用最短的时间掌握并使用一门从未接触的编程语言用于实际项目开发的要诀

虽说是快速掌握，但也必须是系统的学习，绝不能是大致了解或者走马观花。其实对于学习一门新的编程语言来讲，也是有最佳实践的，只要遵照本教程的方法练习，你一定可以在一周内掌握并使用一门新的编程语言开始写项目。但如果要把语言学到精通的程度，除了下面的训练方法，还需要一个真实需求的项目来锻炼自己。

第一部分：学习全新编程语言所需的准备工作（学习过程30分钟，环境准备1小时-1天）

1. 要学习一门新的计算机语言，先确定这个语言是否是图灵完备的，理论上来讲任何一个编程语言只要是图灵完备的，就可以实现一切可编程工作，比如php、java、c语言等，这些都是图灵完备的，因此java可以做的事情c语言也可以做，php能做的事情，java也能做到。但注意，Linux的shell脚本编程就不是图灵完备的，因此很多通过编程语言完成的事情通过shell脚本是无法做到的，类似HTML或者XML这类标记语言虽然能表达语意，但编写这类标记语言连编程都算不上。
2. 如果确定要学习的计算机语言是图灵完备的，那么它也会是可替代的，一定有其他语言也能完成和它同样的事情，而一门计算机语言被创造并且流行起来一定是因为这个语言存在某一方面的优势。所以一定要弄清楚在已经存在其他编程语言的条件下，这个编程语言为什么还会被创造出来，它有哪些优势，适用于哪些场景，又有哪些不足。学习多语言通常有两个大

的优势，一个是可以使用适合的技术解决合适的问题；另一个就是当一个社区里面存在某些技术创新，或者已经有了某种场景的解决方案后，我们可以借鉴这个社区的思想来使用到自己的项目中，哪怕我们使用的是其他编程语言，这样的例子有很多，比如ruby社区中的Rails就在软件工程，项目管理，架构设计上远远领先其他语言的同类框架，因此在php社区中出现了采用同样思想的laravel框架，python社区中也出现了同样设计理念的django框架。

3. 搞定生产、测试、与产品环境，选择称手的开发工具，编写运行简单的程序，如果有条件，跑一下开源项目，至此，第一部分准备工作结束。

第二部分：程序逻辑（学习过程20-30分钟）

由于到这一阶段的学员在之前的训练中已经通过了程序逻辑训练，并且已经有过使用诸如C语言或者javascript语言编写程序的经历，因此已经具备了程序逻辑思想，懂得像计算机一样思考。在上述前提条件下，工程师通常可以在二十分钟内快速弄清楚这门新的编程语言的程序逻辑与法。（主要指 顺序 循环 分支 跳转等语句）

第三部分：变量是程序设计的基础，正确理解编程语言的变量类型才能得心应手的工作，反之不能理解变量则编写程序会非常容易出错（学习过程30分钟，手册阅读2小时，字符串函数训练4小时）

1. 区分变量是静态类型还是动态类型，是强类型还是弱类型
2. 基本类型和封装类型
3. 传递引用还是传递值
4. 时间日期类型的实现和使用。注意：时间日期在软件开发中十分重要（时间日期支持的函数可作为手册查询训练任务）

5. 字符串的实现方式和种类（比如字符数组组成字符串，比如字符串常量池），字符串是可变的还是不可变的。注：此处可增加训练任务，通过阅读和查询手册，全面而系统的了解全部，注意是全部编程语言内置字符串函数，以及正则表达式函数，配合作业任务，至少要做到系统性，语言哪些功能已经内置我得知道，不用背下来，需要可查询，而针对常用的函数，要能够在IDE的提示环境下，不查询手册的情况下直接使用。

第四部分：容器对象学习，通常是指数组，列表，字典等复杂对象类型。任何编程语言要面向实际业务需求完成编码工作，都需要有容器类型的参与。很多早期的编程语言，比如C，C+ +，语言本身不提供容器对象（事实上图灵完备的语言可以通过自己编程实现任何自己所需的容器），语言本身没有提供容器对象虽然不影响这门编程语言成为一个强大的语言，但会严重影响开发效率，而且应对如此常见的需求，不同公司不同程序员如果都去自己编写自己的容器实现，除了浪费大量开发资源外，还会导致程序员之间，甚至组织间代码复用性差，不同项目难以兼容等诸多问题。因此大部分现代化编程语言都内置了容器类型，c+ +这类语言虽然没有内置复杂的容器对象，但是开源的C+ +标准库却提供了比一般语言内置的更为丰富的能力（学习时间20-40分钟，训练时间1-2小时）

1. 容器对象的种类与实现原理
2. 容器的遍历
3. 容器深浅复制
4. 容器的序列化与反序列化
5. 查阅手册，了解全部容器支持的函数并掌握常用容器操作函数

第五部分：编程语言实现抽象的方法、类、对象、接口与函数的实现、使用（学习时间40-100分钟）

1. 函数的定义与调用
2. 函数的参数使用与传递方式（是否有特殊规则如默认参数，多参同名函数，函数运算符重载等）
3. 类和对象的实现方式，语言是否内置面向对象的设计模式支持（没错，面向对象也只是一种设计模式，和之前提到的容器类型一样，不是编程语言必须的，但是因为非常常用，所以部分编程语言内置了，但并不妨碍我们在没有内置面向对象的编程语言中实现面向对象这种设计模式，也就是 封装 继承 多态）
4. 是多继承还是接口，接口是否需要显示声明（提前声明）
5. 常见的设计模式实现，如单例模式，代理模式（委托），观察者模式

第六部分：函数式编程的支持是现代化编程语言的标志，甚至一些古老的面向对象编程语言都开始支持函数式编程，比如java、php（学习时间30分钟）

1. 高阶函数对象
2. 与法特性与编程习惯（如链式与法，闭包等）

第七部分：依赖与项目管理工具与生态支持，如java的maven，gradle，php的composer，ios的CocoaPods，js的npm等（学习时间30分钟）

第八部分：内置高级功能，通常现代化编程语言都支持某些高级功能，比如并发模型，多线程，多进程，系统调用，磁盘文件管理、数据库操作，网络支持等特性，通常在使用一门语言的初期不需要全部精通，但实际用到编程语言某些能力时，需要能随时查阅手册并快速掌握高级功能编写，要达到这一点，至少你应该全面阅读过手册，知道有哪些功能是语言内置可用的。（快速预览编程语言全部模块文档约2小时）