Erin Schnabel
@ebullientworks

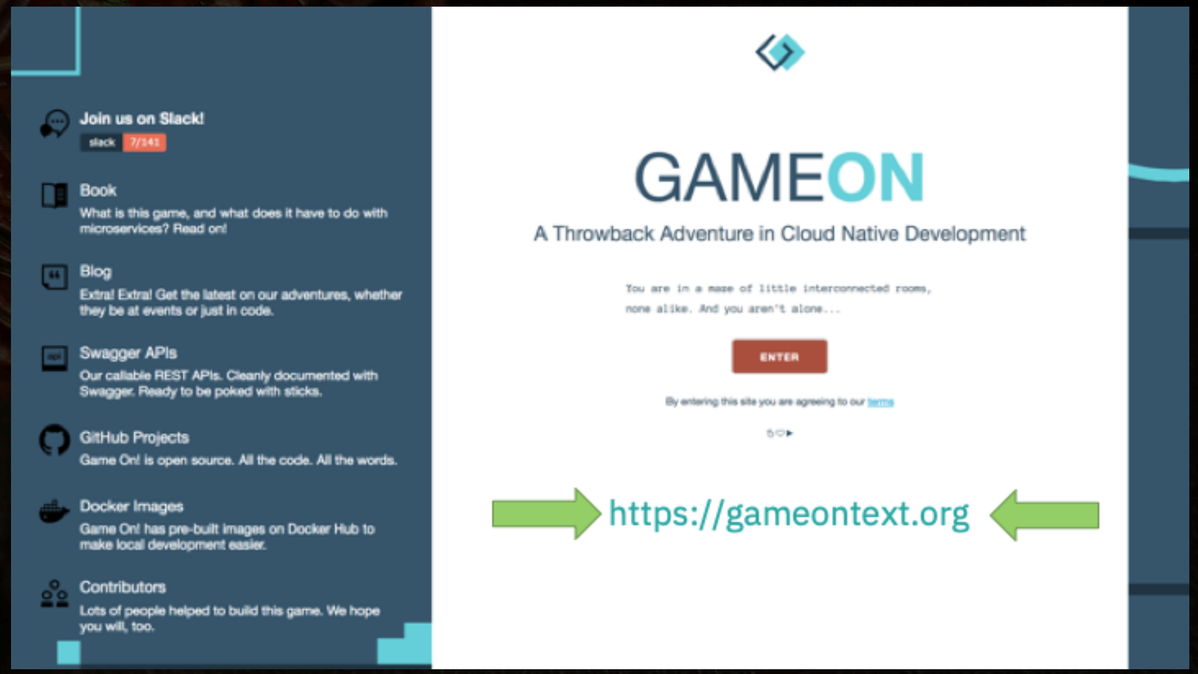
The Mechanics of Metrics

Aggregation across dimensions

A little bit about me

- Developer of things at Red Hat
- IBMer for 21 years
- Java Champion
- Dungeon Master for 12 year olds
- Most importantly:

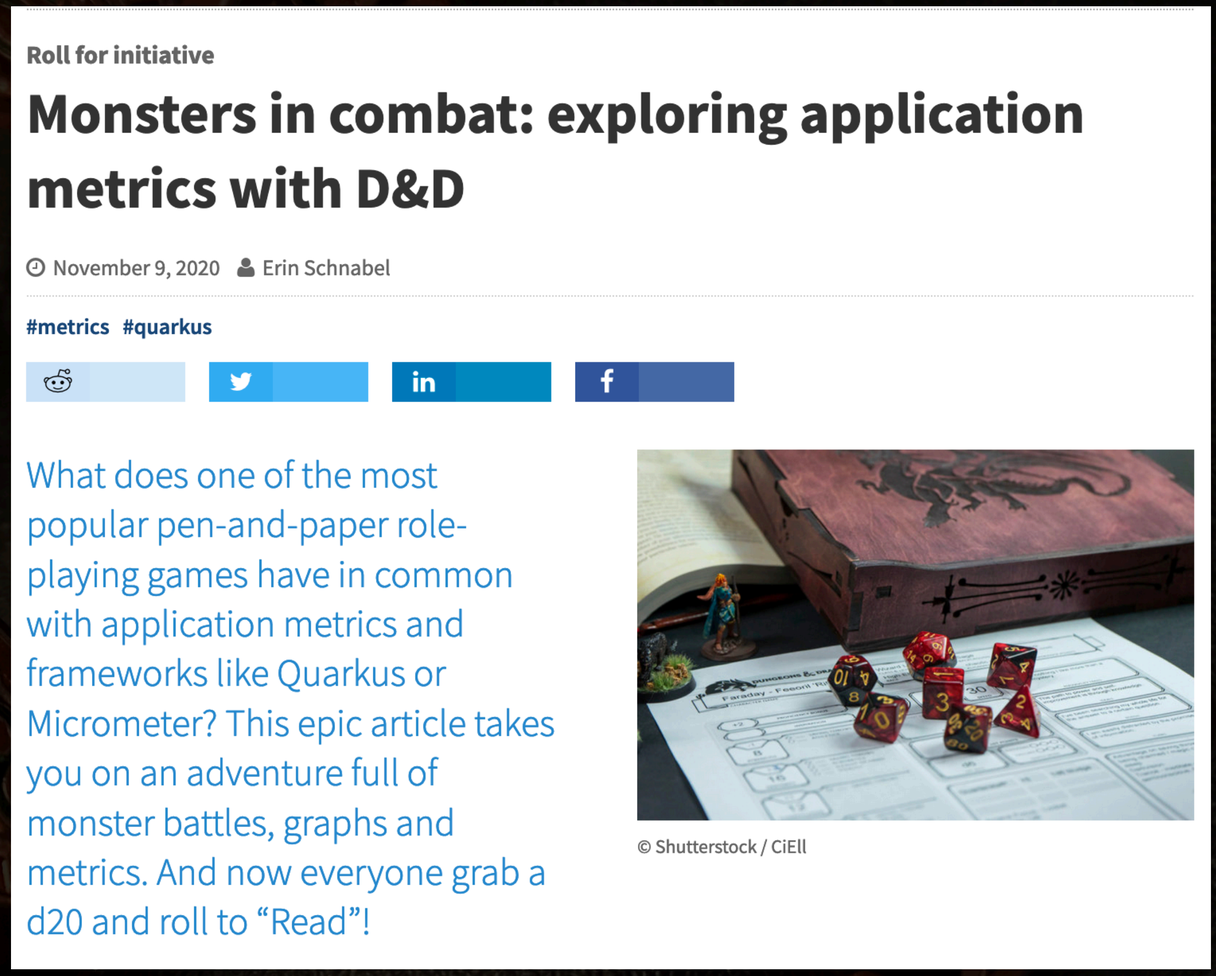
I build ridiculous things.



Monster Combat

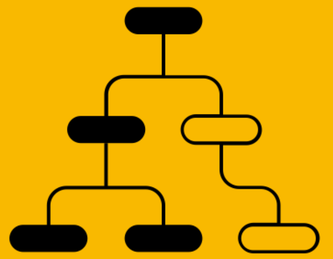
https://github.com/ebullient/monster-combat

https://twitter.com/ebullientworks/status/1328835279659085828?s=20



https://jaxenter.com/metrics-dnd-173311.html

Observability: Which for what?

Service is ready Service is not a zombie	How many times was method x called?	What happened when method x was called	Method x was called
Health Checks	**Metrics**	**Log Entries**	**Distributed Trace**
			
Workload routing System health	Statistics & trends Analytics	Service-centric problem determination	Context + relationships for end–to-end analysis

Time-series data for metrics

How does data change over time?

