Expression support for defining bean definitions

SpEL expressions can be used with XML or annotation-based configuration metadata for defining BeanDefinitions. In both cases the syntax to define the expression is of the form **#{ <expression string> }**

XML based configuration

A property or constructor-arg value can be set using expressions as shown below.

```
<bean id="numberGuess" class="org.spring.samples.NumberGuess">
    <property name="randomNumber" value="#{ T(java.lang.Math).random() * 100.0
}"/>

    <!-- other properties -->
</bean>
```

The variable systemProperties is predefined, so you can use it in your expressions as shown below. Note that you do not have to prefix the predefined variable with the # symbol in this context.

```
<bean id="taxCalculator" class="org.spring.samples.TaxCalculator">
    <property name="defaultLocale" value="#{ systemProperties[user.region] }"/>

    <!-- other properties -->
</bean>
```

You can also refer to other bean properties by name, for example.

```
<bean id="numberGuess" class="org.spring.samples.NumberGuess">
    <property name="randomNumber" value="{ T(java.lang.Math).random() * 100.0
}"/>

    <!-- other properties -->
</bean>

<bean id="shapeGuess" class="org.spring.samples.ShapeGuess">
    <property name="initialShapeSeed" value="{ numberGuess.randomNumber }"/>

    <!-- other properties -->
</bean>
```

Annotation-based configuration

The @Value annotation can be placed on fields, methods and method/constructor parameters to specify a default value.

Here is an example to set the default value of a field variable.

```
public static class FieldValueTestBean

    @Value("#{ systemProperties[user.region] }")
    private String defaultLocale;

    public void setDefaultLocale(String defaultLocale) {
        this.defaultLocale = defaultLocale;
    }

    public String getDefaultLocale() {
        return this.defaultLocale;
    }

}
```

The equivalent but on a property setter method is shown below.

```
public static class PropertyValueTestBean

    private String defaultLocale;

    @Value("#{ systemProperties[user.region] }")
    public void setDefaultLocale(String defaultLocale) {
        this.defaultLocale = defaultLocale;
    }

    public String getDefaultLocale() {
        return this.defaultLocale;
    }

}
```

Autowired methods and constructors can also use the @Value annotation.

```
public class SimpleMovieLister {

    private MovieFinder movieFinder;
    private String defaultLocale;

  @Autowired
    public void configure(MovieFinder movieFinder,
            @Value("#{ systemProperties[user.region] }") String defaultLocale) {
        this.movieFinder = movieFinder;
        this.defaultLocale = defaultLocale;
    }
    // ...
}
public class MovieRecommender {

    private String defaultLocale;

    private CustomerPreferenceDao customerPreferenceDao;

  @Autowired
    public MovieRecommender(CustomerPreferenceDao customerPreferenceDao,
            @Value("#{systemProperties[user.country]}") String defaultLocale) {
        this.customerPreferenceDao = customerPreferenceDao;
        this.defaultLocale = defaultLocale;
    }

    // ...
}
```