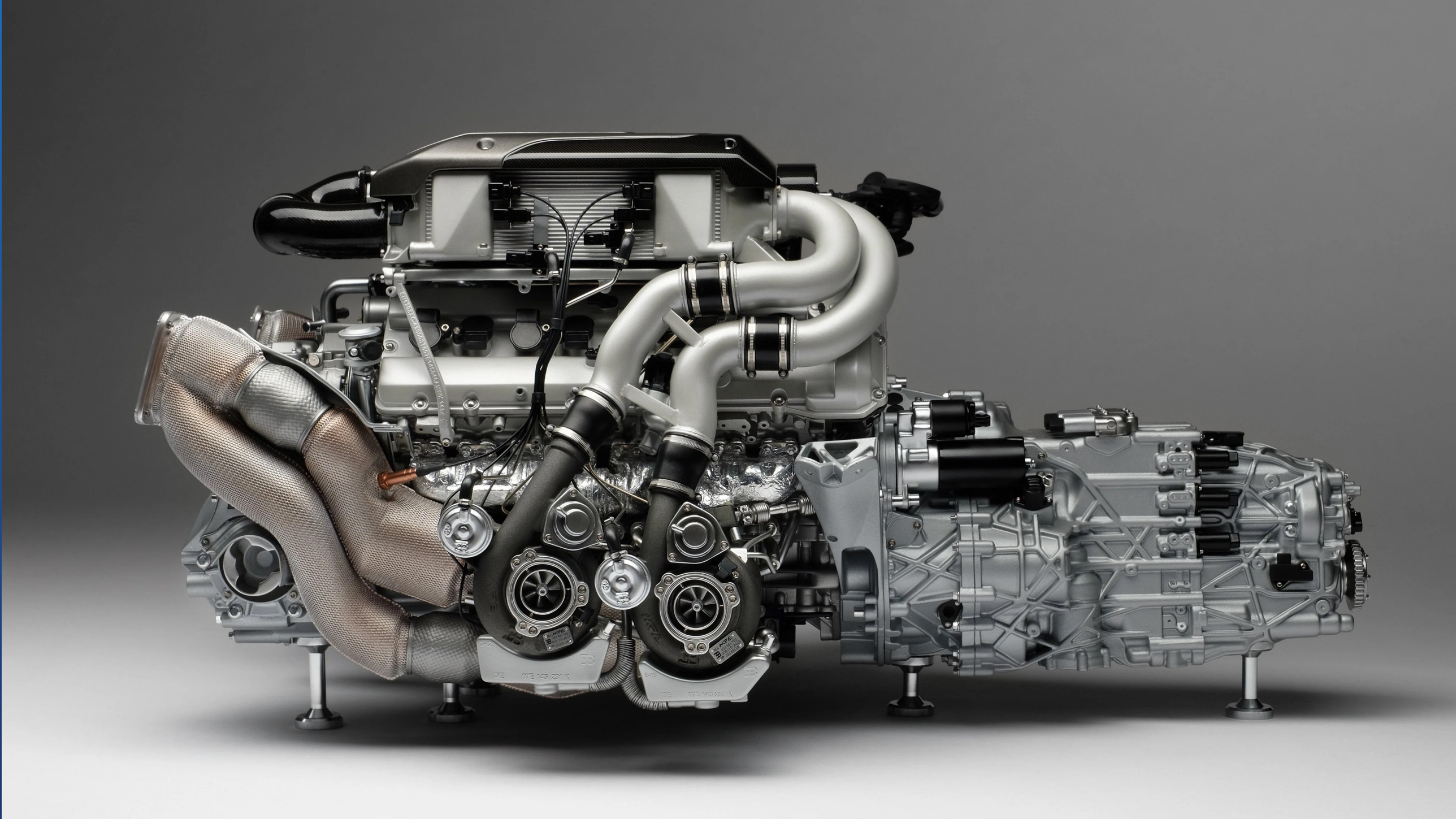
Ruby (on Rails)

DISK {mITup!}
Slatina
08.12.2023.

Vlado Cingel

https://github.com/vlado
vlado@cingel.hr

I love my Wife, Kids
and Ruby

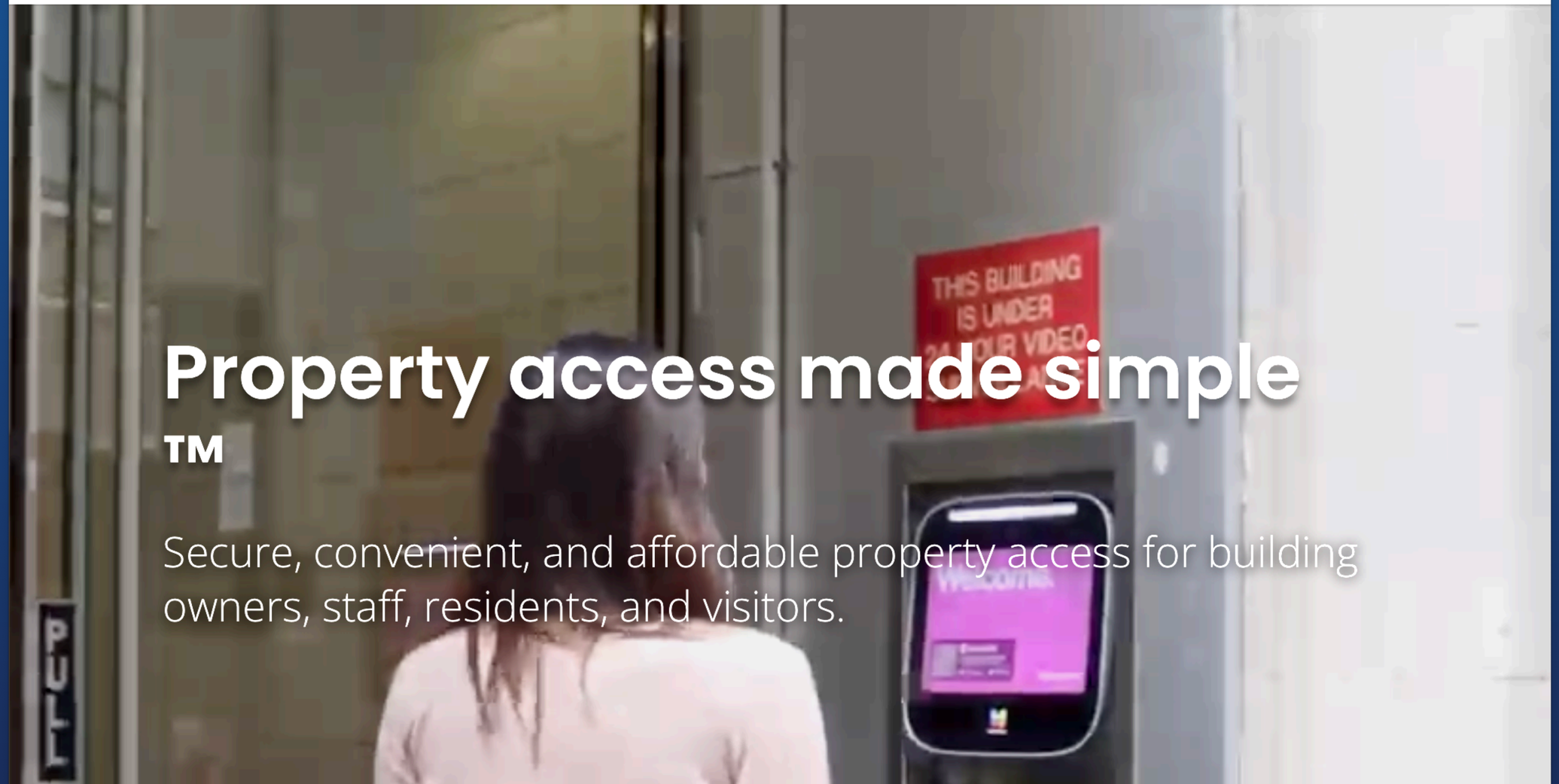



QUALITY
MANAGEMENT
ISO
STANDARDS
PROCESS
APPROACH
IMPROVEMENT
PEOPLE
ORGANIZATION
METHODS
STANDARD
CUSTOMER
SYSTEM
PRODUCTION
CHANGE
CHANGES
EXAMPLE
PRINCIPLES
ONE
STATISTICAL
JAPAN
LEADERSHIP
CONTROL
CUSTOMERS
ENTERPRISE
SIGMA
USE
SOFTWARE
SIX
DEMING
KAIZEN
CONTINUAL
INTERNATIONAL
CALLED
SERVICE
PRODUCTS
CULTURE
ASSURANCE
IMPORTANT
WORK
9001:2008
ACHIEVING
INTEGRATED
MANY
IMPROVE
NUMBER
ALSO
PRACTICES
FACTORS
COUNTRIES
WELL
BASED
NATIONAL
JAPANESE
REQUIREMENTS
INCLUDING
MAJOR
PRODUCT
PERFORMANCE
STANDARDIZATION
PROCESSES
SERIES
FOUNDATION
MANUFACTURE
ASSESSMENT
INDUSTRY
TRANSFORMATIONAL
INTERNAL
COMPONENTS
MUTUALLY
EFFICIENCY
MANUFACTURING
ENTERPRISES
TERMS
INDUSTRIAL
RELEASED
MOTOR
MAKE
WORLD
OFTEN
EVOLUTION
FRAMEWORK
INCLUDE
SUCCESS
GOODS
BUSINESS
USED
DECADES
OTHERS
BENEFICIAL
CREATED
ASSEMBLY
FIRST
THEREFORE
WAR
PART
SOUGHT
BECOME
FOCUS
LIKE
NEW
TOWARDS
HIGH
BETTER
INVOLVEMENT

