RequestMappingHandlerMapping初始化获取拦截器流程

首先是WebMvcConfigurationSupport先进行创建，然后获取WebMvcConfigurer接口实现类添加的拦截器，代码如下：getInterceptors()

*/\*\*  
 \* Return a {****@link*** *RequestMappingHandlerMapping} ordered at 0 for mapping  
 \* requests to annotated controllers.  
 \*/*@Bean  
**public** RequestMappingHandlerMapping requestMappingHandlerMapping() {  
 RequestMappingHandlerMapping mapping = createRequestMappingHandlerMapping();  
 mapping.setOrder(0);  
 mapping.setInterceptors(getInterceptors());  
 mapping.setContentNegotiationManager(mvcContentNegotiationManager());  
 mapping.setCorsConfigurations(getCorsConfigurations());  
  
 PathMatchConfigurer configurer = getPathMatchConfigurer();  
  
 Boolean useSuffixPatternMatch = configurer.isUseSuffixPatternMatch();  
 **if** (useSuffixPatternMatch != **null**) {  
 mapping.setUseSuffixPatternMatch(useSuffixPatternMatch);  
 }  
 Boolean useRegisteredSuffixPatternMatch = configurer.isUseRegisteredSuffixPatternMatch();  
 **if** (useRegisteredSuffixPatternMatch != **null**) {  
 mapping.setUseRegisteredSuffixPatternMatch(useRegisteredSuffixPatternMatch);  
 }  
 Boolean useTrailingSlashMatch = configurer.isUseTrailingSlashMatch();  
 **if** (useTrailingSlashMatch != **null**) {  
 mapping.setUseTrailingSlashMatch(useTrailingSlashMatch);  
 }  
  
 UrlPathHelper pathHelper = configurer.getUrlPathHelper();  
 **if** (pathHelper != **null**) {  
 mapping.setUrlPathHelper(pathHelper);  
 }  
 PathMatcher pathMatcher = configurer.getPathMatcher();  
 **if** (pathMatcher != **null**) {  
 mapping.setPathMatcher(pathMatcher);  
 }  
 Map<String, Predicate<Class<?>>> pathPrefixes = configurer.getPathPrefixes();  
 **if** (pathPrefixes != **null**) {  
 mapping.setPathPrefixes(pathPrefixes);  
 }  
  
 **return** mapping;  
}

又由于RequestMappingHandlerMapping继承自AbstractHandlerMapping，且实现了ApplicationContextAware接口，在注入ApplicationContext时，会触发AbstractHandlerMapping的initApplicationContext方法

*/\*\*  
 \* Initializes the interceptors.  
 \** ***@see*** *#extendInterceptors(java.util.List)  
 \** ***@see*** *#initInterceptors()  
 \*/*@Override  
**protected void** initApplicationContext() **throws** BeansException {  
 extendInterceptors(**this**.**interceptors**);  
 detectMappedInterceptors(**this**.**adaptedInterceptors**);  
 initInterceptors();  
}

detectMappedInterceptors(**this**.**adaptedInterceptors**);

方法会从ioc容器中获取MappedInterceptor.**class**的实例，所以这种类型的HandlerInterceptor,要么通过@Bean注解注入，要么通过我们自定义的WebMvcConfigurer接口实现类注入，不能两者同时存在，否则会出现多个MappedInterceptor