

Чем нам полезен Ruby

в 2020 году

Григорий Петров

Давайте использовать социальные сети:

grigoryvp@evrone.com

t.me/grigoryvp

fb.com/grigoryvp

vk.com/grigoryvp

github.com/grigoryvp

twitter.com/grigoryvp

instagram.com/grigoryvp

Слайды



2

bit.ly/rdev2020

Соцсети



[@grigoryvp](https://t.me/grigoryvp)

Имя



Григорий Петров

Что сейчас будет?

```
{  
  speaker_name: "Григорий Петров",  
  skill_list: ["Ruby", "Python", "JS", "Go", "Rust", "C++"],  
  skill_years: 20,  
  job_title: "DevRel",  
  talk_minutes: 30,  
  questions_at: TALK_END,  
}
```

Краткая история Ruby

Краткая история Ruby

Все началось с беседы Yukihiro Matsumoto и Keiju Ishitsuka в 1993 году*



Краткая история Ruby

Все началось с беседы Yukihiro Matsumoto и Keiju Ishitsuka в 1993 году*



Краткая история Ruby



Краткая история Ruby



Краткая история Ruby



Краткая история Ruby

Октябрь 2004, ДНН* извлекает Rails



* David Heinemeier Hansson

Краткая история Ruby



Краткая история Ruby



Краткая история Ruby



Краткая история Ruby



Краткая история Ruby



Краткая история Ruby

А тут - плато стабильности

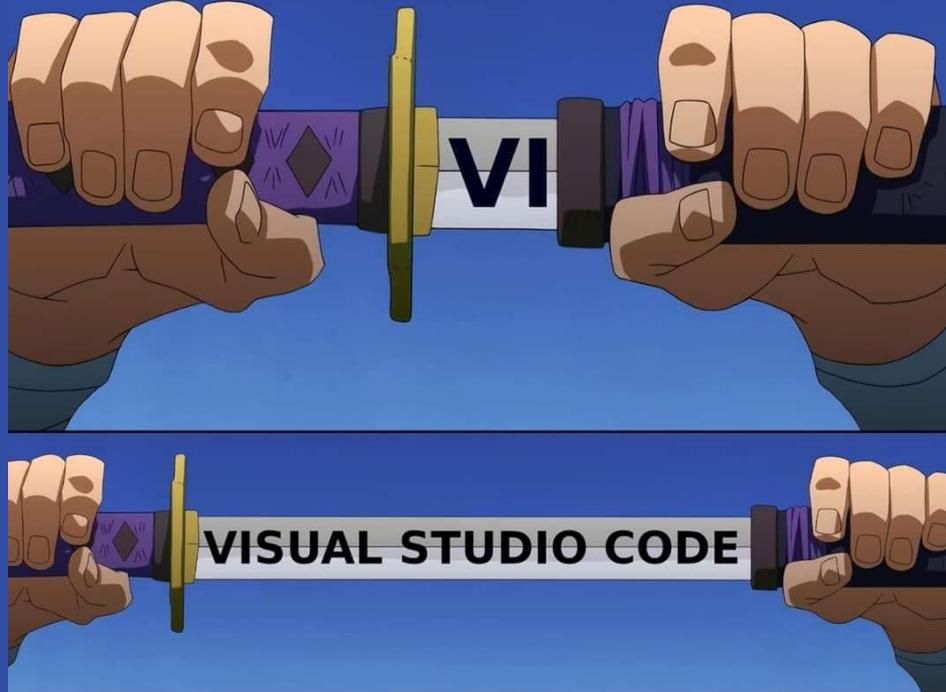


Краткая история Ruby

А тут - плато стабильности



Как оценить популярность технологий?



Как оценить популярность технологий?

→ Только по косвенным признакам.

Как оценить популярность технологий?