Ruby

butterflymx.com

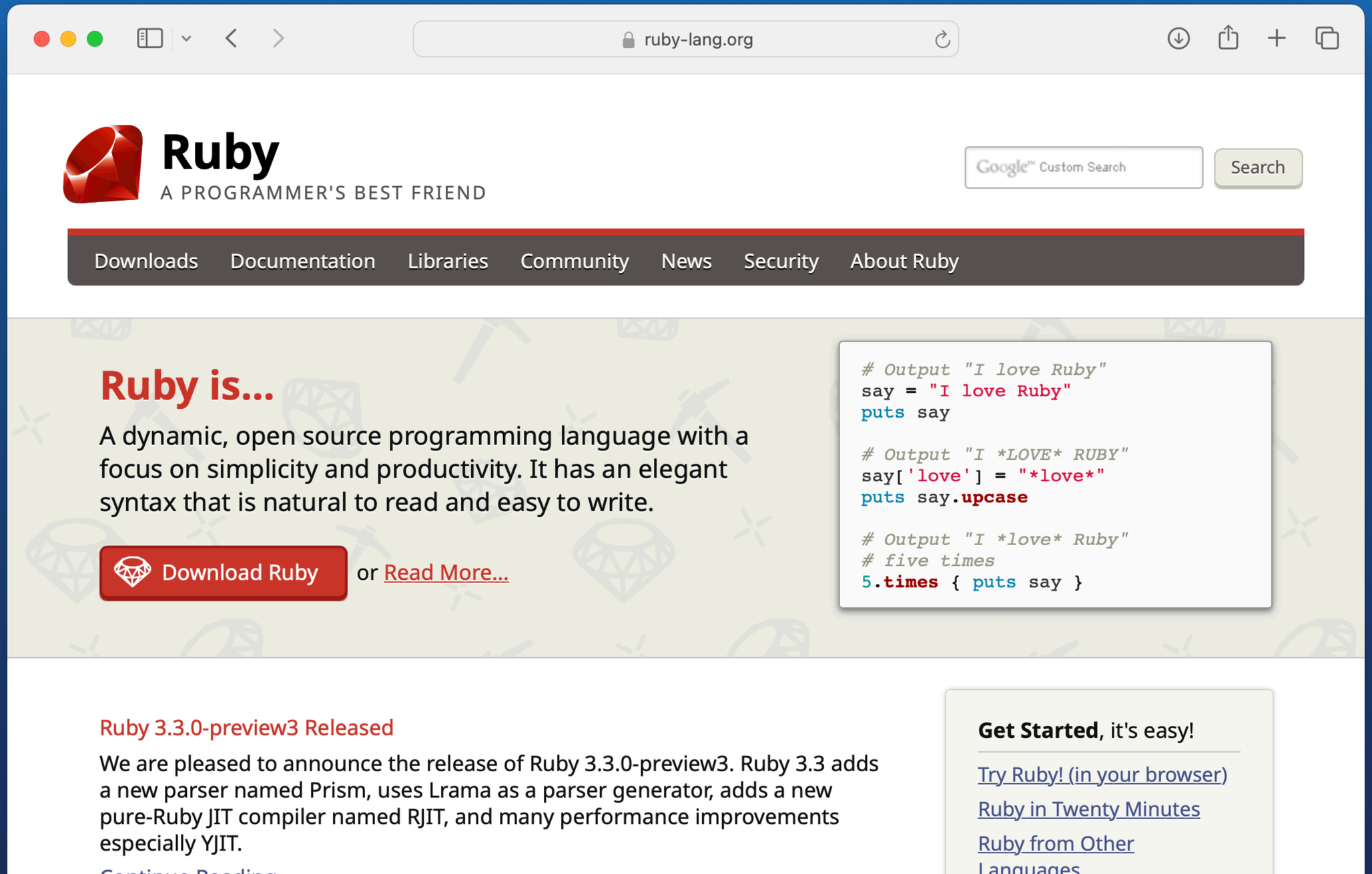
ButterflyMX



Property access made simple™

Secure, convenient, and affordable property access for building owners, staff, residents, and visitors.

Ruby


ruby-lang.org
Ruby
A PROGRAMMER'S BEST FRIEND
Google™ Custom Search
Search
Downloads
Documentation
Libraries
Community
News
Security
About Ruby
Ruby is...
A dynamic, open source programming language with a focus on simplicity and productivity. It has an elegant syntax that is natural to read and easy to write.
Download Ruby
or Read More...
# Output "I love Ruby"
say = "I love Ruby"
puts say
Output "I *LOVE* RUBY"
say['love'] = "*love*"
puts say.upcase
Output "I *love* Ruby"
five times
5.times { puts say }
Ruby 3.3.0-preview3 Released
We are pleased to announce the release of Ruby 3.3.0-preview3. Ruby 3.3 adds a new parser named Prism, uses Lrama as a parser generator, adds a new pure-Ruby JIT compiler named RJIT, and many performance improvements especially YJIT.
Get Started, it's easy!
Try Ruby! (in your browser)
Ruby in Twenty Minutes
Ruby from Other Languages

Yukihiro Matsumoto
“Matz”

1995

“I wanted to design my own language to maximize my freedom“

Matz

“I wanted to design a language in which I could love programming”

Matz

Hello World

```
puts "Hello World”
# => “Hello World”
```

Print “I love Ruby!” 5 times

```ruby
5.times { puts "I love Ruby!" }
# => “I love Ruby!”
# => “I love Ruby!”
# => “I love Ruby!”
# => “I love Ruby!”
# => “I love Ruby!”
```

Check if number is even

```
5.even?
# => false

4.even?
# => true
```

Find odd numbers

```
[1, 4, 5, 7, 12, 15].filter(&:odd?)
# => [1, 5, 7, 15]
```

Iterate over Array

```
[1, 4, 5].each do |number|
  square = number ** 2
  puts "Square of #{number} is #{square}”
end
# => "Square of 1 is 1"
# => "Square of 4 is 16"
# => "Square of 5 is 25"
```

In Ruby everything is an Object

Data Types in Ruby

Strings

```
“Hello World".reverse
# => "dlroW olleH”
```

Numbers

```ruby
1.class
# => Integer

4.5678.class
# => Float

4.5678.round(2)
# => 4.57
```

Booleans

```
true.class
# => TrueClass

false.class
# => FalseClass

true.nil?
# => false

false.to_s
# => “false"
```

Arrays

```
array = [5, 1, 3]

array.size
# => 3

array.min
# => 1

array.max
# => 5

array.sum
# => 9

array.sort
# => [1, 3, 5]
```

Symbols

```
:foo.class
# => Symbol

:foo.to_s
# => “foo"
```

Hashes

```
hash = { "Bulls" => "Chicago", "Lakers" => "LA", "Celtics" => "Boston" }

hash["Bulls"]
# => “Chicago"

hash["Celtics"]
# => "Boston"

hash.keys
# => ["Bulls", "Lakers", "Celtics"]

hash.values
# => ["Chicago", "LA", “Boston"]
```

Blocks

Little anonymous functions that can be passed into methods

```
[1, 4, 5].each do |number|
  square = number ** 2
  puts "Square of #{number} is #{square}”
end
```

Classes and Objects

```ruby
class Club
  def initialize(city, name)
    @city = city
    @name = name
  end

  def full_name
    "#{@city} #{@name}"
  end
end
```

```ruby
class Player
  attr_reader :club

  def initialize(first_name, last_name, club)
    @first_name, @last_name, @club = first_name, last_name, club
  end

  def full_name
    "#{@first_name} #{@last_name}"
  end

  def move_to(new_club)
    @club = new_club
  end
end
```

```ruby
bulls = Club.new("Chicago", "Bulls")
wizards = Club.new("Washington", "Wizards")

michael = Player.new("Michael", "Jordan", bulls)
```

```ruby
michael.full_name
# => "Michael Jordan"

michael.club.full_name
# => "Chicago Bulls"

michael.move_to(wizards)
michael.club.full_name
# => "Washington Wizards"
```

