网站架构（页面静态化，图片服务器分离,负载均衡）方案全解析

文章分类:综合技术

1、HTML静态化其实大家都知道，效率最高、消耗最小的就是纯静态化的html页面，所以我们尽可能使我们的网站上的页面采用静态页面来实现，这个最简单的方法其实也是最有效的方法。但是对于大量内容并且频繁更新的网站，我们无法全部手动去挨个实现，于是出现了我们常见的信息发布系统CMS，像我们常访问的各个门户站点的新闻频道，甚至他们的其他频道，都是通过信息发布系统来管理和实现的，信息发布系统可以实现最简单的信息录入自动生成静态页面，还能具备频道管理、权限管理、自动抓取等功能，对于一个大型网站来说，拥有一套高效、可管理的CMS是必不可少的。除了门户和信息发布类型的网站，对于交互性要求很高的社区类型网站来说，尽可能的静态化也是提高性能的必要手段，将社区内的帖子、文章进行实时的静态化，有更新的时候再重新静态化也是大量使用的策略，像Mop的大杂烩就是使用了这样的策略，网易社区等也是如此。同时，html静态化也是某些缓存策略使用的手段，对于系统中频繁使用数据库查询但是内容更新很小的应用，可以考虑使用html静态化来实现，比如论坛中论坛的公用设置信息，这些信息目前的主流论坛都可以进行后台管理并且存储再数据库中，这些信息其实大量被前台程序调用，但是更新频率很小，可以考虑将这部分内容进行后台更新的时候进行静态化，这样避免了大量的数据库访问请求。

2、图片服务器分离大家知道，对于Web服务器来说，不管是Apache、IIS还是其他容器，图片是最消耗资源的，于是我们有必要将图片与页面进行分离，这是基本上大型网站都会采用的策略，他们都有独立的图片服务器，甚至很多台图片服务器。这样的架构可以降低提供页面访问请求的服务器系统压力，并且可以保证系统不会因为图片问题而崩溃，在应用服务器和图片服务器上，可以进行不同的配置优化，比如apache在配置ContentType的时候可以尽量少支持，尽可能少的LoadModule，保证更高的系统消耗和执行效率。

3、数据库集群和库表散列大型网站都有复杂的应用，这些应用必须使用数据库，那么在面对大量访问的时候，数据库的瓶颈很快就能显现出来，这时一台数据库将很快无法满足应用，于是我们需要使用数据库集群或者库表散列。在数据库集群方面，很多数据库都有自己的解决方案，Oracle、Sybase等都有很好的方案，常用的MySQL提供的Master/Slave也是类似的方案，您使用了什么样的DB，就参考相应的解决方案来实施即可。上面提到的数据库集群由于在架构、成本、扩张性方面都会受到所采用DB类型的限制，于是我们需要从应用程序的角度来考虑改善系统架构，库表散列是常用并且最有效的解决方案。我们在应用程序中安装业务和应用或者功能模块将数据库进行分离，不同的模块对应不同的数据库或者表，再按照一定的策略对某个页面或者功能进行更小的数据库散列，比如用户表，按照用户ID进行表散列，这样就能够低成本的提升系统的性能并且有很好的扩展性。sohu的论坛就是采用了这样的架构，将论坛的用户、设置、帖子等信息进行数据库分离，然后对帖子、用户按照板块和ID进行散列数据库和表，最终可以在配置文件中进行简单的配置便能让系统随时增加一台低成本的数据库进来补充系统性能。

4、缓存缓存一词搞技术的都接触过，很多地方用到缓存。网站架构和网站开发中的缓存也是非常重要。这里先讲述最基本的两种缓存。高级和分布式的缓存在后面讲述。架构方面的缓存，对Apache比较熟悉的人都能知道Apache提供了自己的缓存模块，也可以使用外加的Squid模块进行缓存，这两种方式均可以有效的提高Apache的访问响应能力。网站程序开发方面的缓存，Linux上提供的Memory Cache是常用的缓存接口，可以在web开发中使用，比如用Java开发的时候就可以调用MemoryCache对一些数据进行缓存和通讯共享，一些大型社区使用了这样的架构。另外，在使用web语言开发的时候，各种语言基本都有自己的缓存模块和方法，PHP有Pear的Cache模块，Java就更多了，.net不是很熟悉，相信也肯定有。

5、镜像镜像是大型网站常采用的提高性能和数据安全性的方式，镜像的技术可以解决不同网络接入商和地域带来的用户访问速度差异，比如 ChinaNet和EduNet之间的差异就促使了很多网站在教育网内搭建镜像站点，数据进行定时更新或者实时更新。在镜像的细节技术方面，这里不阐述太深，有很多专业的现成的解决架构和产品可选。也有廉价的通过软件实现的思路，比如Linux上的rsync等工具。

6、负载均衡负载均衡将是大型网站解决高负荷访问和大量并发请求采用的终极解决办法。负载均衡技术发展了多年，有很多专业的服务提供商和产品可以选择，我个人接触过一些解决方法，其中有两个架构可以给大家做参考。

7、硬件四层交换第四层交换使用第三层和第四层信息包的报头信息，根据应用区间识别业务流，将整个区间段的业务流分配到合适的应用服务器进行处理。　第四层交换功能就象是虚 IP，指向物理服务器。它传输的业务服从的协议多种多样，有HTTP、FTP、NFS、Telnet或其他协议。这些业务在物理服务器基础上，需要复杂的载量平衡算法。在IP世界，业务类型由终端TCP或UDP端口地址来决定，在第四层交换中的应用区间则由源端和终端IP地址、TCP和UDP端口共同决定。在硬件四层交换产品领域，有一些知名的产品可以选择，比如Alteon、F5等，这些产品很昂贵，但是物有所值，能够提供非常优秀的性能和很灵活的管理能力。Yahoo中国当初接近2000台服务器使用了三四台Alteon就搞定了

。8、软件四层交换大家知道了硬件四层交换机的原理后，基于OSI模型来实现的软件四层交换也就应运而生，这样的解决方案实现的原理一致，不过性能稍差。但是满足一定量的压力还是游刃有余的，有人说软件实现方式其实更灵活，处理能力完全看你配置的熟悉能力。软件四层交换我们可以使用Linux上常用的LVS来解决，LVS就是Linux Virtual Server，他提供了基于心跳线heartbeat的实时灾难应对解决方案，提高系统的鲁棒性，同时可供了灵活的虚拟VIP配置和管理功能，可以同时满足多种应用需求，这对于分布式的系统来说必不可少。一个典型的使用负载均衡的策略就是，在软件或者硬件四层交换的基础上搭建squid集群，这种思路在很多大型网站包括搜索引擎上被采用，这样的架构低成本、高性能还有很强的扩张性，随时往架构里面增减节点都非常容易。这样的架构我准备空了专门详细整理一下和大家探讨。对于大型网站来说，前面提到的每个方法可能都会被同时使用到，我这里介绍得比较浅显，具体实现过程中很多细节还需要大家慢慢熟悉和体会，有时一个很小的squid参数或者apache参数设置，对于系统性能的影响就会很大，希望大家一起讨论，达到抛砖引玉之效。

用squid做web cache server，而apache在squid的后面提供真正的web服务。当然使用这样的架构必须要保证主页上大部分都是静态页面。这就需要程序员的配合将页面在反馈给客户端之前将页面全部转换成静态页面。

基本看出sina和sohu对于频道等栏目都用了相同的技术，即squid来监听这些IP的80端口，而真正的web server来监听另外一个端口。从用户的感觉上来说不会有任何的区别，而相对于将web server直接和客户端连在一起的方式，这样的方式明显的节省的带宽和服务器。用户访问的速度感觉也会更快。

http://www.dbanotes.net/arch/yupoo\_arch.html

带宽：4000M/S (参考)

服务器数量：60 台左右

Web服务器：Lighttpd, Apache, nginx

应用服务器：Tomcat

其他：Python, Java, MogileFS 、ImageMagick 等

关于 Squid 与 Tomcat

Squid 与 Tomcat 似乎在 Web 2.0 站点的架构中较少看到。我首先是对 Squid 有点疑问，对此阿华的解释是"目前暂时还没找到效率比 Squid 高的缓存系统，原来命中率的确很差，后来在 Squid 前又装了层 Lighttpd, 基于 url 做 hash, 同一个图片始终会到同一台 squid 去，所以命中率彻底提高了"

对于应用服务器层的 Tomcat，现在 Yupoo! 技术人员也在逐渐用其他轻量级的东西替代，而 YPWS/YPFS 现在已经用 Python 进行开发了。

名次解释：

· YPWS--Yupoo Web Server YPWS 是用 Python开发的一个小型 Web 服务器，提供基本的 Web 服务外，可以增加针对用户、图片、外链网站显示的逻辑判断，可以安装于任何有空闲资源的服务器中，遇到性能瓶颈时方便横向扩展。

· YPFS--Yupoo File System 与 YPWS 类似，YPFS 也是基于这个 Web 服务器上开发的图片上传服务器。

【Updated: 有网友留言质疑 Python 的效率，Yupoo 老大刘平阳在 del.icio.us 上写到 "YPWS用Python自己写的，每台机器每秒可以处理294个请求, 现在压力几乎都在10％以下"】

图片处理层

接下来的 Image Process Server 负责处理用户上传的图片。使用的软件包也是 ImageMagick，在上次存储升级的同时，对于锐化的比率也调整过了(我个人感觉，效果的确好了很多)。”Magickd“ 是图像处理的一个远程接口服务，可以安装在任何有空闲 CPU资源的机器上，类似 Memcached的服务方式。

我们知道 Flickr 的缩略图功能原来是用 ImageMagick 软件包的，后来被雅虎收购后出于版权原因而不用了（?）；EXIF 与 IPTC Flicke 是用 Perl 抽取的，我是非常建议 Yupoo! 针对 EXIF 做些文章，这也是潜在产生受益的一个重点。

图片存储层

原来 Yupoo! 的存储采用了磁盘阵列柜，基于 NFS 方式的，随着数据量的增大，”Yupoo! 开发部从07年6月份就开始着手研究一套大容量的、能满足 Yupoo! 今后发展需要的、安全可靠的存储系统“，看来 Yupoo! 系统比较有信心，也是满怀期待的，毕竟这要支撑以 TB 计算的海量图片的存储和管理。我们知道，一张图片除了原图外，还有不同尺寸的，这些图片统一存储在 MogileFS 中。

对于其他部分，常见的 Web 2.0 网站必须软件都能看到，如 MySQL、Memcached 、Lighttpd 等。Yupoo! 一方面采用不少相对比较成熟的开源软件，一方面也在自行开发定制适合自己的架构组件。这也是一个 Web 2.0 公司所必需要走的一个途径。

非常感谢一下 Yupoo! 阿华对于技术信息的分享，技术是共通的。下一个能爆料是哪家?

--EOF--

lighttpd+squid这套缓存是放在另外一个机房作为cdn的一个节点使用的，图中没描绘清楚，给大家带来不便了。

squid前端用lighttpd没用nginx，主要是用了这么久，没出啥大问题，所以就没想其他的了。

URL Hash的扩展性的确不好，能做的就是不轻易去增减服务器，我们目前是5台服务器做一组hash.

我们现在用Python写的Web Server，在效率方面，我可以给个测试数据，根据目前的访问日志模拟访问测试的结果是1台ypws,平均每秒处理294个请求(加载所有的逻辑判断)。

在可靠性上，还不没具体的数据，目前运行1个多月还没有任何异常。

lvs每个节点上都装nginx，主要是为了反向代理及处理静态内容，不过apache已显得不是那么必需，准备逐渐去掉。

我们处理图片都是即时的，我们目前半数以上的服务器都装了magickd服务，用来分担图片处理请求。

http://www.dbanotes.net/review/tailrank\_arch.html

每天数以千万计的 Blog 内容中，实时的热点是什么? Tailrank 这个 Web 2.0 Startup 致力于回答这个问题。

专门爆料网站架构的 Todd Hoff 对 Kevin Burton 进行了采访。于是我们能了解一下 Tailrank 架构的一些信息。每小时索引 2400 万的 Blog 与 Feed，内容处理能力为 160-200Mbps，IO 写入大约在10-15MBps。每个月要处理 52T 之多的原始数据。Tailrank 所用的爬虫现在已经成为一个独立产品：spinn3r。

服务器硬件

目前大约 15 台服务器，CPU 是 64 位的 Opteron。每台主机上挂两个 SATA 盘，做 RAID 0。据我所知，国内很多 Web 2.0 公司也用的是类似的方式，SATA 盘容量达，低廉价格，堪称不二之选。操作系统用的是 Debian Linux 。Web 服务器用 Apache 2.0，Squid 做反向代理服务器。

数据库

Tailrank 用 MySQL 数据库，联邦数据库形式。存储引擎用 InnoDB， 数据量 500GB。Kevin Burton 也指出了 MySQL 5 在修了一些 多核模式下互斥锁的问题(This Bug?)。到数据库的JDBC 驱动连接池用 lbpool 做负载均衡。MySQL Slave 或者 Master的复制用 MySQLSlaveSync 来轻松完成。不过即使这样，还要花费 20％的时间来折腾 DB。

其他开放的软件

任何一套系统都离不开合适的 Profiling 工具，Tailrank 也不利外，针对 Java 程序的 Benchmark 用 Benchmark4j。Log 工具用 Log5j(不是 Log4j)。Tailrank 所用的大部分工具都是开放的。

Tailrank 的一个比较大的竞争对手是 Techmeme，虽然二者暂时看面向内容的侧重点有所不同。其实，最大的对手还是自己，当需要挖掘的信息量越来越大，如果精准并及时的呈现给用户内容的成本会越来越高。从现在来看，Tailrank 离预期目标还差的很远。期待罗马早日建成

YouTube架构学习

关键字: YouTube

原文: YouTube Architecture

YouTube发展迅速，每天超过1亿的视频点击量，但只有很少人在维护站点和确保伸缩性。

平台

Apache

Python

Linux(SuSe)

MySQL

psyco，一个动态的Python到C的编译器

lighttpd代替Apache做视频查看

状态

支持每天超过1亿的视频点击量

成立于2005年2月

于2006年3月达到每天3千万的视频点击量

于2006年7月达到每天1亿的视频点击量

2个系统管理员，2个伸缩性软件架构师

2个软件开发工程师，2个网络工程师，1个DBA

处理飞速增长的流量

Java代码

1. while (true)

2. {

3. identify\_and\_fix\_bottlenecks();

4. drink();

5. sleep();

6. notice\_new\_bottleneck();

7. }

while (true)

{

identify\_and\_fix\_bottlenecks();

drink();

sleep();

notice\_new\_bottleneck();

}

每天运行该循环多次

Web服务器

1，NetScaler用于负载均衡和静态内容缓存

2，使用mod\_fast\_cgi运行Apache

3，使用一个Python应用服务器来处理请求的路由

4，应用服务器与多个数据库和其他信息源交互来获取数据和格式化html页面

5，一般可以通过添加更多的机器来在Web层提高伸缩性

6，Python的Web层代码通常不是性能瓶颈，大部分时间阻塞在RPC

7，Python允许快速而灵活的开发和部署

8，通常每个页面服务少于100毫秒的时间

9，使用psyco(一个类似于JIT编译器的动态的Python到C的编译器)来优化内部循环

10，对于像加密等密集型CPU活动，使用C扩展

11，对于一些开销昂贵的块使用预先生成并缓存的html

12，数据库里使用行级缓存

13，缓存完整的Python对象

14，有些数据被计算出来并发送给各个程序，所以这些值缓存在本地内存中。这是个使用不当的策略。应用服务器里最快的缓存将预先计算的值发送给所有服务器也花不了多少时间。只需弄一个代理来监听更改，预计算，然后发送。

视频服务

1，花费包括带宽，硬件和能源消耗

2，每个视频由一个迷你集群来host，每个视频被超过一台机器持有

3，使用一个集群意味着：

-更多的硬盘来持有内容意味着更快的速度

-failover。如果一台机器出故障了，另外的机器可以继续服务

-在线备份

4，使用lighttpd作为Web服务器来提供视频服务：

-Apache开销太大

-使用epoll来等待多个fds

-从单进程配置转变为多进程配置来处理更多的连接

5，大部分流行的内容移到CDN：

-CDN在多个地方备份内容，这样内容离用户更近的机会就会更高

-CDN机器经常内存不足，因为内容太流行以致很少有内容进出内存的颠簸

6，不太流行的内容(每天1-20浏览次数)在许多colo站点使用YouTube服务器

-长尾效应。一个视频可以有多个播放，但是许多视频正在播放。随机硬盘块被访问

-在这种情况下缓存不会很好，所以花钱在更多的缓存上可能没太大意义。

-调节RAID控制并注意其他低级问题

-调节每台机器上的内存，不要太多也不要太少

视频服务关键点

1，保持简单和廉价

2，保持简单网络路径，在内容和用户间不要有太多设备

3，使用常用硬件，昂贵的硬件很难找到帮助文档

4，使用简单而常见的工具，使用构建在Linux里或之上的大部分工具

5，很好的处理随机查找(SATA，tweaks)

缩略图服务

1，做到高效令人惊奇的难

2，每个视频大概4张缩略图，所以缩略图比视频多很多

3，缩略图仅仅host在几个机器上

4，持有一些小东西所遇到的问题：

-OS级别的大量的硬盘查找和inode和页面缓存问题

-单目录文件限制，特别是Ext3，后来移到多分层的结构。内核2.6的最近改进可能让Ext3允许大目录，但在一个文件系统里存储大量文件不是个好主意

-每秒大量的请求，因为Web页面可能在页面上显示60个缩略图

-在这种高负载下Apache表现的非常糟糕

-在Apache前端使用squid，这种方式工作了一段时间，但是由于负载继续增加而以失败告终。它让每秒300个请求变为20个

-尝试使用lighttpd但是由于使用单线程它陷于困境。遇到多进程的问题，因为它们各自保持自己单独的缓存

-如此多的图片以致一台新机器只能接管24小时

-重启机器需要6-10小时来缓存

5，为了解决所有这些问题YouTube开始使用Google的BigTable，一个分布式数据存储：

-避免小文件问题，因为它将文件收集到一起

-快，错误容忍

-更低的延迟，因为它使用分布式多级缓存，该缓存与多个不同collocation站点工作

-更多信息参考Google Architecture，GoogleTalk Architecture和BigTable

数据库

1，早期

-使用MySQL来存储元数据，如用户，tags和描述

-使用一整个10硬盘的RAID 10来存储数据

-依赖于信用卡所以YouTube租用硬件

-YouTube经过一个常见的革命：单服务器，然后单master和多read slaves，然后数据库分区，然后sharding方式

-痛苦与备份延迟。master数据库是多线程的并且运行在一个大机器上所以它可以处理许多工作，slaves是单线程的并且通常运行在小一些的服务器上并且备份是异步的，所以slaves会远远落后于master

-更新引起缓存失效，硬盘的慢I/O导致慢备份

-使用备份架构需要花费大量的money来获得增加的写性能

-YouTube的一个解决方案是通过把数据分成两个集群来将传输分出优先次序：一个视频查看池和一个一般的集群

2，后期

-数据库分区

-分成shards，不同的用户指定到不同的shards

-扩散读写

-更好的缓存位置意味着更少的IO

-导致硬件减少30%

-备份延迟降低到0

-现在可以任意提升数据库的伸缩性

数据中心策略

1，依赖于信用卡，所以最初只能使用受管主机提供商

2，受管主机提供商不能提供伸缩性，不能控制硬件或使用良好的网络协议

3，YouTube改为使用colocation arrangement。现在YouTube可以自定义所有东西并且协定自己的契约

4，使用5到6个数据中心加CDN

5，视频来自任意的数据中心，不是最近的匹配或其他什么。如果一个视频足够流行则移到CDN

6，依赖于视频带宽而不是真正的延迟。可以来自任何colo

7，图片延迟很严重，特别是当一个页面有60张图片时

8，使用BigTable将图片备份到不同的数据中心，代码查看谁是最近的

学到的东西

1，Stall for time。创造性和风险性的技巧让你在短期内解决问题而同时你会发现长期的解决方案

2，Proioritize。找出你的服务中核心的东西并对你的资源分出优先级别

3，Pick your battles。别怕将你的核心服务分出去。YouTube使用CDN来分布它们最流行的内容。创建自己的网络将花费太多时间和太多money

4，Keep it simple！简单允许你更快的重新架构来回应问题

5，Shard。Sharding帮助隔离存储，CPU，内存和IO，不仅仅是获得更多的写性能

6，Constant iteration on bottlenecks：

-软件：DB，缓存

-OS：硬盘I/O

-硬件：内存，RAID

7，You succeed as a team。拥有一个跨越条律的了解整个系统并知道系统内部是什么样的团队，如安装打印机，安装机器，安装网络等等的人。With a good team all things are possible。

http://hideto.javaeye.com/blog/130815

Google架构学习

关键字: Google

原文：Google Architecture

Google是伸缩性的王者。Google一直的目标就是构建高性能高伸缩性的基础组织来支持它们的产品。

平台

Linux

大量语言：Python，Java，C++

状态

在2006年大约有450,000台廉价服务器

在2005年Google索引了80亿Web页面，现在没有人知道数目

目前在Google有超过200个GFS集群。一个集群可以有1000或者甚至5000台机器。成千上万的机器从运行着5000000000000000字节存储的GFS集群获取数据，集群总的读写吞吐量可以达到每秒40兆字节

目前在Google有6000个MapReduce程序，而且每个月都写成百个新程序

BigTable伸缩存储几十亿的URL，几百千千兆的卫星图片和几亿用户的参数选择

堆栈

Google形象化它们的基础组织为三层架构：

1，产品：搜索，广告，email，地图，视频，聊天，博客

2，分布式系统基础组织：GFS，MapReduce和BigTable

3，计算平台：一群不同的数据中心里的机器

4，确保公司里的人们部署起来开销很小

5，花费更多的钱在避免丢失日志数据的硬件上，其他类型的数据则花费较少

可信赖的存储机制GFS(Google File System)

1，可信赖的伸缩性存储是任何程序的核心需求。GFS就是Google的核心存储平台

2，Google File System - 大型分布式结构化日志文件系统，Google在里面扔了大量的数据

3，为什么构建GFS而不是利用已有的东西？因为可以自己控制一切并且这个平台与别的不一样，Google需要：

-跨数据中心的高可靠性

-成千上万的网络节点的伸缩性

-大读写带宽的需求

-支持大块的数据，可能为上千兆字节

-高效的跨节点操作分发来减少瓶颈

4，系统有Master和Chunk服务器

-Master服务器在不同的数据文件里保持元数据。数据以64MB为单位存储在文件系统中。客户端与Master服务器交流来在文件上做元数据操作并且找到包含用户需要数据的那些Chunk服务器

-Chunk服务器在硬盘上存储实际数据。每个Chunk服务器跨越3个不同的Chunk服务器备份以创建冗余来避免服务器崩溃。一旦被Master服务器指明，客户端程序就会直接从Chunk服务器读取文件

6，一个上线的新程序可以使用已有的GFS集群或者可以制作自己的GFS集群

7，关键点在于有足够的基础组织来让人们对自己的程序有所选择，GFS可以调整来适应个别程序的需求

使用MapReduce来处理数据

1，现在你已经有了一个很好的存储系统，你该怎样处理如此多的数据呢？比如你有许多TB的数据存储在1000台机器上。数据库不能伸缩或者伸缩到这种级别花费极大，这就是MapReduce出现的原因

2，MapReduce是一个处理和生成大量数据集的编程模型和相关实现。用户指定一个map方法来处理一个键/值对来生成一个中间的键/值对，还有一个reduce方法来合并所有关联到同样的中间键的中间值。许多真实世界的任务都可以使用这种模型来表现。以这种风格来写的程序会自动并行的在一个大量机器的集群里运行。运行时系统照顾输入数据划分、程序在机器集之间执行的调度、机器失败处理和必需的内部机器交流等细节。这允许程序员没有多少并行和分布式系统的经验就可以很容易使用一个大型分布式系统资源

3，为什么使用MapReduce？

-跨越大量机器分割任务的好方式

-处理机器失败

-可以与不同类型的程序工作，例如搜索和广告。几乎任何程序都有map和reduce类型的操作。你可以预先计算有用的数据、查询字数统计、对TB的数据排序等等

4，MapReduce系统有三种不同类型的服务器

-Master服务器分配用户任务到Map和Reduce服务器。它也跟踪任务的状态

-Map服务器接收用户输入并在其基础上处理map操作。结果写入中间文件

-Reduce服务器接收Map服务器产生的中间文件并在其基础上处理reduce操作

5，例如，你想在所有Web页面里的字数。你将存储在GFS里的所有页面抛入MapReduce。这将在成千上万台机器上同时进行并且所有的调整、工作调度、失败处理和数据传输将自动完成

-步骤类似于：GFS -> Map -> Shuffle -> Reduction -> Store Results back into GFS

-在MapReduce里一个map操作将一些数据映射到另一个中，产生一个键值对，在我们的例子里就是字和字数

-Shuffling操作聚集键类型

-Reduction操作计算所有键值对的综合并产生最终的结果

6，Google索引操作管道有大约20个不同的map和reduction。

7，程序可以非常小，如20到50行代码

8，一个问题是掉队者。掉队者是一个比其他程序慢的计算，它阻塞了其他程序。掉队者可能因为缓慢的IO或者临时的CPU不能使用而发生。解决方案是运行多个同样的计算并且当一个完成后杀死所有其他的

9，数据在Map和Reduce服务器之间传输时被压缩了。这可以节省带宽和I/O。

在BigTable里存储结构化数据

1，BigTable是一个大伸缩性、错误容忍、自管理的系统，它包含千千兆的内存和1000000000000000的存储。它可以每秒钟处理百万的读写

2，BigTable是一个构建于GFS之上的分布式哈希机制。它不是关系型数据库。它不支持join或者SQL类型查询

3，它提供查询机制来通过键访问结构化数据。GFS存储存储不透明的数据而许多程序需求有结构化数据

4，商业数据库不能达到这种级别的伸缩性并且不能在成千上万台机器上工作

5，通过控制它们自己的低级存储系统Google得到更多的控制权来改进它们的系统。例如，如果它们想让跨数据中心的操作更简单这个特性，它们可以内建它

6，系统运行时机器可以自由的增删而整个系统保持工作

7，每个数据条目存储在一个格子里，它可以通过一个行key和列key或者时间戳来访问

8，每一行存储在一个或多个tablet中。一个tablet是一个64KB块的数据序列并且格式为SSTable

9，BigTable有三种类型的服务器：

-Master服务器分配tablet服务器，它跟踪tablet在哪里并且如果需要则重新分配任务

-Tablet服务器为tablet处理读写请求。当tablet超过大小限制(通常是100MB-200MB)时它们拆开tablet。当一个Tablet服务器失败时，则100个Tablet服务器各自挑选一个新的tablet然后系统恢复。

-Lock服务器形成一个分布式锁服务。像打开一个tablet来写、Master调整和访问控制检查等都需要互斥

10，一个locality组可以用来在物理上将相关的数据存储在一起来得到更好的locality选择

11，tablet尽可能的缓存在RAM里

硬件

1，当你有很多机器时你怎样组织它们来使得使用和花费有效？

2，使用非常廉价的硬件

3，A 1,000-fold computer power increase can be had for a 33 times lower cost if you you use a failure-prone infrastructure rather than an infrastructure built on highly reliable components. You must build reliability on top of unreliability for this strategy to work.

