**Java内存区域与内存溢出异常（大标题）**

**Java虚拟机运行时数据区（中标题）**

**行时数据区（小标题）**

我把代码跑一遍 (正文)

我把代码跑一遍 (突出重点)

**学习的重点在什么地方？（问题）**

**重点回顾**

·Redis服务器的所有数据库都保存在redisServer.db数组中，而数据库的数量则由

redisServer.dbnum属性保存。

·客户端通过修改目标数据库指针，让它指向redisServer.db数组中的不同元素来切换不同的数据库。（一些特别突出的句子重点）

什么是Minor GC、Major GC、Full GC以及触发条件？

从年轻代空间（包括 Eden 和 Survivor 区域）回收内存被称为 Minor GC

第一天：

通用mapper-> mybatis动态sql封装包->增删改查->0 sql

pagehelper-> 实现分页

**电商系统技术特点**

技术新

技术范围广

分布式

高并发、集群、负载均衡、高可用

海量数据

业务复杂

系统安全

**电商模式**

B2B ：商家对商家 案例：阿里巴巴（批发网）

C2C ：客户对客户 案例：淘宝

B2C ：商对客 案例：唯品会

C2B ：客户到企业 案例：肯德基

O2O ：online to offline 线上到线下 案例：美团、饿了么

F2C ：Factory to customer ，直接去厂家拿货，没有中间商做差价

B2B2C ：商家到商家、商家到客户 案例：京东商城、天猫商城

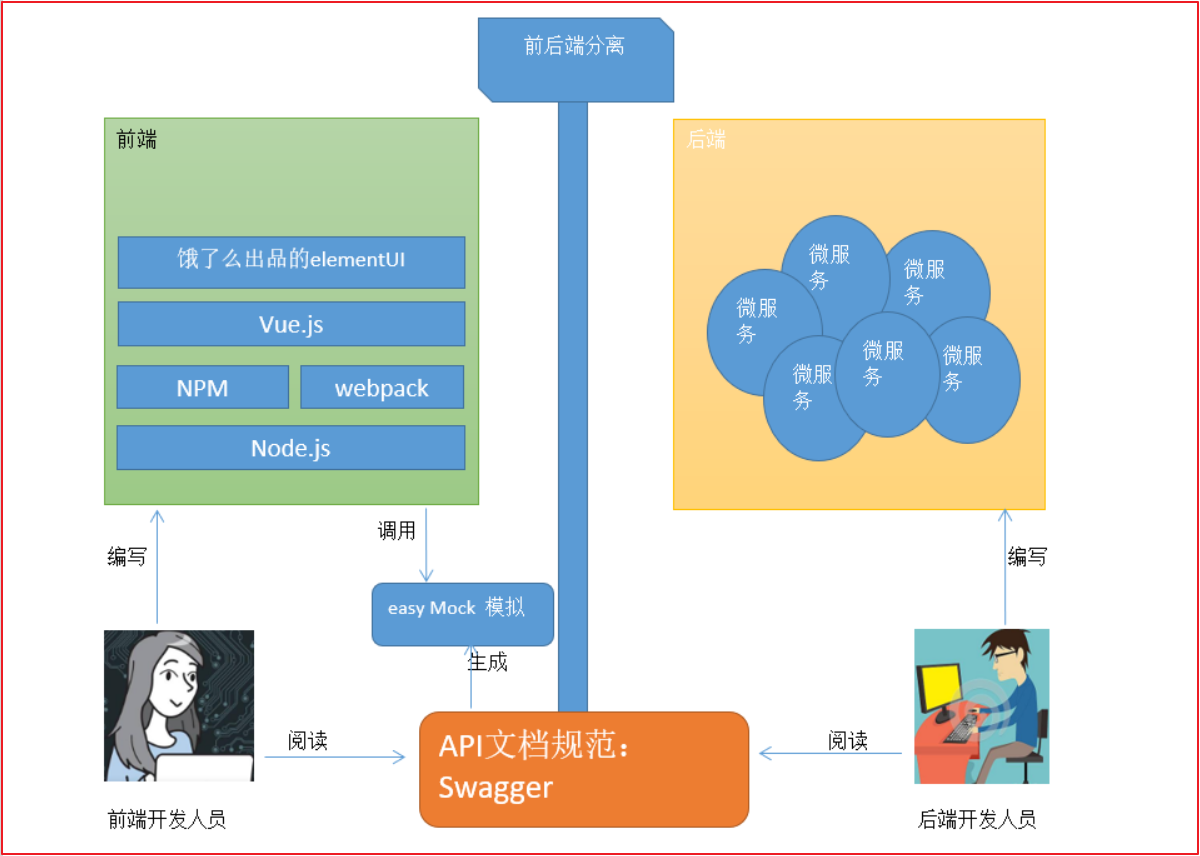
**系统设计**

畅购商城属于B2C电商模式，运营商将自己的产品发布到网站上，会员注册后，在网站上将商品添加到购物车，并且下单，完成线上支付，用户还可以参与秒杀抢购。

**前后端分离**

网站后台的部分采用前后端分离方式。

以前的JavaWeb项目大多数都是java程序员又当爹又当妈，又搞前端，又搞后端。随着时代的发展，渐渐的许多大中小公司开始把前后端的界限分的越来越明确，前端工程师只管前端的事情，后端工程师只管后端的事情。正所谓术业有专攻，一个人如果什么都会，那么他毕竟什么都不精。



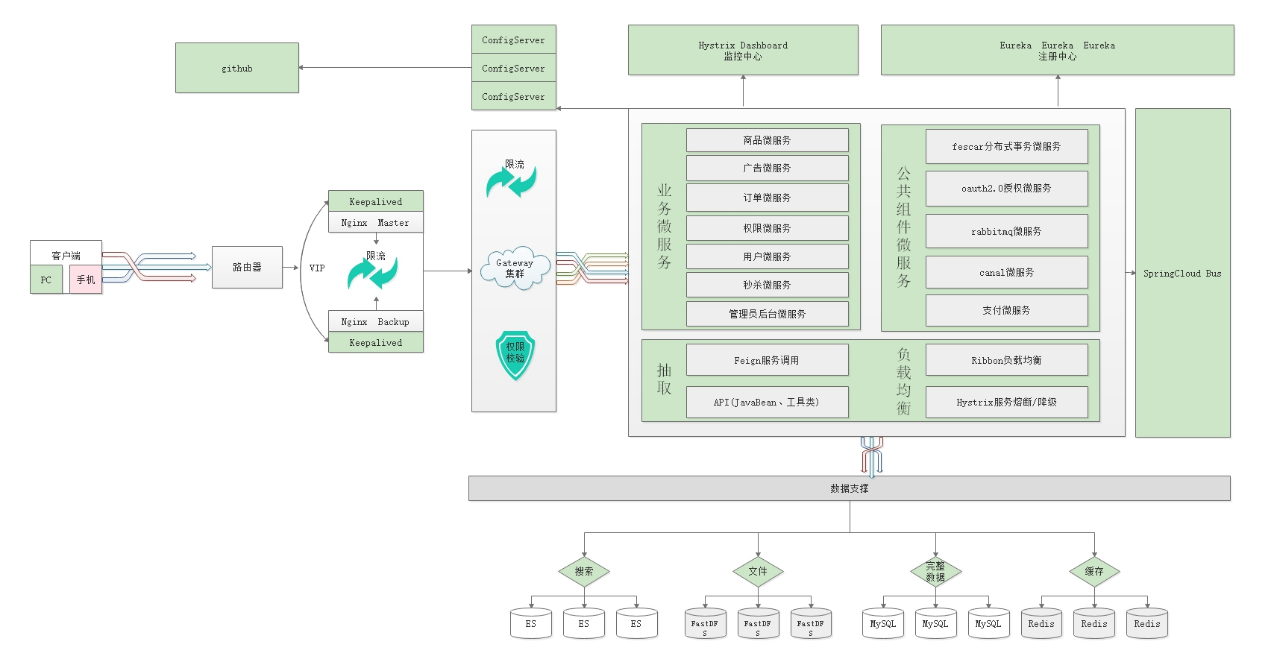
**对于后端java工程师：**

把精力放在设计模式，spring+springmvc，linux，mysql事务隔离与锁机制，mongodb，http/tcp，多线程，分布式架构，弹性计算架构，微服务架构，java性能优化，以及相关的项目管理等等。

