

# Package: tidycwl (via r-universe)

October 12, 2024

**Type** Package

**Title** Tidy Common Workflow Language Tools and Workflows

**Version** 1.0.7

**Maintainer** Soner Koc <soner.koc@sevenbridges.com>

**Description** The Common Workflow Language <<https://www.commonwl.org/>> is an open standard for describing data analysis workflows. This package takes the raw Common Workflow Language workflows encoded in JSON or 'YAML' and turns the workflow elements into tidy data frames or lists. A graph representation for the workflow can be constructed and visualized with the parsed workflow inputs, outputs, and steps. Users can embed the visualizations in their 'Shiny' applications, and export them as HTML files or static images.

**License** AGPL-3

**SystemRequirements** PhantomJS (<https://phantomjs.org>), pandoc (>= 1.12.3) - <https://pandoc.org>.

**VignetteBuilder** knitr

**URL** <https://sbg.github.io/tidycwl/>, <https://github.com/sbg/tidycwl>

**BugReports** <https://github.com/sbg/tidycwl/issues>

**Encoding** UTF-8

**Imports** jsonlite, yaml, dplyr, magrittr, visNetwork, htmlwidgets, webshot

**Suggests** knitr, rmarkdown, shiny

**RoxygenNote** 7.1.2

**Repository** <https://sbg.r-universe.dev>

**RemoteUrl** <https://github.com/sbg/tidycwl>

**RemoteRef** HEAD

**RemoteSha** 8c1003ea80a5ef3b4a76ae0a674d1c15588b62ec

## Contents

export_html	2
export_image	3
get_cwl_version	4
get_edges	5
get_graph	6
get_inputs_id	7
get_inputs_label	7
get_nodes	8
get_outputs_id	9
get_outputs_label	9
get_steps_doc	10
get_steps_id	11
get_steps_label	11
get_steps_revision	12
get_steps_version	13
is_cwl	13
is_draft2	14
is_tool	15
is_v1.0	15
is_v1.1	16
is_workflow	16
parse_cwl	17
parse_inputs	18
parse_meta	18
parse_outputs	19
parse_steps	20
parse_type	20
print.cwl	21
read_cwl	22
read_cwl_json	22
read_cwl_yaml	23
tidycwl_shiny	23
visualize_graph	24
<b>Index</b>	<b>26</b>

---

export\_html

*Export the workflow plot as HTML*

---

### Description

Export the workflow plot as HTML

### Usage

export\_html(g, file, ...)

**Arguments**

`g` Plot rendered by [visualize\\_graph](#).  
`file` File to save HTML into.  
`...` Additional parameters for [visSave](#).

**Value**

HTML file path

**Examples**

```
file_html <- tempfile(fileext = ".html")
flow <- system.file("cwl/sbg/workflow/gatk4-wgs.json", package = "tidycwl") %>% read_cwl_json()
if (rmarkdown::pandoc_available("1.12.3")) {
  get_graph(
    flow %>% parse_inputs(),
    flow %>% parse_outputs(),
    flow %>% parse_steps()
  ) %>%
  visualize_graph() %>%
  export_html(file_html)
}
```

---

export\_image

*Export the workflow plot as PNG, JPEG, or PDF files*

---

**Description**

Export the workflow plot as PNG, JPEG, or PDF files

**Usage**

```
export_image(file_html, file_image, ...)
```

**Arguments**

`file_html` File path to the HTML exported by [export\\_html](#).  
`file_image` File path to the output image. Should end with `.png`, `.pdf`, or `.jpeg`.  
`...` Additional parameters for [webshot](#).

**Value**

Image file path

**Note**

This function uses [webshot](#) to take a screenshot for the rendered HTML of the graph. It requires PhantomJS installed in your system. You can use [install\\_phantomjs](#) to install it.