4，Linux，in-house rack design，PC主板，低端存储

5，Price per wattage on performance basis isn't getting better. Have huge power and cooling issues

6，使用一些collocation和Google自己的数据中心

其他

1，迅速更改而不是等待QA

2，库是构建程序的卓越方式

3，一些程序作为服务提供

4，一个基础组织处理程序的版本，这样它们可以发布而不用害怕会破坏什么东西

Google将来的方向

1，支持地理位置分布的集群

2，为所有数据创建一个单独的全局名字空间。当前的数据由集群分离

3，更多和更好的自动化数据迁移和计算

4，解决当使用网络划分来做广阔区域的备份时的一致性问题(例如保持服务即使一个集群离线维护或由于一些损耗问题)

学到的东西

1，基础组织是有竞争性的优势。特别是对Google而言。Google可以很快很廉价的推出新服务，并且伸缩性其他人很难达到。许多公司采取完全不同的方式。许多公司认为基础组织开销太大。Google认为自己是一个系统工程公司，这是一个新的看待软件构建的方式

2，跨越多个数据中心仍然是一个未解决的问题。大部分网站都是一个或者最多两个数据中心。我们不得不承认怎样在一些数据中心之间完整的分布网站是很需要技巧的

3，如果你自己没有时间从零开始重新构建所有这些基础组织你可以看看Hadoop。Hadoop是这里很多同样的主意的一个开源实现

4，平台的一个优点是初级开发人员可以在平台的基础上快速并且放心的创建健全的程序。如果每个项目都需要发明同样的分布式基础组织的轮子，那么你将陷入困境因为知道怎样完成这项工作的人相对较少

5，协同工作不一直是掷骰子。通过让系统中的所有部分一起工作则一个部分的改进将帮助所有的部分。改进文件系统则每个人从中受益而且是透明的。如果每个项目使用不同的文件系统则在整个堆栈中享受不到持续增加的改进

6，构建自管理系统让你没必要让系统关机。这允许你更容易在服务器之间平衡资源、动态添加更大的容量、让机器离线和优雅的处理升级

7，创建可进化的基础组织，并行的执行消耗时间的操作并采取较好的方案

8，不要忽略学院。学院有许多没有转变为产品的好主意。Most of what Google has done has prior art, just not prior large scale deployment.

9，考虑压缩。当你有许多CPU而IO有限时压缩是一个好的选择。

http://blog.daviesliu.net/2006/09/09/010620/

Lighttpd+Squid+Apache搭建高效率Web服务器

架构原理

Apache通常是开源界的首选Web服务器，因为它的强大和可靠，已经具有了品牌效应，可以适用于绝大部分的应用场合。但是它的强大有时候却显得笨重，配置文件得让人望而生畏，高并发情况下效率不太高。而轻量级的Web服务器Lighttpd却是后起之秀，其静态文件的响应能力远高于 Apache，据说是Apache的2-3倍。Lighttpd的高性能和易用性，足以打动我们，在它能够胜任的领域，尽量用它。Lighttpd对 PHP的支持也很好，还可以通过Fastcgi方式支持其他的语言，比如Python。

毕竟Lighttpd是轻量级的服务器，功能上不能跟Apache比，某些应用无法胜任。比如Lighttpd还不支持缓存，而现在的绝大部分站点都是用程序生成动态内容，没有缓存的话即使程序的效率再高也很难满足大访问量的需求，而且让程序不停的去做同一件事情也实在没有意义。首先，Web程序是需要做缓存处理的，即把反复使用的数据做缓存。即使这样也还不够，单单是启动Web处理程序的代价就不少，缓存最后生成的静态页面是必不可少的。而做这个是 Squid的强项，它本是做代理的，支持高效的缓存，可以用来给站点做反向代理加速。把Squid放在Apache或者Lighttpd的前端来缓存 Web服务器生成的动态内容，而Web应用程序只需要适当地设置页面实效时间即可。

即使是大部分内容动态生成的网站，仍免不了会有一些静态元素，比如图片、JS脚本、CSS等等，将Squid放在Apache或者Lighttp 前端后，反而会使性能下降，毕竟处理HTTP请求是Web服务器的强项。而且已经存在于文件系统中的静态内容再在Squid中缓存一下，浪费内存和硬盘空间。因此可以考虑将Lighttpd再放在Squid的前面，构成 Lighttpd+Squid+Apache的一条处理链，Lighttpd在最前面，专门用来处理静态内容的请求，把动态内容请求通过proxy模块转发给Squid，如果Squid中有该请求的内容且没有过期，则直接返回给Lighttpd。新请求或者过期的页面请求交由Apache中Web程序来处理。经过Lighttpd和Squid的两级过滤，Apache需要处理的请求将大大减少，减少了Web应用程序的压力。同时这样的构架，便于把不同的处理分散到多台计算机上进行，由Lighttpd在前面统一把关。

在这种架构下，每一级都是可以进行单独优化的，比如Lighttpd可以采用异步IO方式，Squid可以启用内存来缓存，Apache可以启用MPM 等，并且每一级都可以使用多台机器来均衡负载，伸缩性很好。

08:31

浏览 (798)

评论 (0)

分类: Java

收藏Spinner

2009 - 10 - 31

缩略显示

Java基础学习笔记

文章分类:Java编程

1.Java语言的特点：面向对象，跨平台，多线程

2.Java运行环境：JVM+Java API

3.数据类型：

boolean 1

char 16

byte 8

short 16

int 32

long 64

float 32

double 64

4.面向对象：

封装：类

继承：单一继承

多态：重写(Overridding)与重载(Overloading)

在子类中，方法的参数个数和返回值都与父类相同，称为重写

在同一个类中，方法的参数不同，称为重载

5.作用域：

private 子类不能继承

default 如果不在同一包中，子类不能继承

protected

public

6.垃圾回收原理：有向图原则(另一种为计数器类型)，可能会有内存泄漏,计数器类型与有向图类型相比：效率更高，但是精度偏低，因为它很难发现变量循环引用的问题

会有内存泄漏的代码：

Vector v=new Vector(10)

for (int i=0;i<10;i++){

Object o=new Object();

v.add(o);

o=null;

}

解决方法：

v=null;

7.栈与堆：原始数据类型存放在栈里，对象数据类型存放在堆中，栈中存放堆中的地址。

8.变量比较：==比较的是栈，原始数据类型相同，但是引用类型值不等，equals比较的是堆，特例是string，java.lang.String类是final类型的,因此不可以继承这个类、不能修改这个类。

9.变量复制：原始数据类型复制的是值，引用数据类型复制的是栈中的引用，所以当一个修改的时候，会影响另一个。对象复制：Object.clone()方法

10.引用传递和值传递：原始数据类型传递的是值，引用数据类型传递的是栈中的引用

11.异常：(Java.lang.Exception)检查时异常，(Java.lang.RuntimeException)运行时异常，((java.lang.Error)错误，检查时异常必须处理

12.集合：collection: set:HashSet TreeSet

list:ArrayList LinkedList Vector Stack

map:HashMap TreeMap HashTable Properties

13.Set里的元素是不能重复的,什么方法来区分重复与否呢? 是用==还是

equals()? 它们有何区别?

Set里的元素是不能重复的,那么用iterator()方法来区分重复与否。equals()是判读两

个Set是否相等。

equals()和==方法决定引用值是否指向同一对象equals()在类中被覆盖,为的是当两个

分离的对象的内容和类型相配的话,返回真值

14.ArrayList, Vector, LinkedList的存储性能和特性

ArrayList和Vector都是使用数组方式存储数据,此数组元素数大于实际存储的数据

以便增加和插入元素,它们都允许直接按序号索引元素,但是插入元素要涉及数组元

素移动等内存操作,所以索引数据快而插入数据慢,Vector由于使用了synchronized方

法(线程安全),通常性能上较ArrayList差,而LinkedList使用双向链表实现存储,按序

号索引数据需要进行前向或后向遍历,但是插入数据时只需要记录本项的前后项即

可,所以插入速度较快。

15、HashMap和Hashtable的区别。

HashMap是Hashtable的轻量级实现(非线程安全的实现),他们都完成了Map接口,

主要区别在于HashMap允许空(null)键值(key),由于非线程安全,效率上可能高于

Hashtable。

HashMap允许将null作为一个entry的key或者value,而Hashtable不允许。

HashMap把Hashtable的contains方法去掉了,改成containsvalue和containsKey。因为

contains方法容易让人引起误解。

Hashtable继承自Dictionary类,而HashMap是Java1.2引进的Map interface的一个实现。

最大的不同是,Hashtable的方法是Synchronize的,而HashMap不是,在多个线程访问

Hashtable时,不需要自己为它的方法实现同步,而HashMap 就必须为之提供外同步。

Hashtable和HashMap采用的hash/rehash算法都大概一样,所以性能不会有很大的差异。

22:23

浏览 (477)

评论 (1)

分类: Java

收藏Spinner

2009 - 07 - 02

缩略显示

定时执行任务的三种方法

定时执行任务的三种方法:

1）java.util.Timer.

2）ServletContextListener.

3）org.springframework.scheduling.timer.ScheduledTimerTask

1）java.util.Timer

这个方法应该是最常用的，不过这个方法需要手工启动你的任务：

Timer timer=new Timer();

timer.schedule(new ListByDayTimerTask(),10000,86400000);

这里的ListByDayTimerTask类必须extends TimerTask里面的run()方法。

2）ServletContextListener

这个方法在web容器环境比较方便，这样，在web server启动后就可以自动运行该任务，不需要手工操作。

将ListByDayListener implements ServletContextListener接口，在contextInitialized方法中加入启动Timer的代码，在 contextDestroyed方法中加入cancel该Timer的代码；然后在web.xml中，加入listener:

<-listener>

<-listener-class>com.qq.customer.ListByDayListener</listener-class>

<-/listener>

3)org.springframework.scheduling.timer.ScheduledTimerTask如果你用spring，那么你不需要写Timer类了，在schedulingContext-timer.xml中加入下面的内容就可以了：

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE beans PUBLIC "-//SPRING//DTD BEAN//EN" "http://www.springframework.org/dtd/spring-beans.dtd">

<beans>

<bean id="timer" class="org.springframework.scheduling.timer.TimerFactoryBean">

<property name="scheduledTimerTasks">

<list>

<ref local="MyTimeTask1"/>

</list>

</property>

</bean>

<bean id="MyTimeTask" class="com.qq.timer.ListByDayTimerTask"/>

<bean id="MyTimeTask1" class="org.springframework.scheduling.timer.ScheduledTimerTask">

<property name="timerTask">

<ref bean="MyTimeTask"/>

</property>

<property name="delay">

<value>10000</value>

</property>

<property name="period">

<value>86400000</value>

</property>

</bean>

</beans>

以上内容转载自：http://dev.csdn.net/author/xtpdcsse/ec8e8080a5b04fa79e7d4828bc807d3f.html

下面是我的实现。

1）利用java.util.Timer. 代码如下

<1>StartThread.java

[java] view plaincopy

package com.jview.main;<BR>

import java.util.Date;<BR>

import java.util.Timer;<BR>

import java.util.TimerTask;<BR>

import org.apache.log4j.Logger;<BR>

public class StartThreadextends Thread {<BR>

private static Logger logger = Logger.getLogger("StartThread");<BR>

public static final int EXECUTE\_CYC = 86400000; //24\*60\*60\*1000毫秒<BR>

int startH = 9;<BR>

int startM = 52;<BR>

private Timer \_timer ;<BR>

private Date \_statDate;<BR>

private Date \_nowDate;<BR>

public StartThread(){<BR>

\_nowDate = new Date();<BR>

\_timer = new Timer();<BR>

\_statDate = newDate(\_nowDate.getYear(),\_nowDate.getMonth(),\_nowDate.getDate(),startH,startM);<BR>

}<BR>

public void StartUp(){<BR>

\_timer.schedule(<BR>

new TimerTask() {<BR>

public void run()<BR>

{<BR>

logger.info("开始统计...");<BR>

try {<BR>

Class.forName("com.jview.stat.StatPlan").newInstance();<BR>

} catch (InstantiationException e) {<BR>

// TODO Auto-generated catch block<BR>

e.printStackTrace();<BR>

} catch (IllegalAccessException e) {<BR>

// TODO Auto-generated catch block<BR>

e.printStackTrace();<BR>

} catch (ClassNotFoundException e) {<BR>

// TODO Auto-generated catch block<BR>

e.printStackTrace();<BR>

}<BR>

}<BR>

},\_statDate,EXECUTE\_CYC);<BR>

}<BR>

public static void main(String[] args) {<BR>

StartThread \_statUp = new StartThread();<BR>

\_statUp.StartUp();<BR>

} <BR>

}

[java] view plaincopy

package com.jview.main;

import java.util.Date;

import java.util.Timer;

import java.util.TimerTask;

import org.apache.log4j.Logger;

public class StartThread extends Thread {

private static Logger logger = Logger.getLogger("StartThread");

public static final int EXECUTE\_CYC = 86400000; //24\*60\*60\*1000毫秒

int startH = 9;

int startM = 52;

private Timer \_timer ;

private Date \_statDate;

private Date \_nowDate;

public StartThread(){

\_nowDate = new Date();

\_timer = new Timer();

\_statDate = new Date(\_nowDate.getYear(),\_nowDate.getMonth(),\_nowDate.getDate(),startH,startM);

}

public void StartUp(){

\_timer.schedule(

new TimerTask() {

public void run()

{

logger.info("开始统计...");

try {

Class.forName("com.jview.stat.StatPlan").newInstance();

} catch (InstantiationException e) {

// TODO Auto-generated catch block

e.printStackTrace();

} catch (IllegalAccessException e) {

// TODO Auto-generated catch block

e.printStackTrace();

} catch (ClassNotFoundException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

},\_statDate,EXECUTE\_CYC);

}

public static void main(String[] args) {

StartThread \_statUp = new StartThread();

\_statUp.StartUp();

}

}

<2>StatPlan.java

[java] view plaincopy

package com.jview.stat;<BR>

import java.util.Calendar;<BR>

import org.apache.log4j.Logger;<BR>

public class StatPlan {<BR>

<BR>

<BR>

private static Logger logger = Logger.getLogger("StatPlan");<BR>

private int i = 0;<BR>

private StatPlanService \_sps ;<BR>

public StatPlan(){<BR>

\_sps = new StatWeekPlan();<BR>

statPlan();<BR>

}<BR>

<BR>

public void statPlan(){<BR>

Calendar \_c = Calendar.getInstance();<BR>

logger.info("stat plan ... 执行"+ i +"次,时间："+\_c.getTime());<BR>

\_sps.StatPlan();<BR>

i++;<BR>

for(int i = 0; i<9999;i++){<BR>

if(i==0 || i==9998){<BR>

logger.info(""+i);<BR>

}<BR>

}<BR>

}<BR>

}

[java] view plaincopy

package com.jview.stat;

import java.util.Calendar;

import org.apache.log4j.Logger;

public class StatPlan {

private static Logger logger = Logger.getLogger("StatPlan");

private int i = 0;

private StatPlanService \_sps ;

public StatPlan(){

\_sps = new StatWeekPlan();

statPlan();

}

public void statPlan(){

Calendar \_c = Calendar.getInstance();

logger.info("stat plan ... 执行"+ i +"次,时间："+\_c.getTime());

\_sps.StatPlan();

i++;

for(int i = 0; i<9999;i++){

if(i==0 || i== 9998){

logger.info(""+i);

}

}

}

}

<3>StatWeekPlan .java

[java] view plaincopy

import org.apache.log4j.Logger;<BR>

<BR>

public class StatWeekPlanextends StatPlanService {<BR>

private static Logger logger = Logger.getLogger("StatWeekPlan");<BR>

public void StatPlan(){<BR>

logger.info("this is statWeekPlan");<BR>

}<BR>

}<BR>

<4>StatPlanService .java<BR>

public class StatPlanService {<BR>

public void StatPlan(){ <BR>

}<BR>

}

[java] view plaincopy

import org.apache.log4j.Logger;

public class StatWeekPlan extends StatPlanService {

private static Logger logger = Logger.getLogger("StatWeekPlan");

public void StatPlan(){

logger.info("this is statWeekPlan");

}

}

<4>StatPlanService .java

public class StatPlanService {

public void StatPlan(){

}

}

2）ServletContextListener.实现

<1> SysStatListener .java

[java] view plaincopy

import java.util.Date;<BR>

import java.util.Timer;<BR>

import java.util.TimerTask;<BR>

import javax.servlet.ServletContextEvent;<BR>

import javax.servlet.ServletContextListener;<BR>

import org.apache.log4j.Logger;<BR>

public class SysStatListenerimplements ServletContextListener{<BR>

private static Logger logger = Logger.getLogger("ListByDayListener");<BR>

private StatTask \_sTask;<BR>

public void contextDestroyed(ServletContextEvent scevent) {<BR>

String status = "停止系统统计线程";<BR>

scevent.getServletContext().log(status);<BR>

logger.info(status);<BR>

\_sTask.shutDown();<BR>

}<BR>

public void contextInitialized(ServletContextEvent scevent) {<BR>

String status = "启动系统统计线程";<BR>

scevent.getServletContext().log(status);<BR>

logger.info(status);<BR>

\_sTask = new StatTask();<BR>

\_sTask.startUp();<BR>

}<BR>

}

[java] view plaincopy

import java.util.Date;

import java.util.Timer;

import java.util.TimerTask;

import javax.servlet.ServletContextEvent;

import javax.servlet.ServletContextListener;

import org.apache.log4j.Logger;

public class SysStatListener implements ServletContextListener{

private static Logger logger = Logger.getLogger("ListByDayListener");

private StatTask \_sTask;

public void contextDestroyed(ServletContextEvent scevent) {

String status = "停止系统统计线程";

scevent.getServletContext().log(status);

logger.info(status);

\_sTask.shutDown();

}

public void contextInitialized(ServletContextEvent scevent) {

String status = "启动系统统计线程";

scevent.getServletContext().log(status);

logger.info(status);

\_sTask = new StatTask();

\_sTask.startUp();

}

}

<2> StatTask .java

[java] view plaincopy

import java.util.Date;<BR>

import java.util.Timer;<BR>

import java.util.TimerTask;<BR>

import org.apache.log4j.Logger;<BR>

public class StatTaskextends Thread {<BR>

private static Logger logger = Logger.getLogger("StartThread");<BR>

public staticfinal int EXECUTE\_CYC =86400000;<BR>

int startH = 9;<BR>

int startM = 52;<BR>

private Timer \_timer ;<BR>

private Date \_statDate;<BR>

private Date \_nowDate;<BR>

public StatTask(){<BR>

\_nowDate = new Date();<BR>

\_timer = new Timer();<BR>

\_statDate = newDate(\_nowDate.getYear(),\_nowDate.getMonth(),\_nowDate.getDate(),startH,startM);<BR>

}<BR>

public void startUp(){<BR>

\_timer.schedule(<BR>

new TimerTask() {<BR>

public void run()<BR>

{<BR>

logger.info("开始统计...");<BR>

try {<BR>

Class.forName("com.jview.stat.StatPlan").newInstance();<BR>

} catch (InstantiationException e) {<BR>

// TODO Auto-generated catch block<BR>

e.printStackTrace();<BR>

} catch (IllegalAccessException e) {<BR>

// TODO Auto-generated catch block<BR>

e.printStackTrace();<BR>

} catch (ClassNotFoundException e) {<BR>

// TODO Auto-generated catch block<BR>

e.printStackTrace();<BR>

}<BR>

}<BR>

},\_statDate,EXECUTE\_CYC);<BR>

}<BR>

public void shutDown(){<BR>

\_timer.cancel();<BR>

}

[java] view plaincopy

import java.util.Date;

import java.util.Timer;

import java.util.TimerTask;

import org.apache.log4j.Logger;

public class StatTask extends Thread {

private static Logger logger = Logger.getLogger("StartThread");

public static final int EXECUTE\_CYC = 86400000;

int startH = 9;

int startM = 52;

private Timer \_timer ;

private Date \_statDate;

private Date \_nowDate;

public StatTask(){

\_nowDate = new Date();

\_timer = new Timer();

\_statDate = new Date(\_nowDate.getYear(),\_nowDate.getMonth(),\_nowDate.getDate(),startH,startM);

}

public void startUp(){

\_timer.schedule(

new TimerTask() {

public void run()

{

logger.info("开始统计...");

try {

Class.forName("com.jview.stat.StatPlan").newInstance();

} catch (InstantiationException e) {

// TODO Auto-generated catch block

e.printStackTrace();

} catch (IllegalAccessException e) {

// TODO Auto-generated catch block

e.printStackTrace();

} catch (ClassNotFoundException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

},\_statDate,EXECUTE\_CYC);

}

public void shutDown(){

\_timer.cancel();

}

<3>在web.xml中添加下面的内容（注：下面的内容放到<filter-mapping>的后面）

<listener>

<listener-class>com.jview.auto.stat.SysStatListener </listener-class>

</listener>

22:58

浏览 (583)

评论 (0)

分类: Java

收藏Spinner

2009 - 03 - 31

缩略显示

13个代码注释的小技巧

这篇文章是由José M. Aguilar在他卓越的博客中以西班牙语的形式首发，其后Timm Martin在获得Aguilar先生的授权下，对该文章进行翻译、修改，并且在DevTopics上发布。

以下13个小技巧可以使得你的代码在长时间内依然能够保持容易理解和维护。

1. 对不同级别的代码进行注释

对于不同级别的代码块，要使用统一的方法来进行注释。例如：

对于每一个类，需要包含一段简明扼要的描述，作者和上一次修改的时间

对于每一个方法，需要包含这个方法的用途，功能，参数以及返回结果

当你在一个团队里面的时候，采用一套注释的标准是非常重要的。当然，使用一种大家都认可的注释约定和工具（例如C#的XML注释和Java的Javadoc）在一定程度上能推动这项任务。

2. 使用段落注释

首先把代码块分解成多个“段落”，每一个段落都执行单一的任务；然后在每一个“段落”开始之前添加注释，告诉阅读代码的人接下来的这段代码是干什么用的

// 检查所有记录都是正确的

foreach (Record record in records)

{

if (rec.checkStatus()==Status.OK)

{

. . .

}

}

// 现在开始进行处理

Context ctx = new ApplicationContext();

ctx.BeginTransaction();

. . .

