# Вызов GRASS GIS из скрипта на Python

Обсудить в форуме Комментариев — 14

Эта страница опубликована в основном списке статей сайта
по адресу http://gis-lab.info/qa/grass-external-scripting.html

Очень часто работа с GRASS GIS происходит в интерактивном режиме, когда пользователь вводит команды в консоли или вызывает их через графический интерфейс пользователя. При этом часто повторяемые действия пользователь может автоматизировать, создав скрипт, который будет доступен для повторного использования. GRASS GIS поддерживает большое количество языков программирования, на которых можно создавать пользовательские скрипты, наиболее распространенными из них являются скриптовый язык командной оболочки (shell, bash) и Python. Пример создания скрипта на языке Python описывается в статье Создание скрипта на Python для ГИС GRASS. После создания скрипта пользователь может запускать его из консоли, в которой запущена ГИС GRASS.

Однако, иногда может потребоваться полностью автономный скрипт, который может работать, даже если консоль GRASS GIS не запущена, например, при вызове скрипта из какой-либо своей программы. Это может быть полезным при автоматической обработке геоданных, когда нужно по расписанию обновлять, генерировать или анализировать данные. Возможно также создание таких скриптов, которые автоматически создают проект GRASS, выполняют определенную работу (импортируют данные, анализируют их) и, по завершении обработки, удаляют проекты. Методика создания подобных скриптов подробно описана в статье On scripting GRASS GIS: Building location-independent command line tools. В данной статье рассматривается лишь часть методов, описанных вышеуказанной работе -- создание скрипта, способного использовать готовые проекты GRASS.

Пример подобного скрипта, написанного на shell, приводится в рецепте, данная статья описывает решение данной задачи для языка Python. Нужно отметить, что описываемый подход можно применить к любому языку программирования, а не только к shell или Python.

## Содержание



## Этапы работы

В общем случае внешний по отношению к ГИС GRASS скрипт должен реализовать следующие основные этапы обработки:

1. Задать значения переменным окружения, которые используются в GRASS GIS (PATH, GISRC и т.д.). Этот этап служит инициализацией и необходим для того, чтобы скрипт был в состоянии определить, где находятся файлы проекта и модули ГИС GRASS и, как следствие, мог их использовать в своей работе.
2. Собственно сама обработка данных, состоящая из последовательности команд GRASS GIS. Это основная часть скрипта, в которой пользователь задает последовательность действий, необходимых для того, чтобы выполнить планируемую обработку данных.
3. Очистка системы от временных файлов. После выполнения необходимого анализа или действий над исходными данными все вспомогательные файлы следует удалить, чтобы не "мусорить" в системе.

Данные этапы более подробно рассматриваются в последующих подразделах.

# Инициализация

## Теория

Описание этапа инициализации рассматривается в Wiki проекта GRASS. В этом подразделе приводится краткий перевод этой страницы.

Как было сказано выше, цель этапа инициализации -- установить переменные окружения так, чтобы скрипт мог получить информацию о том, где искать библиотеки GRASS и файлы проекта. Поэтому нужно задать следующие переменные:

- GISBASE путь к каталогу, в котором установлена ГИС GRASS. Узнать значение переменной можно, если запустить GRASS и в консоли выполнить команду: `echo $GISBASE`
- PATH системная переменная окружения $PATH, в которой нужно прописать пути каталогам, в которых следует искать скрипты и команды GRASS.
- LD_LIBRARY_PATH переменная, в которой перечисляются все каталоги содержащие пользовательские динамические библиотеки.
- GISRC переменная, содержащая путь к файлу, в котором задаются путь к проекту, области и набору карт. Для ГИС GRASS 6.x этот файл обычно называется .grassrc6, но можно создать свой собственный и прописать туда необходимые сведения. В последнем случае в файле должны быть установлены значения следующих переменных GISDBASE, LOCATION, MAPSET (посмотреть формат файла можно, открыв его в текстовом редакторе).

Если речь идет о скрипте на языке Python, то нужно задать еще переменные окружения, описывающие, где хранится интерпретатор языка:

- PYTHONLIB путь к интерпретатору Python.
- PYTHONPATH эта переменная должна указывать на %GISBASE%\etc\python.