- Time is a primary axis (x)
- Data gathered or observed at a regular intervals

- String key with ONE* numeric value
  - Value is observed at collection time
  - Appended as new entry in the series

* “Univariate”. Multivariate data is a very different problem space.

Counter

Micrometer:

```
Dice.setMonitor((k, v) -> {
    registry.counter("dice.rolls", "die", k, "face", label(v))
            .increment();
});
```

Dimensions and cardinality

Labels add dimensions for analysis

```
dice_rolls_total {
    die="d10"
    face="01"
    instance="..."
    job="..."
} 77062
```

Filtered aggregation

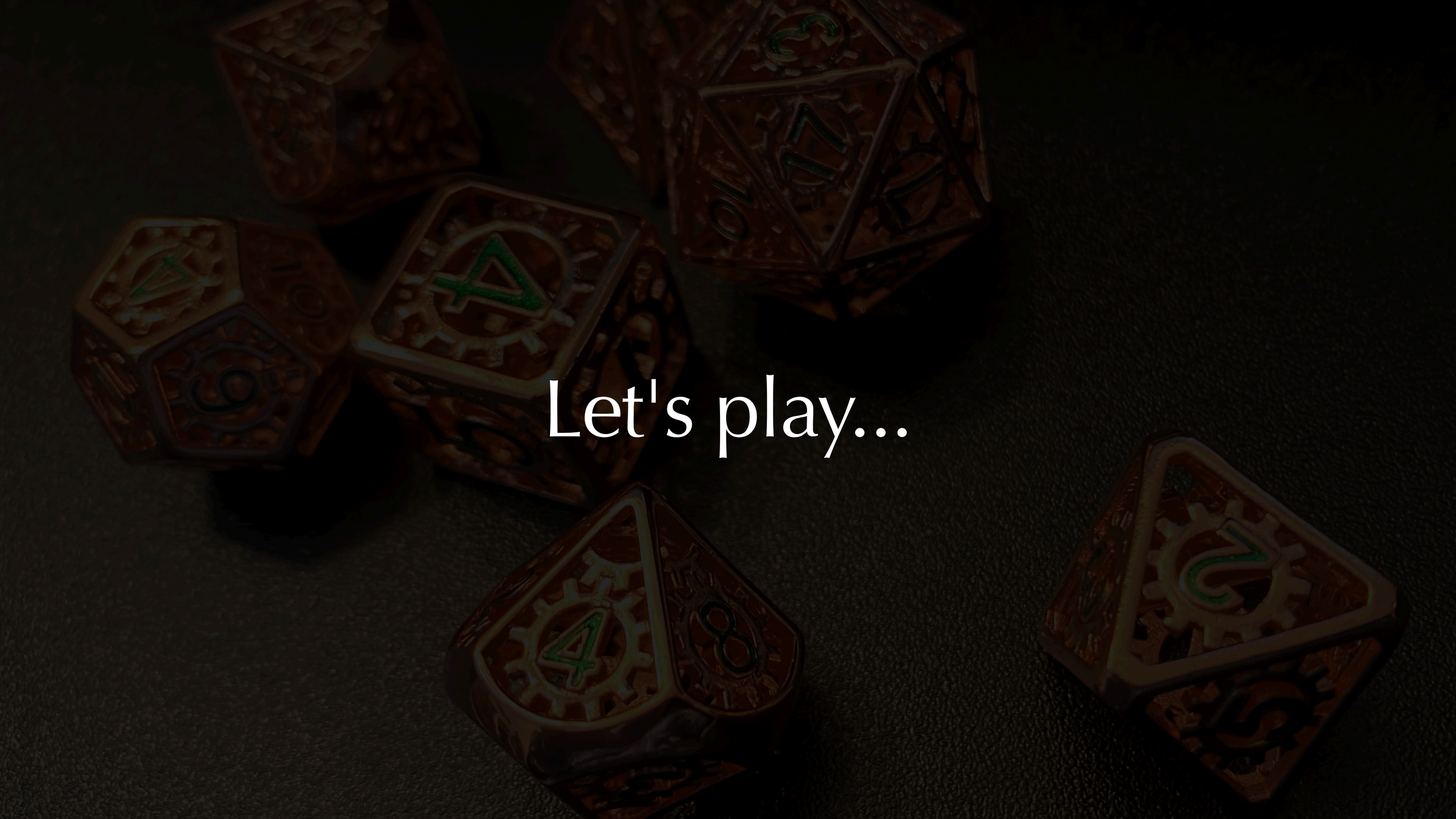
of a **single** value

Cardinality

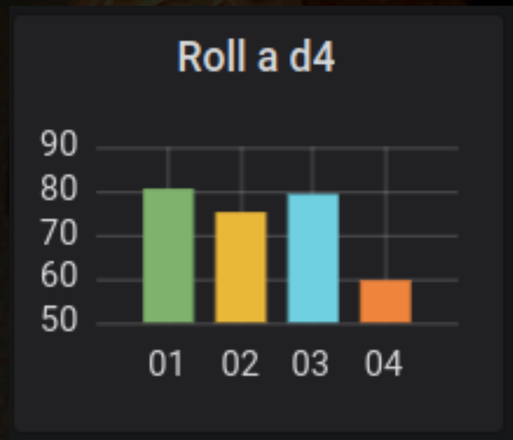
Each unique combination of labels is a series:

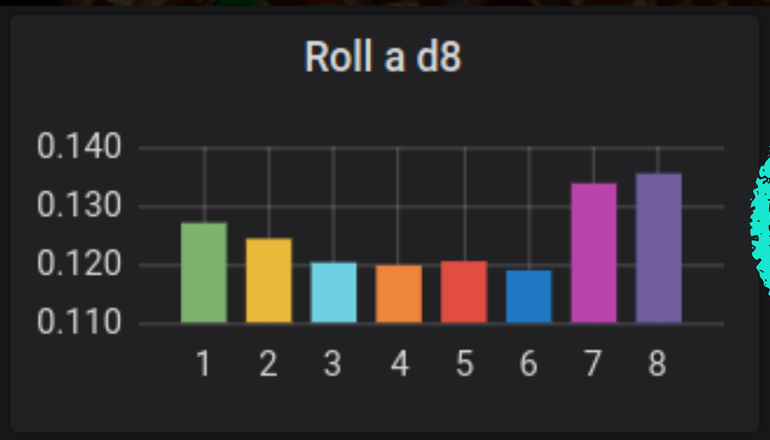
die (6 values),
face (20 values),
instance (3 values),
job (3 values)

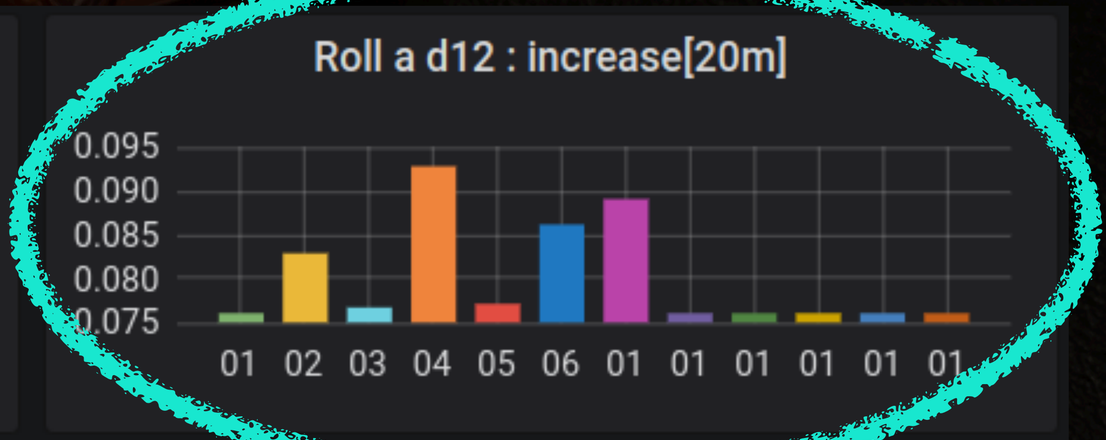
1080 unique time series (max)
(6 * 20 * 3 * 3)

