上午课程核心内容

1. gitlab 环境和搭建

很多公司在项目开发时，为了项目代码的安全，不会选择在 github 或 gitee(码云) 上创建项目代码仓库，而是会先在公司的内网服务器上搭建一个 gitlab 服务，使用 gitlab 在公司内网服务器上创建代码仓库，进行项目源代码的管理。

gitlab 环境 docker 部署：

```
docker run -d --hostname gitlab --restart=always -p 7080:80 -p 2222:22 -p
8443:443 --name gitlab -v /data/gitlab/config:/etc/gitlab -v
/data/gitlab/logs:/var/log/gitlab -v /data/gitlab/data:/var/opt/gitlab
gitlab/gitlab-ce:latest
```

注意：**`gitlab环境占用内存比较多，启动比较缓慢，至少 3 分钟，多则 10 分钟以上`**.

2. Jenkins 对接 gitlab

2.1 Gitlab API Token 认证

jenkins 对接 gitlab 时，除了使用 ssh 认证和 用户名密码的认证方式，还可以使用一种专有的认证方式：**`Gitlab API Token 认证`**

具体配置：

1）在 gitlab 服务上生成 API Token

2）在 jenkins 服务中添加 Gitlab API Token 认证配置

2.2 Hook 触发配置

Hook 触发：通过 hook 触发的配置，可以实现当向指定仓库 git push 代码之后，自动触发对应 jenkins job 任务的执行。

具体配置：

1）安装 GitLab Hook 插件

2）在 jenkins job 配置页面选择 hook 触发方式

注：**需要在 job 配置页面生成 secret token，同时记下 secret token 和 jenkins job 任务的地址**.

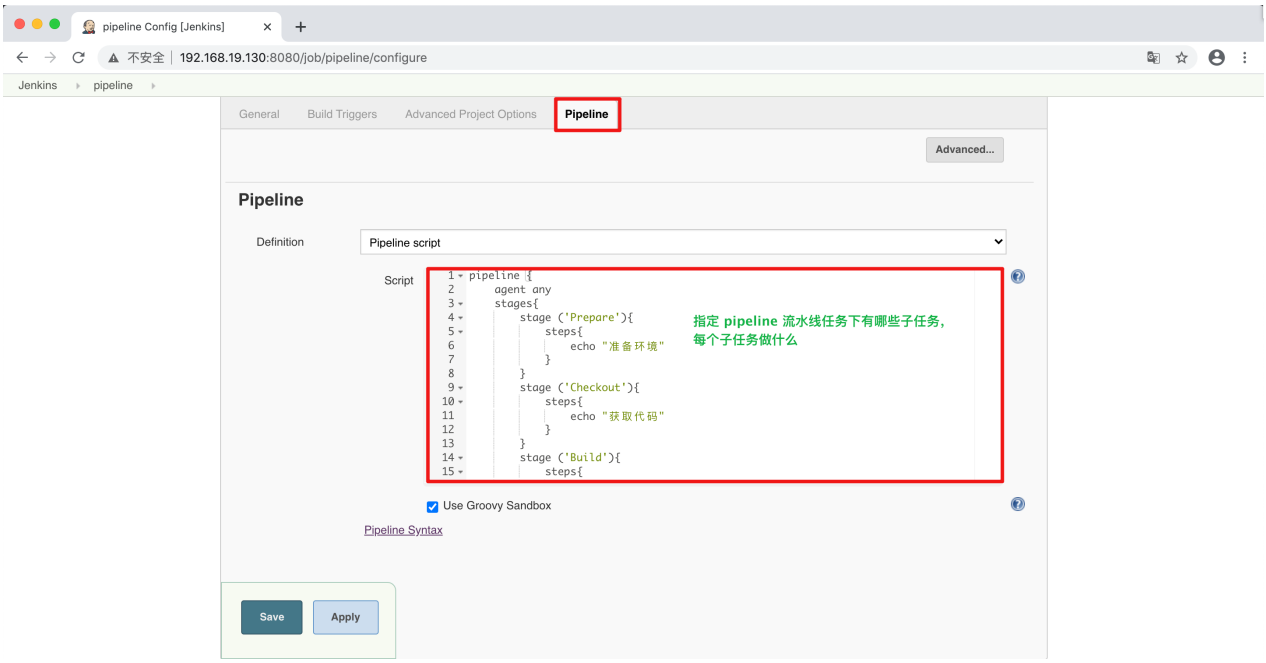
```
http://192.168.19.130:8080/project/gitlab-hook
04c1f748596bf8e8b08b5cf4189b2524
```

