

spring --- analyse

1.为什么说web.xml是父子容器的关系

1.为什么说web.xml是父子容器的关系

```
public class DispatcherServlet extends FrameworkServlet {  
    public static final String MULTIPART_RESOLVER_BEAN_NAME = "multipartResolver";  
    public static final String LOCALE_RESOLVER_BEAN_NAME = "localeResolver";  
    public static final String THEME_RESOLVER_BEAN_NAME = "themeResolver";  
}
```

```
protected WebApplicationContext initWebApplicationContext() {  
    WebApplicationContext rootContext = WebApplicationContextUtils.getWebApplicationContext(this.getServiceLoader());  
    WebApplicationContext wac = null;  
    if (this.webApplicationContext != null) {  
        wac = this.webApplicationContext;  
        if (wac instanceof ConfigurableWebApplicationContext) {  
            ConfigurableWebApplicationContext cwac = (ConfigurableWebApplicationContext)wac;  
            if (!cwac.isActive()) {  
                if (cwac.getParent() == null) {  
                    cwac.setParent(rootContext);  
                }  
                this.configureAndRefreshWebApplicationContext(cwac);  
            }  
        }  
    }  
    if (wac == null) {  
        wac = this.findWebApplicationContext();  
    }  
    if (wac == null) {  
        wac = this.createWebApplicationContext(rootContext);  
    }  
    if (!this.refreshEventReceived) {  
        synchronized(this.onRefreshMonitor) {  
            this.onRefresh(wac);  
        }  
    }  
}
```

1 获取到springioc容器

2 发现自己容器为Null

3 走这个逻辑没找到

4 那就自己创建一个名字为wac的mvc容器

```
protected WebApplicationContext createWebApplicationContext(@Nullable WebApplicationContext parent) {  
    return this.createWebApplicationContext((ApplicationContext)parent);  
}
```

1 进入创建容器的方法，在进去

```
protected WebApplicationContext createWebApplicationContext(@Nullable ApplicationContext parent) {
    Class<?> contextClass = this.getContextClass();
    if (!ConfigurableWebApplicationContext.class.isAssignableFrom(contextClass)) {
        throw new ApplicationContextException("Fatal initialization error in servlet with name '" + this.getServletName() +
    } else {
        ConfigurableWebApplicationContext wac = (ConfigurableWebApplicationContext)BeanUtils.instantiateClass(contextClass);
        wac.setEnvironment(this.getEnvironment());
        wac.setParent(parent);
        String configLocation = this.getContextConfigLocation();
        if (configLocation != null) {
            wac.setConfigLocation(configLocation);
        }

        this.configureAndRefreshWebApplicationContext(wac);
        return wac;
    }
}
```

1 用工具类搞出一个自己的容器来,

2 并设置父容器, 这里我的web.xml中没配置ioc容器, 所以debug---parent属性为Null

3 具体debug效果看下一张图

```
protected WebApplicationContext createWebApplicationContext(@Nullable ApplicationContext parent) { parent: null
    Class<?> contextClass = this.getContextClass(); contextClass: "class org.springframework.web.context.support.XmlWebA
    if (!ConfigurableWebApplicationContext.class.isAssignableFrom(contextClass)) {
        throw new ApplicationContextException("Fatal initialization error in servlet with name '" + this.getServletName()
    } else {
        ConfigurableWebApplicationContext wac = (ConfigurableWebApplicationContext)BeanUtils.instantiateClass(contextClass);
        wac.setEnvironment(this.getEnvironment());
        wac.setParent(parent); parent: null
        String configLocation = this.getContextConfigLocation(); configLocation: "classpath:day04/day04-mvc-config.xml"
        if (configLocation != null) {
            wac.setConfigLocation(configLocation); configLocation: "classpath:day04/day04-mvc-config.xml"
        }

        this.configureAndRefreshWebApplicationContext(wac);
        return wac; wac: "WebApplicationContext for namespace 'dispatchServlet-servlet', started on Thu May 12 08:16:48
    }
}
```

- 为什么可以设置parent原因是请看下图

```
public interface ConfigurableWebApplicationContext extends WebApplicationContext, ConfigurableApplicationContext {
    String APPLICATION_CONTEXT_ID_PREFIX = WebApplicationContext.class.getName() + ":";
    String SERVLET_CONFIG_BEAN_NAME = "servletConfig";

    void setServletContext(@Nullable ServletContext var1);

    void setServletConfig(@Nullable ServletConfig var1);

    @Nullable
    ServletConfig getServletConfig();

    void setNamespace(@Nullable String var1);

    @Nullable
    String getNamespace();
}
```

1 mvc容器类型, 不仅实现了普通版的webApplicationContext还实现了ConfigurableApplicationContext可以设置更多配置的

```
public interface ConfigurableApplicationContext extends ApplicationContext, Lifecycle, Closeable {
    String CONFIG_LOCATION_DELIMITERS = ",; \\t\\n";
    String CONVERSION_SERVICE_BEAN_NAME = "conversionService";
    String LOAD_TIME_WEAVER_BEAN_NAME = "loadTimeWeaver";
    String ENVIRONMENT_BEAN_NAME = "environment";
    String SYSTEM_PROPERTIES_BEAN_NAME = "systemProperties";
    String SYSTEM_ENVIRONMENT_BEAN_NAME = "systemEnvironment";
    String SHUTDOWN_HOOK_THREAD_NAME = "SpringContextShutdownHook";

    void setId(String var1);

    void setParent(@Nullable ApplicationContext var1);

    void setEnvironment(ConfigurableEnvironment var1);

    ConfigurableEnvironment getEnvironment();

    void addBeanFactoryPostProcessor(BeanFactoryPostProcessor var1);

    void addApplicationListener(ApplicationListener<?> var1);
}
```