**Examples**

```
if (interactive()) {
  file_png <- tempfile(fileext = ".png")
  flow <- system.file("cwl/sbg/workflow/gatk4-wgs.json", package = "tidycwl") %>% read_cwl_json()
  get_graph(
    flow %>% parse_inputs(),
    flow %>% parse_outputs(),
    flow %>% parse_steps()
  ) %>%
  visualize_graph() %>%
  export_html(tempfile(fileext = ".html")) %>%
  export_image(file_png, vwidth = 2000, vheight = 3000, selector = "div.vis-network")
}
```

---

get\_cwl\_version

*Get CWL version*

---

**Description**

Get CWL version

**Usage**

```
get_cwl_version(x)
```

**Arguments**

x                   CWL object

**Value**

CWL version number

**Examples**

```
system.file("cwl/sbg/workflow/gatk4-wgs.json", package = "tidycwl") %>%
  read_cwl_json() %>%
  get_cwl_version()

system.file("cwl/sbg/workflow/rnaseq-salmon.json", package = "tidycwl") %>%
  read_cwl_json() %>%
  get_cwl_version()
```

---

get_edges	<i>Get edges in a CWL workflow into a data frame</i>
-----------	--

---

## Description

Get edges in a CWL workflow into a data frame

## Usage

```
get_edges(outputs, steps)
```

## Arguments

outputs	Parsed outputs
steps	Parsed steps

## Value

Data frame containing edge information

## Examples

```
# edges represented by a dictionary
flow <- system.file("cwl/sbg/workflow/rnaseq-salmon.json", package = "tidycwl") %>% read_cwl_json()
get_edges(
  flow %>% parse_outputs(),
  flow %>% parse_steps()
) %>% str()

# edges represented by a list
try(
  flow <- system.file("cwl/sbg/workflow/rnaseq-salmon.cwl", package = "tidycwl") %>% read_cwl_yaml()
)
try(
  get_edges(
    flow %>% parse_outputs(),
    flow %>% parse_steps()
  ) %>% str()
)
```

---

get_graph	<i>Get the CWL workflow graph</i>
-----------	-----------------------------------

---

### Description

Get the CWL workflow graph as a list of two data frames: a data frame of nodes and a data frame of edges.

### Usage

```
get_graph(inputs, outputs, steps)
```

### Arguments

inputs	Parsed inputs
outputs	Parsed outputs
steps	Parsed steps

### Value

List of two data frames containing node and edge information

### Examples

```
# sbg:draft2
flow <- system.file("cwl/sbg/workflow/gatk4-wgs.json", package = "tidycwl") %>% read_cwl_json()
get_graph(
  flow %>% parse_inputs(),
  flow %>% parse_outputs(),
  flow %>% parse_steps()
) %>% str()

# v1.0
flow <- system.file("cwl/sbg/workflow/rnaseq-salmon.json", package = "tidycwl") %>% read_cwl_json()
get_graph(
  flow %>% parse_inputs(),
  flow %>% parse_outputs(),
  flow %>% parse_steps()
) %>% str()
```

---

get_inputs_id	<i>Get ID for inputs</i>
---------------	--------------------------

---

**Description**

Get ID for inputs

**Usage**

```
get_inputs_id(inputs)
```

**Arguments**

inputs	Parsed inputs
--------	---------------

**Value**

Vector of input IDs

**Examples**

```
# inputs represented by a dictionary
system.file("cwl/sbg/workflow/rnaseq-salmon.json", package = "tidycwl") %>%
  read_cwl_json() %>%
  parse_inputs() %>%
  get_inputs_id()

# inputs represented by a list
system.file("cwl/sbg/workflow/rnaseq-salmon.cwl", package = "tidycwl") %>%
  read_cwl_yaml() %>%
  parse_inputs() %>%
  get_inputs_id()
```

---

get_inputs_label	<i>Get label for inputs</i>
------------------	-----------------------------

---

**Description**

Get label for inputs

**Usage**

```
get_inputs_label(inputs)
```

**Arguments**

inputs	Parsed inputs
--------	---------------

**Value**

Vector of input labels

**Examples**

```
# inputs represented by a dictionary
system.file("cwl/sbg/workflow/rnaseq-salmon.json", package = "tidycwl") %>%
  read_cwl_json() %>%
  parse_inputs() %>%
  get_inputs_label()
```

```
# inputs represented by a list
system.file("cwl/sbg/workflow/rnaseq-salmon.cwl", package = "tidycwl") %>%
  read_cwl_yaml() %>%
  parse_inputs() %>%
  get_inputs_label()
```

