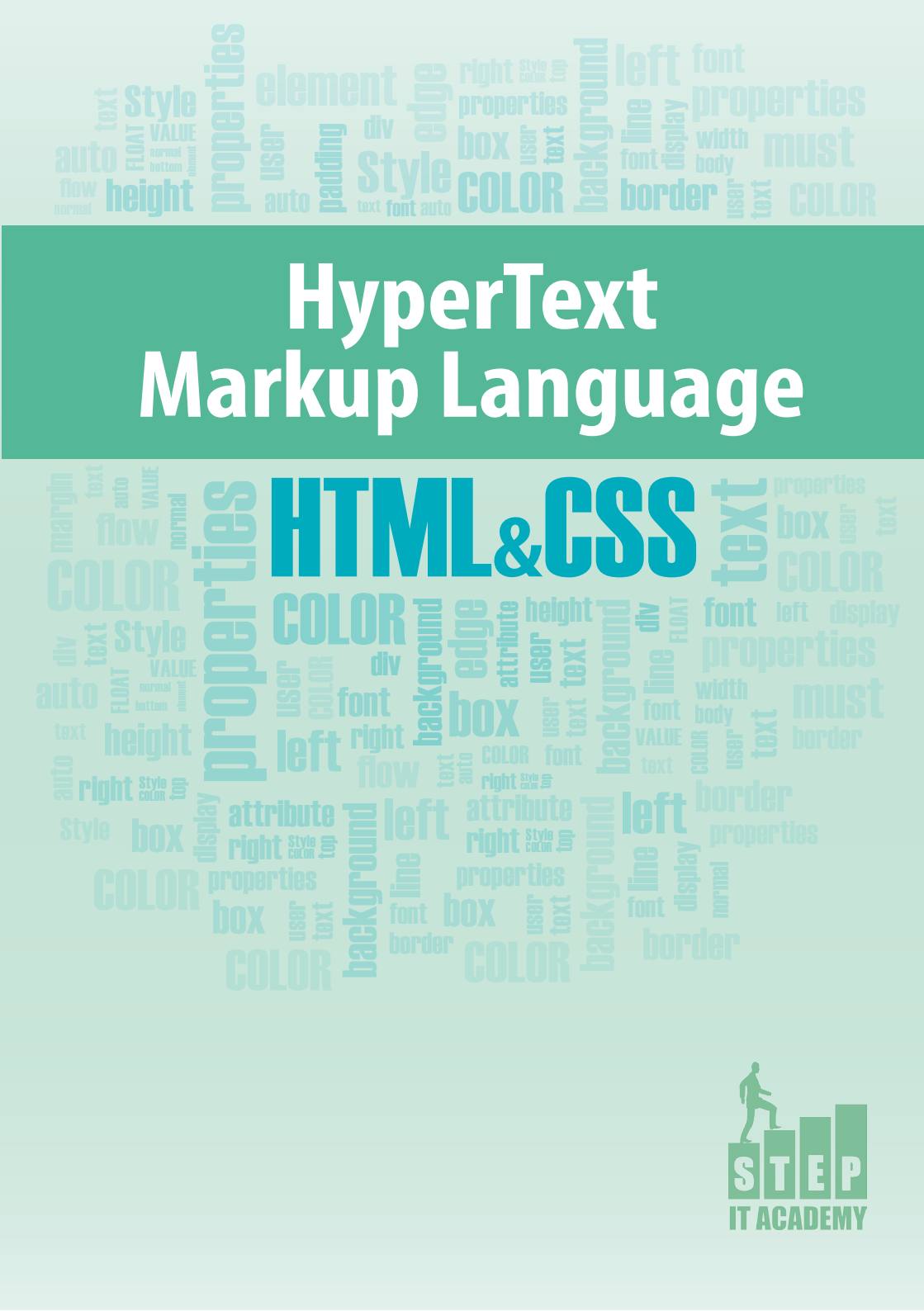
HyperText Markup Language



STEP IT ACADEMY

Unit 5

Block Layout

Contents

Block Layout

layers

Despite the fact that the layer term is pretty often used at website development, detailed explanation of this term is rarely found in website development literature. Hereafter I will often use the layer term, so at first we need to define what it means.

