cannot be
     * decoded; for example, it may be
redacted.
     */
    @available(macOS 10.15, *)
    public enum ArgumentCategory : Int,
@unchecked Sendable {

        case undefined = 0

        case data = 1

        case double = 2

        case int64 = 3

        case string = 4

        case uInt64 = 5
    }
}

/**
 * @class OSLogMessageComponent
 *
 * @abstract
 * The message arguments for a particular
entry. There is one
 * component for each placeholder in the
formatString plus one
 * component for any text after the last
placeholder.
 */
@available(macOS 10.15, *)
open class OSLogMessageComponent :
NSObject, NSSecureCoding {

    /**
     * @property formatSubstring
     *
     * @abstract
     * The text immediately preceding a
placeholder. This can be an
     * empty string if there is nothing
between two placeholders, or
     * between the placeholder and the
bounds of the string.
     */
    @available(macOS 10.15, *)
    open var formatSubstring: String {
get }

    /**
     * @property placeholder
     *
     * @abstract
     * The placeholder text. Is empty for
is the last component.
```

```
     */
    @available(macOS 10.15, *)
    open var placeholder: String { get }

    /**
     * @property argumentCategory
     *
     * @abstract
     * The type of argument corresponding
to the placeholder; see
     *
OSLogMessageComponentArgumentCategory.
     */
    @available(macOS 10.15, *)
    open var argumentCategory:
OSLogMessageComponent.ArgumentCategory {
get }

    /**
     * @property argumentDataValue
     *
     * @abstract
     * The argument as a sequence of
bytes. Can be nil if the
     * argument cannot be decoded (for
example, it could be
     * redacted), or if this is the last
component.
     */
    @available(macOS 10.15, *)
    open var argumentDataValue: Data? {
get }
```

```
    /**
     * @property argumentDoubleValue
     *
     * @abstract
     * The argument as a double-precision
floating point number; the
     * value is undefined if the argument
cannot be decoded or if this
     * is the last component.
     */
    @available(macOS 10.15, *)
    open var argumentDoubleValue: Double
{ get }

    /**
     * @property argumentInt64Value
     *
     * @abstract
     * The argument as a 64-bit signed
integer; the value is undefined
     * if it cannot be decoded or if this
is the last component.
     */
    @available(macOS 10.15, *)
    open var argumentInt64Value: Int64 {
get }

    /**
     * @property argumentNumberValue
     *
     * @abstract
     * The argument as a number. Can be
nil if the argument cannot
```

```
     * be decoded (for example, it could
be redacted), or if this is
     * the last component.
     */
    @available(macOS 10.15, *)
    open var argumentNumberValue:
NSNumber? { get }

    /**
     * @property argumentStringValue
     *
     * @abstract
     * The argument as a string. Can be
nil if the argument cannot
     * be decoded (for example, it could
be redacted), or if this is
     * the last component.
     */
    @available(macOS 10.15, *)
    open var argumentStringValue: String?
{ get }

    /**
     * @property argumentUInt64Value
     *
     * @abstract
     * The argument as a 64-bit unsigned
integer; the value is
     * undefined if the argument cannot
be decoded or if this is the
     * last component.
     */
    @available(macOS 10.15, *)
    open var argumentUInt64Value: UInt64
{ get }
}

@available(macOS 10.15, iOS 15.0, watchOS
8.0, tvOS 15.0, *)
extension OSLogMessageComponent {

    public enum Argument {

        case undefined

        case data(Data)

        case double(Double)

        case signed(Int64)

        case string(String)

        case unsigned(UInt64)
    }

    public var argument:
OSLogMessageComponent.Argument { get }
}

/**
 * @class OSLogPosition
 *
 * @abstract
 * An opaque abstraction representing a
point in a sequence of
```

```
 * entries in the unified logging system.
 *
 * @discussion
 * Generate positions with OSLogStore
instance methods and use them
 * to start viewing entries from a
particular starting point.
 */
@available(macOS 10.15, *)
open class OSLogPosition : NSObject {
}

extension OSLogStore {

    /**
     * @enum OSLogStoreScope
     *
     * @abstract
     * Create a store to a subset of the
libtrace entries.
     *
     * @constant OSLogStoreSystem
     * @constant
OSLogStoreCurrentProcessIdentifier
     * "System" scope indicates the
entire system; i.e., all logs. Entries
can be
     * retrieved for the current calling
process, i.e., matching pid.
     */
    @available(macOS 12.0, *)
    public enum Scope : Int, @unchecked
Sendable {
        @available(macOS 12.0, *)
        case system = 0

        case currentProcessIdentifier = 1
    }
}

/**
 * @class OSLogStore
 *
 * @abstract
 * A set of entries from the unified
logging system. Instances
 * represent a fixed range of entries and
may be backed by a
 * logarchive or the Mac's local store.
 *
 * @discussion
 * Entries in OSLogStore objects are used
by OSLogEnumerator
 * instances; one store can support
multiple OSLogEnumerator
 * instances concurrently.
 */
@available(macOS 10.15, *)
open class OSLogStore : NSObject {

