Оглавление

Введение

Целью данного практического задания является определение конфигурации сайтов, работающих на базе CMS Joomla!.

Чтобы определить конфигурацию сайтов, необходимо узнать

- Версию ядра сайта, работающего на CMS Joomla!
- Перечень установленных расширений Joomla!
- Версии найденных расширений

А также автоматизировать упомянутые пункты анализа сайта путём их реализации на языке Python, где входным параметром будет адрес сайта.

1. Описание CMS и механизма расширений

Прежде чем решать поставленные задачи, необходимо дать определение тому, что такое CMS и как реализован механизм расширений в Joomla.

Система управления контентом (она же CMS) — это программное обеспечение, которое следит за каждым объектом содержимого веб-сайта. Это похоже на то, как в библиотеке хранят книги и следят за ними. Механизм расширений представлен следующими элементами:

1.1 Компонент

Простейшей аналогией будет представление того, что Joomla! — это операционная система и ее компоненты — это приложения рабочего стола. Создаваемое каким-либо компонентом содержимое обычно показано в центре главной части шаблона.

Большинство компонентов состоят из двух частей:

- административной части
- лицевой части

Front-End - это то, что используется для предоставления веб-страниц веб-сайта пользователю.

Back-end (административная часть) предоставляет интерфейс для настройки и управления различных аспектов данного компонента и доступна к нему через административное приложение.

Во фреймворке Joomla! компонент может быть спроектирован двумя способами: возвращать код HTML запрошенной веб-страницы или в шаблоне 'Model-View-Controller'.

MVC(Model-View-Controller) — это шаблон, позволяющий отделять друг от друга логику программного обеспечения и его представление пользователю. Его главное преимущество в том, что модификация каждого компонента может осуществляться независимо.

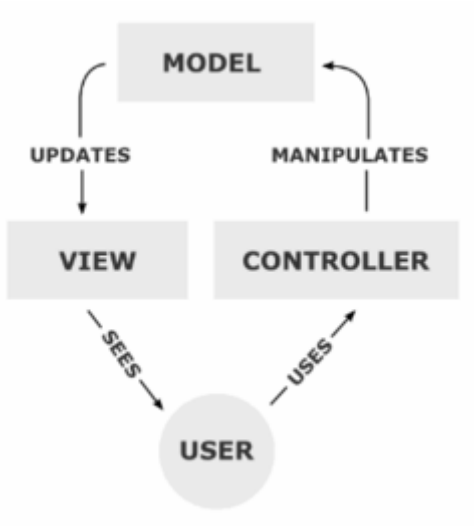

Рис.1, MVC-модель

Модель (Model)

В ней содержатся данные самого приложения. Она реагирует на команды контроллера, изменяя своё состояние.

Представление (View)

Представление — это часть компонента, которая используется для отображения данных из модели в подходящем для взаимодействия с пользователем виде. Для веб-приложения, представление, как правило, будет HTML-страница, которая возвращается пользователю. Представление (вид) извлекает данные из модели и передает данные в шаблон, который заполняется и показывается пользователю.

Контроллер (Controller)

Контроллер (controller) отвечает за операции пользователя. В случае с веб-приложением, операцией пользователя обычно является запрос страницы. Контроллер (controller) определит какой именно запрос был создан конкретным пользователем и соответственно отреагирует на это, оповестив модель об изменении данных.

Таким образом, что модификация каждого компонента может осуществляться независимо.

1.2 Модуль

Модули — это легкие и гибкие расширения, используемые для рендеринга(отрисовки) страниц. В самом просто случае – это просто статическим HTML или текст.

- Эти модули часто представляют собой блоки, расположенные сбоку от компонента на странице.

Хорошо известным примером является модуль входа в систему. А иногда используются для того, чтобы показывать менее важные данные в боковых блоках.

- Модули назначаются для каждого элемента меню

Это означает, что администратор может решить, показывать или скрывать, например, модуль входа в систему в зависимости от того, на какой странице находится пользователь в данный момент.

- Некоторые модули могут быть связаны с компонентами

Например, модуль «последние новости» содержит ссылки на компонент содержимого (com_content) и отображает ссылки на новейшие элементы содержимого. Тем не менее, модули не должны быть жёстко привязаны к компонентам.

- Модули управляются в Joomla! Администратор просмотра с помощью менеджера модулей.

1.3 Плагин

Плагин позволяет вставить в некотором месте выполнения программы (php-скрипта) свой собственный код, который будет изменять или расширять функционал этой программы. при использовании плагина нам не нужно будет вносить изменения в файлы ядра.

Места, куда вставляются такие плагины называются триггерами. Они представляет собой «разъем», в который может подключиться плагин и выполнить свой код. В коде Joomla это выглядит примерно следующим образом:
Код Joomla;
Код Joomla;
Выполнить код плагинов определенного типа, если такие существуют в CMS (это триггер);
Код Joomla;
Код Joomla;

Т. е. в определенном месте выполнения кода Joomla в файлах ядра добавлена строчка, которая ищет все плагины определенного типа, выполняет их последовательно, один за другим. Когда код всех плагинов данного типа будет выполнен, Joomla продолжит выполнять свой основной код.

Каждый плагин Joomla относится к определенному типу. Тип плагина указывает на место его исполнения в CMS и примерный функционал. Кроме того, каждый тип плагинов имеет предопределенный набор триггеров. Назовём некоторые из них:

Тип «Authentication»

Плагины, которые выполняются при авторизации пользователя.

