**Лабораторная работа №1**

**Настройка рабочего окружения**

**Цель:** настроить рабочее окружение для программирования web приложений и сайтов на своем рабочем месте.

**Что такое Visual Studio Code?**

Visual Studio Code или просто VSCode – это бесплатный, популярный и имеющий множество дополнений текстовый редактор, который в первую очередь предназначен для создания и отладки современных веб- и облачных приложений. Разработан он компанией Microsoft и доступен для операционных систем Windows, MacOS и Linux. Распространяется данная программа бесплатно, исходный код её доступен на GitHub.

VSCode поддерживает большое количество языков программирования, включает в себя отладчик, средства для работы с Git, подсветку синтаксиса, технологию автодополнения IntelliSense, инструменты для рефакторинга, терминал и многое другое.

VSCode является достаточно гибким инструментом, расширяемым с помощью плагинов, доступных на [Visual Studio Marketplace](https://marketplace.visualstudio.com/vscode). Открыть панель с расширениями в программе можно через комбинацию клавиш **Ctrl+Shift+X**.

Плагинов для Visual Studio Code очень много, но в рамках обучения рассмотрим только наиболее популярные из них. В VSCode уже встроен такой плагин как Emmet. Если вы не знакомы с ним, то он предназначен для быстрого набора кода. Дополнительно его устанавливать не нужно.

**Установка VS Code**

Для установки VSCode на компьютер необходимо перейти на [этот сайт](https://code.visualstudio.com/). После этого на странице выбрать вашу операционную систему и нажать на значок «Загрузки». После завершения скачивания программы установить её себе на компьютер. Чтобы лучше ориентироваться в программе вы можете установить языковый пакет для русского языка. Для этого откройте панель «Расширения» (Ctrl+Shift+X) и введите в ней текст «Russian Language Pack for Visual Studio Code». После этого выберите его в результатах поиска и установите его.

В Visual Studio Code имеется огромное количество различных плагинов кроме языковых пакетов. Предназначены они для расширения функциональности этого редактора. Все плагины для этого редактора размещены на [Marketplace](https://marketplace.visualstudio.com/vscode).

**Общая настройка VS Code**

VSCode содержит большое количество настроек, с помощью которых вы можете настроить этот редактор под себя. Изменение настроек в VSCode осуществляется в соответствующем окне. Открыть его можно несколькими способами:

* через комбинацию клавиш **Ctrl+**,;
* через пункт меню «Файл -> Настройки -> Параметры» (в версии на английском языке «File -> Preferences -> Settings»);
* нажать на значок шестерёнки и выбрать в открывшемся меню пункт «Параметры» (Settings).

**Список параметров, которые пользователи наиболее часто настраивают:**

* editor.tabsize - число пробелов при табуляции;
* editor.insertSpaces - вставлять ли пробелы при нажатии Tab;
* editor.detectIndentation - нужно ли параметры "#editor.tabsize" и "editor.insertSpaces" определять автоматически при открытии файла на основе его содержимого;
* editor.wordWrap - управляет тем, как следует переносить строки;
* editor.fontSize - размер шрифта в пикселях;
* editor.mouseWheelZoom - нужно ли включать изменение размера шрифта в редакторе при нажатой клавише Ctrl и движении колесика мыши;
* editor.minimap.enabled - включает или отключает отображение мини-карты;
* editor.formatOnSave - выполнять ли автоматическое форматирование файла при его сохранении;
* workbench.startupEditor - управляет тем, что будет отображаться при запуске, если содержимое редактора не было восстановлено из предыдущего сеанса;
* files.insertFinalNewline - если этот параметр включен, то при сохранении файла в его конец вставляется пустая строка;
* files.trimFinalNewlines - если этот параметр активен, то при сохранении файла будут удалены все пустые строки, идущие за последней в конце файла;
* files.trimTrailingWhitespace - если этот параметр включен, то при сохранении файла будут удалены все пробельные символы на концах строк;
* files.autoSave - для включения автосохранения файлов;
* terminal.integrated.cwd - позволяет задать путь явного запуска, по которому будет запущен терминал;
* telemetry.enableTelemetry - включает или отключает отправку сведений об использовании и ошибках в веб-службу Майкрософт;
* telemetry.enableCrashReporter - разрешает отправку отчетов о сбоях в веб-службу Майкрософт;

Изменять настройки можно как глобально, так и конкретно для текущего проекта. Изменение глобальных настроек осуществляется в окне «Параметры» на вкладке «Пользователь». Эти настройки сохраняются в файл «settings.json». Открыть его можно, нажав на значок «Открыть параметры (JSON)».

