

HW #06: Web Spy

1. Описание задания	2
1.1. Web-шпион (spy)	2
1.2. Требования к реализации	4
2. Рекомендации	5
3. Критерии оценивания	6
4. Инструкция по отправке задания	7
5. FAQ (часто задаваемые вопросы)	9
6. Полезные книги	10

1. Описание задания

В этом задании вам нужно написать консольное приложение, которое будет шпионить за интересующим вас конкурентом. В качестве примера “конкурента” возьмем GitLab, который предоставляет ряд платных и бесплатных инструментов.

Ваше приложение должно уметь скачивать страницы из сети Интернет, парсить содержимое HTML, проверять число платных и бесплатных предложений, и самое главное – содержать полный набор тестов, проверяющих кодовую базу без доступа к сети Интернет. Цель задания:

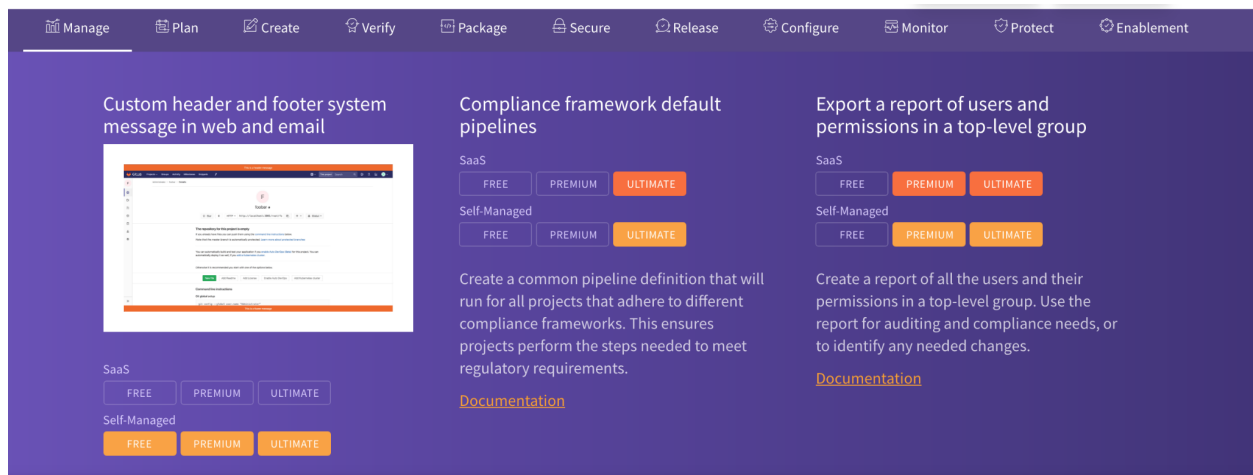
1. Вспомнить как или научиться делать запросы с помощью requests;
2. Научиться разделять интеграционные/системные и юнит-тесты;
3. Научиться парсить содержимое HTML.

1.1. Web-шпион (spy)

Консольный интерфейс библиотеки:

```
$ python3 task_*web_spy.py gitlab1  
free products: 9  
enterprise products: 20
```

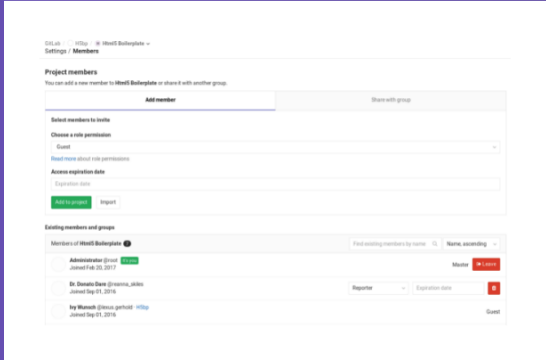
Рассмотрим продукты компании, представленные на странице <https://about.gitlab.com/features/>:



¹ Интерфейс расширяемый, в дальнейшем вы сможете добавить другие интересующие страницы и правила парсинга.

