Theory: Block-level elements

12 minutes 0 / 5 problems solved

§1. What are HTML elements?

As you know, HTML files can be opened in browsers. After receiving the HTML document, the browser reads the tags written in it and then uses them to create an HTML page that users see on their monitor screens.

All you see on the page in your browser viewer are HTML elements. And this is where the difference between HTML tags and HTML elements lies: elements are what the user sees on the browser page, and tags are what the developer writes when creating an HTML document.

> It is worth remembering that not all HTML tags are elements. For example, the `DOCTYPE` instruction, necessary for correct interpretation of code by a browser, and tags in the header of an HTML document are not elements.

There are two main types of page elements: **block-level** and **inline**. Each of them has its own peculiarities.

In this topic, you'll find out what block-level elements are needed for, what properties they possess and how to use them most effectively in creating the structure of HTML code.

§2. Block-level Elements

Block-level elements are mostly used to create the structure of web pages or logically divide an HTML document into parts. Here are some examples of block elements:

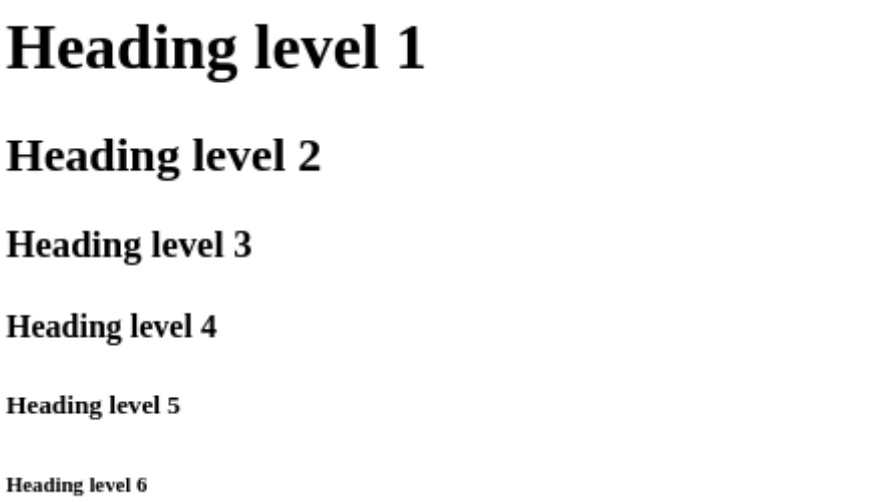
- `<div>` is a universal content divider. It is used to group similar HTML elements:

```
<div>
  <h1>Element h1 is inside the div</h1>
  <p>Element p is also inside the div</p>
</div>
```

- `<h1>-<h6>` are heading tags:

```
<h1>Heading level 1</h1>
<h2>Heading level 2</h2>
<h3>Heading level 3</h3>
<h4>Heading level 4</h4>
<h5>Heading level 5</h5>
<h6>Heading level 6</h6>
```

The result will be displayed in the browser as follows:



Current topic:

Block-level elements

Topic depends on:

× HTML page structure

Topic is required for:

HTML attributes id and class

Basic syntax

DOM

Table of contents:

As you can see, these tags make it easy to identify the headings that convey the importance of the text.

It is recommended to use only one `<h1>` tag on a web page -- this should be the title showing the main theme of the page.

- `<p>` defines a paragraph of text:

```
<p>It's a paragraph of the text</p>
<p>And this is another paragraph</p>
```

This is how it looks in the browser:

It's a paragraph of the text

And this is another paragraph

The text enclosed in this tag is not highlighted in bold, as in the case of `<h1>` - `<h6>`.

- `<pre>` defines a block of formatted text.

```
<pre>
.........../ >  フ
.........| _  _ |
......./` ミ_xノ
....../    |
...../  ヽ   ノ
..../  ヽ   ノ
....|  | | |
/ ￣|   | | |
| (￣ヽ_ヽ_)__)

.\二つ

</pre>
```

By default, any number of spaces occurring in the code in a row are shown as one on a web page. The `<pre>` tag displays the text as you want, with all spaces between characters:

```
..........／>   フ
.........|  _  _ |
.......／` ミ_xノ
....../      |
...../  ヽ   ノ
....|   |  |  |
／￣|   |  |  |
| (￣ヽ__ヽ_)__)
.＼二つ
```

- `<hr>` creates a horizontal line:

```
<hr>
```

Now let's see how it's displayed in the browser:

This is by far not a complete list of block-level elements – there are definitely many more to know.

