

# Melhorando o carregamento do Javascript

Antes de começar, se ainda não tem conhecimento de como medir a velocidade do seu site, o seguinte artigo pode ser de ajuda:

## [Medindo a velocidade do seu site](#)

Melhorar o carregamento das imagens vai aumentar muito a velocidade, mas o Javascript e CSS são duas coisas que podem bloquear a visualização do conteúdo que fica embaixo do chamado destes arquivos. Neste artigo vou falar sobre algumas coisas que podem ser feitas com o Javascript.

A maneira “clássica” de construir a o HTML sempre foi a seguinte:

Content

Dependendo da importância e requisitos do JS, ele podia ser carregado antes do conteúdo ou embaixo antes de fechar o body.

No exemplo anterior, o browser vai começar a ler o arquivo HTML, primeiro vai carregar o CSS, uma vez feito isso, ele procura o JS e o carrega (até agora nada do HTML foi carregado e o usuário ainda fica esperando numa tela em branco) e por fim o resto do documento é carregado.

Talvez no código anterior não seja tão grave, mas tem que pensar o que pode acontecer se tem muito CSS (de você e de um framework como pode ser Bootstrap), livrarias de JS (Jquery), plugins de JS (um mundo de possibilidades... para fazer o site mais devagar). Aqui tudo começa a ficar mais devagar, acima de tudo considerando que o navegador pode levar um tempo em ler o Jav

ascript que já foi carregado.

É por isso que o primeiro que te tem que fazer é pensar em **reduzir ao mínimo o JS**, isso não quer dizer que não pode usá-lo, mas tem que ter a vassoura pronta para varrer tudo o que não seja necessário (e você poder resolver tudo, com esforço mas uma grande recompensa).

Por mais tentador que seja, **pense duas vezes antes de adicionar um novo plugin, app externo ou framework**, todos são granitos que podem afetar o carregamento do site. Recomendo usar [Bundl ePhobia](#) para testar o peso e possível tempo de carregamento do plugin a adicionar.

Meu conselho é que se precisa de plugins, procure um que possa atacar todas as necessidades agora e no futuro; e se for possível, procure no Github se o mesmo está ativo e se tem manutenção, algum commit recente ou seção de ajuda.

Tem que pensar se é realmente necessário ter carregado o Bootstrap JS quando talvez só este fazendo uso dos pop-ups, algo que tranquilamente pode ser resolvido com jQuery sozinho.

Se não tem alternativa pelo menos tem que pensar em carregar todo o JS que não seja crítico nos primeiros segundos do site, como assíncrono, quer dizer que o HTML não ficará bloqueado pela carga deste JS e o navegador irá carregá-lo quando possa.

Por outro lado também tem que pensar quais plugins são críticos e quais não são, pensando os críticos como os que provavelmente tenham que depender de jQuery (para poder carregar jQuery como assíncrono), como por exemplo acontece com Lazy Sizes.

## Jquery assíncrono

Para carregar jQuery de maneira assíncrona simplesmente tem que adicionar o atributo `async="true"` no chamado no tag

## Plugins críticos e não críticos

Como falei anteriormente, é importante dividir os plugins em críticos e não críticos para carregar os mais importantes primeiro e porque pode acontecer que os não críticos tenham dependências com jQuery .

Um exemplo dum plugin crítico e lazy sizes (lazy load), precisado para ir exibindo as imagens principais, enquanto um plugin não crítico pode ser bxSlider ou o JS de Bootstrap. O primeiro porque depende de jQuery e o segundo porque talvez não seja crítico que os pop-ups estejam disponíveis nos primeiros segundos da carga da loja.

Em nossas lojas carregamos os plugins antes de fechar o `body` usando [Twig](#) para que sejam inseridos como código inline e evitar fazer o pedido dum arquivo mais ao navegador.

Dividimos os plugins nos arquivos:

- **external-no-dependencies.js** : Contem plugins críticos, quanto menos melhor.
- **external.js.tpl** : Carrega o resto dos plugins, carga el resto de los plugins, englobados pela função `LS.ready.then`

La forma en la que los cargamos es:

```
{{ "js/external-no-dependencies.js" | static_url | script_tag }}
```

Fazer uso de plugins assíncronos quer dizer que podem carregar qualquer coisa que seja "async"? **Não!** Por mais que tudo fique assíncrono, vai afetar as pontuações de velocidade, sobretudo o "Tempo até interatividade", que é o momento no qual o navegador diz "ok, o documento fica pronto".

Por mais que tenham JS que não bloqueie o carregamento, tem que usá-lo com consciência.

Que segue?

Se já otimizou tudo o que tem a ver com JS, recomendo ler as outras mudanças aplicadas para melhorar a velocidade:

[Melhorando o carregamento das imagens e SVGs](#)

[Melhorando o carregamento do CSS](#)