Обозначим free products те продукты компании, у которых есть бесплатная облачная версия “Available in GitLab SaaS Free”. Для простоты, все остальные продукты будем считать enterprise products (“Not available in SaaS Free”), даже если у них есть бесплатная self-managed версия. Пример free product:

Granular user roles and flexible permissions



SaaS

FREE PREMIUM ULTIMATE

Self-Managed

FREE PREMIUM ULTIMATE

```
▼<a data-placement="top" data-toggle="tooltip" href="/pricing/#gitlab-com" title="Available in GitLab SaaS Free">  
<div class="badge available">FREE</div>  
</a>
```

Страницы в интернете эволюционируют, поэтому дамп (html) страницы для тестирования сохранен на github курса по адресу:

- [github:big-data-team/python-course/./gitlab_features.html](https://github.com/big-data-team/python-course/./gitlab_features.html)

Ключевой акцент задания – это создание Mock внешних объектов, а также разработка и тестирование функциональности без доступа к сети Интернет. Поэтому pytest должен предоставлять возможность запускать только юнит-тесты, только интеграционные тесты, или и те и другие. Пример pytest marker’a slow можно взять с GitHub проекта курса (см. conf_test.py):

- <https://github.com/big-data-team/python-course>



Интеграционные тесты должны проверять, что число “free products” и “enterprise products” остались без изменений. Если же это не так, с помощью assert-сообщения должна выводиться разница между ожидаемыми и полученными величинами.

1.2. Требования к реализации

Все тесты на базовую функциональность без доступа к сети интернет должны быть помечены с помощью:

```
pytest.mark.slow
```

Все тесты, требующие доступ к сети интернет должны быть помечены с помощью:

```
pytest.mark.integration_test
```

Вызов

```
pytest -v test_*web_spy.py --skip-integration
```

должен использовать для тестирования дампы HTML страницы GitLab на вызовы `requests.get`. Дамп HTML лежит на github курса и будет доступен в рабочей директории проекта в период тестирования под именем “`gitlab_features.html`”. Все тесты должны проходить (не должно быть FAILED тестов, могут быть только PASSED и SKIPPED).

Вызов

```
python3 task_*web_spy.py gitlab
```

должен выводить на экран (STDOUT) информацию по числу продуктов компании разного типа:

```
free products: 9
enterprise products: 20
```

Вызов

```
pytest -v test_*web_spy.py --skip-slow
```

должен обращаться в сеть Интернет для скачивания страницы GitLab и сравнивать со значениями free products и enterprise production, представленных в дампе HTML страницы GitLab по примеру того, что выложено на github курса. Файл для сверки “`gitlab_features_expected.html`” должен использоваться в тест-функции. Он будет положен в рабочую директорию проекта в период тестирования.

Шаблон assert-сообщения (**одна строка, без переносов!**):

```
f"expected free product count is {*}, while you calculated {*};  
expected enterprise product count is {*}, while you calculated {*}"
```

2. Рекомендации

Официальная документация:

- <http://python-requests.org/> + <https://pypi.org/project/requests/>
- BeautifulSoup: <https://www.crummy.com/software/BeautifulSoup/bs4/doc/#>
- Soup Sieve: <https://facelessuser.github.io/soupsieve/>
- Scrapy | A Fast and Powerful Scraping and Web Crawling: <https://scrapy.org/>
- lxml: <https://lxml.de/> + <https://lxml.de/parsing.html>

Полезные ссылки для погружения в материалы учебного модуля:

- W3School XML & XPath: https://www.w3schools.com/xml/xml_xpath.asp
- XPath functions: <https://developer.mozilla.org/en-US/docs/Web/XPath/Functions>

Рекомендации по разработке:

- следите за качеством кода и проверяйте “глупые” ошибки с помощью pylint, следите за поддерживаемостью и читаемостью кода;
- держите уровень покрытия кода тестами на уровне 80+%, следуйте TDD (сначала тесты, потом реализация);
- отделяйте фазу рефакторинга от фазы добавления новой функциональности.
 - фиксируем функциональность, все тесты зеленые;
 - проводим рефакторинг;
 - по окончании фазы рефакторинга снова все тесты зеленые;
- следите за скоростью выполнения unit-test'ов, несколько секунд – это хорошо, в противном случае нужно уменьшать размер тестируемых датасетов или разделять тесты на фазы (см. обсуждение про mark.slow);

