

OpenTelemetry Kotlin SIG Meeting Notes

Meeting Time: Mondays 09:00 - 09:45 Pacific Time

Github Repo: <https://github.com/open-telemetry/opentelemetry-kotlin>

Slack Channel: [#otel-kotlin](#)

Meeting Cadence: Weekly

<TEMPLATE>

Attendees:

-

Agenda:

-

Next Meeting: **May 11, 2026**

Attendees:

- Jamie (Embrace)
- Hanson Ho (Embrace)
- Jason Plumb (Splunk)

Agenda:

- Attributes/Baggage API
 - Carlos confirmed that (4+5) is ok current behavior
 - opentelemetry-java implementation for AnyValue:
<https://github.com/open-telemetry/opentelemetry-java/blob/main/api/all/src/main/java/io/opentelemetry/api/common/AttributeKey.java#L87>
 - Action: Add task to create AnyValue function signature
 - <https://github.com/open-telemetry/opentelemetry-kotlin/issues/499>
 - 6/7/8 ok after discussion (no objections yet)
 - Baggage/Propagators implementation in place, Carlos can review when time
- Context propagation
 - Global vs thread-local implementations exist, global is default right now
 - Java has similar problems with lightweight virtual threads
 - Related java issue
<https://github.com/open-telemetry/opentelemetry-java/issues/8002>
 - Action: Hanson write up Context storage tradeoffs for writing instrumentation + storage (code examples helpful)
- (jason) Any updates about the kotlin folks contrib?

- Marcin
- Vsevolod
- Action: Jason reply to thread

Next Meeting: May 4, 2026

Attendees:

- Jason (Splunk)
- Hanson Ho (Embrace)

Agenda:

- (Hanson) Using require()
 - https://github.com/open-telemetry/opentelemetry-kotlin/pull/462#discussion_r3169732449 comment
 - We should consider just defaulting/falling back to the invalid context/state
- [carlos] Attributes discussion:
 - <https://github.com/open-telemetry/opentelemetry-kotlin/issues/401>
 - No SIG today, upon calling SetAttributes() (plural), will clear all existing attributes.
 - Wondering whether the Spec may need a clarification to make it super clear the expected behavior.
 - So we do have consensus now about the implementation –
 - Api has optional setAttributes() (plural) that allows for setting many
 - Does not have to replace/clear other attributes
 - We are consistent with java's implementation already/now.
- AnyValue PR is looking good – but can we add the nullobject case? NullAnyValue or whatever.
 - <https://github.com/open-telemetry/opentelemetry-kotlin/pull/453/changes>

Meeting: Apr 27, 2026

Attendees:

- Jamie (Embrace)
- Hanson Ho (Embrace)
- Jason (Splunk)
- Viorel (Kobudei)

Agenda:

- Jamie: out next week
 - (jason can run the meeting)
- Viorel: environment variable configuration
 - <https://github.com/open-telemetry/opentelemetry-kotlin/issues/47>
 - Action: as long as it's digestible for PRs, order doesn't matter
 - Action: this ships as separate artifact in opentelemetry-java

- Consider splitting into sections based on the sections in the spec, roughly: <https://opentelemetry.io/docs/specs/otel/configuration/sdk-environment-variables/>
- Viorel: currently envvar only implemented on the JVM
 - Action: Create dedicated issue for other platforms (e.g. native)
 - Action: Create milestone for grouping issues
- Jamie: Attributes API
 - <https://github.com/open-telemetry/opentelemetry-kotlin/issues/401>
 - Type encoded in name + param
 - What do other Kotlin APIs do for similar problems?
 - Action: Add support for AnyValue
 - clear/update all attributes a potential approach?
 - Carlos: all SIGs replace attributes similar to Java's approach
 - Hanson: two span processors could potentially overwrite entire attributes, seems too powerful
 - Carlos: investigate what other SIGs are doing
 - Consensus: implement AnyValue, wait for feedback from Carlos, implement other non-contentious implementation details

Meeting: **Apr 20, 2026**

Attendees:

- Hanson Ho (Embrace)
- Jason (Splunk)
- Jamie (Embrace)