Триггеры:

- onUserAuthenticate

Тип «Content»

Плагины, которые выполняются в процессе работы с контентом (создания, изменения, сохранения, удаления, отображения и др.).

Триггеры:

- onContentPrepare
- onContentAfterTitle
- onContentBeforeDisplay
- onContentAfterDisplay

Тип «System»

Плагины, которые выполняются в ходе генерации любой страницы сайта. Триггеры

- onAfterInitialise
- onAfterRender
- onBeforeRender

Теперь, когда описано то, с чем нужно работать, необходимо назвать алгоритм анализа.

2. Этапы анализа Joomla-сайтов

2.1 Анализ с помощью готовых решений

1. Поиск сайтов, использующих CMS Joomla

   a. Чтобы найти нужный сайт для анализа был использован поисковик Shodan с запросом http. component:” joomla”

   b. Иначе можно использовать google dorks, например, inurl:/index.php?option=com_jwallpapers.

   c. Или же использовать publicwww.com с ключом поиска: "X-Content-Encoded-By: Joomla"

   2. Когда нужный сайт был найден, необходимо выяснить версию сайта. Для этого в репозитории Kali Linux есть JoomScan. Мы не будем использовать его по прямому назначению (поиск уязвимостей), а воспользуемся его побочной функцией – поиск версии этой CMS.

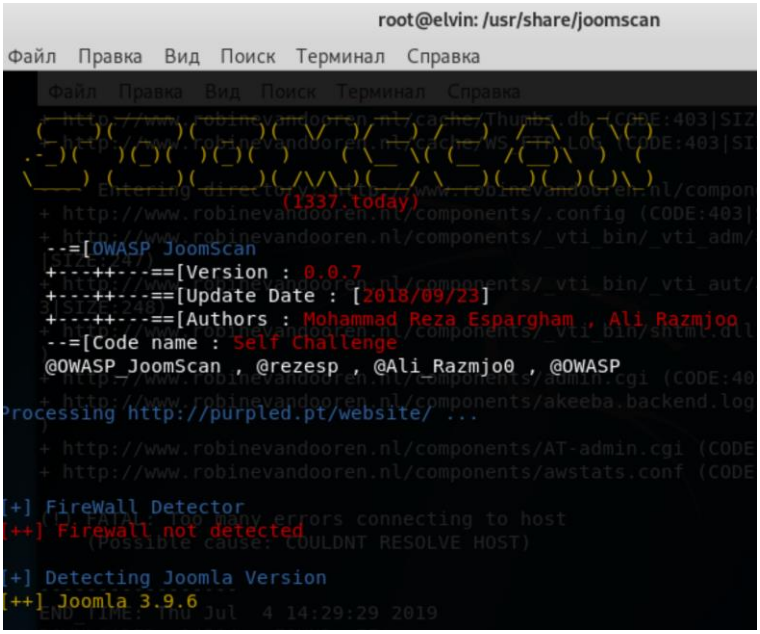

Рис.2, запуск JoomScan

3. Чтобы получить название компонентов и их версии можно, например, заглянуть в соответствующие папки. И чтобы это сделать воспользуемся сканером ZAP.

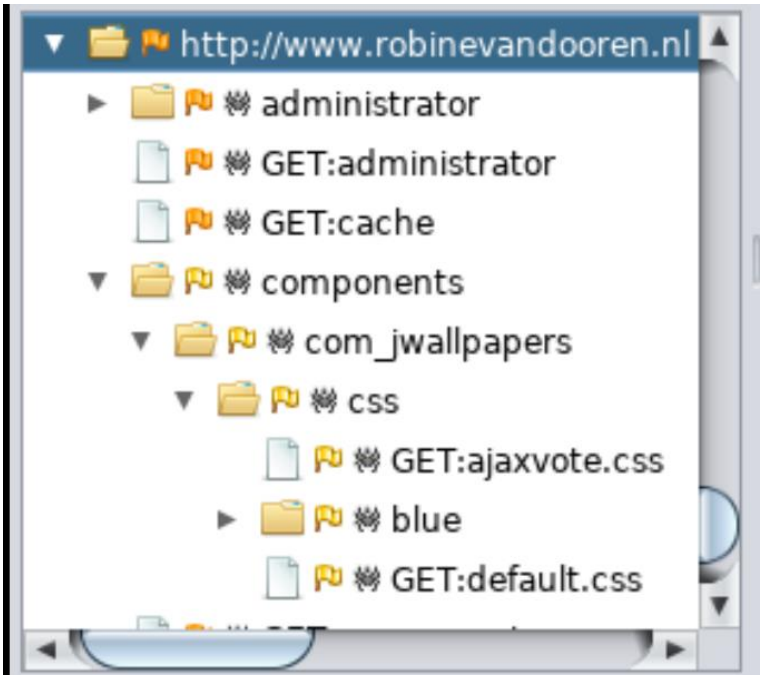

Рис.3, пример запуска ZAP и построения дерева сайта для поиска расширения

2.2 Анализ, произведенный вручную

1. Сайты находятся тем же способом, что в предыдущем варианте анализа
2. В этот раз находить версию сайта будем с помощью такого алгоритма:

- Доступ к сайту по нижеуказанному адресу http://www.[thejoomlawebsite].com/administrator/manifests/files/joomla.xml доступен для Joomla-сайтов версией >= 1.6.0. Соответственно в тегах версии будет указана версия этого сайта
- Для веб-сайтов с версией большей 1.5.0 ,но меньшей 1.5.26 используется такой URL: http://www.[thejoomlawebsite].com/language/en-GB/en-GB.xml
- Для сайтов с версией меньше 1.5.0 используется: http://[thejoomlawebsite].com/modules/custom.xml

3. В силу того, что разработчики редко меняют заводские настройки, пути к директориям расширений остаются стоять по умолчанию, поэтому возможны следующие пути их нахождения:

Компонент:

../administrator/components/com_extension_name/extension_name.xml

Модуль:

../modules/mod_extension_name/mod_extension_name.xml

Plugin:

../plugins/plugin_type/extension_name/extension_name.xml

или же иногда так:

../plugins/plugin_type/extension_name.xml

Сами же названия, версии и описания искомого расширения расположены в тегах <name></name> <version></version> и <description></description> соответственно.

4. Так же просматриваются .css файлы, которые нужно искать в соответствующих названию расширения папках

Компонент:

.. /components/com_extension_name/…

Модуль:

.../modules/mod_extension_name/…

Plugin:

.../plugins/plugin_type/extension_name/…

3.Алгоритм работы программы для нахождения расширений

Пользователь вводит адрес сайта. В первую очередь мы хотим узнать версию сайта. Для этого этот адрес проходит серию вложенных проверок на HTTPError. Вложенные они потому, что, если ссылка на директории версии для 1.6 и выше не подходит, проверяется на 1.5.0 - 1.5.26 или в ином случае, на ниже 1.5. И если всё хорошо, то в сохраненном файле ищется тег с версией, откуда она вытаскивается для пользователя. Затем сохраняем html сайта, в котором будем искать все расширения.

Поиск компонентов происходит в этом html-файле, где посредствам регулярных выражений мы вытаскиваем директории компонентов. Однако эти регулярные выражения должны удовлетворять двум шаблонам поиска:

1. /components/com_...
2. /index.php?option=com_

Первому критерию поиска соответствует такое регулярное выражение: '/com_\S [^\/] +'. Тогда как второму – '=com_\S [^&|^\"]+'.

Однако иногда администраторская директория бывает заблокирована (/administrator/), поэтому надо искать ещё и .css-файлы, где может оказаться информация о компоненте. Поэтому для поиска подобных путей к файлам .css тоже создается регулярное выражение- '/com_\S+.css'.

Поиск модулей работает по тем же самым правилам: ищем директории с характерным видом:

- /modules/mod_ этому соответствует '/mod_\S [^\. +^\/] + [^\W+^\. +] \/+' регулярное выражение

Но мы игнорируем mod_search_searchword, так как это Apache’евскй модуль, а не Jooml’ы.

Затем так же ищем. css, завязанные на модулях с таким регулярным выражением - '/mod_\S+.css'.

Поиск плагинов совершается через регулярные выражения и действия аналогичны вышеупомянутым:

- .../plugins/plugin_type/extension_name – '/plugins/\S+? \/+\S+? \/'

- Для .css путей – '/plugins/\S+.css'

Затем каждая ссылка на xml файл проверяется в цикле на HTTPError (и если ссылка не доступна, то сообщится по какой причине, например, Forbidden) и парсится для нахождения нужной информации. А из каждого .css файла(если нет ошибки) выуживается информация из большого комментария.

Рис.4, блок-схема алгоритма

4.Пример запуска программы, работающей по описанному алгоритму

Чтобы продемонстрировать работоспособность программы, выберем сайт и подадим на вход его URL. Пусть, например, это будет сайт http://barevhayer.am

Программа запрашивает адрес сайта:

```
http://barevhayer.am
http://barevhayer.am/administrator/manifests/files/joomla.xml
The Joomla version is  2.5.11
```

Рис.5, ввод URL и демонстрация того, где ищется версия.

На сайте был найден компонент MuzTvnews. Выводится его версия и его описание:

```
The name of the component is      MuzTvnews

The version is of component      2.1.9

Some description of component      Parent for all PHP based projects
```

Рис.6, версия найденного компонента

Попытка искать версию компонента в .css-файле не увенчалось успехом – в нём не оказалось искомой информации:

```
The info from  http://barevhayer.am/components/com_muztvnews/layouts/strips/themes/cards/cards2.css
No info about component in there
```

Рис.7, поиск компонента по .css

Согласно вышеописанному алгоритму поиска модулей, были найдены следующие модули:

```
The name of the module is        TV Nyut hab Ajax Tertum

The version of module is         1.0.0

Some description of module       MOD_TV_NYUT_HAB_AJAX_TERTUM_XML_DESCRIPTION
```

Рис.8, модуль TV Nyut hab Ajax Tertum и его описание

Во время поиска расширений (в данном случае модулей) в .xml-файлах не всегда удаётся найти подробное описание модуля. В этом случае нашлись только версия и название.

```
The name of the module is    YouBricks Engine

The version of module is    1.0.11

Some description of module
```

Рис.9, поиск модуля YouBrick Engine

Очередной модуль:

```
The name of the module is       Sj News Scrollbar

The version of module is        2.5.0

Some description of module      TPL_YTFRAMEWORK_XML_DESCRIPTION
```

