Model ViewController

Gabriel Voicu
Email: voicu_gabriel@ymail.com
Skype: voicu_gabi
Phone: 0726 283 665

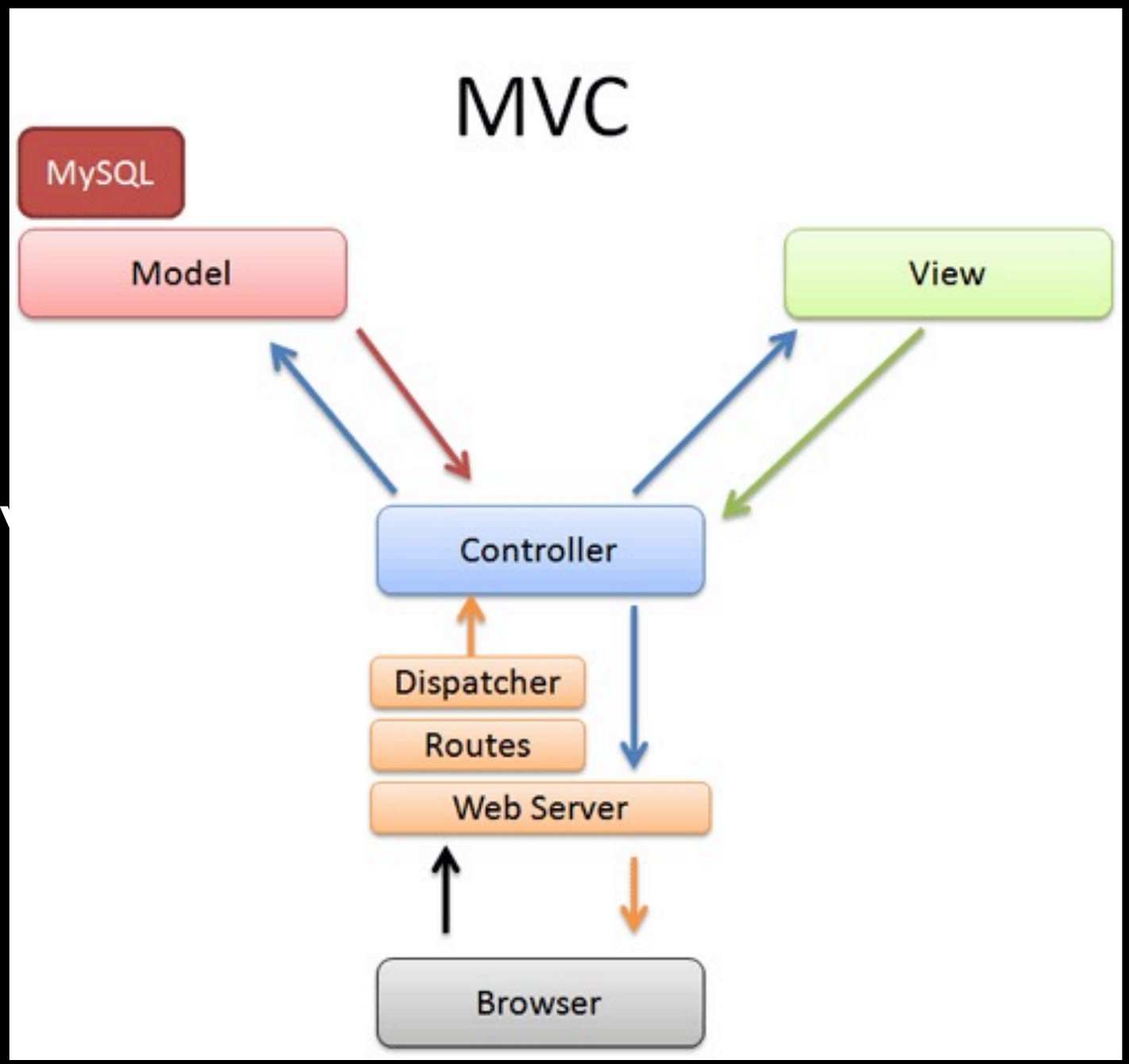
Before

- http://quora.com
- news.ycombinator.com
- html5 weekly, javascript weekly, ruby weekly, etc.
- techcrunch.com

