Зачем вам нужна **Clojure**

Никита Прокопов
tonsky.livejournal.com

Кто мы?

AboutEcho.com

Риалтаймовый веб:
комментарии, потоки, запросы...

Калифорния/Ульяновск/Новосибирск

Кто мы?


ESPN


ADVANCE
DIGITAL


The Washington Post


NEW YORK


UNIVERSAL
UNIVERSAL MUSIC GROUP


SCRIPPS


NBC NEWS


usa
network

Кто мы?

Erlang, OCaml, C:

~15 программистов

45 000 rps в пике

450 серверов

Кто мы?

Javascript:

8 программистов
500 кб кода
300 кб тестов
~200M запросов на CDN в день

Кто мы?

Clojure:

1..5 программистов
9 месяцев
продакшн уже ч/з 3 месяца

Clojure никто не знал

Как случилась Clojure?

Новый проект, нет кода для переиспользования

Twitter Storm

Двухнедельный proof of concept Erlang vs Clojure

Почему Clojure?

Явные сильные стороны:

- Скорость разработки
- Производительность
- Concurrency
- Высокая культура разработки

Что такое Clojure?

JVM-based язык

Общего назначения

Современный LISP

Функциональный

Прагматичный

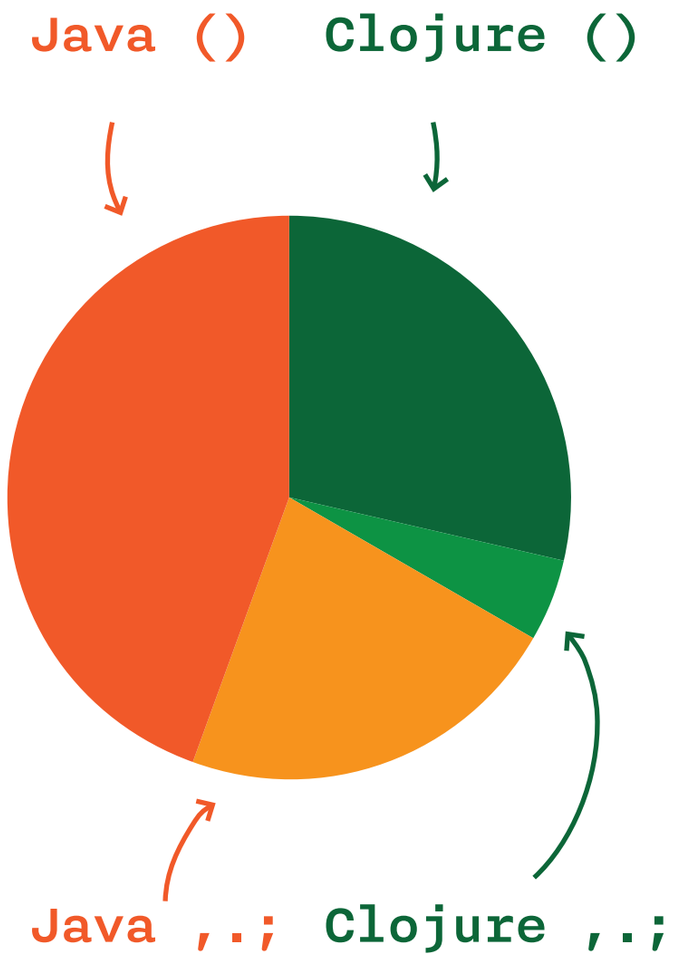
Скобочки? Фу?

```
fun(x, y)

obj.method()

if (x < y) {
  foo();
} else {
  bar();
}

Map<String, Integer>
m = new HashMap<>() {{
  put("x", 1);
  put("y", 2);
}}
```




```
(fun x y)

(.method obj)

(if (< x y)
  (foo)
  (bar))

(let [m {:x 1
         :y 2}])
```

Сильные стороны Clojure

Concurrency

Иммутабельные персистентные структуры данных

Явная модель изменений, высокоуровневые примитивы, транзакционная память

Проще, предсказуемее, компонуемее

Сильные стороны Clojure

Обработка данных

Дата-центричная философия

Удобная стандартная библиотека

ФП — композиция кусков

Extensible Data Notation

Сильные стороны Clojure
Язык общего назначения

Небольшой, выразительный

Компактный синтаксис

Динамический полиморфизм, без ООП

Компонуемые абстракции,
открытость, расширяемость

Кодогенерация ⟨порой⟩

Сильные стороны Clojure

```
public class StringUtils {
    public static boolean isBlank(String str) {
        int strLen;
        if (str == null || (strLen = str.length()) == 0) {
            return true;
        }
        for (int i = 0; i < strLen; i++) {
            if ((Character.isWhitespace(str.charAt(i)) == false)) {
                return false;
            }
        }
        return true;
    }
}
```

