theserverside.com

List and show Git config settings

Cameron McKenzie TechTarget 07 Jun 2020

3-4 minutes

Git has four, and arguably five, separate scopes in which variables can be set. Git cascades through each of these scopes when applying variables, meaning locally scoped variables override global ones and global Git config variables override system ones. All of this can lead to issues when troubleshooting problems with credential managers or added Git remotes, as one can never be quite sure where a given Git variable has been set.

If a variable is defined in multiple scopes, there's an easy way to show what Git config is using at runtime. Just call the git config –get shows the name of the variable.

The follow command will have Git config show the user.email property used by a repository for local commits:

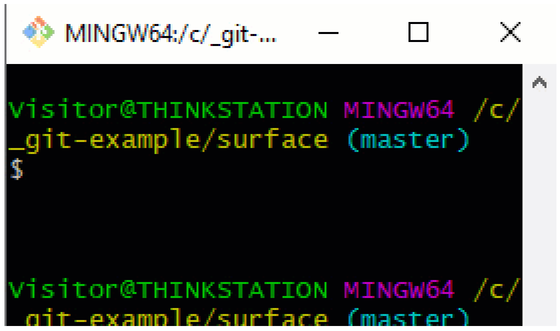
```
config@list (master)
$ git config --get user.email
local@example.com
```

Global git config list example

The git config list command will show all Git config properties throughout all of the variously scoped Git files.

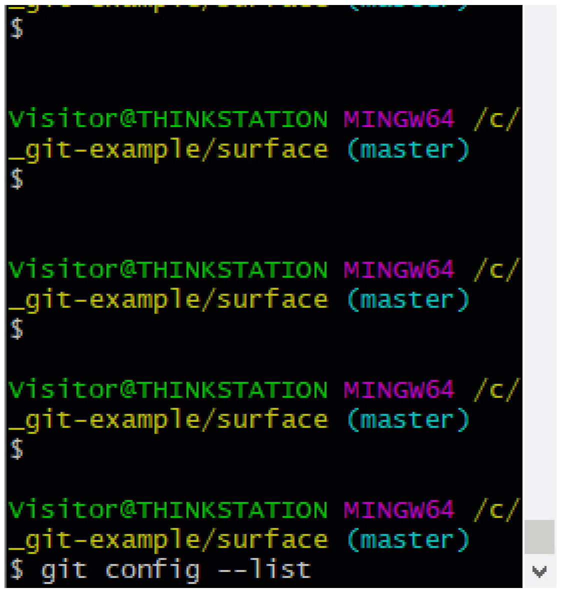
If the value displayed is different from the value expected, then an override is happening in one of the other Git config scopes. The Git config scopes you will commonly encounter include:

- System Git config scope
- Global Git config scope
- Local Git config scope
- Worktree Git config scope
- Portable Git config scope on a Windows machine



```
Visitor@THINKSTATION MINGW64 /c/
_git-example/surface (master)
$


Visitor@THINKSTATION MINGW64 /c/
_git-example/surface (master)
```



```
$

Visitor@THINKSTATION MINGW64 /c/
_git-example/surface (master)
$

Visitor@THINKSTATION MINGW64 /c/
_git-example/surface (master)
$

Visitor@THINKSTATION MINGW64 /c/
_git-example/surface (master)
$

Visitor@THINKSTATION MINGW64 /c/
_git-example/surface (master)
$ git config --list
```

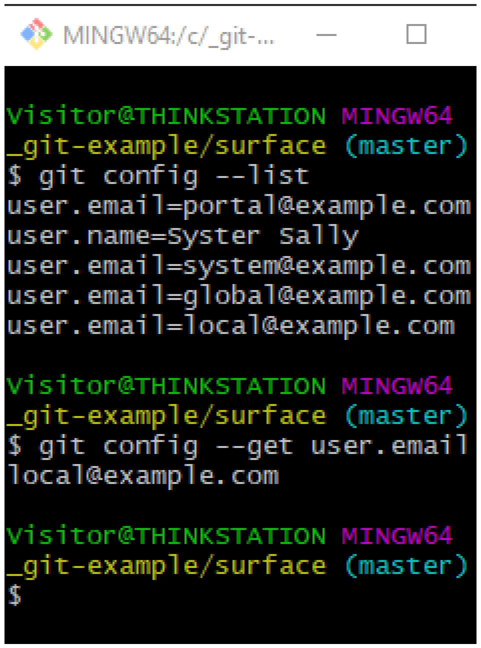
Inspect property values with git config list and get.

Note that the worktree scope is not available if you have not done a Git worktree add in your local project.

If you believe a variable has been set in multiple scopes, you can use the git config –list to show all of the different values assigned to a property.

```
config@list (master)
$ git config --list
user.email=portal@example.com
user.name=Syster Sally
user.email=system@example.com
user.email=global@example.com
user.email=local@example.com
```

Show global git config settings



```
Visitor@THINKSTATION MINGW64
_git-example/surface (master)
$ git config --list
user.email=portal@example.com
user.name=Syster Sally
user.email=system@example.com
user.email=global@example.com
user.email=local@example.com

Visitor@THINKSTATION MINGW64
_git-example/surface (master)
$ git config --get user.email
local@example.com

Visitor@THINKSTATION MINGW64
_git-example/surface (master)
$
```

The git config list command will display values in gitconfig files.

As you can see when we show the Git config user.email property that it is set in all but the optional worktree scope. But at runtime, only the value set locally is used.

If you would like to delete or edit a Git config value manually with a text editor, you can find the Git config file locations through the Git config list command's –show-origin switch. this will tell you not only the names and values of git config properties but will also show Git config file locations for each setting.

```
config@list (master)
$ git config --list --show-origin
file:"C:\\ProgramData/Git/config"
user.email=portal@example.com
file:C:/_git/mingw64/etc/gitconfig
user.name=Syster Sally
file:C:/_git/mingw64/etc/gitconfig
user.email=system@example.com
file:C:/Users/Owner/.gitconfig
user.email=global@example.com
file:.git/config user.email=local@example.com
```

It's not hard to get a command window to show local and global Git config file settings. With the power of the git config list command and its various switches at your fingertips, you should have no troubleshooting any problems that arise from gitconfig property settings errors.