29 OpenResty开发入门

更新时间：2020-03-19 11:05:48

> 没有引发任何行动的思想都不是思想，而是梦想。 —— 马丁

前言

我们在上一篇文章中介绍了 OpenResty 的概念，并且带领大家安装了 OR，写了一个简单的 demo，体会了一把如何爽快的使用 Lua 实现我们的功能。
OR 有很多指令，我不会逐个指令的介绍，大家可以参考我在文章末尾给大家列出来的 参考教程，里面非常详细的列出来了各个指令的详细用法。
我想在本篇文章中介绍一下 OR 框架的工作原理，从高纬度上帮助大家理解它。

重要概念

我们介绍几个在 OR 框架中非常重要的几个概念，理解这些概念有助于我们理解整个框架。

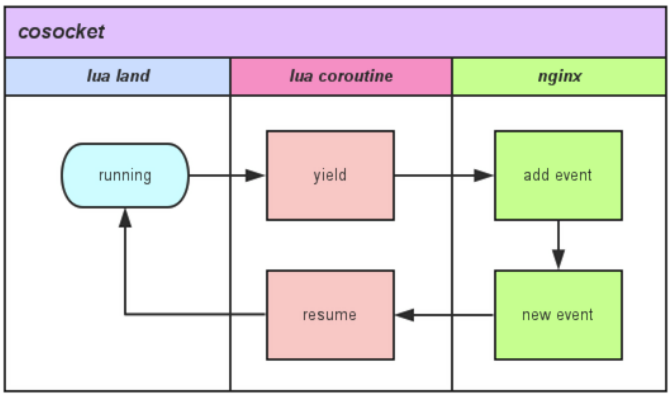
cosocket

这个概念非常的抽象，我在学习的过程中非常的懵逼，当时问了度娘，Google，收效甚微，后来随着自己在工作中的慢慢使用，有了一点理解，在这里分享给大家。

```
cosocket = coroutine + socket
coroutine：协程，go，python，lua中都有协程的概念
socket：套接字，TCP，UPD套接字
```

所以说cosocket就是使用协程实现的套接字管理功能。

是不是也很懵逼？理解这个概念关键在于理解Nginx的异步事件处理机制和Lua中的协程。



(上面的图片是OpenResty最佳实践中的图片)

这个图片非常准确的解释了cosocket的工作原理。

每个网络操作都是一个Lua协程，如果网络事件没有准备好，协程就可以调用yield阻塞自己，同时向Nginx注册一个事件，并将控制权交回给Nginx。等到事件满足的时候，Nginx会通知协程，此时协程调用resume就可以恢复运行。

Lua可以同时启动非常多的协程，每个协程都是一个网络事件，比如有的访问Redis,有的访问MySql，有的访问第三方服务等等，配合Nginx的异步事件处理机制，能够非常高效的完成功能。