Initially, the Netscape company introduced layers by adding the <layer> tag support to its browser. This tag allows hiding/showing current content, setting location relative to the browser window, putting layer over other layers, and loading data to the layer content from a file. All these parameters were changed using JavaScript, and it added options on creating really dynamic content on a page. Despite such impressive set of options, the <layer> tag was not included in the HTML specification and it remained a simple extension in the Netscape browser.

Although the necessity in the specified options became imminent and in the end of 1996 syntax for working with layers was developed and approved in working draft of the CSS Positioning (CSS-P) consortium. Styles took major load: they control appearance of any element, including changing values dynamically via JavaScript. Sadly, object models of browsers for access to elements differed, so a pretty complicated code that would take into account these special issues was to be written.

Now developers of popular browsers adhere to HTML and CSS specifications, which made site developers' lives easier since they need less time to test websites in various browsers.

Nevertheless, there are differences in approaches to browsers and in case they appear, developers adhere to the following.

If websites are slightly different in various browsers, a blind eye is turned to it. To be blunt, they are not fixed. Visitors won't pixel-wise compare a website in various browsers. Although mind that the website should be always displayed correctly and without errors.

If the website has significant differences when displaying in another browser, they are dealt using hacks.

Hack is a set of techniques when a certain browser receives a code that is perceived by this browser only, but ignored by the rest.

Despite that hacks work, it is better to use them to a limited extent or to do without them at all. The thing is, hacks decrease the universality of code, and in order to modify parameters of one element, a programmer needs to make changes in several places simultaneously.

There is another, perspective way: adhere to the CSS specification. Despite that browsers don't fully support it, they progress towards full support of various specifications (HTML, CSS, DOM). Thus, future browsers will be unified and they will display one and the same website correctly.

Let's get back to layers again. It is understood that they are directly related to styles. If so, is each element of the HTML code, to which styles are added, a layer? It is in a way. But we would be confused if we said layer instead of table or paragraph. So let's agree to apply this term to the tags only.

In HTML4 and XHTML, layer is a web page element created using the tag with a style design applied to it.

Thus, the block layout expression or layout using layers consists of constructive usage of the tags and styles. At that, the following principles should be adhered.

Division of content and design

HTML code should contain only markup tags and logical formatting tags, and any design should be beyond the code, it should be in styles. Such approach allows managing appearance of elements and content of a page independently. Thanks to this, several people can work on a website, at that everyone works independently from each other. Designer, layout designer, and programmer work on their tasks on their own, which decreases website development time.

Using the tag often

The tag is very important in block layout, it accomplishes lots of functions. In fact, it's a base for styles that turn it to one thing or another. It does not mean that only this tag is applied, for we need to add images and design text. But when laying out by the means of layers, the tag is a brick of layout, its foundation.

Thanks to this tag, the HTML code falls into several clear blocks, the code becoming more compact than at table layout, besides, search systems index it better.

Tables are applied for representation of table data only

During block layout, tables are used of course, but only when they are required, for example, to display numbers and other table data. An option of refusing tables at all is unwise, moreover harmful.

Let's summarize. In HTML4 and XHTML, layer is a basic element of the web page layout, when styles are actively applied and HTML and CSS specifications are supported. Under such an approach, the tag plays important role, which is often associated with layers. At some point, it is true, so from now forth let's agree to apply the term layer to the tag for which style identifier or class is specified. Thus, the expression "layer with the content name" means that the or tag is used.

Several new markup tags are added to HTML5 to designate various typed block of a page. For instance, <header> and <footer> are used to create sections in the top margin and bottom margin of a document, <nav> is used for navigation, <aside> for side panel. Adding such elements to the HTML specification aimed at decrease of the tag domination and giving more meaning to markup. That's why the term element is actively used in the HTML5 layout, it means a corresponding tag and element it creates.

The above block layout principles are preserved at that, except that is replaced by more meaningful tags in some cases.

Block Layout

As a rule, web page consists of many various elements that can have a complex structure. That's why when creating a web page, there is a need to position these elements in the right way, to style them so they are arranged on a page in the right way. That is, an issue of page markup creation, its layout, appears.

There are various ways, strategies, and types of layout. Initially, table layout was widely used since tables allow splitting web page area into rows and columns very easily. Rows and columns are easy to manage, any content can be easily positioned in them. That is what has determined popularity of table layout.

