Search documentation

Appian

Function Recipes

Appian 7.6

This page lists a collection of function recipes you can use throughout the Appian application. Similar to a cookbook, it shows the individual ingredients that go into creating an expression using Appian functions, the order in which to combine them, and tips on when to use them or modify for preference.

Toggle Us

The expressions listed below are not the only way to accomplish each desired outcome. This page is intended to teach you methods for creating expressions using Appian functions that are practical and efficient, as well as provide working expressions you can implement as is. Just as with cookir recipes, you may want to alter the values in each recipe to accommodate your specific requirements.

The function recipes are categorized by the type of output. Each recipe is titled with the question in mind, "What do you want to do?"

NOTE: For function recipes that require a rule as an Ingredient, create the rule as listed before implementing the function recipe.

If there are additional recipes you would like to see included in this list or questions about the existing ones, feel free to post a message on Forum.

Tu

Re

Ot

Date/Time Results

Convert an Appian Date into a US Calendar Format

Use Case

You want to take Appian Dates, such as **02/05/2011**, and convert them into a US Calendar format (**February 5, 2011**).

Ingredients

- text()

Input

- date (Date)

Expression

- `text(pv!date,"mmm dd,yyyy")`

Convert Appian Dates into a Formal Format

Use Case

You want to take an Appian Date of **02/05/2011** and convert it into a formal format displaying **5th day of February 2011**.

Ingredients

- text()
- format: **ddd**
- format: **mmmm**
- format: **yyyy**

Input

- Datetime (datetime)

Expression

- `text(pv!dateTime, "ddd") & " day of " & text(!dateTime, "mmmm, yyyy")`

Retrieve Next Anniversary Date

Use Case

You want to retrieve next year's anniversary date based on a start date, such as an employee's hire date of **02/05/2011**.

Ingredients

- if()
- today()
- date()
- day()
- month()
- year()

Input

- start (date)

Expression

- `if(AND(month(ri!start )<=month(today()), day(ri!start)<=day(today())), date(1+year(today()), month(ri!start), day(ri!start)), date(year(today()), month(ri`

Start a Timer One Minute After a Process Starts

Use Case

You want to set a timer to start exactly one minute after a process starts, as compared to automatically.

Where to Enter

Configure this in the "Delay until the date and time specified by this expression" on the Setup tab for the Timer Event.

Ingredients

- pp!start_time
- minute()

Expression

- `pp!start_time + minute(1)`

NOTE: To change the time difference from 1 minute to more, modify the 1 as desired.

Add Ten Minutes to a DateTime Value

Use Case

You want to add ten minutes to a datetime value saved as pv!datetime.

Ingredients

- datetime
- intervalds()

Expression

- `pv!datetime + intervalds(0,10,0)`

NOTE: To change the number of minutes added to the datetime value, modify the 10 as desired.

Determine if a Date-Time Value Falls Within Working Hours

Use Case

Users are able to enter any date-time value, but you want to perform an extra validation after users submit the form to determine if a date-time value (saved as **ri!dt**) falls between your company's working hours of 6:00 AM and 5:00 PM.

Ingredients

- or()
- gmt()
- datetime()
- year()
- month()
- day()

Input

- dt (DateTime)

Expression

- `or(ri!dt>gmt(datetime(year(ri!dt),month(ri!dt),day(ri!dt),17, 0, 0)),ri!dt6, 0, 0)))`

NOTE: To change the working hours, modify the 17, 0, 0 and 6, 0, 0 as desired.

- The idea is to work with one consistent time scale, which is done here by the use of timestamps rather than both hours and minutes. This functio constructs new timestamps for the boundaries matching the input.

Return the Next Workday

Use Case

You want to return the date of the next workday based on today's date.

Ingredients

- today()
- workday()

Input

- days (Number)

Expression

- `workday(today(),1))`

NOTE: To change the number of days you want to occur after the current date before determining the next workday, modify the 1 value as desired.

Display the Number of Days Before a Specific Date

Use Case

You want to tell users how many business days are left before a deal closes, if the deal has already closed, or if the deal is closing today.

Ingredients

- if()
- networkdays()
- today()

Input

- closeDate (Date)

Expression

- `if(networkdays(today(), ri!closeDate)<=0, "This deal has closed.", if(networkdays(today(), ri!closeDate)=1, "This deal will close at the end of today.", "Th`

NOTE: To change the text that displays, modify the red text as desired. To base the number of days on a system calendar, replace any instance of `networkdays(today(), ri!closeDate)` with `calworkdays(today(), ri!closeDate, "companyCalendar")` where `companyCalendar` is the name of your system calendar. To include all days (including weekends), replace any instance of `networkdays(today(), ri!closeDate)` with `tointeger(ri!closeDate - today())` .

