

En rolig start

Introduksjon til programmering - Del 2

KRISTIAN BOTNEN, 2024

Øktens agenda

Noen flere datatyper

Funksjoner og moduler

Versjonshåndtering

Noen flere datatyper

Syntax

```
:::python  
def test():  
    print("Hello world")
```

Variabler

```
:::python
# Define some variables
x = 5
name = "Kristian"

# Use our variables
print(x)
print(name)
```

Typers

```
:::python
Text Type:          str
Numeric Types:      int, float, complex
Sequence Types:     list, tuple, range
Mapping Type:        dict
Set Types:           set, frozenset
Boolean Type:        bool
Binary Types:        bytes, bytearray, memoryview
None Type:           NoneType
```

Operatorer

`:::python`

Arithmetic operators `(+, -, *, /, %, **, //)`

Assignment operators `(=, +=, -=, *=, /=, %=, //=, **=, &=, |=, ^=, >>=, <<=)`

Comparison operators `(==, !=, >, <, >=, <=)`

Logical operators `(and, or, not)`

Identity operators `(is, is not)`

Membership operators `(in, not in)`

Bitwise operators `(&, |, ^, ~, <<, >>)`

Python Operators and Booleans Cheat Sheet by [Nouha Thabet](#)

Python Arithmetic Operators

Addition	9 + 2	>> 11
Subtraction	9 - 2	>> 7
Multiplication	9 * 2	>> 18
Division	9 / 2	>> 4.5
Modulus	9 % 2	>> 1
Exponentiation	3 ** 2	>> 81
Floor division	9 // 2	>> 4

Python Assignment Operators

Operator	Example	Same As
=	x = 2	x = 2
+=	x += 2	x = x + 2
-=	x -= 2	x = x - 2
*=	x *= 2	x = x * 2
/=	x /= 2	x = x / 2
%=	x %= 2	x = x % 2
//=	x //= 2	x = x // 2
**=	x **= 2	x = x ** 2

Python Comparison Operators

Equal	x == y
Not equal	x != y
Greater than	x > y
Less than	x < y
Greater than or equal to	x >= y
Less than or equal to	x <= y

Boolean Values

In programming you often need to know if an expression is `True` OR `False`.

You can evaluate any expression in Python, and get the answer.

```
print(5 < 8)          >>> True
print(5 > 8)          >>> False
```

Python Logical Operators

and Returns True if both statements are true

```
x < 5 and x < 10
```

or Returns True if one of the statements is true

```
x < 5 or x < 4
```

not Reverse the result, returns False if the result is true

```
not(x < 5 and x < 10)
```

Python Identity Operators

is Returns true if both variables are the same object

```
x is y
```

is not Returns true if both variables are not the same object

```
x is not y
```

Python Membership Operators

in Returns True if a sequence with the specified value is present in the object

```
x in y
```

not in Returns True if a sequence with the specified value is not present in the object

```
x not in y
```

Python Bitwise Operators

&	AND	Sets each bit to 1 if both bits are 1
	OR	Sets each bit to 1 if one of two bits is 1
^	XOR	Sets each bit to 1 if only one of two bits is 1
~	NOT	Inverts all the bits
<<	Zero fill left shift	Shift left by pushing zeros in from the right and let the leftmost bits fall off
>>	Signed right shift	Shift right by pushing copies of the leftmost bit in from the left, and let the rightmost bits fall off

Int / Float / Complex

```
#!/usr/bin/env python3
# coding: utf-8

:::python
var_int = 1
var_float = 1.0
var_complex = 1j

print(var_int, var_float, var_complex, sep=', ')
print(type(var_int), type(var_float), type(var_complex), sep=', ')
```

String

```
:::python
print("Hello")
print('Hello')
print("""Lorem ipsum dolor sit amet,
consectetur adipiscing elit,
sed do eiusmod tempor incididunt
ut labore et dolore magna aliqua.""")
```

Boolean

```
#!/usr/bin/env python
var_string = "Hello World!"
if ('ello' in var_string): # The values between ( and ) will evaluate to True or False, in this example True
    print("We found 'ello' in our string")
else:
    print("We did not find 'ello' in our string")
```

<i>Index</i>	<i>Verdi</i>	DUPLICATES
0	«Henry»	CHANGEABLE
1	«Ford»	ORDERED
2	«English»	

List

<i>Nøkkel</i>	<i>Verdi</i>	NO-DUPLICATES
«f_Name»	«Henry»	CHANGEABLE
«l_Name»	«Ford»	ORDERED
«language»	«English»	

Dictionary

Collections

<i>Index</i>	<i>Verdi</i>	DUPLICATES
0	«Henry»	UNCHANGABLE
1	«Ford»	ORDERED
2	«English»	

Tuple

<i>Verdi</i>	NO-DUPLICATES
«Henry»	UNCHANGABLE
«Ford»	UNORDERED
«English»	

Set

Lists

:::python

<code>append()</code>	Adds an element at the end of the list
<code>clear()</code>	Removes all the elements from the list
<code>copy()</code>	Returns a copy of the list
<code>count()</code>	Returns the number of elements with the specified value
<code>extend()</code>	Add the elements of a list (or any iterable), to the end of the current list
<code>index()</code>	Returns the index of the first element with the specified value
<code>insert()</code>	Adds an element at the specified position
<code>pop()</code>	Removes the element at the specified position
<code>remove()</code>	Removes the item with the specified value
<code>reverse()</code>	Reverses the order of the list
<code>sort()</code>	Sorts the list

Tuples

`:::python`

`count()` Returns the number of times a specified value occurs in a tuple

`index()` Searches the tuple for a specified value and returns the position of where it was found

Sets

<code>add()</code>	Adds an element to the set
<code>clear()</code>	Removes all the elements from the set
<code>copy()</code>	Returns a copy of the set
<code>difference()</code>	Returns a set containing the difference between two or more sets
<code>difference_update()</code>	Removes the items in this set that are also included in another, specified set
<code>discard()</code>	Remove the specified item
<code>intersection()</code>	Returns a set, that is the intersection of two other sets
<code>intersection_update()</code>	Removes the items in this set that are not present in other, specified set(s)
<code>isdisjoint()</code>	Returns whether two sets have a intersection or not
<code>issubset()</code>	Returns whether another set contains this set or not
<code>issuperset()</code>	Returns whether this set contains another set or not
<code>pop()</code>	Removes an element from the set
<code>remove()</code>	Removes the specified element
<code>symmetric_difference()</code>	Returns a set with the symmetric differences of two sets
<code>symmetric_difference_update()</code>	inserts the symmetric differences from this set and another
<code>union()</code>	Return a set containing the union of sets
<code>update()</code>	Update the set with the union of this set and others

Dictionaries

:::python

clear()	Removes all the elements from the dictionary
copy()	Returns a copy of the dictionary
fromkeys()	Returns a dictionary with the specified keys and value
get()	Returns the value of the specified key
items()	Returns a list containing a tuple for each key value pair
keys()	Returns a list containing the dictionary's keys
pop()	Removes the element with the specified key
popitem()	Removes the last inserted key-value pair
setdefault()	Returns the value of the specified key. If the key does not exist: insert the key, with the specified value
update()	Updates the dictionary with the specified key-value pairs
values()	Returns a list of all the values in the dictionary

Funksjoner og moduler

Funksjoner

```
:::python
def greet_function():
    print("Hello from a function")

greet_function()
```

```
:::python
def fibonacci(n):
    if n <= 1: # If the number is 0, then the answer is 0. If the number is 1, then the answer is 1.
        return n
    else:
        return fibonacci(n - 1) + fibonacci(n - 2) # Each successive fibonacci number is found by adding up the two numbers before it.

print('Fibonacci sequence:')
for i in range(5):
    print(fibonacci(i))
```

Funksjon uten parameter

Funksjon med parameter

Funksjon med parameter og returverdi

Funksjon uten parameter med returverdi

Funksjoner

```
def funksjonsnavn():  
    print("Hei fra funksjon")
```

```
def funksjonsnavn(parameter):  
    print("Hei: " + parameter)
```

```
def funksjonsnavn(parameter):  
    print("Hei")  
    return True
```

```
def funksjonsnavn():  
    print("Hei")  
    return True
```

```
import random

for i in range(10):
    print(random.randint(1, 25))
```

Moduler

```
import numpy as np

x = np.array([1, 2, 3])
print(x)
```

```
$ conda create --name envtest python
$ conda activate envtest
```

```
$ python -m venv envtest
$ source envtest/bin/activate
```

Environments

```
$ python -m venv envtest
$ venv\Scripts\activate.bat
```

```
$ python -m venv envtest
$ venv\Scripts\ Activate.ps1
```

```
# Kommentar
def echo_name(name: str) -> str:
    return(f"Hello {name}")
```

Dokumentasjon

```
"""Gets and echo out a given string.
Parameters
-----
name : string
    A string that is part of a greeting
Returns
-----
string
    a string that contains a greeting
"""
```

```
# Kommentar
def echo_name(name: str) -> str:
    """Gets and echo out a given string.
    Parameters
    -----
    name : string
        A string that is part of a greeting
    Returns
    -----
    string
        a string that contains a greeting
    """

    return(f"Hello {name}")
```

