linux系统编程-应用编程常识--专题讲座

written by 王保明

## 计算机系统组成

### 1计算机系统硬件组成

|  |
| --- |
| * 计算机系统由软件硬件构成 |
| 总线   * 贯穿整个系统的一组电子管道称为总线 * 片内总线 * 系统总线   + 数据总线DB   + 地址总线CB   + 控制总线CB * 外部总线 |
| IO设备   * I/O设备是系统与外界联系的通道 * 键盘鼠标是输入设备 * 显式器是输出设备 * 磁盘既是输入设备也是输出设备 * 输入输出是相对于内存来说的。 |
| 内存   * 内存是一个重要的部件，它是与CPU进行沟通的桥梁。它用来存放程序以及程序要处理的数据，磁盘中的程序要加载到内存才能运行。   处理器 （运算器+控制器）   * 中央处理器（CPU），简称处理器。 * CPU主要有运算器、控制器、寄存器构成 * 取指 * 译码 * 执行 * 写回 * 跳转 |
| 计算机系统硬件组成 |
| 总结：  1、cpu操作内存步骤 通过地址总线写地址、写控制（控制总线）、操作内存（数据总线）  2、硬件角度看hello程序 |
| DMA 传输将数据从一个地址空间复制到另外一个地址空间。当 CPU 初始化这个传输动作，传输动作本身是由 DMA 控制器 来实行和完成。典型的例子就是移动一个外存的区块到芯片内部更快的内存区。像是这样的操作并没有让处理器工作拖延，反而可以被重新安排去处理其他的工作。在实现DMA传输时，是由DMA控制器直接掌管总线，因此，存在着一个总线控制权转移问题。 |

### 2操作系统

|  |
| --- |
| 什么是操作系统   * 操作系统：有效地管理计算机系统中的资源，合理地管理计算机系统的工作流程，方便用户使用的程序的集合。 * 资源   + 软件资源   + 硬件资源 * 操作系统五大任务   + 文件管理   + 处理器管理 cpu 96   + 内存管理   + 设备管理   + 作业管理 |
| 操作系统三个基本抽象   * 操作系统通过三个基本抽象概念来实现这两个功能。（进程、虚拟存储器和文件）。文件是对I/O设备的抽象表示，虚拟存储器是对主存和磁盘I/O设备的抽象表示，进程则是对处理器、主存和I/O设备的抽象表示。 |
| 进程：   * 进程是操作系统对正在运行的程序的一种抽象。一个系统可以运行多个进程，而每个进程好像在独占使用硬件。 |
| 进程上下文切换 |
| **虚拟存储器**   * 虚拟存储器为每个进程提供了一个大的、一致的、私有的地址空间 * 它将内存看成是存储在磁盘上的地址空间的高速缓存，在主存中只保存活动区域，并根据需要在主存与磁盘中来回交换数据。 * 为每个进程提供一致的地址空间，简化了存储管理 * 保护每个进程的地址空间不被其他进程破坏。 |
| **进程虚地址空间** |
| **文件**   * **文件是一系列的字节序列** * **它向应用程序提供了一个统一的视角，来看待系统中各式各样的I/O设备。** |
| **虚拟文件系统VFS**   * **虚拟文件系统是内核实现的一种架构，为用户空间提供统一的文件操作接口，即文件系统调用。它在内核内部为不同的真实文件系统提供一致的抽象接口。** * **用户通过系统用与内核中的虚拟文件系统交互，进而操作实际的文件系统和设备。** |
|  |

