作者：椰丝柔丝

链接：https://www.nowcoder.com/discuss/578614

来源：牛客网

http1.1 http1.0 http2.0 区别

参考

https://www.cnblogs.com/heluan/p/8620312.html

https://blog.csdn.net/ailunlee/article/details/97831912

HTTP1.0和HTTP1.1的区别

1.1 长连接(Persistent Connection)

HTTP1.1支持长连接和请求的流水线处理，在一个TCP连接上可以传送多个HTTP请求和响应，减少了建立和关闭连接的消耗和延迟，在HTTP1.1中默认开启长连接keep-alive，一定程度上弥补了HTTP1.0每次请求都要创建连接的缺点。HTTP1.0需要使用keep-alive参数来告知服务器端要建立一个长连接。

1.2 节约带宽

HTTP1.0中存在一些浪费带宽的现象，例如客户端只是需要某个对象的一部分，而服务器却将整个对象送过来了，并且不支持断点续传功能。HTTP1.1支持只发送header信息（不带任何body信息），如果服务器认为客户端有权限请求服务器，则返回100，客户端接收到100才开始把请求body发送到服务器；如果返回401，客户端就可以不用发送请求body了节约了带宽。

1.3 HOST域

在HTTP1.0中认为每台服务器都绑定一个唯一的IP地址，因此，请求消息中的URL并没有传递主机名（hostname），HTTP1.0没有host域。随着虚拟主机技术的发展，在一台物理服务器上可以存在多个虚拟主机（Multi-homed Web Servers），并且它们共享一个IP地址。**HTTP1.1的请求消息和响应消息都支持host域**，且请求消息中如果没有host域会报告一个错误（400 Bad Request）。

1.4缓存处理

在HTTP1.0中主要使用header里的If-Modified-Since,Expires来做为缓存判断的标准，HTTP1.1则引入了更多的缓存控制策略例如Entity tag，If-Unmodified-Since, If-Match, If-None-Match等**更多可供选择的缓存头来控制缓存策略**。

