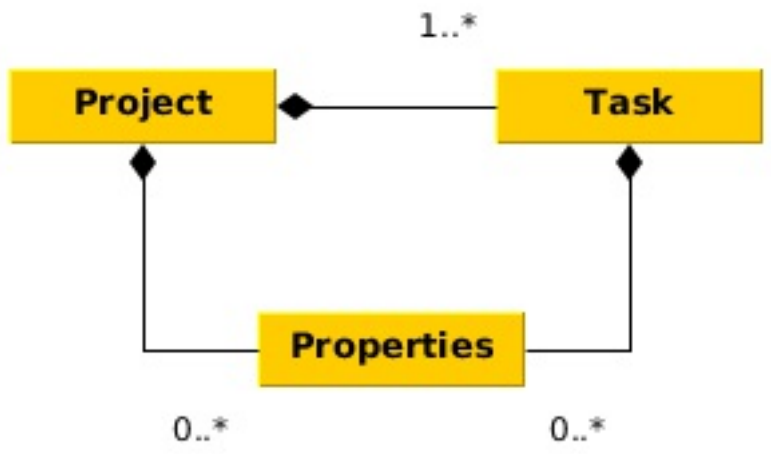
- Каждый grade build состоит из проектов, которые состоят из задачь. Задачи и проекты могут содержать в себе сойства, определяющие их поведение.



- Project - это интерфейс, доступен в build.gradle под именем project (или без имени вовсе):

```
project.setDescription('My Project')
println project.getDescription()
println getDescription()
```

