عنوان: **معرفی کتابخانه Loader برای بارگذاری JS و CSS**
نویسنده: بهنام محمدی
تاریخ: ۱۳۹۴/۰۴/۲۳ ۱۳:۲۰
آدرس: www.dotnettips.info
گروه‌ها: JavaScript, Library

در طراحی صفحات وب، معمولا از فایل‌های JS و CSS مختلفی استفاده می‌شود؛ از کتابخانه‌ها گرفته تا فایل‌ها اصلی برنامه. به صورت خیلی ساده ما تمام این فایل‌ها را به صفحه‌ی لینک می‌کنیم. اما این روش درست نیست و حجم صفحه و تعداد درخواست‌ها به سرور برای بارگذاری فایل‌ها خیلی بیشتر می‌شود. در زمان اجرای یک صفحه‌ی وب مسلما قسمت‌هایی از صفحه وجود دارند که شاید در شرایط خاصی، کاربر این صفحات را ببیند و یا نیاز باشد تا منطقی، توسط یک فایل JS خاص انجام شود. کتابخانه‌های زیادی برای حل این موضوع درست شده‌اند که راهکار آنها به این صورت است که شما در مواقعی که نیاز به این فایل‌ها دارید، آنها را بارگذاری می‌کنید. کتابخانه‌ی Loader یک فایل JS ساده می‌باشد که توسط اینجانب نوشته شده است و در یک پروژه‌ی بزرگ در حال استفاده است. این کتابخانه تا همین الان که 4 سال از عمر پروژه می‌گذرد در حال کار کردن هست و بدون هیچ مشکلی تا الان جواب داده است. بنابراین تصمیم گرفتم تا این کتابخانه را به صورت عمومی منتشر کنم تا شما هم از این کتابخانه استفاده کنید. در زیر کد Core این کتابخانه و نحوه‌ی استفاده از آن را نوشته‌ام و لینک GitHub هم در زیر می‌باشد.

نحوه‌ی استفاده از این کتابخانه بعد از اینکه فایل JS آن‌را به صفحه وصل کردیم، به صورت زیر است که می‌توانید بر حسب نیاز، این تابع را صدا بزنید. کد زیر نحوه‌ی استفاده از این کتابخانه هست. فرض کنید در شرایطی نیاز داریم تا کتابخانه‌ی JSTree را بارگذری کنیم. به جای اینکه از اول فایل‌های JS و CSS آن‌را در صفحه داشته باشیم، خیلی ساده از تابع زیر استفاده می‌کنیم. در این کتابخانه تابع Promise وقتی Fire می‌شود که تمام فایل‌هایی که به صورت پارامتر در تابع Load مشخص شده‌اند، بارگذاری شوند.

```
loader.load([
  'plugin/dropdowntree/css/style.min.css',
  'plugin/dropdowntree/js/jstree.js',
  'plugin/dropdowntree/js/jstree.checkbox.js',
]).promise(function () {
  // run this code promise
});
```

کد هسته‌ی اصلی کتابخانه Loader به صورت زیر هست:

```
/*
    loader version 0.2.1 2015
    loader design by Behnam Mohammadi (http://itten.ir)
*/
window.loader = {
    load: function (urls) {
        var loadCounter = 0;
        var promise = null;
        var ext = '';
        this.promise = function (fun) {
            promise = fun;
        };
        for (var i = 0; i < urls.length; i++) {
            ext = urls[i].substring(urls[i].length - 3);
            if (ext == '.js') {
                var script = document.createElement('script');
                script.src = urls[i];
                script.onload = function () {
                    loadCounter += 1;
                    if (loadCounter == urls.length) {
                        promise();
                    }
                };
                document.body.appendChild(script);
            } else if (ext == 'css') {
                var link = document.createElement('link');
                link.href = urls[i];
                link.rel = 'stylesheet';
                link.type = 'text\css';
                link.onload = function () {
                    loadCounter += 1;
                    if (loadCounter == urls.length) {
                        promise();
                    }
```

```
                };
                document.body.appendChild(link);
            }
        }
        return this;
    }
}
```

حجم این کتابخانه در صورت فشرده سازی کمتر از نیم کیلوبایت هست. لینک GitHub این پروژه را در زیر مشاهده میکنید.

https://github.com/uxitten/loader

نظرات خوانندگان

نویسنده: محسن خان
تاریخ: ۱۳۹۴/۰۴/۲۳ ۲۲:۱

ممنون از شما. این روش احتمالا با روش‌های یکی کردن اسکریپت‌ها یا فایل‌های CSS با هم قابل استفاده نیست (کاهش رفت و برگشت به سرور با یکی کردن چند فایل CSS یا اسکریپت با هم مثل روش‌های bundling & minification در ASP.NET MVC).

نویسنده: بهنام محمدی
تاریخ: ۱۳۹۴/۰۴/۲۴ ۸:۴۸

با سلام ممنون. این کتابخانه وظیفه بارگذاری فایل‌ها به صورت غیر همزمان را دارد و بیشترین کاربرد آن در پروژه‌های SPA هست.