

## **Ruby on Rails -sovelluskehys; case: Translator**

Kristian Wahlroos - 014417003

Helsinki 28.12.2015

HELSINGIN YLIOPISTO

Tietojenkäsittelytieteen laitos

# Sisältö

<b>1</b>	<b>Johdanto</b>	<b>1</b>
1.1	Translator . . . . .	1
1.2	Ruby on Rails . . . . .	3
<b>2</b>	<b>Arkkitehtuuri</b>	<b>6</b>
<b>3</b>	<b>Yleisarkkitehtuuri ja keskeisimmät variaatiopisteet</b>	<b>7</b>
3.1	Yleiskuva kehysrakenteesta . . . . .	7
3.2	Kehyksen erikoistaminen sovelluskohtaisesti . . . . .	7
3.3	Suunnittelumallit . . . . .	7
3.4	Esimerkkitapaukset . . . . .	7
<b>4</b>	<b>Kehyksen ja sovelluksen arviointi</b>	<b>8</b>
4.1	Hyvät ja huonot puolet . . . . .	8
4.2	Laatuskenaariot . . . . .	8
4.3	ATAM . . . . .	8
	<b>Lähteet</b>	<b>9</b>

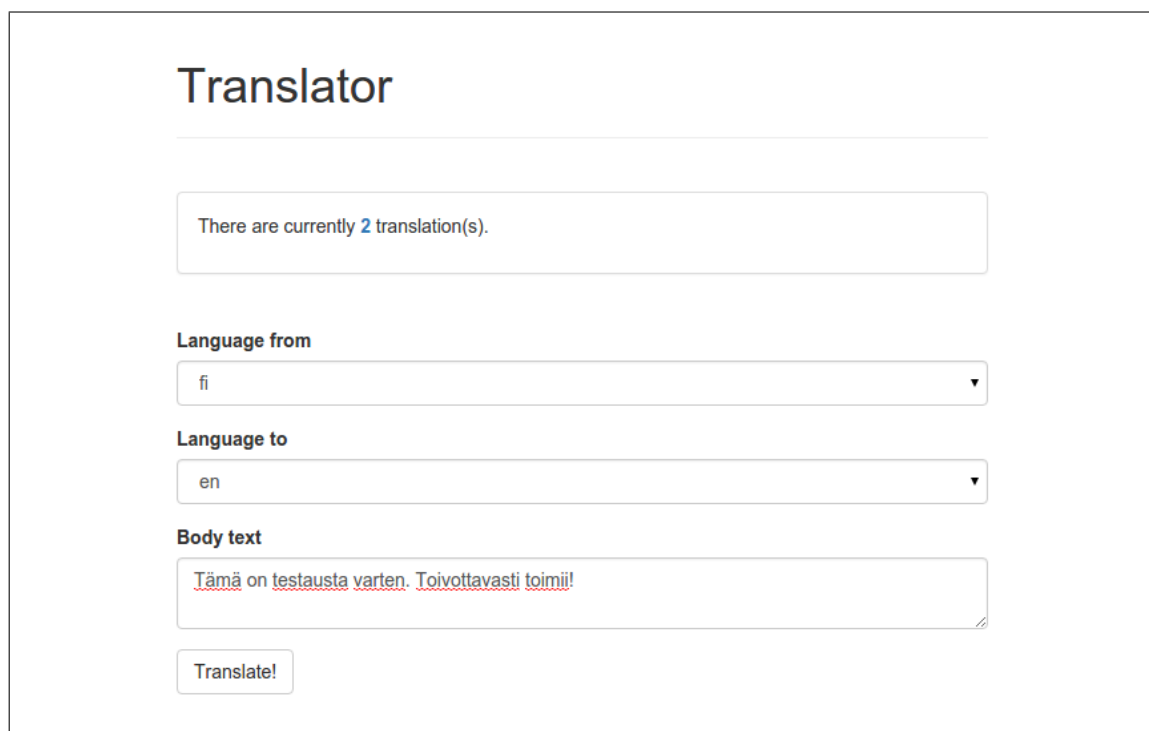
# 1 Johdanto

Tämän harjoitustyön tarkoituksena on tarkastella Ruby on Rails -web sovellushystä ja varsinkin sen arkkitehtuuria tekemäni esimerkkisovelluksen kautta. Teke-mäni esimerkkisovellus on yksinkertainen Internetissä oleva käännössivusto, jonka avulla käyttäjä pystyy kääntämään kolmen eri kielen välillä; suomen, englannin