---

get\_nodes

*Get nodes in a CWL workflow into a data frame*

---

**Description**

Get nodes in a CWL workflow into a data frame

**Usage**

```
get_nodes(inputs, outputs, steps)
```

**Arguments**

inputs	Parsed inputs
outputs	Parsed outputs
steps	Parsed steps

**Value**

Data frame containing node information

**Examples**

```
flow <- system.file("cwl/sbg/workflow/gatk4-wgs.json", package = "tidycwl") %>% read_cwl_json()
get_nodes(
  flow %>% parse_inputs(),
  flow %>% parse_outputs(),
  flow %>% parse_steps()
) %>% str()
```



---

get_outputs_id	<i>Get ID for outputs</i>
----------------	---------------------------

---

**Description**

Get ID for outputs

**Usage**

```
get_outputs_id(outputs)
```

**Arguments**

outputs	Parsed outputs
---------	----------------

**Value**

Vector of output IDs

**Examples**

```
# inputs represented by a dictionary
system.file("cwl/sbg/workflow/rnaseq-salmon.json", package = "tidycwl") %>%
  read_cwl_json() %>%
  parse_outputs() %>%
  get_outputs_id()

# inputs represented by a list
system.file("cwl/sbg/workflow/rnaseq-salmon.cwl", package = "tidycwl") %>%
  read_cwl_yaml() %>%
  parse_outputs() %>%
  get_outputs_id()
```

---

get_outputs_label	<i>Get label for outputs</i>
-------------------	------------------------------

---

**Description**

Get label for outputs

**Usage**

```
get_outputs_label(outputs)
```

**Arguments**

outputs	Parsed outputs
---------	----------------

**Value**

Vector of output labels

**Examples**

```
# inputs represented by a dictionary
system.file("cwl/sbg/workflow/rnaseq-salmon.json", package = "tidycwl") %>%
  read_cwl_json() %>%
  parse_outputs() %>%
  get_outputs_label()

# inputs represented by a list
system.file("cwl/sbg/workflow/rnaseq-salmon.cwl", package = "tidycwl") %>%
  read_cwl_yaml() %>%
  parse_outputs() %>%
  get_outputs_label()
```

---

get\_steps\_doc

*Get documentation/description for steps*

---

**Description**

Get documentation/description for steps

**Usage**

```
get_steps_doc(steps)
```

**Arguments**

steps                    Steps object parsed by [parse\\_steps](#)

**Value**

Vector of step documentation/descriptions

**Examples**

```
# steps represented by a dictionary
system.file("cwl/sbg/workflow/rnaseq-salmon.json", package = "tidycwl") %>%
  read_cwl_json() %>%
  parse_steps() %>%
  get_steps_doc()

# steps represented by a list
system.file("cwl/sbg/workflow/rnaseq-salmon.cwl", package = "tidycwl") %>%
  read_cwl_yaml() %>%
  parse_steps() %>%
  get_steps_doc()
```

---

get_steps_id	<i>Get ID for steps</i>
--------------	-------------------------

---

**Description**

Get ID for steps

**Usage**

```
get_steps_id(steps)
```

**Arguments**

steps                Steps object parsed by [parse\\_steps](#)

**Value**

Vector of step IDs

**Examples**

```
# steps represented by a dictionary
system.file("cwl/sbg/workflow/rnaseq-salmon.json", package = "tidycwl") %>%
  read_cwl_json() %>%
  parse_steps() %>%
  get_steps_id()

# steps represented by a list
system.file("cwl/sbg/workflow/rnaseq-salmon.cwl", package = "tidycwl") %>%
  read_cwl_yaml() %>%
  parse_steps() %>%
  get_steps_id()
```

---

get_steps_label	<i>Get label for steps</i>
-----------------	----------------------------