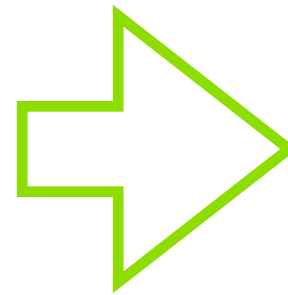
Versjonshåndtering

Intro

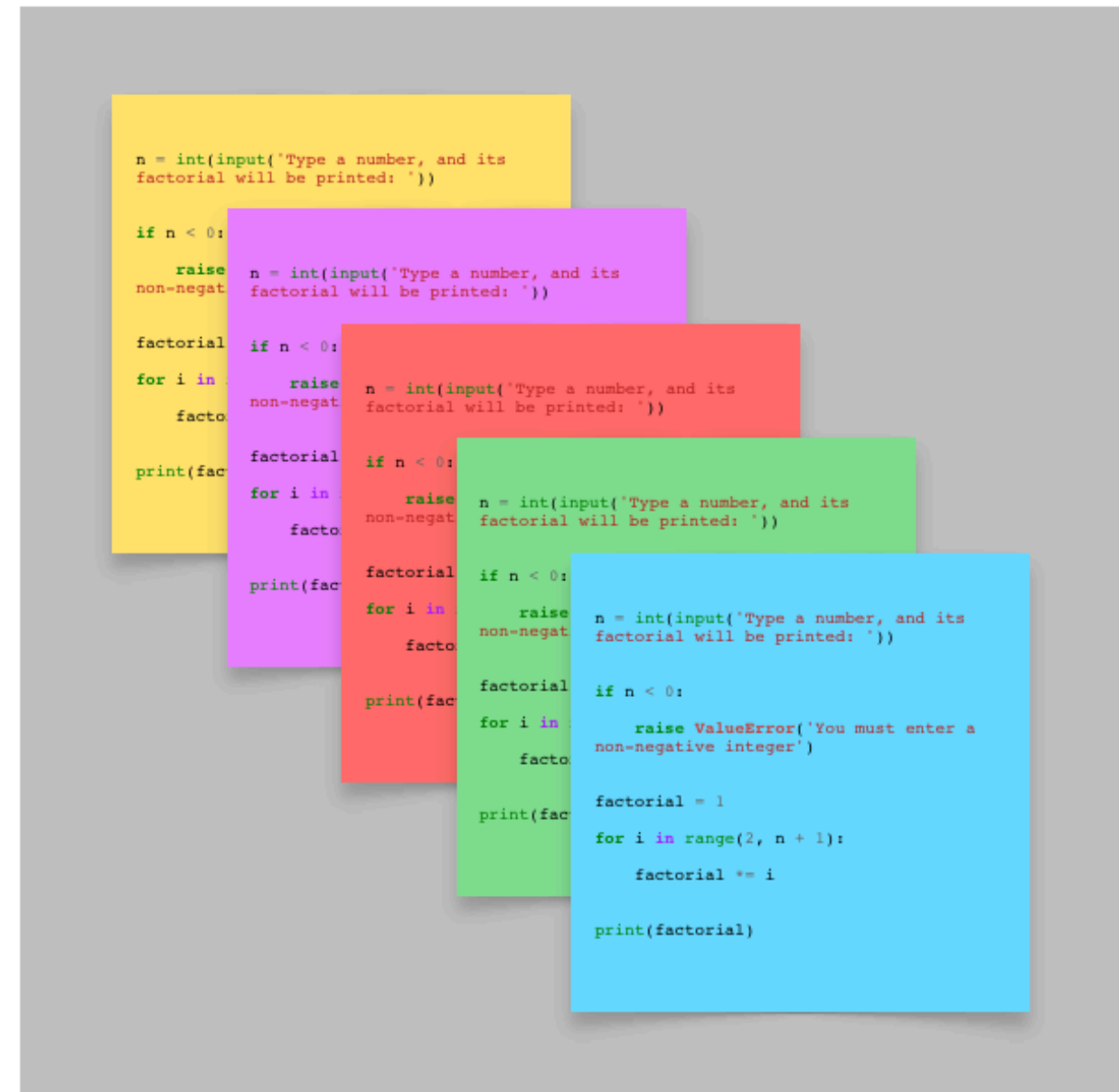
Hvorfor?

- Samarbeid
- Versjonshåndtering
- Gjenoppretting
- Dokumentasjon
- «Sikkerhetskopi»

Filer og mapper



Prosjekt = En samling med filer og mapper



Repository = En samling med filer og mapper, som blir håndtert av versjonskontroll

The diagram illustrates the concept of a repository using a factorial program. It shows a series of overlapping colored rectangles representing different versions of the code. The code evolves from a simple factorial calculation to one that handles non-negative integers and includes error handling.

Timeline:

- Update create_macos_vm_install_dmg.sh Rich Trouton 2 yrs
- Update create_macos_vm_install_dmg.sh ... Rich Trouton 3 yrs
- Update create_macos_vm_install_dmg.sh ... Rich Trouton 4 yrs
- Update create_macos_vm_install_dmg.sh ... Rich Trouton 4 yrs
- Uploading updated create_macos_vm_install_dmg.sh script ... Rich Trouton 6 yrs
- Uploading create_macos_vm_install_dmg script and README ... rtrouton 6 yrs

Code Evolution:

```
n = int(input('Type a number, and its factorial will be printed: '))

if n < 0:
    raise ValueError('You must enter a non-negative integer')

factorial = 1
for i in range(2, n + 1):
    factorial *= i

print(factorial)
```