Рис.10, модуль Sj News Scrollbar с описанием

```
The name of the module is       mod_barev_megamenu

The version of module is        3.2.6

Some description of module      MOD_BAREV_MEGAMENU_MODULE_DESC
```

Рис.11, mod_barev_megamenu с версией и описанием

```
The name of the module is    Universal AJAX Live Search

The version of module is    5.4

Some description of module    Universal AJAX Live Search
```

Рис.12, разработчики заменили дефолтный поисковик на более быстрый

В .css-файле модуля не оказалось информации о версии

```
The info from  http://barevhayer.am/modules/mod_youbricks/css/stylesheet1.css
* @package YJ Module Engine

* @author Youjoomla LLC

* @website Youjoomla.com * @copyright  Copyright (c) 2007 - 2011 Youjoomla LLC.
```

Рис.13, нет информации о версии модуля

Удалось найти большое количество ссылок на .css-файлы тех модулей, которые были уже описаны своими .xml:

```
The info from  http://barevhayer.am/modules/mod_tv_nyut_hab_ajax_tertum/assets/css/styles.css
No info about module in there
The info from  http://barevhayer.am/modules/mod_sj_news_scrollbar/assets/css/styles.css
No info about module in there
The info from  http://barevhayer.am/modules/mod_sj_news_scrollbar/assets/css/jquery.mCustomScrollbar.css
No info about module in there
The info from  http://barevhayer.am/modules/mod_barev_megamenu/assets/css/style.css
No info about module in there
The info from  http://barevhayer.am/modules/mod_barev_megamenu/assets/css/animate.css
No info about module in there
The info from  http://barevhayer.am/modules/mod_barev_megamenu/assets/css/barev-font-awesome.css
No info about module in there
```

Рис.14, нет информации в .css-файлах модулей

Даже когда оказывается комментарий с описанием, версия может быть скрыта:

```
The info from  http://barevhayer.am/modules/mod_barev_megamenu/assets/css/style/red.css
 * @version             $Id$

 * @author              JoomlaUX!

 * @package             Joomla.Site

 * @copyright   Copyright (C) 2008 - 2013 by JoomlaUX. All rights reserved.
```

Рис.15, найдена версия, но она скрыта

Плагины ищутся по вышеописанному алгоритму полного пути:

```
http://barevhayer.am/plugins/system/jutabs/jutabs.xml
The name of the plugin is       System - JU Tabs

The version of plugin is        2.3

Some description of plugin
```

Рис.16, поиск плагина по полному пути

В директории короткого пути нет информации

```
http://barevhayer.am/plugins/system/jutabs.xml
We failed to reach a server via  http://barevhayer.am/plugins/system/jutabs.xml
Reason:  Not Found
Continue attempts...
```

Рис.17, поиск плагина по короткому пути

В конце проверяются .css-плагинов

```
The info from  http://barevhayer.am/plugins/system/jutabs/tabs/jutabs.css
No info about plugin in there
The info from  http://barevhayer.am/plugins/system/jutabs/tabs/themes/style16/tab.css
No info about plugin in there
The info from  http://barevhayer.am/plugins/system/jutabs/tabs/themes/style12/tab.css
No info about plugin in there
```

Рис.18, проверка на наличие версий в .css

Результаты работы

Мы провели анализ сайтов, работающих на CMS Joomla. Дали описание архитектуры CMS Joomla. Описали работу этой CMS с расширениями. Проведенный анализ в два этапа дал представления о web-технологиях, используемых разработчиками. Первый этап заключался в использовании автоматизированных средствах анализа, что дало представление о строении сайта. Второй этап задал алгоритм анализа: поиска версии CMS, поиск .xml и .css файлов, их парсинг для извлечения нужной информации. Получившийся алгоритм лёг в основу написанной программы на Python. Разработанный алгоритм работает быстрее аналогов и может работать как с http, так и https соединениями. Для его работы не требуется никаких словарей, так как работает не по принципу брутфорса. Ближайший аналог по функционалу – Joomscan, не нацелен на определение версии расширений, он лишь находит их вхождения. Он так же не ищет .css-файлы, в которых может лежать полезная информация о том или ином расширении.

В таблице, расположенной ниже, рассматривается некоторая выборка сайтов, исходя из неё, мы узнаем относительную частоту появления расширений в том или ином виде:

Таблица 1, выборка сайтов для статистики

Сайт 1	barevhayer.am
Версия Joomla! - 2.5.11 К компоненту MuzTvnews есть доступ через .xml, .css нет. К модулям YouBricks Engine, TV Nyut hab Ajax Tertum, Sj News Scrollbar, mod_barev_megamenu, Universal AJAX Live Search есть доступ через .xml, .css есть, но они не описаны в них. К плагину jutabs по длинному пути есть доступ через .xml, .css нет	

Сайт 2	ashorooq.net
Версия Joomla! - 1.5.15 К компонентам jcomments, comment нет доступа => через .xml не узнать, .css есть, но они не описаны в них. К остальным компонентам нет доступа=> через .xml не узнать, .css отсутствует К модулям Ytc Content Simple Tabs, Lof CoolFlashNews Module, ccNewsletter Module, Hijri Date, hezaakhbar есть доступ через .xml, .css есть, но они не описаны в них. К плагину Content - AllVideos Reloaded по короткому пути есть доступ через .xml, .css нет	
Сайт 3	www.opel-club.ru
Версия Joomla! - 1.5.15 Компонентов нет К модулям News Pro GK1, DJ Image Slider есть доступ через .xml, .css есть, они описаны в них. К плагину plg_content_mavikthumbnails по короткому пути есть доступ через .xml, .css нет	
Сайт 4	film-uzhasov.ru
Версия Joomla! - 1.5.15 Компонентов нет К модулю News Pro GK4 есть доступ через .xml, .css есть, он описаны в них. Плагинов нет	
Сайт 5	wcbiathlon.ru
Версия Joomla! - 1.5.15 К компоненту gk2_photoslide есть доступ через .xml, .css нет.	

