

Самостоятельная работа №1. Установка языка программирования PHP

Установка и настройка контейнера

Посредством системы виртуализации VirtualBox запущена виртуальная машина Ubuntu Server 20.04.3 LTS (Focal Fossa). Для работы с сервисами, запущенными внутри виртуальной машины, выполнен проброс следующих портов: 22 (ssh); 8080 (http и https из контейнера Docker); 8022 (ssh из контейнера Docker); 3306 (mySQL), 3000 (BrowserSync).

В процессе изучения темы «Task-менеджер GULP + Docker» модуля 4 «Методологии и инструменты современной верстки» был создан образ gulpproject:v1.0 с установленными в нём демоном SSH, веб-сервером Apache 2 и обработчиком задач Gulp. Следующая команда скачивает этот образ из репозитория Docker Hub, пробрасывает из контейнера порты 22, 80 и 3000, создаёт новый контейнер с именем php_apache и запускает его:

```
docker run -it -d -p 8080:80 -p 8022:22 -p 3000:3000 --name php_apache inventivespark/gulpproject:v1.0
```

-it — это флаг позволяющий связать потоки ввода, вывода и ошибок с псевдо-TTY терминалом. Иначе говоря флаг, обозначающий то, что нужно создать сеанс интерактивной работы на подключаемом терминальном устройстве. TTY-терминал — это терминал, который существует только логически, а не физически. Таким образом, один физический терминал может использоваться несколькими процессами.

-d — запускает контейнер в фоновом режиме и выводит на дисплей ID-контейнера.

-p — так называемый «проброс портов». Позволяет обращаться к одинаковым службам, которые по умолчанию работают на 80 (http, https), 22 (ssh), 21 (ftp), 3306 (mySQL) и другие. Параметр строится по следующему принципу:

порт_который_будет_доступен_снаружи :

внутренний_порт_службы_контейнера.

--name позволяет задать контейнеру любое имя для удобного обращения к нему.

Для того, чтобы убедиться, что контейнер запущен и узнать его идентификатор (id), выполняется команда:

```
docker ps
```

Удобнее входить внутрь контейнера, используя его имя, а не идентификатор (id). Для входа в контейнер и работы с ним изнутри выполняется команда, запускающая в контейнере оболочку bash:

```
docker exec -it php_apache /bin/bash
```

-it — это флаг, позволяющий связать потоки ввода, вывода и ошибок с псевдо-TTY терминалом. Иначе говоря флаг, обозначающий то, что нужно создать сеанс интерактивной работы на подключаемом терминальном устройстве.
php_apache — имя контейнера. Вместо него можно задать также его id.
/bin/bash — выбирает к запуску оболочку bash.

При входе в контейнер в приглашении командной строки оболочки bash после символа @ имя компьютера ubuntuserver изменилось на идентификатор контейнера 3ace2e3bf10a.

Для того, чтобы в дальнейшем установка пакетов не прерывалась на диалог "Configuring tzdata", нужно создать символическую ссылку на файл местного часового пояса из доступных в системе:

```
ln -sf /usr/share/zoneinfo/Europe/Moscow /etc/localtime
```

После этого диалог "Configuring tzdata" будет пропускаться. Текущее системное время можно проверить с помощью утилиты date:

```
date
```