---

**Description**

Get label for steps

**Usage**

```
get_steps_label(steps)
```

**Arguments**

steps                Steps object parsed by [parse\\_steps](#)

**Value**

Vector of step labels

**Examples**

```
# steps represented by a dictionary
system.file("cwl/sbg/workflow/rnaseq-salmon.json", package = "tidycwl") %>%
  read_cwl_json() %>%
  parse_steps() %>%
  get_steps_label()

# steps represented by a list
system.file("cwl/sbg/workflow/rnaseq-salmon.cwl", package = "tidycwl") %>%
  read_cwl_yaml() %>%
  parse_steps() %>%
  get_steps_label()
```

---

get_steps_revision	<i>Get revision number for steps</i>
--------------------	--------------------------------------

---

**Description**

Get revision number for steps

**Usage**

```
get_steps_revision(steps)
```

**Arguments**

steps                    Steps object parsed by [parse\\_steps](#)

**Value**

Vector of step revision numbers

**Examples**

```
# steps represented by a dictionary
system.file("cwl/sbg/workflow/rnaseq-salmon.json", package = "tidycwl") %>%
  read_cwl_json() %>%
  parse_steps() %>%
  get_steps_revision()

# steps represented by a list
system.file("cwl/sbg/workflow/rnaseq-salmon.cwl", package = "tidycwl") %>%
  read_cwl_yaml() %>%
  parse_steps() %>%
  get_steps_revision()
```

---

get_steps_version	<i>Get toolkit version for steps</i>
-------------------	--------------------------------------

---

**Description**

Get toolkit version for steps

**Usage**

```
get_steps_version(steps)
```

**Arguments**

steps                Steps object parsed by [parse\\_steps](#)

**Value**

Vector of step toolkit versions

**Examples**

```
# steps represented by a dictionary
system.file("cwl/sbg/workflow/rnaseq-salmon.json", package = "tidycwl") %>%
  read_cwl_json() %>%
  parse_steps() %>%
  get_steps_version()

# steps represented by a list
system.file("cwl/sbg/workflow/rnaseq-salmon.cwl", package = "tidycwl") %>%
  read_cwl_yaml() %>%
  parse_steps() %>%
  get_steps_version()
```

---

is_cwl	<i>Is this a CWL object?</i>
--------	------------------------------

---

**Description**

Is this a CWL object?

**Usage**

```
is_cwl(x)
```

**Arguments**

x                    any object

**Value**

Logical. TRUE if it is a CWL object, FALSE if not.

**Examples**

```
system.file("cwl/sbg/workflow/gatk4-wgs.json", package = "tidycwl") %>%
  read_cwl(format = "json") %>%
  is_cwl()
```

---

 is\_draft2

*Is this CWL draft2?*


---

**Description**

Is this CWL draft2?

**Usage**

```
is_draft2(x)
```

**Arguments**

x                   CWL object

**Value**

Logical. TRUE if it is a CWL draft2 object, FALSE if not.

**Examples**

```
system.file("cwl/sbg/workflow/gatk4-wgs.json", package = "tidycwl") %>%
  read_cwl(format = "json") %>%
  is_draft2()
```

```
system.file("cwl/sbg/workflow/rnaseq-salmon.json", package = "tidycwl") %>%
  read_cwl(format = "json") %>%
  is_draft2()
```

---

is_tool	<i>Is this a CWL command line tool?</i>
---------	---

---

**Description**

Is this a CWL command line tool?

**Usage**

```
is_tool(x)
```

**Arguments**

x	CWL object
---	------------

**Value**

Logical. TRUE if it is a CWL command line tool (instead of a workflow), FALSE if not.

**Examples**

```
system.file("cwl/sbg/tool/bwa-mem.json", package = "tidycwl") %>%  
  read_cwl(format = "json") %>%  
  is_tool()
```

```
system.file("cwl/sbg/workflow/gatk4-wgs.json", package = "tidycwl") %>%  
  read_cwl(format = "json") %>%  
  is_tool()
```

---

is_v1.0	<i>Is this CWL v1.0?</i>
---------	--------------------------

---

**Description**

Is this CWL v1.0?

