



WARGAMING.NET

LET'S BATTLE

ОЧЕРЕДНОЙ СКУЧНЫЙ ДОКЛАД ПРО ЛОГГИРОВАНИЕ

Стас Рудаков

Эпиграф

Невозможно объять необъятное.

Козьма Прутков

Про что будем говорить?

Про что будем говорить?

Про логгирование, там же было написано :)

Про что будем говорить?

Про логгирование, там же было написано :)

А именно:

- ▶ зачем;
- ▶ как;
- ▶ проблемы;
- ▶ очевидные пути решения;
- ▶ неочевидные пути решения;
- ▶ open source инструменты.

Что вообще значит “записать в лог”?

Что вообще значит “записать в лог”?

- ▶ сформировать сообщение

Что вообще значит “записать в лог”?

- ▶ сформировать сообщение
- ▶ по некому событию

Что вообще значит “записать в лог”?

- ▶ сформировать сообщение
- ▶ по некому событию
- ▶ и, если это сообщение нас вообще интересует,

Что вообще значит “записать в лог”?

- ▶ сформировать сообщение
- ▶ по некому событию
- ▶ и, если это сообщение нас вообще интересует,
- ▶ записать или отправить его куда-то,

Что вообще значит “записать в лог”?

- ▶ сформировать сообщение
- ▶ по некому событию
- ▶ и, если это сообщение нас вообще интересует,
- ▶ записать или отправить его куда-то,
- ▶ куда имеют доступ заинтересованные лица.

Зачем

- ▶ Debug
- ▶ Разбор инцидентов

Зачем

- ▶ Debug
- ▶ Разбор инцидентов
- ▶ Сбор статистики
- ▶ Средне- и долгосрочный мониторинг
- ▶ Инструмент для службы поддержки пользователей
- ▶ Аналитика

```
import logging
```

Писать логи очень просто

```
1 import logging
2
3 logging.basicConfig(
4     format='[%(asctime)s] [%(levelname)s] [%(name)s] %(message)s',
5     stream=sys.stdout,
6     level=logging.INFO
7 )
8
9 logger = logging.getLogger("meetup.python.minsk")
10 logger.info("Stas has just started the talk on logging")
```

Писать логи очень просто

```
1 import logging
2
3 logging.basicConfig(
4     format='[%(asctime)s][%(levelname)s][%(name)s] %(message)s',
5     stream=sys.stdout,
6     level=logging.INFO
7 )
8
9 logger = logging.getLogger("meetup.python.minsk")
10 logger.info("Stas has just started the talk on logging")
```

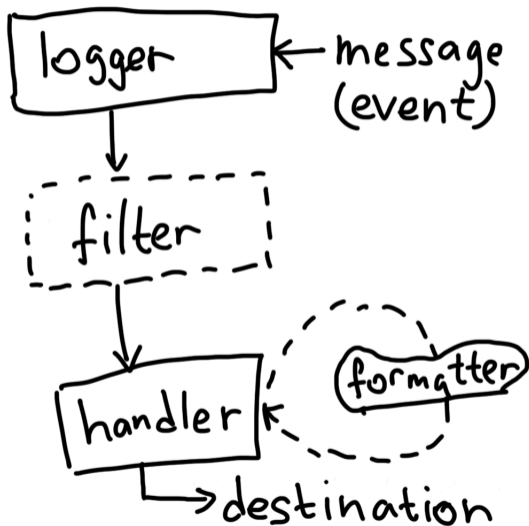
```
1 [2014-01-31 06:23:27,904][INFO][meetup.python.minsk] Stas has
   just started the talk on logging
```


