

# Ruby on Rails

Computerlabor

# Ablauf

- Einführung in Ruby
- Einführung in Ruby on Rails
  - ActiveRecord
  - ActionPack
  - ActiveResource
- Praxis

# Ruby Stichworte

- 1995 erschienen, Open Source
- Entwickelt von Yukihiro Matsumoto
- Vollständig Objektorientiert
- Interpretiert
- Dynamische Typbindung (Duck Typing)
- Garbage Collection

# Ruby Implementationen

- MRI (Matz Ruby Interpret, C)
- YARV (C)
- JRuby (Java)
- Rubinius (Ruby)
- HotRuby (JavaScript/ActionScript)
- IronRuby (.NET Framework)

# Ruby Syntax

```
5.times do
```

```
  puts "Computerlabor"
```

```
done
```

- Alles ist ein Objekt
- {} oder do ... end, () optional
- Keine Variabeldeklaration

# Ruby Syntax

```
10.upto(100) { |i| puts i }
```

# Ruby Funktionen

```
def zaehle(to => 100)
  l.upto(100) { |i| puts i }
end
```

# Ruby Funktionen

```
def summiere(a, b)  
  a+b  
end
```

# Ruby OO

```
class Auto
  def beschleunigen
    puts "Brumm"
  end
end
```

```
auto = Auto.new
auto.beschleunigen
```

# Ruby OO

```
class MegaAuto < Auto
  def bremsen
    puts "Quietsch"
  end
end

auto = MegaAuto.new
auto.beschleunigen
auto.bremsen
```

# Ruby OO

- Einfachvererbung
- Keine abstrakten Klassen
- Mixins anstelle von Interfaces
- Metaprogramming

# Ruby Accessoren

```
class Person
  attr_accessor :name
  attr_reader :alter
end
```

# Ruby Modifikatoren

@instanzvariable

@@klassenvariable

KONSTANTE

public, protected & private

self

# Ruby Blocks

```
def twice  
  yield  
  yield  
end
```

```
twice { puts "Hello" }
```

# Ruby Blocks

```
def twice  
  yield("Olaf")  
  yield("Sergei")  
end
```

```
twice { |name| puts "Hello" + name }
```

# Ruby Iteratoren

```
friends = ['Larry', 'Curly', 'Moe']  
friends.each { |f| print f + "\n" }
```

# Ruby on Rails

RoR

# Ruby on Rails

- 2004, David Heinemeier Hansson
- Extrahiert aus Basecamp
- 100% Ruby

# Ruby on Rails Prinzipien

- DRY (Don't Repeat Yourself)
- Convention over Configuration
- Agile Methodiken (u.a.TDD)

