工作者机制（Worker Mechanism）

Vapula框架使用工作者机制优化任务的运行调度过程。

任务（Task）由用户通过运行时驱动（Driver）构造。

每个任务具有一个虚拟栈（Stack）。

任务和栈的信息依照库（Library）的指定方法（Method）的定义。

任务本身具有启动（Start）、暂停（Pause）、恢复（Resume）、停止（Stop）、重启（Restart）5个控制方法，用户可以调用这些控制方法操作任务。

工作者（Worker）是所有任务的实际执行者。

这些方法内部会完善地实现任务与工作者的交互过程。

工作者具有与任务同名的控制方法。

同时，工作者提供上线（Online）和下线（Offline）两种总控方法。

工作者构造后，本身不具备执行能力。

工作者内部具有1个任务队列和2个线程队列，线程队列区分为前线队列（Head）和后备队列（Back）。

调用工作者上线方法后，工作者会计算前线容量，并构造总数等于前线容量的线程，

将这些线程分别于每个逻辑CPU绑定，再将这些线程推入前线队列，并设置这些线程为前线线程。

此时，后备队列仍为空。

调用工作者下线方法后，工作者会统计当前任务队列，终止所有执行线程并销毁，然后清空任务队列。

任务的执行分为两种，一种是即时执行，一种是排队执行。

**启动**

启动具有时控参数。

时控为0表示用户要求即时执行任务，工作者创建一个线程并安排当前任务，

这样会使工作者立刻调度一个任务启动。

时控不为0表示用户不要求即时执行任务，工作者会将任务推入任务队列。

//当超过等待时间后，工作者仍然会立刻调度任务启动。

即时执行。工作者创建一个新的线程，设置为后备线程，并安排任务并将线程推入后备队列。

排队执行，当前线线程没有任务执行后，会向工作者请求任务，工作者锁定任务队列，取队首任务，安排该任务给前线线程，并解锁任务队列。

**停止**

停止方法具有时控参数。

时控为0，表示用户要求立即强行停止任务。

工作者会终止线程。将线程移出前线并设置为濒死线程。然后工作者创建新的前线线程推入前线队列。

时控不为0，表示用户等待任务主动停止。

任务本身发布控制码。工作者将线程移出前线并设置为濒死线程。

若任务没有主动退出，会再次调用时控为0的停止。

**暂停**

暂停方法具有时控参数。

时控为0，表示用户要求立即挂起任务。

工作者会挂起线程。将线程置为后备，并推入后备队列。

然后获取一个后备队列的空闲线程置入前线。

失败则创建新前线。

**恢复**

调用恢复方法后，工作者会根据任务查询线程。

并检测线程是否是挂起的，如果是则恢复。

**重启**

重启方法具有时控参数。

时控为0，表示用户要求立即重启任务。

工作者会调用停止，时控为0。等待线程结束，然后创建新线程，设为后备。

时控不为0，任务本身会发布控制码。

工作者将等待任务重启，若超时则调用停止，时控为0。

然后将任务推入队列。

注意

当任务没有执行完成时，务必不要将该任务再次推入队列。

因为工作者机制会导致这个任务具有被多次并行调度的可能，

但是虚拟栈只有一个，并行运行一个任务会导致数据破坏。

无论用户使用什么操作系统，同时可以并行调度的实际任务数量是有限的，限制即CPU逻辑核心数量。无限制的多线程调度通常是由操作系统模拟的。

所以应当保证一个软件系统只并行调度等于逻辑核心数量的任务。

排队执行默认即采用

前线容量等于当前Vapula执行环境的逻辑CPU数量与调度负载系数的乘积。

调度负载系数是大于等于1的正整数。