Rails

David Heinemeier Hansson

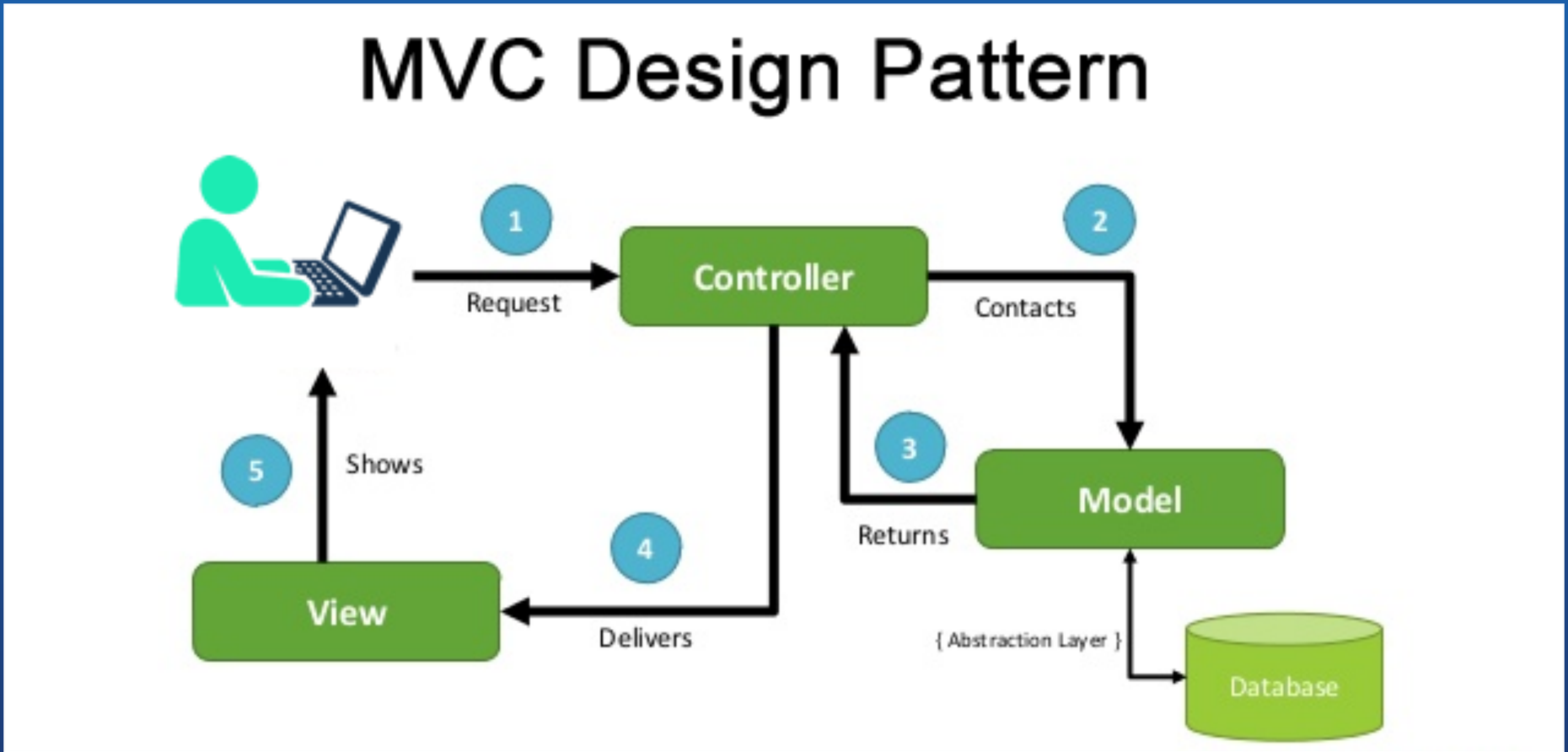
"dhh"

2003

“Ruby on Rails is an open-source web framework that's optimized for programmer happiness and sustainable productivity. It lets you write beautiful code by favoring convention over configuration“

dhh

Model-View-Controller



Model in MVC

- The Model layer represents the domain model (such as Account, Product, Person, Post, etc.) and encapsulates the business logic specific to your application.
- In Rails, database-backed model classes are derived from ActiveRecord::Base. Active Record allows you to present the data from database rows as objects and embellish these data objects with business logic methods.
- Although most Rails models are backed by a database, models can also be ordinary Ruby classes, or Ruby classes that implement a set of interfaces as provided by the Active Model module.

```ruby
# app/models/article.rb
class Article < ApplicationRecord
  def byline
    "'#{title}' written on #{created_at.to_s(:short)}"
  end
end

Article.all
# SELECT * from articles
# => [...]

Article.create(title: "Great article about Rails")
# INSERT INTO articles (name, created_at)
# VALUES ("Great article about Rails”, 2023-12-08 19:18:17)

article = Article.find(1)
# SELECT * from articles WHERE id = 1

article.byline
# => "'Great article about Rails' written on 08.12."
```

View in MVC

- The View layer is composed of "templates" that are responsible for providing appropriate representations of your application's resources.
- Templates can come in a variety of formats, but most view templates are HTML with embedded Ruby code (ERB files).
- Views are typically rendered to generate a controller response or to generate the body of an email.
- In Rails, View generation is handled by Action View.

```
# app/views/articles/show.html.erb
<h1><%= @article.title %></h1>

<%= image_tag @article.cover_image.url %>

<p><%= @article.content %></p>

<%= link_to "Edit", edit_article_path(@article) %>
```

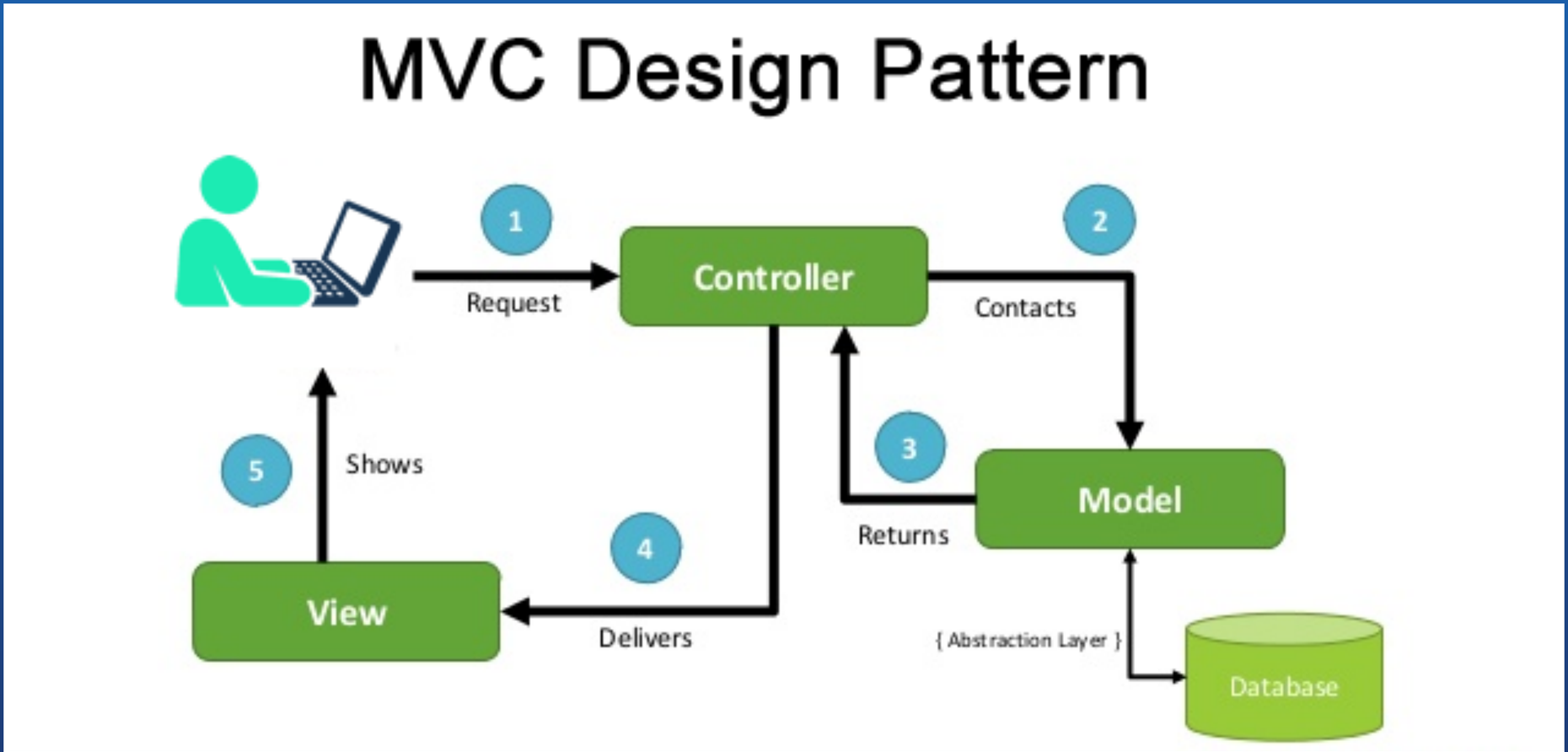
Controller in MVC

- The Controller layer is responsible for handling incoming HTTP requests and providing a suitable response.
- Usually, this means returning HTML, but Rails controllers can also generate XML, JSON, PDFs, mobile-specific views, and more.
- Controllers load and manipulate models, and render view templates in order to generate the appropriate HTTP response.
- In Rails, incoming requests are routed by Action Dispatch to an appropriate controller, and controller classes are derived from ActionController::Base.
- Action Dispatch and Action Controller are bundled together in Action Pack.

```ruby
# app/controllers/articles_controller.rb
class ArticlesController < ApplicationController
  def index
    @articles = Article.all
  end

  def show
    @article = Article.find(params[:id])
  end

  def create
    article = Article.create!(article_params)
    redirect_to article
  end

  private

  def article_params
    params.require(:article).permit(:title, :content)
  end
end
```

```ruby
# config/routes.rb
Rails.application.routes.draw do
  resources :articles do     # /articles, /articles/1
    resources :comments      # /articles/1/comments, /comments/1
  end

  root to: "articles#index" # /
end
```

Model-View-Controller



Rails

- ActiveRecord
- ActiveModel
- ActionPack
- ActionView

- ActiveSupport
- ActiveJob
- ActiveStorage
- ActionMailer
- ActionMailbox
- ActionCable
- ActionText

ActionMailer

Library to generate and send emails

```
article = Article.create(...)

# app/mailers/notifier.rb
class Notifier < ActionMailer::Base
  def new_article(article)
    @article = article
    mail(to: article.subscribers, subject: "New article: #{article.title}")
  end
end

# app/views/notifier/new_article.text.erb
Hello there,
We have new article '<%= @article.title  %>' for you!

message = Notifier.new_article(article)
message.deliver_now
```

