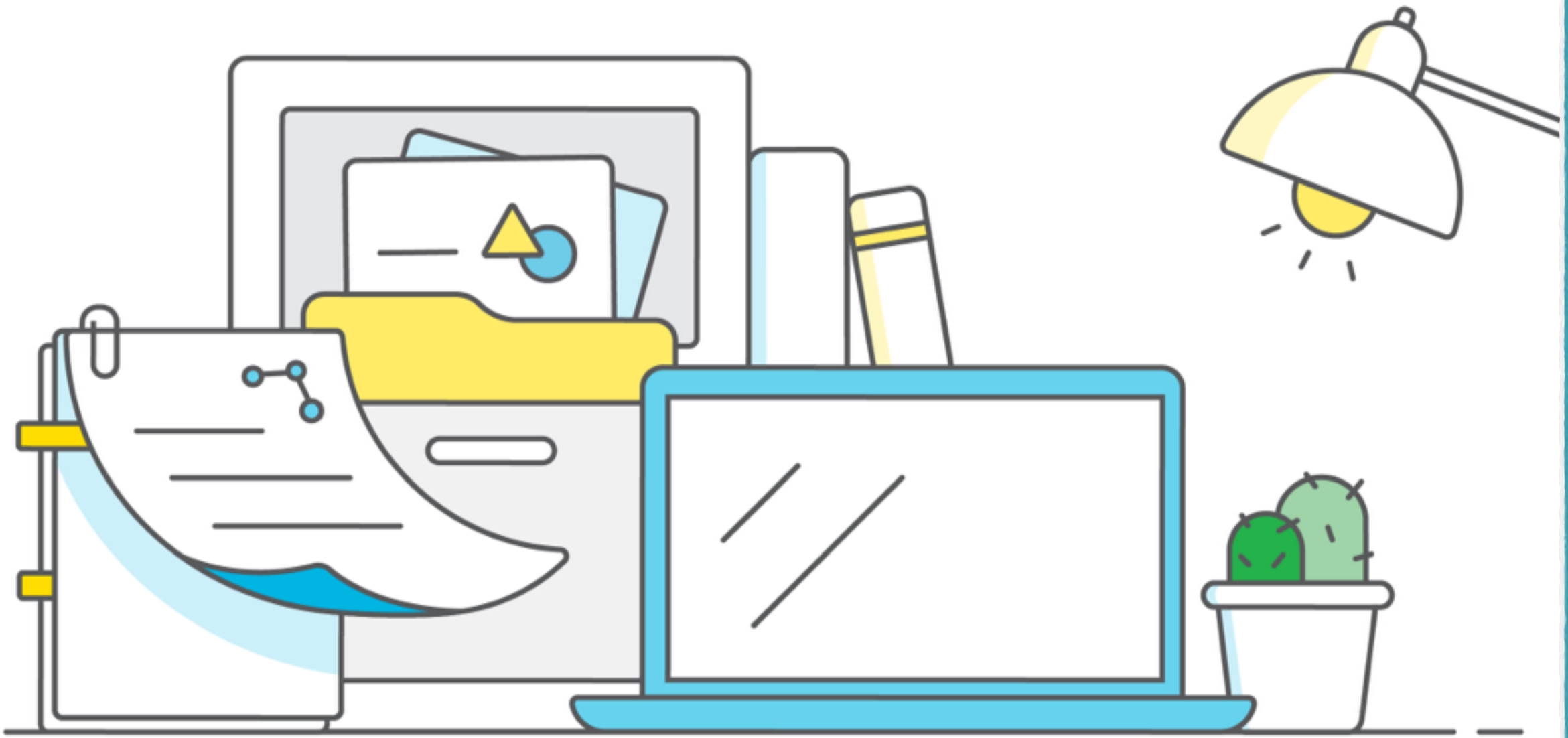


# Spring lesson 5

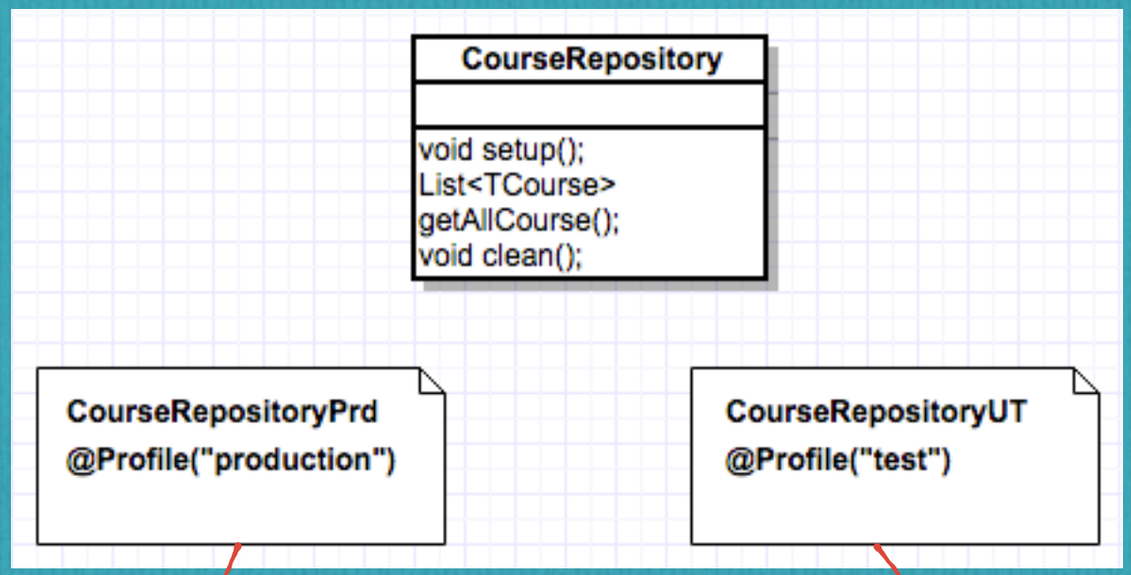
xieqiaoyun

# 本节需掌握

- ☐ 数据库连接 ← 根据自身情况补习哒~

- ☐ 掌握mock数据&单元测试
- ☐ Bean生命周期&容器扩展点

# (1.1) 使用Mockito定义两套Profile的数据源




```java
public void setDataSource(){
    BasicDataSource basicDataSource;
    String connectionString = "jdbc:mysql://localhost:3306/tcourse"
    if (dataSources.containsKey(connectionString)) {
        basicDataSource = dataSources.get(connectionString);
    } else {
        basicDataSource = new BasicDataSource();
        basicDataSource.setDriverClassName("com.mysql.jdbc.Driver")
        basicDataSource.setUrl(connectionString);
        basicDataSource.setUsername("root");
        basicDataSource.setPassword("123456");
        dataSources.put(connectionString, basicDataSource);
    }
    try {
        basicDataSource.getConnection();
        jdbcTemplate = new JdbcTemplate(basicDataSource);
    } catch (SQLException e) {
        e.printStackTrace();
    }
```

```java
public void setup(){
    dataSource = Mockito.mock(DataSource.class);
    connection = Mockito.mock(Connection.class);
    jdbcTemplate = Mockito.mock(JdbcTemplate.class);
    System.out.println("init");
}
public List<TCourse> getAllCourse() {
    List<TCourse> list = new ArrayList<>();
    list.add(new TCourse( id: 1, name: "test1", mark: 90));
    list.add(new TCourse( id: 2, name: "test2", mark: 80));
    list.add(new TCourse( id: 3, name: "test2", mark: 100));
    Mockito.when(
            jdbcTemplate.query(
                    sql: "SELECT id, name, mark FROM course",
                    (rs, rowNum) -> new TCourse(rs.getInt( col
            )
        ).thenReturn(list);
    return list;
}
```

# (1.2) 配置bean

```java
@Configuration
public class AppConfig {

    @Autowired
