It’s all in the mix!

Producing production-ready apps with Spring Boot

Intro

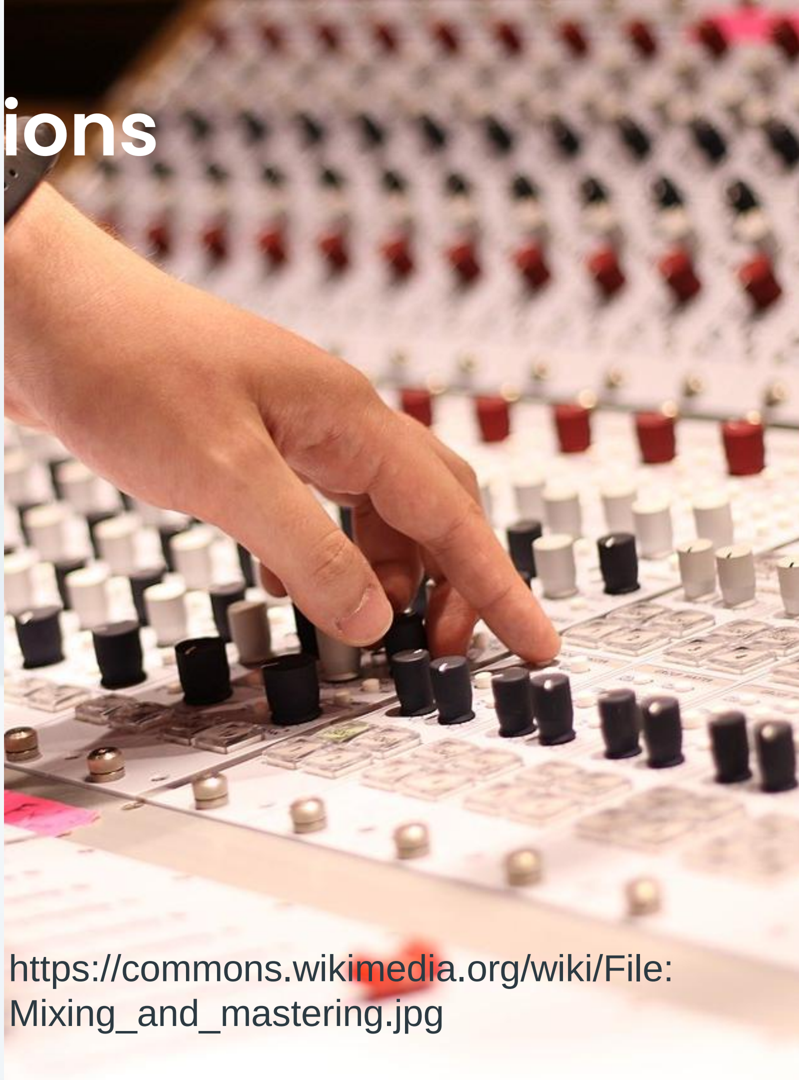
- Many things go into building applications
    - Architecture
    - Business logic
    - Persistence
    - Security
    - APIs

- This talk isn’t about that…

https://upload.wikimedia.org/wikipedia/commons/0/0e/Musician.jpg

Production-Ready Applications

- Need more than that
  - Performance tuning & caching
  - Packaging
  - Observability
- Should integrate with framework features
  - Custom autoconfiguration
  - Error handling in the web tier



https://commons.wikimedia.org/wiki/File:Mixing_and_mastering.jpg

Error Handling in the Web Tier

TRIFORK.