ActiveJob

Framework for declaring jobs and making them run on a variety of queuing backends

```ruby
# app/jobs/notifiy_article_subscribers_job.rb
class NotifyArticleSubscribersJob < ActiveJob::Base
  queue_as :mailings

  def perform(article)
    article.subscribers.where(notified_at: nil).each do |subscriber|
      Notifier.new_article(article, subscriber).deliver_now
      subscriber.update(notifed_at: Time.current)
    end
  end
end

NotifyArticleSubscribersJob.perform_later(article)
NotifyArticleSubscribersJob.set(wait_unit: Date.tommorrow.noon).perform_later(article)
NotifyArticleSubscribersJob.set(wait: 5.minutes).perform_later(article)
```

ActiveStorage

Library to attach cloud and local files to Rails applications

```
class Article < ActiveRecord::Base
  has_one_attached :cover_image
end

# Attach an cover_image to the article.
article.cover_image.attach(
  io: File.open("/path/to/cover.jpg"),
  filename: "cover.jpg",
  content_type: “image/jpeg”
)

# Does the article have an cover_image?
article.cover_image.attached? # => true

# Synchronously destroy the cover_image and actual resource files.
article.cover_image.purge

# Destroy the associated models and actual resource files async, via Active Job.
article.cover_image.purge_later
```

ActiveSupport

Collection of utility classes and standard library extensions that are useful for Rails, and may also be used independently outside Rails

```ruby
# Ruby
"".empty?
# => true

"   ".empty?
# => false

nil.empty?
# => NoMethodError

str.nil? || str.strip.empty?
# => true
```

```ruby
# Rails (ActiveSupport)
"".blank?
# => true

"   ".blank?
# => true

nil.blank?
# => true

str.blank?
# => true
```

ActiveSupport

Collection of utility classes and standard library extensions that are useful for Rails, and may also be used independently outside Rails

```
1.year
2.months
3.weeks
4.days

1.week.ago
3.days.from_now

1.week + 4.days
# => 11.days
```

```
hash = { "foo" => "bar", fiz: "baz" }

hash["foo"]
# => "bar"
hash[:fiz]
# => "baz"
hash[:foo]
# => nil
hash["fiz"]
# => nil

hash = hash.with_indifferent_access

hash[:foo]
# => "bar"
hash["fiz"]
# => "baz"
```

ActionMailbox

Library to receive emails within a Rails application

- Action Mailbox routes incoming emails to controller-like mailboxes for processing in Rails.
- The inbound emails are turned into InboundEmail records using Active Record and feature lifecycle tracking, storage of the original email on cloud storage via Active Storage, and responsible data handling with on-by-default incineration.

ActionCable

Framework to integrate WebSockets with a Rails application

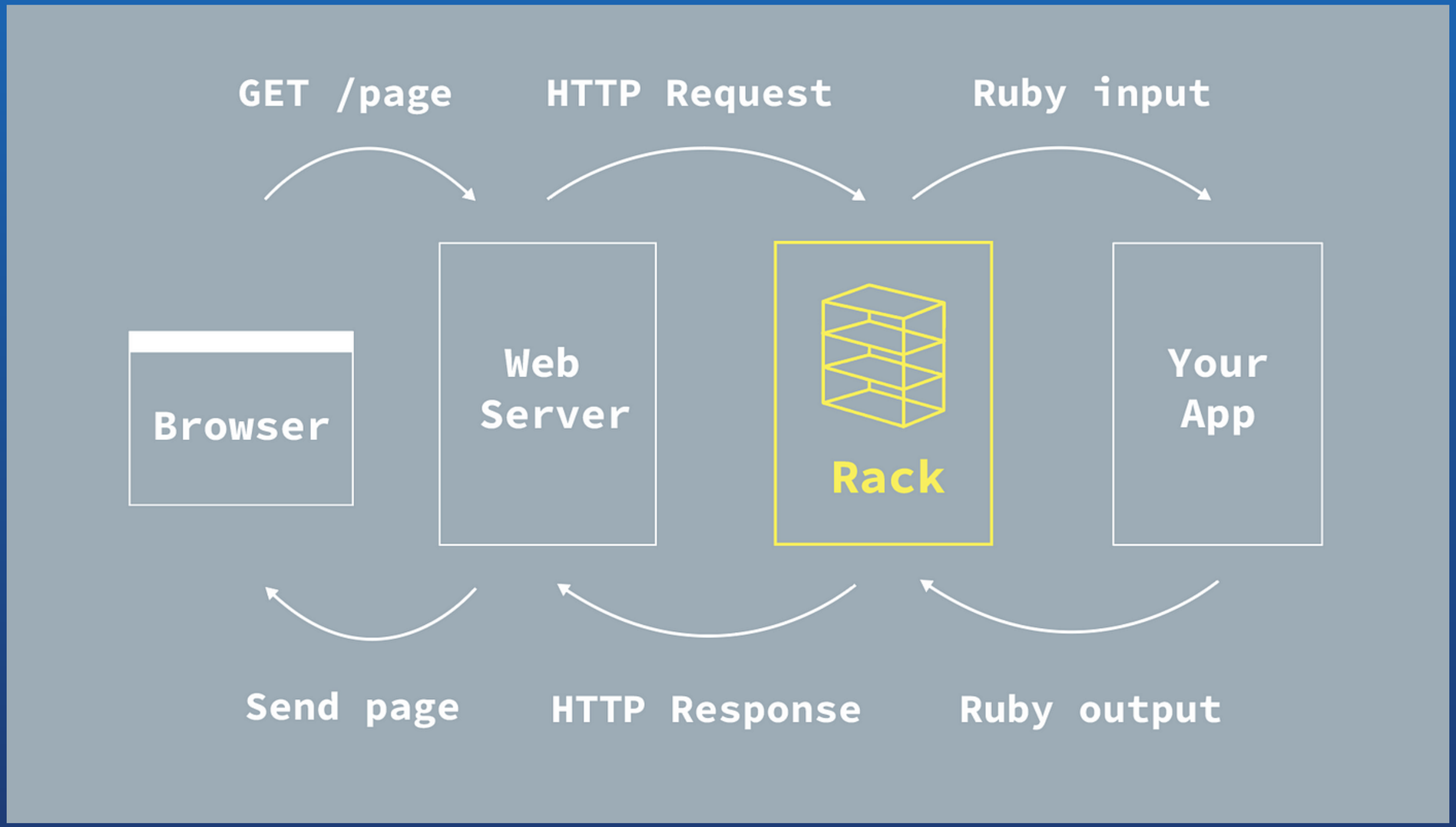
- It allows for real-time features to be written in Ruby in the same style and form as the rest of your Rails application, while still being performant and scalable.
- It's a full-stack offering that provides both a client-side JavaScript framework and a server-side Ruby framework.
- You have access to your entire domain model written with Active Record or your ORM of choice.

ActionText

Library to handle rich text content

- It includes the Trix editor that handles everything from formatting to links to quotes to lists to embedded images and galleries.
- The rich text content generated by the Trix editor is saved in its own RichText model that's associated with any existing Active Record model in the application.
- Any embedded images (or other attachments) are automatically stored using Active Storage and associated with the included RichText model.

Built on Rack



Compatible with all other Ruby web frameworks

Rack

“Rack provides a minimal, modular, and adaptable interface for developing web applications in Ruby. By wrapping HTTP requests and responses in the simplest way possible, it unifies and distills the bridge between web servers, web frameworks, and web application into a single method call.”

```
# Input
{
  "REMOTE_HOST" => "localhost",
  "REQUEST_METHOD" => "GET",
  "REQUEST_URI" => “https://example.com/articles",
  "SERVER_PROTOCOL" => "HTTP/1.1",
  "rack.version" => [1, 2],
  "rack.multithread" => true,
  …
}

# Output
[200, {}, ["Hello World”]]
```

Opinionated

https://dhh.dk/2012/rails-is-omakase.html


RAILS?
2450
WHERE WE'RE GOING
WE DON'T NEED RAILS

rubyonrails.org

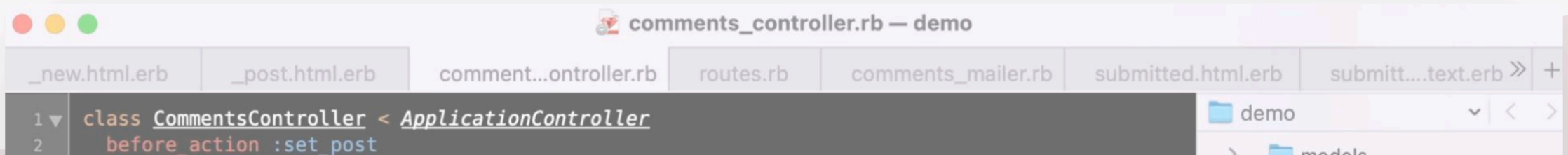
The Rails Job Board is live - Find your next Rails role here.

Guides API Forum Contribute Foundation Team Blog Jobs

Compress the complexity of modern web apps.

Learn just what you need to get started, then keep leveling up as you go. **Ruby on Rails scales from HELLO WORLD to IPO.**

Rails 7.1.2 — released November 10, 2023



Everything you need.

Rails is a full-stack framework. It ships with all the tools needed to build amazing web apps on both the front and back end.

Rendering HTML templates, updating databases, sending and receiving emails, maintaining live pages via WebSockets, enqueuing jobs for asynchronous work, storing uploads in the cloud, providing solid security protections for common attacks. Rails does it all and so much more.

The One Person Framework

Optimized for happiness.

