

Создание семантической разметки по макету

Выполнен на 0%

Введение

Углублённая теория

Подходы к разметке

Работа со спецификацией

Проверка валидности

Часто используемые теги и типовые ошибки

Методика написания семантической разметки

Кейс 1, лёгкий уровень

Кейс 2, лёгкий уровень

Кейс 3, лёгкий уровень

Кейс 4, средний уровень

Кейс 5, средний уровень

Кейс 6, средний уровень

Кейс 7, сложный уровень

Кейс 8, сложный уровень

Кейс 9, сложный уровень

Бонусный кейс 10, лёгкий уровень

Бонусный кейс 11, средний уровень

Бонусный кейс 12, сложный уровень

Главная / Моё обучение / Создание семантической разметки по макету / Углублённая теория /

Проверка валидности

Посмотрите на пример кода ниже и попытайтесь найти ошибки синтаксиса.

```
1  <!DOCTYPE html>
2  <html lang="ru">
3    <head>
4      <meta charset="UTF-8">
5      <title>Барбершоп «Бородинский»</title>
6    </head>
7    <body>
8      <header class="main-header">
9        <nav class="main-navigation">
10         <ul class="site-navigation">
11           <p><a href="info.html">Информация</a></p>
12           <li><a>Новости</a></li>
13         </ul>
14       </nav>
15     </header>
16     <section class="features">
17       <ul class="features-list">
18         <li class="feature-item">
19           <strong>Быстро</strong>
20           <p>Мы делаем свою работу быстро!</p>
21         </li>
22       </div>
23     </section>
24   </body>
25 </html>
```

Код с ошибками в разметке

Правильный ответ

Если вы проделали это упражнение, то должны были найти две ошибки: неправильно закрытый тег и неправильно вложенный тег. Они подсвечены.

```
1  <!DOCTYPE html>
2  <html lang="ru">
3    <head>
4      <meta charset="UTF-8">
5      <title>Барбершоп «Бородинский»</title>
6    </head>
7    <body>
8      <header class="main-header">
9        <nav class="main-navigation">
10         <ul class="site-navigation">
11           <p><a href="info.html">Информация</a></p>
12           <li><a>Новости</a></li>
13         </ul>
14       </nav>
15     </header>
16     <section class="features">
17       <ul class="features-list">
18         <li class="feature-item">
19           <strong>Быстро</strong>
20           <p>Мы делаем свою работу быстро!</p>
21         </li>
22       </div>
23     </section>
24   </body>
25 </html>
```

Ошибки в разметке

И что, действительно верстальщики каждый раз проверяют свой код именно так, вручную? Конечно же, нет. Строгий синтаксис тем и хорош, что его достаточно легко проверять в автоматическом режиме. Поэтому есть специальный сервис для проверки, или валидации, вашего HTML-кода. Называется он — **валидатор W3C**.

Валидатор берёт ваш код и проверяет его на формальные, строгие правила: правильно ли закрыты теги, правильно ли вложены теги и так далее. Для этого валидатор использует всё те же два поля, с которыми мы работали: Categories и Content model — и универсальное правило про правильный порядок вложенности.

Валидатор проверяет всю разметку по спецификации HTML. Чтобы правильно валидировать документ, он использует `<DOCTYPE>`. Если вы написали `<DOCTYPE HTML>` — вы работаете по современной версии HTML 5, в которой в ссылки можно вкладывать заголовок, потому что у ссылки прозрачная модель контента. Но если у вас `<DOCTYPE>` старый, то в этом случае валидатор выдаст ошибку.

1. **Error** Element `p` not allowed as child of element `ul` in this context. (Suppressing further errors from this subtree.)

From line 11, column 11 to line 11, column 13

<p><a href=

Contexts in which element `p` may be used:

Where **flow content** is expected.

Content model for element `ul`:

Zero or more `li` and **script-supporting** elements.

2. **Error** Stray end tag `div`.

From line 22, column 7 to line 22, column 12

</div>

3. **Error** End tag `section` seen, but there were open elements.

From line 23, column 5 to line 23, column 14

</div></section>

4. **Error** Unclosed element `ul`.

From line 17, column 7 to line 17, column 32

<ul class="features-list">

5. **Warning** Section lacks heading. Consider using `h2`–`h6` elements to **add identifying headings to all sections**.

From line 16, column 5 to line 16, column 30

<section class="features">

Пример работы валидатора

Почему валидатор нашёл сразу пять строгих ошибок? Зачастую одна ошибка, например, незакрытый тег, вызывает целый каскад последующих. Чтобы правильно исправлять ошибки при валидации, необходимо делать это сверху вниз. Как только вы замените `<p>` на `` или правильно закроете `<div>`, часть последующих ошибок может исчезнуть.

Отличие ошибки и предупреждения

Ошибки помечаются как **error**, и их нужно обязательно исправлять. Это несоответствие формальному синтаксису языка.

Предупреждения помечаются как **warning**. И это те вещи, которые валидатор проверяет, смотря не на метаданные тега, а на описание, то есть, на его предназначение. Эти предупреждения менее строгие, исправлять их необязательно.

Например, у тега `<section>` есть рекомендации от разработчиков, о том, что неплохо в каждом теге `<section>` иметь заголовок. Если валидатор видит этот тег и не видит в нём заголовка, он может показать вам предупреждение. Необходимо ли это править? Не всегда. Error — обязательный к исправлению, warning — обязательный к размышлению.

Зачем вам, как профессионалам, нужно знать, как устроена спецификация? Дело в том, что иногда встречаются нестандартные вопросы, на которые вы не сможете ответить, используя справочники. Например, можно ли вложить внутрь `<footer>` тег `<section>`? Зная систему, по которой устроен стандарт, вы легко найдёте ответ.

Ознакомились со статьёй?

Сохранить прогресс

Работа со спецификацией

Часто используемые теги и типовые ошибки

Практикум

Тренажёры
Подписка
Для команд и компаний
Учебник по РНР

Профессии

Фронтенд-разработчик
React-разработчик
Фулстек-разработчик
Бэкенд-разработчик

Услуги

Работа наставником
Для учителей
Стать автором

Курсы

HTML и CSS. Профессиональная вёрстка сайтов
HTML и CSS. Адаптивная вёрстка и автоматизация
JavaScript. Профессиональная разработка веб-интерфейсов
JavaScript. Архитектура клиентских приложений
React. Разработка сложных клиентских приложений
РНР. Профессиональная веб-разработка
РНР и Yii. Архитектура сложных веб-сервисов
Node.js. Разработка серверов приложений и API
Анимация для фронтендеров
Вёрстка email-рассылок
Vue.js для опытных разработчиков
Регулярные выражения для фронтендеров
Шаблонизаторы HTML
Алгоритмы и структуры данных
Анатомия CSS-каскада

Блог

С чего начать
Шпаргалки для разработчиков
Отчеты о курсах

Информация

Об Академии
О центре карьеры

Остальное

Написать нам
Мероприятия
Форум