logging: архитектура на пальцах

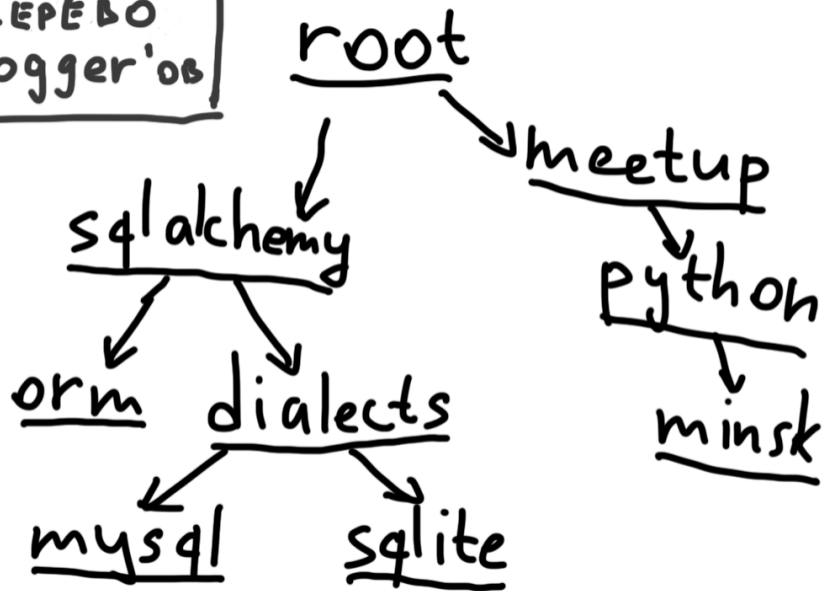
- ▶ сформировать сообщение
- ▶ по некому событию

- ▶ и, если это сообщение нас вообще интересует,

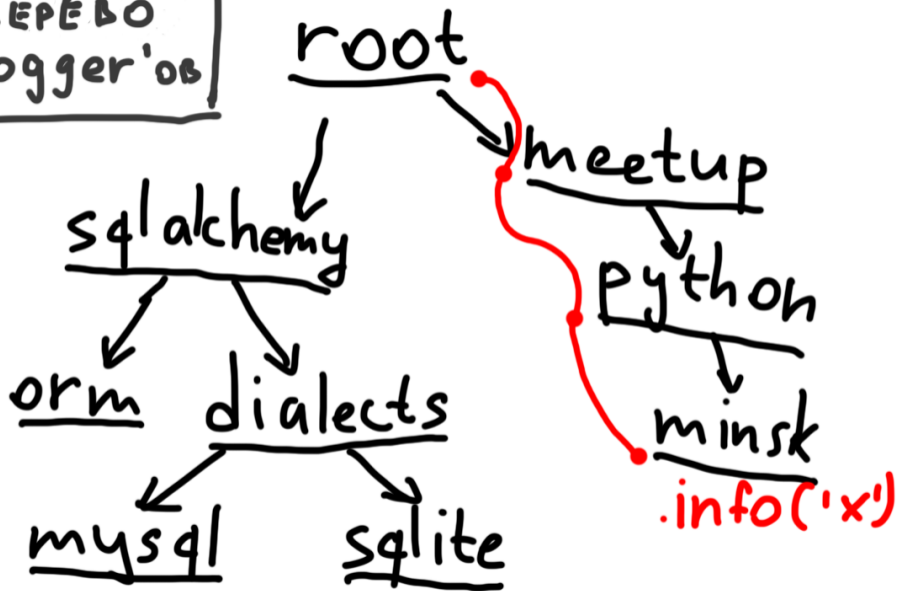
- ▶ записать или отправить его куда-то,
- ▶ куда имеют доступ заинтересованные лица.



