Swift源码编译

编译环境

- Xcode version 11.5 (big sur + Xcode 12.2: 5.3.1)
- Python 2.x
- brew install cmake ninja

编译步骤

第一步：clone swift源码

```
git clone --branch swift-5.2.4-RELEASE https://github.com/apple/swift.git
```

这里我编译的是`swift-5.2.4-Release`，因为在编译源码的时候，最新的就是 `5.2.4 RELEASE`.

如果想要编译最新的，大家可以自行在官网上找到分支，同时注意对应的`Xcode`版本要匹配（这个在官方文档编译`Swift`的时候会有说明）。目前找到的最新分支应该是`5.3.1-release`。

第二步：`update-checkout`

确保你当前的目录是在`swift-source`目录下，然后执行如下命令：

```
./swift/utils/update-checkout --tag swift-5.2.4-RELEASE --clone
```

这一步非常重要，因为 `update-checkout` 会 `clone` 编译 `swift` 相关的库，不然后面在编译 `swift` 源码的过程中一定会失败。

第三步：编译

编译过程中既可以使用`ninja`,也可以使用`Xcode`来进行编译。但是实际测试过程中 `Xcode`编译之后的支持性不是特别的好。所以这里就推荐使用`ninja`来作为编译工具

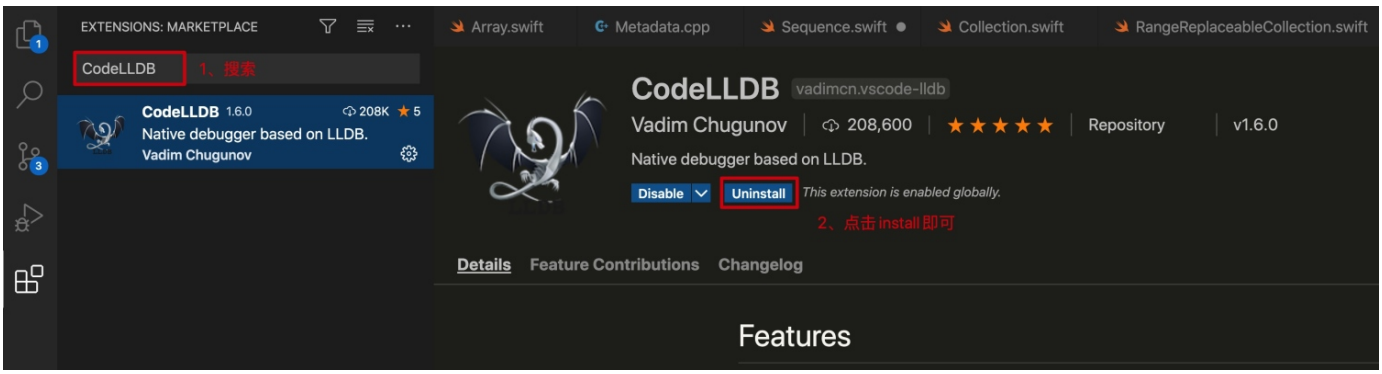
如果还有想要使用`Xcode`编译的，大家可以自己去官网文档当中来进行编译。

执行如下命令进行编译：

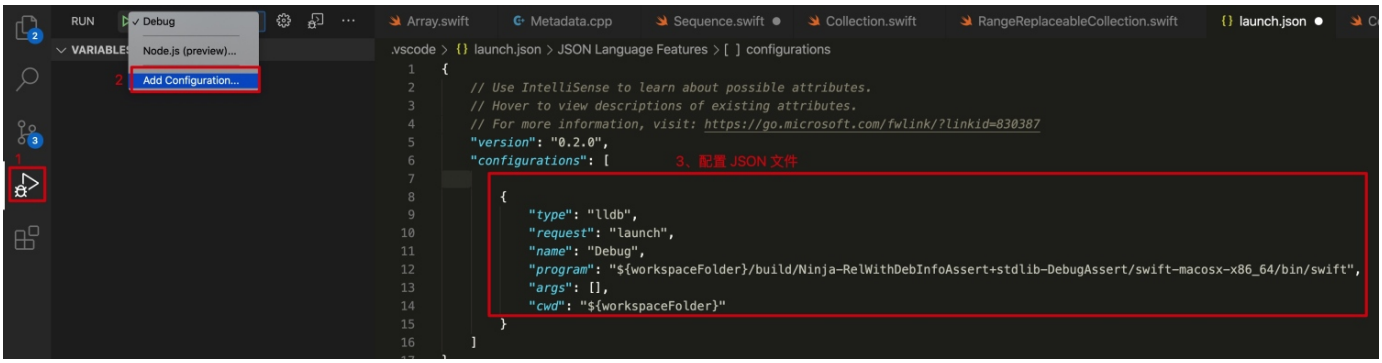
```
./swift/utils/build-script -r --debug-swift-stdlib --lldb
```

