

# INSTITUTO PROVINCIAL DE LA ADMINISTRACIÓN PÚBLICA (IPAP)

---

2022

**IPAP**

SUBSECRETARÍA DE  
GESTIÓN Y EMPLEO PÚBLICO

JEFATURA DE  
GABINETE



GOBIERNO DE LA PROVINCIA DE  
**BUENOS AIRES**

IPAP | PROGRAMA AGENTES DEL ESTADO

# Git y herramientas libres

Agustin Parmisano Sabbione  
2022

# Clase 1

1. Qué es Git. Utilidad e importancia de Git como software libre.
2. Concepto de control de versionado de software.
3. Herramientas libres que usan Git.
4. Utilización en la Administración Pública, ejemplos.
  - a. Github.
  - b. Gitlab.
5. Entornos visuales para Git.
6. Complementos Git para Visual Studio Code.

# Qué es Git.

**Git** es un **software de control de versiones** diseñado por Linus Torvalds (creador del Kernel de Linux), pensando en la **eficiencia**, la **confiabilidad** y **compatibilidad** del **mantenimiento de versiones** de aplicaciones cuando éstas tienen un gran número de **archivos de código fuente**.

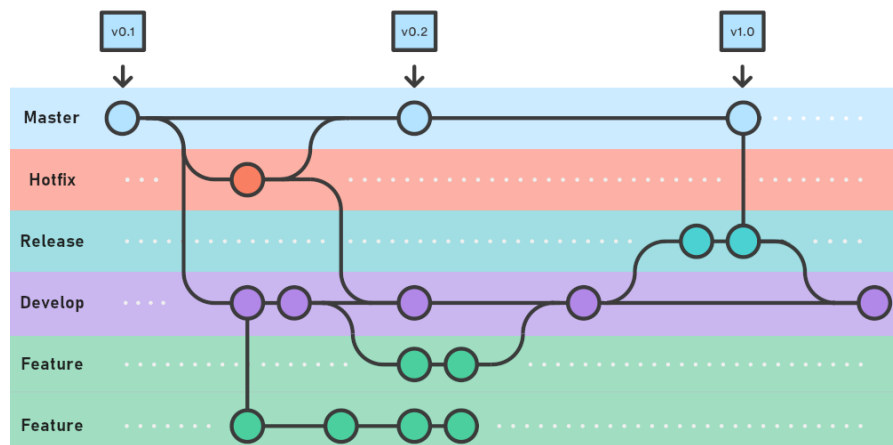


# Utilidad e importancia de Git como software libre

- **Creado por Linux Torvalds, creador del Kernel Linux, en 2005.**
  - Se desprende de la comunidad de desarrollo de Linux.
  - Diseñado para realizar el control de versiones del kernel Linux.
- **Objetivos de Git:**
  - Lograr agilizar el desarrollo.
  - Dar soporte a desarrollos no lineales (múltiples ramas de desarrollo paralelas).
  - Ser 100% Distribuído.
  - Manejar grandes proyectos efectivamente.

# Concepto de control de versionado de software

Se llama control de versiones a la gestión de los diversos cambios que se realizan sobre los elementos de algún producto o una configuración del mismo. Una versión, revisión o edición de un producto, es el estado en el que se encuentra el mismo en un momento dado de su desarrollo o modificación.



# Concepto de control de versionado de software

Los Software de Control de Versiones (VCS) se dividen en dos grandes grupos:

- **VCS Centralizados:** Subversion, Perforce, etc
- **VCS Distribuídos:** Git, Mercurial, etc



