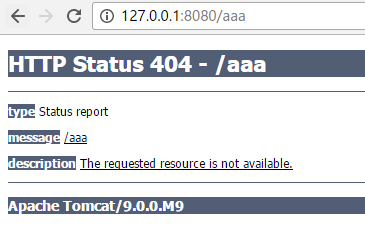
前奏

对于Tomcat、Jetty等Servlet容器而言,当Web应用出现错误时,会响应对应的错误页。以Tomcat为例,下面是默认的404页面和500页面：



这些错误页面的处理,可以通过web.xml的error-page节点进行配置：

<!-- 404 页面不存在错误 -->

<error-page>

<error-code>404</error-code>

<location>/error.jsp</location>

</error-page>

<!-- 500 服务器内部错误 -->

<error-page>

<error-code>500</error-code>

<location>/error.jsp</location>

</error-page>

<!-- java.lang.Exception异常错误,依据这个标记可定义多个类似错误提示 -->

<error-page>

<exception-type>java.lang.Exception</exception-type>

<location>/error.jsp</location>

</error-page>

这些配置在应用启动时,解析web.xml文件的时候,会生成对应ErrorPage对象,并设置到Context属性中;

在应用处理请求时,当发生错误时,会forward到对应的errorPage,进而呈现对应的错误页面。

题外话:forward方式调整在SpringMVC中会执行两次dispatch

SpringBoot采用的是Java Code配置,容器也是可以内嵌的;在进行对应的容器context设置时,将errorPage配置进去,就可以实现自定义错误页显示了.

问题1：errorPage是怎么配置的？

SpringBoot已经为我们写好了错误页处理流程,可以通过配置**server.error.path**指定应用发生错误后的跳转Url;可以在SpringBoot静态资源路径下配置error.[html|jsp]或者在静态资源路径的error目录配置{errorCode}.[ html|jsp]

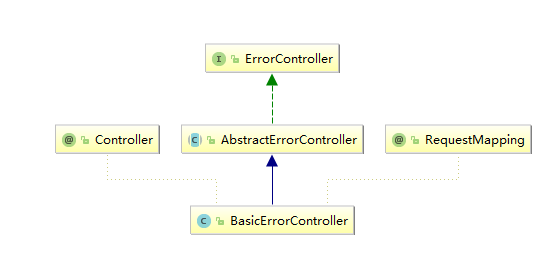
ps ==> 静态资源路径指：/(根目录)、classpath:/META-INF/resources/、classpath:/resources/、classpath:/static/、classpath:/public;优先级从高到低.

静态资源路径也可以通过**spring.resources**进行指定

问题2：errorPage是怎么处理的？

SpringBoot提供了一个BasicErrorController,用于处理错误页后的跳转逻辑。

org.springframework.boot.autoconfigure.web.BasicErrorController,类图继承关系如下：



BasicErrorController提供了两个接口,默认路径都是/error(可以通过**server.error.path**指定)。

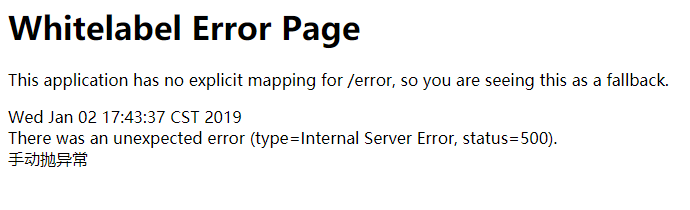
一个用于响应错误页面，一个用于响应错误json数据。

响应错误页面时，会根据DefaultErrorViewResolver，解析出来的页面进行渲染，默认的解析规则是：

1. 解析静态资源路径下的：error/{viewName}.html文件
2. 解析不到再解析error/[4xx|5xx].html文件
3. 再解析不到，则返回默认的view页面(通过BeanName解析器,解析name为error的bean)

remark:1、2步得到的错误页是交给模板引擎进行解析的。模板引擎的加载是在SpringBoot AutoConfigure进行的(在spring.factories文件配置了模板引擎提供类)。

