

Gradle学习系列之八——构建多个Project

在[本系列的上篇文章](#)中，我们讲到了Gradle的依赖管理，在本篇文章中，我们将讲到如何构建多个Project。

请通过以下方式下载本系列文章的Github示例代码：

```
git clone https://github.com/davenkin/gradle-learning.git
```

Gradle为每个build.gradle都会创建一个相应的Project领域对象，在编写Gradle脚本时，我们实际上是在操作诸如Project这样的Gradle领域对象。在多Project的项目中，我们会操作多个Project领域对象。Gradle提供了强大的多Project构建支持。

要创建多Project的Gradle项目，我们首先需要在根（Root）Project中加入名为settings.gradle的配置文件，该文件应该包含各个子Project的名称。比如，我们有一个根Project名为root-project，它包含有两个子Project，名字分别为sub-project1和sub-project2，此时对应的文件目录结构如下：

```
root-project/  
  sub-project1/  
    build.gradle  
  sub-project2/  
    build.gradle  
  build.gradle
```

我翻译的书：



我读过的书：



settings.gradle



root-project本身也有自己的build.gradle文件，同时它还拥有settings.gradle文件位于和build.gradle相同的目录下。此外，两个子Project也拥有他们自己的build.gradle文件。

要将sub-project1和sub-project2加入到root-project的子Project中，我们需要在settings.gradle中加入：

```
include 'sub-project1', 'sub-project2'
```

接下来，我们来定义一个Task用于显示每个Project各自的名称。我们可以在每个build.gradle进行定义，但是这却是一种比较笨的方法，此时我们也完全没有享受到Gradle的多Project构建功能所带来的好处。在Gradle中，我们可以通过根Project的allprojects()方法将配置一次性地应用于所有的Project，当然也包括定义Task。比如，在root-project的build.gradle中，我们可以做以下定义：



```
allprojects {  
    apply plugin: 'idea'  
  
    task allTask << {  
        println project.name  
    }  
}
```



以上Gradle脚本将闭包中的代码应用在所有的Project中，包括root-project本身。我们首先将应用了idea Plugin用于生成IntelliJ工程，其次我们定义了名为allTask的Task，该Task应用于每个Project，作用是输出各个Project的名称。执行“gradle allTask”，命令行输出如下：

```
:allTask
```

昵称：无知者云
园龄：4年8个月
粉丝：88
关注：3
[+加关注](#)

我的标签

社会(17)

诗歌(14)

java(13)

gradle(10)

j2ee(9)

java ee(9)

事务(8)

transaction(5)

Ruby(3)

数据库(3)

[更多](#)

随笔档案(65)

2014年7月 (1)

2014年3月 (1)

2014年2月 (1)

2013年11月 (10)

2013年6月 (2)

2013年4月 (1)

```
root-project
:sub-project1:allTask
sub-project1
:sub-project2:allTask
sub-project2
```

我们看到，该allTask对于每个Project都执行了一次，在执行时输出了当前Project的名称。

除了allprojects()之外，Project还提供了subprojects()方法用于配置所有的子Project（不包含根Project）。比如，我们可以定义Task来只输出各个子Project的名字：

```
subprojects {
    task subTask << {
        println project.name
    }
}
```

执行“gradle subTask”，命令行输出如下：

```
:sub-project1:subTask
sub-project1
:sub-project2:subTask
sub-project2
```

此时的输出中不再包含root-project的名字。

上文中已经提到，在Gradle脚本中，我们实际上是在操作一些领域对象，因此我们可以将groovy的所有语言特性用在Gradle的领域对象上，比如我们可以对Project进行过滤：

```
configure(allprojects.findAll { it.name.startsWith('sub') }) {
    subTask << {
        println 'this is a sub project'
    }
}
```

2013年3月 (1)

2013年2月 (12)

2013年1月 (1)

2012年12月 (2)

2012年11月 (1)

2012年10月 (1)

2012年8月 (1)

2012年7月 (2)

2012年6月 (2)

2012年5月 (3)

2012年4月 (10)

2012年3月 (4)

2012年2月 (7)

2011年12月 (1)

2011年10月 (1)

最新评论

1. Re:毕业两年返校随想

楼主在西电？

--wxyjuly

2. Re:Gradle学习系列之三——读懂Gradle语法

mark

--headchen

3. Re:Gradle学习系列之十一——自定义Plugin (未完待续)

在上面的代码中，我们先找到所有Project中名字以“sub”开头的Project，然后再对这些Project进行配置，在配置中，我们向这些Project的subTask中加入了一条额外的打印语句。

此时如果再执行“gradle subTask”，命令行输出如下：

```
:sub-project1:subTask
sub-project1
this is a sub project
:sub-project2:subTask
sub-project2
this is a sub project
```

到此为止，我们所有的Task定义工作都是在root-project中进行的，而sub-project1和sub-project2中的build.gradle文件依然什么都没有。事实上，我们可以将所有对子Project的配置均放在根Project中进行。在上面的例子中，我们通过allprojects()和subprojects()将所有的子Project都包含在了配置之内，其实我们还可以对单个Project进行单独配置。比如，在root-project的build.gradle中加入：

```
project(':sub-project1') {
    task forProject1 << {
        println 'for project 1'
    }
}
```

以上脚本向sub-project1中加入了名为forProject1的Task，在执行“gradle forProject1”时，终端输出如下：

```
:sub-project1:forProject1
for project 1
```