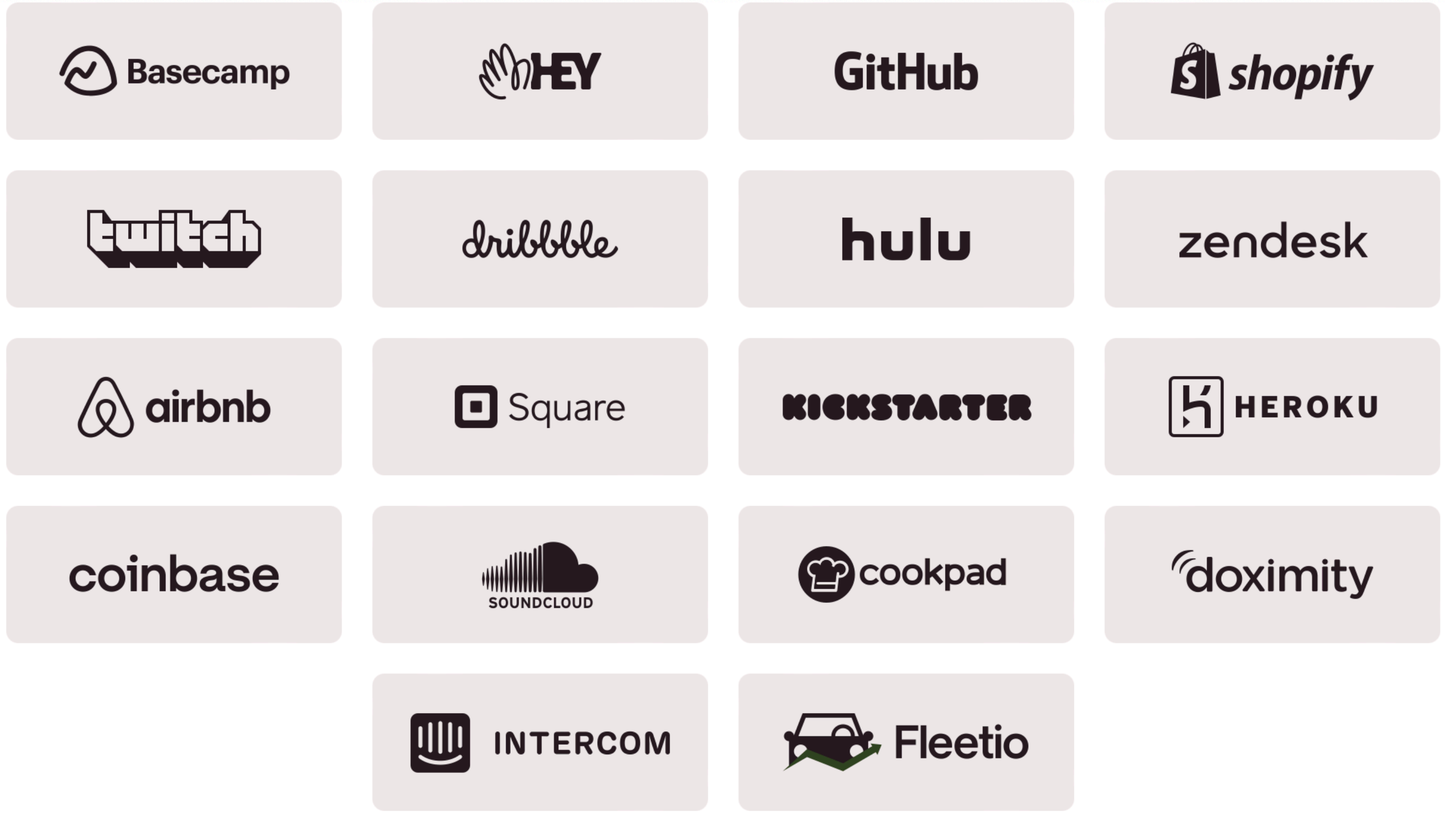
Rails has united and cultivated a strong tribe around a **wide set of heretical thoughts** about the nature of programming and programmers. Understanding these thoughts will help you understand the design of the framework.

Building it together.

Over six thousand people have **contributed code to Rails**, and many more have served the community through evangelism, documentation, and bug reports. Join us!

You're in good company.

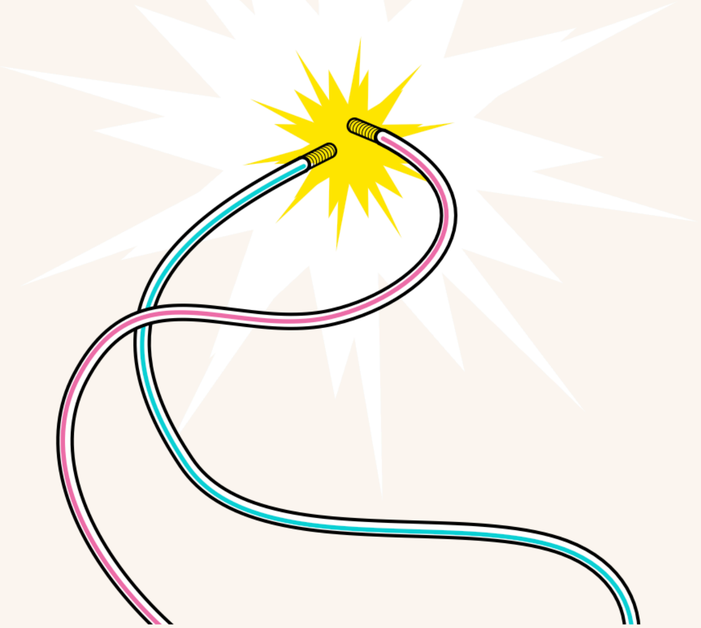
Over the past two decades, Rails has taken countless companies to millions of users and billions in market valuations.



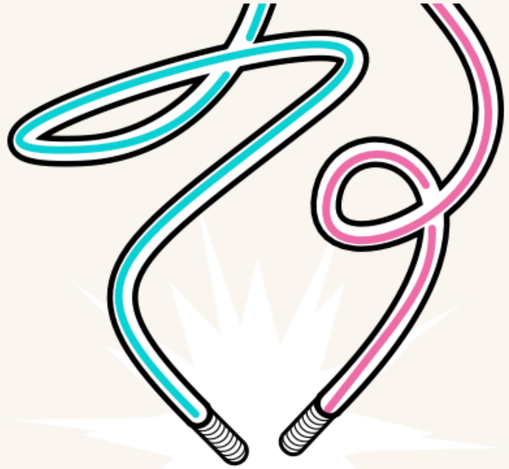
These are just a few of the big names. There have been many hundreds of thousands of apps created with Rails since its inception.

HOTWIRE

HTML OVER THE WIRE



Hotwire is an alternative approach to building modern web applications without using much JavaScript by sending HTML instead of JSON over the wire. This makes for fast first-load pages, keeps template rendering on the server, and allows for a simpler, more productive development experience in any programming language, without sacrificing any of the speed or responsiveness associated with a traditional single-page application.



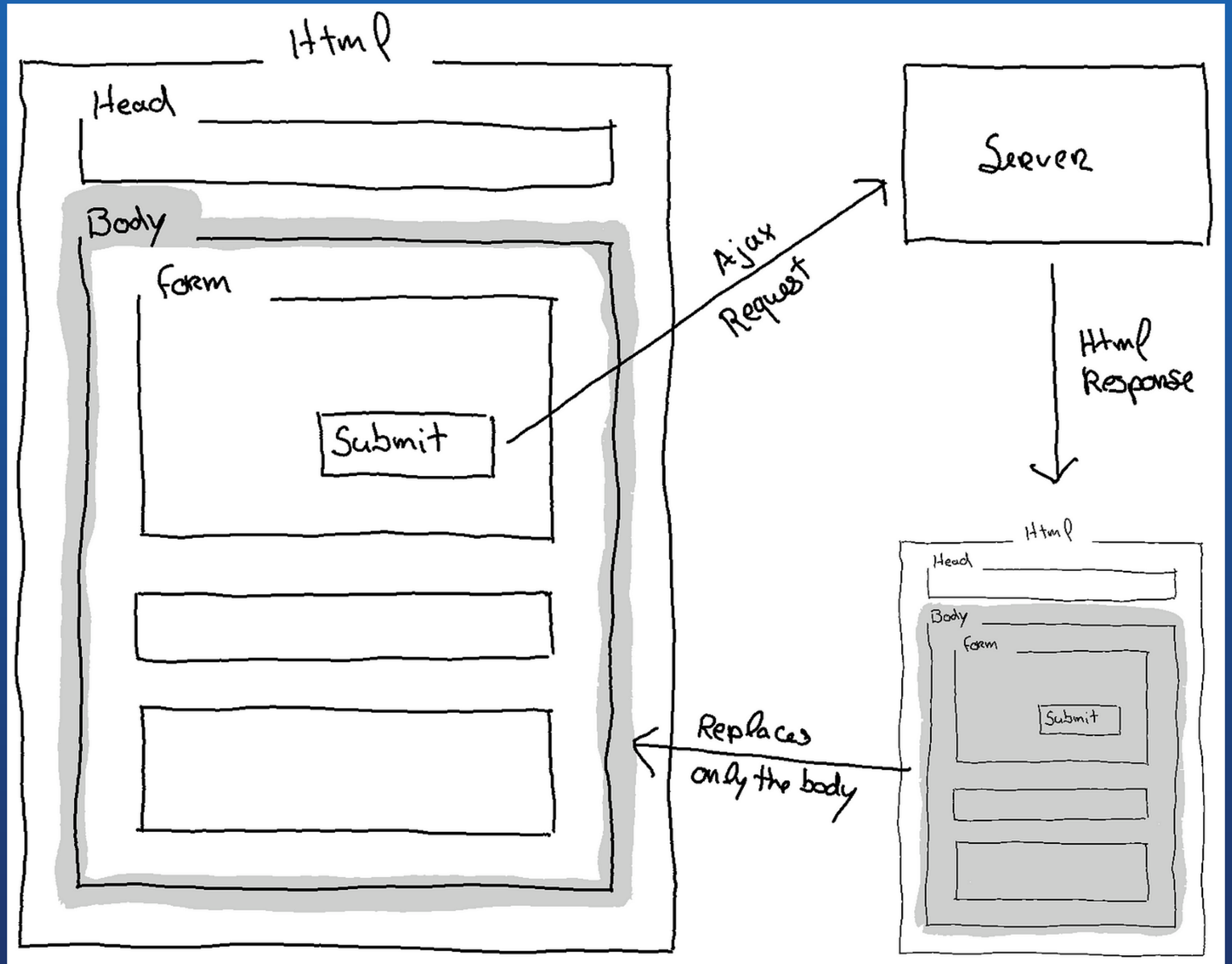
TURBO

The heart of Hotwire is Turbo. A set of complementary techniques for speeding up page changes and form submissions, dividing complex pages into components, and stream partial page updates over WebSocket. All without writing any JavaScript at all. And designed from the start to integrate perfectly with native hybrid applications for iOS and Android.

