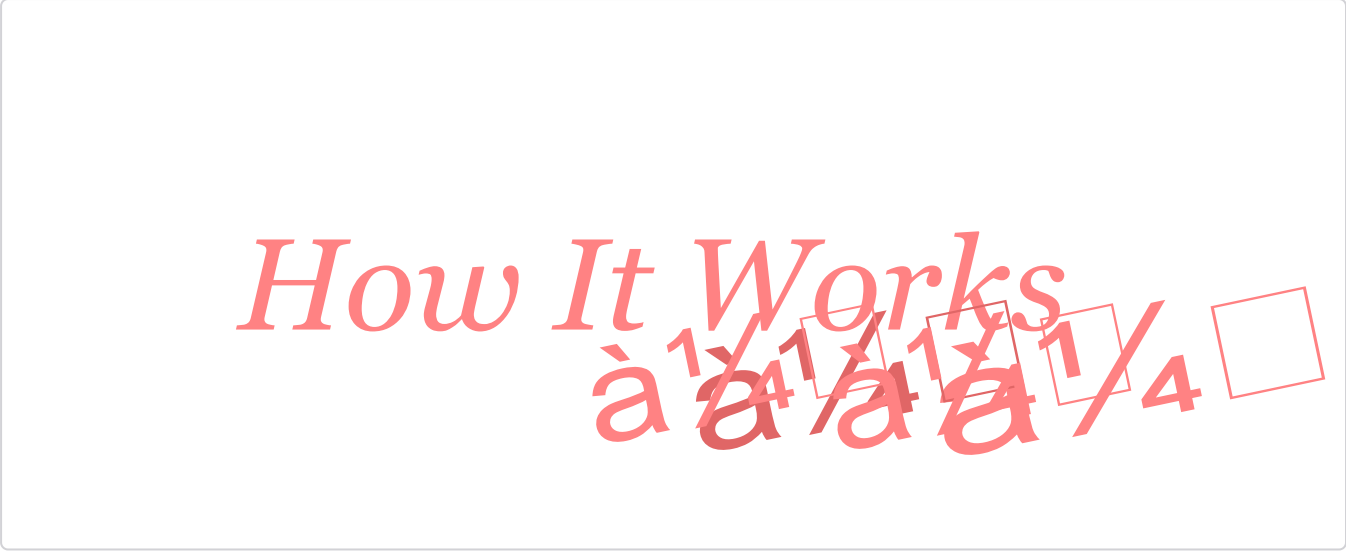
How it Works: Creating a Basic HTML Page

In this lesson, we will understand how we implemented our HTML from the last lesson.
Let's begin!

WE'LL COVER THE FOLLOWING

In the previous lesson, we brought our very first HTML page to life. Now is the time to unveil the mechanics behind our simple HTML page.



Mechanics behind our simple HTML page

We'll look at the **package.json** file first.

The package.json file defined what the `npm start` command should do:

```
{
  "name": "unraveling-html5-2nd-edition",
  "version": "1.0.0",

  "description": "Code samples for the Unraveling HTML5, CSS, and JavaScript book",
  "scripts": {
    "start": "live-server --port=8080"
  },
  "keywords": [],
  "author": "Istvan Novak (dotneteer@hotmail.com)",
  "license": "ISC"
}
```

As you typed `npm start` in the command line, `live-server` started a mini web server listening for requests on port `8080`, launched the web browser, and navigated to the address provided by Educative (the link given underneath the live widget).

As a result of the browser's request, the webserver retrieved the index.html file (the default file for the specified URL). The browser displayed the content of that file.

Now let's see how the browser processes the **index.html** file!

The file starts with the following line:

```
<!DOCTYPE html>
```

When reading it, the browser understands that this file will use **HTML5 markup** HTML5 is the newest version of the **Hypertext Markup Language**.

The file follows the standard structure of the markup that embeds a **head** and a **body** section enclosed between `<head>` and `</head>`, and between `<body>` and `</body>` elements, respectively.

So, what's the *difference* between head and body tags anyway?

- **Head Tag**: Material isn't rendered, meaning it is not displayed on the web page itself. This is usually information about the page such as its title, keywords that may be useful to search engines, and other data that is not considered document content.

- **Body Tag**: Material in the body tag is the actual content of the page and is displayed. Text, images, colors, graphics, audio, etc. can be added to the body. However, since style sheets are now the preferred way to specify a

body. However, since style sheets are now the preferred way to specify a

> document's presentation, the presentational attributes of BODY have been deprecated. More on this later in the CSS chapter.

