Унификация экстента и разрешения растров в QGIS

Обсудить в форуме Комментариев — 17

Эта страница опубликована в основном списке статей сайта по адресу http://gis-lab.info/qa/qgis-rasters-unification.html

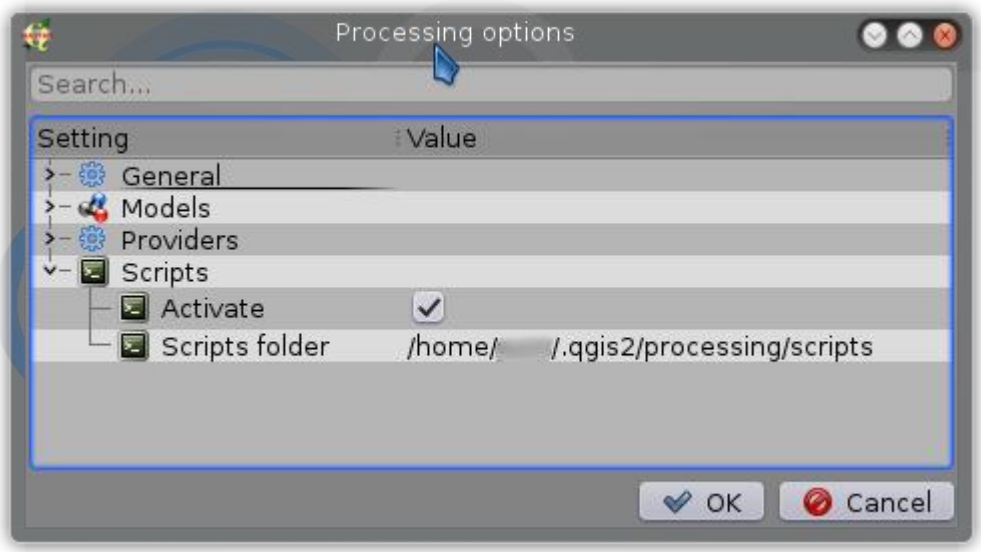
Зачастую для проведения операций растровой алгебры над несколькими растрами одновременно, необходимо, чтобы эти растры имели одинаковые экстент и разрешение. В данной статье описан процесс унификации экстента и разрешения нескольких растров при помощи скрипта, интегрируемого в Processing - модуль геопроцессинга QGIS.

via Misanthrope's Thoughts

Содержание

Установка скрипта

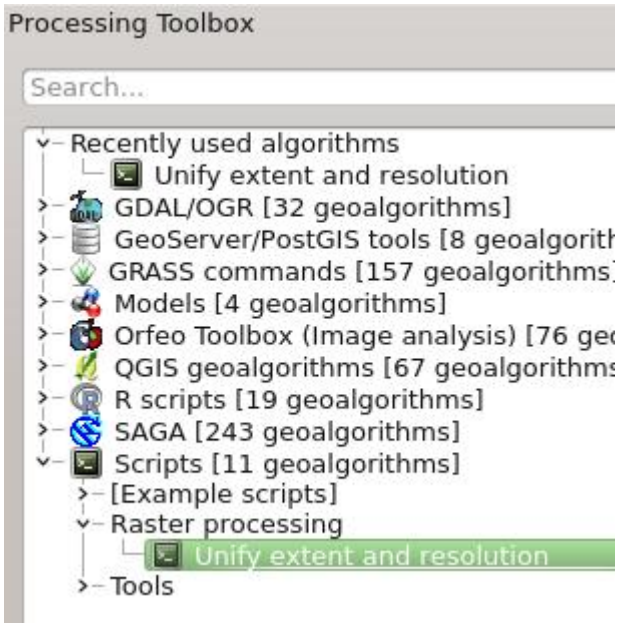
Загрузите архив со скриптом и help-файлом. Распакуйте архив и скопируйте содержимое папки "scripts" в директорию, предназначенную для Python-скриптов модуля Processing в QGIS (например, ~./qgis2/processing/scripts, если вы используете Linux). Если вы не знаете, где находится нужная папка перейдите в Processing -> Options and configuration -> Scripts -> Scripts folder, см. скриншот:



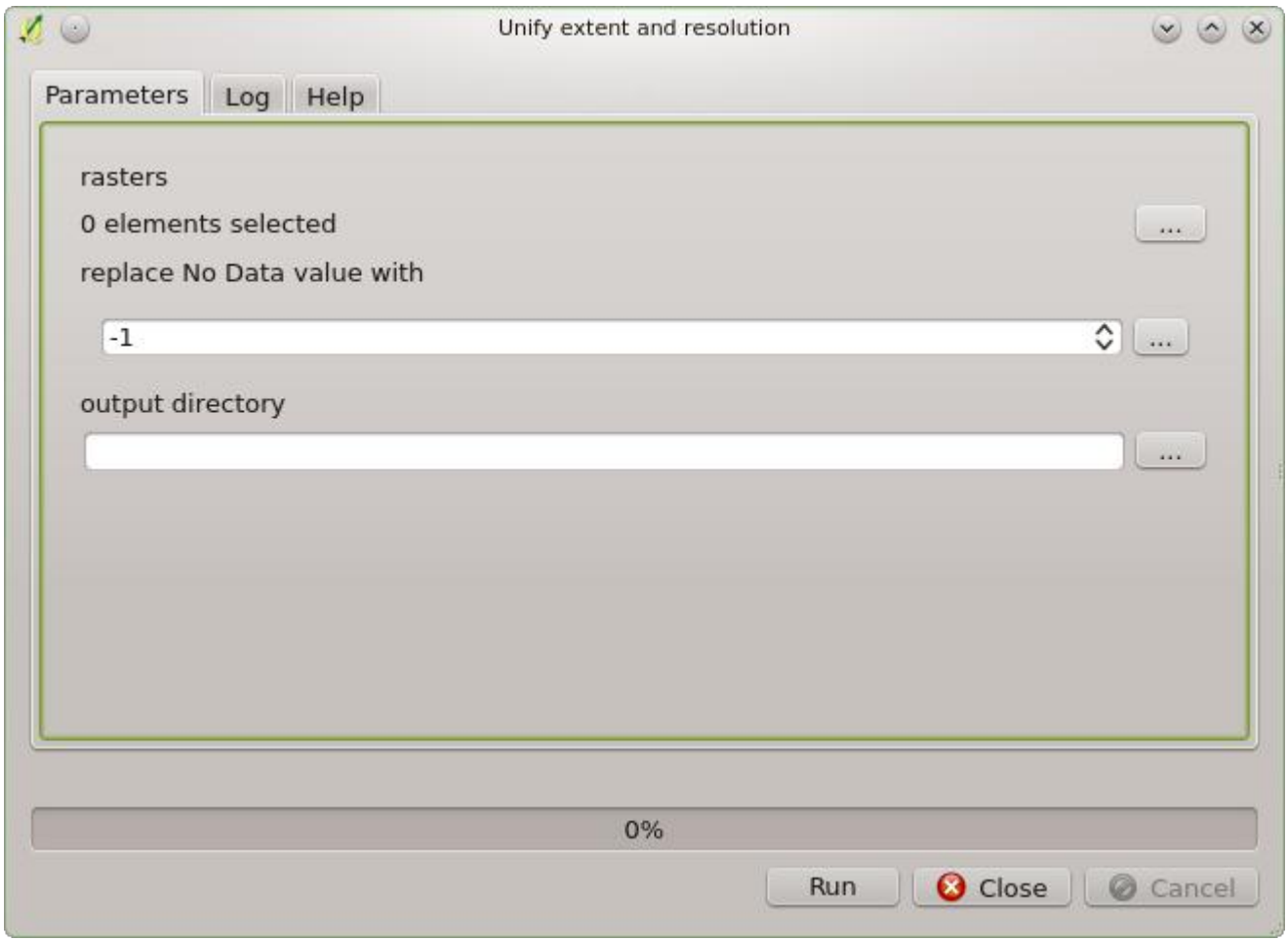
Работа со скриптом

Перезапустите QGIS, если она была запущена. Откройте **Processing Toolbox**. Во вкладке **Scripts** вы увидите

