Владимир Красильщик, Luxoft, Санкт-Петербург

Что надо знать о логировании прагматичному Java-программисту

Москва, 2016

О чем доклад

О чем доклад

- Что было не так с логированием на проекте X

О чем доклад

- Что было не так с логированием на проекте X
- Что мы захотели изменить

О чем доклад

- Что было не так с логированием на проекте X
- Что мы захотели изменить
- Что у нас получилось

О чем доклад

- Что было не так с логированием на проекте X
- Что мы захотели изменить
- Что у нас получилось
- Чему мы научились

Два слова о проекте X

Два слова о проекте X

- ~40 микросервисов

Два слова о проекте X

- ~40 микросервисов
- >16 лет в продакшене

Два слова о проекте X

- ~40 микросервисов
- >16 лет в продакшене
- ~20 сервер-сайд Java-девелоперов

Что было

Что было

В разных микросервисах использовались разные библиотеки логирования

Что было

В разных микросервисах использовались разные библиотеки логирования

- Плюс: становишься “недоэкспертом” в разных библиотеках логирования …

Что было

В разных микросервисах использовались разные библиотеки логирования

- Плюс: становишься “недоэкспертом” в разных библиотеках логирования …
- Минус: big data, high-load, reactive, пиво и дети проходят мимо пока …

Что было

В разных микросервисах использовались разные библиотеки логирования

- Плюс: становишься “недоэкспертом” в разных библиотеках логирования …
- Минус: big data, high-load, reactive, пиво и дети проходят мимо пока …

… пока ты вспоминаешь как правильно писать в этом микросервисе

1) `log.info(“userName is {}”, name);`
2) `log.info(String.format(“userName is %s”, name));`

Что было

Что было

“Jars Hell” библиотек логирования

Что было

“Jars Hell” библиотек логирования

- Плюс: очень интересно разбираться в хитросплетениях “класспасостроения”

Что было

“Jars Hell” библиотек логирования

- Плюс: очень интересно разбираться в хитросплетениях “класспасостроения”
- Минус: забываешь в каком класспасе, тьфу, классе твой ребёнок

Что было

|~/microService1
|---/apache-tomcat
|-------/bin
|-------/conf
|-------/**logs**
|~/microService2
|---/apache-tomcat/
|-------/bin
|-------/conf
|-------/**logs**
|~/microServiceN
|---/apache-tomcat
|-------/bin
|-------/conf
|-------/**logs**

Что было

Что было

Архивирование логов делалось ручками: crontab + sh-скрипт

Что было

Архивирование логов делалось ручками: crontab + sh-скрипт

- Плюс: ты контролируешь
  все сам

Что было

Архивирование логов делалось ручками: crontab + sh-скрипт

- Плюс: ты контролируешь все сам
- Минус: но не контролируешь админов, которые …

Что было

Архивирование логов делалось ручками: crontab + sh-скрипт

- Плюс: ты контролируешь все сам
- Минус: но не контролируешь админов, которые …

… которые все-равно могут снести gnu утилиту, используемую в sh-скрипте в продакшене или удалить расписание из crontab

Что было

Что было

|~/microServiceK
|---/apache-tomcat
|-------/bin
|-------/conf
|-------/logs
|-----------/**archive**

Что было

|~/microServiceK
|---/apache-tomcat
|-------/bin
|-------/conf
|-------/logs
|-----------/**archive**

- Плюс: обновление томката до новой версии становится интересной задачей

Что было

|~/microServiceK
|---/apache-tomcat
|-------/bin
|-------/conf
|-------/logs
|-----------/**archive**

- Плюс: обновление томката до новой версии становится интересной задачей
- Минус: *“Здесь было жестоко убито время”**

* - надпись на парте в родном СПбГЭТУ “ЛЭТИ”

Что было

Что было

Свой велосипед со статическими методами `Log.debug(),Log.info(),…`

Что было

Свой велосипед со статическими методами `Log.debug(),Log.info(),…`

- Плюс: не надо явно указывать имя логера

Что было

Свой велосипед со статическими методами `Log.debug(),Log.info(),…`

- Плюс: не надо явно указывать имя логера
- Минус: ненадежная имплементация …

```
… new Exception().getStackTrace()[4].getClassName()
```

Что захотели изменить

Что захотели изменить

- Одна технологию логирования на все микросервисы

Что захотели изменить

- Одна технологию логирования на все микросервисы
- Избавиться от crontab+sh для архивирования логов

Что захотели изменить

- Одна технологию логирования на все микросервисы
- Избавиться от crontab+sh для архивирования логов
- Упорядочить архивы всех микросервисов

Что захотели изменить

- Одна технологию логирования на все микросервисы
- Избавиться от crontab+sh для архивирования логов
- Упорядочить архивы всех микросервисов
- Удобно просматривать и скачивать логи для анализа

Что захотели изменить

- Одна технологию логирования на все микросервисы
- Избавиться от crontab+sh для архивирования логов
- Упорядочить архивы всех микросервисов
- Удобно просматривать и скачивать логи для анализа
- И уже забыть про логи и начать жить …


пиво
reactive
highload
дети
bigdata
data
log4j
module
logback
logbackclassic
logbackcore
logbackaccess
logbackaudit
projects
version
divided
Logbacks
two
Moreover
provide
significantly
successor
circumstances
Tomcat
easily
JUL
natively
API
log
modules
implements
apply
assimilated
picking
jetty
logging
integrates
switch
known
popular
Servlet
back
groundwork
processing
time
project
different
HTTPaccess
javautilslogging
generic
functionality
three
top
lays
present
big
intended
SLF4J
frameworks
readily
containers
forth
improved
sufficiently
build
Sister
designed
architecture
Note
leaves

А какие библиотеки есть?

А какие библиотеки есть?

1. Java Util Logging (JUL/JDK)

А какие библиотеки есть?

1. Java Util Logging (JUL/JDK)
2. log4j

А какие библиотеки есть?

1. Java Util Logging (JUL/JDK)
2. log4j
3. Logback

А какие библиотеки есть?

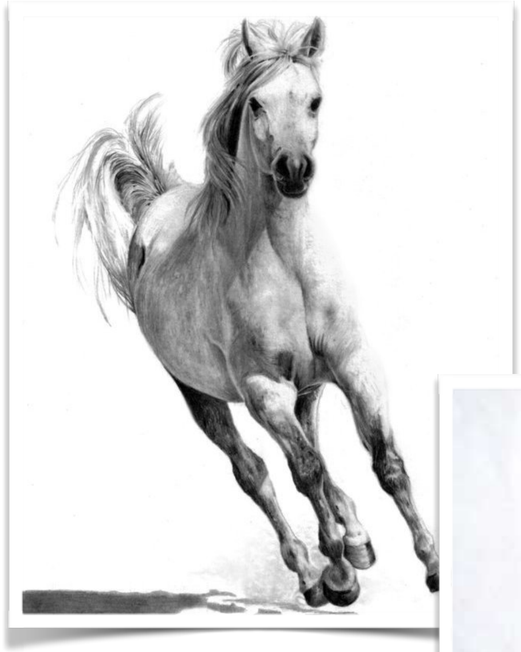
1. Java Util Logging (JUL/JDK)
2. log4j
3. Logback
4. log4j 2

А какие библиотеки есть?

1. Java Util Logging (JUL/JDK)
2. log4j
3. Logback
4. log4j 2

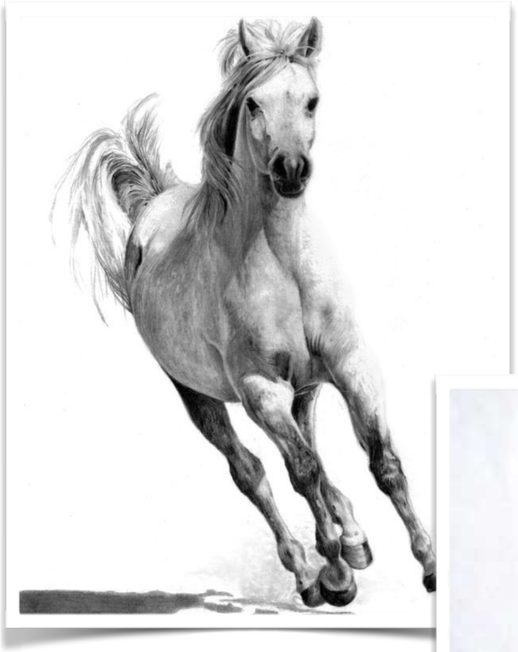
А какие библиотеки есть?

1. Java Util Logging (JUL/JDK)
2. log4j
3. Logback
4. log4j 2



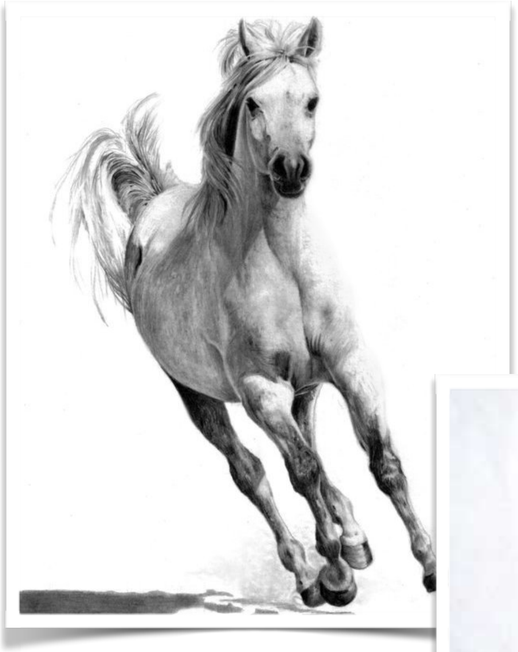
А какие библиотеки есть?

1. Java Util Logging (JUL/JDK)
2. log4j
3. Logback
4. log4j 2
5. Apache Commons Logging (JCL)



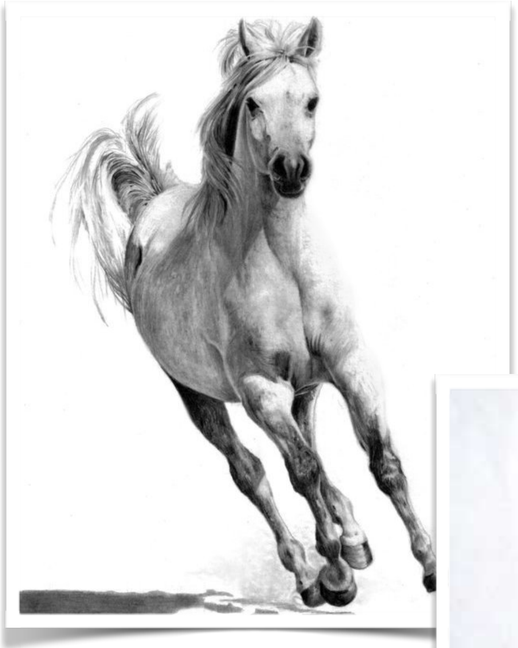
А какие библиотеки есть?

1. Java Util Logging (JUL/JDK)
2. log4j
3. Logback
4. log4j 2
5. Apache Commons Logging (JCL)
6. Simple Logging Facade For Java (SLF4J)



А какие библиотеки есть?

1. Java Util Logging (JUL/JDK)
2. log4j
3. **Logback**
4. log4j 2
5. Apache Commons Logging (JCL)
6. **Simple Logging Facade For Java (SLF4J)**




SLF4J
Simple Logging
Facade for Java


LOGBack™
The Generic, Reliable
Fast & Flexible
Logging Framework

Общий подход в логировании на 1-й квартал 2016 года

Общий подход в логировании на 1-й квартал 2016 года

- Logger

Общий подход в логировании на 1-й квартал 2016 года

- Logger
- Appender

Общий подход в логировании на 1-й квартал 2016 года

- Logger
- Appender
- “in-team-ная связь” Logger—>Appender

Logger

Logger

- `Logger logger = LoggerFactory.getLogger(name);`

Logger

- `Logger logger = LoggerFactory.getLogger(name);`
- `name = “vasiliy_pupkin”;`

Logger

- `Logger logger = LoggerFactory.getLogger(name);`
- `name = “vasiliy_pupkin”;`
- `name = SomeClass.class.getName();`

Logger

- `Logger logger = LoggerFactory.getLogger(name);`
- `name = “vasiliy_pupkin”;`
- `name = SomeClass.class.getName();`
- `“ru.spb.luxoft.x.y.z.SomeClass”`

Logger

- `Logger logger = LoggerFactory.getLogger(name);`
- `name = “vasiliy_pupkin”;`
- `name = SomeClass.class.getName();`
- `“ru.spb.luxoft.x.y.z.SomeClass”`
- `“ru.spb.luxoft”` это родитель для `“ru.spb.luxoft.x.y.z.SomeClass”`

Logger

- `Logger logger = LoggerFactory.getLogger(name);`
- `name = “vasiliy_pupkin”;`
- `name = SomeClass.class.getName();`
- `“ru.spb.luxoft.x.y.z.SomeClass”`
- `“ru.spb.luxoft”` это родитель для `“ru.spb.luxoft.x.y.z.SomeClass”`
- `root<–ru<–spb<–luxoft<–x<–y<–z<–SomeClass`

Logger

- `Logger logger = LoggerFactory.getLogger(name);`
- `name = “vasiliy_pupkin”;`
- `name = SomeClass.class.getName();`
- `“ru.spb.luxoft.x.y.z.SomeClass”`
- `“ru.spb.luxoft”` это родитель для `“ru.spb.luxoft.x.y.z.SomeClass”`
- `root<—ru<—spb<—luxoft<—x<—y<—z<—SomeClass`
- `root<—vasiliy_pupkin`