    CourseRepository dataSource;

    @Bean
    public MyCourseService myCourseService(){
        return new MyCourseService(dataSource);
    }
}
```

+

```java
@Configuration
public class DataSourceConfig {
    @Bean(name="dataSource")
    @Profile("test")
    public CourseRepository getTestRepository(){
        return new CourseRepositoryUT();
    }

    @Bean(name="dataSource")
    @Profile("production")
    public CourseRepository getPrdRepository(){
        return new CourseRepositoryPrd();
```

```java
public static void main(String[] args){
    AnnotationConfigApplicationContext ctx = new AnnotationConfigApplicationContext();
      ctx.getEnvironment().setActiveProfiles("production");
    ctx.getEnvironment().setActiveProfiles("test");
    ctx.register(DataSourceConfig.class);
    ctx.register(AppConfig.class);
    ctx.refresh();

    MyCourseService myCourseService = (MyCourseService) ctx.getBean( name: "myCourseService");
    myCourseService.setup();
    List<TCourse> courseList = myCourseService.getAllCourses();
    for(TCourse course:courseList){
        System.out.println(course.toString());
    }
```

# (1.3) 运行production profile和test profile

## test profile

```
init
TCourse[id=1, name=test1, mark=90]
TCourse[id=2, name=test2, mark=80]
TCourse[id=3, name=test2, mark=100]

Process finished with exit code 0
```

## production profile

```
Creating tables
Sun Sep 10 20:15:56 CST 2017 WARN: Establishing S
TCourse[id=31, name=xqy, mark=61]
TCourse[id=32, name=simida, mark=78]
TCourse[id=33, name=hq, mark=93]
TCourse[id=34, name=meme, mark=33]
TCourse[id=35, name=aaa, mark=62]
TCourse[id=31, name=xqy, mark=61]
TCourse[id=32, name=simida, mark=78]
TCourse[id=33, name=hq, mark=93]
TCourse[id=34, name=meme, mark=33]
TCourse[id=35, name=aaa, mark=62]

Process finished with exit code 0
```

