Потопнин С.О.
Научный руководитель: к.т.н., доцент А.А.Белов
Муромский институт (филиал) федерального государственного образовательного учреждения высшего образования «Владимирский государственный университет имени Александра Григорьевича и Николая Григорьевича Столетовых»
602264, г. Муром, Владимирская обл., ул. Орловская, 23
e-mail: kaf-eivt@yandex.ru

Использование PHP библиотеки Guzzle для формирования http запросов по стандарту PSR-7

HTTP (HyperText Transfer Protocol) – это синхронный прикладной протокол передачи данных седьмого уровня модели OSI, работающий на основе архитектуры взаимодействия клиент-сервер, относящийся к семейству протоколов TCP/IP.

HTTP протокол получил свою популярность из-за простоты и легкости понятия его для человека, так как использует простую структуру формирования запроса(request) и ответа(response). Любой запрос и ответ содержит в себе: статусную строку, заголовок и тело сообщения.

```
GET /index.php HTTP/1.1
Host: example.com
User-Agent: Mozilla/5.0 (X11; U; Linux i686; ru; rv:1.9b5)
Accept: text/html
Connection: close
```

Рис.1. HTTP запрос

```
HTTP/1.0 200 OK
Server: nginx/0.6.31
Content-Language: ru
Content-Type: text/html; charset=utf-8
Content-Length: 1234
Connection: close

... САМА HTML-СТРАНИЦА ...
```

Рис.2. HTTP ответ

HTTP протокол для своей работы по умолчанию использует TCP-порт 80 или 8080. В своих запросах HTTP использует методы GET, POST, PUT, DELETE, также есть более редко используемые методы HEAD, TRACE, OPTION. На данный момент используется версия протокола HTTP 1.1, так как она уже поддерживает постоянное TCP-соединение клиента и сервера в отличии от версии HTTP 1.0, что очень экономит сетевые ресурсы. Также существует версия протокола HTTP 2, которая предоставляет уже более гибкую возможность формирования запросов и ответов, но данная версия протокола на данный момент используется реже, так как очень распространена реализация web-приложений именно на HTTP 1.1.

Протокол HTTPS не является отдельным протоколом от HTTP, а его расширением, к которому добавляется протокол SSL и для передачи данных используется порт 443.

Любое web-приложение использует протокол HTTP и рано или поздно, по мере его роста, ему приходится начинать преобразовать http запросы и ответы в объекты для их правильной и удобной обработки, повторяя некую похожую структуру уже реализованную в других web-приложениях.

Чтобы избежать повторной реализации одной и той же задачи отличающихся в их архитектурном построении был разработан стандарт PSR-7, который задает общий набор интерфейсов для web-приложений, чтобы последние могли использовать одинаковые

абстракции. Это позволяет писать разработчикам повторно используемый, независимый от web-приложения код.

Request и response, с точки зрения стандарта PSR-7, представляют собой абстракцию поверх механизмов встроенных в тот или иной язык программирования. Например, в языке программирования PHP, они полностью заменяют собой суперглобальные массивы, механизм загрузки файлов и многое другое.

В настоящие время в большинстве web-приложений приходится обращаться к сторонним API. В PHP есть расширение cURL(если оно активировано), которое можно использовать для построения запросов к сторонним сервисам, но данное решение приведет к созданию нового кода, которые последующему программисту будет сложно понять. Для того чтобы не создавать новую реализацию для решения задачи, понятную только для самого программиста, который его писал, лучше использовать распространенное стороннее решение, которым является PHP библиотека Guzzle. Начиная с 6 версии данная библиотека уже поддерживает стандарт PSR-7.

Guzzle – это PHP-HTTP-клиент, которые упрощает и стандартизирует HTTP-запросы и интеграцию с web-сервисами. Он может отправлять как синхронные так и асинхронные запросы используя один и тот же интерфейс. Саму библиотеку лучше устанавливать через менеджер пакетных зависимостей composer.

```php
<?php
 chdir(dirname(__DIR__));

 require_once 'vendor/autoload.php';

 use Guzzle\Http\Client;
 use Guzzle\Http\EntityBody;
 use Guzzle\Http\Message\Request;
 use Guzzle\Http\Message\Response;

 /** @var $client Client */
 $client = new Client("https://qrng.anu.edu.au");
```

Рис.3. Инициализация объекта Client.

Первые две строки отвечают за автозагрузку файлов, сгенерированных composer. Далее, при инициализации объекта Client, мы передаем ему URL хоста к которому хотим подключится и осуществлять запросы без query параметров.

```php
/** @var $request Request */
 $request = $client->get('/API/jsonI.php?length=10&type=uint8');

 /** @var $response Response */
 $response = $request->send();

 /** @var $body EntityBody */
 $body = $response->getBody(true);
```

Рис.4. Отправка запросов.

Затем мы задаём путь (/API/jsonI.php); URL параметры (length=10&type=uint8), и указываем тип запроса (GET), вызовом метода $client->get(). Если нам нужно будет выполнить POST запрос, то необходимо воспользоваться методом post(). Для выполнения запроса, вызываем $request->send() и помещаем результат в переменную $response. Если нам необходимо получить ответ от стороннего сервера, воспользуемся методом $response->getBody(). Для получения результата в виде строки, передаём значение TRUE

Помимо данной реализации построения запроса можно воспользоваться более элегантным решением

```php
/** @var $request Request */
 $request = $client->createRequest();

 $request->setPath('/API/jsonI.php');

 $request->getQuery()
     ->set('length', 10)
     ->set('type', 'uint8');
```

Рис.5. Второй вариант формирования запроса.

Во-первых, создаём пустой запрос, вызвав метод $client->createRequest(). Далее можем воспользоваться методом setPath() или getQuery() для формирования строки запроса и параметров.

Литература

1. Ruseller.com: Guzzle простой HTTP PHP клиент [Электронный ресурс]. – URL: https://ruseller.com/lessons.php?rub=37&id=1955.

2. Prowebmastering.ru: Guzzle HTTP клиент [Электронный ресурс]. – URL: https://prowebmastering.ru/guzzle-php.html

3. Semantica.in: Что такое HTTP [Электронный ресурс]. – URL: https://semantica.in/blog/chto-takoe-http.html

4. Habr.com: Разъяснение HTTP 2 [Электронный ресурс]. – URL: https://habr.com/ru/post/221427/

5. Hexlet.io: Стандарт PSR-7 [Электронный ресурс]. – URL: https://ru.hexlet.io/courses/php-mvc/lessons/psr7/theory_unit

6. Coderlessons.com: От HTTP сообщений до PSR-7 [Электронный ресурс]. – URL: https://coderlessons.com/articles/php/ot-http-soobshchenii-do-psr-7-chto-eto-takoe

7. Habr.com: PSR-7 в примерах [Электронный ресурс]. – URL: https://habr.com/ru/post/250343/