接口项目实战

- 项目地址：http://api.lemonban.com/futureloan
- **接口分类**
  - **硬件接口：指的是两个硬件设备之间的连接方式（比如：鼠标和电脑通过USB接口进行连接）**
  - **软件接口：简单来说就是软件程序之间数据交互的通道。**
- **软件接口分类：**
  - **程序内部接口：是客户端与服务器的接口，用来实现客户端和服务器间的数据传递**
  - **外部接口：外部接口常见的典型例子就是通过第三方登录、第三方支付等，通过调用第三方接口并返回当前的系统**
- **常见的接口协议**
  - **webService接口：使用soap协议通过http传输，请求报文和返回报文都是xml格式的。常用的测试工具有:soapUI**
  - **http协议接口：目前使用最广泛的，使用HTTP协议来传输数据，常见的请求方法，有get、post等，常见的测试工具postman,jmeter**
- **什么是接口测试?**
  - **接口测试是测试系统组件间接口的一种测试，接口测试主要用于检测外部系统与系统之间以及内部各个子系统之间的交互点，测试的重点是要检查数据的交换、传递、和控制管理过程，以及系统间的相互逻辑依赖关系等（-----百度百科解释）**
  - **接口测试：本质是基于某种协议，发送一个Request请求给服务器，然后服务器返回一个Response响应数据，然后对响应数据进行分析，判断是否与我们预期的返回一致，从而验证功能是否正确，这就是接口测试。**

- **问题：那么客户端是如何向服务器发送请求?**

一、HTTP协议解读

HTTP协议:超文本传输协议（HyperText Transfer Protocol)是互联网上应用最为广泛的一种网络协议。所有的HTML文件都必须遵守这个标准。设计HTTP最初的目的是为了提供一种发布和接收HTML页面的方法。

HTTPS协议（Hypertext Transfer Protocol over Secure Socket Layer）简单讲是HTTP的安全版，在HTTP下加入SSL层。

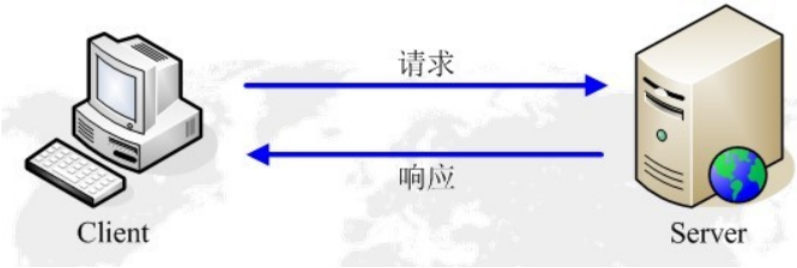
`SSL`（Secure Sockets Layer 安全套接层）主要用于Web的安全传输协议，在传输层对网络连接进行加密，保障在Internet上数据传输的安全。

- `HTTP`的端口号为 80，
- `HTTPS`的端口号为 443

1、HTTP请求的过程：

- 客户端：PC端应用程序 浏览器 APP 小程序 ， 爬虫（代码）

HTTP通信由两部分组成： **客户端请求消息** 与 **服务器响应消息**



1. 当用户在浏览器的地址栏中输入一个URL并按回车键之后，浏览器会向HTTP服务器发送HTTP请求。HTTP请求主要分为“Get”和“Post”两种方法。
2. 当我们在浏览器输入URL http://www.baidu.com 的时候，浏览器发送一个Request请求去获取 http://www.baidu.com 的html文件，服务器把Response文件对象发送回给浏览器。
3. 浏览器分析Response中的 HTML，发现其中引用了很多其他文件，比如Images文件，CSS文件，JS文件。 浏览器会自动再次发送Request去获取图片，CSS文件，或者JS文件。
4. 当所有的文件都下载成功后，网页会根据HTML语法结构，完整的显示出来了。

URL（Uniform / Universal Resource Locator的缩写）：统一资源定位符，是用于完整地描述Internet上网页和其他资源的地址的一种标识方法。

基本格式：`scheme://host[:port#]/path/…/[?query-string][#anchor]`

- scheme：协议(例如：http, https, ftp)
- host：服务器的IP地址或者域名
- port#：服务器的端口（如果是走协议默认端口，缺省端口80）
- path：访问资源的路径
- query-string：参数，发送给http服务器的数据
- anchor：锚（跳转到网页的指定锚点位置）