Например, для системы Windows эти переменые могут быть такими (зависит от конкретных путей установки GRASS):

```
GISBASE= C:\GRASS-64
GISRC= C:\Documents and Settings\user\.grassrc6
LD_LIBRARY_PATH= C:\GRASS-64\lib
PATH= C:\GRASS-64\etc;C:\GRASS-64\etc\python;C:\GRASS-64\lib;C:\GRASS-64\bin;C:\GRASS-
64\extralib;C:\GRASS-64\msys\bin;C:\Python26;
PYTHONLIB= C:\Python26
PYTHONPATH= C:\GRASS-64\etc\python
GRASS_SH= C:\GRASS-64\msys\bin\sh.exe
```

Для Linux переменные могут быть:

```
export GISBASE="/usr/local/grass-6.4.svn/"
export PATH="$PATH:$GISBASE/bin:$GISBASE/scripts"
export LD_LIBRARY_PATH="$LD_LIBRARY_PATH:$GISBASE/lib"
export GISRC="$HOME/.grassrc6"
export PYTHONPATH="$PYTHONPATH:$GISBASE/etc/python"
```

### Реализация

Язык Python несколько облегчает общую схему работы, рассмотренную в предыдущем подразделе, в частности, пользователь не обязан создавать файл с описанием путей проекта и, как следствие, этап очистки системы может быть также пропущен.

Предположим, что пользователь работает с набором данных geosample, и пусть $HOME/grassdata -- каталог, в котором хранится данный набор. Тогда для установки переменных можно воспользоваться следующими командами на языке Python:

```
import os
import sys
gisbase = os.environ['GISBASE'] = "/usr/lib/grass64"
gisdbase = os.path.join(os.environ['HOME'], "grassdata")
location = "geosample"
mapset   = "PERMANENT"
sys.path.append(os.path.join(os.environ['GISBASE'], "etc", "python"))
import grass.script as grass
import grass.script.setup as gsetup
gsetup.init(gisbase,
            gisdbase, location, mapset)
print grass.gisenv()
```

Третья строка задает переменную окружения GISBASE. Четвертая, пятая и шестая строки задают переменные путей к проекту GRASS, которые будут использоваться в команде *gsetup.init*. Обратите внимание на то, что использование команды *gsetup.init* избавляет программиста от создания/редактирования файла путей к проекту (см. переменную GISRC).

## Обработка данных

Здесь происходит собственно обработка данных, которая зависит от целей и задач пользователя. Пользователь может вызывать любые команды GRASS или запускать сторонние модули Python или написать свой собственный обработчик.

Поскольку цель данной статьи -- продемонстрировать этапы предварительной подготовки, необходимые для вызова GRASS GIS из скрипта на Python, а не выполнение того или иного анализа данных, то в качестве пример обработки будем использовать "заглушку", т.е. демонстрационный пример вызова команд GRASS GIS. В качестве такого простого примера покажем, как можно произвести циклическую обработку растров. Следующий кусок кода производит поиск всех растровых файлов и:

1. Выводит называние растра на экран
2. Выводит статистику по данному растру (значение ячейки растра и общая площадь ячеек с данным значением)

```
grass.message('Raster maps:')
for rast in grass.list_strings(type = 'rast'):
    print rast
    grass.run_command('r.stats', input=rast, flags='a')
```

Читатель легко может заменить эти команды на свои, отражающие суть необходимых ему манипуляций с данными.

## Очистка системы

Если пользователь в своем скрипте создавал те или иные вспомогательные файлы, то он должен удалить их по окончании работы скрипта. В первую очередь это относится к файлу с указанием путей к проекту, который хранится в переменной GISRC. Однако, если пользователь для настройки путей проекта использовал команду gsetup.init, то данный шаг можно опустить.

## Итоговый скрипт

Окончательный вид скрипта, готового для использования, приводится ниже:

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-

import os
import sys
gisbase = os.environ['GISBASE'] = "/usr/lib/grass64"
gisdbase = os.path.join(os.environ['HOME'], "grassdata")
location = "geosample"
mapset   = "PERMANENT"
sys.path.append(os.path.join(os.environ['GISBASE'], "etc", "python"))
import grass.script as grass
import grass.script.setup as gsetup
gsetup.init(gisbase,
        gisdbase, location, mapset)
print grass.gisenv()

# Обработка
grass.message('Raster maps:')
for rast in grass.list_strings(type = 'rast'):
    print rast
    grass.run_command('r.stats', input=rast, flags='a')
```

# Ссылки

- Working with GRASS without starting it explicitly (англ.)
- Рецепт создания внешнего для ГИС GRASS скрипта на shell

Обсудить в форуме Комментариев — 14

Последнее обновление: 2014-05-15 01:56

Дата создания: 30.09.2013
Автор(ы): KolesovDmitry