PERFORCE



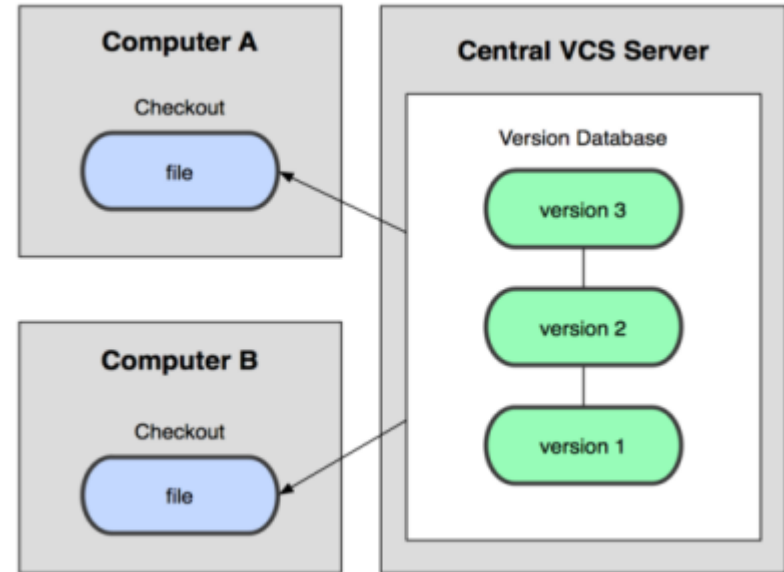
git



mercurial

# VCS Centralizados

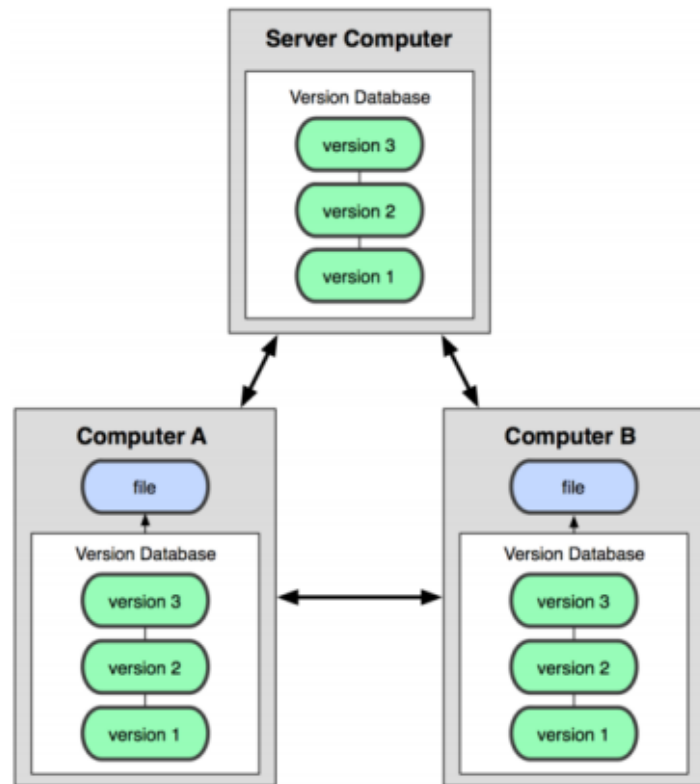
- En Subversion, CVS, Perforce, etc. Existe un servidor como repositorio central (repo) que almacena la “copia oficial” del código.
  - El servidor mantiene la única versión de la historia del repo.
- Se realizan “checkouts” de la única versión del repo en una copia local.
  - Se realizan modificaciones locales
  - Los cambios no son versionados.
- Cuando se terminan de realizar los cambios, se realiza un “check in” devuelta al servidor
  - El check in incrementa la versión del repo.





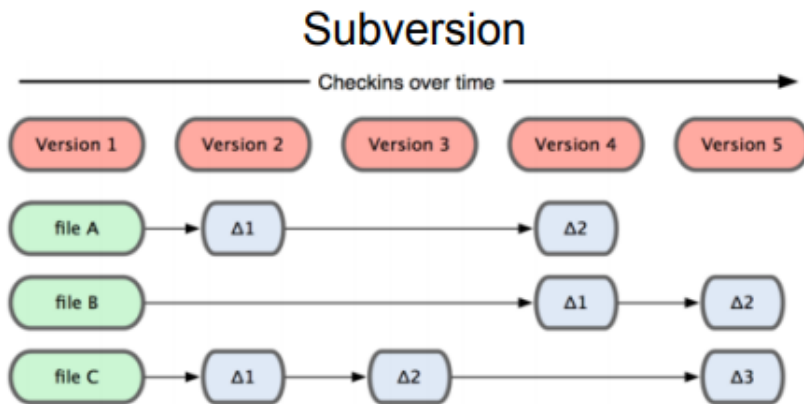
# VCS Distribuido (Git)