2、HTTP请求信息

在浏览器中输入URL地址，访问某个网站，发送一个HTTP请求到服务器的请求消息，包括以下格式：

```
请求行  请求头部  空行  请求数据
```

```
GET https://www.baidu.com/ HTTP/1.1
Host: www.baidu.com
Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
like Gecko) Chrome/54.0.2840.99 Safari/537.36
Accept:text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.
8
Referer: http://www.baidu.com/
Accept-Encoding: gzip, deflate, sdch, br
Accept-Language: zh-CN,zh;q=0.8,en;q=0.6
Cookie:BAIDUID=04E4001F34EA74AD4601512DD3C41A7B:FG=1;BIDUPSID=04E4001F34EA74AD46
01512DD3C41A7B;PSTM=1470329258;MCITY=-343%3A340%3A;BDUSS=nF0MVFiMTVLcUhQ2MxQ0M3S
TZGQUZ4N2hBa1FFRkIzUDI3QlBCZjg5cFdOd1pZQVFBQUFBJCQAADpLvgG0KGyvLrcyfrGAFaq3ldWqt
5XN;H_PS_PSSID=1447_18240_21105_21386_21454_21409_21554; BD_UPN=12314753;
sug=3;sugstore=0;ORIGIN=0;bdime=0;
H_PS_645EC=7e2ad3QHl181NSPbFbd7PRUCE1LlufzxrcFmwYin0E6b%2BW8bbTMKHZbDP0g;
BDSVRTM=0
```

- **1.Host (主机和端口号)**

  Host：对应网址URL中的Web名称和端口号（域名）

- **2. Connection (链接类型)**

  Connection：表示客户端与服务连接类型:keep-alive表示长连接（HTTP/1.1使用 keep-alive 为默认值。）

- **3. Upgrade-Insecure-Requests (升级为HTTPS请求)**

  Upgrade-Insecure-Requests：升级不安全的请求，意思是会在加载 http 资源时自动替换成 https 请求，让浏览器不再显示https页面中的http请求警报。

- **4、User-Agent (浏览器名称)**

  User-Agent：是客户浏览器的名称。

- **5、Accept (传输文件类型)**

  Accept：指浏览器或其他客户端可以接受的文件类型，服务器可以根据它判断并返回适当的文件格式。

  `Accept: text/html, application/xhtml+xml;q=0.9, image/*;q=0.8`：表示浏览器支持的 MIME 类型分别是 html文本、xhtml和xml文档、所有的图像格式资源。

  - **Text**：文本信息，可以是多种字符集和或者多种格式的；
  - **Application**：用于传输应用程序数据或者二进制数据。
  - **q是权重系数**，范围 0 =< q <= 1，q 值越大，越靠前。默认为1，按从左到右排序顺序；若赋值为0，表示浏览器不接受此类型。

- **6. Referer (页面跳转处)**

  Referer：表明产生请求的网页来自于哪个URL，用户是从该 Referer页面访问到当前请求的页面。这个属性可以用来跟踪Web请求来自哪个页面，是从什么网站来的等。

- **7. Accept-Encoding（文件编解码格式）**

  Accept-Encoding：指出浏览器可以接受的编码方式。编码方式不同于文件格式，它是为了压缩文件并加速文件传递速度。浏览器在接收到Web响应之后先解码，然后再检查文件格式，许多情形下这可以减少大量的下载时间。

- **8. Accept-Language（语言种类）**

  Accept-Langeuage：指出浏览器可以接受的语言种类，如en或en-us指英语，zh或者zh-cn指中文，当服务器能够提供一种以上的语言版本时要用到。

- **9、Cache-Control:max-age=0 明确表示不会缓存服务器资源、**
- **10、Accept-Charset（字符编码）**

  Accept-Charset：指出浏览器可以接受的字符编码。

  如果在请求消息中没有设置这个域，缺省是任何字符集都可以接受。
- **11. Cookie （Cookie）**

  Cookie：浏览器用这个属性向服务器发送Cookie。Cookie是在浏览器中寄存的小型数据体，它可以记载和服务器相关的用户信息，也可以用来实现会话功能，以后会详细讲。
- **12. Content-Type (POST数据类型)**