Commit Log:

```
commit 10a9a0a969c790542de09e5d8dc666a2025 (HEAD -> main, origin/main, origin/HEAD)
Author: rtrouton <rtrouton@yahoo.com>
Date: Tue May 16 11:24:01 2023 -0400

    Update README.md

commit 65055ec9c83e070c8b5d8411b6c5688a7770e61
Author: rtrouton <rtrouton@yahoo.com>
Date: Thu Nov 24 09:21:51 2022 -0500

    Update README.md

    Updated OS compatibility information.

commit 8a4c34cdf4bf12693a461f91d472beba2e59e4db
Author: rtrouton <rtrouton@yahoo.com>
Date: Fri Dec 17 09:28:52 2021 -0500

    Update README.md

commit ef94dee9b07d96173e4a1884be6465f5de3880
Author: rtrouton <rtrouton@yahoo.com>
Date: Fri Dec 17 09:25:56 2021 -0500

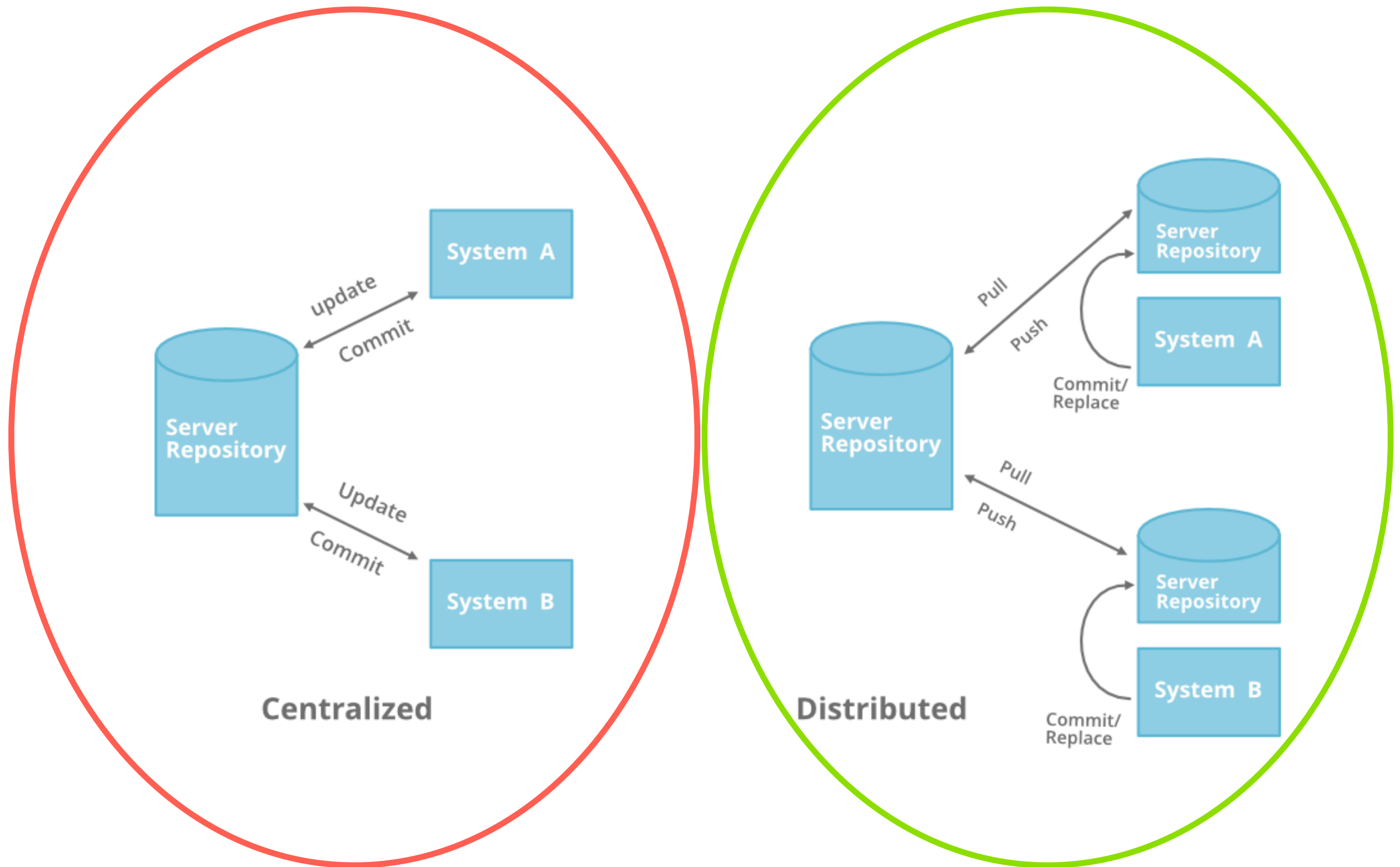
    Update README.md

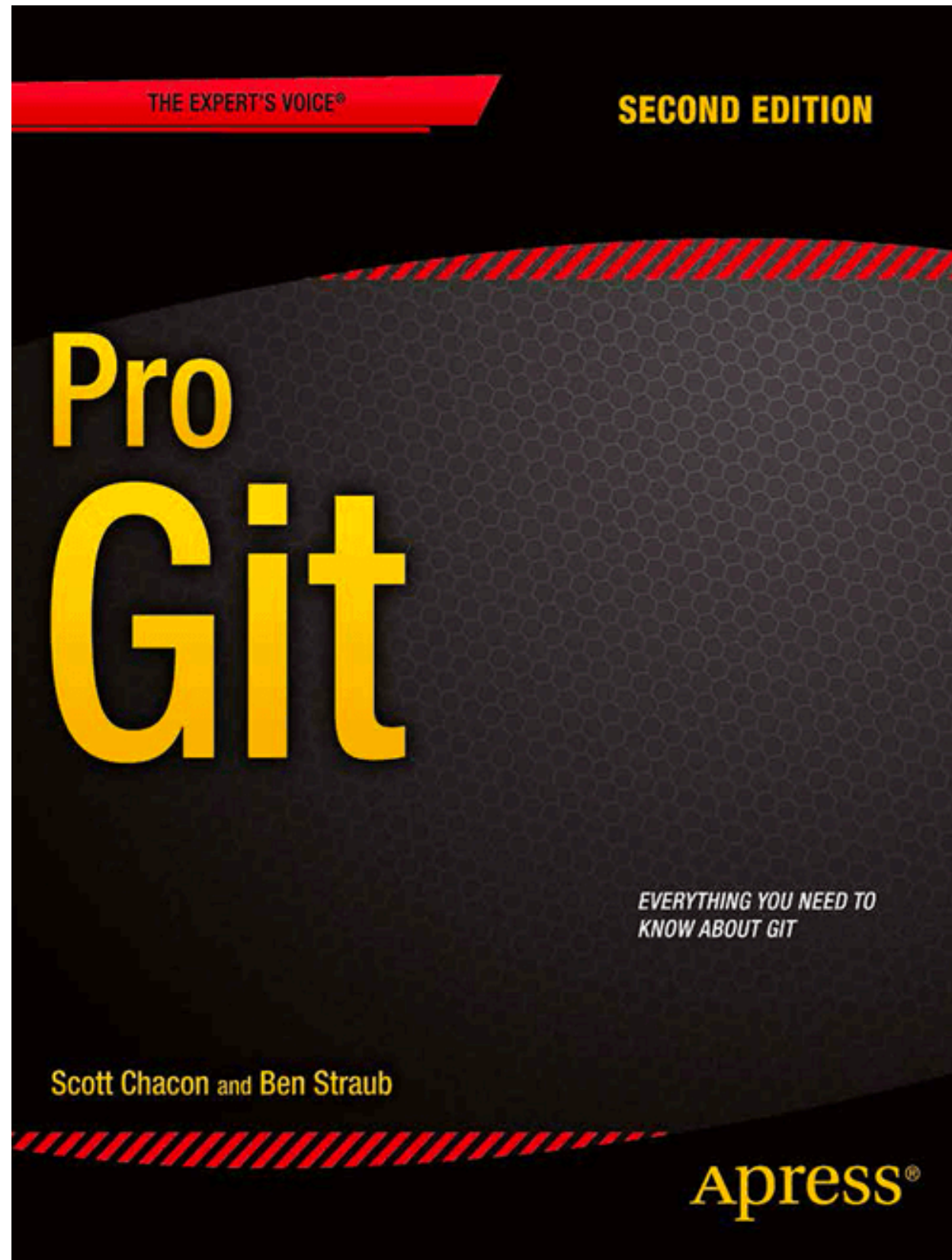
commit 0cca915e33b01a406b0276743ea4e9691d8e99e
Author: rtrouton <rtrouton@yahoo.com>
Date: Tue Sep 28 10:50:30 2021 -0400

    Update README.md

commit e52d3d150522a1d293d47ecd4bef8aa345df8d25
Author: rtrouton <rtrouton@yahoo.com>
Date: Wed Jan 13 17:08:37 2021 -0500

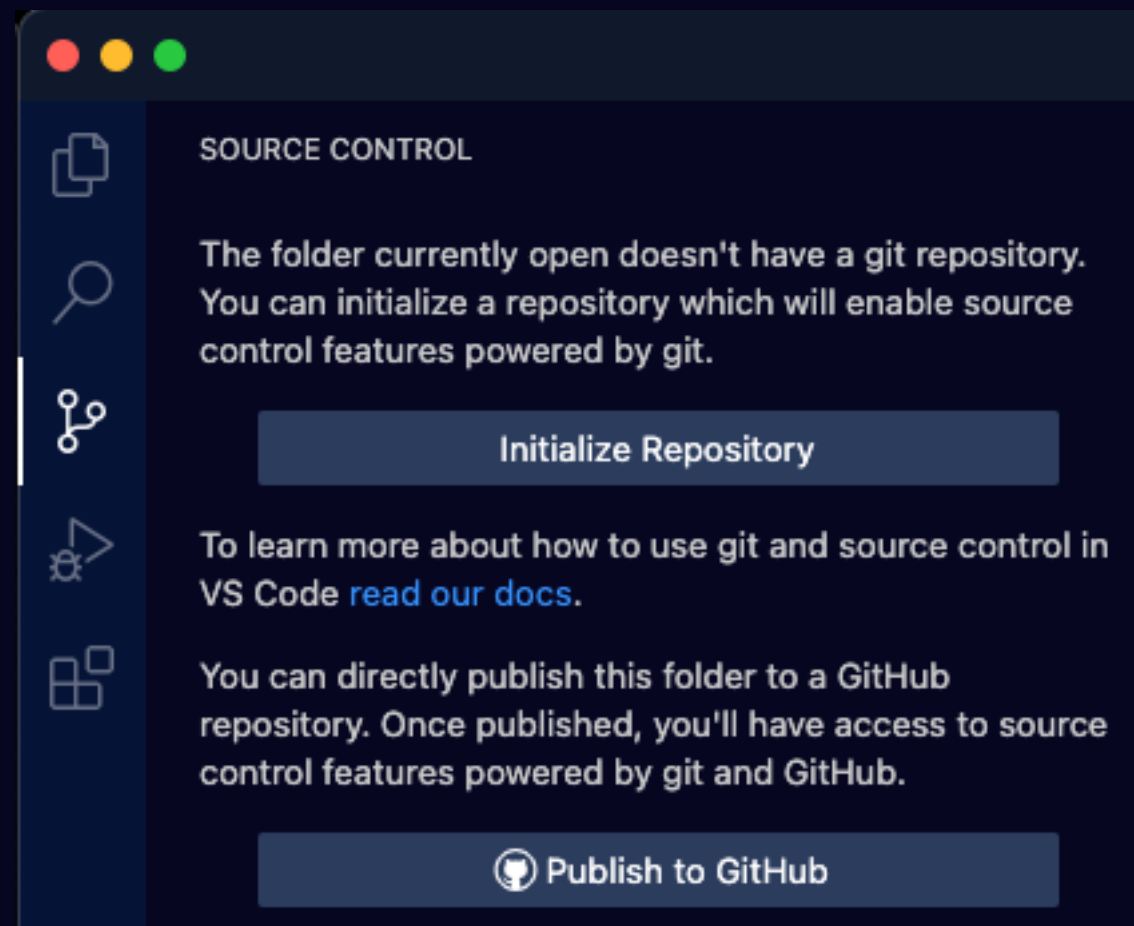
    Update README.md
```