3. 对齐注释行

对于那些在行末写有注释的代码，应该对齐注释行来使得方便阅读

const MAX\_ITEMS = 10; // maximum number of packets

const MASK = 0x1F; // mask bit TCP

有些开发人员使用tab来对齐注释，而另外一些人会用空格来对齐。由于tab在不同的编辑器和集成开发环境中会有所不同，所以最佳的方法是使用空格来对齐注释行。

4. 不要侮辱阅读者的智慧

要避免没用的注释，例如

if (a == 5) //如果a等于5

counter = 0 //把counte设为0

这不单把时间浪费在写没用的注释上面，同时也在分散读者的注意力。

5. 要有礼貌

应当避免没有礼貌的注释，例如“要注意一些愚蠢的用户会输入一个负数”，或者“修正由菜鸟工程师写的愚蠢得可怜的代码而导致的副作用”。这样的注释对于代码的写注释的人来说并没有任何好处，同时你永远都不会知道将来这些注释会被谁来阅读，你的老板，一个客户或者是刚才被你数落的愚蠢得可怜的工程师。

6. 直截了当

不要在注释里面写过多的废话。避免在注释里面卖弄ASCII艺术，写笑话，作诗和过于冗长。简而言之就是保持注释的简单和直接。

7. 使用统一的风格

有些人觉得注释应该让非程序员也能看懂。另外一些人觉得注释需要面对的读者只是程序员。无论如何，正如Successful Strategies for Commenting Code中所说的，最重要的是注释的风格需要统一，并且总是面向相同的读者。就自己而论，我怀疑非程序员是否会去读代码，所以我觉得注释应该面向程序员来写。

8. 在内部使用特殊的标签

当你在一个团队里工作的时候，采用一组一致的标签能帮助不同的程序员沟通。例如，很多团队会采用“TODO”标签来表示一段尚未完成的代码

int Estimate(int x, int y)

{

// TODO: implement the calculations

return 0;

}

标签注释并不会解释代码，它们寻求注意或者是传递信息。但是如果适当地使用这种技术，要记住跟进这段代码并且完成该标签传递的任务。

9. 在写代码的同时添加注释

当你在写代码而且记忆犹新的同时就添加注释。如果等到项目后期才添加注释，会让你事倍功半。“我没有时间写注释”，“我的时间很紧迫”和“项目已经延迟了”，这些都是不写注释的常见借口。有些工程师觉最佳的解决方法是“注释先行”。例如：

public void ProcessOrder()

{

// Make sure the products are available

// Check that the customer is valid

// Send the order to the store

// Generate bill

}

10. 把自己想象为注释的读者（事实上就是如此）

当你正在给代码写注释的时候，不仅仅为日后维护你的代码的开发者考虑，同时也设想一下如果自己就是注释的读者。Phil Haack曾经说过：

“一旦一行代码被敲到文件中, 你就已经要开始维护那一行代码了。”

所以，我们自己就是好（或者坏）注释的第一个受益者（或者受害者）。

11. 更新代码的时候要更新注释

如果注释没有随着代码的修改而更新，那么这些注释将是毫无意义的。代码和注释需要同步，否则注释只会让维护代码的开发者更加痛苦。需要特别注意的是，一些重构的工具会自动更新代码，但是却没有自动更新注释，那么注释就自然而然地过期作废了。

12. 良好可读性代码是注释的金科玉律

对于很多开发者来说，一个基本的原则就是：让代码自己描述自己。虽然有人怀疑这是由不喜欢写注释的程序员所倡导的一场运动，但是无需解释的代码有很大的好处，这些代码更加容易理解甚至让注释变得没有必要。例如，在我的文章Fluid Interfaces中就给大家展示了什么是清晰的无需解释的代码。

Calculator calc = new Calculator();

calc.Set(0);

calc.Add(10);

calc.Multiply(2);

calc.Subtract(4);

Console.WriteLine( “Result: {0}”, calc.Get() );

在这个例子里面，注释就像是违反了第4条技巧那样，变得毫无必要。要写出可读性好的代码，你需要使用适当的命名方式（在经典的 Ottinger’s Rules中有阐述），保证恰当的缩进，并且采用编码风格指导。如果代码不遵守这条技巧，那么注释看起来就好像是为自己不好的代码的写道歉信一样。

13. 跟你的同事分享这些技巧

虽然从第10条技巧中我们已经知道了自己就是好注释的得益者，但是这些技巧对于所有的开发者来说都是很有帮助的，尤其是整个团队都有相同共识的情况下。因此，大方地跟你的同事去分享这些技巧，让我们写出更加容易理解和维护的代码。

08:43

浏览 (469)

评论 (0)

分类: Java

收藏Spinner

2009 - 03 - 10

缩略显示

log4j配置文件的含义

今天试着用了一下log4j来处理java中的日志，感觉良好，顺便记录一下log4j的配置文件log4j.properties各语句的含义。

这是一个数据库配置文件

#这是一个配置文件实例,PropertyConfigurator将使用这个文件 :

#声明一个appender变量名为JDBC

log4j.rootLogger=DEBUG, JDBC

#JDBC是一个JDBCAppender类，这个类可以写消息到数据库

log4j.appender.JDBC=com.benqguru.palau.log.jdbc.test.JDBCAppender

#1.连接数据库的数据库选项

log4j.appender.JDBC.url=jdbc:mysql://localhost:3306/logtest

log4j.appender.JDBC.username=root

log4j.appender.JDBC.password=

#2.指定你自己的JDBCConnectionHandler的连接器选项

log4j.appender.JDBC.connection\_class=com.benqguru.palau.log.jdbc.test.MyConnectionHandler

#3.指定一个静态的SQL语句的SQL选项，这个语句将在每次消息事件发生时被执行

log4j.appender.JDBC.sql=INSERT INTO LOGTEST (id, msg, created\_on, created\_by) VALUES (1, @MSG@, sysdate, 'me')

#4. 指定数据库中一个表的表选项。

log4j.appender.JDBC.table=logtest

#5.描述表的重要列的列选项(非空列是必须被描述的)

log4j.appender.JDBC.columns=id\_seq~EMPTY id~ID~MyIDHandler msg~MSG created\_on~TIMESTAMP created\_by~STATIC~Thomas Fenner

#6.定义消息布局器的布局器选项(可选)

log4j.appender.JDBC.layout=org.apache.log4j.PatternLayout

log4j.appender.JDBC.layout.ConversionPattern=%m

#7.定义消息事件缓冲器的大小的缓冲器选项(可选)

log4j.appender.JDBC.buffer\_size=1

#8.定义自动提交的提交选项(可选)

log4j.appender.JDBC.docommit=N

##########下面是英文说明#############

#Date - %d{DATE}[slf5s.DATE]

#Priority - %p[slf5s.PRIORITY]

#NDC - %x[slf5s.NDC]

#Thread - %t[slf5s.THREAD]

#Category - %c[slf5s.CATEGORY]

#Location - %l[slf5s.LOCATION]

#Message - %m[slf5s.MESSAGE]

#

#log4j.appender.R.layout.ConversionPattern=[slf5s.start]%d{DATE}[slf5s.DATE]%n/

# %p[slf5s.PRIORITY]%n%x[slf5s.NDC]%n%t[slf5s.THREAD]%n/

# %c[slf5s.CATEGORY]%n%l[slf5s.LOCATION]%n%m[slf5s.MESSAGE]%n%n

##########下面是中文说明#############

#%m 输出代码中指定的消息

#%p 输出优先级，即DEBUG，INFO，WARN，ERROR，FATAL

#%r 输出自应用启动到输出该log信息耗费的毫秒数

#%c 输出所属的类目，通常就是所在类的全名

#%t 输出产生该日志事件的线程名

#%n 输出一个回车换行符，Windows平台为“/r/n”，Unix平台为“/n”

#%d 输出日志时间点的日期或时间，默认格式为ISO8601，也可以在其后指定格式，

#比如：%d{yyy MMM dd HH:mm:ss,SSS}，输出类似：2002年10月18日 22：10：28，921

#%l 输出日志事件的发生位置，包括类目名、发生的线程，以及在代码中的行数。举例：Testlog4.main(TestLog4.java:10)

这是一个普通的配置文件

#log4j会解析这个文件

log4j.debug=false

#暂时还不清楚这两个属性的作用

log4j.disableOverride=true

log4j.disable=INFO

#设置记录所有类的日志的优先级别

log4j.rootLogger=DEBUG,dest1,dest2

#设置这个包记录日志为假的话

#dist1,dist2就不会记录com.benqguru.commons.logging.test.LoggingSample的日志,只有dist3会记录.

#反之,会记录,这样就会重复记录

log4j.additivity.com.benqguru.commons.logging.test.LoggingSample=false

#特别指定某个特殊包的日志级别和目标设备

log4j.category.com.benqguru.commons.logging.test.LoggingSample=WARN, dest3

#这个目标设备用来debug

log4j.appender.dest1=org.apache.log4j.ConsoleAppender

#log4j.appender.dest1.layout=org.apache.log4j.SimpleLayout

log4j.appender.dest1.layout=org.apache.log4j.PatternLayout

#%-4r %-5p [%t] %37c %3x - %m%n

log4j.appender.dest1.layout.ConversionPattern=%d %p %c - <%m> (%F.%M:%L) %t%n

#这个目标设备用来trace

log4j.appender.dest2=org.apache.log4j.RollingFileAppender

#该文件记录日志的级别(INFO以下的级别不记录)

log4j.appender.dest2.Threshold=INFO

#文件保存路径

log4j.appender.dest2.File=c:/log4jlog.htm

#是否往文件追加信息

log4j.appender.dest2.Append=true

#设置文件最大值

log4j.appender.dest2.MaxFileSize=5KB

#设置备份文件的最大数量

log4j.appender.dest2.MaxBackupIndex=10

#使用一个html格式来记录日志

log4j.appender.dest2.layout=org.apache.log4j.HTMLLayout

#是否打印该消息的代码行

log4j.appender.dest2.layout.LocationInfo=true

#设置该日志的html的标题

log4j.appender.dest2.layout.Title=My app title

#这个目标设备用来trace指定类或包

log4j.appender.dest3=org.apache.log4j.DailyRollingFileAppender

#log4j.appender.dest3.Threshold=DEBUG

log4j.appender.dest3.File=c:/SpecPackageLog.htm

log4j.appender.dest3.Append=false

log4j.appender.dest3.layout=org.apache.log4j.HTMLLayout

log4j.appender.dest3.layout.LocationInfo=true

log4j.appender.dest3.layout.Title=My app title

13:37

浏览 (896)

评论 (2)

分类: Java

收藏Spinner

2009 - 03 - 10

缩略显示

Java Regex(Java正则表达式)

在Sun的API中对在对 “最大匹配Greedy”“最小匹配Reluctant”“完全匹配Possessive”的描述，不能让我明白他们有什么区别，现在将我对这三种匹配模式的理解写出来，供大家参考。

1、Greediness（贪心）匹配：

X?、X\*、X+、X{n，}都是最大匹配。例如你要用“<.+>”去匹配“a<tr>aava </tr>abb”，也许你所期待的结果是想匹配“<tr>”，但是实际结果却会匹配到“<tr>aava </tr>”。这是为什么呢？下面我们跟踪下最大匹配的匹配过程。

①“<”匹配字符串的“<”。②“.+”匹配字符串的“tr>aava </tr>ab”，在进行最大匹配时，它把两个“>”都匹配了，它匹配了所有字符，直到文本的最后字符“b” ③这时，发现不能成功匹配“>”，开始按原路回退，用“a”与“>”匹配，直到“ab”前面的“>”匹配成功。

2、Reluctant(Laziness)最小匹配

X?、X\*、X+、X{n，}都是最大匹配。好，加个？就成了Laziness匹配。例如X??、X\*?、X+?、X{n，}?都是最小匹配，其实X{n，m}?和X{n }?有些多余。

最小匹配意味者，.+? 匹配一个字符后，马上试一试>的匹配可能，失败了，则.+? 再匹配一个字符，再马上试一试>的匹配可能。JDK文档中Greedy 和 Reluctant，它是以eat一口来隐喻的，所以翻译成贪吃和（勉强的）厌食最贴切了。不过我喜欢最大匹配、最小匹配的说法。

3、Possessive完全匹配

与最大匹配不同，还有一种匹配形式：X?+、X\*+、X++、X{n，}+等，成为完全匹配。它和最大匹配一样，一直匹配所有的字符，直到文本的最后，但它不由原路返回。也就是说，一口匹配，搞不定就算了，到也干脆，偶喜欢。

Use a possessive quantifier for situations where you want to seize all of something without ever backing off; it will outperform the equivalent greedy quantifier in cases where the match is not immediately found.

13:35

浏览 (2123)

评论 (0)

分类: Java

收藏Spinner

2009 - 03 - 05

缩略显示

学习正则表达式最佳的材料

[java] view plaincopy

import java.util.regex.Matcher;<BR>

import java.util.regex.Pattern;<BR>

<BR>

public class RegularExpression {<BR>

<BR>

public static void main(String[] args) {<BR>

// 简单认识正则表达式的概念<BR>

<BR>

// p("abc".matches("..."));<BR>

// p("a8729a".replaceAll("//d", "-"));<BR>

// Pattern p = Pattern.compile("[a-z]{3}");<BR>

// Matcher m = p.matcher("fgh");<BR>

// p(m.matches());<BR>

// p("aaas".matches("[a-z]{3,}"));<BR>

<BR>

// 初步认识. \* + ?<BR>

<BR>

// p("a".matches("."));<BR>

// p("aa".matches("aa"));<BR>

// p("aaaa".matches("a\*"));<BR>

// p("aaaa".matches("a+"));<BR>

// p("".matches("a\*"));<BR>

// p("aaaa".matches("a?"));<BR>

// p("".matches("a?"));<BR>

// p("a".matches("a?"));<BR>

// p("214523145234532".matches("//d{3,100}"));<BR>

// p("192.168.0.231".matches("//d{1,3}//.//d{1,3}//.//d{1,3}//.//d{1,3}"));<BR>

// p("192".matches("[0-2][0-9][0-9]"));<BR>

<BR>

// 范围<BR>

<BR>

// p("a".matches("[abc]"));<BR>

// p("a".matches("[^abc]"));<BR>

// p("A".matches("[a-zA-Z]"));<BR>

// p("a".matches("[a-z]|[A-Z]"));<BR>

// p("A".matches("[a-z[A-Z]]"));<BR>

// p("R".matches("[A-Z&&[RFG]]"));<BR>

<BR>

// 认识/s /w /d /<BR>

<BR>

// p(" /n/r/t".matches("//s{4}"));<BR>

// p(" ".matches("//S"));<BR>

// p("a\_8".matches("//w{3}"));<BR>

// p("abc888&^%".matches("[a-z]{1,3}//d+[&^#%]+"));<BR>

// p("//".matches("////"));<BR>

<BR>

// POSIX Style<BR>

<BR>

// p("a".matches("//p{Lower}"));<BR>

<BR>

// boundary<BR>

<BR>

// p("hello sir".matches("^h.\*"));<BR>

// p("hello sir".matches(".\*ir$"));<BR>

// p("hello sir".matches("^h[a-z]{1,3}o//b.\*"));<BR>

// p("hellosir".matches("^h[a-z]{1,3}o//b.\*")); // whilte lines<BR>

// p("/n".matches("^[//s&&[^//n<BR>

// p("aaa 8888c".matches(".\*//d{4}."));<BR>

// p("aaa8888c".matches(".\*//b//d{4}."));<BR>

// p("aaa8888c".matches(".\*//d{4}."));<BR>

// p("aaa 8888c".matches(".\*//b//d{4}."));<BR>

<BR>

// email<BR>

<BR>

// p("asdfasdfsafsf@dsdfsdf.com".matches("[//w[.-]]+@[//w[.-]]+//.[//w]+"));<BR>

<BR>

// matches find lookingAt<BR>

<BR>

// Pattern p = Pattern.compile("//d{3,5}");<BR>

// String s = "123-34345-234-00";<BR>

// Matcher m = p.matcher(s);<BR>

// p(m.matches());<BR>

// m.reset();<BR>

// p(m.find());<BR>

// p(m.start() + "-" + m.end());<BR>

// p(m.find());<BR>

// p(m.start() + "-" + m.end());<BR>

// p(m.find());<BR>

// p(m.start() + "-" + m.end());<BR>

// p(m.find());<BR>

// //p(m.start() + "-" + m.end());<BR>

// p(m.lookingAt());<BR>

// p(m.lookingAt());<BR>

// p(m.lookingAt());<BR>

// p(m.lookingAt());<BR>

<BR>

// replacement<BR>

<BR>

// Pattern p = Pattern.compile("java", Pattern.CASE\_INSENSITIVE);<BR>

// Matcher m = p<BR>

// .matcher("java Java JAVa JaVa IloveJAVA you hateJava afasdfasdf");<BR>

// StringBuffer buf = new StringBuffer();<BR>

// int i = 0;<BR>

// while (m.find()) {<BR>

// i++;<BR>

// if (i % 2 == 0) {<BR>

// m.appendReplacement(buf, "java");<BR>

// } else {<BR>

// m.appendReplacement(buf, "JAVA");<BR>

// }<BR>

// }<BR>

// m.appendTail(buf);<BR>

// p(buf);<BR>

<BR>

// group<BR>

<BR>

// Pattern p = Pattern.compile("(//d{3,5})([a-z]{2})");<BR>

// String s = "123aa-34345bb-234cc-00";<BR>

// Matcher m = p.matcher(s);<BR>

// while (m.find()) {<BR>

// p(m.group());<BR>

// }<BR>

<BR>

// qulifiers<BR>

<BR>

// Pattern p = Pattern.compile(".{3,10}+[0-9]");<BR>

// String s = "aaaa5bbbb68";<BR>

// Matcher m = p.matcher(s);<BR>

// if (m.find())<BR>

// p(m.start() + "-" + m.end());<BR>

// else<BR>

// p("not match!");<BR>

<BR>

// non-capturing groups<BR>

<BR>

// Pattern p = Pattern.compile(".{3}(?=a)");<BR>

// String s = "444a66b";<BR>

// Matcher m = p.matcher(s);<BR>

// while (m.find()) {<BR>

// p(m.group());<BR>

// }<BR>

<BR>

// back refenrences<BR>

<BR>

// Pattern p = Pattern.compile("(//d(//d))//2");<BR>

// String s = "122";<BR>

// Matcher m = p.matcher(s);<BR>

// p(m.matches());<BR>

<BR>

// flags的简写<BR>

Pattern p = Pattern.compile("java", Pattern.CASE\_INSENSITIVE);<BR>

p("Java".matches("(?i)(java)"));<BR>

}<BR>

<BR>

public staticvoid p(Object o) {<BR>

System.out.println(o);<BR>

}<BR>

<BR>

}<BR>

[java] view plaincopy

import java.util.regex.Matcher;

import java.util.regex.Pattern;

public class RegularExpression {

public static void main(String[] args) {

// 简单认识正则表达式的概念

// p("abc".matches("..."));

// p("a8729a".replaceAll("//d", "-"));

// Pattern p = Pattern.compile("[a-z]{3}");

// Matcher m = p.matcher("fgh");

// p(m.matches());

// p("aaas".matches("[a-z]{3,}"));

// 初步认识. \* + ?

// p("a".matches("."));

// p("aa".matches("aa"));

// p("aaaa".matches("a\*"));

// p("aaaa".matches("a+"));

// p("".matches("a\*"));

// p("aaaa".matches("a?"));

// p("".matches("a?"));

// p("a".matches("a?"));

// p("214523145234532".matches("//d{3,100}"));

// p("192.168.0.231".matches("//d{1,3}//.//d{1,3}//.//d{1,3}//.//d{1,3}"));

// p("192".matches("[0-2][0-9][0-9]"));

// 范围

// p("a".matches("[abc]"));

// p("a".matches("[^abc]"));

// p("A".matches("[a-zA-Z]"));

// p("a".matches("[a-z]|[A-Z]"));

// p("A".matches("[a-z[A-Z]]"));

// p("R".matches("[A-Z&&[RFG]]"));

// 认识/s /w /d /

// p(" /n/r/t".matches("//s{4}"));

// p(" ".matches("//S"));

// p("a\_8".matches("//w{3}"));

// p("abc888&^%".matches("[a-z]{1,3}//d+[&^#%]+"));

// p("//".matches("////"));

// POSIX Style

// p("a".matches("//p{Lower}"));

// boundary

// p("hello sir".matches("^h.\*"));

// p("hello sir".matches(".\*ir$"));

// p("hello sir".matches("^h[a-z]{1,3}o//b.\*"));

// p("hellosir".matches("^h[a-z]{1,3}o//b.\*")); // whilte lines

// p("/n".matches("^[//s&&[^//n

// p("aaa 8888c".matches(".\*//d{4}."));

// p("aaa8888c".matches(".\*//b//d{4}."));

// p("aaa8888c".matches(".\*//d{4}."));

// p("aaa 8888c".matches(".\*//b//d{4}."));

// email

// p("asdfasdfsafsf@dsdfsdf.com".matches("[//w[.-]]+@[//w[.-]]+//.[//w]+"));

// matches find lookingAt

// Pattern p = Pattern.compile("//d{3,5}");

// String s = "123-34345-234-00";

// Matcher m = p.matcher(s);

// p(m.matches());

// m.reset();

// p(m.find());

// p(m.start() + "-" + m.end());

// p(m.find());

// p(m.start() + "-" + m.end());

// p(m.find());

// p(m.start() + "-" + m.end());

// p(m.find());

// //p(m.start() + "-" + m.end());

// p(m.lookingAt());

// p(m.lookingAt());

// p(m.lookingAt());

// p(m.lookingAt());

// replacement

// Pattern p = Pattern.compile("java", Pattern.CASE\_INSENSITIVE);

// Matcher m = p

// .matcher("java Java JAVa JaVa IloveJAVA you hateJava afasdfasdf");

// StringBuffer buf = new StringBuffer();

// int i = 0;

// while (m.find()) {

// i++;

// if (i % 2 == 0) {

// m.appendReplacement(buf, "java");

// } else {

// m.appendReplacement(buf, "JAVA");

// }

// }

// m.appendTail(buf);

// p(buf);

// group

// Pattern p = Pattern.compile("(//d{3,5})([a-z]{2})");

// String s = "123aa-34345bb-234cc-00";

// Matcher m = p.matcher(s);

// while (m.find()) {

// p(m.group());

// }

// qulifiers

// Pattern p = Pattern.compile(".{3,10}+[0-9]");

// String s = "aaaa5bbbb68";

// Matcher m = p.matcher(s);

// if (m.find())

// p(m.start() + "-" + m.end());

// else

// p("not match!");

// non-capturing groups

// Pattern p = Pattern.compile(".{3}(?=a)");

// String s = "444a66b";

// Matcher m = p.matcher(s);

// while (m.find()) {

// p(m.group());

// }

// back refenrences

// Pattern p = Pattern.compile("(//d(//d))//2");

// String s = "122";

// Matcher m = p.matcher(s);

// p(m.matches());

// flags的简写

Pattern p = Pattern.compile("java", Pattern.CASE\_INSENSITIVE);

p("Java".matches("(?i)(java)"));

}

public static void p(Object o) {

System.out.println(o);

}

}

14:05

浏览 (630)

评论 (1)

分类: Java

收藏Spinner

2009 - 02 - 06

缩略显示

Java snippets02

[java] view plaincopy

if (Character.isLowerCase(ch[i])) {<BR>

char temp = Character.toUpperCase(ch[i]);<BR>

ch[i] = temp;<BR>

} else if (Character.isUpperCase(ch[i])) {<BR>

char temp = Character.toLowerCase(ch[i]);<BR>

ch[i] = temp;<BR>

}

[java] view plaincopy

if (Character.isLowerCase(ch[i])) {

char temp = Character.toUpperCase(ch[i]);

ch[i] = temp;

} else if (Character.isUpperCase(ch[i])) {

char temp = Character.toLowerCase(ch[i]);

ch[i] = temp;

}

[java] view plaincopy

while (true) {<BR>

flag = 0;<BR>

result = tc.getNum();<BR>

>

input = Integer.parseInt(br.readLine());<BR>

>

while (flag == 0) {<BR>

if (input == result) {<BR>

System.out.println("very good!");<BR>

flag = 1;<BR>

} else {<BR>

flag = 0;<BR>

System.out.println("no,please try again.");<BR>

input = Integer.parseInt(br.readLine());<BR>

}<BR>

}<BR>

}

[java] view plaincopy

while (true) {

flag = 0;

result = tc.getNum();

input = Integer.parseInt(br.readLine());

while (flag == 0) {

if (input == result) {

System.out.println("very good!");

flag = 1;

} else {

flag = 0;

System.out.println("no,please try again.");

input = Integer.parseInt(br.readLine());

}

}

}

[java] view plaincopy

L1: while (true) {<BR>

int question = new Times().question();<BR>

L2: while (true) {<BR>

int answer = scanner.nextInt();<BR>

if (answer == question) {<BR>

System.out.println("very good!");<BR>

continue L1;<BR>

} else {<BR>

System.out.println("no,please try again.");<BR>

continue L2;<BR>

}<BR>

}<BR>

}

[java] view plaincopy

L1: while (true) {

int question = new Times().question();

L2: while (true) {

int answer = scanner.nextInt();

if (answer == question) {

System.out.println("very good!");

continue L1;

} else {

System.out.println("no,please try again.");

continue L2;

}

}

}

[java] view plaincopy

char[] c1 = s1.toCharArray();

[java] view plaincopy

char[] c1 = s1.toCharArray();

[java] view plaincopy

Map<Integer, String> m =new HashMap<Integer, String>();<BR>

m.put(1, "one");<BR>

m.put(2, "two");<BR>

m.put(3, "three");<BR>

Object o[] = m.values().toArray();

[java] view plaincopy

Map<Integer, String> m = new HashMap<Integer, String>();

m.put(1, "one");

m.put(2, "two");

m.put(3, "three");

Object o[] = m.values().toArray();

[java] view plaincopy

GregorianCalendar ca = new GregorianCalendar();<BR>

System.out.println(ca.get(GregorianCalendar.AM\_PM));

[java] view plaincopy

GregorianCalendar ca = new GregorianCalendar();

System.out.println(ca.get(GregorianCalendar.AM\_PM));

14:43

浏览 (360)

评论 (0)

分类: Java

收藏Spinner

2009 - 02 - 06

缩略显示

Java snippets01

[java] view plaincopy

for (int i =0; i < number.length; i++) {<BR>

for (int j = 0; j < number.length - i - 1; j++) {<BR>

if (number[j] < number[j + 1]) {<BR>

int temp = number[j];<BR>

number[j] = number[j + 1];<BR>

number[j + 1] = temp;<BR>

}<BR>

}<BR>

}<BR>

[java] view plaincopy

for (int i = 0; i < number.length; i++) {

for (int j = 0; j < number.length - i - 1; j++) {

if (number[j] < number[j + 1]) {

int temp = number[j];

number[j] = number[j + 1];

number[j + 1] = temp;

}

}

}

[java] view plaincopy

BufferedReader buffer = new BufferedReader(new InputStreamReader(System.in));<BR>

String str = buffer.readLine();

[java] view plaincopy

BufferedReader buffer = new BufferedReader(new InputStreamReader(System.in));

String str = buffer.readLine();

[java] view plaincopy

Scanner scanner = new Scanner(System.in);<BR>

String line = scanner.nextLine();

[java] view plaincopy

Scanner scanner = new Scanner(System.in);

String line = scanner.nextLine();

