



## GIS mit Ruby on Rails

**Pirmin Kalberer**  
**Sourcepole AG, Bad Ragaz**  
**[www.sourcepole.ch](http://www.sourcepole.ch)**



**./configure && make  
&& make install**

```
apt-get install postgis
```

# **XML, SOAP**

# Http, REST

**CVS**

**git**

**Linux?**



**Linux!**

**RUP**

# **Agile Software- Entwicklung**

# PHP Web-GIS

# **Web-Framework, Mashup**

**Sourceforge**

**github**

# OSS GIS 2000



**OSS GIS 2010?**

# Ruby

**Interpretierte, objektorientierte &  
dynamisch typisierte Skriptsprache**

# Rails

**Fullstack MVC Web-Applikations  
Framework**



# Ruby - Philosophie

eine dynamische, freie Programmiersprache,  
die sich **einfach anwenden** und  
produktiv einsetzen lässt.

Sie hat eine **elegante Syntax**, die man  
leicht lesen und schreiben kann.



# Ruby - Programmiersprache

- › Japan 1995, Yukihiro Matsumoto
- › Open Source
- › Mischung aus Perl, Smalltalk, Eiffel, Ada und Lisp
- › Hohe Qualität der Bibliotheken



```
5.times do  
  puts "ruby ist cool!".upcase  
end
```

```
RUBY IST COOL!  
RUBY IST COOL!  
RUBY IST COOL!  
RUBY IST COOL!  
RUBY IST COOL!
```



# Rails Framework

- › Dänemark 2004, David Heinemeier Hansson
- › Open Source MIT Lizenz
- › Fullstack MVC Web-Framework
- › Wurde aus bestehender Anwendung extrahiert
- › 20 Konferenzen 1. Halbjahr 2010
- › Bücher: >100



# Rails Features

- › „Don't repeat yourself” (DRY)
- › Convention over Configuration
- › MVC, REST,...
- › AJAX
- › Test-Framework
- › Generatoren, Scaffolding
- › Plugins



Das **Model** bildet die zugrundeliegende Datenstruktur

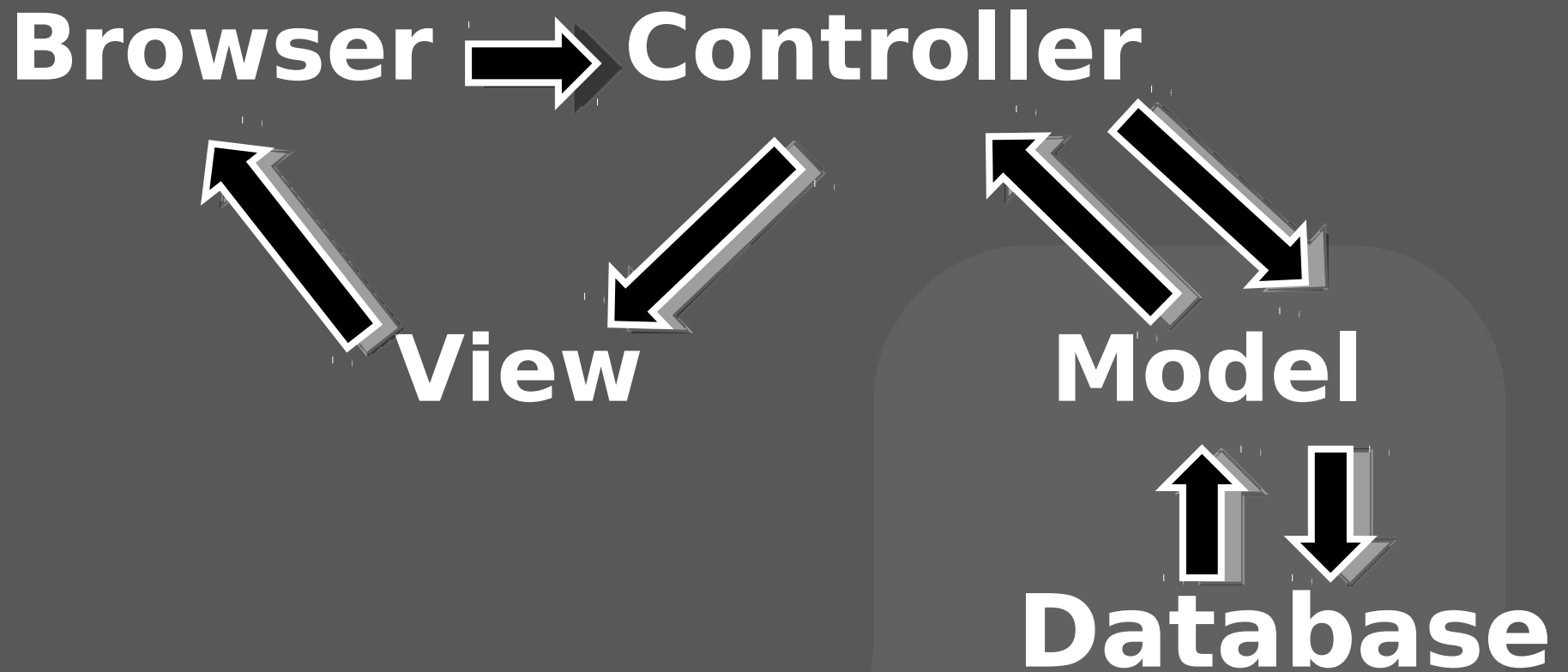
Die **View** ist die Darstellung der Datenstruktur