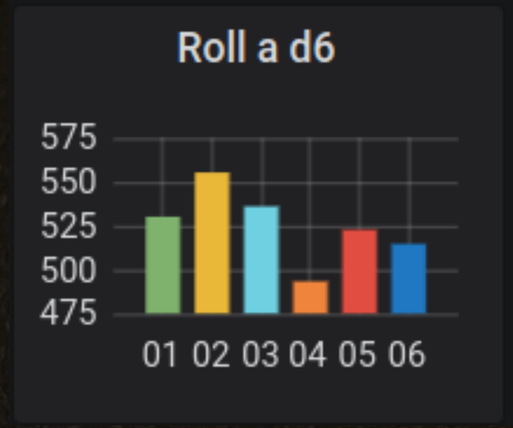

Let's play...

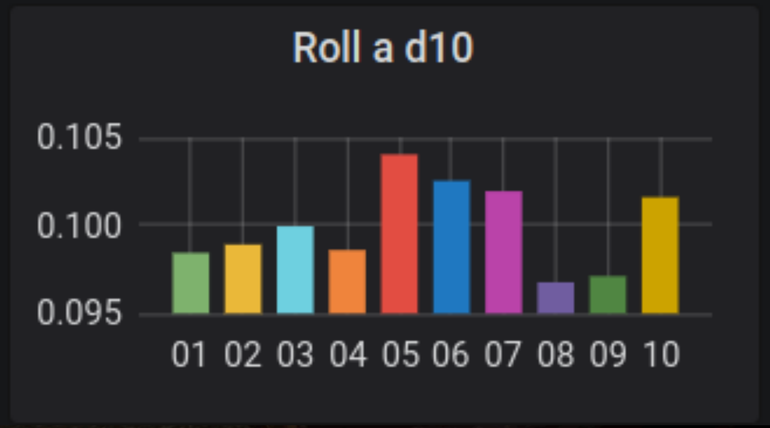
You may find more than just numeric trends...

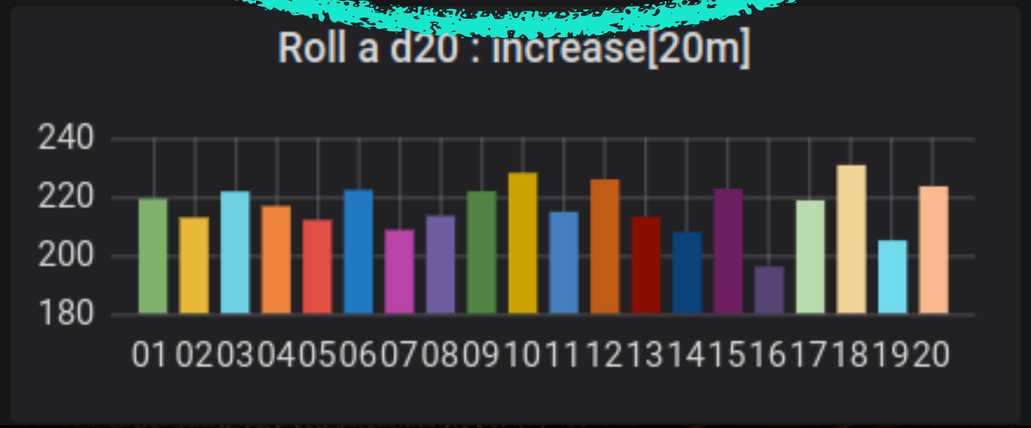












Attacks : Hits and Misses

- AC == Armor class

  Attacker rolls a d20...