### 3linux内核

|  |
| --- |
|  |
|  |
|  |
| 常识   Linux是最受欢迎的自由电脑[操作系统](http://baike.baidu.com/view/880.htm" \t "_blank)内核。它是一个用C语言写成，符合POSIX标准的类Unix操作系统。Linux最早是由[芬兰](http://baike.baidu.com/view/20509.htm" \t "_blank)[黑客](http://baike.baidu.com/view/1960.htm" \t "_blank) Linus Torvalds为尝试在英特尔x86架构上提供自由免费的类Unix操作系统而开发的。该计划开始于1991年，在计划的早期有一些Minix [黑客](http://baike.baidu.com/view/1960.htm" \t "_blank)提供了协助，而今天全球无数程序员正在为该计划无偿提供帮助。 技术上说Linux是一个内核。“内核”指的是一个提供[硬件抽象层](http://baike.baidu.com/view/500774.htm" \t "_blank)、磁盘及文件系统控制、多任务等功能的[系统软件](http://baike.baidu.com/view/7860.htm" \t "_blank)。一个内核不是一套完整的操作系统。一套基于Linux内核的完整操作系统叫作Linux操作系统，或是GNU/Linux。 |
| [Linux](http://baike.baidu.com/view/1634.htm" \t "_blank)是一个一体化[内核](http://baike.baidu.com/view/1366.htm" \t "_blank)（monolithic kernel）系统。[设备驱动程序](http://baike.baidu.com/view/15565.htm" \t "_blank)可以完全访问硬件。Linux内的[设备驱动程序](http://baike.baidu.com/view/15565.htm" \t "_blank)可以方便地以[模块化](http://baike.baidu.com/view/182267.htm" \t "_blank)（modularize）的形式设置，并在系统运行期间可直接装载或[卸载](http://baike.baidu.com/view/386432.htm" \t "_blank)。 |
| 架构  [Linux](http://baike.baidu.com/view/1634.htm" \t "_blank)是一个一体化[内核](http://baike.baidu.com/view/1366.htm" \t "_blank)（monolithic kernel）系统。[设备驱动程序](http://baike.baidu.com/view/15565.htm" \t "_blank)可以完全访问硬件。Linux内的[设备驱动程序](http://baike.baidu.com/view/15565.htm" \t "_blank)可以方便地以[模块化](http://baike.baidu.com/view/182267.htm" \t "_blank)（modularize）的形式设置，并在系统运行期间可直接装载或[卸载](http://baike.baidu.com/view/386432.htm" \t "_blank) 体系结构属性 在讨论大型而复杂的系统的[体系结构](http://baike.baidu.com/view/1188494.htm" \t "_blank)时，可以从很多角度来审视系统。[体系结构](http://baike.baidu.com/view/1188494.htm" \t "_blank)分析的一个目标是提供一种方法更好地理解[源代码](http://baike.baidu.com/view/60376.htm" \t "_blank)。  Linux 内核实现了很多重要的[体系结构](http://baike.baidu.com/view/1188494.htm" \t "_blank)属性。在或高或低的层次上，内核被划分为多个子系统。Linux 也可以看作是一个整体，因为它会将所有这些基本服务都集成到内核中。这与[微内核](http://baike.baidu.com/view/928365.htm" \t "_blank)的[体系结构](http://baike.baidu.com/view/1188494.htm" \t "_blank)不同，后者会提供一些基本的服务，例如[通信](http://baike.baidu.com/view/15007.htm" \t "_blank)、[I/O](http://baike.baidu.com/view/300881.htm" \t "_blank)、[内存](http://baike.baidu.com/view/1082.htm" \t "_blank)和[进程管理](http://baike.baidu.com/view/364947.htm" \t "_blank)，更具体的服务都是插入到微内核层中的。  随着时间的流逝，Linux 内核在[内存](http://baike.baidu.com/view/1082.htm" \t "_blank)和 CPU 使用方面具有较高的效率，并且非常稳定。但是对于 Linux 来说，最为有趣的是在这种大小和复杂性的前提下，依然具有良好的[可移植性](http://baike.baidu.com/view/1936417.htm" \t "_blank)。Linux 编译后可在大量处理器和具有不同[体系结构](http://baike.baidu.com/view/1188494.htm" \t "_blank)约束和需求的平台上运行。一个例子是 Linux 可以在一个具有[内存管理](http://baike.baidu.com/view/4541016.htm" \t "_blank)单元（MMU）的处理器上运行，也可以在那些不提供[MMU](http://baike.baidu.com/view/969924.htm" \t "_blank)的处理器上运行。