**技术架构**



**系统架构图**



Nginx ： 可抗5万并发

Keepalived : 解决Nginx单点故障，保证Nginx 高可用 , VIP虚拟IP

Nginx限流: 可以根据ip限流，可以根据访问速率限流

OpenResty : 可以提升nginx处理并发的能力(10k-1000k)

微服务网关Gateway：

1. 将用户请求路由到不同的微服务

2. 限流

3. 权限验证

Feign ：

来实现微服务之间的调用，默认使用Ribbon的负载均衡

数据搜索：elastic search

数据持久化 ：mysql

缓存 ：redis

文件 ：FastDFS

vm : username : root password : 123456

mysql :

mysql -u root -p

123456

**数据库设计**

大型应用里面，推荐使用反三范式

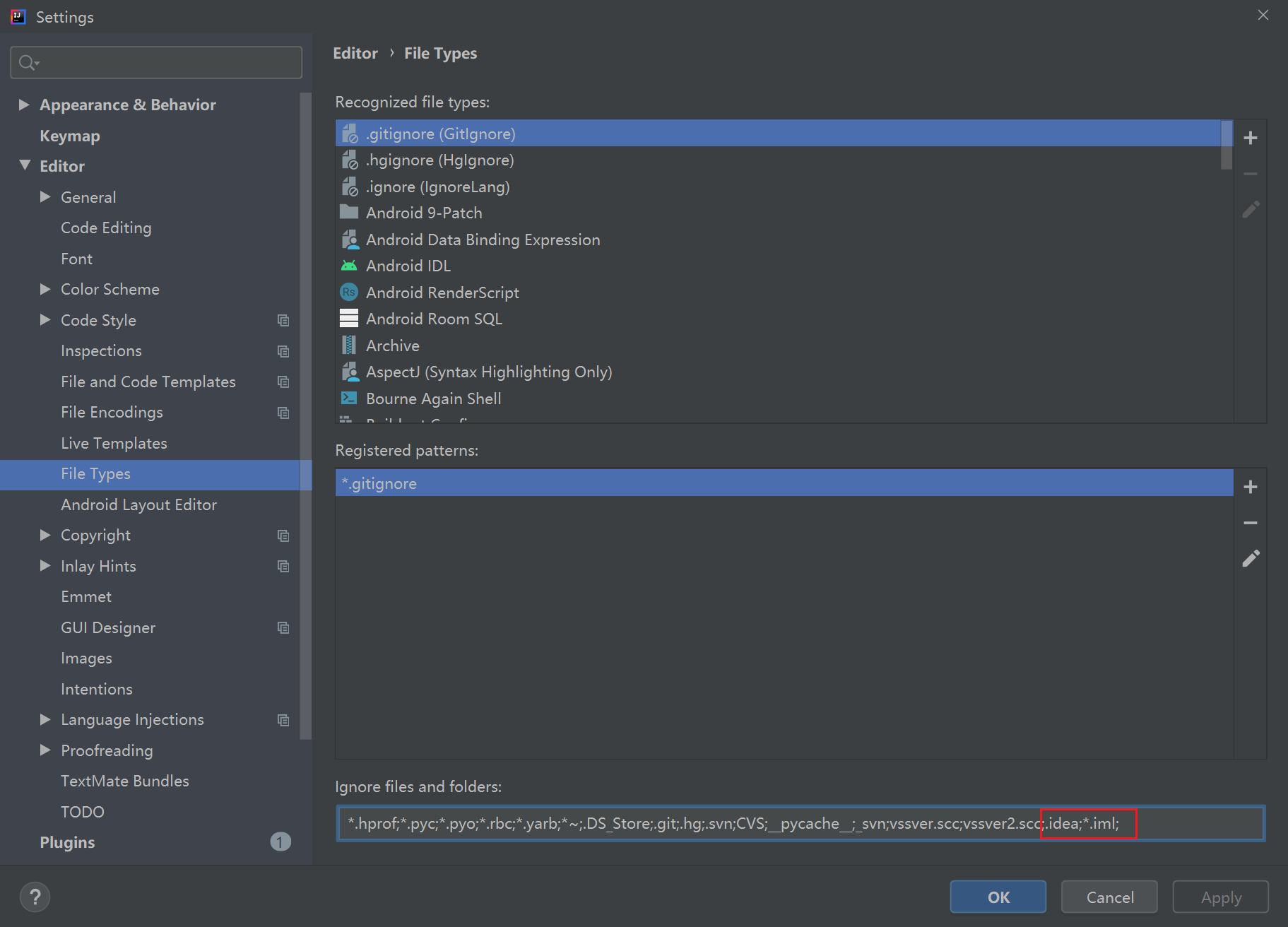
1. 不要创建外键（增删改不用检查依赖）

2. 数据冗余操作（增加冗余外键，不要关联外表）

父工程：均为pom工程 (需要pom.xml中指定package)

其他: 均为jar 工程 (不指定package，默认是jar工程)

隐藏不想看到的文件：



全局查找

ctrl + h 或者通过 Edit 》Find 》Find In Path

全局替换

点击 Edit 》Find 》Replace In Path 或者设置一个快捷键

IDEA中设置Run Dashboard

微服务项目的开发过程中，工程会非常多，经常要启动很多个服务，才能完成一项测试。启动的多了，容易开发者带来错乱的感觉，很不方便管理。在idea作为开发工具时，推荐一个很好用的功能--Run Dashboard。

新版的idea :　viem -> Tool Windows -> Services -> + -> application. (win)