3. Критерии оценивания

Балл за задачу складывается из:

- **40%** - реализация функционала по анализу предложений конкурентов
- **20%** - правильное разделение на юнит и интеграционные тесты
 - каждый тест относится либо к юнит (`mark.slow`) или интеграционным (`mark.integration_test`)
 - для каждого окружения (`slow / integration`) существует как минимум один тест
 - не существует тестов, которые помечены как "skipped" одновременно в обоих окружениях
 - все юнит-тесты проходят
- **20%** - качество покрытия юнит-тестами
 - оценка качества проводится автоматически вызовом `pytest`:
 - `PYTHONPATH=. pytest -v --cov=task_*_web_spy test_*_web_spy.py --skip-integration`
 - уровень покрытия тестами должен быть выше 80%
 - проверяем код Python версии 3.7 с помощью `pytest==6.0.1`
 - точная формула: $20\% \times \min([\text{test_coverage} / 0.8], 1.0)$
- **10%** - корректная работа интеграционных тестов
 - оценка качества проводится автоматически вызовом `pytest`:
 - `PYTHONPATH=. pytest -v --cov=task_*_web_spy test_*_web_spy.py --skip-slow`
 - должен выводиться ожидаемый `assert`
- **10%** - поддерживаемость и читаемость кода
 - в общем случае см. Clean Code и [Google Python Style Guide](#)
 - оценка качества будет проводиться автоматическим вызовом `pylint`:
 - `pylint task_*.py`
 - качество кода должно оцениваться выше 8.0 / 10.0
 - проверяем код Python версии 3.7 с помощью `pylint==2.5.3`
 - точная формула: $10\% \times \min([\text{lint_quality} / 8.0], 1.0)$

Discounts (скидки и другие акции):

- **100%** за плагиат в решениях (всем участникам процесса)
- **100%** за посылку решения после hard deadline
- **30%** за посылку решения в после soft deadline и до hard deadline
- **5%** за каждую посылку после 2й посылки в день (каждый день можно делать до 2х посылок без штрафа)

лучший балл с 1-й попытки: 100%



лучший балл со 2-й попытки: 100%

лучший балл с 3-й попытки: 95%

лучший балл с 4-й попытки: 90%

4. Инструкция по отправке задания

Оформление задания:

- Код задания (Short name): **HW06:Web Spy**
- Выполненное ДЗ запакуйте в архив `PY-MADE-2021-Q4_<Surname>_<Name>_HW#.zip`, пример -- `PY-MADE-2021-Q4_Dral_Alexey_HW06.zip`. (Проверяйте отсутствие пробелов и невидимых символов после копирования имени отсюда.²) Если ваше решение лежит в папке `my_solution_folder`, то для создания архива `hw.zip` на Linux и Mac OS выполните команду³:
 - `zip -r hw.zip my_solution_folder/*`
- На Windows 7/8/10: необходимо выделить все содержимое директории `my_solution_folder/` нажать правую кнопку мыши на одном из выделенных объектов, выбрать в открывшемся меню "Отправить >", затем "Сжатая ZIP-папка". Теперь можно переименовать архив.
- Решение задания должно содержаться в одной папке.
- Перед проверкой убедитесь, что дерево вашего архива выглядит так:
 - `| PY-MADE-2021-Q4_<Surname>_<Name>_HW06.zip`
 - `| ---- task_<Surname>_<Name>_web_spy.py`
 - `| ---- test_<Surname>_<Name>_web_spy.py`
 - `| ---- conftest.py`
 - ~~`| ---- gitlab_features.html*`~~
 - ~~`| ---- gitlab_features_expected.html`~~
 - `| ---- *.html5`
 - При несовпадении дерева вашего архива с представленным деревом, ваше решение не будет возможным автоматически проверить, а значит, и оценить его.
- Для того, чтобы сдать задание, необходимо:
 - Зарегистрироваться и залогиниться в сервисе [Everest](#)
 - Перейти на страницу приложения: [MADE Python Grader](#)
 - Выбрать вкладку Submit Job (если отображается иная).