Linux 内核的[uClinux](http://baike.baidu.com/view/163694.htm" \t "_blank)移植提供了对非 MMU 的支持 开发和规范 核心的开发和规范一直是由Linux社区控制着，版本也是唯一的。实际上，操作系统的[内核版本](http://baike.baidu.com/view/6744074.htm" \t "_blank)指的是在Linux本人领导下的开发小组开发出的系统内核的版本号。自1994年3月14日发布了第一个正式版本Linux 1.0以来，每隔一段时间就有新的版本或其修订版公布。  Linux将标准的GNU许可协议改称Copyleft，以便与Copyright相对照。通用的公共许可（GPL）允许用户销售、拷贝和改变具有Copyleft的[应用程序](http://baike.baidu.com/view/330120.htm" \t "_blank)。当然这些程序也可以是Copyright的，但是你必须允许进一步的销售、拷贝和对其代码进行改变，同时也必须使他人可以免费得到修改后的[源代码](http://baike.baidu.com/view/60376.htm" \t "_blank)。事实证明，GPL对于Linux的成功起到了极大的作用。它启动了一个十分繁荣的商用Linux阶段，还为[编程](http://baike.baidu.com/view/3281.htm" \t "_blank)人员提供了一种凝聚力，诱使大家加入这个充满了慈善精神的Linux运动。 主要子系统 **系统调用接口**：SCI 层提供了某些机制执行从[用户空间](http://baike.baidu.com/view/4274331.htm" \t "_blank)到内核的[函数调用](http://baike.baidu.com/view/2369016.htm" \t "_blank)。正如前面讨论的一样，这个接口依赖于[体系结构](http://baike.baidu.com/view/1188494.htm" \t "_blank)，甚至在相同的处理器家族内也是如此。SCI 实际上是一个非常有用的[函数调用](http://baike.baidu.com/view/2369016.htm" \t "_blank)多路复用和多路分解服务。在 ./[linux](http://baike.baidu.com/view/1634.htm" \t "_blank)/kernel 中您可以找到 SCI 的实现，并在 ./[linux](http://baike.baidu.com/view/1634.htm" \t "_blank)/arch 中找到依赖于[体系结构](http://baike.baidu.com/view/1188494.htm" \t "_blank)的部分。  [进程管理](http://baike.baidu.com/view/364947.htm" \t "_blank)：进程管理的重点是进程的执行。在内核中，这些进程称为线程，代表了单独的处理器虚拟化（线程代码、数据、[堆栈](http://baike.baidu.com/view/93201.htm" \t "_blank)和 CPU[寄存器](http://baike.baidu.com/view/6159.htm" \t "_blank)）。在[用户空间](http://baike.baidu.com/view/4274331.htm" \t "_blank)，通常使用进程 这个术语，不过 Linux 实现并没有区分这两个概念（进程和线程）。内核通过 SCI 提供了一个[应用程序编程接口](http://baike.baidu.com/view/185287.htm" \t "_blank)（[API](http://baike.baidu.com/view/16068.htm" \t "_blank)）来创建一个新进程（fork、exec 或 Portable Operating System Interface [POSⅨ] 函数），停止进程（kill、exit），并在它们之间进行[通信](http://baike.baidu.com/view/15007.htm" \t "_blank)和同步（signal 或者 POSⅨ 机制）。  [进程管理](http://baike.baidu.com/view/364947.htm" \t "_blank)还包括处理活动进程之间共享 CPU 的需求。内核实现了一种新型的[调度算法](http://baike.baidu.com/view/2963962.htm" \t "_blank)， 不管有多少个线程在竞争 CPU，这种算法都可以在固定时间内进行操作。这种算法就称为 O⑴ 调度程序，这个名字就表示它调度多个线程所使用的时间和调度一个线程所使用的时间是相同的。O⑴ 调度程序也可以支持多处理器（称为对称多处理器或 SMP）。您可以在 ./linux/kernel 中找到[进程管理](http://baike.baidu.com/view/364947.htm" \t "_blank)的[源代码](http://baike.baidu.com/view/60376.htm" \t "_blank)，在 ./linux/arch 中可以找到依赖于[体系结构](http://baike.baidu.com/view/1188494.htm" \t "_blank)的源代码。  [内存管理](http://baike.baidu.com/view/4541016.htm" \t "_blank)：内核所管理的另外一个重要资源是内存。为了提高效率，如果由硬  [VFS 在用户和文件系统之间提供了一个交换层](http://baike.baidu.com/picture/573460/573460/0/34bbf8cddce9b61c0fb3458a?fr=lemma&ct=single)  VFS 在用户和文件系统之间提供了一个交换层  管理[虚拟内存](http://baike.baidu.com/view/976.htm" \t "_blank)，内存是按照所谓的内存页 方式进行管理的（对于大部分[体系结构](http://baike.baidu.com/view/1188494.htm" \t "_blank)来说都是 4KB）。