Agenda:

- Jamie: Release branch protection changes need review (taking same approach as opentelemetry-android)
 - Action: Jamie release after resolved
- Jamie: Bug reported in protobuf encoding: <https://github.com/open-telemetry/opentelemetry-kotlin/issues/409>
 - Double/Float, Long/Int
 - We probably want to cast at point of input from our API
 - Ask how attributes are being set for this use-case
 - Java <https://github.com/open-telemetry/opentelemetry-java/pull/7973>
- Hanson: Semantic conventions release
 - Will be released as the new SDK is released and not independently
- Jamie: Talk about API stabilisation again? Some more milestones are created: <https://github.com/open-telemetry/opentelemetry-kotlin/milestones>
 - Attributes probably priority, also Context/Resource next candidates?
 - Create issue for discussion, also potentially check against spec-compliance-matrix
- Carlos: Java SIG maintainers recommended NOT offering a global object by default, but offer it later on if/as needed (FYI it is a recommended but optional operation in the Spec)

- However, if/when this gets added later, the lack of no-op objects may be problematic.
- Already mentioned reasons:
 - Uniformity with other SIGs (e.g. Python).
 - For native instrumentation (e.g. Okhttp instrumenting their operations out-of-the-box), they need a no-op by default (and it can get cumbersome to do the differentiation between the no-op and an actual SDK when the user configures one).

•

Meeting: Apr 13, 2026

Attendees:

- Jamie (Embrace)
- Hanson Ho (Embrace)

Agenda:

- Jamie: release workflow failing due to new branch permissions
 - Release branch protection added means otelbot can't push branches
 - Jamie: Check what opentelemetry-android is doing?
- Jamie: Continue Logging API stabilisation discussion
 - Logging API depends on things that require stabilisation first:
 - Context
 - Attributes
 - Baggage
 - OpenTelemetry object itself?
 - No-op implementation
 - Propagation
 - Logging + Tracing API need to be stable together at same time
- Hanson: No-op implementation followup
 - api + noop in separate packages
 - Docs of current suggested approach:
<https://github.com/open-telemetry/opentelemetry.io/pull/9345/changes#diff-1d03868115c58fa764691d96790757249e993b3207873e56aa842c9b1ccb5e0fR135>
 - General approach: global object that gets no-op or actual implementation
 - Java discourages due to init/order problems

Meeting: Apr 6, 2026

Attendees:

- Jason (Splunk)
- Hanson Ho (Embrace)

Agenda:

- [jason] - security issues remaining
 - Much better shape now – just two remaining
 - New repo “maintained” detected (will fix itself)
 - Jason will make the release branches protected
- [carlos] Logging API stabilization
 - Anything that the API depends on HAS to be stabilized as well, e.g. AttributesMutator, PLUS anything global that is exposed, e.g. OpenTelemetry object.
 - Essentially, everything in the API package, e.g. Context.
 - Noop implementation has to exist in the API.
 - <https://github.com/open-telemetry/opentelemetry-specification/blob/62472705e13651a5d75adeb16367ccd30bcd8834/specification/logs/noop.md>
 - We have them, but they are in a separate package. Is this ok?
 - Does this pass muster at the spec level?
 - The idea here is that the app decides which implementation to use – noop is available or use the jvm sdk or use the kotlin sdk.
 - Instrumentations should never pick which implementation to use
 - AI: Carlos to get a second opinion from TC
 - AI: Hanson to summarize to the kotlin group
 - Context.current() and what the default is might be a larger discussion
 - Will revisit when Jamie is back
 - Double check that Android maintainers are happy with the way AttributesMutator is defined?
 - Any is missing from the api, but we can defer that to later since it is new
 - Should we combine some of the typed methods?
 - Will weaver provide type info for attributes etc one day?
 - Severity UNKNOWN vs UNDEFINED (Java)
 - Does this really matter?
 - Does the ExperimentalApi have to go away before release?
 - Java has an incubator package for API/SDK that has experimental features available for testing.
 - In android, we like this annotation because it allows us to flag something as non-stable without having to jump through the packaging issues that java has
 - We have to stabilize the entire API – we’re not talking about just doing the logging api
 - All of this:
 - <https://github.com/open-telemetry/opentelemetry-kotlin/blob/main/api/api/jvm/api.api>
 - OPTION: move stuff out, then move it back in piecemeal
 - More of a hassle really