- En Git, mercurial, etc., no se realiza un “checkout” de un repo central.
  - Se realizan “clone” y “pull” de los cambios del repo.
- El repo local es una copia completa de todo lo que hay en el servidor (repo) remoto
- Muchas operaciones son locales:
  - Check in/out del repo local.
  - commit de cambios al repo local.
  - El repo local mantiene la versión de la historia.
- Cuando están listos los cambios se pueden realizar un “push” de los cambios hacia el servidor remoto

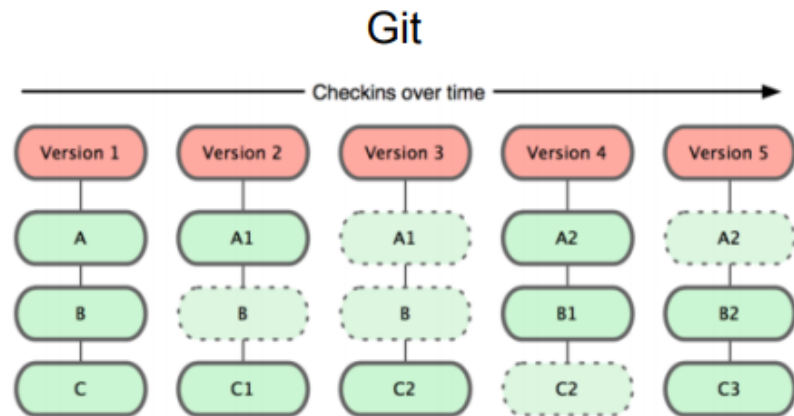


# Git Snapshots (Instantáneas)

- VCS Centralizados como Subversión rastrea los datos de la versión en cada archivo individual.



- Git mantiene "instantáneas" de todo el estado del proyecto.
  - Cada versión de check in del código general tiene una copia de cada archivo.
  - Algunos archivos cambian en un checkin dado, otros no.
  - Más redundancia, pero más rápido.



# Herramientas libres que usan Git

**Linux Kernel** (<https://github.com/torvalds/linux>): Linux es un núcleo mayormente libre semejante al núcleo de Unix. Es uno de los principales ejemplos de software libre y de código abierto.



**Tensorflow** (<https://github.com/tensorflow/tensorflow>): es una biblioteca de código abierto para aprendizaje automático a través de un rango de tareas, y desarrollado por Google para satisfacer sus necesidades de sistemas capaces de construir y entrenar redes neuronales para detectar y descifrar patrones y correlaciones, análogos al aprendizaje y razonamiento usados por los humanos.



# Herramientas libres que usan Git

**Kubernetes(K8s)** (<https://github.com/kubernetes/kubernetes>):

Kubernetes es una plataforma de sistema distribuido de código libre para la automatización del despliegue, ajuste de escala y manejo de aplicaciones en contenedores que fue originalmente diseñado por Google y donado a la Cloud Native Computing Foundation.



**kubernetes**

**Ansible** (<https://github.com/ansible/ansible>): Ansible es una plataforma de software libre para configurar y administrar ordenadores. Combina instalación multi-nodo, ejecuciones de tareas ad hoc y administración de configuraciones. Adicionalmente, Ansible es categorizado como una herramienta de orquestación



**ANSIBLE**

# Herramientas libres que usan Git

**REACT NATIVE** (<https://github.com/facebook/react-native>): React Native, es un framework de código abierto creado por Meta Platforms, Inc. Se utiliza para desarrollar aplicaciones para Android, Android TV, iOS, macOS, tvOS, Web, Windows y UWP al permitir que los desarrolladores usen React con las características nativas de estas plataformas.