Linux 包括了管理可用[内存](http://baike.baidu.com/view/1082.htm" \t "_blank)的方式，以及物理和虚拟映射所使用的硬件机制。  不过[内存管理](http://baike.baidu.com/view/4541016.htm" \t "_blank)要管理的可不止 4KB[缓冲区](http://baike.baidu.com/view/266782.htm" \t "_blank)。Linux 提供了对 4KB[缓冲区](http://baike.baidu.com/view/266782.htm" \t "_blank)的抽象，例如 slab 分配器。这种[内存管理](http://baike.baidu.com/view/4541016.htm" \t "_blank)模式使用 4KB[缓冲区](http://baike.baidu.com/view/266782.htm" \t "_blank)为基数，然后从中分配结构，并跟踪内存页使用情况，[比如](http://baike.baidu.com/view/6814120.htm" \t "_blank)哪些内存页是满的，哪些页面没有完全使用，哪些页面为空。这样就允许该模式根据系统需要来动态调整[内存](http://baike.baidu.com/view/1082.htm" \t "_blank)使用。  为了支持多个用户使用[内存](http://baike.baidu.com/view/1082.htm" \t "_blank)，有时会出现可用内存被消耗光的情况。由于这个原因，页面可以移出[内存](http://baike.baidu.com/view/1082.htm" \t "_blank)并放入磁盘中。这个过程称为交换，因为页面会被从[内存](http://baike.baidu.com/view/1082.htm" \t "_blank)交换到硬盘上。[内存管理](http://baike.baidu.com/view/4541016.htm" \t "_blank)的[源代码](http://baike.baidu.com/view/60376.htm" \t "_blank)可以在 ./[linux](http://baike.baidu.com/view/1634.htm" \t "_blank)/mm 中找到。  [虚拟文件系统](http://baike.baidu.com/view/671797.htm" \t "_blank)：虚拟文件系统（VFS）是 Linux 内核中非常有用的一个方面，因为它为文件系统提供了一个通用的接口抽象。VFS 在 SCI 和内核所支持的文件系统之间提供了一个交换层。  VFS 在用户和文件系统之间提供了一个交换层  在 VFS 上面，是对诸如 open、close、read 和 write 之类的函数的一个通用 API 抽象。在 VFS 下面是文件系统抽象，它定义了上层函数的实现方式。它们是给定文件系统（超过 50 个）的插件。文件系统的[源代码](http://baike.baidu.com/view/60376.htm" \t "_blank)可以在 ./[linux](http://baike.baidu.com/view/1634.htm" \t "_blank)/fs 中找到。  文件系统层之下是[缓冲区](http://baike.baidu.com/view/266782.htm" \t "_blank)缓存，它为文件系统层提供了一个通用函数集（与具体文件系统无关）。这个缓存层通过将数据保留一段时间（或者随即预先读取数据以便在需要是就可用）优化了对[物理设备](http://baike.baidu.com/view/5059658.htm" \t "_blank)的访问。[缓冲区](http://baike.baidu.com/view/266782.htm" \t "_blank)缓存之下是[设备驱动程序](http://baike.baidu.com/view/15565.htm" \t "_blank)，它实现了特定[物理设备](http://baike.baidu.com/view/5059658.htm" \t "_blank)的接口。 特性 如果 Linux 内核的[可移植性](http://baike.baidu.com/view/1936417.htm" \t "_blank)和效率还不够好，Linux 还提供了其他一些特性，它们无法划分到上面的分类中。  作为一个生产操作系统和[开源软件](http://baike.baidu.com/view/444964.htm" \t "_blank)，Linux 是测试新协议及其增强的良好平台。Linux 支持大量[网络协议](http://baike.baidu.com/view/16603.htm" \t "_blank)，包括典型的 TCP/IP，以及高速网络的扩展（大于 1 Gigabit Ethernet [GbE] 和 10 GbE）。Linux 也可以支持诸如[流控制传输协议](http://baike.baidu.com/view/1333903.htm" \t "_blank)（SCTP）之类的协议，它提供了很多比 TCP 更高级的特性（是传输层协议的接替者）。  Linux 还是一个动态内核，支持动态添加或删除[软件组件](http://baike.baidu.com/view/551014.htm" \t "_blank)。被称为动态可加载内核模块，它们可以在引导时根据需要（当前特定设备需要这个模块）或在任何时候由用户插入。  Linux 最新的一个增强是可以用作其他操作系统的操作系统（称为[系统管理](http://baike.baidu.com/view/635537.htm" \t "_blank)程序）。该系统对内核进行了修改，称为基于内核的[虚拟机](http://baike.baidu.com/view/1132.htm" \t "_blank)（KVM）。这个修改为[用户空间](http://baike.baidu.com/view/4274331.htm" \t "_blank)启用了一个新的接口，它可以允许其他操作系统在启用了 KVM 的内核之上运行。除了运行 Linux 的其他实例之外， Microsoft&reg; Windows&reg; 也可以进行虚拟化。惟一的限制是底层处理器必须支持新的虚拟化指令[4] 。 |