Logger

Logger

- ```
  logger.info(“hello”);
  logger.debug(“hello”);
  ```
  ```

Logger

- ```
  logger.info(“hello”);
  logger.debug(“hello”);
  ```
- Event:
  ```
  message = ”hello”
  Level   = Level.INFO
  ```
  ```

Logger

- ```
  logger.info(“hello”);
  logger.debug(“hello”);
  ```
- Event:
  ```
  message = ”hello”
  Level   = Level.INFO
  ```
- Уровни “относительной ужасности”:
  error—>warn—>info—>debug—>fine
  ```

Appender

Appender

- Appender - знает куда отправлять события

Appender

- Appender - знает куда отправлять события
- Console

Appender

- Appender - знает куда отправлять события
- Console
- File

Appender

- Appender - знает куда отправлять события
- Console
- File
- Socket

Appender

- Appender - знает куда отправлять события
- Console
- File
- Socket
- Database

Appender

- Appender - знает куда отправлять события
- Console
- File
- Socket
- Database
- Abstract-Spherical-Horse-In-Vacuum

“in-team-ная связь”
Logger—>Appender

“in-team-ная связь” Logger—>Appender

- `ru.spb.luxoft.x.y.z.SomeClass—>fb` (1)

“in-team-ная связь” Logger—>Appender

- `ru.spb.luxoft.x.y.z.SomeClass—>fb` (1)
- `ru.spb.luxoft—>vk` (2)

“in-team-ная связь” Logger—>Appender

- `ru.spb.luxoft.x.y.z.SomeClass—>fb` (1)
- `ru.spb.luxoft—>vk` (2)
- `root—>ok` (3)

“in-team-ная связь” Logger—>Appender

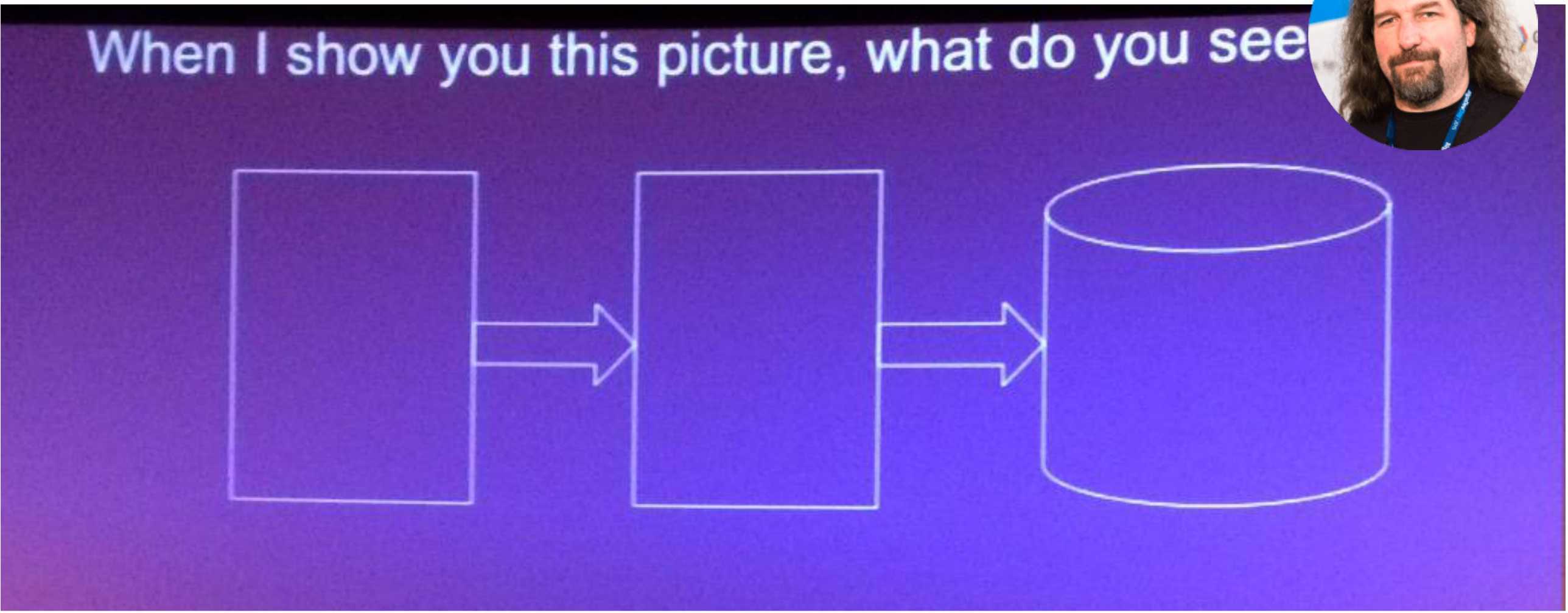
- `ru.spb.luxoft.x.y.z.SomeClass—>fb` (1)
- `ru.spb.luxoft—>vk` (2)
- `root—>ok` (3)
- `LoggerFactory.`*`getLogger`*`(SomeClass.`**`class`**`)`
  `.info(“I <3 Big Data!”)?`

“in-team-ная связь” Logger—>Appender

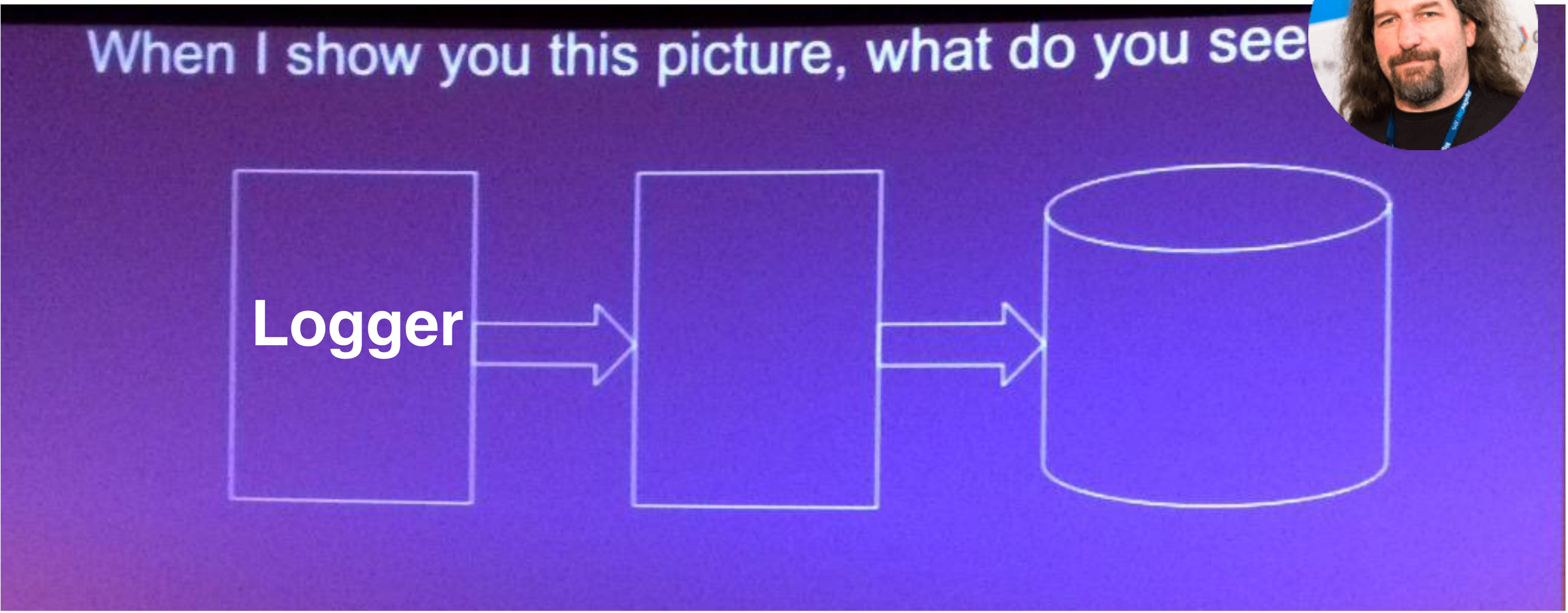
- `ru.spb.luxoft.x.y.z.SomeClass—>fb` (1)
- `ru.spb.luxoft—>vk` (2)
- `root—>ok` (3)
- `LoggerFactory.getLogger(SomeClass.class)`
  `.info(“I <3 Big Data!”)?`
- `root<—ru<—spb<—luxoft<—x<—y<—z<—SomeClass`

“in-team-ная связь” Logger—>Appender

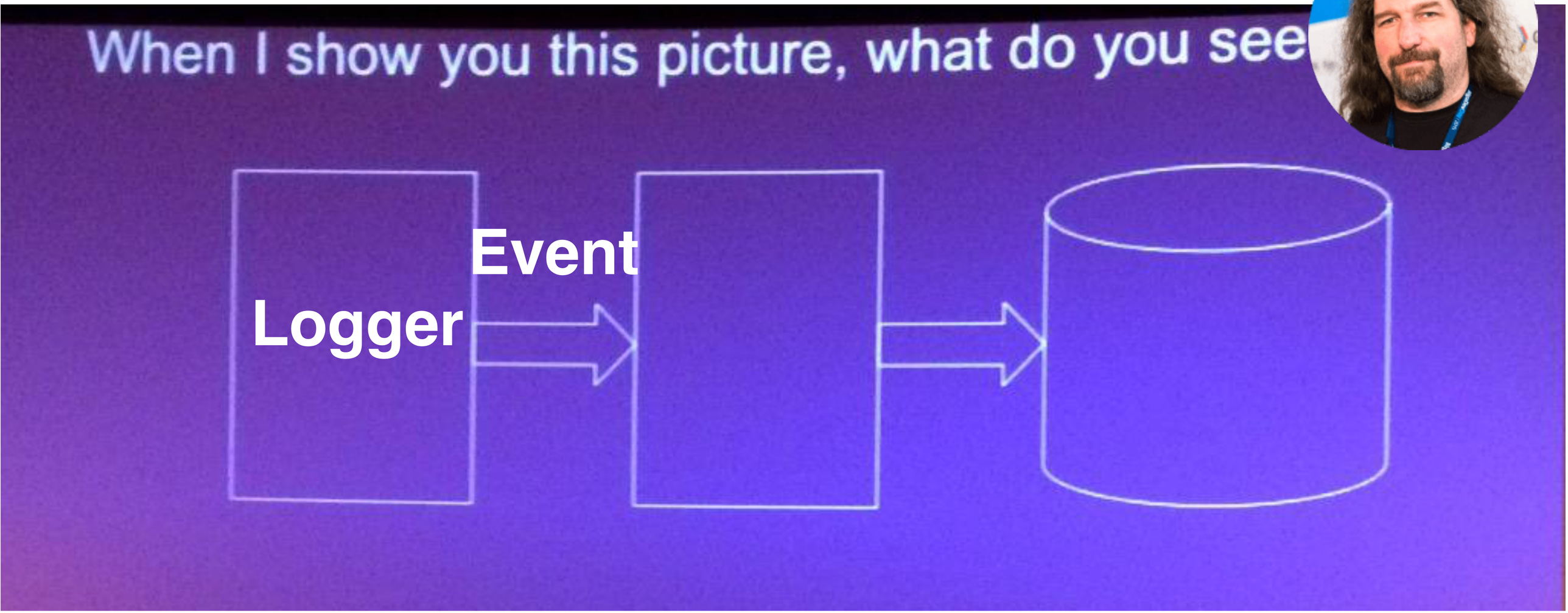
- ru.spb.luxoft.x.y.z.SomeClass—>fb (1)
- ru.spb.luxoft—>vk (2)
- root—>ok (3)
- LoggerFactory.*getLogger*(SomeClass.**class**)
  .info(“I <3 Big Data!”)?
- root<—ru<—spb<—luxoft<—x<—y<—z<—SomeClass
- additivity …



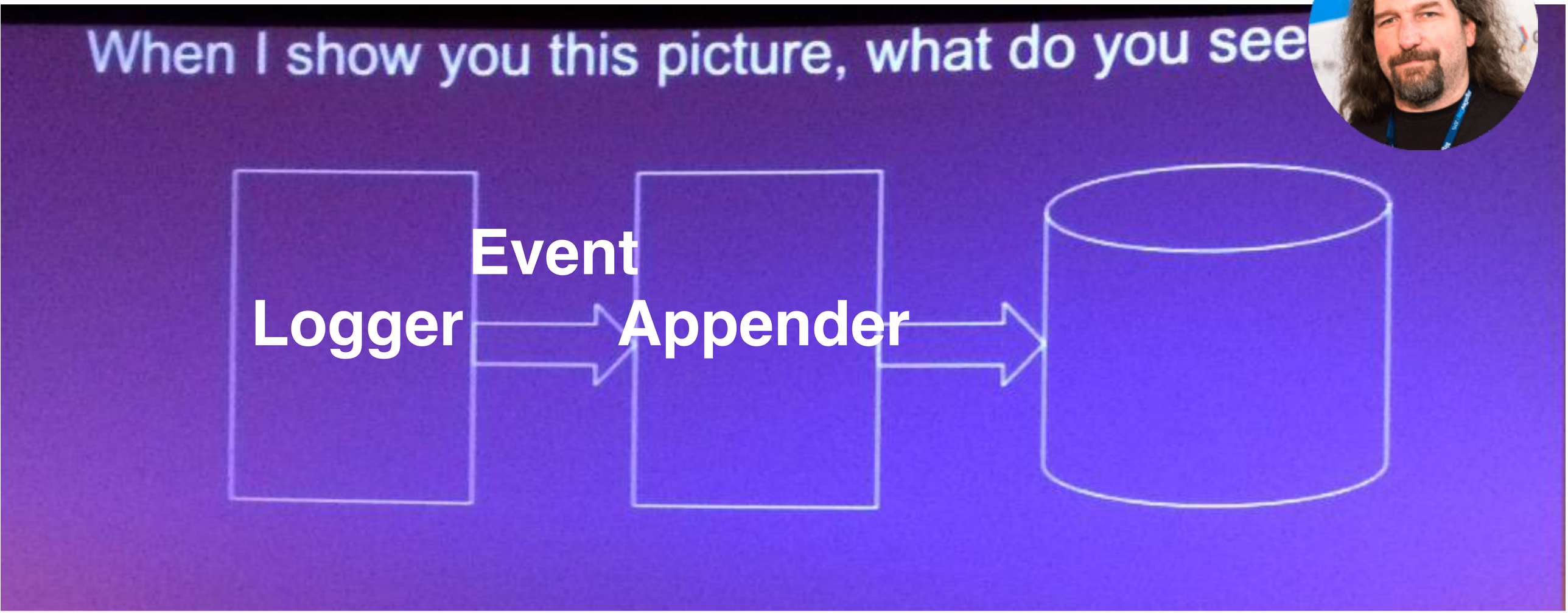
Универсальная Программная Архитектурная Диаграмма (УПАД)
by @tedneward



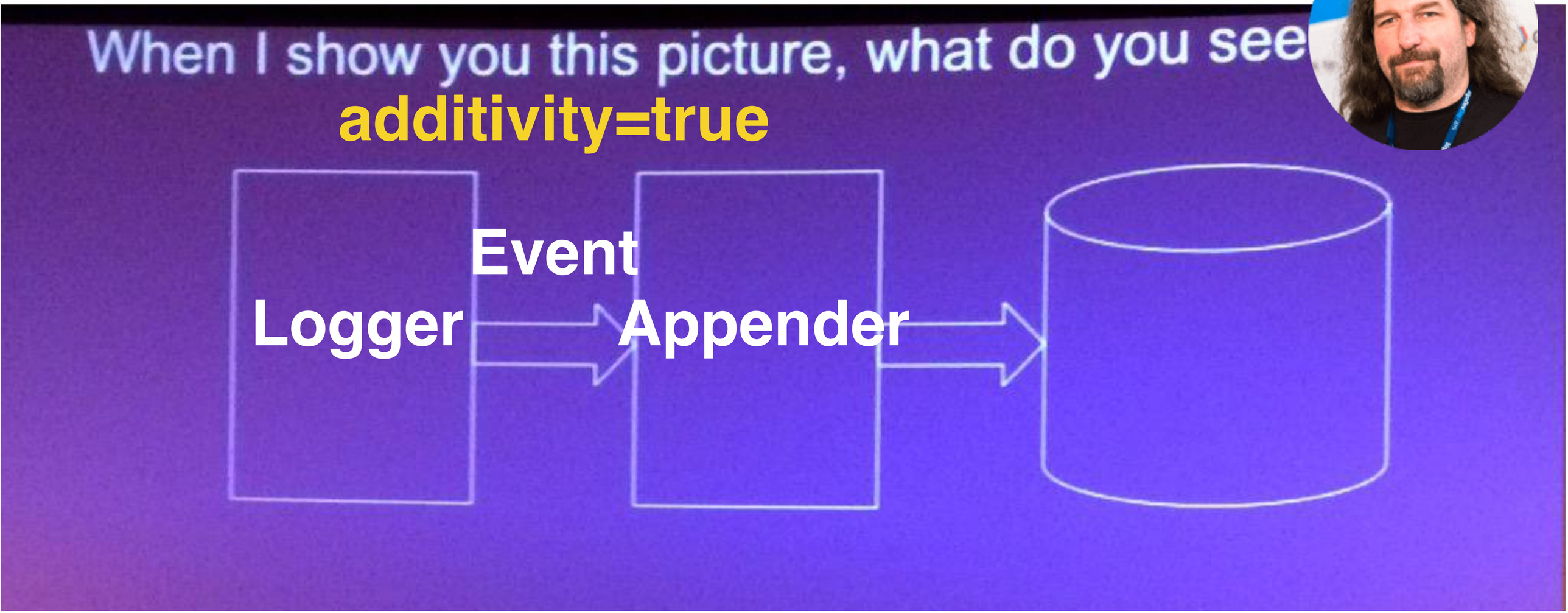
Универсальная Программная Архитектурная Диаграмма (УПАД)
by @tedneward



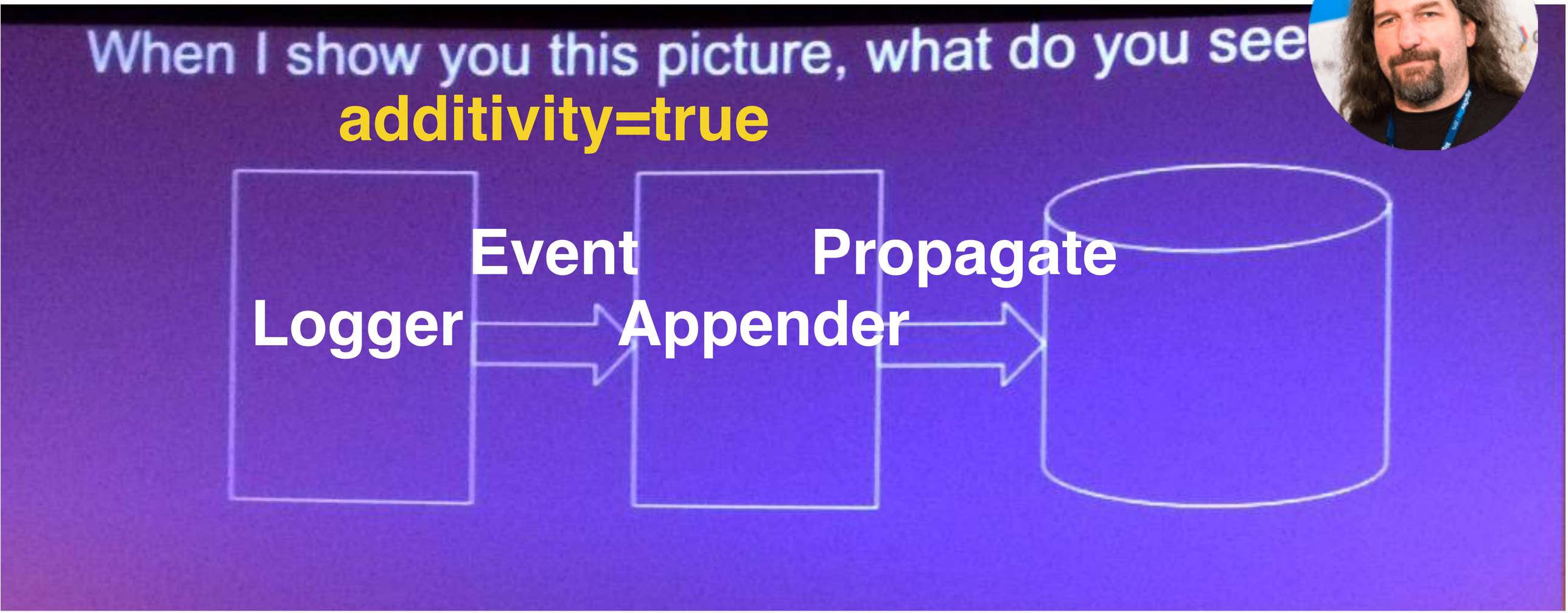
Универсальная Программная Архитектурная Диаграмма (УПАД)
by @tedneward



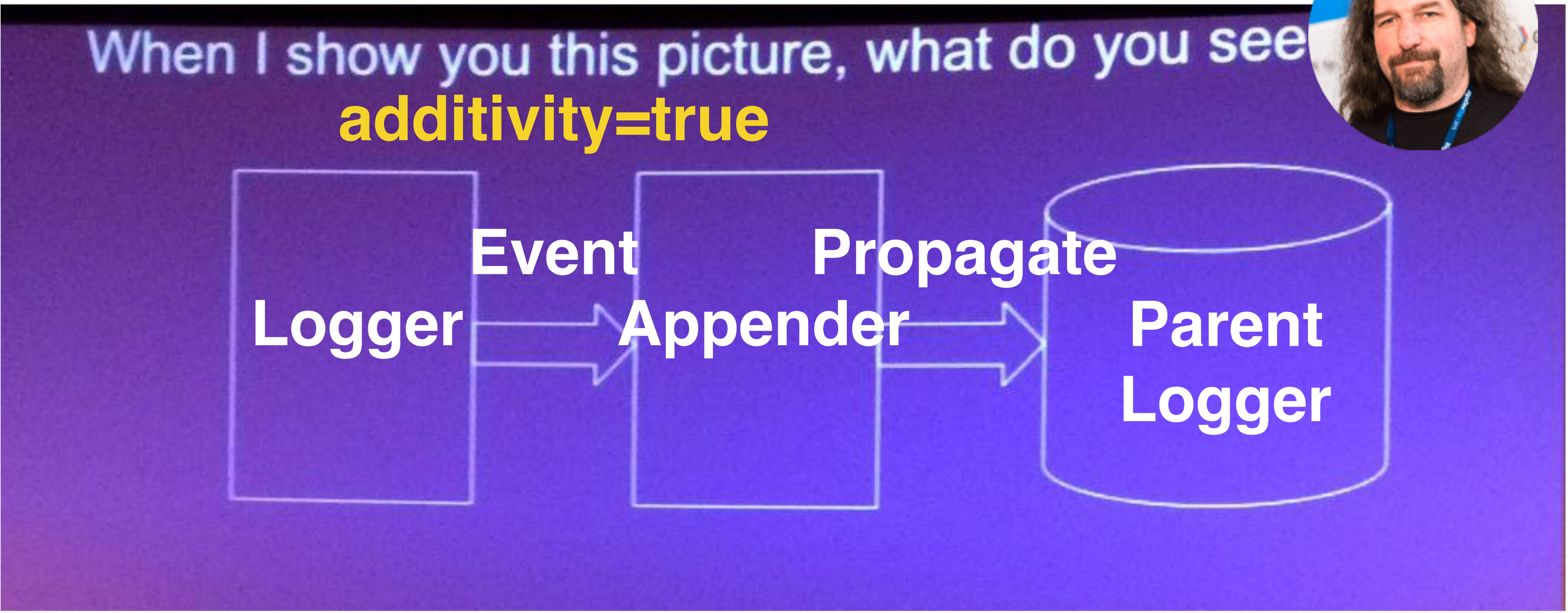
Универсальная Программная Архитектурная Диаграмма (УПАД)
by @tedneward



Универсальная Программная Архитектурная Диаграмма (УПАД)
by @tedneward



Универсальная Программная Архитектурная Диаграмма (УПАД)
by @tedneward



Универсальная Программная Архитектурная Диаграмма (УПАД)
by @tedneward

“in-team-ная связь”
Logger—>Appender

“in-team-ная связь” Logger—>Appender

- Уровень “относительной ужасности” in-team-ной связи

“in-team-ная связь” Logger—>Appender

- Уровень “относительной ужасности” in-team-ной связи
- `Logger logger = LoggerFactory.getLogger(“vasiliy_pupkin”);`
  `vasiliy_pupkin—>Abstract-Spherical-Horse-In-Vacuum`
  `Level = Level.INFO`

“in-team-ная связь” Logger—>Appender

- Уровень “относительной ужасности” in-team-ной связи
- `Logger logger = LoggerFactory.getLogger(“vasiliy_pupkin”);`
  `vasiliy_pupkin—>Abstract-Spherical-Horse-In-Vacuum`
  `Level = Level.INFO`
- `logger.error() - V`
  `logger.warn() - V`
  `logger.info() - V`
  `logger.debug() - X`
  `logger.fine() - X`

error—>warn—>info—>debug—>fine

Что получилось и чему научились

SLF4J

SLF4J

API в slf4j-api.jar

SLF4J

API в slf4j-api.jar

