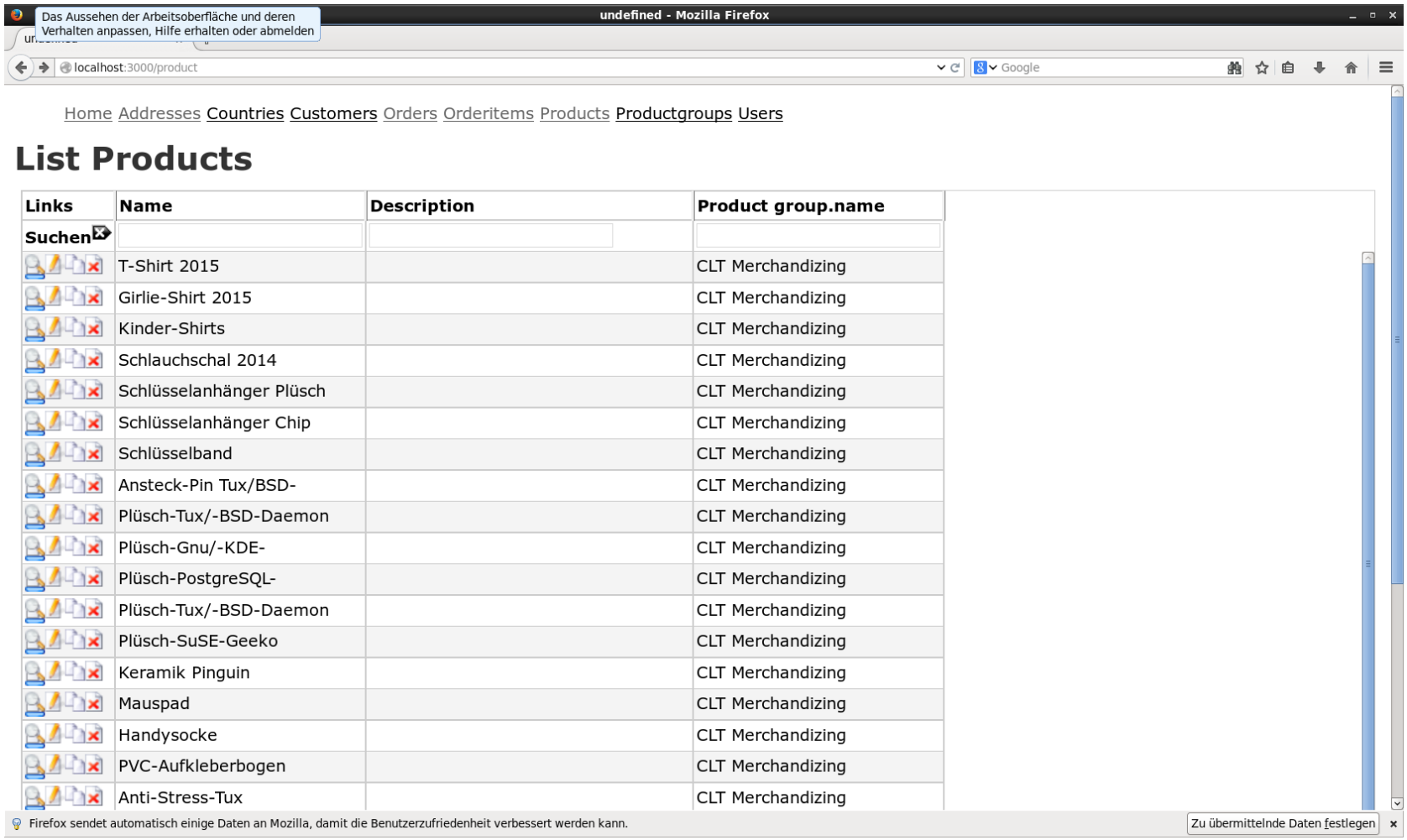
Die Esda Scaffolding Extension für Ruby on Rails

