x!

px

Aplicaes Web com Ruby On Rails

Com.pensar 2016

Luiz Alberto Ferreira Gomes

Curso de Cincia da Computao

8 de abril de 2017

Aplicao Web (1)

- Executada pelos usurios via um endereo de um servidor web na rede
- Utiliza um navegador (em ingls: *browser*) para iniciar sua execuo
- Consiste de uma coleo de *scripts* no cliente e no servidor, pginas HTML, folhas de estilos e etc.
  - outros recursos que podem estar espalhados por vrios servidores.
- Exemplos: webmail, lojas virtuais, homebanking, wikis, blogs e etc.

Aplicao Web (2)

- H um pouco mais do que isso:
  - Rede de Computadores:
    - a Internet, um sistema global de redes de computadores interconectadas.
    - utiliza o conjunto de protocolos TCP/IP.
  - Web (World Wide Web):
    - um sistema de documentos (em ingls: *web pages*) vinculados que so acessados atravs da Internet via protocolo HTTP.
    - Web pages contm documentos hypermedia: textos, grficos, imagens, vdeos e outros recursos multimdia, juntamente com *hiperlinks* para outras pginas
    - Hiperlinks formam a estrutura bsica da Web.
    - A estrutura da Web a que a torna til e de valor.

px

Aplicao Web (3)

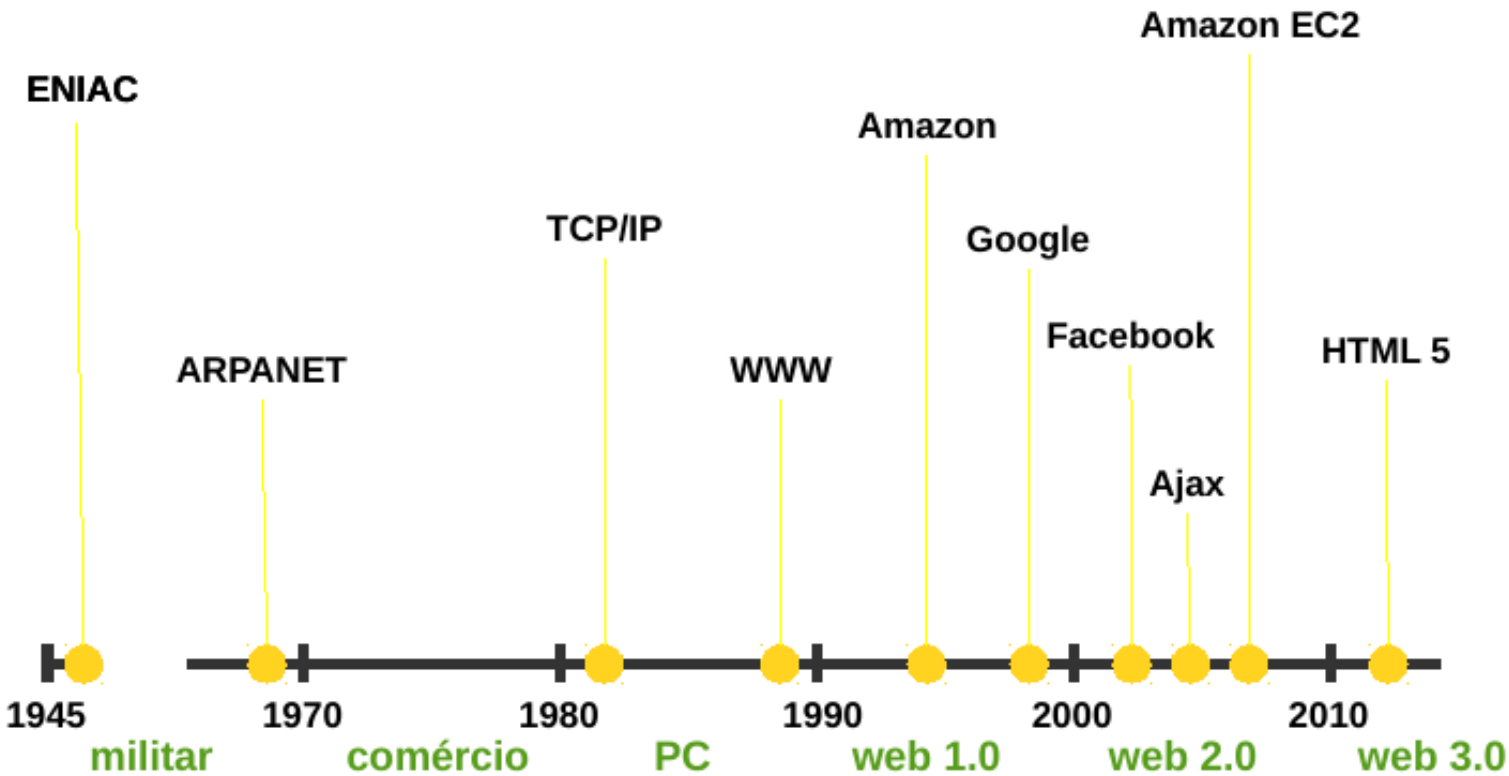
- Vantagens:
  - Convenincia pela utilizao um web browser como cliente.
  - Compatibilidade inerente entre plataformas.
  - Habilidade de atualizar e manter as aplicaes web sem instalao e distribuio de software em vrios clientes em potencial.
  - Reduo dos custos de TI.
- Desvantagens:
  - Interfaces com usurio ainda no so to boas quanto as das aplicaes tradicionais.
  - Maior risco de comprometimento da privacidade e segurana dos dados.

px

Aplicao Web (4)

- Mais difcil de desenvolver e depurar do que uma aplicao tradicional, pois existem mais partes a se considerar.

Histrico


ENIAC
ARPANET
TCP/IP
WWW
Amazon
Google
Facebook
Ajax
Amazon EC2
HTML 5
1945
1970
1980
1990
2000
2010
militar
comércio
PC
web 1.0
web 2.0
web 3.0

px

Web 1.0, 2.0 e 3.0

- Web 1.0 : pginas estticas e primeiros modelos de negcios.
- Web 2.0 : interactividade(Ajax), redes sociais e comrcio eletrnico.
- Web 3.0 : 'Web Inteligente', interpretao da informao auxiliada por mquina
  - exemplo: sistemas de recomendao.
- Base tecnolgica da Web 2.0 e 3.0.
  - javascript, xml, json(ajax).
  - interoperabilidade via Web Services.
  - infraestrutura via modelos de computao em nuvem (IAAS, PAAS e SAAS)
  - aplicaes mveis

Modelos de Computao em Nuvem (1)

- IAAS (Infraestructure As A Service) : fornece a insfraestrutura computacional fsica ou mquinas virtuais e outros recursos discos, firewalls, endereos IP e etc.
  - exemplos: Amazon EC2, Windows Azure, Google Compute Engine.
- PAAS (Platform as a Service) : fornece plataformas computacionais que tipicamente incluem sistemas operacionais, ambientes para execuo de programas, bancos de dados, servidores web e etc.
  - exemplos: AWS Elastic Beanstalk, Windows Azure, Heroku e Google App Engine

px

Modelos de Computao em Nuvem (2)

- SAAS (Software as a Service) : fornece acesso sob demanda s aplicaes de software, sem que o usurio tem que se preocupar com sua instalao, configurao e execuo.
  - exemplos: Google Apps e Microsoft 365.

px

Arquiteturas de Aplicaes Web (1)

- As aplicaes web modernas envolvem uma quantidade significativa de complexidade.
  - especialmente no lado do servidor.
- Uma tpica aplicao web envolve inmeros protocolos, linguagens de programao e tecnologias que compem a pilha de tecnologia web.
- Desenvolver, manter e ampliar uma aplicao web complexa difcil.
  - mas, construindo-o usando uma base de princpios de slidos de projeto pode-se simplificar cada uma dessas tarefas.

px

Arquiteturas de Aplicaes Web (2)

- Engenheiros de software usam abstraes para lidar com este tipo de complexidade.
  - *Design patterns* fornecem abstraes teis para sistemas orientados a objetos.

