

【How to】以子模块的方式添加OUI


由 earth 朱梦园 or 曾澳创建, 最后修改于一月 09, 2025

- [说明](#)
- [官方指南](#)
- [添加lua-eco子模块](#)
- [添加OUI子模块](#)
- [查看 lua-eco 和 oui 的 git 子仓库管理](#)
- [添加其他子模块流程](#)
- [子模块和主仓库“绑定”概念](#)
- [删除子模块](#)
- [使用 git submodule update --init --recursive](#)

说明


T750 方案默认的 web 管理界面是 Luci，由于后续我们将使用 OUI 作为新的web框架，所以需先在 T750 方案中随时部署 OUI。

官方指南

 请先阅读官方指南，大概对其有所了解

- <https://zhaojh329.github.io/oui/zh/>
- <https://github.com/zhaojh329/lua-eco>

添加lua-eco子模块

 lua-eco 依赖 libev -高性能事件循环库，确保存在 /feeds/packages/libs/libev库
lua-eco 的 gitlab 仓库为 http://192.168.10.16:9090/gitlab/giec_cpe/lua-eco.git

1、在 /t750-rg500leu-mt7915-source-code/目录下导入 lua-eco 子模块

```
cd t750-rg500leu-mt7915-source-code/  
git submodule add http://192.168.10.16:9090/gitlab/giec_cpe/lua-eco.git feeds/packages/lang/lua-eco
```

这里是将 lua-eco 作为 git 子模块添加到 git 主仓库中，但是实际 lua-eco 也存在 git 子仓库：log 和 ssl ，所以在/t750-rg500leu-mt7915-source-code/feeds/packages/lang/lua-eco 下执行

展开源码

2、在 /t750-rg500leu-mt7915-source-code/feeds/packages/lang/Lua-eco 目录下创建创建 Makefile

