



3. Rails Básico

p. Continuando o desenvolvimento

i. Sistema: gerenciador de horas trabalhadas

1. Já temos o modelo user. O que mais é necessário?

ii. Ponto de partida: models e migrations

iii. Precisaremos de Project, TaskType, ProjectMembership e TimeLog

iv. Comandos:

1. `ruby script/generate model Project name:string
description:text`

2. `ruby script/generate model TaskType name:string
project:belongs_to`

3. `ruby script/generate model ProjectMembership joined:date
leaved:date user:belongs_to project:belongs_to`

4. `ruby script/generate model TimeLog user:belongs_to
task_type:belongs_to description:text tak_date:date`

v. São gerados models, migrations, tests e fixtures para cada model

1. Por enquanto daremos prioridade a models e migrations

q. Conferindo as migrations geradas

i. Antes de executar `rake db:migrate`, devemos checar o código fonte gerado para as migrations e fazer as alterações necessárias

1. `create_projects.rb`

2. `class CreateProjects < ActiveRecord::Migration`

`def self.up`

`create_table :projects do |t|`

`t.string :name, :null => false, :limit => 50`

`t.text :description`

`t.timestamps`

`end`

`end`

`def self.down`

`drop_table :projects`

`end`

3. Alterações:

a. Apenas restrição de nome não nulo e no máximo 50 caracteres

4. `create_tasks_type.rb`

5. `class CreateTasksTypes < ActiveRecord::Migration`

`def self.up`

`create_table :tasks_types do |t|`

`t.string :name, :null => false, :limit => 50`

`t.belongs_to :project, :null => false`



- ```
 t.timestamps
 end
 add_index :tasks_types, [:project_id, :name], :unique
=> true
 end
 del self.down
 drop_table :tasks_types
 end
6. Alterações
 a. Mesmas restrições em :name
 b. belongs_to criará um campo integer com o nome
 passado + sufixo _id
 i. Referência a outro modelo
 c. É criado um índice
 i. Necessário passar o nome da coluna criada
7. create_project_memberships.rb
8. class CreateProjectMemberships < ActiveRecord::Migration
 def self.up
 create_table :project_memberships do |t|
 t.date :joined, :null => false
 t.date :leaved
 t.belongs_to :user, :null => false
 t.belongs_to :project, :null => false
 t.timestamps
 end
 add_index :project_memberships, [:user_id,
:project_id], :unique => true
 end
 del self.down
 drop_table :project_memberships
 end
9. Alterações
 a. Nenhuma novidade: restrições de preenchimento,
 referência a campos e criação de índice
10. create_time_logs.rb
11. class CreateTimeLogs < ActiveRecord::Migration
 def self.up
 create_table :time_logs do |t|
 t.belongs_to :user, :null => false
 t.belongs_to :task_type, :null => false
 t.text :description
```



UNIVERSIDADE FEDERAL DO PIAUÍ  
CENTRO DE CIÊNCIAS DA NATUREZA  
DEPARTAMENTO DE INFORMÁTICA E ESTATÍSTICA

---

- ```
t.date :tak_date, :null => false
t.time :worked_hours, :null => false
t.timestamps

end

end

del self.down

drop_table :time_logs

end
```
12. Alterações
 - a. Nenhuma novidade: restrições de preenchimento, referência a campos e criação de índice
 - b. Campos :user_id, :task_type_id, :task_date obrigatórios
 13. Validações no banco são desnecessárias?
 - a. Serão refeitas nos modelos...
 - b. Uma validação a mais nunca é demais
 - i. Banco consistente
 14. Podemos executar: rake db:migrate
- r. Escrevendo modelos
- i. Restrições de relacionamento
 - ii. user.rb
 - iii.

```
class User < ActiveRecord::Base
  has_many :project_memberships, :conditions=>“leaved is null”
  has_many :time_logs
  has_many :projects, :through => :project_memberships
  validates_presence_of :login, :password, :name, :email
  ...
  validates_uniqueness :email
end
```
 - iv. Queremos, a partir de um usuário, saber quais projetos ele está relacionado e quantas horas trabalhou
 1. User tem muitos (has_many) ProjectMemberships e TimeLogs
 2. Terceiro has_many (:through)
 - a. Tem muitos projetos, mas vem de memberships
 - b. Relacionamento indireto
 - c. Possibilidade de adicionar “conditions”
 - i. Usuário só tem memberships nos projetos que ainda participa
 - v. projects.rb
 - vi.

```
class Project < ActiveRecord::Base
  has_many :task_types
```