раздел **Raster processing**, а в нём - скрипт **Unify extent and resolution**:

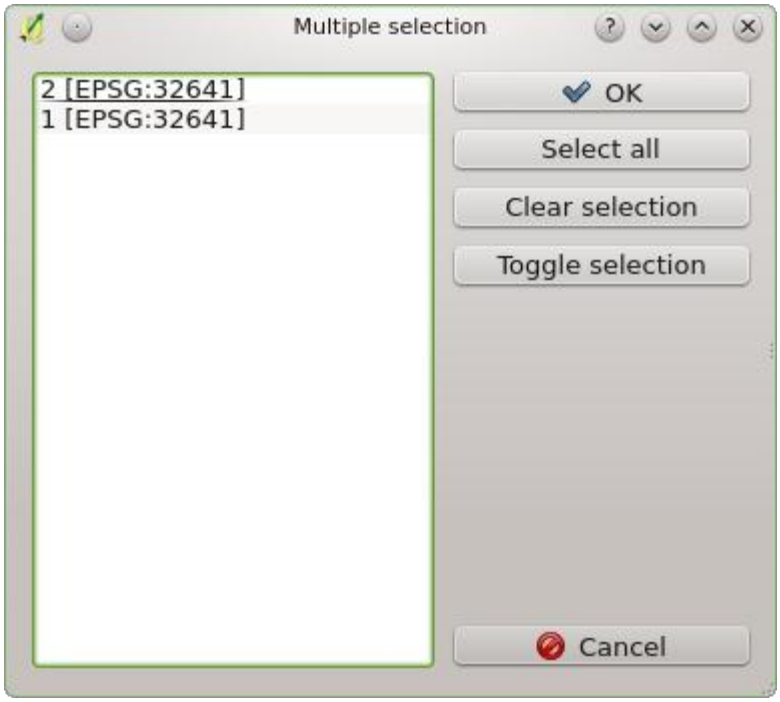


Запустив его вы увидите диалоговое окно:



Обратите внимание, что во вкладке Help находится описание скрипта (на английском языке).

В поле "**rasters**" выберите 2 или более растра, экстент и разрешение которых надо сделать одинаковыми. Системы координат всех растров должны быть идентичными.

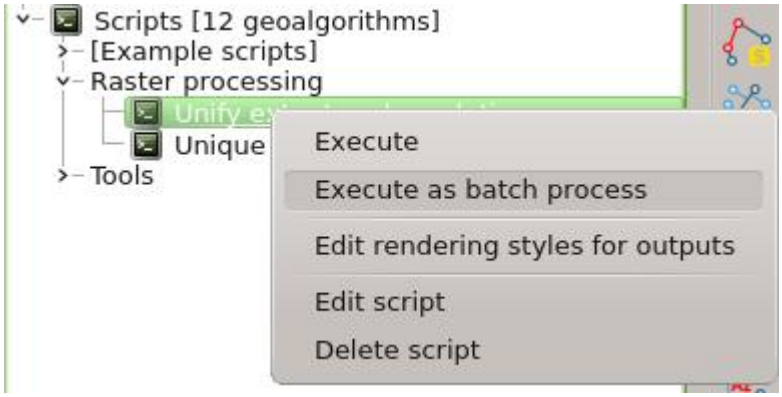


В поле "**replace No Data value with**" введите значение, которое будет использовано для новых пикселей, добавляемых в растр, а также заменит в результирующих растрах значения No Data обрабатываемых растров. Обратите внимание, что этому значению в результирующих растрах не будет присвоен атрибут No Data - они будут обрабатываться любым ПО, как обычные пиксели.

