1. DefautListableBeanFactory -- bean注册，后续的增删改查功能

2. XmlBeanDefinitionReader：XML配置文件的读取，解析，注册

ClassPathXmApplicationContext：利用XmlBeanDefinitionReader工具，读取配置并加载到spring容器，然后初始化

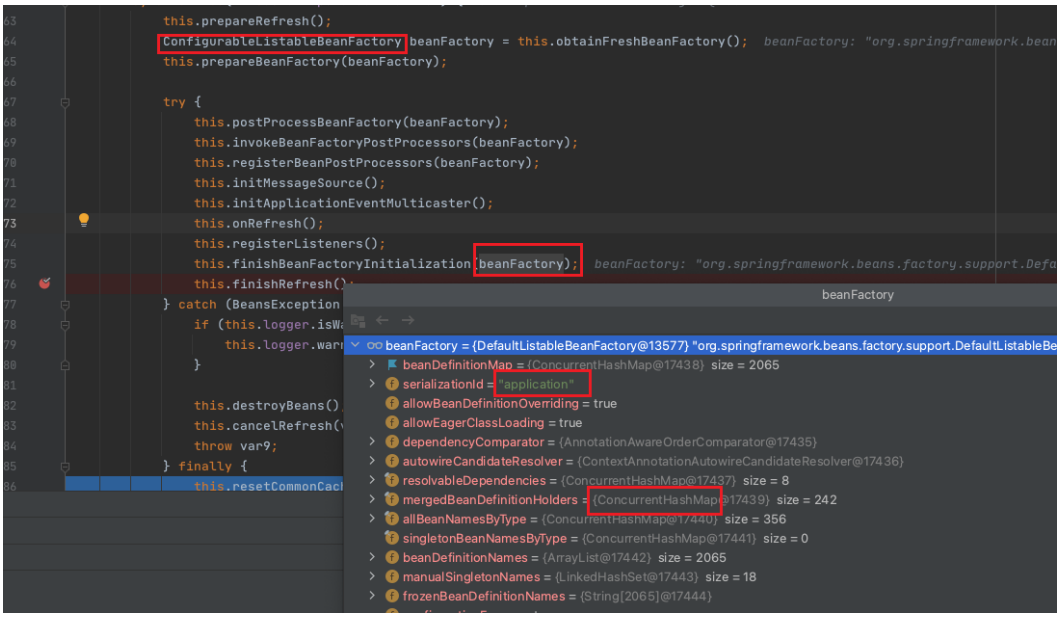
其中的refresh方法（spring容器启动的核心方法）涵盖了：xml文件读取，bean的创建，实例化，几乎所有applicationContext中提供的内容

BeanDifinition和Resource的定位，载入，和注册的三个基本过程

3.obtainFreshBeanFactory：此上下文的基础Bean工厂的实际刷新，关闭之前的bean工厂（如有）并初始化上一个声明周期的下一阶段的新bean工厂

关注数据存在哪了，不然还是了解不透彻

bean的数据都在下面这个ConfigurableListableBeanFactory里（DefautListableBeanFactory）



特别的AbstractApplicaitonContext的refresh方法

```
public void refresh() throws BeansException, IllegalStateException {
    synchronized(this.startupShutdownMonitor) {
        // 1、调用容器准备刷新的方法，获取容器的当前时间，为容器设置同步标识
        this.prepareRefresh();
        2、调用子类执行refreshBeanFactory方法 进行资源文件的载入和注册
        创建ioc容器，配置启动参数，开启注解的自动装配，loadBeanDifintion对beanDifinition
        ConfigurableListableBeanFactory beanFactory = this.obtainFreshBeanFactory
        3、为beanFactory配置容器特性，例如类加载器，事件处理器
        this.prepareBeanFactory(beanFactory);

        try {
            4、这个方法其实是在添加BeanpostProcessor（大多数Context都是这么做的，
            如AnnotationConfigServletWebServerApplicationContext有三点作用

                1、特定四个bean(ServletRequest,ServletResponse,HttpSession,WebReque
```

```
            2、这个是在beanFactory加载完毕，尚未构建beanDifinition和实例化bean之前，
            这个方法是一个待实现的方法，留给各Context实现自己的业务逻辑（也就是beanFact
                例如：修改BeanFactory 或者注册BeanPostProcessor等等
            3、通过scan和register完成注解扫描加载BeanDifinition,通过springboot的启
                注意：scanner只能完成@Component@Service@Controller@Repository的
        this.postProcessBeanFactory(beanFactory);
        5、调用这个beanFactory中所有注册的BeanFactoryPostProcessor的bean，这其实是
                操作三种对象并调用他们重写的方法即postProcessBeanDefinitionRegistry
                1、beanFactoryPostProcessors中的对象，
               2、3、 继承BeanDifnitionRegisterPostProcessor和BeanFactoryPostPr
           举例
               我们通常在使用 Mybatis + Spring 时，经常用到的 org.mybatis.spring.m
               MapperScannerConfigurer 在 postProcessBeanDefinitionRegistry 方
                   主要是：扫描 basePackage 指定的目录，将该目录下的类（通常是 DAO/MA
                   因此，我们可以看到我们项目中的 DAO（MAPPER）接口，通常都没有使用注解
                   但是我们还是可以在 Service 服务中，使用 @Autowire 注解来将其注入到
        this.invokeBeanFactoryPostProcessors(beanFactory);
        6、将BeanPostProcessors注册到BeanFactory 的 beanPostProcessors 缓存中，跟
            BeanPostProcessor接口定义了after和before方法，可以在实例的初始化方法前后
        this.registerBeanPostProcessors(beanFactory);
        7、初始化信息源，跟国际化有关
        this.initMessageSource();
        8、初始化容器的事件传播器
        this.initApplicationEventMulticaster();
        9、调用子类的某些特殊bean的初始化方法
        this.onRefresh();
        10、为事件传播器注册事件监听器
        this.registerListeners();
        11、初始化所有剩余的单例Bean，初始化所有的单例bean（这是重点）
        this.finishBeanFactoryInitialization(beanFactory);
        12、初始化容器的声明周期事件处理器，并发布容器的声明周期时间
        this.finishRefresh();
    } catch (BeansException var9) {
        if (this.logger.isWarnEnabled()) {
            this.logger.warn("Exception encountered during context initializa
        }
        13、销毁已经创建的bean
        this.destroyBeans();
        14、取消刷新操作，重置容器同步标识
        this.cancelRefresh(var9);
```

```
55              throw var9;
56          } finally {
57              this.resetCommonCaches();
58          }
59
60      }
61  }
```