[java] view plaincopy

InputStreamReader in = null;<BR>

OutputStreamWriter out = null;<BR>

BufferedReader br = null;<BR>

<BR>

in = new InputStreamReader(new FileInputStream(System<BR>

.getProperty("user.dir")<BR>

+ "//old.txt"), "UTF-8");<BR>

br = new BufferedReader(in);<BR>

out = new OutputStreamWriter(new FileOutputStream(<BR>

"C://WINDOWS//new.txt"), "UTF-8");

[java] view plaincopy

InputStreamReader in = null;

OutputStreamWriter out = null;

BufferedReader br = null;

in = new InputStreamReader(new FileInputStream(System

.getProperty("user.dir")

+ "//old.txt"), "UTF-8");

br = new BufferedReader(in);

out = new OutputStreamWriter(new FileOutputStream(

"C://WINDOWS//new.txt"), "UTF-8");

[java] view plaincopy

String[] temp;<BR>

temp = text.split("//s{1,}");// 按照空格分割字符串，多个空格作为一个空格对字符串进行分割

[java] view plaincopy

String[] temp;

temp = text.split("//s{1,}");// 按照空格分割字符串，多个空格作为一个空格对字符串进行分割

[java] view plaincopy

String s = String.valueOf(1);<BR>

if (s.charAt(i) == '1'){}<BR>

[java] view plaincopy

String s = String.valueOf(1);

if (s.charAt(i) == '1'){}

[java] view plaincopy

System.nanoTime();

[java] view plaincopy

System.nanoTime();

[java] view plaincopy

SimpleDateFormat sdt = new SimpleDateFormat("yyyy-MM-DD HH:mm:ss");<BR>

String date = sdt.format(System.currentTimeMillis());

[java] view plaincopy

SimpleDateFormat sdt = new SimpleDateFormat("yyyy-MM-DD HH:mm:ss");

String date = sdt.format(System.currentTimeMillis());

[java] view plaincopy

DecimalFormat df = (DecimalFormat) NumberFormat.getInstance();<BR>

df.applyPattern("00");

[java] view plaincopy

DecimalFormat df = (DecimalFormat) NumberFormat.getInstance();

df.applyPattern("00");

[java] view plaincopy

Calendar current\_time = new GregorianCalendar();<BR>

Calendar target\_time = new GregorianCalendar(2009, Calendar.MAY,1);<BR>

long diffMillis = target\_time.getTimeInMillis()<BR>

- current\_time.getTimeInMillis();<BR>

long diffDays = diffMillis / (24 \*60 \* 60 \* 1000) + 1;<BR>

SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-DD");<BR>

<BR>

System.out.println("距" + sdf.format(target\_time.getTimeInMillis())<BR>

+ "还有：" + diffDays + "天");

[java] view plaincopy

Calendar current\_time = new GregorianCalendar();

Calendar target\_time = new GregorianCalendar(2009, Calendar.MAY, 1);

long diffMillis = target\_time.getTimeInMillis()

- current\_time.getTimeInMillis();

long diffDays = diffMillis / (24 \* 60 \* 60 \* 1000) + 1;

SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-DD");

System.out.println("距" + sdf.format(target\_time.getTimeInMillis())

+ "还有：" + diffDays + "天");

[java] view plaincopy

Arrays.sort(array);

[java] view plaincopy

Arrays.sort(array);

[java] view plaincopy

StringBuilder s1 = new StringBuilder(temp);<BR>

// StringBuffer s1 = new StringBuffer(temp);<BR>

System.out.println(s1.reverse());

[java] view plaincopy

StringBuilder s1 = new StringBuilder(temp);

// StringBuffer s1 = new StringBuffer(temp);

System.out.println(s1.reverse());

13:28

浏览 (395)

评论 (0)

分类: Java

收藏Spinner

2009 - 02 - 06

缩略显示

双色球算号器

我是个彩民，特别喜欢买双色球。最近看了几种算法，写了一个很简单的双色球的算号器，分享给大家，仅供娱乐，呵呵

[java] view plaincopy

package com.tester.luckly;<BR>

<BR>

import java.util.Random;<BR>

import java.util.Scanner;<BR>

import java.util.Set;<BR>

import java.util.TreeSet;<BR>

<BR>

public class DoubleBall {<BR>

private Set<Integer> hs;<BR>

private static boolean flag = true;<BR>

<BR>

public enum oddOrEven {<BR>

Odd, Even<BR>

}<BR>

<BR>

public enum aOrBOrC {<BR>

A, B, C<BR>

}<BR>

<BR>

public static void main(String[] args) {<BR>

DoubleBall db = new DoubleBall();<BR>

System.out<BR>

.println("====================Red Balls============================");<BR>

String url1 = "http://map.zhcw.com/ssq//html/h7fenqu\_ascstr=20.html";<BR>

System.out.println();<BR>

System.err.println(url1);<BR>

Scanner scanner1 = new Scanner(System.in);<BR>

System.out.println("Please select:1.Input your own two numbers");<BR>

System.out<BR>

.println(" 2.Input the range of the two numbers");<BR>

int select = scanner1.nextInt();<BR>

if (select == 1) {<BR>

flag = false;<BR>

} else {<BR>

flag = true;<BR>

}<BR>

System.out.println("Please input two numbers like this: 1 2");<BR>

scanner1.nextLine();<BR>

String[] str = scanner1.nextLine().split(" ");<BR>

Set<Integer> redBall = db.calcRedBall(Integer.parseInt(str[0]), Integer<BR>

.parseInt(str[1]));<BR>

<BR>

System.out<BR>

.println("====================Blue Ball============================");<BR>

String url2 = "http://sports.sohu.com/s2007/0445/s252476794/";<BR>

System.out.println();<BR>

System.err.println(url2);<BR>

Scanner scanner2 = new Scanner(System.in);<BR>

<BR>

System.out.println("1.Please select:" + oddOrEven.Odd +" or "<BR>

+ oddOrEven.Even);<BR>

String select1 = scanner2.nextLine();<BR>

<BR>

System.out.println("2.Please select:" + aOrBOrC.A +" or " + aOrBOrC.B<BR>

+ " or " + aOrBOrC.C);<BR>

String select2 = scanner2.nextLine();<BR>

<BR>

int blueBall = db.calcBlueBall(select1, select2);<BR>

<BR>

System.out<BR>

.println("====================The Result============================");<BR>

System.out.println();<BR>

for (int i : redBall) {<BR>

System.err.print(i + " ");<BR>

}<BR>

System.err.println("| " + blueBall);<BR>

String url3 = "http://www.bwlc.net/dhtz/";<BR>

System.err.println(url3);<BR>

}<BR>

<BR>

public Set<Integer> calcRedBall(int begin,int end) {<BR>

hs = new TreeSet<Integer>();<BR>

<BR>

/\*<BR>

\* 固定恒码：就是每期都必备的号码，通常为2-3个，如02，13，27。 这2-3个号码长期备选（至少50期）。<BR>

\*/<BR>

hs.add(10);<BR>

hs.add(27);<BR>

<BR>

/\*<BR>

\* 边缘捡“胆”:就是在边缘码“05、10、15、20、25、30”及“01、<BR>

\* 06、11、16、21、26、31”共13个号码中巧妙地捡出胆码。之所以 把边缘码作为“胆”码的一种选取方法，是因为从历史中奖号码来看，<BR>

\* 几乎每一期都会在具备这种特性的号码中出现2-3个。<BR>

\*/<BR>

int[] array = { 5, 15, 20, 25, 30, 01, 06, 11, 16, 21, 26, 31 };<BR>

for (int i = 0; i < 2; i++) {<BR>

Random random = new Random();<BR>

hs.add(array[random.nextInt(array.length)]);<BR>

}<BR>

<BR>

/\*<BR>

\* 重码追邻：在上上期出号的左右选取的上期号码，这话不好理解，现举例说明：双色球2007109期开01 04 07 08 13 14；<BR>

\* 110期开02 04 07 15 24 28；其中这期的04<BR>

\* 07，刚好是109期二、三位的重码，那么111期的重码该选取哪个呢？重码追邻就是在110期的04 07 的左右选取，110期04 07<BR>

\* 的左右是02 15。所以111期的备选号就是02 15 了。实际上111期重码刚好开的是02。重码追邻一般选取2-3个号码。<BR>

\*/<BR>

/\* ========= 或者========= \*/<BR>

/\* 旺区落“胆”。就是在最近几期热号区选择胆码。比如，如果最近的5期内，在中区12~22出号比较密集，那么就要在这个区域里选取3个胆码。这样选择的理由是热码恒热原理，即号码总是在某个区域相对集中出现。 \*/<BR>

if (flag == true) {<BR>

hs.add(begin);<BR>

hs.add(end);<BR>

} else {<BR>

Random random = new Random();<BR>

hs.add(begin + random.nextInt(end - begin + 1));<BR>

}<BR>

/\* Luck number! \*/<BR>

while (hs.size() < 6) {<BR>

Random random = new Random();<BR>

hs.add(1 + random.nextInt(33));<BR>

}<BR>

<BR>

return hs;<BR>

}<BR>

<BR>

public int calcBlueBall(String select1, String select2) {<BR>

int begin = 1;<BR>

int end = 16;<BR>

<BR>

if (select2.equals(aOrBOrC.A.toString())) {<BR>

begin = 1;<BR>

end = 5;<BR>

} else if (select2.equals(aOrBOrC.B.toString())) {<BR>

begin = 6;<BR>

end = 10;<BR>

} else {<BR>

begin = 11;<BR>

end = 16;<BR>

}<BR>

<BR>

Random random = new Random();<BR>

int blueBall = begin + random.nextInt(end - begin +1);<BR>

if (select1.equals(oddOrEven.Odd.toString())) {<BR>

while (blueBall % 2 == 0) {<BR>

blueBall = begin + random.nextInt(end - begin + 1);<BR>

}<BR>

} else {<BR>

while (blueBall % 2 == 1) {<BR>

blueBall = begin + random.nextInt(end - begin + 1);<BR>

}<BR>

}<BR>

<BR>

return blueBall;<BR>

}<BR>

}

[java] view plaincopy

package com.tester.luckly;

import java.util.Random;

import java.util.Scanner;

import java.util.Set;

import java.util.TreeSet;

public class DoubleBall {

private Set<Integer> hs;

private static boolean flag = true;

public enum oddOrEven {

Odd, Even

}

public enum aOrBOrC {

A, B, C

}

public static void main(String[] args) {

DoubleBall db = new DoubleBall();

System.out

.println("====================Red Balls============================");

String url1 = "http://map.zhcw.com/ssq//html/h7fenqu\_ascstr=20.html";

System.out.println();

System.err.println(url1);

Scanner scanner1 = new Scanner(System.in);

System.out.println("Please select:1.Input your own two numbers");

System.out

.println(" 2.Input the range of the two numbers");

int select = scanner1.nextInt();

if (select == 1) {

flag = false;

} else {

flag = true;

}

System.out.println("Please input two numbers like this: 1 2");

scanner1.nextLine();

String[] str = scanner1.nextLine().split(" ");

Set<Integer> redBall = db.calcRedBall(Integer.parseInt(str[0]), Integer

.parseInt(str[1]));

System.out

.println("====================Blue Ball============================");

String url2 = "http://sports.sohu.com/s2007/0445/s252476794/";

System.out.println();

System.err.println(url2);

Scanner scanner2 = new Scanner(System.in);

System.out.println("1.Please select:" + oddOrEven.Odd + " or "

+ oddOrEven.Even);

String select1 = scanner2.nextLine();

System.out.println("2.Please select:" + aOrBOrC.A + " or " + aOrBOrC.B

+ " or " + aOrBOrC.C);

String select2 = scanner2.nextLine();

int blueBall = db.calcBlueBall(select1, select2);

System.out

.println("====================The Result============================");

System.out.println();

for (int i : redBall) {

System.err.print(i + " ");

}

System.err.println("| " + blueBall);

String url3 = "http://www.bwlc.net/dhtz/";

System.err.println(url3);

}

public Set<Integer> calcRedBall(int begin, int end) {

hs = new TreeSet<Integer>();

/\*

\* 固定恒码：就是每期都必备的号码，通常为2-3个，如02，13，27。 这2-3个号码长期备选（至少50期）。

\*/

hs.add(10);

hs.add(27);

/\*

\* 边缘捡“胆”:就是在边缘码“05、10、15、20、25、30”及“01、

\* 06、11、16、21、26、31”共13个号码中巧妙地捡出胆码。之所以 把边缘码作为“胆”码的一种选取方法，是因为从历史中奖号码来看，

\* 几乎每一期都会在具备这种特性的号码中出现2-3个。

\*/

int[] array = { 5, 15, 20, 25, 30, 01, 06, 11, 16, 21, 26, 31 };

for (int i = 0; i < 2; i++) {

Random random = new Random();

hs.add(array[random.nextInt(array.length)]);

}

/\*

\* 重码追邻：在上上期出号的左右选取的上期号码，这话不好理解，现举例说明：双色球2007109期开01 04 07 08 13 14；

\* 110期开02 04 07 15 24 28；其中这期的04

\* 07，刚好是109期二、三位的重码，那么111期的重码该选取哪个呢？重码追邻就是在110期的04 07 的左右选取，110期04 07

\* 的左右是02 15。所以111期的备选号就是02 15 了。实际上111期重码刚好开的是02。重码追邻一般选取2-3个号码。

\*/

/\* ========= 或者========= \*/

/\* 旺区落“胆”。就是在最近几期热号区选择胆码。比如，如果最近的5期内，在中区12~22出号比较密集，那么就要在这个区域里选取3个胆码。这样选择的理由是热码恒热原理，即号码总是在某个区域相对集中出现。 \*/

if (flag == true) {

hs.add(begin);

hs.add(end);

} else {

Random random = new Random();

hs.add(begin + random.nextInt(end - begin + 1));

}

/\* Luck number! \*/

while (hs.size() < 6) {

Random random = new Random();

hs.add(1 + random.nextInt(33));

}

return hs;

}

public int calcBlueBall(String select1, String select2) {

int begin = 1;

int end = 16;

if (select2.equals(aOrBOrC.A.toString())) {

begin = 1;

end = 5;

} else if (select2.equals(aOrBOrC.B.toString())) {

begin = 6;

end = 10;

} else {

begin = 11;

end = 16;

}

Random random = new Random();

int blueBall = begin + random.nextInt(end - begin + 1);

if (select1.equals(oddOrEven.Odd.toString())) {

while (blueBall % 2 == 0) {

blueBall = begin + random.nextInt(end - begin + 1);

}

} else {

while (blueBall % 2 == 1) {

blueBall = begin + random.nextInt(end - begin + 1);

}

}

return blueBall;

}

}

12:29

浏览 (2864)

论坛浏览 (4047)

评论 (7)

分类: Java

收藏Spinner

2009 - 02 - 05

缩略显示

关于用Java I/O 复制文件的问题

今天遇到了一个极其郁闷的问题，想写一段代码，可以给windows自动安装一种字体。

原理就是将4个ttf字体文件复制到C://WINDOWS//Fonts//目录下。本来以为很简单，但用java I/O复制过去的字体不能使用（将记事本的字体改成DejaVuSansMono，如果有效果变化，就是正常的），直接手动复制同样的文件过去，就可以使用。不知道问题出在哪里？

哪位朋友帮忙看看，万分感谢，字体文件在附件中，代码如下：

[java] view plaincopy

package com.test;<BR>

import java.io.FileInputStream;<BR>

import java.io.FileOutputStream;<BR>

import java.io.IOException;<BR>

import java.nio.channels.FileChannel;<BR>

<BR>

public class FontInstall {<BR>

<BR>

public staticvoid main(String[] args) {<BR>

try {<BR>

String[] fonts = { "DejaVuSansMono-Oblique.ttf",<BR>

"DejaVuSansMono-BoldOblique.ttf", "DejaVuSansMono.ttf",<BR>

"DejaVuSansMono-Bold.ttf" };<BR>

System.out.println();<BR>

for (int i =0; i < fonts.length; i++) {<BR>

// Create channel on the source<BR>

FileChannel srcChannel = new FileInputStream(System<BR>

.getProperty("user.dir")<BR>

+ "//" + fonts[i]).getChannel();<BR>

<BR>

// Create channel on the destination<BR>

FileChannel dstChannel = new FileOutputStream(<BR>

"C://WINDOWS//Fonts//" + fonts[i]).getChannel();<BR>

<BR>

// Copy file contents from source to destination<BR>

dstChannel.transferFrom(srcChannel, 0, srcChannel.size());<BR>

<BR>

// Close the channels<BR>

srcChannel.close();<BR>

dstChannel.close();<BR>

}<BR>

} catch (IOException e) {<BR>

}<BR>

}<BR>

}<BR>

[java] view plaincopy

package com.test;

import java.io.FileInputStream;

import java.io.FileOutputStream;

import java.io.IOException;

import java.nio.channels.FileChannel;

public class FontInstall {

public static void main(String[] args) {

try {

String[] fonts = { "DejaVuSansMono-Oblique.ttf",

"DejaVuSansMono-BoldOblique.ttf", "DejaVuSansMono.ttf",

"DejaVuSansMono-Bold.ttf" };

System.out.println();

for (int i = 0; i < fonts.length; i++) {

// Create channel on the source

FileChannel srcChannel = new FileInputStream(System

.getProperty("user.dir")

+ "//" + fonts[i]).getChannel();

// Create channel on the destination

FileChannel dstChannel = new FileOutputStream(

"C://WINDOWS//Fonts//" + fonts[i]).getChannel();

// Copy file contents from source to destination

dstChannel.transferFrom(srcChannel, 0, srcChannel.size());

// Close the channels

srcChannel.close();

dstChannel.close();

}

} catch (IOException e) {

}

}

}

用另一种写法试了下，也是不行，复制过去的文件大小是相同的，用比较工具比较也没问题。

[java] view plaincopy

package com.test;<BR>

<BR>

import java.io.FileInputStream;<BR>

import java.io.FileOutputStream;<BR>

import java.io.IOException;<BR>

import java.io.InputStream;<BR>

import java.io.OutputStream;<BR>

<BR>

public class FontInstall2 {<BR>

<BR>

public staticvoid main(String[] args) {<BR>

try {<BR>

String[] fonts = { "DejaVuSansMono-Oblique.ttf",<BR>

"DejaVuSansMono-BoldOblique.ttf", "DejaVuSansMono.ttf",<BR>

"DejaVuSansMono-Bold.ttf" };<BR>

System.out.println();<BR>

for (int i =0; i < fonts.length; i++) {<BR>

InputStream in = new FileInputStream(System<BR>

.getProperty("user.dir")<BR>

+ "//" + fonts[i]);<BR>

OutputStream out = new FileOutputStream("C://WINDOWS//Fonts//"<BR>

+ fonts[i]);<BR>

<BR>

// Transfer bytes from in to out<BR>

byte[] buf = newbyte[1024];<BR>

int len;<BR>

while ((len = in.read(buf)) > 0) {<BR>

out.write(buf, 0, len);<BR>

}<BR>

in.close();<BR>

out.close();<BR>

}<BR>

} catch (IOException e) {<BR>

}<BR>

}<BR>

}<BR>

[java] view plaincopy

package com.test;

import java.io.FileInputStream;

import java.io.FileOutputStream;

import java.io.IOException;

import java.io.InputStream;

import java.io.OutputStream;

public class FontInstall2 {

public static void main(String[] args) {

try {

String[] fonts = { "DejaVuSansMono-Oblique.ttf",

"DejaVuSansMono-BoldOblique.ttf", "DejaVuSansMono.ttf",

"DejaVuSansMono-Bold.ttf" };

System.out.println();

for (int i = 0; i < fonts.length; i++) {

InputStream in = new FileInputStream(System

.getProperty("user.dir")

+ "//" + fonts[i]);

OutputStream out = new FileOutputStream("C://WINDOWS//Fonts//"

+ fonts[i]);

// Transfer bytes from in to out

byte[] buf = new byte[1024];

int len;

while ((len = in.read(buf)) > 0) {

out.write(buf, 0, len);

}

in.close();

out.close();

}

} catch (IOException e) {

}

}

}

新字体.zip (600.1 KB)

下载次数: 10

17:12

浏览 (907)

论坛浏览 (739)

评论 (5)

分类: Java

收藏Spinner

2009 - 02 - 02

缩略显示

Java util之常用数据类型特性盘点

java.util就相当于c++的STL，是Java的一个非常重要的包，有很多常用的数据类型，不同数据类型有不同的用途，而有些数据类似乎很相似，怎样选择应用，就需要对它们进行辨析。

下面列出了这些数据类型的特点，根据这些特点，就可以有针对性的选用

\* 蓝色为接口，绿色为具体实现类

\* 缩进的层次结构，就是implement或extend的层次关系

\* 每个接口或类都具备其所有上层接口、类的特性

Collection

........|--------List

........|..........|----------ArrayList

........|..........|----------Vector

........|..........|.............|-----Stack

........|..........|----------LinkedList

........|--------Set

...................|----------HashSet.

...................|.............|-----LinkedHashSet

...................|----------SortedSet

.................................|-----TreeSet

Iterator

.....|-------ListIterator

Map

.....|------Hashtable

.....|..........|------Properties

.....|------HashMap

.....|..........|------LinkedHashMap

.....|------WeakHashMap

.....|------SortedMap

................|------TreeMap

Collection.

●..实现该接口及其子接口的所有类都可应用clone()方法，并是序列化类.

.....List.

.....●..可随机访问包含的元素

.....●..元素是有序的

.....●..可在任意位置增、删元素

.....●..不管访问多少次，元素位置不变

.....●..允许重复元素

.....●..用Iterator实现单向遍历，也可用ListIterator实现双向遍历

..........ArrayList

..........●..用数组作为根本的数据结构来实现List

..........●..元素顺序存储

..........●..新增元素改变List大小时，内部会新建一个数组，在将添加元素前将所有数据拷贝到新数组中

..........●..随机访问很快，删除非头尾元素慢，新增元素慢而且费资源

..........●..较适用于无频繁增删的情况

..........●..比数组效率低，如果不是需要可变数组，可考虑使用数组

..........●..非线程安全

.

..........Vector.

..........●..另一种ArrayList，具备ArrayList的特性

..........●..所有方法都是线程安全的（双刃剑，和ArrayList的主要区别）

..........●..比ArrayList效率低

...............Stack

...............●..LIFO的数据结构

..........LinkedList.

..........●..链接对象数据结构（类似链表）

..........●..随机访问很慢，增删操作很快，不耗费多余资源

..........●..非线程安全

.....Set.

.....●..不允许重复元素，可以有一个空元素

.....●..不可随机访问包含的元素

.....●..只能用Iterator实现单向遍历

..........HashSet

..........●..用HashMap作为根本数据结构来实现Set

..........●..元素是无序的

..........●..迭代访问元素的顺序和加入的顺序不同

..........●..多次迭代访问，元素的顺序可能不同

..........●..非线程安全

...............LinkedHashSet

...............●..基于HashMap和链表的Set实现

...............●..迭代访问元素的顺序和加入的顺序相同

...............●..多次迭代访问，元素的顺序不便

...............●..因此可说这是一种有序的数据结构

...............●..性能比HashSet差

...............●..非线程安全

..........SortedSet

..........●..加入SortedSet的所有元素必须实现Comparable接口

..........●..元素是有序的

...............TreeSet.

...............●..基于TreeMap实现的SortedSet

...............●..排序后按升序排列元素

...............●..非线程安全

－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－

Iterator..

●..对Set、List进行单向遍历的迭代器

..........ListIterator.

..........●..对List进行双向遍历的迭代器

－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－

Map

●..键值对，键和值一一对应

●..不允许重复的键.

.....Hashtable.

.....●..用作键的对象必须实现了hashcode()、equals()方法，也就是说只有Object及其子类可用作键

.....●..键、值都不能是空对象

.....●..多次访问，映射元素的顺序相同

.....●..线程安全的

..........Properties

..........●..键和值都是字符串

.....HashMap

.....●..键和值都可以是空对象

.....●..不保证映射的顺序

.....●..多次访问，映射元素的顺序可能不同

.....●..非线程安全

...............LinkedHashMap

...............●..多次访问，映射元素的顺序是相同的

...............●..性能比HashMap差

.....WeakHashMap..

.....●..当某个键不再正常使用时，垃圾收集器会移除它，即便有映射关系存在

.....●..非线程安全

.....SortedMap.

.....●..键按升序排列

.....●..所有键都必须实现.Comparable.接口.

...............TreeMap.