Daniel Schreiber

22. März 2015

Demo


Das Aussehen der Arbeitsoberfläche und deren Verhalten anpassen, Hilfe erhalten oder abmelden
undefined - Mozilla Firefox
localhost:3000/product
Google
Home Addresses Countries Customers Orders Orderitems Products Productgroups Users
List Products
Links
Name
Description
Product group.name
Suchen
T-Shirt 2015 CLT Merchandizing
Girlie-Shirt 2015 CLT Merchandizing
Kinder-Shirts CLT Merchandizing
Schlauchschal 2014 CLT Merchandizing
Schlüsselanhänger Plüsch CLT Merchandizing
Schlüsselanhänger Chip CLT Merchandizing
Schlüsselband CLT Merchandizing
Ansteck-Pin Tux/BSD- CLT Merchandizing
Plüsch-Tux/-BSD-Daemon CLT Merchandizing
Plüsch-Gnu/-KDE- CLT Merchandizing
Plüsch-PostgreSQL- CLT Merchandizing
Plüsch-Tux/-BSD-Daemon CLT Merchandizing
Plüsch-SuSE-Geeko CLT Merchandizing
Keramik Pinguin CLT Merchandizing
Mauspad CLT Merchandizing
Handysocke CLT Merchandizing
PVC-Aufkleberbogen CLT Merchandizing
Anti-Stress-Tux CLT Merchandizing
Firefox sendet automatisch einige Daten an Mozilla, damit die Benutzerzufriedenheit verbessert werden kann.
Zu übermittelnde Daten festlegen

Hintergrund

- Entstanden aus Esda Warenwirtschaft (ab 2006)
- Ziel: Standardoberfläche für Stammdatentabellen
  - Wenig Code pro Tabelle
  - Standardverhalten sinnvoll, aber anpassbar
  - Performant für große Datenmengen
- GPLv3
- Rails 3.2/4.1

Ruby on Rails - Kurz und knapp

- MVC-Framework für Webanwendungen
  - Models: Tabelle = Klasse, Record = Instanz
  - Controller: Logik hinter den URIs
  - View: rendert Template für Client
  - Router: mappt URIs auf Controllermethoden
- Designprinipien:
  - DRY - Don't repeat yourself
  - Convention over configuration
  - opinionated software
- Ausgefeilte Introspection-Möglichkeiten
- Voreinstellungen für Dateisystemlayout, Klasse- und Methodennnamen

ActiveRecord - Rails' ORM

- Philosophie: die Datenbank ist dumm
- SQLite, PostgreSQL, MyS^WMariaDB
- Felder aus DB auslesen
- Beziehungen im Model angeben
- Datenvalidierung im Model, nicht in DB
- Trigger
  - before_*/after_*
  - save, update, create
  - validate
  - destroy
  - after_commit, after_rollback
- unterstützt Transaktionen – auch verschachtelt

ActionPack

- ActionController – Anwendungslogik
  - HTTP Parameter verarbeiten
  - Daten aus Datenbank lesen/schreiben
  - Daten für Template zusammensammeln
  - Template angeben
  - i18n, content negotiation, CSRF Schutz
- ActionView – Templates rendern
  - verschiedene Templatesysteme: ERB, Builder, weitere aus gems
  - Partials
  - XSS Schutz per default
- Helper – Hilfen für Formulare, Links, Bilder etc