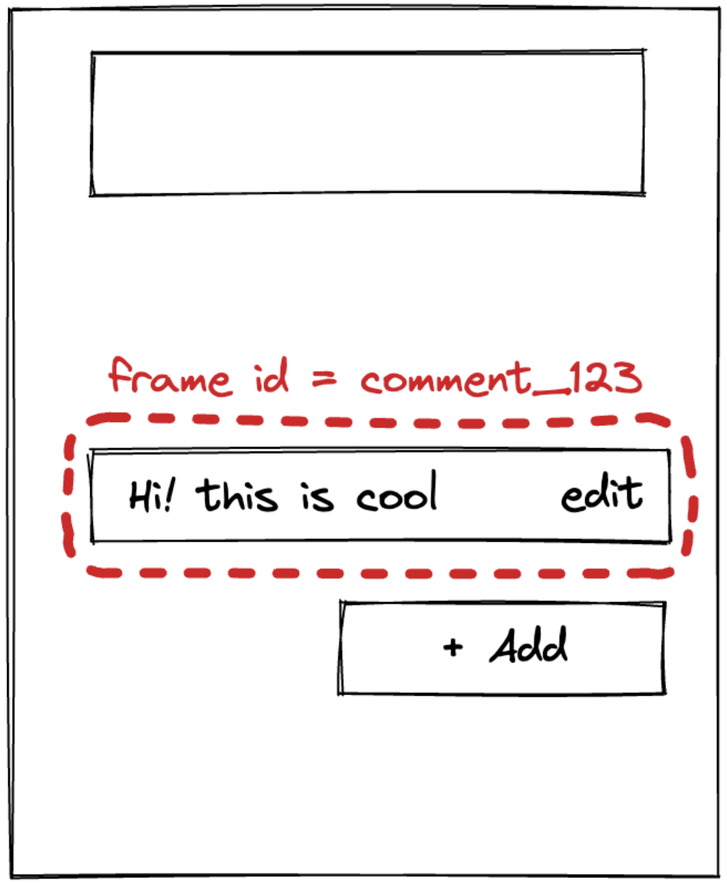
Turbo

- **Turbo Drive** accelerates links and form submissions by negating the need for full page reloads.
- **Turbo Frames** decompose pages into independent contexts, which scope navigation and can be lazily loaded.
- **Turbo Streams** deliver page changes over WebSocket, SSE or in response to form submissions using just HTML and a set of CRUD-like actions.
- **Turbo Native** lets your majestic monolith form the center of your native iOS and Android apps, with seamless transitions between web and native sections.

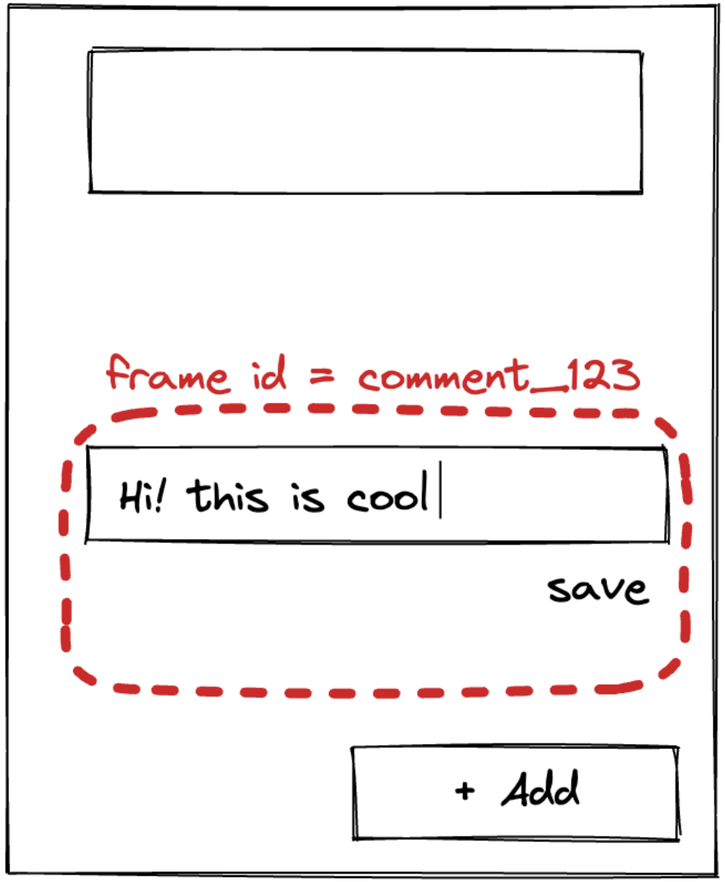
Turbo Drive


Html
Head
Body
Form
Submit
Server
Ajax
Request
Html
Response
Html
Head
Body
Form
Submit
Replaces
only the body