...............●..基于红黑树的SortedMap实现

...............●..非线程安全

12:43

浏览 (315)

评论 (1)

分类: Java

收藏Spinner

2009 - 02 - 02

缩略显示

一些常用的正则表达式

正则表达式是一种通用的标准，大部分计算机语言都支持正则表达式，包括as3，这里转摘出了一些常用的正则表达式语句，大家用到的时候就不用自己写了

^/d+$　　//匹配非负整数（正整数 + 0）

^[0-9]\*[1-9][0-9]\*$　　//匹配正整数

^((-/d+)|(0+))$　　//匹配非正整数（负整数 + 0）

^-[0-9]\*[1-9][0-9]\*$　　//匹配负整数

^-?/d+$　　　　//匹配整数

^/d+(/./d+)?$　　//匹配非负浮点数（正浮点数 + 0）

^(([0-9]+/.[0-9]\*[1-9][0-9]\*)|([0-9]\*[1-9][0-9]\*/.[0-9]+)|([0-9]\*[1-9][0-9]\*))$　　//匹配正浮点数

^((-/d+(/./d+)?)|(0+(/.0+)?))$　　//匹配非正浮点数（负浮点数 + 0）

^(-(([0-9]+/.[0-9]\*[1-9][0-9]\*)|([0-9]\*[1-9][0-9]\*/.[0-9]+)|([0-9]\*[1-9][0-9]\*)))$　　//匹配负浮点数

^(-?/d+)(/./d+)?$　　//匹配浮点数

^[A-Za-z]+$　　//匹配由26个英文字母组成的字符串

^[A-Z]+$　　//匹配由26个英文字母的大写组成的字符串

^[a-z]+$　　//匹配由26个英文字母的小写组成的字符串

^[A-Za-z0-9]+$　　//匹配由数字和26个英文字母组成的字符串

^/w+$　　//匹配由数字、26个英文字母或者下划线组成的字符串

^[/w-]+(/.[/w-]+)\*@[/w-]+(/.[/w-]+)+$　　　　//匹配email地址

^[a-zA-z]+://匹配(/w+(-/w+)\*)(/.(/w+(-/w+)\*))\*(/?/S\*)?$　　//匹配url

匹配中文字符的正则表达式： [/u4e00-/u9fa5]

匹配双字节字符(包括汉字在内)：[^/x00-/xff]

匹配空行的正则表达式：/n[/s| ]\*/r

匹配HTML标记的正则表达式：/<(.\*)>.\*<//>|<(.\*) //>/

匹配首尾空格的正则表达式：(^/s\*)|(/s\*$)

匹配Email地址的正则表达式：/w+([-+.]/w+)\*@/w+([-.]/w+)\*/./w+([-.]/w+)\*

匹配网址URL的正则表达式：^[a-zA-z]+://(/w+(-/w+)\*)(/.(/w+(-/w+)\*))\*(/?/S\*)?$

匹配帐号是否合法(字母开头，允许5-16字节，允许字母数字下划线)：^[a-zA-Z][a-zA-Z0-9\_]{4,15}$

匹配国内电话号码：(/d{3}-|/d{4}-)?(/d{8}|/d{7})?

匹配腾讯QQ号：^[1-9]\*[1-9][0-9]\*$

下表是元字符及其在正则表达式上下文中的行为的一个完整列表：

/ 将下一个字符标记为一个特殊字符、或一个原义字符、或一个后向引用、或一个八进制转义符。

^ 匹配输入字符串的开始位置。如果设置了 RegExp 对象的Multiline 属性，^ 也匹配 ’/n’ 或 ’/r’ 之后的位置。

$ 匹配输入字符串的结束位置。如果设置了 RegExp 对象的Multiline 属性，$ 也匹配 ’/n’ 或 ’/r’ 之前的位置。

\* 匹配前面的子表达式零次或多次。

+ 匹配前面的子表达式一次或多次。+ 等价于 {1,}。

? 匹配前面的子表达式零次或一次。? 等价于 {0,1}。

{n} n 是一个非负整数，匹配确定的n 次。

{n,} n 是一个非负整数，至少匹配n 次。

{n,m} m 和 n 均为非负整数，其中n <= m。最少匹配 n 次且最多匹配 m 次。在逗号和两个数之间不能有空格。

? 当该字符紧跟在任何一个其他限制符 (\*, +, ?, {n}, {n,}, {n,m}) 后面时，匹配模式是非贪婪的。非贪婪模式尽可能少的匹配所搜索的字符串，而默认的贪婪模式则尽可能多的匹配所搜索的字符串。

. 匹配除 "/n" 之外的任何单个字符。要匹配包括 ’/n’ 在内的任何字符，请使用象 ’[./n]’ 的模式。

(pattern) 匹配pattern 并获取这一匹配。

(?:pattern) 匹配pattern 但不获取匹配结果，也就是说这是一个非获取匹配，不进行存储供以后使用。

(?=pattern) 正向预查，在任何匹配 pattern 的字符串开始处匹配查找字符串。这是一个非获取匹配，也就是说，该匹配不需要获取供以后使用。

(?!pattern) 负向预查，与(?=pattern)作用相反

x|y 匹配 x 或 y。

[xyz] 字符集合。

[^xyz] 负值字符集合。

[a-z] 字符范围，匹配指定范围内的任意字符。

[^a-z] 负值字符范围，匹配任何不在指定范围内的任意字符。

/b 匹配一个单词边界，也就是指单词和空格间的位置。

/B 匹配非单词边界。

/cx 匹配由x指明的控制字符。

/d 匹配一个数字字符。等价于 [0-9]。

/D 匹配一个非数字字符。等价于 [^0-9]。

/f 匹配一个换页符。等价于 /x0c 和 /cL。

/n 匹配一个换行符。等价于 /x0a 和 /cJ。

/r 匹配一个回车符。等价于 /x0d 和 /cM。

/s 匹配任何空白字符，包括空格、制表符、换页符等等。等价于[ /f/n/r/t/v]。

/S 匹配任何非空白字符。等价于 [^ /f/n/r/t/v]。

/t 匹配一个制表符。等价于 /x09 和 /cI。

/v 匹配一个垂直制表符。等价于 /x0b 和 /cK。

/w 匹配包括下划线的任何单词字符。等价于’[A-Za-z0-9\_]’。

/W 匹配任何非单词字符。等价于 ’[^A-Za-z0-9\_]’。

/xn 匹配 n，其中 n 为十六进制转义值。十六进制转义值必须为确定的两个数字长。

/num 匹配 num，其中num是一个正整数。对所获取的匹配的引用。

/n 标识一个八进制转义值或一个后向引用。如果 /n 之前至少 n 个获取的子表达式，则 n 为后向引用。否则，如果 n 为八进制数字 (0-7)，则 n 为一个八进制转义值。

/nm 标识一个八进制转义值或一个后向引用。如果 /nm 之前至少有is preceded by at least nm 个获取得子表达式，则 nm 为后向引用。如果 /nm 之前至少有 n 个获取，则 n 为一个后跟文字 m 的后向引用。如果前面的条件都不满足，若 n 和 m 均为八进制数字 (0-7)，则 /nm 将匹配八进制转义值 nm。

/nml 如果 n 为八进制数字 (0-3)，且 m 和 l 均为八进制数字 (0-7)，则匹配八

12:34

浏览 (268)

评论 (0)

分类: Java

收藏Spinner

2009 - 01 - 21

缩略显示

编程经典问题

【程序1】

题目：古典问题：有一对兔子，从出生后第3个月起每个月都生一对兔子，小兔子长到第三个月后每个月又生一对兔子，假如兔子都不死，问每个月的兔子总数为多少？

1.程序分析： 兔子的规律为数列1,1,2,3,5,8,13,21....

【程序2】

题目：判断101-200之间有多少个素数，并输出所有素数。

1.程序分析：判断素数的方法：用一个数分别去除2到sqrt(这个数)，如果能被整除，

则表明此数不是素数，反之是素数。

【程序3】

题目：打印出所有的"水仙花数"，所谓"水仙花数"是指一个三位数，其各位数字立方和等于该数本身。例如：153是一个"水仙花数"，因为153=1的三次方＋5的三次方＋3的三次方。

1.程序分析：利用for循环控制100-999个数，每个数分解出个位，十位，百位。

【程序4】

题目：将一个正整数分解质因数。例如：输入90,打印出90=2\*3\*3\*5。

程序分析：对n进行分解质因数，应先找到一个最小的质数k，然后按下述步骤完成：

(1)如果这个质数恰等于n，则说明分解质因数的过程已经结束，打印出即可。

(2)如果n<>k，但n能被k整除，则应打印出k的值，并用n除以k的商,作为新的正整数你n,重复执行第一步。

(3)如果n不能被k整除，则用k+1作为k的值,重复执行第一步。

【程序5】

题目：利用条件运算符的嵌套来完成此题：学习成绩>=90分的同学用A表示，60-89分之间的用B表示，60分以下的用C表示。

1.程序分析：(a>b)?a:b这是条件运算符的基本例子。

【程序6】

题目：输入两个正整数m和n，求其最大公约数和最小公倍数。

1.程序分析：利用辗除法。

【程序7】

题目：输入一行字符，分别统计出其中英文字母、空格、数字和其它字符的个数。

1.程序分析：利用while语句,条件为输入的字符不为'/n'.

【程序8】

题目：求s=a+aa+aaa+aaaa+aa...a的值，其中a是一个数字。例如2+22+222+2222+22222(此时共有5个数相加)，几个数相加有键盘控制。

1.程序分析：关键是计算出每一项的值。

【程序9】

题目：一个数如果恰好等于它的因子之和，这个数就称为"完数"。例如6=1＋2＋3.编程 找出1000以内的所有完数。

【程序10】

题目：一球从100米高度自由落下，每次落地后反跳回原高度的一半；再落下，求它在 第10次落地时，共经过多少米？第10次反弹多高？

【程序11】

题目：有1、2、3、4个数字，能组成多少个互不相同且无重复数字的三位数？都是多少？

1.程序分析：可填在百位、十位、个位的数字都是1、2、3、4。组成所有的排列后再去 掉不满足条件的排列。

【程序12】

题目：企业发放的奖金根据利润提成。利润(I)低于或等于10万元时，奖金可提10%；利润高于10万元，低于20万元时，低于10万元的部分按 10%提成，高于10万元的部分，可可提成7.5%；20万到40万之间时，高于20万元的部分，可提成5%；40万到60万之间时高于40万元的部分，可提成3%；60万到100万之间时，高于60万元的部分，可提成1.5%，高于100万元时，超过100万元的部分按1%提成，从键盘输入当月利润I，求应发放奖金总数？

1.程序分析：请利用数轴来分界，定位。注意定义时需把奖金定义成长整型。

【程序13】

题目：一个整数，它加上100后是一个完全平方数，再加上168又是一个完全平方数，请问该数是多少？

1.程序分析：在10万以内判断，先将该数加上100后再开方，再将该数加上268后再开方，如果开方后的结果满足如下条件，即是结果。请看具体分析：

【程序14】

题目：输入某年某月某日，判断这一天是这一年的第几天？

1.程序分析：以3月5日为例，应该先把前两个月的加起来，然后再加上5天即本年的第几天，特殊情况，闰年且输入月份大于3时需考虑多加一天。

【程序15】

题目：输入三个整数x,y,z，请把这三个数由小到大输出。

1.程序分析：我们想办法把最小的数放到x上，先将x与y进行比较，如果x>y则将x与y的值进行交换，然后再用x与z进行比较，如果x>z则将x与z的值进行交换，这样能使x最小。

【程序16】

题目：输出9\*9口诀。

1.程序分析：分行与列考虑，共9行9列，i控制行，j控制列。

【程序17】

题目：猴子吃桃问题：猴子第一天摘下若干个桃子，当即吃了一半，还不瘾，又多吃了一个 第二天早上又将剩下的桃子吃掉一半，又多吃了一个。以后每天早上都吃了前一天剩下 的一半零一个。到第10天早上想再吃时，见只剩下一个桃子了。求第一天共摘了多少。

1.程序分析：采取逆向思维的方法，从后往前推断。

【程序18】

题目：两个乒乓球队进行比赛，各出三人。甲队为a,b,c三人，乙队为x,y,z三人。已抽签决定比赛名单。有人向队员打听比赛的名单。a说他不和x比，c说他不和x,z比，请编程序找出三队赛手的名单。

1.程序分析：判断素数的方法：用一个数分别去除2到sqrt(这个数)，如果能被整除， 则表明此数不是素数，反之是素数。

【程序19】

题目：打印出如下图案（菱形）

\*

\*\*\*

\*\*\*\*\*\*

\*\*\*\*\*\*\*\*

\*\*\*\*\*\*

\*\*\*

\*

1.程序分析：先把图形分成两部分来看待，前四行一个规律，后三行一个规律，利用双重 for循环，第一层控制行，第二层控制列。

【程序20】

题目：有一分数序列：2/1，3/2，5/3，8/5，13/8，21/13...求出这个数列的前20项之和。

1.程序分析：请抓住分子与分母的变化规律。

【程序21】

题目：求1+2!+3!+...+20!的和

1.程序分析：此程序只是把累加变成了累乘。

【程序22】

题目：利用递归方法求5!。

1.程序分析：递归公式：fn=fn\_1\*4!

【程序23】

题目：有5个人坐在一起，问第五个人多少岁？他说比第4个人大2岁。问第4个人岁数，他说比第3个人大2岁。问第三个人，又说比第2人大两岁。问第2个人，说比第一个人大两岁。最后问第一个人，他说是10岁。请问第五个人多大？

1.程序分析：利用递归的方法，递归分为回推和递推两个阶段。要想知道第五个人岁数，需知道第四人的岁数，依次类推，推到第一人（10岁），再往回推。

【程序24】

题目：给一个不多于5位的正整数，要求：一、求它是几位数，二、逆序打印出各位数字。

【程序25】

题目：一个5位数，判断它是不是回文数。即12321是回文数，个位与万位相同，十位与千位相同。

【程序26】

题目：请输入星期几的第一个字母来判断一下是星期几，如果第一个字母一样，则继续 判断第二个字母。

1.程序分析：用情况语句比较好，如果第一个字母一样，则判断用情况语句或if语句判断第二个字母。

【程序27】

题目：求100之内的素数

【程序28】

题目：对10个数进行排序

1.程序分析：可以利用选择法，即从后9个比较过程中，选择一个最小的与第一个元素交换， 下次类推，即用第二个元素与后8个进行比较，并进行交换。

【程序29】

题目：求一个3\*3矩阵对角线元素之和

1.程序分析：利用双重for循环控制输入二维数组，再将a[i][i]累加后输出。

【程序30】

题目：有一个已经排好序的数组。现输入一个数，要求按原来的规律将它插入数组中。

1. 程序分析：首先判断此数是否大于最后一个数，然后再考虑插入中间的数的情况，插入后此元素之后的数，依次后移一个位置。

【程序31】

题目：将一个数组逆序输出。

1.程序分析：用第一个与最后一个交换。

【程序32】

题目：取一个整数a从右端开始的4～7位。

程序分析：可以这样考虑：

(1)先使a右移4位。

(2)设置一个低4位全为1,其余全为0的数。可用~(~0<<4)

(3)将上面二者进行&运算。

【程序33】

题目：打印出杨辉三角形（要求打印出10行如下图）

1.程序分析：

1

1 1

1 2 1

1 3 3 1

1 4 6 4 1

1 5 10 10 5 1

【程序34】

题目：输入3个数a,b,c，按大小顺序输出。

1.程序分析：利用指针方法。

【程序35】

题目：输入数组，最大的与第一个元素交换，最小的与最后一个元素交换，输出数组。

【程序36】

题目：有n个整数，使其前面各数顺序向后移m个位置，最后m个数变成最前面的m个数

【程序37】

题目：有n个人围成一圈，顺序排号。从第一个人开始报数（从1到3报数），凡报到3的人退出圈子，问最后留下的是原来第几号的那位。

【程序38】

题目：写一个函数，求一个字符串的长度，在main函数中输入字符串，并输出其长度。

【程序39】

题目：编写一个函数，输入n为偶数时，调用函数求1/2+1/4+...+1/n,当输入n为奇数时，调用函数1/1+1/3+...+1/n(利用指针函数)

【程序40】

题目：字符串排序。

【程序41】

题目：海滩上有一堆桃子，五只猴子来分。第一只猴子把这堆桃子凭据分为五份，多了一个，这只猴子把多的一个扔入海中，拿走了一份。第二只猴子把剩下的桃子又平均分成五份，又多了一个，它同样把多的一个扔入海中，拿走了一份，第三、第四、第五只猴子都是这样做的，问海滩上原来最少有多少个桃子？

【程序42】

题目：809\*??=800\*??+9\*??+1 其中??代表的两位数,8\*??的结果为两位数，9\*??的结果为3位数。求??代表的两位数，及809\*??后的结果。

【程序43】

题目：求0—7所能组成的奇数个数。

【程序44】

题目：一个偶数总能表示为两个素数之和。

【程序45】

题目：判断一个素数能被几个9整除

【程序46】

题目：两个字符串连接程序

【程序47】

题目：读取7个数（1—50）的整数值，每读取一个值，程序打印出该值个数的＊。

【程序48】

题目：某个公司采用公用电话传递数据，数据是四位的整数，在传递过程中是加密的，加密规则如下：每位数字都加上5,然后用和除以10的余数代替该数字，再将第一位和第四位交换，第二位和第三位交换。

【程序49】

题目：计算字符串中子串出现的次数

【程序50】

题目：有五个学生，每个学生有3门课的成绩，从键盘输入以上数据（包括学生号，姓名，三门课成绩），计算出平均成绩，况原有的数据和计算出的平均分数存放在磁盘文件"stud"中。

16:43

浏览 (1671)

评论 (2)

分类: Java

收藏Spinner

2009 - 01 - 20

缩略显示

Java中格式化输出数字

在实际工作中，常常需要设定数字的输出格式，如以百分比的形式输出，或者设定小数位数等，现稍微总结如下。

主要使用的类：java.text.DecimalFormat

1。实例化对象，可以用如下两种方法：

DecimalFormat df=(DecimalFormat)NumberFormat.getInstance();

DecimalFormat df1=(DecimalFormat) DecimalFormat.getInstance();

因为DecimalFormat继承自NumberFormat。

2。设定小数位数

系统默认小数位数为3，如：

DecimalFormat df=(DecimalFormat)NumberFormat.getInstance();

System.out.println(df.format(12.3456789));

输出：12.346

现在可以通过如下方法把小数为设为两位：

df.setMaximumFractionDigits(2);

System.out.println(df.format(12.3456789));

则输出为：12.35

3。将数字转化为百分比输出，有如下两种方法：

(1)

df.applyPattern("##.##%");

System.out.println(df.format(12.3456789));

System.out.println(df.format(1));

System.out.println(df.format(0.015));

输出分别为：1234.57% 100% 1.5%

(2)

df.setMaximumFractionDigits(2);

System.out.println(df.format(12.3456789\*100)+"%");

System.out.println(df.format(1\*100)+"%");

System.out.println(df.format(0.015\*100)+"%");

输出分别为：

1,234.57% 100% 1.5%

4。设置分组大小

DecimalFormat df1=(DecimalFormat) DecimalFormat.getInstance();

df1.setGroupingSize(2);

System.out.println(df1.format(123456789));

输出：1,23,45,67,89

还可以通过df1.setGroupingUsed(false);来禁用分组设置，如：

DecimalFormat df1=(DecimalFormat) DecimalFormat.getInstance();

df1.setGroupingSize(2);

df1.setGroupingUsed(false);

System.out.println(df1.format(123456789));

输出：123456789

5。设置小数为必须为2位

DecimalFormat df2=(DecimalFormat) DecimalFormat.getInstance();

df2.applyPattern("0.00");

System.out.println(df2.format(1.2));

输出：1.20

java.text.DecimalFormat学习笔记

例子：

import java.text.\*;

public class Untitled1 {

public static void main(String[] args) {

//---------------------------------------------

//定义一个数字格式化对象，格式化模板为".##"，即保留2位小数.

DecimalFormat a = new DecimalFormat(".##");

String s= a.format(333.335);

System.err.println(s);

//说明：如果小数点后面不够2位小数，不会补零.参见Rounding小节

//---------------------------------------------

//-----------------------------------------------

//可以在运行时刻用函数applyPattern(String)修改格式模板

//保留2位小数，如果小数点后面不够2位小数会补零

a.applyPattern(".00");

s = a.format(333.3);

System.err.println(s);

//------------------------------------------------

//------------------------------------------------

//添加千分号

a.applyPattern(".##/u2030");

s = a.format(0.78934);

System.err.println(s);//添加千位符后,小数会进三位并加上千位符

//------------------------------------------------

//------------------------------------------------

//添加百分号

a.applyPattern("#.##%");

s = a.format(0.78645);

System.err.println(s);

//------------------------------------------------

//------------------------------------------------

//添加前、后修饰字符串，记得要用单引号括起来

a.applyPattern("'这是我的钱$',###.###'美圆'");

s = a.format(33333443.3333);

System.err.println(s);

//------------------------------------------------

//------------------------------------------------

//添加货币表示符号(不同的国家，添加的符号不一样

a.applyPattern("/u00A4");

s = a.format(34);

System.err.println(s);

//------------------------------------------------

//-----------------------------------------------

//定义正负数模板,记得要用分号隔开

a.applyPattern("0.0;'@'-#.0");

s = a.format(33);

System.err.println(s);

s = a.format(-33);

System.err.println(s);

//-----------------------------------------------

//综合运用，正负数的不同前后缀

String pattern="'my moneny'###,###.##'RMB';'ur money'###,###.##'US'";

a.applyPattern(pattern);

System.out.println(a.format(1223233.456));

}

}

总结：

要生成一个DecimalFormat对象，一般只要通过NumberFormat类工厂的getInstance()来取得一个 NumberFormat对象再将其转换成DecimalFormat对象，然后通过DecimalForat对象的applyPattern()来动态改变数据的现示格式模板，通过format()方法取得格式化后的数字。同时,DecimalFormat提供了许多的方法来返回格式化后的数字的某一部份，这些方法如：getNegativeSuffix()。这个类的难点主要是在模板的书写及理解上。其实主要的就是针对一个数字的正负形式来设定不同的格式显示。这里要特别注意的是使用在模板上的特殊字符代表有特殊的意义，如下表所示：

Symbol Description

0 a digit

# a digit, zero shows as absent

. placeholder for decimal separator

, placeholder for grouping separator

E separates mantissa and exponent for exponential formats

; separates formats

- default negative prefix

% multiply by 100 and show as percentage

? multiply by 1000 and show as per mille

¤ currency sign; replaced by currency symbol; if doubled, replaced by international currency symbol; if present in a pattern, the monetary decimal separator is used instead of the decimal separator

X any other characters can be used in the prefix or suffix

' used to quote special characters in a prefix or suffix

例如：如果模板中含有#，意思是指这个#号可代表一个或多个数字如果该位的数字是零的话则省略该位。另：注意“#,##0.0#;(#)”这个模板的意思是指数字的负数形式跟正数的一样。

12:15

浏览 (1451)

评论 (0)

分类: Java

收藏Spinner

2009 - 01 - 08

缩略显示

右键运行jar文件的方法

经常需要运行一些打成jar文件格式的Java程序，每次都需要切换到DOS下运行命令：java -jar FileName.jar ，非常的麻烦。

其实可以将jar文件的默认打开方式设置成jre中的javaw，设置好后可以双击jar文件运行。但这样的话，需要查看包中内容的时候又非常麻烦，需要将jar文件拖到解压缩工具中。

今天突发奇想，改写了一个批处理文件，可以将java -jar命令加到鼠标右键中：

reg add "HKCR/\*/shell/Run\_Jar" /ve /d Run\_Jar /f

reg add "HKCR/\*/shell/Run\_Jar/command" /ve /d "java -jar %%1" /f

reg add "HKCR/Folder/shell/Run\_Jar" /ve /d Run\_Jar /f

reg add "HKCR/Folder/shell/Run\_Jar/command" /ve /d "java -jar %%1" /f

将上面的代码保存为bat文件，然后双击执行，执行后，鼠标右键中会出现Run\_Jar选项。

这下你就可以在你想运行的jar文件上点击右键，然后Run\_Jar，怎么样，比较方便吧：）

稍微解释一下上面的命令：

reg add "HKCR/\*/shell/Run\_Jar"

在注册表中的HKEY\_CLASSES\_ROOT/\*shell/ 目录下添加键值，后面参数的意思分别是：

/ve 为注册表项添加空白值名<无名称>

/d 要分配给添加的注册表 ValueName 的数据

/f 不用提示就强行改写现有注册表项

12:11

浏览 (3231)

评论 (2)

分类: Java

收藏Spinner

2009 - 01 - 07

缩略显示

开心网外挂开发手记

开讲。

做一个Web游戏外挂需要的准备知识：

1) 需要有耐心

2) 熟悉HTML, JavaScript，特别是FORM

3) 熟悉HTTP协议，特别是Cookie, URL的编码方式和POST, GET内容格式

4) 熟悉游戏本身，能抽象出最优的赚钱/升级的数学模型

5) 掌握一门语言，白菜萝卜都可以，我比较喜欢用Python和C#

6) 需要一些抓包的工具，比如Fiddler

好了，来个例子，开心网争车位.

首先，我们看看一个正常用户玩的时候是怎么玩的。

1) 浏览器输入 www.kaixin001.com ，输入用户名，密码，点登陆

2) 登陆后选择左侧的“争车位”，进入到争车位

3) 看看谁在我的车位里，一一贴条

4) 看看自己的车哪些停车时间操作15分钟了，一一找个空的不免费的车位挪过去

5) 登出开心网，关闭浏览器

看看让程序怎么来实现1-6。

1) 登陆开心网

在Fiddler的帮助下，我们知道登陆是把FORM提交到/login/login.php，POST内容是url=%2F&email=xxx@xxx.com&password=xxx

用C#可以方便的完成这个POST操作，然后得到服务器的返回，然后根据返回的内容里找一个关键字就可以判断这次登陆操作是否成功了。（比如我找的就是"<title>我的首页 - 开心网</title>"）

2) 登陆争车位

这次是一个GET操作，URL是/app/app.php?aid=1040。

需要说明一下的是，这里没有提供用户名密码，服务器怎么能知道是哪个用户呢？Cookie在这里就发挥了它的作用。

C#里存放Cookie的方法是new 一个CookieContainer，然后所有的HttpWebRequest的CookieContainer都用它。

好了，取得这个页面以后，我们能得到很多信息：（这些信息是JSON格式）

a) 好友列表，每个好友的状态（在线/车位满）

b) 自己的车库信息，停了那些好友的哪些车，分别赚了多少钱

c) 自己的汽车信息，多少辆车，分别停在哪里，赚了多少钱

根据这些信息，我们可以得到一个停放的列表（车位不满的好友列表）

3) 贴条

贴条是一个POST操作，URL是/parking/post.php，内容是verify=xxx&parkid=yyy&p=1&\_=

parkid很好理解，车库信息里直接可以取到，那么verify怎么得到的呢？

我当初的第一反应是Cookie数据通过某种运算得到的，不过后来我看了登陆争车位的html后才发现原来这个verify是这样写的：

...

