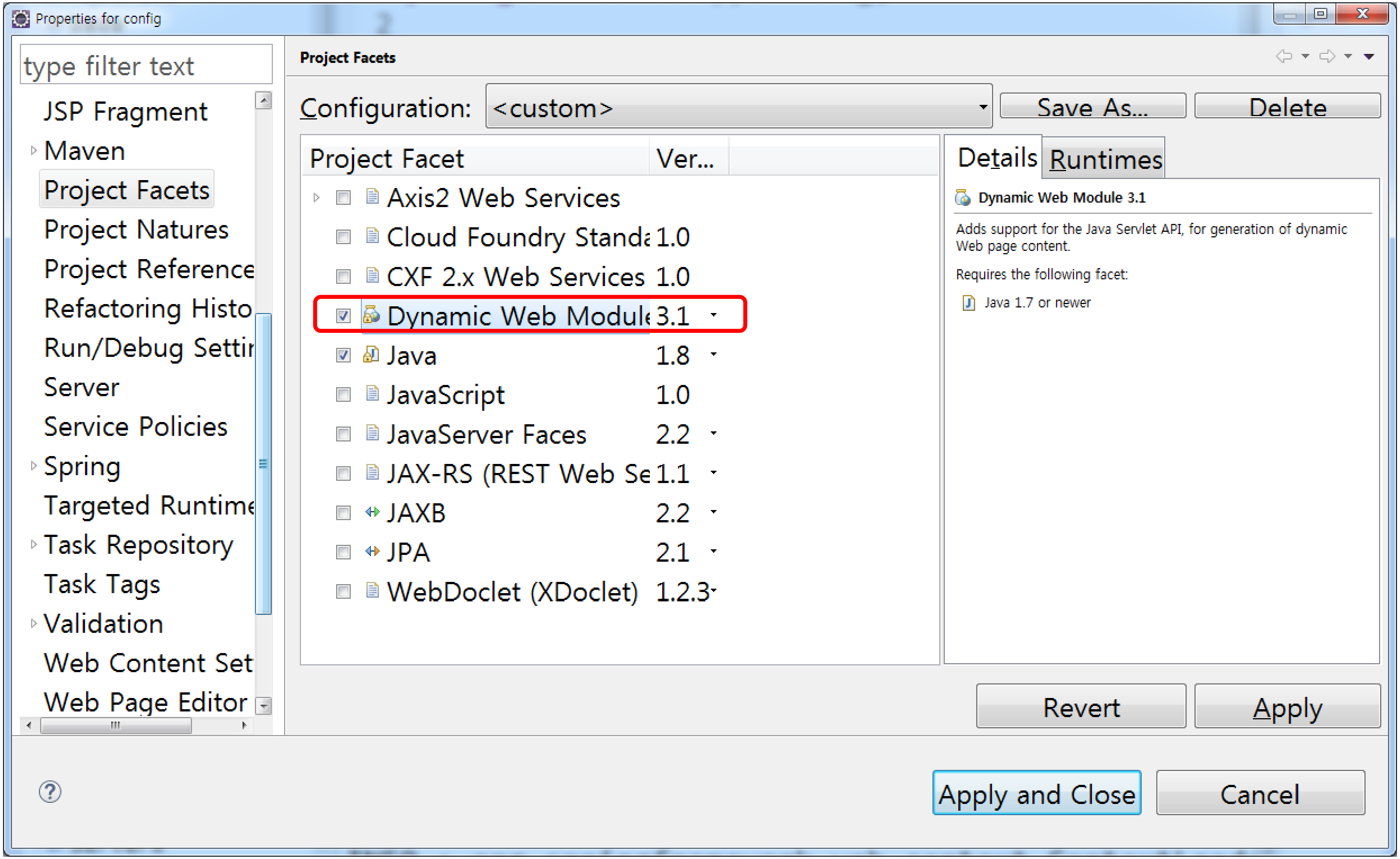
스프링 MVC 자바 설정

❖ Servlet 3.0 부터 web.xml 대신 자바 설정 클래스 사용가능

- 웹 서버 기동시
  - AbstractAnnotationConfigDispatcherServletInitializer 자식 클래스 또는 WebApplicationInitializer 인터페이스 구현 클래스 검색
  - 존재한다면 해당 클래스를 설정으로 사용