第四步：使用 `VSCode` 来调试 `Swift`。

首先我们需要安装一个插件`CodeLLDB`



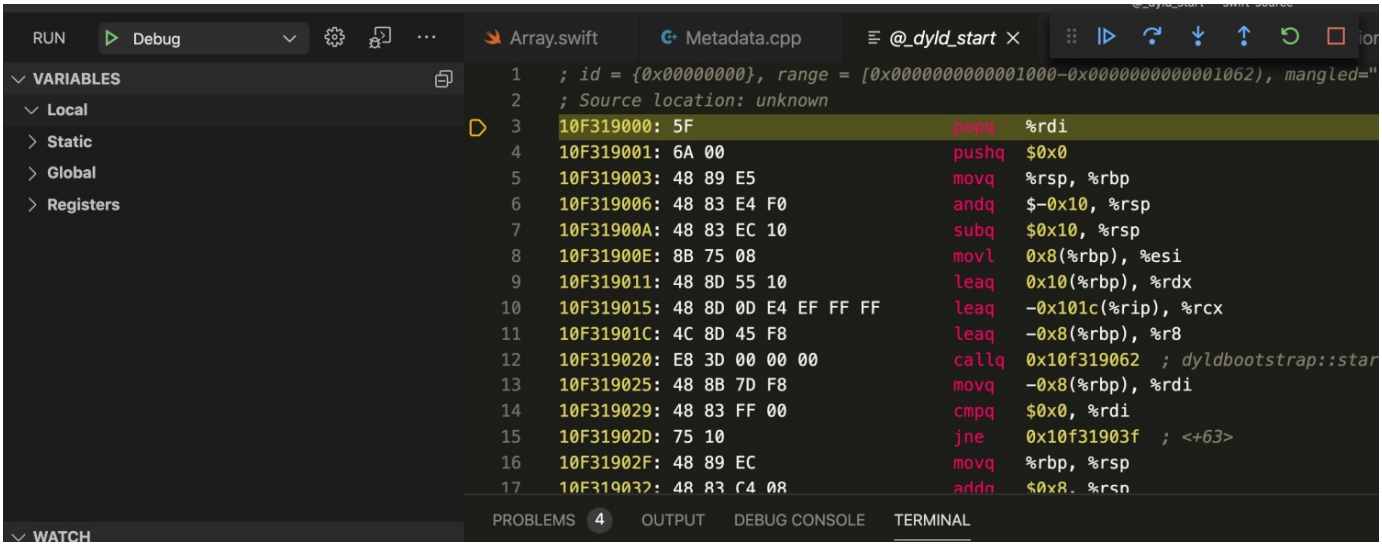
接下来配置 `JSON` 文件



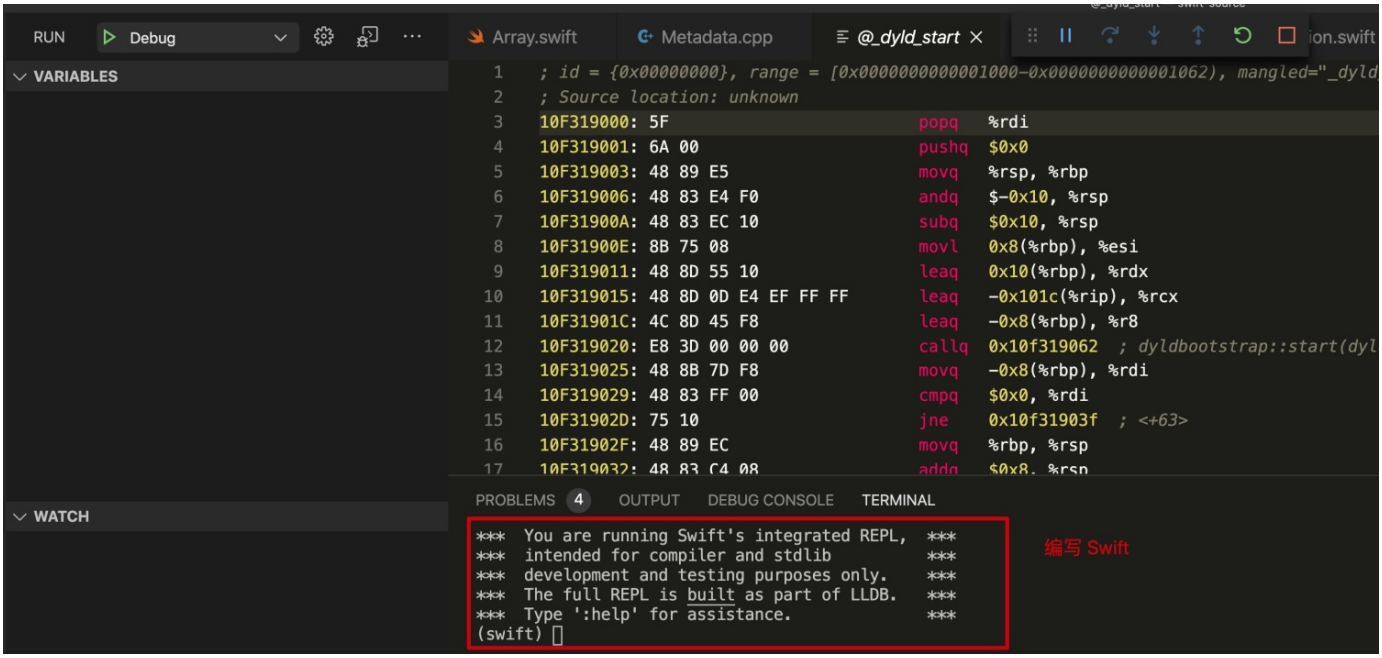
```
{
    "version": "0.2.0",
    "configurations": [

        {
            "type": "lldb",
            "request": "launch",
            "name": "Debug",
            "program": "${workspaceFolder}/build/Ninja-RelWithDebInfoAssert+stdlib-DebugAssert/swift-macosx-x86_64/bin/swift",
            "args": [],
            "cwd": "${workspaceFolder}"
        }
    ]
}
```

