Internationalization

I18n

Florent Delannoy

Well Railed October 2011

Internationali~~z~~ation

s

I18n

Florent Delannoy

Well Railed October 2011

Use UTF-8

Use UTF-8

(Please)

History

Before 2007: i18n in Rails was a mess

Mr. Rails i18n: Sven Fuchs

Now everything's perfect and everyone's happy.

So!

```
t  :symbol
```

```
t :symbol
I18n.translate
```

app/views/home/base.html.erb

```
<h1>WellRailed i18n demo</h1>
<p>Hello!</p>
```

↓

app/views/home/step1.html.erb

```
<h1><%=t :title %></h1>
<p><%=t :hello %></p>
```

config/locale/en.yml

```
en:
  title: “WellRailed i18n demo,
in English (en, default)”
  hello: “Hi!”
```

Internationalisation is the process of designing a software application so that it can be adapted to various languages and regions without engineering changes.

Localisation is the process of adapting internationalised software for a specific region or language by adding locale-specific components and translating text.

— *Wikipedia*

i18n

1. Abstract strings away
2. Find and set locale

Set locale — good

Set locale — good

TLD

google.fr

Set locale — good

TLD

google.fr

Subdomain

fr.wikipedia.org

Set locale — good

TLD

google.fr

Subdomain

fr.wikipedia.org

URL params

example.com/fr/home
example.com/?lang=fr

Set locale — bad

Set locale — bad

Cookie

Breaks REST

Set locale — bad

Cookie

Breaks REST

accept-language header

Very unreliable, browser-dependant, breaks REST

Set locale — bad

Cookie

Breaks REST

accept-language header

Very unreliable, browser-dependant, breaks REST

GeoIP

Very bad.

(And breaks REST too)

l10n

1. Use defined locale...

2. ...to serve locale-specific content

```
l   :symbol
```

```
l :symbol
```

```
I18n.localize
```

app/views/home/step2.html.erb

```erb
<%=t :'step2.date' %> <%=l Time.now, :day => @day %>

<%=t :'step2.day' %> <%=l Time.now, :format => :daymonth %>

<%=t :'step2.money' %> <%= number_to_currency @amount %>

<%=t :'step2.readable' %> <%= number_to_human @bignum %>

<%=t :'step2.distance' %> <%= number_to_human @distance, :units => :distance %>

<%=t :'step2.when.world_cup',
     :since_finals => distance_of_time_in_words(Time.now, @since_finals) %>
```

So do I have to rewrite the boilerplate for each locale?

No! i18n-rails gem

https://github.com/svenfuchs/rails-i18n/

Default localisation standard for most languages

More

Left-To-Right (LTR) support: Arabic, Hebrew...

Use a different backend:

- store your translations in the db / somewhere else
- add your own i18n logic

Different pluralisation rules: Russian, Bosnian

Translate routes

Web front-end: github.com/mynewsdesk/translate

Globalize3 gem: translations in your model

- e.g. `post.title` changes depending on the locale

Thanks!