

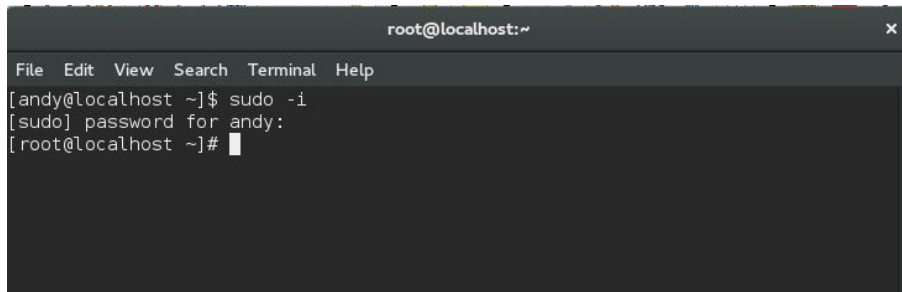
Universidad de San Carlos de Guatemala
Facultad de Ingenieria
Escuela de Ciencias
Analisis y Diseño 2

MANUAL DE INSTALACION DE GIT

Andrea Virginia Chavarría Guzmán
2009-20081

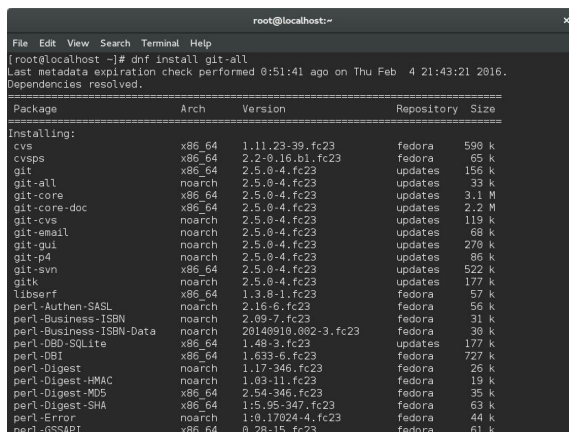
Para este manual se usará como base Fedora 23 de OS.

Paso1: Nos logueamos como root ya sea
`sudo -i`
`sudo su`



```
root@localhost:~  
File Edit View Search Terminal Help  
[andy@localhost ~]$ sudo -i  
[sudo] password for andy:  
[root@localhost ~]#
```

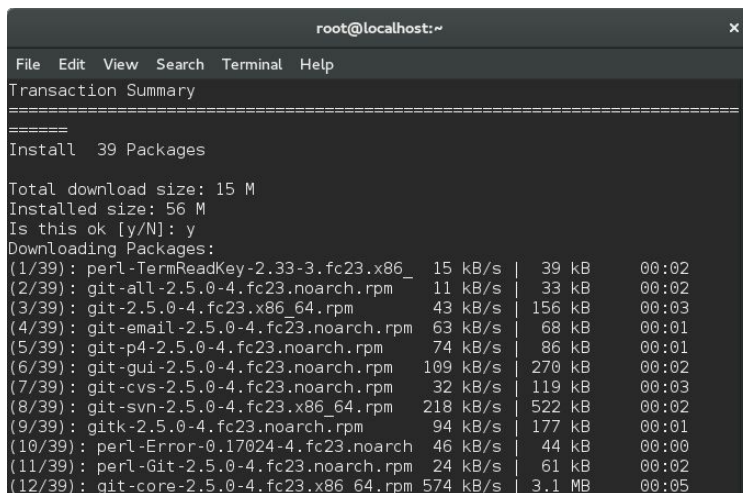
Paso2: Procedemos a instalar Git con el comando siguiente
`dnf install git-all`



```
root@localhost:~  
File Edit View Search Terminal Help  
[root@localhost ~]# dnf install git-all  
Last metadata expiration check performed 0:51:41 ago on Thu Feb 4 21:43:21 2016.  
Dependencies resolved.  
=====
```