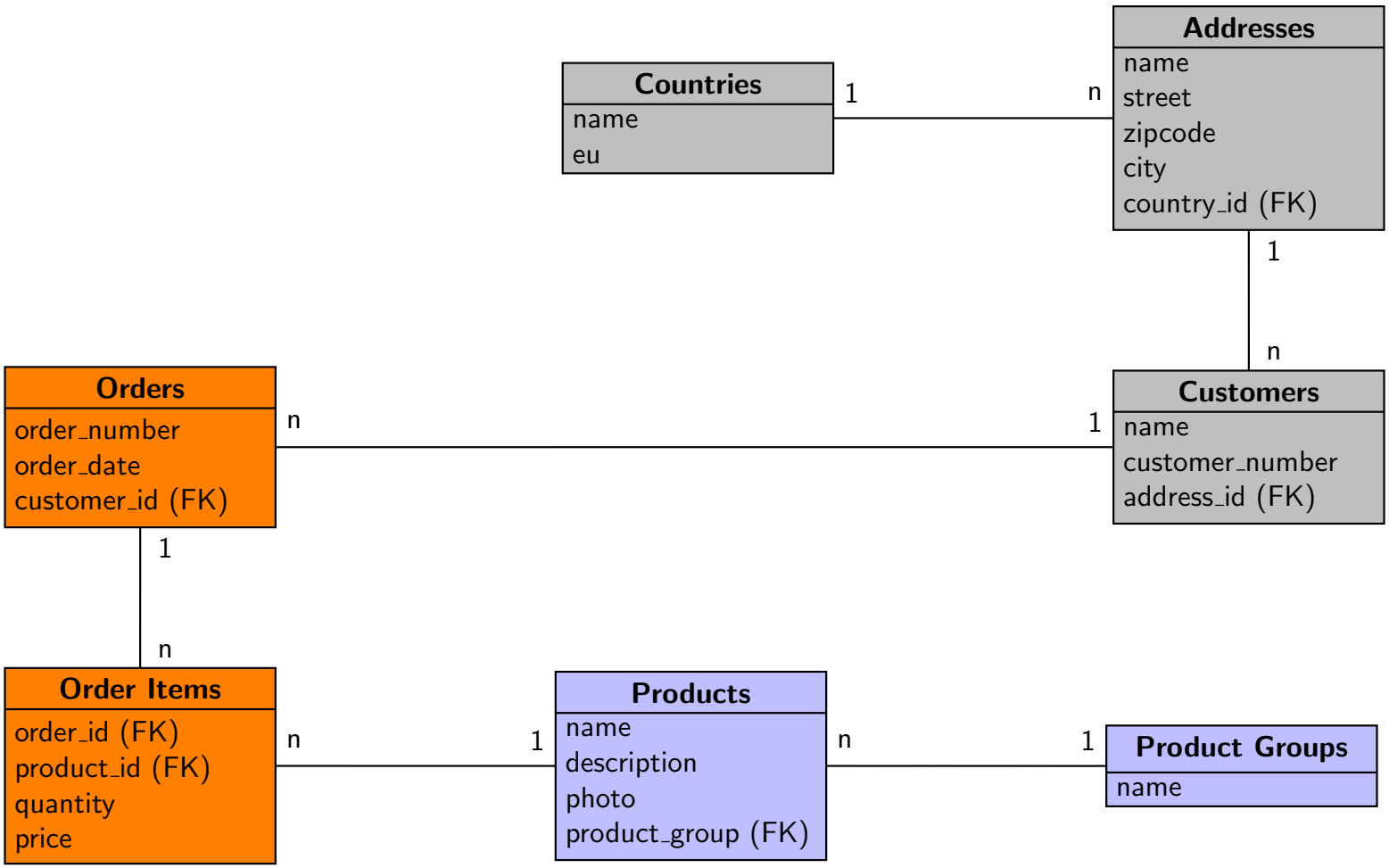
Beispielanwendung

```
git clone https://gitorious.org/scaffold_example_app/
    scaffold_example_app.git
cd scaffold_example_app
bundle install
rake db:migrate
rails server
```

Eine eigene Anwendung

```
rails new webshop -m "https://raw.githubusercontent.
    com/schrd/scaffolding_esda/master/app_template/
    scaffolding_app_template.rb"
cd webshop
bundle install
rake db:migrate
rake db:seed
# Models erstellen
# Controller generieren
```

Datenstrukturen



Datenmodell in Rails

```
rails g model country name:string eu:boolean
rails g model address name:text street:string city:
    string "zipcode:string{10}" country:references:
    index
rails g model customer name:string customer_number:
    integer address:references:index

rails g model product_group "name:string{30}"
rails g model product name:string description:text
    photo:binary product_group:references:index
```

Datenmodell in Rails

```
rails g model order order_number:integer order_date:
    date customer:references:index
rails g model order_item order:references:index
    product:references:index quantity:integer price:
    decimal
```

- Migrations anpassen: NOT NULL angeben

Controller generieren

```
for controller in country address customer product
    product_group order order_item; do
  rails g esda:scaffold_controller $controller
done

rake db:migrate
rake db:seed
rails s &
sleep 2
firefox http://localhost:3000
```

Reihenfolge der Felder in Formularen

- Standard: alle Datenbankfelder alphabetisch, Fremdschlüssel dereferenzieren
- Anpassung über @scaffold_fields Variable oder scaffold_fields Klassenmethode

```
class Address < ActiveRecord::Base
  belongs_to :country
  has_many :customers

  @scaffold_fields = %w(name street city zipcode
      country)
end
```