**VUE.js** (<https://github.com/vuejs/vue>): Vue.js es un framework de JavaScript de código abierto para la construcción de interfaces de usuario y aplicaciones de una sola página con licencia del MIT.



# Utilización en la Administración Pública

**República Argentina** (<https://github.com/argob>): Repositorio oficial del Gobierno de la República Argentina.

**Datos Argentina** (<https://github.com/datosgobar>): repositorio oficial del equipo de Datos de la Nación Argentina.

**Gobierno de la Ciudad de Buenos Aires** (<https://github.com/gcba>): Repositorio oficial del GCBA.

**Buenos Aires Data** (<https://github.com/datosgcba>): Repositorio oficial de la iniciativa de datos abiertos de la Ciudad de Buenos Aires.

Si [buscamos](#) en **Google** las palabras **git gob ar** o **gitlab gob ar** aparecerán más casos.

# Github

GitHub (<https://github.com/>) es una forja para alojar proyectos utilizando el sistema de control de versiones Git. Se utiliza principalmente para la creación de código fuente de programas de ordenador. El software que opera GitHub fue escrito en Ruby on Rails. Desde enero de 2010, GitHub opera bajo el nombre de GitHub, Inc.



# Gitlab

Gitlab Inc. (<https://about.gitlab.com/>) es una compañía de [núcleo abierto](#) y es la principal proveedora del software GitLab, un servicio web de [forja](#), control de versiones y DevOps basado en Git.






# Entornos visuales para Git

Si bien existen numerosos entornos visuales para manejar Git para cualquier servidor (github, gitlab, o git instalado en nuestros servidores) lo ideal es aprender a manejar Git desde la consola.

En la [página oficial de Git](#) podemos encontrar una gran variedad de clientes GUI tanto para Linux como para Windows como GitHub desktop, SourceTree, TortoiseGit y **GitKraken** como los más utilizados.

 **git** --local-branching-on-the-cheap

[About](#)  
[Documentation](#)  
[Downloads](#)  
[GUI Clients](#)  
[Logos](#)  
[Community](#)

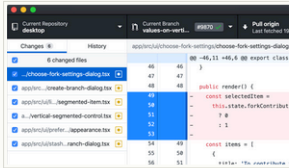
The entire [Pro Git book](#) written by Scott Chacon and Ben Straub is available to [read online for free](#). Dead tree versions are available on [Amazon.com](#).

## GUI Clients

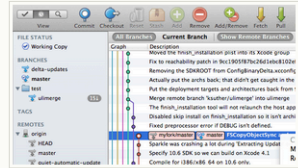
Git comes with built-in GUI tools for committing ([git-gui](#)) and browsing ([gitk](#)), but there are several third-party tools for users looking for platform-specific experience.

If you want to add another GUI tool to this list, just [follow the instructions](#).

[All](#) [Windows](#) [Mac](#) [Linux](#) [Android](#) [iOS](#)



**GitHub Desktop**  
Platforms: Mac, Windows  
Price: Free  
License: MIT




**SourceTree**  
Platforms: Mac, Windows  
Price: Free  
License: Proprietary

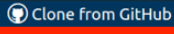
# Complementos Git para Visual Studio Code

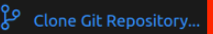
**Visual Studio Code** es un **editor de código** fuente desarrollado por Microsoft para Windows, Linux, macOS y Web. Incluye soporte para la depuración, **control integrado de Git**, resaltado de sintaxis, finalización inteligente de código, fragmentos y refactorización de código.



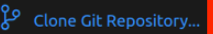
# Complementos Git para Visual Studio Code

1  Podemos empezar un proyecto desde VisualCode clonando desde GitHub siguiendo estos 3 pasos.

3  Provide repository URL or pick a repository source. remote sources

2  Visual Studio Code Editing evolved






Start

- New File...
- Open File...
- Open Folder...
-  Clone Git Repository...