§3. Features of block-level elements

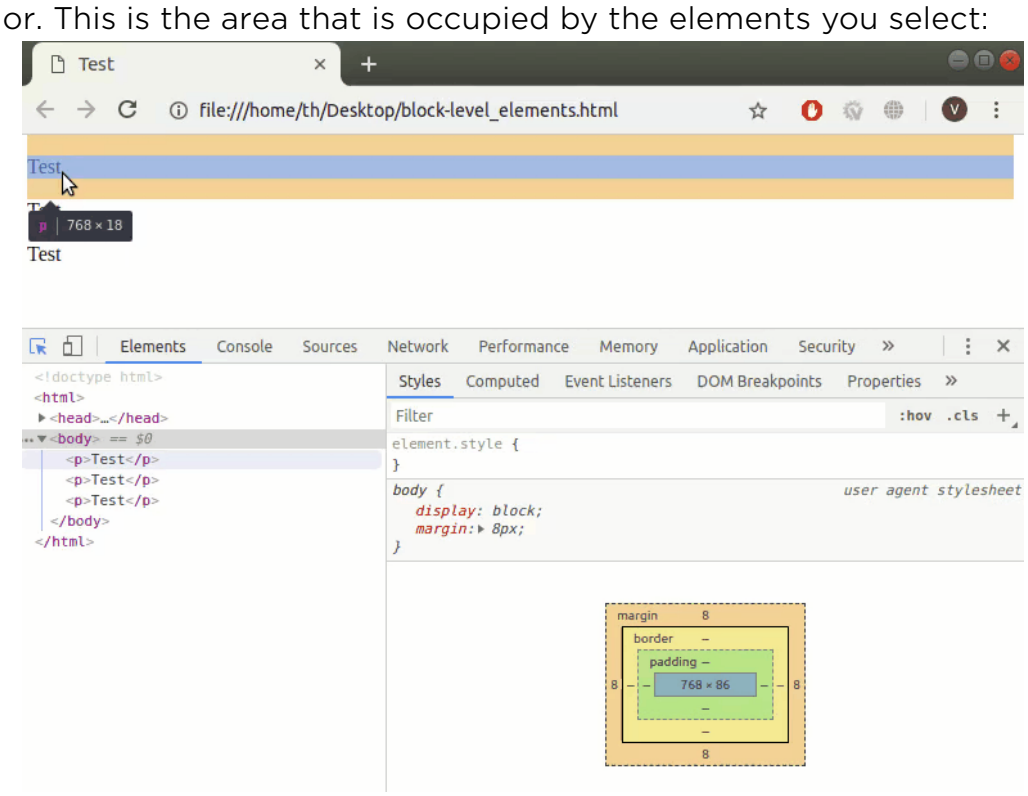
The following features are characteristic of block-level elements:

1. You can put both inline and other block-level elements in them.
2. Before and after, the browser makes a line break. Take a look at the behavior of block-level elements and compare it with that of inline

elements to better understand this feature:

3. The width of block-level elements occupies all the allowed space. That is, if the element is located inside `<body>`, it will occupy the entire width of the web page. If the block-level element is inside another, it will begin to occupy the entire width of the element in which it is located.

   To verify it, just press the *Ctrl+Shift+I* or *Cmd+Opt+I* keys on a web page and point the mouse cursor at the object of interest. In the browser, rectangles of different height and width will be highlighted in color. This is the area that is occupied by the elements you select:



4. The height of the block-level element is calculated automatically by the browser based on the block contents.

Block level elements behave like that because they act like containers that contain and take all content within it.

Separation of elements into block-level and inline was used up to version 4.0. In HTML5 these concepts were replaced by a more complex set of content categories, according to which each HTML element must follow the rules determining what information can be stored in it. We shall take a closer look at that a bit later; for now let's cement what we learned so far with a bit of practice as this knowledge is likely to come in handy sometime.

Report a typo

234 users liked this theory. **6** didn't like it. **What about you?**

Start practicing

Comments (2) Hints (0) Useful links (0) Show discussion