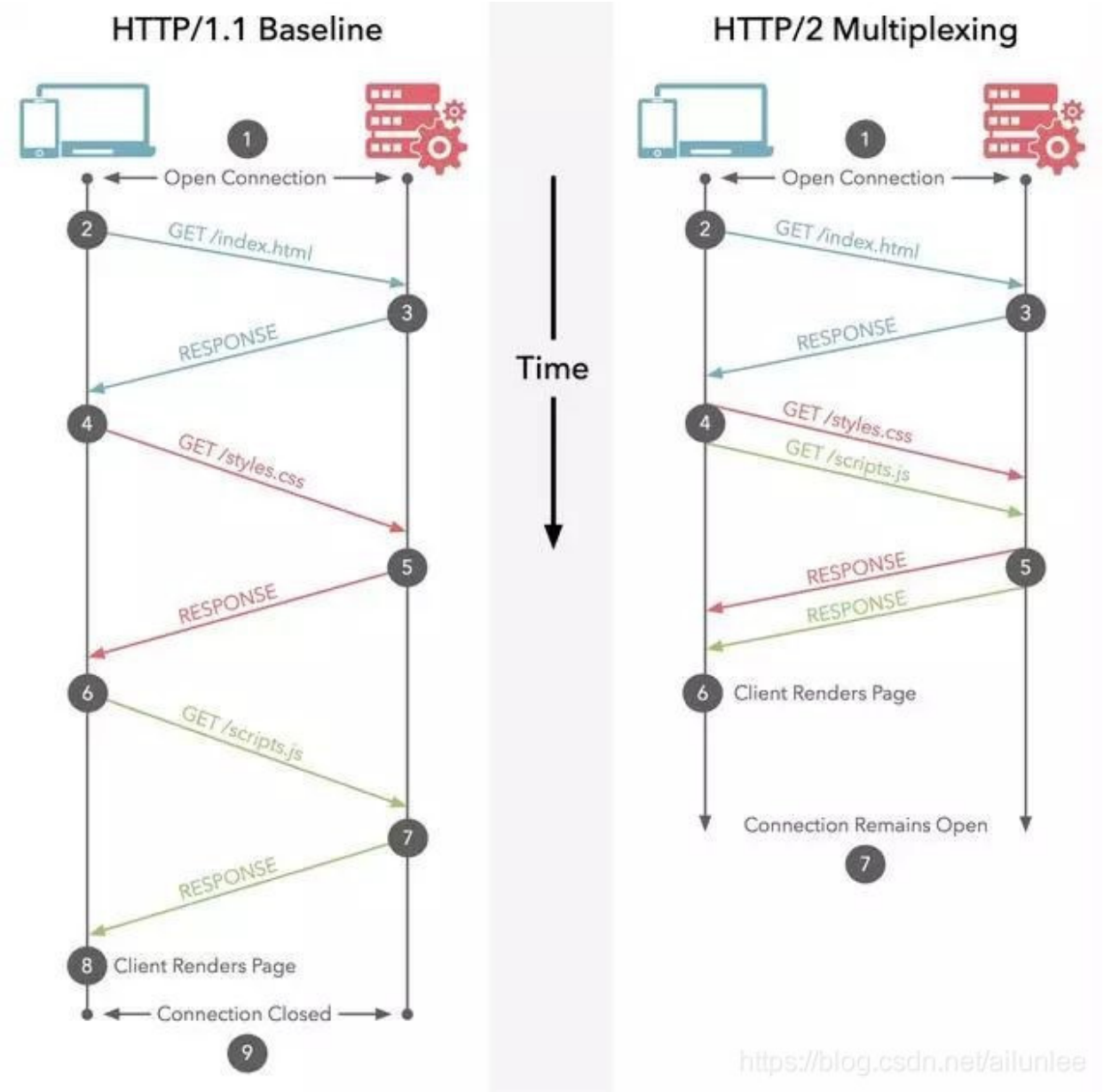
1.5错误通知的管理

在HTTP1.1中**新增了24个错误状态响应码**，如409（Conflict）表示请求的资源与资源的当前状态发生冲突；410（Gone）表示服务器上的某个资源被永久性的删除。

2 HTTP1.1和HTTP2.0的区别

2.1 多路复用

HTTP2.0使用了多路复用的技术，做到同一个连接并发处理多个请求，而且并发请求的数量比HTTP1.1大了好几个数量级。HTTP1.1也可以多建立几个TCP连接，来支持处理更多并发的请求，但是创建TCP连接本身也是有开销的。



2.2 头部数据压缩

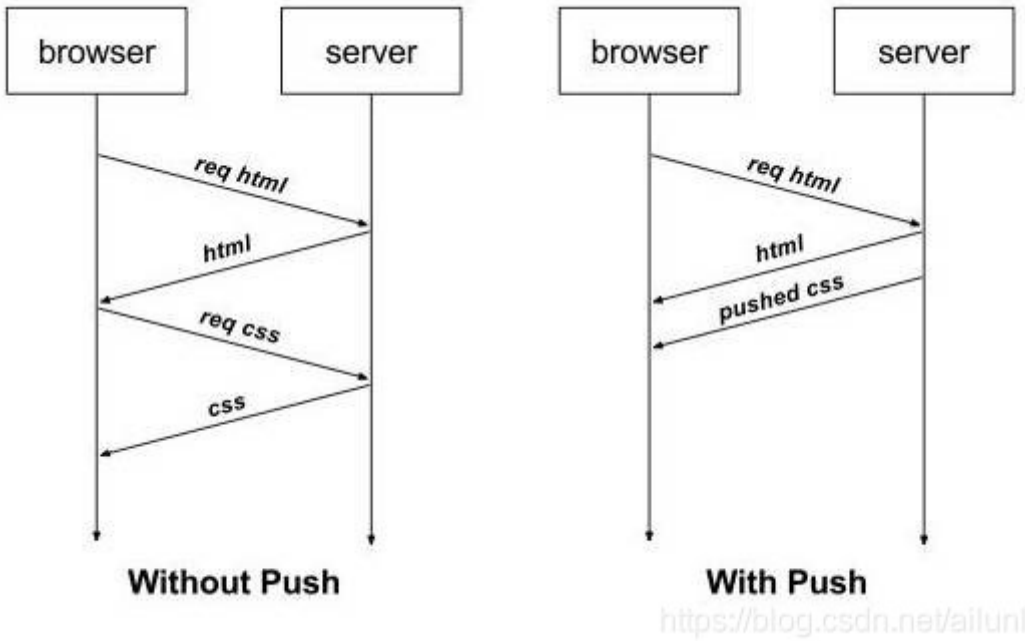
在HTTP1.1中，HTTP请求和响应都是由状态行、请求/响应头部、消息主体三部分组成。一般而言，消息主体都会经过gzip压缩，或者本身传输的就是压缩过后的二进制文件，但状态行和头部却没有经过任何压缩，直接以纯文本传输。随着Web功能越来越复杂，每个页面产生的请求数也越来越多，导致消耗在头部的流量越来越多，尤其是每次都要传输UserAgent、Cookie这类不会频繁变动的内容，完全是一种浪费。

