DEVELOP A PYTHON SCRIT

BY

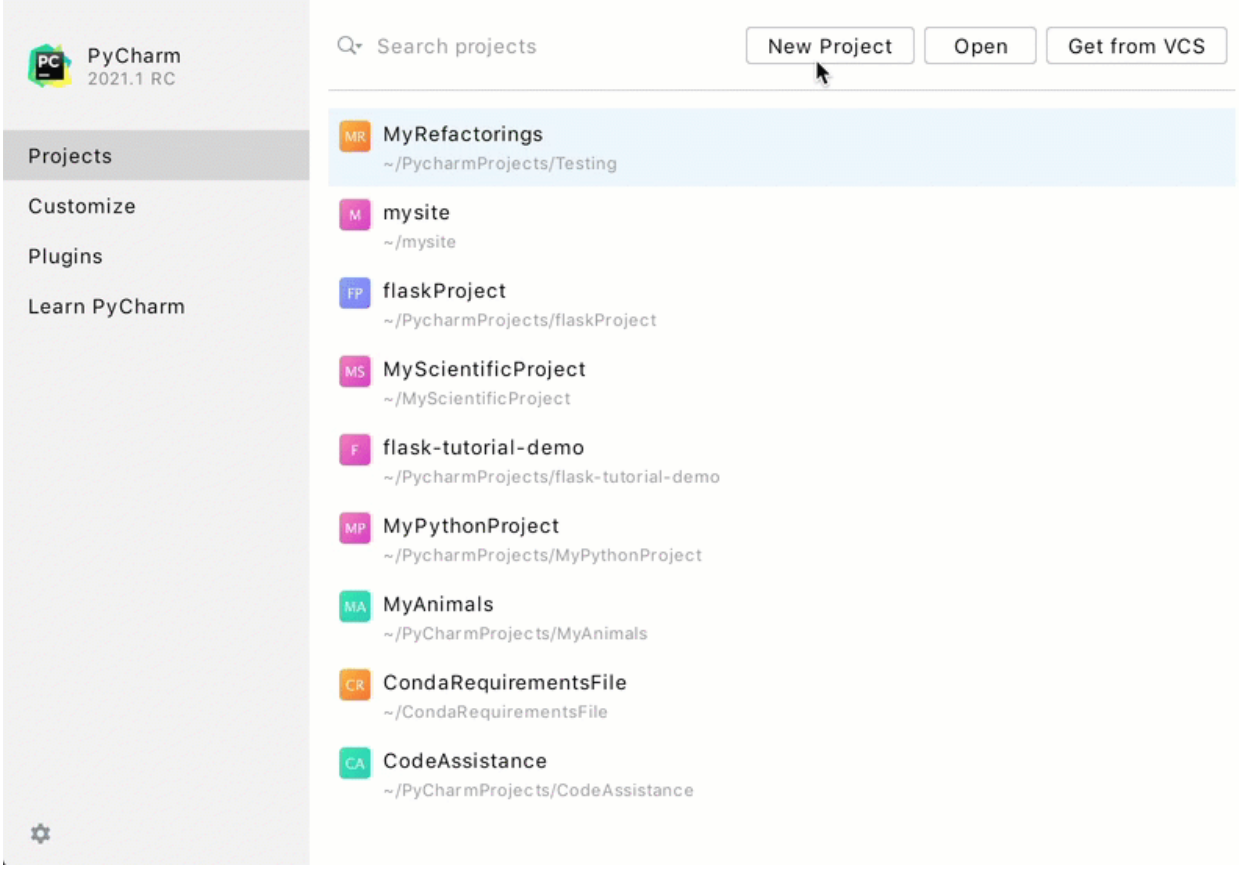
K.GAYATHRI

952319104012

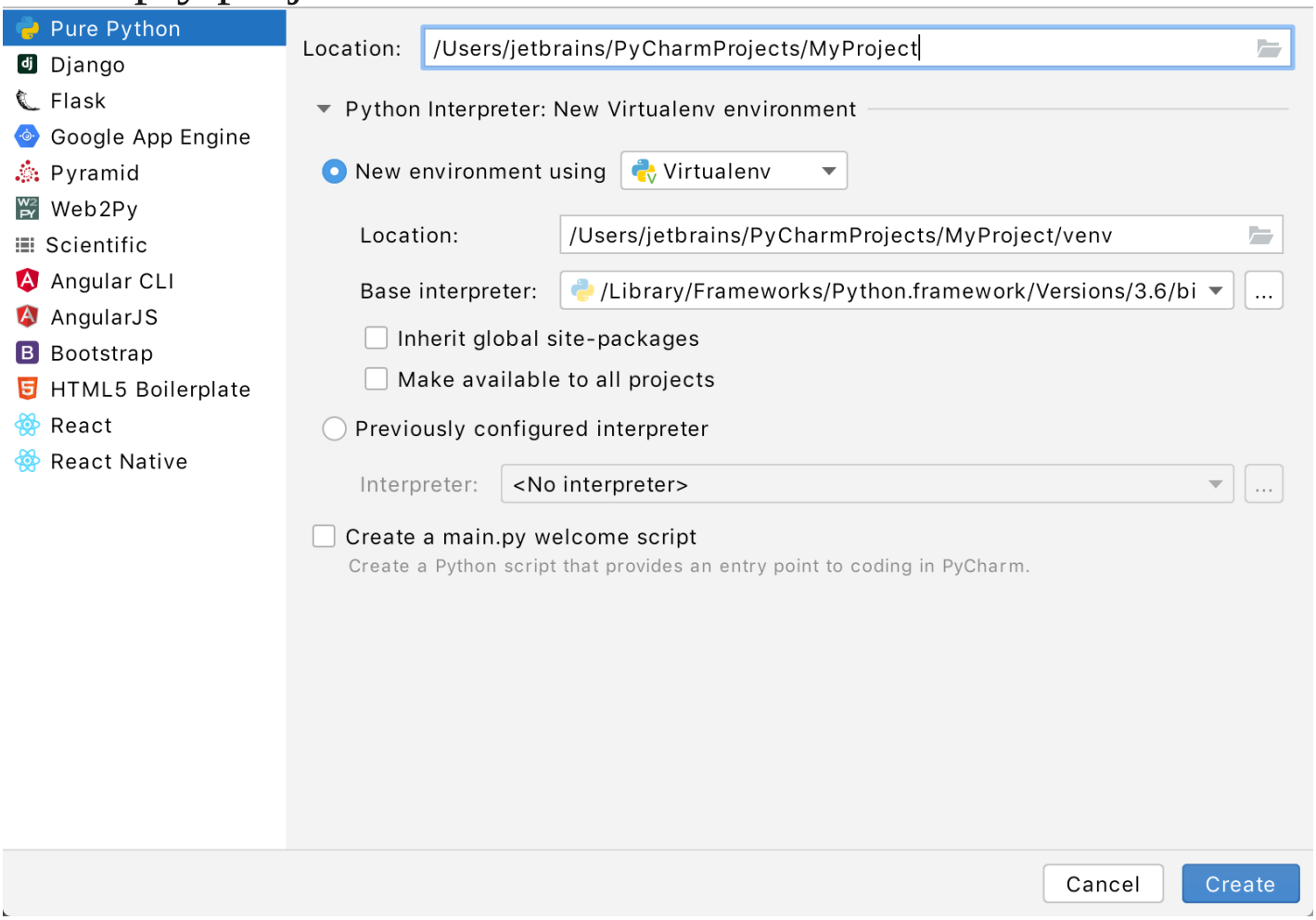
DEVELOP A PYTHON SCRIPT

In the Project tool window, select the project root (typically, it is the root node in the project tree), right-click it, and select File | New .... Select the option Python File from the context menu, and then type the new filename. PyCharm creates a new Python file and opens it for editing.

Step 1. Create and run your first Python project



1. If you're on the Welcome screen, click New Project. If you've already got any project open, choose File | New Project from the main menu.
2. Although you can create projects of various types in PyCharm, in this tutorial let's create a simple Pure Python project. This template will create an empty project.

