

# Package ‘AnVILWorkflow’

April 5, 2024

**Title** Run workflows implemented in Terra/AnVIL workspace

**Version** 1.2.0

**Date** 2023-5-30

**Description** The AnVIL is a cloud computing resource developed in part by the National Human Genome Research Institute. The main cloud-based genomics platform deployed by the AnVIL project is Terra. The AnVILWorkflow package allows remote access to Terra implemented workflows, enabling end-user to utilize Terra/ AnVIL provided resources - such as data, workflows, and flexible/scalable computing resources - through the conventional R functions.

**Depends** R (>= 4.2.0),

**Imports** utils, AnVIL, httr, methods, jsonlite

**Suggests** knitr, tibble, BiocStyle

**License** Artistic-2.0

**biocViews** Infrastructure, Software

**Encoding** UTF-8

**LazyData** true

**VignetteBuilder** knitr

**Roxygen** list(markdown = TRUE)

**RoxygenNote** 7.2.3

**BugReports** <https://github.com/shbrief/AnVILWorkflow/issues>

**git\_url** <https://git.bioconductor.org/packages/AnVILWorkflow>

**git\_branch** RELEASE\_3\_18

**git\_last\_commit** 9f58777

**git\_last\_commit\_date** 2023-10-24

**Repository** Bioconductor 3.18

**Date/Publication** 2024-04-05

**Author** Sehyun Oh [aut, cre] (<<https://orcid.org/0000-0002-9490-3061>>)

**Maintainer** Sehyun Oh <shbrief@gmail.com>

**R topics documented:**

<code>.biobakery_currentInput</code>	2
<code>.get_workflow_fullname</code>	3
<code>.get_workspace_fullname</code>	3
<code>.nonMetadataOutputs</code>	4
<code>.stop_quietly</code>	5
<code>availableAnalysis</code>	5
<code>cloneWorkspace</code>	6
<code>currentInput</code>	7
<code>findInputName</code>	8
<code>getDashboard</code>	8
<code>getOutput</code>	9
<code>monitorWorkflow</code>	10
<code>runWorkflow</code>	11
<code>setCloudEnv</code>	12
<code>stopWorkflow</code>	12
<code>updateInput</code>	13

<b>Index</b>	<b>15</b>
--------------	-----------

---

`.biobakery_currentInput`

*Check the current input arguments*

---

**Description**

Check the current input arguments

**Usage**

```
.biobakery_currentInput(config)
```

**Arguments**

<code>config</code>	Terra workflow configuration. Output from the <code>avworkflow_configuration_get()</code> function.
---------------------	---

**Value**

A list length of two, including `inputListPath` and `inputFilePath`.

### Examples

```
library(AnVIL)
if (gcloud_exists() && nzchar(avworkspace_name())) {
  config <- avworkflow_configuration_get(
    workflow_namespace = "mtx_workflow_biobakery_version3",
    workflow_name = "mtx_workflow_biobakery_version3",
    workspace_namespace = "waldronlab-terra-rstudio",
    workspace_name = "mtx_workflow_biobakery_version3_template")
  biobakery_inputs <- .biobakery_currentInput(config)
}
```

---

.get\_workflow\_fullname

*Get the workflow namespace and name*

---

### Description

Use this internally when [setCloudEnv](#) is already run.

### Usage

```
.get_workflow_fullname(workspaceName, workflowName = NULL)
```

### Arguments

**workspaceName** A character. Name of the workspace to use.

**workflowName** A character. Name of the workflow to run. If a single workflow is available under the selected workspace, this function will check the input of that workflow under the default (NULL). If there are multiple workflows available, you should specify the workflow.

### Value

A character of workflow\_namespace/workflow\_name

---

.get\_workspace\_fullname

*Get the fullname of the workspace*

---

### Description

Get the fullname of the workspace

**Usage**

```
.get_workspace_fullname(workspaceName)
```

**Arguments**

`workspaceName` Character(1). Name of the template workspace name you want to clone. You can provide name or namespace/name.