ДЕРЕВО
logger'ov

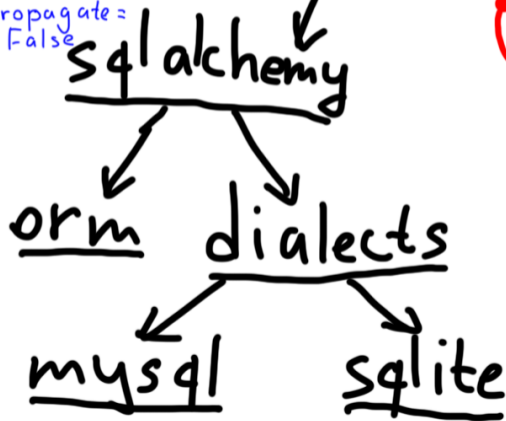


ДЕРЕВО
logger'ов



ДЕРЕВО
logger'ов

propagate =
False



root

meetup

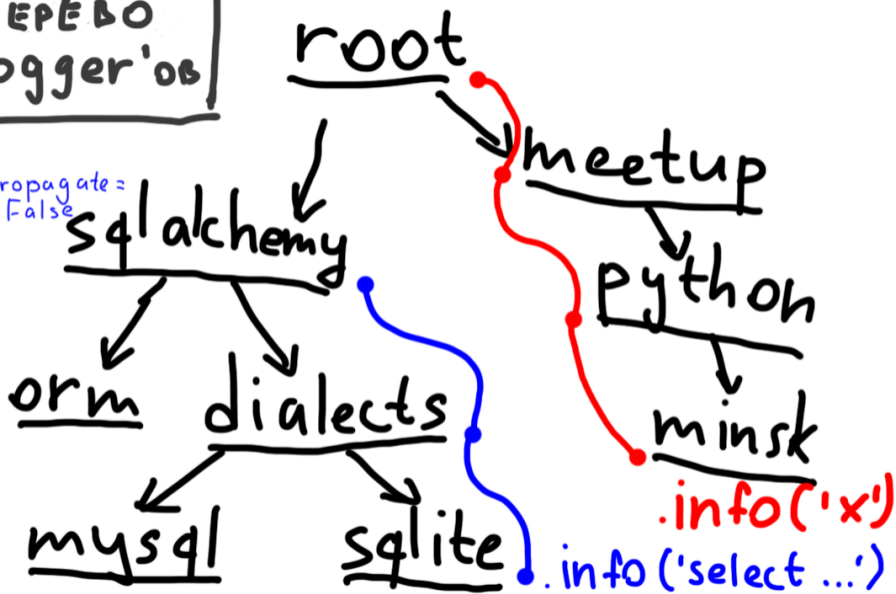
python

minsk

.info('x')

ДЕРЕВО
logger'ов

propagate =
False



ДЕРЕВО
logger'ов

root

ВСЕ УЗЛЫ-
СИГНАТОРЫ!

propagate=
False

sqlalchemy

orm

dialects

mysql

sqlite

meetup

python

minsk

.info('x')

.info('select ...')

Конфигурация

```
1 CFG = {
2     'formatters': {
3         'simple': {
4             'format': '%(asctime)s %(
5                 message)s'}},},
6     'filters': {
7         'pwd': {
8             '()': 'proj.PwdFilter',
9             'foo': 'bar'},},},
10    'handlers': {
11        'console':{
12            'level': 'DEBUG',
13            'class': 'logging.'
14                'StreamHandler',
15            'formatter': 'simple'}},
```

```
16        'mail_adm': {
17            'level': 'ERROR',
18            'class': 'proj.'
19                'AdminEmailHandler',
20            'filters': ['pwd']}},},
21    'loggers': {
22        '': {
23            'handlers': ['console'],
24            'level': 'INFO',
25            'filters': ['pwd']},
26        'django.request': {
27            'handlers': ['mail_adm'],
28            'level': 'ERROR',
29            'propagate': False},},},
30    logging.config.dictConfig(CFG)
```

tips n tricks: уровни логгирования

tips n tricks: уровни логгирования

▶ CRITICAL = 50

```
logger.critical(u'проблема, после которой приложение не может  
восстановиться без постороннего вмешательства')
```

tips n tricks: уровни логгирования

- ▶ `CRITICAL = 50`

`logger.critical(u'проблема, после которой приложение не может восстановиться без постороннего вмешательства')`

