


## Вариативные работы

### BCP 1

Экспериментальная проверка корректности документов (валидация) на языках разметки

**Markup Validation Service**  
Check the markup (HTML, XHTML, ...) of Web documents

**Jump To:** [Notes and Potential Issues](#) [Congratulations · Icons](#)

**This document was successfully checked as -//B//DTD STUD 1.0 Strict//RU!**

<b>Result:</b>	Passed, <b>1 warning(s)</b>
<b>Source :</b>	<pre>&lt;?xml version="1.0" encoding="utf-8"?&gt; &lt;!DOCTYPE таблица PUBLIC "-//B//DTD STUD 1.0 Strict//RU" "http://kodaktor.ru/j/dtd_8d259"&gt; &lt;таблица&gt;   &lt;студент имя="Иванов"&gt;     &lt;отметка дисциплина="Информационные технологии"&gt;5&lt;/отметка&gt;   &lt;/студент&gt;   &lt;студент имя="Петрова"&gt;     &lt;отметка дисциплина="Мультимедиа"&gt;3&lt;/отметка&gt;     &lt;отметка дисциплина="Веб-дизайн"&gt;4&lt;/отметка&gt;     &lt;отметка дисциплина="Графика"&gt;5&lt;/отметка&gt;   &lt;/студент&gt; &lt;/таблица&gt;</pre>
<b>Encoding :</b>	utf-8 <input type="button" value="(detect automatically)"/>
<b>Doctype :</b>	-//B//DTD STUD 1.0 Strict//RU <input type="button" value="(detect automatically)"/>
<b>Root Element:</b>	таблица

### BCP 2

#### Функциональное программирование на языке JS

Функциональное программирование — парадигма программирования, в которой процесс вычисления трактуется как вычисление значений функций в математическом понимании последних (в отличие от функций как подпрограмм в процедурном программировании).

Принципы функционального программирования:

1. чистые функции;
2. иммутабельность;
3. ссылочная прозрачность;
4. функции как объекты первого класса;
5. функции высшего порядка;

## 6. рекурсия.

Чистыми называют функции, которые не имеют побочных эффектов ввода-вывода и памяти (они зависят только от своих параметров и возвращают только свой результат). Чистые функции для одних и тех же аргументов возвращают одинаковый результат.

Пример чистой функции:

```
function multiply(a, b) {  
  return a * b;  
}
```

Иммутабельность - невозможность изменения сущности. Пример функции:

```
function slugify(str) {  
  return str.toLowerCase().trim().split(' ').join('-');  
}  
  
let str = 'some url text';  
slugify(str);
```

Приведённые в коде функции позволяют создать функцию для преобразования строки, не меняющую исходную строку.

Если функция неизменно возвращает один и тот же результат для одних и тех же передаваемых ей входных значений, она обладает ссылочной прозрачностью.

Идея восприятия функций как объектов первого класса заключается в том, что такие функции можно рассматривать как значения и работать с ними как с данными. При этом можно выделить следующие возможности функций:

- Ссылки на функции можно хранить в константах и переменных и через них обращаться к функциям.
- Функции можно передавать другим функциям в качестве параметров.
- Функции можно возвращать из других функций.

Функции высших порядков — это такие функции, которые могут принимать в качестве аргументов и возвращать другие функции. В JavaScript стандартные методы массивов `filter`, `map` и `reduce` принимают в качестве аргументов функции.

# Подбор палитры

Выбрать другие цвета

Количество цветов:



#12f7bb, #30ef9a, #4de779, #6be059, #88d838, #a6d017

Сценарий реализован с использованием библиотеки `chroma.js`.

Шкала показывается от двух случайно выбранных цветов, которые генерируются с помощью метода `chroma.random()`. Для создания шкалы используется метод `chroma.scale(colors)`.

Код: <https://pastebin.com/0DE59uXa>

---

## BCP 4

### Настройка линтера кода

ESLint - это статический анализатор кода, который может находить синтаксические ошибки, баги и неточности форматирования.

