Cells Cheatsheet

Version 1.0 (Sept 2011)

Generator

Generate CartCell	`rails g cell Cart show redraw`
HAML views	`rails g cell Cart show -e haml`
RSpec test	`rails g cell Cart show -t rspec`

Options processing

No options	`render_cell(:cart, :show)`
	
	`def show`
	`  render`

Keyword options	`render_cell(:cart, :show,`
	`              :user => current_user, ...)`
	
	`def show(opts)`
	`  user = opts[:user]`

Arguments list	`render_cell(:cart, :show, current_user, @items)`
	
	`def show(user, items)`
	`  raise 'Get out!' unless user.logged_in?`

Default args	`render_cell(:cart, :show, @items)`
	
	`def show(items=[])`
	`  items.each do |item|`

Rendering

Note: Feel free to use `#render` in views the same way as in state methods.

Render view	`def show`
	`  render`
	`end`

Alternative view	`render :view => :highlighted`
With layout	`render :layout => :boxed`
Pass locals	`render :locals => {:user => user}`
Arbitrary text	`render :text => 'alert(42);'`

Invoke alternative state	`def show`
	`  render :state => :update`
	`  # => will send mail, too!`
	`end`
	
	`def update`
	`  send_mail`
	`  render`
	`end`

Pass args to rendered state

```
render({:state => :header}, Time.now)

def header(time)
  render
```

Render twice and wrap

```
def show
  "<div>" + render(:view => header) +
    render + "</div>"
end
```

Control Flow & Initialization

Redirecting

```
render_cell(:cart, :show) do |cell|
  redirect_to "/login" and return if cell.no_items?
end
```

Initialization

```
render_cell(:cart, :show) do |cell|
  cell.highlight! if current_user.vip?
end
```

View Inheritance

Inheriting views

```
|-- cells
|   |-- cart_cell.rb
|   |-- cart
|   |   `-- show.haml
|   |-- vip_cart_cell.rb
|   |-- vip

class VipCartCell < CartCell
  def show
    render  # => will use cart/show.haml.
  end
```

Overriding views

```
|-- cells
|   |-- cart_cell.rb
|   |-- cart
|   |   `-- header.haml
|   |   `-- show.haml
|   |-- vip_cart_cell.rb
|   |-- vip
|   |   `-- header.haml

class VipCartCell < CartCell
  def show
    render :view => :header + # => renders vip_cart/header.haml.
      render                  # => renders cart/show.haml.
  end
```

Builders

Add builder

```
CartCell.build do |opts|
  VipCartCell if current_user.vip?
end
```

`render_cell(:cart, show)` instantiates and uses `VipCartCell` if condition applies.

Caching

TTL	`class CartCell < Cell::Base`
	`  cache :show, :expires_in => 5.minutes`

Custom cache key	` cache :show do`
	`   :user => current_user.name`
	`  end`

Condition	`  cache :show, :if => Time.now.year > 2010`

Using state options	`render_cell :cart, :show, current_user`
	
	`class CartCell < Cell::Base`
	`  cache :show do |user|`
	`    :user => user.id`
	`  end`

Configuration

Adding view path	`Cell::Base.append_view_path "app/view_models"`

Initializer	`Cells.setup do |config|`
	`  config.append_view_path "app/view_models"`
	`end`

Test::Unit

Rendering	`class CartCellTest < Cell::TestCase`
	`  tests CartCell`
	
	`  it "shows user name" do`
	`    invoke :show, :user => @user_1`
	`    assert_select ".name", "Kevin to the T"`

Unit test	` it "responds to #vip?" do`
	`    assert cell(:cart, :user => @user_1).vip?`