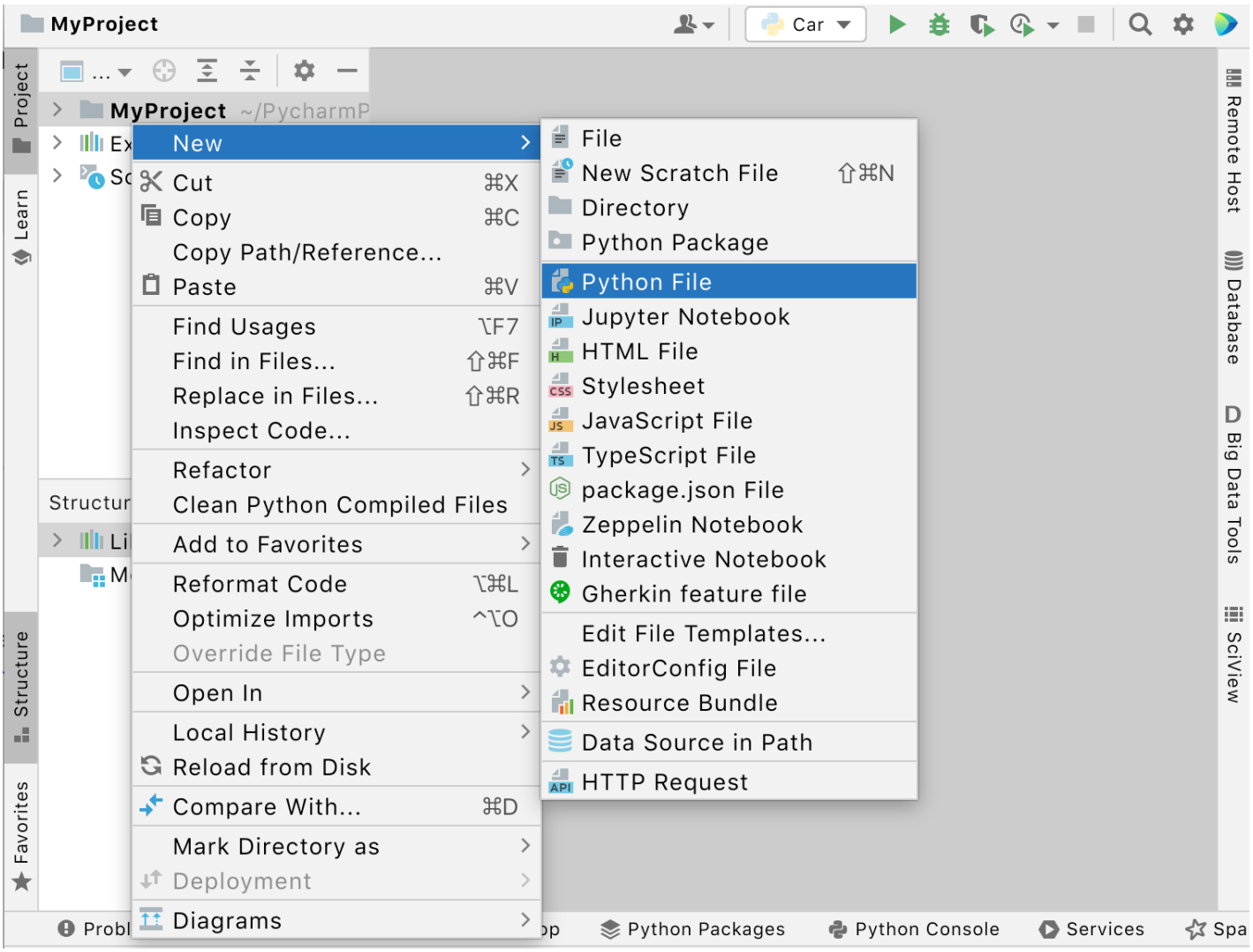


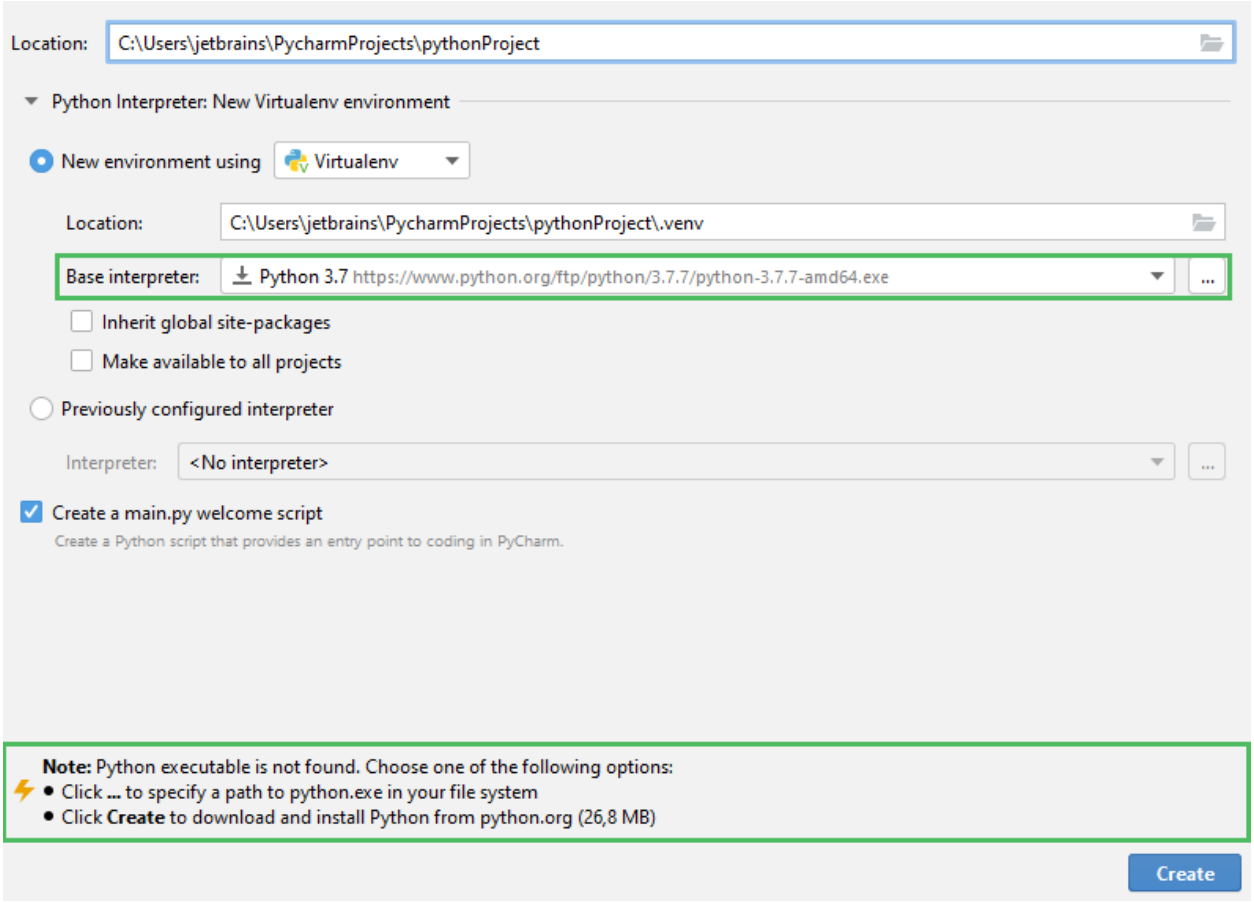
3. Choose the project location. Click button next to the Location field and specify the directory for your project.

4. Also, deselect the Create a main.py welcome script checkbox because you will create a new Python file for this tutorial.
5. Python best practice is to create a virtualenv for each project. In most cases, PyCharm create a new virtual environment automatically and you don't need to configure anything. Still, you can preview and modify the venv options. Expand the Python Interpreter: New Virtualenv Environment node and select a tool used to create a new virtual environment. Let's choose Virtualenv tool, and specify the location of the environment and the base Python interpreter used for the new virtual environment.
    6. Specify a path to the Python executable (in case of non-standard installation)
    7. Download and install the latest Python versions from python.org
    8. Install Python using the Command-Line Developer Tools (macOS only).