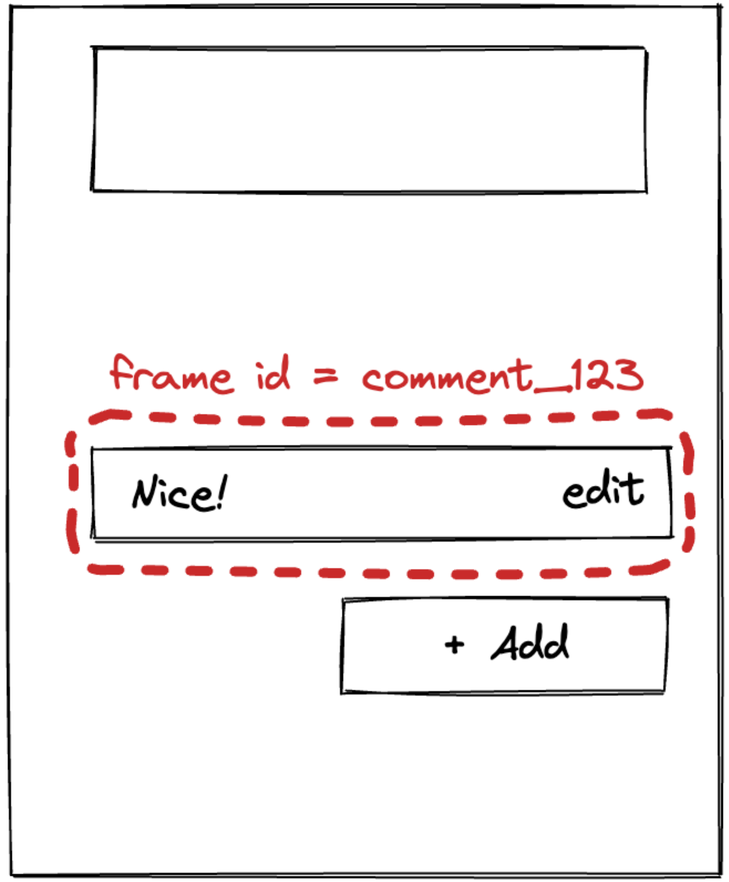
Turbo Frame



GET comments#edit

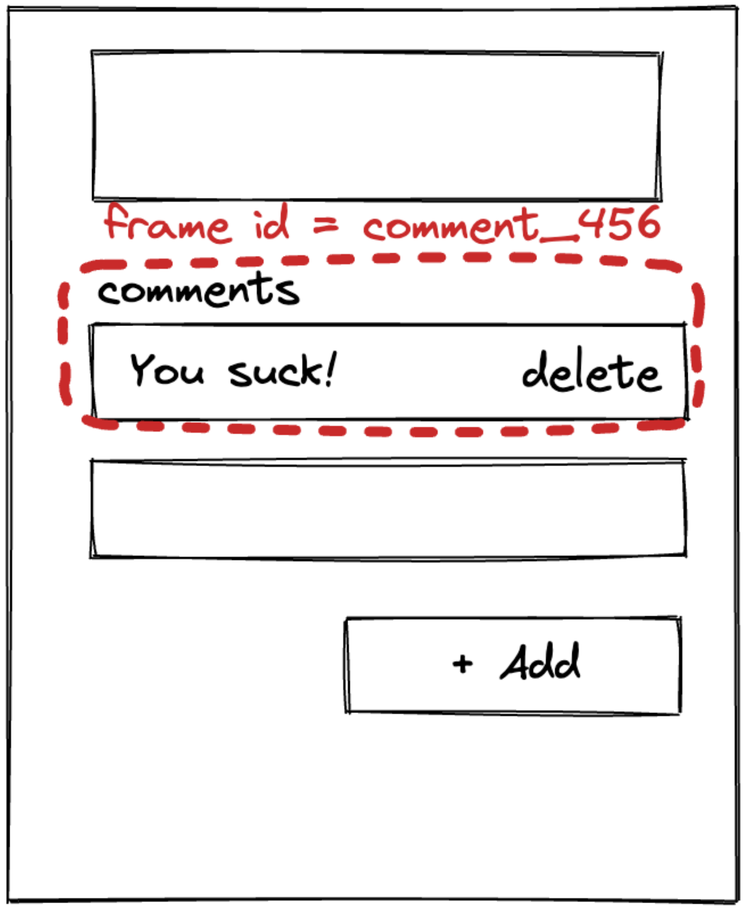


comments#update
redirect to comment#show



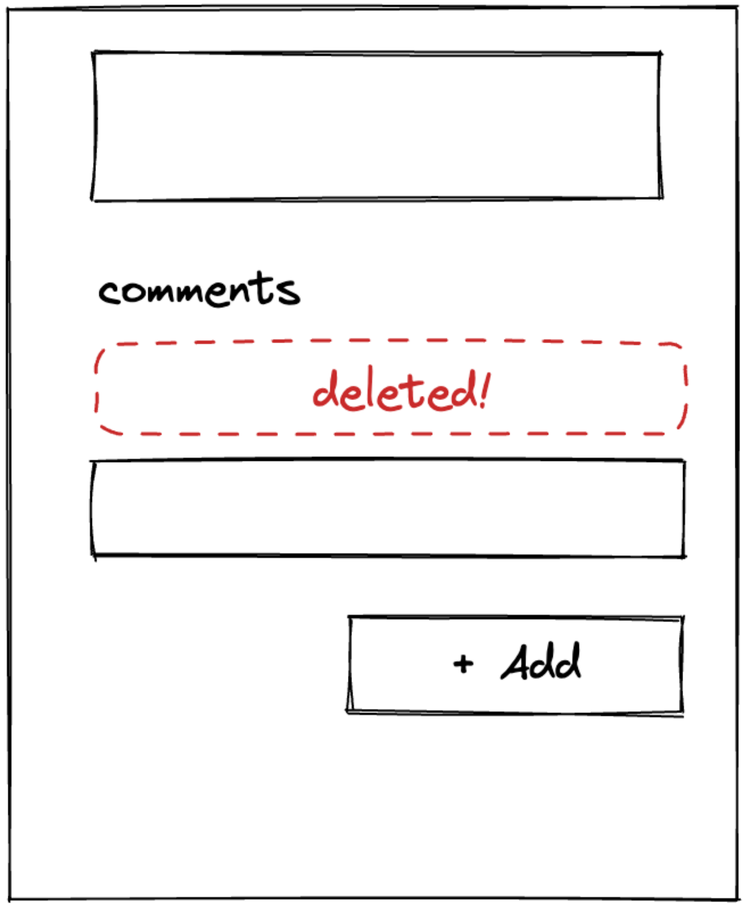
Turbo Stream

Server-generated JavaScript response (SJR)



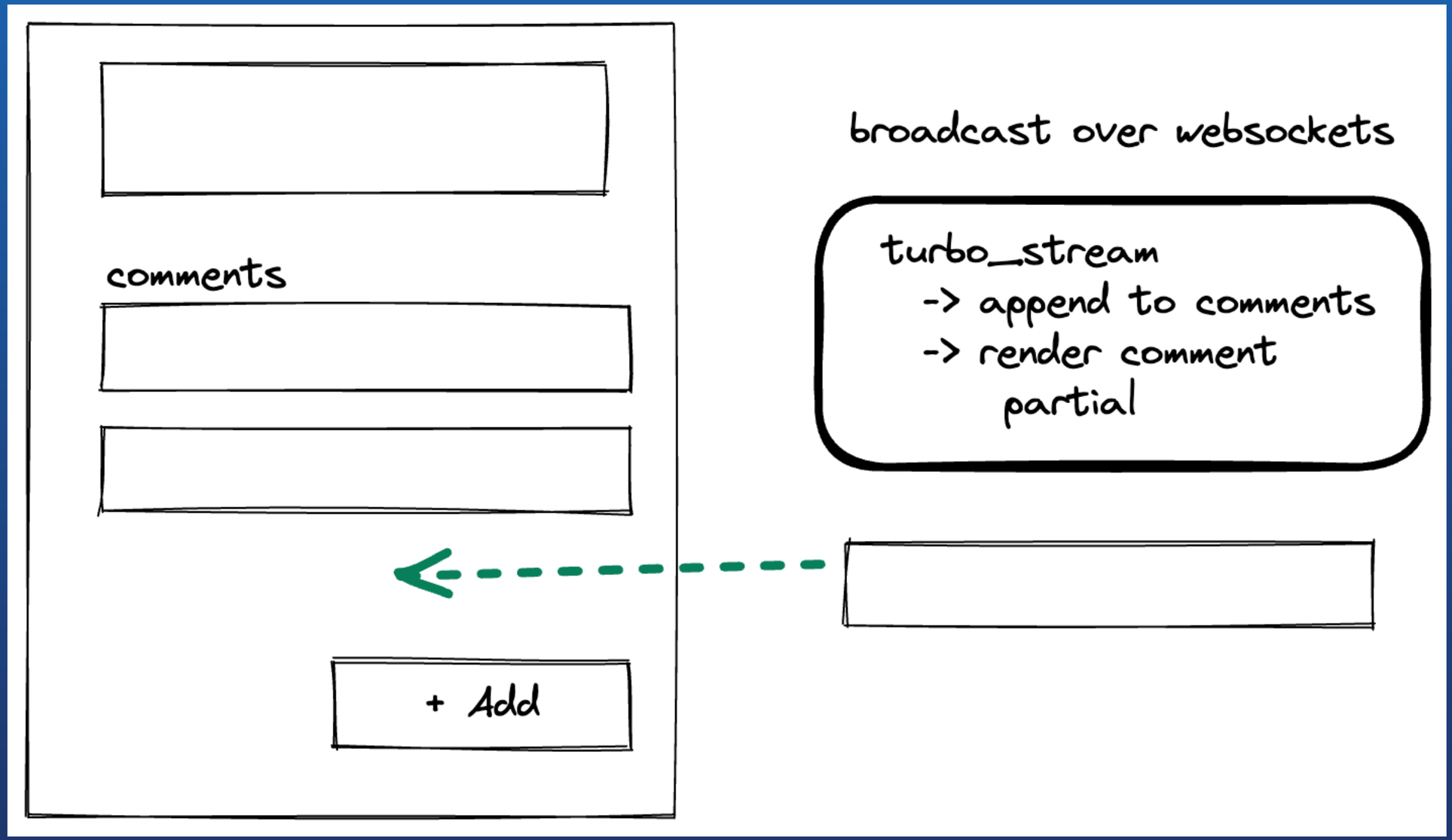
DELETE comments#destroy

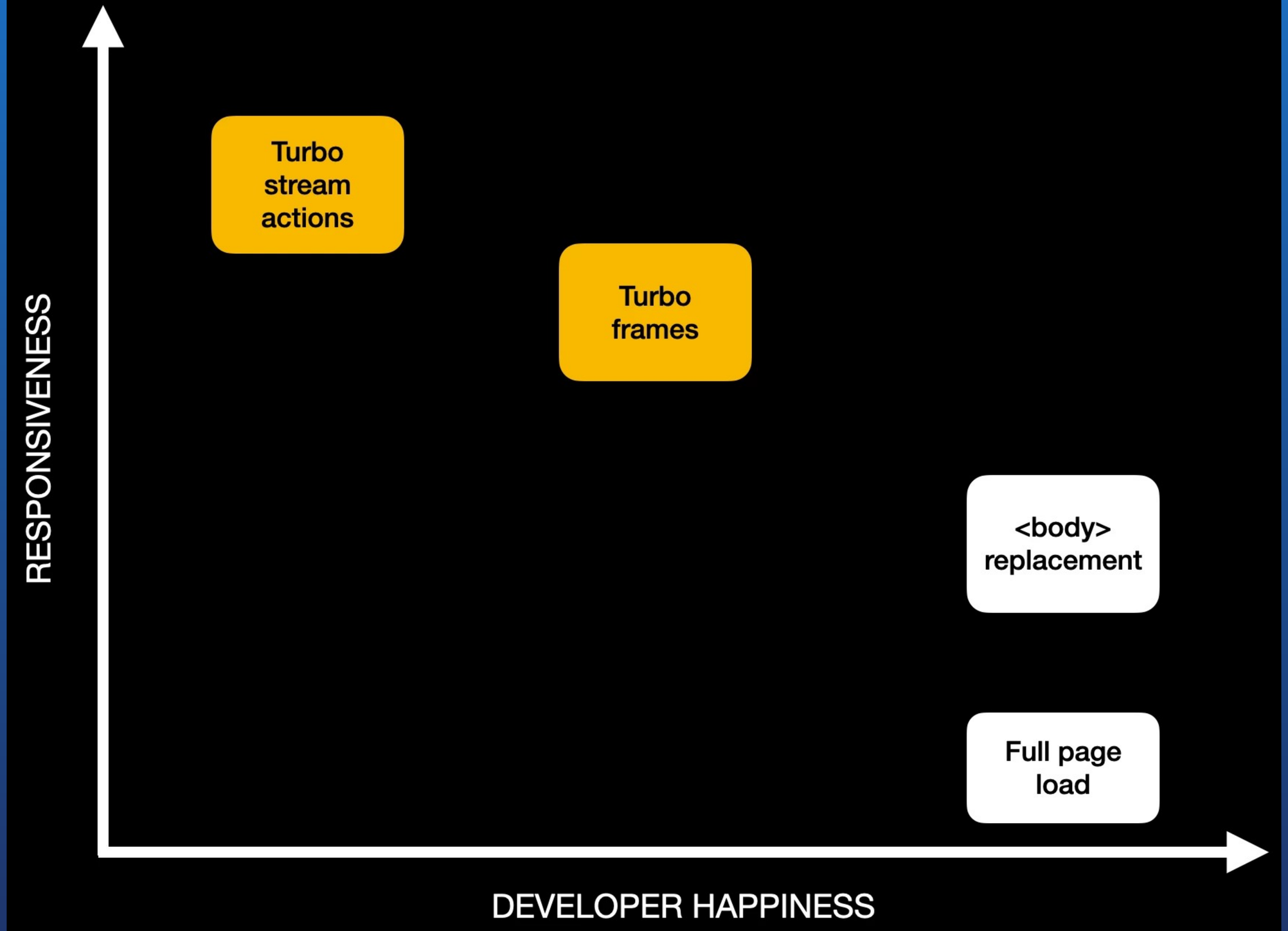
```
turbo_stream
  -> remove
  -> comment_456
```