² Онлайн инструмент для проверки: <https://www.soscisurvey.de/tools/view-chars.php>

³ Флаг `-r` значит, что будет совершен рекурсивный обход по структуре директории

⁴ Будет доступно в окружении в период тестирования, в архив паковать не надо, чтобы не занимать 1 МБ данных каждой посылкой

⁵ Архив с тестовыми данными должен занимать **менее 200 КБ** пространства на жестком диске



- Выбрать в качестве "Task" значение: **HW06:Web Spy**⁶
- Загрузить в качестве "Task solution" файл с решением
- В качестве Access Token указать тот, который был выслан по почте
- **Перед отправкой задания**, оставьте, пожалуйста, отзыв о домашнем задании по ссылке: https://rebrand.ly/pymade2021q4_feedback_hw. Это позволит нам скорректировать учебную нагрузку по следующим заданиям (в зависимости от того, сколько часов уходит на решение ДЗ), а также ответить на интересующие вопросы.

Внимание: если до дедлайна остается меньше суток, и вы знаете (сами проверили или коллеги сообщили), что сдача решений сломана, обязательно сдайте свое решение, прислав нам ссылку на выполненное задание (Job) на почту с темой письма "Short name. ФИО.". Например: "**HW06:Web Spy**. Иванов Иван Иванович." Таким образом, мы сможем увидеть какое решение у вас было до дедлайна и сможем его оценить. Пример ссылки:

- <https://everest.distcomp.org/jobs/67893456230000abc0123def>

Любые вопросы / комментарии / предложения пишите согласно [предложениям](#) на портале.

Всем удачи!

⁶ Сервисный ID: python.web_spy



5. FAQ (часто задаваемые вопросы)

"You are not allowed to run this application", что делать?

Если Вы видите надпись "You are not allowed to run this application" во вкладке Submit Job в Everest, то на данный момент сдача закрыта (нет доступных для сдачи домашних заданий, по техническим причинам или другое). Попробуйте, пожалуйста, еще раз через некоторое время. Если Вы еще ни разу не сдавали, у коллег сдача работает, но Вы видите такое сообщение, сообщите нам об этом.

Grader показывает 0 или < 0 , а отчет (Grading report) не помогает решить проблему

Ситуации:

- система оценивания показывает оценку (Grade) < 0 , а отчет (Grading report) не помогает решить проблему. Пример: в случае неправильно указанного access token система вернет -401 и информацию о том, что его нужно поправить;
- система показывает 0 и в отчете (Grading report) не указано, какие тесты не пройдены. Пример: вы отправили невалидный архив (rar вместо zip), не приложили нужные файлы (или наоборот приложили лишние - временные файлы от Mac OS и т.п.), рекомендуется проверить содержимое архива в консоли:

```
unzip -l your_solution.zip
```

Если Вы столкнулись с какой-то из них присылайте ссылку на выполненное задание (Job) в чат курса. Пример ссылки:

<https://everest.distcomp.org/jobs/67893456230000abc0123def>

Как правильно настроить окружение, чтобы оно совпадало с тестовым окружением?

1. Если еще не установлено, то установите conda
<https://docs.conda.io/projects/conda/en/latest/user-guide/install/>
2. Настройте окружение для разработки на основе README.md курса
<https://github.com/big-data-team/python-course>
3. Скачайте необходимые датасеты для выполнения задания
<https://github.com/big-data-team/python-course#study-datasets>



6. Полезные книги

Рекомендуется взять в домашнюю библиотеку книгу, пропагандирующую написание "чистого кода":

- Clean Code: A Handbook of Agile Software Craftsmanship
- автор: Robert C. Martin
- опубликовано: August 2008

Вероятно есть перевод, но поскольку автор курса читал эту книгу только в оригинале, то за качество перевода не отвечает ;)