任务调度(TaskManage)实现分析

# 一 功能概要

在现实的业务中，经常会用到一些自定义的服务，当服务过多时，管理起来就比较难，而且对服务的统一监控就更是不可能的了。任务调度平台可以对服务进行统一管理与监控，保证了服务的可靠性，这些服务实现统一的接口，开发和部署都比较方便。

# 二 模块分析

## 数据库

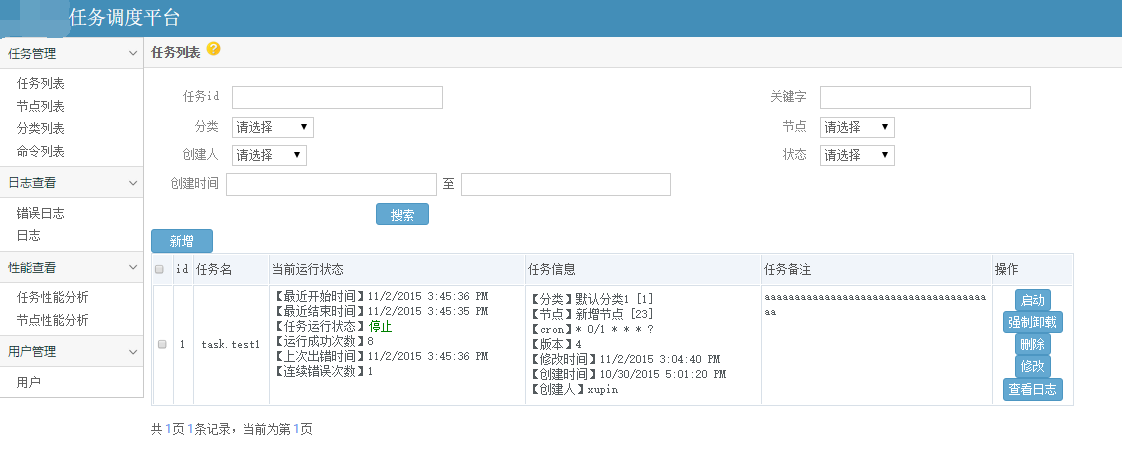
|  |  |  |
| --- | --- | --- |
| 表名 | 简要 | 说明 |
| tb\_category | 任务分类表 | 任务属于一个分类 |
| tb\_command | 任务命令表 | 对任务的命令保存在数据库中，节点会从该表中读取没有执行的命令并标记执行状态 |
| tb\_error | 错误日志表 |  |
| tb\_log | 一般日志表 | 包括类型：  常用日志（任务内写的普通日志），  系统日志，  系统错误日志，  常用错误日志（任务内写的错误日志） |
| tb\_node | 节点表 | 该任务平台的所有节点（任务运行在节点中） |
| tb\_performance | 节点和任务性能记录表 |  |
| tb\_task | 任务表 | 该平台的所有任务 |
| tb\_tempdata | 任务数据库中的临时数据表 | 任务可以将临时数据保存在数据库，当然也可以保存在本地 |
| tb\_user | 任务调度web用户表 | 登录 |
| tb\_version | 任务版本库 | 任务的起始版本为1，以后每添加一个版本就会加1，也可以在原版本上修改。 |

## 任务调度Web

Web很简单，涉及到的是对Task数据库的修改和对应的显示，能显示节点的状态和任务的状态，节点一段时间没有心跳则会为停止状态，任务的状态为数据库里保存的状态，可能不为实际状态。

需要说明的是web不直接对任务进行操作，做的仅是添加操作命令到数据库，由节点读取命令执行，如果节点停止，这个命令也不会执行。可在命令列表页面查看执行状态。

网站截图：



## 节点（Windows服务）

节点是任务调度的核心，每个节点为一个服务进程，

一个节点服务按命令来调度多个任务，命令来源于web，任务对应一个服务，任务定义了服务的各类信息，包括服务的程序启动入口文件名。调度平台可以调度多个节点，一个节点服务管理多个服务在本机上运行。（避免程序冲突，几个os不要部署多个相同的服务）。

节点通过数据库访问和web api与系统进行通信。

节点内主要包含：

* 多个监控线程（见【五】）
* 一个运行任务池，每个任务在各自独立的进程中运行，内部包含一个任务执行计划
* 命令读取执行线程

服务在启动时会从本地和任务web上获取配置信息，这些信息包括：

* 节点编号（本地或web获取）
* 数据库连接字符串（web获取）
* 任务web的地址（本地获取）

节点的具体工作内容，请看【四】。

## 具体任务

？？？

任务是一个服务程序，主要的运行类 需要继承IHostServices接口，规范服务程序的简单示例（此处加链接）

任务会在任务调度web中会定制一个执行计划，服务通过Quartz来管理任务的定时执行，服务会定义一个任务类并作为参数注册到Quartz中，同时将执行计划作为参数注册到Quartz,这样就实现了定时调度任务类。执行计划作为参数使用Corn表达式，Corn将在下节说明。

