开发模式：

windows 宿主机开发 + centos 虚拟机执行

> windows 和 centos 通过 共享文件夹进行文件同步

优点：

- 开发方便、工具丰富
- 运行环境，与真实环境相同

windows 和 centos 如何 通过 共享文件夹进行文件同步

windows有虚拟机的配置目录

centos 可以通过 配置的挂载目录，进行访问

共享文件夹，进行实施同步

Lua工程的开发工具：

idea 企业版 + lua 插件

Lua工程的模块结构：

图中所示的工程结构都处于工程的src目录下，包含了一下两大部分的内容：

- 第一部分为Nginx的配置，
- 第二部分为Lua脚本的目录结构
- 第三部分为静态文件目录

第一个大的部分Nginx的配置，可以进一步细分，包含了多块内容：

（1）Nginx的配置文件目录

（2）Nginx的调试日志目录

第二个大的部分是Lua脚本的目录结构。

Lua脚本统一放在了src/luaScript（名称自己定）目录下，luaScript目录结构，可以进一步细分，包含了多块内容：

（1）src/luaScript/initial目录，用于存放Lua程序初始化时需要加载的其他Lua脚本，比如mobdebug.lua调试脚本。

（2）src/luaScript/module目录，用于存放业务模块的Lua脚本，比如helloworld.lua。

（3）src/luaScript/redis用于存放操作redis的一些公共方法的代码，比如分布式锁Lock.lua。这里仅仅以redis为例，说明如果是一些耦合度较高的Lua模块，可以在src/luaScript目录下单独建一个子目录。

（4）其他的模块目录

第三个大的部分是静态js、html的目录结构。

Lua工程的配置文件注意要点

配置搜索目录 lua_package_path

Lua包路径，如果需要配置多个路径，路径之间使用分号";"分隔。末尾的两个双分号";;"，表示加上Nginx默认Lua包搜索路径，其中包含了Nginx的安装目录下的lua目录。

开启调试开关 lua_code_cache

```
#调试模式（即关闭lua脚本缓存)
lua_code_cache off;
```

Lua工程的运行脚本工具：

Lua脚本的模块化 开发

与Java类似，实际开发的Lua代码需要进行分模块开发。Lua中的一个模块对应于一个Lua脚本文件。使用require指令导入Lua模块，第一次导入模块后，所有Nginx进程全局共享模块的数据和代码，每个Worker进程需要时会得到此模块的一个副本，不需要重复导入，从而提高Lua应用的性能。接下来，演示开发一个简单的Lua模块，用来存放公共的基础对象和基础函数。

代码清单：src/luaScript/module/common/basic.lua

```
--定义一个应用程序公共的Lua对象app_info
```

```
local app_info = { version = "0.10" }
--增加一个path属性，保存Nginx进程所保存的Lua模块路径，包括conf文件配置的部分路径
app_info.path = package.path;


--局部函数,取得最大值
local function max(num1, num2)
    if (num1 > num2) then
        result = num1;
    else
        result = num2;
    end
    return result;
end
--统一的模块对象
local _Module = {
    app_info = app_info;
    max = max;
}
return _Module
```

创建一个Lua脚本src/luaScript/module/demo/helloworld.lua 来调用刚才所定义的这个基础模块src/luaScript/module/common/basic.lua文件。

helloworld.lua的代码如下：

```
// 代码清单：src/luaScript/module/demo/helloworld.lua

--- 启动调试
local mobdebug = require("luaScript.initial.mobdebug");
mobdebug.start();
--导入自定义的模块
local basic = require("luaScript.module.common.basic ");

--使用模块的成员属性
ngx.say("Lua path is: " .. basic.app_info.path);
ngx.say("<br>" );
--使用模块的成员方法
ngx.say("max 1 and 11 is: ".. basic.max(1,11) );
```

在使用require内置函数导入Lua模块时，对于多级目录下的模块，使用require("目录1.目录2.模块名")的形式进行加载，源目录之间的“/”斜杠分割符号改成“.”点号分隔符。这一点和Java的包名的分隔符很类似。