ja ruotsin. Itse sivusto on erittäin yksinkertainen eikä toiminnallisuutta ole hirveästi kääntämisen lisäksi, vaan itse keskittyminen tapahtuu sovelluksen pinnan alle kuten arkkitehtuuriin ratkaisuihin joita Ruby on Rail-kehys tuo mukanaan.

## 1.1 Translator

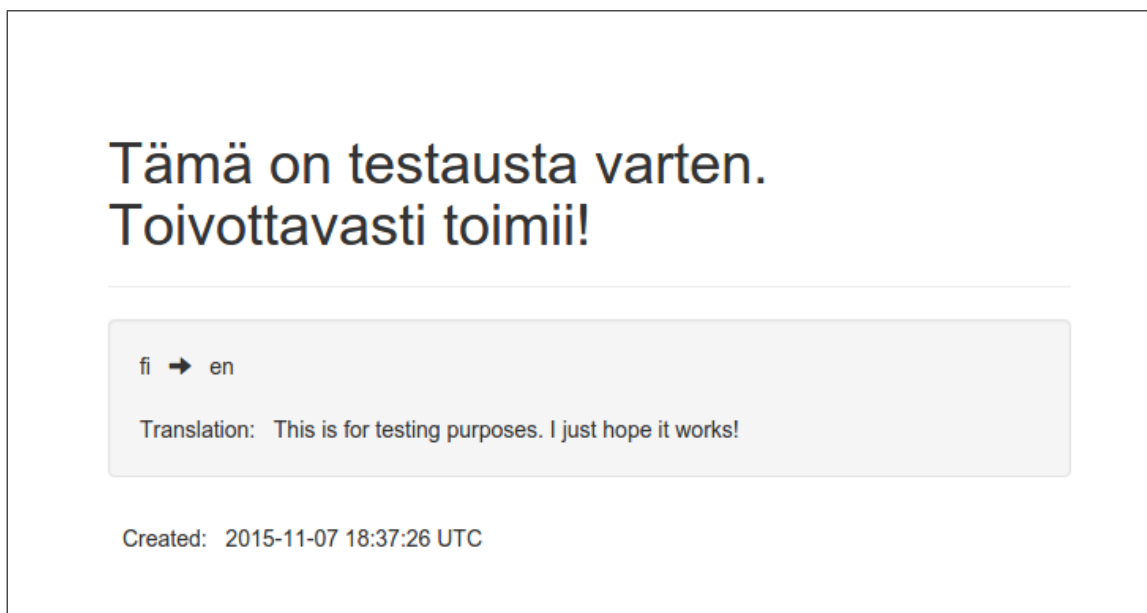
Sovellukseneeni viittaaan tästä alkaen nimellä *Translator*.

Translatorin pääidea on siis nimensä mukaisesta kääntää kieleltä toiselle. Käännök-sen logiikka hoidetaan ulkopuolista API:a käyttäen [YANDEX], joka hoitaa käytän-nössä kääntämisen sovelluslogiikan. Seuraavassa esitettävä kuvasarja esittää kääntä-misen käyttäjän näkökulmasta.



The screenshot shows a web application titled "Translator". Below the title, there is a message box that says "There are currently 2 translation(s)". Underneath, there are two dropdown menus: "Language from" with "fi" selected and "Language to" with "en" selected. Below these is a text input field labeled "Body text" containing the text "Tämä on testausta varten. Toivottavasti toimii!". At the bottom, there is a button labeled "Translate!".

