GENESYS™

Genesys Knowledge Management User Guide

Functions

4/26/2025

Functions

When composing formulas, you can use many built-in functions. *Functions* are predefined formulas that perform calculations using values, called *arguments,* which you supply. To use a function, write its name, followed by an opening parenthesis, the arguments for the function separated by commas, and a closing parenthesis.

Function arguments may be of any data type, although individual functions may place restrictions on their arguments. Function arguments may be constants or formulas. The `Length` function, for example, takes a single string argument and returns its length in characters. This formula evaluates to 13:

`<$Length("Hello, world!")$>`

As another example, the`Date` function takes individual date components (year, month, day, and so on), and constructs a date/time value. The formula below evaluates to `1965-11-23 09:03:10`:

`<$Date(1965, 11, 23, 9, 3, 10)$>`

Functions may act as arguments to other functions. The `WeekdayName` function takes a single date/time argument and returns the day of the week as a string. The formula below evaluates to `Tuesday`:

`<$WeekdayName(Date(1965, 11, 23, 9, 3, 10))$>`

This formula evaluates to 7:

`<$Length(WeekdayName(Date(1965, 11, 23, 9, 3, 10))$>)`

Detailed descriptions of all available functions are provided in the Genesys eServices Field Codes Reference Manual.

Using Objects

All object/property pairs are also available in the `Variables` drop-down menu in the Knowledge Manager `Field Code Editor`.
Object properties can be of any data type. `Agent.FullName`, for example, is a string, but `Interaction.DateCreated` is a date/time.
The data type of an object property can even be another object. For example, `Contact.EmailAddresses` yields another object called a `ContactEmailAddressList`. In cases such as this, you can access the properties of the resulting object by entering a period (.), followed by the property name, just as before. For example, the formula below evaluates to the number of e-mail addresses assigned to the contact:

`<$Contact.EmailAddresses.Count$>`

Some object properties require arguments just as functions do. For these properties, write the

arguments, enclosed in parentheses after the property name, just as before.

For example, the ContactEmailAddressList object has a property named `Exists,` which you can use to test whether a particular e-mail address is assigned to a contact. The data type of this property is Boolean (true/false), and it takes one argument, the e-mail address to test. For example:

```
<$Contact.EmailAddresses.Exists("samd@acme.com")$>
```

Detailed descriptions of all objects and their properties are provided in the Genesys eServices Field Codes Reference Manual.