Design Patterns (1)

Definio (Design Patterns)

Um padro de projeto uma descrio da colaborao de objetos que interagem para resolver um problema de software em geral dentro de um contexto particular.

- Um design pattern um modelo abstrato que pode ser aplicado recorrentemente.
- A idia aplicar padres de projeto, a fim de resolver problemas especficos que ocorrem durante a construo de sistemas reais.
- Os padres de projeto fornecem uma maneira de comunicar as solues em um projeto, ou seja, a terminologia que engenheiros de software usam para falar sobre projetos.

Modelo Cliente-Servidor (1)

- A arquitetura cliente-servidor a arquitetura mais bsica para descrever a cooperao entre os componentes de uma aplicao web.
- A arquitetura cliente-servidor pode ser subdividia em:
  - servidor que "escuta"por requisies e fornece os servios ou recursos de acordo com cada uma.
  - cliente que estabelece a conexo com o servidor para requisitar servios ou recursos.
- Existe um protocolo request/response associado com qualquer arquitetura cliente-servidor.

Modelo Cliente-Servidor (2)

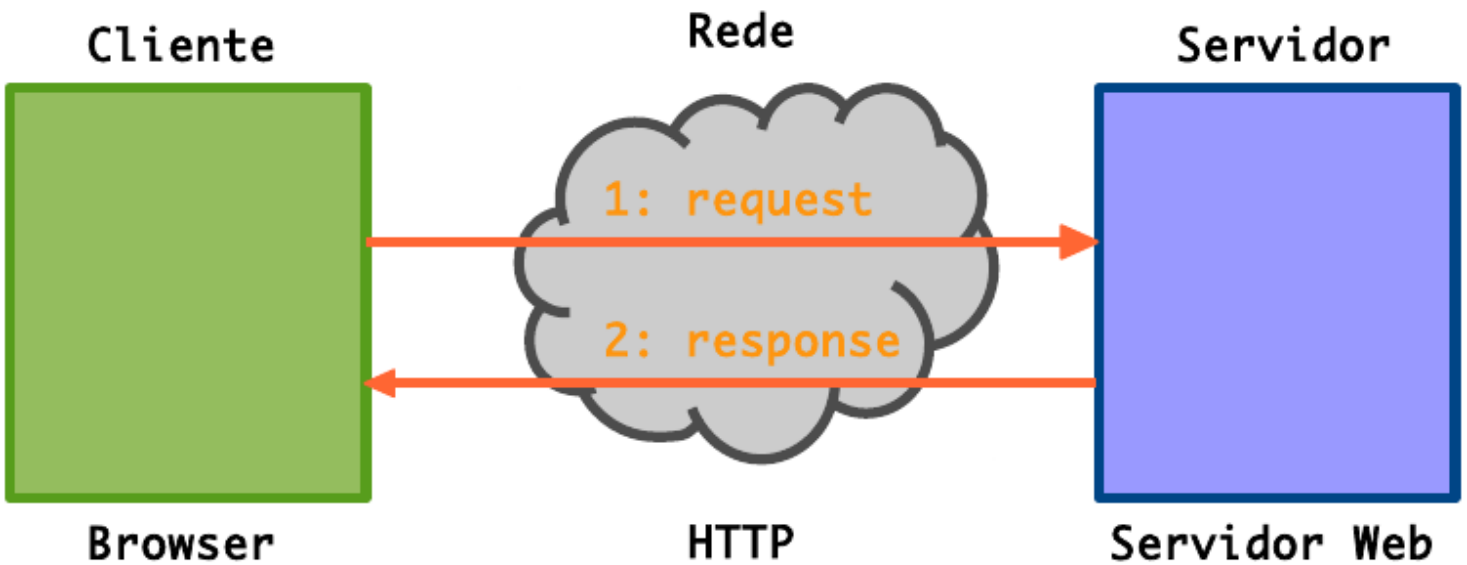


Figura: Arquitetura cliente servidor.

px

Modelo Cliente-Servidor (3)

- sem dvida o padro de projeto de arquitetura mais conhecido
- O ponto chave de uma arquitetura cliente-servidor distribuir os componentes de uma aplicao entre o cliente o servidor de alguma forma.
  - o servidor realiza as tarefas, consultas e transaes
  - o cliente fica com uma responsabilidade menor: a de receber informaes
- A fim de construir aplicaes web complexas, vrios design patterns ajudam a organizar como peas so dispostas dentro da arquitetura cliente-servidor.

px

Arquitetura N-Tier (1)

Definio (Arquitetura N-Tier)

A arquitetura n-tier um *design pattern* muito til que estrutura o modelo cliente-servidor.

- Este padro de projeto baseado no conceito de quebrar um sistema em partes diferentes ou camadas que podem ser separados fisicamente:
  - cada camada responsvel por fornecer uma funcionalidade especfica ou coesa.
  - uma camada apenas interage com as camadas adjacentes a ela por meio de uma estrutura bem definida por meio de interfaces.

px

Arquitetura N-Tier (2)

Exemplos (Arquitetura 2-Tier)

- Servidores de impresso
- Aplicaes web antigas:
  - Interface com o usurio (navegador) residia no cliente (thin).
  - Servidor fornecia as pginas estticas (HTML).
  - Interface entre os dois via *Hypertext Transfer Protocol* (HTTP).

- Camadas adicionais aparecem quando a funcionalidade do aplicativo ainda mais dividida.
- Quais so as vantagens de um tal projeto?
  - A abstrao fornece um meio para gerenciar a complexidade.