Der **Controller** enthält die eigentliche Programmlogik





# Rails Architektur





## ActiveRecord

- › **ORM: Objektrelationales Mapping**
- › **Enthält Business-Logik**
- › **Beziehungen zwischen Modellen**
- › **Validierung**



## ActionView

- › Repräsentiert die Sicht auf die Daten
- › HTML, XML, EMail Inhalte oder Javascript
- › kann Ruby Code enthalten
- › Helfer Klassen enthalten View Logik



## ActionController

- › steuert den Kontrollfluss der Anwendung
- › redirected zu anderen Controller Aktionen
- › stellt der View Daten und Methoden zur Verfügung
- › wählt die View
- › sendet View zurück an den Client



## Model

```
class Customer < ActiveRecord::Base
  has_many :projects
  validates_presence_of :name,
    :minimum => 50, :message => "zu kurz!"
end
```

## Controller

```
def list
  @customers = Customer.all(:aktiv => true)
end
```



## View

```
<ul>
  <% for customer in @customers %>
    <li><%= link_to customer.name, customer %>
      ( <%= link_to "ändern",
          edit_customer_path(customer) %>)
    </li>
  <% end %>
</ul>
```



# GIS-Bibliotheken für RoR

- › **GeoKit, Graticule** und **acts\_as\_geocodable**: Geokodierung und Distanzberechnungen
- › **GeoRuby**: Spatial DB Adapter
- › **Ruby bindings**: OGR/GDAL, Mapserver
- › **MapLayers**: Integration von OpenLayers und OGC Service-Publikation von Geodaten
- › **MapFish**: Web Mapping Framework mit REST-Protokoll als Client-Server Schnittstelle



- › Integration von OpenLayers in RoR
- › Publizierung von Rails-Models als WFS, KML und GeoRSS

## Karte einbinden (Controller):

```
@map = MapLayers::Map.new("map")  
  do |map, page|  
    page << map.add_layer(MapLayers::GOOGLE)  
    page << map.zoom_to_max_extent  
  end
```





## View:

```
<html>
  <head>
    <%= map_layers_includes :google =>
      "ABQIAAAA..." %>
  </head>
  <body>
    <div id="map"/>
    <%= @map.to_html %>
  </body>
</html>
```



## Web-Service publizieren (Controller):

```
class WeatherStationsController <  
  ApplicationController
```

```
  map_layer :weather_stations,  
            :geometry => :geom
```

```
end
```