Package	Arch	Version	Repository	Size
Installing:				
cvs	x86_64	1.11.23-39.fc23	fedora	598 k
cvsps	x86_64	2.2-9.10.b1.fc23	fedora	65 k
git	x86_64	2.5.0-4.fc23	updates	156 k
git-all	noarch	2.5.0-4.fc23	updates	33 k
git-core	x86_64	2.5.0-4.fc23	updates	3.1 M
git-core-doc	x86_64	2.5.0-4.fc23	updates	2.2 M
git-cvs	noarch	2.5.0-4.fc23	updates	119 k
git-email	noarch	2.5.0-4.fc23	updates	68 k
git-gui	noarch	2.5.0-4.fc23	updates	270 k
git-p4	noarch	2.5.0-4.fc23	updates	86 k
git-svn	x86_64	2.5.0-4.fc23	updates	522 k
gitk	noarch	2.5.0-4.fc23	updates	177 k
libserf	x86_64	1.3.8-1.fc23	fedora	57 k
perl-Authen-SASL	noarch	2.16-6.fc23	fedora	56 k
perl-Business-ISBN	noarch	2.09-7.fc23	fedora	31 k
perl-Business-ISBN-Data	noarch	20140910.002-3.fc23	fedora	30 k
perl-DBD-SQLite	x86_64	1.48-3.fc23	updates	177 k
perl-DBI	x86_64	1.633-6.fc23	fedora	727 k
perl-Digest	noarch	1.17-346.fc23	fedora	26 k
perl-Digest-HMAC	noarch	1.03-11.fc23	fedora	19 k
perl-Digest-MD5	x86_64	2.54-346.fc23	fedora	35 k
perl-Digest-SHA	x86_64	1:5.95-347.fc23	fedora	63 k
perl-Error	noarch	1:0.17024-4.fc23	fedora	44 k
perl-GSSAPI	x86_64	0.29-15.fc23	fedora	61 k

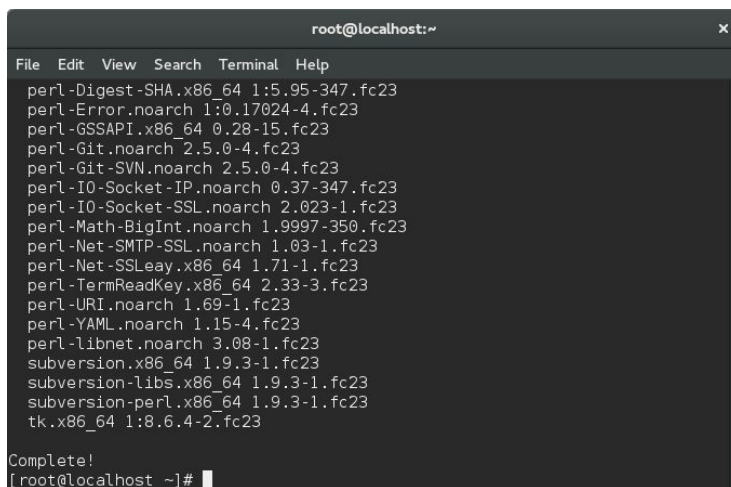
Nos preguntara si queremos instalar colocamos Y y comenzara a descargar los paquetes.



```
root@localhost:~  
File Edit View Search Terminal Help  
Transaction Summary  
=====
```

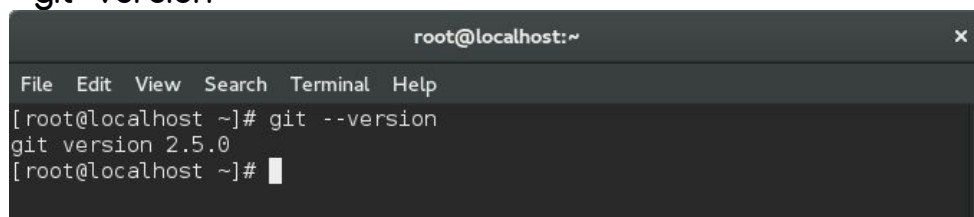
Install 39 Packages			
Total download size: 15 M			
Installed size: 56 M			
Is this ok [y/N]: y			
Downloading Packages:			
(1/39):	perl-TermReadKey-2.33-3.fc23.x86_	15 kB/s 39 kB	00:02
(2/39):	git-all-2.5.0-4.fc23.noarch.rpm	11 kB/s 33 kB	00:02
(3/39):	git-2.5.0-4.fc23.x86_64.rpm	43 kB/s 156 kB	00:03
(4/39):	git-email-2.5.0-4.fc23.noarch.rpm	63 kB/s 68 kB	00:01
(5/39):	git-p4-2.5.0-4.fc23.noarch.rpm	74 kB/s 86 kB	00:01
(6/39):	git-gui-2.5.0-4.fc23.noarch.rpm	109 kB/s 270 kB	00:02
(7/39):	git-cvs-2.5.0-4.fc23.noarch.rpm	32 kB/s 119 kB	00:03
(8/39):	git-svn-2.5.0-4.fc23.x86_64.rpm	218 kB/s 522 kB	00:02
(9/39):	gitk-2.5.0-4.fc23.noarch.rpm	94 kB/s 177 kB	00:01
(10/39):	perl-Error-0.17024-4.fc23.noarch	46 kB/s 44 kB	00:00
(11/39):	perl-Git-2.5.0-4.fc23.noarch.rpm	24 kB/s 61 kB	00:02
(12/39):	git-core-2.5.0-4.fc23.x86_64.rpm	574 kB/s 3.1 MB	00:05

