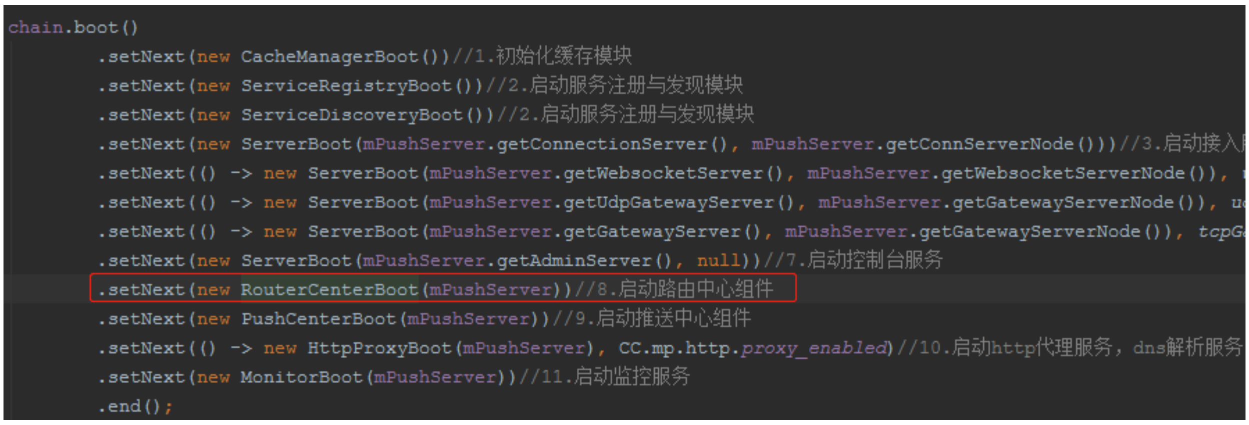
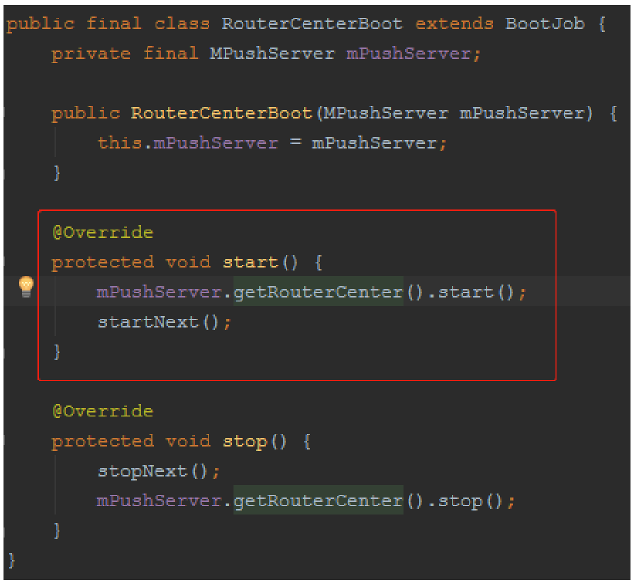
初始化路由服务



```java
chain.boot()
        .setNext(new CacheManagerBoot())//1.初始化缓存模块
        .setNext(new ServiceRegistryBoot())//2.启动服务注册与发现模块
        .setNext(new ServiceDiscoveryBoot())//2.启动服务注册与发现模块
        .setNext(new ServerBoot(mPushServer.getConnectionServer(), mPushServer.getConnServerNode()))//3.启动接入
        .setNext(() -> new ServerBoot(mPushServer.getWebsocketServer(), mPushServer.getWebsocketServerNode()),
        .setNext(() -> new ServerBoot(mPushServer.getUdpGatewayServer(), mPushServer.getGatewayServerNode()), u
        .setNext(() -> new ServerBoot(mPushServer.getGatewayServer(), mPushServer.getGatewayServerNode()), tcpG
        .setNext(new ServerBoot(mPushServer.getAdminServer(), null))//7.启动控制台服务
        .setNext(new RouterCenterBoot(mPushServer))//8.启动路由中心组件
        .setNext(new PushCenterBoot(mPushServer))//9.启动推送中心组件
        .setNext(() -> new HttpProxyBoot(mPushServer), CC.mp.http.proxy_enabled)//10.启动http代理服务，dns解析服务
        .setNext(new MonitorBoot(mPushServer))//11.启动监控服务
        .end();
```