Web Error Handling

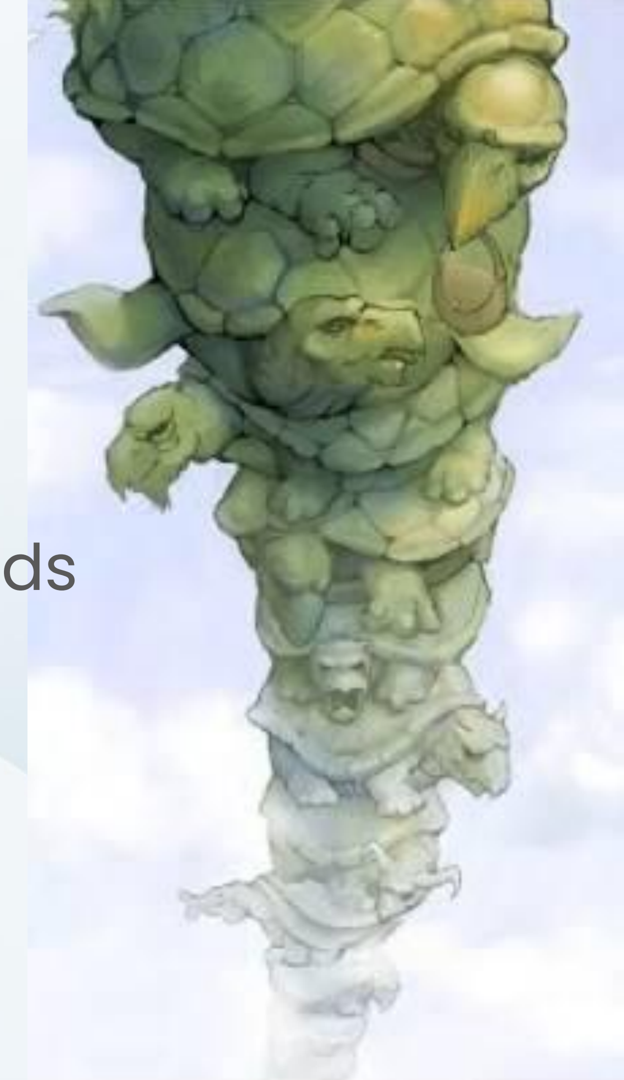
- Response for uncaught and framework exceptions
- Default: white-label error for uncaught

- Common approach: @(Rest)ControllerAdvice with @ExceptionHandlers

- Spring 6.0+ also supports RFC 7807 ProblemDetail

ControllerAdvice & ExceptionHandler

- @ExceptionHandler also called for nested cause
  - any level, since Spring 5.3
- Spring considers all handler methods
- No match?
  Then try cause, repeatedly



Demo

TRIFORK.

Further Notes

- Extend DefaultErrorAttributes to resolve errors
- Extend ResponseEntityExceptionHandler for RFC 7807 support
  - Handles common Spring MVC exceptions
  - Hides validation errors, though…

