QGIS in R with qgisprocess :: CHEAT SHEET

Mission

The main objective is provide to the R interface the most popular open-source desktop GIS program like QGIS. This package is a re-implementation of the functionality provided by the archived **RQGIS** package, which was partially revived in the **RQGIS3** package.

Features

This package makes it easier to use native functions from qgis and some from gdal, grass and saga.

Provider	Algorithms
qgis	51+196(c++)+1(3D)
gdal	56
grass	301
saga	366
Total count	971

```
> qgis_algorithms( )
```

Show a tibble with algorithms

Installation

main package

```
> install.packages('remotes')
> remotes::install_github('paleolimbot/qgisprocess')
> library(qgisprocess)
```

GNU/Linux

```
> qgis_configure()
> qgis_version()
> qgis_algorithms()
```

Windows

```
Specify path to QGIS installation on Windows
options("qgisprocess.path" = "C:/Program
Files/QGIS 3.16/bin/qgis_process-qgis.bat")
> qgis_configure()
> qgis_version()
> qgis_algorithms()
```

By docker

1. Get started with the installation of docker in your machine
2. Download the image of geocomputation

```
> docker pull geocompr/geocompr:qgis-ext
```

3. run to image of geocomputation with docker

```
> docker run -d -p 8786:8787 -v
$(pwd):/home/rstudio/data -e PASSWORD=pw
geocompr/geocompr:qgis-ext
```

Input functions

The package offers new functionalities of Input to have a workflow of an easy manner inside of R.

```
qgis_show_help(algorithm ='native:creategrid')
```

Show a description of the function to use

Algorithm name

```
qgis_arguments(algorithm ='native:creategrid')
```

Show all the parameters of the function.

Run the algorithms

```
qgis_run_algorithm(
  algorithm = 'native:creategrid',
  TYPE = 4,
  EXTENT = c('794599, 798208, 8931775,8935384'),
  HSPACING = 1000 ,
  VSPACING = 1000,
  CRS = 'EPSG:32717',
  OUTPUT = 'grid'
      )
```

Create a function based on the algorithm to use

```
grid_fun ← qgis_function('native:creategrid')
grid_fun(
  TYPE = 4,
  EXTENT = c('794599,798208,8931775,8935384'),
  HSPACING = 1000,
  VSPACING = 1000,
  CRS = 'EPSG:32717',
  OUTPUT = 'grid'
     )
```

Output functions

qgisprocess give us new functionalities of output for vector, raster and other format file, and it is possible loads it to our environment work.

```
qgis_output(x = output_run_alg ,which ='OUTPUT')
```

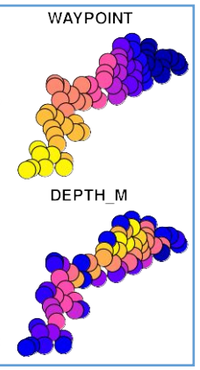
```
qgis_tmp_base( )
qgis_tmp_file( ".csv" )
qgis_tmp_vector( )
qgis_tmp_raster( )
```

A character vector indicating the location of a temporary file.

%>% integration

qgisprocess also provide us two functions that wraps qgis_run_algorithms with the argument default .value = TRUE to make it more usable within other R codes.

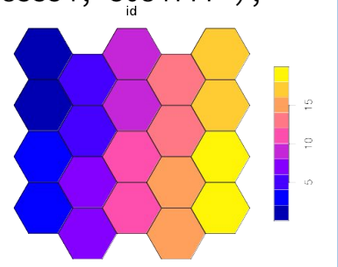
```
qgis_pipe(
 .data = system.file(
            'longlake/longlake_depth.gpkg',
             package = 'qgisprocess'),
  Algorithm = 'native:buffer',
  DISTANCE = 100,
      )%>%
qgis_output('OUTPUT') %>%
st_read( ) %>% plot( )

qgis_fun(algorithm =, . . . ,.quiet = TRUE)
```



Workflows

Vector data

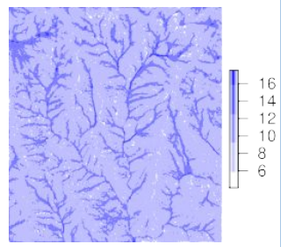
```
depth ← st_read(
system.file('longlake/longlake_depth.gpkg',
package = 'qgisprocess'))
grid_fun ← qgis_function("native:creategrid")
grid_fun(
   TYPE = 4,
   EXTENT = c('409967, 411658, 5083354, 5084777'),
   HSPACING = 400,
   VSPACING = 400,
   CRS = 'EPSG:26920',
   OUTPUT = 'grid') %>%
 qgis_output('OUTPUT') %>%
 st_read() %>% select(id) %>%
 plot()
```



Raster data

```
dem ← raster(
              system.file('spdata/altitude.gkpg',
               package = 'qgisprocess')
qgis_run_algorithm(
  algorithm ='saga:sagawetnessindex',
  DEM = dem,
  TPI = 'tpi.sdat') %>%
  qgis_output(which = "TWI" ) %>%
  read_stars( ) %>%
  plot(col=cpt(pal='ocal_blues')
  )
```

TWI



QGIS en R con qgisprocess :: CHEAT SHEET

Misión

El objetivo principal es proporcionar a la interfaz de R el programa SIG de escritorio de código abierto más popular como QGIS. Este paquete es una reimplementación de la funcionalidad proporcionada por el paquete archivado **RQGIS**, que fue parcialmente revivido en el paquete **RQGIS3**.