Lokalt

Git init

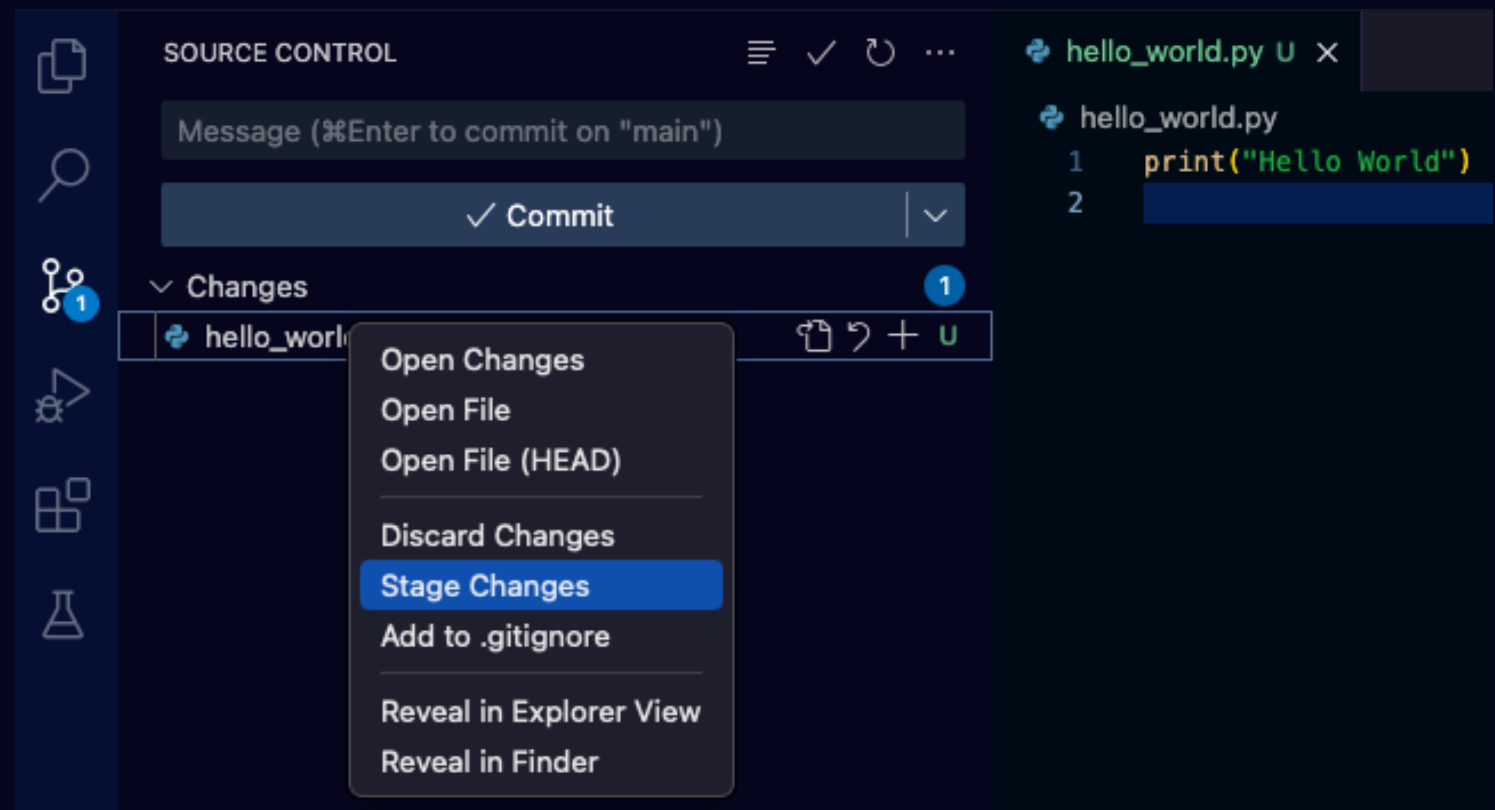
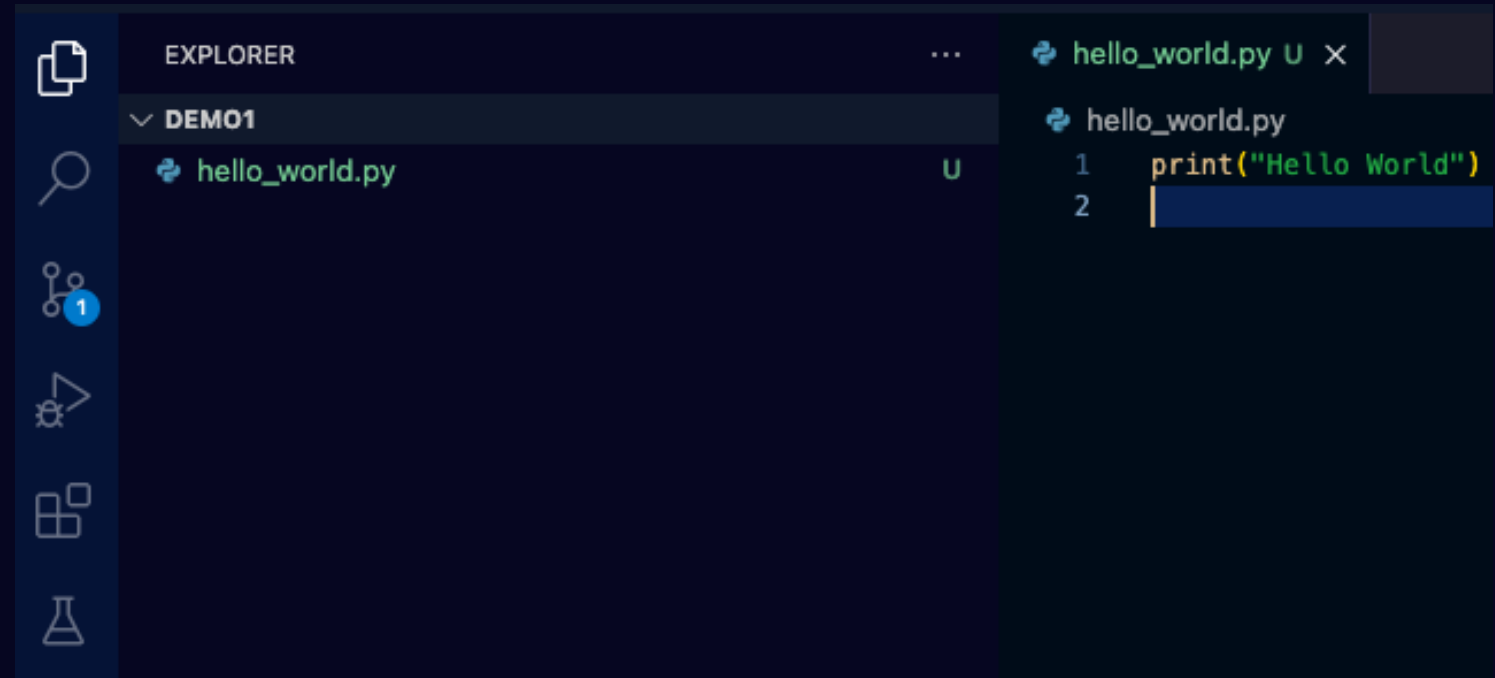