**Value**

Character(1) of workspaceNamespace/workspaceName

**Examples**

```
library(AnVIL)
if (gcloud_exists() && nzchar(avworkspace_name())) {
  .get_workspace_fullname(workspaceName = "Bioconductor-Workflow-DESeq2")
}
```

---

`.nonMetadataOutputs` *Subset to non-metadata output files*

---

**Description**

Subset to non-metadata output files

**Usage**

```
.nonMetadataOutputs(workflowOutputs)
```

**Arguments**

`workflowOutputs`

A data frame of workflow outputs with four columns: file, workflow, task, and path. Returned value from [avworkflow\\_files](#).

**Value**

A character vector containing the names of non-metadata output files

---

.stop\_quietly      *Stop the execution without error messages*

---

**Description**

Stop the execution without error messages

**Usage**

```
.stop_quietly()
```

**Value**

Stop the function call without warning/error messages.

---

availableAnalysis      *Find the available analysis*

---

**Description**

This function shows the available analyses and the brief descriptions of them.

**Usage**

```
availableAnalysis(curatedOnly = TRUE, keyword = NULL)
```

**Arguments**

curatedOnly      Default is TRUE, returning only workspaces that offer simplified input configuration by this package. If it is set to FALSE, all the workspaces

keyword      Default is NULL. When this argument is provided as a character(1), it will return only the workspaces containing the keyword and the user has an access to.

**Value**

A data frame. The analysis columns shows the name of the available analyses, which is the required input (analysis argument) for the functions implemented in AnVILWorkflow package.

**Examples**

```
library(AnVIL)
if (gcloud_exists() && nzchar(avworkspace_name())) {availableAnalysis()}
```

cloneWorkspace

*Clone template workspace*

---

**Description**

This function makes your own copy of the existing workspace, selected through `templateName` or `analysis`. Your copied/cloned workspace name will be `workspaceName` and any computing cost will be charged to the billing linked to your `billingProjectName`. You should provide at least one argument `templateName` or `analysis`.

**Usage**

```
cloneWorkspace(  
  workspaceName,  
  templateName = "",  
  analysis = NULL,  
  accountEmail = gcloud_account(),  
  billingProjectName = gcloud_project()  
)
```

**Arguments**

<code>workspaceName</code>	Name of the workspace you are creating
<code>templateName</code>	Character(1). Name of the template workspace name you want to clone. You can provide name or namespace/name.
<code>analysis</code>	Character(1). Name of the analysis you want to clone it's workspace. The list of available analyses can be found using <a href="#">availableAnalysis</a> .
<code>accountEmail</code>	Character(1). Email linked to Terra account
<code>billingProjectName</code>	Character(1). Name of the billing project

**Value**

Name of the cloned workspace

**Examples**

```
library(AnVIL)  
if (gcloud_exists() && nzchar(avworkspace_name())) {  
  cloneWorkspace(workspaceName = "salmon",  
                 templateName = "Bioconductor-Workflow-DESeq2")  
}
```

---

currentInput	<i>Check the current input arguments</i>
--------------	--

---

### Description

Check the current input arguments

### Usage

```
currentInput(  
  workspaceName,  
  workflowName = NULL,  
  requiredInputOnly = TRUE,  
  analysis = NULL  
)
```

### Arguments

workspaceName	Name of the workspace
workflowName	Name of the workflow to run. If a single workflow is available under the specified workspace, this function will check the input of that workflow under the default (NULL). If there are multiple workflows available, you should specify the workflow.
requiredInputOnly	Under the default (TRUE), only the required inputs are returned.
analysis	If specified, only the minimally required inputs for a given workflow will be returned.

### Value

A data.frame for the inputs defined in a workflow configuration.

