

Socket使用

socket使用的库函数

1.创建套接字

```
1 Socket (af,type,protocol) //建立地址和套接字的联系
2 bind(sockid, local addr, addrlen) //服务器端侦听客户端的请求
3 listen( Sockid ,quenlen) //建立服务器/客户端的连接 (面向连接TCP)
```

2.客户端请求连接

```
1 Connect(sockid, destaddr, addrlen) //服务器端等待从编号为Sockid的Socket上接
2 newsockid=accept(Sockid, Clientaddr, paddrlen) //发送/接收数据
```

3.面向连接：

```
1 send(sockid, buff, buflen)
2 recv()
```

4.面向无连接：

```
1 sendto(sockid,buff,...,addrlen)
2 recvfrom()
```

5.释放套接字

```
1 close(socked)
```

在iOS中以NSStream(流)来发送和接收数据,可以设置流的代理，对流状态的变化做出相应的动作(连接建立，接收到数据，连接关闭)。

- NSStream：数据流的父类，用于定义抽象特性，例如：打开、关闭代理，NSStream继承自CFStream(CoreFoundation)
- NSInputStream：NSStream的子类，用于读取输入
- NSOutputStream：NSStream的子类，用于写输出。

服务端先不提， 客户端代码大概如下：

```
1 //需要导入<arpa inet.h="">, <netdb.h>
2 - (void)test
```

很不错，原型阶段可以使用Mockplus，非常快速的画出自己的想
cindy2 评论了 从idea到原型构建一个小程序...

期待国产Mockplus越来越好。

cindy2 评论了 优质产品需求文档（PRD）三大原则...

有没有demo?

nikolxm 评论了 Runloop优化列表滑动卡顿...

原来搞过一个这样的功能，也是在tableView上添加倒计时，而且是不同的无为刀法 评论了 多个cell中展示倒计时，本地和服务器时间差异解决...

正打算做输入身份证那个自定义键盘呢。太及时了。小码哥底层班的视频可XXcc_学无止境 评论了 深入讲解iOS键盘三：自定义键盘的两种方法...

难得的干货!

蜗壳后仰美如画 评论了 iOS Push的前世今生

相关帖子

听说现在代码混淆后，审核过不了？

实战方案：2018年大型企业、公共机构如何IT高效运维的

请问如何读取pod Assets中的字体文件？

tableview的表头的高度如何自适应

cocos 鼠标垫用坏了

求救，登录开发者账号时提示Need assistance with accessing your developer account?

离线地图

丐帮app是什么软件

vps手机端开发需要什么知识

听说现在代码混淆后，审核过不了？

实战方案：2018年大型企业、公共机构如何IT高效运维的

请问如何读取pod Assets中的字体文件？

tableview的表头的高度如何自适应

cocos 鼠标垫用坏了

求救，登录开发者账号时提示Need assistance with accessing your developer account?

离线地图

丐帮app是什么软件

vps手机端开发需要什么知识

```
3 {
4     NSString * host=@"123.33.33.1";
5     NSNumber * port = @1233;
6     // 创建 socket
7     int socketFileDescriptor = socket(AF_INET, SOCK_STREAM, 0);
8     if (-1 == socketFileDescriptor) {
9         NSLog(@"创建失败");
10        return;
11    }
12    // 获取 IP 地址
13    struct hostent * remoteHostEnt = gethostbyname([host UTF8String])
14    if (NULL == remoteHostEnt) {
15        close(socketFileDescriptor);
16        NSLog(@"%@",@"无法解析服务器的主机名");
17        return;
18    }
19    struct in_addr * remoteInAddr = (struct in_addr *)remoteHostEnt->
20    // 设置 socket 参数
21    struct sockaddr_in socketParameters;
22    socketParameters.sin_family = AF_INET;
23    socketParameters.sin_addr = *remoteInAddr;
24    socketParameters.sin_port = htons([port intValue]);
25    // 连接 socket
26    int ret = connect(socketFileDescriptor, (struct sockaddr *) &socketParameters, sizeof(socketParameters));
27    if (-1 == ret) {
28        close(socketFileDescriptor);
29        NSLog(@"连接失败");
30        return;
31    }
32    NSLog(@"连接成功");
33 }
```

