Estándares de formato de código en Practicum de Yandex

Prólogo

Estas son las directrices internas de Practicum para formatear el código HTML y CSS.

Seguir estas directrices es importante ya que:

- vitarás algunos errores comunes que cometen los principiantes en HTML y CSS
- Estas directrices te ayudarán a aprender a formatear correctamente el código
- Te acostumbrarás a seguir las directrices internas, algo que tendrás que hacer en cualquier empresa tecnológica
- Estas hacen que el código sea más fácil de leer para tus revisores de código y futuros compañeros de trabajo

Estas directrices se aplican a los archivos de proyecto que contienen código HTML y CSS.

Herramientas de autoformateo

Editorconfig

En lugar de utilizar la sangría de cuatro espacios que viene por defecto en muchos editores de código, o una combinación de tabulaciones y espacios, tendrás que aplicar dos espacios a tu código. Aunque esto no sea fundamental en los primeros proyectos, es mejor desarrollar el hábito desde el principio.

```
<div class="parent">
  <div class="child">
  </div>
</div>
```

Pero, ¿por qué? La sangría de dos espacios hace que el código sea más compacto, y este es el estándar más utilizado por las empresas tecnológica.

Afortunadamente, en la mayoría de los editores de código, es fácil configurar la tecla Tab para establecer la sangría de dos espacios.

Visual Studio Code no es una excepción. Editorconfig es la forma más popular de hacerlo. Puedes encontrar instrucciones en la documentación oficial sobre cómo hacerlo directamente en Visual Studio Code.

Todo lo que tienes que hacer es buscar el plugin en el menú «Extensiones» e instalarlo.

Una vez instalado el plugin, crea un archivo `.editorconfig` en la raíz del proyecto con el siguiente contenido:

```
# http://editorconfig.org

# Una propiedad especial que debe ser especificada en la parte superior del archivo fuera de
# cualquier sección. Configura a true para detener la búsqueda de archivos .editorconfig en el archivo actual root = true
root = true

[*]
# Estilo de sangría
# Valores posibles: tabulación, espacio
indent_style = space

# Tamaño de la sangría en caracteres a un solo espacio
# Valores posibles: un número entero, tabulación
indent_size = 2

# Carácter de final de línea
```

```
# Possible values - lf, crlf, cr
end_of_line = lf

# Codificación de caracteres del archivo
# Possible values - latin1, utf-8, utf-16be, utf-16le
charset = utf-8

# Indica si se recortan los espacios en blanco al final de las líneas
# Valores posibles: true, false
trim_trailing_whitespace = true

# Desactivar para Markdown porque los espacios en blanco al final son importantes
[*.md]
trim_trailing_whitespace = false

# Indica si el archivo debe terminar con una nueva línea
# Possible values - true, false
insert_final_newline = true
```

Prettier

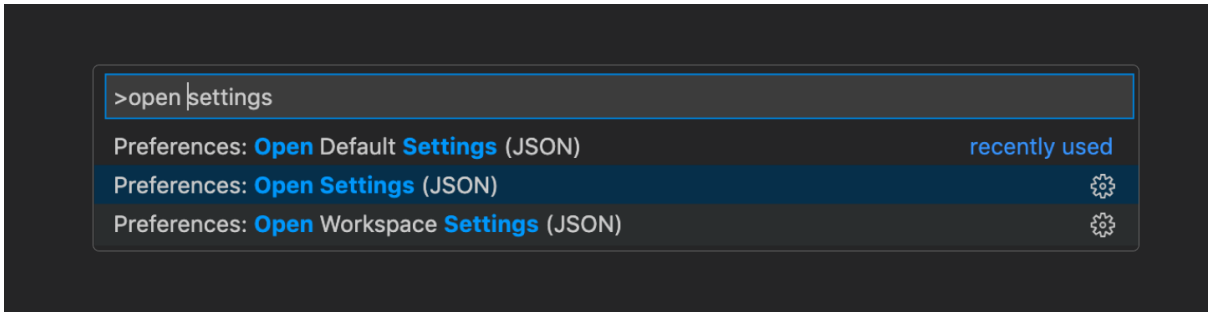
Más allá de los aspectos básicos manejados por EditorConfig, hay muchas formas de formatear el código. Pero en lugar de ofrecerte docenas de reglas que tienes que recordar, ¡una gran parte del formateo puede hacerse por ti de forma automática! Aquí te mostraremos cómo configurar el formateador Prettier, que hará precisamente eso.