See also: calworkdays() and tointeger()

Convert an XML String into a Value of Type Datetime

Use Case

You have a localized XML string representing a date and time and need to convert it to a value of type Datetime in the GMT timezone.

Ingredients

- with()
- split()

- left()
- right()
- datetime()
- gmt()

Input

- dateText (Text)

Expression

```
=with(
  local!fullArray: split(ri!dateText, "T"),
  local!dateArray: split(local!fullArray[1], "-"),
  local!timeArray: split(left(local!fullArray[2], 12), ":"),
  local!smsArray: split(local!timeArray[3], "."),
  local!timeZone: "GMT" & right(local!fullArray[2], 6),
  local!rawDate: datetime(
    local!dateArray[1],
    local!dateArray[2],
    local!dateArray[3],
    local!timeArray[1],
    local!timeArray[2],
    local!smsArray[1],
    local!smsArray[2]
  ),
  gmt(
    local!rawDate,
    local!timeZone
 )
)
```

yields `9/25/2013 1:50 AM GMT+00:00` where `ri!dateText` = `2013-09-24T19:50:24.192-06:00`

Number Results

Return the Number of Active Employees in Your Application

Use Case

You want to quickly display how many employee records exist in your application.

Ingredients

- topaginginfo()
- totalCount

Expression

- `myQueryRule(topaginginfo(1,0)).totalCount`

NOTE: When using a batchsize of 0 and attempting to only retrieve a total count based on a query, the function detailed above is better to use than `count(myQueryRule())`. If you used `count(myQueryRule())`, the system would run the main query; whereas using `.totalcount`, the system only executes
count query against the database.

Return the Total Number of Revisions Made to a Document

Use Case

You want to show the number of times a document has had a new version saved.

Ingredients

- document()

Input

- doc (Document)

Expression

- `document(ri!doc,"totalNumberOfVersions")-1)`

Add a List of Integer Values with Another List of Integer Values

Use Case

You want to take a list of integers (for example, 10,20,30) and add the corresponding integers from a different list (for example, 50,70,90) to each original value in the first list.

- Where `pv!x = {10,20,30}` and `pv!y = {40,50,60}`.

Ingredients

- processVariable

Input

- pv!x (Integer Array)
- pv!y (Integer Array)

Expression

- `pv!x+pv!y` yields `{50,70,90}`

Return a Random Integer in a Range

Use Case

You want to return a random integer between two integers, inclusive.

Ingredients

- rand()
- tointeger()

Input

- min (Number (Integer))
- max (Number (Integer))

Expression

- `ri!min + tointeger(rand() * (ri!max - ri!min))`

Text Results

Truncate Text After 50 Characters

Use Case

You want to truncate the remaining part of a text value once it surpasses 50 characters by replacing the excess text with an ellipsis (**...**).

Ingredients

- if()
- len()

Input

- text (Text)

Expression

- `if(len(ri!text)>50, left(ri!text, 50)&"...", ri!text)`

NOTE: To change the amount of characters allowed before truncating, modify the 50 value as desired.

Display the Full Name of a User

Use Case: You want to display the full name of a user despite knowing only the username.

Ingredients

- user()

Input

- user (User)

Expression

- `user(ri!user,"firstName") & " " & user(ri!user,"lastName")`

Capitalize the First Letter of Every Word in a Text Value

Use Case

You want to convert a text value to have initial caps in order to use it as an announcement.

Ingredients

- proper()

Input

- Employee out of office(Text)

Expression

- `proper("Employee out of office")` yields `Employee Out Of Office`

NOTE: To produce initial caps for your own text, modify the Employee out of office text value as desired.

Populate a User's Name on a Form

Use Case

You want to create a form that populates the user's full name when he/she opens the form.

This recipe requires the following value be enabled on the Other tab of the User Input Task node:

- **Refresh default values every time the task form is viewed.**

Ingredients

- loggedInUser()
- user()

Input

- property (Text)

Expression

- `user(loggedinuser(),"firstName") & " " & user(loggedInUser(),"lastName")`

NOTE: To change the user information that is populated on the form (for example, to show his/her displayName), modify the property (firstName) value(s) as desired. To display just the first name, only enter the first half of the recipe (`user(loggedinuser(),"firstName")`).

Strip Characters Not Supported in Folder Names from a PV of Type Text

User Case

You are using a process variable (saved as `ri!text`) to populate folder names for the Create Folder Smart Service and want to remove any characters from the process variable values that are not supported in folder names in order to avoid possible errors.