下面是默认的错误页：



如果是响应错误json,则会从DefaultErrorAttributes获取异常信息,作为一个map,返回,该异常信息拥有的Key有：

**timestamp:错误发生的时间戳**

**status:http错误码**

**error: http错误码对应的说明**

**exception:异常类名**

**message:错误消息**

**errors:参数绑定错误**

**trace:出错的堆栈信息**

**path:请求出错的路径**

下面是一个栗子：

{

"timestamp": 1546056544666,

"status": 500,

"error": "Internal Server Error",

"exception": "java.lang.Error",

"message": "手动抛异常",

"path": "/exception/error"

}

想要自定义配置页,在不提供自己的ErrorController的情况下,通过在静态资源路径下,配置下面的静态路径：

error/error.html

error/4xx.html

error/5xx.html

第二种是提供自己的ErrorController(需要实现

org.springframework.boot.autoconfigure.web ErrorController)

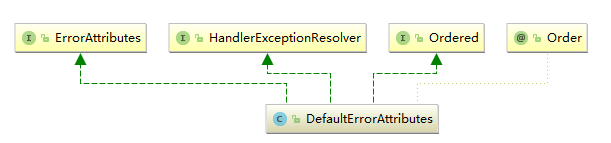
通过上面分析,影响到错误结果解析的有：

org.springframework.boot.autoconfigure.web ErrorController

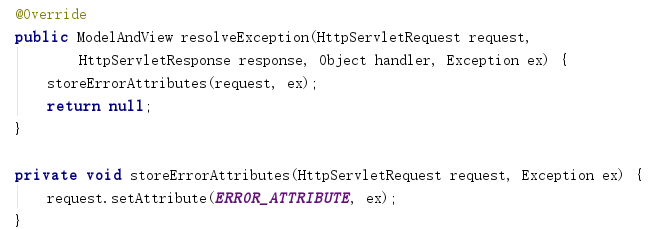
org.springframework.boot.autoconfigure.web. ErrorAttributes

org.springframework.boot.autoconfigure.template. TemplateAvailabilityProvider

1. ErrorAttribute,用于存放异常时的信息,默认使用org.springframework.boot.autoconfigure.web.DefaultErrorAttributes



DefaultErrorAttributes实现了HandlerExceptionResolver,这是SpringMVC用于请求错误时的异常解析器。DefaultErrorAttributes只是将发生的异常设置到request的Attribute中：



1. 默认错误页：Whitelabel Error Page

当没有可用的TemplateAvailabilityProvider时,会进行默认错误页的配置

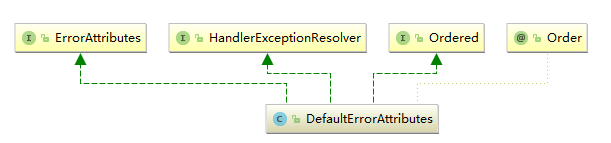
1. 默认错误页解析器：DefaultErrorViewResolver

用于实现错误页的1、2步解析规则

MVC异常自动配置类

org.springframework.boot.autoconfigure.web.ErrorMvcAutoConfiguration

1. 配置ErrorAttribute,用于存放异常时的信息,默认使用org.springframework.boot.autoconfigure.web.DefaultErrorAttributes