Rails Resources

- Website
- Rails Tutorial (http://ruby.railstutorial.org/)
- RailsCasts (http://railscasts.com)
- Rails Guides (http://guides.rubyonrails.org/)
- Books
- Rails 3 Way
- Agile Web Development with Rails

Remeber


MVC
MySQL
Model
View
Controller
Dispatcher
Routes
Web Server
Browser

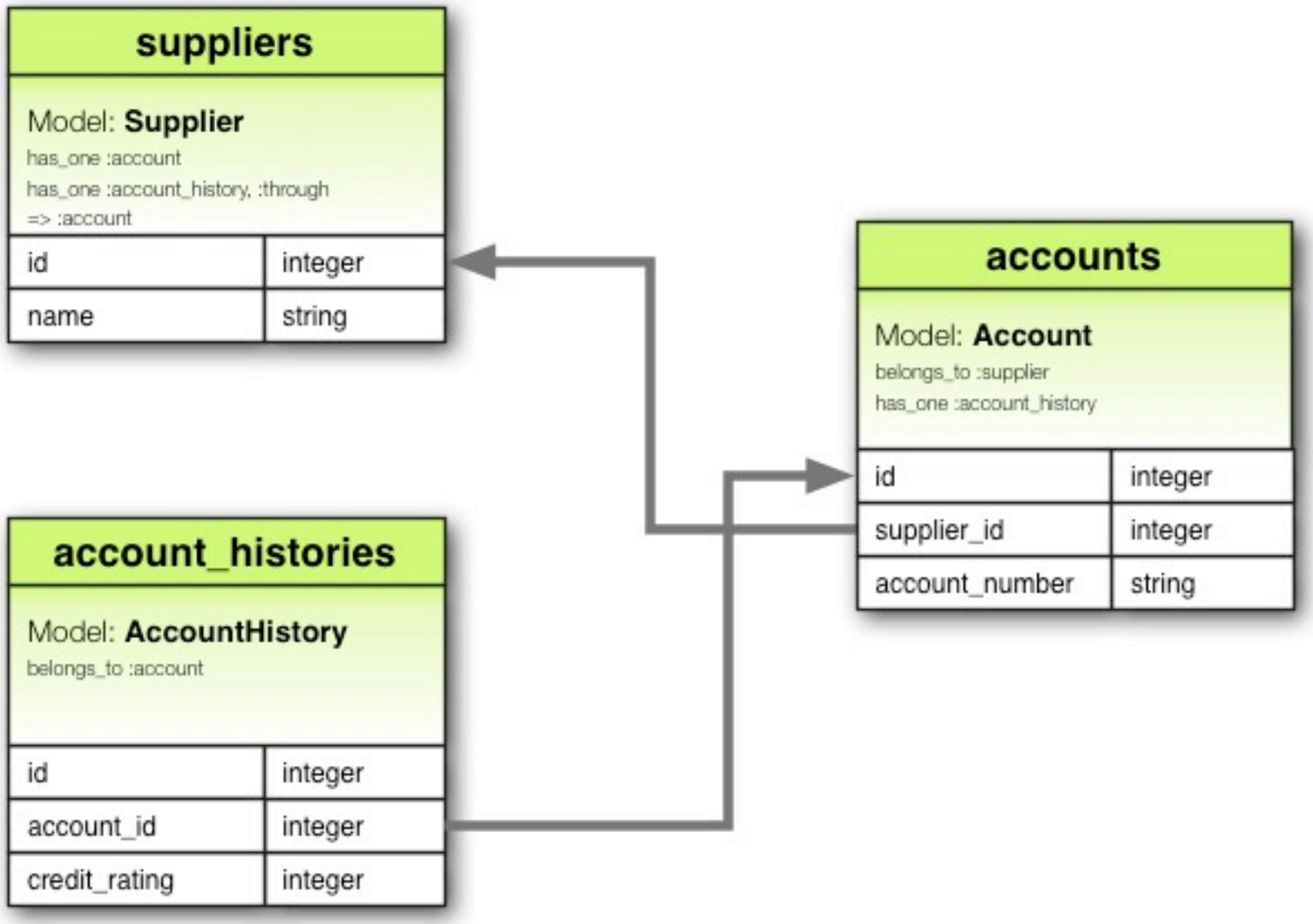
- Vie

Model

- Se ocupa de partea de stocare si procesare a datelor din aplicatie
- Celalte componente ale unei aplicatii (controller / plugin-uri / cozi de task-uri) se conecteaza la el