```
Logger logger = LoggerFactory.getLogger(SomeClass.class);
logger.info(“hello”);
```

SLF4J

SLF4J

Binding

SLF4J

Binding

- logback-classic-1.1.3.jar

SLF4J

Binding

- logback-classic-1.1.3.jar
- slf4j-**log4j**12-1.7.12.jar

SLF4J

Binding

- logback-classic-1.1.3.jar
- slf4j-**log4j**12-1.7.12.jar
- slf4j-**jdk**14-1.7.12.jar

SLF4J

Binding

- logback-classic-1.1.3.jar
- slf4j-**log4j**12-1.7.12.jar
- slf4j-**jdk**14-1.7.12.jar
- slf4j-**jcl**-1.7.12.jar

Troubleshooting Multiple Bindings

Troubleshooting Multiple Bindings

SLF4J: Class path contains multiple SLF4J bindings.

SLF4J: Found binding in [jar:file:/export/apps/userR/serviceK/apache-tomee-webprofile/lib/**slf4j-jdk14-1.7.2.jar**!/org/slf4j/impl/StaticLoggerBinder.class]

SLF4J: Found binding in [jar:file:/export/apps/userR/serviceK/apache-tomee-webprofile/lib/**logback-classic-1.0.7.jar**!/org/slf4j/impl/StaticLoggerBinder.class]

SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.

SLF4J: Actual binding is of type [org.slf4j.impl.**JDK**14LoggerFactory]

Troubleshooting Multiple Bindings

Troubleshooting Multiple Bindings

