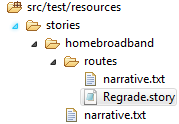
1. стурктура папки

RESOURCES\STORIES\capabilities{**ONE WORD?**}\features{**ONE WORD?**}\story1+story2...



**Всё маленькими буквами? Кроме стори файлов**

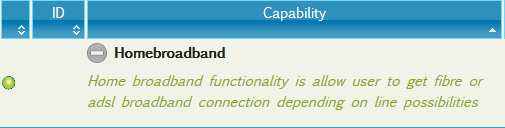
2. Содержание папок

capabilities описываются с помощью narrative.txt файла:

Narrative:

Home broadband functionality is allow user to get fibre or adsl broadband connection depending on line possibilities

В итоге этот текст попадёт в Requirement tracking секцию:



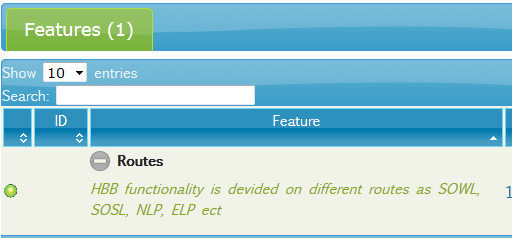
Features это subfolder of capability(optional)

описываются с помощью narrative.txt файла:

Narrative:

HBB functionality is devided on different routes as SOWL, SOSL, NLP, ELP ect

В итоге этот текст попадёт в Requirement tracking секцию ниже capability:



Стори файлы описываются при помощи narrative текста внутри стори:

Narrative:

Regrade functionality allows to change users's current package

For this purposes he should switch to regrade route and register order as regrade customer

Current package selection on the reg form is required step

Regrade line number: 01179575564

Meta:

@tag priority:P1,test type:regression

@Priority p1

@Area hbb

@Component test

@Type regression

Scenario: 01.For the regrade customers, regrade screen should be shown

Given user is on the home page

When go to HBB page

And click registration

And select media 'NO\_MEDIA'

And click consumers plans

And select route 'orange mobile'

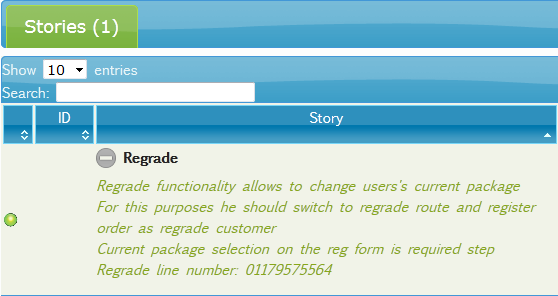
And select contract 'paym'

And select discount 'broadband'

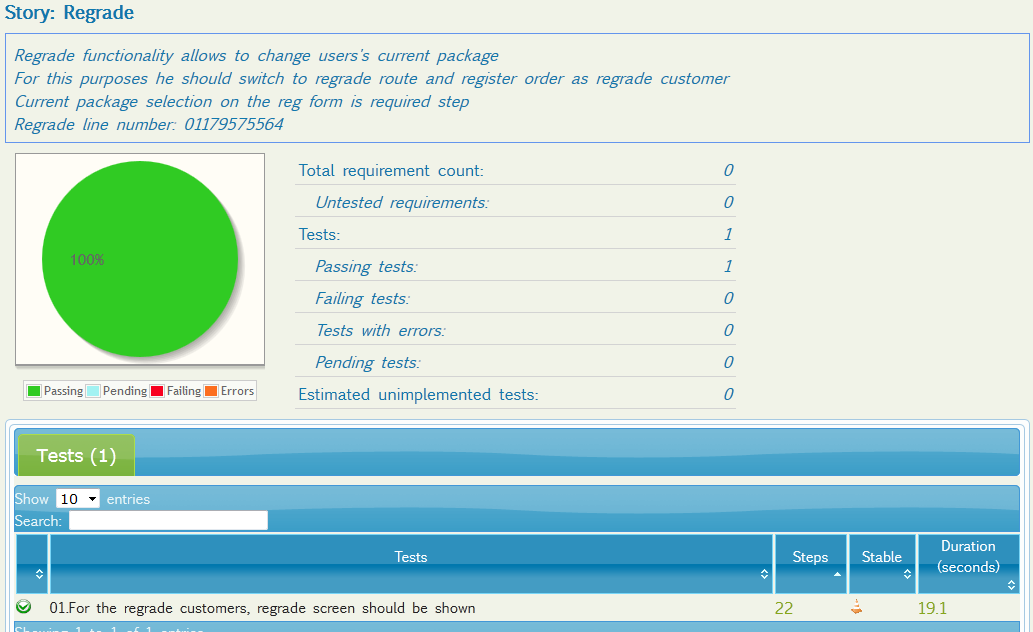
And do linecheck with '01179575564', 'BS167DX'