# Ruby on Rails Patterns

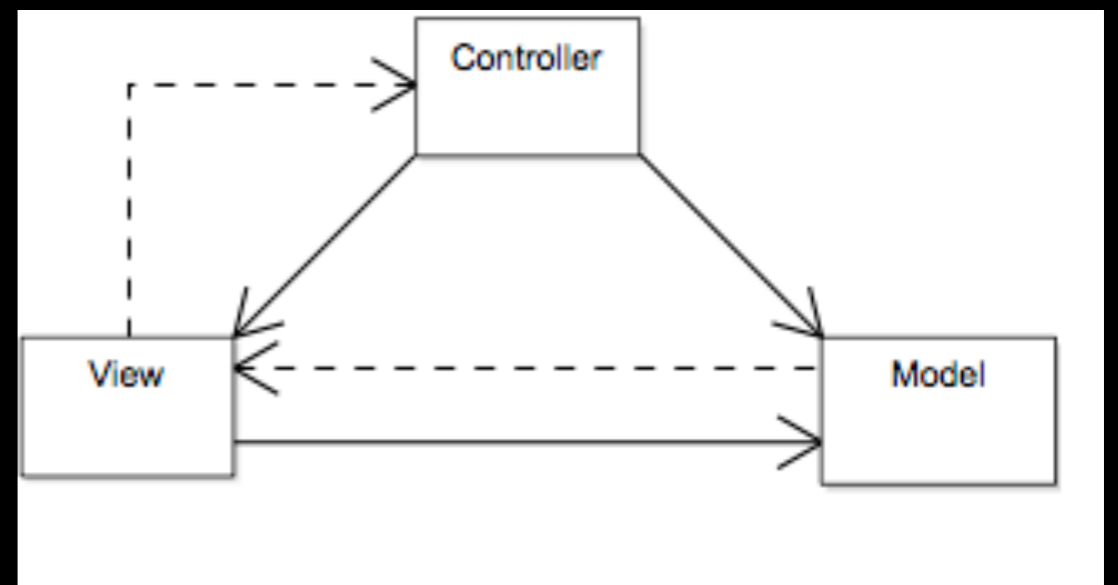
- Model View Controller
- Active Record

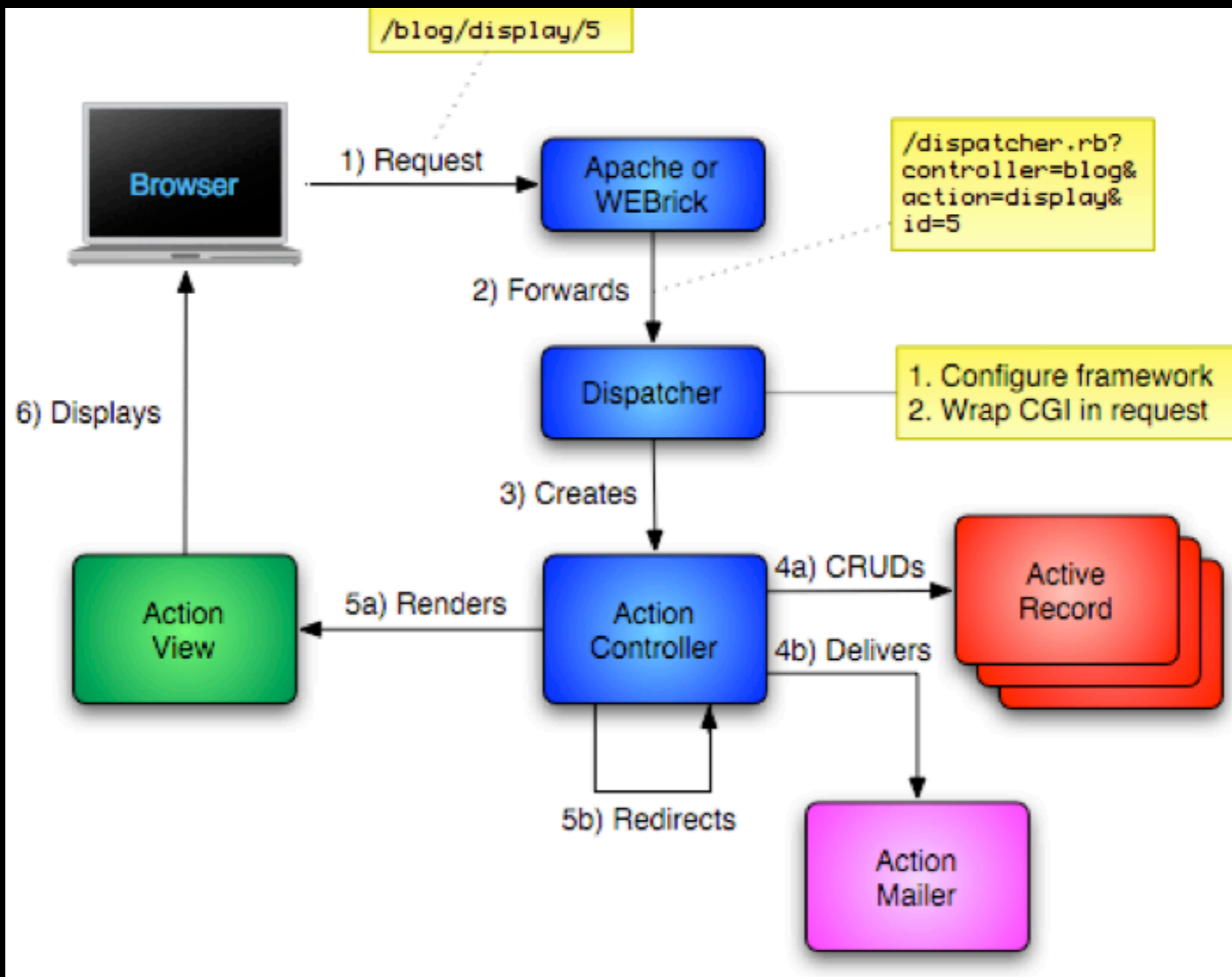
# Model View Controller

Model = Datenstruktur

View = Darstellung der Struktur

Controller = Programmlogik





# Typische Anfrage

# Active Record

- Tabelle ist eine Klasse
- Spalte ist ein Attribut
- Objekt ist ein Datensatz

# Active Record

- `class Person < ActiveRecord::Base`  
`end`
- `Person.find(1)`
- `Person.find_by_name 'Pascal'`
- `Person.find_by_name_and_firstname 'Pascal',`  
`'Zubiger'`

# Active Record Relationen

- `has_many + belongs_to`
- `has_and_belongs_to`

# Active Record Migrations

```
class NeueTabelle < ActiveRecord::Migration

  def self.up
    create_table :users do |table|
      table.string :name
      table.string :login, :null => false
      table.string :password, :limit => 32, :null => false
      table.string :email
    end
  end

  def self.down
    drop_table :users
  end

end
```

# Active Record Validations

`validates_acceptance_of :eula`

`validates_associated :comments`

`validates_presence_of :name`

`validates_confirmation_of :password`

`validates_length_of :login, :minimum => 6`

`validates_length_of :password, :within => 8..25`

`validates_numericality_of :year, :integer_only => true`

`validates_confirmation_of :password`

`validates_uniqueness_of :email`

`validates_format_of :email, :with => /\A([\^@\s]+)@\((?:  
[-a-z0-9]+\.\.)+[a-z]{2,})\Z/i`

# Rails routes.rb

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

  map.connect 'products/:id', :controller => 'catalog',
              :action => 'view'