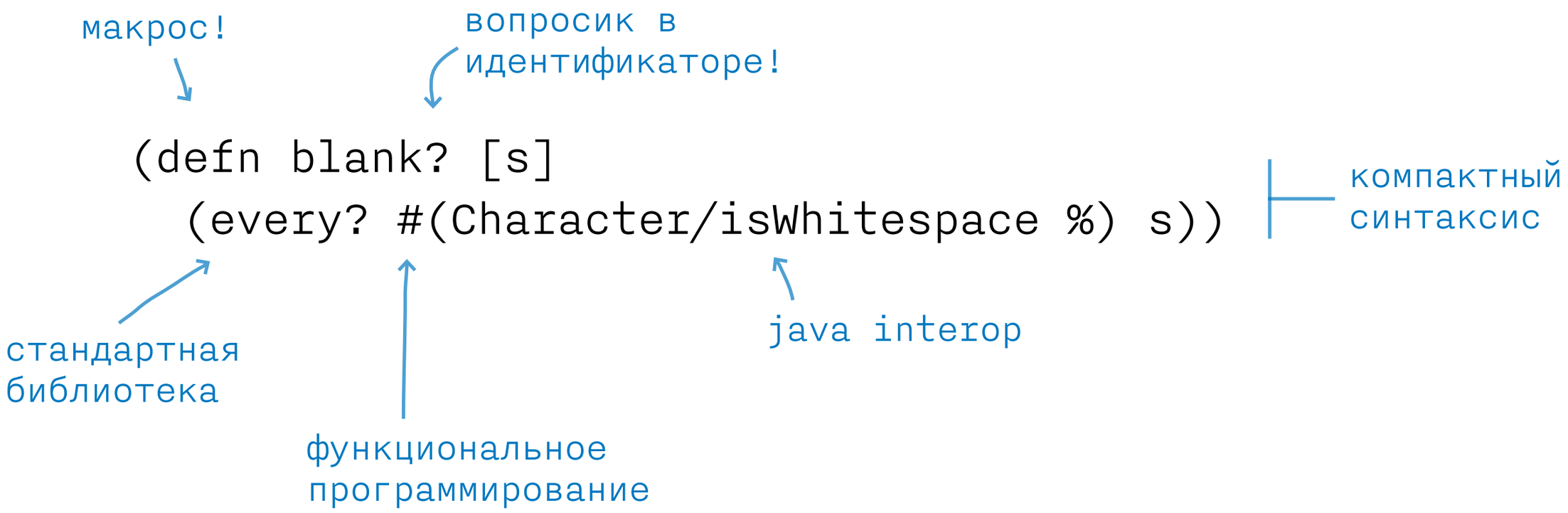
[](){} ×28

Сильные стороны Clojure

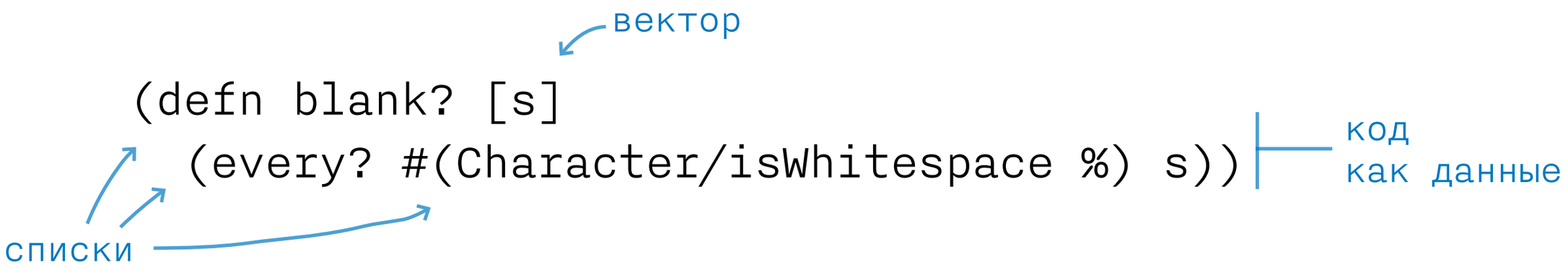
```
(defn blank? [s]
  (every? #(Character/isWhitespace %) s))
```

