**GridSim类**

该类是整个仿真的初始化、运行和停止过程中起主要作用的类。必须要在创建任何其他用户级GridSim实体前初始化，来建立仿真环境。调用类中的startGridSimulation方法来开始网格仿真。所有资源和用户实体必须在调用初始化和开始仿真的方法之间被实例化！

貌似继承了GridSim的类就是用户实体类，在main方法中都会分配一个该类变量，作为网格用户对象。

**public** **static** **void** init(**int** numUser, Calendar cal, **boolean** traceFlag,

String[] excludeFromFile, String[] excludeFromProcessing,

String reportWriterName){}

注释：用于初始化GridSim参数。该方法必须在创建任何实体前调用！在该方法中，将会创建以下的网格实体：GridSimRandom，GridStatistics，GridInformationService和GridSimShutdown。Calendar对象必须指定使用Calendar.getInstance()来指示模拟开始时间。模拟时间在处理提前预约功能时非常重要。（自5.0版本以来，就不再初始化GridSimRandom对象了）。

参数：numUser 创建的网格用户实体总数。该参数表示GridSimShutdown对象应该在向其他实体发出终止信号之前等待用户实体的END\_OF\_SIMULATION信号

cal 模拟的开始时间。若该值为null，则时间将会从Calendar.getInstance获得

traceFlag 如果需要记录GridSim的跟踪信息，则将该值设为true

excludeFromFile 一个String数组，存储要被移除的统计数据的文件列表

excludeFromProcessing 一个String数组，存储被排除写入文件的程序处理列表

reportWriterName 一个ReportWriter实体名。该实体可以在gridbroker包中找到

**public static void** startGridSimulation() **throws** NullPointerException{}

注释：开始GridSim仿真的执行。它等待所以实体完成执行，例如，所有实体线程都在body方法退出后到达non-RUNNABLE状态。然后，它会杀死所有实体的线程。注意，该方法应该在创建并添加了所有实体，并且他们的端口都互联之后调用

异常：在初始化前创建实体或实体名为空时，抛出空指针异常

**protected** Gridlet gridletReceive(){}

注释：获取网格任务，它是属于当前等待在该实体的缓存队列的第一个事件。如果没有事件，则无限期的等待一个事件到来

返回值：一个网格任务实体，如果发生了错误则返回null

**public static int** getEntityId(String entityName){}

注释：给定实体名称获取ID。实体名不能为空！正常的返回结果（即ID值）需>=-1，若等于-1，则实体ID没有找到

参数：entityName 一个实体名

**public void** gridSimHold(**double** duration){}

注释：使实体保持duration个单位的仿真时间

参数：duration 需要保持的总时间

**public static** LinkedList getGridResourceList(){}

注释：向GIS实体发送请求以获取所有的网格资源列表（包括支持提前预约的资源）

返回值：一个链表，存储内容为网格资源ID（Integer对象形式），如果GIS实体没有创建，则返回null

**protected** Object receiveEventObject(){}

注释：获取一个对象，它是属于等待在该实体当前缓存队列的第一个事件的。如果没有事件，则无限期的等待一个事件的到来

返回值：一个事件对象

说明：该方法通常是send请求之后的一个接收对象的方法。该方法返回值类型为Object，所以可以接收任何类型的对象。比如send请求是希望获取资源的属性信息，则该方法就可以返回一个ResourceCharacteristics类型的对象，不过接收之后别忘了强制类型转换！

**protected boolean** gridletSubmit(Gridlet gl, **int** resID){}

注释：无延迟的发送一个网格任务到目的网格资源。默认情况下没有指示方法执行成功的确认（ACK）。返回true则表示该任务已被提交到资源，否则返回false。

网格任务提交失败，可能是由以下一个或多个原因造成的：

如果参数中的ACK状态被设置为false；

如果一个资源ID不存在；

如果一个任务ID不存在；

如果网格资源的userID不存在；

如果延迟时间为负数；

如果网格资源对象为空；

如果一个网格任务对象之前已经完成执行。

send方法也能实现提交任务的功能，只要将标签设置为GRIDLET\_SUBMIT即可！

参数：gl 一个将要被发送的网格任务对象

resID 一个唯一的资源ID

说明：终极方法是下面这个，参数最全，其他方法都是调用这个方法

**protected** **boolean** gridletSubmit(Gridlet gl, **int** resourceID, **double** delay, **boolean** ack, **int** netServiceLevel)