新版的idea :　viem -> Tool Windows -> Services -> + -> springboot. (mac)

雪花算法（id生成器）

一个tomcat，部署一套代码，使用雪花算法，每秒能够生成26万个不重复的id。

每个tomcat顶多也就几百并发，绰绰有余，对于12306和淘宝绰绰有余。

dao -> service -> controller

alt + 回车 + 回车 + 回车 实现类自动填充接口新方法

@RequestBody 前台传递过来的参数是json格式的数据

@ResponseBody 后台返回给前台的数据是json格式的数据

前台传递给后台参数有json字符串时，需要用@PostMapping

google 浏览器推荐安装的插件:

json转换插件

电脑如何分屏显示？

1、打开一个文档。

2、同时按下键盘上面win键和向左方向键（左方向键即占桌面左半侧，右方向键即占桌面右半侧）。

3、再打开一个窗口，同时按下win键盘上面win键和向右方向键，完成。

**遇到的问题：**

所有微服务抛出的异常都在一个全局异常处理里面统一处理，而不是在每个微服务里面都写一套异常的处理。

大体上的逻辑是，写入在一个公共服务，它不是一个独立的服务，没有启动类，也不需要注册到Eureka，它的作用是提供一些公用的组件、工具类和依赖等等，如Lombok包等.

主要是使用@ControllerAdvice + @ExceptionHandler来实现

**问题：其他的微服务使用不到公共服务里面的全局异常处理类**

原因： @ControllerAdvice注解的类，必须要让spring扫描到才能生效

启动类的扫描范围是启动类同级以及子级的包

因此可以把用户服务的启动类的配置做了如下更改：

@SpringBootApplication(scanBasePackages ="com.changgou" )

项目是maven项目，但看不到右侧maven目录：

找到pom.xml，右键 + add as maven project.