Properties for config

type filter text

- JSP Fragment
- Maven
- Project Facets
- Project Natures
- Project Reference
- Refactoring Histo
- Run/Debug Settir
- Server
- Service Policies
- Spring
- Targeted Runtime
- Task Repository
- Task Tags
- Validation
- Web Content Set
- Web Page Editor

Project Facets

Configuration: <custom>

Save As... Delete

Project Facet	Ver...
□ Axis2 Web Services	
□ Cloud Foundry Standa	1.0
□ CXF 2.x Web Services	1.0
☑ Dynamic Web Modul	3.1
☑ Java	1.8
□ JavaScript	1.0
□ JavaServer Faces	2.2
□ JAX-RS (REST Web Se	1.1
□ JAXB	2.2
□ JPA	2.1
□ WebDoclet (XDoclet)	1.2.3

Details | Runtimes

Dynamic Web Module 3.1

Adds support for the Java Servlet API, for generation of dynamic Web page content.

Requires the following facet:

Java 1.7 or newer

Revert Apply

Apply and Close Cancel

❖ pom.xml

```
    :

    <!-- Servlet 3.1.0 / JSP 2.3.1 -->
    <dependency>
       <groupId>javax.servlet</groupId>
       <artifactId>javax.servlet-api</artifactId>
       <version>3.1.0</version>
       <scope>provided</scope>
    </dependency>

    <dependency>
       <groupId>javax.servlet.jsp</groupId>
       <artifactId>javax.servlet.jsp-api</artifactId>
       <version>2.3.1</version>
       <scope>provided</scope>
    </dependency>
```

❖ pom.xml

```
         :
   </dependencies>
   <build>
      <plugins>
         <plugin>
            <groupId>org.apache.maven.plugins</groupId>
            <artifactId>maven-war-plugin</artifactId>
            <version>2.6</version>
            <configuration>
               <failOnMissingWebXml>false</failOnMissingWebXml>
            </configuration>
         </plugin>
            :
      </plugins>
   </build>
</project>
```

❖ 자바 설정 클래스

- 기존 설정 xml 파일 삭제
  - web.xml, servlet-context.xml, root-context.xml

- edu.iot.app.config 패키지에 자바 설정 클래스 작성
  - WebAppInitializer
    - web.xml 대체
  - WebConfig
    - servlet-context.xml 대체
  - RootConfig
    - root-context.xml 대체

❖ RootConfig.java

```
@Configuration
@ComponentScan(basePackages= {"edu"},
      excludeFilters= {
         @Filter(type=FilterType.ANNOTATION, value=EnableWebMvc.class)

      })
public class RootConfig {

}
```

❖ WebConfig.java

```
@Configuration
@EnableWebMvc    // <annotation-driven /> 설정
@ComponentScan("edu.iot")
public class WebConfig extends WebMvcConfigurerAdapter {

   // 기본 ViewResolver 설정, 메서드 명이 빈의 id가 됨
   @Bean
   public ViewResolver viewResolver() {
      InternalResourceViewResolver resolver =
                                new InternalResourceViewResolver();
      resolver.setPrefix("/WEB-INF/views/");
      resolver.setSuffix(".jsp");
      resolver.setOrder(10);
      return resolver;
   }
```

❖ WebConfig.java

```
   // 정적 자원 경로 설정
   // <resources mapping="/resources/**" location="/resources/"/> 설정
   @Override
   public void addResourceHandlers(
                             ResourceHandlerRegistry registry) {

      registry.addResourceHandler("/resources/**")
            .addResourceLocations("/resources/")
            .setCachePeriod(0);
   }

}
```