3）在 gitlab 对应代码仓库中进行配置，进行 hook 触发设置

2.3 Pipeline 流水线任务

之前创建演示都是 free style 风格的任务，Pipeline流水线风格的任务就是把一个大的任务划分成多个小的子任务，然后分别进行执行，方便查看任务执行到了哪个阶段，并且在出现问题之后，可以快速定位问题，流程编排与可视化。

插件：需要安装 Pipeline 插件



组成元素：

概念	说明
Stage：阶段	一个Pipeline可以划分为若干个Stage，每个Stage代表阶段任务中的一组操作 子任务
Node：节点	一个Node就是一个Jenkins节点，或者是Master或node，是任务执行时候的目标主机环境
Step：步骤	Step是最基本的操作单元，表示完成阶段任务的一系列的命令组成，它遵循jenkinsfile语法 子任务中的每一步骤

指定任务：

- 声明式语法

```
pipeline {
    agent any
    stages {
        stage ('子任务'){
            steps{
              # 具体每一步 step
              ...
            }
        },
        ...
        stage ('子任务'){
            steps{
            }
        }
    }
}
```

- 脚本式语法

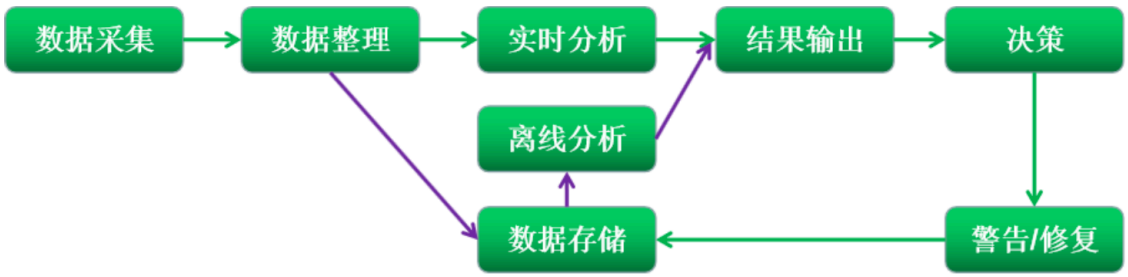
```
node ('工作节点'){
    stage '子任务1'
        任务1执行命令
    stage '子任务2'
        任务2执行命令
    stage '子任务3'
        任务3执行命令
    stage '子任务4'
        任务4执行命令
    stage '子任务5'
        任务5执行命令
}
```

BlueOcean插件：配合Pipeline，显示一个更加直观流水线任务流程。

3. ELK 简介

数据采集和分析需求：

运营阶段在整个项目的生命过程中，它的持续时间是最长的。怎么在长久持续运营的过程中，保证产品能够持续稳定的运行下去？？？需要收集项目运营过程中的产生的数据：日志、记录...，及时对一些数据进行分析，根据分析的结果来及时发现问题、优化流程。"数据采集和处理过程"