But table layout does not create the most flexible pages design-wise, which is especially relevant in the world without a uniform screen resolution, instead it has large TV screens, small tablet and phablet screens, very small smartphone screens, etc. Table layout didn't manage to satisfy this variety of screens. So it was gradually replaced by block layout. Block layout is a relative name of layout methods and techniques, when the CSS's float property is mainly used for layout of web pages, and the main building element of web pages is , i.e. block. Using the float property and div or other elements, you can create a page structure consisting of several columns like in table layout, but it is much more flexible.

The float property was discussed in one of the previous topics. Now we use it to create a two-column web page. Let's suppose that we have standard header and footer at the top and bottom, and two columns in the center: menu column or sidebar and main content column.

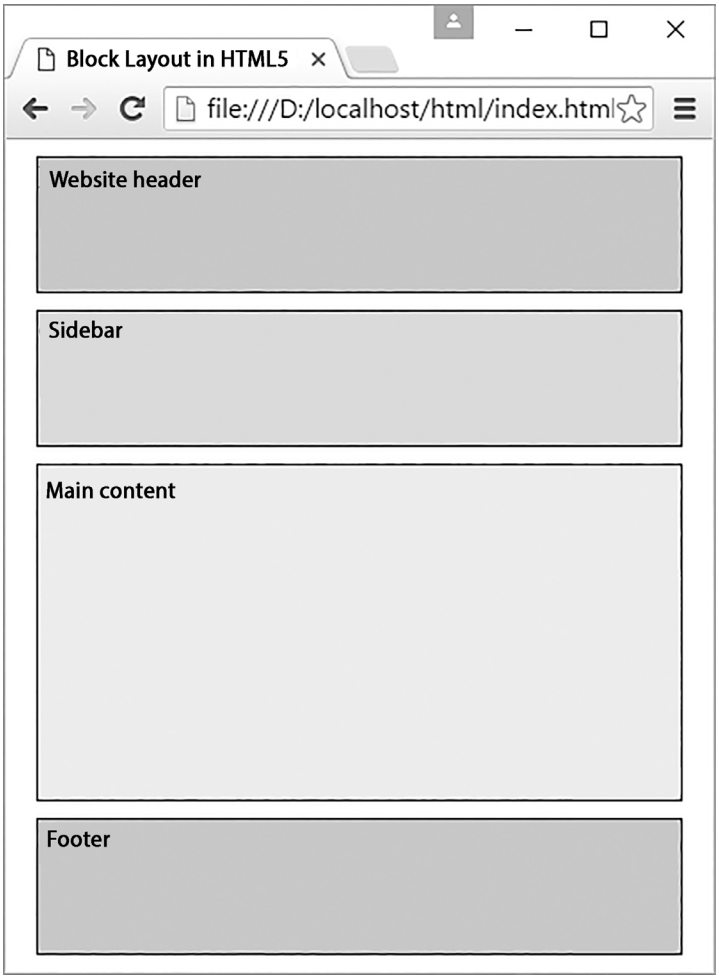
First we will specify all blocks. When working with elements that use wrapping and the float property, their order is important since the float element code that has the float property should go before the element that wraps the floating element. That is, the sidebar block will be stretched up to the main content block:

```
<!DOCTYPE html>
<html>

    <head>
        <meta charset="utf-8">
        <title>Block Layout in HTML5</title>
        <style>
            div{
                margin: 10px;
                border: 1px solid black;
                font-size: 20px;
                height: 80px;
            }
            #header{
                background-color: #ccc;
            }
            #sidebar{
                background-color: #ddd;
            }
            #main{
                background-color: #eee;
                height: 200px;
            }
            #footer{
                background-color: #ccc;
            }
        </style>
    </head>

    <body>
        <div id="header">Website header</div>
        <div id="sidebar">Sidebar</div>
        <div id="main">Main content</div>
        <div id="footer">Footer</div>
    </body>

</html>
```



Height, border, and block margins are added for decoration in this case, in order to identify the block area and separate it from others.

Then in order to move the sidebar block to the left relative to the main content block and to receive the wrapping effect, we need to specify the float: left property and preferred width of the sidebar block. Width may be fixed, for example,

150px or 8 em. Or you can also use percentage, for example, 30% — 30% of the body container width. On the one hand, fixed width blocks are easier to manage, but on the other hand, percentage value of width allows creating more flexible, responsive blocks that change their size relative to the change of the browser window size.

