**云笔代码版本规范**

**说明**

本文描述了在云笔技术研发过程中，对源码的管理方式。

**版本工具**

使用：GIT

**版本号**

使用四位版本号，其中版本号规范如下：

1.0.0.0

其中，从左到右，依次称为：一位版本号，二位版本号，三位版本号，四位版本号。

**一位版本号**

从1开始计数，描述系统发生大的架构变更时，进行+1。如：系统架构更新、业务架构更新。

**二位版本号**

从0开始计数，描述系统发生模块级别的变更时，进行+1。如用户子模块变更，增加支付功能等。

**三位版本号(发布的最小版本)**

从0开始计数，通常每次拉取一个版本时，使用的版本号，作为平常的如：bug更新、小功能增加等。

**四位版本号**

从0开始计数，每一次的提交测试，使用的版本号。

**版本地址**

目前，公司大规模地使用了Coding的服务（[http://coding.net](http://coding.net/)）。

**隐私及权限**

对于公司内部的项目，如果属于通用级别的，代码级别为公开，即所有人都可以看到并可以直接fork。属于业务级别的，则为私有。

**公司项目地址**

<https://coding.net/u/yunbix>

**代码开发**

版本规范如下：

1. master：开发版本
2. tag：发布版本

**版本的开发**

代码开发过程中，开发主要在master下进行，开发完成后，则直接进行代码tag操作，根据版本号规范，生成新的tag即可。

假设，我们在代码的master下进行开发：

`git clone <https://git.coding.net/yunbix/someproject.git> localworkplace`

**版本打包**

开发完成第一个测试版本时，进行打tag操作：

`git tag v1.0.0.0 -m "提交测试"`

测试进行测试，使用的也是这个版本来进行测试：

git pull origin v1.0.0.0

git checkout v1.0.0.0

即可获取到对应版本的代码。

如果在开发过程中，存在需要并行开发的情况，那么这个时候，就需要对代码进行分支操作了：

git branch v1.0.1

git checkout v1.0.0

开发完成后，如果需要发布，那么首先需要将代码合并入master：

git checkout master

git merge v1.0.1

顺利的话，这里就完成了。如果出现了代码冲突，那么解决对应的冲突即可。