<script language=javascript>

var fs2\_pars = "f1";

var g\_verify = "383639\_1040\_383639\_1221703779\_49963b942199e94b88e405d0f7b1651d";

var g\_first = parseInt("0");

var g\_touid = parseInt("0");

var g\_checkswf = 0;

.....

很好，直接去这个从HTML取得的值就好了。

4) 挪车位

这个操作稍微复杂了一点点，具体的HTTP请求大家可以去分析一下，我分析的结果如下：

a) 是否是免费车位

在JavaScript里有代码：

v\_park\_free = (v\_parkid >> 16) & 0xff;

if (v\_park\_free)

{

v\_mycar += '免费车位';

}

else

{

v\_mycar += '私家车位';

}

c) 读取邻居家车库信息

URL: /parking/neighbor.php

POST内容: verify=xxx&puid=yyy&\_=

d) 读取邻居家车库信息

URL: /parking/user.php

POST内容: verify=xxx&puid=yyy&\_=

e) 停车

URL: /parking/park.php

POST内容: verify=xxx&park\_uid=xxx&parkid=xxx&carid=xxx&neighbor=xxx&a=1&first\_fee\_parking=0&\_=

5) 登出

GET URL: /login/logout.php

恭喜你，基本的操作你都明白了。

时间晚了，该睡觉了，一些高级主题改天再写~

--------------Google Code 分割线-------------

http://code.google.com/p/kaixin/

13:15

浏览 (1994)

评论 (0)

分类: Java

收藏Spinner

2009 - 01 - 07

缩略显示

给学习J2EE的朋友一些值得研究的开源项目

给学习J2EE的朋友一些值得研究的开源项目

江苏 无锡 缪小东

这篇文章写在我研究J2SE、J2EE近三年后。前3年我研究了J2SE的Swing、Applet、Net、RMI、Collections、IO、 JNI......研究了J2EE的JDBC、Sevlet、JSP、JNDI…..不久我发现这些好像太浮浅了：首先，我发现自己知道的仅仅是java 提供的大量的API，根本不能很好地使用它；其次，我根本就没有学到任何有助于写程序的知识，此时我也只不过能写个几页的小程序。出于这个幼稚的想法我研究了JDK中Collections、Logger、IO…..的源代码，发现这个世界真的很神奇，竟然有如此的高手――利用java语言最最基本的语法，创造了这些优秀的Framework。从此一发不可收拾，我继续研究了J2EE的部分，又发现这是一个我根本不能理解的方向（曾经有半年停滞不前），为什么只有接口没有实现啊！后来由于一直使用Tomcat、Derby等软件突然发现：哦！原来J2EE仅仅是一个标准，只是一个架构。真正的实现是不同提供商提供的。接着我研究了MOM4J、OpenJMS、Mocki、HSQLD……发现这些就是J2EE的实现啊！原来软件竟会如此复杂，竟会如此做….规范和实现又是如何成为一体的呢？通过上面的研究发现：原来J2EE后面竟然有太多太多理念、太多太多的相似！这些相似就是其背后的理念――设计模式！（很幸运，在我学java的时候，我一般学java的一个方向就会读一些关于设计模式的书！很幸运，到能领略一点的时候能真正知道这是为什么！）其实模式就是一种思维方式、就是一种理念……模式是要运用到程序中的，只有从真正的项目中才能领会模式的含义……

学得越多，发现懂得越少！在学习过程中发现一些很有用，很值得学习的开源项目，今天在此推荐给大家。

一、 JavaServlet和JSP方向

很多人都是从Servlet和JSP步入J2EE的。它就是J2EE的表现层，用于向客户呈现服务器上的内容。J2EE很重要的方面。不罗嗦了！大家都知道的！下面就开始推荐吧！

1. Jakarta Tomcat

Apache基金会提供的免费的开源的Serlvet容器，它是的Jakarta项目中的一个核心项目，由Apache、Sun和其它一些公司（都是IT 界的大鳄哦）及个人共同开发而成，全世界绝大部分Servlet和Jsp的容器都是使用它哦！由于Sun的参与和支持，最新的Servlet和Jsp规范总能在Tomcat中得到体现。

不过它是一个非常非常全的Serlvet容器，全部源码可能有4000页，对于初学者或者一般的老手可能还是比较大了！在你有能力时推荐研究！下载地址：http://jakarta.apache.org/tomcat/index.html

下面推荐两个小一点的吧！

2. Jetty

Jetty是一个开放源码的HTTP服务器和Java serverlet容器。源代码只有1000页左右，很值得研究。有兴趣可以去http://jetty.mortbay.com/下载看看。我曾经翻了一下，只是目前没有时间。（都化在博客上了，等博客基本定型，且内容完整了，再干我热衷的事件吧！）

3. Jigsaw

Jigsaw是W3C开发的HTTP，基于Java 的服务器，提供了未来 Web 技术发展的蓝图。W3C知道吧！（太有名气了，很多标准都是它制订的！有空经常去看看吧！）下载网址：http://www.w3.org/Jigsaw 代码仅仅1000页左右。

4. Jo!

Jo!是一个纯Java的实现了Servlet API 2.2, JSP 1.1, 和HTTP/1.1的Web服务器。它的特性包括支持servlet tag,支持SSI，高级线程管理，虚拟主机，数据缓存，自动压缩text或HTML文件进行传输，国际化支持，自动重新加载Servlet、Jsp，自动重新加载web工程文件(WARs)，支持WAR热部署和一个Swing控制台。jo!可以被用做jboss和jakarta avalon-phoenix的web容器。下载地址http://www.tagtraum.com/ 。我极力推荐大家在研究Tomcat之前研究该软件，主要是其比Tomcat小多了，且开发者提供比较全的手册。该方向研究这两个也就可以了！

二、 JDBC方向

很多人都喜欢JDBC，数据库吗！很深奥的东西，一听就可以糊弄人。其实等你真正研究了数据库的实现后发现，接口其实真的太简单，太完美了！要想设计如此优秀的框架还是需要学习的。下面就推荐几个数据库的实现吧！

1. Hypersonic SQL

Hypersonic SQL开源数据库方向比较流行的纯Java开发的关系型数据库。好像不是JDBC兼容的，JDBC的很多高级的特性都没有支持，不过幸好支持ANSI- 92 标准 SQL语法。我推荐它主要是它的代码比较少1600页左右，如此小的数据库值得研究，而且他占的空间很小，大约只有160K，拥有快速的数据库引擎。推荐你的第一个开源数据库。下载地址：http://hsqldb.sourceforge.net/。

2. Mckoi DataBase

McKoiDB 和Hypersonic SQL差不多，它是GPL 的license的纯Java开发的数据库。他的 JDBC Driver 是使用 JDBC version 3 的 Specifaction。 他也是遵循 SQL-92 的标准，也尽量支持新的 SQL 特色, 并且支持 Transaction 的功能。两个可以选一个吧！下载地址：http://mckoi.com/database/。

3. Apache Derby

学Java的数据库我建议使用Apache Derby ，研究数据库想成为一个数据库的高手我建议你先研究Apache Derby。Apache Derby是一个高质量的、纯 Java开发的嵌入式关系数据库引擎，IBM® 将其捐献给Apache开放源码社区，同时IBM的产品CloudSpace是它对应的产品。Derby是基于文件系统，具有高度的可移植性，并且是轻量级的，这使得它非常便于发布。主要是没有商业用户的很好的界面，没有其太多的功能。不过对于我们使用数据库、研究数据库还是极其有用的。对于中小型的企业说老实话你也不要用什么Oracle、SqlServer了，用Derby就可以了，何况是开源的呢！只要能发挥其长处也不容易啊！下载地址：http://incubator.apache.org/derby。

不过在没有足够的能力前，不要试图读懂它！注释和源代码15000页左右，我一年的阅读量！能读下来并且能真正领会它，绝对高手！你能读完Derby的源代码只有两种可能：1.你成为顶尖的高手――至少是数据库这部分；2.你疯了。选择吧！！！！作为我自己我先选择Hypersonic SQL这样的数据库先研究，能过这一关，再继续研究Derby！不就是一年的阅读量吗！我可以化3年去研究如何做一个数据库其实还是很值得的！有的人搞 IT一辈子自己什么都没有做，也根本没有研究别人的东西！

作为一个IT落后于别国若干年的、从事IT的下游产业“外包”的国家的IT从业人员，我认为还是先研究别人的优秀的东西比较好！可以先研究别人的，然后消化，学为己用！一心闭门造车实在遗憾！

三、 JMS方向

JMS可能对大家来说是一个比较陌生的方向！其实JMS是一个比较容易理解，容易上手的方向。主要是Java消息服务，API也是相当简单的。不过在企业应用中相当广泛。下面就介绍几个吧！

1. MOM4J

MOM4J是一个完全实现JMS1.1规范的消息中间件并且向下兼容JMS1.0与1.02。它提供了自己的消息处理存储使它独立于关系数据与语言，它的客户端可以用任何语言开发。它可以算是一个小麻雀，很全实现也比较简单！它包含一个命名服务器，一个消息服务器，同时提供自己的持续层。设计也相当的巧妙，完全利用操作系统中文件系统设计的观念。代码也很少，250页左右，最近我在写该实现的源代码阅读方面的书，希望明年年中能与大家见面！下载地址：http://mom4j.sourceforge.net/index.html。

2. OpenJMS

OpenJMS是一个开源的Java Message Service API 1.0.2 规范的实现，它包含有以下特性：

1. 它既支持点到点（point-to-point）（PTP）模型和发布/订阅（Pub/Sub）模型。

2. 支持同步与异步消息发送 。

3. JDBC持久性管理使用数据库表来存储消息 。

4. 可视化管理界面。

5. Applet支持。

6. 能够与Jakarta Tomcat这样的Servlet容器结合。

7. 支持RMI, TCP, HTTP 与SSL协议。

8. 客户端验证 。

9. 提供可靠消息传输、事务和消息过滤。

很好的JMS方向的开源项目！我目前也在研究它的源代码！学习它可以顺便研究JNDI的实现、以及网络通信的细节。这是我JMS方向研究的第二个开源项目。代码量1600页左右吧！下载地址：http://openjms.sourceforge.net/index.html

3. ActiveMQ

ActiveMQ是一个开放源码基于Apache 2.0 licenced 发布并实现了JMS 1.1。它能够与Geronimo，轻量级容器和任Java应用程序无缝的给合。主要是Apache的可以任意的使用和发布哦！个人比较喜欢Apache 的源代码！下载地址：http://activemq.codehaus.org/

4. JORAM

JORAM一个类似于openJMS分布在ObjectWeb之下的JMS消息中间件。ObjectWeb的产品也是非常值得研究的！下面我还会给大家另外一个ObjectWeb的产品。下载地址：http://joram.objectweb.org/

我个人推荐：OpenJMS和ActiveMQ！

四、 EJB方向

EJB一个比较“高级”的方向。Sun公司曾经以此在分布式计算领域重拳出击。不过自从出现了Spring、Hibernation……后似乎没落了！这个方向单独开源的也比较少，主要EJB是和JNDI、JDBC、JMS、JTS、JTA结合在一起的是以很少有单独的。下面推荐两个不过好像也要下载其它类库。

1. EasyBeans

ObjectWeb的一个新的项目，一个轻量级的EJB3容器，虽然还没有正式发布，但是已经可以从它们的subversion仓库中检出代码。代码量比较小600页左右，熟读它可以对网络编程、架构、RMI、容器的状态设计比较了解了！即学会EJB又能学习其它设计方法何乐而不为哦！下载地址：http://easybeans.objectweb.org/

2. OpenEJB

OpenEJB是一个预生成的、自包含的、可移植的EJB容器系统，可以被插入到任意的服务器环境，包括应用程序服务器，Web服务器，J2EE平台， CORBA ORB和数据库等等。OpenEJB 被用于 Apple的WebObjects。听起来很好，我目前没有研究过。不知道我就不推荐了。下载地址：http://www.openejb.org/

五、 J2EE容器

上面谈了这么多，都是J2EE的各个方向的。其实J2EE是一个规范，J2EE的产品一般要求专业提供商必须提供它们的实现。这些实现本身就是J2EE容器。市场上流行的J2EE容器很多，在开源领域流行的只有很少，很少。其中最著名的是JBoss。

1. JBoss

在J2EE应用服务器领域，Jboss是发展最为迅速的应用服务器。由于Jboss遵循商业友好的LGPL授权分发，并且由开源社区开发，这使得Jboss广为流行。另外，Jboss应用服务器还具有许多优秀的特质。

其一，它将具有革命性的JMX微内核服务作为其总线结构；

其二，它本身就是面向服务的架构（Service-Oriented Architecture，SOA）；

其三，它还具有统一的类装载器，从而能够实现应用的热部署和热卸载能力。因此，它是高度模块化的和松耦合的。Jboss用户的积极反馈告诉我们，Jboss应用服务器是健壮的、高质量的，而且还具有良好的性能。为满足企业级市场日益增长的需求，Jboss公司从2003年开始就推出了 24\*7、专业级产品支持服务。同时，为拓展Jboss的企业级市场，Jboss公司还签订了许多渠道合作伙伴。比如，Jboss公司同HP、 Novell、Computer Associates、Unisys等都是合作伙伴。

在2004年6月，Jboss公司宣布，Jboss应用服务器通过了Sun公司的J2EE认证。这是Jboss应用服务器发展史上至今为止最重要的里程碑。与此同时，Jboss一直在紧跟最新的J2EE规范，而且在某些技术领域引领J2EE规范的开发。因此，无论在商业领域，还是在开源社区，Jboss 成为了第一个通过J2EE 1.4认证的主流应用服务器。现在，Jboss应用服务器已经真正发展成具有企业强度（即，支持关键级任务的应用）的应用服务器。

Jboss 4.0作为J2EE认证的重要成果之一，已经于2004年9月顺利发布了。同时，Jboss 4.0还提供了Jboss AOP（Aspect-Oriented Programming，面向方面编程）组件。近来，AOP吸引了大量开发者的关注。它提供的新的编程模式使得用户能够将方面（比如，事务）从底层业务逻辑中分离出来，从而能够缩短软件开发周期。用户能够单独使用Jboss AOP，即能够在Jboss应用服务器外部使用它。或者，用户也可以在应用服务器环境中使用它。Jboss AOP 1.0已经在2004年10月发布了。 很有名吧！可以下载一个用一下，下载地址：http://www.jboss.org/

关于JBoss的使用资料也非常多，甚至比商业软件的还多。有机会研究吧！

2. JOnAS

JOnAS是一个开放源代码的J2EE实现，在ObjectWeb协会中开发。整合了Tomcat或Jetty成为它的Web容器，以确保符合 Servlet 2.3和JSP 1.2规范。JOnAS服务器依赖或实现以下的Java API：JCA、JDBC、JTA 、JMS、JMX、JNDI、JAAS、JavaMail 。下载地址：http://jonas.objectweb.org/

3．Apache Geronimo

Apache Geronimo 是 Apache 软件基金会的开放源码J2EE服务器，它集成了众多先进技术和设计理念。 这些技术和理念大多源自独立的项目，配置和部署模型也各不相同。 Geronimo能将这些项目和方法的配置及部署完全整合到一个统一、易用的模型中。作为符合J2EE标准的服务器，Geronimo提供了丰富的功能集和无责任 Apache 许可，具备“立即部署”式J2EE 1.4容器的各种优点，其中包括：

1. 符合J2EE1.4标准的服务器 。

2. 预集成的开放源码项目 。

3. 统一的集成模型 。

4. 可伸缩性、可管理性和配置管理功能。

我一直比较推荐Apache的产品。主要是可以任意自由地使用。下载地址：http://incubator.apache.org/projects/geronimo/

六、 其它

讲了这么多大家可能很厌烦了！是不是很多很多啊！其实不然，我们不会的太多太多了！不会的太多太多了。不管你是不是J2EE高手，还是J2SE高手，有些东西你要绝对很精明的。例如：1.Java的Collections Framework就是java的数据结构了，不仅要吃透它，还要能按照需要扩展它，利用其思想创建一个自己的数据结构。2.网络编程肯定要会吧，现在以及以后很多程序都是不在同一台机器上的，不会网络怎么行哦！3.IO肯定要会的吧！你的程序难道不用输入输出数据啊！整个IO包加NIO也有600多页的源代码哦！4.JDBC你要会吧！数据库都不会，在你的企业应用中你的数据又保存到哪里啊！文件中――太落后了吧！典型的没有学过J2EE。尽管数据库背后也是采用文件保存的。5.Serverlet、JSp你要是做网页做网站，肯定要做到。问你一个简单的问题，网页中如何实现分页啊！有具体方法的就在本文章后发言吧！6. Ant要会吧！java语言中发布的工具，类似与c中的make工具。7.JUnit用过吧！单元测试软件。你不要啊！你的软件就没有bug！你牛！（建议大家研究研究其源代码，很有用的框架，包含大量的设计模式，源代码不到100页！看了只能感叹――高手就是高手）细心的朋友可以看到在你使用的很多 IDE工具中都有JUnit哦！就是它。

一切的一切才刚刚开始！有兴趣，有需要你可以研究数据库连接池的框架，如：C3P0、Jakarta DBCP、 DBPool….可以研究J2EE框架Spring……. Web框架Struts……持久层框架Hibernate…..甚至开发工具Eclipse…..Sun领导的点对点通信的JXTA…..报表工具 JFreeChart、JasperReports…..分布式网络编程的CORBA、网络通信的JGROUPS、XML解析的xerces…..（在不经意间开源已经步入你的电脑，不信啊！你JDK的安装目录jdk1.6.0 src com sun org apache就是Xerces，一个XML解析的著名的开源 项目）

不管怎么样我还是建议从基本的做起，学精J2SE，熟读它的源码，准确了解其设计理念，然后分头击破J2EE――一口吃不成一个胖子！不要贪多贪广！脚踏实地就可以了！

假如你是初学者，请你到我的博客看那些我给你推荐的书籍！见《Java 推荐读物与源代码阅读》这篇文章。假如你是一位J2SE高手想进入J2EE方向，请先了解设计模式，然后搞懂其各个方向的API，接着可以研究各个方向的开源项目提高自己！充实自己！接着就是服务于你的行业，做你想干的事！谢谢！你已经是高手了，不需要来听我唠叨了！不过我还是恳切希望你能到我的博客给我唠叨！

13:13

浏览 (979)

评论 (3)

分类: Java

收藏Spinner

2009 - 01 - 07

缩略显示

Java 推荐读物与源代码阅读

Java 推荐读物与源代码阅读

江苏无锡 缪小东

1. Java语言基础

谈到Java语言基础学习的书籍，大家肯定会推荐Bruce Eckel的《Thinking in Java》。它是一本写的相当深刻的技术书籍，Java语言基础部分基本没有其它任何一本书可以超越它。该书的作者Bruce Eckel在网络上被称为天才的投机者，作者的《Thinking in C++》在1995年曾获SoftwareDevelopment Jolt Award最佳书籍大奖，《Thinking in Java》被评为1999年Java World“最爱读者欢迎图书”，并且赢得了编辑首选图书奖。作者从1986年至今，已经发表了超过150篇计算机技术文章，出版了6本书（其中4本是关于C++的），并且在全世界做了数百次演讲。他是《Thinking in Java》、《Thinking in C++》、《C++ Inside & Out》《Using C++》和《Thinking in Patterns》的作者，同时还是《Black Belt C++》文集的编辑。他的书被读者称为“最好的Java参考书……绝对让人震惊”；“购买Java参考书最明智的选择”；“我见过的最棒的编程指南”。作者的非凡才华，极其跨越语言的能力，使作者被选为Java发展10年间与Java关系最密切的10个人物之一。

《Thinking in Java》讲述了Java语言的方方面面，很多Java语言的老手都评价“这是一本将Java语言讲得相当丑陋的书”。该书谈及了java语言的很多细节，每一个方面都是相当深刻的。通过本书你可以看到“丑陋的”java语言。

网络上关于java语言讲解的视频很多很多，其中不凡有垃圾。《翁恺—JAVA语言》可能是你学习java语言基础的唯一选择，该讲座基本按照《Thinking in Java》这本书讲解，其中不凡有翁老师的很多有意思的笑话。我很幸运学习就是从此视频开始的。内容包括30讲，我总共看了3遍。

不过，对于初学者我不太推荐使用《Thinking in Java》，我比较推荐Prentice Hall PTR 的《Core Java 2》国内称为《Java 2 核心技术》，目前是第七版。网络上大家都可以下载到电子版。Oreilly的《Java in a nutshell》也是一个不错的选择。读完以上两本后，你可以看看翁恺老师的视频，接着可以研究《Thinking in Java》了。

2. Java数据结构

市面上关于Java数据结构的书本身就很少很少。大致有APress 的《Java Collections》，Jones 和Bartlett 的《Data Structures in Java》、《Object-oriented Data Structures Using Java》以及Prentice Hall 出版的《Data Structures and Algorithms in Java》 (Dec 19, 2005)还有一本就是《Data Structures And Algorithms With Object-oriented Design Patterns In Java》。很幸运我的第一本英文书就是APress 的《Java Collections》（本书在国内可能根本就没有中文版――只能下载英文版了），很不错，讲得很有条例、很简单，是一本完完全全Java Collections API介绍的书籍，其中不凡有扩展API的例子。这是我推荐你学习java数据结构的唯一一本好书。其它的Jones 和Bartlett的那两本国内好像有一本中文版，想看你也可以看看。

在学习完API后，你可以看看java.util包中对应的类了。不过只有在学习过设计模式后你才有可能完全理解整个Java Collections Framework。Java Collections Framework使用了很多著名的设计模式如：迭代器（Iterator）模式，工厂方法模式、装饰器模式、适配器模式等等。通过研究 java.util包中数据结构的源代码，你可以知道臭名昭著的Properties类的设计了，同时可能基本具备设计简单的数据结构的能力了。

所谓学习无止境，学习完Sun提供了Java Collections Framework后，你可以研究Apche的另一个Java Collections Framework，很有意思哦。互为补充的两个Framework。

在大家学习、研究Java Collections之前，我提示一下Java Collections主要包括以下三部分：接口（Interface）、实现（Implemention）和算法（Algorithm）。

1. 接口主要有List、Set、Queue和 Map。List 、Se t和Queue是 Collection接口的子接口。

2. 实现主要是实现这些接口的具体类。如实现List接口的ArrayList、LinkedList、Stack和Vector；实现Set接口的 HashSet、TreeSet 和LinkedHashSet；实现Queue接口的PriorityQueue、SynchronousQueue等等；实现Map接口的 HashMap、TreeMap、Hashtable、Properties、WeakHashMap等等。

3. 算法主要是由Arrays类和Collections类提供的，它是整个Java Collection Framework算法的核心。支持各种类型的排序，查找等常用操作。

Java Collections中包含两个版本的数据结构，主要是原先的支持同步的数据结构和后来不支持同步的数据结构。

Java Collection Framework在使用Comparator和Comparable接口支持排序。同时提供新旧两个版本的迭代器Iterator和Enumeraton，以及它们如何转换等等。

在java.util包中的Obserable接口和Observer类是考察者模式的核心。

……

3. Java IO

市面上关于IO的书籍也仅仅只有Oreilly出版社的两本，都是Elliotte Rusty Harold的著作。两本书的风格基本一致，推荐阅读是第一版的《Jvava I/O》，讲得比较浅显，内容相对比较集中，实例也很多。第二版今年5月国外才出版，很有幸我在网络上下载了第二版，讲得极其详细――726页的大块头（我化了两个星期），这次将NIO和IO和在一起，还包括J2ME部分的，不过串口、并口通信部分好像类库支持不够，自己不能实际操作。

与第一版的《Jvava I/O》一起的Oreilly还有一本《Jvava NIO》，也是很不错的哦。

大家在依次阅读完《Jvava I/O》以及《Jvava NIO》后，可以研究java.io包中的源代码了。在大家研究源代码前我给点提示：

Java的io包主要包括：

1. 两种流：字节流（byte Stream）和字符流（character stream），这两种流不存在所谓的谁代替谁、谁比谁高级之说，它们互为补充，只是侧重点不同而已。

2. 两种对称：1.字节流、字符流的对称；2.输入、输出的对称。

3. 一个桥梁：将字节流转变为字符流的InputStreamReader和OutputStreamWriter。

其中必须注意：

1. PipedInputStream和PipedOutputStrem是两个比较有趣的类。

2. 支持Buffered的流是我们经常使用的类。

3. 装饰器（Decorator）模式在java最著名的应用就是用于io的设计。仔细研究各个Filter流与具体流的关系，多看设计模式的书籍。相信你会有所所获。

4. 学习好io包，是研究net包，rmi包……的基础哦！

4 . Java数据库

数据库的书籍太多太多了，也是太烂太烂了！这方面的书我基本都研究过，推荐的你就看看Apress的《JDBC Recipes A Problem Solution Approach 》很不错，国外2005年底才出版，（国内好像没有中文版，不过出了中文版也不一定值得看――国内经常将国外的书翻译得一塌糊涂、不堪入目）不过我们真的很幸运，网络上有电子版的。值得一看。推荐我看的第一本比较满意的――Wiley出版的《Java Database Bible》，讲得很不错！Sun公司自己的关于JDBC API介绍的那一本《JDBC API Tutorial andRefernece》也不错。我第二本JDBC的就是研究的这套API。

不过目前这些书都是一些相对比较浮浅的API应用的书籍。有机会我会给大家带来介绍JDBC API以及JDBC实现内部细节的书！我尽快努力，同时希望得到大家的支持！

顺便给学习JDBC的朋友一点提示：

JDBC的学习和使用主要是这套API，其使用过程也是极其简单，下面是使用JDBC的一般流程：

1. 加载某个数据库的驱动（Driver类），通常使用Class.forName(“驱动的类名“);

2. 连接数据库――

Connection con = DriverManager.getConnection(url,username,password)；

3. 得到会话――Statement stmt = con.createStatement();

4. 执行操作――Result rs = stmt.executeQuery(“SQL查询语句”)；

5. 处理结果――

while(rs.next()){

String col1 = rs.getString(1);

……

}

简单吧！整个JDBC中可以变化的一般是：

1. 可以由Connection对象创建Statement、PreparedStatement和CallableStatement创建三种类型的Statement。

2. 可以创建多种类型的ResultSet：支持单向移动和个自由移动；可更新的和不可更新的；支持不同等级的交易的…..