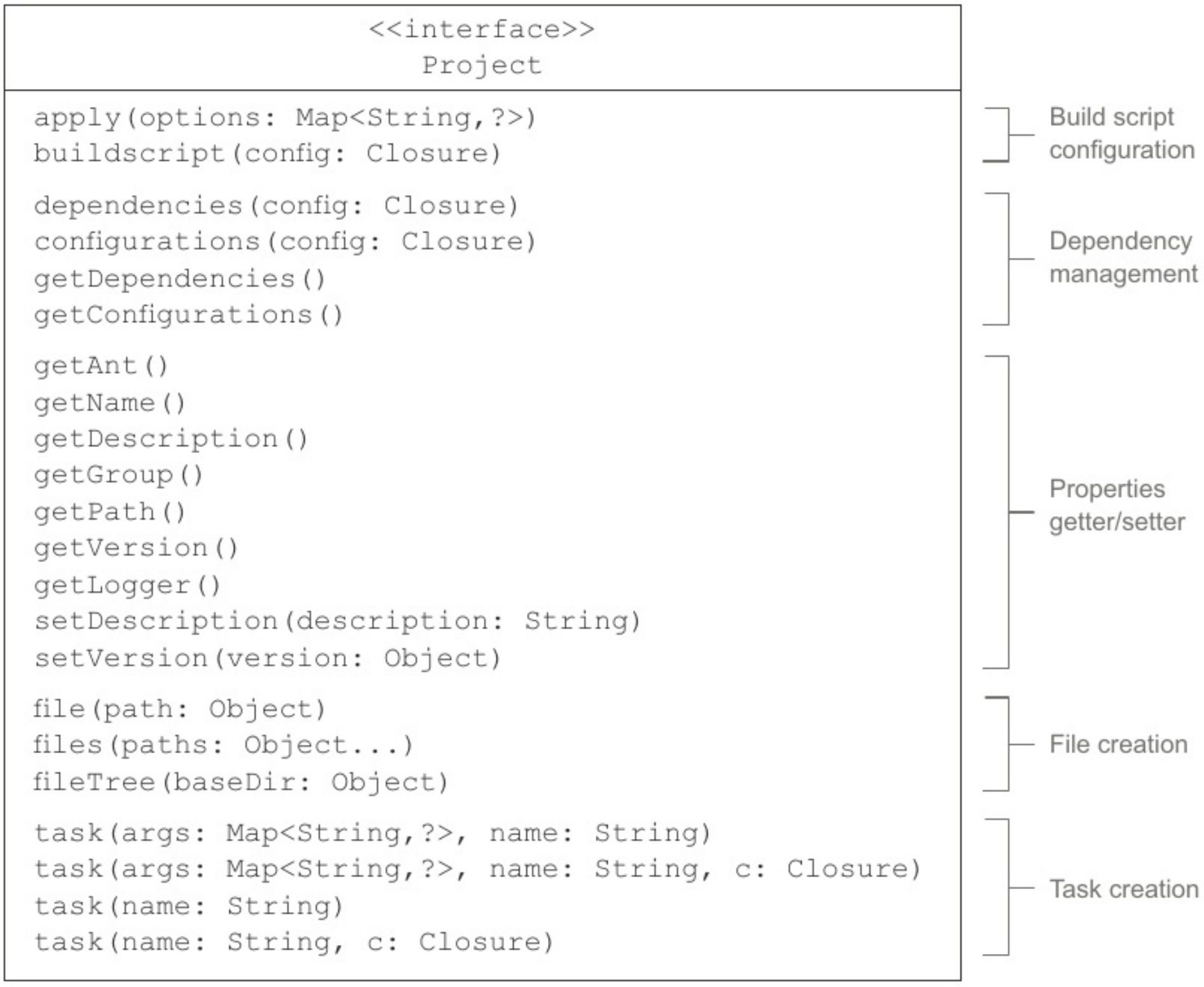
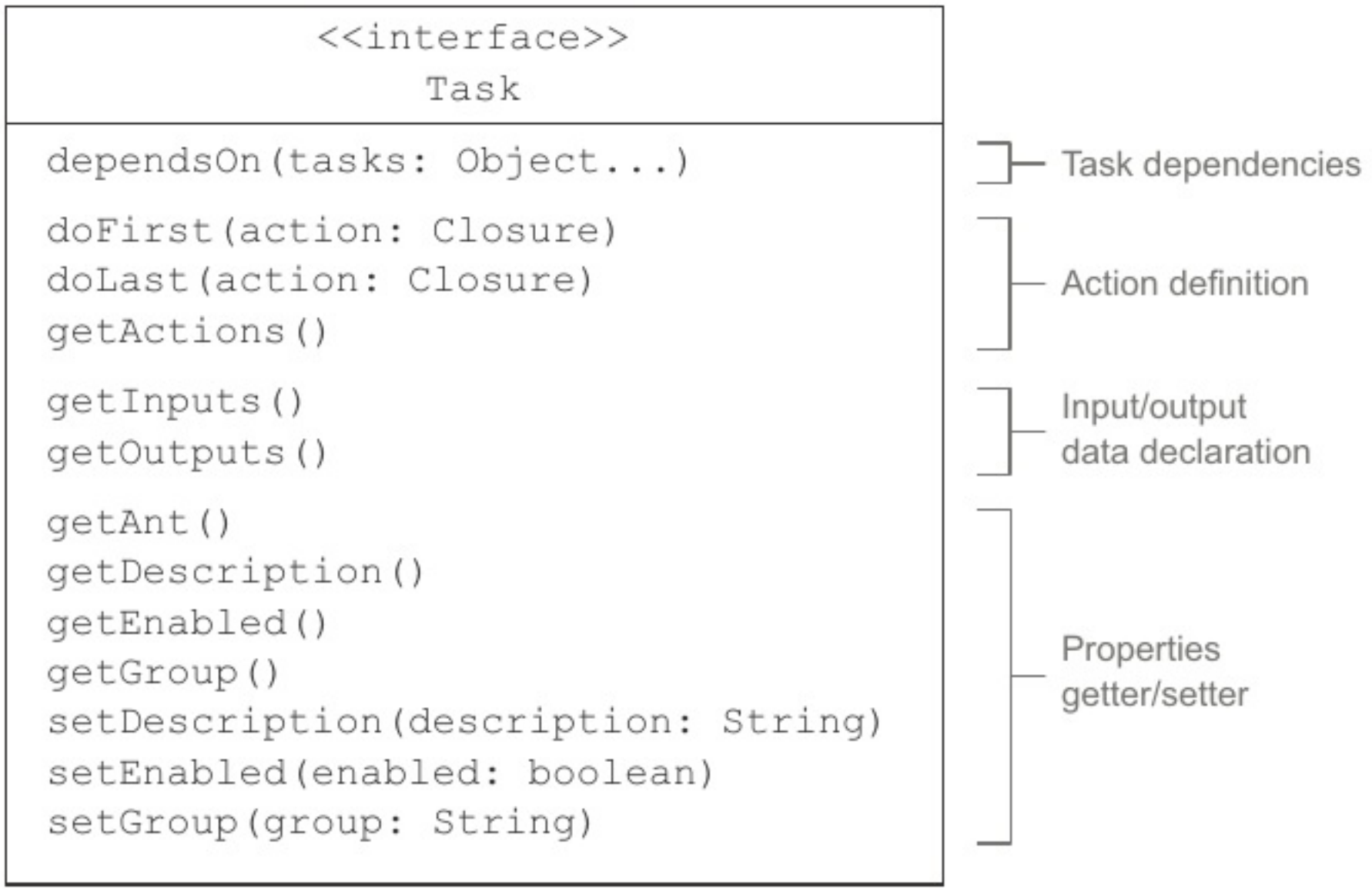