# 系统编程和应用编程

|  |
| --- |
| * 系统编程   在操作系统之上利用系统调用、C库进行对系统资源进行访问。如apache 、gcc、gdb 等   * 应用编程   在更高层次的编程接口或者库之上构建应用程序。如android程序（android sdk）、iphone程序（iphone sdk）、QT程序设计（QT）MFC程序设计（MFC）等。 |
| **系统调用在系统中所处的位置**   * **所有操作系统都提供多种服务的入口点，由此程序向系统核请求服务。这些入口点被称之为系统调用(system call)，** |
| **C库**   * **这里我们所说的C库（libc），指的是标准C定义的Ｃ函数的集合。如标准输入输出函数、字符串处理函数、动态存储分配函数、日期时间函数、数学函数等。** * **GNU发布的libc称为glibc** |
| **系统调用与C库关系**   * 系统调用与Ｃ库从形式上来看都C函数 * C库函数有些是调用系统调用来实现的，比如说malloc、free调用brk，printf调用write系统用，有些函数不需要任何系统调用，比如abs、strcpy、atoi等，因为它并不是必需要使用内核服务 * 系统调用通常提供的是最小界面，而Ｃ库函数通常提供更复杂的功能。 |
| **内核如何处理系统调用**   * **每个系统调用被赋予一个系统调用号** * **在i386平台上，执行一个系统调用是通过int 0x80指令完成的。** * **eax存放系统调用号** * **ebx、ecx、edx、esi、edi存储系统调用参数，对于超过5个参数的系统调用，用一个寄存器指向用户空间存储所有系统调用参数的缓存。** |
| **错误处理**   * **在系统编程中错误通常通过函数返回值来表示，并通过特殊变量errno来描述。** * **errno这个全局变量在<errno.h>头文件中声明如下：extern int errno;** * **错误处理函数**   + **perror**   + **strerror** |