Kuva 1: Käyttäjä haluaa kääntää tämän lauseen suomesta englantiin. Osoite: /



Kuva 2: Uudelleenohjataan selain näyttämään tulos. Osoite: /translations/:id

From	Body text	To	Translation	
fi	hei nimeni on Kaapo	en	hey, my name is Kaapo	<a href="#">View</a>
fi	Hei maailma!	en	Hello world!	<a href="#">View</a>
fi	Tämä on testausta varten. Toivottavasti toimii!	en	This is for testing purposes. I just hope it works!	<a href="#">View</a>

Kuva 3: Käyttäjä on navigoinut itsensä kaikkien käännösten sivulle klikkaamalla linkkiä pääsivulla (linkki on tallennettujen käännösten määrä). Osoite: /translations.

Edellinen kuvasarja eteni siis käyttäjän selaimessa seuraavasti:

Root (/) → tallennettu käännös (/translations/:id, jossa :id korvautuu tietokantaan tallennetun tietueen id:llä) → root (/) → kaikki käännökset (/translations).

## 1.2 Ruby on Rails

Ruby on Rails on itsessään sovelluskehys, joka on luotu vuonna 2003 David Heinemeier Hansson toimesta [ROR]. Sen näkyvimpiä ominaisuuksia ovat MVC-malli, REST-rajapinnat sekä Convention over Configuration -periaate. Nämä kaikki patternit tulevat esille käytännössä kaikissa Ruby on Rails -kehyksellä tuotetuissa projekteissa ja moni tunteeikin kehyksen juuri näistä edellä mainituista laatuviivista, joita kehyksen käyttö melko automaattisesti tuo mukanaan.

**MVC-malli** Ruby on Rails:n Modelina toimii ActiveRecordissa säilytettävät tietokantaobjektit. ActiveRecord huolehtii käytännössä koko toteutettavan järjestelmän logiikasta ja abstrahoi vahvan rajapinnan avulla konkreettista tietokantaa niin, että kehittäjän on helppo vaihtaa tietokanta sekä luoda uusia tietokantaobjekteja.

ActiveRecord itsessään toteuttaa Active Record-patternin [AR], joka määrittelee, että miten luokat ja tietokanta kuvautuu toisilleen. Rubyssä tämä kuvautuminen on tehty niin, että tietokannan tietue on aina luokka ja taulut luokan kenttiä. Luokkien metodeilla usein muutetaan vain ja ainoastaan tietokantataulun rivejä, jolloin olion sisäinen tila muuttuu. Luomassani projektissa uuden käännöksen ja sen tallentaminen tietokantaan on hyvin yksinkertaista tämän vuoksi, eikä kehittäjän tarvitse tietää, että taustalla pyörii SQLite tietokantana:

```
1 t = Translation.new
  ***
3 kenttien alustus
  ***
5 t.save
```

ActiveRecordin avulla myös suorat tietokantakyselyt on abstrahoitu pois. Kyselyt toimivat aina luokan nimen kautta, jolloin esimerkiksi omassa sovelluksessani kaikki suomesta käännetyt käännökset löytyvät seuraavasti:

```
1 Translation.all.where language_from: "fi "
```

Viewin vastuuta Ruby on Railsissä hoitaa ActionView, joka on näkymä tietokannan datalle. Kaikessa yksinkertaisuudessaan siis näytettävä HTML-sivu käyttäjälle. Se ladataan Controllerin toimesta oikeasta kansioista oikealla hetkellä ja usein HTML-tiedosto sisältääkin upotettua Ruby-koodia. Tämä Ruby-koodi on suurimaksi osaksi toiminnallisuutta näyttää vastaavan mallin tietoja HTML-muodossa. Esimerkiksi sovellukseni kaikki tehdyt käännökset näytettävä sivu on suurimmaksi osaksi upotettua Ruby-koodia, joka iteroi tietokannan kaikki käännökset ja hakee niiden tietokantataulut taulukkoon näytettäväksi dataksi.