  Content-Type：POST请求里用来表示的内容类型

3、HTTP响应信息

服务器接收到请求后，返回的HTTP响应也由四个部分组成，分别是：

```
状态行   消息报头   空行   响应正文
```

```
HTTP/1.1 200 OK
Access-Control-Allow-Credentials: true
Access-Control-Allow-Origin: https://m.baidu.com,https:
Connection: keep-alive
Content-Encoding: gzip
Content-Type: text/html
Date: Tue, 12 Mar 2019 07:58:18 GMT
Search_result: OK
Server: Apache
Tracecode: 34988465750563182602031215
Tracecode: 34988458200827947786031215
Vary: Accept-Encoding
Transfer-Encoding: chunked
```

1，Cache-Control：must-revalidate, no-cache, private。

这个值告诉客户端，服务端不希望客户端缓存资源，在下次请求资源时，必须要从新请求服务器，不能从缓存副本中获取资源。

- Cache-Control是响应头中很重要的信息，当客户端请求头中包含Cache-Control:max-age=0请求，明确表示不会缓存服务器资源时,Cache-Control作为作为回应信息，通常会返回no-cache，意思就是说，"那就不缓存呗"。
- 当客户端在请求头中没有包含Cache-Control时，服务端往往会定,不同的资源不同的缓存策略，比如说oschina在缓存图片资源的策略就是Cache-Control：max-age=86400,这个意思是，从当前时间开始，在86400秒的时间内，客户端可以直接从缓存副本中读取资源，而不需要向服务器请求。

2. Connection：keep-alive

这个字段作为回应客户端的Connection：keep-alive，告诉客户端服务器的tcp连接也是一个长连接，客户端可以继续使用这个tcp连接发送http请求。

3. Content-Encoding:gzip

告诉客户端，服务端发送的资源是采用gzip编码的，客户端看到这个信息后，应该采用gzip对资源进行解码。

4. Content-Type：text/html;charset=UTF-8

告诉客户端，资源文件的类型，还有字符编码，客户端通过utf-8对资源进行解码，然后对资源进行html解析。通常我们会看到有些网站是乱码的，往往就是服务器端没有返回正确的编码。

5. Date：Sun, 21 Sep 2016 06:18:21 GMT

这个是服务端发送资源时的服务器时间，GMT是格林尼治所在地的标准时间。http协议中发送的时间都是GMT的，这主要是解决在互联网上，不同时区在相互请求资源的时候，时间混乱问题。

6. Expires:Sun, 1 Jan 2000 01:00:00 GMT

这个响应头也是跟缓存有关的，告诉客户端在这个时间前，可以直接访问缓存副本，很显然这个值会存在问题，因为客户端和服务器的时间不一定会都是相同的，如果时间不同就会导致问题。所以这个响应头是没有Cache-Control：max-age=*这个响应头准确的，因为max-age=date中的date是个相对时间，不仅更好理解，也更准确。

7. Pragma:no-cache

这个含义与Cache-Control等同。

8.Server：Tengine/1.4.6

这个是服务器和相对应的版本，只是告诉客户端服务器的信息。

9. Transfer-Encoding：chunked

这个响应头告诉客户端，服务器发送的资源的方式是分块发送的。一般分块发送的资源都是服务器动态生成的，在发送时还不知道发送资源的大小，所以采用分块发送，每一块都是独立的，独立的块都能标示自己的长度，最后一块是0长度的，当客户端读到这个0长度的块时，就可以确定资源已经传输完了。

10. Vary: Accept-Encoding

告诉缓存服务器，缓存压缩文件和非压缩文件两个版本，现在这个字段用处并不大，因为现在的浏览器都是支持压缩的。

11. Cookie：通过在 客户端 记录的信息确定用户的身份。

12. Session：通过在 服务器端 记录的信息确定用户的身份。

4、HTTP响应状态码

```
# 1xx:信息
100 Continue
服务器仅接收到部分请求，但是一旦服务器并没有拒绝该请求，客户端应该继续发送其余的请求。
101 Switching Protocols
服务器转换协议：服务器将遵从客户的请求转换到另外一种协议。

# 2xx:成功
200 OK
请求成功（其后是对GET和POST请求的应答文档）
201 Created
请求被创建完成，同时新的资源被创建。
202 Accepted
供处理的请求已被接受，但是处理未完成。
203 Non-authoritative Information
文档已经正常地返回，但一些应答头可能不正确，因为使用的是文档的拷贝。
204 No Content
没有新文档。浏览器应该继续显示原来的文档。如果用户定期地刷新页面，而Servlet可以确定用户文档足够新，这个状态代码是很有用的。
205 Reset Content
```

没有新文档。但浏览器应该重置它所显示的内容。用来强制浏览器清除表单输入内容。
206 Partial Content
客户发送了一个带有Range头的GET请求，服务器完成了它。