**Usage**

```
is_v1.0(x)
```

**Arguments**

x	CWL object
---	------------

**Value**

Logical. TRUE if it is a CWL v1.0 object, FALSE if not.

**Examples**

```

system.file("cwl/sbg/workflow/gatk4-wgs.json", package = "tidycwl") %>%
  read_cwl(format = "json") %>%
  is_v1.0()

system.file("cwl/sbg/workflow/rnaseq-salmon.json", package = "tidycwl") %>%
  read_cwl(format = "json") %>%
  is_v1.0()

```

---

is\_v1.1

*Is this CWL v1.1?*


---

**Description**

Is this CWL v1.1?

**Usage**

```
is_v1.1(x)
```

**Arguments**

x CWL object

**Value**

Logical. TRUE if it is a CWL v1.1 object, FALSE if not.

**Examples**

```

system.file("cwl/sbg/workflow/rnaseq-salmon.json", package = "tidycwl") %>%
  read_cwl(format = "json") %>%
  is_v1.1()

```

---

is\_workflow

*Is this a CWL workflow?*


---

**Description**

Is this a CWL workflow?

**Usage**

```
is_workflow(x)
```



**Arguments**

x                   CWL object

**Value**

Logical. TRUE if it is a CWL workflow (instead of a command line tool), FALSE if not.

**Examples**

```
system.file("cwl/sbg/workflow/gatk4-wgs.json", package = "tidycwl") %>%
  read_cwl(format = "json") %>%
  is_workflow()
```

```
system.file("cwl/sbg/tool/bwa-mem.json", package = "tidycwl") %>%
  read_cwl(format = "json") %>%
  is_workflow()
```

---

parse\_cwl                   *Parse a CWL workflow*

---

**Description**

Parse a CWL workflow and return the metadata, inputs, outputs, and steps in a list.

**Usage**

```
parse_cwl(x)
```

**Arguments**

x                   CWL object

**Value**

List of CWL metadata, inputs, outputs, and steps

**Examples**

```
system.file("cwl/sbg/workflow/rnaseq-salmon.cwl", package = "tidycwl") %>%
  read_cwl_yaml() %>%
  parse_cwl() %>%
  names()
```

---

parse_inputs	<i>Parse the inputs of the CWL workflow into a data frame</i>
--------------	---

---

**Description**

Parse the inputs of the CWL workflow into a data frame

**Usage**

```
parse_inputs(x, simplify = TRUE)
```

**Arguments**

x	CWL object
simplify	Simplify the list as a data frame?

**Value**

List or data frame of inputs

**Examples**

```
system.file("cwl/sbg/workflow/rnaseq-salmon.json", package = "tidycwl") %>%  
  read_cwl_json() %>%  
  parse_inputs() %>%  
  names()  
  
system.file("cwl/sbg/workflow/rnaseq-salmon.cwl", package = "tidycwl") %>%  
  read_cwl_yaml() %>%  
  parse_inputs() %>%  
  names()
```

---

parse_meta	<i>Parse the metadata in the CWL workflow</i>
------------	---

---

**Description**

Parse the metadata in the CWL workflow

**Usage**

```
parse_meta(x)
```

**Arguments**

x	CWL object
---	------------

**Value**

List of CWL metadata

**Examples**

```
system.file("cwl/sbg/workflow/gatk4-wgs.json", package = "tidycwl") %>%  
  read_cwl(format = "json") %>%  
  parse_meta()
```

---

parse_outputs	<i>Parse the outputs of the CWL workflow into a data frame</i>
---------------	--

---

**Description**

Parse the outputs of the CWL workflow into a data frame

**Usage**

```
parse_outputs(x, simplify = TRUE)
```

**Arguments**

x	CWL object
simplify	Simplify the list as a data frame?