3. 数据输入的批处理。

4. 结果集中特殊类型（Blob、Clob、Arrary和Ref、Struct）列的操作。

5. 这些特殊类型的录入数据库。

6. javax.sql包中特殊结果集（CachedRowSet、JdbcRowSet、WebRowSet）的操作。

7. 其它的就是一个DataSource了，也很简单！一个J2EE中的被管理对象

简单吧！相信大家很快就会征服JDBC。

5. Java 网络编程

网络编程――一个神秘的、充满挑战的方向。不过在谈Java网络编程之前首先感谢Sun公司的开发人员，因为它们天才的设想，充满智慧的架构，使广大java程序员学习java网络编程变得异常简单。

Java网络编程方面的书，我推荐O'Reilly的《Java Network Programming》，目前已经第三版了，以前的版本市面上肯定有！网络上早有第三版的电子版，国外2004年出版，706页哦！讲得很全，比较深入，太深入的可能由于Sun有些东西没有完全公开，所以也就不好讲了，有兴趣的可以下载看看！第二本还是O'Reilly 1998年出版的《Java distributed computing 》，基础部分写得比较详细，后面的实例还是值得研究的。

在大家阅读这些书之前，给大家一点提示：

java网络编程其实相对比较简单，入门也很快很快。java网络编程主要包括两个部分：1.Socket；2.URL部分。不过第二部分也完全建立在第一部分的基础上。

1. Socket包括客户端的Socket和服务器端的ServerSocket。还有就是DatagramSocket和DatagramPacket，它对应于UDP通信协议。 总之，Socket部分是建立其它高级协议的基础。

2. URL类是一个网络资源定位器，通常和具体的网络协议如HTTP，FTP，Telnet……相关。通过该类可以连接网络上的资源，通过其 openStream可以以io包中的流（InputStream）的形式读取网络资源；通过其OpenConnection方法，可以打开一个连接，在此连接上可以不仅可以完成读的操作，还可以完成写的操作。

Java的网络编程大体包括以上两部分。网络编程和IO以及多线程部分非常密切，在学习此部分前大家一定对这两部分了解比较透彻。

学习了以上部分你可以研究java.net包中的与此相关的源代码了！研究所有的源代码还为时尚早。在整个net包中包含：ContentHandlerFactory、URLStreamHandlerFactory、URLStreamHandler、 URLClassLoader等辅助类，它们构成了java.net网络编程的框架，通过研究其源代码，你不仅可以快速理解java.net包，还可以为以后扩展该包打下基础，甚至可以将此思维方式运用到自己的项目中。

到此为止你对java.net包应该才了解60％，还有一部分你可以使用JDecompiler之类的反编译软件打开你JDK安装目录下/jdkxxx /jre/lib目录中的rt.jar，用WinRAR之类的软件打开它的sun.net包，反编译所有的文件，它是URL类工作的细节。当研究完该 sun.net包，你就会对整个网络编程很熟悉很熟悉了。

一切看起来我们已经对网络编程很精通了。其实不然，刚刚开始而已，要想深入，请继续吧！网络上很多优秀的网络编程库甚至软件可以为我们“添加功力”。如 Apache的HttpCore和HTTPConnection 是两个和HTTP协议相关库；JGroups是研究分布式通信、群组通信的必读库；接着我们可以研究P2P的软件包，如Sun公司的JXTA，它可能是 java平台点对点通信未来的标准哦！接着你可以研究成熟得不得了，使用极其广泛得P2P软件Azureus！www.sourceforge.net可以下载到！

千里之行始于足下！Just do it ！（目前我也只研究了net包，其它的会在不久的将来继续深入。Sun公司因为某些原因没有公开net的其它实现细节，在其允许将其源代码以文字的形式加以研究，以及允许将其没有公开的实现写入书中时，我很希望能出一本java网络编程的书籍，以飧广大读者！！）

6. Servlet和JSP

Servlet、JSP的书也是满地都是！值得推荐的也仅仅两三本。实推Addison Wiley的《Servlets and JavaServer pages ：The J2EE Technology Web Tier》，又是一本很厚的哦！国外2003年出版、784页，讲得比较全，例子也很多，特别是第八章Filter，举了几个不错的例子。其它所有我看到的关于Servlet和JSP的书都没有如此深入的！（可能有我没有看到而已）。O’reilly的《Java Servlet Programming》和《Java Server Pages》相对比较好懂一些，可以读读！

在大家学习Servlet和Jsp之前我还是要提醒一下：

本质上说Servlet就是一个实现Servlet接口的、部署于服务器端的服务器端的程序罢了！它可以象写其它任何java应用程序一样编写，它可以操作数据库、可以操作本地文件、可以连接本地EJB……编写Servlet程序的一般流程为：

1. 继承一个HttpServlet类；

2. 覆盖其doGet、doPost方法；

3. 在覆盖方法的内部操作方法参数HttpServletRequest和HttpServletResponse。

4. 读取请求利用HttpServletRequest。利用HttpServletRequest你可以操作Http协议的协议头、可以得到请求的操作方法、可以得到请求的路径、可以得到请求的字符串、以及和请求客户相关的信息，更主要的你可以得到Cookie和HttpSession这两个对象。

5. 利用Cookie你可以操作“甜心”对象或者将其写入HttpServletResponse中。

6. 向客户输出信息可以使用HttpServletResponse。使用HttpServletResponse可以写入各种类型的协议头、可以增加Cookie、可以重定向其它URL、可以向客户发送Http协议的状态码。

7. 利用HttpSession在会话内完成你想实现的任何功能。

同时Servlet还提供了一些事件和事件监听器（简单的观察者模式而已）。还有就是过滤器（Filter）和包装器（ServletRequestWrapper、ServletResponseWrapper）――简单的流的使用和装饰器模式的使用。

学习Sevlet、JSP必然要部署到服务器中，记住通常文件部署的步骤和参数的设置以及在程序中如何使用就可以了。

完全理解Servlet后，学习jsp相对比较容易了！Jsp完全建立在Servlet的基础上，它是为了迎合那些喜欢在Html文档中嵌入脚本（如：PHP之类的网页编程语言）的程序员的需要罢了！学起来也相当的容易！

一切看起来似乎那么的风平浪静，简单好学！简单的表象背后有其复杂的机理。要想对Servlet和Jsp彻底研究，你得研究Tomcat等开源软件的具体实现。它无非就是一个服务器，在客户利用网页通过HTTP协议向服务器发送请求后，服务器将此HTTP请求转化为相应的 HttpServletRequest对象，调用你编写的Servlet罢了，在你的Servlet中你肯定操作了此 HttpServletRequest了吧，同时操作了HttpServletResponse了吧，服务器就将此 HttpServletResponse按照HTTP协议的要求利用HTTP协议发送给你的浏览器了！在服务器端的Jsp网页在被客户请求后，Tomcat会利用编译软件，使用javax.servlet.jsp包中的模板，编译此jsp文件，编译后就是一个Servlet！以后的操作和 Servlet完全一样哦！

在Servlet和Jsp的基础上出现了，所谓的高级技术：JSTL，Struts……无非就是一些标签和MVC模式的使用。

继续前进吧！胜利就在前方！！

7. 多线程

一个看起来很神秘，却很容易上手、很难精通的方向！

我推荐两本我感觉很好的书籍。首先是我第一本能上手看的这方面的书，Sams 1998年出版的《Java Thread Programming》，写得暴好，很容易读懂，我有空还时常看当时的笔记！要知道怎么好你自己看吧！第二本OReilly三次出版的《Java Threads》，最新是2004版，国内好像有中文版，推荐你还是看英文版的吧！书中谈到了与多线程相关的N个方向，如IO、Swing、 Collection等等。

给大家一点提示吧！java类库中与多线程相关的类不是很多，主要有：Thread、ThreadGroup以及ThreadLocal和 InheritableThreadLocal四个类和一个Runnable接口；关键字synchronize、volatile ；以及Object对象的wait、notify、notifyAll方法！

1 Thread是多线程的核心类，提供了一系列创建和操作多线程的方法。

2 ThreadGroup是一个管理Thread的工具类。

3 ThreadLocal和InheritableThreadLocal为Thread提供了一个类似保险箱功能的存储线程对象的类！

4 Runnable不用说了吧！

5 synchronize是同步方法和同步块的核心哦！多个线程调用此方法时，只有一个线程可以使用此方法，其它方法阻塞，从而保证被操作对象内部状态完整性。某个线程调用带有synchronize的方法或块时会得到该对象的对象锁，完成块中的操作后释放此对象锁，从而其它对象可以继续操作。

6 wait、notify、notifyAll提供了有效的等待/通知机制。Java语言中每一个对象都有一个休息室，任何线程在其操作的对象的状态不满足的情况下，在该对象的休息室中休息，释放对象锁；当其它线程操作该对象后，唤醒休息室中的线程，它们再检查条件，当条件满足后，执行相应的操作。

多线程大致就这么多基础的！简单吗！这对于一个真正的程序员应该是不够的，真正对多线程要有所掌握，请您研究java.util.concurrent包吧！大师Doug Lea的作品，原先是一个开源的一致性编程的库，后来被Sun公司并入java类库。作者的网站上也有另外一个版本的该类库！值得研究的好东西!Hibernation、OpenJMS等开源软件都使用了此包！

8. 设计模式

谈到设计模式很多人多会推荐GOF的那本，该书在Amzon上是五星级的推荐书籍。不过对于学习java没多久的、特别是java初学者，我很不推荐这本书。主要是该书的例子基本都是C++的，很多细节没有讲述得足够清楚。

我给大家推荐的第一本是阎宏博士的《Java 与模式》，它是第一本中国人自己写的关于设计模式的书籍，写的比较有趣，融合了很多中华民族的文化和观念，例子、类图都比较多，且相对简单！非常不错的入门书籍――又是大块头哦！

其次我推荐Wiley出版社出版的《Pattern In Java》一套三本，我才看了第一本，好像第二本不怎么样，第三本还不错！

第三本是中文翻译版的关于多线程模式的（很难得的中文翻译版）中国铁道出版社2003年出版的《Java多线程设计模式》，将多线程模式讲得非常浅显，配有大量的图例，每章都有习题，最后有答案！我研究多线程模式就是由它开始的！

第四本，今年出版的Head First系列的《Head First Design Pattern》，秉承Head First系列图书的优点，大量的类图、丰富的实例、有趣的注解，值得购买！

其次在J2EE方向你可以研究阅读Addison Wesley 2002年出版的《Patterns of Enterprise Application Architecture》，众多大腕的作品，讲企业消息集成的！Sun提供的《J2EE PATTERNS SL500》也很好！晚了推荐那一本Amzon 4星半的《Holub on patterns》，大师的作品，提供了，很值得研究的例子，不过对上面四本不是很熟悉的读者，最好不要读它！可能会让你比较累！

我学习设计模式经过一段很曲折的路线，前前后后大约看了20本，阎宏博士的《Java 与模式》我看了4遍，还排除我第一次基本没看懂的看！记得研一时老师给我们讲了GOF的那本，作为选修课，我和它们计算机系的硕士、博士们一起，到最后一个班40－50个人，不超过3个人明白，我也没有明白任何一点（基础差吧――主要我对C++语言一点都不了解），凭我不伏输的性格，我认为我对java语言理解还可以，我就借了《Java 与模式》，结果还是基本没看懂。很有幸的是读研三时，听过了上交大饶若楠老师关于Java OOP语言的讲座，我懂了组合书籍模式等三种设计模式后，对其它模式有了强烈的兴趣和要征服它的愿望！工作后我买的第一本就是《Java 与模式》，第一遍花了2个月研究了这个1000多页的大块头，后来第三遍15天左右就可以搞定，笔记记了一大本！从此一发不可收拾。

选对书、埋头研究。相信很快就会入门的！

学习Java语言8个简单的部分，这只是我们研究Java语言的开始！这些都懂了充其量一个java程序员而已，后面的路很长很长！我们可以继续研究数据库实现的源代码、Servlet服务器的源代码、RMI、EJB、JNDI、面向方面编程、重构、ANT工具、Eclipse工具、Spring工具、 JBoss、JOnAS、Apache Geronimo等J2EE服务器！研究了这些你可能会成为一个出色的J2EE Architecture！你可以继续研究剖析器、编译器、JNODE（java写的操作系统）……

感谢大家有此耐心，听我罗罗嗦嗦大半天！感谢大家的阅读，感谢群里的朋友！这篇文章主要应群里朋友的呼声――不知道如何选书、不知道从何看起！大半天的功夫完成赶此文章，字句上难免有失误，同时由于能力有限不凡有错误！请阅读后批评指正！

上面基本是我研究java语言的顺序，以上书籍都是我阅读过的，不存在替任何出版社宣传的成分！有的方法可能不适合你，假如你能收获一点，两点甚至更多，请你不要吝啬推荐给你的朋友――共同学习！

感谢大家的阅读；感谢互联网的设计者；感谢java的设计师；感谢www.open-open.com和www.sourceforge.net网站！

13:12

浏览 (4034)

评论 (9)

分类: Java

收藏Spinner

2009 - 01 - 07

缩略显示

JAVA简单编码规则

一、规范存在的意义

应用编码规范对于软件本身和软件开发人员而言尤为重要，有以下几个原因：

1、好的编码规范可以尽可能的减少一个软件的维护成本 , 并且几乎没有任何一个软件，在其整个生命周期中，均由最初的开发人员来维护；

2、好的编码规范可以改善软件的可读性，可以让开发人员尽快而彻底地理解新的代码；

3、好的编码规范可以最大限度的提高团队开发的合作效率；

4、长期的规范性编码还可以让开发人员养成好的编码习惯，甚至锻炼出更加严谨的思维；

二、命名规范

1、一般概念

1、尽量使用完整的英文描述符

2、采用适用于相关领域的术语

3、采用大小写混合使名字可读

4、尽量少用缩写，但如果用了，必须符合整个工程中的统一定义

5、避免使用长的名字（小于 15 个字母为正常选择）

6、避免使用类似的名字，或者仅仅是大小写不同的名字

7、避免使用下划线（除静态常量等）

2、标识符类型说明

1、包（ Package ）的命名

Package 的名字应该采用完整的英文描述符，都是由一个小写单词组成。并且包名的前缀总是一个顶级域名，通常是com，edu，gov ，mil ，net ，org 等<如：com.yjhmily.test>

2、类（ Class ）的命名

类名应该是个一名词，采用大小写混合的方式，每个单词的首字母大写。尽量保证类名简洁而富于描述。使用完整单词，避免缩写词 ( 除非工程内有统一缩写规范或该缩写词被更广泛使用，像 URL ， HTML)<如： FileDescription>

3、接口（ Interface ）的命名

基本与 Class 的命名规范类似。在满足 Classd 命名规则的基础之上，保证开头第一个字母为 ”I” ，便于与普通的 Class 区别开。其实现类名称取接口名的第二个字母到最后，且满足类名的命名规范；<如： IMenuEngine>

4、枚举（ Enum ）的命名

基本与 Class 的命名规范类似。在满足 Classd 命名规则的基础之上，保证开头第一个字母为 ”E” ，便于与普通的 Class 区别开。<如： EUserRole>

5、异常（ Exception ）的命名

异常（ Exception ） 通常采用字母 e 表示异常，对于自定义的异常类，其后缀必须为 Exception<如： BusinessException>

6、方法（ Method ）的命名

方法名是一个动词，采用大小写混合的方式，第一个单词的首字母小写，其后单词的首字母大写。方法名尽可能的描述出该方法的动作行为。返回类型为 Boolean 值的方法一般由“ is ”或“ has ”来开头<如： getCurrentUser() 、 addUser() 、 hasAuthority()>

7、参数（ Param ）的命名

第一个单词的首字母小写，其后单词的首字母大写。参数量名不允许以下划线或美元符号开头，虽然这在语法上是允许的。参数名应简短且富于描述。<如： public UserContext getLoginUser(String loginName);>

8、常量字段 （ Constants ）的命名

静态常量字段（ static final ） 全部采用大写字母，单词之间用下划线分隔；<如： public static final Long FEEDBACK;public static Long USER\_STATUS;>

三、注释规范

一个很好的可遵循的有关注释的经验法则是：

问问你自己，你如果从未见过这段代码，要在合理的时间内有效地明白这段代码，你需要一些什么信息？？？

1、一般概念

1、注释应该增加代码的清晰度

2、保持注释的简洁

3、在写代码之前或同时写注释

4、注释出为什么做了一些事，而不仅仅是做了什么

2、注释哪些部分

1、Java 文件：必须写明版权信息以及该文件的创建时间和作者；

2、类：类的目的、即类所完成的功能，以及该类创建的时间和作者名称；多人一次编辑或修改同一个类时，应在作者名称处出 现多人的名称；

3、接口： 在满足类注释的基础之上，接口注释应该包含设置接口的目的、它应如何被使用以及如何不被使用。在接口注释清楚的前提下对应的实现类可以不加注释；

4、方法注释： 对于设置 (Set 方法 ) 与获取 (Get 方法 ) 成员的方法，在成员变量已有说明的情况下，可以不加注释；普通 成员方法要求说明完成什么功能，参数含义是什么且返回值什么；另外方法的创建时间必须注释清楚，为将来的维护和阅读 提供宝贵线索；

5、方法内部注释： 控制结构，代码做了些什么以及为什么这样做，处理顺序等，特别是复杂的逻辑处理部分，要尽可能的给出 详细的注释；

6、参数： 参数含义、及其它任何约束或前提条件；

7、属性： 字段描述；

8、局部 ( 中间 ) 变量： 无特别意义的情况下不加注释；

3、注释格式

遵循工程规定的统一注释格式，一般情况下会以 codetemplates.xml 格式的文件导入 IDE(Eclipse) 或者用Eclipse默认的；

四、代码格式规范

遵循工程规定的统一代码格式，一般情况下直接使用 IDE(Eclipse) 自带的默认代码格式对代码进行格式化；

五、其他规范

JSP 文件命名

采用完整的英文描述说明 JSP 所完成的功能，尽可能包括一个生动的动词，第一个字母小写，如： viewMessage.jsp 、 editUser.jsp 等。

六、工程特有命名规范

1、持久层

1、 Hibernate 映射文件及实体

与数据库表名称完全对应；

如： Advertisement.hbm.xml 、 Advertisement.java

2、数据访问 DAO

DAO 接口和实现类名称必须完全符合正常接口和实现类的命名规则，且最后以 ”DAO” 结尾

DAO 内的数据访问方法必须足够抽象的描述出对数据库的基本 CRUD 操作；

如： ICrossAdDAO( 接口 ) 、 CrossAdDAO( 实现类 )

3、各种操作数据库的 HQL 配置文件

HQL 文件的个数原则上与系统的 Services 层的服务个数相等，且以服务名称命名 HQL 文件；

如： resource.hbm.xml

2、服务层

1、服务接口和实现

服务接口和实现类必须完全符合正常接口和实现类的命名规则；以工程定义的服务名为主体，并统一以 ”Serv” 结尾

如： IResourceServ( 服务接口 ) 、 ResourceServ( 接口实现类 )

2、服务接口方法

方法名是一个动词，采用大小写混合的方式，第一个单词的首字母小写，其后单词的首字母大写。方法名尽可能的描述出该 方法的动作行为。

返回类型为 Boolean 值：用“ is ”或“ has ”来开头

得到某数据： get+ 数据描述名词复数 + 数据类型；

得到所有数据： get+All+ 数据描述名词复数 + 数据类型；

通过 XXX 得到 / 查询某数据： get/query+ 数据描述名词复数 + 数据类型 +By+ 条件；

添加某数据： save/add+ 数据描述名词 ()

更新某数据： save/update+ 数据描述名词；

删除某数据： delete/remove+ 数据描述名词；

3、业务对象

业务名称 +BO

4、查询参数对象

凡是继承 Abst\*\*\*QuerySpec 的查询参数类全部满足以下规则：

Query+ 所要查询的数据描述名词 +Spec

作为参数传入时，参数名必须为：所要查询的数据描述名词 +Spec

如： QueryProgramSpec;

3、MVC 层

1、Action 控制层

Action 类名：功能模块名称 +Action ；

Actoin 方法名称尽可能的描述出页面迁移的去向

如： LoginAction( 登录用 action) ， toWelcome( 转向欢迎页的 action 方法 )

2、资源文件

系统全局资源文件： globalMessages\_+ 字符编码类型 +.properties

功能模块内部的资源文件： package.properties

4、Spring 配置文件

1、Action 相关配置文件

文件目录： WebRoot/WEB-INF/spring/action/ 功能模块名称 +\_ApplicationContext.xml

2、Services 相关配置文件

文件目录： WebRoot/WEB-INF/spring/services/Services\_ApplicationContext.xml

3、全局性配置文件

文件目录： WebRoot/WEB-INF/spring/工程名+\_ApplicationContext.xml

5、JSP 文件

采用完整的英文描述说明 JSP 所完成的功能，尽可能包括一个生动的动词，第一个字母小写，如： viewMessage.jsp、editUser.jsp 等。

13:05

浏览 (414)

评论 (0)

分类: Java

收藏Spinner

2008 - 12 - 08

缩略显示

Java读书目录

关键字: java

统计一下，争取每年能读3本书

01.《成功通过Sun认证 Java2程序员考试》

02.《Java实例技术手册》

03.《数据结构与算法分析--JAVA 语言描述》

04.《Junit In Action》

05.《设计模式精解》

06.《SCWCD认证专家应考指南》

07.《精通Struts》

08.《精通Hibernate》

09.《Spring In Action》

10.《Webwork In Action》

11.《J2EE核心模式》

12.《Ajax基础教程》

13.《Ajax In Action》

14.《应用RAILS进行敏捷WEB开发》

15.《ECLIPSE SWT/JFACE开发实战精解》

15:30

浏览 (493)

评论 (0)

分类: Java

收藏Spinner

2008 - 12 - 04

缩略显示

我所喜欢的Java书籍作者（国内）

文章分类:IT生活关键字: java 书籍

首先说明，我不是书托，为了避免这种嫌疑，我只评论一下2007年以前出版的书籍作者。其实那些作者也不需要我这种无名小卒来宣传，只是我决心以后尽量只看英文电子版和影印版的Java书籍了，所以写下点东西，留个纪念。

看了很多本Java书籍，其中国内的作者有这样几位给我留下了深刻的印象：

第一位，可以算得上是我Java的入门导师，张洪斌。我在2003年刚开始学习Java的时候，是伴随着考SCJP认证为目的的。在学校的图书馆里翻了很久，终于找到了一本《成功通过SUN认证 JAVA2程序员考试》，这本书可能是当时仅有的跟SCJP认证有关的书了。认真读下去，发现张老师的写作风格非常幽默，讲起概念也非常清晰。最后我以高分通过了SCJP，很有一种以后只买张老师的书的冲动。后来我也又买了一本《例释JAVA2企业版（J2EE）程序设计》，主要讲解的是EJB2.1的内容，可是实习的时候才发现EJB中除了无状态会话Bean以外，其他的部分很少有公司用，我只看了几页就不得不放到一旁，学习其他更有用的知识了。以后张老师出的书也都是有关Jboss，Websphere，Weblogic的，而我用的最多的是Tomcat，所以以后也再也没有买来看。

第二位，是我在学习Design Pattern的时候，最钦佩的人，彭晨阳。那个时候为了准备SCJD认证考试，不得不学习Design Pattern，受GoF的影响，我一直觉得Design Pattern就是天书中才该有的内容（谁让他们用C++去讲解设计模式啊，简直是难上加难）。但是无意中搜到了彭大侠的网站jdon，发现彭大侠可以将 Design Pattern用最通俗易懂的方式讲解出来，并融入到自己的jive论坛的设计当中，对彭大侠的崇拜如滔滔江水，连绵不绝（不要扁我）。图书馆中有一本彭大侠编写的《JAVA实用系统开发指南》，我借了几次，都没有办法读完。不是因为书不好，是因为书中没有提供完整的源代码，而我当时的能力又没有办法完全领悟其中的奥妙，所以只能放弃。后来还是下了一本《JAVA实用系统开发指南》的电子版，留着以后备用。

第三位，也是唯一一位的女作者，孙卫琴。在很多人眼中，女程序员的技术可能比不上男程序员，但孙JJ用她写的书证明了她技术的功底。我有两本她写的书：《精通Struts》《精通Hibernate》。两本书都是在2005年Java十年庆典的时候，用Sun论坛上的积分兑换的。（2005年是我的幸运年，我不但得到了James Gosling的亲笔签名书籍， 而且还和James Gosling合影留念）。正好我2006年开始工作的时候，是用SSH来做项目，所以有机会对两本书详细的阅读了一下。感觉孙JJ的讲解非常详细，而且在书中也描述了很多在实际项目中会遇到的技术点。唯一遗憾的是《精通Hibernate》中Hibernate的版本是Hibernate2，2006年的时候，已经出Hibernate3了，而且2和3的变化非常大，所有的包名都变了，我用Hibernate2进行学习，Hibernate3进行开发，升级步骤让我头疼了好一阵。后来孙JJ又出了很多书籍，不过都是一些Java基础教程，我就没有购买。

与孙JJ之间的故事还有一段小插曲，我曾经自己建立了一个Sun公司认证资料的网站，后来因为2年没有续费，网站关闭了，很多网友就写信索求一些难找的资料。突然有一天，我发现其中的一位居然是我最敬佩的孙JJ，原来孙JJ正在编写《TOMCAT与JAVA WEB开发技术详解(第2版)》，其中有EL和JSTL的内容。孙JJ还在信中非常认真的询问了一下SCWCD认证中JSP2.0的着重点，以及 JSP2.0在企业中的应用是否广泛（我那时非常认真的告诉她，EL和JSTL用的人太少。。。）从这点可以看出，孙JJ在写书的时候非常认真的取材，这让我对她更增加了几分敬意。

最后，不得不提一位我所见过的最烂的Java书籍作者，名字就不提了。本来是想只写最喜欢的作者的，但是说到了《精通Struts》《精通 Hibernate》，就不得不提一下他，因为这本书叫做《精通Spring》，与前两本书是同一出版社。我买下它，是因为前两本书真的很不错，以为同一个出版社，质量应该差不多，但买完以后就傻眼了。此书是我有史以来见过的最烂的书！看书还不如直接看Spring的手册，因为书的内容大部分都是摘录的。更可恨的是，该作者在china-pub书评上自己没事就给自己打五颗星，增加自己的等级，读者提出点意见的时候，他居然跑上来与广大的读者对骂。。。。。看来IT界也有素质低的啊！后来，我终于等到了《Spring in Action》的出版，才真正弄明白了什么是IoC和AOP，有了这次经历以后，我就决定尽量少买国内作者的书籍了。

大概就这么多了，其实最近我也又买了一本国人写的Java书籍，不过为了证明我不是在做广告，就先不说了。我希望国内的Java程序员能不断的推出更好的Java高级编程相关的书籍，让我们Java界的技术更上一层楼，呵呵。

10:28

浏览 (1057)

评论 (0)

分类: Java

收藏Spinner

2008 - 11 - 25

缩略显示

java的md5加密类

MD5算法是将数据进行不可逆加密的算法有较好的安全性，在国内如寿信的安全支付平台就采用此算法。

源代码如下

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

MD5 算法的Java Bean

Last Modified:10,Mar,2001

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

import java.lang.reflect.\*;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

md5 类实现了RSA Data Security, Inc.在提交给IETF

的RFC1321中的MD5 message-digest 算法。

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

public class MD5 {

/\* 下面这些S11-S44实际上是一个4\*4的矩阵，在原始的C实现中是用#define 实现的，

这里把它们实现成为static final是表示了只读，切能在同一个进程空间内的多个

Instance间共享\*/

static final int S11 = 7;

static final int S12 = 12;

static final int S13 = 17;

static final int S14 = 22;

static final int S21 = 5;

static final int S22 = 9;

static final int S23 = 14;

static final int S24 = 20;

static final int S31 = 4;

static final int S32 = 11;

static final int S33 = 16;

static final int S34 = 23;

static final int S41 = 6;

static final int S42 = 10;

static final int S43 = 15;

static final int S44 = 21;

static final byte[] PADDING = { -128, 0, 0, 0, 0, 0, 0, 0, 0,

0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,

0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,

0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };

/\* 下面的三个成员是MD5计算过程中用到的3个核心数据，在原始的C实现中

被定义到MD5\_CTX结构中

\*/

private long[] state = new long[4]; // state (ABCD)

private long[] count = new long[2]; // number of bits, modulo 2^64 (lsb first)

private byte[] buffer = new byte[64]; // input buffer

/\* digestHexStr是MD5的唯一一个公共成员，是最新一次计算结果的

　 16进制ASCII表示.