大概就是这样，因为是C语言的，所以看起来不是很方便，一般开发中都会使用比较简单的方法，如下。

CocoaAsyncSocket

iOS的socket实现是特别简单的，可以使用用github的开源类库cocoaasyncsocket简化开发，cocoaasyncsocket是支持tcp和ump的。代码大概如下：

```
1 - (IBAction)connectToServer:(id)sender {
2     // 1.与服务器通过三次握手建立连接
3     NSString *host = @"133.33.33.1";
4     int port = 1212;
5     //创建一个socket对象
6     _socket = [[GCDAsyncSocket alloc] initWithDelegate:self delegateQueue:nil];
7     //连接
8     NSError *error = nil;
9     [_socket connectToHost:host onPort:port error:&error];
10    if (error) {
11        NSLog(@"%@",error);
12    }
13    }
14    #pragma mark -socket的代理
15    #pragma mark 连接成功
16    -(void)socket:(GCDAsyncSocket *)sock didConnectToHost:(NSString *)host error:(NSError *)error {
17        NSLog(@"%s",__func__);
18    }
19    #pragma mark 断开连接
20    -(void)socketDidDisconnect:(GCDAsyncSocket *)sock withError:(NSError *)error {
21    if (err) {
22        NSLog(@"连接失败");
23    }else{
24        NSLog(@"正常断开");
25    }
26    }
27    #pragma mark 数据发送成功
28    -(void)socket:(GCDAsyncSocket *)sock didWriteDataWithTag:(long)tag {
29    NSLog(@"%s",__func__);
30    //发送完数据手动读取，-1不设置超时
31    [sock readDataWithTimeout:-1 tag:tag];
32    }
33    #pragma mark 读取数据
34    -(void)socket:(GCDAsyncSocket *)sock didReadData:(NSData *)data withOffset:(NSUInteger)offset {
35    NSString *receiverStr = [[NSString alloc] initWithData:data encoding:NSUTF8StringEncoding];
36    NSLog(@"%s %@",__func__,receiverStr);
37    }
```

 微博



CocoaChina

+ 加关注

在北京的小伙伴赶快行动起来啊！



2月27日 15:52 转发(1) | 评论(1)

[吃瓜]CocoaChina一周精选： 周一：程序员如何做到两年待遇 20K+?http://t.cn/R8rBcqb 周二：YYCache 源码解析http://t.cn/R8rBcqG 周三：在 iOS 中实现区块链 [幻灯]http://t.cn/R8rBcq5 周四：日媒：旅行青蛙 App Store 下载中国占 95%，日本仅 2%http://t.cn/R8rBcqt 周五：iOS11界面交互设计规范

下面是原理补充， 有兴趣的朋友可以细看。

网络七层协议

网络七层协议由下往上分别为物理层、数据链路层、网络层、传输层、会话层、表示层和应用层。其中物理层、数据链路层和网络层通常被称作媒体层，是网络工程师所研究的对象；传输层、会话层、表示层和应用层则被称作主机层，是用户所面向和关心的内容。

HTTP协议对应于应用层，TCP协议对应于传输层，IP协议对应于网络层，HTTP协议是基于TCP连接的,三者本质上没有可比性。TCP/IP是传输层协议，主要解决数据如何在网络中传输；而HTTP是应用层协议，主要解决如何包装数据。**Socket**是应用层与**TCP/IP**协议族通信的中间软件抽象层，是它的一组接口。