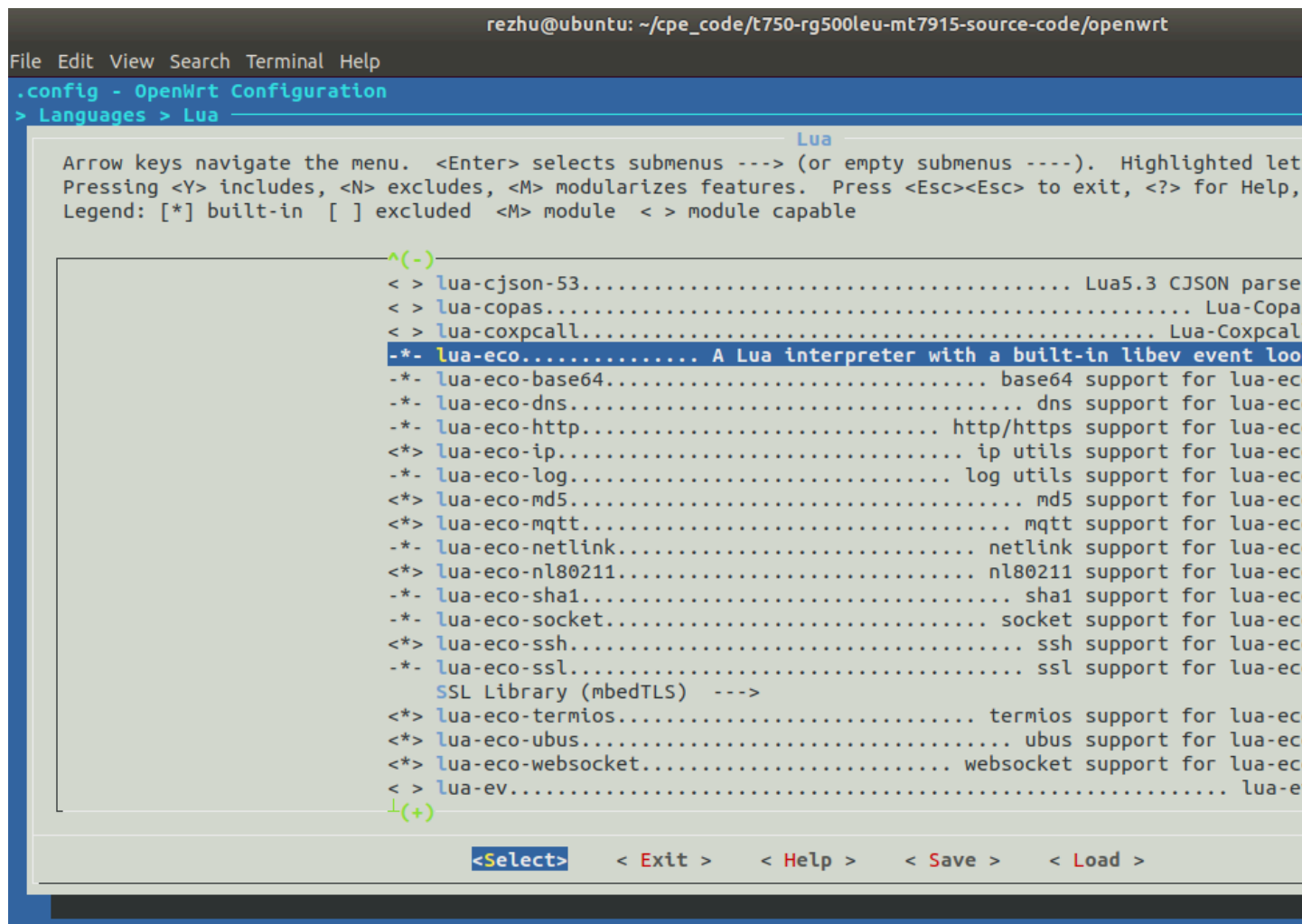
展开源码

随后在/t750-rg500leu-mt7915-source-code/openwrt执行

```
./scripts/feeds update -a  
./scripts/feeds install -a
```

```
make menuconfig
```

此时应该已经出现了 lua-eco



由于我们新增了Makefile，需要再子仓库提交一次，然后在主仓库提交一次

```
cd /t750-rg500leu-mt7915-source-code/feeds/packages/lang/lua-eco
git status
```

```
git add Makefile
git commit -m "Add Makefile"
```

```
cd /t750-rg500leu-mt7915-source-code/
git add feeds/packages/lang/lua-eco
git commit -m "XXX"
```

添加OUI子模块

使用下行命令从gitlab仓库中拉取 OUI 源码

```
git submodule add http://192.168.10.16:9090/gitlab/giec_cpe/oui.git feeds/oui
```