第二天：FastDFS

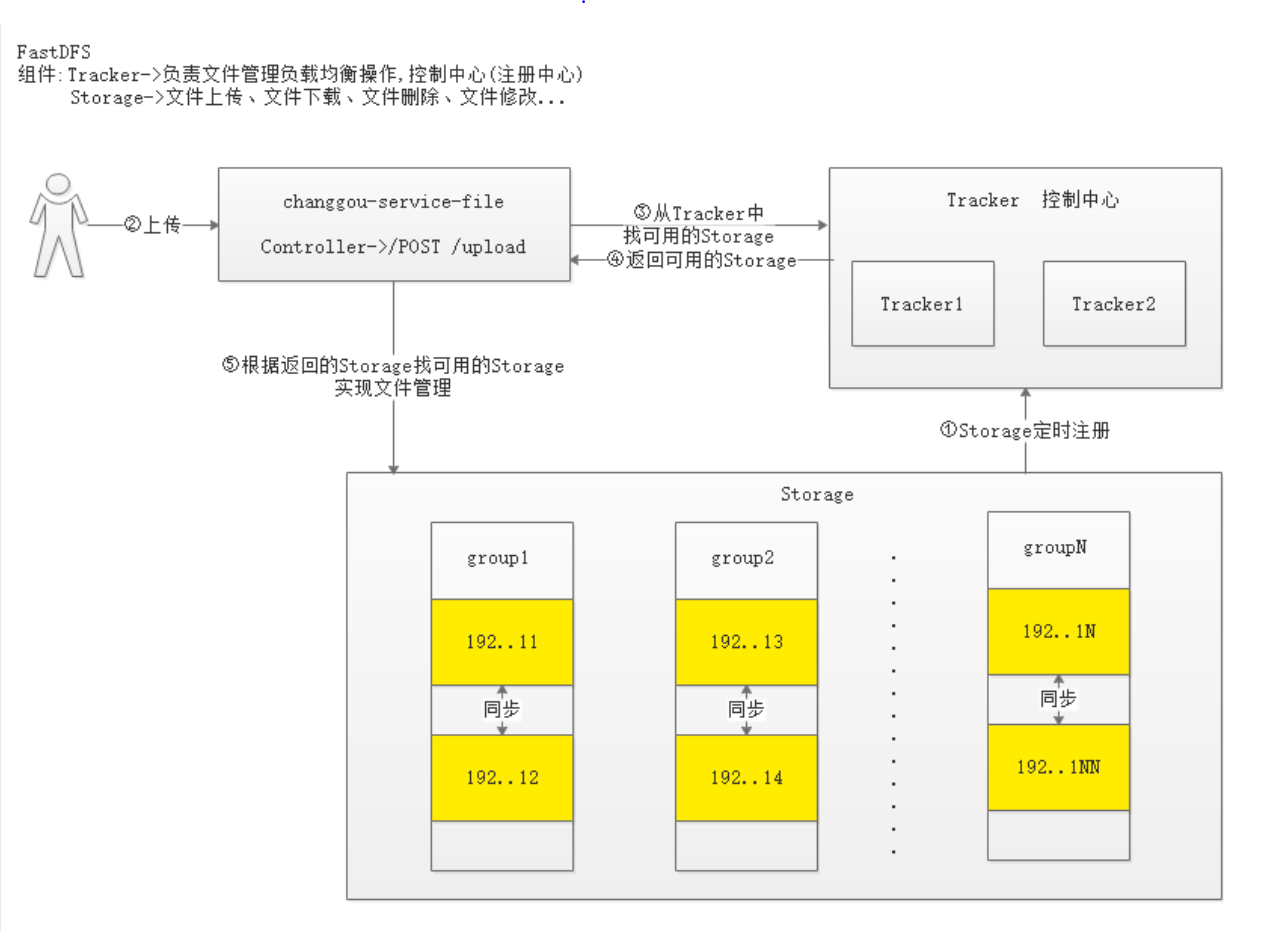
FastDFS是一个开源的轻量级分布式文件系统，它对文件进行管理，功能包括：文件存储、文件同步、文件访问（文件上传、文件下载）等。

FastDFS为互联网量身定制，充分考虑了冗余备份、负载均衡、线性扩容等机制，并注重高可用、高性能等指标，使用FastDFS很容易搭建一套高性能的文件服务器集群提供文件上传、下载等服务。

FastDFS 架构包括 Tracker server 和 Storage server。

Tracker server 作用是负载均衡和调度，通过 Tracker server 在文件上传时可以根据一些策略找到Storage server 提供文件上传服务。可以将 tracker 称为调度服务器。Storage server 作用是文件存储，客户端上传的文件最终存储在 Storage 服务器上，可以将storage称为存储服务器。