这里有一个问题：我们是在root-project下执行的命令，因此照理说Gradle会认为forProject1是定义在所有的Project上，而此时只有sub-project1才拥有该Task，Gradle应该抛出异常指示在root-project和sub-project2上找不到该Task才对，为什么它还是执行成功了呢？原因在于：只有当一个Task没有在任何

gin（本系列完）
DateAndTimePluginExtension 这个里面的timeFormat 是不是没有用，为什么样定义呢？
--灵感仰
4. Re:Gradle学习系列之一——Gradle快速入门
您好，图中这个是否应该是taskA？
--Mr.XuFeng
5. Re:Gradle学习系列之十一——自定义Plugin（本系列完）
博主你真是太棒了！循序渐进通俗易懂。我这两天看gradle看地稀里糊涂的，终于在你这弄明白了。谢谢你！
--无意义

阅读排行榜
1. Gradle学习系列之一——Gradle快速入门(32411)
2. Gradle学习系列之二——创建Task的多种方法(10429)
3. Gradle学习系列之七——依赖管理(9418)
4. 重温大师经典：Martin Fowler的持续集成(9321)
5. Gradle学习系列之三——读懂Gradle语法(9247)

评论排行榜
1. 重温大师经典：Martin Fowler的持续集

Project中定义时，Gradle才会将其当做异常。否则，Gradle会在所有拥有该Task的Project上执行该Task。

一旦有了多个Project，他们之间便会存在着依赖关系。Gradle的Project之间的依赖关系是基于Task的，而不是整个Project的。

现在，让我们来看一个Project依赖的例子。比如sub-project1中有taskA和taskB，taskA依赖于taskB：



```
task taskA << {
    println 'this is taskA from project 1'
}

task taskB << {
    println 'this is taskB from project 1'
}


taskA.dependsOn taskB
```



在执行“gradle taskA”时，终端输出：

```
:sub-project1:taskB
this is taskB from project 2
:sub-project1:taskA
this is taskA from project 1
```

这个很容易理解，两个Task都是属于sub-project1的。但是，让我们再向其中加入一些复杂性。我们在sub-project2中定义taskC和taskD，然后使taskA再依赖于taskC，又使taskB依赖于taskD：



```
//sub-project1:
taskA.dependsOn ':sub-project2:taskC'
```

成(12)
2. 5个小时写一个扑克牌游戏——金钩钓鱼(10)
3. Gradle学习系列之二——创建Task的多种方法(6)
4. 用Spring MVC3+Ant+Jenkins+SVN+Tomcat做一个持续集成例子(5)
5. Gradle学习系列之九——自定义Task类型(4)

推荐排行榜
1. Java加载资源文件的两种方法(8)
2. Gradle学习系列之一——Gradle快速入门(7)
3. 重温大师经典：Martin Fowler的持续集成(6)
4. Java事务之一——Java事务的基本问题(5)
5. 用Spring MVC3+Ant+Jenkins+SVN+Tomcat做一个持续集成例子(4)

```
taskB.dependsOn ':sub-project2:taskD'

//sub-project2:
task taskC << {
    println 'this is taskC from project 2'
}

task taskD << {
    println 'this is taskD from project 2'
}
```



此时再执行“gradle taskA”，终端输出如下：

```
:sub-project2:taskD
this is taskD from project 2
:sub-project1:taskB
this is taskB from project 1
:sub-project2:taskC
this is taskC from project 2
:sub-project1:taskA
this is taskA from project 1
```



分析一下：taskA依赖于taskB，而taskB又依赖于taskD，所以sub-project1的taskD首先得到了执行，然后再执行sub-project1的taskB。之后，又由于taskA依赖于taskC，故Gradle再次转向sub-project1执行taskC，最后才执行taskA。

在[下一篇文章](#)中，我们将讲到如何自定义Task类型。

标签: [gradle](#)

好文要顶

关注我

收藏该文





[无知者云](#)
[关注 - 3](#)
[粉丝 - 88](#)
[+加关注](#)

2

推荐

0


反对

(请您对文章做出评价)

« 上一篇 : [Gradle学习系列之七——依赖管理](#)
» 下一篇 : [Gradle学习系列之九——自定义Task类型](#)

posted @ 2013-11-15 09:28 无知者云 阅读(6080) 评论(0) 编辑 收藏

[刷新评论](#) [刷新页面](#) [返回顶部](#)

 注册用户登录后才能发表评论，请 [登录](#) 或 [注册](#)，[访问网站首页](#)。

【推荐】50万行VC++源码: 大型组态工控、电力仿真CAD与GIS源码库

【推荐】融云即时通讯云 - 豆果美食、Faceu等亿级APP都在用



最新IT新闻:

- 特斯拉正在测试8.0版固件：界面大改版+新Autopilot功能
- 苹果联手NASA推出音乐短片庆祝“朱诺号”太空任务
- 印度售价24元的智能手机预订量超7000万部 官方称首发供货20万
- 冤家路窄 甲骨文被判违反协议需向惠普赔偿30亿美元
- Facebook和App命里犯冲？新闻聚合应用Paper将于本月底关闭
- » 更多新闻...

最新知识库文章:

- 编程同写作，写代码只是在码字
- 遇见程序员男友
- 设计师的视觉设计五项修炼
- 我听到过的最精彩的一个软件纠错故事
- 如何避免软件工程中最昂贵错误的发生
- » 更多知识库文章...

Copyright ©2016 无知者云