¿Qué es Prettier? Prettier es un «formateador de código de opinión». Este se centra en el estilo, lo que significa que, dado un código desagradable, lo hará agradable, de manera que sea coherente para todos los miembros de tu equipo. Cuando decimos «obstinado» nos referimos a que tiene muchos ajustes específicos activados por defecto, por lo que, si los valores predeterminados te funcionan bien, no necesitas configurarlos.

Configuración de Prettier

Primero, instala la extensión Prettier VS Code. Entonces configuraremos VS Code para utilizar Prettier de forma predeterminada.

Para abrir tu archivo de configuración de VS Code, abre el panel de comandos `cmd-shift-p` o `ctrl-shift-p` y escribe «Abrir configuración». Selecciónalo de la lista como se indica a continuación:



Alt: A search bar with the text "Open Settings" and several suggested options below. The suggestions are "Preferences"Open Default Settings (JSON)", "Preferences"Open Settings (JSON)", and "Preferences"Open Workspace Settings (JSON)".

Agrega esta configuración a tu `settings.json` para establecer a Prettier como el formateador predeterminado para estos idiomas:

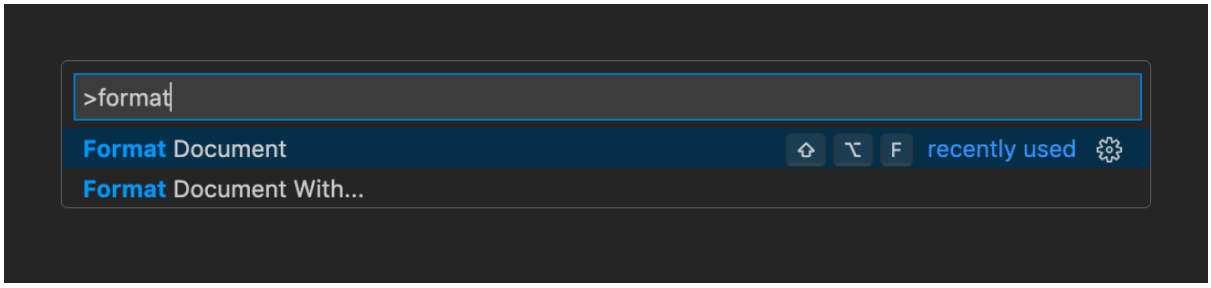
```
"[css]": {
  "editor.defaultFormatter": "esbenp.prettier-vscode"
},
"[html]": {
  "editor.defaultFormatter": "esbenp.prettier-vscode"
},
"[javascript]": {
```

```
  "editor.defaultFormatter": "esbenp.prettier-vscode"
},
"[javascriptreact]": {
  "editor.defaultFormatter": "esbenp.prettier-vscode"
},
"[json]": {
  "editor.defaultFormatter": "esbenp.prettier-vscode"
},
```

Si deseas que VS Code formatee tu código automáticamente para que nunca tengas que activarlo manualmente, te recomendamos estas configuraciones:

```
"files.autoSave": "onFocusChange", // guardar cada vez que cambies de ventana activa
"editor.formatOnSave": true, // formatear cada vez que guardes
```

O, si solo quieres formatear el código manualmente, puedes desactivar estas configuraciones, y abrir el panel de comandos de nuevo para buscar «Formatear documento». También se te mostrará qué atajos de teclado puedes utilizar para formatear tu archivo rápidamente sin recurrir al panel de comandos.



Alt: A search bar with the text "Format" entered and several suggested options below. The suggestions are "Format Document" and "Format Document With..."

