3种爬取网站的常见方法：

1、爬取网站地图

2、遍历每个网页的数据库ID

3、跟踪网页链接

下载网页download.py：

|  |
| --- |
| def download(url):  print('Downloading', url)  try:  html = urllib.request.urlopen(url).read()  except urllib.request.URLError as e:  print('Download error:', e.reason)  html = None  return html |

下载网页的HTTP错误大家都不陌生，对于HTTP错误，4xx错误发生在请求存在问题时，而5xx错误则发生在服务端存在问题时。所以，我们只需要确保download函数在发生Sxx错误时重试下载即可。下载页面re\_download.py（重新下载功能）：

|  |
| --- |
| def download(url, num\_retries= 2): # num\_retries是重新下载次数  print('Downloading', url)  try:  html = urllib.request.urlopen(url).read()  except urllib.request.URLError as e:  print('Download error:', e.reason)  html = None  if num\_retries > 0:  if hasattr(e, 'code') and 500 <= e.code < 600:  # recursively retry 5xx HTTP errors  return download(url , num\_retries - 1)  return html |

为了下载更可靠，需要控制用户代理的设定。下载页面user\_agent\_download.py（用户代理）：

|  |
| --- |
| def download(url, user\_agent='wswp', num\_retries=2):  print('Downloading', url)  headers = {'User-agent': user\_agent}  req = urllib.request.Request(url, headers=headers) # 使用Request来包装请求  try:  html = urllib.request.urlopen(req).read() # 使用urlopen来获取页面  except urllib.request.URLError as e:  print('Download error:', e.reason)  html = None  if num\_retries > 0:  if hasattr(e, 'code') and 500 <= e.code < 600:  # recursively retry 5xx HTTP errors  return download(url , num\_retries - 1)  return html |

至此，一个灵活的下载函数，可以在开发中得到复用。该函数能够捕获异常、重试下载并设置用户代理。