К модулю Nice Ajax Poll есть доступ через .xml, .css нет. Плагинов нет	
Сайт 6	ciit.zp.ua
Версия Joomla! - 2.5.8 К компонентам COM_K2, COM_GCALENDAR, com_content, есть доступ через .xml, .css есть, но они не описаны в них. К модулям JComments Latest, Lof K2SlideShow Module, .css есть, но они не описаны в них. К плагинам JA T3 Framework и System - JA Tabs по короткому и длинному пути соответственно есть доступ через .xml, .css есть, но они не описаны в них.	
Сайт 7	adibudi.com
Версия Joomla! – 0...1 Компонентов нет К модулям News Pro GK4, YT Megamenu, .css есть, они описаны в них. Плагинов нет	
Сайт 8	www.palabos.org
Версия Joomla! – 2.5.14 Компонентов нет К модулям AiDaNews 2, Pro Image Flow есть доступ через .xml, .css есть, но они не описаны в них. К плагину PLG_CONTENT_HIGHSLIDE по длинному пути есть доступ через .xml, .css есть, он описан в них.	

Сайт 9	karn.tv
Версия Joomla! – 1.5.15 Компонентов нет К модулям EasyBlog, Content Page, Nastia News Design, Cassrina Hover Image есть доступ через .xml, .css есть, они описаны в них. К плагинам Multithumb, System – Mootools по короткому пути есть доступ через .xml, .css есть, но он не описан в них.	
Сайт 10	hvatit-bolet.ru
Версия Joomla! – 1.5.15 К компонентам JComments, Comments, есть доступ через .xml, .css есть, они описаны в них К модулям News Pro GK4, VTEM Slides есть доступ через .xml, .css есть, но они не описаны в них. К плагину Mootools по короткому пути есть доступ через .xml, .css есть, но он не описан в них.	

Из 10 рассмотренных сайтов 5 из них не имеют компонентов, то есть в 50% случаев у сайта может не оказаться компонентов, 3 не имеют плагинов. Это означает, что с вероятностью в 30% у выбранного сайта не будет плагинов. Такое происходит по решению администратора сайта. На двух сайтах(то есть в 20% случаев) мы получили информацию о компоненте через .xml, но не через .css, потому что такого файла нет или не удалось его найти, это произошло по причине отсутствия прямой ссылки на главной странице сайта. Два сайта (ещё 20% случаев), где возможно получить информацию о компоненте через .xml и существует .css-файл, но описания в последнем нет. Один сайт (такой сайт появляется с вероятностью в 10%) с описанием компонентов в обоих файлах. У одного сайта (такие встречаются с вероятностью в 10 %) не получить доступ к компонентам, но они описаны .css файлами, однако в них ничего нет. Четыре

сайта (такие встречаются с вероятностью в 40%) с описанием модулей в .xml и .css. Пять сайтов (такие встречаются с вероятностью в 50%) с описанием модулей только в .xml файле при наличии .css-файла, но отсутствия информации в нём. Один сайт (такие встречаются с вероятностью в 10%) с модулями, описанными только в .xml-файле и отсутствующим .css-файлом. На одном сайте (такие встречаются с вероятностью в 10%) есть доступ к плагинам по длинному пути, но нет. css-файла. На двух сайтах (такие встречаются с вероятностью в 20%) по длинному пути находится плагины, но в одном из них нет описания в .css, а в другом есть. На двух сайтах (такие встречаются с вероятностью в 20%) на коротком пути лежат плагины, но нет .css-файлов. На трех сайтах (такие встречаются с вероятностью в 30%) с короткими путями плагинов. css есть, но нет описания этих плагинов.