```
mvn dependency:tree -DoutputFile=tree.txt
-Dincludes=org.slf4j -Dverbose > output
```

Troubleshooting Multiple Bindings

```
<dependency>
    <groupId>org.apache.openejb</groupId>
    <artifactId>openejb-core</artifactId>
    <version>4.7.1</version>
    <scope>provided</scope>
    <exclusions>
        <exclusion>
            <groupId>org.slf4j</groupId>
            <artifactId>slf4j-jdk14</artifactId>
        </exclusion>
    </exclusions>
    </dependency>
<dependency>
```

Нежданчик #0: некоторые либы используют явно log4j или JUL/JDK

Нежданчик #0: некоторые либы используют явно log4j или JUL/JDK

- Что Spring использует для логирования?

Нежданчик #0: некоторые либы используют явно log4j или JUL/JDK

- Что Spring использует для логирования?
- Spring использует JCL!

Bridge в SLF4J

Bridge в SLF4J

- jcl-**over**-slf4j.jar

Bridge в SLF4J

- jcl-**over**-slf4j.jar
- log4j-**over**-slf4j.jar

Bridge в SLF4J

- jcl-**over**-slf4j.jar
- log4j-**over**-slf4j.jar
- jul-**to**-slf4j.jar (ничего не смущает?)

Bridge в SLF4J - проблемы

Bridge в SLF4J - проблемы

- не работает, если конфигурация сделана программно

Bridge в SLF4J - проблемы

- не работает, если конфигурация сделана программно
- **jul-to**-slf4j.jar && slf4j-**jdk**14.jar —> StackOverflowError
  log4j-over-slf4j.jar && slf4j-**log4j**12.jar —> StackOverflowError

Bridge в SLF4J - проблемы

- не работает, если конфигурация сделана программно
- **jul-to**-slf4j.jar && slf4j-**jdk**14.jar —> StackOverflowError
  log4j-over-slf4j.jar && slf4j-**log4j**12.jar —> StackOverflowError
- jul-**to**-slf4j.jar —> SLF4JBridgeHandler на root и …

Bridge в SLF4J - проблемы

- не работает, если конфигурация сделана программно
- **jul-to**-slf4j.jar && slf4j-**jdk**14.jar —> StackOverflowError
  log4j-over-slf4j.jar && slf4j-**log4j**12.jar —> StackOverflowError
- jul-**to**-slf4j.jar —> SLF4JBridgeHandler на root и …
- … и проседает производительность

“Швейцарский” файловый аппендер

“Швейцарский” файловый аппендер

```
<appender name="SERVICE" class=“ch.qos.logback.core.rolling.RollingFileAppender"> (1)
```

“Швейцарский” файловый аппендер

```
<appender name="SERVICE" class=“ch.qos.logback.core.rolling.RollingFileAppender"> (1)
        <file>${catalina.base}/logs/serviceK.log</file>                            (2)
```

“Швейцарский” файловый аппендер

```
<appender name="SERVICE" class=“ch.qos.logback.core.rolling.RollingFileAppender"> (1)
        <file>${catalina.base}/logs/serviceK.log</file>                           (2)
        <rollingPolicy class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy"> (3)
```

“Швейцарский” файловый аппендер

```
 <appender name="SERVICE" class=“ch.qos.logback.core.rolling.RollingFileAppender"> (1)
        <file>${catalina.base}/logs/serviceK.log</file>                              (2)
        <rollingPolicy class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy"> (3)
            <fileNamePattern>${user.home}/archive/serviceK/%d{yyyy-MM-dd}/serviceK-
%d{yyyy-MM-dd_HH-mm,aux}.log.gz</fileNamePattern>                                    (4)
```

“Швейцарский” файловый аппендер

```
 <appender name="SERVICE" class=“ch.qos.logback.core.rolling.RollingFileAppender"> (1)
        <file>${catalina.base}/logs/serviceK.log</file>                             (2)
        <rollingPolicy class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy"> (3)
            <fileNamePattern>${user.home}/archive/serviceK/%d{yyyy-MM-dd}/serviceK-
%d{yyyy-MM-dd_HH-mm,aux}.log.gz</fileNamePattern>                                   (4)
            <maxHistory>90</maxHistory>                                             (5)
```

“Швейцарский” файловый аппендер

```
 <appender name="SERVICE" class=“ch.qos.logback.core.rolling.RollingFileAppender"> (1)
        <file>${catalina.base}/logs/serviceK.log</file>                              (2)
        <rollingPolicy class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy"> (3)
            <fileNamePattern>${user.home}/archive/serviceK/%d{yyyy-MM-dd}/serviceK-
%d{yyyy-MM-dd_HH-mm,aux}.log.gz</fileNamePattern>                                    (4)
            <maxHistory>90</maxHistory>                                              (5)
        </rollingPolicy>
```

“Швейцарский” файловый аппендер

```
 <appender name="SERVICE" class=“ch.qos.logback.core.rolling.RollingFileAppender"> (1)
        <file>${catalina.base}/logs/serviceK.log</file>                              (2)
        <rollingPolicy class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy"> (3)
            <fileNamePattern>${user.home}/archive/serviceK/%d{yyyy-MM-dd}/serviceK-
%d{yyyy-MM-dd_HH-mm,aux}.log.gz</fileNamePattern>                                    (4)
            <maxHistory>90</maxHistory>                                              (5)
        </rollingPolicy>
        <encoder>
```

“Швейцарский” файловый аппендер

```
 <appender name="SERVICE" class=“ch.qos.logback.core.rolling.RollingFileAppender"> (1)
        <file>${catalina.base}/logs/serviceK.log</file>                             (2)
        <rollingPolicy class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy"> (3)
            <fileNamePattern>${user.home}/archive/serviceK/%d{yyyy-MM-dd}/serviceK-
%d{yyyy-MM-dd_HH-mm,aux}.log.gz</fileNamePattern>                                   (4)
            <maxHistory>90</maxHistory>                                             (5)
        </rollingPolicy>
        <encoder>
            <pattern>%d [%22.22thread] [%-5level] [%logger{0}] %msg%n%ex</pattern> (6)
```

“Швейцарский” файловый аппендер

```
 <appender name="SERVICE" class=“ch.qos.logback.core.rolling.RollingFileAppender"> (1)
        <file>${catalina.base}/logs/serviceK.log</file>                             (2)
        <rollingPolicy class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy"> (3)
            <fileNamePattern>${user.home}/archive/serviceK/%d{yyyy-MM-dd}/serviceK-
%d{yyyy-MM-dd_HH-mm,aux}.log.gz</fileNamePattern>                                   (4)
            <maxHistory>90</maxHistory>                                             (5)
        </rollingPolicy>
        <encoder>
            <pattern>%d [%22.22thread] [%-5level] [%logger{0}] %msg%n%ex</pattern> (6)
        </encoder>
```

“Швейцарский” файловый аппендер

```
 <appender name="SERVICE" class=“ch.qos.logback.core.rolling.RollingFileAppender"> (1)
        <file>${catalina.base}/logs/serviceK.log</file>                             (2)
        <rollingPolicy class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy"> (3)
            <fileNamePattern>${user.home}/archive/serviceK/%d{yyyy-MM-dd}/serviceK-
%d{yyyy-MM-dd_HH-mm,aux}.log.gz</fileNamePattern>                                   (4)
            <maxHistory>90</maxHistory>                                             (5)
        </rollingPolicy>
        <encoder>
            <pattern>%d [%22.22thread] [%-5level] [%logger{0}] %msg%n%ex</pattern> (6)
        </encoder>
</appender>
```

Миграционые службы

Миграционые службы

- http://www.slf4j.org/migrator.html - миграция на SLF4J из JCL, log4j или JUL/JDK

Миграционые службы

- http://www.slf4j.org/migrator.html - миграция на SLF4J из JCL, log4j или JUL/JDK

  Ограничения:
  1) fatal() - X
  2) Object в параметрах - X
  3) 2 логера в одной строке кода - X

Миграционые службы

- http://www.slf4j.org/migrator.html - миграция на SLF4J из JCL, log4j или JUL/JDK

  Ограничения:
  1) fatal() - X
  2) Object в параметрах - X
  3) 2 логера в одной строке кода - X

- http://logback.qos.ch/translator/ - миграция на Logback из log4j

Миграционые службы

- http://www.slf4j.org/migrator.html - миграция на SLF4J из JCL, log4j или JUL/JDK

  Ограничения:
  1) fatal() - X
  2) Object в параметрах - X
  3) 2 логера в одной строке кода - X

- http://logback.qos.ch/translator/ - миграция на Logback из log4j

  Ограничения:
  1) только из конфигурации через *.properties
  2) генерит немного кривую конфигурацию Logback

Нежданчик #1: GWT

Нежданчик #1: GWT

- Прошёлся migrator-ом по всему коду

Нежданчик #1: GWT

- Прошёлся migrator-ом по всему коду
- В девелоп моде все пучком

Нежданчик #1: GWT

- Прошёлся migrator-ом по всему коду
- В девелоп моде все пучком
- При сборке продакшена падает ошибка компиляции класса `org.slf4j.LoggerFactory`

Нежданчик #1: GWT

- Прошёлся migrator-ом по всему коду
- В девелоп моде все пучком
- При сборке продакшена падает ошибка компиляции класса `org.slf4j.LoggerFactory`
- JUL/JDK надо оставить в GWT UI коде чтобы генерировалось `console.debug()` в javascript

Нежданчик #2: Hibernate

Нежданчик #2: Hibernate

- Проблема: надо подебажить запросы Hibernate после перехода на новую версию

Нежданчик #2: Hibernate

- Проблема: надо подебажить запросы Hibernate после перехода на новую версию
- `show_sql=true` —> и тишина

Нежданчик #2: Hibernate

- Проблема: надо подебажить запросы Hibernate после перехода на новую версию
- `show_sql=true` —> и тишина
- Добавил “in-team-ной связи” для логеров Hibernate —> и тишина

Нежданчик #2: Hibernate

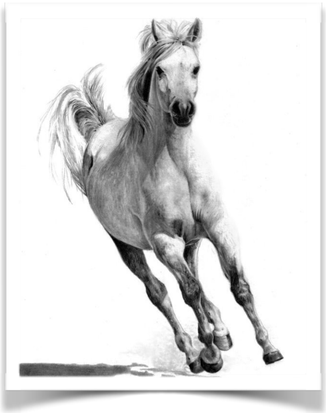
- Проблема: надо подебажить запросы Hibernate после перехода на новую версию
- `show_sql=true` —> и тишина
- Добавил “in-team-ной связи” для логеров Hibernate —> и тишина
- Hibernate 4.X.X+ использует JBoss Logging!

Нежданчик #2: Hibernate

- Проблема: надо подебажить запросы Hibernate после перехода на новую версию
- `show_sql=true` —> и тишина
- Добавил “in-team-ной связи” для логеров Hibernate —> и тишина
- Hibernate 4.X.X+ использует JBoss Logging!
- JBoss Logging - это ещё один адаптер!

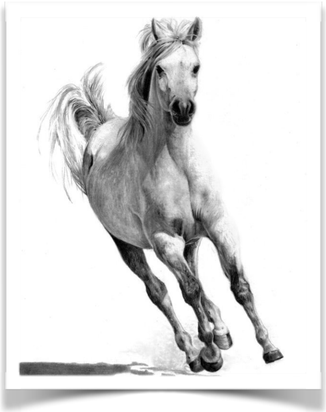
Нежданчик #2: Hibernate

- Проблема: надо подебажить запросы Hibernate после перехода на новую версию
- `show_sql=true` —> и тишина
- Добавил “in-team-ной связи” для логеров Hibernate —> и тишина
- Hibernate 4.X.X+ использует JBoss Logging!
- JBoss Logging - это ещё один адаптер!
- Приоритеты: JBoss—>log4j—>SLF4J—>JUL/JDK



Нежданчик #2: Hibernate

- Проблема: надо подебажить запросы Hibernate после перехода на новую версию
- `show_sql=true` —> и тишина
- Добавил “in-team-ной связи” для логеров Hibernate —> и тишина
- Hibernate 4.X.X+ использует JBoss Logging!
- JBoss Logging - это ещё один адаптер!
- Приоритеты: JBoss—>log4j—>SLF4J—>JUL/JDK
- `-Dorg.jboss.logging.provider=slf4j` (Не делайте так!)



Нежданчик #3: временная недетерминированность

Нежданчик #3: временная недетерминированность

```
<fileNamePattern>${user.home}/archive/microServiceK/
%d{yyyy-MM-dd}/serviceK-%d{yyyy-MM-dd_HH-mm,aux}.log.gz
</fileNamePattern>
```

Нежданчик #3: временная недетерминированность