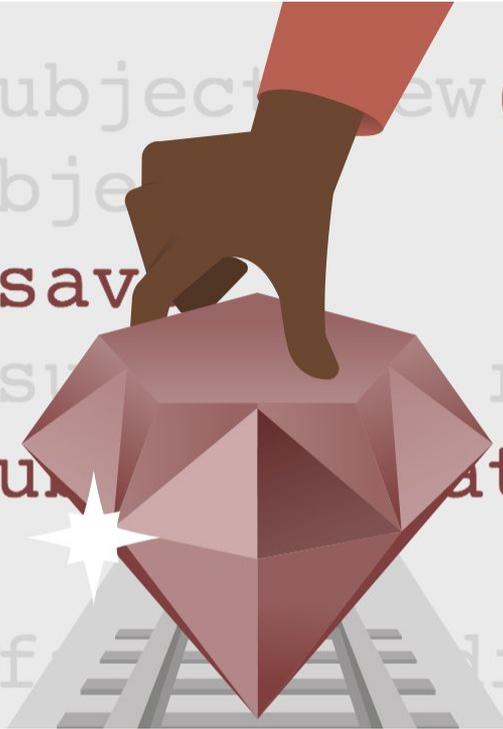
- ✓ Только по косвенным признакам.
- Статистика - сложная штука.

Как оценить популярность технологий?

- ✓ Только по косвенным признакам.
- ✓ Статистика - сложная штука.
- Поэтому поговорим о фактах.

Ruby on Rails

```
@subject = Subject.new(subject_params)
# Save the object
if @subject.save
  # If save successful, redirect to the
  redirect(subject_path)
else
  # If save failed, display the form
```



Ruby on Rails

```
@subject = Subject.new(subject_params)
# Save the object
if @subject.save
  # If save successful, redirect to the
  redirect(subject_path)
else
  # If save failed, display the form
```



Ruby on Rails

```
@subject = Subject.new(subject_params)
# Save the object
if @subject.save
  # If save successful, redirect to the
  redirect(subject_path)
else
  # If save fails, display the form
```



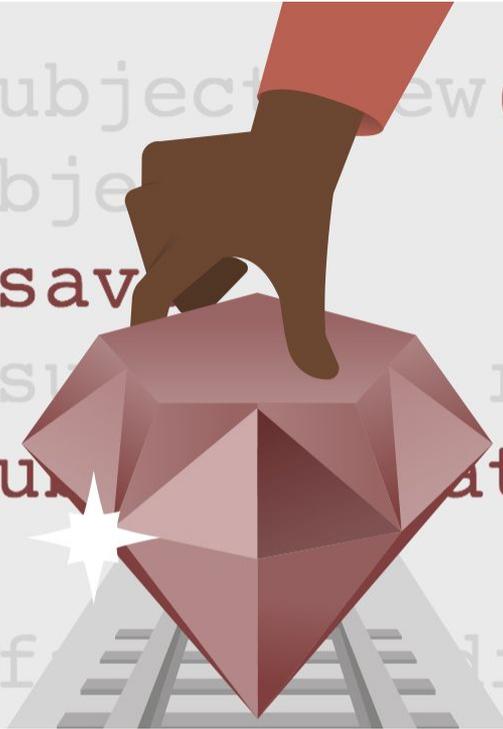
Ruby on Rails

```
@subject = Subject.new(subject_params)
# Save the object
if @subject.save
  # If save successful, redirect to the
  redirect(subject_path)
else
  # If save failed, display the form
```



