聊聊图片资源的渐进式加载及懒加载

现状

- 图片资源对网页很重要，小到 icon，中到商品介绍图，大到整个页面就是一张图
- 图片提升用户体验不说，甚至减少了前端开发的工作量（一个页面用几张图拼起来）
- 图片资源往往比较大，动辄几十 kb 甚至上百 kb，所以在用之前需要先进行压缩
- 设计师给图前先做了压缩的话还好，如果设计师没做，就得自己去在线压缩，但其实压缩后，图片的体积还是很大（相比较各种三方依赖）
- 图片资源多的页面，用户等待页面加载完成的时间也会变长，同时又会涉及到用户的网络状况，结果是，或多或少都会影响用户体验

方案

那针对图片资源网站，如何提升用户体验呢?

1. 缩小图片文件体积，使用 webp 格式
2. 针对 icon 图片，使用 image sprites，或用 SVG 或 图标字体替换图片
3. 提供图片占位符
4. 延迟加载图片
5. ...

下面我们针对3、4两点做优化的实践。

实践

我准备了一个 **今日必应** 的页面，里面会展示 6.28 到 7.4 每天的必应美图，由于是 bing.com 里的背景图，所以每张图片的大小基本都在 300kb 以上。
（注意：需要打开 chrome 的开发者工具，禁用浏览器缓存，不然无法看到效果， 可选的把网速限制为 *Fast 3G* ）

原始版本

首先，先体验一下原始版本。图片加载很慢，但最直观的感受是图片加载完成后页面会抖动一下，体验很差。

图片占位符

页面抖动是因为图片在加载之前高度是 0，加载完成后，导致下方元素位置发生变化，产生抖动，简单的为图片预留一个空间就能解决，查看效果。

现在每张图片在加载之前都默认会有一片空白区域，我们可以对这块区域加以利用，比如说：

- 给占位符填充颜色，查看效果
- 给占位符增加 loading 动画，查看效果
- **图片渐进式加载**
- …

目前为止，我见过最优雅的加载方式，是 Medium 里文章头图的加载，可以预览一下效果。

因为宽度是撑满视口的，所以这类图片的尺寸都很大。如果什么都不做，对于页面性能的影响是灾难级的，但是这里的图片做了渐进式加载后，加载这种大图不但没有让用户体验减分，甚至是大大加分的。

这种渐进式加载的效果，看起来很棒，而且实现的原理很简单，先看看实现后的效果，查看效果，再看看实现的代码。

图片懒加载

现在我们页面中的图片，已经有一个比较令人愉悦的加载效果了，但如果页面中有上百张图片，但我们打开页面时默认只能看到 10 张，然后就关闭了页面，那剩下的 90 张图片依旧会被浏览器加载，既会影响我们想看的 10 张图片的加载速度，又会产生不必要的流量消耗，所以就需要延迟加载 **处于视口之外的图片**。

那如何判断图片是否进入了视口呢？如果之前大家有了解过，或实现过图片懒加载，就会知道，可以监听窗口的 `scroll` 和 `resize` 事件，再用 `getBoundingClientRect` 方法来判断图片进入视口。这种方法浏览器的兼容性比较好，但很容易产生性能问题，好在现在有了新的 Intersection Observer API 能帮我们更轻松的来做判断，也推荐大家读一篇介绍 **Intersection Observer API** 的文章。

先看一下实现图片懒加载后的效果，查看效果。实现的代码也不复杂，对于支持 **Intersection Observer API** 的浏览器，我们使用该 API 提供的方法判断图片是否进入视口，对于不支持的浏览器，我们用兼容性更好的事件监听方法来实现。只有当图片进入视口后，再做渐进式的加载，而不是像之前的那样，在 `DOMContentLoaded` 事件被触发后，就去加载所有图片。再看下实现的代码。