Translators Quick Guide
V0.9

Translators Quick Guide

Index

1 What is i18x? 3
2 Simple Placeholder Tags 3
3 Placeholder Attributes 3
4 Special Character Handling 3
5 White Spaces and Space-Characters 4
6 Open- and Close-Placeholder-Tags 4
7 Conditional Rules using IF-Attributes 5
8 Expression Attribute 5
9 Named Expressions by Definition Tag 6
10 Number Formats by Definition Tag 6

1 What is i18x?

i18x is a universal translation format to translate text phrases including placeholders, translation rules, unit conversions and value formats. The translation rules also allows the handling of grammar- and syntax-rules.

2 Simple Placeholder Tags

Placeholders are defined in i18x by XML like tags. A single tag is represented in this way:

```
<placeholdername/>
```

The beginning of a tag is a less-than-character followed by the placeholders name and closed by a slash-character and greater-than-character. This form is called a closed tag.

When the application called the translation function with a text phrase, the named placeholders will be passed to the translation function.

Example:

i18x Text Phrase

```
Hello <firstname/> <lastname/>!
```

Placeholders

```
{“firstname”:”Peter”,“lastname”:”Temple”}
```

Output

```
Hello Peter Temple!
```

3 Placeholder Attributes

Placeholders can have one or more attributes to define special handlings of it. The general form for placeholder attributes is:

```
<placeholdername attribute=”attributevalue”/>
```

There will be many attributes defined in i18x. Here an example of using the attribute *repeat*:

i18x Text Phrase

```
We need <whatweneed repeat=”3”/>!
```

Placeholders

```
{“whatweneed”:”bananas”}
```

Output

```
We need bananasbananasbananas!
```

4 Special Character Handling

Because some characters are used to define the tags and rules, this characters can not used inside a text which have to output. E.g. to use less-than-characters a special tag have to used.

Example:

i18x Text Phrase

```
<number1/> <lt/> <number2/>!
```

Placeholders

```
{“number1”:5,“number2”:8}
```

Output

```
5 < 8
```

Auto Tag	Output Value
<lt/>	<
<gt/>	>
<space/>	(space character)

5 White Spaces and Space-Characters

The translation routine will remove all white spaces like tabulator- or newline-. Also multiple space-characters will be reduced to single spaces.

Example:

i18x Text Phrase

```
\t\t\nHello      <firstname/>    \n<lastname/>!
```

Placeholders

```
{“firstname”:”Peter”,“lastname”:”Temple”}
```

Output

```
Hello Peter Temple!
```

To create an output with multiple spaces use the space-tag:

i18x Text Phrase

```
\t\t\nHello      <firstname/><space repeat=”3”/>\n<lastname/>!
```

Placeholders

```
{“firstname”:”Peter”,“lastname”:”Temple”}
```

Output

```
Hello Peter   Temple!
```

6 Open- and Close-Placeholder-Tags

Up to here we only use single tags. Another form of tags is to defined placeholder-ranges by using placeholder-open- and placeholder-close-tags.

Placeholder-Open-Tag

```
<placeholdername>
```

Placeholder-Close-Tag

```
</placeholdername>
```

It is important to close all tags and keep tag hierarchy. Never use open tag which not follows by a close tag. In this way tags can be define hier….

Unlike single-tags, open- and close-tags does not create outputs by itself. The output is created by the inside content.

7 Conditional Rules using IF-Attributes

In i18x it is possible to create conditional rules depending on placeholder values.

i18x Text Phrase

```
<noofresults if="0">No results.</noofresults>
<noofresults if="1">One result.</noofresults>
<noofresults if="1+"><noofresults/> results.</noofresults>
```

Placeholders

```
1.{“noofresults”:0}
2.{“noofresults”:1}
3.{“noofresults”:2}
4.{“noofresults”:10}
```

Outputs

```
1. No results.
2. One result.
3. 2 results.
4. 10 results.
```

8 Expression Attribute

With an expression Attribute the placeholder value can be changed by a formula. This is very helpful to convert different units used in different languages or cultures.

i18x Text Phrase

```
Today´s temperature is <celsiustemperature expression="=x*1.8+32"/>°F.
```

Placeholders

```
{“celsiustemperature”:20}
```

Output

```
Today´s temperature is 68°F.
```

The expression attribute value have to begin with an equal sign following by a statement, where *x* is the current value of the placeholder which includes this attribute. The new placeholder value is given by the result of the statement.

This elements can be used inside a statement:

Element	Description
=	Indicates an expression statement. Have to be the first character.
x	The value of the placeholder which includes the expression attribute.
+	Addition of values.
-	Subtraction of values.
*	Multiplication of values.
/	Division of values.
(*y*)	Bracketing of a statement *y*.
round(*y*)	Round function of statement *y*.

ceil(***y***)	Ceil function of statement *y*.
floor(***y***)	Floor function of statement *y*.
abs(***y***)	Absolute function of statement *y*.

9 Named Expressions by Definition Tag

With the definition tag a named expression can be define:

i18x Text Phrase

```
<definition type="expression" name="CelsiusToFahrenheit" expresseion="x*1.8+32"/>
Today´s temperature is <temperature expression="CelsiusToFahrenheit"/>°F.
```

Placeholders

```
{“celsiustemperature”:20}
```

Output

```
Today´s temperature is 68°F.
```

These definition-attributes are available in an expression definition:

Attribute	Description
name	The name of the new expression definition.
type	Have to be a value of “expression” to define a named expression.
expression	The statement of the expression.

The expression is available in the global namespace of i18x, so a definition which is defined in any text phrase can be accessed from all other text phrases. In most cases the application programmers will defines extra separated text phrases to define all global named expressions which then can used by all other text phrases.

10 Number Formats by Definition Tag

The definition tag with a type-attribute value of “format”. To define complex number formatting

Example of a simple two decimal number format:

Example of a simple date format:

Attribut Name	Description
i	A string with the integer part of the given value.
$i_0..i_n$	Characters of the integer part of the given value, where n is the position of the character. Counting starts at the least significant position.
f	Expression of the definition.
$f_0..f_n$	Only format definition: Character replacement of value
r	Only format definition: Character to fill up integer leading empty placeholders.

$r_0..r_n$	
x	The given value.
xa	The absolute value of the given value.
day	The gregorian calendar day of the month of users local date based on given value*.
month	The gregorian calendar month of users local date based on given value*.
year	The gregorian calendar year of users local date based on given value*.
weekday	The gregorian calendar weekday (0..6 / Monday to Sunday) of users local date based on given value*.
hour	Hour of users local date based on given value*.
minute	Minute of users local date based on given value*.
second	Second of users local date based on given value*.
millisecond	Millisecond of users local date based on given value*.
weekofyear	Week number of the year of users local date based on given value*.
dayofyear	Day number of the year of users local date based on given value*.