Ingredients

- stripwith()

Input

- text (text)

Expression

- `stripwith(ri!text,"</span>,/,;,:,?,',',>,<,*,],[")`

NOTE: To change which characters to remove from the process variable value, modify the characters in red.

Duplicate Results

Create a List with Ten Occurrences of the Number 1

Use Case

You want to create a list that displays the value "1" ten times.

Ingredients

- repeat()

Input

- Number (Integer)

Expression

- `repeat(10, 1)`

NOTE To change the number of times "1" is listed, modify the 10 number value as desired.

Create a List with Ten Occurrences of the Same Word

Use Case

You want to create a list that displays the word "Hello" ten times.

Ingredients

- repeat()

Input

- Number (Integer)
- Hello (Text)

Expression

- `repeat(10, "Hello")`

NOTE: To change the number of times "Hello" is listed, modify the 10 number value as desired. To change the word that is repeated, modify the Hello text value as desired.

Array Results

Remove all Values in an Array

Use Case

You want to remove all values in an array so it is considered empty.

Ingredients

- ldrop()
- count()

Input

- Array (Array)

Expression

- `ldrop(array, count(array))`

NOTE: Instead of the word array, enter the pv name or array reference as needed.

Return All Notes Created by a User that Contain a Specific Word

Use Case

From an array of notes, you want to only return those where the note creator is `john.smith` and it contains the text `fixed`.

Ingredients

- filter()
- **Rule**: `isExpectedNote`
  - **Rule Definition**: `and(ri!note.creator="john.smith", like(ri!note.text, "*fixed*"))`

    NOTE: To change the name or text by which to search for, modify the john.smith and fixed values as desired.
  - **Rule Ingredients**: and() and like()
  - **Rule Input**: note (Any Type)

```
Note
|- text (Text)
|- creator (Text)
```

Expression

- `filter(rule!isExpectedNote, pv!notes)`

Return All Notes Created by a User that Contain Any of Three Different Words

Use Case

From an array of notes, you want to only return those where the note creator is `john.smith` and it contains the text `fixed`, `done`, or `closed`.

Ingredients

- filter()
- pp!initiator
- **Rule**: `isNoteByCreatorAndHasText`
  - **Rule Definition**: `and(ri!note.creator="john.smith", contains(ri!hasText, ri!note.text))`

    NOTE: To change the name by which to search for, modify the john.smith value as desired.
  - **Rule Ingredients**: and(), user(), and like()
  - **Rule Inputs**: `note` (Any Type), `creator` (User - Single), and `hasText` (Text - Single)

```
Note
|- text (Text)
|- creator (Text)
```

  NOTE: Rule Input `note` is the array to operate on and `hasText` is the context parameter.

- **Constant**: `TEXT_LIST` (text - multiple)
    - Values should include `fixed`, `done`, and `closed`.

NOTE: To change the text by which to search for, modify the values for the constant.

Input

- notes (Array)

Expression

- `filter(rule!isNoteByCreatorAndHasText, pv!notes, pp!initiator, cons!TEXT_LIST)`

Return All Notes from an Array that Adhere to Two Non-Array Context Parameters

Use Case

From an array of notes, you want to only return those that meet a note creator parameter and a text parameter.

Ingredients

- filter()
- pp!initiator
- **Rule**: `isNoteByCreatorAndHasText`
    - **Rule Definition**: `and(ri!note.creator=user(ri!creator, "username"), like(ri!note.text, "*" & ri!hasText & "*"))`
    - **Rule Ingredients**: and(), user(), and like()
    - **Rule Inputs**: `note` (Any Type), `creator` (User - Single), and `hasText` (Text - Single)

```
Note
|- text (Text)
|- creator (Text)
```

  NOTE: Rule Input `note` is the array to operate on, `creator` is the first context parameter, and `hasText` is the second context parameter

- **Constant**: `TEXT_LIST` (text - multiple)
    - Values should include `fixed`, `done`, and `closed`.

  NOTE: To change the text by which to search for, modify the values for the constant.

Input

- notes (Array)

Expression

- `filter(rule!isNoteByCreatorAndHasText, pv!notes, pp!initiator, cons!SOME_TEXT)`

Repeat Each Item in an Array Based on the Values of a Different Array

Use Case

You want to repeat each item in an array (for example, `pv!textList`) a certain number of times as defined by the values in a different array (for example, `pv!frequencyList`).

- Where `pv!textList = {"a", "b", "c"}` and `pv!frequencyList = {2, 3, 1}`, the output would be `{"a", "a", "b", "b", "b", "c"}`.