3xx:重定向
300 Multiple Choices
多重选择。链接列表。用户可以选择某链接到达目的地。最多允许五个地址。
301 Moved Permanently
所请求的页面已经转移至新的url。
302 Moved Temporarily
所请求的页面已经临时转移至新的url。
303 See Other
所请求的页面可在别的url下被找到。
304 Not Modified
未按预期修改文档。客户端有缓冲的文档并发出了一个条件性的请求（一般是提供If-Modified-Since头表示客户只想比指定日期更新的文档）。服务器告诉客户，原来缓冲的文档还可以继续使用。
305 Use Proxy
客户请求的文档应该通过Location头所指明的代理服务器提取。
306 Unused
此代码被用于前一版本。目前已不再使用，但是代码依然被保留。
307 Temporary Redirect
被请求的页面已经临时移至新的url。

4xx:客户端错误
400 Bad Request
服务器未能理解请求。
401 Unauthorized
被请求的页面需要用户名和密码。
401.1
登录失败。
401.2
服务器配置导致登录失败。
401.3
由于 ACL 对资源的限制而未获得授权。
401.4
筛选器授权失败。
401.5
ISAPI/CGI 应用程序授权失败。
401.7
访问被 Web 服务器上的 URL 授权策略拒绝。这个错误代码为 IIS 6.0 所专用。
402 Payment Required
此代码尚无法使用。
403 Forbidden
对被请求页面的访问被禁止。
403.1
执行访问被禁止。
403.2
读访问被禁止。
403.3
写访问被禁止。
403.4
要求 SSL。
403.5
要求 SSL 128。
403.6
IP 地址被拒绝。
403.7
要求客户端证书。

403.8
站点访问被拒绝。
403.9
用户数过多。
403.10
配置无效。
403.11
密码更改。
403.12
拒绝访问映射表。
403.13
客户端证书被吊销。
403.14
拒绝目录列表。
403.15
超出客户端访问许可。
403.16
客户端证书不受信任或无效。
403.17
客户端证书已过期或尚未生效。
403.18
在当前的应用程序池中不能执行所请求的 URL。这个错误代码为 IIS 6.0 所专用。
403.19
不能为这个应用程序池中的客户端执行 CGI。这个错误代码为 IIS 6.0 所专用。
403.20
Passport 登录失败。这个错误代码为 IIS 6.0 所专用。
404 Not Found
服务器无法找到被请求的页面。
404.0
没有找到文件或目录。
404.1
无法在所请求的端口上访问 Web 站点。
404.2
Web 服务扩展锁定策略阻止本请求。
404.3
MIME 映射策略阻止本请求。
405 Method Not Allowed
请求中指定的方法不被允许。
406 Not Acceptable
服务器生成的响应无法被客户端所接受。
407 Proxy Authentication Required
用户必须首先使用代理服务器进行验证，这样请求才会被处理。
408 Request Timeout
请求超出了服务器的等待时间。
409 Conflict
由于冲突，请求无法被完成。
410 Gone
被请求的页面不可用。
411 Length Required
"Content-Length" 未被定义。如果无此内容，服务器不会接受请求。
412 Precondition Failed
请求中的前提条件被服务器评估为失败。
413 Request Entity Too Large
由于所请求的实体的太大，服务器不会接受请求。
414 Request-url Too Long
由于url太长，服务器不会接受请求。当post请求被转换为带有很长的查询信息的get请求时，就会发生这种情况。
415 Unsupported Media Type

由于媒介类型不被支持，服务器不会接受请求。
416 Requested Range Not Satisfiable
服务器不能满足客户在请求中指定的Range头。
417 Expectation Failed
执行失败。
423
锁定的错误。

5xx:服务器错误
500 Internal Server Error
请求未完成。服务器遇到不可预知的情况。
500.12
应用程序正忙于在 Web 服务器上重新启动。
500.13
Web 服务器太忙。
500.15
不允许直接请求 Global.asa。
500.16
UNC 授权凭据不正确。这个错误代码为 IIS 6.0 所专用。
500.18
URL 授权存储不能打开。这个错误代码为 IIS 6.0 所专用。
500.100
内部 ASP 错误。
501 Not Implemented
请求未完成。服务器不支持所请求的功能。
502 Bad Gateway
请求未完成。服务器从上游服务器收到一个无效的响应。
502.1
CGI 应用程序超时。 ·
502.2
CGI 应用程序出错。
503 Service Unavailable
请求未完成。服务器临时过载或当机。
504 Gateway Timeout
网关超时。
505 HTTP Version Not Supported
服务器不支持请求中指明的HTTP协议版本