Recent

- swoogo-test-django ~/Documentos/dev-proyectos/CRPG
- curso-docker ~/Documentos
- ejemplos\_redes ~/Documentos/curso-docker
- persistent ~/Documentos/curso-docker/ejemplos\_redes
- swoogo-api ~/Documentos/dev-proyectos/CRPG
- More...

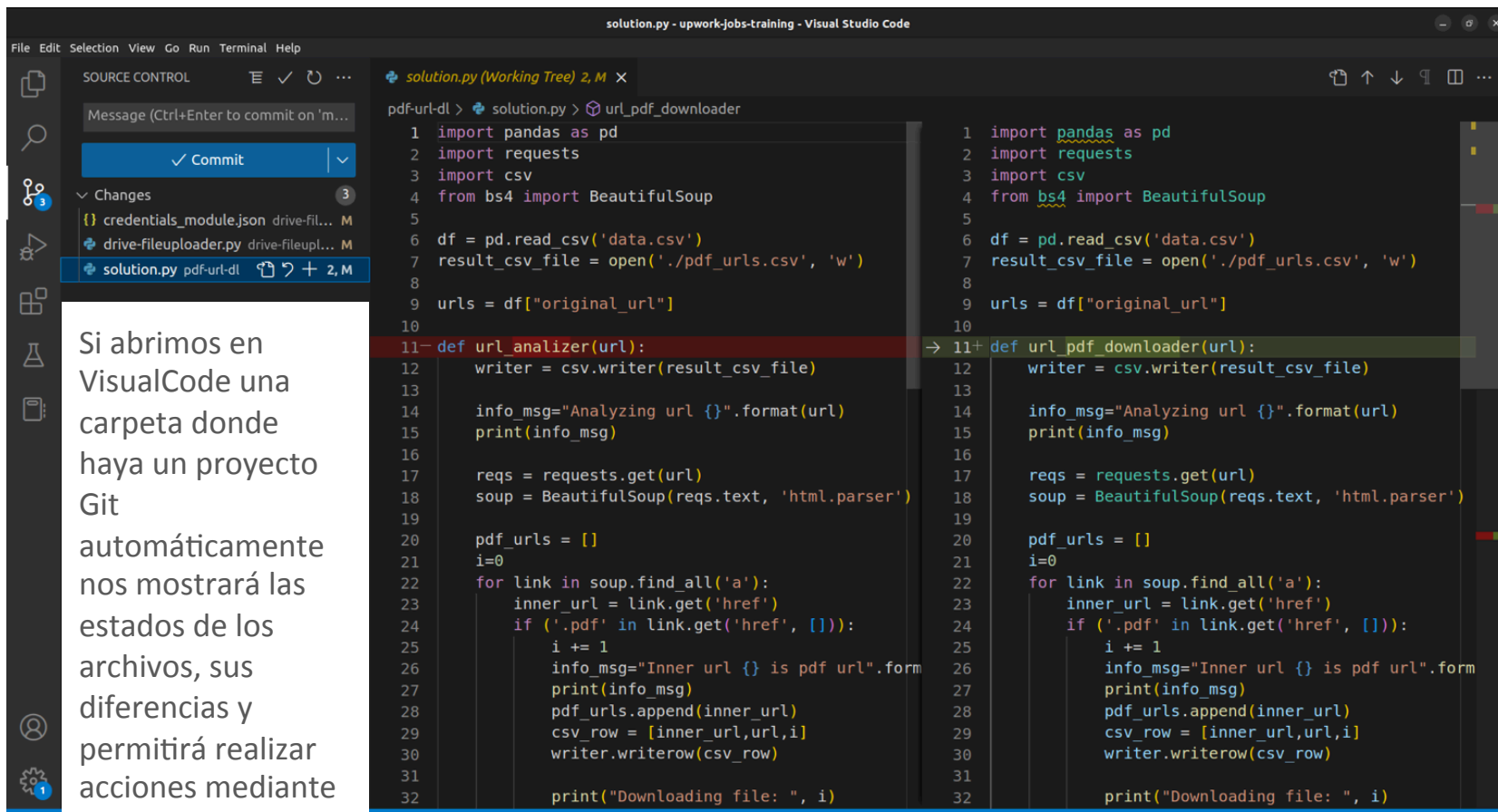
Walkthroughs

-  Get Started with VS Code Discover the best customizations to make VS Code yours.
-  Learn the Fundamentals Jump right into VS Code and get an overview of the must-have features.
-  Get started with JavaScript and Node.js **New**
-  Boost your Productivity
-  Get started with Python development **Updated**

More...