Ingredients

- apply()
- repeat()
- merge()

Inputs

- textList (Array)
- frequencyList (Array)

Expression

- `apply(fn!repeat, merge(pv!frequencyList, pv!textList))`

NOTE: To change which array process variables to use, modify the textList value to the array process variable that should determine the text to repea and modify the frequencyList value to the array process variable that should determine how many times each text value in the aforementioned array should be repeated.

Extract a List from an EntityData Array of the Data Written to a Single Entity

Use Case

You want to extract the list of data written to an entity (for example, Opportunity) based on the output generated by using the Write to Multiple Data Store Entities Smart Service to write to three entities: Opportunity, Contact, and Line Item.

See also: Write to Multiple Data Store Entities Smart Service

NOTE: The explanation of the configuration is longer than the solution. If you configure the Write to Multiple Data Store Entities Smart Service to upda data for three entities (Opportunity, Contact, and Line Item), each entity may show up more than once in the EntityData input array. Consequently, tl output of the smart service ("Stored Values") would contain a dynamic number of elements for each EntityData.

To get the list of all Opportunities that were updated by the smart service, you need to append every Data field where the entity is equal to Opportunit

Ingredients

- apply()
- StoredValue (EntityData)
- OPPORTUNITY_ENTITY (Data Store Entity)
- **Rule**: `dataForEntity`
  - **Rule Definition**: `if(ri!entityData.Entity=ri!entity, ri!entityData.Data, {})`
  - **Rule Ingredient**: if()
  - **Rule Inputs**: `entityData` (Any Type) and `entity` (Any Type)

    NOTE: `entityData` is the array to operate on and `entity` is the context parameter

Expression

- `apply(rule!dataForEntity, ac!StoredValues, pv!OPPORTUNITY_ENTITY)`

NOTE: To change the entity by which to pull the list of data from, modify the OPPORTUNITY_ENTITY value as desired.

Sort an Array in Process

Use Case

You want to sort an array of values from a process variable as part of a process.

Ingredients

- todatasubset()
- pagingInfoNodeInput (PagingInfo) - node input in the process you want to sort the array
- processVariableArray (Text) - process variable that holds the array of values you want to sort

You'll need to create a node input (for example `pagingInfoNodeInput`) of type PagingInfo with the `sort.field` value equal to the field you want to sort the array by.

Expression

Use this expression as the expression for a node output in the same node you created the `pagingInfoNodeInput`.

- `todatasubset(pv!processVariableArray, ac!pagingInfoNodeInput).data`

Boolean Results

Determine if Items in an Array are Contained Within Another Array

Use Case

You want to see if any of the items in an array (saved as `pv!originalArray`) are contained within another array (saved as `pv!compareArray`).

Ingredients

- contains()
- any()

Expression Using Partial Evaluation

- `any(fn!contains(ri!compareArray, _), ri!originalArray)`

See also: Partial Evaluation of Rules and Type Constructors

Expression NOT Using Partial Evaluation

Additional Ingredient

- **Rule**: `myRule`
  - **Rule Definition**: `contains(ri!array, ri!item)`
  - **Rule Inputs**: `array` and `item`

New Expression

- `any(ri!myrule, ri!originalArray, ri!compareArray)`

The intermediate rule is necessary because `contains()` takes the original array as the first parameter and the array to check as the second parameter, whereas the looping function syntax sends the parameters in the opposite order.

For example, to generate a list of persons from two parallel arrays with person first names and last names:

```
apply(
  type!Person(firstName:_, lastName:_),
  merge(pv!firstNames, pv!lastNames)
)
```

Link Results

Create an Internal Dashboard Link that Refreshes the Dashboard Content

Use Case

You want to create a link that refreshes the dashboard content when a user clicks it.

Ingredients

- linktoprocessdashboardinternal()
- pp!id

Input

- Refresh (text)

Expression

- `linktoprocessdashboardinternal(pp!id, "Refresh")`

NOTE: To change the title of the refresh link, modify the Refresh text value as desired.

Matching Results

Return a CDT from an Array that Contains a Field Value Matching Another Value

Use Case

You have an array of department CDTs saved to a process variable called `pv!departments` each containing a field called `Id`, and you want to retrieve tl
CDT with an `Id` value matching the value of a process variable.

Ingredients

- displayvalue()
- Process Variable: `pv!departmentId` (Integer)
- Process Variable: `departments` (CDT Array)
- Custom Data Type:

```
department (Text)
   |- Id (Integer)
   |- Address (Text)
```

Expression

- `displayvalue(pv!departmentId, pv!departments.Id, pv!departments, "none")`

NOTE: To change the value that displays if the `pv!departmentId` doesn't match any `department.Id` values, modify the none text value as desired.