链滴

spring 相关工具类

作者：Pleuvoir

原文链接：https://ld246.com/article/1522496478985

来源网站：链滴

记录Spring相关工具类。

---

1. 方便的获取容器中的Bean

```
/**
 * {@link org.springframework.context.ApplicationContext} 包装类，方便获取容器中的Bean
 * <p>注意：需要在容器中注册后方可使用
 * @author pleuvoir
 *
 */
public class ApplicationContextWrap implements ApplicationContextAware {

    private static ApplicationContext applicationContext;

    @Override
    public void setApplicationContext(ApplicationContext applicationContext)
            throws BeansException {
        ApplicationContextWrap.applicationContext = applicationContext;
    }

    public static <T> T getBean(Class<T> clazz) {
        return applicationContext.getBean(clazz);
    }

    @SuppressWarnings("unchecked")
    public static <T> T getBean(String beanName) {
        return (T) applicationContext.getBean(beanName);
    }
```

```
}
```

因为实现了ApplicationContextAware接口，注册方式略有不同：

```
<bean name="applicationContextWrap" class="com.pleuvoir.spring.ApplicationContextWrap"
/>
```

或者直接使用注解：

```
@Bean(name = "applicationContextWrap")
public ApplicationContextWrap applicationContextWrap(){
    return new ApplicationContextWrap();
}
```

2. Environment包装类 扩展了属性类型

```
/**
 * {@link org.springframework.core.env.Environment} 包装类，提供了更为简便的数据获取方法
 * @author pleuvoir
 *
 */
public class EnvironmentWrap implements EnvironmentAware {

    private Environment environment;

    @Override
    public void setEnvironment(Environment environment) {
        this.environment = environment;
    }

    /**
     * 获取Integer类型的数据，若数据格式不可转换为Integer类型，将抛出异常{@link NumberFor
atException}
     * @param key
     * @return
     */
    public Integer getInteger(String key) {
        String val = environment.getProperty(key);
        return Integer.valueOf(val);
    }

    /**
     * 获取Integer类型的数据，并提供默认的值，若数据格式不可转换为Integer类型或者为null时，
会返回默认值
     * @param key
     * @param defaultVal
     * @return
     */
    public Integer getInteger(String key, Integer defaultVal) {
        Integer i = null;
        try {
            i = getInteger(key);
        } catch (NumberFormatException e) {
```

```
        }
        if (i == null) {
            i = defaultVal;
        }
        return i;
    }

    /**
     * 获取Long类型的数据，若数据格式不可转换为Long类型，将抛出异常{@link NumberFormatEx
eption}
     * @param key
     * @return
     */
    public Long getLong(String key) {
        String val = environment.getProperty(key);
        return Long.valueOf(val);
    }

    /**
     * 获取Long类型的数据，并提供默认的值，若数据格式不可转换为Long类型或者为null时，将会
回默认值
     * @param key
     * @param defaultVal
     * @return
     */
    public Long getLong(String key, Long defaultVal) {
        Long i = null;
        try {
            i = getLong(key);
        } catch (NumberFormatException e) {
        }
        if (i == null) {
            i = defaultVal;
        }
        return i;
    }

    /**
     * 获取Double类型的数据，若数据格式不可转换为Double类型，将抛出异常{@link NumberFor
atException}
     * @param key
     * @return
     */
    public Double getDouble(String key) {
        String val = environment.getProperty(key);
        return Double.valueOf(val);
    }

    /**
     * 获取Double类型的数据，并提供默认的值，若数据格式不可转换为Double类型或者为null时，
会返回默认值
     * @param key
     * @param defaultVal
     * @return
```

```
     */
    public Double getDouble(String key, Double defaultVal) {
        Double i = null;
        try {
            i = getDouble(key);
        } catch (NumberFormatException e) {
        }
        if (i == null) {
            i = defaultVal;
        }
        return i;
    }

    /**
     * 获取BigDecimal类型的数据，若数据格式不可转换为BigDecimal类型，将抛出异常
     * {@link NumberFormatException}
     * @param key
     * @return
     */
    public BigDecimal getBigDecimal(String key) {
        String val = environment.getProperty(key);
        return new BigDecimal(val);
    }

    /**
     * 获取BigDecimal类型的数据，并提供默认的值，若数据格式不可转换为BigDecimal类型或者为n
ll时，将会返回默认值
     * @param key
     * @param defaultVal
     * @return
     */
    public BigDecimal getBigDecimal(String key, BigDecimal defaultVal) {
        BigDecimal i = null;
        try {
            i = getBigDecimal(key);
        } catch (NumberFormatException e) {
        }
        if (i == null) {
            i = defaultVal;
        }
        return i;
    }

    /**
     * 获取Boolean类型的数据，若数据格式不可转换为Boolean类型，将返回false
     * @param key
     * @return
     */
    public Boolean getBoolean(String key) {
        String val = environment.getProperty(key);
        return Boolean.valueOf(val);
    }

    /**
```

```
     * 获取String类型的数据
     * @param key
     * @return
     */
    public String getString(String key) {
        return environment.getProperty(key);
    }

}
```

同上：

```
<bean name="environmentWrap" class="com.pleuvoir.spring.EnvironmentWrap" />
```

```
<br>
```

// TODO 其他生命周期提供的扩展点。