The content of the whole markup is put within `<html>` and `</html>` elements. The *skeleton* of the page looks like this:

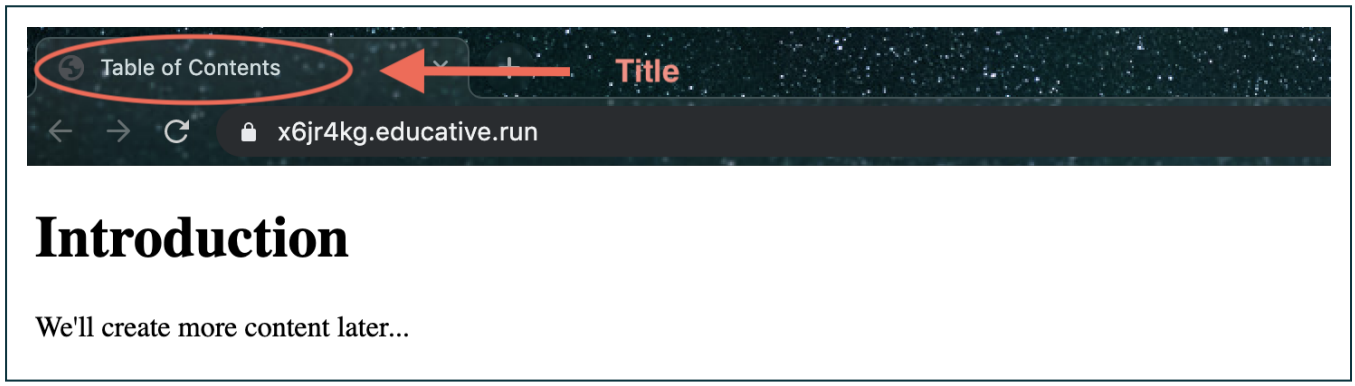
```
<!DOCTYPE html>
<html>
  <head></head>
  <body></body>
</html>
```

The markup you typed in **Step 1** adds three HTML markup elements to the originally empty page:

- `<title>`
- `<h1>`
- `<p>`

The `<title>` tag

As discussed above, the text within the `<title>` tag that is placed inside the `<head>` tag is a **metadata element**. This means it won't be rendered in the page. It defines the title of the page to be displayed *outside of the page* in the web browser, in most cases it defines it as the caption of the tab holding the page, as shown in the image below:



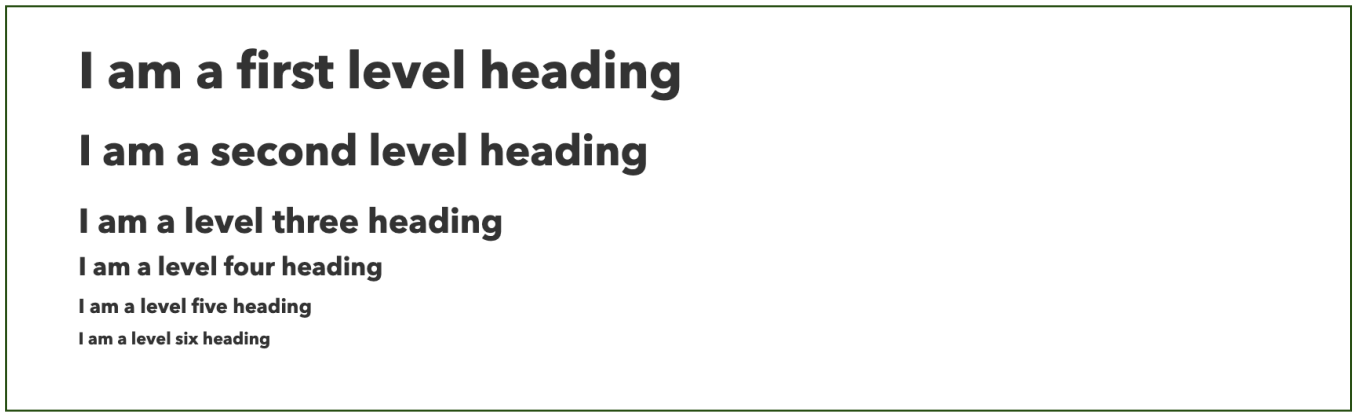
Title of the webpage displayed outside of the webpage

The `<h1>` tag

The `<h1>` element defines a **first level heading**. A first level heading is the

most important heading, hence it is largest of all headings.

The number next to h specifies the level of the heading. So `<h1>` is the first level heading, `<h2>` is the second level heading, and so on. `<h6>` defines the least important heading.



<h1> till <h6> level headings

The `<p>` tag

The `<p>` tag describes a simple paragraph. These are nested into the `<body>` section and they are rendered as the content of the page, as shown in the figure below.

two separate <p> tags used to make two separate paragraphs

In **Step 5**, you added a new `<p>` element to the markup as follows:

```
<p>We'll create more content later...</p>
<p>Live coding works!</p>
```

When the mini web server sent back the index.html file to the browser, it added a short JavaScript code that synchronized the web page with the server.

The server detected that the index.html file had been changed and notified the browser of this fact. The client-side JavaScript received this notification, re-queried index.html, and refreshed the screen.

Complete live demo at your service!

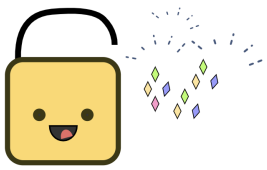
The complete implementation of the exercise from the previous lesson is given below for you to play around and experiment with.

> 📜**Note:** Remember to start the live server using the `npm start` command for this exercise.

Learn and enjoy! :)

```
<!DOCTYPE html>
<html>
<head>
  <title>Table of Contents</title>
</head>
<body>
  <h1>Introduction</h1>
  <p>We'll create more content later...</p>
  <p>Live coding works!</p>
</body>
</html>
```

Achievement unlocked! 🎉



Congratulations! You've learned to make your very first HTML page!

Good job! Give yourself a round of applause!

In the *next lesson*, we'll get a brief overview of how to format an HTML page.