Controlleria kutsutaan ActionController:ksi Ruby on Railsissa ja se vastaa koko järjestelmään kohdistuvien pyyntöjen reitittämisestä sekä yleisestä datan välityksestä. Omassa järjestelmässäni TranslationController vastaa kaikesta käännöksiin liittyvästä toiminnallisuudesta, kuten esimerkiksi aloitussivun näyttämisen logiikasta. Railsissa konventiot tulevatkin vahvasti esille, koska Translation-luokasta vastaavan kontrollerin on oltava samanniminen kuin luokkakin, mutta monikossa (englannin-s-päätteellä). Tähän kontrolleriin on myös kirjoitettu kaikki mahdolliset toiminnot, joita mallille on mahdollista tehdä ja näiden avulla rakennetaan myös näkymät.

**REST-rajapinta** Koko Ruby on Railsin käyttöönotto ja sen konventioiden noudattaminen tekee toteutettavasta järjestelmästä melkein pakostikkin REST-rajapintaa noudattavan järjestelmän, koska useat konventiot ovat pakollisia käyttää ja vaikka niiden kiertäminen on teknisesti mahdollista, on usein vain helpompi alistua valmiiksi määriteltyihin konventioihin. Esimerkiksi luodessani Translation-mallia, oli minun pakko luoda TranslationsController-kontrolleri, mutta ennen niiden linkittämistä on tiedostoon /config/routes.rb kerrottava, että mikä osoite linkittyy mihinkin kontrolleriin. Omassa projektissani olen lisännyt seuraavan rivin kyseiseen tiedostoon

```
1 resources :translations, only: [:show, :new, :create]
```

Tämän avulla olen saanut automaattisesti käyttöön seuraavat osoitteet

```
1 translations      POST /translations(..format)      translations#create
  new_translation  GET  /translations/new(..format)    translations#new
3 translation       GET  /translations/:id(..format)    translations#show
```

**Konventiot** - kansiorakenne (app) - routes.rb - Gems - database.yml - migraatiot - nimeäminen - <https://github.com/bbatsov/rails-style-guide> - luokkien linkitys (UML-style)

## 2 Arkkitehtuuri

Roolijako here (N-tier + <http://www.tutorialspoint.com/ruby-on-rails/images/rails-framework.gif>)



### **3 Yleisarkkitehtuuri ja keskeisimmät variaatiopisteet**

Luokkakaavio, sekvenssikaavio, stereotyyppit.

#### **3.1 Yleiskuva kehysrakenteesta**

#### **3.2 Kehyksen erikoistaminen sovelluskohtaisesti**

#### **3.3 Suunnittelumallit**

#### **3.4 Esimerkkitapaukset**

## 4 Kehyksen ja sovelluksen arviointi

### 4.1 Hyvät ja huonot puolet

Lähdekirjallisuus

### 4.2 Laatuskenaariot

### 4.3 ATAM

## Lähteet

- AR        Active Record -pattern. [https://en.wikipedia.org/wiki/Active\\_record\\_pattern](https://en.wikipedia.org/wiki/Active_record_pattern). [28.12.2015]
- BPS98     Bray, T., Paoli, J. ja Sperberg-McQueen, C., Extensible Markup Language (XML) 1.0. W3C Recommendation 10-February-1998. <http://www.w3.org/TR/1998/REC-xml-19980210>. [18.1.2000]
- EMN01     Erkiö, H., Mäkelä, M., Nykänen, M. ja Verkamo, I., Opinnäytetyön ulkoasun malli. Tieteellisen kirjoittamisen kurssiin liittyvä julkaisematon moniste, Tietojenkäsittelyopin laitos, Helsinki, 2001.
- ROR        RoR - Ruby on Rails. <http://rubyonrails.org/>. [28.12.2015]
- YANDEX    Yandex - Translator API. <https://tech.yandex.com/translate/>. [07.11.2015]