  # purchase_url(:id => product.id)
  map.purchase 'products/:id/purchase', :controller => 'catalog',
          :action => 'purchase'

  map.connect "", :controller => "welcome"

  # Install the default route as the lowest priority.
  map.connect ':controller/:action/:id.:format'
  map.connect ':controller/:action/:id'

end
```

# Rails Controller

```
class WorldController <
  ApplicationController

  def hello

    render :text => 'Hello world'

  end

end
```

# Rails CRUD Controller

```
class BookController < ApplicationController
  def list; end
  def show; end
  def new; end
  def create; end
  def edit; end
  def update; end
  def delete; end
end
```

# Rails Controller

```
def list
```

```
  @books = Book.find(:all)
```

```
end
```

# Rails Controller

```
def show
```

```
  @book = Book.find(params[:id])
```

```
end
```

# Rails Controller

```
def new
```

```
  @book = Book.new
```

```
  @subjects = Subject.find(:all)
```

```
end
```

# Rails Controller

```
def create
  @book = Book.new(params[:book])
  if @book.save
    redirect_to :action => 'list'
  else
    @subjects = Subject.find(:all)
    render :action => 'new'
  end
end
```

# Rails Controller

```
def edit
  @book = Book.find(params[:id])
  @subjects = Subject.find(:all)
end
```

# Rails Controller

```
def update
  @book = Book.find(params[:id])
  if @book.update_attributes(params[:book])
    redirect_to :action => 'show', :id => @book
  else
    @subjects = Subject.find(:all)
    render :action => 'edit'
  end
end
```

# Rails Controller

```
def delete
  Book.find(params[:id]).destroy
  redirect_to :action => 'list'
end
```

# Rails Views