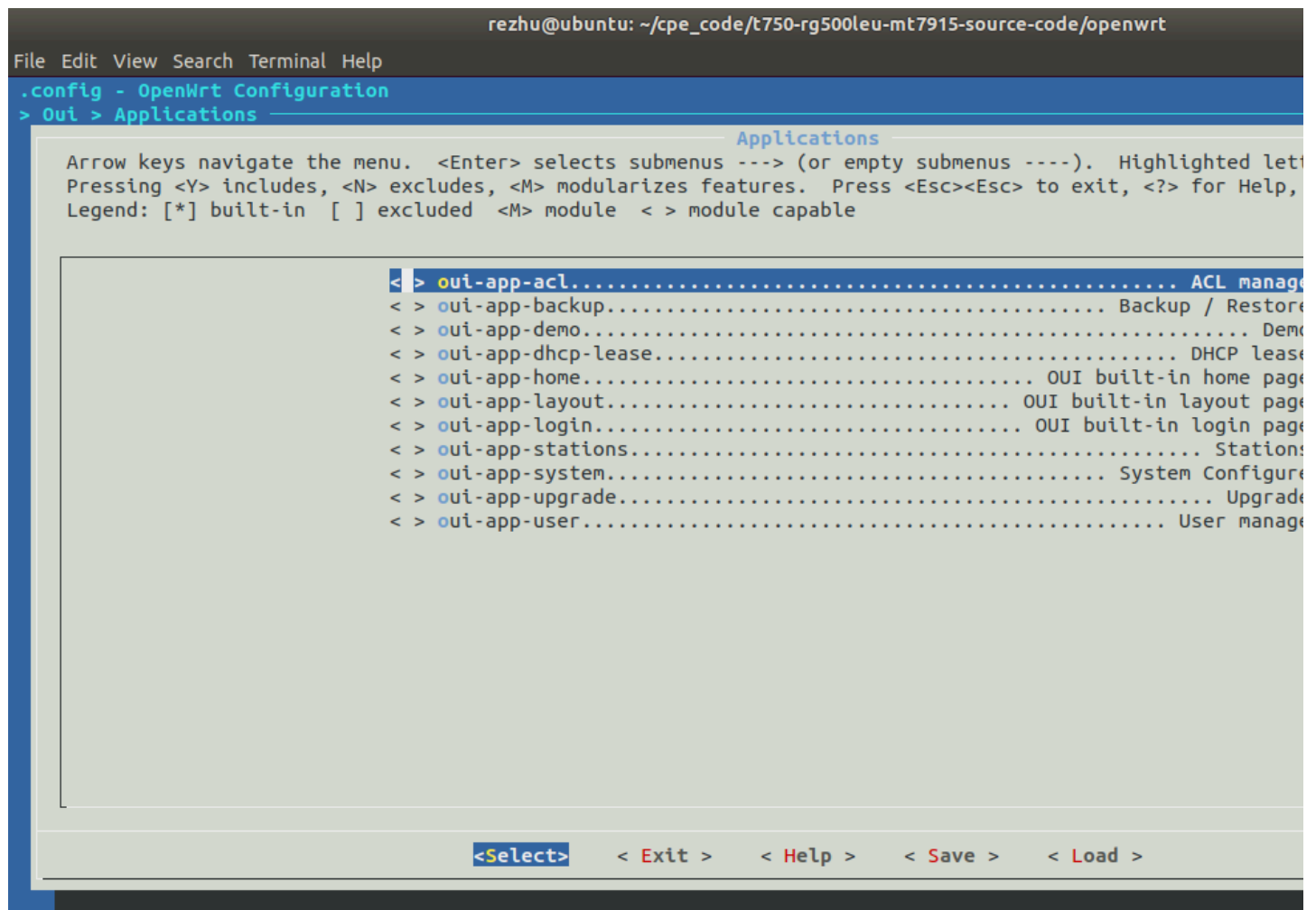
更改 /t750-rg500leu-mt7915-source-code/openwrt/feeds.conf.default 文件：

```
1
2 增加下面这行
src-link oui ../../feeds/oui
```

随后在 /t750-rg500leu-mt7915-source-code/openwrt 目录下执行

```
./scripts/feeds update -a
./scripts/feeds install -a
make menuconfig
```

此时已经可以看到 OUI 相关的 app



在编译过程中，可能会弹出与 node 相关的错误，若与版本号有关，参考

[【How to】在 Ubuntu 1804 安装高版本 node](#)

也可也在 oui-ui-core/Makefile 中给 node 命令增加 --verbose 参数，得到更多 log 输出

```
$(NPM) --prefix $(PKG_BUILD_DIR)/htdocs install --verbose
```

查看 lua-eco 和 oui 的 git 子仓库管理

上述操作将 lua-eco 和 oui 作为 git 主仓库的子仓库添加了过来，当执行完 `git submodule add <url> <path>` 后

在 /t750-rg500leu-mt7915-source-code 下会出现 .gitmodules 文件，该配置文件保存了项目 URL 与已经拉取的本地目录之间的映射。

该文件也像 .gitignore 文件一样受到主仓库的 git 版本控制。

```
cat .gitmodules
```

```
[submodule "feeds/oui"]
  path = feeds/oui
  url = http://192.168.10.16:9090/gitlab/giec_cpe/oui.git
[submodule "feeds/packages/lang/lua-eco"]
  path = feeds/packages/lang/lua-eco
  url = http://192.168.10.16:9090/gitlab/giec_cpe/lua-eco.git
```

```
git submodule status
```

```
628212e00cbcf723b798777d44e7ecb84dd8570e feeds/oui (heads/master)
bdca677c1870f001e4d82be479372373b7f5947f feeds/packages/lang/lua-eco (v3.5.2-4-gbdca677)
```