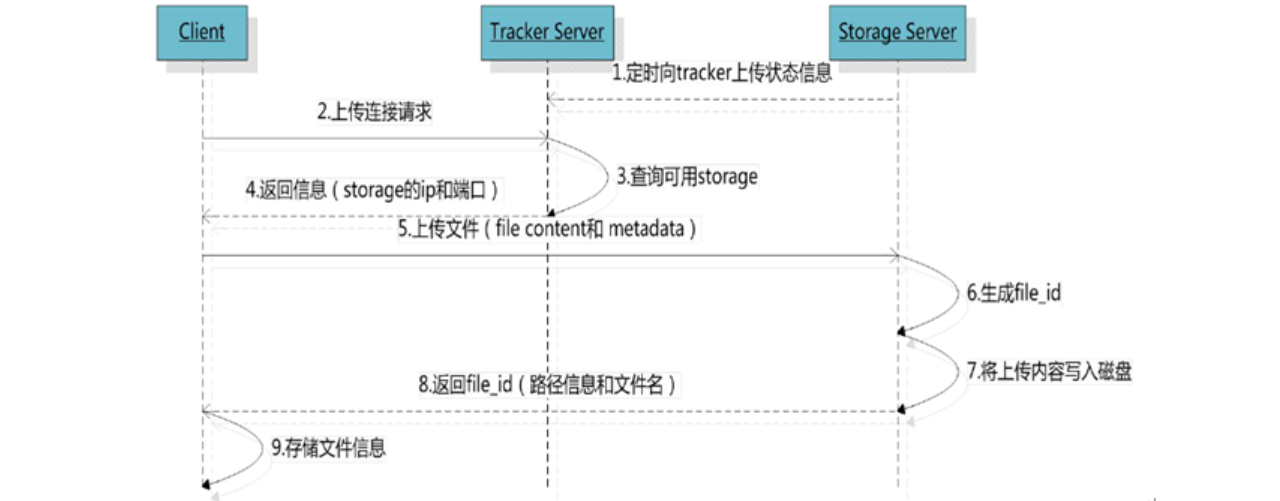
工作原理：



注意：每个组里面是容灾机制，是同步关系，不是主从。

多个组之间够成集群关系。

上传流程 :