```
<% if @books.blank? %>
  <p>There are not any books currently in the system.</p>
<% else %>
  <p>These are the current books in our system</p>
  <ul id="books">
    <% @books.each do |c| %>
      <li><%= link_to c.title, {:action => 'show', :id => c.id} -%></li>
    <% end %>
  </ul>
<% end %>
<p><%= link_to "Add new Book", {:action => 'new' }%></p>
```

# Rails Views

```
<h1>Add new book</h1>

<% form_for :book, @book, :url => { :action => "create" } do |f| %>

  <%= f.label :title %>
  <%= f.text_field :title %><br/>

  <%= f.label :price %>
  <%= f.text_field :price %><br/>

  <%= f.label :subject %>
  <%= f.collection_select(:subject_id, @subjects, :id, :name) %><br/>

  <br/>

  <%= f.label :description %><br/>
  <%= f.text_area :description %><br/>

  <br/>

  <%= submit_tag "Create" %>

<% end %>

<%= link_to 'Back', { :action => 'list' } %>
```

# Rails Views

- *Layouts*
- *Partials*

# IDE's

- Textmate, Vim, Emacs ;-)
- Netbeans
- Aptana
- Eclipse + RDT
- Eclipse + DLTK
- IntelliJ IDEA
- 3rd Rails

# ScreenCasts

- [railscast.com](http://railscast.com)
- [peepcode.com](http://peepcode.com)

# Bücher

- Programming Ruby (PragProg - PickAxe)
- Agile Web Development with Rails (PP)
- The Ruby Way (AWW)
- The Rails Way (AWW)
- Advanced Rails Recipes

# Ressourcen

- <http://www.slideshare.net/jweiss/an-introduction-to-ruby-on-rails/>
- <http://www.slideshare.net/raindoll/ruby-on-rails-einfhrung/>
- <http://www.tutorialspoint.com/ruby-on-rails>
- <http://www.codyfauser.com/2005/11/20/rails-rjs-templates>

END

Mehr gefällig?

Standard Methods	Verb	Path	Action
<i>plural_path</i>	GET	/teams	index
<i>singular_path(id)</i>	GET	/teams/1	show
<i>new_singular_path</i>	GET	/teams/new	new
<i>plural_path</i>	POST	/teams	create
<i>edit_singular_path(id)</i>	GET	/teams/1;edit	edit
<i>singular_path(id)</i>	PUT	/teams/1	update
<i>singular_path(id)</i>	DELETE	/teams/1	destroy

# REST

# REST routes.rb