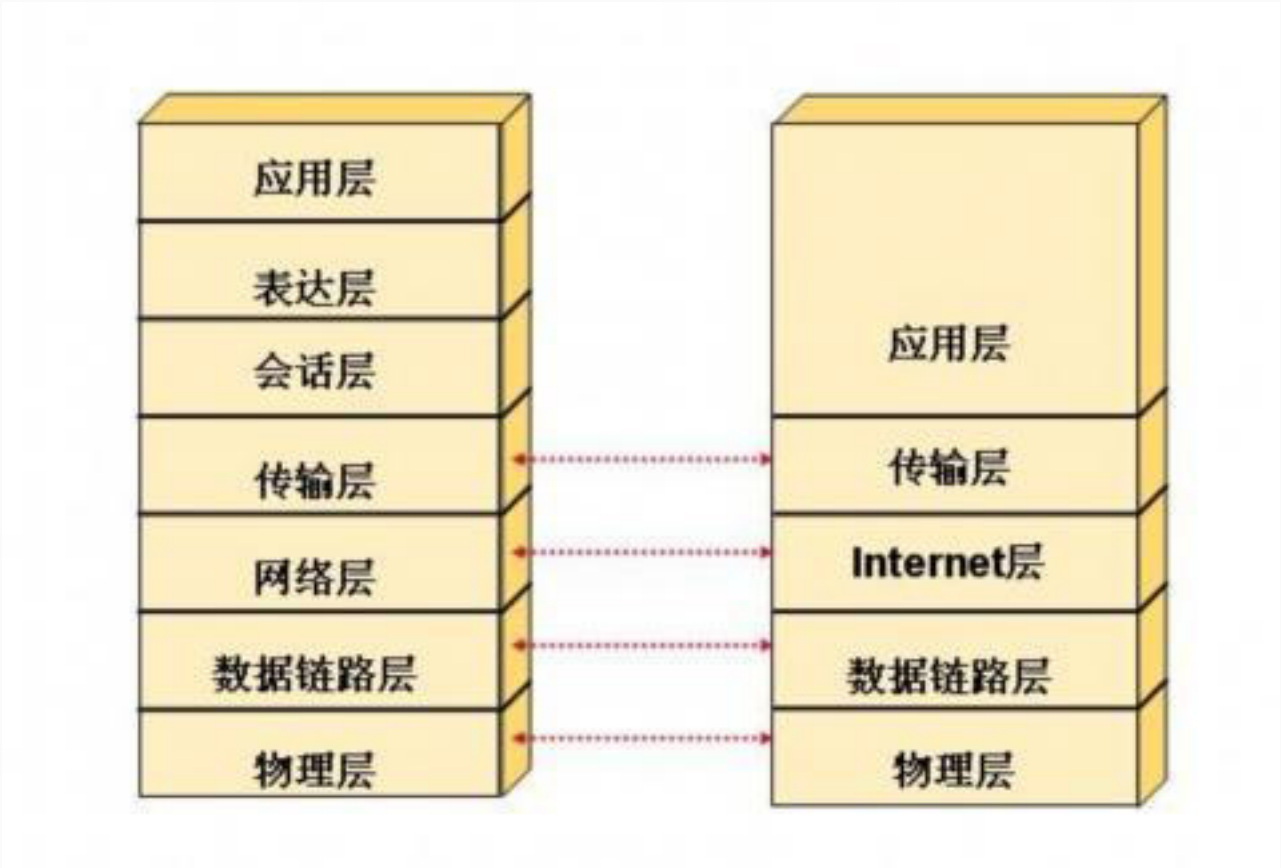


网络七层协议

TCP/IP五层模型

TCP/IP五层模型的协议分为：应用层、传输层、网络层、数据链路层和物理层。中继器、集线器、还有我们通常说的双绞线也工作在物理层；网桥（现已很少使用）、以太网交换机（二层交换机）、网卡（其实网卡是一半工作在物理层、一半工作在数据链路层）在数据链路层；路由器、三层交换机在网络层；传输层主要是四层交换机、也有工作在四层的路由器。

TCP/IP协议中的应用层处理七层模型中的第五层、第六层和第七层的功能。TCP/IP协议中的传输层并不能总是保证在传输层可靠地传输数据包，而七层模型可以做到。TCP/IP协议还提供一项名为UDP（用户数据报协议）的选择。UDP不能保证可靠的数据包传输。