客户端上传文件后,存储服务器将文件 ID 返回给客户端，此文件 ID 用于以后访问该文件的索引信息。文件索引信息包括：组名，虚拟磁盘路径，数据两级目录，文件名。



组名：文件上传后所在的 storage 组名称。

虚拟磁盘路径：storage 配置的虚拟路径。

数据两级目录：storage 服务器在每个虚拟磁盘路径下创建的两级目录，用于存储数据

文件。

文件名：与文件上传时不同。是由存储服务器根据特定信息生成，文件名包含：源存储

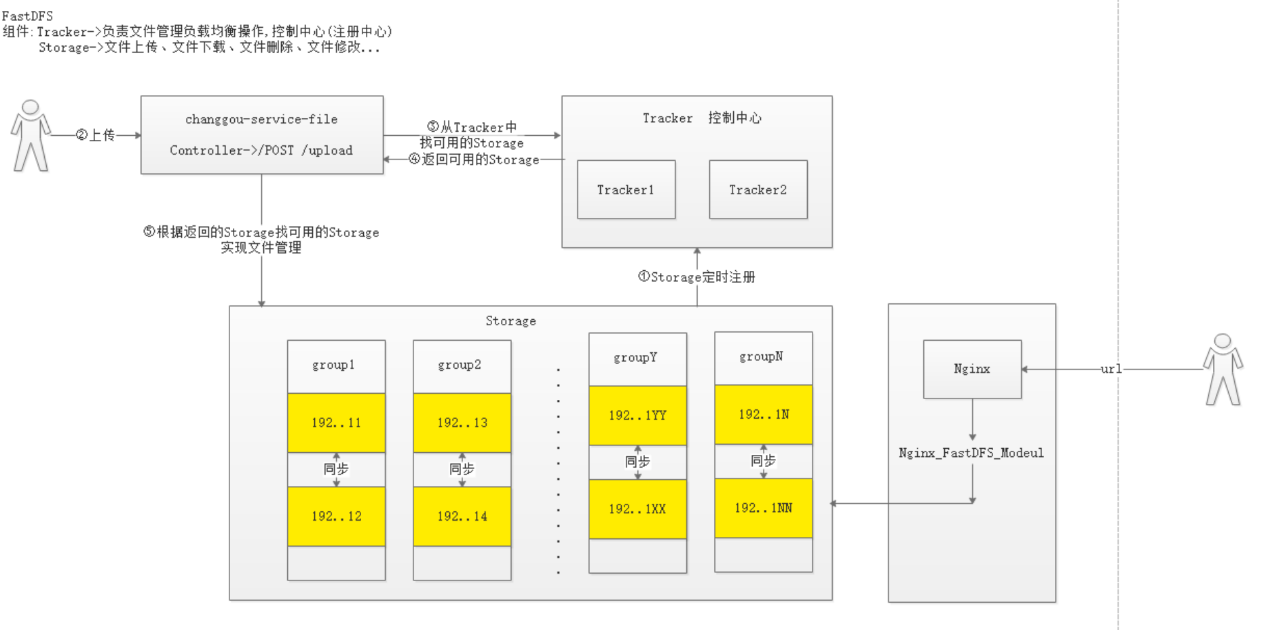
服务器 IP 地址、文件创建时间戳、文件大小、随机数和文件拓展名等信息。

访问 url :

http://ip:port/group1/ M00 /02/44 wkgDre34E8WaaaaaaaaaaGKFFSDFSDFFSD.JPG

Nginx访问图片流程：

这个nginx\_fastdfs\_model 模块是安装stograge和tracker的时候自动安装的，用户可以根据url去访问图片。



Dubbo的工作流程 : (FastDFS的原理类似Dubbo)

Diagram

Description automatically generated

启动FastDFS :

1. 拉取镜像

docker pull morunchang/fastdfs

1. 运行tracker

docker run -d --name tracker --net=host morunchang/fastdfs sh tracker.sh

1. 运行storage

docker run -d --name storage --net=host -e TRACKER\_IP=192.168.211.132:22122 -e GROUP\_NAME=group1 morunchang/fastdfs sh storage.sh

使用的网络模式是–net=host, 192.168.211.132是宿主机的IP

group1是组名，即storage的组

如果想要增加新的storage服务器，再次运行该命令，注意更换 新组名

1. 开启启动设置(linux 开机自启动)

docker update --restart=always tracker

docker update --restart=always storage

**遇到的问题 :**

**FastDFS中的图片调用api删除之后，刷新浏览器url获取图片，发现还是可以获取到，清空浏览器缓存，则成功获取不到.但不可能每次让用户清空浏览器缓存，怎么做？**

add\_header Cache-Control no-store;

添加到storage的nginx.conf里面的location里面，重启docker.

分类、规格参数、参数表关系：

选择分类，会涉及到一个模版id.

根据模版id，会涉及到一个规格参数.

除此之外，此模版可能还有一些额外的参数，存在参数表中. (通过模版id去search)

品牌管理

相册管理

规格参数管理

相关的表有3个 :

tb\_template 表（模板表）

tb\_spec 表（ 规格表）

tb\_para 表（参数表）

模板与规格是一对多关系 ，模板与参数是一对多关系

分类管理 (已完成)

第三天

**SPU与SKU概念理解**

SPU：通俗点讲，同款商品的公共属性抽取。

例如：\*\*华为P30 就是一个 SPU\*\*

表 : tb\_spu

SKU: 库存量单位, 是物理上不可分割的最小存货单元。

通俗点讲，即同款商品中，某类或者某个商品的独有属性。

例如：\*\*华为P30 红色 64G 就是一个 SKU\*\*

表 : tb\_sku

商品一般有3级分类，所有商品都发布在第3级分类下。

tb\_spu + tb\_sku = 商品表

1个分类对应多个品牌

1个品牌对应多个商品

1个分类对应一个规格参数模版

一个规格参数模版对应多个规格、多个参数

application.properties 中文乱码问题解决 ：

Graphical user interface

Description automatically generated

File -> Settings -> Editor -> File Encodings

将Properties Files (\*.properties)下的Default encoding for properties files设置为UTF-8，将Transparent native-to-ascii conversion前的勾选上。

mac 打开强制退出程序窗口 :

　　使用快捷键：Command+Option+Esc来打开“强制退出应用程序”的窗口，然后选中你需要退出的程序，再点右下方的“强制退出”即可。