在从web新建任务的时候，会有一个Json配置，这个配置在运行时在当前对象的AppConfig中，该值为一个字典。

任务在执行时可以设置临时数据，保存和读取有两个方法：

* 使用本地文件
* 使用数据库

注：

1 web定制的Corn 数据要传递给服务，服务加载这个Corn ,所以需要统一定义服务加载Corn的规范：可以加Corn 填入到web.config 中的指定条目中，在启动服务程序

2 定制的json 需要能够加载到服务程序中，可以统一定义服务加入读取aa.json 格式文件，来载入web 中定义的json

3 节点服务开启关闭任务的服务，执行计划会定时实例化服务中的任务类并执行，当中的并发，同步方式 由服务内部自己定义

# 三 任务调度Corn表达式介绍

Quartz使用类似于Linux下的Cron表达式定义时间规则，Cron表达式由6或7个由空格分隔的时间字段组成，如表所示：

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 是否必须 | 允许值 | 特殊字符 |
| 秒 | 是 | 0-59 | , - \* / |
| 分钟 | 是 | 0-59 | , - \* / |
| 小时 | 是 | 0-23 | , - \* / |
| 日期 | 是 | 1-31 | , - \* ? / L W C |
| 月份 | 是 | 1-12或者 JAN-DEC | , - \* / |
| 星期 | 是 | 1-7或SUN-SAT | , - \* ? / L C # |
| 年 | 否 | 1970-2099 | , - \* / |

以上为Corn表达式的结构，在实际使用中，可能用自定义的corn更简单实用。

自定义表达式有: Simple, RunOnce 几种Corn表达式：

**Simple表达式**

意义：一个简单的设置任务运行频率的自定义表达式。

格式:[Simple,运行间隔(单位:秒,空表示1秒【默认】),总共运行次数(单位:次,空表示int.MaxValue【默认】),开始时间(空表示立即开始【默认】),结束时间(空表示不限制)]

示例:[Simple,1,1,2012-01-01 17:25,2016-01-01 17:25]

**RunOnce表达式**

意义：表示任务仅运行一次。等同于[Simple,,1,,]

格式:[RunOnce]

示例:[RunOnce]

在任务调度web上有表达式生成的工具。

# 四 实现详解

## 整体流程

节点配置：

添加节点windows服务，配置正常节点和网站的地址

## 任务命令执行

对节点内任务的启动、停止等都是通过命令实现的，具体的这些命令的执行请看下小节。节点内有一个监控命令的线程，每隔一小段时间会从数据库里读取命令进行执行，执行完成会更新命令的执行状态。

下图能理解执行过程：



## 任务命令集（当前支持的命令）

* **开始命令**

1、 按任务 服务名称在任务池查找，如果有则查看进程是否在运行，正常就返回，否则

2、 如果没有 从数据库取出这个任务的相关信息，并取这个任务最新版本的版本信息和文件；

3、保存该文件到节点的任务dll缓存目录

4、解压文件到任务ID对应的目录

5、复制节点的共享程序集到这个任务ID目录，这里要注意，如果共享程序集在该任务dll中有修改，需要更新共享程序的对应程序集。

6、根据任务配置。

7、初始化这个实例的参数，如：连接字符串，任务的AppConfig等，

8、将任务添加到节点的任务池，开启这个任务，并更新这个任务状态为运行。

任务池：包括任务的名称，id，任务对应服务的进程的名片，名片包括进程id 等信息

9、 如果这些步骤出错，中断任务服务并从任务池中卸载任务

如果这些步骤出错，将从池中卸载任务。

* **停止命令**

1、从节点的任务池中查找这个任务，如果没找到，返回，否则

2、找到任务所在进程，关闭进程，如果有抛出异常超时异常，记录异常并返回失败，否则

3、从任务池中删除任务

4、更新任务状态为停止

# 五 节点内部监控任务介绍

节点基本任务：

* **NodeHeartBeatMonitor**:节点的心跳监控者，每小段时间会运行一次，来更改库中当前节点的状态，来说明节点的运行是否正常
* **PingTaskWebMonitor**: （暂时不加）每段时间设用任务平台的api，确定管理网站运行正常
* **TaskPerformanceMonitor**:（待解读）节点中每个任务运行情况，如cpu、内存使用情况，监控节点性能状态。

*TaskRecoverMonitor与TaskStopMonitor中有一个细节，为了保证任务状态修改的时间差问题，只会使用上次记录的状态与当前状态的交集来得出结果并记录。*

# *六 任务调度Web功能用例*

## *6.1概述*

## *6.2 新增任务*

## *6.3 更新任务*

## *6.4 启动*

*判断之前任务是否在运行，已运行则记录已经在运行的日志， 返回成功；未运行则判断任务dll目录中有没有当前任务程序集，没有则先重新下载任务程序集，然后开启任务程序集。*

## *6.5 关闭*

## *6.6 强制卸载*

*先检查任务的状态，如果运行中则先停止任务进程，再卸载的任务程序集。重新启动后会按最新的任务程序版本运行*