参数：delay 延迟的时间，若立即执行，则该参数为0.0

ack 确认状态。设置为true则想要知道该方法的结果，反之或不关心结果，则设置为false

netServiceLevel 可以设置该参数以便支持差异化服务的网络实体可以提供服务

**GridSimCore类**

**protected** **void** send(String entityName, **double** delay, **int** gridSimTag, Object data){}

注释：发送一个事件或消息到另一个实体。如果网络带宽在方阵中扮演很重要的角色，建议使用带有一个输出端口参数的send()方法。不过，实体必须有网络实体，比如Input和Output端口（创建实体期间通过通过给定传输速率或带宽速度来指定）。

参数：entityName 目标实体名

delay 从当前仿真时间到事件应该被提交的时间的长度（若为负数则将被改为0）

gridSimTag 用户定义的代表事件或消息类型的数字（有常量）

data 一个数据的引用，和事件一起被发送（Object类型，什么都能传）

特殊：四个参数的方法可以用于发送网格任务。一共有6个重载的send方法，其中有一个方法比这个send方法少一个data参数，也在例子中出现了，它可以用于发送模拟结束的信息（GridSimTags.END\_OF\_SIMULATION）

**protected void** terminateIOEntities()

注释：它终止实体的管理网络通信的信道。可以被显式调用来关闭网络通信信道。建议素有继承自GridSim类的实体，显式调用该方法来终止构造方法GridSim(String, double)中创建的Input和Output

**eduni.simjava.Sim\_entity类**

这是一个超级父类！GridSim extends GridSimCore，而GridSimCore extends Sim\_entity。body()方法是该类中的方法，几乎所有需要进行仿真的类都要重写该方法。这是解决GridSim实体间通信最核心的方法。

body()方法应该是个回调方法，调用startGridSimulation方法后，进入body方法，执行仿真内容。所以一般情况下，body方法最后都要有各种关闭。

**Machine类**

**public** Machine(**int** id, **int** numPE, **int** ratingPE) {}

注释：分配一个新的Machine对象（构造方法，参数均应大于0）

参数：id 机器的ID

numPE 该Machine的PE总数

ratingPE 处理器的计算能力，机器中一个处理器的MIPS（每秒处理的百万指令数）速率（一般是看选择的机器型号规定该值，377比较常用）

**ResourceCharacteristics类**

**public** ResourceCharacteristics(String architecture, String OS,

MachineList machineList, **int** allocationPolicy,

**double** timeZone, **double** costPerSec){}

注释：分配一个新的ResourceCharacteristics对象。如果参数传入的时区是无效的，则默认情况下时区设置为GMT+0

参数：architecture 资源的体系结构

OS 使用的操作系统

machineList 一个资源里的机器列表

allocationPolicy 资源的分配策略

timeZone 拥有这些预留资源的用户所在当地时区（范围是GMT-12到GMT+13）

costPerSec 使用这些资源的每秒代价

**public int** getResourceID(){}

**public** String getResourceName(){}

**public** String getResourceArch(){}

**public** String getResourceOS(){}

**public** MachineList getMachineList(){}

**public** Machine getMachineWithFreePE(){}

**public** Machine getMachineWithFreePE(**int** numPE) {}

**public int** getResourceAllocationPolicy(){}

**public double** getResourceTimeZone(){}

**public int** getMIPSRatingOfOnePE(){}

**public int** getMIPSRatingOfOnePE(**int** id, **int** peID) {}

**public int** getMIPSRating(){}

**public double** getCPUTime(**double** gridletLength, **double** load) {}

**public int** getNumPE(){}

**public int** getNumFreePE(){}

**public int** getNumBusyPE(){}

**public double** getCostPerSec(){}

**public double** getCostPerMI(){}

**public int** getNumMachines(){}

**public int** getNumFailedMachines(){}

注释：以上方法都是资源属性类中返回资源的各种属性的方法，返回的内容是该类私有成员变量相关的内容（私有，其他类访问不到，只能通过这些get方法）

**GridResource类**

**public** GridResource(String name, **double** baud\_rate, **long** seed,

ResourceCharacteristics resource, **double** peakLoad,

**double** offPeakLoad, **double** relativeHolidayLoad,

LinkedList weekends, LinkedList holidays) **throws** Exception {}

注释：分配一个新的网格资源对象。如果想自定义不同类型的网格资源对象，需要使用另一个构造方法（GridResource(String, double, ResourceCharacteristics, ResourceCalendar, AllocPolicy)），并重写processOtherEvent(Sim\_event)方法