```
<fileNamePattern>${user.home}/archive/microServiceK/
%d{yyyy-MM-dd}/serviceK-%d{yyyy-MM-dd_HH-mm,aux}.log.gz
</fileNamePattern>
```

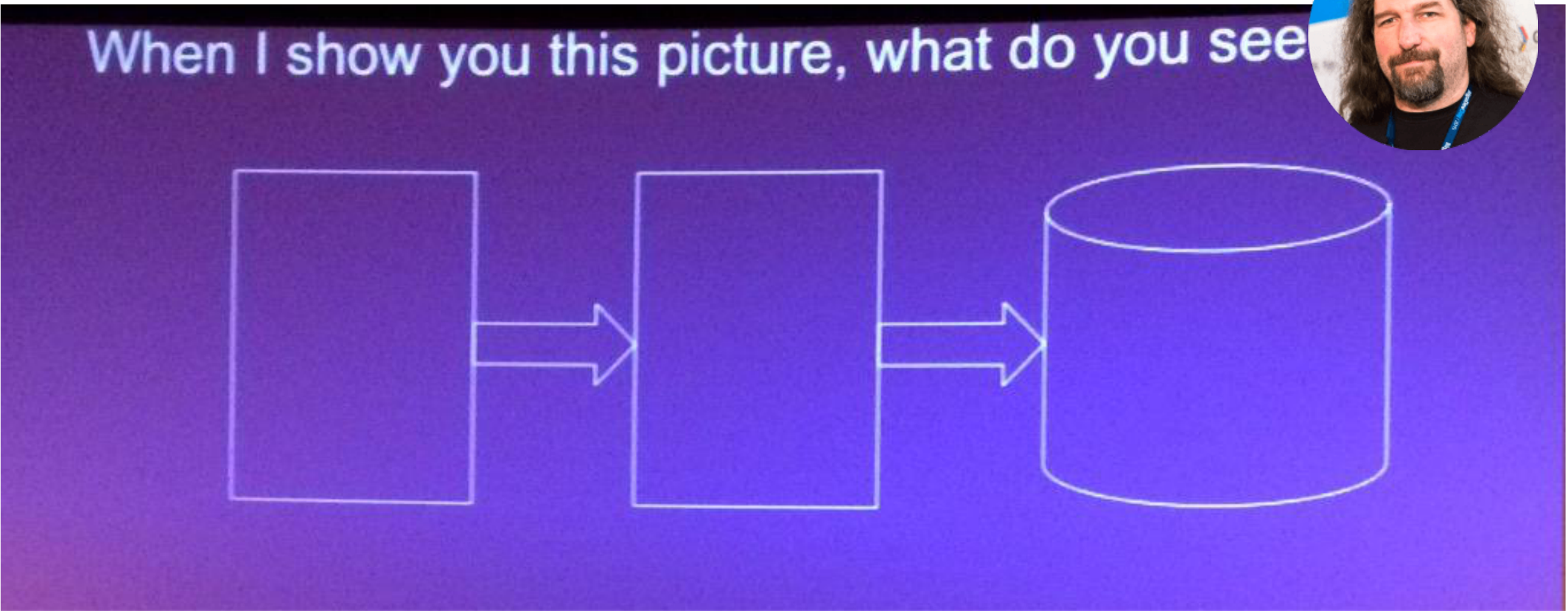
- В результате архивы появлялись, но в 00:13, 00:21, 07:14, …
  Но почему?

Нежданчик #3: временная недетерминированность

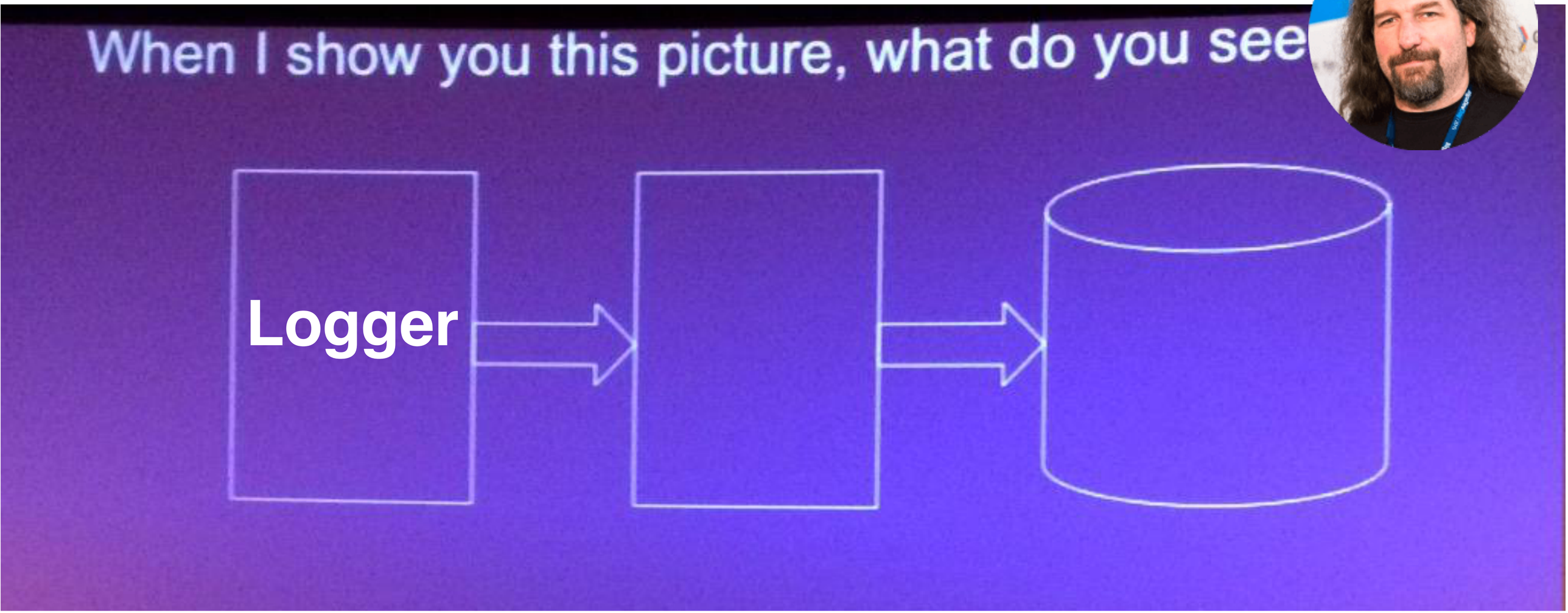
```
<fileNamePattern>${user.home}/archive/microServiceK/
%d{yyyy-MM-dd}/serviceK-%d{yyyy-MM-dd_HH-mm,aux}.log.gz
</fileNamePattern>
```

- В результате архивы появлялись, но в 00:13, 00:21, 07:14, …
  Но почему?
- Архивирование, ротация и подчистка файлов не выполняются по расписанию!

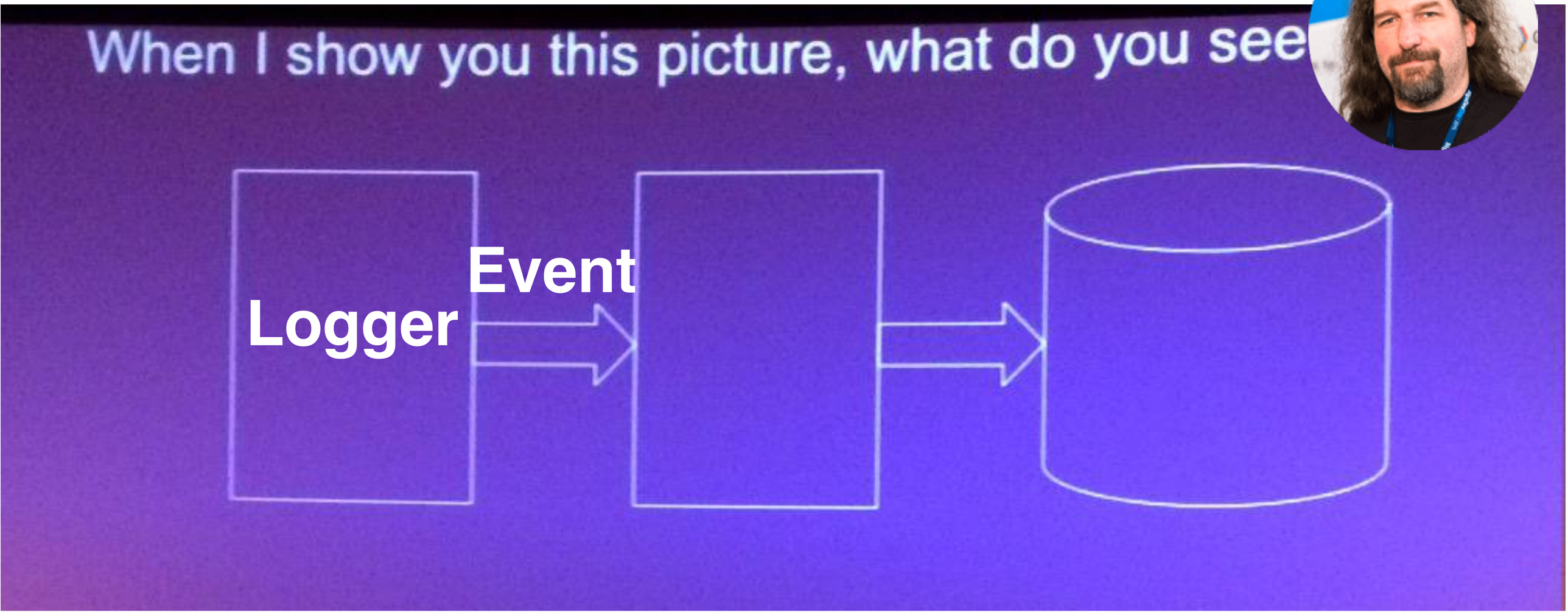
  А тогда как?



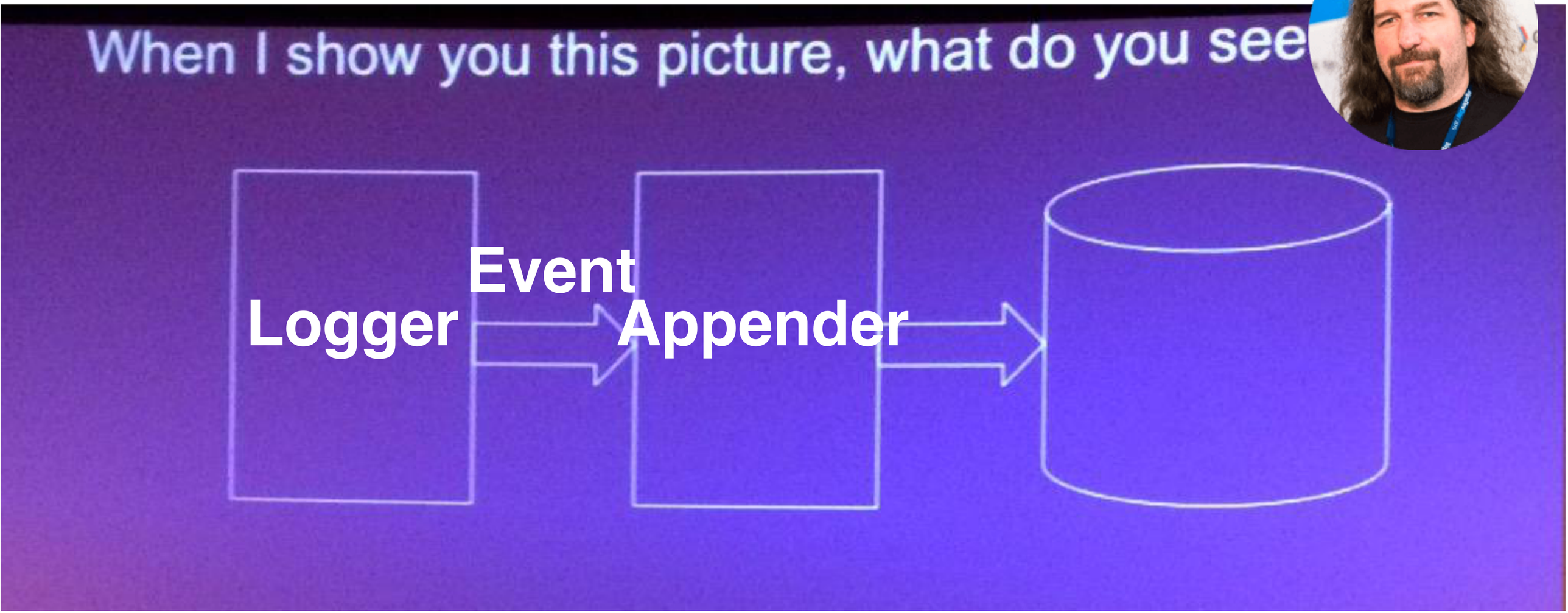
Универсальная Программная Архитектурная Диаграмма (УПАД)
by @tedneward



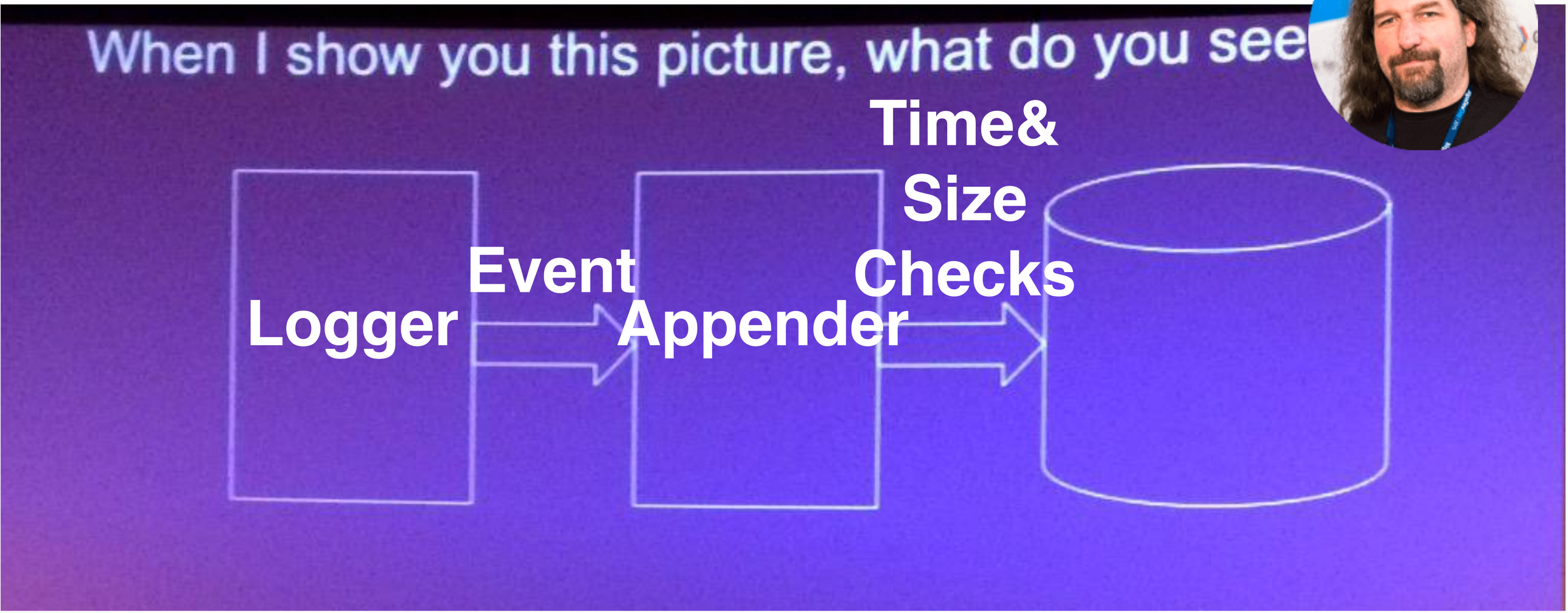
Универсальная Программная Архитектурная Диаграмма (УПАД)
by @tedneward



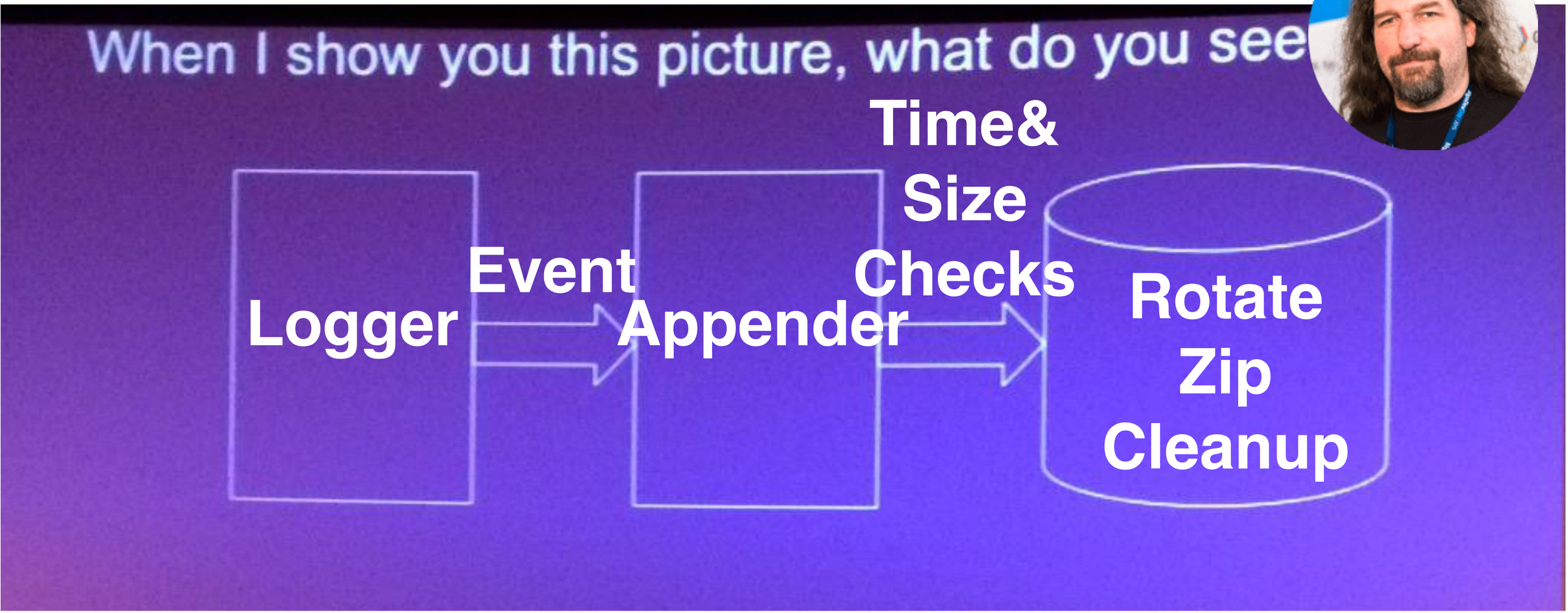
Универсальная Программная Архитектурная Диаграмма (УПАД)
by @tedneward



Универсальная Программная Архитектурная Диаграмма (УПАД)
by @tedneward



Универсальная Программная Архитектурная Диаграмма (УПАД)
by @tedneward



Универсальная Программная Архитектурная Диаграмма (УПАД)
by @tedneward

Mapped Diagnostic Context (MDC)

Mapped Diagnostic Context (MDC)

- `MDC.put("personalId", personalId);`

Mapped Diagnostic Context (MDC)

- `MDC.put("personalId", personalId);`
- `<pattern>%X{personalId} - %m%n</pattern>`

Mapped Diagnostic Context (MDC)

- `MDC.put("personalId", personalId);`
- `<pattern>%X{personalId} - %m%n</pattern>`
- `MDC.remove(“personalId");`

Страна советов

Страна советов

- Отдельный файл для всех внешних коммуникаций

Страна советов

- Отдельный файл для всех внешних коммуникаций
- AOP / BCI - для массового логирования методов

Страна советов

- Отдельный файл для всех внешних коммуникаций
- AOP / BCI - для массового логирования методов
- Мониторинг:

```
log.warn(Alerts.YELLOW("Bad thing's happened", exc));
log.error(Alerts.RED("Very bad thing's happened", exc));
```

Страна советов

- Отдельный файл для всех внешних коммуникаций
- AOP / BCI - для массового логирования методов
- Мониторинг:
```
log.warn(Alerts.YELLOW("Bad thing's happened", exc));
log.error(Alerts.RED("Very bad thing's happened", exc));
```
- Heart-beat thread - `log.info("I'm alive!");`

Страна советов

- Отдельный файл для всех внешних коммуникаций
- AOP / BCI - для массового логирования методов
- Мониторинг:

```
log.warn(Alerts.YELLOW("Bad thing's happened", exc));
log.error(Alerts.RED("Very bad thing's happened", exc));
```

- Heart-beat thread - `log.info("I'm alive!");`
- Решение для просмотра и скачивания логов …

Логирование в микросервисах и кластере

Логирование в микросервисах и кластере

- Микросервисы: специализированные взаимодействующие компоненты

Логирование в микросервисах и кластере

- Микросервисы: специализированные взаимодействующие компоненты
- Кластер: параллелизм+отказоустойчивость+высокая доступность

Логирование в микросервисах и кластере

- Микросервисы: специализированные взаимодействующие компоненты
- Кластер: параллелизм+отказоустойчивость+высокая доступность

    - Logstash
      Elastic Search
      Kibana

Логирование в микросервисах и кластере

- Микросервисы: специализированные взаимодействующие компоненты
- Кластер: параллелизм+отказоустойчивость+высокая доступность

- Logstash
  Elastic Search
  Kibana
- Appender
  может писать в
  базу данных,
  JMS и т. д.

Логирование в микросервисах и кластере

- Микросервисы: специализированные взаимодействующие компоненты
- Кластер: параллелизм+отказоустойчивость+высокая доступность

- Logstash
  Elastic Search
  Kibana
- Appender
  может писать в
  базу данных,
  JMS и т. д.
- Splunk или
  Logmatic

Логирование в облаке

Логирование в облаке

AWS, OpenShift, Jelastic, Microsoft Azure, и т д

Логирование в облаке

AWS, OpenShift, Jelastic, Microsoft Azure, и т д

Убедитесь, что апгрейд или выключение виртуалки,
а также изменения инфраструктуры не убьют Ваши
драгоценные логи и архивы!

Логирование в Docker-е

Логирование в Docker-е

Монтируйте папку с логами через
VOLUME, -v вне контейнера на хосте

API development

API development

- Не используйте JUL/JDK просто потому что он уже есть в Java!

API development

- Не используйте JUL/JDK просто потому что он уже есть в Java!
- Не используйте JUL/JDK просто потому что он уже есть в Java!!

API development

- Не используйте JUL/JDK просто потому что он уже есть в Java!
- Не используйте JUL/JDK просто потому что он уже есть в Java!!
- Используйте связку SLF4J+“рабочая лошадка”, например, logback

API development

- Не используйте JUL/JDK просто потому что он уже есть в Java!
- Не используйте JUL/JDK просто потому что он уже есть в Java!!
- Используйте связку SLF4J+“рабочая лошадка”, например, logback
- Подумайте, а нужна ли библиотека “рабочей лошадки” как транзитивная зависимость вашим клиентам?

API development

- Не используйте JUL/JDK просто потому что он уже есть в Java!
- Не используйте JUL/JDK просто потому что он уже есть в Java!!
- Используйте связку SLF4J+“рабочая лошадка”, например, logback
- Подумайте, а нужна ли библиотека “рабочей лошадки” как транзитивная зависимость вашим клиентам?
- Паттерн “волшебная системная пропертя” а-ля `verbose`

Сколько проблем в коде?

```
private static final Logger log =
LoggerFactory.getLogger(SomeClass.class);     (1)

…

void withdraw(String login, Long amount) {    (2)

 if (log.isDebugEnabled()) {                  (3)

  log.debug(“starting withdrawing”);          (4)

 }

 log.info(“login ”+login+“,amount ”+amount); (5)

}
```

Сколько проблем в коде?

```
private static final Logger LOG =
LoggerFactory.getLogger(SomeClass.class);     (1)

…

void withdraw(String login, Long amount) {    (2)

 if (log.isDebugEnabled()) {                          (3)

  LOG.debug(“starting withdrawing”);                  (4)

 }

 LOG.info(“login {},amount {}”,login,amount);(5)