- For more control, consider Wim Deblauwe’s Error Handling Spring Boot Starter

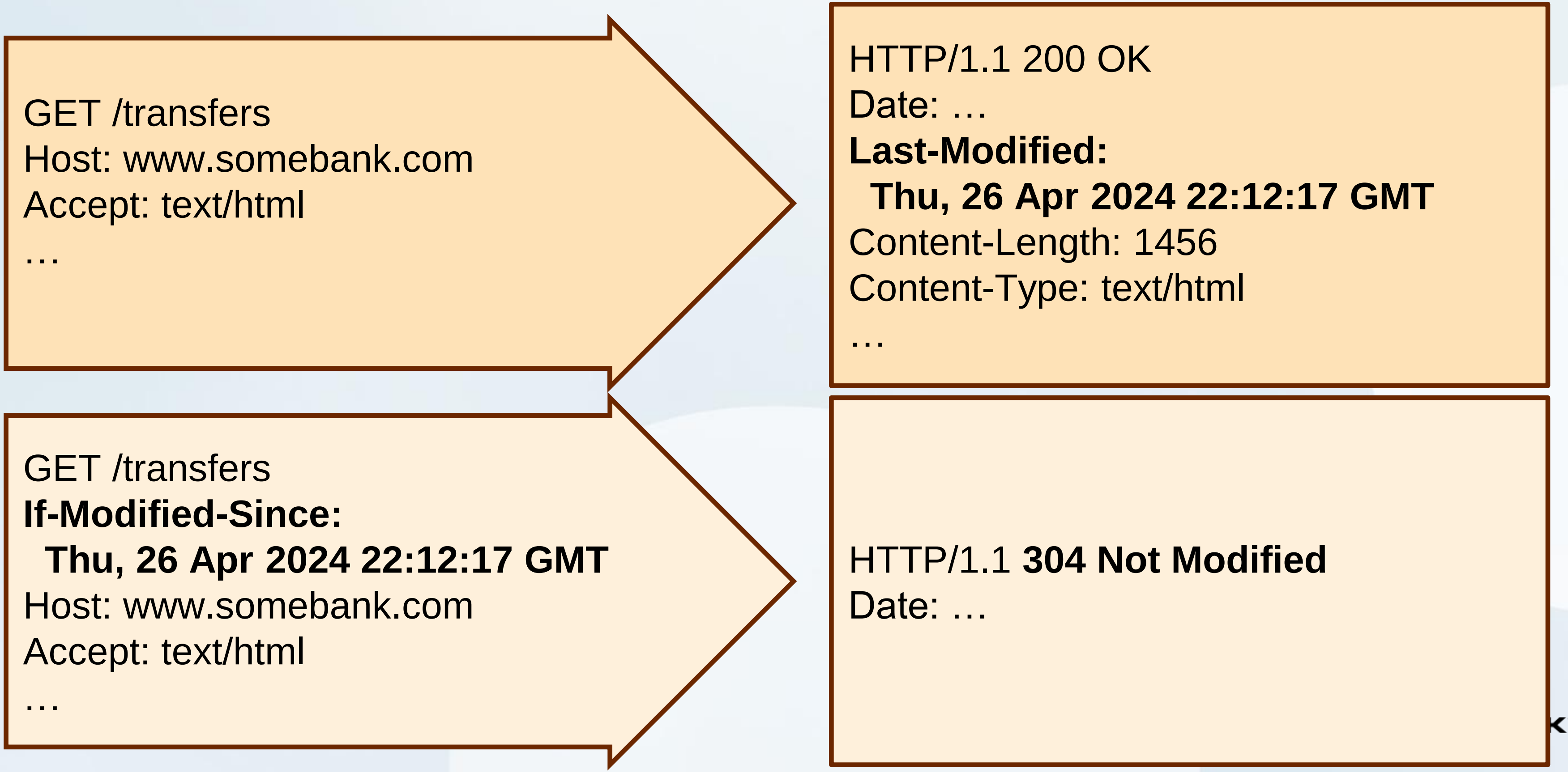
HTTP Caching

HTTP Caching

- Part of HTTP spec since 1.0
- Aimed at browsers originally, also relevant to API clients

- Widely supported
  - Tools, platforms, languages, frameworks, …
- But often overlooked for non-static resources

Last-Modified (since HTTP 1.0)



