JavaScript Tutorial

Adding a script to HTML

Ilya Kantor
Contributed: kichinsky

Tweet

Like 17

A script can be put anywhere on the page. The most useful places include:

- Inside the Head tag of the document
- At the bottom of the document, right before closing `BODY`

..But, generally scripts can be put anywhere.

When the browser renders an HTML-page and finds a `<script>` tag - it switches into Javascript mode and executes the code inside. Once executed the browser continues rendering the rest of the page.

Page rendering and `SCRIPT`

The following example demonstrates how the browser switches in and out of JavaScript mode.

```
<html>
<body>
  <h1>Counting rabbits</h1>

  <script>
    for(var i=1; i<=3; i++) {
      alert("Rabbit "+i+" out of the hat!")
    }
  </script>

  <h1>...Finished counting</h1>

</body>
</html>
```

Open the code in new window

Note the order of execution in the example above:

1. When the page starts rendering, only the beginning of the document is shown.
2. Browser meets a script and runs it, executing `alert` 3 times, and then.
3. After the script is complete, the browser returns to HTML, and *only then* the rest of the page is shown.

Moving scripts into `HEAD`

If the HTML may be large, where is the best place to put JavaScript? If you want a script to execute early, before the page is displayed, then the `HEAD` section is a good place.

```
<html>
  <head>
    <script>
      function count_rabbits() {
        for(var i=1; i<=3; i++) {
          alert("Rabbit "+i+" out of the hat!");
        }
      }
    </script>
  </head>

  <body>

  <h2>Press the button to start</h2>

    <input type="button" onclick="count_rabbits()" value="Count
rabbits!"/>

  </body>

</html>
```

Putting scripts into `HEAD` is a common and easy practice, but highly optimized sites use another method.

Scripts at the end of `BODY`

A script can also be at the bottom of page body. In this case it executes after the page is shown.

- Good, because user doesn’t have to wait for scripts.
- Bad, because the functions become available after the HTML is loaded. A user has a chance to click on button which may not work. Usually adding special code that hides functionality until the script has loaded resolves the problem.

By the way, CSS styles must be declared in the HEAD according to the HTML standard. Only scripts are allowed to be placed anywhere.

External scripts

Usually, most JavaScript code is put into an external file, which is attached to HTML, like this:

```
<script src="/path/to/script.js"></script>
```

The `/path/to/script.js` is a relative path. If you have a specific location including the full URL that is the absolute path. Relative paths are relative to your current location on the site.

File `/path/to/script.js` contains JavaScript code, which will execute immediately after browser recieves the file.

This is very handy, because the same file may be used on many pages. If the web-server is configured correctly, the browser will cache the file and will not download it every time.

Here is how it looks like:

```
<html>
  <head>
    <script src="/files/tutorial/browser/script/rabbits.js"></script>
  </head>

  <body>
    <input type="button" onclick="count_rabbits()" value="Count
rabbits!"/>
  </body>

</html>
```

Open the code in new window

Here is the contents of `/files/tutorial/browser/script/rabbits.js`:

```
function count_rabbits() {
    for(var i=1; i<=3; i++) {
        // operator + concatenates strings
        alert("Rabbit "+i+" out of the hat!")
    }
}
```

Open the code in new window

Note, there are no `SCRIPT` tags inside this file. `SCRIPT` tags are used are only in HTML.

External scripts block page rendering in same way as embedded scripts do.

So, if an external script is in `HEAD` then page will not be shown until the script is downloaded and executed.

Closing `</script>` and XHTML

Note that usually one can't use XML-style self-closing tags like `<script src="..."/>` even if DOCTYPE XHTML is specified.

A separate closing `</script>` must be present, replacing it with shorter `<script src="..."/>` doesn't work.

That's if your server uses `Content-Type: text/html` which makes a browser to parse the page in "HTML-mode".

To attach several scripts - use several tags:

```
<script src="/js/script1.js"></script>
<script src="/js/script2.js"></script>
...
```

If `src` attribute is present then tag contents is ignored.

That is, one can't attach external file and execute code in single `<script>`. Two separate tags are needed: first one with `src` for external file, and the second one without `src`, but with the code, which will be executed after that file.

Modern markup for the `<script>` tag.

Nowadays only validators(tools which check pages for correctness in terms of standards) identify incorrect markup. Sometimes people use old ugly code and it works.

Although the correct markup is useful to know. At least you will be able to tell the difference between professional JavaScript and messy javascript written many years ago.

Attribute `<script type=...>`

The older HTML4 standard required this attribute to be set, but HTML5 allows it to be absent. A correct pre-HTML5 value was `type="text/javascript"`.

If you put an unsupported value into `type`, e.g. `<script type="text/html">`, the contents will be ignored. This is a trick used add data that was not rendered to the page. The browser does not execute or show `<script>` with unknown type. Such script is like a div with the `style="display:none"`.

Attribute `<script language=...>`

You may find this attribute in out dated scripts. It is obsolete and don't use it for JavaScript.

Comments before and after scripts

Old manuals and tutorials sometimes recommend to "hide" JavaScript from browsers that don't support it, using HTML-comments: `<!-- ... -->`.

The browser which required such tricks (very old Nescape) is dead for a long time. Other browsers ignore comments. Don't put them, there's no need indeed. – Delete this just state that its no longer required.

Make the "rabbits" example with external script work on your PC.

- Open the file online: rabbits_ext.html.
- Put both HTML and JS on your hard drive and make sure it opens and button works.

Note. If you are opening html from your *filesystem*, not from a site, then "/my/script.js" will be an *absolute path*, from filesystem root.

You also can use *relative paths* like "script.js" for files in same folder with HTML, or "my/script.js" for the file in subfolder.

Open solution

Summary

Scripts can be embedded directly using `SCRIPT` or added as external files by a `SCRIPT` with `src="path/to/file.js"` attribute. An external script file is pure JavaScript.

In either way, HTML page rendering is blocked until the script is downloaded and executed.

That's why an advanced approach is to put the scripts at the bottom of BODY, but in this case a user can start interacting with a page before the Javascript has Loaded.

Putting scripts in `HEAD` is easy and guarantees that they will be available before the page is shown.

‹ Decorators Browser environment ›