- ▶ `ERROR = 40`

`logger.error(u'проблема, из-за которой приложение работает не в штатном режиме')`

tips n tricks: уровни логгирования

- ▶ CRITICAL = 50

`logger.critical(u'проблема, после которой приложение не может восстановиться без постороннего вмешательства')`

- ▶ ERROR = 40

`logger.error(u'проблема, из-за которой приложение работает не в штатном режиме')`

- ▶ WARNING = 30 *# уровень по умолчанию*

`logger.warning(u'проблема, которая не мешает работе приложения, но на которую стоит обратить внимание')`

tips n tricks: уровни логгирования

- ▶ CRITICAL = 50

`logger.critical(u'проблема, после которой приложение не может восстановиться без постороннего вмешательства')`

- ▶ ERROR = 40

`logger.error(u'проблема, из-за которой приложение работает не в штатном режиме')`

- ▶ WARNING = 30 *# уровень по умолчанию*

`logger.warning(u'проблема, которая не мешает работе приложения, но на которую стоит обратить внимание')`

- ▶ INFO = 20

`logger.info(u'обычная запись в лог')`

tips n tricks: уровни логгирования

- ▶ CRITICAL = 50

`logger.critical(u'проблема, после которой приложение не может восстановиться без постороннего вмешательства')`

- ▶ ERROR = 40

`logger.error(u'проблема, из-за которой приложение работает не в штатном режиме')`

- ▶ WARNING = 30 *# уровень по умолчанию*

`logger.warning(u'проблема, которая не мешает работе приложения, но на которую стоит обратить внимание')`

- ▶ INFO = 20

`logger.info(u'обычная запись в лог')`

- ▶ DEBUG = 10

`logger.debug(u'логи, которые помогли нам при отладке приложения и которые вряд ли пригодятся в будущем')`

tips n tricks: traceback

```
1 try:
2     do_something_unsafe()
3 except Exception as e:
4     logger.exception('Unexpected exception')
5     raise
```

tips n tricks: traceback

```
1 try:
2     do_something_unsafe()
3 except Exception as e:
4     logger.exception('Unexpected exception')
5     raise
```

В лог отправится:

```
1 [2014-01-29 23:35:04,393][examples][ERROR] Unexpected exception
2 Traceback (most recent call last):
3   File "stacktrace1.py", line 22, in <module>
4     do_something_unsafe()
5   File "stacktrace1.py", line 18, in do_something_unsafe
6     down_rabbit_hole()
7   File "stacktrace1.py", line 17, in down_rabbit_hole
8     sender = message.sender
9 AttributeError: 'NoneType' object has no attribute 'sender'
```

tips n tricks: traceback

```
1 try:
2     validate_data(data)
3 except ValidationError as e:
4     logger.exception('Validation failed')
```

А в лог запишется

```
1 [2014-01-29 08:33:53,439][examples][ERROR] Validation failed
2 Traceback (most recent call last):
3   File "stacktrace2.py", line 27, in <module>
4     validate_data(data)
5   File "stacktrace2.py", line 22, in validate_data
6     check_values(data)
7   File "stacktrace2.py", line 21, in check_values
8     raise ValidationError('security key is missing')
9 ValidationError: security key is missing
```


tips n tricks: traceback

```
1 try:
2     validate_data(data)
3 except ValidationError as e:
4     logger.exception('Validation failed')
```

А в лог запишется

```
1 [2014-01-29 08:33:53,439][examples][ERROR] Validation failed
2 Traceback (most recent call last):
3   File "stacktrace2.py", line 27, in <module>
4     validate_data(data)
5   File "stacktrace2.py", line 22, in validate_data
6     check_values(data)
7   File "stacktrace2.py", line 21, in check_values
8     raise ValidationError('security key is missing')
9 ValidationError: security key is missing
```

Но разве уровень **ERROR** нам подходит?