Monday March 30 2026

Attendees:

- Jamie (Embrace)
- Hanson Ho (Embrace)
- Jason (Splunk)
- Francisco Prieto (Canary Technologies)

Agenda:

- Jamie: When should we close milestones and what should criteria be? E.g. Logging API: <https://github.com/open-telemetry/opentelemetry-kotlin/milestone/1>
 - Close when all issues complete, *but* add a placeholder issue that marks an API surface as stable
 - We have the power to declare APIs stable after our own due diligence
 - Consider another issue that checks against spec-compliance-matrix before marking stable
 - Ask TC for review after above done
 - Leave around a month or so for feedback to come in after doing our internal work
- Jamie: Continuing on with API spec compliance + Tracing API milestones
- Jamie: The last release was March 11th. Should we release again this week?
 - Ship next week
- Hanson: [Blog post!](#)
 - As the project matures/stabilizes there are other otel channels for continuing promotion/engagement:
 - Slides from EU Maintainer summit: https://docs.google.com/presentation/d/1h_bQL3Tzoom7vMQnVV1_TXoJ7p4OEyrQQXUubaExIsY/edit?slide=id.ged3401ed36_1_0#slide=id.ged3401ed36_1_0
 - Engage the end-user sig ([repo](#), [doc](#))
 - Possible [youtube video](#) (humans of otel or...)
 - [OTel me...](#)
 - [OTel in practice...](#)
 - Ask them about other ideas for promotion/engagement
 - Henrik (is it observable): <https://isitobservable.io/> could do a video/interview
 - Consider adding a kotlin microservice to the [otel-demo](#)
 - I wonder if anybody from JetBrains might be interested in helping contribute...
 - ...or at least to help spread awareness (it's validation for KMP that this project exists perhaps)

CANCELLED: Monday March 23 2026

****As folks are on holiday and at Kubecon****

Monday March 16 2026

Attendees:

- Jamie (Embrace)
- Jason (Splunk)
- Hanson Ho (Embrace)
- Francisco Prieto (Canary Technologies)

Agenda:

- Hanson: Issue templates
 - Should create a couple of issues as a template
 - Consensus: defer until it becomes painful
- Hanson: Policy for AI disclosure in PRs
 - opentelemetry-android has [AGENTS.md](#) for reference
 - Checkbox to disclose AI tools as part of PR (see Java for example)
 - Action: Hanson take a look
 - May potentially need to remove a year or so from now?
 - Related: <https://github.com/open-telemetry/community/pull/3166>
 - Related java instrumentation PR template example: <https://github.com/open-telemetry/opentelemetry-java-instrumentation/pull/15563>
- Jamie: Please review getting started docs: <https://github.com/open-telemetry/opentelemetry.io/pull/9345>
- Jamie: Still need to get a link on the community calendar: <https://github.com/open-telemetry/community/pull/3262>
 - Jamie: recreate PR and fix build breakages
- Hanson: [Readable Span interface in API](#)
 - Proposed extension function to get readable span (currently non-public)
 - Internal/unstable symbol that isn't part of the library consumer API would be acceptable
 - Needs an explicit opt-in annotation (separate from ExperimentalApi?) or separate module
 - Action: Hanson alter issue to be like spec, add explicit opt-in API
- Jason: security issues on the repo
 - Action: Jamie initial triage and bump
- Fran: data instrumentation, where should it go?
 - Do we want to maintain donated instrumentation at this point?
 - Good to accept for now under an instrumentation module, mark as alpha etc & provide with warnings

- Action: Jamie create subdirectory named instrumentation
- Action: Jamie initial review of PR
- RecordException function deprecated
 - Action: Carlos check with spec about whether this can be deleted

Monday March 09 2026

Attendees:

- Jamie - Embrace
- Hanson Ho (Embrace)
- Jason (Splunk)

Agenda:

- Jamie: Release this week? Last was Feb 20th
 - Keep up-to-date with opentelemetry-java
 - Yes, let's ship this week
 - Jamie release mid-week
- Jamie: Getting started docs available for opentelemetry.io, please take a look <https://github.com/open-telemetry/opentelemetry.io/pull/9345>
 - Agreement on opentelemetry.io being source of truth for our docs
 - How to check whether docs code samples are correct?
 - Defer for later? Create an issue for automation perhaps
- Jamie: Was going to create a bunch of issues that map to the spec-compliance matrix and label them appropriately (e.g. help-wanted). Any objections?
 - Grouping with milestones and labels is critical
 - Start with one API surface then get feedback on how issues look
 - Jamie: add issues to GH
- Support `_all_` the targets? <https://github.com/open-telemetry/opentelemetry-kotlin/issues/229>
 - Would accept PR
 - Higher priorities for SIG right now
 - Make it work for one platform first
 - Jamie: respond
- Hanson: Discuss read/write API again? <https://github.com/open-telemetry/opentelemetry-kotlin/issues/259>
 - Hanson: summarize discussion on GH issue
 - Extension function
- Hanson: Issue templates
- Hanson: agent aided PR documentation

Monday March 02 2026

Attendees:

- Jamie Lynch (Embrace)
- Jason (Splunk)
- Hanson Ho (Embrace)
-

Agenda:

- Jamie: Calendar invite setup, Kotlin SIG PR still in-flight:
<https://github.com/open-telemetry/community/pull/3262>
 - Double-check if anything else needs changing on community repo for Kotlin
- Jamie: Blog post on opentelemetry.io in progress, reviews welcome:
<https://github.com/open-telemetry/opentelemetry.io/pull/9293>
- Jamie: Continuing to work on API's spec compliance. In the meantime should we consider getting something added to the spec-compliance-matrix in the spec repo?
<https://github.com/open-telemetry/opentelemetry-specification/pull/4907>
 - Consensus: ok to merge after review
 - Smaller PRs for more complex areas such as context propagation
- Jamie: Release cadence?
 - Sort-of-convention on Java: monthly
 - Approvers help with release then maintainer hits merge? Spreads knowledge
 - Investigate publishing snapshots for bleeding edge integrations
 - Consensus: monthly (document in release notes, with caveat that this is just convention)
- Jason: iOS build instructions
 - XCode setup required (should add as pre-requisite for setup) (Hanson)

Monday February 23 2026

Attendees:

- Jamie Lynch (Embrace)
- Jason (Splunk)
- Francisco Prieto (Canary Technologies)
- Alolita Sharma (OTEL GC)
- Carlos (OTEL TC)
- Hanson Ho (Embrace)

Agenda:

- Kotlin SIG PR: <https://github.com/open-telemetry/community/pull/3262>

- Update google doc link - Alolita Sharma will add
- Where does the zoom meeting go? Just in the readme? - Yes to avoid Zoom bombing incidents
- Project board
 - Create from milestones on Github (see other OpenTelemetry repos for example)
 - Example: <https://github.com/orgs/open-telemetry/projects/186> - Go SIG Roadmap updated by maintainers and could be reviewed for the first 5 minutes in each SIG meeting
- Meeting cadence: weekly or bi-weekly?
 - Weekly at 45 min consensus
 - Revisit in around a month
- What should explicit minCompileSdk be?
<https://github.com/open-telemetry/opentelemetry-kotlin/pull/214>
 - Consensus: 34 ok
 - AarMetadata contains minimum AGP version, investigate further?
<https://developer.android.com/reference/tools/gradle-api/8.13/com/android/build/api/variant/AarMetadata>
- Discuss read-only API approach for capturing telemetry, versus read-write API. The Attributes API is a good conversation starter for this:
<https://github.com/open-telemetry/opentelemetry-kotlin/pull/160>
 - Embrace present use-cases for further discussion
 - Renames can be split off into another PR, keep same relationships for now
- Spec SIG meeting notes should be read by maintainers