Git init

\$ git init

Oppretter et nytt repository på din maskin

Git add

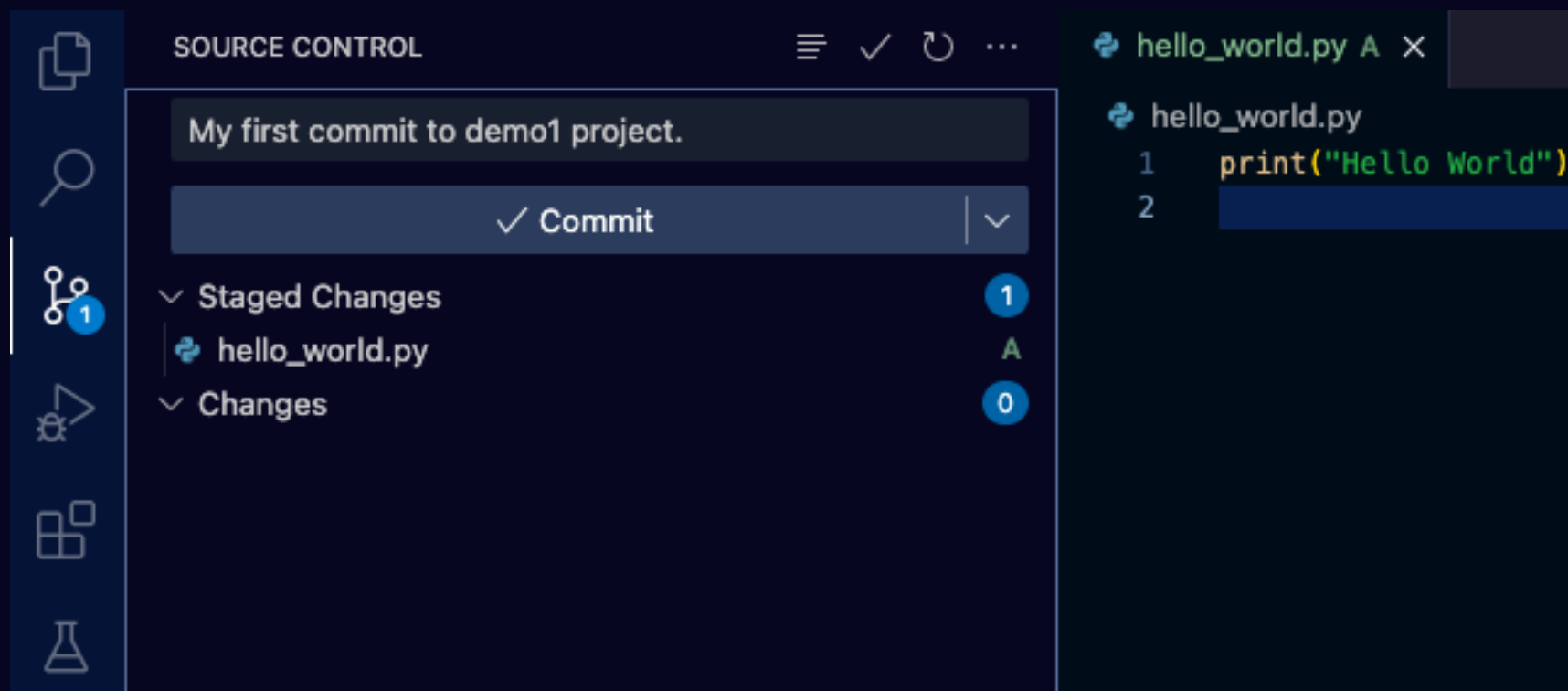


Git add

```
$ git add filnavn1 filnavn2
```

Legg til en eller flere filer til repositoriet

Git commit

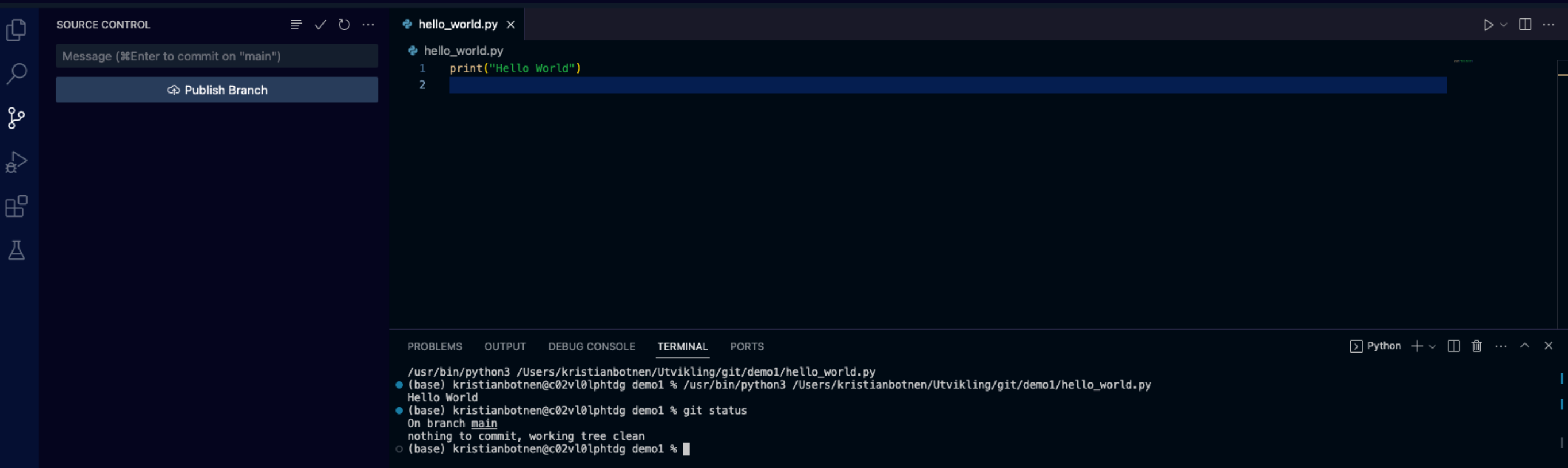


Git commit

```
$ git commit -m «A descriptive message»
```

Registrer nye endringer i repositoret

Git status



The screenshot shows a code editor interface with a dark theme. On the left, a sidebar contains icons for Explorer, Search, Source Control, Run and Debug, Extensions, and Testing. The Source Control panel is active, showing a commit message input field with the placeholder "Message (%Enter to commit on 'main')", a "Publish Branch" button, and a "Commit" button. The main editor area displays a file named `hello_world.py` with the following content:

```
1 print("Hello World")
2
```

Below the editor, a terminal window is open, showing the output of running the Python script and checking the Git status:

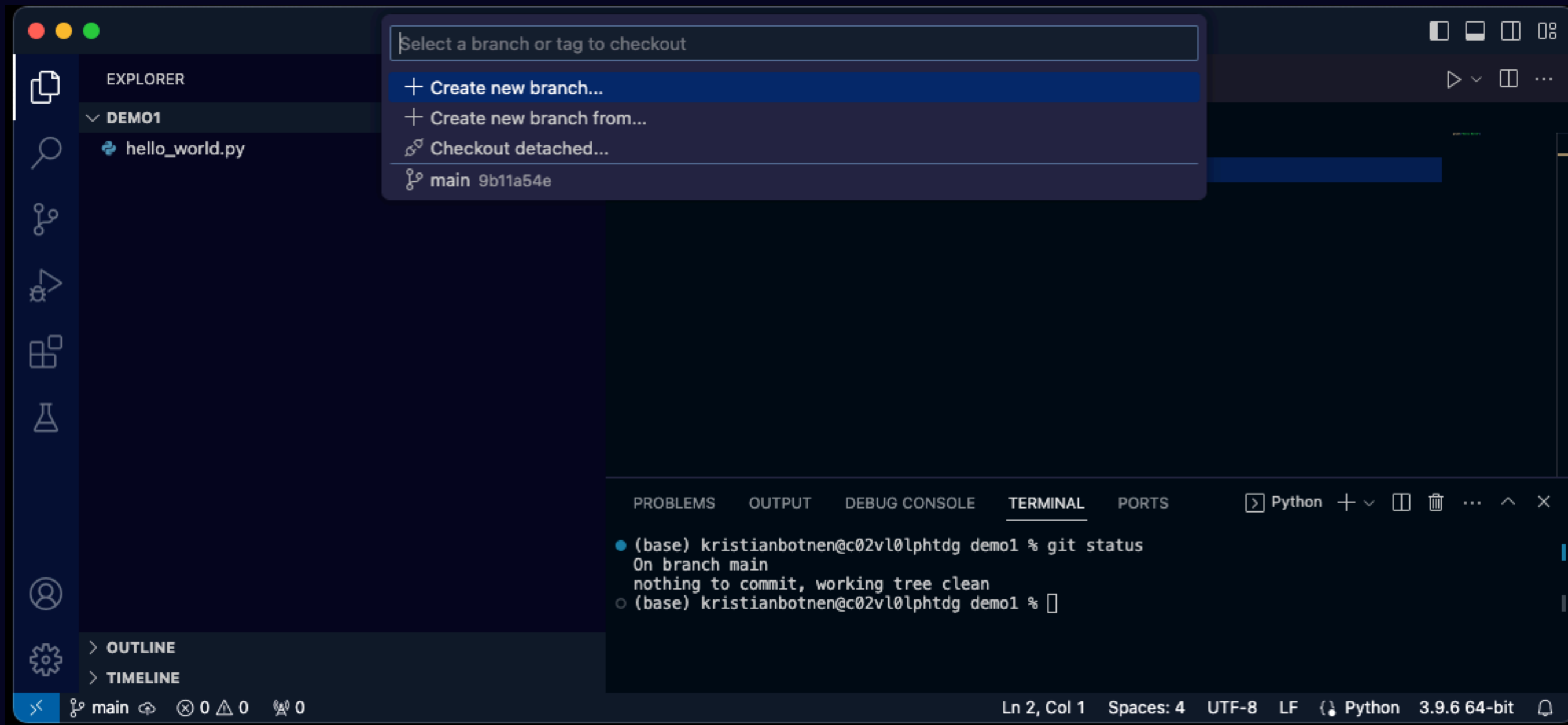
```
/usr/bin/python3 /Users/kristianbotnen/Utvikling/git/demo1/hello_world.py
• (base) kristianbotnen@c02vl0lphtdg demo1 % /usr/bin/python3 /Users/kristianbotnen/Utvikling/git/demo1/hello_world.py
Hello World
• (base) kristianbotnen@c02vl0lphtdg demo1 % git status
On branch main
nothing to commit, working tree clean
○ (base) kristianbotnen@c02vl0lphtdg demo1 %
```