Figure 4.2 Main entry point of a Gradle build—the `Project` interface

- Task - это тоже интерфейс:



- Проекты и задачи имеют некоторое количество свойств, доступных через setters\getters.
- Многие классы Gradle позволяют добавлять доп. свойства используя пр-во имен ext. Такие свойства хранятся ввиде Map<String,?>:

```
project.ext.myProp = 'some value'
ext{
  myOtherProp='some value'
}
//после создания доп. свойства можно не использовать
//ext для доступа к нему
println project.myProp
myProp='some other value'
```

- gradle.properties позволяет указывать свойства проекта:

```
//gradle.properties
myProp='value'
//build.gradle
println project.myProp
```

Tasks

- Задачи могут принадлежать к определенным группам и иметь описания, gradle tasks отражает это:

```
task printVersion {
    group = 'versioning'
    description = 'Prints project version.'
    doLast {
      //gradle содержит свою реализацию логгера
      //на основе SLF4j
      logger.quiet "Version: $version"
    }
}
```

- задачи могу содержать любое количество closures в doLast и doFirst:

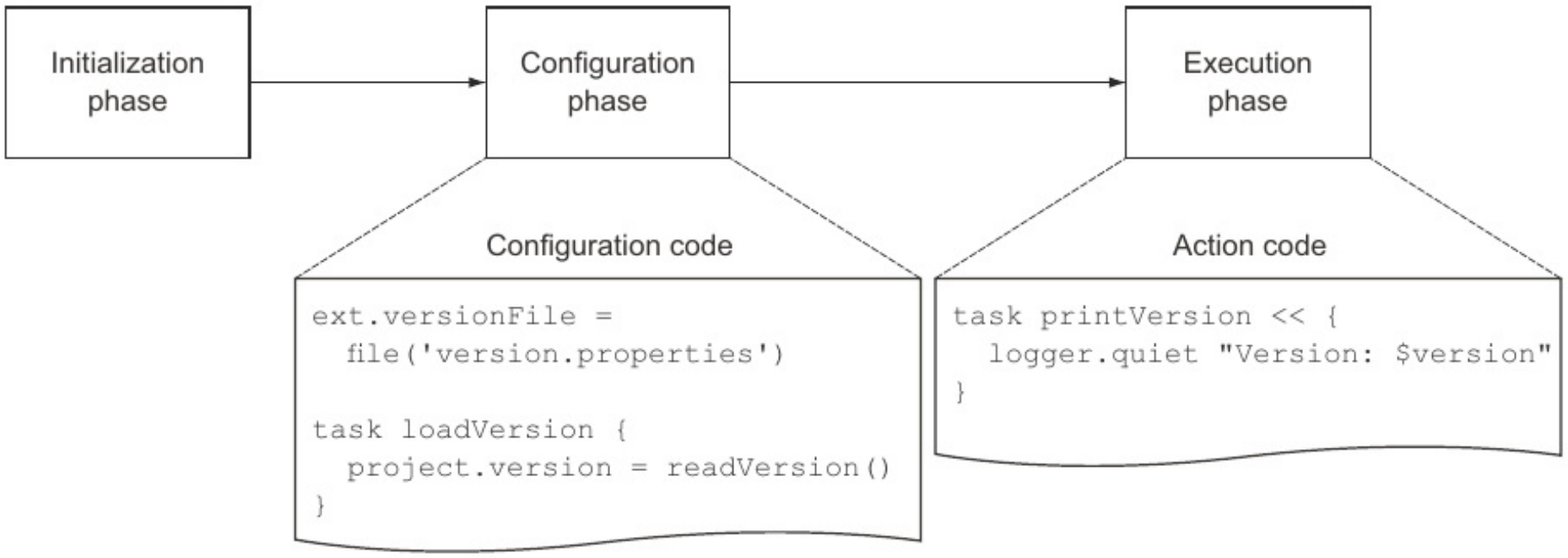
```
task someTask{
}
someTask.doLast {some action}
someTask.doLast {some other action}
```

- Finallizer tasks: задача second всегда выполняется после задачи first и служит например для освобождения ресурсов, используемых задачей first

```
task first << { println "first" }
task second << { println "second" }
first.finalizedBy second
```

Build Phases

- Инициализация: создается экз. project.
- Конфигурирование: происходит конфигурация всех объектов.
- Выполнение: выполняются указанные задачи.



Task inputs and outputs

Figure 4.6 Gradle determines if a task needs to be executed though its inputs/ outputs.