Arquitetura N-Tier (3)

- Camadas podem ser atualizados ou substitudos de forma independente a medida que os requisitos ou tecnologia.
  - a nova s precisa usar as mesmas interfaces que a antiga utilizada.
- Ele fornece um equilbrio entre inovao e padronizao.
- Sistemas tendem a ser muito mais fceis de construir, manter e atualizar.

px

Arquitetura 3-Tiers (1)

- Uma das mais comuns a arquitetura em 3 camadas:
  - Apresentao
    - a interface com o usurio.
  - Aplicao (lgica)
    - recupera modifica e/ou exclui dados na camada de dados, e envia os resultados do processamento para a camada de apresentao.
  - Camada de dados
    - a fonte dos dados associados ao aplicativo.
- As aplicaes web modernas frequentemente so construdas utilizando uma arquitetura em 3 camadas:
  - Apresentao
    - o navegador web do usurio.

px

Arquitetura 3-Tiers (2)

- Aplicao (lgica)
  - o servidor web e lgica associada com gerao de contedo web dinmico.
  - por exemplo, a coleta e formatao do resultados de uma pesquisa.
- Camada de dados
  - um banco de dados.

px

Agenda

px

Ruby on Rails (1)

- Ruby on Rails (Rails) framework construdo na linguagem Ruby para o desenvolvimento de aplicaes web
  - Rails fornecido em uma gem Ruby (gem um pacote Ruby)
- Rails fornece uma extenso conjunto de geradores de cdigo e scripts de automao de testes
- Um conjunto de ferramentas adicionais so fornecidos como parte do ecossistema Rails:
  - Rake - utilitrio similar ao **make do Unix** para criar e migrar bancos de dados, limpar sesses de uma Web app
  - WEBrick - servidor web de desenvolvimento para execuo de aplicaes Rails

Ruby on Rails (2)

- SQLite - um servidor de banco de dados simples pr-instalado como o Rails
- Rack Middleware - interface padronizado para interao entre um servidor web e uma Web App

- Algumas empresas que utilizam Rails: Twitter, Hulu, GitHub, Yellow Pages e etc

px

Filosofia do Rails (1)

- Ruby on Rails 100% open-source, disponvel por meio da MIT License: ⟨http://opensource.org/licenses/mit-license.php⟩.
- Conveno acima da Configurao (em ingls: *Convention over Configuration* (CoC))
  - se nomeao segue certas convenes, no h necessidade de arquivos de configurao.

Exemplo:

```
FilmesController#show -> filmes_controler.rb
FilmesController#show -> views/filmes/show.html.e
```


px

Filosofia do Rails (2)

- "Don't Repeat Yourself" (DRY) sugere que escrever que o mesmo cdigo vrias vezes uma coisa ruim
- O *Representational State Transfer* (REST) o melhor padro para desenvolvimento de aplicaes web
  - organiza a sua aplicao em torno de recursos e padres HTTP (verbs)

Histrico (1)

- David Heinemeier Hanson derivou o Ruby on Rails a partir do BaseCamp – uma ferramenta de gesto de projetos da empresa 37Signals.
  - a primeira verso de cdigo aberto (em ingls: *open source*)foi liberada em julho de 2004.
  - mas direitos para que outros desenvolvedores colaborassem com o projeto foram liberados em fevereiro de 2005.
- Em agosto de 2006, o Ruby on Rails atingiu um marco importante quando a Apple dicidiu distribudo juntamente com a verso do seu sistema operacional Mac OS X v10.5 "Leopard"
  - nesse mesmo no o Rails comeou a ganhar muita ateno da comunidade de desenvolvimento web.

Histrico (2)

- Rails utilizado por diversas companhias, como por exemplo:
  - Airbnb, BaseCamp, Disney, GitHub, Hulu, Kickstarter, Shopify e Twitter.

px

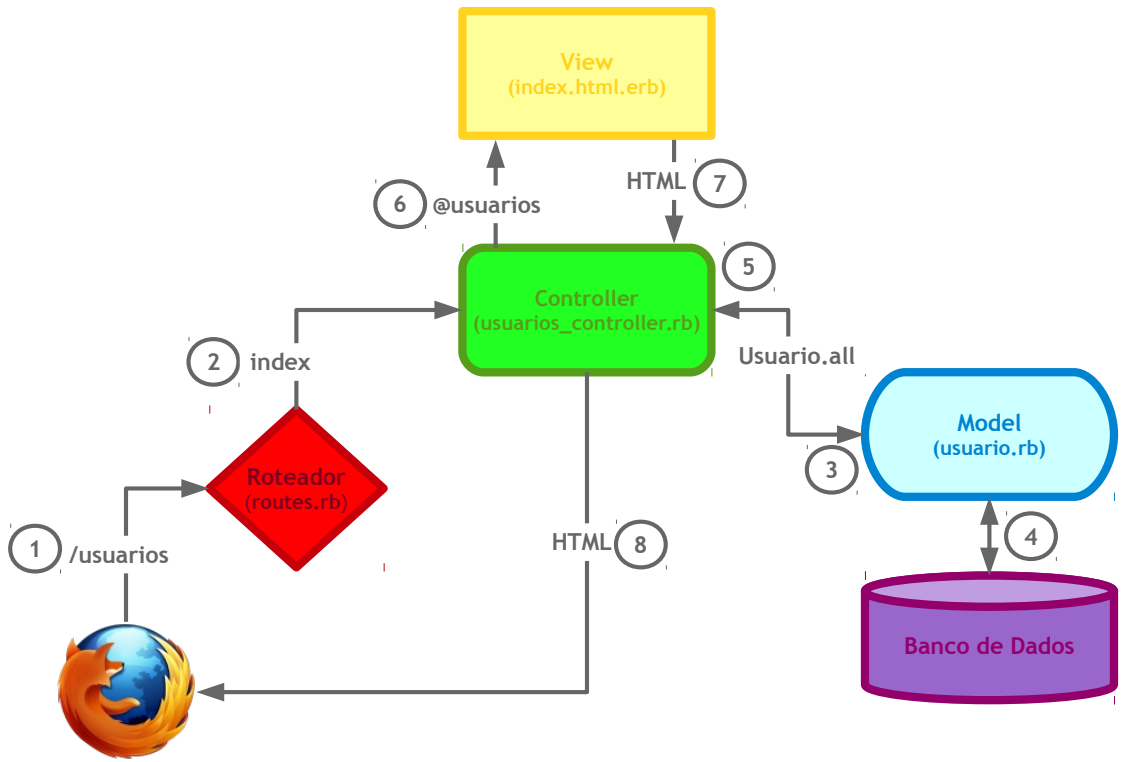
Histrico (3)

Verso	Data
1.0	13 de dezembro de 2005
1.2	19 de janeiro de 2007
2.0	07 de dezembro de 2007
2.1	01 de junho de 2008
2.2	21 de novembro de 2008
2.3	16 de maro de 2009
3.0	29 de agosto de 2010
3.1	31 de agosto de 2011
3.2	20 de janeiro de 2012
4.0	25 de junho de 2013
4.1	08 de abril de 2014

Tabela: Evoluo histrica do Ruby on Rails

Model-View-Controller

- O framework Rails contrudo em cima do Design Pattern Model View Controller(MVC):



px

Hora de Colocar a Mo na Massa

- Conecte-se na mquina com o usurio `a1550099999` e senha 333333
  1. Inicie uma janela de terminal e digite no prompt: [style=BashInputBasicStyle] $rails new my_app$
  2. Mude para o diretrio da aplicao (RAILS.root) [style=BashInputBasicStyle] $cd new my_app$
  3. Execute o servidor web embutido: [style=BashInputBasicStyle] *rails s*
  4. Abra uma janela do navegador e digite: [style=BashInputBasicStyle] $http://localhost:3000$

px

Estrutura de uma Aplicao Rails (1)