Nos mostrara lo siguiente que la instalación a finalizado.



```
root@localhost:~  
File Edit View Search Terminal Help  
perl-Digest-SHA.x86_64 1:5.95-347.fc23  
perl-Error.noarch 1:0.17024-4.fc23  
perl-GSSAPI.x86_64 0.28-15.fc23  
perl-Git.noarch 2.5.0-4.fc23  
perl-Git-SVN.noarch 2.5.0-4.fc23  
perl-IO-Socket-IP.noarch 0.37-347.fc23  
perl-IO-Socket-SSL.noarch 2.023-1.fc23  
perl-Math-BigInt.noarch 1.9997-350.fc23  
perl-Net-SMTP-SSL.noarch 1.03-1.fc23  
perl-Net-SSLeay.x86_64 1.71-1.fc23  
perl-TermReadKey.x86_64 2.33-3.fc23  
perl-URI.noarch 1.69-1.fc23  
perl-YAML.noarch 1.15-4.fc23  
perl-libnet.noarch 3.08-1.fc23  
subversion.x86_64 1.9.3-1.fc23  
subversion-libs.x86_64 1.9.3-1.fc23  
subversion-perl.x86_64 1.9.3-1.fc23  
tk.x86_64 1:8.6.4-2.fc23  
  
Complete!  
[root@localhost ~]#
```

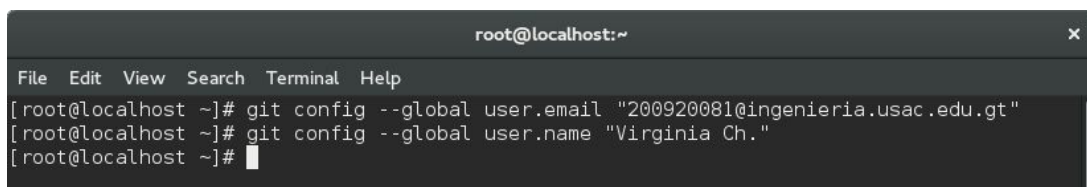
Paso 3: Para hacer una verificación que Git esta instalado lo hacemos así
git --version



```
root@localhost:~  
File Edit View Search Terminal Help  
[root@localhost ~]# git --version  
git version 2.5.0  
[root@localhost ~]#
```

Configuración de Git

Ya con Git instalado debemos ingresar información de nosotros para que los mensajes de commit sean generados con la información correcta, para esto usamos el comando **git config**
git config --global user.name "Tu nombre"
git config --global user.email "tumail@mail.com"



```
root@localhost:~  
File Edit View Search Terminal Help  
[root@localhost ~]# git config --global user.email "200920081@ingenieria.usac.edu.gt"  
[root@localhost ~]# git config --global user.name "Virginia Ch."  
[root@localhost ~]#
```

Verificamos que los datos han sido añadidos correctamente así
git config --list

```
root@localhost:~  
File Edit View Search Terminal Help  
[root@localhost ~]# git config --list  
user.name=Virginia Ch.  
user.email=200920081@ingenieria.usac.edu.gt  
[root@localhost ~]#
```