- В качестве входных данных задачи могут выступать: директории,файлы, свойства.
- Выходные данные могут быть: директория, файлы

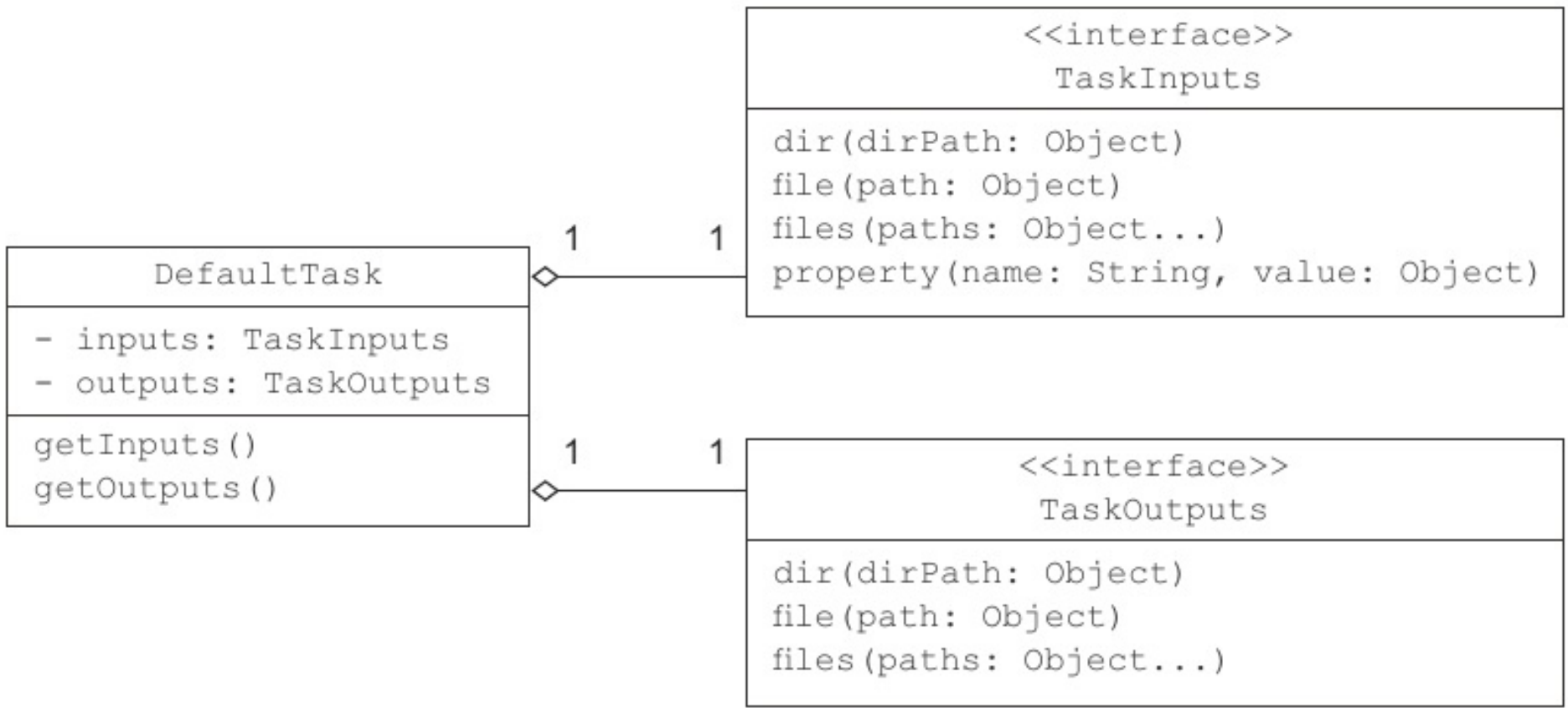


Figure 4.7 The class `DefaultTask` defines task inputs and outputs.

- Задача считается up-to-date если её входные и выходные данные не изменились с момента последнего запуска.
- Напишем задачу, которая изменяет версию на релиз:

Listing 4.5 Adding incremental build support via inputs/outputs

```
task makeReleaseVersion(group: 'versioning', description: 'Makes project
                        ➥ a release version.') {
   inputs.property('release', version.release)
   outputs.file versionFile

   doLast {
      version.release = true
      ant.propertyfile(file: versionFile) {
         entry(key: 'release', type: 'string', operation: '=', value: 'true')
      }
   }
}
```

Inputs/outputs are declared during configuration phase

As the version file is going to be modified it's declared as output file property

Declaring version release property as input