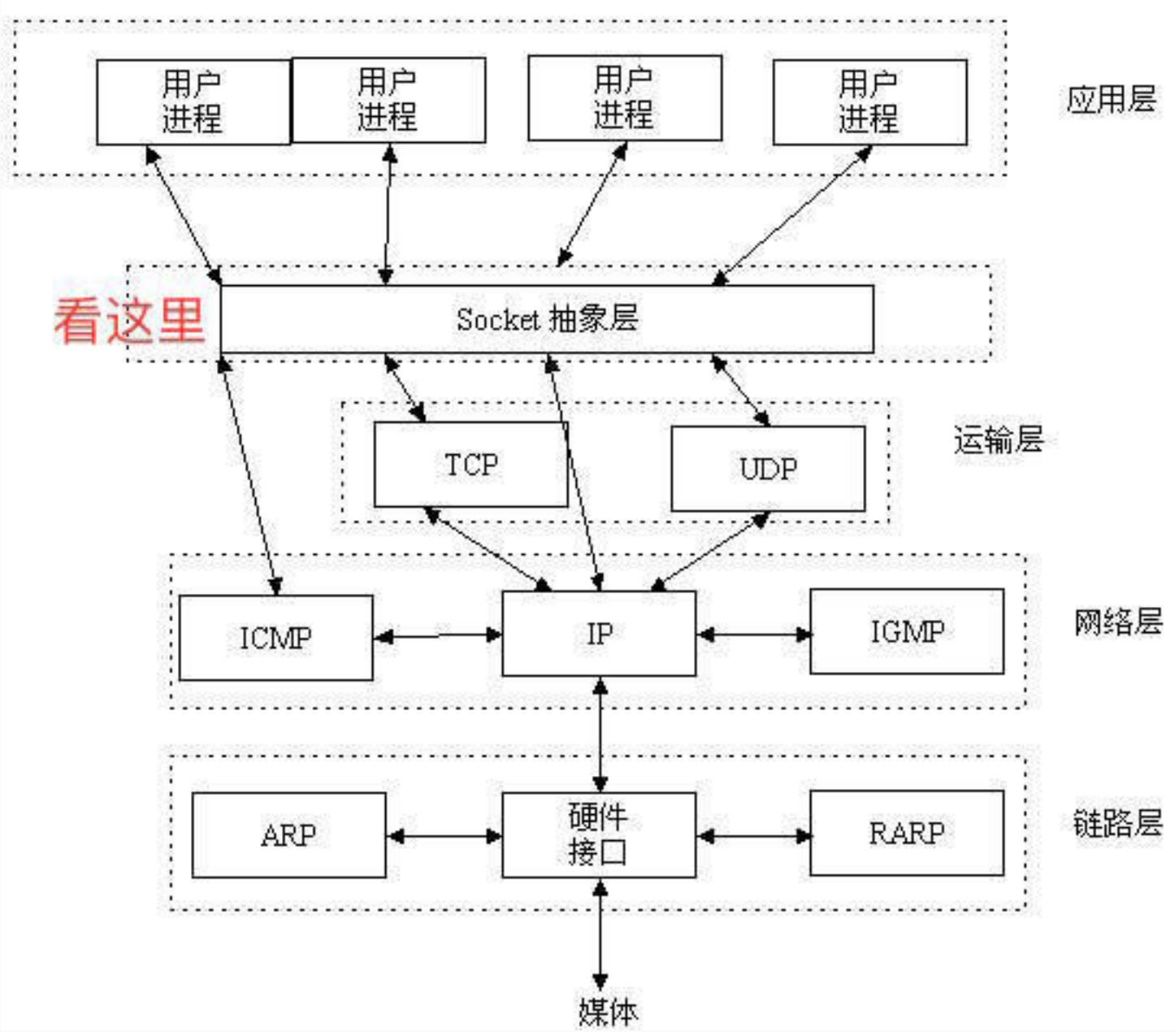
对应关系

- TCP：面向连接、传输可靠(保证数据正确性,保证数据顺序)、用于传输大量数据(流模式)、速度慢，建立连接需要开销较多(时间，系统资源)。
- UDP：面向非连接、传输不可靠、用于传输少量数据(数据包模式)、速度快。

TCP是一种流模式的协议，UDP是一种数据报模式的协议。

在传输数据时，可以只使用传输层（TCP/IP）， 但是那样的话，由于没有应用层， 便无法识别数据内容， 如果想要使传输的数据有意义， 则必须使用应用层协议（HTTP、FTP、TELNET等）， 也可以自己定义应用层协议。

WEB使用HTTP作传输层协议， 以封装HTTP文本信息， 然后使用TCP/IP做传输层协议将它发送到网络上。Socket是对TCP/IP协议的封装，Socket本身并不是协议， 而是一个调用接口（API）， 通过Socket， 我们才能使用TCP/IP协议。



Socket的位置

TCP连接

要想明白Socket连接，先要明白TCP连接。手机能够使用联网功能是因为手机底层实现了TCP/IP协议，可以使手机终端通过无线网络建立TCP连接。TCP协议可以对上层网络提供接口，使上层网络数据的传输建立在“无差别”的网络之上。