**Пример файла «settings.json»:**

{

"editor.wordWrap": "wordWrapColumn",

"editor.wordWrapColumn": 120,

"editor.minimap.enabled": false,

"editor.formatOnSave": true,

"editor.tabSize": 2,

"editor.detectIndentation": false,

"editor.fontSize": 17,

"editor.mouseWheelZoom": true,

"workbench.startupEditor": "none",

"files.insertFinalNewline": true,

"files.trimFinalNewlines": true,

"files.trimTrailingWhitespace": true,

"telemetry.enableTelemetry": false,

"telemetry.enableCrashReporter": false,

"files.autoSave": "afterDelay",

"files.autoSaveDelay": 10000

}

Кстати, изменять настройки также можно просто посредством редактирования этого файла.

Сохранение настроек для рабочей директории выполняется в специальный файл «settings.json», который будет добавлен в папку «.vscode». Настройка параметров для рабочей директории (проекта) можно также выполнить просто посредством редактирования этого файла.

**Настройка VS Code для HTML и CSS**

Visual Studio Code обеспечивает базовую поддержку при написании HTML и CSS из коробки. Имеется подсветка синтаксиса, умные дополнения с IntelliSense и настраиваемое форматирование. VSCode также имеет отличную поддержку Emmet. Зачем нужен Emmet? Он позволяет очень быстро писать код. Например, Emmet аббревиатура ul>li\*3>span.item-$ после нажатии клавиши Tab создаст следующий код:

<ul>

<li><span class="item-1"></span></li>

<li><span class="item-2"></span></li>

<li><span class="item-3"></span></li>

</ul>

В CSS аббревиатура Emmet как dn создаст код display: none.

VSCode имеет встроенные средства для форматирования кода. Настроить параметры форматирования можно в настройках. Находятся они в разделах «Расширения -> HTML» и «Расширения -> CSS».

Комбинация клавиш для выполнения форматирования в VSCode: **Shift+Alt+F**. Функциональность VS Code при работе с HTML и CSS можно улучшить с помощью расширений.

**Вот перечень некоторых из них:**

**Material Icon**

* **Auto Rename Tag** – автоматически изменяет имя закрывающего тега при переименовывании открывающегося;
* **Auto Close Tag** – автоматически добавляет закрывающий HTML/XML тег при вводе закрывающей скобки открывающегося тега (кроме HTML, это дополнение добавляет эту возможность в JavaScript и многие другие языки);
* **HTMLHint** – плагин для статического анализа HTML кода;
* **HTML CSS Support** - поддержка CSS для документов HTML;
* **IntelliSense for CSS class names in** **HTML** - плагин для предложения вариантов завершения имени CSS класса в HTML на основе определений, найденных им в вашем рабочем пространстве;
* **Autoprefixer -** для автоматического добавления CSS свойств с префиксами;
* **CSS Peek** - позволяет посмотреть свойства, прикреплённые к классу или идентификатору без переключения на CSS файл, в котором они описаны;
* **Prettier - Code formatter** - для форматирования кода (HTML, CSS, JavaScript и др.);
* **Project Manager** – для удобного управления вашими проектами, независимо от того, где они расположены;
* **LiveServer** – локальный сервер с возможностью автоматического обновления страниц в браузере при их изменении (запустить сервер можно посредством нажатия на иконку «Go Live» в строке состояния или выбора пункта «Open with Live Server» в контекстном меню области редактора);
* **Settings Sync** – плагин для синхронизации настроек VS Code (хранение настроек осуществляется на GitHub);
* **GitLens** – расширяет возможности Git встроенного в этот редактор (он позволяет очень просто понять почему и когда были изменены те или иные строки кода, изучить историю и развитие кодовой базы проекта и многое другое).

**Текстовый редактор Atom**

Atom — это отличный, многофункциональный текстовый редактор, с возможностью установки огромного количества расширений, позволяющих приспособить его практически к чему угодно. Сам редактор разработан компанией GitHub на основе их платформы Electron, которая включает Node.js и движок рендеринга Chrome.

Благодаря этому она позволяет разрабатывать приложения для рабочего стола на основе веб-технологий. Мы разберёмся, как пользоваться редактором Atom для веб-разработки, какие плагины нужно установить, а также как правильно всё настроить, чтобы получить удобное рабочее окружение.

**Установка Atom.**

Для операционной системы Windows или MacOS редактор Atom можно загрузить с GitHub репозитория проекта или с [официального сайта](https://atom.io/). Затем надо распаковать полученный архив и запустить atom.exe. Программа портативная, поэтому установки не требует. После завершения установки вы можете найти программу в главном меню системы.