启动服务



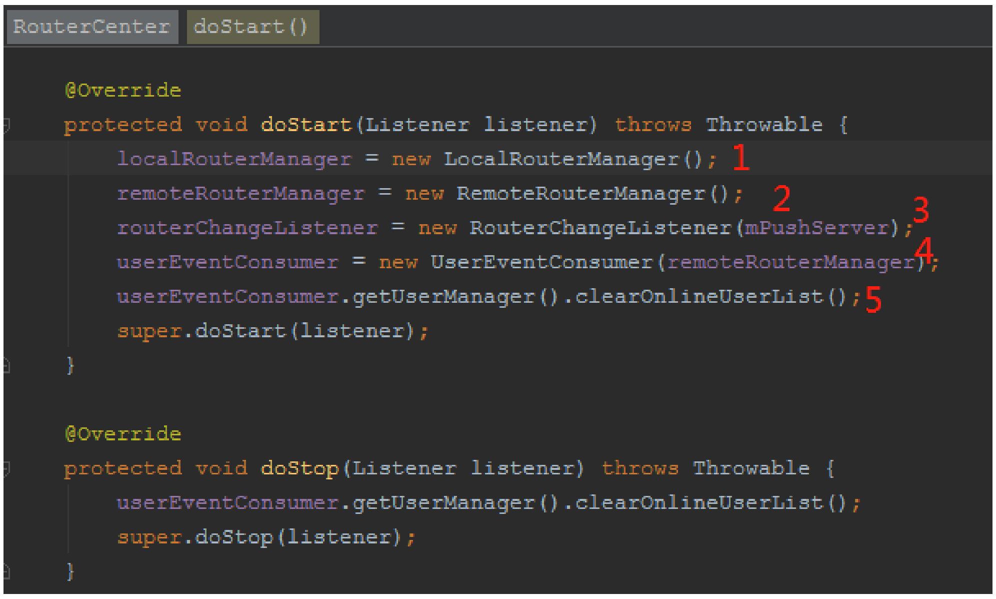
```java
public final class RouterCenterBoot extends BootJob {
    private final MPushServer mPushServer;

    public RouterCenterBoot(MPushServer mPushServer) {
        this.mPushServer = mPushServer;
    }

    @Override
    protected void start() {
        mPushServer.getRouterCenter().start();
        startNext();
    }

    @Override
    protected void stop() {
        stopNext();
        mPushServer.getRouterCenter().stop();
    }
}
```

调用RouterCenter->BaseService#start()，然后start()最终调用子类RouterCenter#dostar()



```java
RouterCenter  doStart()

    @Override
    protected void doStart(Listener listener) throws Throwable {
        localRouterManager = new LocalRouterManager(); 1
        remoteRouterManager = new RemoteRouterManager();  2
        routerChangeListener = new RouterChangeListener(mPushServer); 3
        userEventConsumer = new UserEventConsumer(remoteRouterManager); 4
        userEventConsumer.getUserManager().clearOnlineUserList(); 5
        super.doStart(listener);
    }

    @Override
    protected void doStop(Listener listener) throws Throwable {
        userEventConsumer.getUserManager().clearOnlineUserList();
        super.doStop(listener);
    }
```

1 初始化本地路由管理器

* 本地路由信息的添加、删除、获取；

* 订阅连接关闭事件ConnectionCloseEvent，删除本地路由，发布离线事件UserOfflineEvent；

2 初始化远程路由管理器

* 通过SPI，找到mpush-cache模块中CacheManagerFactory接口的实现类RedisCacheManagerFactory，得到RedisManager实例；
* 远程路由信息的添加、删除、获取；
* 订阅连接关闭事件ConnectionCloseEvent，将远程路由信息修改为离线(connId=null)；

3 初始化路由变更监听器

* 订阅路由变更事件RouterChangeEvent
* 根据路由类型(本地Or远程)，如果是本地路由，则发送踢人消息到客户端；否则广播踢人消息到MQ；
* 订阅MQ的广播消息，如果conn在本地机器，发送踢人消息到客户端；

4 初始化用户事件消费者

用户在线列表的key为 mp:oul:127.0.0.1

* 初始化UserManager

  踢人、清空在线用户列表、将用户添加到在线列表中、从在线列表中删除用户；

  统计在线用户数量、获取在线用户列表；

* 订阅用户在线事件UserOnlineEvent

  将用户添加到在线列表中；

  发布MQ在线消息ONLINE_CHANNEL给订阅方；

* 订阅用户离线事件UserOfflineEvent

  从在线列表中删除用户；

  发布MQ离线消息OFFLINE_CHANNEL给订阅方；

5 清除属于这台机器上的在线用户