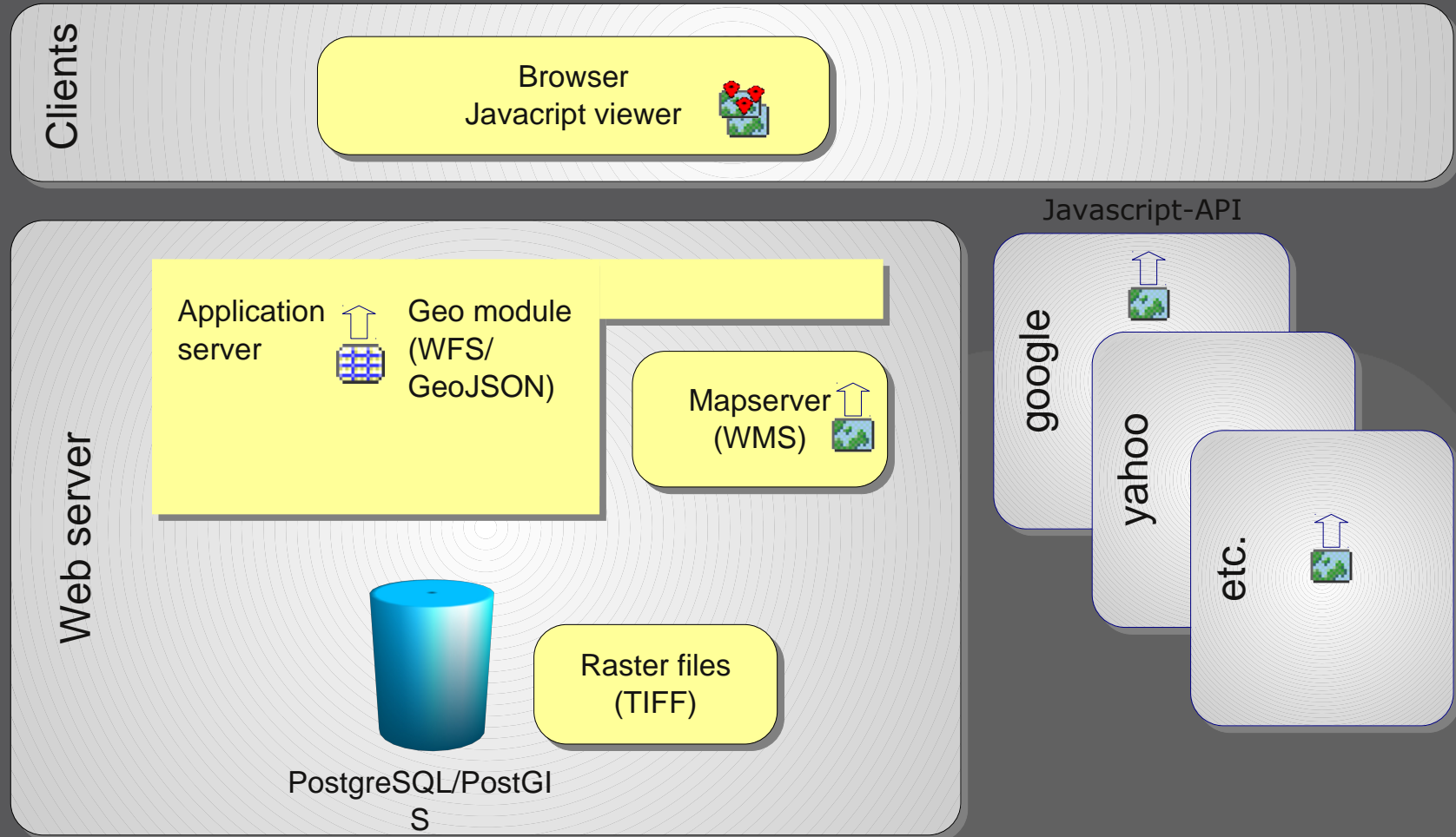
› **WFS-Service**

› **KML-Service**

› **GeoRSS-Service**



# „Web 2.0 GIS-Architektur“





# UMN Mapserver

## WMS/WFS Server:

```
require "mapscript"

class Mapserver
  include Mapscript

  def initialize(app, mapfile)
    @wms = MapObj.new(mapfile)
  end

  def call(env)
    req = OWSRequest.new
    #...
    retval = @wms.OWSDispatch(req)
    #...
  end
end
```



