

Reporte de Laboratorio 0

Manejo de versiones, compilación y documentación

Dunia Barahona - B40806

11 de septiembre de 2016

Índice

1. Código	1
2. GitHub	3
3. Conclusiones	10

1. Código

1. Se subieron a repositorios tres programas y su documentación:

```
1  #!/usr/bin/python
2  # -*- coding: utf-8 -*-
3
4  import sys
5
6  ##
7  # @file
8  # @author Dunia Barahona <s4sibarahona@gmail.com>
9  # @version 1.0
10 # @section DESCRIPCIÓN
11 #
12 # Este es un programa escrito en Python que toma de la línea de comandos una lista de
13 # números reales e imprime en pantalla el resultado de la suma de dichos números.
14
15 ## @return el resultado de la suma de los números ingresados.
16 def suma_n():
17     ## Variable de tipo entero.
18     bolsita = 0
19     for x in range(1, len(sys.argv)):
20         bolsita += int(sys.argv[x])
21     print bolsita
22
23
24 suma_n()
```

Figura 1: suma.py

```

1  #include <stdio.h>
2  #include <stdlib.h>
3  /**
4   * @file
5   * @author Dunia Barahona <s4si@hotmail.com>
6   * @version 1.0
7   *
8   * @section DESCRIPCIÓN
9   *
10  * Este es un programa escrito en C que toma de la línea de comandos una lista de
11  * números reales e imprime en pantalla el resultado de la suma de dichos números.
12  */
13
14  /**
15   * @param cant argumento de tipo entero, es la cantidad de elementos que recibe.
16   * @param **ents un puntero de constantes, son los valores recibidos.
17   * @return el resultado de la suma de dichos números.
18  */
19  int sum(int cant, char **ents){
20      int i=0;
21      double bolsita=0;
22      for(i; i<cant; i++){
23          bolsita += atof(ents[i]);
24      }
25      int temp= ("%i", bolsita);
26      printf("%i\n", temp);
27      return 0;
28  }
29  /**
30   * Esta es la función principal.
31   * @param argc es la cantidad de elementos que recibe.
32   * @param **argv son los valores recibidos.
33  */
34  int main(int argc, char **argv) {
35      sum(argc, argv);
36  }

```

Figura 2: suma.c

```

3  #include "iostream"
4  #include "string"
5  using namespace std;
6  /**
7   * @file
8   * @author Dunia Barahona <s4si@hotmail.com>
9   * @version 1.0
10  *
11  * @section DESCRIPCIÓN
12  *
13  * Este es un programa escrito en C++ que toma de la línea de comandos una lista de
14  * números reales e imprime en pantalla el resultado de la suma de dichos números.
15  */
16
17  /**
18  * @param cant argumento de tipo entero, es la cantidad de elementos que recibe.
19  * @param **ents un puntero de constantes, son los valores recibidos.
20  * @return el resultado de la suma de dichos números.
21  */
22  int sum(int cant, char **ents){
23      int i=0;
24      double bolsita=0;
25      for(i; i<cant; i++){
26          bolsita += atof(ents[i]);
27      }
28      cout <<bolsita<< endl; /*imprime*/
29      return 0;
30  }
31  /**
32  * Esta es la función principal.
33  * @param argc es la cantidad de elementos que recibe.
34  * @param **argv son los valores recibidos.
35  */
36  int main(int argc, char **argv) {
37      sum(argc, argv);
38  }

```

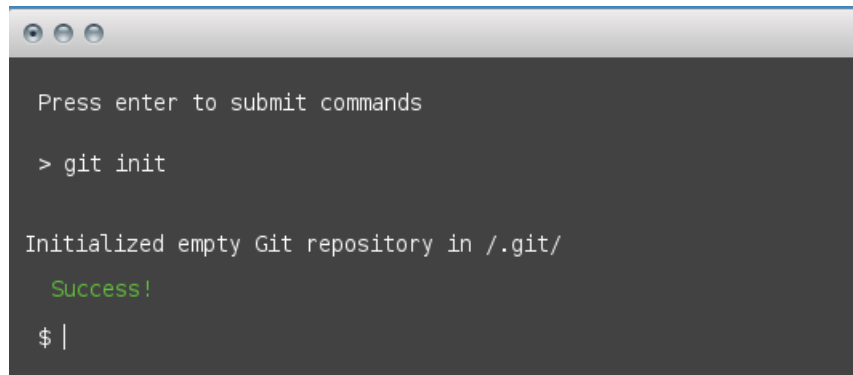
Figura 3: suma.cpp

2. Se creó un **Makefile** con cuatro *targets*:

- Compilar.
- Borrar.
- Ejecutar.
- All: este se encarga de correr los primeros tres en orden.