### Examples

```
library(AnVIL)  
if (gcloud_exists() && nzchar(avworkspace_name())) {  
  currentInput(workspaceName = "Bioconductor-Workflow-DESeq2")  
}
```

findInputName            *Find the root entity name*

---

### Description

Find the root entity name

### Usage

```
findInputName(workspaceName, rootEntity = "", nameOnly = TRUE)
```

### Arguments

workspaceName	Name of the workspace
rootEntity	A character. Type of root entity for Terra's data model. For example, participant, participant_set, sample, etc.
nameOnly	Under the default (TRUE), only the names of a given root entity type will be returned.

### Value

A character vector of input names under the given root entity.

### Examples

```
library(AnVIL)
if (gcloud_exists() && nzchar(avworkspace_name())) {
  .findInputName(
    workspaceName = "Bioconductor-Workflow-DESeq2",
    rootEntity = "participant_set")
}
```

---

getDashboard            *Print out Dashboard contents*

---

### Description

This function prints out the Dashboard contents of the target workspace. You can provide either workspaceName or analysis. If both values are provided, this function will use workspaceName argument over analysis argument.

### Usage

```
getDashboard(workspaceName = "", analysis = NULL)
```



**Arguments**

workspaceName	The name of the workspace you want to get the overview provided through the Dashboard.
analysis	The name of the analysis use want to check the Dashboard of. The list of available analyses can be found with availableAnalysis().

**Value**

The last modified date as a message, followed by the Dashboard contents from the target workspace.

**Examples**

```
library(AnVIL)
if (gcloud_exists() && nzchar(avworkspace_name())) {
  getDashboard(analysis = "salmon")
  getDashboard(workspaceName = "Bioconductor-Workflow-DESeq2")
}
```

---

getOutput

*Download output files from Terra*


---

**Description**

Download output files from Terra

**Usage**

```
getOutput(
  workspaceName,
  submissionId = NULL,
  keyword = NULL,
  dest_dir = ".",
  dry = TRUE
)
```

**Arguments**

workspaceName	Name of the workspace
submissionId	Submission Id. If it's not provided, the most recent submission id with the 'succeeded' status will be used.
keyword	A character string containing a regular expression to be matched in the output file name. Under the default NULL, all the outputs from the workflow, including log files, will be returned.
dest_dir	Path to the directory where downloaded files are saved
dry	To download the output data, set dry = FALSE.

**Value**

If "dry=TRUE", this function will return a data frame with two columns named 'filename' and 'name'.

- filename: Name of the actual output files.
- name: Name of the output defined in your workflow script. This is how configuration refers the outputs.

**Examples**

```
library(AnVIL)
if (gcloud_exists() && nzchar(avworkspace_name())) {
  getOutput(workspaceName = "Bioconductor-Workflow-DESeq2")
}
```

---

monitorWorkflow	<i>Check the status of submitted jobs</i>
-----------------	---

---

**Description**

Check the status of submitted jobs

**Usage**

```
monitorWorkflow(workspaceName)
```

**Arguments**

workspaceName    Character(1). Name of the workspace

**Value**

A tibble summarizing submitted workflow jobs. Contains information such as submission Id, submission date, and submission status.

**Examples**

```
library(AnVIL)
if (gcloud_exists() && nzchar(avworkspace_name())) {
  monitorWorkflow(workspaceName = "Bioconductor-Workflow-DESeq2")
}
```

---

runWorkflow	<i>Launch Terra workflow</i>
-------------	------------------------------

---

## Description

Launch Terra workflow

## Usage

```
runWorkflow(  
  workspaceName,  
  workflowName = NULL,  
  useCallCache = TRUE,  
  inputName = NULL  
)
```

## Arguments

workspaceName	Name of the workspace that contains the workflow(s) you want to launch.
workflowName	Name of the workflow to run.
useCallCache	A logical. Under the default condition (TRUE), call cache will be used.
inputName	Name of you input entity. If the workflow is using Terra's data model, this is required. The available entities can be found using the <code>findInputName</code> function.

## Value

This function will print out whether the call for workflow launching was successful or not.