Usando git

Teniendo un archivo en mi caso un html nos colocamos en la carpeta donde se encuentra nuestro proyecto le decimos a git que queremos esta area como nuestro ambiente de trabajo así:

git init

```
andy@localhost:~/Documents/Ejemplo  
File Edit View Search Terminal Help  
[andy@localhost Ejemplo]$ git init  
Initialized empty Git repository in /home/andy/Documents/Ejemplo/.git/  
[andy@localhost Ejemplo]$
```

Ahora para añadir archivos al repositorio de git creado usamos **git add**. Con este comando añadimos todos los archivos que se encuentran en la carpeta ejemplo y para verificar los añadidos usamos **git status**

```
andy@localhost:~/Documents/Ejemplo  
File Edit View Search Terminal Help  
[andy@localhost Ejemplo]$ git add .  
[andy@localhost Ejemplo]$ git status  
On branch master  
  
Initial commit  
  
Changes to be committed:  
  (use "git rm --cached <file>..." to unstage)  
  
    new file:   index.html  
[andy@localhost Ejemplo]$
```

Luego hacemos un commit este es un pequeño mensaje explicando los cambios que se han realizado
git commit -m "Mensaje"

```
andy@localhost:~/Documents/Ejemplo
File Edit View Search Terminal Help
[andy@localhost Ejemplo]$ git commit -m "Prueba"
[master 79a4c0b] Prueba
```

En el commit actual al que estamos es conocido como HEAD para ver este usamos **git show HEAD**

```
andy@localhost:~/Documents/Ejemplo
File Edit View Search Terminal Help
commit 79a4c0b50db06a332da590d00470d781c9ed6beb
Author: Virginia Ch <200920081@ingenieria.usac.edu.gt>
Date: Fri Feb 5 17:12:20 2016 -0600

Prueba

diff --git a/index.html b/index.html
index 3b9c0c2..f83b8b7 100755
--- a/index.html
+++ b/index.html
@@ -12,11 +12,6 @@

<P>Esto es un parrafo con informacion
super importante. Notese que las lineas salen pegadas aun dejando
espacios, saltos de linea, etc. <BR> Si pongo esto
:
```

Podemos ver tambien los cambios sobre todo (historial)
git log

```
andy@localhost:~/Documents/Ejemplo
File Edit View Search Terminal Help
[andy@localhost Ejemplo]$ git log
commit 79a4c0b50db06a332da590d00470d781c9ed6beb
Author: Virginia Ch <200920081@ingenieria.usac.edu.gt>
Date: Fri Feb 5 17:12:20 2016 -0600

Prueba

commit 7f50a3720d44fd19344833257dc37b3d52e84d80
Author: andy <andy@localhost.localdomain>
Date: Fri Feb 5 17:07:43 2016 -0600

First commit
[andy@localhost Ejemplo]$
```

Para crear una rama se utiliza el comando
git branch nombrerama

```
andy@localhost:~/Documents/Ejemplo
File Edit View Search Terminal Help
[andy@localhost Ejemplo]$ git branch develop
```

Para cambiar de una rama a otra se puede hacer de la siguiente manera
git checkout nombrerama

```
andy@localhost:~/Documents/Ejemplo
File Edit View Search Terminal Help
[andy@localhost Ejemplo]$ git checkout develop
Switched to branch 'develop'
[andy@localhost Ejemplo]$
```

Para recuperar el estado de un comit anterior se utiliza
git checkout SHA(los primeros 7)

```
andy@localhost:~/Documents/Ejemplo
File Edit View Search Terminal Help
[andy@localhost Ejemplo]$ git log
commit 13638b6d52a121c05ab67d68523b278cec283e20
Author: Virginia Ch <200920081@ingenieria.usac.edu.gt>
Date: Fri Feb 5 17:55:14 2016 -0600

    otro commit

commit 7f50a3720d44fd19344833257dc37b3d52e84d80
Author: andy <andy@localhost.localdomain>
Date: Fri Feb 5 17:07:43 2016 -0600

    First commit
[andy@localhost Ejemplo]$ git checkout 7f50a37
Note: checking out '7f50a37'.

You are in 'detached HEAD' state. You can look around, make experimental
changes and commit them, and you can discard any commits you make in this
state without impacting any branches by performing another checkout.

If you want to create a new branch to retain commits you create, you may
do so (now or later) by using -b with the checkout command again. Example:

    git checkout -b <new-branch-name>

HEAD is now at 7f50a37... First commit
[andy@localhost Ejemplo]$
```