2. GitHub

Documentación del tutorial:

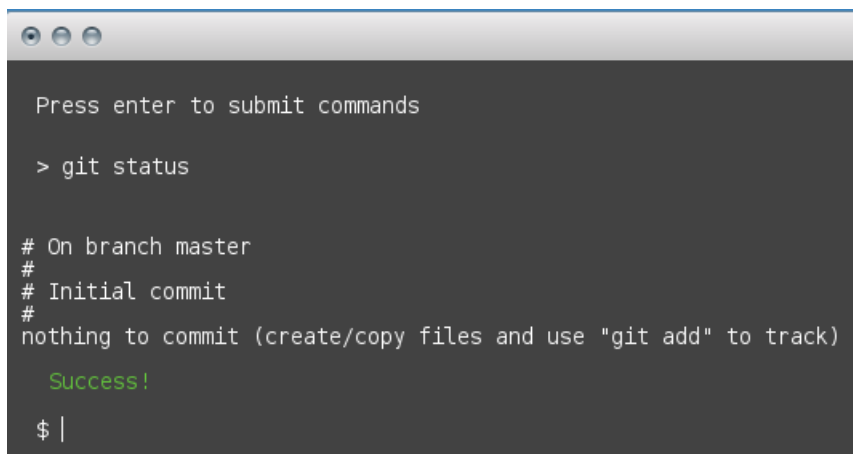


```
Press enter to submit commands

> git init

Initialized empty Git repository in /.git/
  Success!
$ |
```

Figura 4: Inicialización de repositorio

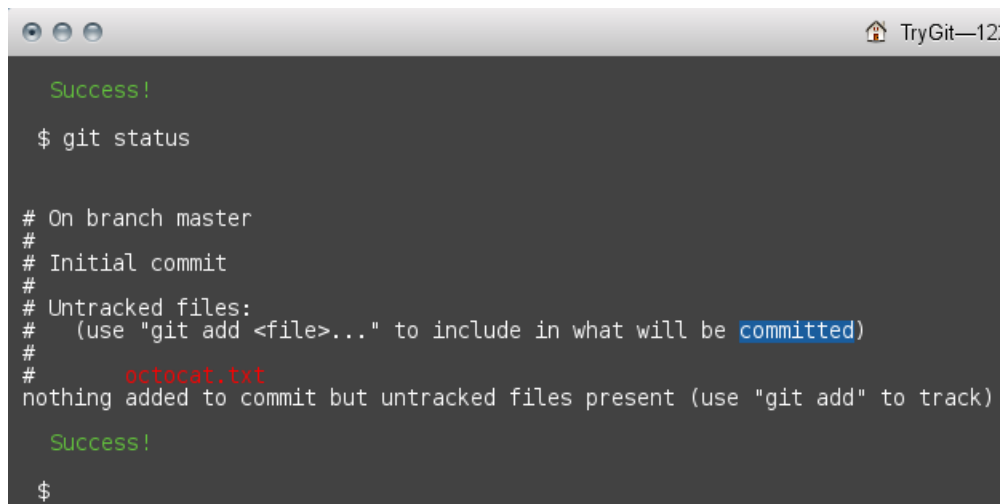


```
Press enter to submit commands

> git status

# On branch master
#
# Initial commit
#
nothing to commit (create/copy files and use "git add" to track)
  Success!
$ |
```

Figura 5: git status



```
TryGit—12

  Success!
$ git status

# On branch master
#
# Initial commit
#
# Untracked files:
#   (use "git add <file>..." to include in what will be committed)
#
#       octocat.txt
nothing added to commit but untracked files present (use "git add" to track)
  Success!
$
```

Figura 6: git status

```
$ git add octocat.txt

Nice job, you've added octocat.txt to the Staging Area

$ git status

# On branch master
#
# Initial commit
#
# Changes to be committed:
#   (use "git rm --cached <file>..." to unstage)
#
#       new file:   octocat.txt
#
```

Figura 7: Hacer que revise cambios en archivo

```
$ git add '*.txt'

Success!

$ git status

# On branch master
# Changes to be committed:
#   (use "git reset HEAD <file>..." to unstage)
#
#       new file:   blue_octocat.txt
#       new file:   octofamily/baby_octocat.txt
#       new file:   octofamily/momma_octocat.txt
#       new file:   red_octocat.txt
#
```

