НАСТОЛЬНОЕ ПРИЛОЖЕНИЕ ДЛЯ ЛЮБИТЕЛЕЙ НОВОСТЕЙ МИРА КИНО

Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь

Шпаков Н.И.

Жвакина А.В. – канд. техн. Наук, доцент

Каждый год выходит много тысяч фильмов, а за новостями из мира кино практически невозможно уследить даже самому искушенному ценителю. Несмотря на то, что сейчас все больше и больше людей имеют доступ к информационным технологиям, интернет соединение может в самый неподходящий момент оказаться недостаточно качественным для посещения любимого сайта.

Данное приложение было создано с расчетом на снижение потребления интернет трафика и также на быстрый и удобный доступ к самым популярным в мире кино-ресурсам, к новостям которых пользователи могут получить доступ сразу же после выбора ресурса. Приложение состоит из двух частей:

1) Серверной части, реализованной на высокоуровневом языке программирования Python версии 3.6.0;
2) Визуальной пользовательской части с использованием инструментов Qt, которая будет использовать уже в дальнейшем данные с сервера.

Работу схемы данной связки можно увидеть на иллюстрации ниже:

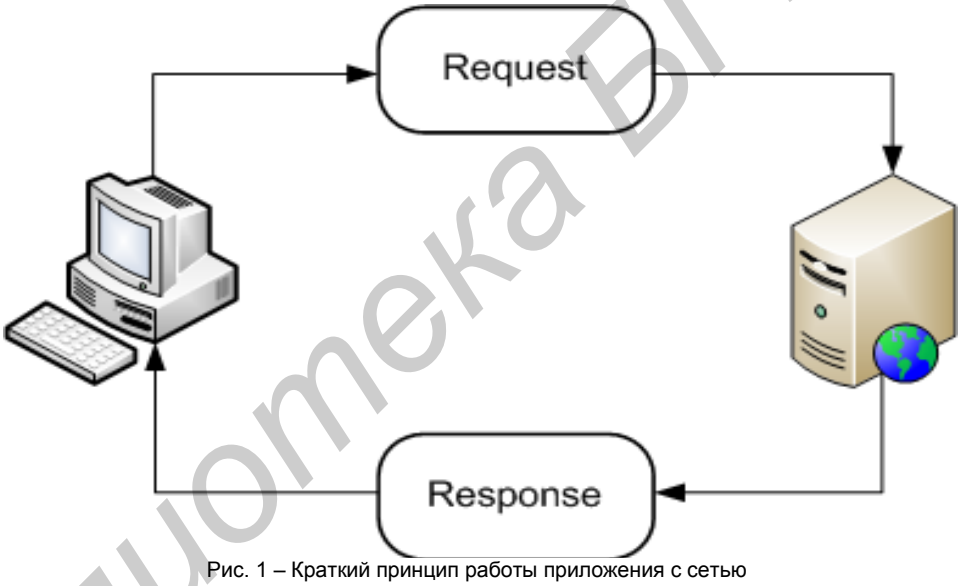

Рис. 1 – Краткий принцип работы приложения с сетью

Сперва клиент посылает сигнал на получение каких-то данных у сервера. В данном случае нам будут возвращаться данные в формате JSON, которые в дальнейшем будут представлены в виде форматированного списка новостей с возможностью их просмотра. Сам же сервер обрабатывает по определенному алгоритму данные из языка текстовой разметки html при помощи таких модулей языка программирования Python, как BeautifulSoup, lxml и requests. Через метод requests сервер получает исходные сырые данные, которые должны быть обработаны с помощью BeautifulSoup и lxml для вынесения нужных нам тегов с данными, а затем форматирование их в JSON формат. Сами данные обрабатываются циклически, позволяя за одну загрузку просмотреть сразу несколько новостных страниц.

После получения JSON данных клиент начинает их обработку. С использованием существующих инструментов Qt представить полученные данные в приемлемый вид не представляет никакого труда для самого приложения.

