



Formula Class

Contains methods to get a builder for creating a formula instance and to update all formula fields on the input SObjects.

Namespace

[System](#)

Usage

Use the Formula class in conjunction with the [FormulaBuilder](#) and [FormulaInstance](#) classes in the [FormulaEval](#) namespace.

See [Formula Evaluation in Apex](#).

Example

This example creates a formula instance using `Formula.builder()` and the [FormulaBuilder](#) methods.

```
FormulaEval.FormulaInstance ff = Formula.builder()
    .withType(Account.SObjectType)
    .withReturnType(FormulaEval.FormulaReturnType.STRING)
    .withFormula('{!name} ({!website})')
    .parseAsTemplate(true)
    .build();
```

- [Formula Methods](#)

Formula Methods

The following are methods for `Formula`.

- [builder\(\)](#)
Creates an instance of `FormulaBuilder` for configuring the formula with formula expression, context type, and output data type as inputs.
- [recalculateFormulas\(subjects\)](#)
Updates (recalculates) all formula fields on the input SObjects.

builder()

Creates an instance of `FormulaBuilder` for configuring the formula with formula expression, context type, and output data type as inputs.

Signature

```
public static formulaeval.FormulaBuilder builder()
```

Return Value

Type: [FormulaEval.FormulaBuilder](#)



```
public static List<System.FormulaRecalcResult> recalculateFormulas(List<SObject> subjects)
```

Parameters

subjects

Type: List<SObject>

List of sObjects whose formula fields are to be recalculated.

Return Value

Type: List<FormulaRecalcResult Class>

Usage

Recalculate formula fields on new or queried SObjects. If all data is present on the SObjects, SOQL limits are not affected. If the data required to evaluate a formula field is missing, that data is retrieved and limits are changed accordingly.

The new formula values are stored in the SObjects themselves and overwrite previous values of formula fields.

Example

```
Account a = new Account();
a.Name = 'Salesforce';
a.BillingCity = 'San Francisco';
List<Account> accounts = new List<Account>{a};

List<FormulaRecalcResult> results = Formula.recalculateFormulas(accounts);
System.assert(results[0].isSuccess());
// Option 1
System.debug('New value: ' + accounts[0].get('My_Formula_Field__c'));
// Option 2
System.debug('New value: ' + results[0].getSObject().get('My_Formula_Field__c'));
```

DID THIS ARTICLE SOLVE YOUR ISSUE?

Let us know so we can improve!

[Share your feedback](#)

DEVELOPER CENTERS

Heroku
MuleSoft
Tableau
Commerce Cloud
Lightning Design System

POPULAR RESOURCES

Documentation
Component Library
APIs
Trailhead
Sample Apps

COMMUNITY

Trailblazer Community
Events and Calendar
Partner Community
Blog
Salesforce Admins





© Copyright 2025 Salesforce, Inc. [All rights reserved.](#) Various trademarks held by their respective owners. Salesforce, Inc.
Salesforce Tower, 415 Mission Street, 3rd Floor, San Francisco, CA 94105, United States

[Privacy Information](#) [Terms of Service](#) [Legal](#) [Use of Cookies](#) [Trust](#) [Cookie Preferences](#)

☒ [Your Privacy Choices](#) [Responsible Disclosure](#) [Contact](#)