Intro to Metrics

What are metrics?

Metrics are pre-aggregated timeseries data representing observations of numeric quantities and associated attributes.

Observations

- (sync) when an event of interest happens we can observe a value related to that event, e.g.,
  - how long an HTTP request took
  - how many bytes were in the request body
  - a query was made to a datastore
- (async) sometimes we want to periodically observe a value without recording every change to that value
  - how many items are in a queue waiting for processing
  - how much memory is allocated to a process
  - what is the temperature in a room

Numeric Quantities

Metrics deal with numbers, both integer and floating point.

It doesn't make sense to observe `red`, but it can make sense to observe `balloons{color=red} 99`

Associated Attributes

Attributes can be included in the observation to provide context for understanding the observation, e.g.

- what URL was requested
- what was the HTTP status code of the response
- which datastore was queried
- what color was my balloon

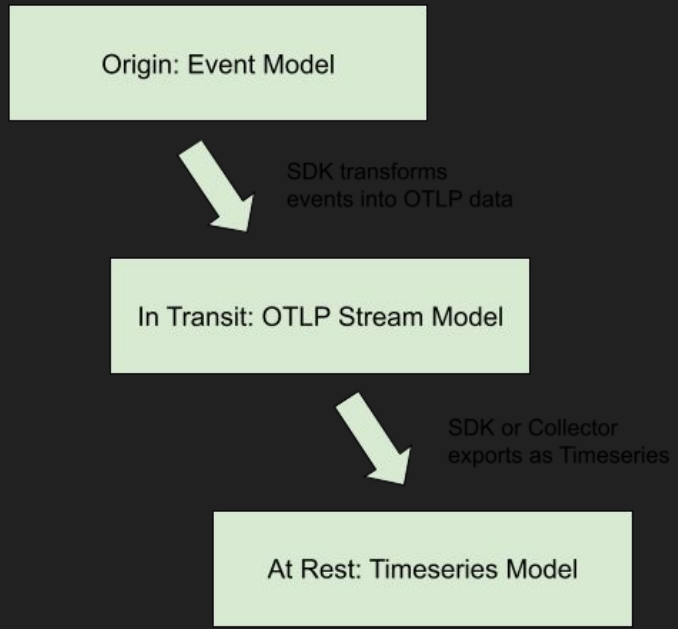
Pre-Aggregated Timeseries

Different aggregations can be used to preserve different amounts of information from the included observations

- sum
- min/max/last value
- fixed-bucket histogram
- exponential histogram

Data Models

- Event Model
- OTLP
- Timeseries Model



Metrics API and SDK

- MeterProvider is the entry point of the API. It provides access to Meters
  - API MeterProvider only provides ability to create a Meter
  - SDK MeterProvider accepts configuration for Resource association, metric processing, and exporters
- Meter is responsible for creating Instruments
- Instrument is responsible for reporting Measurements

Instruments

- Synchronous
  - Allows for direct observation of a value related to work occurring in a distinct context
- Asynchronous
  - Allows for observation of values that may not relate directly to any individual context within which work is performed, but instead relate to the instrumented scope of work as a whole
  - Adds a callback parameter to the parameters used to create synchronous instruments
  - This callback is invoked when metrics are aggregated and can record observations to instruments registered with the callback

Counter

- Tracks monotonically increasing values
  - That's a fancy way of saying they never go down
  - They might stay the same or go up, but never down
- Defaults to `Sum` aggregation
  - Observing `[1,4,2,3]` results in the aggregated value `10`
- Has an asynchronous variant

UpDownCounter

- Same as `Counter` without requirement for monotonicity
- Defaults to `Sum` aggregation
  - Observing `[1,4,-2,3]` results in the aggregated value `6`
- Has an asynchronous variant

Gauge

- Tracks values that may change independently of identifiable events within the observed system
- Defaults to `LastValue` aggregation
  - Observing `[1,4,2,3]` results in the aggregated value `3`
- Asynchronous only
  - An experimental synchronous `Gauge` instrument is being worked on

Histogram

- Records observations of values in a statistically meaningful distribution
- Defaults to `ExplicitBucketHistogram` aggregation
  - Optionally can use `ExponentialHistogram` aggregation for high-resolution, self-adjusting distributions
  - May include minimum, maximum, arithmetic sum, and count of values observed
  - Observing `[1,4,2,3]` results in the aggregated value `{min: 1, max: 4, sum: 10, count: 4}` and a set of bucket counts such as `{1: 1, 3: 3, 5: 4}`
- Synchronous only