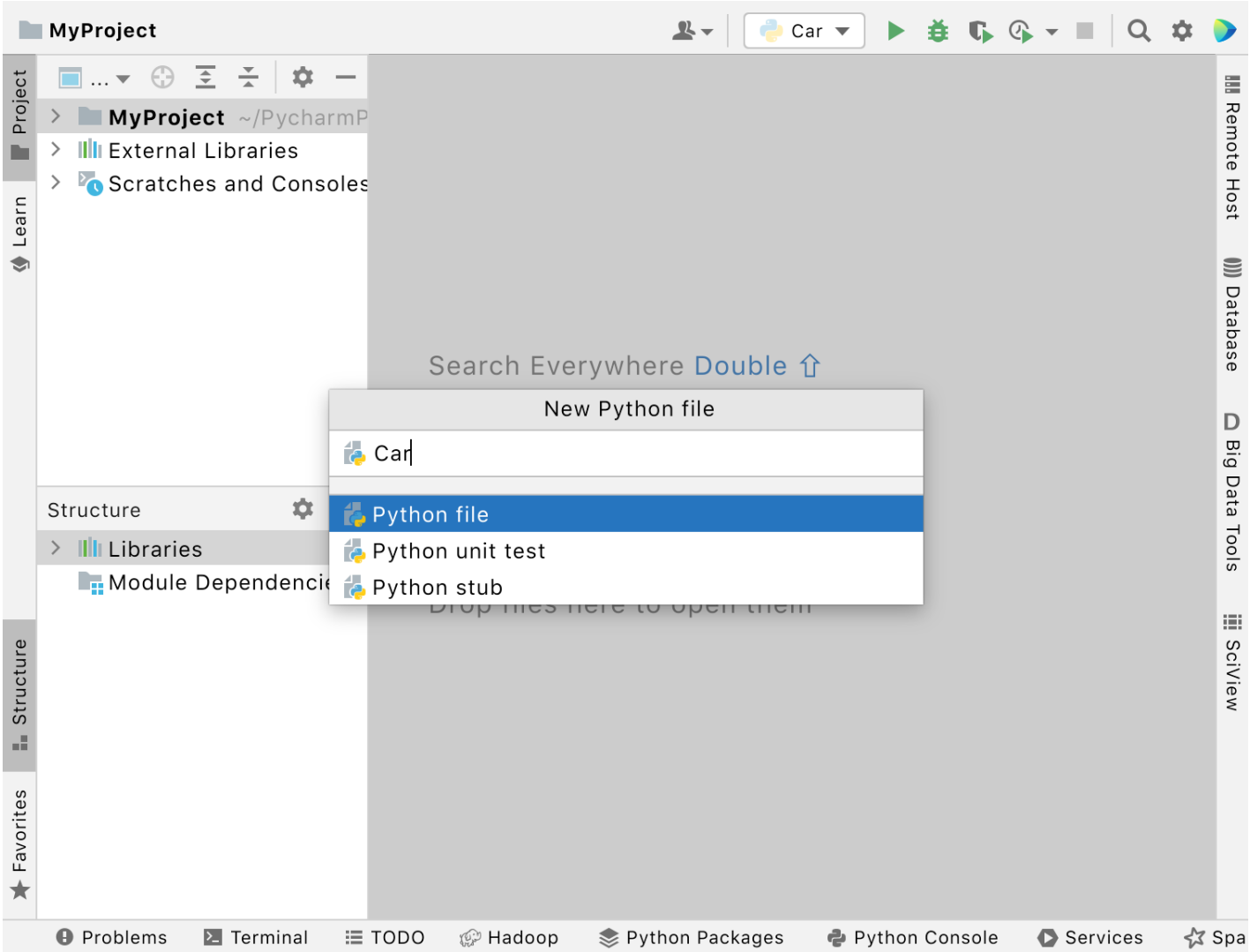
Create a Python file

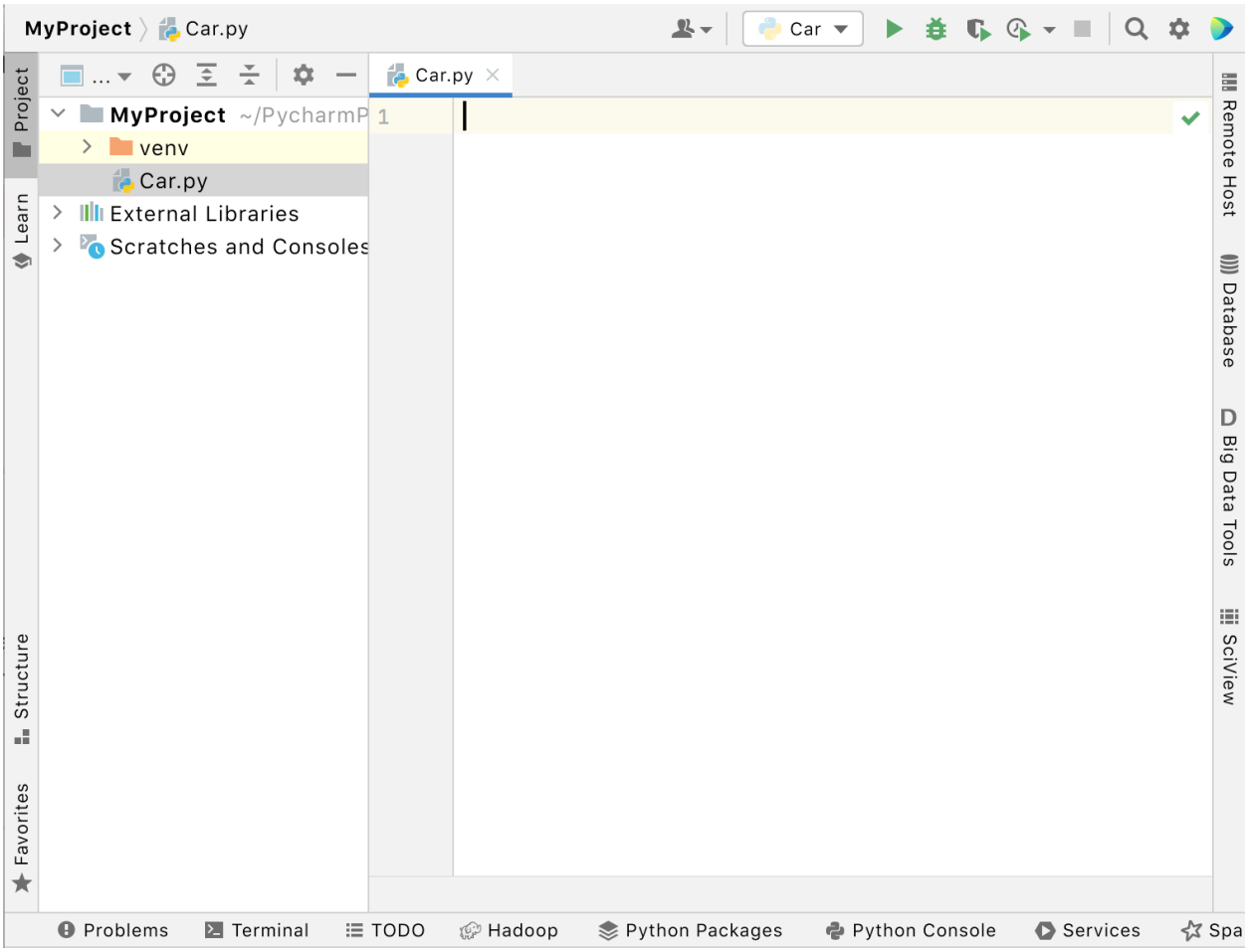
1. In the Project tool window, select the *project root* (typically, it is the root node in the project tree), right-click it, and select File | New ....


MyProject
Car
Project
Learn
Structure
Favorites
Remote Host
Database
Big Data Tools
SciView
MyProject ~/PycharmP
New
Cut ⌘X
Copy ⌘C
Copy Path/Reference...
Paste ⌘V
Find Usages ⌥F7
Find in Files... ⇧⌘F
Replace in Files... ⇧⌘R
Inspect Code...
Refactor
Clean Python Compiled Files
Add to Favorites
Reformat Code ⌥⌘L
Optimize Imports ^⌥O
Override File Type
Open In
Local History
Reload from Disk
Compare With... ⌘D
Mark Directory as
Deployment
Diagrams
File
New Scratch File ⇧⌘N
Directory
Python Package
Python File
Jupyter Notebook
HTML File
Stylesheet
JavaScript File
TypeScript File
package.json File
Zeppelin Notebook
Interactive Notebook
Gherkin feature file
Edit File Templates...
EditorConfig File
Resource Bundle
Data Source in Path
HTTP Request
Python Packages
Python Console
Services



Select the option Python File from the context menu, and then type the new filename.