tips n tricks: traceback

```
1 try:
2     validate_data(data)
3 except ValidationError as e:
4     logger.warning('Validation failed', exc_info=True)
```

Получаем

```
1 [2014-01-29 08:33:53,439][examples][WARNING] Validation failed
2 Traceback (most recent call last):
3   File "stacktrace2.py", line 27, in <module>
4     validate_data(data)
5   File "stacktrace2.py", line 22, in validate_data
6     check_values(data)
7   File "stacktrace2.py", line 21, in check_values
8     raise ValidationError('security key is missing')
9 ValidationError: security key is missing
```

tips n tricks: “ленивое” форматирование

```
1 logger.info(  
2     'Request id=%s message=%r' % (100500, {'event': 'meetup'})  
3 )
```

tips n tricks: “ленивое” форматирование

```
1 logger.info(  
2     'Request id=%s message=%r' % (100500, {'event': 'meetup'})  
3 )
```

Зачем форматировать строку, если мы не знаем, попадет ли она в лог?

tips n tricks: “ленивое” форматирование

```
1 logger.info(  
2     'Request id=%s message=%r' % (100500, {'event': 'meetup'})  
3 )
```

Зачем форматировать строку, если мы не знаем, попадет ли она в лог?

```
1 logger.info(  
2     'Request id=%s message=%r', 100500, {'event': 'meetup'}  
3 )
```

tips n tricks: “ленивое” форматирование

```
1 logger.info(  
2     'Request id=%s message=%r' % (100500, {'event': 'meetup'})  
3 )
```

Зачем форматировать строку, если мы не знаем, попадет ли она в лог?

```
1 logger.info(  
2     'Request id=%s message=%r', 100500, {'event': 'meetup'}  
3 )
```

args** работают, *kwargs** не работают

tips n tricks: ротация
в один прекрасный день...

tips n tricks: ротация

в один прекрасный день. . . no space left on device

tips n tricks: ротация

в один прекрасный день... no space left on device

```
from logging.handlers import *
```

```
1 RotatingFileHandler(  
2     filename, mode='a',  
3     maxBytes=0, backupCount=0,  
4     encoding=None, delay=False  
5 )
```

```
1 $ ls logs  
2 meetup.log  
3 meetup.log.1  
4 meetup.log.2  
5 meetup.log.3
```

tips n tricks: ротация

в один прекрасный день... no space left on device

```
from logging.handlers import *
```

```
1 RotatingFileHandler(  
2     filename, mode='a',  
3     maxBytes=0, backupCount=0,  
4     encoding=None, delay=False  
5 )
```

```
1 $ ls logs  
2 meetup.log  
3 meetup.log.1  
4 meetup.log.2  
5 meetup.log.3
```

```
1 TimedRotatingFileHandler(  
2     filename,  
3     when='h', interval=1,  
4     backupCount=0, encoding=None,  
5     delay=False, utc=False)
```

```
1 $ ls logs  
2 meetup.log  
3 meetup.log.2014-01-31_17  
4 meetup.log.2014-01-31_18  
5 meetup.log.2014-01-31_19
```

tips n tricks: ротация

в один прекрасный день... no space left on device

```
from logging.handlers import *
```

```
1 RotatingFileHandler(  
2     filename, mode='a',  
3     maxBytes=0, backupCount=0,  
4     encoding=None, delay=False  
5 )
```

```
1 TimedRotatingFileHandler(  
2     filename,  
3     when='h', interval=1,  
4     backupCount=0, encoding=None,  
5     delay=False, utc=False)
```

```
1 $ ls logs  
2 meetup.log  
3 meetup.log.1  
4 meetup.log.2  
5 meetup.log.3
```

```
1 $ ls logs  
2 meetup.log  
3 meetup.log.2014-01-31_17  
4 meetup.log.2014-01-31_18  
5 meetup.log.2014-01-31_19
```

Если варианты выше не подходят, можно использовать утилиту logrotate и

```
1 WatchedFileHandler(fname, mode='a', encoding=None, delay=False)
```

Sentry
getsentry.com

Sentry - это...