**Как пользоваться редактором Atom**

1. Установка пакетов Atom. Для установки нового пакета на вкладке Welcome Guide выберите **Install a package**, а затем **Open Installer**. Это же окно можно открыть через главное меню. Для этого откройте **Edit -> Preferences -> Install**. Здесь есть поле поиска, в которое нужно ввести имя пакета для установки. Далее для установки пакета нужно выбрать пакет из списка найденных и нажать кнопку **Install**. Когда установка завершится, вы сможете открыть его настройки кнопкой **Settings** или деактивировать его кнопкой **Disable**.
2. Русификация Atom. Я не рекомендую русифицировать программу. Если вы разработчик, так как английский вам обязательно понадобится и не нужно избегать возможности его практиковать. К тому же ничего сложного в меню нет, а в интернете есть множество инструкций на русском. Но если совсем тяжело или не в много ту, то для этого установите пакет atom-i18n. Это не просто русификатор Atom, пакет содержит поддержку огромного количества языков. Когда пакет будет установлен, нажмите кнопку **Settings** под именем пакета и в поле **Language** выберите Русский. Затем вам останется перезагрузить редактор. Теперь главное меню и контекстное меню будут на русском.

Уже сейчас редактор кода Atom готов к использованию, но мы можем сделать его ещё лучше.

**Работа с html и css**

1. **Emmet** - пишем много кода с помощью простых сокращений. Нажатие на Tab разворачивает сокращение (пример использования был приведен выше, так же дополнительно вынесено отдельным файлом: Базовый синтаксис плагина EMMET);
2. **HTML Preview** - смотрите результат своей работы, не выходя из редактора (аналог Plunker)
3. **W3c Validation** - проверьте ваш HTML и CSS код на наличие ошибок
4. **Multi-cursor** - редактируйте сразу несколько строк/слов обновременно
5. **Linter** - подсвечивает ошибки в коде
6. **Autoprefixer** - добавляет префиксы к CSS для работы во всех браузерах
7. **Auto-close HTML** - автоматически закрывает открытые html-теги

**Что такое GIT**

Git - система управления контроля версиями, репозитории Git содержат файлы, историю, конфигурации управляемые Git. Данные могут отправляться из локальной папки в Git репозиторий и обратно, локальное состояние подразумевает рабочую папку которая содержит измененные файлы готовые для помещения в .git каталог, при помещении в данный каталог - Commit, указываются комментарии к данному коммиту, что по факту является историей, удаленный репозиторий содержит ветки Branches, основная ветка называется Master, данные могут отправляться как в Master, так и другие ветки (ответвления) проекта.

GIT является одной из самых популярных систем. Её отличие от других программ — отсутствие графической версии. Поэтому работа с Git ведётся через командную строку. В разных операционных системах свои программы для взаимодействия с Git.

В Windows их две: PowerShell и cmd.exe. В Ubuntu это Terminal. Самая популярная программа на macOS тоже называется Terminal. Если вам не подходит встроенная в систему программа для работы с командной строкой, вы можете поставить свою. Например, написанную на JavaScript программу Hyper, которая работает на любой операционной системе. На Windows популярны программы Cmder и Git Bash, а на macOS — iTerm.

В мире разработки такие программы называют «терминал» или «консоль». А работает это так: мы вводим команду и получаем реакцию машины: сообщение об ошибке, запрос на подтверждение информации, результат выполненных действий.

**Установка GIT**

Если вы ранее не работали с GIT, то для начала его нужно установить. В зависимости от системы нужно выбрать свой вариант.

**Установка GIT в Linux (Ubuntu)**

В зависимости от вашего дистрибутива Linux требуется установить через консоль, например в ubuntu эта команда будет иметь следующий вид:

**sudo apt install git**

**Установка на macOS**

1. Скачиваем Git со страницы [проекта](https://git-scm.com/download/mac).
2. Запускаем загруженный файл.
3. Система может показать окно с ошибкой, где будет написано, что файл скачан с неавторизованного сайта и инсталлятор не может быть запущен. В таком случае нужно зайти в «Системные настройки» — «Безопасность» (Security and Privacy), в появившемся окне будет сообщение об ошибке и кнопка Open anyway (Всё равно открыть). Нажимаем.
4. Система покажет окно, уточняющее хотите ли вы запустить установку. Подтверждаем действие.
5. Установщик проведёт через все необходимые шаги.

**Установка в Windows**

Скачайте exe-файл инсталлятора с [сайта Git](https://git-scm.com/download/win) и запустите его. Это Git для Windows, он называется msysGit. Установщик спросит добавлять ли в меню проводника возможность запуска файлов с помощью Git Bash (консольная версия) и GUI (графическая версия). Подтвердите действие, чтобы далее вести работу через консоль в Git Bash. Остальные пункты можно оставить по умолчанию. Проверим, что Git установлен.