Arquivo/Pasta	**Descrio**
app	Arquivos contendo os principais cdigos da aplicao, incluindo modelos, vises, controladores e auxiliares(*helpers*)
app/assets	Arquivos contendo folhas de estilos (CSS), cdigos Javascript e imagens da aplicao
bin	Arquivos ou scripts executveis
config	Configuraes da aplicao
db	Migraes, esquema e outros arquivos relacionados ao banco de dados
doc	Documentao do sistema
lib	Bibliotecas auxiliares
lib/assets	Arquivos contendo folhas de estilos (CSS), cdigos Javascript e imagens das bibliotecas

px

Estrutura de uma Aplicao Rails (2)

Arquivo/Pasta	Descrio
log	Informaes de log
public	Pginas que podem ser acessadas publicamente via navegador, tais como pginas de erros
test	Testes da nossa aplicao
tmp	Arquivos temporrios como cache e informaes de sesses
vendor	Dependncias e bibliotecas de terceiros
vendor/assets	Arquivos contendo folhas de estilos (CSS), cdigos Javascript e imagens de terceiros
README.rdoc	Uma breve descrio da aplicao
Rakefile	Tarefas que podem ser executadas pelo comando rake
Gemfile	Pacotes(gems) necessrios para a aplicao
Gemfile.lock	Uma lista de gems utilizadas para garantir que todas as cpias da aplicao utilizam as mesmas verses de gems
config.ru	Um arquivo de configurao para o Rack Middleware
.gitignore	Define de arquivos ou padres de arquivos que devero ser ignorados pelo Git

Agenda

px

Especificao do Blog App (1)

1. Blog uma contrao de "weblog", um site de discusso ou troca de informaes publicado na Web.
2. Existem dois tipos de participantes: o administrador e o usurio
3. O administrador do blog deve ser capaz de entrar novas postagens, tipicamente em ordem cronolgica inversa.
4. Os usurios devem ser capazes de visitar o blog e escrever comentrios sobre as postagens.
5. O administrador do blog deve ser capaz de modificar e ou remover qualquer postagem ou comentrio.
6. Os usurios no devem ser capazes de modificar postagens ou comentrios de outros usurios.

Passos Iniciais do Blog App (1)

1. Inicie uma janela de terminal e digite no prompt: [style=BashInputBasicStyle] *cd* rails new blog
2. Utilize o gerador `scaffold` para criar os componentes MVC para as postagens e os comentrios [style=BashInputBasicStyle] *railsgeneratescaffoldpost title* : *stringbody* : *text* rails generate scaffold comment post¡*d* : *integer body* : *text*
3. Gere as tabelas `post` e `comment` no banco de dados [style=BashInputBasicStyle] *rakedb* : *migrate*
4. Visualize todas as URLs reconhecidas pela sua aplicao digitando: [style=BashInputBasicStyle] *rakeroutes*

px

Passos Iniciais do Blog App (2)

5. Inicie o servidor web embutido: [style=BashInputBasicStyle] *railss*
6. Abra uma janela do navegador e digite: [style=BashInputBasicStyle] *http* : //*localhost* : 3000/*posts*

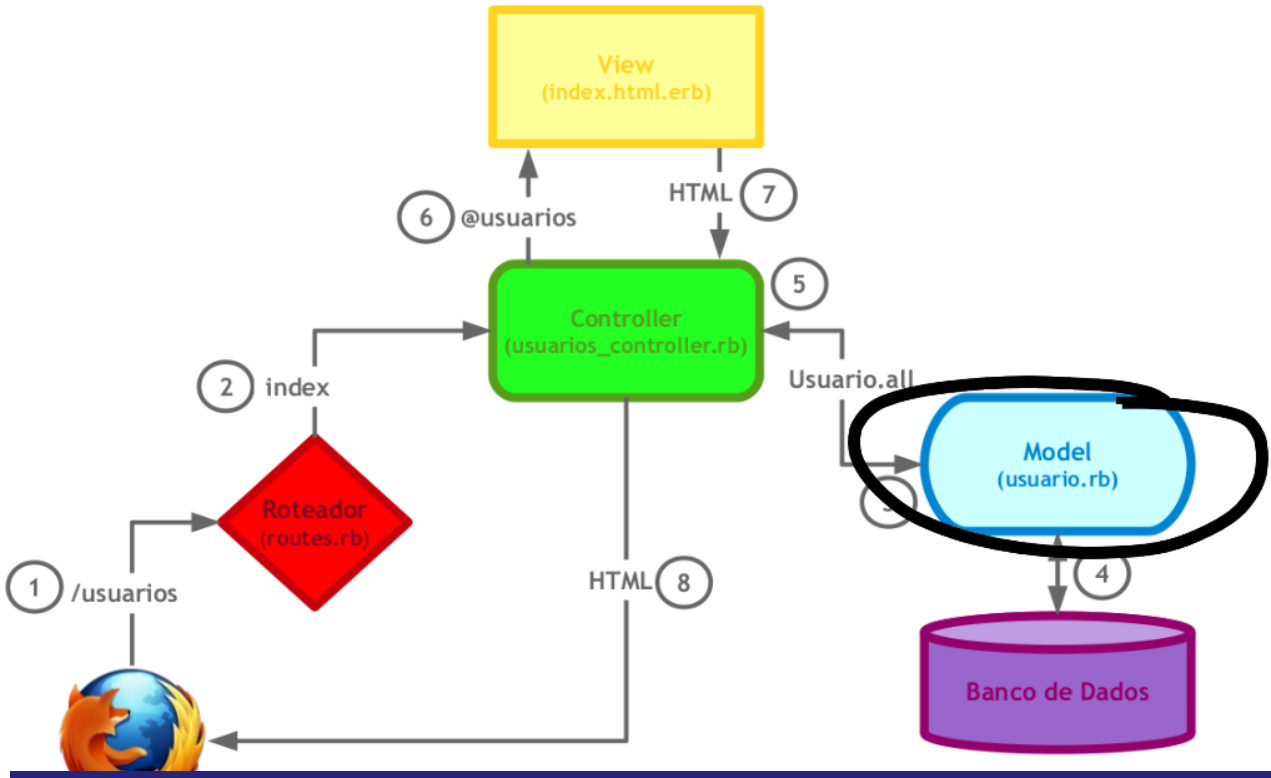
px

Agenda

px

Model Component

- O modelo gerencia os dados, a lgica e as regras de neggcios da aplicao.



px

Banco de Dados Relacionais (1)

- Um aspecto importante da programao web a habilidade de coletar, armazenar e recuperar diferentes formas de dados
  - uma das formas mais populares so os bancos de dados relacionais
- Um banco de dados relacional baseado entidades, denominadas tabelas, no relacionamento, associaes, entre elas
- O continer fundamental em um banco de dados relacional denominado de database ou schema
  - podem incluir estruturas de dados, os dados propriamente ditos e permisses de acesso

Banco de Dados Relacionais (2)

- Os dados so armazenados em tabelas e as tabelas so divididas em linhas e colunas. Por exemplo:

Tabela: comment

id	post_id	body
10	1	Ruby realmente...
11	2	Rails facilita...
13	2	Concordo, ...

px

Banco de Dados Relacionais (3)

- Relacionamentos so estabelecidos entre tabelas para que a consistncia dos dados seja mantida em qualquer situao e podem ser:
  - 1:1
  - 1:N
  - N:M

Tabela: comment

id	post_id	body
10	1	Ruby realmente...
11	2	Rails facilita...
13	2	Concordo, ...

