UX Design and Education for Effective Monitoring Tools
Hi!👋

- I’m Amy!
- Observability engineer!
- Also on the plenary panel tomorrow at 4pm!
- amynguyen.net
  @amyngyn
Agenda

- Motivation
- Sharing what you know
- Designing what your users want
- Recap
Agenda

- Motivation
  - Sharing what you know
  - Designing what your users want
  - Recap
UX and Your Situation
UX and Your Situation

team
UX and Your Situation

documentation

team
UX and Your Situation

- Team
- Documentation
- Tools
UX and Your Situation

- Team
- Documentation
- Tools

probably paying for vendors TBH?
UX and Your Situation

this talk

tools

documentation

team

probably paying for vendors TBH?
UX and Your Situation

things we can all control

documentation

tools

probably paying for vendors TBH?

this talk
Why should we care about user experience?
Why should we care about user experience?
Why should we care about user experience?

- **Prevent misunderstandings**: not everyone is (or should have to be) an expert at interpreting monitoring data
Why should we care about user experience?

- **Prevent misunderstandings:** not everyone is (or should have to be) an expert at interpreting monitoring data

- **Developer velocity:** help people reach conclusions faster, help your company move faster
Why should we care about user experience?

- **Prevent misunderstandings:** not everyone is (or should have to be) an expert at interpreting monitoring data

- **Developer velocity:** help people reach conclusions faster, help your company move faster

- **Data democracy:** you don’t know what questions people want to answer with their own data
Why should we care about user experience?

- **Prevent misunderstandings:** not everyone is (or should have to be) an expert at interpreting timeseries data.

- **Developer velocity:** help people reach conclusions faster, help your company move faster.

- **Data democracy:** you don’t know what questions people want to answer with their own data.
Why should we care about user experience?

Because we can help engineers work *correctly, quickly, and independently.*
“The fastest way to become a 10x engineer is to help 10 other engineers do their jobs better.”

- Wayne Gretzky

- Michael Scott
Agenda

- Motivation
- Sharing what you know
- Designing what your users want
- Recap
Sharing what you know
Sharing what you know

1. **Education vs intuition:** Don’t overload people with too much information.
stats.example.metric.errors > 5
maxSeries(redacted_metric) > 0.97

Aggregate series by average

Check every 30 seconds
Drop edges
Drop none
For empty data, treat as passing
For graphite metrics, replace null with
Test Alert
Sharing what you know

1. **Education vs intuition**: Don’t overload people with too much information.

2. **Best practices**: Use your expertise to determine the most helpful default behavior.
maxSeries(redacted_metric) > 0.7

From 10 minutes ago until now

Aggregate series by average

Check every 30 seconds

For empty data, do nothing

For graphite metrics, replace null with

Test Alert
Sharing what you know

1. **Education vs intuition:** Don’t overload people with too much information.

2. **Best practices:** Use your expertise to determine the most helpful default behavior.

3. **Potential pitfalls:** Make it hard to do the wrong thing.
Sample Alert

maxSeries(redacted_metric) > 0.7

When

From 10 minutes ago until now

Aggregate series by

- average
- final point
- max
- min
- sum
- percentile

For empty data, do nothing

For graphite metrics, replace null with

Test Alert
It looks like you're trying to alert on the most recent data.

Are you sure you want to do that?
Agenda

- Motivation
- Sharing what you know
- Designing what your users want
- Recap
Designing what your users want
Designing what your users want

1. **Performance:** Do whatever it takes to make it fast.
Performance: Cool Ideas!
Performance: Cool Ideas!

- Backend
Performance: Cool Ideas!

- Backend
  - Roll-up data over long time ranges
Performance: Cool Ideas!

- Backend
  - Roll-up data over long time ranges
  - Store latest data in memory (e.g., Facebook’s Gorilla paper and Beringei project)
Performance: Cool Ideas!

- Backend
  - Roll-up data over long time ranges
  - Store latest data in memory (e.g., Facebook’s Gorilla paper and Beringei project)
  - Add a cache layer (e.g., Turn’s Splicer project)
Performance: Cool Ideas!

