Measure proposal: mathematical operators!

Ben Allen

21 November 2024

Measure revision – mathematical operators

The new version of the explainer for Measure – which I'm now calling Measurement – is updated to include the following: mathematical operations:

- Add or subtract two Measurements of same dimensions
- Multiply or divide Measurements by a scalar
- Divide a Measurement by another Measurement to get an *x*-per-*y* compound measurement
- Multiply a Measurement by a dimensionally compatible Measurement (for example, to get an area measurement from two length measurements)

Measure revision – options bag

Measurements keep track of many pieces of data:

- The `value`
- The `unit` in which the value is measured
- The `precision` of the measurement
- The type of thing being measured (`usage`)
- The `exponent` to which the base unit is raised.

In this revision, I'm managing it by putting everything but value in an options bag

Measure revision – mixed units

Mixed unit input and output are handled as follows:

1. Mixed units are input in terms of the *largest* unit. A Measurement of unit `"foot-and-inch"` that's 5 feet, 6 inches would have `value: 5.5`
2. Measurements have a new `toComponents` Measurement method that returns an array containing each component of the measurement (only really useful in case of mixed units)

Measure revision – Precision is a pain!

Okay, how *should* we handle precision? In our last conversations we decided that precision representing fractional digits is the least-bad option. But is it though?

Note: the Java Units of Measurement API appears to deal with this by not handling precision at all.

Measure revision – Lingering questions

- Is the options bag a good idea, a bad idea, a terrible idea?
- How should we internally store Measurements with units like cubic centimeters? Is the exponent option a good idea?
- What am I missing? (I'm sure I'm missing a ton)