## Generator:

```
script/generate mapfish_resource Summit
```

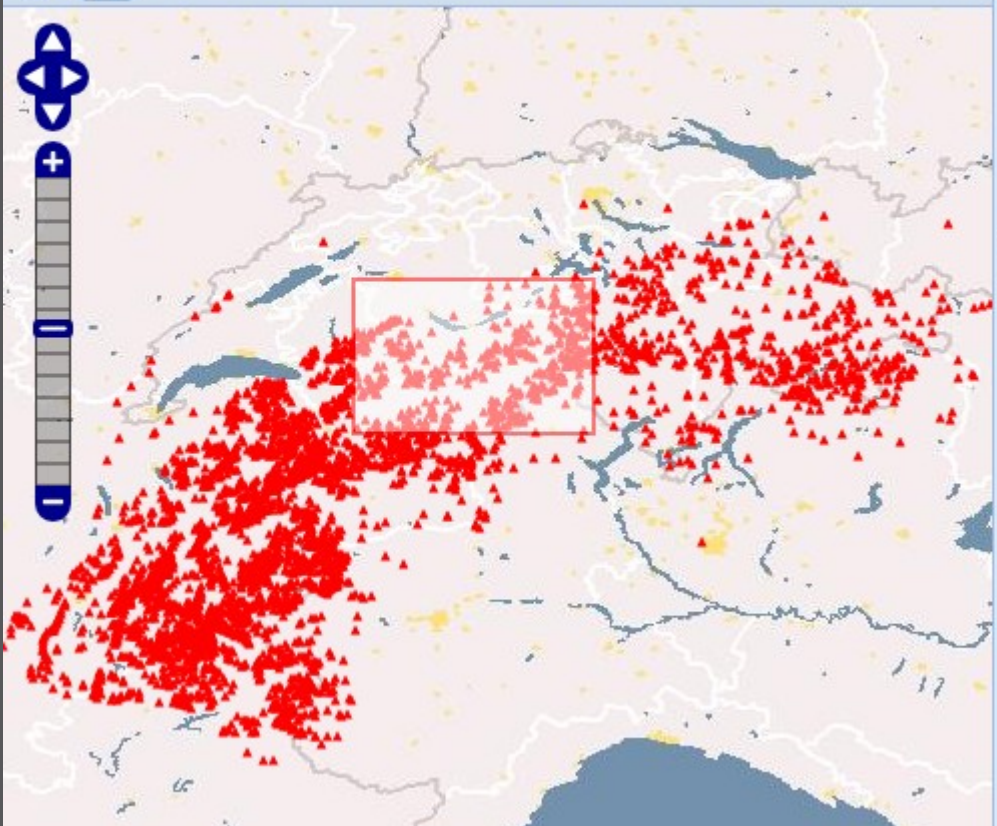
## Generierte Controller-Code:

```
def index
  @summits =
    Summit.find_by_mapfish_filter(params)
  render :json => @summits.to_gejson
end
```



## MapFish, Search demo

Map



Search criteria

Name:

Min elevation:

Max elevation:

Help note: search criteria specified above are used in conjunction with map queries.

Search results

name	elevation
Mont Pucel	3054
Mont Durand	3713
Mont de l'Étoile	3370
Monte Leone	3553
Monts Rouges	3167
Monte Cervandone	3210

Clear markers



# Mapfish – Unterschiede Pylon

- › Client kompatibel / REST-Schnittstelle
- › Riesige Auswahl an Rails-Plugins
- › Literatur, Schulungen, etc.
- › Automatisches OR-Mapping
- › Migrations
- › Integriertes Test-Framework



# Links

- › <http://rubyonrails.org/>
- › [http://wiki.github.com/pka/map\\_layers](http://wiki.github.com/pka/map_layers)
- › <http://mapfish.org/doc/implementations/rails.html>



**OSS GIS 2010?**

**Ruby on Rails!**



## Danke!



**Pirmin Kalberer**  
**<pka at sourcepole.ch>**