Tabela: post

id	title	body
1	A Linguagem Ruby	Ruby legal.
2	O Framework Rais	O Rais facilita...

px

Banco de Dados Relacionais (4)

px

SQLite (1)

- O banco de dados que o Rails utiliza em diversos ambientes (desenvolvimento, teste e produo) especificado em: config/database.yml

[style=RubyInputStyle, firstline=7]codigos/blog$_1$/*config*/*database.yml*

- Rails usa por padro o SQLite como gerenciador padro
  - relacional, embutido, sem servidor, configurao zero, transacional, suporta SQL

px

SQLite (2)

ATENO: SQLite no um banco de dados para produo !

- Banco de dados de produo populares: MySQL e PostgreSQL

px

Database Console

- O comando rails db fornece uma console para acesso aos bancos dados MySQL, PostgreSQL e SQLite.

[style=BashInputBasicStyle, basicstyle=, keepspaces=true]

*railsdbSQLiteversion*3.8.7.12014 − 10 − 2913 : 59 : 56*Enter*" .*help*" *forusagehints*.*sqlite* > .*headersonsqlite* >

.*modecolumnssqlite* > *select* ∗ *fromposts*; *idtitlebodycreated*$_a$*tupdated*$_a$*t* − − − − − − − − − − − − − − − − − −

− −

− −5*ALinguagemRubyRubyelegal*.2016 − 04 − 3022 :

45 : 20.6363632016 − 04 − 3022 : 45 : 20.636363*sqlite* >

- Dica: utilize headers on e mode coluns

px

Hora de Colocar a Mo na Massa (1)

- Inicialize na pasta da aplicao a console do banco de dados e configure a sua exibio: [style=BashInputBasicStyle] *railsdbsqlite* > *.headersonsqlite* > *.modecolumns*
- Exiba os colunas da tabela `posts`: [style=BashInputBasicStyle] sqlite¿ .schema posts

px

Hora de Colocar a Mo na Massa (2)

- Crie um novo `post` e salve no banco de dados: [style=BashInputBasicStyle] sqlite¿ INSERT INTO posts (title, body, created$_at$, *updated*$_a t$)*VALUES*(”*Com.pensar*2016”,”*Temvarioscurso*
  05 – 0319 : 50 : 00”,”2016 – 05 – 0319 : 50 : 00”);
- Exiba todos os `posts`: [style=BashInputBasicStyle] sqlite¿ SELECT * FROM posts;
- Exiba todos os `posts` ordenados pelo ttulo (title): [style=BashInputBasicStyle] sqlite¿ SELECT * FROM posts ORDER BY title;

px

Hora de Colocar a Mo na Massa (3)

- Exiba um `post`: [style=BashInputBasicStyle] sqlite¿ SELECT * FROM posts LIMIT 1
- Exiba o `post` cujo `id` 2: [style=BashInputBasicStyle] sqlite¿ SELECT * FROM posts WHERE id=2;
- Atualize o ttulo `post` cujo o `id` 2: [style=BashInputBasicStyle] sqlite¿ UPDATE posts SET title="Novo titulo"WHERE id=2;
- Remova `post` cujo o `id` 2: [style=BashInputBasicStyle] sqlite¿ DELETE FROM posts WHERE id=2;

px

Migrations (1)

- Como podemos rastrear e desfazer alteraes em um banco de dados?
- No existe uma maneira fcil - manualmente confuso e propenso a erros.
- Tipicamente, comandos SQL so dados para criar e modificar tabelas em um banco de dados
- Mas se houver a necessidade de trocar o banco de dados "durante o voo"?
  - por exemplo, desenvolve-se em SQLite e implanta-se em MySQL.

SOLUO: Migrations

px

Migrations (3)

- A cada vez que o scaffold executado na aplicao, o Rails cria um arquivo de migration de banco de dados. Este arquivo armazenado em db/migrate
- Por exemplo: o arquivo 20160430140114_create_posts.rb

[style=RubyInputStyle]codigos/blog$_1$/*db*/*migrate*/20160430140114$_c$*re*

- Rails utiliza o comando rake para executar os migrations e fazer as alteraes no banco de dados.

[style=BashInputBasicStyle] *rakedb* : *migrate*

px

Object-Relational Mapping (1)

- Um ORM preenche a lacuna entre banco de dados relacionais e as linguagens de programao orientadas a objetos
- Simplifica bastante a escrita de cdigos para acessar o banco de dados.
- Tipicamente, comandos SQL so dados para criar e modificar tabelas em um banco de dados
- No Rails, o Model do MVC utiliza algum framework de ORM

Active Record (1)

- ActiveRecord o nome do ORM padro do Rails?

[style=RubyInputStyle, caption=app/models/post.rb]codigos/blog/app/models/post.rb

Onde est cdigo ?
R: Metaprogramao + Conveno

- Para que "mgica" ocorra:
  - o ActiveRecord tem que saber como encontrar o banco de dados (ocorre via config/database.yml)
  - **(Conveno)** existe uma tabela com o nome no plural da subclasse `ActiveRecord::Base`
  - **(Conveno)** espera-se que a tabela tenha uma chave primrio denominada id

px

Object-Relational Mapping (1)

- Um ORM preenche a lacuna entre banco de dados relacionais e as linguagens de programao orientadas a objetos
- Simplifica bastante a escrita de cdigos para acessar o banco de dados.
- Tipicamente, comandos SQL so dados para criar e modificar tabelas em um banco de dados
- No Rails, o Model do MVC utiliza algum framework de ORM

Hora de Colocar a Mo na Massa (1)

- Inicialize na pasta da aplicao a console do Rails (no a do banco de dados): [style=BashInputBasicStyle] *railsc*
- Exiba os atributos da classe `Post`: [style=BashInputBasicStyle] irb(main):004:0¿ Post.column$_n$*ames*
- Crie um novo `post` e salve no banco de dados: [style=BashInputBasicStyle] irb(main):005:0¿ p1 = Post.new irb(main):006:0¿ p1.title=”Temperatura em Pocos”irb(main):007:0¿ p1.body=”Esta muito frio...”irb(main):008:0¿ p1.save
- Exiba todos os `posts`: [style=BashInputBasicStyle] irb(main):007:0¿ Post.all

px

Hora de Colocar a Mo na Massa (2)

- Exiba todos os `posts` ordenados pelo ttulo (title): [style=BashInputBasicStyle] irb(main):007:0¿ Post.all.order(title: :asc)
- Exiba um `post`: [style=BashInputBasicStyle] irb(main):007:0¿ Post.first
- Exiba o `post` cujo `id` 2: [style=BashInputBasicStyle] irb(main):007:0¿ $\text{Post.find}_b y(id : 2)$
- Atualize o ttulo do primeiro `post`: [style=BashInputBasicStyle] irb(main):007:0¿ p1=Post.first irb(main):008:0¿ p1.update(title: "um novo titulo")
- Remova do primeiro `post`: [style=BashInputBasicStyle] irb(main):007:0¿ p1=Post.first irb(main):008:0¿ p1.destroy

px

Validao em Aplicaes Web

- Validao de Dados o processo para garantir que a aplicao web operem corretamente. Exemplo:
  - garantir a validao do e-mail, nmero do telefone e etc
  - garantir que as "regras de negcios"sejam validadas
- A vulnerabilidade mais comum em aplicao web a injeo SQL

Client Side

- Envolve a verificao de que os formulrios HTML sejam preechidos corretamente
  - JavaScript tem sido tradicionalmente utilizado.
  - HTML5 possui "input type"especficos para checagem.
  - Funciona melhor quando combinada com validaes do lado do servidor.