Git status

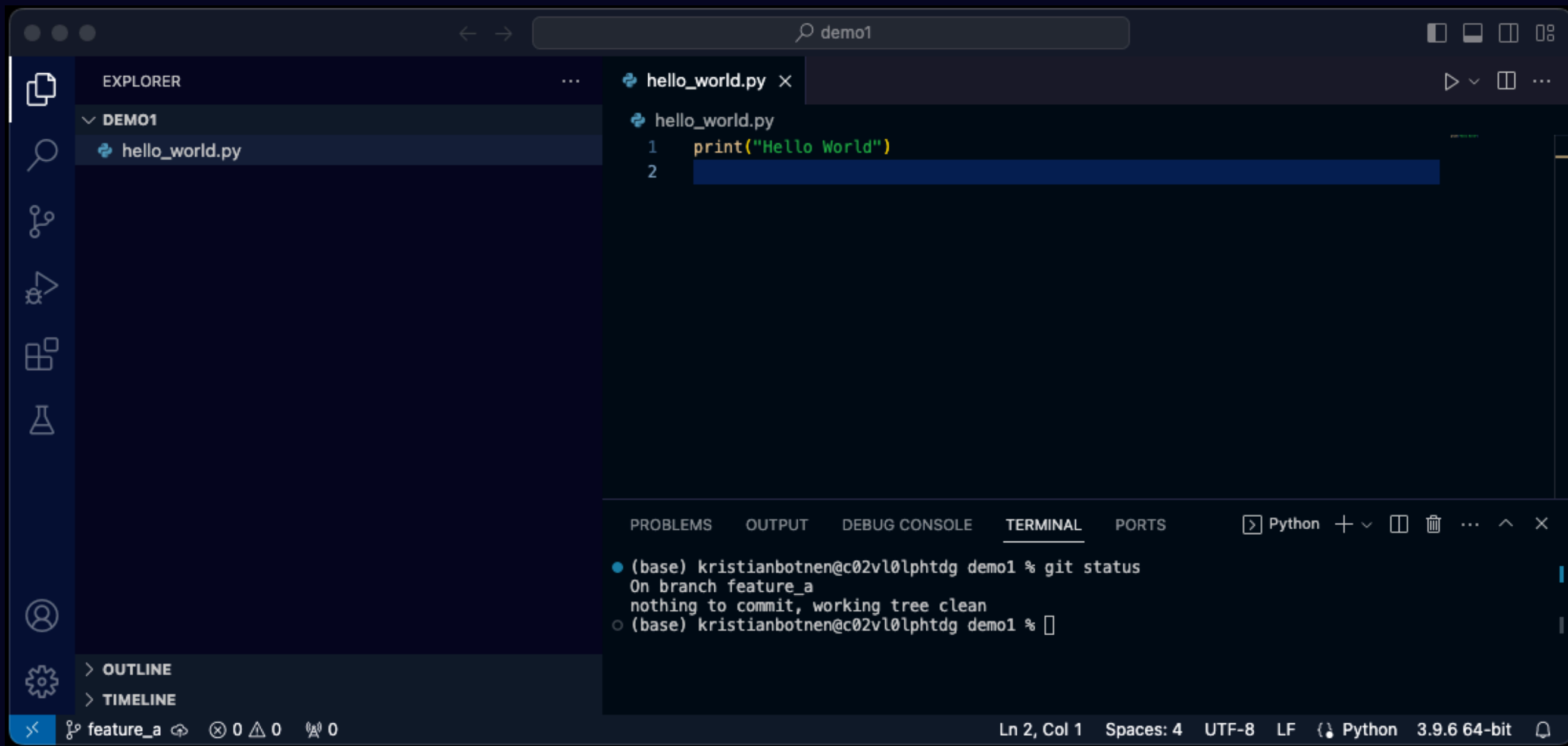
\$ git status

Sjekk status på ditt lokale repositorie

Git branch



Git branch 2



Git branch

```
$ git checkout -b rts1245
```

Oppretter en ny gren i repositoriet ditt

Git branch

```
$ git branch -d rts1245
```

Sletter en gren i repositoriet ditt

Git log

The screenshot shows a code editor interface with a dark theme. The top bar includes a search bar with the text "demo1". The left sidebar contains the Explorer, Outline, and Timeline views. The Explorer view shows a file named "hello_world.py" under a folder named "DEMO1". The Outline view shows the file "hello_world.py". The Timeline view shows a commit message "My first commit to demo1 project. Kristian Botnen 3 hrs" and a status "File Saved". The main editor area shows the file "hello_world.py" with the following code:

```
1 print("Hello World")
2
```

The bottom panel shows the Terminal view with the following output:

```
(base) kristianbotnen@c02vl0lphtdg demo1 % git log
commit 9b11a54e2c0df9b83481a35e9de804da55ac4841 (HEAD -> feature_a, main)
Author: Kristian Botnen <kristian.botnen@uib.no>
Date: Tue Oct 31 11:34:58 2023 +0100

    My first commit to demo1 project.
(base) kristianbotnen@c02vl0lphtdg demo1 %
```

The status bar at the bottom shows the current branch "feature_a", the number of changes (0), the file encoding (UTF-8), the line and column numbers (Ln 2, Col 1), the number of spaces (4), the line feed (LF), the Python version (Python 3.9.6 64-bit), and the file icon.

Git log

\$ git log

Viser endringsloggen for en gren

Git checkout

The screenshot shows the Visual Studio Code interface with a Git checkout menu open. The menu is titled "Select a branch or tag to checkout" and lists the following options:

- + Create new branch...
- + Create new branch from...
- Checkout detached...
- feature_a 9b11a54e
- main 9b11a54e

The Explorer view on the left shows a project named "DEMO1" with a file named "hello_world.py". The Outline view shows a timeline for "hello_world.py" with a commit titled "My first commit to demo1 project. Kristian Botnen 3 hrs" and a "File Saved" event.

The Terminal view at the bottom shows the following commands and output:

```
(base) kristianbotnen@c02vl0lphtdg demo1 % git branch -va
* feature_a 9b11a54 My first commit to demo1 project.
  main      9b11a54 My first commit to demo1 project.
(base) kristianbotnen@c02vl0lphtdg demo1 % git checkout main
```

The status bar at the bottom indicates the current file is "feature_a" and the Python interpreter is "Python 3.9.6 64-bit".

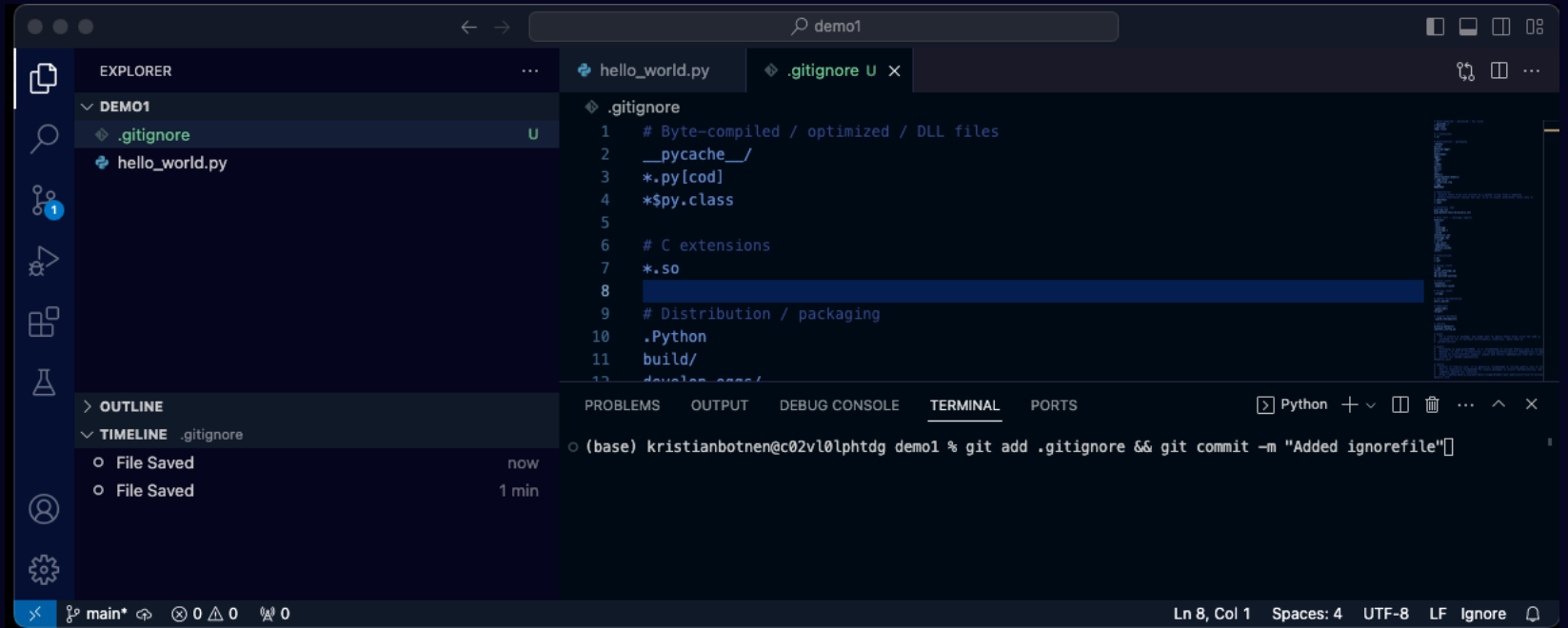
Git checkout

\$ git checkout master

\$ git checkout rts12345

Bytter mellom grener i repositoriet

Git ignore



This screenshot shows the initial state of a .gitignore file in VS Code. The Explorer panel on the left shows a project named 'DEMO1' containing a '.gitignore' file and a 'hello_world.py' file. The .gitignore file is selected and its content is displayed in the editor. The content includes comments and patterns for ignoring byte-compiled files, C extensions, and distribution/packaging files. The Timeline panel shows two 'File Saved' events. The Terminal panel shows the command `git add .gitignore && git commit -m "Added ignorefile"` being executed.

