Behaviour driven development (BDD) i Gherkin

Behaviour driven development je agilna tehnika u razvoju softvera koja podstiče saradnju između programera i netehničkih ili poslovnih učesnika u razvoju softvera. Prvi put se javlja 2003. kao odgovor na Test driven developement. Tehnika je napredovala poslednjih nekoliko godina.

BDD se fokusira na dobijanje jasnog razumevanja željenog ponašanja softvera, kroz diskusiju zainteresovanih strana. Ova tehnika proširuje TDD pisanjem test scenarija na prirodnom jeziku, koji mogu razumeti ljudi koji se ne bave programiranjem. Ovo pomaže programerima da se fokusiraju na kodiranje, umesto na tehničke detalje, sa druge strane smanjujući napor potreban za "prevođenje" između jezika u kome je kod pisan i domen-specifičnog jezika koji koriste korisnici, projekt-menadžeri i ostali netehnički učesnici u razvoju softvera. Više detalja na :
http://pythonhosted.org/behave/philosophy.html

Tvorac BDD principa je *Dan North*, čiji blog je dostupan na:

http://dannorth.net/

Gherkin

Gherkin predstavlja domen-specifičan jezik blizak poslovnim korisnicima, pomoću kojeg je moguće opisati ponašanje softvera, bez ulaženja u detalje implementacije.

Korišćenje Gherkin jezika ima dva cilja:

- dokumentacija i
- automatizacija testova

Još jedna korisna osobina Gherkin jezika, u kombinaciji sa nekim od alata koji se koriste za BDD (Cucumber, Behave, Lettuce, JBehave):

"when it yells in red it's talking to you, telling you what code you should write."

Gherkin jezik, korišćenjem predefinisanog formata, omogućava opis ponašanja aplikacije, razbijanjem ponašanja u specifične situacije (primere ponašanja). Opis ponašanja aplikacije je proces u

kojem je potrebno da učestvuju sve strane (klijenti, poslovni analitičari, programeri i sl.) u cilju dobijanja što kompletnijeg opisa ponašanja. Opisi ponašanja su u bilo kom momentu podložni promenama.

Više o “korisničkim pričama”:

http://dannorth.net/whats-in-a-story/

Gherkin sintaksa

Opis funkcionalnosti aplikacije deli se na funkcije, koje se po Gherkin sintaksi nazivaju **Feature**. Opisi funkcija nalazi se u fajlovima koji imaju ***.feature*** ekstenziju**.**

Gherkin je linijski orijentisan jezik, poput Python-a, koji koristi uvlačenje sadržaja (*indentation*) za definisanje strukture. Moguće je koristiti *tab* ili *space,* pri čemu je preporuka da se koristi *space,* zbog bolje portabilnosti između operativnih sistema. Većina linija u ***.feature*** fajlu imaju na svom početku neku rezervisanu reč.

Pisanje komentara je moguće na bilo kojoj liniji u fajlu, pri čemu komentari počinju sa 0 ili više praznih mesta, praćeno znakom **#** i tekstom komentara.

Parser deli strukturu ***.feature*** fajlova u *features (*funkcije*), scenarios (*scenariji ponašanja*)* i *steps* (koraci)*.* Prilikom pokretanja automatskih testova, svaka *steps* sekcija uparuje se sa implementacijom koraka na ciljnom programskom jeziku. Sadržaj ***.feature*** fajla izgleda ovako:

```
1: Feature: Some terse yet descriptive text of what is desired
2:   Textual description of the business value of this feature
3:   Business rules that govern the scope of the feature
4:   Any additional information that will make the feature easier to understand
5:
6:   Scenario: Some determinable business situation
7:     Given some precondition
8:       And some other precondition
9:      When some action by the actor
10:      And some other action
11:      And yet another action
12:     Then some testable outcome is achieved
13:      And something else we can check happens too
14:
15:  Scenario: A different situation
16:      ...
```

Linija nakon rezervisane reči **Feature** je početak opisa jedne funkcije. Linije 2-4 neće biti parsirane, i služe opisu namene ove funkcije. Linija 6, nakon rezervisane reči **Scenario** je početak prvog scenarija. Na linijama 7-13 nalazi se opis koraka prvog scenarija. Od linije 15 počinje drugi scenario.

Rezervisane reči

Gherkin sintaksa obuhvata ograničen broj rezervisanih reči namenjenih lakšem parsiranju od strane alata i boljem razdvajanju semantički različitih segmenata opisa funkcionalnosti aplikacije.

Feature

Rezervisana reč **Feature** nalazi se na početku opisa svake funkcije, praćena naslovom i eventualnim detaljnijim opisom funkcije. Svaki **Feature** sadrži jedan ili više scenarija ponašanja.

Scenario

Rezervisana reč **Scenario** nalazi se na početku opisa svake „varijante“ izvršavanja određene funkcije (specifične situacije). Nakon ove ključne reči nalazi se kratak opis scenarija, koji ga jasno može razlikovati u odnosu na druge scenarije. Svaki **Scenario** sastoji se od niza koraka, koji detaljno opisuju njegovo izvršavanje, a za šta se koriste rezervisane reči **Given, When i Then.**

Given

Rezervisana reč **Given** predstavlja početak sekcije, u okviru koje je potrebno pripremiti početno stanje sistema, na osnovu kog se vrši opis ponašanja. U okviru **Given** sekcije potrebno je izbegavati pominjanje bilo kakve korisničke interakcije.

Primer: kreiranje sloga u bazi podataka, postavljanje baze podataka u određeno stanje, **kreiranje određene strukture *Workspace-a*** *i sl.*

When

Rezervisana reč **When** nalazi se na početku sekcije namenjene spoljašnjoj interakciji (korisnika ili nekog drugog sistema/dela sistema) sa našim sistemu, koja proizvodi (ili ne) neku promenu stanja sistema.