注意：上述 program 的文件路径和你编译的文件路径相同即可

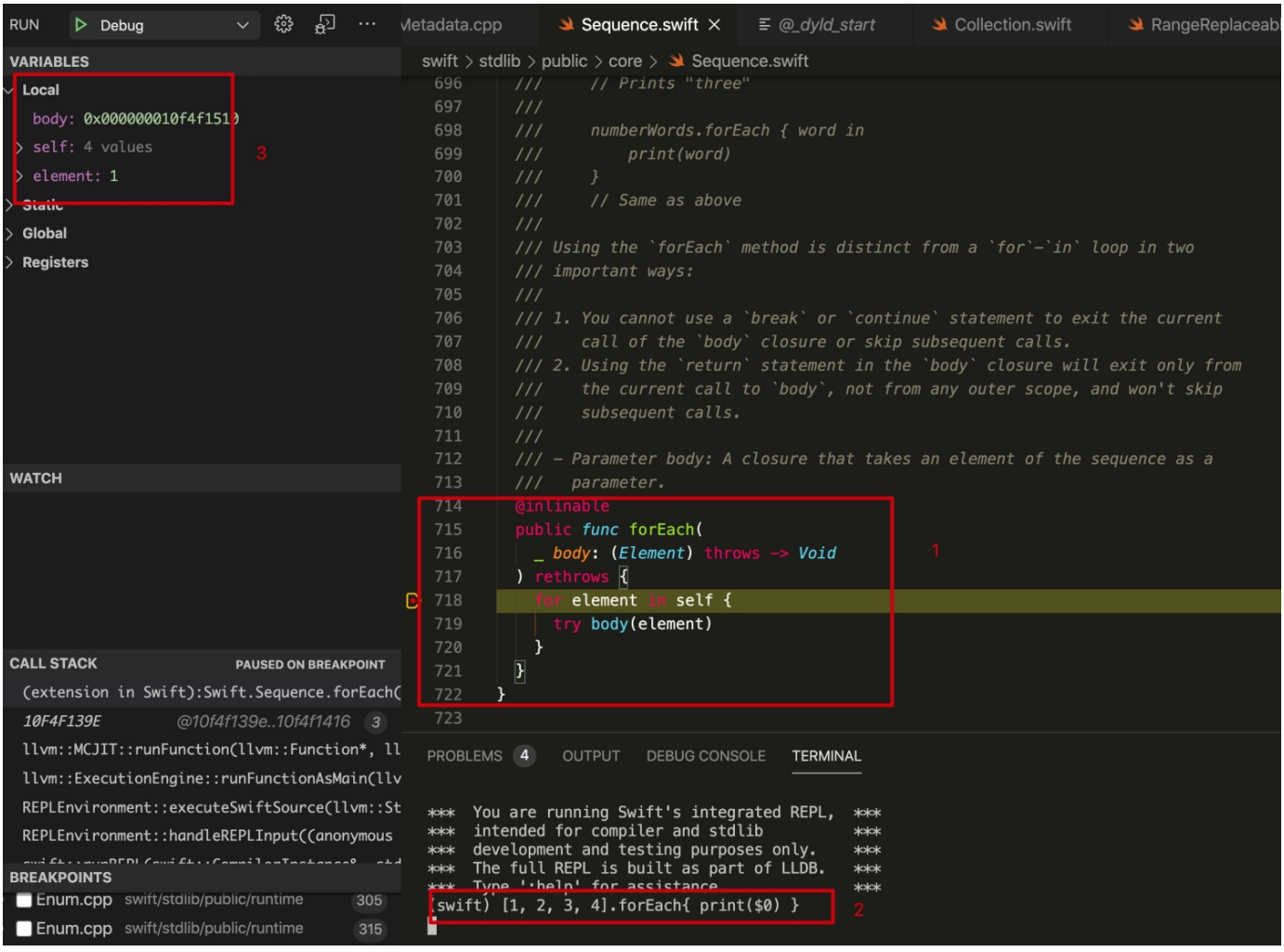
`run` 起来之后：



过掉断点之后：

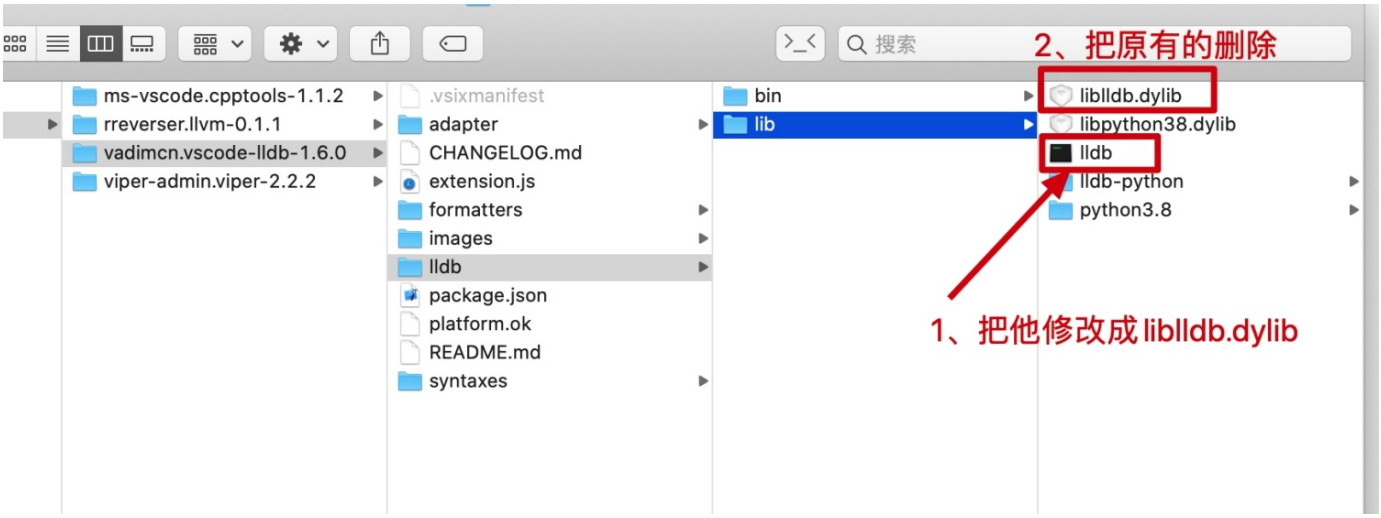


e.g:



在调试`.swift`文件的时候，可能区域 3 不显示，解决办法如下：

修改 `CodeLLDB` 的 `lib`文件下面的 `dylib` 文件



其中 `lldb` 的可执行文件在如下目录