linux应用编程市场需求

|  |
| --- |
| 广州博庭诚招金融软件开发工程师    于2004年成立的广州博庭计算机科技有限公司位于广州天河广纺联创意园，是专注于证券金融软件与服务的技术公司。多年来，博庭科技在业界拥有优秀口碑。广州团队的产品**《飞狐交易师》**软件是众多技术分析发烧者心目中的标杆，至今拥有大量忠实用户。在台湾长期坐拥市占率第一。    博庭科技在量化投资交易领域尤其拥有强大实力。量化交易是指利用计算机结合一定的数据模型实现投资理念，执行投资策略的过程。量化交易在国际市场占交易量70%，是国际投资的主流先进方法，而在国内也是最热门的新兴技术，是未来的专业投资主流趋势。前景无可限量。    博 庭拥有广州和上海两个团队。博庭科技既开发面向大众的金融软件，也为高端客户订制交易软件。正在研发的《金魔方》软件在继承飞狐交易师强大功能的基础上， 融合先进的量化分析研发和交易功能，致力于成为中国期货、期权、股票、和现货市场领先的量化交易软件。上海团队拥有证券柜台系统/账户管理系统、风控系 统、高频交易系统、各大证券期货交易接口系统、手机客户端。    我们可以为客户提供从后台到前台的一个完整的交易服务全产品链。我们的客户包括各类交易所、券商、期货商、基金、证券投资人。    另外，广州团队和上海团队各自运营两个不同风格的量化的私募基金，均拥有骄人的投资业绩。    本次招聘员工主要工作于广州。同时，根据本人意愿和技能特点也可能在上海甚至海外工作。    我们提供的福利和待遇：薪资6000-2万。社保和公积金。带薪年假。文体活动。学习机会。以及私募基金投资收益等。    **招聘要求：**    招聘职位：软件开发工程师    工作职责：    1.负责证券量化分析和交易软件的客户端及服务器模块设计、编码、调试、测试等工作。  2.参与相关质量活动，确保设计、实现、测试工作按时保质完成。    职位要求：  1、对金融和证券投资拥有浓厚兴趣。平时关心相关知识。  2、计算机、通信、软件工程、自动化、数学或相关专业，本科及以上学历。  3、精通C/C++语言/JAVA软件编程，熟悉TCP/IP协议、Internet网络的基本知识、熟悉STL。  4、熟练使用windows或Linux开发、编译、调试环境，如VC、。  5、能够熟练阅读和理解英文资料。  6、深刻理解Windows/Linux/Unix内存管理机制、进程/线程、各种进程间通信方式、消息事件通知机制和异步机制者加分。  7、精通编译原理者加分。 |
| 初级linux开发工程师   |  | | --- | | **职位职能:**  软件工程师 | | **职位描述:**  岗位职责： 1）项目任务执行 l 完成开发经理下达的项目任务  l 积极配合开发经理，保证项目的质量和进度  2）技术任务执行 l 完成技术主管下达的技术研究任务  l 积极与同组员工分享技术知识；  l协助同事解决技术问题  3）领导交办的其他事项  任职资格： 1) 掌握c、c++语言基础和特性。  2) 掌握计算机相关技术基础，包括操作系统、数据结构、汇编语言、组成原理、计算机网络、编译原理、设计模式等。  3) 熟练应用gcc/g++/gdb等开发调试工具。  4) 熟悉shell脚本，能进行makefile编写。  5) 熟练使用常用的数据库sqlite、mysql、oracle等。  6) 熟练使用常用的网络工具。 |   **[北京鼎普科技股份有限公司](http://search.51job.com/list/co,c,1284425,000000,10,1.html" \t "_blank)** |
| |  | | --- | |  | | **职位职能:**  高级软件工程师  软件工程师 | | **职位描述:**  岗位职责：  1.Linux平台下服务软件开发； 2.优化算法和性能； 3.脚本的自动化配置部署；   任职资格：  1.熟悉Linux，2年以上的开发经验； 2.精通C或Python或Perl开发，2年以上开发经验； 3.具备良好的分析解决问题能力； 4.学习能力强，善于沟通，富有团队合作精神，能承受一定的工作压力； 5.有大并发高性能服务器开发经验优先； |   **[北京中软冠群软件技术有限公司](http://search.51job.com/list/co,c,235896,000000,10,1.html" \t "_blank)** |
|  |
| |  | | --- | |  | | **职位职能:**  高级软件工程师 | | **职位描述:**  岗位职责： 1.负责Wifi networking开发- Linux kernel/module, system programming。  任职资格： 1.3年以上软件开发相关经验, 熟悉Embedded system&RTOS； 2.熟悉C/C++ 开发； 3.具备Network Programming 经验； 4.熟悉linux内核开发。 |   高级linux开发工程师 **[360安全卫士](http://search.51job.com/list/co,c,804886,000000,10,1.html" \t "_blank)** |

# 学到什么程度可以找工作

这个问题太难回答？

你要找什么样的工作

**linux内核开发**

linux内核常用开发模型

字符设备开发 块设备开发 网络设备开发usb设备开发。。。

驱动总线模型

linux内核 并发机制。。。。（中断、自旋锁、队列（任务））

linux内核内存管理。。。。。。。。。。。。。。。。。

linux内核移植

bootload /内核移植。。。。和硬件 绑定 + 2410 2440 6410 mini

linux应用开发

1多进程多线程的客户端 并发服务器。。。高效率的处理业务。。。。

多进程多线程机制

进程间锁机制。。。。。。线程锁 （同步和互斥 信号量的两种用法）

进程间通讯（共享内存、消息队列、socket、）

=======》死的知识点。。。

2、报文的打包机制

自定义报文

json xml der

3、TCP/IP的深入编程

tcpip写入深入分析 （粘包无边界、连接数、5种服务器模型）

4、=======》商业代码级别的服务框架。。。

===============》