Приложение

```
import re
from bs4 import BeautifulSoup
import requests
from requests.exceptions import ConnectionError
import urllib.request
import shutil
from urllib.request import Request, urlopen
from urllib.error import  URLError
comp=[]
comp1=[]# to get comps in proper way without duplicates
comp1_css=[]
mod=[]
mod1=[]# get separating css and other stuff
mod_css=[]
plug=[]
plug1=[]
plug_css=[]
def get_num(x):
    return (''.join(ele for ele in x if ele.isdigit() or ele == '.'))
def version_joomla_finder(li):
    mystr=li;
    j_ver_160=('/administrator/manifests/files/joomla.xml')
    j_ver_150_26=('/language/en-GB/en-GB.xml')
    j_ver_1=('/modules/custom.xml')
#For Joomla websites >= 1.6.0

    print(mystr+j_ver_160)# get the url to access the xml
   #-----------------------------------------------------------------------
```

```
            req = Request(mystr+j_ver_160)
            try:
             response = urlopen(req)
            except URLError as e:
             if hasattr(e, 'reason'):
                #print('We failed to reach a server.')
                #print('Reason: ', e.reason)
                #For Joomla websites >= 1.5.0 and <= 1.5.26
                req1 = Request(mystr+j_ver_150_26)
                try:
                 response = urlopen(req1)
                except URLError as e:
                 if hasattr(e, 'reason'):
                    req2 = Request(mystr+j_ver_1)
                    try:
                     response = urlopen(req2)
                    except URLError as e:
                     if hasattr(e,'reason'):
                        print(e.reason)
                    else:
                       with urllib.request.urlopen(mystr+j_ver_150_26) as response, open('file_name.txt', 'wb') as out_file:
                            shutil.copyfileobj(response, out_file)
                else:
                  with urllib.request.urlopen(mystr+j_ver_150_26) as response, open('file_name.txt', 'wb') as out_file:
                    shutil.copyfileobj(response, out_file)
            else:
            # everything is fine
            # Download the file from `url` and save it locally under `file_name`:
```

```
            with urllib.request.urlopen(mystr+j_ver_160) as response, open('file_name.txt', 'wb') as out_file:
                shutil.copyfileobj(response, out_file)
        #-------------------------------------------------------------------------
        out_file.close()
        out_file=open('file_name.txt')
        ver = '<version>'

        for i in out_file:
            #line = out_file.readline()
           # print (i)
            if ver in i:
                break
        out_file.close()

        #print(get_num(i))
        return get_num(i)


    def saveinhtml(li):
        with urllib.request.urlopen(li) as response, open('the_site.txt', 'wb') as out_file:
            shutil.copyfileobj(response, out_file)
            out_file.close()

    def strcmp(res1,res2):
        if res1 in res2:
            return True
        else:
```

```
        return False
# Just trying to find every comp on a site
def look_for_comp(li):
    out_file=open('the_site.txt','r',encoding='Latin-1')
    mytext=out_file.read()
    #result = re.findall(r'/com_\w+\D', mytext)
    result = re.findall(r'/com_\S[^\/]+', mytext)
    #print (result)
    for x in result:
        if not x in comp:
            comp.append(x) # now we got comp directories in comp
    #additional search for componetnts
    result2 = re.findall(r'=com_\S[^&|^\"]+', mytext)
    for x in result2:
        if not x in comp:
            comp.append(x) # now we got  ALL comp directories in comp
    for i in range(len(comp)):
        if '=' in comp[i]:
            comp[i]=comp[i][1:len(comp[i])]
    #print(comp)
    for i in range(len(comp)):
        if '/' in comp[i]:
            comp[i]=comp[i][1:len(comp[i])]
    for x in comp:
        if not x in comp1:
            comp1.append(x)

    #print(comp)
    # let us find .css now
    result3 = re.findall(r'/com_\S+.css', mytext)
```

```
    #print(result3)
    for x in result3:
        if not x in comp1_css:
            comp1_css.append(x) #now we got .css directories
    #print(comp)
    print('\n')
    out_file.close()

    print(comp1)
    print(comp1_css)

def look_for_mod(li):
    out_file=open('the_site.txt','r',encoding='Latin-1')
    mytext=out_file.read()
    #result = re.findall(r'/com_\w+\D', mytext)
    result = re.findall(r'/mod_\S[^\.+^\/]+[^\W+^\.+]\/+', mytext)
    #print (result)
    for x in result:
        if not x in mod:
            mod.append(x) # now we got comp directories in mod

    #print(mod)
    # let us find .css now
    result2 = re.findall(r'/mod_\S+.css', mytext)
    #print(result2)
    for x in result2:
        if not x in mod:
            mod.append(x) #now we got .css directories
    print(mod)
    out_file.close()
```

```
def look_for_plug(li):
    out_file=open('the_site.txt','r',encoding='Latin-1')
    mytext=out_file.read()
    result = re.findall(r'/plugins/\S+?\/+\S+?\/', mytext) # looking for directories for xml
   # print(result)
    for x in result:
        if not x in plug:
            plug.append(x) # now we got comp directories in plug
#    print(plug)
    #All for xml have found
    #now looking for .css
    result2 = re.findall(r'/plugins/\S+.css', mytext)
   # print(result2)
    for x in result2:
        if not x in plug:
            plug.append(x) # now we got comp directories in plug
    print(plug)
    out_file.close()



def info_comp(li):

    for i in range(len(comp1)):
        li1=li+'/administrator/components'
        buf=comp1[i]
        # fixing / and /
        buf=buf='/'+buf+'/'
```

```
                buf1=buf[5:len(buf)-1]+'.xml'
                li1=li1+buf+buf1
               # print(li1)
                 # chekning the url we've got by the http response
                req = Request(li1)
                try:
                   response = urlopen(req)
                except URLError as e:
                   if hasattr(e, 'reason'):
                      print('We failed to reach a server via ', li1)
                      print('Reason: ', e.reason)
                      print('Continue attempts...')
                      continue
                   elif hasattr(e, 'code'):
                      print('The server couldn\'t fulfill the request via ', li1)
                      print('Error code: ', e.code)
                      print('Continue attempts...')
                      continue
                else:
                   with urllib.request.urlopen(li1) as response, open('the_xml.txt', 'wb')
as out_file:
                         shutil.copyfileobj(response, out_file)
                out_file.close()
                out_file=open('the_xml.txt','r',encoding='Latin-1')
                # searching for name , version and description
                name='<name>'
                description='<description>'
                version='<version>'
                for x in out_file:
                   if name in x:
```

```
            x=x.replace("<name>","")
            x=x.replace("</name>","")
            print('The name of the component is ',x)
        if description in x:
            x=x.replace("<description>","")
            x=x.replace("</description>","")
            print('Some description of component ',x)
        if version in x:
            x=x.replace('<version>',"")
            x=x.replace('</version>',"")
            print('The version is of component ',x)


def info_comp_css(li):
    for i in range(len(comp1_css)):
        li1=li+'/components'
        li1=li1+comp1_css[i]
        req = Request(li1)
        try:
            response = urlopen(req)
        except URLError as e:
            if hasattr(e, 'reason'):
                print('We failed to reach a server using css via',li1)
                print('Reason: ', e.reason)
                continue
            elif hasattr(e, 'code'):
                print('The server couldn\'t fulfill the request via',li1)
                print('Error code: ', e.code)
                continue
        else:
```

```
                with urllib.request.urlopen(li1) as response, open('the_css.txt', 'wb') as
out_file:
                    shutil.copyfileobj(response, out_file)
            out_file.close()
            out_file=open('the_css.txt','r',encoding='Latin-1')
            print('The info from ',li1)
            author='* @author'
            package='* @package'
            version='* @version'
            copyright1='* @copyright'
            mytext=out_file.read()
            result = re.findall(r'@author', mytext)
            if not result:
               print('No info about component in there')
            else:
               out_file.seek(0)
               for j in out_file:
                  if author   in j:
                     print(j)
                  if package in j:
                     print(j)
                  if version in j:
                     print(j)
                  if copyright1 in j:
                     print(j)

        def info_mod(li):
           for i in range(len(mod1)):
              li1=li+'/modules'+mod1[i]
              buf=mod1[i][1:len(mod1[i])-1]+'.xml'
```

```
            li1=li1+buf
            #print(li1)
            # chekning the url we've got by the http response
            req = Request(li1)
            try:
               response = urlopen(req)
            except URLError as e:
               if hasattr(e, 'reason'):
                  print('We failed to reach a server via ', li1)
                  print('Reason: ', e.reason)
                  print('Continue attempts...')
                  continue
               elif hasattr(e, 'code'):
                  print('The server couldn\'t fulfill the request via ', li1)
                  print('Error code: ', e.code)
                  print('Continue attempts...')
                  continue
            else:
               with urllib.request.urlopen(li1) as response, open('the_xml1.txt', 'wb')
as out_file:
                     shutil.copyfileobj(response, out_file)
            out_file.close()
            out_file=open('the_xml1.txt','r',encoding='Latin-1')
            # searching for name , version and description
            name='<name>'
            description='<description>'
            version='<version>'
            for x in out_file:
               if name in x:
                  x=x.replace("<name>","")
```

```
            x=x.replace("</name>","")
            print('The name of the module is ',x)
        if description in x:
            x=x.replace("<description>","")
            x=x.replace("</description>","")
            print('Some description of module ',x)
        if version in x:
            x=x.replace('<version>',"")
            x=x.replace('</version>',"")
            print('The version of module is ',x)
#       out_file.close()


def info_mod_css(li):
    for i in range(len(mod_css)):
        li1=li+'/modules'
        li1=li1+mod_css[i]
        req = Request(li1)
        try:
            response = urlopen(req)
        except URLError as e:
            if hasattr(e, 'reason'):
                print('We failed to reach a server using css via',li1)
                print('Reason: ', e.reason)
                continue
            elif hasattr(e, 'code'):
                print('The server couldn\'t fulfill the request via',li1)
                print('Error code: ', e.code)
                continue
        else:
```

```
                with urllib.request.urlopen(li1) as response, open('the_css1.txt', 'wb')
as out_file:
                        shutil.copyfileobj(response, out_file)
            out_file.close()
            out_file=open('the_css1.txt','r',encoding='Latin-1')
            print('The info from ',li1)
            author='* @author'
            package='* @package'
            version='* @version'
            copyright1='* @copyright'
            mytext=out_file.read()
            result = re.findall(r'/\*\*', mytext)
            if not result:
               print('No info about module in there')
            else:
               out_file.seek(0)
               for j in out_file:
                  if author   in j:
                     print(j)
                  if package in j:
                     print(j)
                  if version in j:
                     print(j)
                  if copyright1 in j:
                     print(j)

        def cut_back(buf):
           buf=buf[1:len(buf)-1]
           r=buf[::-1]
           return(r[0:r.find('/')])
```

```
        def info_plug_long(li):
            for i in range(len(plug1)):
             li1=li+plug1[i]
             buf=plug1[i]
             buf1=cut_back(buf)
             buf1=buf1[::-1]
             li1=li1+buf1+'.xml'
             print(li1)
             # chekning the url we've got by the http response
             req = Request(li1)
             try:
                response = urlopen(req)
             except URLError as e:
                if hasattr(e, 'reason'):
                    print('We failed to reach a server via ', li1)