Server Side

- A validao feita aps a submisso do formulrio HTML
  - banco de dados(stored procedure) - dependente do banco de dados
  - no controlador - veremos mais tarde que no se pode colocar muita lgica no controlador (controladores magros)
  - no modelo - boa maneira de garantir que dados vlidos sejam armazenados no banco de dados (database agnostic)
  - Funciona melhor quando combinada com validaes do lado do servidor.

px

Validao em Rails (1)

- Objetos em um sistema OO como tendo um ciclo de vida
  - eles so criaddos, atualizados mais tarde e tambm destruidos.
- Objetos ActiveRecord tm mtodos que podem ser chamados, a fim de assegurar a sua integridade nas vrias fases do seu ciclo de vida.
  - garantir que todos os atributos so vlidos antes de salv-lo no banco de dados
- Callbacks so mtodos que so invocados em um ponto do ciclo de vida dos objetos ActiveRecord
  - eles so "ganchos"para gatilhos para acionar uma lgica quando houver alteraes de seus objetos

px

Validao em Rails

- Validations so tipo de callbacks que podem ser utilizados para garantir a validade do dado em um banco de dados
- Validao so definidos nos modelos. Exemplo: [style=RubyInputStyle] class Person ¡ ActiveRecord::Base validates$_p$*resence*$_o$*f* : *name*validates$_n$*umeracality*$_o$*f* : *age*, : *only*$_i$*nteger* => *true*validates$_c$*onfirmation*$_o$*f* : *email*validates$_l$*ength*$_o$*f* : *password*, : *in* => 8..20*end*

px

Hora de Colocar a Mo na Massa

- Modifique o arquivo `app/models/post.rb` para exigir que o usurio digite o ttulo e o texto do blog: [style=RubyInputStyle] class Post ¡ ActiveRecord::Base validates$_p$*resence*$_o$*f* : *title*, : *bodyend*
- Modifique o arquivo `app/models/comment.rb` para exigir que o usurio digite texto do comentrio blog: [style=RubyInputStyle] class Post ¡ ActiveRecord::Base validates$_p$*resence*$_o$*f* : *bodyend*
- Inicie o servidor web embutido e teste se a validao est funcionando

px

Associaes em Rails (1)

- O gerador `scaffold` utiliza por padro o ActiveRecord. Isto significa:
  - Tabelas para postagens e comentrios foram criadas quando executamos as migraes
  - Um conexo com o banco de dados estabelecida
  - O ORM configurado para as postagens e comenttios foi criado - o "M"do MVC.
- No entanto, uma coisa est faltando:
  - tem-se que assegurar que qualquer comentrios sejam associados s suas postagens

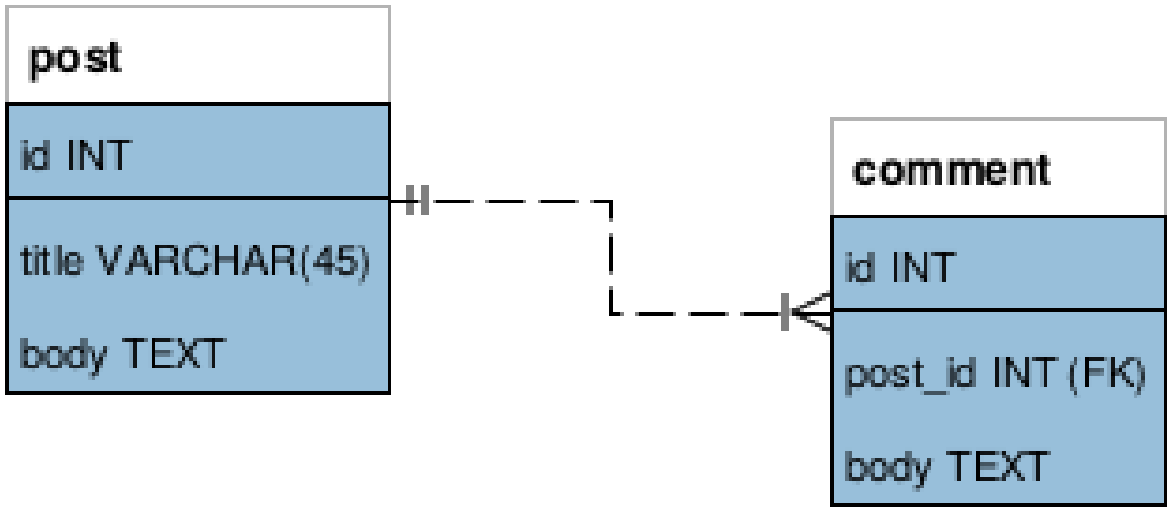
px

Associaes em Rails (2)

- Para tornar os modelos em Rails totalmente funcionais precisamos adicionar associaes:
  - cada postagem precisa saber os comentrios associado a ele
  - cada comentrio precisa saber qual a postagem ele pertence
- H uma relao muitos-para-um entre comentrios e postagens uma:

px

Associaes em Rails (3)



- O ActiveRecord contm um conjunto de mtodos de classe para vinculao de objetos por meio de chaves estrangeiras
- Para habilitar isto, deve-se declarar as associaes dentro dos modelos usando:

px

Associaes em Rails (4)

Associao	**Modelo Pai**	**Modelo Filho**
Um-para-um	`has_one`	`belongs_to`
Muitos-para-um	`has_many`	`belongs_to`
Muitos-para-muitos	`has_and_belongs_to_many`	*na tabela juno

px

Hora de Colocar a Mo na Massa (1)

- Modifique o arquivo `app/models/post.rb` para associar o post aos seus comentrio: [style=RubyInputStyle] class Post ¡ ActiveRecord::Base validates$_p$*resence*$_o$*f* : *title*, : *bodyhas*$_m$*any* : *commentsend*
- Modifique o arquivo `app/models/comment.rb` para associar o comentrio ao seu post: [style=RubyInputStyle] class Comment ¡ ActiveRecord::Base validates$_p$*resence*$_o$*f* : *bodybelongs*$_t$*o* : *postend*

px

Hora de Colocar a Mo na Massa (2)

