Лабораторная работа 06а

ПвИ

**Задание 00**

1. Разработайте два сервера **06-00http** и **06-00express** с использованием модуля http и пакета express соответственно.
2. Оба сервера должны обрабатывать запросы, описанные в следующих заданиях.

**Задание 01** /connection?set=***set***

1. При GET-запросе **/connection** в окно браузера вывести текущее значение параметра **keepAliveTimeout.**
2. При GET-запросе **/connection?set=*set*** установить новое значение системного параметра **keepAliveTimeout=*set*** и вывести в окно браузера сообщение, что установлено новое значение параметра **keepAliveTimeout=*set***.
3. Продемонстрируйте влияние системного параметра **keepAliveTimeout** на работу приложения**.**

**Задание 02** /headers

1. Отобразите в окне браузера все заголовки запроса и ответа. Объясните назначение каждого заголовка.

**Задание 03**/user/***id***

1. Проанализируйте значениe параметра ***id***.
2. Если ***id*** имеет числовое значение, то выведите из файла users.json информацию о пользователе c соответствующим ***id***.
3. Иначе возвращайте клиенту ошибку (например, неверный параметр, пользователя с таким id нет). Для express-сервера ошибку необходимо обработать в специальном middleware для ошибок.

**Задание 04** middleware

1. Создайте промежуточное ПО (middleware), в котором будет выводиться в консоль информация о каждом запросе (метод, путь, время).
2. Примените промежуточное ПО ко всем маршрутам.

**Задание 05** status-code

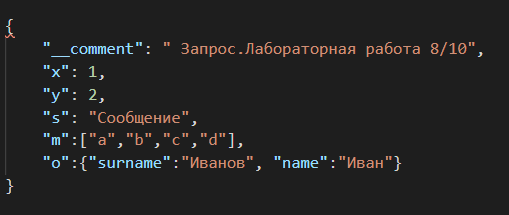
1. При получении запроса с неподдерживаемым url или методом, отличным от GET/POST, сформируйте ответ, имеющий необходимый статус и собственное пояснение к статусу.

**Задание 06** /formparameter

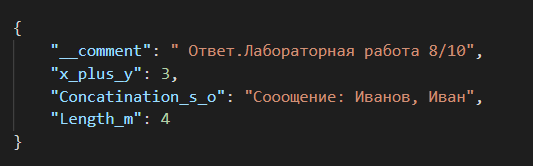
1. Используйте HTML-форму, включающую теги **input** с **type**: **text**, **number, checkbox, radiobutton,** тег **textarea**, а также два тега **input type=submit**, имеющих одно и тоже имя, но разные значения.
2. В окно браузера выведите значения параметров, полученных в запросе.

**Задание 07** /json

1. Принимайте POST-запросы, содержащие данные в json-формате и отправляйте ответы в json-формате.
2. Сообщение в запросе имеет следующую структуру:



1. Сообщение в ответе имеет следующую структуру:



Поле **x+y** ответасодержитсумму полей **x** и **y** запроса.

Поле **Concatination\_s\_o** ответа-конкатенациюполей **s** и свойств объекта **о** запроса.

Поле **Length\_m** ответа-количество элементов в массиве **m** запроса.

1. Проверьте работоспособность приложения с помощью POSTMAN.

**Задание 08** /public/***filename***

1. В ответ на запрос высылается файл с именем ***filename*** из директории **public.**
2. Если файл ***filename*** ненайден, возвращается ответ со статусом 404.
3. Проверьте работоспособность приложения с помощью браузера.

**Задание 09** /upload

1. В ответ на GET-запрос на **/upload** высылается web-форма, позволяющая отправить POST-запрос на **/upload,** присылающийсерверу файл.
2. Сервер сохраняет файл в директории **public.**
3. Проверьте работоспособность приложения с помощью браузера.

**Задание 10** Ответьте на следующие вопросы.

1. Поясните назначение заголовка **Content-Type.**
2. Поясните назначение заголовка **Accept.**
3. Для чего используется значение **multipart/form-data** заголовка **Content-Type.**
4. Как с помощью тега **form**, обеспечить значение **multipart/form-data** заголовка **Content-Type.**
5. Какое значение заголовка **Content-Type** отправляется тегом **form** в запросе по умолчанию.
6. Где и в каком формате передаются параметры в GET-запросе?
7. Где и в каком формате передаются параметры в POST-запросе?
8. Поясните понятие **JSON**?
9. Поясните понятие **XML**?