Figura 8: Hacer que revise cambios en todos los archivos .txt

```
$ git commit -m 'Add all the octocat txt files'

[master 3852b4d] Add all the octocat txt files
4 files changed, 4 insertions(+)
create mode 100644 blue_octocat.txt
create mode 100644 octofamily/baby_octocat.txt
create mode 100644 octofamily/momma_octocat.txt
create mode 100644 red_octocat.txt

Success!

$ git log

commit 3852b4db1634463d0bb4d267edb7b3f9cd02ace1
Author: Try Git <try_git@github.com>
Date: Sat Oct 10 08:30:00 2020 -0500

    Add all the octocat txt files
```

Figura 9: git commit

```
$ git remote add origin https://github.com/try-git/try_git.git

Success!

$ git push -u origin master

Branch master set up to track remote branch master from origin.
Success!

$ git pull origin master

Updating 3852b4d..3e70b0f
```

Figura 10: Repositorio remoto

```
$ git diff HEAD

diff --git a/octocat.txt b/octocat.txt
index 7d8d808..e725ef6 100644
--- a/octocat.txt
+++ b/octocat.txt
@@ -1,1 @@
-A Tale of Two Octocats
+[m]A Tale of Two Octocats and an Octodog

Success!

$ git add octofamily/octodog.txt

Success!
```

Figura 11: Diferencias desde el último commit

```
~~~~~

$ git diff --staged

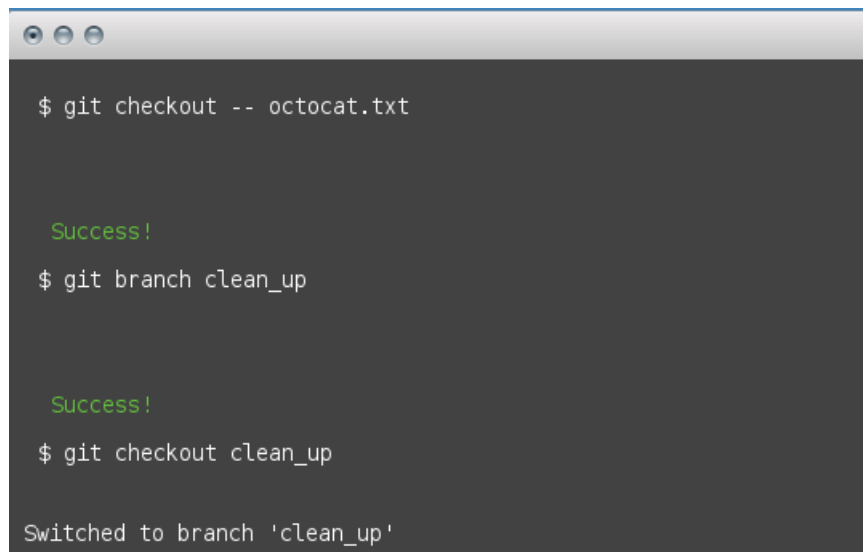
diff --git a/octofamily/octodog.txt b/octofamily/octodog.txt
new file mode 100644
index 0000000..cfbc74a
--- /dev/null
+++ b/octofamily/octodog.txt
@@ -0,0 +1 @@
+[m]woof

Success!

$ git reset octofamily/octodog.txt

Success!
```

Figura 12: Cambios que se han dado



```
$ git checkout -- octocat.txt

Success!

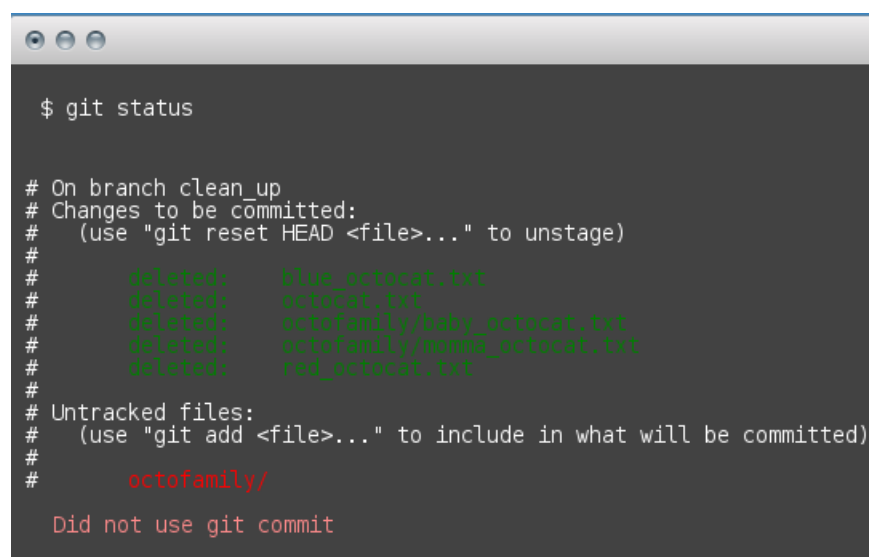
$ git branch clean_up

Success!