Обновление списка репозиторий и обновление установленных пакетов

```
apt-get update && apt-get upgrade -y
```

```
root@ubuntuserver:~# docker ps -a
CONTAINER ID   IMAGE                COMMAND                  CREATED        STATUS        PORTS        NAMES
root@ubuntuserver:~# docker run -it -d -p 8080:80 -p 8022:22 -p 3000:3000 --name
php_apache inventivespark/gulpproject:v1.0
3ace2e3bf10a4b6ccab93aaf1481b241b0053ffee778a75db8a1d370703fda0d
root@ubuntuserver:~# docker ps
CONTAINER ID   IMAGE                COMMAND                  CREATED        STATUS        PORTS        NAMES
3ace2e3bf10a   inventivespark/gulpproject:v1.0   "bash"                  9 seconds ago   Up 8
seconds       0.0.0.0:3000->3000/tcp, :::3000->3000/tcp, 0.0.0.0:8022->22/tcp, :::80
22->22/tcp, 0.0.0.0:8080->80/tcp, :::8080->80/tcp   php_apache
root@ubuntuserver:~# docker exec -it php_apache /bin/bash
root@3ace2e3bf10a:/# ln -sf /usr/share/zoneinfo/Europe/Moscow /etc/localtime
root@3ace2e3bf10a:/# date
Wed Feb 23 21:59:10 UTC 2022
root@3ace2e3bf10a:/# apt-get update && apt-get upgrade -y
```

Установка PHP и настройка Apache 2

Установка интерпретатора языка программирования PHP 7.2:

```
apt-get install php php-curl php-gd php-xml php-soap php-intl php-mysql -y
```

```
root@3ace2e3bf10a:/# apt-get install php php-curl php-gd php-xml php-soap php-intl php-mysql -y
```

Команда, выводящая установленную версию PHP:

```
php -v
```

```
root@3ace2e3bf10a:/# php -v
PHP 7.2.24-0ubuntu0.18.04.10 (cli) (built: Oct 25 2021 17:47:59) ( NTS )
Copyright (c) 1997-2018 The PHP Group
Zend Engine v3.2.0, Copyright (c) 1998-2018 Zend Technologies
    with Zend OPcache v7.2.24-0ubuntu0.18.04.10, Copyright (c) 1999-2018, by Zend Technologies
```

Для связки Apache 2 и PHP нужно установить следующий пакет:

```
apt-get install libapache2-mod-php -y
```

```
root@3ace2e3bf10a:/# apt-get install libapache2-mod-php -y
```

Для того, чтобы Apache 2 обрабатывал код PHP, содержащийся внутри файлов HTML, в текстовом редакторе nano открывается конфигурационный файл Apache 2:

```
nano /etc/apache2/mods-enabled/php7.2.conf
```

```
root@3ace2e3bf10a:/# nano /etc/apache2/mods-enabled/php7.2.conf
```

```
GNU nano 2.9.3 /etc/apache2/mods-enabled/php7.2.conf

<FilesMatch ".+\.ph(ar|p|tml)$">
    SetHandler application/x-httpd-php
</FilesMatch>
<FilesMatch ".+\.phps$">
    SetHandler application/x-httpd-php-source
    # Deny access to raw php sources by default
    # To re-enable it's recommended to enable access to the files
    # only in specific virtual host or directory
    Require all denied
</FilesMatch>
<FilesMatch ".+\.html$">
    SetHandler application/x-httpd-php
</FilesMatch>
# Deny access to files without filename (e.g. '.php')
```

^G Get Help	^O Write Out	^W Where Is	^K Cut Text	^J Justify	^C Cur Pos
^X Exit	^R Read File	^_ Replace	^U Uncut Text	^T To Spell	^_ Go To Line

В конфигурационный файл добавляются следующие строки:

```
<FilesMatch ".+\.html$">
```

```
    SetHandler application/x-httpd-php
```

```
</FilesMatch>
```

После сохранения файла конфигурации выполняется запуск демона SSH и веб-сервера Apache 2

