Nimbus fw main theme:

/\*\*

\* Configures classloader to load resources from custom locations\*/

@Configuration

**public** **class** WebConfig **extends** WebMvcConfigurerAdapter {

**private** **static** **final** String TARGET\_FRONTEND = "file:./target/frontend/";

**private** **static** **final** String CLASSPATH\_STATIC = "classpath:./static/";

**public** **static** **final** Map<String, String[]> RESOURCE\_MAPPINGS = **new** HashMap<>();

**static** {

// The second argument of String[] is to tell where to look in a jar file.

RESOURCE\_MAPPINGS.put("/index.html", **new** String[] {"file:./target/frontend/index.html", CLASSPATH\_STATIC});

RESOURCE\_MAPPINGS.put("/systemjs\*", **new** String[] {TARGET\_FRONTEND, CLASSPATH\_STATIC});

RESOURCE\_MAPPINGS.put("/scripts/\*\*", **new** String[] {"file:./target/frontend/scripts/", "classpath:/static/scripts/"});

RESOURCE\_MAPPINGS.put("/styles/\*\*", **new** String[] {"file:./target/frontend/styles/", "classpath:/static/styles/"});

RESOURCE\_MAPPINGS.put("/js/\*\*", **new** String[] {"file:./target/frontend/js/", "classpath:/static/js/"});

RESOURCE\_MAPPINGS.put("/node\_modules/\*\*", **new** String[] {"file:./target/frontend/node\_modules/", "classpath:/static/node\_modules/"});

RESOURCE\_MAPPINGS.put("/utils/\*\*", **new** String[] {"file:./target/frontend/utils/", "classpath:/static/utils/"});

RESOURCE\_MAPPINGS.put("/webapp/\*\*", **new** String[] {"file:./target/frontend/webapp/", "classpath:/static/webapp/"});

RESOURCE\_MAPPINGS.put("/resources/\*\*", **new** String[] {CLASSPATH\_STATIC});

RESOURCE\_MAPPINGS.put("/vendor\*\*bundle\*js", **new** String[] {TARGET\_FRONTEND, CLASSPATH\_STATIC});

RESOURCE\_MAPPINGS.put("/vendor\*\*bundle\*js", **new** String[] {TARGET\_FRONTEND, CLASSPATH\_STATIC});

RESOURCE\_MAPPINGS.put("/polyfills\*\*bundle\*js", **new** String[] {TARGET\_FRONTEND, CLASSPATH\_STATIC});

RESOURCE\_MAPPINGS.put("/inline\*\*bundle\*js", **new** String[] {TARGET\_FRONTEND, CLASSPATH\_STATIC});

RESOURCE\_MAPPINGS.put("/scripts\*\*bundle\*js", **new** String[] {TARGET\_FRONTEND, CLASSPATH\_STATIC});

RESOURCE\_MAPPINGS.put("/main\*\*bundle\*js", **new** String[] {TARGET\_FRONTEND, CLASSPATH\_STATIC});

RESOURCE\_MAPPINGS.put("//styles\*\*bundle\*css", **new** String[] {TARGET\_FRONTEND, CLASSPATH\_STATIC});

RESOURCE\_MAPPINGS.put("/\*\*ttf", **new** String[] {TARGET\_FRONTEND, CLASSPATH\_STATIC});

RESOURCE\_MAPPINGS.put("/\*\*ttf", **new** String[] {TARGET\_FRONTEND, CLASSPATH\_STATIC});

RESOURCE\_MAPPINGS.put("/\*\*woff", **new** String[] {TARGET\_FRONTEND, CLASSPATH\_STATIC});

RESOURCE\_MAPPINGS.put("/\*\*woff2", **new** String[] {TARGET\_FRONTEND, CLASSPATH\_STATIC});

RESOURCE\_MAPPINGS.put("/browser-sync\*", **new** String[] {TARGET\_FRONTEND, CLASSPATH\_STATIC});

RESOURCE\_MAPPINGS.put("/updates\*", **new** String[] {TARGET\_FRONTEND, CLASSPATH\_STATIC});

RESOURCE\_MAPPINGS.put("/jslibs/\*\*", **new** String[] {TARGET\_FRONTEND, CLASSPATH\_STATIC});

}

--

@Configuration

**public** **class** DefaultProcessConfig {

@Bean

**public** ActivitiExpressionManager activitiExpressionManager(){

**return** **new** ActivitiExpressionManager();

}

@Bean

**public** BPMGateway bpmGateway(BeanResolverStrategy beanResolver){

**return** **new** ActivitiBPMGateway(beanResolver);

}

@Bean(name="default.\_new$execute?fn=\_initEntity")

