CDOJ 基本架构文档

Fish<lyhypacm@gmail.com>

本文档中红色字部分是不在第一阶段计划之内的功能描述

目录

数据库综述

数据库框架和原来相比，基本的架构没有太大变化，有变动的如下：

代码管理

代码和提交记录不再是通过 Id 相等来对接，原来的设计出现了提交记录和代码不对应的 BUG，这个问题非常容易出现。所以我们通过外键的形式来避免这类问题的出现。

题目表相关

加了一个表，来实现题目的分类的功能。标签的管理在后台由管理员进行。
题目表中加入了表示数据组数的字段，改写了内核，每次测试单组数据，前端可以动态显示数据测到哪一组了。
此外题目加了一个难度字段，数值是从 0 到 10，表示题目的难度，这个也是由管理员来控制的。

编译语言控制

动态语言管理，用来动态添加删除编译语言，新内核将加入语言相关的扩展。

文章管理

多元化的文章类型，用一个表来表示各种各样的文章，通过配置文件管理的方式得到文章内容的框架。
比如说一个文章是帖子类型，它是纯文本形式，就可以用一个文本框来表示它。一个文章如果是视频类型，它就是用一个视频插件的形式表示它。但是它们在数据库中，是存在一个表里面的。
除了公共的字段，剩余的字段是用 JSON 组织在一起的，放在一个字段之中。
类型的描述用 XML 文件来表示，统一配置。

系统架构综述

本系统分三层。
上层是展现给用户的 Web 层，用于和用户交互，这一层面本身就是一个 MVC 架构的子系统。
中间层是 Service 层，用于做上下层的交互，从数据库中取出没有测过的提交，交给底层评测。
底层是 Core 层，是整个评测的核心，将中间层传输来的代码进行评测，最后返回给中间层。

数据库引擎

数据库选用 MySQL5 系列引擎，数据库连接池用 BoneCP 库管理数据库连接。

Web 层

介绍

用于向用户展现所有的功能，包括普通用户可以看到的 Problem Contest Status 之类的页面，还包括了管理员的管理页面。

架构

前端用 bootstrap 开源框架显示页面，显示内容以 js 控制为主。
后台用 Spring3 + Hibernate4 + Struts2 的框架，经典 MVC 模型。

基本开发思路

Condition

用于将一系列的前端信息转化为 Criteria 对象。
对于 Condition 的实例可以参见 JavaDoc，主要是用注释的方式来实现方便的条件处理。

DTO

数据传输对象(Data Transform Object)，用来做前端和后台的数据交换，最后用 DTO 生成 Entity，这个生成的方法是用反射机制自动完成的，不用一个 DTO 写一个方法，对于 Override 的方法，只要实现不能反射的字段即可。

View

用来提供给前端显示的视图类，将前端要显示的内容放在这个类中。构造函数是要以 Entity 为参数的，不能直接构造。对于题目这样的列表和详细信息内容相差过大的情况，可以写多个视图类来解决这个问题。

全局拦截器

实现一个全局拦截器，在 Action 中实现两个方法，一个是 action 执行前运行的，一个是 action 执行后运行的。然后用全局拦截器通过反射机制调用这两个方法即可。

重载 XML 解析器

自己实现一个 XML 解析的类，用来方便自己对 XML 文档进行解析。

Service 层

介绍

对数据库中没有评测过的提交记录进行管理，将其交予 Core 层进行评测。

架构 1：本机测试

使用原内核，一个主线程，多个子线程，主线程用于管理全局的评测队列，子线程用于评测单个提交代码。
评测的方法是在子线程中 fork 出内核进程，测完后 fetch 到其输出，根据输出得到评测的结果。

架构 2：分布式

使用新内核，一个主线程，多个子线程，各个线程的作用和架构 1 相同。
评测的方式是用网络通讯，将代码经过网络传输给内核，内核评测完后将结果返回回来 ，然后写入数据库。

Core 层

介绍

将代码进行编译运行，判断是否得到正确的答案，此外还要考虑内存、使用的时间等的计算，将结果返回给 Service 层。

架构 1：本机测试

使用原内核，评测过程中，编译过程是两个进程，子进程编译，主进程计算编译的时间。运

行也是两个进程，子进程运行，父进程用来做到监控的目的。将结果以 stdout 的方式输出。

架构 2：分布式

新内核，和原内核相比，区别在于，用多进程加网络通信的方式，对于新的请求就新建进程来执行架构 1 中主进程的流程，主进程继续监听端口。
数据的传输用网络的形式而不是 stdin、stdout。