Características

Este paquete facilita el uso de funciones nativas de qgis y algunas de gdal, grass y saga.

Provider	Algorithms
qgis	51+196(c++)+1(3D)
gdal	56
grass	301
saga	366
Total count	971

```
> qgis_algorithms( )
```

Muestra todos los algoritmos disponibles en un tibble.

Instalación

Paquete principal

```
> install.packages('remotes')
> remotes::install_github('paleolimbot/qgisprocess')
> library(qgisprocess)
```

GNU/Linux

```
> qgis_configure()
> qgis_version()
> qgis_algorithms()
```

Windows

```
Especifica el path de QGIS al instalar
options("qgisprocess.path" = "C:/Program
Files/QGIS 3.16/bin/qgis_process-qgis.bat")
> qgis_configure()
> qgis_version()
> qgis_algorithms()
```

By docker

1. Comience con la instalación de docker en su máquina
2. Descargar la imagen de geocomputation

```
> docker pull geocompr/geocompr:qgis-ext
```

3. Ejecuta la image de geocomputation con docker

```
> docker run -d -p 8786:8787 -v
$(pwd):/home/rstudio/data -e PASSWORD=pw
geocompr/geocompr:qgis-ext
```

Funciones de entrada

El paquete ofrece nuevas funcionalidades de Input para tener un flujo de trabajo de manera fácil dentro de R.

```
qgis_show_help(algorithm ='native:creategrid')
```

Mostrar una descripción de la función a utilizar

nombre del algoritmo

```
qgis_arguments(algorithm ='native:creategrid')
```

Mostrar todos los parámetros de la función

Ejecutar los algoritmos

```
qgis_run_algorithm(
  algorithm = 'native:creategrid',
  TYPE = 4,
  EXTENT = c('794599, 798208, 8931775,8935384'),
  HSPACING = 1000 ,
  VSPACING = 1000,
  CRS = 'EPSG:32717',
  OUTPUT = 'grid'
       )
```

Crear una función basada en el algoritmo a utilizar

```
grid_fun ← qgis_function('native:creategrid')
grid_fun(
  TYPE = 4,
  EXTENT = c('794599,798208,8931775,8935384'),
  HSPACING = 1000,
  VSPACING = 1000,
  CRS = 'EPSG:32717',
  OUTPUT = 'grid'
     )
```

Funciones de salida

qgisprocess nos da nuevas funcionalidades de salida para archivos vectoriales, raster y otros formatos, y es posible cargarlo a nuestro entorno de trabajo.

```
qgis_output(x = output_run_alg ,which ='OUTPUT')
```

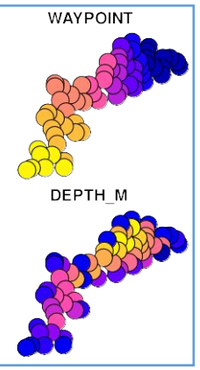
```
qgis_tmp_base( )
qgis_tmp_file( ".csv" )
qgis_tmp_vector( )
qgis_tmp_raster( )
```

Un vector de caracteres que indica la ubicación de un archivo temporal.

Integración de %>%

qgisprocess también nos proporciona dos funciones que envuelve qgis_run_algorithms con el argumento default .value = TRUE para hacerlo más utilizable dentro de otros códigos de R.

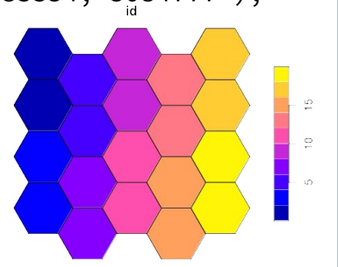
```
qgis_pipe(
 .data = system.file(
            'longlake/longlake_depth.gpkg',
             package = 'qgisprocess'),
  Algorithm = 'native:buffer',
  DISTANCE = 100,
      )%>%
qgis_output('OUTPUT') %>%
st_read( ) %>% plot( )

qgis_fun(algorithm =, . . . ,.quiet = TRUE)
```



Flujo de trabajo

Vector data

```
depth ← st_read(
system.file('longlake/longlake_depth.gpkg',
package = 'qgisprocess'))
grid_fun ← qgis_function("native:creategrid")
grid_fun(
   TYPE = 4,
   EXTENT = c('409967, 411658, 5083354, 5084777'),
   HSPACING = 400,
   VSPACING = 400,
   CRS = 'EPSG:26920',
   OUTPUT = 'grid') %>%
 qgis_output('OUTPUT') %>%
 st_read() %>% select(id) %>%
 plot()
```



Raster data

```
dem ← raster(
              system.file('spdata/altitude.gkpg',
               package = 'qgisprocess')
qgis_run_algorithm(
  algorithm ='saga:sagawetnessindex',
  DEM = dem,
  TPI = 'tpi.sdat') %>%
  qgis_output(which = "TWI" ) %>%
  read_stars( ) %>%
  plot(col=cpt(pal='ocal_blues')
  )
```