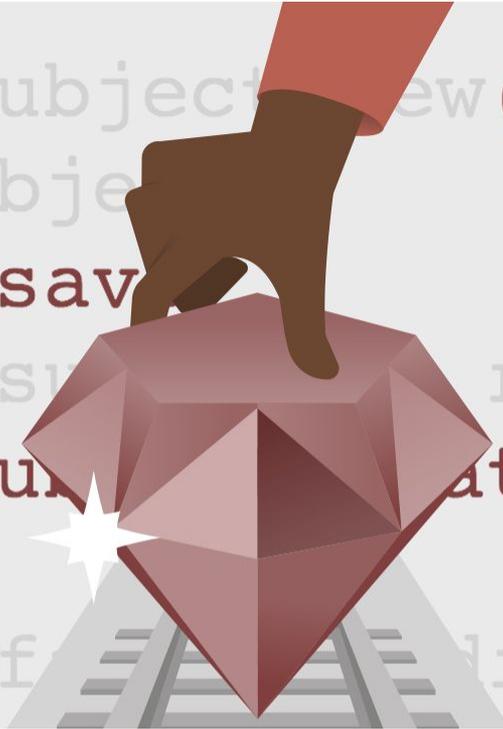
Ruby on Rails

```
@subject = Subject.new(subject_params)
# Save the object
if @subject.save
  # If save successful, redirect to the
  redirect(subject_path)
else
  # If save failed, display the form
```



Ruby on Rails

```
@subject = Subject.new(subject_params)
# Save the object
if @subject.save
  # If save successful, redirect to the
  redirect(subject_path)
else
  # If save failed, display the form
```



Ruby on Rails

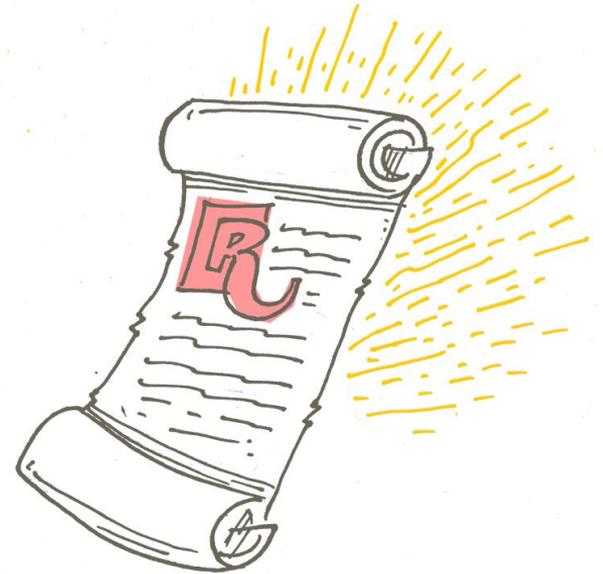
```
@subject = Subject.new(subject_params)
# Save the object
if @subject.save
  # If save successful, redirect to the
  redirect(subject_path)
else
  # If save failed, display the form
```



Ruby on Rails

→ "Рельсы" - это про opinionated и convention over configuration.

1. **Optimize for programmer happiness**
2. **Convention over Configuration**



Ruby on Rails

- ✓ "Рельсы" - это про opinionated и convention over configuration.
- DSL для максимального уровня абстракций.

```
describe UsersController, type: :controller do
  before do
    allow(controller).to receive(:current_user).and_return(nil)
  end

  describe "GET #new" do
    subject { get :new }

    it "returns success" do
      expect(subject).to be_success
    end
  end
end
```

Ruby on Rails

- ✓ "Рельсы" - это про opinionated и convention over configuration.
- ✓ DSL для максимального уровня абстракций.
- Подход "Majestic monolith".



Ruby on Rails

- ✓ "Рельсы" - это про opinionated и convention over configuration.
- ✓ DSL для максимального уровня абстракций.
- ✓ Подход "Majestic monolith".
- Точка интеграции популярных библиотек и решений.

"There's a gem for everything", например dry-rb

Ruby on Rails

- ✓ "Рельсы" - это про opinionated и convention over configuration.
- ✓ DSL для максимального уровня абстракций.
- ✓ Подход "Majestic monolith".
- ✓ Точка интеграции популярных библиотек и решений.
- Большинство Ruby разработчиков умеют в рельсы.

No fragmentation for Angular and Rails, big companies like that

Ruby on Rails

- ✓ "Рельсы" - это про opinionated и convention over configuration.
- ✓ DSL для максимального уровня абстракций.
- ✓ Подход "Majestic monolith".
- ✓ Точка интеграции популярных библиотек и решений.
- ✓ Большинство Ruby разработчиков умеют в рельсы.
- Быстрая разработка бизнес-логики и простого фронта.