# (1.4) JUnit单元测试

```java
@RunWith(SpringJUnit4ClassRunner.class)
@ContextConfiguration(classes = {AppConfig.class, DataSourceConfig.class})
@ActiveProfiles("test")
public class MyCourseServiceTest {

    @Autowired
    MyCourseService myCourseService;

    @Before
    public void createConn(){
        System.out.println("before class: prepare data");
        myCourseService.setup();
    }

    @Test
    public void getAllCourses(){
        System.out.println("run test...");
        List<TCourse> courses = myCourseService.getAllCourses();
        for(TCourse course:courses){
            System.out.println(course.toString());
        }
    }

    @After
    public void removeData(){
        System.out.println("after class: remove data");
        myCourseService.clean();
    }
```

1 test passed - 644ms

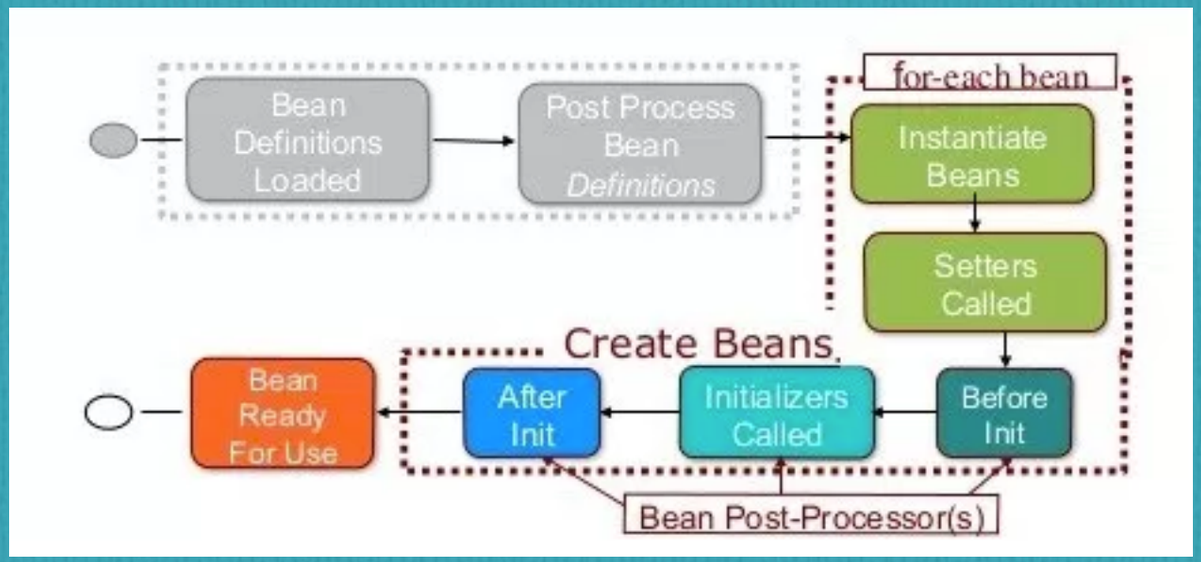
```
信息: Refreshing org.springframework.context.support.GenericApplicationContext@735b5592: sta
before class: prepare data
init
run test...
TCourse[id=1, name=test1, mark=90]
TCourse[id=2, name=test2, mark=80]
TCourse[id=3, name=test2, mark=100]
after class: remove data
```

(

# 三个容器扩展点

- 自定义Bean的初始化过程，实现BeanPostProcessor接口
- 自定义Bean元数据，实现BeanFactoryPostProcessor接口，BeanFactoryPostProcessor作用在BeanPostProcessor之前
- 自定义容器初始化逻辑，实现FactoryBean接口

# Bean生命周期&BeanPostProcessor的作用范围




Bean实例化后，init方法之前和之后

# (2)自定义BeanPostProcessor检查Bean定义属性

```java
@Component
public class MyBeanPostProcessor implements BeanPostProcessor{
    @Override
    public Object postProcessBeforeInitialization(Object o, String s) throws BeansException {
        Field[] fields = o.getClass().getDeclaredFields();
        Class[] classes = o.getClass().getInterfaces();
        if (fields.length > 4){
            System.out.println(String.format("waring %s has %d properties:",s,fields.length));
            for(Field field:fields){
                System.out.println(String.format("     property %s type %s",field.getName(),field.getType().toString()));
            }
        }
        if (classes.length > 2){
            System.out.println(String.format("waring %s has implement %d interfaces: ",s,classes.length));
            for(Class clazz:classes){
                System.out.println(String.format("     interface %s : ",clazz.toString()));
            }
        }
        return o;
    }

    @Override
    public Object postProcessAfterInitialization(Object o, String s) throws BeansException {
        return o;
    }
```

MyBeanPostProcessorTest

```
     property CGLIB$emptyArgs type class [Ljava.lang.Object;
     property $$beanFactory type interface org.springframework.beans.factory.BeanFactory
waring myBean2 has 5 properties:
     property id type int
     property firstname type class java.lang.String
     property lastname type class java.lang.String
     property age type int
     property mark type int
MyBean[id=0, name=first bean]@392781299
MyBean2[id=0, fistname=null, lastname=null, age=0, mark=0]@1822383117

Process finished with exit code 0
```

代码已上传github

# Thank You

–xieqiaoyun