$ git checkout clean_up

Switched to branch 'clean_up'
```

Figura 13: git checkout



```
$ git status

# On branch clean_up
# Changes to be committed:
#   (use "git reset HEAD <file>..." to unstage)
#
#       deleted:    blue_octocat.txt
#       deleted:    octocat.txt
#       deleted:    octofamily/baby_octocat.txt
#       deleted:    octofamily/nonna_octocat.txt
#       deleted:    red_octocat.txt
#
# Untracked files:
#   (use "git add <file>..." to include in what will be committed)
#
#       octofamily/
#
Did not use git commit
```

Figura 14: git status


```
$ git commit -m "Remove all the cats"

[clean_up 63540fe] Remove all the cats
5 files changed, 5 deletions(-)
delete mode 100644 blue_octocat.txt
delete mode 100644 octocat.txt
delete mode 100644 octofamily/baby_octocat.txt
delete mode 100644 octofamily/momma_octocat.txt
delete mode 100644 red_octocat.txt

Success!

$ git checkout master

Switched to branch 'master'

Success!
```

Figura 15: git commit

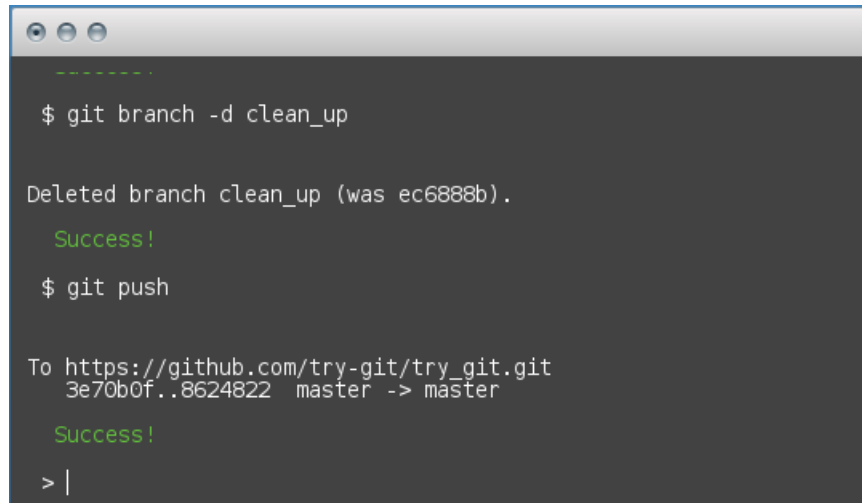
```
$ git merge clean_up

Updating 3852b4d..ec6888b
Fast-forward
 blue_octocat.txt      | 1 -
  octocat.txt          | 1 -
 octofamily/baby_octocat.txt | 1 -
 octofamily/momma_octocat.txt | 1 -
  red octocat.txt      | 1 -
5 files changed, 5 deletions(-)
delete mode 100644 blue_octocat.txt
delete mode 100644 octocat.txt
delete mode 100644 octofamily/baby_octocat.txt
delete mode 100644 octofamily/momma_octocat.txt
delete mode 100644 red_octocat.txt

Success!

$
```

Figura 16: Unión de *branches*

A terminal window with a dark background and light text. The window has three small circular icons in the top-left corner. The text inside the terminal shows the execution of two git commands. The first command is 'git branch -d clean_up', which results in the message 'Deleted branch clean_up (was ec6888b).' followed by 'Success!'. The second command is 'git push', which results in the message 'To https://github.com/try-git/try_git.git' followed by '3e70b0f..8624822 master -> master' and 'Success!'. The prompt '> |' is visible at the bottom.

```
-----  
$ git branch -d clean_up  
  
Deleted branch clean_up (was ec6888b).  
  
Success!  
  
$ git push  
  
To https://github.com/try-git/try_git.git  
3e70b0f..8624822 master -> master  
  
Success!  
  
> |
```

Figura 17: Eliminar rama

3. Conclusiones

1. Doxygen es una herramienta muy completa, que facilita el proceso de documentación.
2. \LaTeX es un programa que permite crear documentos de buena calidad de manera sencilla.
3. GitHub requiere de práctica para hacer uso de todas las opciones que ofrece.