После того, как все действия по установке завершены, убедимся, что Git появился в системе компьютера. Откройте терминал и введите **git --version**, должна появиться текущая версия программы на вашей машине. Эта проверка подходит для всех операционных систем.

**Настройка Git**

После установки производим настройку своего профиля вводя в терминал поочереди команды:

* git config --global user.name "ВАШЕ\_ИМЯ"
* git config --global user.email АДРЕС

Заменив значения ВАШЕ\_ИМЯ и АДРЕС вашими значениями.

После указания своих данных, можно их просмотреть:

* git config --global --list

Обратите внимание, что в командах, указанных выше, есть опция --global. Это значит, что такие данные будут сохранены для всех ваших действий в Git и вводить их больше не надо. Если вы хотите менять эту информацию для разных проектов, то в директории проекта вводите эти же команды, только без опции --global.

**GITHUB**

GitHub — веб-сервис, который основан на системе Git. Это такая социальная сеть для разработчиков, которая помогает удобно вести коллективную разработку IT-проектов. Здесь можно публиковать и редактировать свой код, комментировать чужие наработки, следить за новостями других пользователей. Чтобы начать работу с GitHub, нужно зарегистрироваться на сайте, если вы ещё этого не сделали.

После того как у вас будет создан аккаунт в Github можно будет начать полноценно работать с ним.

Копирование репозитория Git в локальную папку

Для начала определим, что такое репозиторий. Это рабочая директория с вашим проектом. По сути, это та же папка с HTML, CSS, JavaScript и прочими файлами, что хранится у вас на компьютере, но находится на сервере GitHub. Поэтому вы можете работать с проектом удалённо на любой машине, не переживая, что какие-то из ваших файлов потеряются — все данные будут в репозитории при условии, что вы их туда отправите.

Копировать или клонировать репу c GitHub можно по HTTPS или SSH.

Команда для копирования репозитория:

* git clone ССЫЛКА\_НА\_РЕПОЗИТОРИЙ

После клонирования переходим в папку репозитория:

* cd НАЗВАНИЕ\_РЕПОЗИТОРИЯ

Смотрим статус:

* git status

Добавление данных в Git или коммит (commit)

Создаем файл с текстом:

* echo "This example Git text file" > example.txt

Смотрим статус:

* git status

Видим, что у нас есть файл готовый для загрузки в Git, добавляем его в репозиторий:

* git add example.txt

Снова смотрим статус, видим, что у нас появился новый файл example.txt, добавляем данный файл в репозиторий используя git commit:

* git commit -m "This first example text file"

**Отправка данных в онлайн Git репозиторий**

Отправить данные в репу можно используя команду git push:

* git push origin master

После ввода аутентификационных данных, произойдет отправка в онлайн репозиторий, после чего можно обновить страницу на GitHub дабы убедиться в том, что новый файл добавлен в него (репозиторий).

Обратите внимание, что если вы используете двухфакторную авторизацию на github и являетесь пользователем linux, то вам нужно в настройках, в разделе «Developers setting» создать новый Personal access tokens и данный токкен использовать вместо пароля при аутентификации.

**VISUAL STUDIO CODE + GIT**

Данное ПО является хорошим выбором для комфортной работы с GIT и GITHUB. В VSCode есть встроенный терминал, который можно вызвать с помощью комбинации клавиш: **ctrl + `**

Внутри данного терминала вы можете использовать те же команды для работы с GIT которые были описаны выше.

**Настройка терминала VS Code в Windows**

По умолчанию консоль VS Code в Windows запускает стандартную командную строку Windows, которая не может работать с GIT, чтобы все работало нужно запустить терминал bash, для этого выполним следующие действия:

1. Убедимся, что ранее нами был скачан и установлен [git](https://git-scm.com/download/win" \t "_blank), подробнее данный процесс был описан выше.
2. Запускаем Visual Studio Code.
3. Зажимаем сочетание клавиш **Ctrl + Shift + P** вводим **open user setting** и из выпадающего меню выбираем **Open User Settings**.
4. В параметры поиска вводим terminal.integrated.shell.windows и заменяем cmd.exe на C:\\Program Files\\Git\\bin\\bash.exe (если вы установили git в другую директорию то ссылка будет иметь вид относительно установленной вами директории).
5. Перезагружаем Visual Stuio Code
6. Открываем терминал и проверяем, что запущен bash терминал

**Вывод:** Опишите и сравните два текстовых редактора VSCode и Atom. Опишите работу с Git.