✓ Show welcome page on startup

# Complementos Git para Visual Studio Code

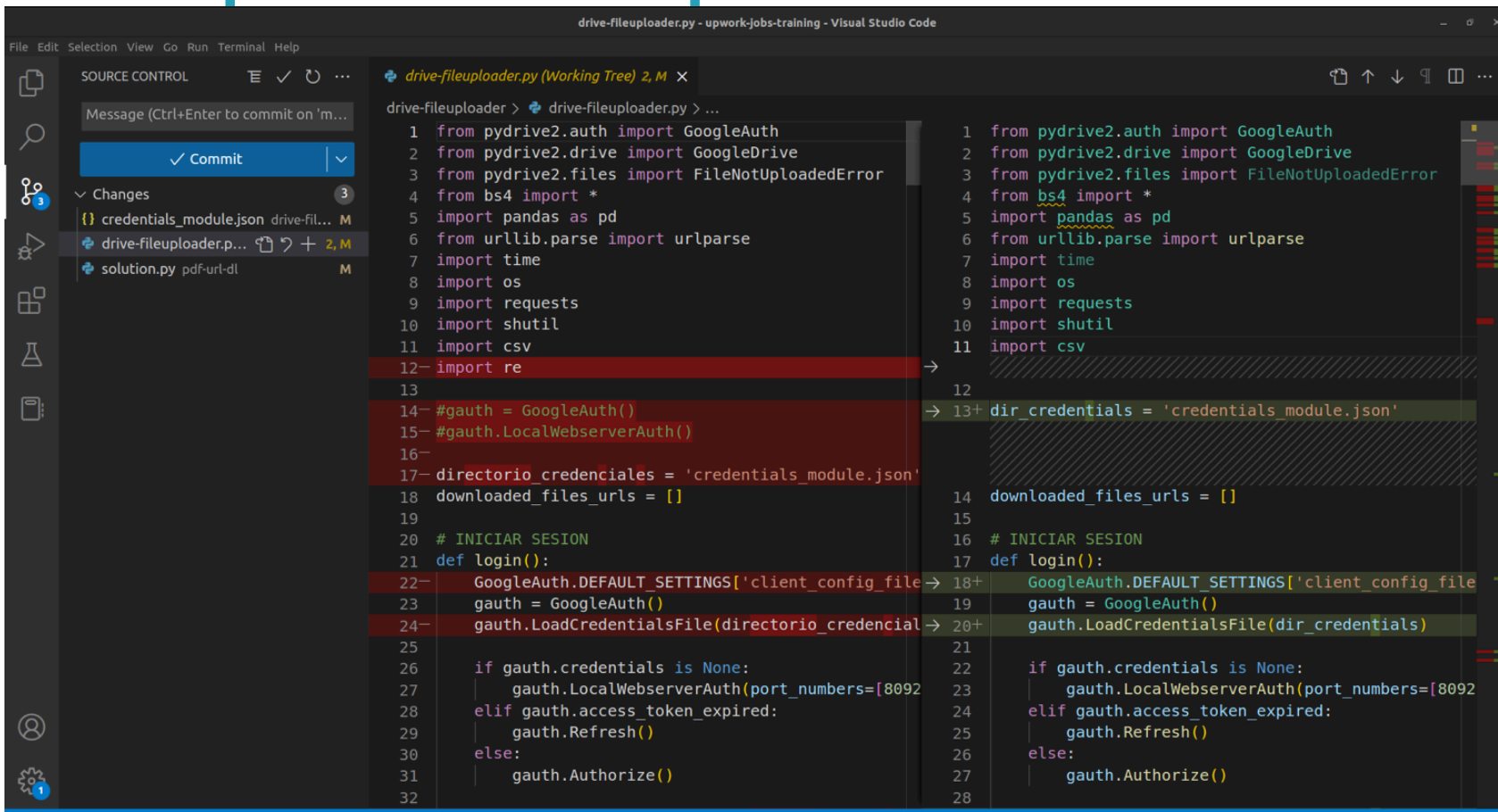


The screenshot shows the Visual Studio Code interface with the Git extension. On the left, the 'SOURCE CONTROL' panel displays a list of files: 'credentials\_module.json', 'drive-fileuploader.py', and 'solution.py'. The 'solution.py' file is selected, showing a diff between the current state and the last commit. The diff highlights changes in the 'url\_analyzer' and 'url\_pdf\_downloader' functions. The 'url\_analyzer' function is shown on the left, and the 'url\_pdf\_downloader' function is shown on the right. The diff shows that the 'url\_analyzer' function has been renamed to 'url\_pdf\_downloader' and its logic has been updated to handle PDF URLs.

Si abrimos en VisualCode una carpeta donde haya un proyecto Git automáticamente nos mostrará los estados de los archivos, sus diferencias y permitirá realizar acciones mediante

```
1 import pandas as pd
2 import requests
3 import csv
4 from bs4 import BeautifulSoup
5
6 df = pd.read_csv('data.csv')
7 result_csv_file = open('./pdf_urls.csv', 'w')
8
9 urls = df["original_url"]
10
11- def url_analyzer(url):
12     writer = csv.writer(result_csv_file)
13
14     info_msg="Analyzing url {}".format(url)
15     print(info_msg)
16
17     reqs = requests.get(url)
18     soup = BeautifulSoup(reqs.text, 'html.parser')
19
20     pdf_urls = []
21     i=0
22     for link in soup.find_all('a'):
23         inner_url = link.get('href')