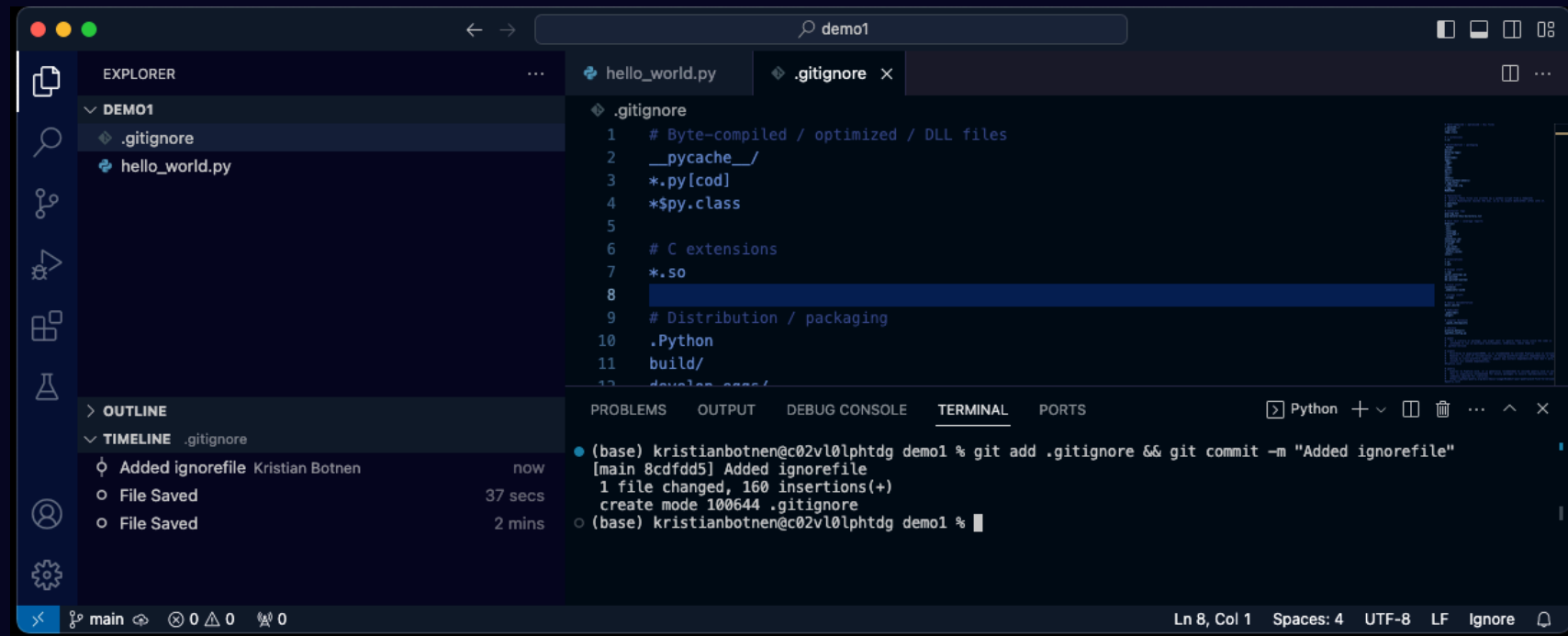
```
.gitignore
1 # Byte-compiled / optimized / DLL files
2 __pycache__/
3 *.py[cod]
4 *$py.class
5
6 # C extensions
7 *.so
8
9 # Distribution / packaging
10 .Python
11 build/
12 developer.*
```

Timeline:

- File Saved (now)
- File Saved (1 min)

Terminal:

```
(base) kristianbotnen@c02vl0lphtdg demo1 % git add .gitignore && git commit -m "Added ignorefile"
```



This screenshot shows the final state of the .gitignore file in VS Code after a commit. The Explorer panel on the left shows the project 'DEMO1' with the '.gitignore' file and 'hello_world.py' file. The .gitignore file is selected and its content is displayed in the editor. The Timeline panel shows three events: 'Added ignorefile' by Kristian Botnen, and two 'File Saved' events. The Terminal panel shows the command `git add .gitignore && git commit -m "Added ignorefile"` being executed, followed by the output of the command.

```
.gitignore
1 # Byte-compiled / optimized / DLL files
2 __pycache__/
3 *.py[cod]
4 *$py.class
5
6 # C extensions
7 *.so
8
9 # Distribution / packaging
10 .Python
11 build/
12 developer.*
```

Timeline:

- Added ignorefile Kristian Botnen (now)
- File Saved (37 secs)
- File Saved (2 mins)

Terminal:

```
(base) kristianbotnen@c02vl0lphtdg demo1 % git add .gitignore && git commit -m "Added ignorefile"
[main 8cdfdd5] Added ignorefile
1 file changed, 160 insertions(+)
create mode 100644 .gitignore
(base) kristianbotnen@c02vl0lphtdg demo1 %
```