## Examples

```
library(AnVIL)  
if (gcloud_exists() && nzchar(avworkspace_name())) {  
  if ("salmon" %in% avworkspaces())$name  
  runWorkflow(workspaceName = "salmon")  
}
```

---

setCloudEnv	<i>Setup Google Cloud Account and Project</i>
-------------	---

---

**Description**

Setup Google Cloud Account and Project

**Usage**

```
setCloudEnv(
  accountEmail = gcloud_account(),
  billingProjectName = gcloud_project(),
  message = TRUE
)
```

**Arguments**

accountEmail	Character(1). Email linked to your Terra account.
billingProjectName	Character(1). Name of the billing project, which is the gcloud account.
message	Under the default (TRUE), this function will print out Google Cloud Account and Billing Project set in the working environment

**Value**

Terra/AnVIL working environment - Google Cloud billing account and the billing project name - will be printed out.

**Examples**

```
library(AnVIL)
if (gcloud_exists()) {
  setCloudEnv()
}
```

---

stopWorkflow	<i>Abort submitted job</i>
--------------	----------------------------

---

**Description**

Abort submitted job

**Usage**

```
stopWorkflow(workspaceName, submissionId = NULL, dry = TRUE)
```

**Arguments**

workspaceName	Name of the workspace
submissionId	A character. Submission ID you want to abort. You can find the submission id using <code>monitorSubmission</code> function. If it is not defined, the most recent submission will be aborted.
dry	Logical(1) when TRUE (default), report the consequences but do not perform the action requested. When FALSE, perform the action.

**Value**

This function will print out whether the call for workflow abortion was successful or not. In case it was unsuccessful, the diagnosis will be suggested as a part of the message.

**Examples**

```
library(AnVIL)
if (gcloud_exists() && nzchar(avworkspace_name())) {
  if ("salmon" %in% avworkspaces()$name)
    stopWorkflow(workspaceName = "salmon")
}
```

---

updateInput	<i>Update the input</i>
-------------	-------------------------

---

**Description**

Update the input

**Usage**

```
updateInput(
  workspaceName,
  inputs,
  workflowName = NULL,
  dry = TRUE,
  verbose = TRUE
)
```

**Arguments**

workspaceName	Name of the workspace
inputs	A tibble containing new input values. Provide the modify version of the current input table, which is a returned value from <code>currentInput</code> function.

workflowName	Name of the workflow to run. If a single workflow is available under the specified workspace, this function will check the input of that workflow under the default (NULL). If there are multiple workflows available, you should specify the workflow.
dry	Logical(1). When TRUE (default), report the updated configuration but do not perform the action requested in Terra. When FALSE, inputs in Terra/AnVIL will be updated.
verbose	Logical(1). When TRUE (default), this function will print the updated input.

### Value

With verbose=TRUE, a list of updated inputs will be printed. A successful execution of the function will update the input configuration of the target workflow in Terra/AnVIL.

### Examples

```
library(AnVIL)
if (gcloud_exists() && nzchar(avworkspace_name())) {
  if ("salmon" %in% avworkspaces()) {
    inputs <- currentInput(workspaceName = "salmon")
    ## Modify the contents of 'inputs' table for your analysis
    updateInput(workspaceName = "salmon", inputs = inputs)
  }
}
```

# Index

## \* **internal**

- .biobakery\_currentInput, 2
- .get\_workflow\_fullname, 3
- .get\_workspace\_fullname, 3
- .nonMetadataOutputs, 4
- .stop\_quietly, 5

- .biobakery\_currentInput, 2
- .get\_workflow\_fullname, 3
- .get\_workspace\_fullname, 3
- .nonMetadataOutputs, 4
- .stop\_quietly, 5

availableAnalysis, 5, 6  
avworkflow\_files, 4

cloneWorkspace, 6  
currentInput, 7, 13

findInputName, 8

getDashboard, 8  
getOutput, 9

monitorWorkflow, 10

runWorkflow, 11

setCloudEnv, 3, 12  
stopWorkflow, 12

updateInput, 13