5、HTTP请求方法

根据HTTP标准，HTTP请求可以使用多种请求方法。

序号	方法	描述
1	GET	用于获取资源（没有请求体）
2	POST	向指定资源提交数据进行处理请求（例如提交表单或者上传文件），数据被包含在请求体中。POST请求可能会导致新的资源的建立和/或已有资源的修改
3	PATCH	用于更新服务器的数据（局部更新）
4	DELETE	用于服务器删除指定的数据
5	PUT	用于更新服务器的数据（数据整体更新）

HTTP请求常用的`Get`和`Post`两种方法

- GET是从服务器上获取数据，POST是向服务器传送数据
- GET请求参数显示，都显示在浏览器网址上，HTTP服务器根据该请求所包含URL中的参数来产生响应内容，即“Get”请求的参数是URL的一部分。 例如： `http://www.baidu.com/s?wd=Chinese&tt=9999`
- POST请求参数在请求体当中，消息长度没有限制而且以隐式的方式进行发送，通常用来向HTTP服务器提交量比较大的数据（比如请求中包含许多参数或者文件上传操作等），请求的参数包含在“Content-Type”消息头里，指明该消息体的媒体类型和编码，

二、鉴权、授权

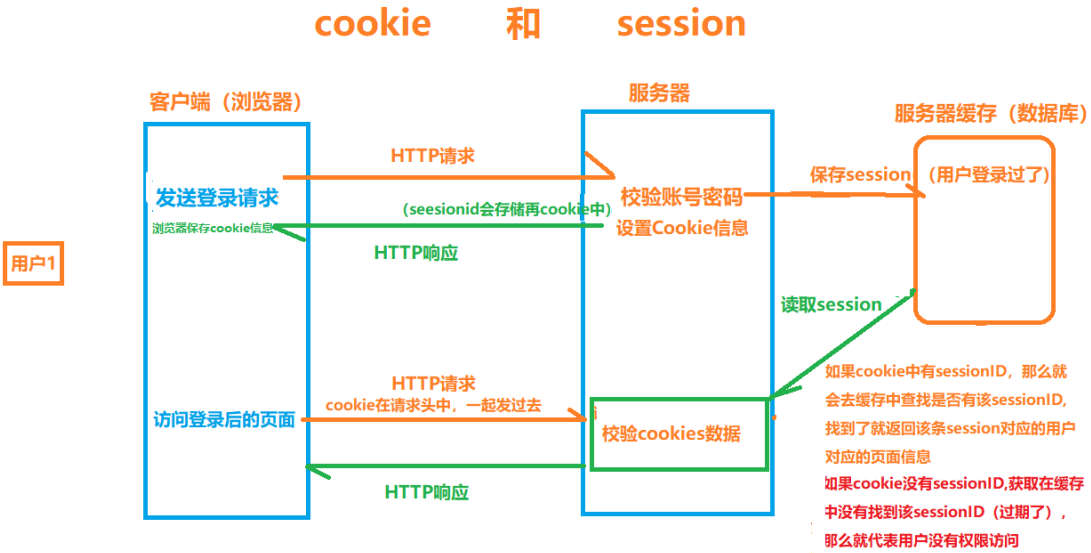
- **问题：浏览器登录了某些网站（腾讯视频），下次访问为什么不要登录?**

两句话理解鉴权授权：

授权：相当于给一个通行证，

鉴权：鉴定是否有权限访问（判断有没有通行证）

1、Cookies 和 session



- **cookie**

- Cookie 是在 HTTP 协议下，服务器或脚本可以维护用户信息的一种方式。Cookie 是由 Web 服务器保存在用户浏览器（客户端）上的小文本文件，它可以包含有关用户的信息。无论何时用户访问到服务器，都会带上该服务器的cookie信息。
- 一般 Cookie 都是有效期的， Cookie 只在浏览器上保存一段规定的时间，一旦超过规定的时间，该 Cookie 就会被系统清除。

- **Session**
  - Session将数据存储在服务器中，服务器会为每一个用户创建一条session，用户访问服务器的时候需要拿着sessionid去表明自己的身份。
  - Session的实现是基于Cookie, Session需要借助于Cookie来存储sessionID。
- **cookie和session都是开发设置的，不要去想怎么设置，跟你没关系！！！**

2、token