Then regrade screen is shown

В итоге этот текст попадёт в Requirement tracking секцию ниже feature:



Сценарии внутри стори файла описываются своим названием после слова Scenario:



3. Таги:

Существует возможность добавлять кастомные поля фильтрации в отчёт. Реализуется посредствам «@tag» meta

Meta:

@tag priority:P1,test type:regression

В этом случае будут добавлены 2 фильтра

Priorities и Test Types



4. Фильтрация

Существует возможность, посредствам jbehave запускать определённые стори. Реализуется с помощью тегов внутри стори(см выше):

@Priority p1

@Area hbb

@Component test

@Type regression

Выбор фильтра передаётся через командную строку:

clean integration-test thucydides:aggregate **-Dmetafilter="+Component test"**

заупустить всё – «\*»

**-Dmetafilter="groovy: (Area == 'edit' && Project == 'bbreg') || Area == '1'"**

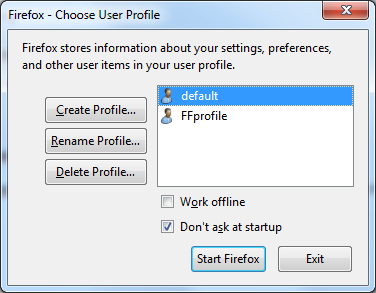
Другой вариант – использваония findStoriesCalled("mask string”); метода:

findStoriesCalled("\*\*/myfeature/\*.story");

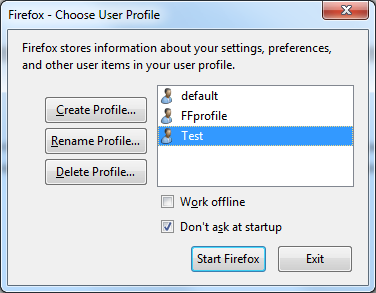
5. Создание профайла для FireFox и настройка по запуску тестов с данным профайлом

Закрываем сам FireFox если он открыт. Выполняем а Rune (в Win 7 Пуск и в строке поиска) «firefox.exe -p»

Откроется вот такое окошко:

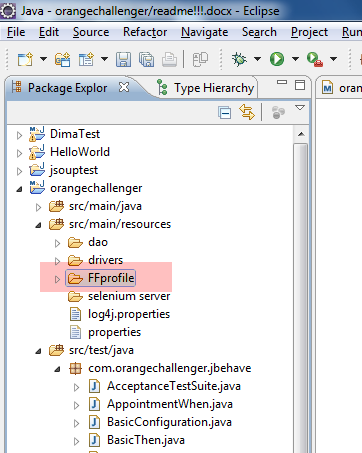


В нем нажимаем «Create profile» создаем профайл, обзываем его как-нить (например «Test») и выбираем папку, куда он сохраниться. Жмем «Finish» и он появляется у нас в списке:



Жмем «Start Firefox» нам открывается чистенький Firefox, ставим в него все необходимые нам аддоны, настраиваем их и закрываем браузер.

Затем идем в папочку куда мы его сохранили, перем все ее содержимое и копаем в папку FFprofile нашего проекта:



Теперь, в файл с настройками сусидизиса - **thucydides.properties** добавляем строчку **webdriver.firefox.profile=src/main/resources/FFprofile**

Все, готово, теперь у нас будет тянуться необходимый нам профиль при старте тестов в FF.