# 代理模式

代理模式是当调用方不方便直接访问一个对象或者不满足需要的时候，提供一个替身 对象来控制对这个对象的访问。 调用方实际上访问的是替身对象。替身对象对请求做出一些处理之 后，再把请求转交给本体对象。

proxy-1

proxy-1

## 应用场景

比如我们平时用的代理软件就是代理模式。

比如我们做图片懒加载就可以用到代理模式。

## 实现

我们以图片懒加载为例，实现一下代理模式。

如下代码只会在id为img1的元素出现在视口才会加载图片。

proxy-3

proxy-3

如果没有出现在视口，是不会加载的：

proxy-2

proxy-2

function lazyloadImg(node, src) {  
 var intersectionObserver = new IntersectionObserver(function(entries) {  
 if (entries[0].intersectionRatio > 0) return node.src = src;  
 });  
 // start observing  
 intersectionObserver.observe(node);  
}  
  
lazyloadImg(document.querySelector('#img1'), 'http://lucifer.ren/imgs/a.png')

如果不适用代理模式的话，代码可能是：

// 在某个时候  
document.querySelector('#img1').src = 'http://lucifer.ren/imgs/a.png';

另外我们设置可以在代理中增加一些逻辑判断，比如图片如果很小就直接发，如果比较大就等会。 或者预先展示一个占位图片，等到图片加载之后再替换。

我们可以发现，通过使用代理模式，我们的代码很容易松耦合，帮助我们写出更加内聚的代码。