**Value**

List or data frame of outputs

**Examples**

```
system.file("cwl/sbg/workflow/rnaseq-salmon.json", package = "tidycwl") %>%  
  read_cwl_json() %>%  
  parse_outputs() %>%  
  names()  
  
system.file("cwl/sbg/workflow/rnaseq-salmon.cwl", package = "tidycwl") %>%  
  read_cwl_yaml() %>%  
  parse_outputs() %>%  
  names()
```

---

parse_steps	<i>Parse the steps of the CWL workflow into a data frame</i>
-------------	--

---

**Description**

Parse the steps of the CWL workflow into a data frame

**Usage**

```
parse_steps(x)
```

**Arguments**

x CWL object

**Value**

List or data frame of steps

**Examples**

```
# steps represented by a dictionary
system.file("cwl/sbg/workflow/rnaseq-salmon.json", package = "tidycwl") %>%
  read_cwl_json() %>%
  parse_steps() %>%
  nrow()

# steps represented by a list
system.file("cwl/sbg/workflow/rnaseq-salmon.cwl", package = "tidycwl") %>%
  read_cwl_yaml() %>%
  parse_steps() %>%
  length()
```

---

parse_type	<i>Parse CWL content type</i>
------------	-------------------------------

---

**Description**

Parse CWL content type

**Usage**

```
parse_type(x)
```

**Arguments**

x CWL object

**Value**

CWL content type (Workflow or CommandLineTool)

**Examples**

```
system.file("cwl/sbg/workflow/gatk4-wgs.json", package = "tidycwl") %>%
  read_cwl(format = "json") %>%
  parse_type()
```

```
system.file("cwl/sbg/tool/bwa-mem.json", package = "tidycwl") %>%
  read_cwl(format = "json") %>%
  parse_type()
```

---

print.cwl	<i>Print CWL objects</i>
-----------	--------------------------

---

**Description**

Print a brief summary of the CWL object.

**Usage**

```
## S3 method for class 'cwl'
print(x, ...)
```

**Arguments**

x	An object of class cwl.
...	Additional parameters for <code>print</code> (not used).

**Value**

The input cwl object.

**Examples**

```
path <- system.file("cwl/sbg/workflow/rnaseq-salmon.json", package = "tidycwl")
flow <- read_cwl(path, format = "json")
flow
```

---

read_cwl	<i>Read a CWL file into a list</i>
----------	------------------------------------

---

**Description**

Read a CWL file into a list

**Usage**

```
read_cwl(file, format = c("json", "yaml"))
```

**Arguments**

file	A file path, character string, or connection.
format	CWL storage format. "json" or "yaml".

**Value**

List representation of the input CWL

**Examples**

```
system.file("cwl/sbg/workflow/rnaseq-salmon.json", package = "tidycwl") %>%  
  read_cwl(format = "json")  
  
system.file("cwl/sbg/workflow/rnaseq-salmon.cwl", package = "tidycwl") %>%  
  read_cwl(format = "yaml")
```

---

read_cwl_json	<i>Read a CWL file (JSON format) into a list</i>
---------------	--

---

**Description**

Read a CWL file (JSON format) into a list

**Usage**

```
read_cwl_json(file)
```

**Arguments**

file	A file path, JSON string, or connection.
------	--

**Value**

List representation of the input CWL

**Examples**

```
system.file("cwl/sbg/workflow/rnaseq-salmon.json", package = "tidycwl") %>%
  read_cwl_json()
```

---

read_cwl_yaml	<i>Read a CWL file (YAML format) into a list</i>
---------------	--

---

**Description**

Read a CWL file (YAML format) into a list

**Usage**

```
read_cwl_yaml(file)
```

**Arguments**

file            A file path, YAML string, or connection.

**Value**

List representation of the input CWL

**Examples**

```
system.file("cwl/sbg/workflow/rnaseq-salmon.cwl", package = "tidycwl") %>%
  read_cwl_yaml()
```

---

tidycwl_shiny	<i>Shiny bindings for tidycwl</i>
---------------	-----------------------------------

---

**Description**

Output and renderer functions for using tidycwl within Shiny apps and interactive R Markdown documents.

