



Curso Completo



Aula 033

PORO's

PORO's

Plain Old Ruby Object

PORO's

Classes simples que não herdam de nenhuma outra.

```
class Car  
end
```

Como fazer?

```
class RegistrationController < ApplicationController
  def create
    user = User.new(registration_params)
    if user.valid? && ip_valid?(registration_ip)
      user.save!
      user.add_bonuses
      user.synchronize_related_accounts
      user.send_email
    end
  end
end
```

```
class RegistrationService
  def fire!(params)
    user = User.new(params)
    if user.valid? && ip_validator.valid?(registration_ip)
      user.save!
      after_registered_events(user)
    end
    user
  end

  private

  def after_registered_events(user)
    BonusesCreator.new.fire!(user)
    AccountsSynchronizator.fire!(user)
    EmailSender.fire!(user)
  end

  def ip_validator
    @ip_validator ||= IpValidator.new
  end
end
```

Como fazer?

```
class RegistrationController < ApplicationController
  def create
    user = RegistrationService.new.fire!(registration_params)
  end
end
```

Comparando...

```
class RegistrationController < ApplicationController
  def create
    user = User.new(registration_params)
    if user.valid? && ip_valid?(registration_ip)
      user.save!
      user.add_bonuses
      user.synchronize_related_accounts
      user.send_email
    end
  end
end
```

```
class RegistrationController < ApplicationController
  def create
    user = RegistrationService.new.fire!(registration_params)
  end
end
```


PORO's

- A maior vantagem é de tornar a classe responsável por uma única coisa (Single Responsibility Principle - SRP).

PORO's

- Complexidade do código reduzida, mais explícita e direta;
- Facilitação da legibilidade;
- Redução de acoplamento;
- Código limpo e testável;
- Facilidade de evolução.



Exemplo Rails?

Exemplo Rails?

```
└─ app
  └─ assets
  └─ controllers
    └─ backoffice
      └─ categories_controller
        └─ category_service.rb
        └─ admins_controller.rb
        └─ categories_controller.rb
```

Exemplo Rails?

```
class Backoffice::CategoriesController::CategoryService
  attr_accessor :category

  def self.create(params_category)
    @category = Category.new(params_category)

    if @category.valid?
      @category.save!
    end

    @category
  end
end
```

Exemplo Rails?

```
def create
  @category = CategoryService.create(params_category)

  unless @category.errors.any?
    redirect_to backoffice_categories_path, notice: "A categoria foi criada com sucesso."
  else
    render :new
  end
end
```