24         if ('.pdf' in link.get('href', [])):
25             i += 1
26             info_msg="Inner url {} is pdf url".format(inner_url)
27             print(info_msg)
28             pdf_urls.append(inner_url)
29             csv_row = [inner_url,url,i]
30             writer.writerow(csv_row)
31
32     print("Downloading file: ", i)
11+ def url_pdf_downloader(url):
12     writer = csv.writer(result_csv_file)
13
14     info_msg="Analyzing url {}".format(url)
15     print(info_msg)
16
17     reqs = requests.get(url)
18     soup = BeautifulSoup(reqs.text, 'html.parser')
19
20     pdf_urls = []
21     i=0
22     for link in soup.find_all('a'):
23         inner_url = link.get('href')
24         if ('.pdf' in link.get('href', [])):
25             i += 1
26             info_msg="Inner url {} is pdf url".format(inner_url)
27             print(info_msg)
28             pdf_urls.append(inner_url)
29             csv_row = [inner_url,url,i]
30             writer.writerow(csv_row)
31
32     print("Downloading file: ", i)
```

# Complementos Git para Visual Studio Code



# Pregunta/Debate

Para comentar en el foro

¿Qué tan importante es aprender este tipo de herramientas? ¿Por qué?

¿Qué tipos de archivo deberían subirse a un servidor Git y cuáles no?

¿Qué ventajas y desventajas nos brinda el Software Libre?

[ipap.gba.gob.ar](http://ipap.gba.gob.ar)

**IPAP**

SUBSECRETARÍA DE  
GESTIÓN Y EMPLEO PÚBLICO

JEFATURA DE  
GABINETE



GOBIERNO DE LA PROVINCIA DE  
**BUENOS AIRES**