- Backend
  - Roll-up data over long time ranges
  - Store latest data in memory (e.g., Facebook’s Gorilla paper and Beringei project)
  - Add a cache layer (e.g., Turn’s Splicer project)

yeah sure get back to us in 6 months
Performance: Cool Ideas!

- **Backend**
  - Roll-up data over long time ranges
  - Store latest data in memory (e.g., Facebook’s Gorilla paper and Beringei project)
  - Add a cache layer (e.g., Turn’s Splicer project)

- **Frontend**
Performance: Cool Ideas!

- **Backend**
  - Roll-up data over long time ranges
  - Store latest data in memory (e.g., Facebook’s Gorilla paper and Beringei project)
  - Add a cache layer (e.g., Turn’s Splicer project)

- **Frontend**
  - Don’t reload existing data if the user changes the time window
Performance: Cool Ideas!

- **Backend**
  - Roll-up data over long time ranges
  - Store latest data in memory (e.g., Facebook’s [Gorilla paper](#) and [Beringei project](#))
  - Add a cache layer (e.g., [Turn’s Splicer project](#))

- **Frontend**
  - Don’t reload existing data if the user changes the time window
  - Prevent the user from requesting the data incessantly
Performance: Cool Ideas!

- **Backend**
  - Roll-up data over long time ranges
  - Store latest data in memory (e.g., Facebook’s [Gorilla paper](#) and [Beringei project](#))
  - Add a cache layer (e.g., Turn’s [Splicer project](#))

- **Frontend**
  - Don’t reload existing data if the user changes the time window
  - Prevent the user from requesting the data incessantly
  - Lazy-load graphs on a dashboard
Designing what your users want

1. **Performance:** Do whatever it takes to make it fast.

2. **Exploration:** Make it easy to try things without fear.
don’t mess with this
don’t mess with this
don't mess with this

or this
don't mess with this

or this
don't change it
don’t mess with this

don’t change it

or this

It doesn’t deserve to be treated this way
it’s just trying to help
Accessibility is for everyone
Accessibility is for everyone

- Color contrast
Accessibility is for everyone

- Color contrast
Accessibility is for everyone

- Color contrast
- Customizable color schemes
Accessibility is for everyone

- Color contrast
- Customizable color schemes
- Navigation bar that isn’t entirely controlled by JavaScript
Accessibility is for everyone

- Color contrast
- Customizable color schemes
- Navigation bar that isn’t entirely controlled by JavaScript
- Keyboard navigation
Accessibility is for everyone

- Color contrast
- Customizable color schemes
- Navigation bar that isn’t entirely controlled by JavaScript
- Keyboard navigation
- Obvious focus on what needs attention
Exploration through previews
Exploration through previews: terraform plan
Exploration through previews: alert previews

[chef][errors] Chef is failing more frequently

Chef failed in 104 seconds on i-516e4bfd | #account:prod #env:prod #host:i-516e4bfd
Chef updated 5 resources out of 406 resources total.

- dd-disk_encrypt[encrypted] (dd-disk:encrypt line 29)
- users_manage[sysadmin] (users:sysadmins line 23)
- python_pip[setuptools] (python:pip line 50)
- chef_handler[Chef::Handler::Datadog] (datadog:dd-handler line 32)
- users_manage[dogs] (ddusers:default line 18)

Errno::ENOEXEC: package[openssh-server] (openssh:default line 28) had an error: Errno::ENOEXEC: Cannot allocate memory - fork(2)
Exploration recap

1. Don’t break basic browser features.

2. Design for accessibility.

3. Make it easy to try things without making permanent changes.
Designing what your users want

1. **Performance**: Do whatever it takes to make it fast.

2. **Exploration**: Make it easy to try things without fear.