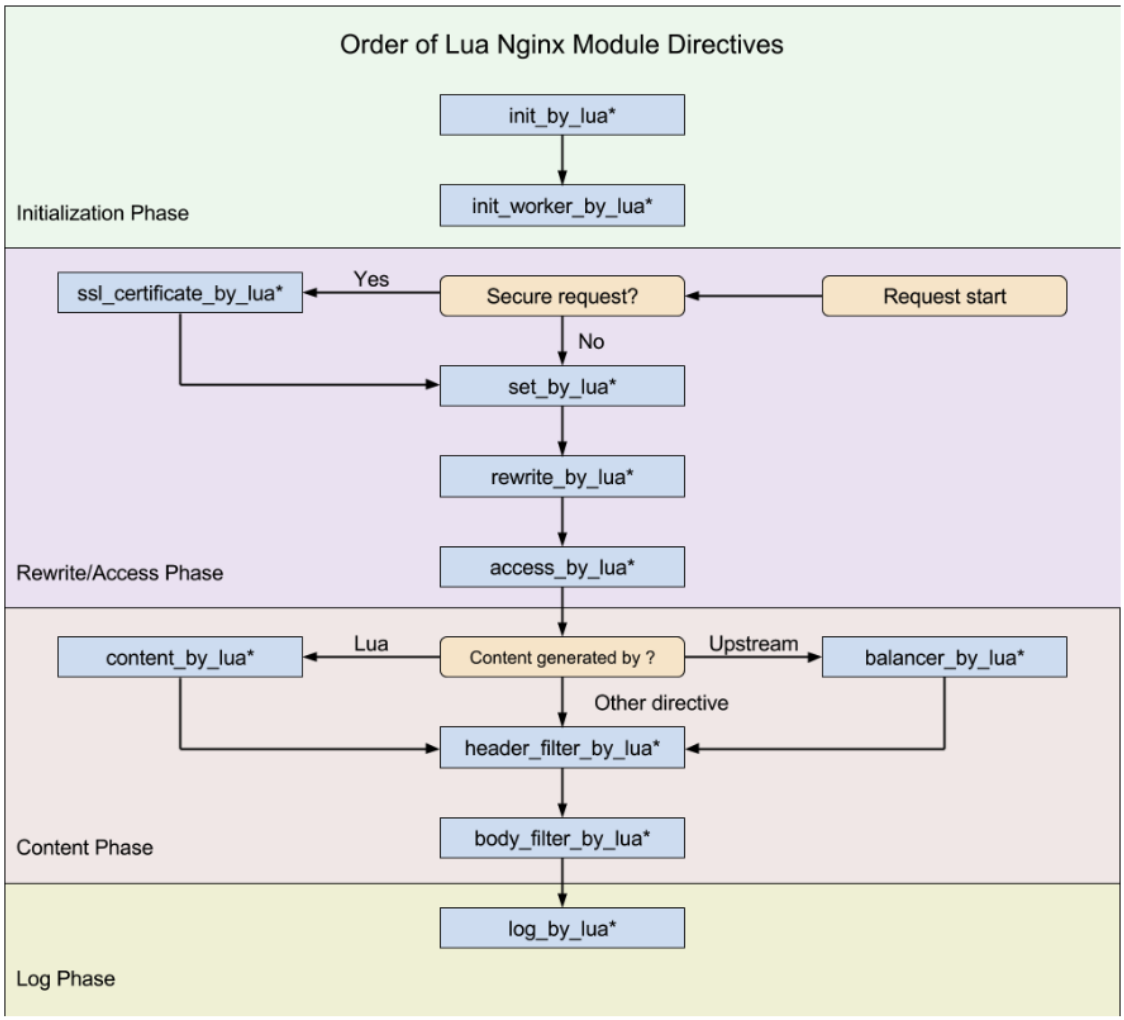
运行阶段

Nginx处理一个请求的过程其实是被分为了11个阶段，如下：

```
NGX_HTTP_POST_READ_PHASE
NGX_HTTP_POST_READ_PHASE
NGX_HTTP_POST_READ_PHASE
NGX_HTTP_REWRITE_PHASE
NGX_HTTP_POST_REWRITE_PHASE
NGX_HTTP_POST_REWRITE_PHASE
NGX_HTTP_ACCESS_PHASE
NGX_HTTP_POST_ACCESS_PHASE
NGX_HTTP_POST_ACCESS_PHASE
NGX_HTTP_CONTENT_PHASE
NGX_HTTP_LOG_PHASE
```

这种方式非常的好。就像工厂中的流水线一样，我们可以在每个阶段只处理自己感兴趣的部分，这样的话，每个部分实现的逻辑也很清晰，便于解耦。

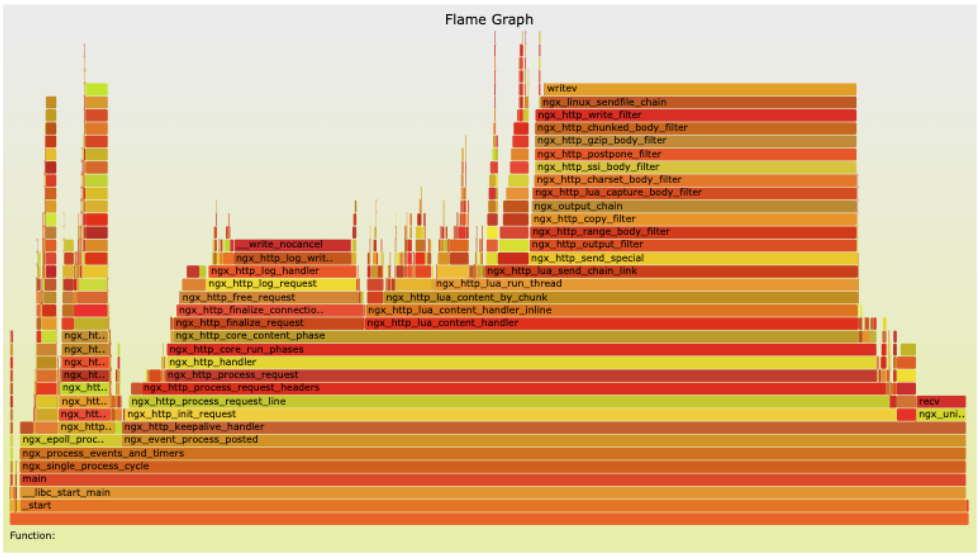
我们说过OR是一个第三方模块，那么它必然会介入到上面的某一个或者某几个阶段，我们看一下官方的介绍：



上图是官网(参考教程的第二个)上的介绍，OP介入到了Nginx中的REWRITE， ACCESS，CONTENT以及LOG这几个阶段，并且详细的画出了各类指令起作用的阶段，这是非常重要的，我们对照上图以及指令就可以深刻的理解指令起作用的阶段。

常用工具

如果我们在业务中遇到一些比较奇怪的问题，比如CPU的占用率和实际业务使用的CPU不符合，或者Nginx的吞吐率很低，这个时候应该怎么办呢？这里给大家介绍了一种定位问题的工具，叫做火焰图(形似火焰)。我在公司业务开发中使用的也是这个工具，非常的高效方便。



这个工具帮助我在工作中排查了很多的问题，大家要深入了解一下。

我在这篇文章中没有介绍什么干货，更多的是从一个比较高的维度给大家介绍了OpenResty的比较重要的一些内容。
个人觉得，这些东西是我在学习OR过程中遇到的比较费解的东西，我花费了大量的事件学习，但是当时也收效甚微的一些东西。希望对大家有所帮助。

参考教程

我在学习OpenResty的过程中，找过比较多的资料，做了对比之后，比较推荐两个教程供大家参考：

- 官网:先按照官网的Getting Started自己动手做一下
- github中OR项目
- OpenResty中文Api
- OpenResty最佳实践

}

28 OpenResty 的作用及环境搭建