  20 – critical hit (HIT)
  1 – critical miss (MISS)

- DC == Difficulty class

  Defender rolls a d20...

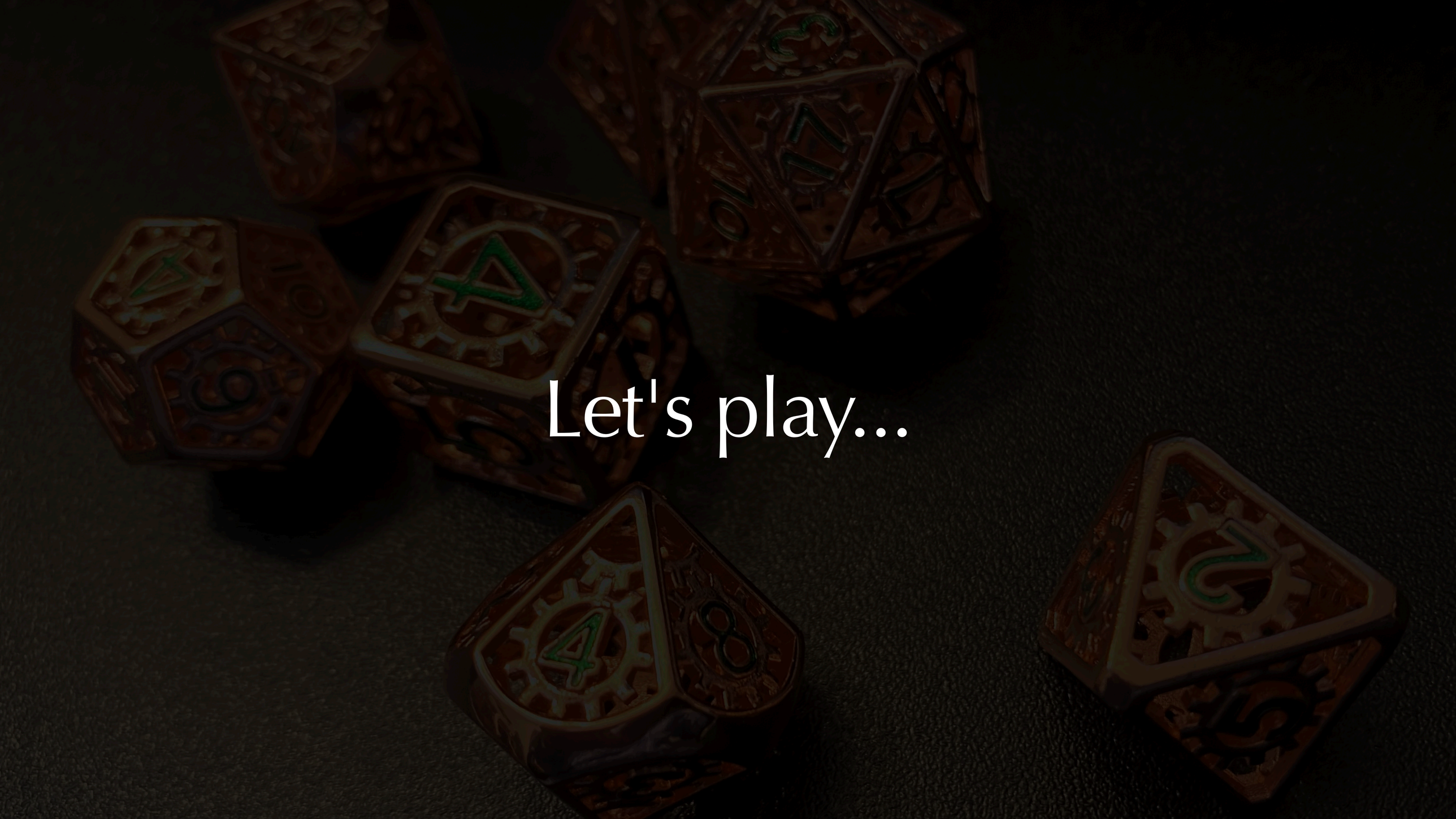
```
: oneRound:
    Troll(LARGE GIANT){AC:15,HP:84(8d10+40),STR:18(+4),DEX:13(+1),CON:20(+5)
    Pit Fiend(LARGE FIEND){AC:19,HP:300(24d10+168),STR:26(+8),DEX:14(+2),CON

: attack: miss: Troll(36) -> Pit Fiend(100)
: attack: miss: Troll(36) -> Pit Fiend(100)
: attack: hit> Troll(36) -> Pit Fiend(97) for 9 damage using Claws[7hit,11(2
: attack: hit> Pit Fiend(97) -> Troll(10) for 22 damage using Bite[14hit,22(
: attack: MISS: Pit Fiend(97) -> Troll(10)
: attack: HIT> Pit Fiend(97) -> Troll(0) for 34 damage using Mace[14hit,15(2

: oneRound: survivors
    Pit Fiend(LARGE FIEND){AC:19,HP:300(24d10+168),STR:26(+8),DEX:14(+2),CON
```