UNIVERSIDADE FEDERAL DO PIAUÍ
CENTRO DE CIÊNCIAS DA NATUREZA
DEPARTAMENTO DE INFORMÁTICA E ESTATÍSTICA

- ```
has_many :project_memberships, :conditions=>"leaved is null"
has_many :users, :through => project_memberships
has_many :time_logs, :through => :task_types
validates_presence_of :name
validates_uniqueness_of :name
validates_length_of :name, :within => 5..50
end
```
- vii. Tipo de tarefa pertencente ao projeto
1. Nome deve ser único e com tamanho entre 5 e 50 caracteres
- viii. project\_membership.rb
- ix. class ProjectMembership < ActiveRecord::Base
- ```
  belongs_to :user  
  belongs_to :project  
  validates_associated :user  
  validates_associated :Project  
  validates_presence_of :joined  
end
```
- x. task_type.rb
- xi. class TaskType < ActiveRecord::Base
- ```
 belongs_to :project
 has_many :time_logs
 validates_associated :project
 validates_uniqueness_of :name, :scope => :project_id
 validates_presence_of :name
 validates_length_of :name, :within => 5..50
end
```
- xii. time\_log.rb
- xiii. class TimeLog < ActiveRecord::Base
- ```
  belongs_to :user  
  belongs_to :task_type  
  validates_presence_of :task_date  
  validates_presence_of :worked_hours  
end
```
- xiv. Registro de tempo trabalhado de um usuário em um tipo de tarefa
- xv. Observe que o nome do campo task_date foi gerado errado (tak_date)
1. Correção através de nova migration
 - a. ruby script/generate migration fix_typo
 - b. Editamos o arquivo /db/migration/<data>_fix_tipo.rb
 - c. class FixTypo < ActiveRecord::Migration
def self.up



UNIVERSIDADE FEDERAL DO PIAUÍ
CENTRO DE CIÊNCIAS DA NATUREZA
DEPARTAMENTO DE INFORMÁTICA E ESTATÍSTICA

```
        rename_column :time_logs, :tak_date,  
        :task_date  
      end  
      def self.down  
        rename_column :time_logs,  
        :task_date, :tak_date  
      end  
    end
```

end

d. Não se altera migration que já foi executada

s. Gerando todo o código

i. Scaffold não gera código realmente útil, mas é bastante útil no desenvolvimento

1. `ruby script/generate scaffold Project name:string description:text`
2. `ruby script/generate scaffold TaskType name:string project:belongs_to`
3. `ruby script/generate scaffold ProjectMembership joined:date leaved:date user:belongs_to project:belongs_to`
4. `ruby script/generate scaffold TimeLog user:belongs_to description:text task_date:date worked_hours:time`

ii. Praticamente a mesma coisa da geração de modelos

1. Poderia ser usada desde início, mas a ideia é fazer a aplicação em partes mesmo
2. Erro no nome do campo foi corrigido
3. Adicionado o campo horas trabalhadas no TimeLog

iii. Já é possível navegar na aplicação, porém os relacionamentos não funcionam

1. Scaffold não gera código para relacionamentos

iv. Correção de rotas

1. Não faz sentido listar, por exemplo, `project_memberships`

v. `ActionController::Routing::Routes.draw` do `|map|`

```
map.resources :users do |usr|  
  usr.resources :time_logs  
  usr.resources :project_memberships  
end
```

end

```
map.resources :projects do |proj|  
  proj.resources :task_types do |tt|  
    tt.resources :time_logs  
  end  
end
```

```
proj.resources :project_memberships
```

end



UNIVERSIDADE FEDERAL DO PIAUÍ
CENTRO DE CIÊNCIAS DA NATUREZA
DEPARTAMENTO DE INFORMÁTICA E ESTATÍSTICA

```
map.connect ':controller/:action/:id'  
map.connect ':controller/:action/:id.:format'
```

end

- vi. Agora, podemos acessar os membros de um projeto, por exemplo
- vii. rake routes
- t. Um pouco de segurança na aplicação
- u. Um layout menos confuso para a aplicação
- v. Associando usuários a projetos
- w. Adicionando tipos de tarefas a um projeto
- x. Cadastro das horas trabalhadas
- y. Um relatório para a aplicação
- z. Limpando um pouco o código e se livrando um pouco do “inglês”