****

**i18x**

Translators Quick Guide

V0.9

Index

[1 What is i18x? 3](#_Toc379728996)

[2 Simple Placeholder Tags 3](#_Toc379728997)

[3 Placeholder Attributes 3](#_Toc379728998)

[4 Special Character Handling 4](#_Toc379728999)

[5 White Spaces and Space-Characters 4](#_Toc379729000)

[6 Open- and Close-Placeholder-Tags 4](#_Toc379729001)

[7 Conditional Rules using IF-Attributes 5](#_Toc379729002)

[8 Expression Attribute 6](#_Toc379729003)

[9 Named Expressions by Definition Tag 7](#_Toc379729004)

[10 Number Formats by Definition Tag 7](#_Toc379729005)

[11 Online Resources 8](#_Toc379729006)

[12 Release History 9](#_Toc379729007)

[13 Legal Information 10](#_Toc379729008)

[13.1 MIT Licence Terms 10](#_Toc379729009)

# 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.

# 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!

# 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 sense, which is used to give information about the sense of a text pharse :

i18x Text Phrase

Title is ‘<mytitle sense=”title of the application”/>’.

Placeholders

{“mytitle”:”MyApp”}

Output

Title is ‘MyApp’.

Another example of using the attribute repeat, which produces multiple outputs of the given placeholder-tag:

i18x Text Phrase

We need <whatweneed repeat=”3”/>!

Placeholders

{“whatweneed”:”bananas”}

Output

We need bananasbananasbananas!

# 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) |

# 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!

# 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. Open- und close-tags are especially useful in combination with the if-Attribute.

# 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.

The if-Attribute value contains the condition to make the content between the open- and the close-tag visible or not.

|  |  |
| --- | --- |
| Element | Description |
| x | Returns true, if placeholder value is the same as x. |
| x= | Returns true, if placeholder value is the same as x. |
| x+ | Returns true, if placeholder value is greater than x. |
| x- | Returns true, if placeholder value is less than x. |
| x~ | Returns true, if placeholder value is not equal to x. |
| | | Character to separate multiple conditions. |

A more complex example:

i18x Text Phrase (English-US)

Dear<space/>

<age if="14|14+">

<gender if="1">Mr.</gender><space/>

<gender if="0">\

<married if="0">Ms.</married>

<married if="1">Mrs.</married>

</gender><space/>

<firstnames/> <lastname/>

</age>

<age if="14-">

<firstnames/>

</age>,<newline/>

i18x Text Phrase (German-DE)

<age if="14|14+">

Sehr<space/>

<gender if="0">geehrte Frau</gender>

<gender if="1">geehrter Herr</gender><space/>

<firstnames/> <lastname/>

</age>

<age if="14-">

<gender if="0">Liebe</gender>

<gender if="1">Lieber</gender><space/>

<firstnames/>

</age>,<newline/>

Placeholders

1. {"age":15,"married":1,"gender":1,"firstnames":"Michael","lastname":"Miller"}
2. {"age":5,"married":0,"gender":1,"firstnames":"John","lastname":"Smith"}
3. {"age":12,"married":0,"gender":0,"firstnames":"Tina","lastname":"Jonson"}
4. {"age":30,"married":1,"gender":0,"firstnames":"Jenny","lastname":"Taylor"}
5. {"age":30,"married":0,"gender":0,"firstnames":"Maria","lastname":"Temple"}

Outputs (English-US)

1. Dear Mr. Michael Miller,
2. Dear John,
3. Dear Tina,
4. Dear Mrs. Jenny Taylor,
5. Dear Ms. Maria Temple,

Outputs (German-DE)

1. Sehr geehrter Herr Michael Miller,
2. Lieber John,
3. Liebe Tina,
4. Sehr geehrte Frau Jenny Taylor,
5. Sehr geehrte Frau Maria Temple,

# 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. |

# Named Expressions by Definition Tag

With the definition tag a named expression can be define. The defined expression is then available in the global namespace of i18x, so a definition which is defined in any text phrase can be used in all other text phrases of the i18x-application.

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 | Has to be a value of “expression” to define a named expression. |
| expression | The statement of the expression. (inline expression statement only) |

# 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:

The following predefined tags are available inside a format definition:

|  |  |
| --- | --- |
| Auto Tag Name | Description |
| i | A string with the integer part of the given value. |
| i0..in | 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. |
| f0..fn | Only format definition: Character replacement of value |
| r | Only format definition: Character to fill up integer leading empty placeholders. |
| r0..rn |  |
| 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\*. |

# Online Resources

* <http://www.dongleware.com/i18x/validator.html>  
  A i18x validation and testing website. There are also more i18x text phrase examples available.
* I18x at GitHub

# Release History

|  |  |  |
| --- | --- | --- |
| Date | Version | Release Notes |
| 2014-02-09 | 0.9.0 | First version. |

# Legal Information

This document is published under [MIT](http://en.wikipedia.org/wiki/MIT_License) license and are for free and commercial use.

Written by Meinolf Amekudzi.

©2014



Dongleware Verlags GmbH

Am Güterbahnhof 18

69181 Leimen

Germany

FON +49 6224 17 52 40

USt-ID: DE 152 563 028

Steuernummer: FAHD 32492/11803

Handelsregister: Amtsgericht Mannheim HRB 334574

Sitz der Gesellschaft: Leimen

Geschäftsführer: Meinolf Amekudzi

[www.dongleware.com](http://www.dongleware.com)

## MIT Licence Terms

Permission is hereby granted, free of charge, to any person obtaining a copy of this documentation, to deal without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of it, and to permit persons to whom the document is furnished to do so, subject to the following conditions:  
  
The above copyright notice and this permission notice shall be included in all copies or substantial portions of the document.  
  
THE DOCUMENT IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE DOCUMENTATION OR THE USE OR OTHER DEALINGS IN THE DOCUMENT.