}
```

Безопасность в логировании: уязвимости

Безопасность в логировании: уязвимости

- Логирование персональных данных, а также ценной коммерческой информации

Безопасность в логировании: уязвимости

- Логирование персональных данных, а также ценной коммерческой информации
- Log Injection (OWASP): перевод строки, другие атаки …

Безопасность в логировании: уязвимости

- Логирование персональных данных, а также ценной коммерческой информации
- Log Injection (OWASP): перевод строки, другие атаки …
- Javascript Injection

Безопасность в логировании: уязвимости

- Логирование персональных данных, а также ценной коммерческой информации
- Log Injection (OWASP): перевод строки, другие атаки …
- Javascript Injection
- Атаки на мониторинговые тулы, базы данных и т.д.

Безопасность в логировании: уязвимости

- Логирование персональных данных, а также ценной коммерческой информации
- Log Injection (OWASP): перевод строки, другие атаки …
- Javascript Injection
- Атаки на мониторинговые тулы, базы данных и т.д.
- Открытое логирование паролей, логинов, урлов и хостов

Безопасность в логировании: решения

Безопасность в логировании: решения

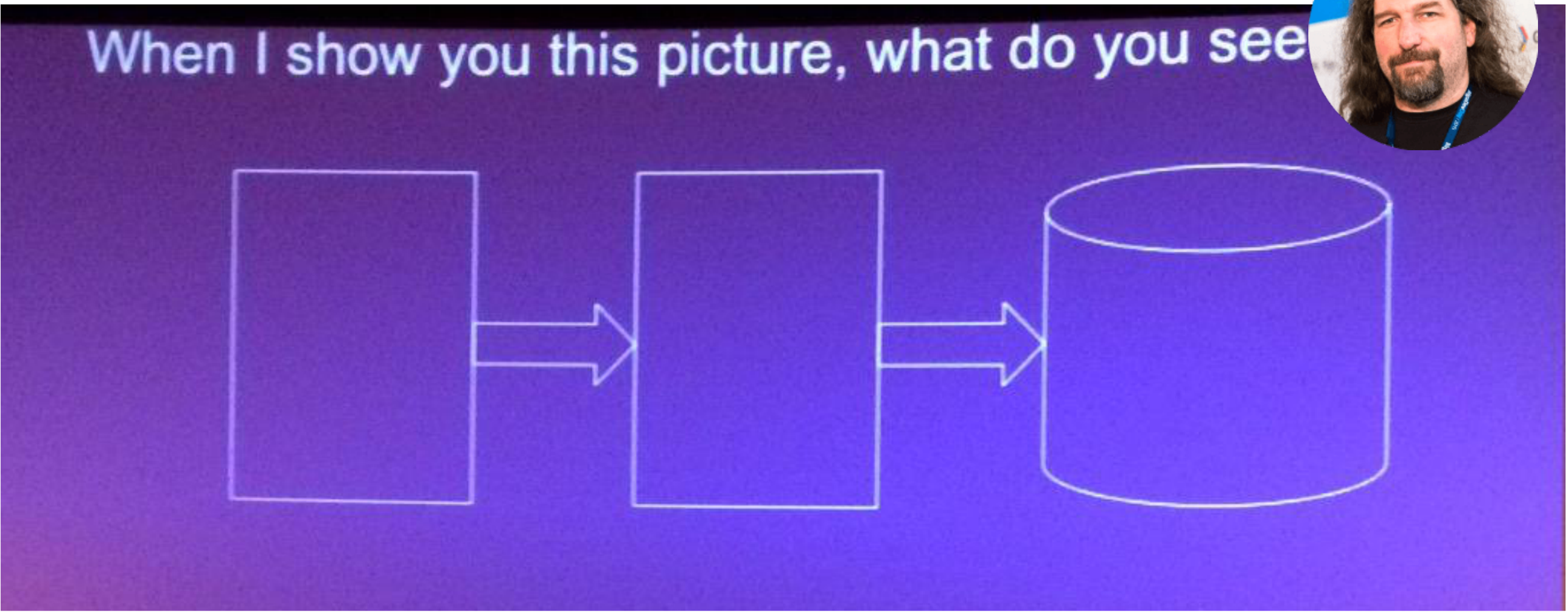
- veracode.com рекомендует логировать только константные выражения из предопределенного пула строк

Безопасность в логировании: решения

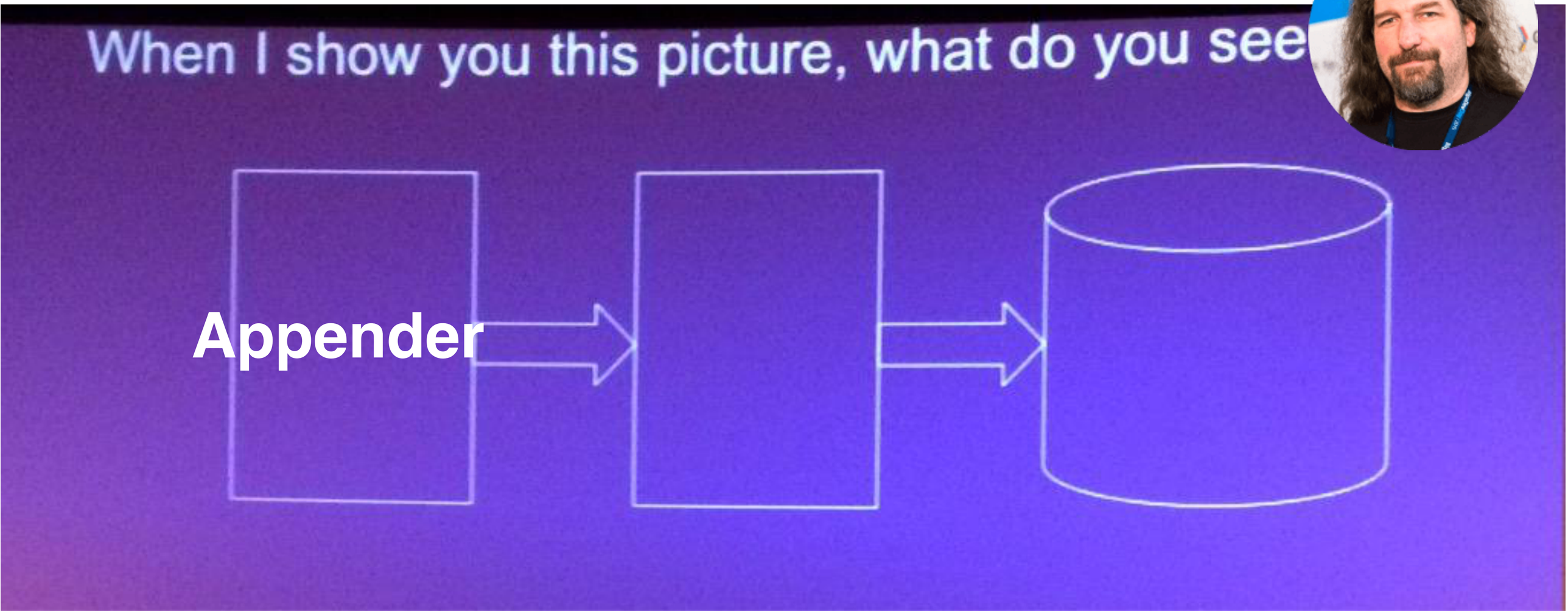
- veracode.com рекомендует логировать только константные выражения из предопределенного пула строк
- https://github.com/javabeanz/owasp-security-logging (экранирование, шифрация, маскирование, фильтрация, аудит)

Безопасность в логировании: решения

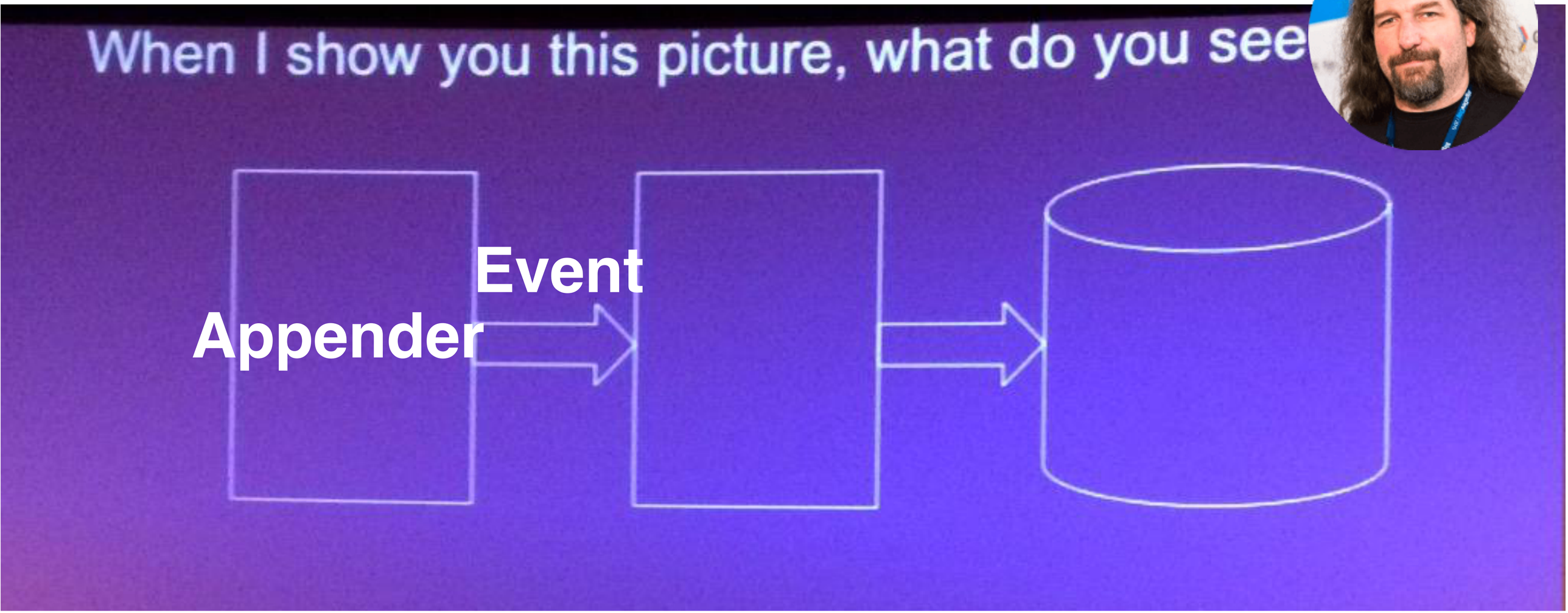
- veracode.com рекомендует логировать только константные выражения из предопределенного пула строк
- https://github.com/javabeanz/owasp-security-logging (экранирование, шифрация, маскирование, фильтрация, аудит)
- Фильтры и конвертеры в logback



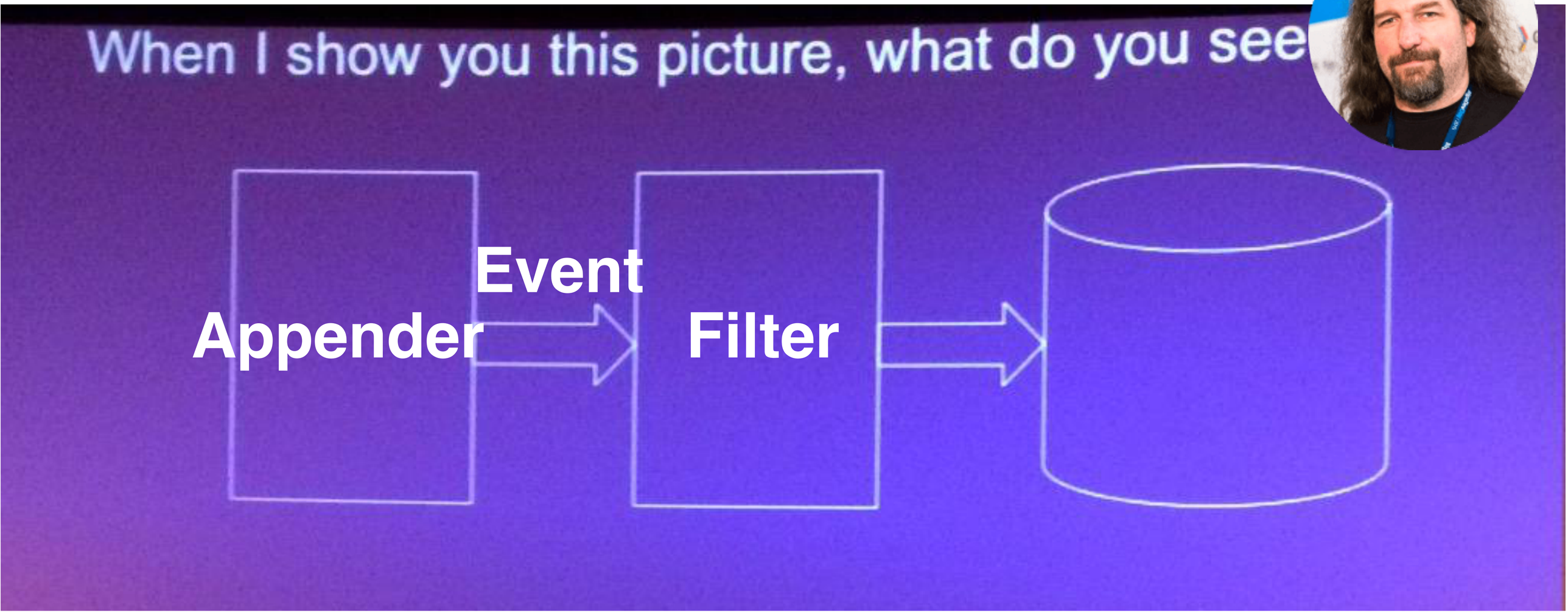
Универсальная Программная Архитектурная Диаграмма (УПАД)
by @tedneward



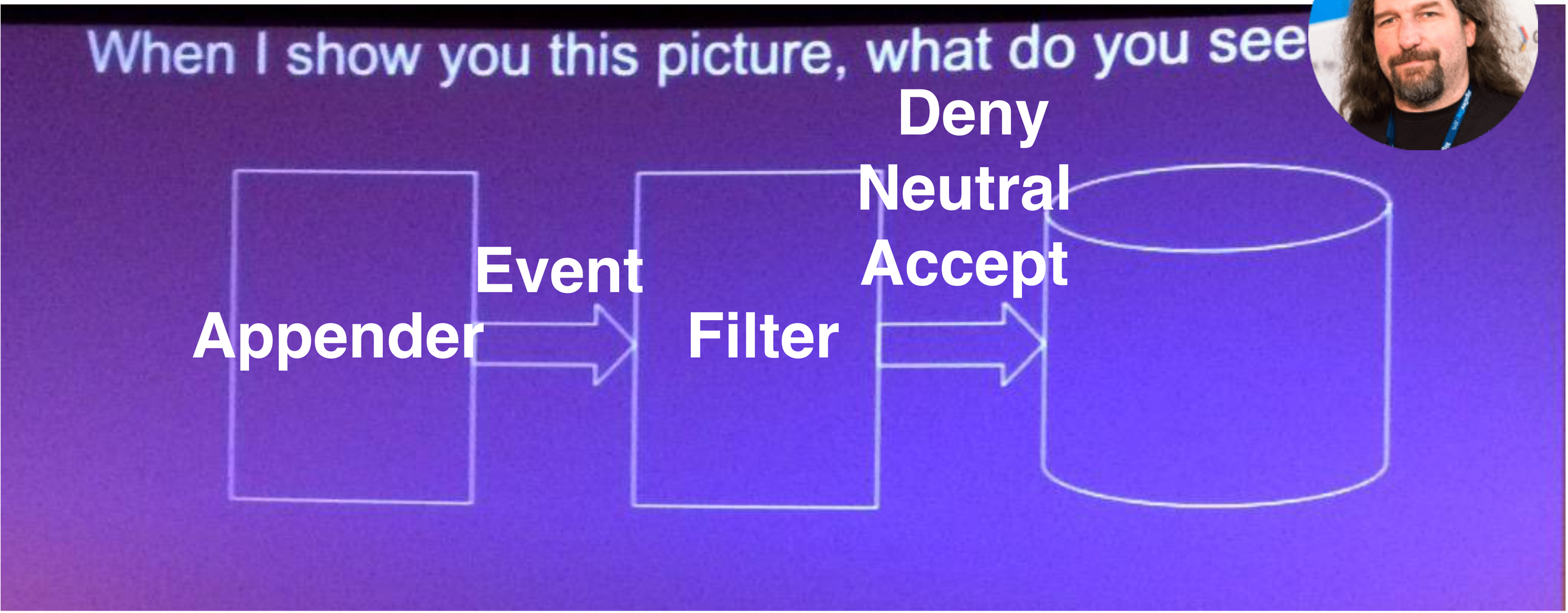
Универсальная Программная Архитектурная Диаграмма (УПАД)
by @tedneward



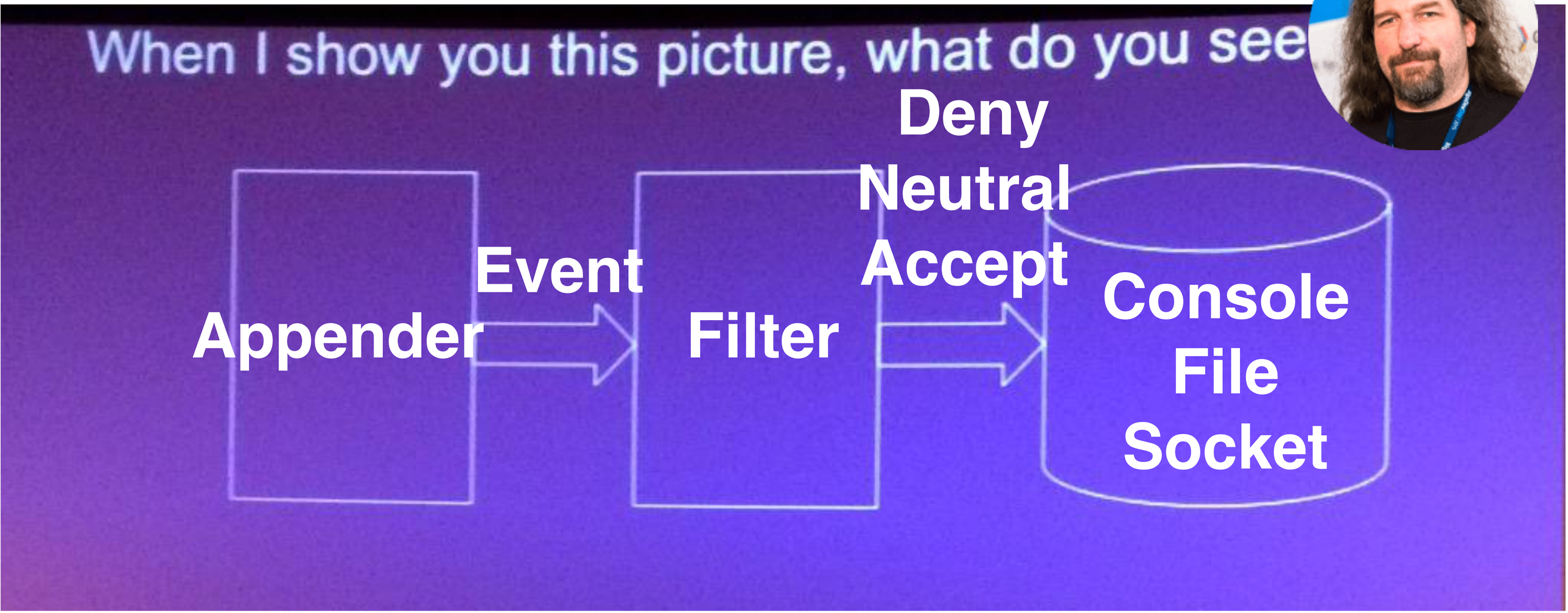
Универсальная Программная Архитектурная Диаграмма (УПАД)
by @tedneward



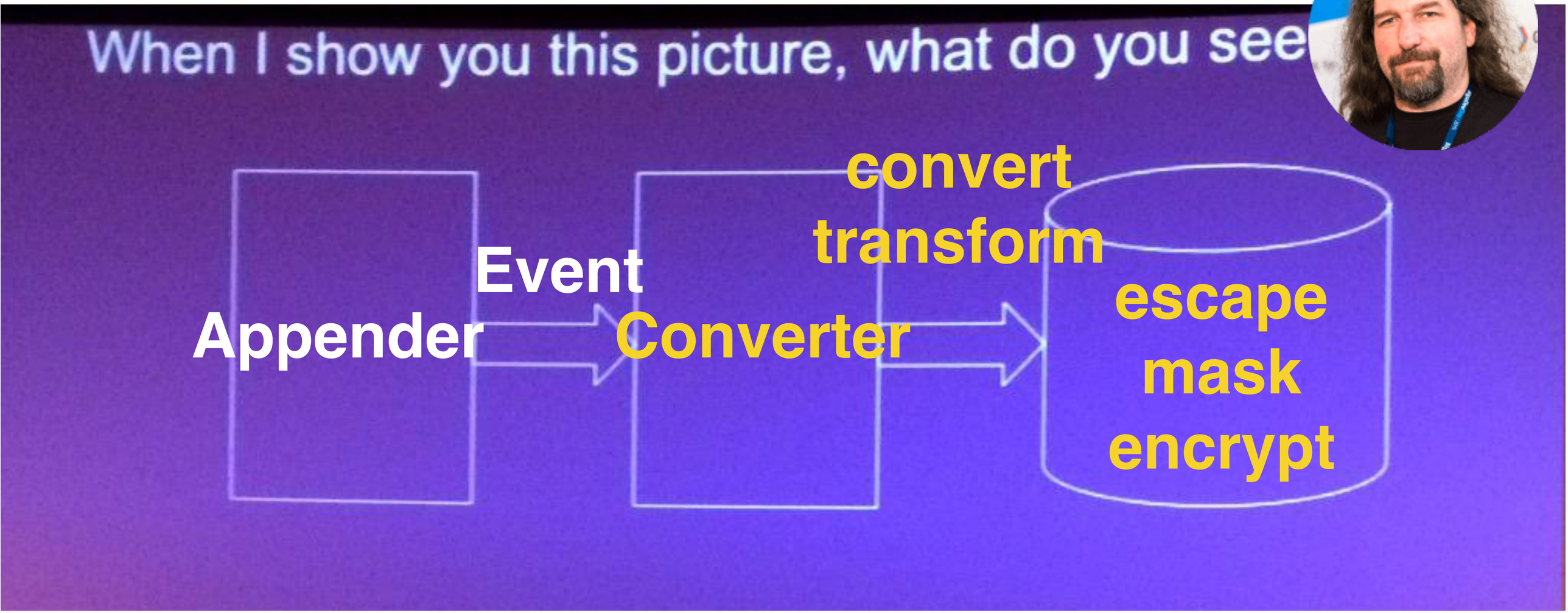
Универсальная Программная Архитектурная Диаграмма (УПАД)
by @tedneward



Универсальная Программная Архитектурная Диаграмма (УПАД)
by @tedneward

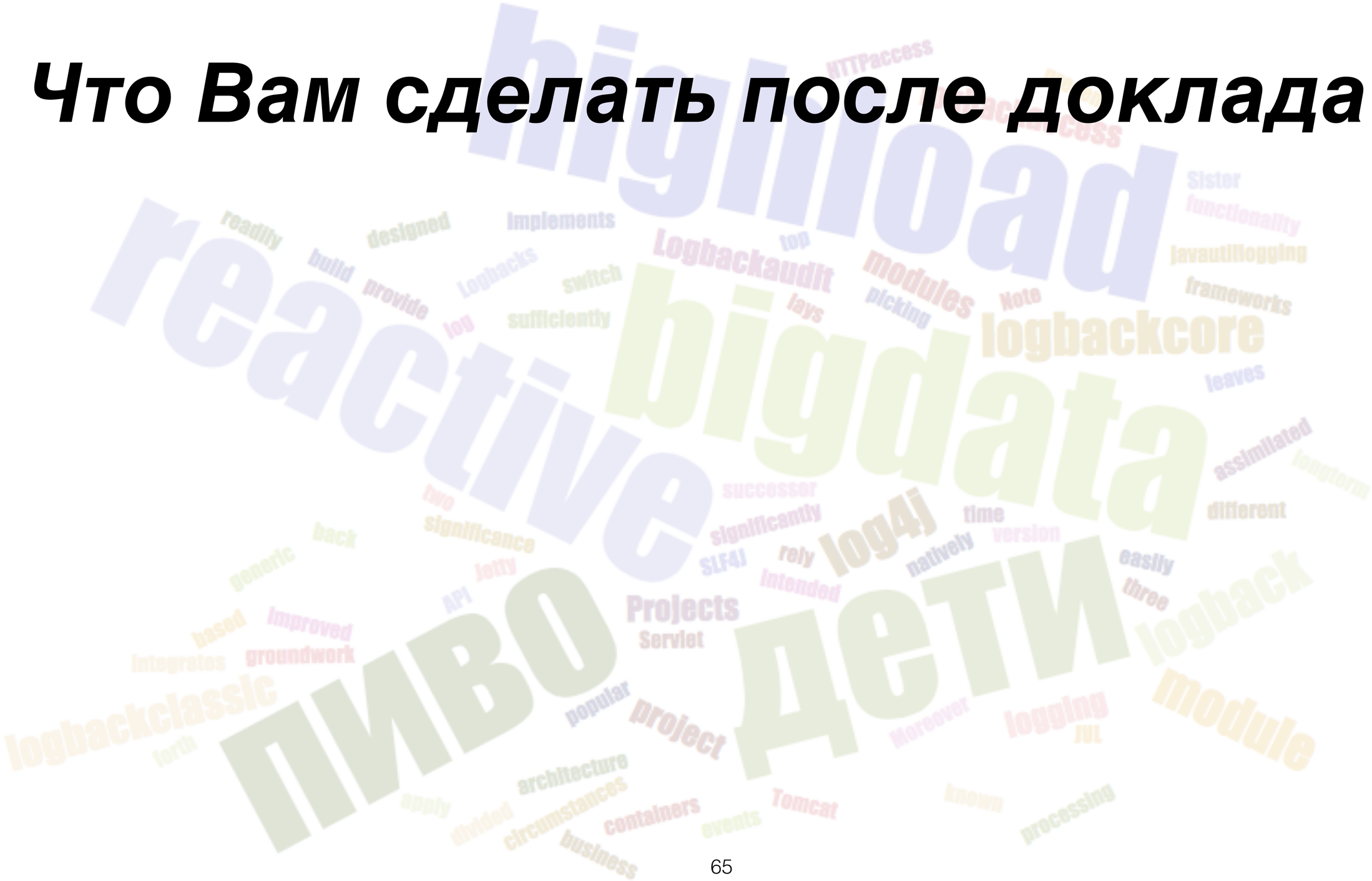


Универсальная Программная Архитектурная Диаграмма (УПАД)
by @tedneward



Универсальная Программная Архитектурная Диаграмма (УПАД)
by @tedneward

Что Вам сделать после доклада?



Что Вам сделать после доклада?

- ***Очистить Ваш класспас от месива библиотек логирования, подтягивающихся транзитивно и бесконтрольно Вашими maven, gradle, sbt и ivy с ними***