                    print('Reason: ', e.reason)
                    print('Continue attempts...')
                    continue
                elif hasattr(e, 'code'):
                    print('The server couldn\'t fulfill the request via ', li1)
                    print('Error code: ', e.code)
                    print('Continue attempts...')
                    continue
             else:
                with urllib.request.urlopen(li1) as response, open('the_xml2.txt', 'wb')
as out_file:
                        shutil.copyfileobj(response, out_file)
             out_file.close()
             out_file=open('the_xml2.txt','r',encoding='Latin-1')
```

```
        # searching for name , version and description
        name='<name>'
        description='<description>'
        version='<version>'
        for x in out_file:
            if name in x:
                x=x.replace("<name>","")
                x=x.replace("</name>","")
                print('The name of the plugin is ',x)
            if description in x:
                x=x.replace("<description>","")
                x=x.replace("</description>","")
                print('Some description of plugin ',x)
            if version in x:
                x=x.replace('<version>',"")
                x=x.replace('</version>',"")
                print('The version of plugin is ',x)


def info_plug_short(li):
    for i in range(len(plug1)):
        plug1[i]=plug1[i][:-1]
        li1=li+plug1[i]+'.xml'
        print(li1)
        # chekning the url we've got by the http response
        req = Request(li1)
        try:
            response = urlopen(req)
        except URLError as e:
            if hasattr(e, 'reason'):
                print('We failed to reach a server via ', li1)
```

```
                print('Reason: ', e.reason)
                print('Continue attempts...')
                continue
            elif hasattr(e, 'code'):
                print('The server couldn\'t fulfill the request via ', li1)
                print('Error code: ', e.code)
                print('Continue attempts...')
                continue
        else:
            with urllib.request.urlopen(li1) as response, open('the_xml3.txt', 'wb')
as out_file:
                shutil.copyfileobj(response, out_file)
        out_file.close()
        out_file=open('the_xml3.txt','r',encoding='Latin-1')
        # searching for name , version and description
        name='<name>'
        description='<description>'
        version='<version>'
        for x in out_file:
            if name in x:
                x=x.replace("<name>","")
                x=x.replace("</name>","")
                print('The name of the plugin is ',x)
            if description in x:
                x=x.replace("<description>","")
                x=x.replace("</description>","")
                print('Some description of plugin ',x)
            if version in x:
                x=x.replace('<version>',"")
                x=x.replace('</version>',"")
```

```
                print('The version of plugin is ',x)


        def info_plug_css(li):

            for i in range(len(plug_css)):
                li1=li+plug_css[i]
                req = Request(li1)
                try:
                    response = urlopen(req)
                except URLError as e:
                    if hasattr(e, 'reason'):
                        print('We failed to reach a server using css via',li1)
                        print('Reason: ', e.reason)
                        continue
                    elif hasattr(e, 'code'):
                        print('The server couldn\'t fulfill the request via',li1)
                        print('Error code: ', e.code)
                        continue
                else:
                    with urllib.request.urlopen(li1) as response, open('the_css2.txt', 'wb')
as out_file:
                            shutil.copyfileobj(response, out_file)
                out_file.close()
                out_file=open('the_css2.txt','r',encoding='Latin-1')
                print('The info from ',li1)
                author='* @author'
                package='* @package'
                version='* @version'
                copyright1='* @copyright'
```

```
        mytext=out_file.read()
        result = re.findall(r'/\*\*', mytext)
        if not result:
            print('No info about plugin in there')
        else:
            out_file.seek(0)
            for j in out_file:
                if author   in j:
                    print(j)
                if package in j:
                    print(j)
                if version in j:
                    print(j)
                if copyright1 in j:
                    print(j)

print("Enter the URL")
li=[]
li=input()
if li[-1]=='/':
    #li.pop()
    li=li[:-1] #say eatshit you slash
#print(li);
#print('You entered'+li)
#r=requests.get('https://github.com')
#print(r)

print('The Joomla version is ',version_joomla_finder(li))
#------------------------------------------------------------------------
#NOW lets see components, modules and plugs
```

```
saveinhtml(li) # save the html of the site

look_for_comp(li)
look_for_mod(li)
look_for_plug(li)
#-------------------------------------------------------------------------
# when we got all comps, mods and plugs , let head us to find info about them


# let's start with comps
if not comp:
    print('No components on the site')
info_comp(li) # in this func we will fix the url and get the info
info_comp_css(li)
if not mod:
    print('No modules on the site')
# separating /mod xml  and .css mod
for i in range(len(mod)):
        if '.css' in mod[i]:
            mod_css.append(mod[i])
        else:
            if not '.js' in mod[i]:
                mod1.append(mod[i])
info_mod(li)
info_mod_css(li)
if not plug:
    print('No plugins on the site')
# now plugins
for i in range(len(plug)):
```

```
    if '.css' in plug[i]:
        plug_css.append(plug[i])
    else:
        if not '.js' in plug[i]:
            plug1.append(plug[i]) #rest
info_plug_long(li)
info_plug_short(li)
info_plug_css(li)
```

Список литературы

1. https://joomla.stackexchange.com/questions/24882/how-to-find-the-version-of-extensions-used-on-a-joomla-website-without-access-to/24883?noredirect=1#comment32356_24883
2. https://wedal.ru/uroki-joomla/plaginy-joomla-podrobnyj-razbor-prostym-yazykom.html
3. https://www.itoctopus.com/how-to-quickly-know-the-version-of-any-joomla-website
4. https://docs.joomla.org/Component
5. https://docs.joomla.org/Module