[](){} ×8

Анатомия Clojure



Анатомия Clojure



Clojure как Java

Уважает платформу

Прямой interop в Java

Генерация .class =>
дергаем Clojure из Java-проекта

Lein работает прямо
с Maven-репозиториями

Clojure как Java

Писать Java на Clojure проще, чем на самой Java

Разгоняется до скорости Java

Clojure как Python

Динамическая компиляция

Быстрое прототипирование

Быстрее, чем в Питоне ⟨REPL⟩

Компактнее и лаконичнее,
чем в Питоне ⟨ФП⟩

Потенциал для оптимизации

Clojure как Bash*

Удобный перочинный нож

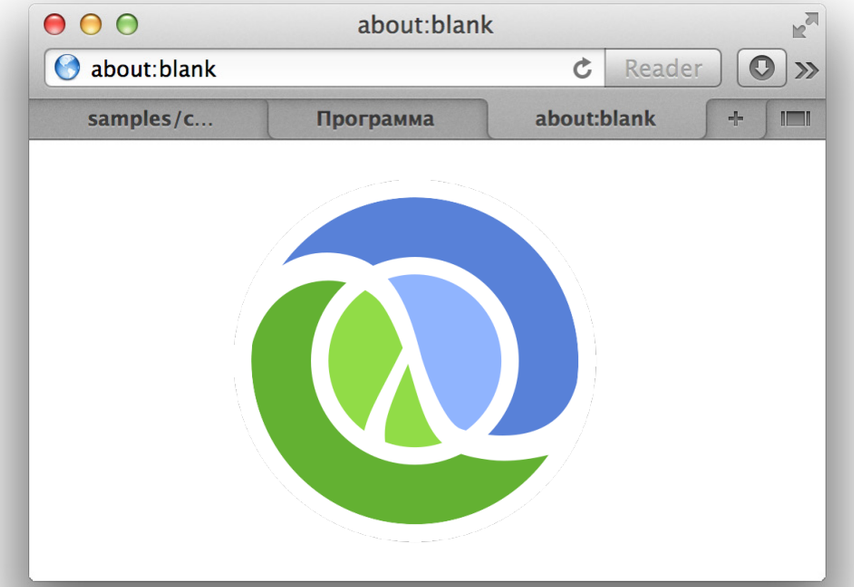
Особенно если запущен REPL

Особенно хорошо — разбор,
анализ и трансформация данных

* Обобщенный

Clojure как JavaScript
ClojureScript

Clojure в браузере



ClojureScript

Зачем?

Писать web и не сойти с ума

Трудно масштабировать JS проект:

- нужна хорошая архитектура
- нужна жесткая дисциплина
- нужна согласованность по тулзам

Проблема не в синтаксисе

JavaScript

Типы
Структуры данных
Зависимости
Неймспейсы
Полиморфизм
Типовые операции

JavaScript

- Типы
- Структуры данных
- Зависимости
- Неймспейсы
- Полиморфизм
- Типовые операции

CoffeeScript?

~~Типы~~

~~Структуры данных~~

~~Зависимости~~

~~Неймспейсы~~

~~Полиморфизм~~

~~Типовые операции~~

Меньше кнопок нажимать

ClojureScript

Правильная семантическая модель:

- модулей
- данных
- вычислений
- состояния

ClojureScript

Стандартные решения стандартных проблем

```
реализаций import:     0
ООП-фреймворков:       0
альтерн. синтаксисов: 0
альтерн. коллекций:    0
monad tutorials:       0
```

ClojureScript
Компиляция

Google Closure-совместим

Сжимает

Генерирует кроссбраузерный код

Оптимизирует лучше человека*

Ужасная отладка**

* Потенциально
** Уже почти нет

ClojureScript

Компиляция

Не надо ждать стандартов	2 года
Не надо ждать смерти IE	7..10 лет
Не надо ждать библиотек	0..∞

Destucturing	Real data structures
Compact function syntax	Array comprehensions
Vararg parameters	Maps with non-string keys
Modules/imports	For-of
Lexical scope	Multiline strings

Уже сегодня, уже сейчас

ClojureScript
Довесок

Иммутабельность
Персистентные структуры данных
ФП
Макросы
Строгая типизация
Протоколы

...в браузере! уже сегодня!

ClojureScript

1,5 года

Достаточно быстр и легок

Стабильное API

Production-ready

Clojure сообщество,
или зачем интересоваться Clojure

Clojure сделает из вас более лучшего инженера

Много хорошего кода

Правильные ценности

Правильная философия

Clojure сообщество,
или зачем интересоваться Clojure

Площадка для экспериментов

Крайне полезные лекции!