    /**
     * @method localStoreAndReturnError
     *
     * @abstract
     * Create an OSLogStore representing
```

```
the Mac's local store.
     *
     * @param error
     * If initialization is unsuccessful
--- for example, this process
     * does not have access to local logs
--- return nil and set this
     * parameter to a pointer to an error
object describing the reason.
     *
     * @discussion
     * This enables processing of a
sequence of logs as of the particular
     * point in time when this object is
created.
     *
     * Gaining access to the local
unified logging system requires
     * permission from the system. The
caller must be run by an admin
     * account.
     */
    @available(macOS 10.15, *)
    open class func local() throws ->
Self

    /**
     * @method storeWithScope
     *
     * @abstract
     * Create an OSLogStore for a subset
of entries in the local store.
     *
     * @param scope
     * The kind of subset the OSLogStore
is for.
     *
     * @param error
     * If initialization is unsuccessful,
return nil and set this parameter to a
     * pointer to an error object that
describes the reason.
     */
    @available(macOS 12.0, *)
    public convenience init(scope:
OSLogStore.Scope) throws

    /**
     * @method storeWithURL
     *
     * @abstract
     * Create an OSLogStore based on a
logarchive.
     *
     * @param url
     * The path identifying a logarchive
to be read.
     *
     * @param error
     * If initialization is unsuccessful
--- for example, the path is not
     * to a valid logarchive or the
logarchive is not compatible because
     * it is from a newer version ---
return nil and set this parameter
     * to a pointer to an error object
```

```
that describes the reason.
     */
    @available(macOS 10.15, *)
    public convenience init(url: URL)
throws

    @available(macOS, introduced: 10.15,
deprecated: 12.0, message: "Use one of
the factory methods")
    public init()

    /**
     * @method positionWithDate
     *
     * @abstract
     * Return a position representing the
time specified.
     *
     * @param date
     * The date to look for.
     *
     * @discussion
     * If there are multiple occurences
of the same time --- if, for
     * example, there was a time change
during the range of entries
     * --- the earliest occurrence is
used.
     */
    @available(macOS 10.15, *)
    open func position(date: Date) ->
OSLogPosition
```

```
    /**
     * @method positionWithTimeIntervalSinceEnd
     *
     * @abstract
     * Return a position representing an offset since the end of the time
     * range that the entries span.
     *
     * @param seconds
     * The seconds to add to the last time point in the range of entries.
     */
    @available(macOS 10.15, *)
    open func position(timeIntervalSinceEnd seconds: TimeInterval) -> OSLogPosition

    /**
     * @method positionWithTimeIntervalSinceLatestBoot
     *
     * @abstract
     * Return a position representing time since the last boot in the
     * series of entries.
     *
     * @param seconds
     * The seconds to add to the boot time point in the log time range.
     *
     * @discussion
     * Negative seconds would create an
```

```
ambiguous or imprecise position;
     * this function asserts that the
interval is positive.
     */
    @available(macOS 10.15, *)
    open func
position(timeIntervalSinceLatestBoot
seconds: TimeInterval) -> OSLogPosition
}

@available(macOS 10.15, iOS 15.0, watchOS
8.0, tvOS 15.0, *)
extension OSLogStore {

    public func getEntries(with options:
OSLogEnumerator.Options = [], at
position: OSLogPosition? = nil, matching
predicate: NSPredicate? = nil) throws ->
AnySequence<OSLogEntry>
}

@available(macOS 10.15, iOS 15.0, watchOS
8.0, tvOS 15.0, *)
extension OSLogStore {

    public func getEntries(with options:
OSLogEnumerator.Options = [], at
position: OSLogPosition? = nil, matching
predicate: NSPredicate? = nil) throws ->
AnySequence<OSLogEntry>
}

@available(macOS 10.15, iOS 15.0, watchOS
8.0, tvOS 15.0, *)
extension OSLogMessageComponent {

    public enum Argument {

        case undefined

        case data(Data)

        case double(Double)

        case signed(Int64)

        case string(String)

        case unsigned(UInt64)
    }

    public var argument:
OSLogMessageComponent.Argument { get }
}
```