默认含有的属性有：

**timestamp:错误发生的时间戳**

**status:http错误码**

**error: http错误码对应的说明**

**exception:异常类名**

**message:错误消息**

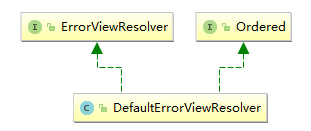
**errors:参数绑定错误**

**trace:出错的堆栈信息**

**path:请求出错的路径**

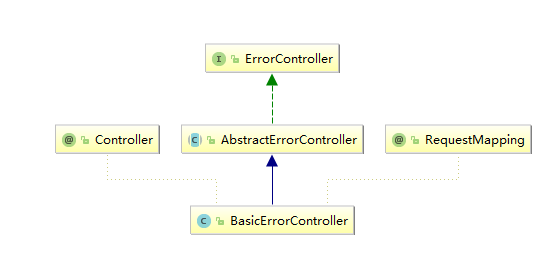
1. 配置视图解析器,用于解析ModelAndView

org.springframework.boot.autoconfigure.web.DefaultErrorViewResolver

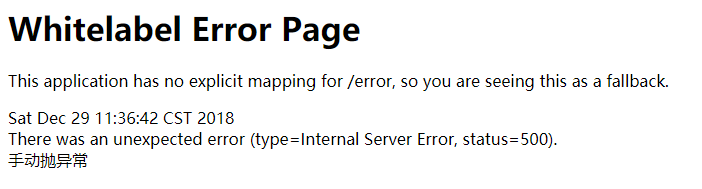


1. 配置异常控制器,发生异常时跳转的url,默认是/error,可以通过**server.error.path**进行配置

org.springframework.boot.autoconfigure.web.BasicErrorController



1. 配置默认的错误View,View名为**error**样式如下(当找不到错误视图时使用):



1. 定义错误页注册器,用于配置容器的错误页

错误路径可以通过**server**.**error.path进行配置,默认是：/error**

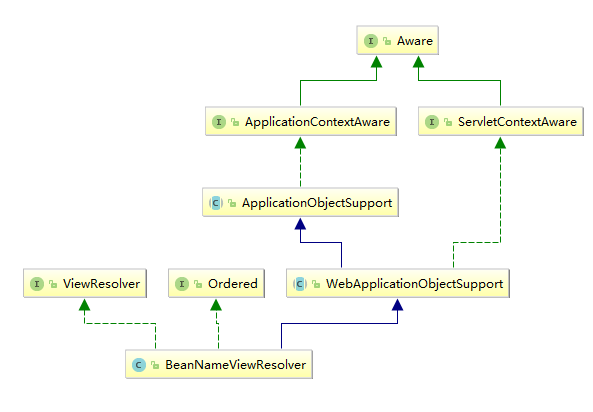
ErrorPageCustomizer

在每个容器进行configureContext时会使用到

org.springframework.boot.context.embedded.tomcat.TomcatEmbeddedServletContainerFactory

上面的内容只是SpringBoot提供的错误页处理方式,而不是SpringMVC的异常处理。

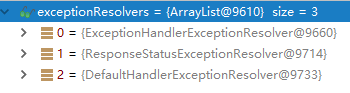
默认错误页处理流程：



org.springframework.web.servlet.view.BeanNameViewResolver

从Spring Context将viewName作为bean的名称，获取对应View类型的bean，然后返回。

SpringMVC默认会添加3个ExceptionHandlerResolver



当遇到异常时,是一个个进行处理的,能处理就处理,不能就让下一个进行处理

下面是SpringBoot提供的模板引擎：

*# Template availability providers***org.springframework.boot.autoconfigure.template.TemplateAvailabilityProvider**=**\  
org.springframework.boot.autoconfigure.freemarker.FreeMarkerTemplateAvailabilityProvider,\  
org.springframework.boot.autoconfigure.mustache.MustacheTemplateAvailabilityProvider,\  
org.springframework.boot.autoconfigure.groovy.template.GroovyTemplateAvailabilityProvider,\  
org.springframework.boot.autoconfigure.thymeleaf.ThymeleafTemplateAvailabilityProvider,\  
org.springframework.boot.autoconfigure.web.JspTemplateAvailabilityProvider**

只要引入响应的类库，则会启动对应的模板引擎，

**Thymeleaf模板引擎默认解析:classpath:/templates/\*.html文件**