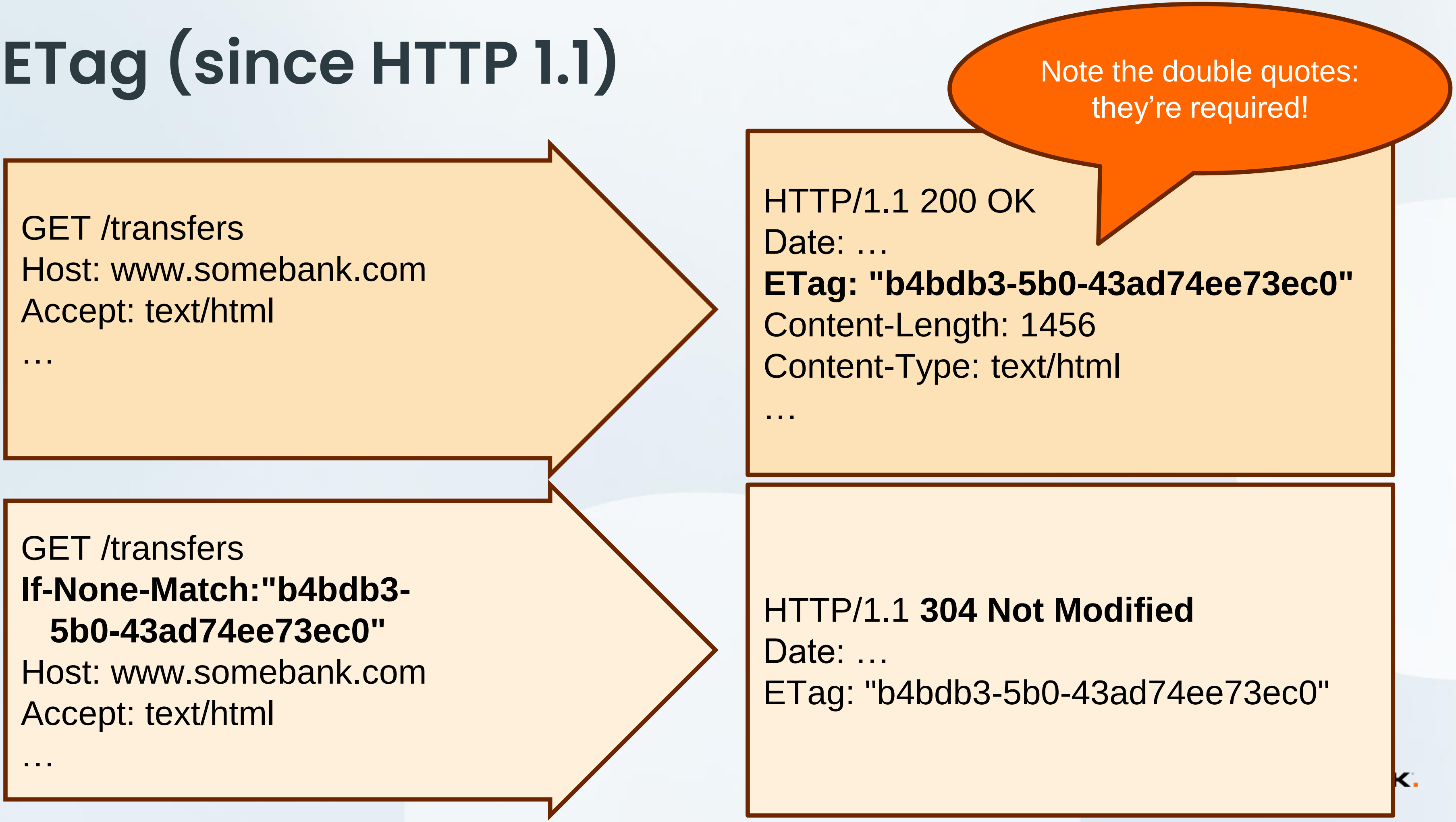
GET /transfers
Host: www.somebank.com
Accept: text/html
…

HTTP/1.1 200 OK
Date: …
Last-Modified:
Thu, 26 Apr 2024 22:12:17 GMT
Content-Length: 1456
Content-Type: text/html
…

GET /transfers
If-Modified-Since:
Thu, 26 Apr 2024 22:12:17 GMT
Host: www.somebank.com
Accept: text/html
…

HTTP/1.1 **304 Not Modified**
Date: …

ETag (since HTTP 1.1)



Note the double quotes: they're required!

```
GET /transfers
Host: www.somebank.com
Accept: text/html
…
```

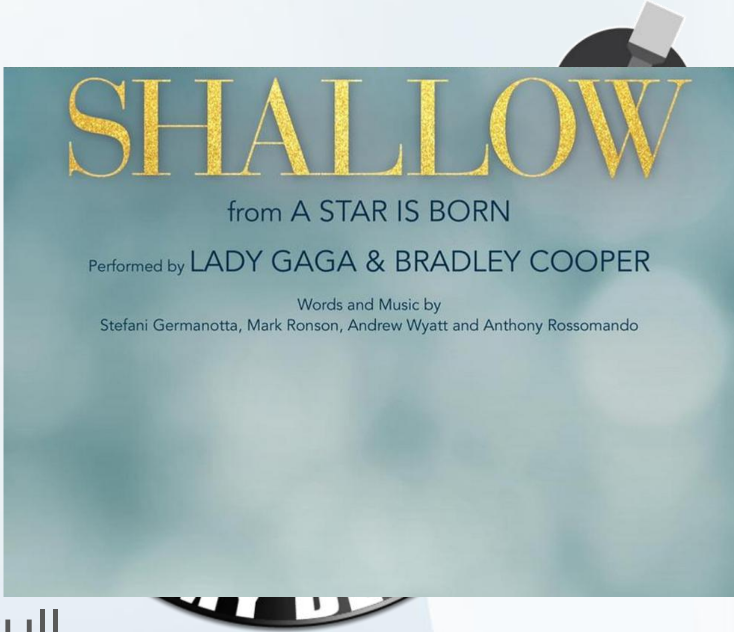
```
HTTP/1.1 200 OK
Date: …
ETag: "b4bdb3-5b0-43ad74ee73ec0"
Content-Length: 1456
Content-Type: text/html
…
```

```
GET /transfers
If-None-Match:"b4bdb3-5b0-43ad74ee73ec0"
Host: www.somebank.com
Accept: text/html
…
```

```
HTTP/1.1 304 Not Modified
Date: …
ETag: "b4bdb3-5b0-43ad74ee73ec0"
```