Felder in Auflistung ändern

- Standard: wie in Formularen
- Anpassung über @scaffold_browse_fields Variable

```
class Address < ActiveRecord::Base
  belongs_to :country
  has_many :customers

  @scaffold_fields = %w(name street city zipcode
      country)
  @scaffold_browse_fields = %w(name street city
      zipcode country.name)
end
```

Darstellung von Fremdschlüsselwerten

- Standard: Wert des Feldes name, wenn nicht vorhanden, dann to_s
- Überschreiben durch implementieren der scaffold_name Methode

```
class Address < ActiveRecord::Base
  belongs_to :country
  has_many :customers

  @scaffold_fields = %w(name street city zipcode
      country)
  @scaffold_browse_fields = %w(name street city
      zipcode country.name)

  def scaffold_name
    "#{self.name} aus #{self.city}"
  end
end
```

Änderbarkeit von Feldern

- Standard: alle Felder änderbar
- Überschreiben durch implementieren der feldname_immutable? Methode

```
class Order < ActiveRecord::Base
  belongs_to :customer
  has_many :order_items

  def order_number_immutable?
    not self.new_record?
  end
end
```

Extra Spalten in Auflistung

- Jede Modelfunktion kann als Spalte verwendet werden
- Hinzufügen zu scaffold_browse_fields

```
class Order < ActiveRecord::Base
  belongs_to :customer
  has_many :order_items
  @scaffold_browse_fields = self.scaffold_fields + ["
      value"]

  def value
    self.order_items.sum("quantity * price")
  end
end
```

Suche in Auflistung

- Berechnete Spalten können durchsucht werden
- Klassenmethode build_conditions_for_spaltenname implementieren
- Widget für Suchemaske erstellen

```
class Order < ActiveRecord::Base
  def self.build_conditions_for_value(table, params_part, param_name)
    p_ge = params_part[param_name].try(:[], :from)
    p_le = params_part[param_name].try(:[], :to)

    conditions, condition_params = [], []
    unless p_ge.blank?
      conditions << "sum(quantity␣*␣price)␣>=␣?"
      condition_params << BigDecimal.new(p_ge)
    end
    unless p_le.blank?
      conditions << "sum(quantity␣*␣price)␣<=␣?"
      condition_params << BigDecimal.new(p_le)
    end
    return [], [] if conditions.size == 0
    subselect = "#{table}.id␣IN␣(SELECT␣order_id␣from␣order_items␣group␣by␣order_id␣HAVING␣#{conditions.
          join('␣AND␣')})"
    return [subselect], condition_params
  end
end
```

Widget für Suchmaske

```
module OrderHelper
  def input_search_for_order_value(record_name,
      param_column_name, prefix, value, options)

    to_number_search_field_tag(record_name,
       param_column_name, prefix, value, options)
  end
end
```

Inline Association

- **eine** has_many Assocation kann inline dargestllt werden

```
class Order < ActiveRecord::Base
  def self.inline_association
    self.reflect_on_association(:order_items)
  end
end
```

Controller

- Parameter für scaffold Aufruf
- überschreiben einzelner Methoden
- before/after/around-Filter
- Für Scaffolding Templates: @instance-Variable passend befüllen

Nutzbare Helpermethoden

- Eigenes View Template im Standardpfad ablegen, z.B. app/views/order/edit.html.erb
- record_show(@instance)
- record_form(@instance)
- scaffold_field_name(@instance, feldname)
- scaffold_value(@instance, feldname)
- editable_scaffold_value(@instance, feldname)
- scaffold_field(@instance, feldname)

Eigenes Widget für Feld

- Helper implementieren

```
module ProductHelper
  def product_photo_field(record, field, name_prefix)
    file_field_tag(html_name(record.class, field,
       name_prefix)) +
    unless record.new_record?
      image_tag(download_url_for(record, :photo))
    else
      ""
    end
  end
end
```

TODO

- Dokumentation verbessern
- Auf Rails 4.2 Kompatibilität prüfen
- Responsive Design
- Design anpassbarer
- i18n verbessern
- Code aus Rails 2.x/3.x Zeiten aufräumen

Fragen, Anmerkungen, Wünsche, Bugs . . .

. . . bitte an . . .

schrd@linux-tage.de

https://github.com/schrd/scaffolding_esda