- ▶ open source агрегатор сообщений об ошибках,
- ▶ написанный на Python + Django + Celery,
- ▶ с клиентами для многих языков и платформ.

The screenshot displays the Sentry web interface. At the top, there are navigation tabs for 'Sentry', 'Events', and 'Team'. The main content area shows a list of error events with the following details:

- Event 1:** 10 occurrences, 'sentry.tasks.fetch_source in fetch_javascript_source', 8 hours ago, root. Status: Resolved (green checkmark).
- Event 2:** 4 occurrences, 'urllib2 in http_error_default', 8 hours ago, sentry.tasks.fetch_source.fetch_javascript_source. Status: Resolved (green checkmark).
- Event 3:** 44 occurrences, 'django.db.backends.postgresql_psycopg2.base in execute', 8 hours ago, root. Status: Resolved (green checkmark).
- Event 4:** 15 occurrences, 'amqp.method_framing in read_method', 2 days ago, celery.worker.consumer. Status: Resolved (green checkmark).
- Event 5:** 3 occurrences, 'django.db.backends.postgresql_psycopg2.base in execute', 2 days ago, root. Status: Unresolved (red circle).
- Event 6:** 3 occurrences, 'django.db.backends.postgresql_psycopg2.base in execute', 2 days ago, root. Status: Unresolved (red circle).

The right sidebar contains several filter sections:

- DAILY USAGE:** 10 events.
- BOOKMARKS:** All Events, Only Bookmarks.
- STATUS:** Unresolved (x).
- LOGGER:** Select a logger.
- SERVER NAME:** Select a server name.
- URL:** Search for a url.
- USER EMAIL:** Search for a user email.

Sentry Events Team Sentry ckj

Sentry (Internal) TypeError: NoneType cannot be coerced to bar Search query or message id

1.5k sentry.management.commands.send_fake_data in exception
TypeError: NoneType cannot be coerced to bar
26 minutes ago foo.bar

The system marked this event as a regression — just now
chris@getsentry.com marked this event as resolved — 2013-01-28

← Newer Event Event at Feb. 16, 2013, 2:23 p.m. UTC Older Event →

Exception Stacktrace User Additional Data Versions

Exception

Types: TypeError
Value: NoneType cannot be coerced to bar
Location: sentry/management/commands/send_fake_data.py in exception , line 39

Stacktrace (most recent call last) Raw

sentry/management/commands/send_fake_data.py in exception

```
34. # return client.capture('Query', query=queries.next(), engine=engine.next(), time_spent=duration, data={'logger': loggers.next(), 'site': 'sql'})
35.
36. def exception(client):
37.     timestamp = datetime.datetime.utcnow() - datetime.timedelta(seconds=timestamps.next())
38.     try:
39.         raise exceptions.next()
40.     except Exception:
41.         email = emails.next()
42.         return client.capture('Exception', data={
43.             'logger': loggers.next(),
44.             'site': 'web',
```

Aggregate

Tags

Similar Events

AGGREGATE DETAILS

Status: Unresolved
First Seen: Jan. 19, 2013
Reopened At: 16 hours ago
Last Seen: 24 minutes ago

This event is publicly visible. [Link](#)

ACTIONS

Remove Event Data

LEVEL

1.0k error

LOGGER

1.0k foo.bar

SERVER NAME

1.0k Chris-MacBook-Pro.Local

SITE

1.0k web

Sentry: как подключить

```
1 from raven.handlers.logging import SentryHandler
2 handler = SentryHandler('http://public:secret@example.com/1',
3                          level=logging.WARNING)
```

Sentry: как подключить

```
1 from raven.handlers.logging import SentryHandler
2 handler = SentryHandler('http://public:secret@example.com/1',
3                          level=logging.WARNING)
```

```
4 logger = logging.getLogger('coffeemaker')
5 logger.addHandler(handler)
```