Primer: interakcija sa GUI elementom, poziv određene funkcije i sl.

Then

Rezervisana reč **Then** predstavlja početak sekcije u okviru koje se vrši provera rezultata izvršavanja određene funkcije sistema. Rezultati bi se trebali odnositi na poslovnu vrednost vidljivu akteru koji je stupio u interakciju sa našim sistemom (izveštaj, poruka na GUI i sl.). Proverava se poklapanje rezultata sa opisanim očekivanim rezultatima.

And, But

U slučaju pojave više **Given, When** ili **Then** sekcija, moguće je iskoristiti rezervisane reči **And** ili **But** u cilju dobijanja prirodnije strukture opisa. Prilikom parsiranja reči **And/But** biće zamenjeni jednom od reči **Given/When/Then,** i to poslednjom koja je prethodno iskorišćena. Primer:

```
Scenario: Multiple Givens
  Given one thing
  Given another thing
  Given yet another thing
  When I open my eyes
  Then I see something
  Then I don't see something else
```

```
Scenario: Multiple Givens
  Given one thing
    And another thing
    And yet another thing
  When I open my eyes
  Then I see something
    But I don't see something else
```

Background

Backround sekcija omogućava postavljanje određenog stanja aplikacije za sve scenarije u okviru jednog feature-a (stanje do kojeg se dođe u okviru jednog scenarija inače se ne prenosi u druge scenarije). Piše se na isti način kao i **Scenario** i izvršava se pre svakog scenarija.

```
Feature: Multiple site support
  As a Mephisto site owner
  I want to host blogs for different people
  In order to make gigantic piles of money

  Background:
    Given a global administrator named "Greg"
    And a blog named "Greg's anti-tax rants"
    And a customer named "Dr. Bill"
    And a blog named "Expensive Therapy" owned by "Dr. Bill"

  Scenario: Dr. Bill posts to his own blog
    Given I am logged in as Dr. Bill
    When I try to post to "Expensive Therapy"
    Then I should see "Your article was published."

  Scenario: Dr. Bill tries to post to somebody else's blog, and fails
    Given I am logged in as Dr. Bill
    When I try to post to "Greg's anti-tax rants"
    Then I should see "Hey! That's not your blog!"

  Scenario: Greg posts to a client's blog
    Given I am logged in as Greg
    When I try to post to "Expensive Therapy"
    Then I should see "Your article was published."
```

Scenario outline

U nekim slučajevima, potrebno je testirati isti set koraka (scenario) za različite vrednosti. Da bi se izbeglo kopiranje istih scenarija, uz izmenu samo datih vrednosti, moguće je korišćenje **Scenario outline**-a. U tu svrhu potrebno je iskoristiti tzv. **placeholder**-e, koji se nalaze između znakova **< >** i predstavljaju mesto na kome će biti ubačena neka od vrednosti navedenih u **Examples** sekciji. Primer:

```
Scenario: eat 5 out of 12
  Given there are 12 cucumbers
  When I eat 5 cucumbers
  Then I should have 7 cucumbers

Scenario: eat 5 out of 20
  Given there are 20 cucumbers
  When I eat 5 cucumbers
  Then I should have 15 cucumbers
```

```
Scenario Outline: eating
  Given there are <start> cucumbers
  When I eat <eat> cucumbers
  Then I should have <left> cucumbers

  Examples:
    | start | eat | left |
    |  12   |  5  |  7   |
    |  20   |  5  |  15  |
```

Step data

Svakom koraku mogu biti pridruženi podaci u obliku teksta ili tabele.

Tekst je moguće proslediti kao blok teksta između trostrukih navodnika """. Nivo indentacije mora biti najmanje zadržan u svim linijama prosleđenog teksta. Primer:

```
Scenario: some scenario
  Given a sample text loaded into the frobulator
    """
    Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do
    eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut
    enim ad minim veniam, quis nostrud exercitation ullamco laboris
    nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in
    reprehenderit in voluptate velit esse cillum dolore eu fugiat
    nulla pariatur. Excepteur sint occaecat cupidatat non proident,
    sunt in culpa qui officia deserunt mollit anim id est laborum.
    """
  When we activate the frobulator
  Then we will find it similar to English
```

Tabele se mogu proslediti jednostavnim unosom tabele kao u **Examples** sekciji, uvučene u odnosu na korak na koji se odnose. Broj kolona mora biti fiksan. Kolone su razdvojene znakom |. Primer:

```
Scenario: some scenario
  Given a set of specific users
     | name      | department  |
     | Barry     | Beer Cans   |
     | Pudey     | Silly Walks |
     | Two-Lumps | Silly Walks |

 When we count the number of people in each department
 Then we will find two people in "Silly Walks"
  But we will find one person in "Beer Cans"
```

Tags

Moguće je "označavanje" delova ***.feature*** fajla, kojima je u okviru poziva funkcija alata, ili u okviru implemetacije testova, moguće pristupati. Moguće je označavanje sledećih elemenata ***.feature*** fajla: **feature, scenario, scenario outline.**

Behave

Behave je alat za BDD u programskom jeziku Python. Omogućava pokretanje ***.feature*** fajlova i njihovo mapiranje na implementaciju napisanu u ovom programskom jeziku.

Za korišćenje Behave alata potrebno je kreirati folder u kojem će se nalaziti svi ***.feature*** fajlovi, i dodatni podfolder **steps** u okviru kojeg će se nalaziti fajlovi sa ***.py*** ekstenzijom u kojima će se nalaziti implementacije koraka. Podfolder **steps** za potrebe dosadašnjeg rada treba ostaviti prazan.

Korisni linkovi:

1. http://pythonhosted.org/behave/