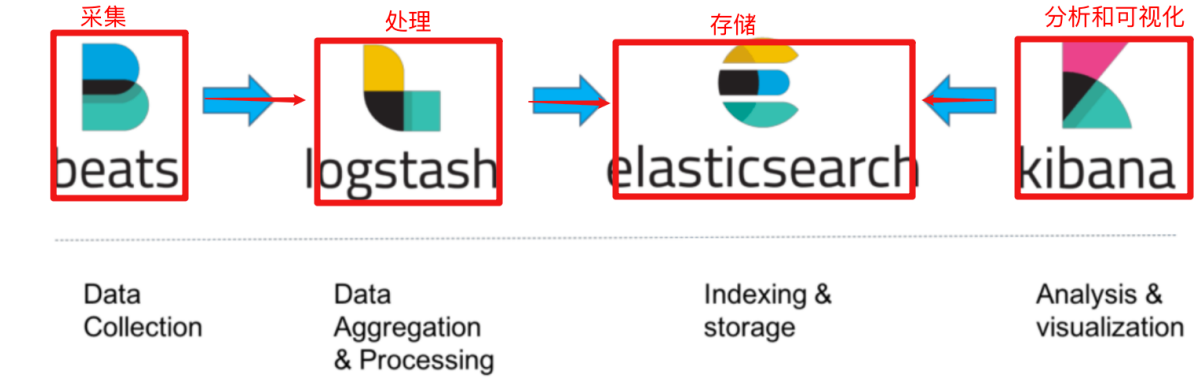
ELK 简介：

两三年前ELK还是一套日志分析平台的解决方案，但是随着 Elastic 公司发展，除了 ELK 这3个软件，它又引入了一些组件，目的就是把 ELK 打造成一个开源的数据分析解决方案。

- ElasticSearch：数据存储和搜索
- LogStash：数据处理
- Kibana：数据分析和结果可视化
- FileBeats：数据采集

经典组合：

对于小型的应用项目开发环境，ELK的四个组件可以实现一个非常经典的组合：



下午课程核心内容

1. ELK 环境搭建

在 192.168.19.130 和 192.168.19.131 虚拟机中已经搭建了 ELK 的一整套环境，直接使用即可，将来需要新搭建时可以参考讲义。

2. ElasticSearch 集群启动

作用：数据的存储和索引。

① 分别在 192.168.19.130 和 192.168.19.131 上启动 elasticsearch

```
# 切换账户
su - elastic
# 启动elasticsearch
elasticsearch -d
```

> 启动之后 elasticsearch 会监听 9200 和 9300 端口。

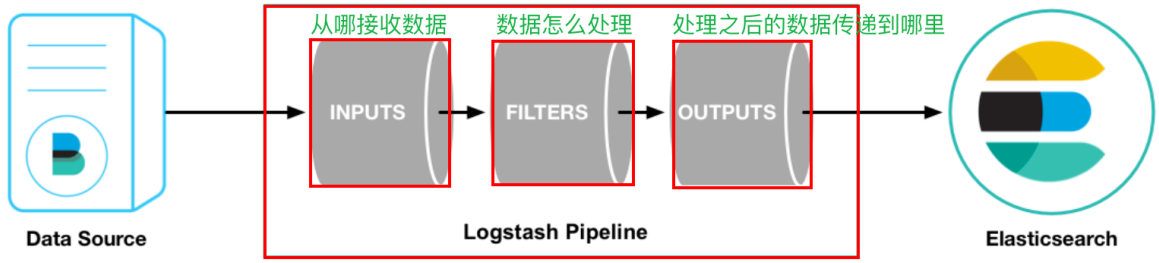
注意：192.168.19.130 虚拟机上启动 elasticsearch，需要先把原来 java 环境去做一个替换，替换为：java11

```
cd /data/softs/
tar -xf openjdk-11+28_linux-x64_bin.tar.gz -C /data/server/
cd /data/server/
rm java
ln -s jdk-11/ java
# 查询java版本
java -version
```

注意：192.168.19.131 虚拟机上启动 elasticsearch 之前需要先将 docker 中的 elasticsearch 关闭

```
# 否则会有端口口冲突
docker stop elasticsearch
```

