http://www.php-internals.com/book/?p=chapt01/01-02-code-structure

第一章 准备工作和背景知识

**千里之行，始于足下。**

在开始进入PHP的内核实现之前，需要做一些准备工作，也需要了解一些背景知识。本章主要涉及PHP源码的获取， PHP源码的编译，从而得到我们的调试环境。

接下来，我们将简单描述PHP源码的结构以及在\*nix环境和Windows环境下如何阅读源码。 最后我们介绍在阅读PHP源码过程中经常会遇到的一些语句。

如果你没有接触过PHP，或者对PHP的历史不太了解，我们推荐你先移步[百度百科 PHP](http://baike.baidu.com/view/99.htm)， 这里有PHP非常详细的历史介绍，它包括PHP的诞生，PHP的发展，PHP的应用，PHP现有三大版本的介绍以及对于PHP6的展望等。

## 第一节 环境搭建

在开始学习PHP实现之前，我们需要一个实验和学习的环境。下面介绍一下怎样在\*nix环境下准备和搭建PHP环境。

(\*nix指的是类Unix环境，比如各种Linux发行版，FreeBSD， OpenSolaris， Mac OS X等操作系统)

### 1.获取PHP源码

为了学习PHP的实现，首先需要下载PHP的源代码。下载源码首选是去[PHP官方网站http://php.net/downloads.php](http://php.net/downloads.php)下载， 如果你喜欢使用svn/git等版本控制软件，也可以使用svn/git来获取最新的源代码。

# git 官方地址

git clone https://git.php.net/repository/php-src.git

# 也可以访问github官方镜像

git clone git://github.com/php/php-src.git

cd php-src && git checkout PHP-5.3 # 签出5.3分支

笔者比较喜欢用版本控制软件签出代码，这样做的好处是能看到PHP每次修改的内容及日志信息， 如果自己修改了其中的某些内容也能快速的查看到，如果你想修复PHP的某个Bug或者提交新功能的话， 有版本控制也会容易的多，更多信息可以参考附录：[怎样为PHP做贡献](http://www.php-internals.com/book/?p=D-how-to-contribute)。

目前PHP已经[迁移到Git](http://www.php.net/archive/2012.php" \l "id2012-03-20-1)了，PHP的wiki上有关于 [迁移到Git的说明](https://wiki.php.net/vcs/gitfaq)，以及[使用Git的流程](https://wiki.php.net/vcs/gitfaq)   
~~在笔者编写这些内容的时候PHP版本控制是还基于SVN的，上面提到的github镜像地址目前已经没有同步更新了， 由于把svn同步到git会对系统性能造成明显影响，加上社区还没有就到底是否迁移到git达成一致，所以也就停止了更新。 目前很多开源软件都开始转向了分布式版本控制系统(~~[~~DVCS~~](http://en.wikipedia.org/wiki/Distributed_revision_control)~~)， 例如Python语言在转向DVCS时对目前的分布式版本控制系统做了一个[详细的对比](http://www.python.org/dev/peps/pep-0374/)， 如果以前没有接触过，笔者强烈建议试试这些版本控制软件。~~现在Github的同步是实时的。 所以习惯Github的话，基本上可以把Github当做官方版本库了。

### 2.准备编译环境

在\*nix环境下，需要安装编译构建环境。如果你用的是Ubuntu或者是用apt做为包管理的系统，可以通过如下命令快速安装：

sudo apt-get install build-essential

如果你使用的是Mac OS X，则需要安装Xcode。Xcode可以在Mac OS X的安装盘中找到，如果你有Apple ID的话， 也可以登陆苹果开发者网站<http://developer.apple.com/>下载。

如果你不愿意下载庞大的Xcode，也可以去<https://github.com/kennethreitz/osx-gcc-installer>下载安装包， 只安装所需的命令行工具。

### 3. 编译

下一步可以开始编译了，本文只简单介绍基本的编译过程，不包含Apache的PHP支持以及Mysql等模块的编译。 相关资料请自行查阅相关文档。 如果你是从svn/git签出的代码则需要执行代码根目录的buildconf脚本以生成所需要的构建脚本。

cd ~/php-src

./buildconf

执行完以后就可以开始configure了，configure有很多的参数，比如指定安装目录，是否开启相关模块等选项：

有的系统自带的autoconf程序版本会有Bug，可能导致扩展的配置无法更新，如果在执行./buildconf时 报错，可以根据出错信息安装合适版本的autoconf工具。

./configure --help # 查看可用参数

为了尽快得到可以测试的环境，我们仅编译一个最精简的PHP。通过执行 ./configure --disable-all来进行配置。 以后如果需要其他功能可以重新编译。如果configure命令出现错误，可能是缺少PHP所依赖的库，各个系统的环境可能不一样。 出现错误可根据出错信息上网搜索。 直到完成configure。configure完成后我们就可以开始编译了。

./configure --disable-all

make

在\*nix下编译过程序的读者应该都熟悉经典的configure make，make install吧。执行make之后是否需要make install就取决于你了。 如果install的话最好在configure的时候是用prefix参数指定安装目录， 不建议安装到系统目录， 避免和系统原有的PHP版本冲突。 在make 完以后，在sapi/cli目录里就已经有了php的可以执行文件. 执行一下命令：

./sapi/cli/php -v

-v参数表示输出版本号，如果命令执行完后看到输出php版本信息则说明编译成功。 如果是make install的话可以执行$prefix/bin/php这个路径的php。 当然如果是安装在系统目录或者你的prefix目录在$PATH环境变量里的话，直接执行php就行了。

在只进行make而不make install时，只是编译为可执行二进制文件，所以在终端下执行的php-cli所在路径就是php-src/sapi/cli/php。

后续的学习中可能会需要重复configure make 或者 make && make install 这几个步骤。

### Windows环境的编译

Windows环境下的编译可以参考官方Wiki的说明：<https://wiki.php.net/internals/windows/stepbystepbuild>

### 推荐书籍和参考

* [Gnu Make中文书册](http://www.yayu.org/book/gnu_make/index.html)
* 《Autotools A Practioner's Guide》