Rails называют "технологией стартапов"

Ruby on Rails

- ✓ "Рельсы" - это про opinionated и convention over configuration.
- ✓ DSL для максимального уровня абстракций.
- ✓ Подход "Majestic monolith".
- ✓ Точка интеграции популярных библиотек и решений.
- ✓ Большинство Ruby разработчиков умеют в рельсы.
- ✓ Быстрая разработка бизнес-логики и простого фронта.
- Встроенный Webpack.

Ruby on Rails

- ✓ "Рельсы" - это про opinionated и convention over configuration.
- ✓ DSL для максимального уровня абстракций.
- ✓ Подход "Majestic monolith".
- ✓ Точка интеграции популярных библиотек и решений.
- ✓ Большинство Ruby разработчиков умеют в рельсы.
- ✓ Быстрая разработка бизнес-логики и простого фронта.
- ✓ Встроенный Webpack.
- AWS Lambda Serverless & Jets.

dry-rb



Author: Rostislav Katin

dry-rb

→ 22 библиотеки высокоуровневой логики.

dry-rb

- ✓ 22 библиотеки высокоуровневой логики.
- Идеи, давно доступные в Java Spring.

dry-rb

- ✓ 22 библиотеки высокоуровневой логики.
- ✓ Идеи, давно доступные в Java Spring.
- Идеи, породившие таких последователей как dry-python.

Ключевые части экосистемы



Ключевые части экосистемы

→ Bundler



Ключевые части экосистемы

- ✓ Bundler
- RubyGems



Ключевые части экосистемы

- ✓ Bundler
- ✓ RubyGems
- Rake



Ключевые части экосистемы

- ✓ Bundler
- ✓ RubyGems
- ✓ Rake
- RSpec



Ключевые части экосистемы

- ✓ Bundler
- ✓ RubyGems
- ✓ Rake
- ✓ RSpec
- Rubocop



Ключевые части экосистемы

- ✓ Bundler
- ✓ RubyGems
- ✓ Rake
- ✓ RSpec
- ✓ Rubocop
- Sorbet и Steep



Слабые места



Слабые места

→ Machine Learning.

Слабые места

- ✓ Machine Learning.
- Асинхронность с Fibers непопулярна.

Слабые места

- ✓ Machine Learning.
- ✓ Асинхронность с Fibers непопулярна.
- Слабая документация *

* Если сравнивать с Real Python

Слабые места

- ✓ Machine Learning.
- ✓ Асинхронность с Fibers непопулярна.
- ✓ Слабая документация.
- Низкая популярность за пределами Rails.

Выводы



[VK.COM/NIK_ARAGUA](https://vk.com/nik_aragua)



Выводы

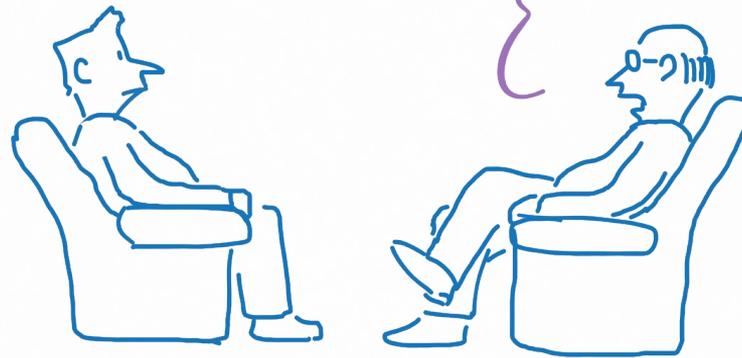


[VK.COM/NIK_ARAGUA](https://vk.com/NIK_ARAGUA)



Это всё!

ДАВАЙТЕ ОБСУДИМ™



Кур. Анастасии для Григория ПЕТРОВА *AM*