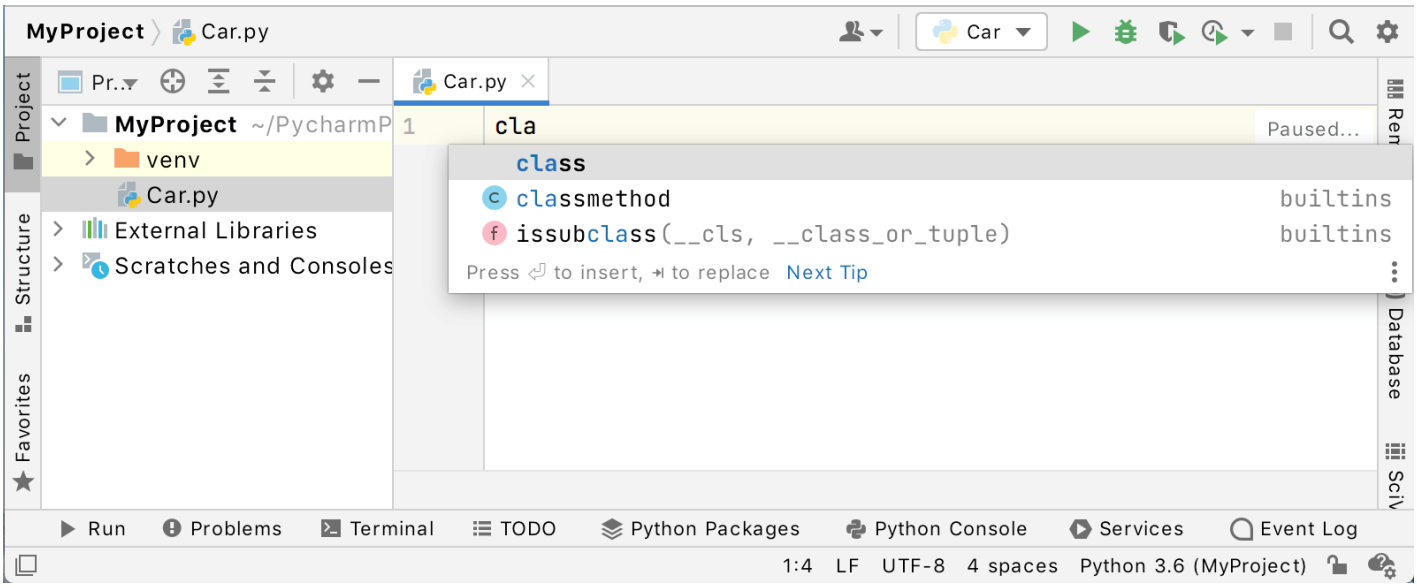

MyProject
Car
Project
Learn
MyProject ~/PycharmP
External Libraries
Scratches and Consoles
Search Everywhere Double ⇧
New Python file
Car
Python file
Python unit test
Python stub
Structure
Libraries
Module Dependencie
Remote Host
Database
Big Data Tools
SciView
Structure
Favorites
Problems
Terminal
TODO
Hadoop
Python Packages
Python Console
Services
Spa



Edit Python code

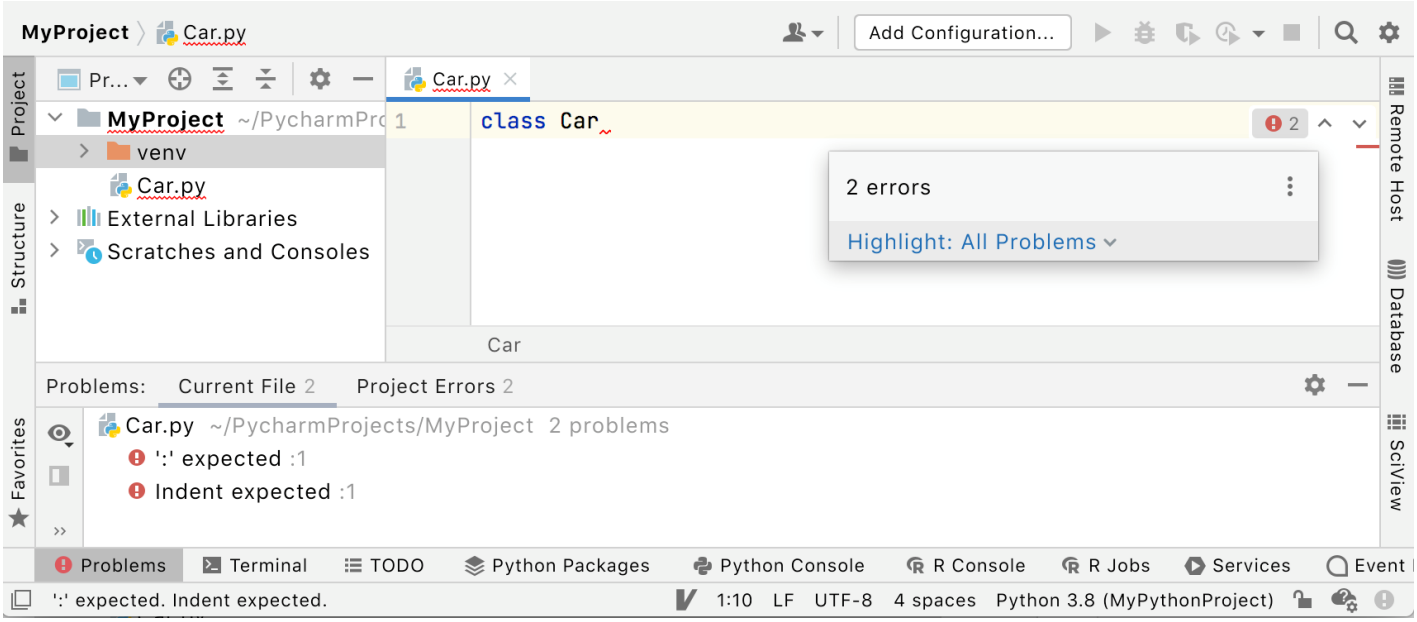
Let's start editing the Python file you've just created.