Git ignore

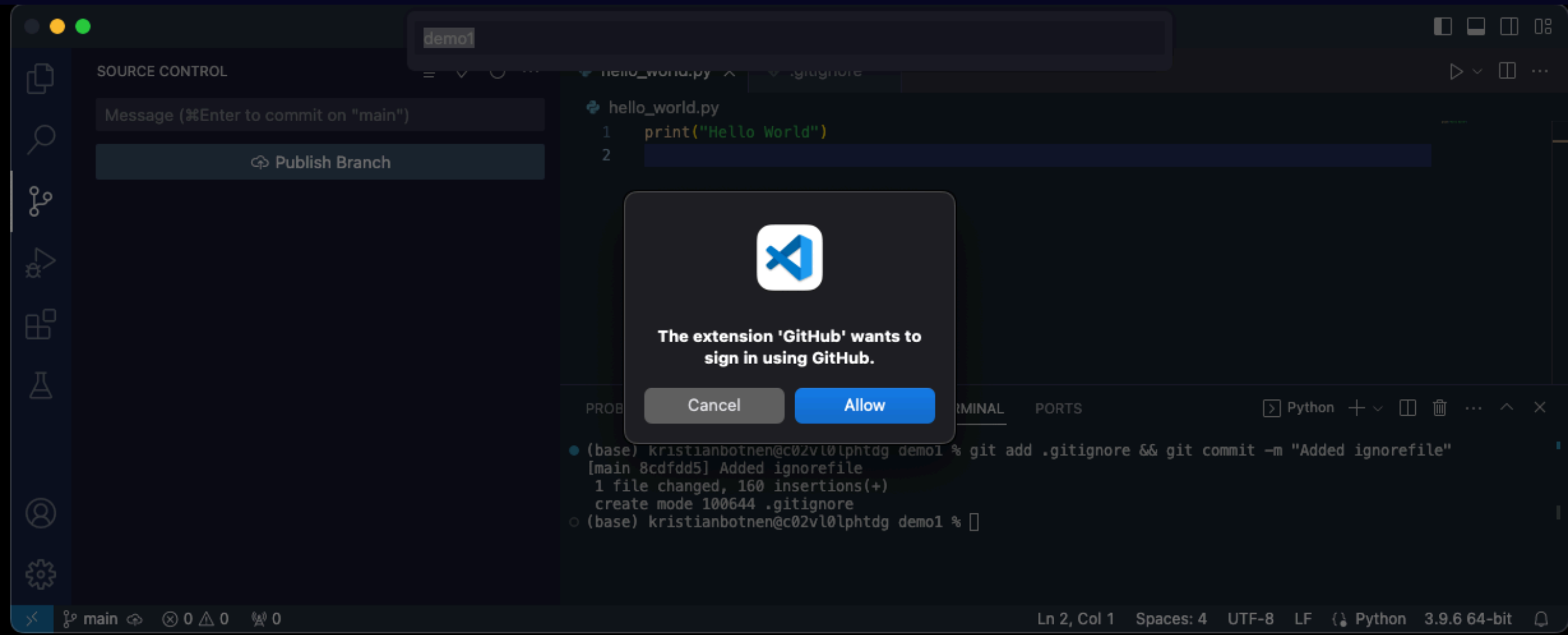
\$ touch .gitignore

Oppgaver - Del 1 - Lokalt

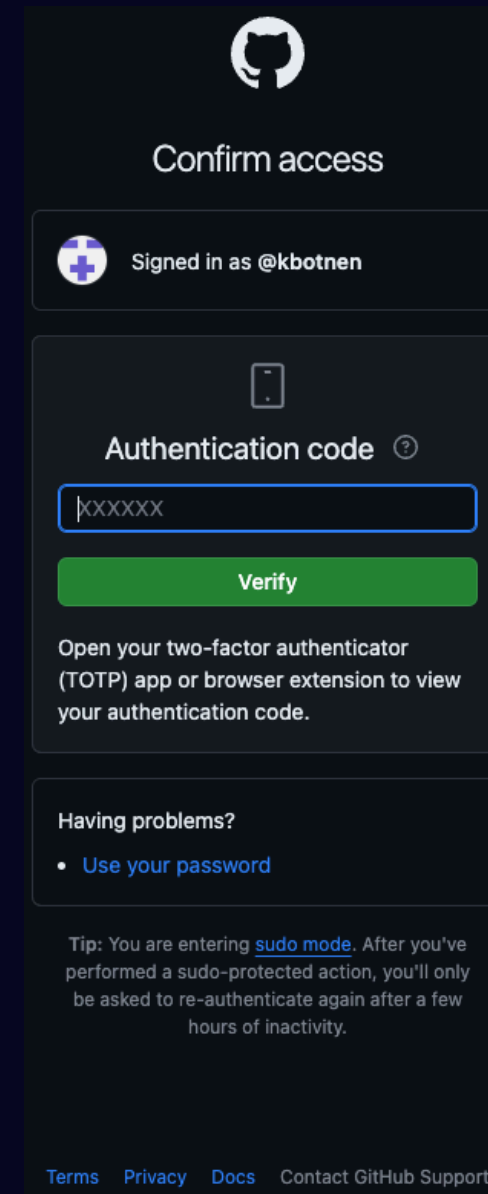
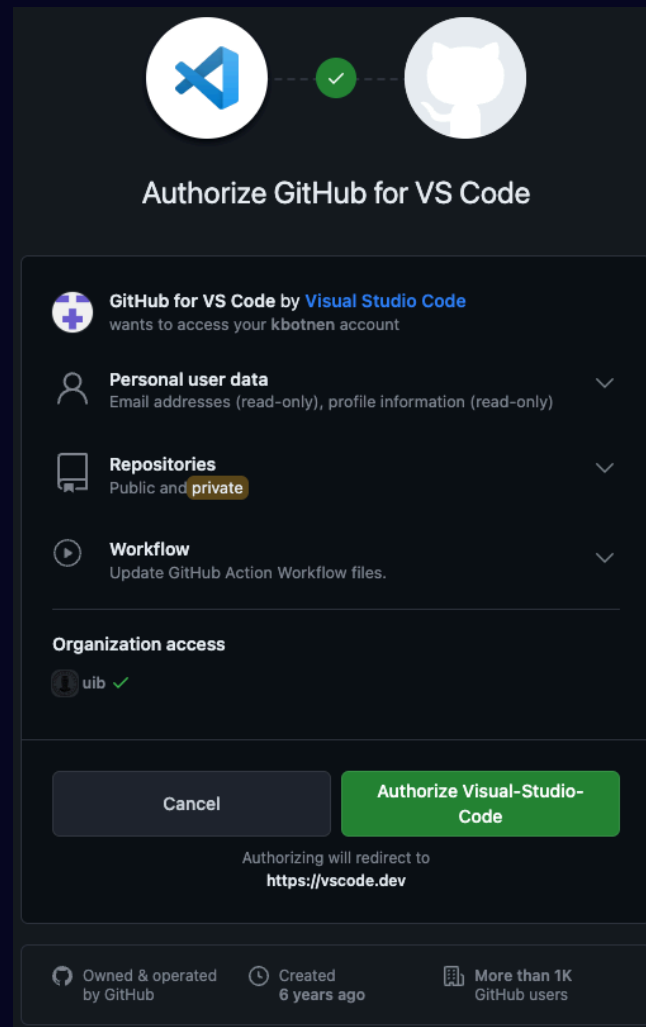
- 1: Lag et nytt repo lokalt på din maskin
- 2: Opprett noen filer og registrer endringene
- 3: Opprett en ny gren, bytt til denne og gjenta punkt 2
- 4: Lag en .gitignore fil og tilpass denne til ditt repo / prosjekt
- 5: Registrer endringene du gjorde i punkt 4

Server

Github - Go public!



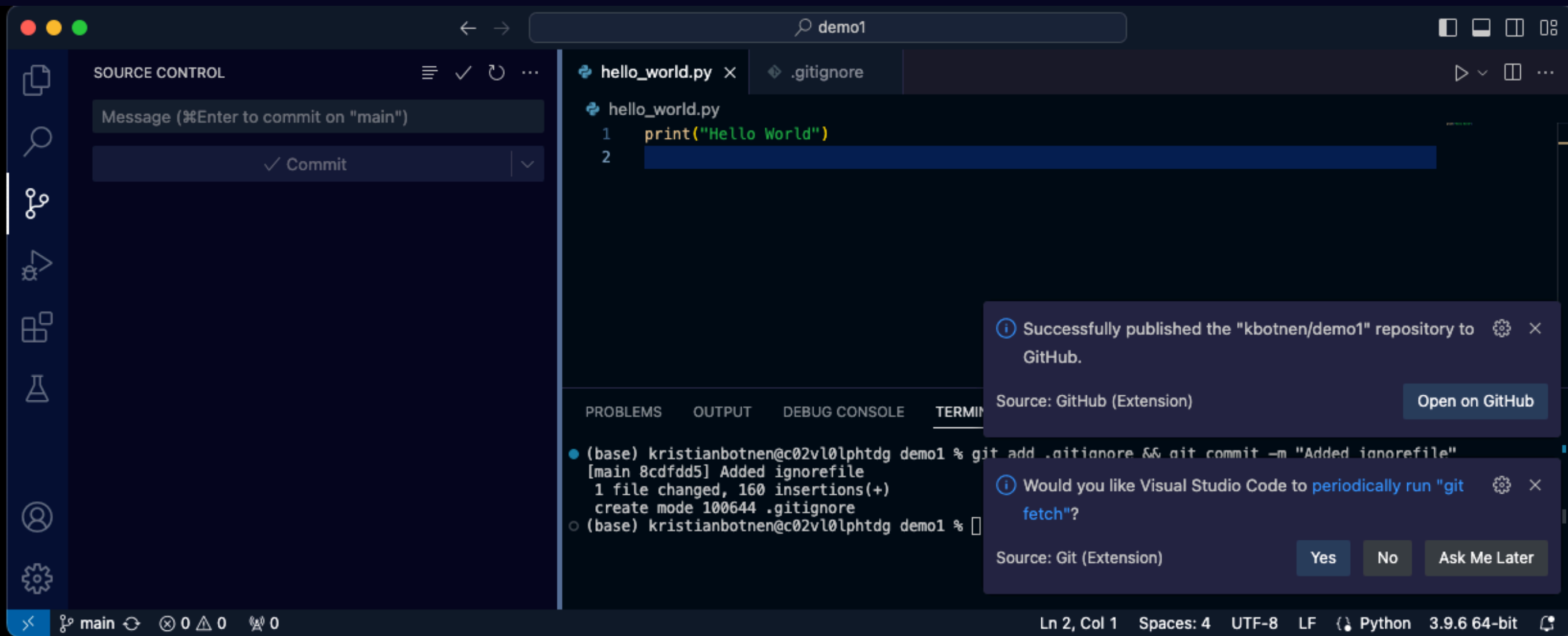
Github - Go public!





Github - Go public!














Github - Go public!











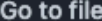
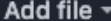

Github - Go public!

  kbotnen / demo1


Type  to search 

 Code  Issues  Pull requests  Actions  Projects  Wiki  Security  Insights  Settings


 demo1 Public  Pin  Unwatch 1  Fork 0  Star 0

 main  1 branch  0 tags  Go to file  Add file  Code


Kristian Botnen Added ignorefile8cdfdd5 11 minutes ago🕒 2 commits


 .gitignore

Added ignorefile11 minutes ago


 hello_world.py


My first commit to demo1 project.4 hours ago


Help people interested in this repository understand your project by adding a README.


About

No description, website, or topics provided.

 Activity

 0 stars

 1 watching

 0 forks

Releases

No releases published

[Create a new release](#)

Packages

No packages published

[Publish your first package](#)

Languages

Python

100.0%

Oppgaver - Del 2 - Server

- 1: Publisert det du gjorde i oppgave 1 til et offentlig GitHub repo
- 2: Gjør endringer i koden din v.h.a nettsiden på GitHub
- 3: Synkroniser endringene du gjorde i punkt 2 til ditt lokale repo
- 4: Gjør endringer i koden din v.h.a Visual Studio Code
- 5: Synkroniser / publiser endringene du gjorde i punkt 4 til ditt lokale repo

Kommandoer

GIT Push

\$ git push

Sender endringene dine til repositorie på server

GIT Push origin

```
$ git push origin rts12345
```

Sender endringene dine til en spesifikk gren på server

GIT Fetch

\$ git fetch

Henter oppdatert info fra server

GIT Pull

\$ git pull

Henter oppdatert info fra server, og fletter den samtidig

GIT Branch

\$ git branch -va

Lister ut alle brancher lokalt og server

GIT Merge

```
$ git merge rts12345
```

Fletter endringene fra rts12345 grenen inn i grenen du står i