The last step is to specify margin between the main content block and sidebar block. Since the wrapping block can wrap a floating element on the right and at the bottom, if the floating element has a lower height, then we need to set a margin equal to the floating element width as a minimum. For example, if the sidebar width is 150px, we can set width of the main content block to 170px which allows creating blank space between two blocks.

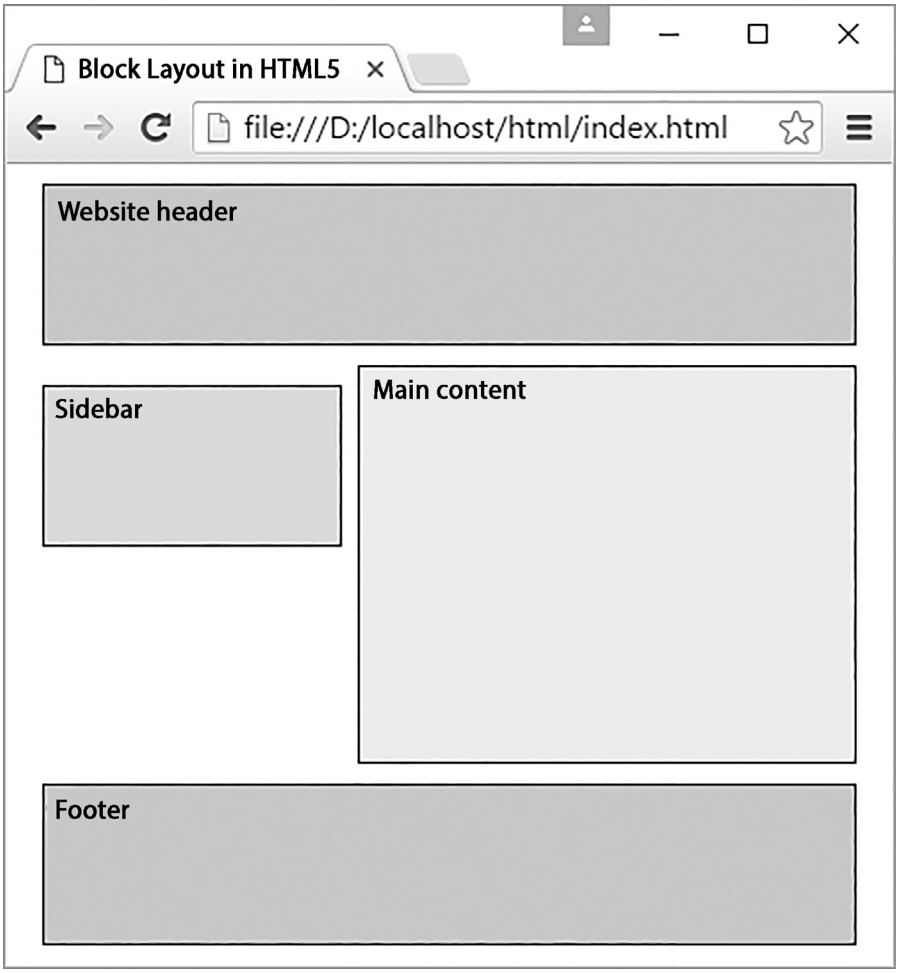
At that, it is not recommended to set width of the main content block explicitly since browsers widen it automatically to prevent it from occupying the whole space available.

So, taking into account everything above, let's alter styles of the sidebar blocks and main content as follows:

```
#sidebar{
    background-color: #ddd;
    float: left;
    width: 150px;
}

#main{
    background-color: #eee;
    height: 200px;
    margin-left: 170px; /* 150px (sidebar width) +
                10px + 10px (2 margins) */
}
```

As the result, we get a sidebar to the left of the main block:



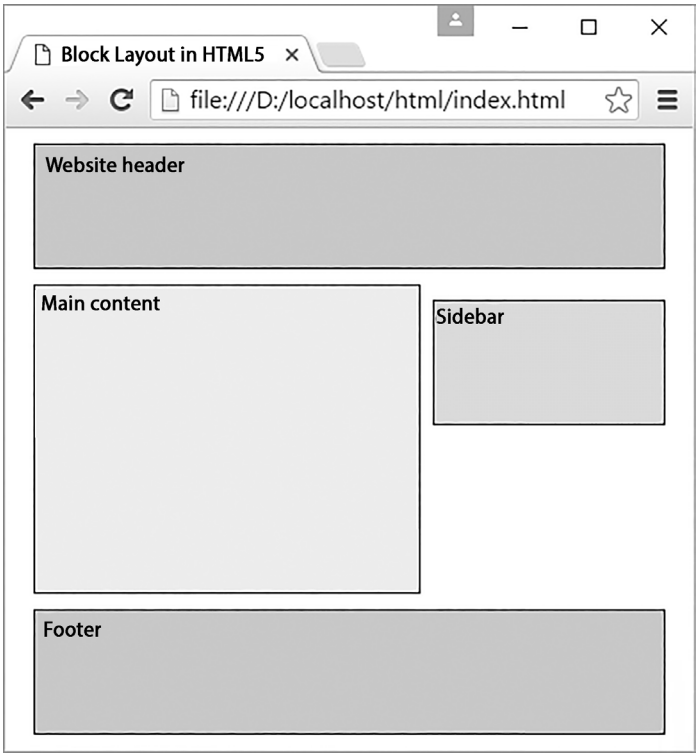
Block height is specified arbitrarily for better clarity; in real life, height is automatically specified by the browser as a rule.

Creation of the right sidebar is identical to creation of the left one, we just need to set the float: right value for the sidebar, and margin right for the main content block:

```
#sidebar{
    background-color: #ddd;
    float: right;
    width: 150px;
}

#main{
    background-color: #eee;
    height: 200px;
    margin-right: 170px;
}
```

At that, the html markup is the same: sidebar block still must precede the main content block.



Block Layout: Part 2

The previous topic covered creation of a page with two columns. We can add a larger amount of columns in such a way and create a more complicated structure. For example, let's add a second sidebar to the web page:

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>Block Layout in HTML5</title>
        <style>
            div{
                margin: 10px;
                border: 1px solid black;
                font-size: 20px;
                height: 80px;
            }
            #header{
                background-color: #ccc;
            }
            #leftSidebar{
                background-color: #ddd;
            }
            #rightSidebar{
                background-color: #bbb;
            }
            #main{
                background-color: #eee;
                height: 200px;
            }
            #footer{
                background-color: #ccc;
            }
        </style>
    </head>
```

```
    <body>
        <div id="header">Header of the web site</div>
        <div id="leftSidebar">Left sidebar</div>
        <div id="rightSidebar">Right sidebar</div>
        <div id="main">Main content</div>
        <div id="footer">Footer</div>
    </body>
</html>
```

Here again, the code of both sidebars should go up to the main content block that wraps them.

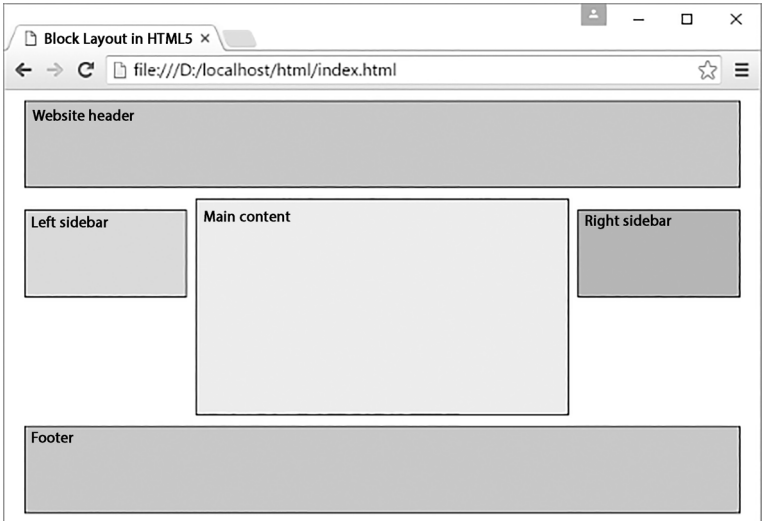
Now let's change styles of both sidebars and main content:

```
#leftSidebar{
    background-color: #ddd;
    float: left;
    width: 150px;
}

#rightSidebar{
    background-color: #bbb;
    float: right;
    width: 150px;
}

#main{
    background-color: #eee;
    height: 200px;
    margin-left: 170px;
    margin-right: 170px;
}
```

Again, we should specify width and the float property for both floating blocks (sidebars) one of them has the left value, and the other right.



Unit 5.
Block Layout