1. Start with declaring a class. Immediately as you start typing, PyCharm suggests how to complete your line:



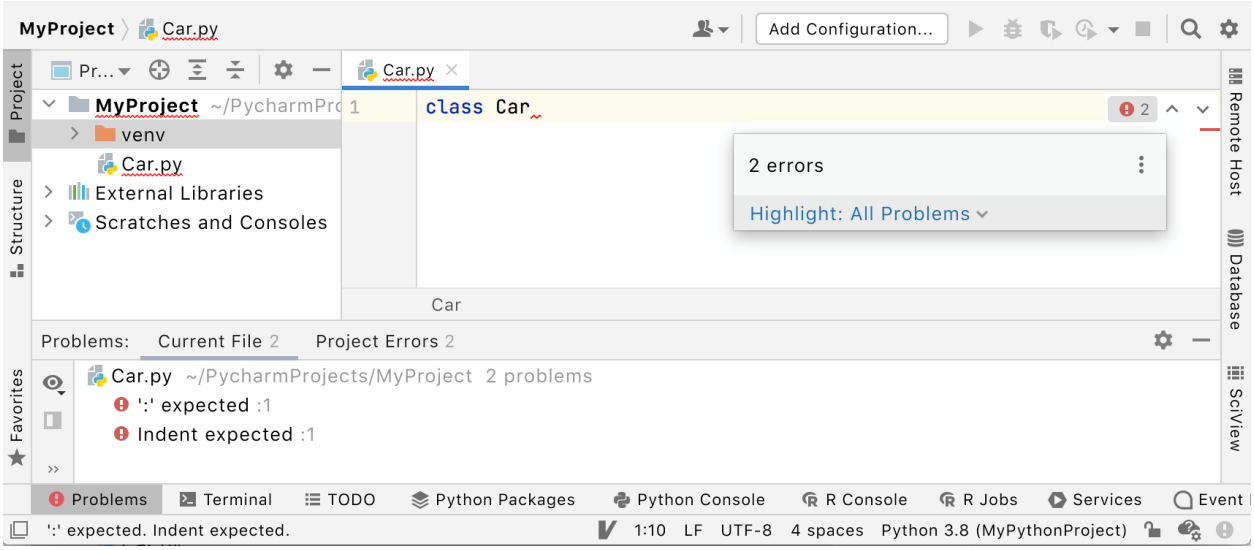
Choose the keyword class and type the class name, Car.

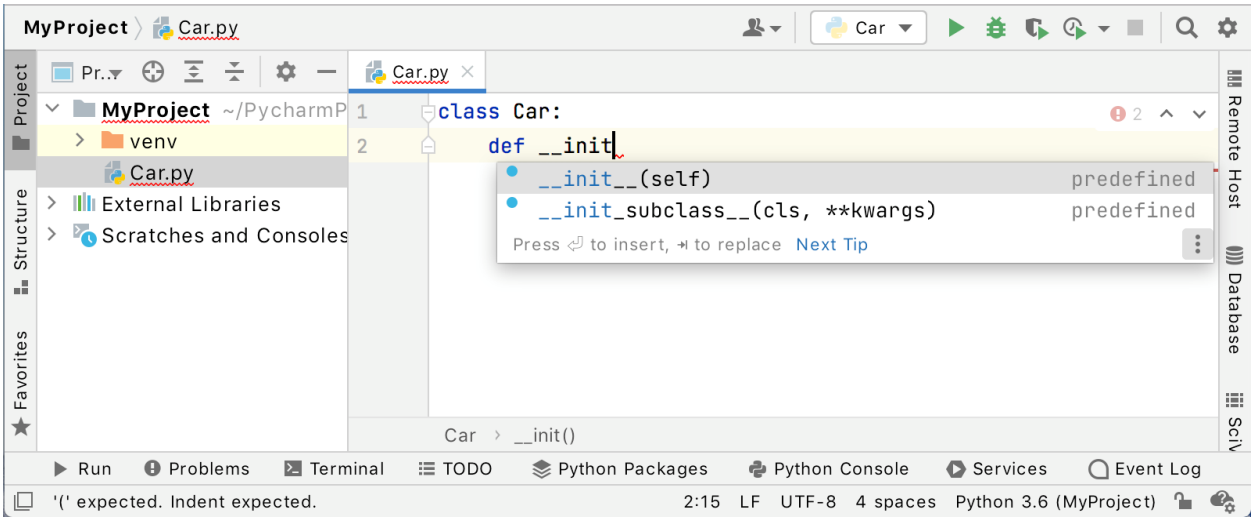
PyCharm informs you about the missing colon, then expected indentation:

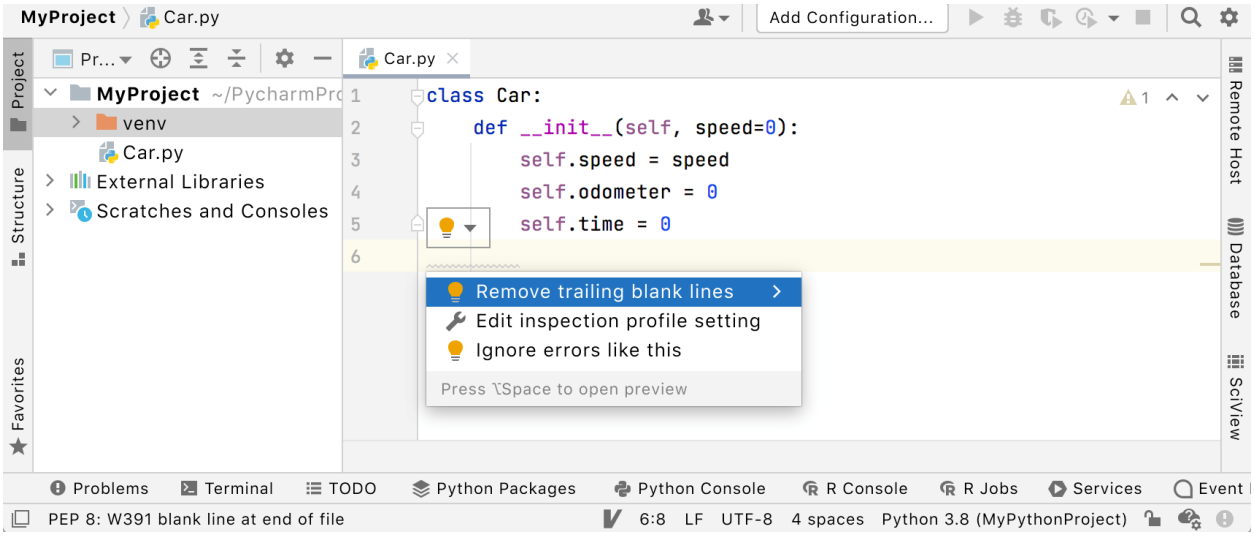


Note that PyCharm analyses your code on-the-fly, the results are immediately shown in the inspection

indicator in the upper-right corner of the editor. This inspection indication works like a traffic light: when it is green, everything is OK, and you can go on with your code; a yellow light means some minor problems that however will not affect compilation; but when the light is red, it means that you have some serious errors. Click it to preview the details in the Problems tool window

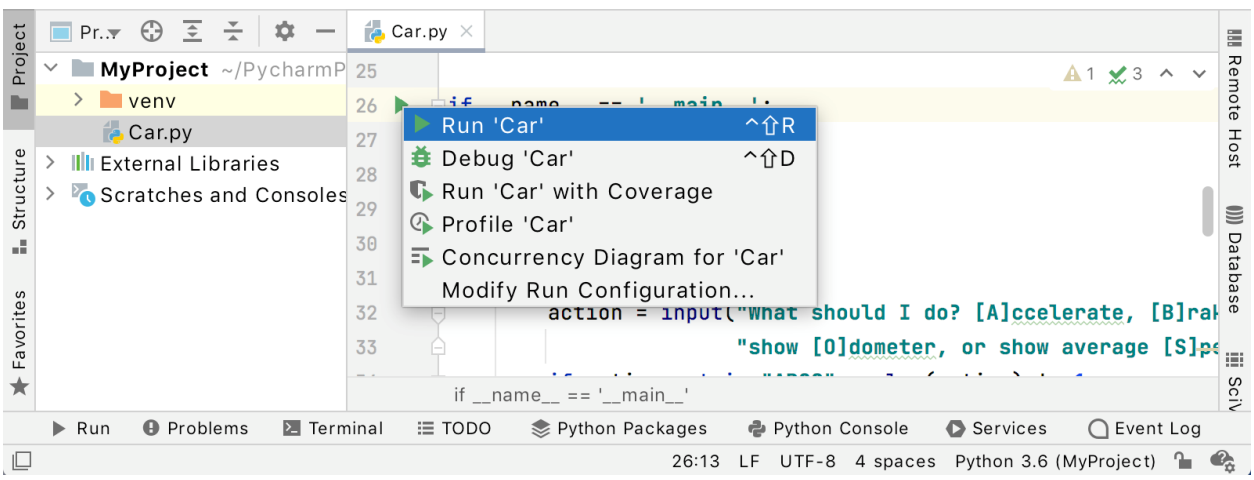







```python
class Car:

    def __init__(self, speed=0):
        self.speed = speed
        self.odometer = 0
        self.time = 0

    def say_state(self):
        print("I'm going {} kph!".format(self.speed))

    def accelerate(self):
        self.speed += 5

    def brake(self):
        if self.speed < 5:
            self.speed = 0
        else:
            self.speed -= 5

    def step(self):
        self.odometer += self.speed
        self.time += 1

    def average_speed(self):
        if self.time != 0:
            return self.odometer / self.time
        else:
            pass


if __name__ == '__main__':
```

```
    My car = Car()
    Print ("I'm a car!")
    While True:
        Action = input ("What should I do? [A]celerate, [B]rake "
                 "Show [O]Dometer or show average [S]peed?").Upper ()
        If action not in "ABOS" or len (action) != 1:
            Print ("I don't know how to do that")
            Continue
        If action == 'A':
            My_car accelerate()
        elif action == 'B':
            My_car brake()
        elif action == 'O':
            print("The car has driven {} kilometers"
format(my_car.odometer))
        elif action == 'S':
            print("The car's average speed was {} kph"
format(My_car.average_speed()))
        My_car.step()
        My_car say_state
```



MyProject 〉 Car.py

Car.py

```
if __name__ == '__main__':

    my_car = Car()
    print("I'm a car!")
    while True:
        action = input("What should I do? [A]ccelerate, [B]rake, "
                       "show [O]dometer, or show average [S]peed?").upper()
        if action not in "ABOS" or len(action) != 1:
            print("I don't know how to do that")
            continue
```

if __name__ == '__main__' 〉 while True

Run: Car

```
/Users/jetbrains/PycharmProjects/MyPythonProject/venv/bin/python
  /Users/jetbrains/PycharmProjects/MyProject/Car.py
I'm a car!
What should I do? [A]ccelerate, [B]rake, show [O]dometer, or show average
[S]peed?
```

MyProject 〉 Car.py

Car

Edit Configurations...

Save 'Car' Configuration

Car

MyProject ~/Pycha

venv

Car.py

External Libraries

Scratches and Cons

Car.py

```
class Car:
    def __init__(self, speed
        self.speed = speed
        self.odometer = 0
        self.time = 0
```