Для работы ESLint необходимо установить Node.js и npm.

#### Настройка линтера eslint для Visual Studio Code

##### Установка eslint

```
npm install -D eslint
```

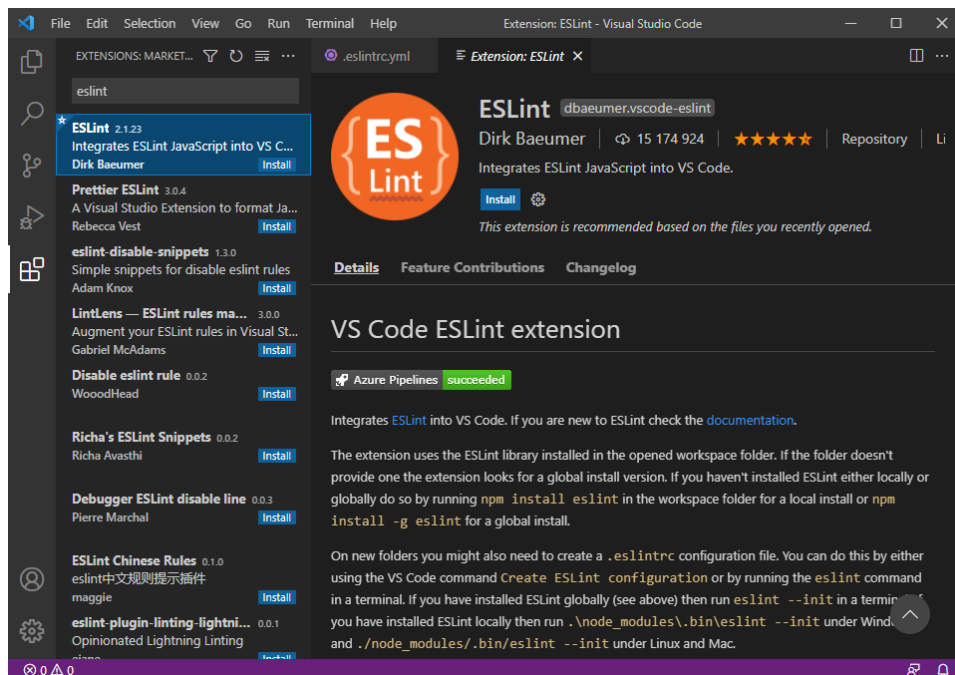
##### Установка плагинов eslint

```
npm install -D eslint-config-airbnb-base eslint-plugin-import
```

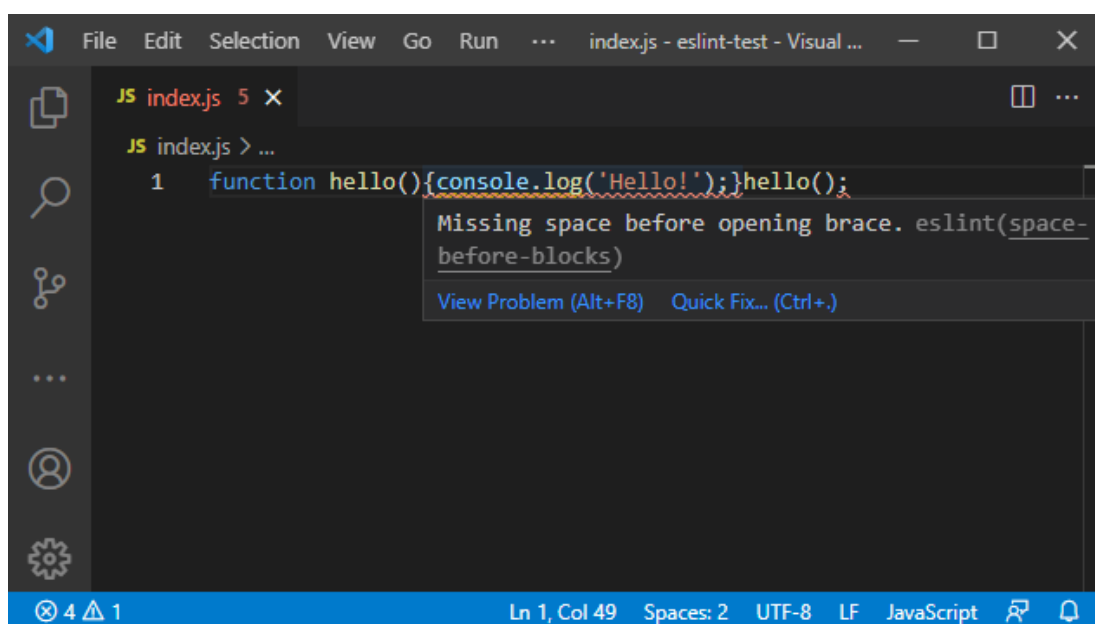
## Создание конфигурационного файла .eslintrc.yml