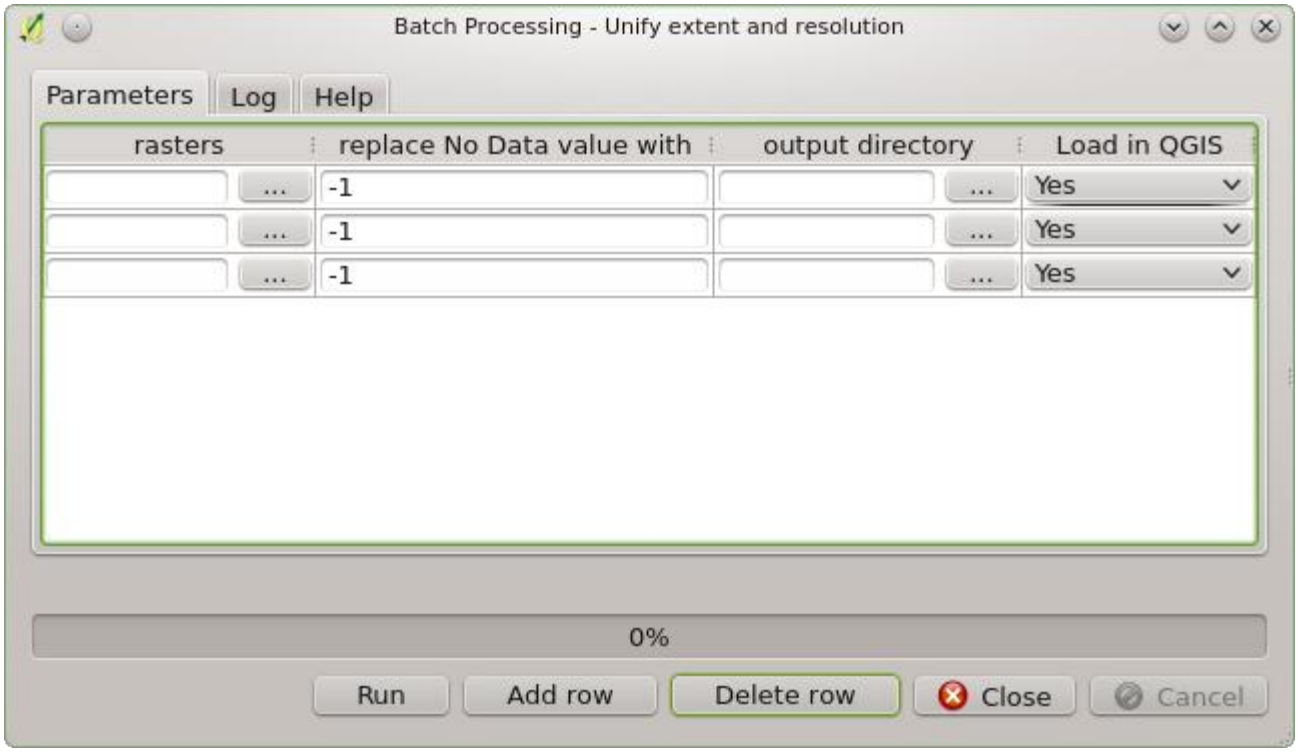
В поле "**output directory**" укажите путь к папке в которую должны быть сохранены результаты. Результирующие файлы будут названы следующим образом. Первая часть имени файла будет соответствовать названию исходного файла, к нему будет добавлен постфикс "_unified", например: "raster_1.tif" -> "raster_1_unified.tif".

После окончания основной работы скрипта появится сообщение о том, что растры были унифицированы и пользователю будет предоставлена возможность добавить результаты на карту, если он этого пожелает:

Для запуска скрипта в режиме пакетной обработки кликните по скрипту в Processing Toolbox правой кнопкой мыши и выберите "Execute as batch process":



Появится такое диалоговое окно, где вы сможете настроить все параметры, указанные выше:



Как это работает

Опишем только основные идеи, использованные в алгоритме. Исходный код здесь подробно обсуждаться не будет, так как скрипт содержит свыше десятка функций и несколько классов (для Qt-сообщений). Кроме того, скрипт снабжен достаточным количеством комментариев.

