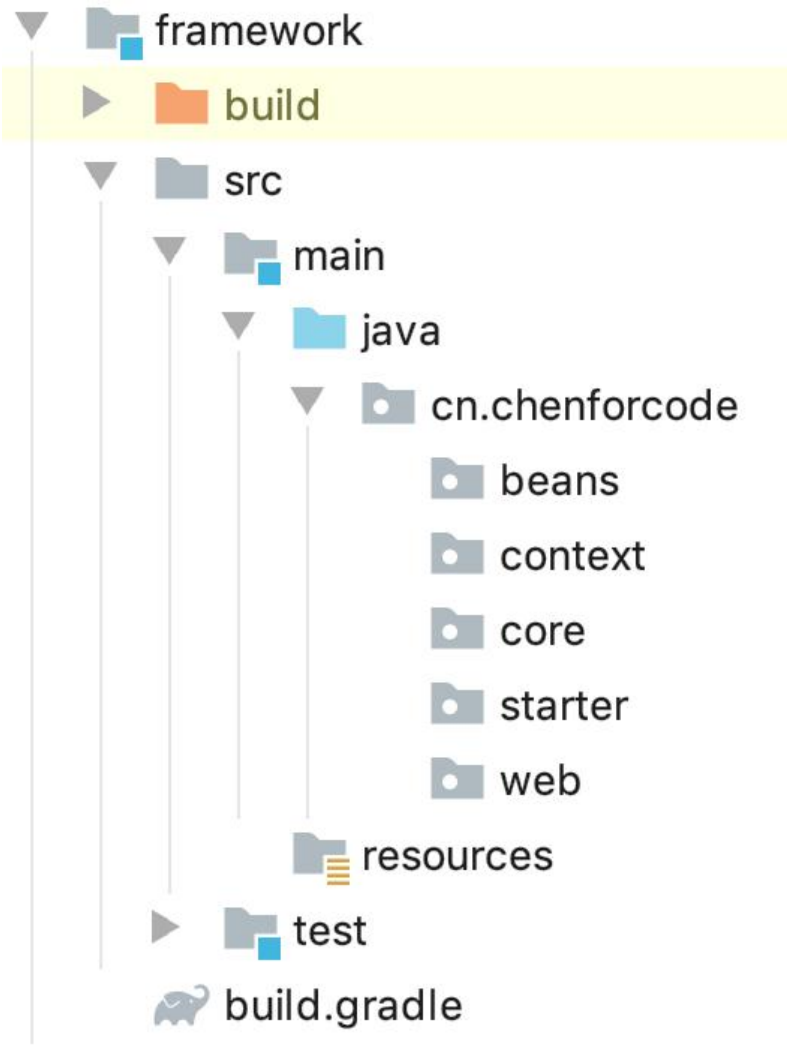
链滴

mini-spring 第一期：仿 spring 结构搭建

作者：ChenforCode
原文链接：https://ld246.com/article/1572444162394
来源网站：链滴

1.spring的核心功能模块有core模块，data模块，web模块，core模块中主要有beans包，context，core包。所以在frame模块中建立如下结构



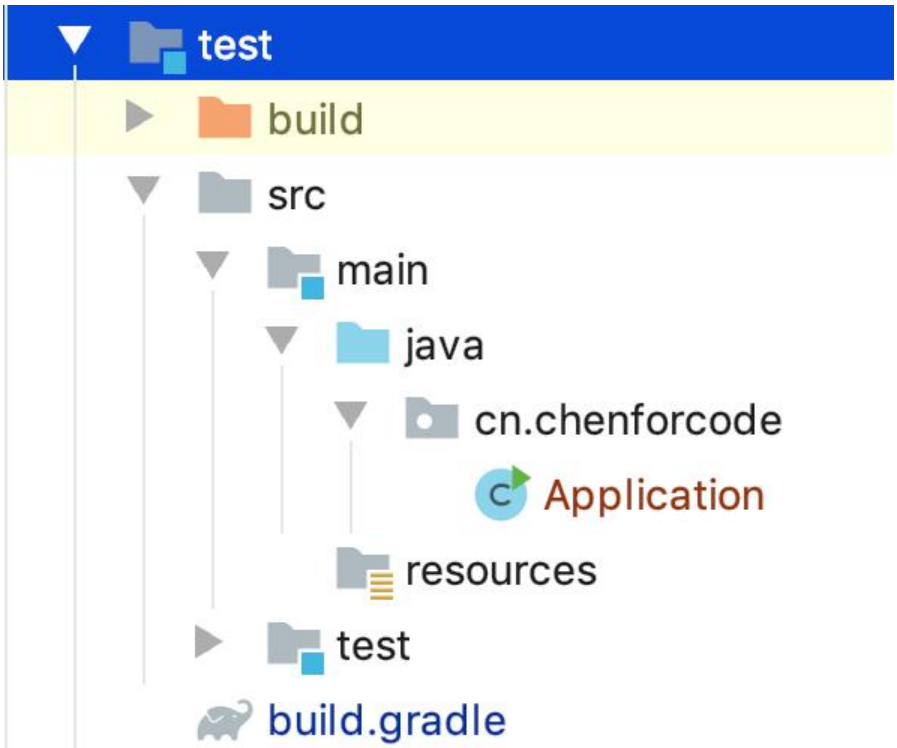
2.结构设计

@1：实现core模块，包括core包，context包，beans包

@2：实现web模块，集成web和webmvc

@3：添加starter，实现类似于spring boot的启动方式

3.实现启动器，在test模块下建立Applicaition类



编写Applicaition类

```
package cn.chenforcode;

public class Application {
    public static void main(String[] args) {
        System.out.println("Hello World!");
    }
}
```

4.打包项目，运行

```
gradle build
java -jar test/build/libs/test-1.0-SNAPSHOT.jar
```

这个时候出现了错误：test/build/libs/test-1.0-SNAPSHOT.jar中没有主清单属性

这是因为没有在打包的时候指定主类，在test模块下的build.gradle下编写相应的配置

```
jar {
    manifest {
        attributes "Main-Class": "cn.chenforcode.Application"
    }
    from {
        configurations.compile.collect {
            it.isDirectory() ? it : zipTree(it)
        }
    }
}
```

主要含义

manifest：指定主类

from：将所有的依赖都递归打包：如果是目录，就递归进入，如果不是目录就打包

5.framework模块打包

在test模块的gradle文件中添加,compile中添加的是要打包进去的兄弟模块的名称

```
dependencies {
    testCompile group: 'junit', name: 'junit', version: '4.12'
    compile(project(':framework'))
}
```

6.在frame模块的starter包中添加代码

```
package cn.chenforcode.starter;

public class MiniApplication {
    public static void run(Class<?> cls, String[] args) {
        System.out.println("Hello Mini-spring!");
    }
}
```

7.在test中调用这个代码

```
package cn.chenforcode;

import cn.chenforcode.starter.MiniApplication;

public class Application {
    public static void main(String[] args) {
        System.out.println("Hello World!");
        MiniApplication.run(Application.class, args);
    }
}
```

8.再次打包运行

```
pkucoder:mini-spring pkucoder$ gradle build

BUILD SUCCESSFUL in 758ms
5 actionable tasks: 5 executed
pkucoder:mini-spring pkucoder$ java -jar test/build/libs/test-1.0-SNAPSHOT.jar
Hello World!
Hello Mini-spring!
```