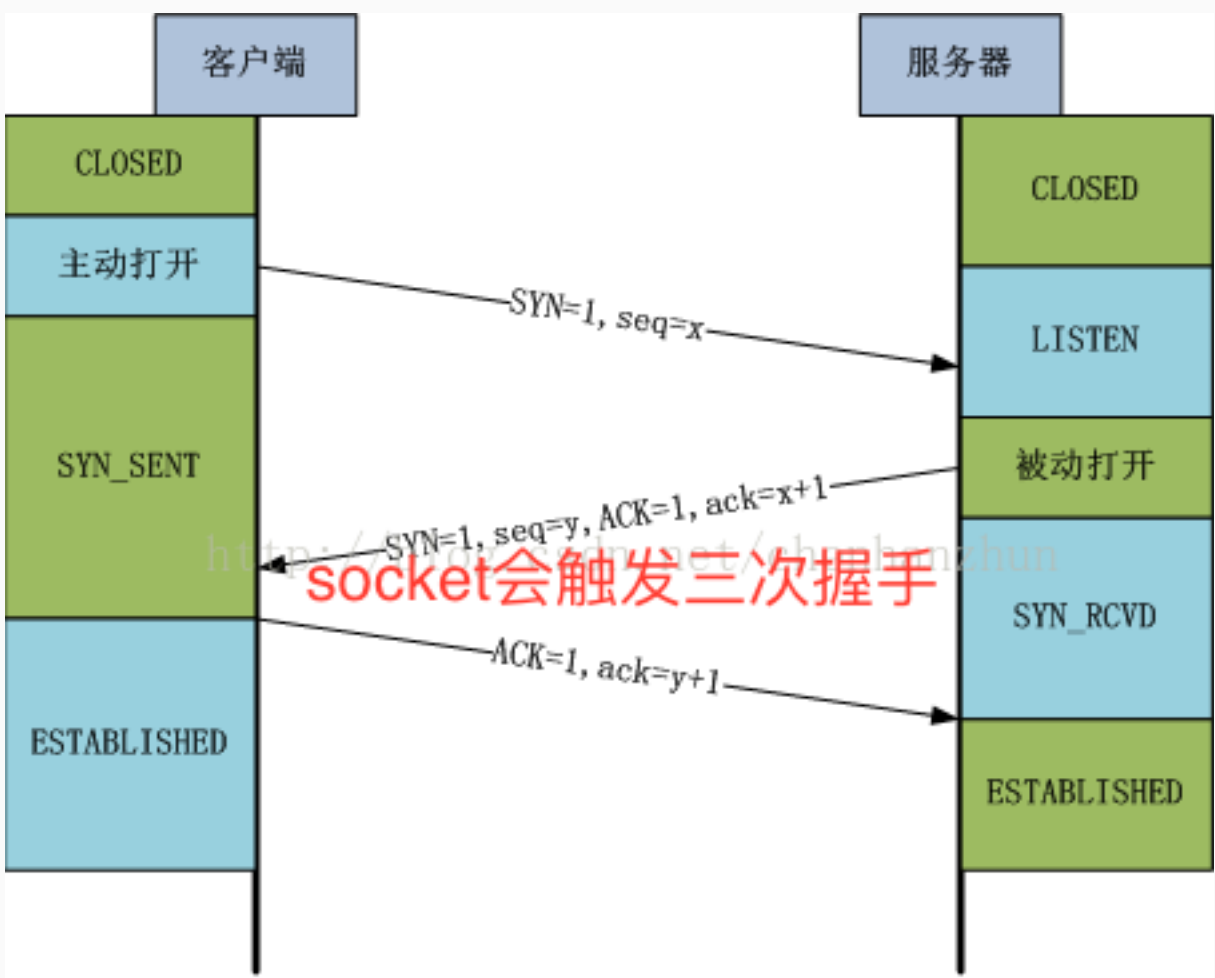
建立起一个TCP连接需要经过“三次握手”：

第一次握手： 客户端发送syn包(syn=j)到服务器， 并进入SYN_SEND状态， 等待服务器确认；

第二次握手： 服务器收到syn包， 必须确认客户的SYN（ack=j+1）， 同时自己也发送一个SYN包（syn=k）， 即SYN+ACK包， 此时服务器进入SYN_RECV状态；

第三次握手： 客户端收到服务器的SYN + ACK包， 向服务器发送确认包ACK(ack=k+1)， 此包发送完毕， 客户端和服务器进入ESTABLISHED状态， 完成三次握手。