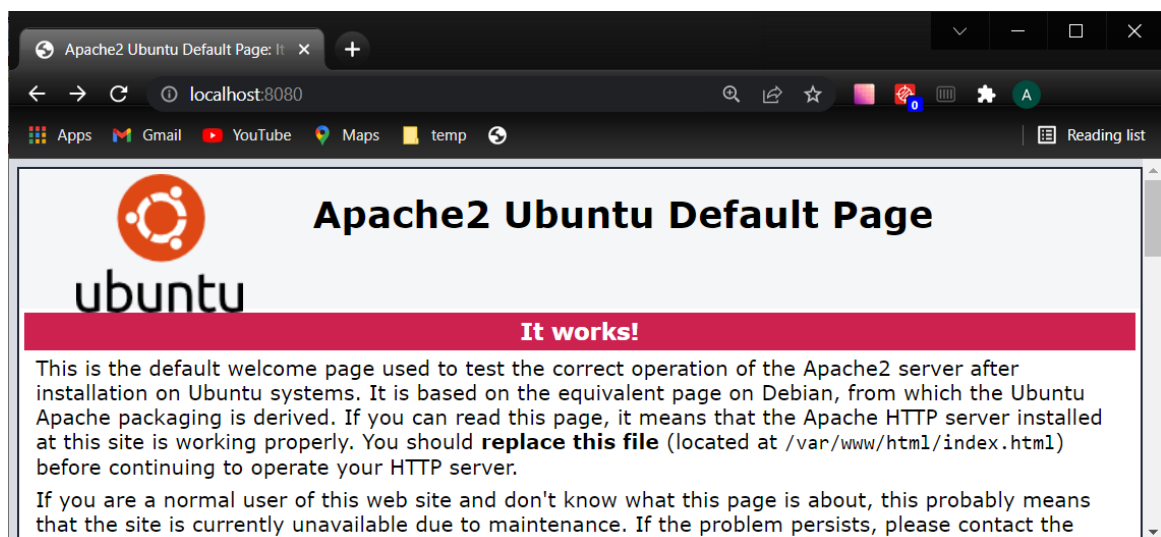
```
service ssh start; service apache2 start
```

Проверка статуса демона SSH и сервера Apache 2:

```
service ssh status; service apache2 status
```

```
root@3ace2e3bf10a:/# service ssh start; service apache2 start
* Starting OpenBSD Secure Shell server sshd [ OK ]
* Starting Apache httpd web server apache2
AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 172.17.0.2. Set the 'ServerName' directive globally to suppress this message
*
root@3ace2e3bf10a:/# service ssh status; service apache2 status
* sshd is running
* apache2 is running
```

Теперь к Docker контейнеру можно присоединиться по протоколу SSH, а приветственная страница веб-сервера Apache 2 доступна из браузера по адресу localhost:8080



Переход в директорию Apache 2:

```
cd /var/www/html/
```

Создание в ней поддиректории phpProject:

```
mkdir phpProject
```

Установка для этой директории полных прав доступа (каждый пользователь может читать, редактировать и запускать на выполнение):

```
chmod -R 777 phpProject
```

```
root@3ace2e3bf10a:/# cd /var/www/html/  
root@3ace2e3bf10a:/var/www/html# mkdir phpProject  
root@3ace2e3bf10a:/var/www/html# chmod -R 777 phpProject
```

На рабочей машине создаётся директория PHP и производится переход в неё:

```
$ mkdir PHP && cd PHP
```