Для обработки растров используется не QGIS API, а GDAL API. Работа алгоритма состоит в следующем:

- Для каждого угла обрабатываемых растров вычисляются значения X и Y. Из них выбираются максимальные и минимальные значения. Вычисление координат углов базируется на основе аффинной трансформации, лежащей в основе модели данных GDAL:

```
Xgeo = GT0 + Xpixel*GT1 + Yline*GT2
Ygeo = GT3 + Xpixel*GT4 + Yline*GT5
```

где Xgeo - значение координаты X, Ygeo - значение координаты Y, Xpixel - номер столбца пикселя в растре, Ygeo - номер строки пикселя в растре, GT2 и GT4 вращение пикслея по осям X и Y соответственно, GT1 - ширина пикселя, GT5 - высота пикселя, GT0 и GT3 - соответственно X-координата и Y-координата верхнего левого угла верхнего левого пикселя ратсра, GT - параметры геотрансформации ратра, возвращаемые функцией gdal.GetGeoTransform().

- Далее берётся первый растр из поступивших на обработку и из него берётся информация о проекции и горизонтальных и вертикальных размерах пикселя. Эта информация вкупе с данными о финальном экстенте будет использована для создания болванок унифицированных растров.
- Наконец, для каждого из обрабатываемых растров создаётся унифицированная болванка на которую переносятся значения исходного растра, а оставшееся место (ведь экстент результирующего растра не меньше экстента исходного растра) забивается значениями, указанными пользователем. Это происходит следующим образом. Для каждого пикселя болванки рассчитываются её пространственные координаты (на основе формулы, представленной выше). Вычисляется, какому пикселю оригинального растра соответствуют данные координаты. Полученное значение пикселя оригинального растра записывается в болванку. Если координаты выходят за пределы оригинального растра или значение данного пикселя помечено, как No Data, то в болванку записывается значение, определённое пользователем.

В любой момент времени не происходит работы более, чем с двумя растрами (исходный и результирующий), таким образом, количество растров, которые можно подать на обработку за раз практически не ограничено.

Если вы таки захотите покопаться в исходном коде, то начните с функции **main()**:

```
def main(rasters_list, no_data):
```

```
  # проверяем корректность указанного пути сохранения результатов
  if not checkFolder(output_directory):
    return NoPathGiven()

  # проверяем, совпадают ли проекции растров
  if not checkCRS(rasters_list):
    return WrongCRS()

  # получаем экстент для результирующих ратров
  fin_coordinates = finCoordinates(rasters_list)

  # собираем необходимую информацию для обработки
  processing_list = processingList(rasters_list, output_directory)
  raster_1 = rasters_list[0]
  raster_1 = gdal.Open(raster_1)
  main_geo_transform = raster_1.GetGeoTransform() # параметры геотрансформации
эталонного растра
  proj = raster_1.GetProjection()
  if not no_data:
    no_data = raster_1.GetRasterBand(1).GetNoDataValue()
    if not no_data:
      no_data = -9999

  # обрабатываем растры
  for tupl in processing_list:
    raster = gdal.Open(tupl[0])
    output = ExtendRaster(raster, fin_coordinates, tupl[1], main_geo_transform, proj,
no_data)
    raster = None
    if output != True:
      return None

  # выводим сообщение об успешном завершении
  result = Success()
  return result.initUI(processing_list)
```

Вместо заключения

Пожелания и предложения по работе скрипта можно оставлять в соответствующей теме форума.

Ссылки по теме

- Работа с растрами при помощи GDAL и Python
- Геопроцессинг с SEXTANTE для QGIS
- Подсчёт уникальных значений в растре при помощи SEXTANTE и QGIS

Обсудить в форуме Комментариев — 17

Последнее обновление: 2014-05-15 02:08

Дата создания: 05.01.2014
Автор(ы): SS_Rebelious