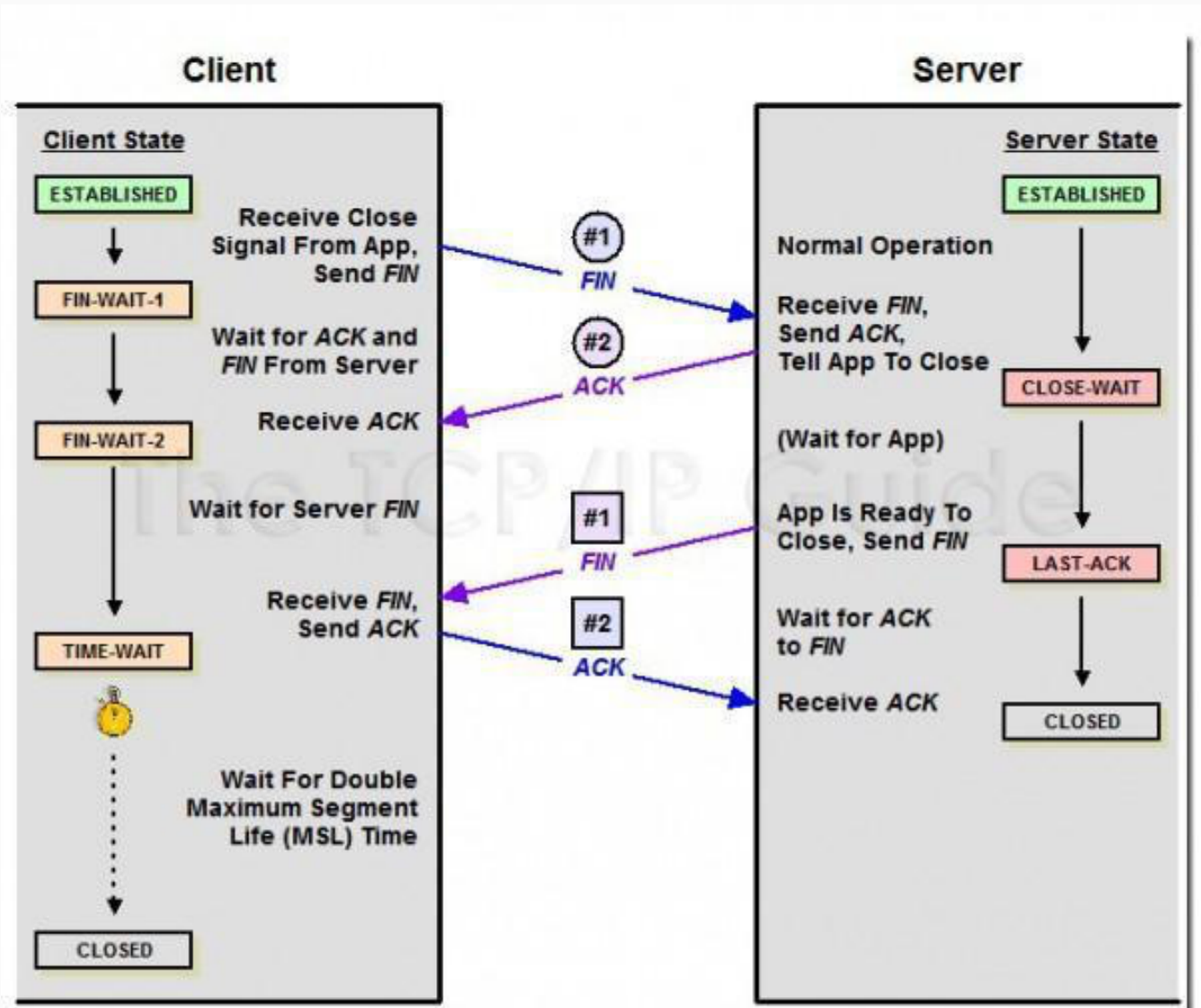
三次握手(Three-way Handshake)即建立一个TCP连接时， 需要客户端和服务端总共发送3个包。三次握手的目的是连接服务器指定端口， 建立TCP连接,并同步连接双方的序列号和确认号并交换TCP 窗口大小信息。在socket编程中， 客户端执行connect()时,将触发三次握手。



三次握手

握手过程中传送的包里不包含数据，三次握手完毕后，客户端与服务器才正式开始传送数据。理想状态下，TCP连接一旦建立，在通信双方中的任何一方主动关闭连接之前，TCP 连接都将被一直保持下去。断开连接时服务器和客户端均可以主动发起断开TCP连接请求，断开过程需要经过“四次握手”。

TCP连接的拆除需要发送四个包，因此称为四次握手(four-way handshake)。在socket编程中，任何一方执行close()操作即可产生握手（有地方称为“挥手”）操作。



TCP连接的拆除

之所以有“三次握手”和“四次握手”的区别，是因为连接时当Server端收到Client端的SYN连接请求报文后，可以直接发送SYN+ACK报文。其中ACK报文是用来应答的，SYN报文是用来同步的。但是关闭连接时，当Server端收到FIN报文时，很可能并不会立即关闭SOCKET，所以只能先回复一个ACK报文，告诉Client端，“你发的FIN报文我收到了”。只有等到我Server端所有的报文都发送完了，我才能发送FIN报文，因此不能一起发送。故需要四步握手。

