Spring技术栈

2021年9月18日 21:32

Spring的两种技术栈

传统阻塞式Servlet技术栈

非阻塞式Reactive（响应式）技术栈

阻塞式Servlet技术栈

基于传统的Servlet的容器技术，利用Servlet来实现基于Web的服务器端的应用

具体实现

当客户端向服务器端发起请求的时候，服务器端针对**每一个请求开启一个线程**来处理这个请求

在这个线程里，会读取网络数据、读写数据库、做计算，然后产生返回给前端的结果并返回回去

整个过程都在一个独立的线程中完成。

当请求很多时，就会开辟很多线程，占用CPU和内存。一般我们会建立一个**线程池**，来限制能够同时处理的请求数。线程的数量受制于**内存大小、CPU资源**

这种技术之所以被称为阻塞式，是因为每一个请求中都会有一些慢速的操作，如读取网络数据、读取数据库，**与io有关的操作会使线程处于等待状态**，无法充分利用cpu的资源。因为我们定义了线程池来限制能同时处理请求的数量，这些处于**阻塞状态的线程就会使得新到来的请求无法得到及时的处理**，即使CPU

非阻塞式Reactive技术栈

采用与Servlet完全不同的想法来处理

具体实现

它会把整个任务分解成若干个小任务，如IO慢速任务和计算类高速任务

当一个请求发送到响应式的服务器时，不是用一个线程来处理这个请求，它会把IO这样的**慢速操作置于一个事件队列中，来侦听它是否有完成**，所以慢速操作并不会独占一个线程，从而让有限的资源被慢速操作阻塞。

在慢速操作完成了之后，才会**开启线程去执行高速的操作**

这样的方式其实是**基于事件驱动的思想**

Servlet技术栈和Reactive技术栈的不同

两种技术栈中采用的技术框架完全不同，如：

Servlet：

- 使用SpringMVC来实现服务器端的控制器，来处理Http的请求以及返回Http的响应
- 在逻辑层主要使用JDBC或JPA去访问数据库
- 基于Tomcat服务器

Reactive：

- 采用WebFlux来实现服务器端的控制器层
- 基于Netty这种异步的高速服务器
- 采用Mongo等来访问数据库

Servlet技术栈和Reactive技术栈的适用场合

两种技术栈并没有明显的优劣比较，在大多数情况下，我们使用Servlet技术栈，因为它比较符合我们传统的编程习惯

对于io操作频繁、高并发大负载的系统，Servlet技术栈会使得资源无法得到充分的利用，我们会采用响应式技术栈，如为服务器的网关，本身操作并不多，大多属于网络io