**public** FunctionHandler<?, ?> defaultActionNewInitFunctionHandler(BeanResolverStrategy beanResolver){

**return** **new** DefaultActionNewInitEntityFunctionHandler<>(beanResolver);

}

@Bean(name="default.\_get$execute?fn=param")

**public** FunctionHandler<?, ?> defaultParamFunctionHandler(BeanResolverStrategy beanResolver){

**return** **new** DefaultParamFunctionHandler<>(beanResolver);

}

@Bean(name="default.\_nav$execute?fn=default")

**public** PageIdEchoNavHandler<?> pageIdEchoNavHandler(){

**return** **new** PageIdEchoNavHandler<>();

}

@Bean(name="default.\_process$execute?fn=\_set")

**public** SetFunctionHandler<?,?> setFunctionHandler(){

**return** **new** SetFunctionHandler<>();

}

@Bean(name="default.\_process$execute?fn=\_update")

**public** UpdateFunctionHandler<?,?> updateFunctionHandler(){

**return** **new** UpdateFunctionHandler<>();

}

@Bean(name="default.\_process$execute?fn=\_setByRule")

**public** FunctionHandler<?,?> setByRuleFunctionHandler(){

**return** **new** SetByRuleFunctionalHandler<>();

}

@Bean(name="default.\_process$execute?fn=\_add")

**public** AddFunctionHandler<?,?> addFunctionHandler(){

**return** **new** AddFunctionHandler<>();

}

@Bean(name="default.\_process$execute?fn=\_bpm")

**public** StatelessBPMFunctionHanlder<?,?> statelessBPMFunctionHanlder(BeanResolverStrategy beanResolver){

**return** **new** StatelessBPMFunctionHanlder<>(beanResolver);

}

@Bean(name="expressionEvaluator")

**public** ExpressionEvaluator expressionEvaluator(BeanResolverStrategy beanResolver){

**return** **new** SpelExpressionEvaluator();

}

@Bean(name="commandExecutorTaskDelegate")

**public** CommandExecutorTaskDelegate commandExecutorTaskDelegate(BeanResolverStrategy beanResolver){

**return** **new** CommandExecutorTaskDelegate(beanResolver);

}

@Bean(name="default.\_search$execute?fn=lookup")

**public** FunctionHandler<?, ?> lookupFunctionHandler(){

**return** **new** DefaultSearchFunctionHandlerLookup<>();

}

@Bean(name="default.\_search$execute?fn=example")

**public** FunctionHandler<?, ?> exampleFunctionHandler(){

**return** **new** DefaultSearchFunctionHandlerExample<>();

}

@Bean(name="default.\_search$execute?fn=query")

**public** FunctionHandler<?, ?> queryFunctionHandler(){

**return** **new** DefaultSearchFunctionHandlerQuery<>();

}

@Bean(name="default.\_process$execute?fn=\_eval")

**public** EvalFunctionHandler<?,?> evalFunctionHandler(ExpressionManager expressionManager){

**return** **new** EvalFunctionHandler(expressionManager);

}

}

public interface **BeanResolver**

A bean resolver can be registered with the evaluation context and will kick in for @myBeanName and &myBeanName expressions. The & variant syntax allows access to the factory bean where relevant.

|  |  |
| --- | --- |
| **Modifier and Type** | **Method and Description** |
| java.lang.Object | [**resolve**](https://docs.spring.io/spring-framework/docs/current/javadoc-api/org/springframework/expression/BeanResolver.html#resolve-org.springframework.expression.EvaluationContext-java.lang.String-)(**[EvaluationContext](https://docs.spring.io/spring-framework/docs/current/javadoc-api/org/springframework/expression/EvaluationContext.html" \o "interface in org.springframework.expression)** context, java.lang.String beanName)  Look up the named bean and return it. |

**resolve**

java.lang.Object resolve([EvaluationContext](https://docs.spring.io/spring-framework/docs/current/javadoc-api/org/springframework/expression/EvaluationContext.html" \o "interface in org.springframework.expression) context,

java.lang.String beanName)

throws [AccessException](https://docs.spring.io/spring-framework/docs/current/javadoc-api/org/springframework/expression/AccessException.html)

Look up the named bean and return it. If attempting to access a factory bean the name will have a & prefix.

**Parameters:**

context - the current evaluation context

beanName - the name of the bean to lookup

**Returns:**

an object representing the bean

**Throws:**

[AccessException](https://docs.spring.io/spring-framework/docs/current/javadoc-api/org/springframework/expression/AccessException.html) - if there is an unexpected problem resolving the named bean