Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií
Ilkovičova 2 , 842 16 Bratislava

Extrakcia dát z webu

[WebExtraction]

Metodika pre testovanie

Tím: č. 16, WebX
Vedúci tímu: Ivan Srba
Členovia tímu: Ján Brechtl, Tomáš Juhaniak, Martin Kalužník, Rastislav Krchňavý, Michal Kren, Martin Lacek, Andrej Vaculčiak
Akademický rok: 2016/2017
Autor: Andrej Vaculčiak
Verzia číslo: 1.0
Dátum poslednej zmeny: 06.05.2017

1 Úvod

Účelom tohto dokumentu, ako aj z názvu vyplýva, je oboznámiť čitateľa so spôsobmi vytvárania testov spolu so spôsobmi testovania serverovej časti systému.

Touto metodikou sa riadi tím č. 16 WebX v rámci predmetu tímový projekt v akademickom roku 2016/2017. Metodikou sa riadia všetci členovia tímu, ktorí pracujú na serverovej časti systému.

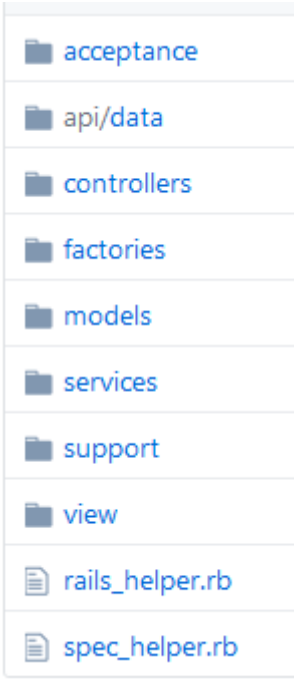
2 Skratky a značky

3 Postupy

V nasledujúcich podkapitolách sú uvedené postupy a spôsoby písania testov pre serverovú časť systému naprogramovanú v Ruby on Rails.

3.1 Umiestnenie testov

Všetky testy pre serverovú časť sa nachádzajú priamo v projekte, a to v adresári *spec*. V tomto adresári sa nachádzajú podadresáre, ktorými sa testy kategorizujú podľa štruktúry celého projektu. Situácia je zobrazená na Obr. 1.



Obr. 1 – Podadresáre adresára spec

Podľa toho, na ktorej časti aktuálne pracujeme, podľa toho si zvolíme príslušný adresár a doňho buď pridáme nový súbor vo formáte .rb s názvom **<nazov>_spec.rb**.

3.2 Vytvorenie testu

Ak ideme vytvárať test pre súčasť systému, ktorá ešte nemá vytvorený samostatný súbor, vytvoríme najskôr nový súbor. Ak už tento súbor existuje, doplníme doň len ďalší test, ktorý testuje danú, nami implementovanú funkcionalitu.

Každý súbor má formát:

```
require 'rspec'
require 'rails_helper'

describe '<Example of test>' do

      it '<should test something>' do
```

```
            <doplnenie samotného testu>
      end

end
```

Ak chceme len doplniť ďalší test, pridáme ďalšiu stať typu `it '<should test something>' do,` do už existujúceho súboru s testami.

3.2.1 Kedy vytvárať testy?

Nakoľko sa snažíme o TDD (Test Driven Development), najvhodnejšie je vytváranie testov ešte pred samotnou implemetnáciou funkcionality, no nakoľko nie vždy vieme, ako bude výsledná funkcionalita implementovaná, aké funkcie sa použijú a v akom formáte bude výstup alebo dokonca testujeme front-end, kde sa tiež nie vždy dá určiť, čo všetko do jednotlivých obrazoviek pribudne a ako to presne bude vyzerať, vytvárame testy po skončení implementácie funkcionality.

3.2.2 Používanie „factories“

Factories alebo továrne slúžia na vytváranie objektov, rôznych tried a rôznych atribútov, pričom ich primárny účel je minimalizovať duplicitu kódu alebo rozsah kódu samotného testu kvôli nutnosti definície atribútov jednotlivých objektov použitých v teste.

Jeden súbor, napríklad factory pre používateľov *users.rb*, obsahuje viacero tovární, ktoré vytvoria rôznych používateľov.

Príklad:

```
factory :other_user, class: User do
      email 'test1@example.com'
      password ‘password’
      name ‘name1’
      surname ‘surname1’
end
```

Táto factory nám vytvorí objekt používateľa triedy User so špecifickými atribútmi. Ak chceme v teste vytvoriť objekt, ktorý nám takáto factory vráti, použijeme nasledovný príkaz:

```
user = create(:other_user)
```

3.2.3 Spôsob a konvencie pri písaní testu

1. Definície a popisy sú vždy písané v angličtine.
2. Pri písaní kódu testu sa postupuje podľa Metodiky pre vývoj
3. Test **NESMIE** obsahovať operáciu return.
4. Pre kontrolu správnosti výsledku nepoužívame konštrukciu **if** ale **expect**, ktorá v prípade nesplnenia stanovenej podmienky spôsobí pád testu.

Príklad:

Namiesto:

```
if (a == 4)
      return
else
      return ERROR
end
```

Použijeme:

```
expect(a).to eq 4
```