参数：name 与该实体关联的名字（因为simjava包中的Sim\_entity类需要该参数）

baud\_rate 网络连接或带宽速度

seed 初始的seed

resource 一个ResourceCharacteristics对象

peakLoad 高峰时段的负载

offPeakLoad 低峰时段的负载

relativeHolidayLoad 假期时间的负载

weekends 一个包含周末的链表

holidays 一个包含公共假期的链表

**Gridlet类**

**public** Gridlet(**int** gridletID, **double** gridletLength,

**long** gridletFileSize, **long** gridletOutputSize){}

注释：分配一个新的网格任务对象。网格任务的长度，输入输出文件大小应该大于等于1。默认情况下，这个构造方法会设置该对象的历史。（另有一个构造方法可以设置是否记录对象历史，就是在此构造器基础上加了一个布尔类型的参数）

参数：gridletID 该网格任务的唯一ID

gridletLength 网格任务在网格资源中被执行所需的指令数（MI），任务的长度

gridletFileSize 该任务提交到网格资源之前的文件大小（以字节为单位）

gridletOutputSize该任务在网格资源中处理结束之后的文件大小（字节）

**public void** setGridletStatus(**int** newStatus) **throws** Exception{}

注释：设置该网格任务的状态码（Gridlet类中有很多相关的常量表示状态码），若输入的状态码是无效的，则会抛异常（范围是0到8，闭区间）

参数：newStatus 网格任务状态码（有常量，如：SUCCESS，FAILED，CANCELED等）

**public void** setUserID(**int** id){}

注释：设置一个网格任务的使用者或拥有者的ID。设置用户ID是非常重要的，因为不设置的话，网格任务将无法在网格资源中被执行（很简单的方法，但是很必要！）

参数：id 用户的ID

**GridletList类**

该类无带参数的构造方法，只有一个默认的无参构造方法。GridletList继承LinkedList<Gridlet>。

该类有一个sort()方法，实现按任务长度排序的功能；一个indexOf(int gridletId, int userId)方法，用来寻找指定的网格任务（根据用户和任务ID）在集合中的下标值；一个get(int gridletId, int userId)方法，返回给定的任务。

另外，该类含有一个私有的内部类，类名为OrderLength，实现了Comparator<Gridlet>接口，实现了接口的compare(Gridlet a, Gridlet b)方法，使两个任务按长度比较。sort()方法用到了该内部类。

**GridSimStandardPE类**

随机创建网格任务时用到了该类。

为一个标准PE（Processing Element）定义MIPS率（Million Instructions Per Second），或让用户自定义标准PE的MIPS或SPEC（Standard Performance Evaluation Corporation）率。该值可以应用于为GridSim资源创建带有相关MIPS或SPEC率的PE。

该类可以用于定义网格任务关于标准PE的性能资源或执行时间。

该类中所有方法和属性都是Static的，所以将构造方法声明为私有的。类中有一个静态初始化块，给属性MIPSRating设置初始值为100。我们可以通过setRating(int rating)方法修改该属性值。

**GridSimRandom类**

**public** **static** **double** real(**double** value, **double** lessFactor, **double** moreFactor,

**double** randDouble) **throws** IllegalArgumentException{}

注释：将预测值或估计值映射到实际值，范围是(1-lessFactor)\*value到(1+moreFactor)\*value

使用的公式是value\*(1-lessFactor+(lessFactor+moreFactor)\*randDouble)

lessFactor>=0.0，moreFactor<=1.0

参数：value 估计值

lessFactor 标准值到最小值的因数

moreFactor 标准值到最大值的因数

randDouble 0.0到1.0之间均匀分布的一个double值

非法参数异常 以上四个参数不是0或正数时，抛异常。另外，moreFactor和randomDouble的最大值是1.0

**public static int** intSample(**int** range) **throws** IllegalArgumentException{}

注释：从java.util.Random类获取随机的整型值，方法内部就是用一个Random对象产生一个range范围内的整型值。若参数range为负数，则抛出IllegalArgumentException

参数：range 要返回的随机值的范围（范围要大于等于0）

返回值：一个伪随机数，[0, range)内均匀分布的整型数值

**public static double** doubleSample(){}

注释：从java.util.Random类获取随机的double值，方法内部就是调用了Random类的nextDouble方法（看GridSimRandom代码可以发现，有一个Random类的成员变量，初始化块和构造方法分别给他创建对象和设置seed值，与Random的用法相同）

返回值：返回下一个伪随机数，该数是由该随机数生成序列产生的0.0到1.0之间均匀分布的double值