ETag Calculation

- **Shallow:**
  - Derive full response, then hash
  - +: saves bandwith & latency
  - −: still does all the work
- **Deep:**
  - Determine value *without* deriving full response
  - +: additional savings; don’t do all the work
  - −: harder and not always feasible


SHALLOW
from A STAR IS BORN
Performed by LADY GAGA & BRADLEY COOPER
Words and Music by
Stefani Germanotta, Mark Ronson, Andrew Wyatt and Anthony Rossomando

Demo

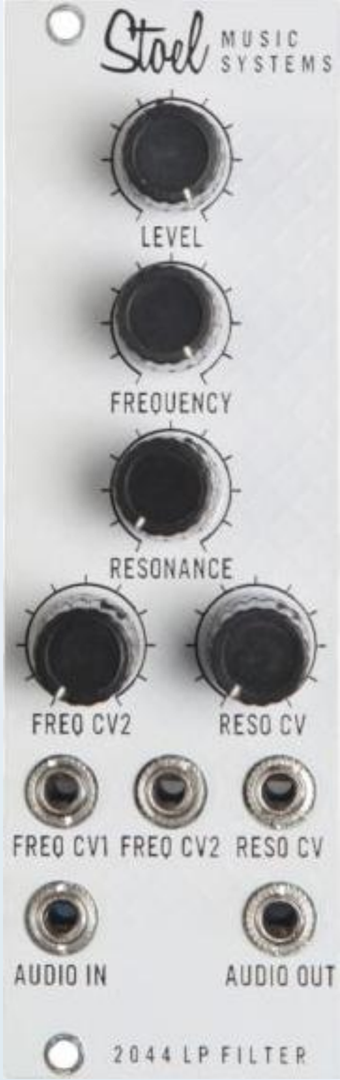
Final Comments

- Consider what mappings could benefit from HTTP Caching
  - Shallow or deep
- Even for resources that change frequently, caching might provide benefits
  - E.g. if clients poll frequently for the same resource
- Check if your clients respect the caching headers!

Ad-hoc HTTP Client Request Logging

Logging HTTP Client Requests

- Not provided by RestTemplate/~Client
- Underlying client support not suitable for production:
- No filtering or masking
    - Secrets, PII, cruft, …
- Always on, if enabled



Ad-hoc / Per-request Logging

- Enable logging for certain requests only
  - Based on header
- Propagate header downstream
  - As “*baggage*”;
    part of built-in tracing support

- Debug in production!

How To Build

- ClientHttpRequestInterceptor implementation
  - Applies to both RestTemplate and RestClient
  - Logs when enabled or on-demand
  - Can filter, mask, etc.
- Requires requests/responses to be *buffered*
- Register using *auto-configuration*

Auto-configuration

Configuration classes with conditions

- Applied automatically when containing jar is on the classpath
- Evaluated *after* regular app config
  - Can be ordered
- Often provide their own @ConfigurationProperties