- Crie um novo `post` e salve no banco de dados: [style=BashInputBasicStyle] irb(main):005:0¿ p1 = Post.new irb(main):006:0¿ p1.title="Associacao"irb(main):007:0¿ p1.body="Vinculando com o comentario"irb(main):008:0¿ p1.save
- Crie um novo `comment` e o vincule a um `post`: [style=BashInputBasicStyle] irb(main):005:0¿ c1 = Comment.new irb(main):006:0¿ c1.body="Comentario vinculado"irb(main):007:0¿ c1.save irb(main):008:0¿ p1.comments ¡¡ c1

px

Hora de Colocar a Mo na Massa (3)

- Consulte os comentrios do `post` p1:
  [style=BashInputBasicStyle] irb(main):005:0¿ p1.comments.all

- Consulte os comentrios 2 do `post` p1:
  [style=BashInputBasicStyle] irb(main):005:0¿
  p1.comments.where(id: 2)
- Consulte o `post` do comentrio c1:
  [style=BashInputBasicStyle] irb(main):005:0¿ c1.post

px

Agenda

px

Action Controller

- Um Action Controller classe Ruby contendo uma ou mais aes
- Cada ao responsvel pela resposta a uma requisio
- Quando uma ao concluda a viso de mesmo nome renderizada
- Uma ao deve estar mapeada no arquivo routes.rb que gerado pelo scaffold: [style=RubyInputStyle] Rails.application.routes.draw do resource :posts resource :comments end

px

Representational State Transfer

- Rails utilza Representational State Transfer(REST) para mapear os recursos (resources) de uma aplicao:
  - List todos os recursos disponveis
  - Show um recurso especfico
  - Destroy um recurso existente
  - Provide a way to create um novo recurso
  - Create um novo recurso
  - Provide a way to update um recurso existente
  - Update um recurso existente

px

Ao: Index (1)

- Ao que recupera todas as postagens do blog
- (Implicitamente) procura pelo template index.html.erb para renderizar a resposta [style=RubyInputStyle, caption=controllers/posts_controller.rb] Class PostsController ¡ ApplicationController
  GET /posts GET /posts.json def index @posts = Post.all end

px

Ao: Index (2)

- index.html.erb: [style=RubyInputStyle, caption=views/posts/index.html.erb] ... ¡tbody¿ ¡¡tr¿ ¡td¿¡¡td¿¡¡td¿¡¡td¿¡¡td¿¡¡/tr¿ ¡¡/tbody¿ ....

px

Ao: Show (1)

- Recupera uma postagem especfica no parmetro id passado como parte da URL
- (Implicitamente) procura pelo show.html.erb para renderizar a resposta [style=RubyInputStyle, caption=controllers/posts_controller.rb] Class PostsController ¡ ApplicationController before$_{a}ction$: $set_{p}ost$, $only$: [: $show$, : $edit$, : $update$, : $destroy$] GET /posts/1 GET /posts/1.json def show end private def set$_{p}ost$@$post$ = $Post.find(params[:id])end$

px

Ao: Show (2)

- show.html.erb: [style=RubyInputStyle, caption=views/posts/show.html.erb] ¡p¿ ¡strong¿Title:¡/strong¿ ¡i/p¿ ¡p¿ ¡strong¿Body:¡/strong¿ ¡i/p¿ ¡i

px

respond_to (1)

- Rails helper que especifica como responder a uma requisio baseado no formato da requisio

[style=RubyInputStyle, caption=controllers/posts_controller.rb]
Class PostsController ¡ ApplicationController
POST /posts POST /posts.json def create @post = Post.new(post$_p$arams)
respond$_t$odo|format|if @post.saveformat.htmlredirect$_t$o@post, notice :

px

redirect_to

- Ao invs de renderizar um template - envia uma resposta ao navegador: "go here"
- Usualmente utiliza uma URL completa como um parmetro
  - pode ser tanto uma URL ou uma rota nomeada
- Se o parmetro um objeto - Rails tentara gerar uma URL para aquele objeto

Ao: Destroy (1)

- Remove uma postagem especfica pelo parmetro id passado como parte da URL [style=RubyInputStyle, caption=posts_controller.rb] Class PostsController ¡ ApplicationController
  before$_{a}ction$: $set_{p}ost$, $only$: [: $show$, : $edit$, : $update$, : $destroy$]

DELETE /posts/1 DELETE /posts/1.json def destroy @post.destroy respond$_{t}odo|format|format.html redirect_{t}oposts_{u}rl$, $notice$:' $Postwas...$
  private Use callbacks to share common setup or constraints between actions. def
  set$_{p}ost@post$ = $Post.find(params[:id])end$

px

Ao: New (1)

- Cria um novo objeto post(vazio)
- (Implicitamente) procura pelo new.html.erb para renderizar a resposta [style=RubyInputStyle, caption=posts_controller.rb] Class PostsController ¡ ApplicationController GET /posts/new def new @post = Post.new end

px

Ao: New (2)

- new.html.erb: [style=RubyInputStyle, caption=views/posts/new.html.erbb] ¡h1¿New Post¡/h1¿ ¡¡

px

Ao: Create (1)

- Cria um novo objeto post como os parmetros que foram passados pelo formulrio new
- Tenta salvar o objeto no banco de dados
- Se sucesso, redireciona para o template show

Ao: Create (2)

- Se insucesso, renderiza o template new novamente [style=RubyInputStyle, caption=controllers/posts_controller.rb] Class PostsController ¡ ApplicationController def create @post = Post.new(post$_p$*arams*) respond$_t$*odo*|*format*|*if* @*post*.*saveformat*.*html*|*redirect*$_t$*o*@*post*, *notice* : private Never trust parameters from the scary internet, only allow the white list through. def post$_p$*aramsparams*.*require*(: *post*).*permite*(: *title*, : *content*)*end*
  - a linha 20 implementa strong parameters para aumentar a segurana da aplicao

px

Hora de Colocar a Mo na Massa

- Modifique o post_params para o cdigo abaixo: [style=RubyInputStyle, caption=controllers/posts_controller.rb] Class PostsController ¡ ApplicationController private Never trust parameters from the scary internet, only allow the white list through. def post$_p$*aramsparams*.*require*(: *post*).*permite*(: *title*, : *content*)*paramsend*
  - tente, agora, criar um post (volte agora para o cdigo original).

px

Flash (1)

- **Problema:** Queremos redirecionar um usurio para uma pgina diferente do nosso site, mas ao mesmo tempo fornecer a ele algum tipo de mensagem. Exemplo: "Postagem criada !"
- **Soluo:** flash - uma hash onde a dado persiste por exatamente UMA requisio APS a requisio corrente
- Um contedo pode ser colocado em um flash assim: [style=RubyInputStyle, caption=controllers/posts_controller.rb] flash[:attribute] = value
- Dois atributos comuns so :notice(good) e :alert (bad)

px

Flash (2)

- Estes dois atributos (:notice ou :alert) podem ser colocados no redirect_to
- show.html.erb: [style=RubyInputStyle, caption=views/posts/show.html.erb] ¡p id=”notice≫¡¡p¿ ¡strong¿Title:¡/strong¿ ¡¡/p¿ ¡p¿ ¡strong¿Body:¡/strong¿ ¡¡/p¿ ¡¡

px

Ao: Edit (1)