ORM

- Object-Relational Mapper
- Converteste date existente in formate diferite si incompatibile unele cu celelalte in entitati orientate obiect




```
class Supplier < ActiveRecord::Base
  has_one :account
  has_one :account_history, :through => :account
end

class Account < ActiveRecord::Base
  belongs_to :supplier
  has_one :account_history
end

class AccountHistory < ActiveRecord::Base
  belongs_to :account
end
```

ActiveRecord

- Cel mai folosit ORM din Rails
- Alte ORM-uri: DataMapper,
- Simplu de folosit
- Compatibil cu majoritatea bazelor de date folosite in industrie: PostgreSQL, MySQL, Oracle, SQLite, etc.

Operatii

- Creare model:
  - rails generate model Song title:string length:integer
- Adaugare coloana:
  - rails generate migration AddArtistToSong artist:string

Tasks

- Generati modelul Playlist care sa contina elementele
  - “name” de tip string
  - “player_name” de tip string

Creare Obiect

- Creare obiect
  - Metoda 1:
    - a = Song.new(:title => “SONG_NAME”, :duration => 125, :author => “SONG_AUTHOR”)
    - a.save
  - Metoda 2:
    - Song.create!(:title => “SONG_NAME”, :duration => 125, :author => “SONG_AUTHOR”)

Editare Obiect

- Metoda 1
  - song.update_attributes(:title => “NEW_SONG_NAME”)
- Metoda 2
  - song.title = “OTHER_SONG_NAME”
  - song.save

Tasks

- Creati 3 cantece si apoi updatati titlul ultimului

Metode de lucru cu modelele

- Song.all
- Song.first
- Song.last
- Song.where(title: “TITLE”)
- Song.where(“duration > ?”, 100)
- Song.where(title: “TITLE”).order(“created_at DESC”)
- Song.where(title: “TITLE”).count
- TODO: Rulati-le pe toate :-)

Validari

```
class Song < ActiveRecord::Base

  # validates_uniqueness_of :title, :author

  validates :title, :uniqueness => true, :presence => true, :length
=> {:minimum => 2}

  validates :author, :presence => true,

                :format => { :with => /[a-zA-z]+/ }

end

> s = Song.new(:title => “test”)

> s.vaild?
```

Filtre

```
class Song < ActiveRecord::Base

  before_save :count_song_save_attempts

  after_save :send_song_to_playlist

  private

  def count_song_save_attempts

    puts “Inainte de save”

  end

  def send_song_to_playlist

    puts “Dupa save”

  end

end
```

```
class Song < ActiveRecord::Base
        attr_accessible :title, :author, :duration
        attr_accessor :popularity


        def nice_print
          “#{author} - #{title}”
        end
end
> song.popularity
```

Relatii intre clase

- has_one

class Song

belongs_to :author

end

class Author

has_one :song

end

- Legatura se face prin elementul “author_id” din clasa Song
- author.build_song(title: “Show must go on”, duration: 240)
- author.create_song(title: “Show must go on”, duration: 240)

Tasks

- Creati modelul Author ce are coloanele
  - name de tip string
  - starting_date de tip datetime
- Adaugati coloana author_id la tabela Song
- Adaugati coloana playlist_id la tabela Song
- Adaugati relatii de tipul has_many intre Playlist <=> Song si Song <=> Author
- Creati cel putin un exemplu de playlist si autor

Relatii intre clase

- has_many

```
class Song                    class Author
  belongs_to :author            has_many :songs
end                           end
```

- Legatura se face prin elementul “author_id” din clasa Song
- author.songs.build(title: “Show must go on”, duration: 240)
- author.songs.create(title: “Show must go on”, duration: 240)

Accesare date modele conexe

- Accesare directa
  - songs = author.songs
  - songs.first.author.name
- songs = Song.includes(:author).where(“authors.name” => “Greenday”)
- songs.first.author.name