Sentry: как подключить

```
1 from raven.handlers.logging import SentryHandler
2 handler = SentryHandler('http://public:secret@example.com/1',
3                          level=logging.WARNING)
```

```
4 logger = logging.getLogger('coffeemaker')
5 logger.addHandler(handler)
```

```
6 logger.warning('We are running out of milk')
7 logger.error('Not enough milk for a cappuccino')
```

Sentry: как подключить

```
1 from raven.handlers.logging import SentryHandler
2 handler = SentryHandler('http://public:secret@example.com/1',
3                          level=logging.WARNING)
```

```
4 logger = logging.getLogger('coffeemaker')
5 logger.addHandler(handler)
```

```
6 logger.warning('We are running out of milk')
7 logger.error('Not enough milk for a cappuccino')
```

HTTP? Серьезно?

А теперь давайте вспомним, что мы с этими логами собирались делать

- ▶ Debug
- ▶ Разбор инцидентов

А теперь давайте вспомним, что мы с этими логами собирались делать

- ▶ Debug
- ▶ Разбор инцидентов
- ▶ Сбор статистики
- ▶ Средне- и долгосрочный мониторинг
- ▶ Инструмент для службы поддержки пользователей
- ▶ Аналитика

Назад в будущее

Современные вызовы:

- ▶ хотим убедиться, что логи не были модифицированы злоумышленником после взлома;
- ▶ переходим от программ к сложным распределенным системам — хочется иметь логгирование уровня всей системы, а не отдельных приложений;
- ▶ быстрый поиск;
- ▶ метрики.

elasticsearch
elasticsearch.org

elasticsearch - это...

- ▶ распределенный
- ▶ полнотекстовый
- ▶ поисковый движок

elasticsearch - это...

- ▶ распределенный
- ▶ полнотекстовый
- ▶ поисковый движок
- ▶ с RESTful web-интерфейсом,
- ▶ с документами без схемы
- ▶ и возможностями агрегации документов.

elasticsearch - это...

- ▶ распределенный
 - ▶ полнотекстовый
 - ▶ поисковый движок
 - ▶ с RESTful web-интерфейсом,
 - ▶ с документами без схемы
 - ▶ и возможностями агрегации документов.
-
- ▶ Написан на Java
 - ▶ с использованием библиотеки Apache Lucene.
 - ▶ Используется в Mozilla, GitHub и многих других компаниях (если вам нужно убедить своего менеджера).

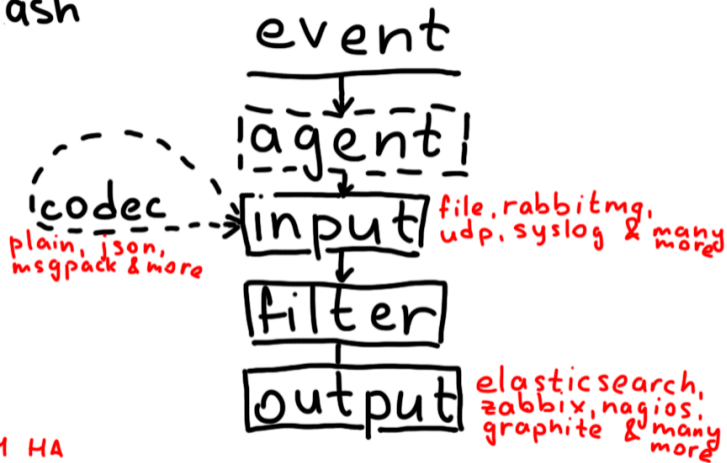
elasticsearch - это...

- ▶ распределенный
 - ▶ полнотекстовый
 - ▶ поисковый движок
 - ▶ с RESTful web-интерфейсом,
 - ▶ с документами без схемы
 - ▶ и возможностями агрегации документов.
-
- ▶ Написан на Java
 - ▶ с использованием библиотеки Apache Lucene.
 - ▶ Используется в Mozilla, GitHub и многих других компаниях (если вам нужно убедить своего менеджера).