- Recupera uma postagem especfica no parmetro id passado como parte da URL
- (Implicitamente) procura pelo edit.html.erb para renderizar a resposta [style=RubyInputStyle, caption=controllers/posts_controller.rb] Class PostsController ¡ ApplicationController
  $before_action : set_post, only : [: show, : edit, : update, : destroy]$

GET /posts/1/edit def edit end
  private def $set_post @post = Post.find(params[: id])end$

px

Ao: Edit (2)

- edit.html.erb: [style=RubyInputStyle, caption=controllers/posts_controller.rb] ¡h1¿Editing Post¡/h1¿ ¡¡¡

px

Ao: Update (1)

- Recupera um objeto post utilizando o parmetro id
- Atualiza o objeto post com os parmetros que foram passados pelo formulrio edit
- Tenta atualizar o objeto no banco de dados
- Se sucesso, redireciona para o template show
- Se insucesso, renderiza o template edit novamente [style=RubyInputStyle, caption=posts_controller.rb] Class PostsController ¡ ApplicationController PATCH/PUT /posts/1 PATCH/PUT /posts/1.json def update respond$_t$*odo*|*format*|*if* @*post*.*update*(*post*$_p$*arams*)*format*.*html**redire* *post*).*permite*(: *title*, : *content*)*end*

px

Hora de Colocar a Mo na Massa (1)

- Modifique o arquivo de rotas para aninhar os comentrios s postagens e reinicie o servidor: [style=RubyInputStyle, caption=config/routes.rb] Rails.application.routes.draw do resources :comments resources :posts do resources :comments end end
- Crie um novo `post` e salve no banco de dados: [style=BashInputBasicStyle] irb(main):005:0¿ p1 = Post.new irb(main):006:0¿ p1.title=”Whatsapp bloqueado”irb(main):007:0¿ p1.body=”A justia bloqueou o Whatsapp...”irb(main):008:0¿ p1.save

px

Hora de Colocar a Mo na Massa (2)

- Crie um novo `comment` e o vincule a um `post`: [style=BashInputBasicStyle] irb(main):005:0¿ c1 = Comment.new irb(main):006:0¿ c1.body=”O que fazer agora ???”irb(main):007:0¿ c1.save irb(main):008:0¿ p1.comments ¡¡ c1
- Digite no navegador no endereo ⟨http://localhost:3000/posts/id/comments⟩. Onde o id o id do post criado anteriormente.
- Agora crie um novo blog e digite novamente ⟨http://localhost:3000/posts/id/comments⟩. Onde o id do blog que acabou de ser criado.(Temos um problema)

px

Hora de Colocar a Mo na Massa (3)

- Modifique o cdigo do template views/posts/show.html.erb. Insira o cdigo abaixo, aps a linhas 10 (abaixo do pargrafo do body). [style=RubyInputStyle] ¡h2¿Comments¡/h2¿ ¡div id=”comments» ¡¡¡p¿ ¡strong¿Posted ¡¡¡/p¿ ¡¡¡/div¿
- Agora no navegador visualize uma postagem que tenha comentrios.
- Acrescente o cdigo a seguir logo abaixo do cdigo anterior no arquivo views/posts/show.html.erb: [style=RubyInputStyle] ¡¡p¿ ¡¡¡/p¿ ¡p¿ ¡¡/p¿ ¡

px

Hora de Colocar a Mo na Massa (4)

- Modifique a aco create do controlador controllers/comments_controller.rb: [style=RubyInputStyle] def create @post = Post.find(params[:post_id])@comment = @post.comments.create(comment_params) respond_to do|format|if @comment.saveformat.htmlredirect_to@post,
- Escolha uma postagem qualquer e escreva alguns comentrios.
- Remova a rota absoluta para comentrios no arquivo de rotas e reinicie o servidor: [style=RubyInputStyle, caption=config/routes.rb] Rails.application.routes.draw do resources :comments resources :posts do resources :comments end end

px

Agenda

px

Action View

- Arquivo HTML com a extenso .erb
  - ERb uma biblioteca que permite a colocao de cdigo Ruby no HTML
- Dois padres a aprender:
  - <% ...cdigo ruby..%> avalia o cdigo Ruby
  - <%= ...cdigo ruby..%> retorna o resultado do cdigo avaliado

px

Partials (1)

- Rails encoraja o princpio DRY
- O laioute da aplicao mantida em um nico local no arquivo application.html.erb
- O cdigo comum dos templates ser reutilizado em mltiplos templates
- Por exemplo, os formulrios do edit e do new - so realmente muito diferentes ?
- Partials so similares aos templates regulares, mas ele possuem capacidades mais refinadas
- Nomes de partials comeam com underscore (_)

px

Partials (2)

- Partials so renderizados com render 'partialname' (sem underscore)
- render tambm aceita um segundo argumento, um hash com as variveis locais utilizadas no partial
- Similar a passagem de variveis locais, o render pode receber um objeto
- <%= render @post %> renderizara app/views/posts/_posts.html.erb com o contedo da variavel @post
- <%= render @posts %> renderiza uma coleo e equivalente a: [style=RubyInputStyle, caption=controllers/posts_controller.rb] ¡¡¡

px

Partials (3)

- _form.html.erb [style=RubyInputStyle, caption=views/posts/_form.html.erb] ¡¡¡div id=”error$_e$*xplanation*» < *h*2 >< *prohibitedthispostfrombeingsaved* :< /*h*2 >< *ul* ><< *li* ><<<< /*ul* >< /*div* >< ¡div class=”field» ¡¡¡/div¿ ¡div class=”field» ¡¡¡/div¿ ¡div class=”actions» ¡¡/div¿ ¡

px

Form Helpers (1)

- form_for gere a tag form para o objeto passado como parmetro
- Rails utiliza a mtodo POST por padro
- Isto faz sentido:
  - uma password no passada como parmetro na URL
  - qualquer modificao dever ser feita via POST e no GET

[style=RubyInputStyle, caption=views/posts/_form.html.erb] ¡... ¡

px

f.label

- Gera a tag HTML label
- A descrio pode ser personalizada passando um segundo parmetro [style=RubyInputStyle] ¡div class=”field» ¡¡¡/div¿

px

f.text_field

- Gera o campo input type=”text”
- Utilize :placeholder para mostrar um valor dentro do campo [style=RubyInputStyle] ¡div class=”field» ¡¡¡/div¿

px

f.text_area

- Similar ao f.text_field, mas gera um text area de tamanho (40 cols x 20 rows)
- O tamanho pode ser modificado atravs do atriburo size: [style=RubyInputStyle] ¡div class=”field» ¡¡¡/div¿

px

Outros Form Helpers

- date_select
- search_field
- telephone_field
- url_field
- email_field
- number_field
- range_field

px

f.submit

- Renderiza o boto submit
- Aceita o nome do boto submit como primeiro argumento
- Se o nome no for fornecido - gera um baseado no modelo e na ao. Por exemplo: "Create Post"ou "Update Post"[style=RubyInputStyle] ¡div class="actions» ¡¡/div¿
- Mais form helpers: ⟨http://guides.rubyonrails.org/form_helpers.html⟩

px