**Usage**

```
cwl_output(outputId, width = "100%", height = "600px")
```

```
render_cwl(expr, env = parent.frame(), quoted = FALSE)
```

**Arguments**

outputId	output variable to read from
width, height	Must be a valid CSS unit (like "100%", "600px", "auto") or a number, which will be coerced to a string and have "px" appended.
expr	An expression that generates a CWL graph
env	The environment in which to evaluate expr.
quoted	Is expr a quoted expression (with quote())? This is useful if you want to save an expression in a variable.

**Value**

An output or render function that enables the use of the widget within Shiny apps.

**Examples**

```
if (interactive()) {
  library("shiny")
  library("tidycwl")

  cwl_folder <- system.file("cwl/sbg/workflow/", package = "tidycwl")
  file_all <- list.files(cwl_folder)
  cwl_name <- file_all[which(tools::file_ext(file_all) == "json")]

  ui <- fluidPage(
    selectInput("cwl_file", "Select a CWL file:", cwl_name),
    cwl_output("cwl_plot", height = "800px")
  )

  server <- function(input, output, session) {
    output$cwl_plot <- render_cwl({
      flow <- paste0(cwl_folder, input$cwl_file) %>% read_cwl_json()
      get_graph(
        flow %>% parse_inputs(),
        flow %>% parse_outputs(),
        flow %>% parse_steps()
      ) %>% visualize_graph()
    })
  }

  shinyApp(ui, server)
}
```

---

 visualize\_graph

*Visualize the CWL workflow*


---

**Description**

Visualize the CWL workflow



## Usage

```
visualize_graph(  
  g,  
  hierarchical = TRUE,  
  direction = "LR",  
  separation = 300,  
  palette = c("#C3C3C3", "#FF8F00", "#00AAA8"),  
  width = "100%",  
  height = 600  
)
```

## Arguments

<code>g</code>	Graph generated by <a href="#">get_graph</a> .
<code>hierarchical</code>	Enable the hierarchical layout? Default is TRUE.
<code>direction</code>	Direction of the hierarchical layout. Options include "LR", "RL", "UD", and "DU" (up-down, down-up, left-right, right-left). Default is "LR".
<code>separation</code>	Level separation parameter from <a href="#">visHierarchicalLayout</a> .
<code>palette</code>	Three-color palette for inputs, outputs, and steps.
<code>width</code>	Canvas width, see <a href="#">visNetwork</a> . Default is "100%".
<code>height</code>	Canvas height, see <a href="#">visNetwork</a> . Default is 600.

## Value

A [visNetwork](#) output.

## Examples

```
flow <- system.file("cwl/sbg/workflow/gatk4-wgs.json", package = "tidycwl") %>% read_cwl_json()  
get_graph(  
  flow %>% parse_inputs(),  
  flow %>% parse_outputs(),  
  flow %>% parse_steps()  
) %>% visualize_graph()
```

# Index

`cwl_output (tidycwl_shiny)`, 23

`export_html`, 2, 3

`export_image`, 3

`get_cwl_version`, 4

`get_edges`, 5

`get_graph`, 6, 25

`get_inputs_id`, 7

`get_inputs_label`, 7

`get_nodes`, 8

`get_outputs_id`, 9

`get_outputs_label`, 9

`get_steps_doc`, 10

`get_steps_id`, 11

`get_steps_label`, 11

`get_steps_revision`, 12

`get_steps_version`, 13

`install_phantomjs`, 3

`is_cwl`, 13

`is_draft2`, 14

`is_tool`, 15

`is_v1.0`, 15

`is_v1.1`, 16

`is_workflow`, 16

`parse_cwl`, 17

`parse_inputs`, 18

`parse_meta`, 18

`parse_outputs`, 19

`parse_steps`, 10–13, 20

`parse_type`, 20

`print`, 21

`print.cwl`, 21

`read_cwl`, 22

`read_cwl_json`, 22

`read_cwl_yaml`, 23

`render_cwl (tidycwl_shiny)`, 23

`tidycwl_shiny`, 23

`visHierarchicalLayout`, 25

`visNetwork`, 25

`visSave`, 3

`visualize_graph`, 3, 24

`webshot`, 3