Но как связать логгирование и elasticsearch?

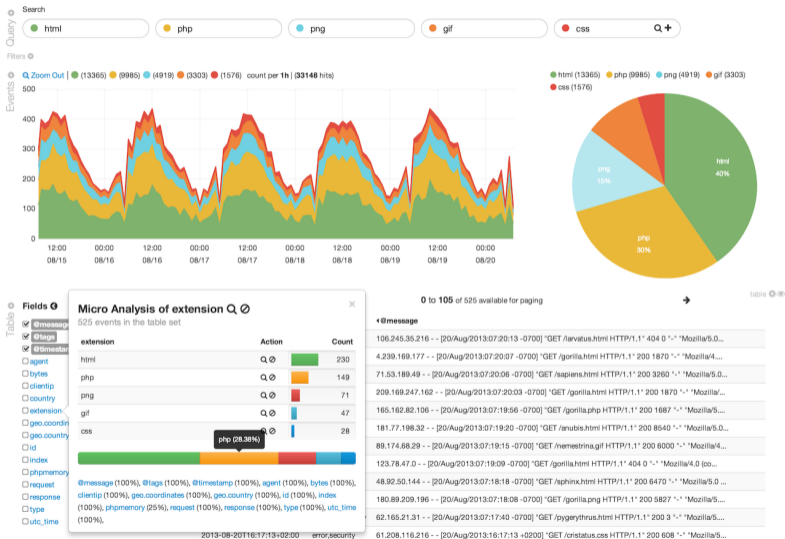
logstash
logstash.net

АРХИТЕКТУРА logstash



НАПИСАН НА
IRuby

logstash + elasticsearch + kibana = nice web UI



structlog — последний ингредиент

```
1 >>> from structlog import get_logger
2 >>> log = get_logger()
3 >>> log = log.bind(user='anonymous', some_key=23)
4 >>> log = log.bind(user='hynek', another_key=42)
5 >>> log.info('user.logged_in', happy=True)
6 some_key=23 user='hynek' another_key=42 happy=True event='user.
  logged_in'
```

<http://www.structlog.org/>

Что мы не объяли?

- ▶ Syslog Protocol
- ▶ Логи в реляционной базе данных
- ▶ journald
- ▶ logplex, fluentd
- ▶ Агенты для сбора логов
- ▶ Что, собственно, писать в логи.

Выводы

- ▶ Батарейка logging вполне подходит для 95% случаев¹.

¹По статистике, не стоит доверять 75% статистики

Выводы

- ▶ Батарейка logging вполне подходит для 95% случаев¹.
- ▶ Архитектуры систем логгирования похожи как на микро, так и на макро уровне.

¹По статистике, не стоит доверять 75% статистики

Выводы

- ▶ Батарейка logging вполне подходит для 95% случаев¹.
- ▶ Архитектуры систем логгирования похожи как на микро, так и на макро уровне.
- ▶ Open source инструменты помогают выжать из логов очень много полезных свойств.

¹По статистике, не стоит доверять 75% статистики

Выводы

- ▶ Батарейка logging вполне подходит для 95% случаев¹.
- ▶ Архитектуры систем логгирования похожи как на микро, так и на макро уровне.
- ▶ Open source инструменты помогают выжать из логов очень много полезных свойств.
- ▶ Я вас всех порядочно утомил.

¹По статистике, не стоит доверять 75% статистики

СПАСИБО ЗА ВНИМАНИЕ. ВОПРОСЫ?

Стас Рудаков

<mailto:stas@garage22.net>

[https://raw.githubusercontent.com/nott/talks/logging.pdf](https://raw.githubusercontent.com/nott/talks/master/logging.pdf)

<http://docs.python.org/3/library/logging.html>

<http://www.structlog.org/>

<https://www.getsentry.com/>

<http://logstash.net/>