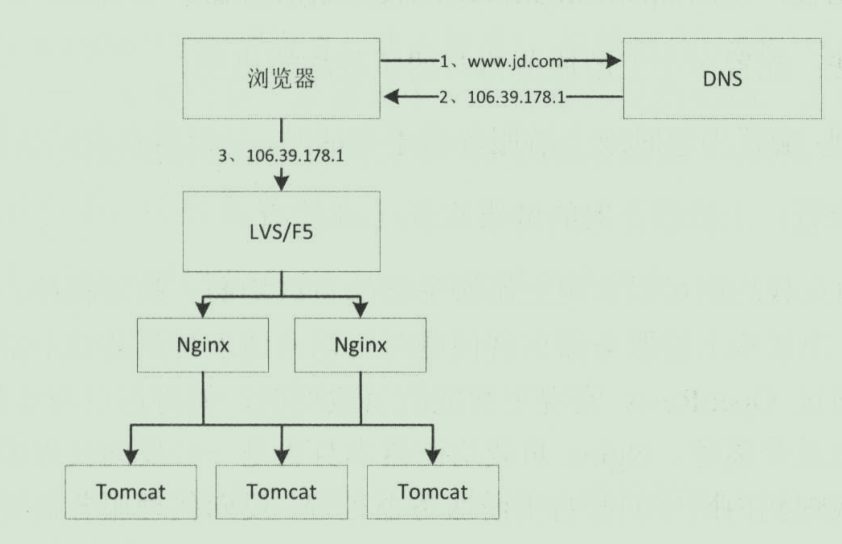
负载均衡和反向代理

当我们的应用单实例不能支撑用户的请求时，就需要扩容。从一台服务器扩容为多台服务器。在请求的时候，浏览器首先会查询DNS服务器获取对应的IP,然后通过IP访问对应的服务。



upstream server：上游服务器。指的是nginx负载均衡到处理业务的服务器，也称为real server真实处理业务的服务器。

负载均衡要关心的问题：

上游服务器配置：upstream server

负载均衡算法：配置多个上游服务器时的负载均衡机制

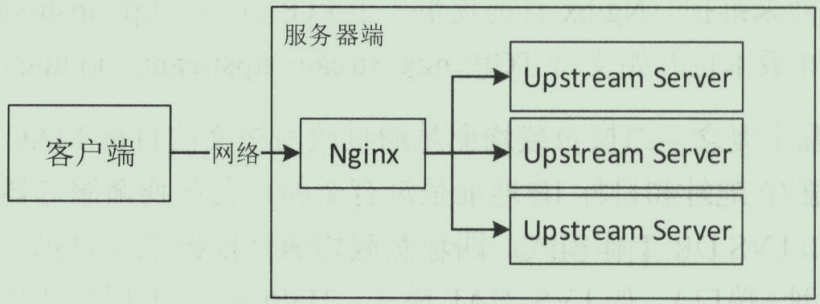
失败重试：当超时或者上游服务器不可用时，是否需要重试其它上游服务器

心跳检测：上游服务器的健康检查/心跳检查

使用Nginx来做负载均衡可以实现上游服务器的负载均衡、故障转移、失败重试、健康检查

当某台服务器出现问题时可以将其它转发到其他上游服务器来保障高可用

Nginx作为负载均衡器的同时本身也是一台反向代理服务器，将用户的请求通过Nginx代理到内网的某台上游服务器处理，反向代理服务器可以对响应的结果进行缓存、压缩等处理来提升性能！



upstream server 配置

#脑健康app

upstream mobile {

server 10.10.12.106:9111 weight=1;

server 10.10.12.107:9111 weight=2;

}

upstream server的主要配置

IP地址和端口：配置上游服务器的IP和端口

权重：weight用来配置权重，默认都是1权重分配的越多这台服务器接收的请求就越多。上面的配置，如果客户端发送的三次请求，会有一个由106服务器处理，两个交给107服务器处理。根据服务器的实际处理能力来进行权重设置。

配置proxy\_pass来处理用户请求

server {

listen 80;

server\_name 10.10.12.108;

location / {

proxy\_pass http://mobile;

}

}

当访问Nginx 10.10.12.108时，会将请求反向代理到mobile配置的upstream server

负载均衡算法

负载均衡算法决定了用户的请求到来时选择哪一台上游服务器来进行处理，默认使用的是round-robin(轮询)

round-robin：轮询，默认的负载均衡算法、配合weight配置可以实现基于权重的轮询

ip\_hash：根据客户IP进行负载均衡，相同的IP负载到同一个上游服务器

失败重试

upstream mobile {

server 10.10.12.106:9111 max\_files=2 fail\_timeout=10s weight=1;

server 10.10.12.107:9111 max\_files=2 fail\_timeout=10s weight=2;

}

配置上游服务器的max\_files和fail\_timeout来指定每个上游服务器，当fail\_timeout时间内失败了max\_files次请求，则认为该上游服务器不可用，然后移除该服务器。fail\_timeout时间后会再次将该上游服务器加入到存活的服务器列表进行重试。