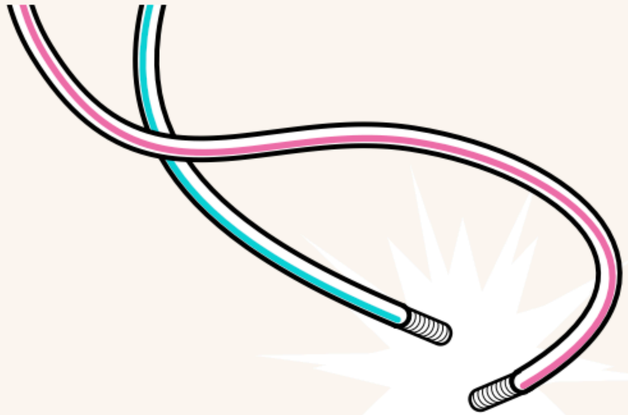


Turbo Stream

Broadcast with ActionCable




RESPONSIVENESS
Turbo stream actions
Turbo frames
<body> replacement
Full page load
DEVELOPER HAPPINESS



STIMULUS

While Turbo usually takes care of at least 80% of the interactivity that traditionally would have required JavaScript, there are still cases where a dash of custom code is required. Stimulus makes this easy with a HTML-centric approach to state and wiring.

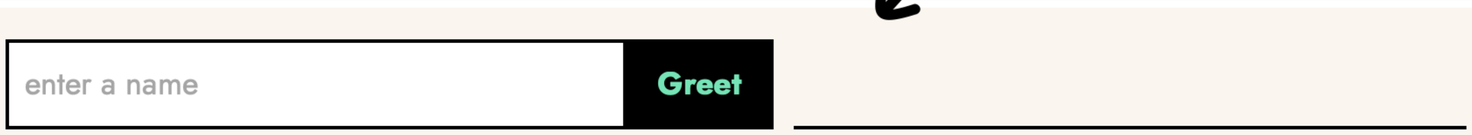
A modest JavaScript framework for the HTML you already have.

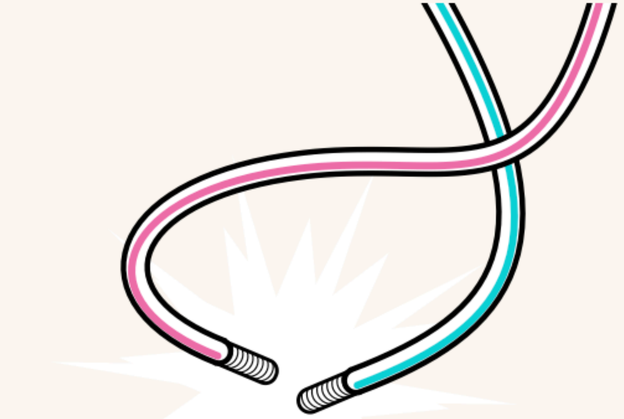
Sprinkle your HTML with controller, target, and action attributes:

```html
<!--HTML from anywhere-->
<div data-controller="hello">
  <input data-hello-target="name" type="text">

  <button data-action="click->hello#greet">
    Greet
  </button>

  <span data-hello-target="output">
  </span>
</div>
```

Write a compatible controller and watch Stimulus bring it to life:

```js
// hello_controller.js
import { Controller } from "stimulus"

export default class extends Controller {
  static targets = [ "name", "output" ]

  greet() {
    this.outputTarget.textContent =
      `Hello, ${this.nameTarget.value}!`
  }
}
```





STRADA

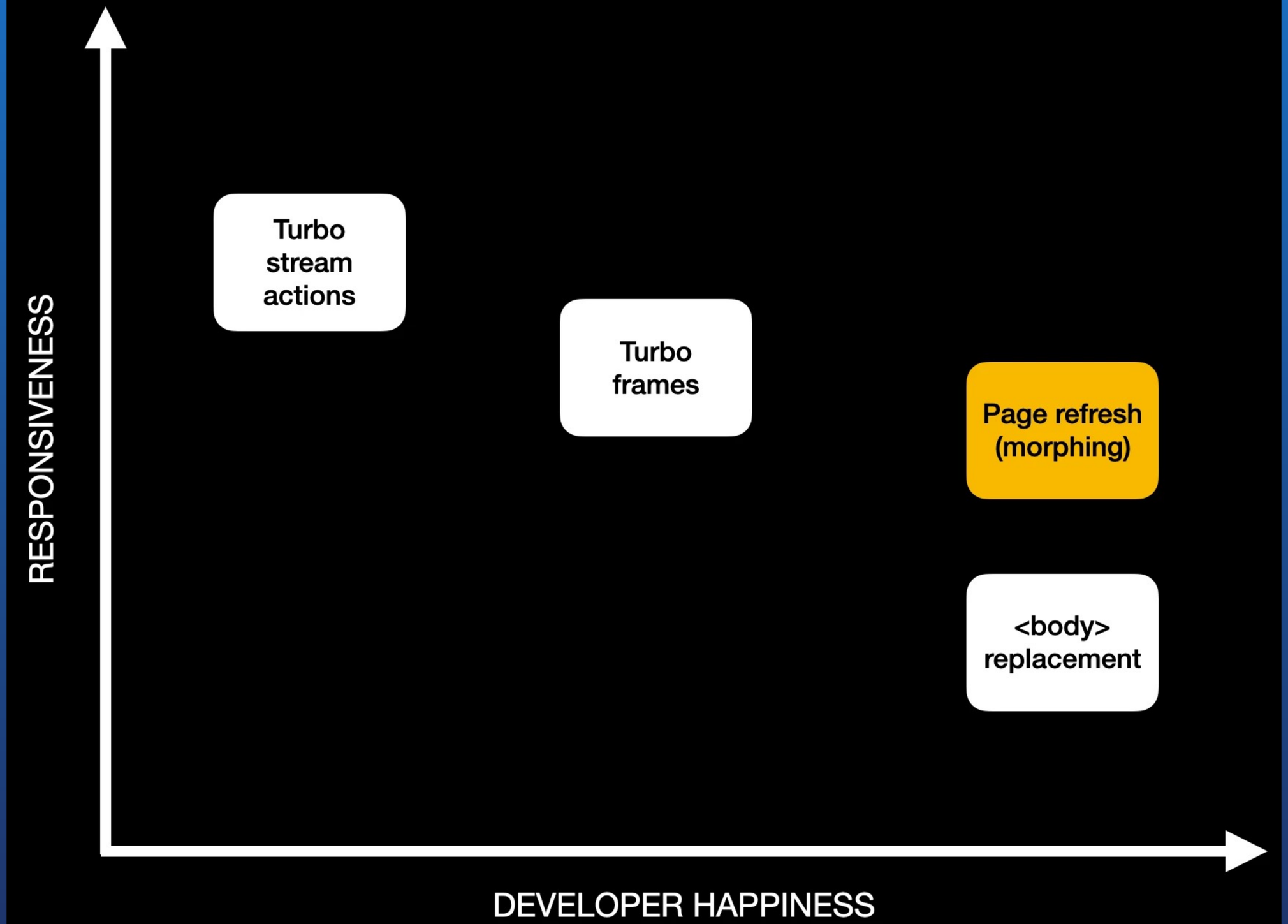
Standardizes the way that web and native parts of a mobile hybrid application talk to each other via HTML bridge attributes. This makes it easy to progressively level-up web interactions with native replacements.

Strada

- Enables you to create fully native controls in your hybrid mobile apps, driven by the web. Build web components and native components that work together in WebView screens to elevate your Turbo Native apps to the next level.
- **Strada Web** enables you to use your existing HTML to create web components that send and respond to messages with components in your native apps.
- **Strada iOS** and **Strada Android** enable you to create native components that receive and reply to messages from web components that are present on the page. All without writing any JavaScript in your native apps.

Turbo 8 Morphing

Beta


RESPONSIVENESS
Turbo stream actions
Turbo frames
Page refresh (morphing)
<body> replacement
DEVELOPER HAPPINESS

Ruby is great option for web development

Where to start

- https://rubyonrails.org/
- https://guides.rubyonrails.org/
- https://api.rubyonrails.org/
- https://hotwired.dev/

Live demo?

THANK YOU!

THANK YOU!

QUESTIONS?

https://github.com/vlado
vlado@cingel.hr

BONUS SLIDE :)

https://www.destroyallsoftware.com/talks/wat