❖ WebConfig.java (web.xml 대체)

```
public class WebAppInitializer
   extends AbstractAnnotationConfigDispatcherServletInitializer{

   @Override
   protected Class<?>[] getRootConfigClasses() {
      // 전역 수준의 설정 클래스
      return new Class<?>[] {RootConfig.class};
   }

   @Override
   protected Class<?>[] getServletConfigClasses() {
      // DispatcherServlet의 설정 클래스
      return new Class<?>[] {WebConfig.class};
   }

   @Override
   protected String[] getServletMappings() {
      // DispatcherServlet을 "/"에 맵핑
      return new String[] {"/"};
   }
```

❖ WebConfig.java

```
   // UTF-8 문자 인코딩 필터 설정
    @Override
    protected Filter[] getServletFilters() {

        CharacterEncodingFilter characterEncodingFilter =
                 new CharacterEncodingFilter();
        characterEncodingFilter.setEncoding("UTF-8");
        characterEncodingFilter.setForceEncoding(true);
        return new Filter[] { characterEncodingFilter};
    }
}
```

스프링 MVC 자바 설정
- Tiles 설정 -

❖ TilesConfig.java

```java
@Configuration
public class TilesConfig {

   @Bean
   public TilesConfigurer tilesConfigurer() {
      TilesConfigurer conf = new TilesConfigurer();
      conf.setDefinitions("/WEB-INF/tiles/tiles-layout.xml");
      return conf;
   }

   @Bean
   public ViewResolver tilesViewResolver() {
      UrlBasedViewResolver resolver = new UrlBasedViewResolver();
      resolver.setViewClass(TilesView.class);
      resolver.setOrder(2);
      return resolver;
   }
}
```

❖ WebConfig.java

```
@Configuration
@EnableWebMvc
@EnableTransactionManagement
@ComponentScan("edu.iot")
@Import({ TilesConfig.class })
public class WebConfig extends WebMvcConfigurerAdapter {
   :
}
```

스프링 MVC 자바 설정
- MutipartResolver -

❖ WebConfig.java

```
@Configuration
@EnableWebMvc
@EnableTransactionManagement
@ComponentScan("edu.iot")
public class WebConfig extends WebMvcConfigurerAdapter {
   :

   @Bean
   public MultipartResolver multipartResolver() {
      return new CommonsMultipartResolver();
   }

}
```

스프링 MVC 자바 설정
- BeanNameViewResolver -

❖ WebConfig.java

```
@Configuration
@EnableWebMvc
@EnableTransactionManagement
@ComponentScan("edu.iot")
public class WebConfig extends WebMvcConfigurerAdapter {

   @Bean
   public ViewResolver beanNameViewResolver() {
      BeanNameViewResolver resolver = new BeanNameViewResolver();
      resolver.setOrder(1);
      return resolver;
   }

   :
}
```

스프링 MVC 자바 설정
- 데이터베이스 접속 설정-

❖ DatabaseConfig.java

```
@Configuration
@PropertySource("classpath:database.properties")
@EnableTransactionManagement
public class DatabaseConfig {
    @Autowired private Environment env;

    @Bean(destroyMethod = "close")
    public BasicDataSource dataSource() {
        BasicDataSource dataSource = new BasicDataSource();
        dataSource.setDriverClassName(
                             env.getProperty("database.driver"));
        dataSource.setUrl(env.getProperty("database.url"));
        dataSource.setUsername(env.getProperty("database.username"));
        dataSource.setPassword(env.getProperty("database.password"));
        return dataSource;
    }
```

❖ DatabaseConfig.java

```
    @Bean
    public PlatformTransactionManager transactionManager() {
        return new DataSourceTransactionManager(dataSource());
    }
}
```