Directrices generales sobre el formato del código

Prettier y EditorConfig no pueden encargarse de todo, así que algunas cosas las tendrás que controlar tú mismo. Las hemos recopilado a continuación:

HTML

Validar tu código HTML

Comprueba que tu código sea válido. Las herramientas como el validador de HTML del W3C, están disponibles para ayudarte a hacer exactamente esto. Comprobar periódicamente la validez de tu código te permitirá detectar los errores antes de que se acumulen.

Utilizar correctamente los elementos semánticos

Utiliza los elementos HTML para los propósitos previstos. Por ejemplo, los elementos `<h1>` a `<h6>` sirven para dar formato a los encabezados, las etiquetas `<p>` son para los párrafos y el elemento `<a>` es para los enlaces. Esto proporciona accesibilidad a las personas con problemas de visión, hace que el código sea más reutilizable y mejora los resultados cuando el sitio es analizado por un motor de búsqueda u otras herramientas.

```
<!-- no está recomendado -->
<div onclick="goToRecommendations();">All recommendations</div>

<!-- recomendado -->
<a href="recommendations/">All recommendations</a>
```

Establecer textos `alt` para las imágenes

Para mejorar la accesibilidad, describe las imágenes con texto utilizando el atributo `alt`.

El contenido alternativo es un aspecto importante para que tu página web sea accesible. Un usuario con discapacidad visual podría confiar en el valor del atributo `alt` para entender una página.

Para las imágenes decorativas y otras imágenes que no necesitan texto alternativo, agrega un atributo `alt` vacío, es decir, `alt=""`, para que el navegador pueda omitirlas en lugar de leer el nombre del archivo en voz alta.

```
<!-- no está recomendado -->
<img src="spreadsheet.png">

<!-- recomendado -->
<img src="spreadsheet.png" alt="Screenshot of a spreadsheet.">
```

No utilices estilos en línea

La mejor práctica es mantener el marcado (HTML) y el estilo (CSS) en archivos separados. Esto hace que sea más fácil mantener tus estilos organizados y reutilizables.

```
<!-- no está recomendado -->
<h1 style="font-size: 30px; font-family: Roboto, sans-serif">A lovely title</h1>

<!-- recomendado -->
<h1 class="page-header">A lovely title</h1>
```

```
/* in .css file */
.page-header {
  font-size: 30px;
  font-family: Roboto, sans-serif;
}
```

CSS

Validar el código CSS

Al igual que con HTML, puedes utilizar un validador para comprobar que tu código sea válido. El validador de CSS de W3C detecta errores de sintaxis y estructura.

La validación garantiza que la página web se mostrará en un navegador de forma fiable y correcta.

Dale nombres significativos a las clases y a los identificadores

Utiliza nombres que reflejen lo que hacen los bloques y los elementos. Esto hace que el código sea más fácil de leer y entender. Utiliza los selectores de clase para trabajar con CSS.

Utiliza la metodología BEM.

```
/* no está recomendado: un nombre sin sentido */
.rumpelstiltskin {}

/* no está recomendado: un nombre descriptivo general */
.button-green {}

/* recomendado: un nombre descriptivo específico */
.gallery {}
```

Una última palabra

¿Son estas normas vinculantes?

Las reglas establecidas aquí no son limitaciones ni normas técnicas. Romperlas no supone ninguna catástrofe. Son nuestras directrices internas para formatear el código y ayudan a mantener las cosas claras y coherentes.

Trabajando con código desconocido

Cuando trabajes en una base de código preexistente, como, por ejemplo, cuando te contraten después de este curso, deberás seguir utilizando el estilo existente de esa base de código (a menos que se te indique lo contrario).

Antes de empezar a trabajar en el código, infórmate del formato que tiene el código con el que vas a trabajar. Sigue los mismos principios que los anteriores desarrolladores del proyecto, aunque no coincidan con los tuyos. Cualquier inconsistencia en el estilo hace que sea más difícil de entender para cualquier otra persona que lea el código.