Custom Auto-configuration

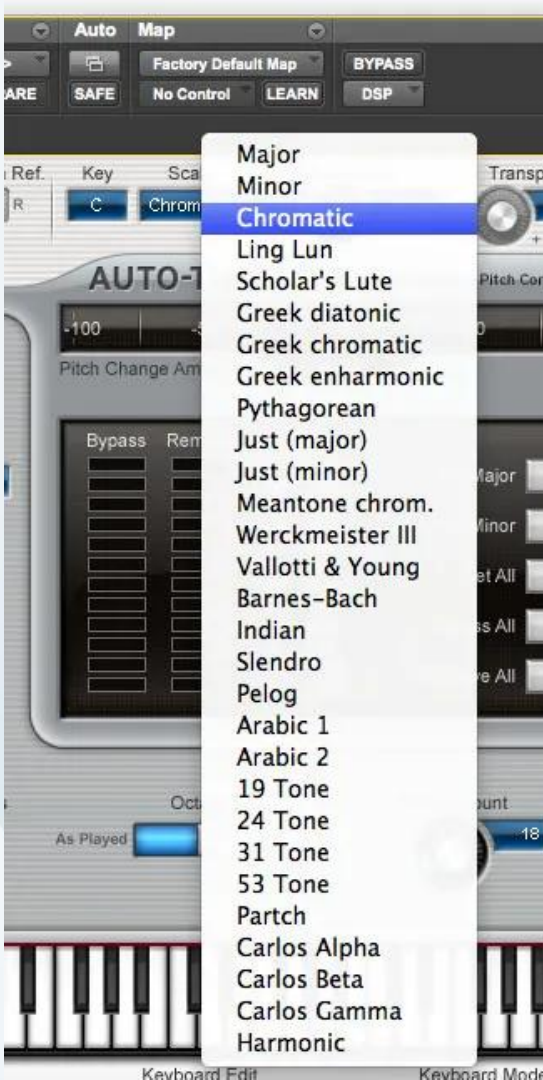
Very useful in your own apps

- Company-wide, or in multi-module project
- Support common patterns
  - like “local” or “test” Spring profile
- Examples:
  - Logging, HTTP client config, metrics, security, Jackson
  - MVC: error handling, common filters or argument resolvers, …

TRIFORK.

Demo

Actuator Autotune



Actuator Configuration Masking

- Config inspectable via Actuator endpoints
  - /env, /configprops
- Might contain secrets / sensitive values

- All masked by default
  - Configurable via property

Demo

Actuator Configuration Masking

- Older Boot versions only masked sensitive fields
  - But no common definition of “sensitive”
- Can do this yourself by providing bean of type SanitizingFunction!

Demo

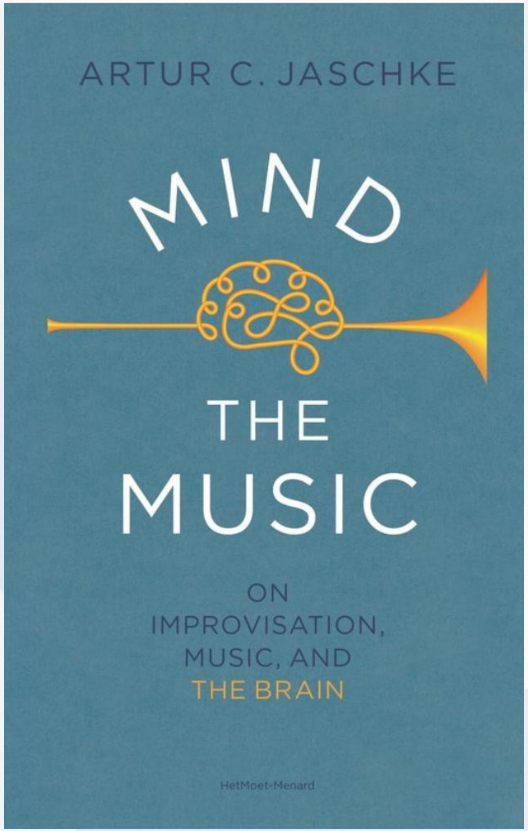
Metrics Mixing

Metrics in Spring Boot

- Many built-in metrics exposed via Micrometer
  - JVM, HTTP client/server, thread pools, etc.
- Easy to add your own
  - Programmatically or via annotations

Mind Your Metrics: Common Concerns

- Standard tags
  - host, service, …
- Restricting tag cardinality
  - # of possible values
- Restricting # of metrics
  - To reduce costs
- For HTTP client metrics: ensure host is included



Demo

Coda

- There's more to production-ready apps than app logic
- Boot provides all you need, but maybe not out-of-the-box
  - Don't be afraid to tune
  - Learn what your options are!
- Build mini-frameworks for your apps
  - After all, it's ***your mix***!

Thank you!

https://github.com/jkuipers/in-the-mix-demos

CTO TRIFORK AMSTERDAM

Joris Kuipers

trifork.nl

@jkuipers, @triforkams

TRIFORK.