❖ RootConfig.java

```
@Configuration
@ComponentScan(basePackages= {"edu"},
      excludeFilters= {
         @Filter(type=FilterType.ANNOTATION, value=EnableWebMvc.class)

   })
@Import({
      DatabaseConfig.class
   })
public class RootConfig {

}
```

❖ WebConfig.java

```
@Configuration
@EnableWebMvc
@EnableTransactionManagement
@ComponentScan("edu.iot")
public class WebConfig extends WebMvcConfigurerAdapter {
    :
}
```

스프링 MVC 자바 설정
- MyBatis 설정 -

❖ MybatisConfig.java

```java
@Configuration
public class MybatisConfig {

   @Bean
   public SqlSessionFactory sqlSessionFactory(DataSource dataSource,
            ApplicationContext applicationContext) throws Exception {

      org.apache.ibatis.session.Configuration configuration
                    = new org.apache.ibatis.session.Configuration();
      configuration.setCacheEnabled(false);
      configuration.setMapUnderscoreToCamelCase(true);
      configuration.setUseGeneratedKeys(true);

      final SqlSessionFactoryBean sessionFactory =
                                     new SqlSessionFactoryBean();
      sessionFactory.setDataSource(dataSource);
      sessionFactory.setConfiguration(configuration);
```

❖ MybatisConfig.java

```
      // sessionFactory.setConfigLocation(
      //             applicationContext.getResource(
      //                      "classpath:config/mybatis-config.xml"));

      sessionFactory.setTypeAliasesPackage("edu.iot.common.model");
      sessionFactory.setMapperLocations(
                    applicationContext.getResources(
                             "classpath:mapper/**/*.xml"));

      return sessionFactory.getObject();
   }

   @Bean
   public MapperScannerConfigurer mapperScannerConfigurer() {
      MapperScannerConfigurer conf = new MapperScannerConfigurer();
      conf.setBasePackage("edu.iot.**.dao");
      return conf;
   }
}
```

❖ RootConfig.java

```
@Configuration
@ComponentScan(basePackages= {"edu"},
      excludeFilters= {
         @Filter(type=FilterType.ANNOTATION, value=EnableWebMvc.class)

   })
@Import({
      DatabaseConfig.class,
      MybatisConfig.class
   })
public class RootConfig {

}
```

스프링 MVC 자바 설정
- 인터셉터 설정 -

❖ WebConfig.java

```
:
public class WebConfig extends WebMvcConfigurerAdapter {
      :
   @Bean
   public LoginInterceptor loginInterceptor() {
      return new LoginInterceptor();
   }

   @Bean
   public AdminInterceptor adminInterceptor() {
      return new AdminInterceptor();
   }
```

❖ WebConfig.java

```
   @Override
   public void addInterceptors(InterceptorRegistry registry) {

      registry.addInterceptor(loginInterceptor())
            .addPathPatterns("/admin/**")
            .addPathPatterns("/gallery/**")
            .addPathPatterns("/member/**")
            .addPathPatterns("/board/**")
            .excludePathPatterns("/member/avata/*")
            .excludePathPatterns("/board/list")
            .excludePathPatterns("/board/view/**");

      registry.addInterceptor(adminInterceptor())
            .addPathPatterns("/admin/**");
   }

}
```

스프링 MVC 자바 설정
- 예외 설정 -

❖ WebConfig.java

```
@Bean
HandlerExceptionResolver errorHandler() {
   SimpleMappingExceptionResolver s =
                  new SimpleMappingExceptionResolver();

   // 예외명과 뷰이름 맵핑
   Properties p = new Properties();
   p.setProperty(DataAccessException.class.getName(),
                  "error/db_error");
   s.setExceptionMappings(p);
   // 디폴트 에러 뷰 이름 설정
   s.setDefaultErrorView("error/error");

   // 뷰 이름 별 응답 코드 설정
   // s.addStatusCode("error/db_error", 500);
   // s.setDefaultStatusCode(500);
   return s;
}
```