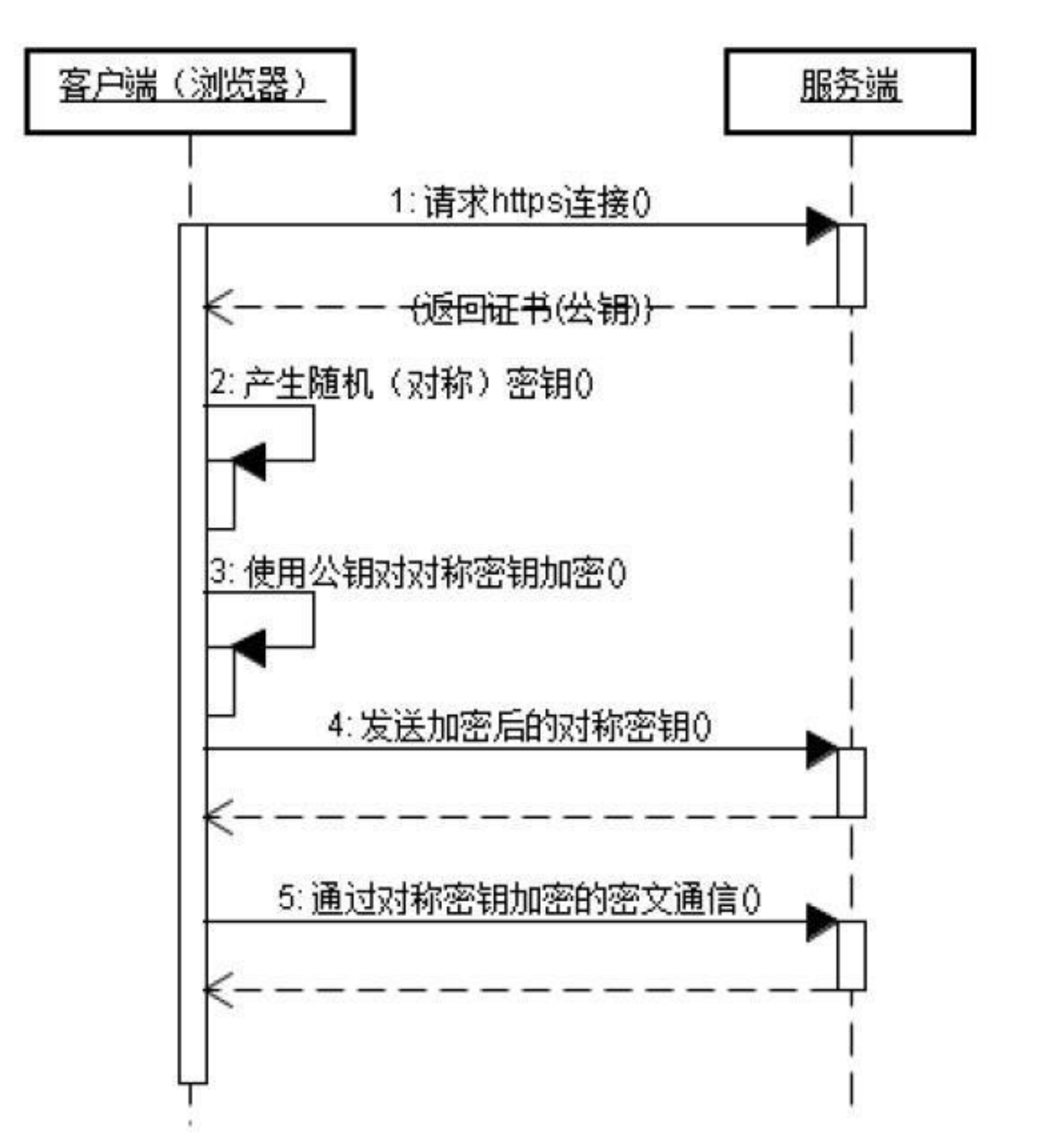
HTTP连接

HTTP协议即超文本传送协议(HypertextTransfer Protocol)，是Web联网的基础，也是手机联网常用的协议之一，HTTP协议是建立在TCP协议之上的一种应用。

HTTP连接最显著的特点是客户端发送的每次请求都需要服务器回送响应，在请求结束后，会主动释放连接。从建立连接到关闭连接的过程称为“一次连接”。因此HTTP连接是一种“短连接”，要保持客户端程序的在线状态，需要不断地向服务器发起连接请求。若服务器长时间无法收到客户端的请求，则认为客户端“下线”，若客户端长时间无法收到服务器的回复，则认为网络已经断开。在HTTP 1.0中，客户端的每次请求都要求建立一次单独的连接，在处理完本次请求后，就自动释放连接。在HTTP 1.1中则可以在一次连接中处理多个请求，并且多个请求可以重叠进行，不需要等待一个请求结束后再发送下一个请求。

HTTPS（Hyper Text Transfer Protocol over Secure Socket Layer），是以安全为目标的HTTP通道，是HTTP的安全版。在HTTP下加入SSL层，HTTPS的安全基础是SSL，因此加密的详细内容就需要SSL。HTTPS存在不同于HTTP的默认端口及一个加密/身份验证层（在HTTP与TCP之间）。HTTP协议以明文方式发送内容，不提供任何方式的数据加密，如果攻击者截取了Web浏览器和网站服务器之间的传输报文，就可以直接读懂其中的信息，因此HTTP协议不适合传输一些敏感信息。

https协议需要到ca申请证书；http是超文本传输协议，信息是明文传输，https 则是具有安全性的ssl加密传输协议；http和https使用的是完全不同的连接方式，用的端口也不一样，前者是80，后者是443；http的连接很简单，是无状态的，HTTPS协议是由SSL+HTTP协议构建的可进行加密传输、身份认证的网络协议。