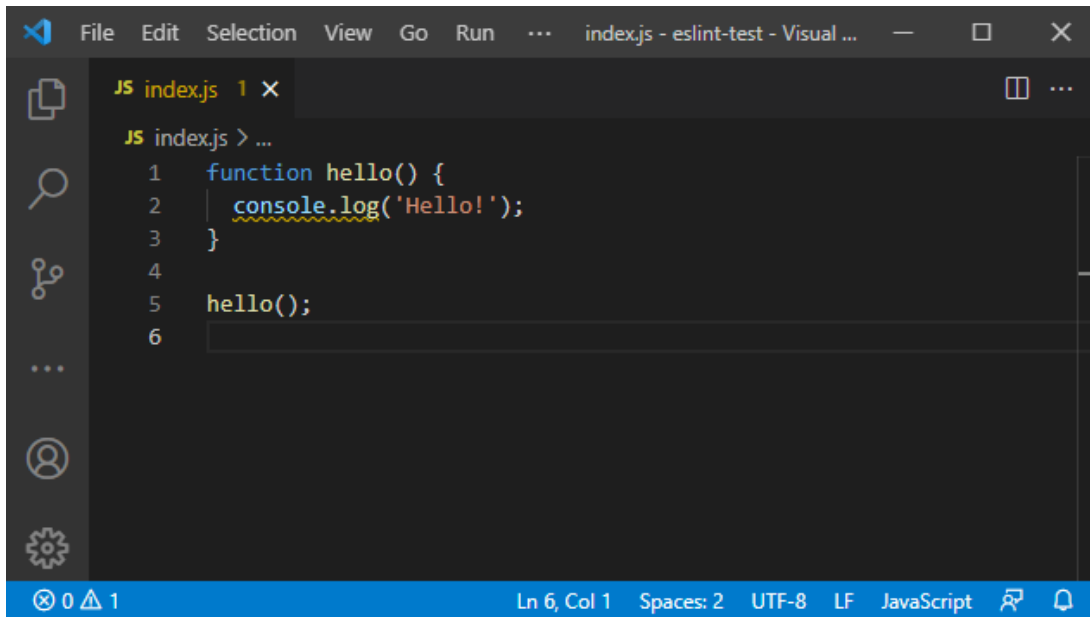
```
extends:  
  - 'airbnb-base'  
env:  
  node: true  
  browser: true
```

## Установка плагина ESLint



## Исправление кода с учётом рекомендаций линтера:





The image shows a screenshot of the Visual Studio Code editor interface. The title bar at the top indicates the file is 'index.js - eslint-test - Visual ...'. The editor window displays a JavaScript file named 'index.js' with the following code:

```
1 function hello() {  
2   console.log('Hello!');  
3 }  
4  
5 hello();  
6
```

The code is written in a dark theme. The 'console.log' function is underlined with a yellow squiggly line, indicating a linting error. The status bar at the bottom shows 'Ln 6, Col 1', 'Spaces: 2', 'UTF-8', 'LF', 'JavaScript', and a search icon.