HTTP1.1不支持header数据的压缩，HTTP2.0使用HPACK算法对header的数据进行压缩，这样数据体积小了，在网络上传输就会更快。

2.3 服务器推送

服务端推送是一种在客户端请求之前发送数据的机制。网页使用了许多资源：HTML、样式表、脚本、图片等等。在HTTP1.1中这些资源每一个都必须明确地请求。这是一个很慢的过程。浏览器从获取HTML开始，然后在它解析和评估页面的时候，增量地获取更多的资源。因为服务器必须等待浏览器做每一个请求，网络经常是空闲的和未充分使用的。

为了改善延迟，HTTP2.0引入了server push，它允许服务端推送资源给浏览器，在浏览器明确地请求之前，免得客户端再次创建连接发送请求到服务器端获取。这样客户端可以直接从本地加载这些资源，不用再通过网络。



redis长度过长怎么优化？哪个api，数据量超过多少效率会变低？

参考

https://blog.csdn.net/beyond59241/article/details/78889867/

Redis使用过程中经常会有各种大key的情况， 比如：

- 1： 单个简单的key存储的value很大
- 2： hash， set，zset，list 中存储过多的元素（以万为单位）

由于redis是单线程运行的，如果一次操作的value很大会对整个redis的响应时间造成负面影响，所以，业务上能拆则拆，下面举几个典型的分拆方案。

1、单个简单的key存储的value很大

1.1、 改对象需要每次都整存整取

可以尝试将对象分拆成几个key-value， 使用multiGet获取值，这样分拆的意义在于分拆单次操作的压力，将操作压力平摊到多个redis实例中，降低对单个redis的IO影响；

1.2、该对象每次只需要存取部分数据

可以像第一种做法一样，分拆成几个key-value， 也可以将这个存储在一个hash中，每个field代表一个具体的属性，使用hget,hmget来获取部分的value，使用hset，hmset来更新部分属性

2、 hash， set，zset，list 中存储过多的元素

类似于场景一种的第一个做法，可以将这些元素分拆。

以hash为例，原先的正常存取流程是 hget(hashKey, field) ; hset(hashKey, field, value)

现在，固定一个桶的数量，比如 10000， 每次存取的时候，先在本地计算field的hash值，模除 10000， 确定了该field落在哪个key上。

newHashKey = hashKey + (*hash*(field) % 10000）;

hset (newHashKey, field, value) ;

hget(newHashKey, field)

set, zset, list 也可以类似上述做法.

但有些不适合的场景，比如，要保证 lpop 的数据的确是最早push到list中去的，这个就需要一些附加的属性，或者是在 key的拼接上做一些工作（比如list按照时间来分拆）。

MySQL做过哪些优化？覆盖索引？limit两个参数区别？MySQL分页优化的其他方法

回表、索引覆盖、最左匹配、索引下推

优化Limit查询

子查询的分页方式

优化步骤1 :使用有索引的列或主键进行Order by操作

SELECT film_ id, description FROM sakila.film ORDER BY film_ id LIMIT 50,5;

优化步骤2 :记录上次返回的主键,在下次查询时使用主键过滤

SELECT film_ id, description FROM sakila.film WHERE film_ id > 55 and film_ id <=60 ORDER BY film_ id LIMIT 1, 5;

*避免了数据量大时扫描过多的记录

concurrenthashmap如何保证线程安全，说说你的理解

https://www.cnblogs.com/junjiang3/p/8686290.html

arraylist linkedlist 使用场景

https://blog.csdn.net/m0_37637141/article/details/81943468

本地线程和守护线程的区别，Thread.setDemon();

线程状态

线程池参数？线程池为什么用new的不好？

1、每次new Thread，新建对象性能差

2、缺乏统一管理，可能导致线程创建过多，死机等。

3、缺乏更多功能，如：定时执行，定期执行，线程中断等。