添加其他子模块流程

若需要将其他git仓库作为子模块添加，参考如下：

```
git submodule add <url> <path>
```

// 这条命令会在当前的git管理中添加一个子模块，此时 `git status` 会出现 `.gitmodules` 出现变动，此时 `<pa`

```
git submodule init
```

```
git submodule update
```

// 或者执行二合一命令：

```
git submodule update --init --recursive
```

// 此时将会从 `<url>` 拉取代码到指定 `<path>`

当创建了子模块后，`.gitmodules` 会出现变动，同时在 `.git/module` 目录下也会找到刚才添加的子模块，并且在 `.git/config` 中也可以查看到对应新添加的子模块

```
rezhu@ubuntu:~/cpe_code/T750-RG500LEU-MT7915$ cat .gitmodules
[submodule "feeds/packages/lang/lua-eco"]
    path = feeds/packages/lang/lua-eco
    url = http://192.168.10.16:9090/gitlab/giec_cpe/lua-eco.git
[submodule "feeds/oui"]
    path = feeds/oui
    url = http://192.168.10.16:9090/gitlab/giec_cpe/oui.git
rezhu@ubuntu:~/cpe_code/T750-RG500LEU-MT7915$ cat .git/config
[core]
    repositoryformatversion = 0
    filemode = true
    bare = false
    logallrefupdates = true
[remote "origin"]
    url = http://192.168.10.16:9090/gitlab/giec_cpe/T750-RG500LEU-MT7915.git
    fetch = +refs/heads/*:refs/remotes/origin/*
[branch "giec-base-oui"]
    remote = origin
    merge = refs/heads/giec-base-oui
[submodule "feeds/packages/lang/lua-eco"]
    url = http://192.168.10.16:9090/gitlab/giec_cpe/lua-eco.git
    active = true
[submodule "feeds/oui"]
    url = http://192.168.10.16:9090/gitlab/giec_cpe/oui.git
    active = true
```

子模块和主仓库“绑定”概念

对于主仓库来说，子模块是有可能在当前开发状态下多次更新的，为了避免混乱，**不要自动更新到最新的子模块代码，除非你明白自己在干什么**

原因如下：

子模块的提交状态是和主分支的提交存存在“**绑定**”关系的，在 Git 中，`git submodule init` 和 `git submodule update` 的作用是将子模块的代码同步到当前仓库的提交绑定记录（即子模块在当前父仓库提交中指定的特定提交）

```
git submodule init
```

作用：

- 从父仓库的 `.gitmodules` 文件中读取子模块的配置，并在 `.git/config` 中生成对应的子模块配置

行为：

- **不会真正克隆子模块代码，也不会检查出任何子模块的内容，仅初始化子模块的元数据**

```
git submodule update
```

作用：

- 将子模块的内容同步到当前父仓库绑定的提交。

行为：

- 如果子模块未克隆，则克隆子模块的代码仓库。
- 检出父仓库提交中记录的子模块特定提交（也就是指向的具体提交 SHA）
- **不会拉取子模块的最新代码，而是切换到父仓库指定的子模块提交记录**

所以若想进行子模块的更新，使用子模块的项目必须手动更新才能包含最新的提交:

```
cd /子模块目录下
git pull
```

随后在主仓库下，可以使用 git status 查看：

```
rezhu@ubuntu:~/cpe_code/T750-RG500LEU-MT7915$ git status
On branch 30-feature-online-user-display-package
Your branch is up to date with 'origin/30-feature-online-user-display-package'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   feeds/giec/packages/ota_update/config/ota_config
        modified:   feeds/oui (new commits)
```

可见其中会显示出子模块的“new commits”，若本次开发需要用到正是该版本的子模块，在主仓库 commit 时，请 add 这个“new commits”的子仓库，随后将这个子模块的更新一并提交。

此时就已经建立了“绑定”关系，既当前主仓库的 commit 与 子模块的 commit “绑定”，随后在任何时刻，即便子模块有了新的更新，但是只要主仓库处于这个 commit 下，执行

```
git submodule init
git submodule update
```