Среди главных преимуществ перед обычным подходом просмотра и “серфинга” в интернете данное решение несет большую экономию интернет трафика, который может оказаться в дефиците у пользователя, или же просто вне зоны полноценного использования. Поэтому, обрабатывая лишь нужные нам текстовые документы, мы экономим большое количество трафика. Также это приложение позволяет получить доступ к нужным данным в кратчайшие сроки, просто запустив его с рабочего стола пользователя.

Среди минусов можно отметить то, что функционал приложения будет значительно уступать полноценным кино-ресурсам. Также одним из самых важных недостатков системы является то, что и серверная часть, и клиентская постоянно будет нуждаться в доработках чтобы увеличить спектр предоставляемых ресурсов, так как для каждого сайта используется свой подход в его создании. Точно также и приложение нуждается в уникальном подходе к каждому ресурсу для вынесения необходимой нам

информации.

Исследованиеподдержанопроектом CERES. Centers of Excellence for young RESearchers (Reg.no. 544137-TEMPUS-1-2013-SK-JPHES),

Список использованных источников:

1. Документация по языку Python. [Электронный ресурс] – Режим доступа: https://docs.python.org/3/ . − Дата доступа: 03.04.2017.
2. Документация по средствам Qt. [Электронный ресурс] – Режим доступа: http://doc.qt.io/qt-5/json.html. − Дата доступа: 03.04.2017.

АВТОМАТИЗАЦИЯ ТЕСТИРОВАНИЯ С ПРИМЕНЕНИЕМ BDD ПОДХОДА

Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь

Жданко Т.В.

Матвейчук. Н. М. – канд. физ.-мат. наук, доцент

Тестирование является состовляющей частью разработки программного обеспечения. Возрастающее количество исходного кода, а также возрастающее число разработчиков, вызывает все большее количество ошибок в разрабатываемом приложении. Одной из основных функций тестирования является обнаружение этих ошибок как можно раньше. Скорость, с которой можно идентифицировать эти ошибки, полностью зависит от подхода тестирования, количества тестировщиков, их опыта, а также инструментов, используемых в процессе тестирования. Эффективная автоматизация тестирования является одной из наиболее важных составляющих быстрой и непрерывной поставки высококачественного программного обеспечения.

Существуют различные подходы к автоматизации тестирования программного обеспечения. Мы часто слышим такие аббревиатуры, как TDD и BDD. Подход разработки через тестирование (TDD, англ. TestDrivenDevelopment) заключается в написании модульных, интеграционных и функциональных тестов для разрабатываемого приложения, перед написанием самого кода приложения. Т.е. сперва разрабатываются тесты, проверяющие корректность работы еще не реализованного программного кода. Разумеется, этот тест не проходит. Затем разрабатывается код, который выполняет действия, требуемые для успешного прохождения теста.

BDD(англ. BehaviourDrivenDevelopment) подход заключается в описании поведения системы с точки зрения эксперта предметной области, а не программиста. Здесь рассматривается не результат выполнения какого-либо модуля, а та работа, которую он выполняет. Этот принцип рассматривают как продолжение TDD.

Одним из инструментов программной поддержки BDD является Cucumber. Тесты пишутся с помощью сценариев, которые понятны любому пользователю. В сценариях используются следующие элементы BDD синтаксиса:

- Given - условие;
- When - событие;
- Then - результат.

Cucumber интерпретирует тесты на используемом языке программирования и использует Selenium WebDriver для управления тестами в браузере.

На рисунке 1 представлен простой сценарий авторизации пользователя.

```
@run
Feature: Test Login

Scenario Outline: Login Success and Failure
    Given I navigate to the mock application
    When I try to login with '<type>' credentials
    Then I should see that I logged in '<status>'

Examples:
    | type      | status        |
    | valid     | successfully      |
    | invalid   | unsuccessfully    |
```

Рисунок 1 – Листинг тестового сценария

Как мы видим, Cucumber может использовать один и тот же тест многократно, используя различные входные данные, которые представлены в таблице. В файлах со сценариями используются теги (@run), с