Мало сил — приходится искать хорошие решения

Clojure сообщество
Доклады Rich Hickey про CS

Hammock-driven development

Are we there yet?

Simple made easy

TBD ⟨так вот называется,
о дизайне систем⟩

Clojure сообщество

Доклады Rich Hickey про Clojure

Clojure Concurrency

RH Unveils ClojureScript

RH on Datomic

Clojure в жизни

Clojure в Echo

Опрошено 5 разработчиков

1..5 мес. работы на Clojure

До этого никто Clojure не знал

Clojure в Echo

Насколько сложно читать?

- несложно, дело привычки
- читать сложнее Erlang-а
- Python ⟨2,3⟩
  Java, Erlang ⟨4⟩
  Clojure⟨6,7⟩
- примерно Ruby ⟨без Rails⟩
- очень зависит от автора

Clojure в Echo
Насколько сложно писать?

- очень легко
- легче, чем в ООП языках
- меньше кода, только суть
- упирается в понимание кода
- Проблем с отладкой нет
  ⟨отладочная печать она и в Африке отладочная печать⟩

Clojure в Echo

Когда начинает получаться?

— неделя

— от недели и больше

— недели две

— с учетом, что есть опыт в ФП

Clojure в Echo

Наиболее сложные области

- concurrency примитивы
- двухсторонний interop
- meta параметры
- идеология

Clojure в Echo

Полезно

- гибкость, лаконичность
- особенно чувствуется при переключении на другой язык
- скорость написания кода («опа-опа и готово»)
- java-библиотеки
- удобна для файлов конфигурации

Clojure в Echo

Что раздражает?

- скобки ⟨1 чел.⟩
- привязанность к Java ⟨2 чел.⟩
- непрозрачность кода из-за макросов ⟨2 чел.⟩
- медленный старт, тяжеловесность платформы ⟨2 чел.⟩

Clojure в Echo
Общее впечатление

- Большая неограниченная свобода. Можно писать как угодно, в любом стиле. Зеркало разработчика.
- Идеальна для соло проектов и плоха для командной разработки.
- Писать на Clojure очень легко, поэтому мы так много пишем и переписываем то, что пишем.

Clojure в Echo
Twitter Storm

Начиналось всё хорошо

Clojure в Echo
Twitter Storm

Фреймворк, не библиотека

Всё делает сам

Нужно интегрировать с ним тулзы

Нужно специальное тестирование

Clojure в Echo
Twitter Storm

Не переконфигурируется на лету

Баги в реализации ⟨leaks, deploy⟩

Фиксирует версии библиотек
⟨zookeeper, clojure, логгинг, вебстек?⟩

Заменили на plain old functions
+ систему управления кластером
⟨скоро⟩

Clojure в Echo
Midje: unit-testing

Начиналось всё хорошо

Clojure в Echo
Midje: unit-testing

Простые тесты писать проще*

```
⟨is ⟨= ⟨:peer res⟩ :p1⟩⟩

⟨fact
  res => ⟨contains {:peer :p1}⟩⟩
```

* Ну не сложнее точно

Clojure в Echo
Midje: unit-testing

Простые тесты писать проще*

```
(is (= (:peer res) :p1))

(fact
  res => (contains {:peer :p1}))
```

* Ну не сложнее точно

Clojure в Echo
Midje: unit-testing

```
(fact
  res => (contains {:peer :p1}))
```