Micrometer: Timer & Distribution Summary

- Records value

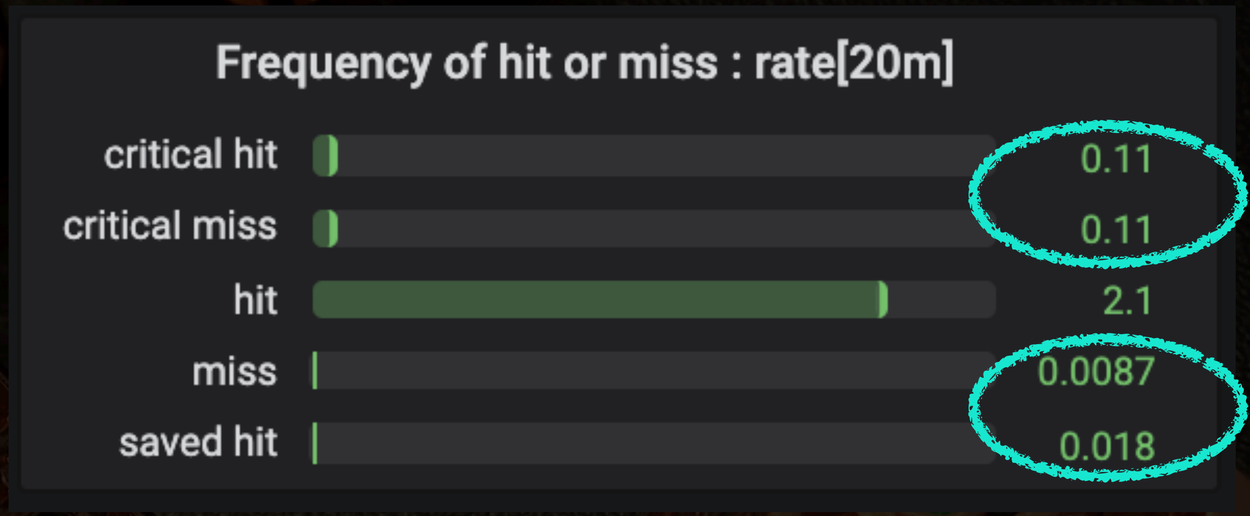
```
registry.summary("round.attacks",
        "hitOrMiss", event.hitOrMiss(),
        "attackType", event.getAttackType(),
        "damageType", event.getType())
        .record((double) event.getDamageAmount());
```

- Default metrics: count, sum, max
- Optional: cumulative histogram buckets, pre-computed quantile


Let's play...

Other discoveries...

	max	avg
critical saved hit	26.00	23.13
critical hit	25.88	25.68
hit	12.92	12.85
saved hit	11.49	11.07
miss	0	0
critical miss	0	0




Thank you!