HTTPS

Socket连接与HTTP连接的不同

通常情况下**Socket**连接就是**TCP**连接，因此Socket连接一旦建立，通信双方即可开始相互发送数据内容，直到双方连接断开。但在实际应用中，客户端到服务器之间的通信防火墙默认会关闭长时间处于非活跃状态的连接而导致 Socket 连接断连，因此需要通过轮询告诉网络，该连接处于活跃状态。

而HTTP连接使用的是“请求—响应”的方式，不仅在请求时需要先建立连接，而且需要客户端向服务器发出请求后，服务器端才能回复数据。



微信扫一扫

订阅每日移动开发及APP推广热点资讯

公众号：CocoaChina

我要投稿

收藏文章

分享到：




113113

上一篇：[iOS夯实：RunLoop](#)

下一篇：[GitHub前50名的Objective-C动画相关库](#)

相关资讯	
<ul style="list-style-type: none">■ 读懂「 唱吧KTVHTTPCache 」设计思想■ HTTP协议中的401授权认证机制在iOS上的实现■ 【iOS开发】之CocoaAsyncSocket使用■ 关于 iOS HTTP2.0 的一次学习实践■ iOS即时通讯进阶 - CocoaAsyncSocket源码解析(Read篇)	<ul style="list-style-type: none">■ iOS 通俗易懂的HTTP网络■ iOS网络协议----HTTP/TCP/IP浅析■ SRWebSocket源码浅析■ iOS~URLCache探索■ 一步一步构建你的iOS网络层 - HTTP篇

我来说两句




你怎么看？快来评论一下吧！

您还没有登录！请 [登录](#) 或 [注册](#)

发表评论


所有评论（59）


- 


Nothing1

mark mark

2017-09-08 14:39:49

 0


 0


回复
- 


jzwvip

mark

2017-04-12 10:56:09

 0


 0


回复
- 

低调的魅力

udp呢？

2017-03-15 18:24:57

 0

 0

回复

	零零九	2017-03-15 15:39:16
不好意思 我想问一下 socket怎么上传项目中mp3的文件		
		<div> 0</div> <div> 0</div> <div>回复</div>
	kotercy	2016-08-20 02:50:47
群主你好，我在编写socket的时候照你写的去做，引入GCDAsyncSocket，初始化socket对象，代理disconnect也正常执行了，但是一直无法接收服务器的数据包，didreaddata这个方法一直不执行，可不可以给点意见，小白求带		
		<div> 1</div> <div> 2</div> <div>回复</div>
	super_sun	2016-11-23 18:28:14
回复： kotercy [sock readDataWithTimeout:-1 tag:tag];//需要手动获取数据		
		<div> 1</div> <div> 2</div> <div>回复</div>
	caodaxun	2016-07-18 21:54:07
socket一般用在哪里？		
		<div> 0</div> <div> 2</div> <div>回复</div>
	我们私奔吧	2016-07-14 23:53:47
赞		
		<div> 0</div> <div> 1</div> <div>回复</div>
	a123478we	2016-07-05 19:02:42
不错不错		
		<div> 0</div> <div> 1</div> <div>回复</div>
	a123478we	2016-07-05 19:02:32
不错不错		
		<div> 0</div> <div> 1</div> <div>回复</div>
	a123478we	2016-07-05 19:02:14
不错不错		
		<div> 0</div> <div> 1</div> <div>回复</div>
	a123478we	2016-07-05 19:01:52
不错不错		
		<div> 0</div> <div> 1</div> <div>回复</div>
	callmefrank	2016-06-23 23:14:57
mark		
		<div> 0</div> <div> 1</div> <div>回复</div>
	minglin_chen	2016-06-23 19:26:26
Mark		
		<div> 0</div> <div> 1</div> <div>回复</div>
	猫咪瞎跳	2016-06-22 19:34:45
mark		
		<div> 0</div> <div> 2</div> <div>回复</div>

	tomorrow_ mark	2016-06-20 02:52:31	 0  1 回复
	Alphts mark	2016-06-20 02:37:33	 0  1 回复
	kassign Mark	2016-06-13 20:53:49	 0  1 回复
	sunshine199110 mark	2016-06-08 17:50:21	 0  1 回复
	棒棒糖之kiss mark	2016-06-08 01:11:00	 0  3 回复
	2026_Cloud 还不错的文章	2016-06-07 17:27:00	 0  4 回复

更多评论