3. Logstash 软件启动和使用



作用：从指定的数据源获取数据，对数据进行处理，然后将数据传输到指定的地方。

```
# 从屏幕接收数据，然后将数据显示到屏幕上
logstash -e 'input { stdin { } } output { stdout {} }'

# 从屏幕接收数据，然后将数据存储到 elasticsearch 的 message 索引中

logstash -e 'input { stdin{} } output { elasticsearch { hosts =>
["master.itcast.com:9200"] index => "message" } }'

# 使用配置文件启动
logstash -f 配置文件.conf
```

4. Kibana 软件启动

作用：从 elasticsearch 中获取数据，对于数据进行分析，并进行结果的可视化显示

启动：

```
# 切换到专有账号
su - kibana
# 启动：前台启动，方便终止
kibana
```

访问：192.168.19.131:5601

5. Filebeats 软件启动和使用

作用：从指定的地方采集数据、收集数据，传输到指定的地方。

使用：

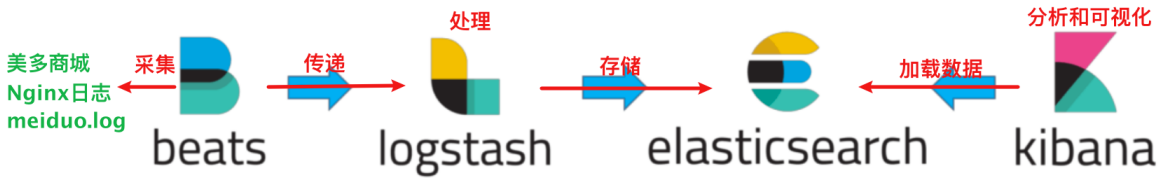
1）首先需要有个采集的配置文件：

```
filebeat.inputs:
- type: log
  enabled: true
  paths:
    - /var/log/*.log
output.elasticsearch:
  hosts: ["master.itcast.com:9200"]
  template.name: "filebeat"
```

2）启动Filebeats，进行数据采集

```
filebeat -e -c 采集配置文件
```

6. ELK - 简单实践案例



我们首先将美多商城项目的nginx日志信息输出到定制的目录 `/var/log/nginx/meiduo.log` 里面，然后接下来，我们就以meiduo_mall的日志数据为对象，使用filebeat来获取这些日志，将其输入到logstash中，logstash接收到数据后，定制显示格式，将其输入到elasticsearch中，kibana从elasticsearch中获取数据，并展示到当前界面。

实现过程：

第一步：配置美多商城的 Nginx 日志，输出到 /var/log/nginx/meiduo.log 文件中

```
# vi /etc/nginx/conf.d/meiduo.conf
```

```
upstream meiduo {
        # server 192.168.19.131:8001;
        # server 192.168.19.131:8002;

        server 192.168.19.130:8001;
        server 192.168.19.130:8002;
}

server {
        listen 80;
        server_name www.meiduo.site;

        # 添加access_log配置，配置将日志文件输出到：/var/log/nginx/meiduo.log
        access_log /var/log/nginx/meiduo.log;

        location = / {
                root /data/meiduo/front_page/;
                index index.html;
                try_files $uri $uri/ =404;
        }

        location /static {
                alias /data/meiduo/front_page/;
        }
```

第二步：配置 filebeat，从 meiduo.log 中采集数据，将采集的数据传递给 logstash

```
# vi /data/server/filebeat/meiduo.yml
```

```
filebeat.inputs:
- type: log
  enabled: true
  paths:
    - /var/log/nginx/meiduo.log
output.logstash:
  hosts: ["node.itcast.com:5044"]
```

第三步：配置 logstash，接收 filebeat 传递的数据，将数据存储到 elasticsearch

```
# vi /data/server/logstash/config/meiduo.conf
```

```
input {
    beats {
        port => 5044
    }
}
output{
    elasticsearch {
        hosts => ["master.itcast.com:9200"]
        index => "meiduo_mall-%{+YYYY.MM.dd}"
    }
}
```

第四步：启动 kibana，从 elasticsearch 加载美多商城日志的索引数据，进行展示