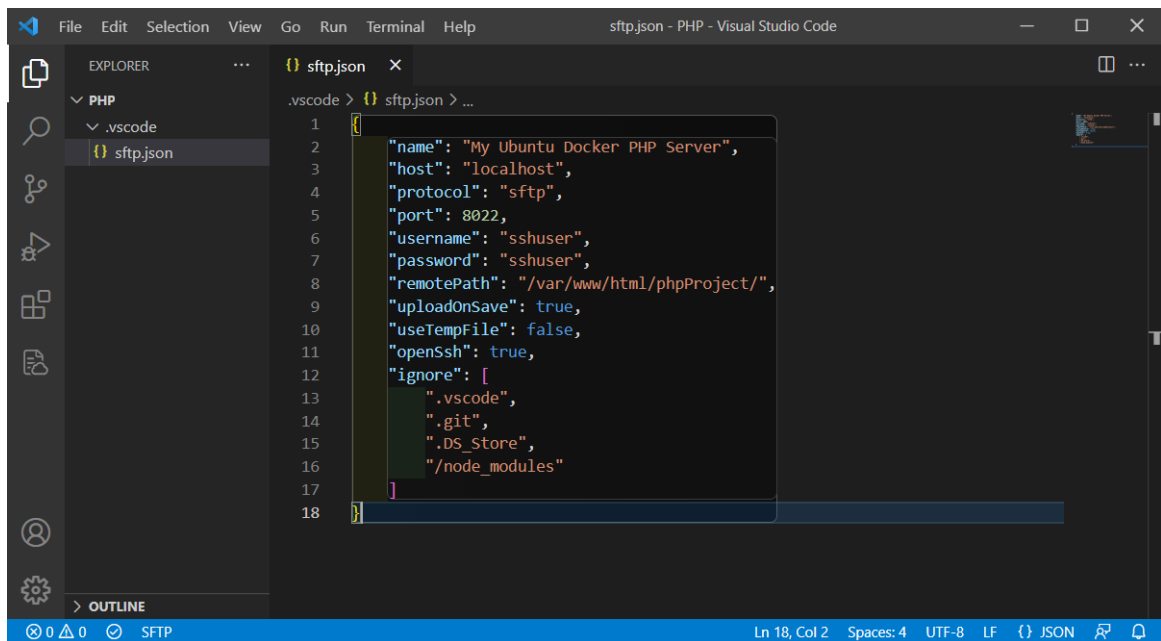
Запуск редактора Visual Studio Code в директории PHP:

```
$ code .
```

```
ARTY@i7 MINGW64 /d  
$ mkdir PHP && cd PHP  
  
ARTY@i7 MINGW64 /d/PHP  
$ code .
```

В редакторе кода VS Code устанавливается расширение SFTP от Natizyskunk. Для того, чтобы синхронизировать с сервером директорию PHP, в редакторе VS Code нажимается комбинация клавиш Ctrl+Shift+P и выполняется команда SFTP:Config. После этого расширение SFTP в директории PHP создаст поддиректорию .vscode с файлом конфигурации. В открывшемся файле sftp.json указываются данные для подключения по SSH к Docker-контейнеру внутри Ubuntu Server:

- **name:** имя сервера;
- **host:** адрес сервера;
- **protocol:** протокол;
- **port:** порт (проброшен в виртуальной машине);
- **username** и **password:** имя пользователя и пароль для доступа к контейнеру;
- **remotePath:** абсолютный путь к директории на сервере (в контейнере), куда будут загружаться файлы;
- **uploadOnSave:** автоматическая загрузка файлов на сервер при их сохранении;
- **ignore:** директории, которые не предназначены для загрузки на сервер и выгрузки с него.