```
map.resources :users, :sessions
```

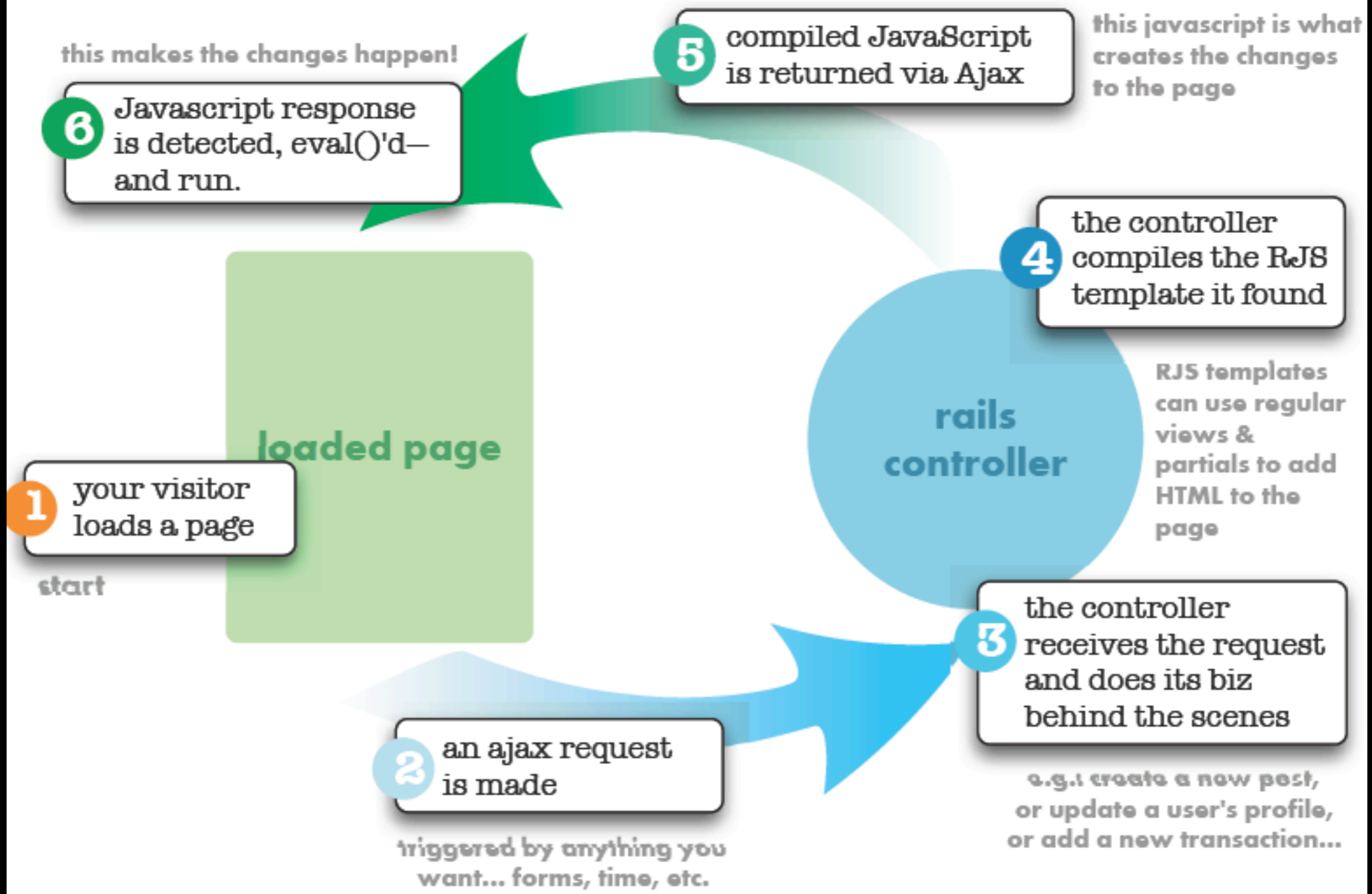
```
map.resources :teams do |teams|
```

```
  teams.resources :players
```

```
end
```

# REST helpers

```
link_to "Destroy",  
team_path(@team),  
:confirm => "Are you sure?",  
:method => :delete
```



# Ruby Java Script (RJS)

# RJS View

```
<%= javascript_include_tag :defaults %>
```

```
<h1 id='header'>RJS Template Test</h1>
```

```
  <ul id='list'>
```

```
    <li>Dog</li>
```

```
    <li>Cat</li>
```

```
    <li>Mouse</li>
```

```
</ul>
```

```
<%= link_to_remote("Add a fox",  
  :url =>{ :action => :add }) %>
```

# RJS Controller

```
def add  
end
```

# add.rjs.erb

```
page.insert_html :bottom, 'list',  
  content_tag("li", "Fox")  
page.visual_effect :highlight, 'list', :duration => 3  
page.replace_html 'header',  
  'RJS Template Test Complete!'
```

# Capistrano Deployment

- `gem install -y capistrano`
- `capify .`
- `cap -T`

# Capistrano Deployment

```
set :application, "set your application name here"  
set :repository, "set your repository location here"
```

```
set :deploy_to, "/var/www/#{application}"
```

```
set :scm, :subversion
```

```
role :app, "your app-server here"  
role :web, "your web-server here"  
role :db, "your db-server here", :primary => true
```

# Capistrano Deployment

```
# Initialies Setup  
cap deploy:setup
```

```
# Abhängigkeiten prüfen  
cap -q deploy:check
```

```
# Erster Deploy und Migrationen ausführen  
cap deploy:cold
```

```
# Weitere Deploys  
cap deploy
```

```
# Rollback  
cap deploy:rollback
```