Что Вам сделать после доклада?

- Очистить Ваш класспас от месива библиотек логирования, подтягивающихся транзитивно и бесконтрольно Вашими maven, gradle, sbt и ivy с ними
- Если Вы пишете API, подумать о класспасе ваших клиентов

Что Вам сделать после доклада?

- Очистить Ваш класспас от месива библиотек логирования, подтягивающихся транзитивно и бесконтрольно Вашими maven, gradle, sbt и ivy с ними
- Если Вы пишете API, подумать о класспасе ваших клиентов
- Досконально изучить вашу связку адаптер+”рабочая лошадка”

Что Вам сделать после доклада?

- Очистить Ваш класспас от месива библиотек логирования, подтягивающихся транзитивно и бесконтрольно Вашими maven, gradle, sbt и ivy с ними
- Если Вы пишете API, подумать о класспасе ваших клиентов
- Досконально изучить вашу связку адаптер+”рабочая лошадка”
- Придумать как удобно доставать и просматривать логи с продакшена

Что Вам сделать после доклада?

- Очистить Ваш класспас от месива библиотек логирования, подтягивающихся транзитивно и бесконтрольно Вашими maven, gradle, sbt и ivy с ними
- Если Вы пишете API, подумать о класспасе ваших клиентов
- Досконально изучить вашу связку адаптер+”рабочая лошадка”
- Придумать как удобно доставать и просматривать логи с продакшена
- Принять во внимание текущее и возможное будущее окружение: один сервер, кластер, виртуалка, облако, Docker

Что Вам сделать после доклада?

- Очистить Ваш класспас от месива библиотек логирования, подтягивающихся транзитивно и бесконтрольно Вашими maven, gradle, sbt и ivy с ними
- Если Вы пишете API, подумать о класспасе ваших клиентов
- Досконально изучить вашу связку адаптер+”рабочая лошадка”
- Придумать как удобно доставать и просматривать логи с продакшена
- Принять во внимание текущее и возможное будущее окружение: один сервер, кластер, виртуалка, облако, Docker
- Проверить Ваш код на уязвимости по безопасности, докрутить экранирование строк, маскирование или криптование подозрительных аргументов

Что Вам сделать после доклада?

- Очистить Ваш класспас от месива библиотек логирования, подтягивающихся транзитивно и бесконтрольно Вашими maven, gradle, sbt и ivy с ними
- Если Вы пишете API, подумать о класспасе ваших клиентов
- Досконально изучить вашу связку адаптер+”рабочая лошадка”
- Придумать как удобно доставать и просматривать логи с продакшена
- Принять во внимание текущее и возможное будущее окружение: один сервер, кластер, виртуалка, облако, Docker
- Проверить Ваш код на уязвимости по безопасности, докрутить экранирование строк, маскирование или криптование подозрительных аргументов
- Ежедневно использовать Универсальную Программную Архитектурную Диаграмму (УПАД) Теда Ньюворда, чтобы прослыть крутым архитектором

Москва, 2016

Спасибо!

Владимир Красильщик, Luxoft, Санкт-Петербург
vladimir.krasilschik@gmail.com
https://ru.linkedin.com/in/vladimirkrasilschik
http://drozd4j.blogspot.ru