Kurs programowania Ruby on Rails

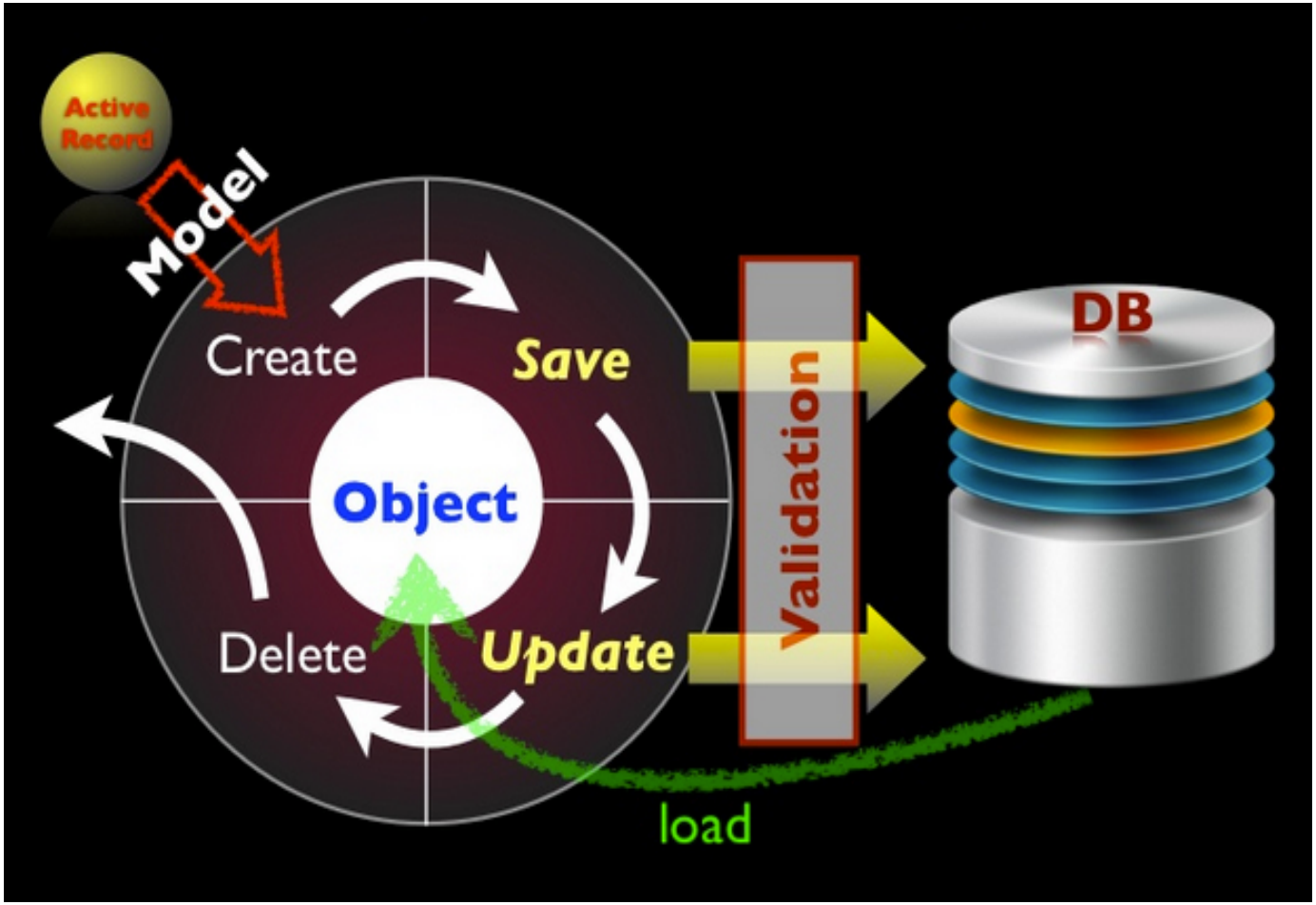
Callbacki

Callbacki

Callbacki są metodami wywoływanymi w pewnym momencie cyklu życia obiektu. Używając callbacków możemy napisać działania, który zadziałą kiedykolwiek obiekt modułu ActiveRecord zostanie utworzony, zachowany, zauktalizowany, zwalidowany, załadowany z bazy danych.

Cykl życia obiektu

Szkolenie Rails



Callbacki

- jako zwykła metoda bądź jako blok

- przed lub po cyklem życia obiektu

create,save,update,delete,validate,load

- private lub protected

Przykładowy callback


```
class User < ActiveRecord::Base

  validates_presence_of :login, :email

  before_validation :ensure_login_has_a_value

  protected

  def   ensure_login_has_a_value

    if login.nil?

      self.login = email unless email.blank?

    end

  end

End
```

Źródło http://guides.rubyonrails.org/

Przykładowy callback – jako blok


```
class User < ActiveRecord::Base

  validates :login, :email, presence: true


  before_create do |user|

    user.name = user.login.capitalize if      user.name.blank?

  end

End
```

Źródło http://guides.rubyonrails.org/

Dostępne callbacki

CREATE	UPDATE	DESTROY
before_validation	before_validation	before_destroy
after_validation	after_validation	around_destroy
before_save	before_save	after_destroy
around_save	before_update	
after_create	around_update	
after_save	after_update	

Typy w bazie danych

Szkolenie Rails

string	text	integer	boolean	decimal
float	binary	date	time	datetime

Warunkowe callbacki

- używane z :if i :unless z Symbolem
- używane z :if i :unless z Stringiem
- używane z :if i :unless z Proc

Warunkowe callbacki

Symbol:

```
class Order <ActiveRecord::Base

  before_save :normalize_card_number , :if => :paid_with_card?
```

String:

```
Class Order < ActiveRecord::Base

before_save :normalize_card_number, :if =>" paid_with_card?"
```

Proc:

```
Class Order < ActiveRecord::Base

   before_save :normalize_card_number, :if =>Proc.new {|lorder| order.paid_with_card? }
```

After_find , After initialize

Szkolenie Rails

```
class User < ActiveRecord::Base
  after_initialize do |user|
    puts "You have initialized an object!"
  end
  after_find do |user|
    puts "You have found an object!"
  end
end
```

Metody wywołujące callbacki

-create

-create!

-decrement!

-destroy

-destory_all

-increment!

-save

-save!

Metody wywołujące callbacki c.d.

-save(:validate =>false)

-toggle!

-update

-update_attribute

-update_attributes

-update

-update_attribute

-update_attributes

-valid?

Metody pomijące callbacki

- decrement
- decrement_counter
- delete
- delete_all
- find_by_sql
- increment
- increment_counter

Metody pomijące callbacki c.d.

- toggle
- touch
- update_column
- update_all
- update_counters

Callback tylko dla pewnego cyklu życia obiektu

Szkolenie Rails

```
class User <ActiveRecord::Base

after_validation :set_location, on: [:create,:update]

# przekazujemy tablicę symboli, bądź pojedynczy #symbol

def set_lcoation

  self.location =" Warszawa"

end
```

Zatrzymanie wykonania

KOLEJKA:

- walidowanie dla modeli
- rejestracja callbacków
- operację na bazie danych

ZATRZYMANIE:

Before_ callbacks

:false lub wyjątek ->zatrzymanie->ROLLBACK

Przykład błędnego wykonania


```
class Article < ActiveRecord::Base

before_update :close_comments, :if =>:more_than_ten_comments

Protected

def close_comments

self.open_for_comments = false

end

end
```

Kurs programowania Ruby on Rails

Callbacki

Koniec