3. **Simplicity**: Make it easy to figure out what to do.
Am I so out of touch?
sum:10m-avg:stats.example.metric.errors{host=my_host, other_tag=some_value}
manually type in your metric if you know the name somehow

sum:10m-avg:stats.example.metric.errors{host=my_host, other_tag=some_value}
manually type in your metric if you know the name somehow

```
sum:10m-avg:stats.example.metric.errors{host=my_host, other_tag=some_value}
```
manually type in your metric if you know the name somehow

sum:10m-avg:stats.example.metric.errors{host=my_host, other_tag=some_value}

aggregator

hope you remember what tags are available for this metric lol
manually type in your metric if you know the name somehow

secret bonus: you can downsample?!

aggregator

hope you remember what tags are available for this metric lol
manually type in your metric if you know the name somehow

aggregator

secret bonus: you can downsample?!

hope you remember what tags are available for this metric lol
### Metrics

**stat.redacted.metric**

**My Example Metric**

<table>
<thead>
<tr>
<th>Grouping Tags</th>
<th>Non-grouping Tags</th>
</tr>
</thead>
<tbody>
<tr>
<td>go</td>
<td>service</td>
</tr>
<tr>
<td>host</td>
<td>bar</td>
</tr>
<tr>
<td>foo</td>
<td>topic</td>
</tr>
<tr>
<td>joinPartition</td>
<td>baz</td>
</tr>
<tr>
<td>partition</td>
<td>version</td>
</tr>
<tr>
<td>stage</td>
<td>qux</td>
</tr>
</tbody>
</table>

**OpenTSDB** - sum

**Rate**

**Downsample**
<table>
<thead>
<tr>
<th>Metric</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>stat.redacted.metric</td>
<td></td>
</tr>
</tbody>
</table>

**My Example Metric**

<table>
<thead>
<tr>
<th>Tag</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>go</td>
<td></td>
</tr>
<tr>
<td>host</td>
<td>foo</td>
</tr>
<tr>
<td>partition</td>
<td>stage</td>
</tr>
<tr>
<td>service</td>
<td>bar</td>
</tr>
<tr>
<td>topic</td>
<td>baz</td>
</tr>
<tr>
<td>version</td>
<td>qux</td>
</tr>
</tbody>
</table>

**Available Options**
- OpenTSDB
- sum
- Rate
- Downsample

Available options you probably don’t need to touch
manual entry for power users

<table>
<thead>
<tr>
<th>Grouping Tags</th>
<th>host</th>
<th>foo</th>
<th>joinPartition</th>
</tr>
</thead>
<tbody>
<tr>
<td>go</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>partition</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Non-grouping Tags</td>
<td>service</td>
<td>bar</td>
<td>topic</td>
</tr>
</tbody>
</table>

My Example Metric

stat.redacted.metric

OpenTSDB

sum

Rate

Downsample
<table>
<thead>
<tr>
<th>Metrics</th>
<th>Statistics</th>
<th>Raw</th>
<th>Display</th>
<th>YAML</th>
</tr>
</thead>
<tbody>
<tr>
<td>stat.redacted.metric</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**My Example Metric**

<table>
<thead>
<tr>
<th>Grouping Tags</th>
<th></th>
<th></th>
<th></th>
<th>Non-grouping Tags</th>
</tr>
</thead>
<tbody>
<tr>
<td>go</td>
<td>host</td>
<td>foo</td>
<td>joinPartition</td>
<td></td>
</tr>
<tr>
<td>partition</td>
<td>stage</td>
<td>service</td>
<td>bar</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>topic</td>
<td>baz</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>version</td>
<td>qux</td>
<td></td>
</tr>
</tbody>
</table>

relevant information needed to create a query!
Agenda

- Motivation
- Sharing what you know
- Designing what your users want
- Recap
"The fastest way to become a 10x engineer is to help 10 other engineers do their jobs better.

- Wayne Gretzky"

- Michael Scott
UX and Your Situation

this talk

doctoration

tools

probably paying for vendors TBH?

things we can all control

team
Sharing what you know

1. **Education vs intuition:** Don’t overload people with too much information.

2. **Best practices:** Use your expertise to determine the most helpful default behavior.

3. **Potential pitfalls:** Make it hard to do the wrong thing.
Designing what your users want

1. **Performance**: Do whatever it takes to make it fast.

2. **Exploration**: Make it easy to try things without fear.

3. **Simplicity**: Make it easy to figure out what to do.
Thanks! 😊

(also check out the plenary panel! 4pm tomorrow!)