\*/

public String digestHexStr;

/\* digest,是最新一次计算结果的2进制内部表示，表示128bit的MD5值.

\*/

private byte[] digest = new byte[16];

/\*

getMD5ofStr是类MD5最主要的公共方法，入口参数是你想要进行MD5变换的字符串

返回的是变换完的结果，这个结果是从公共成员digestHexStr取得的．

\*/

public String getMD5ofStr(String inbuf) {

md5Init();

md5Update(inbuf.getBytes(), inbuf.length());

md5Final();

digestHexStr = "";

for (int i = 0; i < 16; i++) {

digestHexStr += byteHEX(digest[i]);

}

return digestHexStr;

}

// 这是MD5这个类的标准构造函数，JavaBean要求有一个public的并且没有参数的构造函数

public MD5() {

md5Init();

return;

}

/\* md5Init是一个初始化函数，初始化核心变量，装入标准的幻数 \*/

private void md5Init() {

count[0] = 0L;

count[1] = 0L;

///\* Load magic initialization constants.

state[0] = 0x67452301L;

state[1] = 0xefcdab89L;

state[2] = 0x98badcfeL;

state[3] = 0x10325476L;

return;

}

/\* F, G, H ,I 是4个基本的MD5函数，在原始的MD5的C实现中，由于它们是

简单的位运算，可能出于效率的考虑把它们实现成了宏，在java中，我们把它们

　　实现成了private方法，名字保持了原来C中的。 \*/

private long F(long x, long y, long z) {

return (x & y) | ((~x) & z);

}

private long G(long x, long y, long z) {

return (x & z) | (y & (~z));

}

private long H(long x, long y, long z) {

return x ^ y ^ z;

}

private long I(long x, long y, long z) {

return y ^ (x | (~z));

}

/\*

FF,GG,HH和II将调用F,G,H,I进行近一步变换

FF, GG, HH, and II transformations for rounds 1, 2, 3, and 4.

Rotation is separate from addition to prevent recomputation.

\*/

private long FF(long a, long b, long c, long d, long x, long s,

long ac) {

a += F (b, c, d) + x + ac;

a = ((int) a << s) | ((int) a >>> (32 - s));

a += b;

return a;

}

private long GG(long a, long b, long c, long d, long x, long s,

long ac) {

a += G (b, c, d) + x + ac;

a = ((int) a << s) | ((int) a >>> (32 - s));

a += b;

return a;

}

private long HH(long a, long b, long c, long d, long x, long s,

long ac) {

a += H (b, c, d) + x + ac;

a = ((int) a << s) | ((int) a >>> (32 - s));

a += b;

return a;

}

private long II(long a, long b, long c, long d, long x, long s,

long ac) {

a += I (b, c, d) + x + ac;

a = ((int) a << s) | ((int) a >>> (32 - s));

a += b;

return a;

}

/\*

md5Update是MD5的主计算过程，inbuf是要变换的字节串，inputlen是长度，这个

函数由getMD5ofStr调用，调用之前需要调用md5init，因此把它设计成private的

\*/

private void md5Update(byte[] inbuf, int inputLen) {

int i, index, partLen;

byte[] block = new byte[64];

index = (int)(count[0] >>> 3) & 0x3F;

// /\* Update number of bits \*/

if ((count[0] += (inputLen << 3)) < (inputLen << 3))

count[1]++;

count[1] += (inputLen >>> 29);

partLen = 64 - index;

// Transform as many times as possible.

if (inputLen >= partLen) {

md5Memcpy(buffer, inbuf, index, 0, partLen);

md5Transform(buffer);

for (i = partLen; i + 63 < inputLen; i += 64) {

md5Memcpy(block, inbuf, 0, i, 64);

md5Transform (block);

}

index = 0;

} else

i = 0;

///\* Buffer remaining input \*/

md5Memcpy(buffer, inbuf, index, i, inputLen - i);

}

/\*

md5Final整理和填写输出结果

\*/

private void md5Final () {

byte[] bits = new byte[8];

int index, padLen;

///\* Save number of bits \*/

Encode (bits, count, ;

///\* Pad out to 56 mod 64.

index = (int)(count[0] >>> 3) & 0x3f;

padLen = (index < 56) ? (56 - index) : (120 - index);

md5Update (PADDING, padLen);

///\* Append length (before padding) \*/

md5Update(bits, ;

///\* Store state in digest \*/

Encode (digest, state, 16);

}

/\* md5Memcpy是一个内部使用的byte数组的块拷贝函数，从input的inpos开始把len长度的

　　　　　 字节拷贝到output的outpos位置开始

\*/

private void md5Memcpy (byte[] output, byte[] input,

int outpos, int inpos, int len)

{

int i;

for (i = 0; i < len; i++)

output[outpos + i] = input[inpos + i];

}

/\*

md5Transform是MD5核心变换程序，有md5Update调用，block是分块的原始字节

\*/

private void md5Transform (byte block[]) {

long a = state[0], b = state[1], c = state[2], d = state[3];

long[] x = new long[16];

Decode (x, block, 64);

/\* Round 1 \*/

a = FF (a, b, c, d, x[0], S11, 0xd76aa478L); /\* 1 \*/

d = FF (d, a, b, c, x[1], S12, 0xe8c7b756L); /\* 2 \*/

c = FF (c, d, a, b, x[2], S13, 0x242070dbL); /\* 3 \*/

b = FF (b, c, d, a, x[3], S14, 0xc1bdceeeL); /\* 4 \*/

a = FF (a, b, c, d, x[4], S11, 0xf57c0fafL); /\* 5 \*/

d = FF (d, a, b, c, x[5], S12, 0x4787c62aL); /\* 6 \*/

c = FF (c, d, a, b, x[6], S13, 0xa8304613L); /\* 7 \*/

b = FF (b, c, d, a, x[7], S14, 0xfd469501L); /\* 8 \*/

a = FF (a, b, c, d, x[8], S11, 0x698098d8L); /\* 9 \*/

d = FF (d, a, b, c, x[9], S12, 0x8b44f7afL); /\* 10 \*/

c = FF (c, d, a, b, x[10], S13, 0xffff5bb1L); /\* 11 \*/

b = FF (b, c, d, a, x[11], S14, 0x895cd7beL); /\* 12 \*/

a = FF (a, b, c, d, x[12], S11, 0x6b901122L); /\* 13 \*/

d = FF (d, a, b, c, x[13], S12, 0xfd987193L); /\* 14 \*/

c = FF (c, d, a, b, x[14], S13, 0xa679438eL); /\* 15 \*/

b = FF (b, c, d, a, x[15], S14, 0x49b40821L); /\* 16 \*/

/\* Round 2 \*/

a = GG (a, b, c, d, x[1], S21, 0xf61e2562L); /\* 17 \*/

d = GG (d, a, b, c, x[6], S22, 0xc040b340L); /\* 18 \*/

c = GG (c, d, a, b, x[11], S23, 0x265e5a51L); /\* 19 \*/

b = GG (b, c, d, a, x[0], S24, 0xe9b6c7aaL); /\* 20 \*/

a = GG (a, b, c, d, x[5], S21, 0xd62f105dL); /\* 21 \*/

d = GG (d, a, b, c, x[10], S22, 0x2441453L); /\* 22 \*/

c = GG (c, d, a, b, x[15], S23, 0xd8a1e681L); /\* 23 \*/

b = GG (b, c, d, a, x[4], S24, 0xe7d3fbc8L); /\* 24 \*/

a = GG (a, b, c, d, x[9], S21, 0x21e1cde6L); /\* 25 \*/

d = GG (d, a, b, c, x[14], S22, 0xc33707d6L); /\* 26 \*/

c = GG (c, d, a, b, x[3], S23, 0xf4d50d87L); /\* 27 \*/

b = GG (b, c, d, a, x[8], S24, 0x455a14edL); /\* 28 \*/

a = GG (a, b, c, d, x[13], S21, 0xa9e3e905L); /\* 29 \*/

d = GG (d, a, b, c, x[2], S22, 0xfcefa3f8L); /\* 30 \*/

c = GG (c, d, a, b, x[7], S23, 0x676f02d9L); /\* 31 \*/

b = GG (b, c, d, a, x[12], S24, 0x8d2a4c8aL); /\* 32 \*/

/\* Round 3 \*/

a = HH (a, b, c, d, x[5], S31, 0xfffa3942L); /\* 33 \*/

d = HH (d, a, b, c, x[8], S32, 0x8771f681L); /\* 34 \*/

c = HH (c, d, a, b, x[11], S33, 0x6d9d6122L); /\* 35 \*/

b = HH (b, c, d, a, x[14], S34, 0xfde5380cL); /\* 36 \*/

a = HH (a, b, c, d, x[1], S31, 0xa4beea44L); /\* 37 \*/

d = HH (d, a, b, c, x[4], S32, 0x4bdecfa9L); /\* 38 \*/

c = HH (c, d, a, b, x[7], S33, 0xf6bb4b60L); /\* 39 \*/

b = HH (b, c, d, a, x[10], S34, 0xbebfbc70L); /\* 40 \*/

a = HH (a, b, c, d, x[13], S31, 0x289b7ec6L); /\* 41 \*/

d = HH (d, a, b, c, x[0], S32, 0xeaa127faL); /\* 42 \*/

c = HH (c, d, a, b, x[3], S33, 0xd4ef3085L); /\* 43 \*/

b = HH (b, c, d, a, x[6], S34, 0x4881d05L); /\* 44 \*/

a = HH (a, b, c, d, x[9], S31, 0xd9d4d039L); /\* 45 \*/

d = HH (d, a, b, c, x[12], S32, 0xe6db99e5L); /\* 46 \*/

c = HH (c, d, a, b, x[15], S33, 0x1fa27cf8L); /\* 47 \*/

b = HH (b, c, d, a, x[2], S34, 0xc4ac5665L); /\* 48 \*/

/\* Round 4 \*/

a = II (a, b, c, d, x[0], S41, 0xf4292244L); /\* 49 \*/

d = II (d, a, b, c, x[7], S42, 0x432aff97L); /\* 50 \*/

c = II (c, d, a, b, x[14], S43, 0xab9423a7L); /\* 51 \*/

b = II (b, c, d, a, x[5], S44, 0xfc93a039L); /\* 52 \*/

a = II (a, b, c, d, x[12], S41, 0x655b59c3L); /\* 53 \*/

d = II (d, a, b, c, x[3], S42, 0x8f0ccc92L); /\* 54 \*/

c = II (c, d, a, b, x[10], S43, 0xffeff47dL); /\* 55 \*/

b = II (b, c, d, a, x[1], S44, 0x85845dd1L); /\* 56 \*/

a = II (a, b, c, d, x[8], S41, 0x6fa87e4fL); /\* 57 \*/

d = II (d, a, b, c, x[15], S42, 0xfe2ce6e0L); /\* 58 \*/

c = II (c, d, a, b, x[6], S43, 0xa3014314L); /\* 59 \*/

b = II (b, c, d, a, x[13], S44, 0x4e0811a1L); /\* 60 \*/

a = II (a, b, c, d, x[4], S41, 0xf7537e82L); /\* 61 \*/

d = II (d, a, b, c, x[11], S42, 0xbd3af235L); /\* 62 \*/

c = II (c, d, a, b, x[2], S43, 0x2ad7d2bbL); /\* 63 \*/

b = II (b, c, d, a, x[9], S44, 0xeb86d391L); /\* 64 \*/

state[0] += a;

state[1] += b;

state[2] += c;

state[3] += d;

}

/\*Encode把long数组按顺序拆成byte数组，因为java的long类型是64bit的，

只拆低32bit，以适应原始C实现的用途

\*/

private void Encode (byte[] output, long[] input, int len) {

int i, j;

for (i = 0, j = 0; j < len; i++, j += 4) {

output[j] = (byte)(input[i] & 0xffL);

output[j + 1] = (byte)((input[i] >>> & 0xffL);

output[j + 2] = (byte)((input[i] >>> 16) & 0xffL);

output[j + 3] = (byte)((input[i] >>> 24) & 0xffL);

}

}

/\*Decode把byte数组按顺序合成成long数组，因为java的long类型是64bit的，

只合成低32bit，高32bit清零，以适应原始C实现的用途

\*/

private void Decode (long[] output, byte[] input, int len) {

int i, j;

for (i = 0, j = 0; j < len; i++, j += 4)

output[i] = b2iu(input[j]) |

(b2iu(input[j + 1]) << |

(b2iu(input[j + 2]) << 16) |

(b2iu(input[j + 3]) << 24);

return;

}

/\*

b2iu是我写的一个把byte按照不考虑正负号的原则的＂升位＂程序，因为java没有unsigned运算

\*/

public static long b2iu(byte b) {

return b < 0 ? b & 0x7F + 128 : b;

}

/\*byteHEX()，用来把一个byte类型的数转换成十六进制的ASCII表示，

　因为java中的byte的toString无法实现这一点，我们又没有C语言中的

sprintf(outbuf,"%02X",ib)

\*/

public static String byteHEX(byte ib) {

char[] Digit = { 0,1,2,3,4,5,6,7,8,9,

A,B,C,D,E,F };

char [] ob = new char[2];

ob[0] = Digit[(ib >>> 4) & 0X0F];

ob[1] = Digit[ib & 0X0F];

String s = new String(ob);

return s;

}

}

注：

倒数第9行我觉得应该是

char[] Digit = { '0', '1', '2', '3', '4', '5', '6', '7', '8', '9', 'A',

'B', 'C', 'D', 'E', 'F' };

因为是转来的，我就不改了

10:38

浏览 (247)

评论 (0)

分类: Java

收藏Spinner

2008 - 11 - 25

缩略显示

jsp生成随机验证码图片

生成有4个随机数字和杂乱背景的图片，数字和背景颜色会改变，服务器端刷新（用history.go(-1)也会变）

原型参考ALIBABA http://china.alibaba.com/member/showimage

------------产生验证码图片的文件-----image.jsp-------------------------------------------

<%@ page contentType="image/jpeg" import="java.awt.\*,java.awt.image.\*,java.util.\*,javax.imageio.\*" %>

<%!

Color getRandColor(int fc,int bc){//给定范围获得随机颜色

Random random = new Random();

if(fc>255) fc=255;

if(bc>255) bc=255;

int r=fc+random.nextInt(bc-fc);

int g=fc+random.nextInt(bc-fc);

int b=fc+random.nextInt(bc-fc);

return new Color(r,g,b);

}

%>

<%

//设置页面不缓存

response.setHeader("Pragma","No-cache");

response.setHeader("Cache-Control","no-cache");

response.setDateHeader("Expires", 0);

// 在内存中创建图象

int width=60, height=20;

BufferedImage image = new BufferedImage(width, height, BufferedImage.TYPE\_INT\_RGB);

// 获取图形上下文

Graphics g = image.getGraphics();

//生成随机类

Random random = new Random();

// 设定背景色

g.setColor(getRandColor(200,250));

g.fillRect(0, 0, width, height);

//设定字体

g.setFont(new Font("Times New Roman",Font.PLAIN,18));

//画边框

//g.setColor(new Color());

//g.drawRect(0,0,width-1,height-1);

// 随机产生155条干扰线，使图象中的认证码不易被其它程序探测到

g.setColor(getRandColor(160,200));

for (int i=0;i<155;i++)

{

int x = random.nextInt(width);

int y = random.nextInt(height);

int xl = random.nextInt(12);

int yl = random.nextInt(12);

g.drawLine(x,y,x+xl,y+yl);

}

// 取随机产生的认证码(4位数字)

String sRand="";

for (int i=0;i<4;i++){

String rand=String.valueOf(random.nextInt(10));

sRand+=rand;

// 将认证码显示到图象中

g.setColor(new Color(20+random.nextInt(110),20+random.nextInt(110),20+random.nextInt(110)));// 调用函数出来的颜色相同，可能是因为种子太接近，所以只能直接生成

g.drawString(rand,13\*i+6,16);

}

// 将认证码存入SESSION

session.setAttribute("rand",sRand);

// 图象生效

g.dispose();

// 输出图象到页面

ImageIO.write(image, "JPEG", response.getOutputStream());

%>

---------------使用验证码图片的文件---------a.jsp------------------------------------

<%@ page contentType="text/html;charset=gb2312" %>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">

<html>

<head>

<title>认证码输入页面</title>

<meta http-equiv="Content-Type" content="text/html; charset=gb2312">

<META HTTP-EQUIV="Pragma" CONTENT="no-cache">

<META HTTP-EQUIV="Cache-Control" CONTENT="no-cache">

<META HTTP-EQUIV="Expires" CONTENT="0">

</head>

<body>

<form method=post action="check.jsp">

<table>

<tr>

<td align=left>系统产生的认证码：</td>

<td><img border=0 src="image.jsp"></td>

</tr>

<tr>

<td align=left>输入上面的认证码：</td>

<td><input type=text name=rand maxlength=4 value=""></td>

</tr>

<tr>

<td colspan=2 align=center><input type=submit value="提交检测"></td>

</tr>

</table>

</form>

</body>

</html>

-----------------验证的页面----------check.jsp

<%@ page contentType="text/html; charset=gb2312" language="java" import="java.sql.\*" errorPage="" %>

<html>

<head>

<title>认证码验证页面</title>

<meta http-equiv="Content-Type" content="text/html; charset=gb2312">

<META HTTP-EQUIV="Pragma" CONTENT="no-cache">

<META HTTP-EQUIV="Cache-Control" CONTENT="no-cache">

<META HTTP-EQUIV="Expires" CONTENT="0">

</head>

<body>

<%

String rand = (String)session.getAttribute("rand");

String input = request.getParameter("rand");

%>

系统产生的认证码为： <%= rand %><br>

您输入的认证码为： <%= input %><br>

<br>

<%

if (rand.equals(input)) {

%>

<font color=green>输入相同，认证成功！</font>

<%

} else {

%>

<font color=red>输入不同，认证失败！</font>

<%

}

%>

</body>

</html>

注：使用上述代码时会抛出一个异常：getOutputStream() has already been called for this response

tomcat5下jsp出现getOutputStream() has already been called for this response异常的原因和解决方法

在tomcat5下jsp中出现此错误一般都是在jsp中使用了输出流（如输出图片验证码，文件下载等），

没有妥善处理好的原因。

具体的原因就是

在tomcat中jsp编译成servlet之后在函数\_jspService(HttpServletRequest request, HttpServletResponse response)的最后

有一段这样的代码

finally {

if (\_jspxFactory != null) \_jspxFactory.releasePageContext(\_jspx\_page\_context);

}

这里是在释放在jsp中使用的对象，会调用response.getWriter(),因为这个方法是和

response.getOutputStream()相冲突的！所以会出现以上这个异常。

然后当然是要提出解决的办法，其实挺简单的（并不是和某些朋友说的那样--

将jsp内的所有空格和回车符号所有都删除掉），

在使用完输出流以后调用以下两行代码即可：

out.clear();

out = pageContext.pushBody();

更多0

更多 0

上一篇下图片地图有关

下一篇在Applet上显示一组图片 每1秒换下一个图片

主题推荐程序员eclipse插件apache基金会http服务器google file system

猜你在找

SurfaceViewHQL语句from后面接类名,而非表名编码格式问题camera 模块知识BuildPath引起的问题oracle数据库插入数据时异常纯CSS3打造精美按钮Android中写入读取XMLmysql删除重复数据Spring AOP 实现原理

查看评论

暂无评论

发表评论

用 户 名：

hao707822882

评论内容：

插入代码

\* 以上用户言论只代表其个人观点，不代表CSDN网站的观点或立场

快速回复 TOP

核心技术类目

全部主题 Java VPN Android iOS ERP IE10 Eclipse CRM JavaScript Ubuntu NFC WAP jQuery 数据库 BI HTML5 Spring Apache Hadoop .NET API HTML SDK IIS Fedora XML LBS Unity Splashtop UML components Windows Mobile Rails QEMU KDE Cassandra CloudStack FTC coremail OPhone CouchBase 云计算 iOS6 Rackspace Web App SpringSide Maemo Compuware 大数据 aptech Perl Tornado Ruby Hibernate ThinkPHP Spark HBase Pure Solr Angular Cloud Foundry Redis Scala Django Bootstrap

个人资料

访问我的空间

CARROTHWQ

访问：15990次

积分：568分

排名：千里之外

原创：15篇转载：186篇译文：0篇评论：4条

文章搜索

文章分类

java(2)

java GUI(3)

java3D(0)

java Web(1)

google(0)

java JDBC(0)

java +(1)

java + Mysql(2)

java cs(1)

文章存档

2014年05月(14)

2014年04月(20)

2014年03月(8)

2014年02月(4)

2013年12月(20)

展开

阅读排行

一个程序员多年的收藏(772)

Google 地图切片URL地址解析(242)

java关于使用了未经检查或者不安全的操作(232)

java swing基础与事件处理(208)

Java实现图片的放缩(205)

教你如何从Google Map爬数据(切片）(181)

java爬虫技术(179)

网络常用端口号(176)

在SWT中，如何点击一个按钮跳转到另外一个窗口？监听事件如何编写？(174)

监听树的选择事件(168)

评论排行

java登录界面(2)

Google 地图切片URL地址解析(2)

java中使用LinkedList类的实例(0)

ArrayList和数组间的相互转换(0)

java synchronized详解(0)

MySQL数据库(0)

如何使用 Jtree(0)

java JTable有什么设置方法(0)

JAVA SWING编程技术应用：JTABLE表格案例分享(0)

web(0)

推荐文章

最新评论

Google 地图切片URL地址解析

CARROTHWQ: @w\_y\_x\_2001:你是不是下同一个区域的图片的次数他多了！google每天提供的次数是有限制的...

Google 地图切片URL地址解析

阳夏居士: 楼主你好！我下载Google瓦片，构建的和浏览器访问一样的URL,用wininet方式下载瓦片失败，...

java登录界面

CARROTHWQ: @u010602808:嘿嘿！你过奖了！我也是新手上路！

java登录界面

爱你一呀: 楼主好棒，像楼主学习！！！

公司简介|招贤纳士|广告服务|银行汇款帐号|联系方式|版权声明|法律顾问|问题报告|合作伙伴|论坛反馈

网站客服 杂志客服 微博客服 webmaster@csdn.net 400-600-2320

京 ICP 证 070598 号

北京创新乐知信息技术有限公司 版权所有

江苏乐知网络技术有限公司 提供商务支持

Copyright © 1999-2014, CSDN.NET, All Rights Reserved GongshangLogo