不会拉取最新的子模块，而是定位到“绑定”好的子模块的 commit 上

```
rezhu@ubuntu:~/cpe_code/T750-RG500LEU-MT7915$ git submodule init
rezhu@ubuntu:~/cpe_code/T750-RG500LEU-MT7915$ git submodule update
Submodule path 'feeds/oui': checked out '84f230a57fa6e50381e07a0fbad94ce32d8c0a32
rezhu@ubuntu:~/cpe_code/T750-RG500LEU-MT7915$ cd feeds/oui/
rezhu@ubuntu:~/cpe_code/T750-RG500LEU-MT7915/feeds/oui$ git branch
* (HEAD detached at 84f230a)
  giec_dev
rezhu@ubuntu:~/cpe_code/T750-RG500LEU-MT7915/feeds/oui$
```

在gitlab中也有显示：

giec-base-oui ▾


T750-RG500LEU-MT7915 / feeds /

+

▾

History

Find file

Name	Last commit
..	
giec/packages	[GK1801] 更改无产测数据时WiFi的默认密码
luci	add luci-app-onliner luci-app-arpcbind
mtk	add luci-app-onliner luci-app-arpcbind
packages	[GK1801] 修复了子模块初始化异常的问题
quectel	Restore files after moving .git directory
 oui @ 84f230a5	[GK1801] 修复了子模块初始化异常的问题

(这就是所谓的“绑定”)

若确定开发是需要最新的提交，则需要进入到子模块路径中，手动拉取最新提交，并且在最后主仓库 commit 时选中子模块的 "new commit"，这将重新定向子模块的指向。

```
rezhu@ubuntu:~/cpe_code/T750-RG500LEU-MT7915$ git status
g0n branch 30-feature-online-user-display-package
Your branch is up to date with 'origin/30-feature-online-user-display-package'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   feeds/giec/packages/ota_update/config/ota_config
        modified:   feeds/oui (new commits)
```

```
rezhu@ubuntu:~/cpe_code/T750-RG500LEU-MT7915$ git add feeds/oui/
rezhu@ubuntu:~/cpe_code/T750-RG500LEU-MT7915$ git status
On branch 30-feature-online-user-display-package
Your branch is up to date with 'origin/30-feature-online-user-display-package'.

Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   feeds/oui
```

由于主仓库与子模块存在一一对应的“绑定”关系，故需要确保子仓库存在，若子仓库由于某些原因删除了提交记录，会导致主分支无法找到对应的子模块提交记录，从而报错：

```
fatal: remote error: upload-pack: not our ref 2e91c0d11b41368ce8eee7c941c2b7c97689d4a8
fatal: The remote end hung up unexpectedly
Fetched in submodule path 'packages/lang/lua-eco', but it did not contain 2e91c0d11b41368ce8eee7c941c2b7c97689d4a8. Direct fetching of th
at commit failed.
```

解决方法：删除子模块，解除指向，确定子模块仓库正常，然后重新添加子模块并提交（相当于替换子模块）

删除子模块

有时子模块的项目维护地址发生了变化，或者需要替换子模块，就需要删除原有的子模块。

删除子模块较简单，和添加刚好相反，步骤如下：

1. rm -rf 子模块目录 删除子模块目录及源码
2. vim .gitmodules 删除项目目录下.gitmodules文件中子模块相关条目
3. vim .git/config 删除配置项中子模块相关条目
4. rm .git/module/* 删除模块下的子模块目录，每个子模块对应一个目录，注意只删除对应的子模块目录即可

使用 git submodule update --init --recursive

在前面指出

```
git submodule init
git submodule update

git submodule update --init --recursive
```

这两个命令等价，这里并不严谨，因为后者**具有额外递归操作**：如果子模块中还有嵌套的子模块（即子模块的子模块），也会自动初始化和更新

如果没有嵌套的子模块，git submodule init/update 和单独执行 git submodule update --init --recursive 才是等价的，因此请**直接使用 git submodule update --init --recursive**