Поведение неконсистентно

Обычная clojure не подходит

Свои checkers писать нереально

Адское макропрограммирование

Нет junit.xml вывода

и даже автор не представляет, как его добавить

Clojure в Echo

Midje clojure.test: unit-testing

```
(is (= (:peer res) :p1))
```

Прямой, тупой, бесхитростный

Расширяется во все стороны

Чистая дистилированная clojure

Нет mocking, только bindings :(

Clojure в Echo

Положительный опыт

clojure.data.*

clojure.tools.*

http.async.client

riemann

clj-redis

clj-oauth

amotoen

nippy

ring

compojure

clojurescript

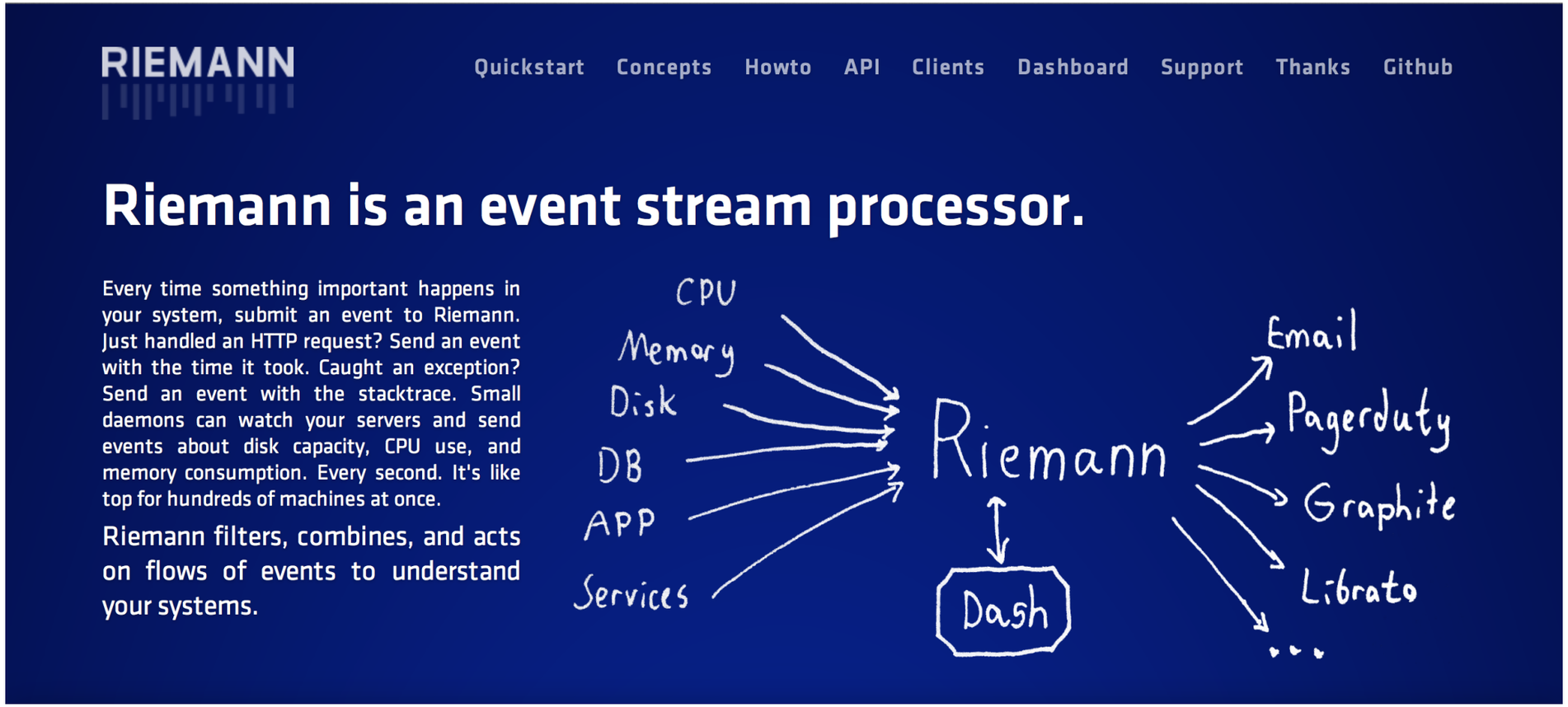
enfocus

jayq

shoreleave

Success stories

Riemann



Success stories
Datomic — БД нового типа

Summary

- Clojure was made for this kind of app
- Fast enough at all levels
- Most key subsystems < 1000 lines
- A ton of concurrency, no sweat
- Leverage interop - Hornetq, Guava etc
- Startup time could be better
- Datomic is Simple

Success stories

Prismatic – crawling, ML

Our Design Approach

We tend to roll our own.

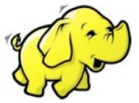
Libraries >> Frameworks

99.9% Clojure, 0.1% Java

Success stories

Prismatic – crawling, ML

One Slide Summary

1. **F**ine-grained **C**omposable **A**bstractions > Monolithic Frameworks 
2.  lets you make **FCA**
3.  **<3** 

Преимущества Clojure

Обработка данных

Concurrency

Быстрая разработка

Доступ к JVM

Когда использовать?

Параллельная обработка общего state

Обработка больших массивов данных ⟨анализ, обучение⟩

Большие приложения в браузере

Полезные ресурсы

- clojure.org
- clojuredocs.org
- tonsky.livejournal.com
- The Joy of Clojure
- Clojure Programming
- Programming Clojure

Спасибо за внимание!

tonsky.livejournal.com

jobs@aboutecho.com

Новосибирск, март 2013