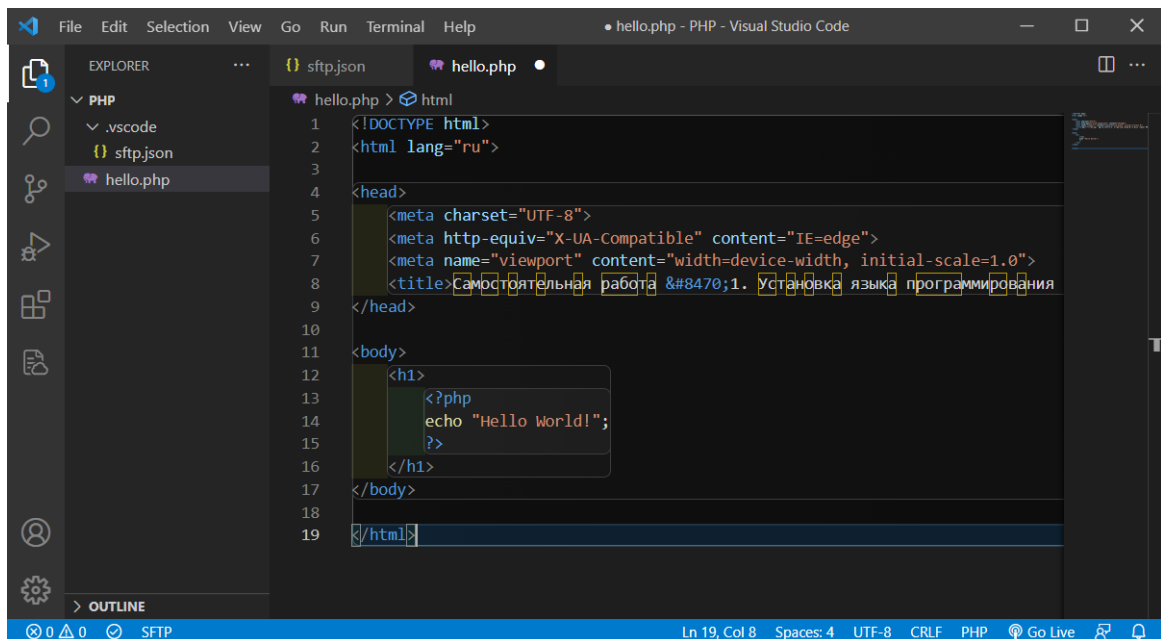


The screenshot shows the Visual Studio Code interface with the Explorer sidebar on the left displaying the file structure: PHP > .vscode > sftp.json. The main editor window is open to sftp.json, showing a JSON configuration for SFTP. The status bar at the bottom indicates 'Ln 18, Col 2', 'Spaces: 4', 'UTF-8', 'LF', and 'JSON'.

```
1 {
2   "name": "My Ubuntu Docker: PHP Server",
3   "host": "localhost",
4   "protocol": "sftp",
5   "port": 8022,
6   "username": "sshuser",
7   "password": "sshuser",
8   "remotePath": "/var/www/html/phpProject/",
9   "uploadOnSave": true,
10  "useTempFile": false,
11  "openSsh": true,
12  "ignore": [
13    ".vscode",
14    ".git",
15    ".DS_Store",
16    "/node_modules"
17  ]
18 }
```

В боковом меню с файлами внутри редактора VS Code открывается локальная директория PHP. Внутри неё производится нажатие правой кнопки мыши, выбор Sync Remote -> Local для того, чтобы синхронизировать удаленную директорию с локальной. Данная операция перезапишет всё, что есть в локальной директории PHP! Нужно соблюдать осторожность, чтобы не перепутать её с командой Sync Local -> Remote — это затрет все файлы в Docker контейнере!

В редакторе Visual Studio Code в директории PHP создается файл hello.php, в который добавляется HTML страница с простейшей программой на языке PHP, затем файл сохраняется и автоматически загружается на сервер (в контейнер):



The screenshot shows the Visual Studio Code interface with the Explorer sidebar on the left displaying the file structure: PHP > .vscode > sftp.json > hello.php. The main editor window is open to hello.php, showing an HTML document with a PHP snippet. The status bar at the bottom indicates 'Ln 19, Col 8', 'Spaces: 4', 'UTF-8', 'CRLF', 'PHP', and 'Go Live'.

```
1 <!DOCTYPE html>
2 <html lang="ru">
3
4 <head>
5   <meta charset="UTF-8">
6   <meta http-equiv="X-UA-Compatible" content="IE=edge">
7   <meta name="viewport" content="width=device-width, initial-scale=1.0">
8   <title>Самостоятельная работа &#8470;1. Установка языка программирования
9 </head>
10
11 <body>
12   <h1>
13     <?php
14       echo "Hello World!";
15     ?>
16   </h1>
17 </body>
18
19 </html>
```

По умолчанию, сервер Apache работает на 80-м порту, но при запуске контейнера порт 80 был проброшен на порт 8080. Страница с загруженным файлом hello.php доступна из браузера по адресу <http://localhost:8080/phpProject/hello.php>

