



VIRTUÁLNÍ PROSTŘEDÍ V PYTHONU

Andrej Pčelovodov



NOTEPAD++

Andrej Pčelovodov



NOTEPAD++

1. Běžte na

<https://github.com/AndreiBee/Python-Tutorials>



NOTEPAD++

1. Běžte na

<https://github.com/AndreiBee/Python-Tutorials>

2. Nainstalujte si:

Notepad++



VIRTUALNÍ PROSTŘEDÍ = PÍSKOVIŠTĚ



PÍSKOVIŠTĚ



PYTHON VENV

Andrej Pčelovodov



VIRTUALNÍ PROSTŘEDÍ = PÍSKOVIŠTĚ



PÍSKOVIŠTĚ



PYTHON VENV

- Každé dítě má svůj
vlastní prostor

🧪 VIRTUALNÍ PROSTŘEDÍ = PÍSKOVIŠTĚ



PÍSKOVIŠTĚ



PYTHON VENV

- Každé dítě má svůj **vlastní prostor**
- Hračky se **nemíchají** s ostatními



VIRTUALNÍ PROSTŘEDÍ = PÍSKOVIŠTĚ



PÍSKOVIŠTĚ



PYTHON VENV

- Každé dítě má svůj **vlastní prostor**
- Hračky se **nemíchají** s ostatními
- Když se rozbije, **prostě ho zrušíš**



VIRTUALNÍ PROSTŘEDÍ = PÍSKOVIŠTĚ



PÍSKOVIŠTĚ



PYTHON VENV

- Každé dítě má svůj **vlastní prostor**
- Hračky se **nemíchají** s ostatními
- Když se rozbije, **prostě ho zrušíš**
- Neovlivní ostatní děti



VIRTUALNÍ PROSTŘEDÍ = PÍSKOVIŠTĚ



PÍSKOVIŠTĚ

- Každé dítě má svůj **vlastní prostor**
- Hračky se **nemíchají** s ostatními
- Když se rozbije, **prostě ho zrušíš**
- Neovlivní ostatní děti



PYTHON VENV

- Každý projekt má **vlastní prostředí**



VIRTUALNÍ PROSTŘEDÍ = PÍSKOVIŠTĚ



PÍSKOVIŠTĚ

- Každé dítě má svůj **vlastní prostor**
- Hračky se **nemíchají** s ostatními
- Když se rozbije, **prostě ho zrušíš**
- Neovlivní ostatní děti



PYTHON VENV

- Každý projekt má **vlastní prostředí**
- Knihovny se **nemíchají** s jinými projekty



VIRTUALNÍ PROSTŘEDÍ = PÍSKOVIŠTĚ



PÍSKOVIŠTĚ

- Každé dítě má svůj **vlastní prostor**
- Hračky se **nemíchají** s ostatními
- Když se rozbije, **prostě ho zrušíš**
- Neovlivní ostatní děti



PYTHON VENV

- Každý projekt má **vlastní prostředí**
- Knihovny se **nemíchají** s jinými projekty
- Když se rozbije, **smažeš venv**



VIRTUALNÍ PROSTŘEDÍ = PÍSKOVIŠTĚ



PÍSKOVIŠTĚ

- Každé dítě má svůj **vlastní prostor**
- Hračky se **nemíchají** s ostatními
- Když se rozbije, **prostě ho zrušíš**
- Neovlivní ostatní děti



PYTHON VENV

- Každý projekt má **vlastní prostředí**
- Knihovny se **nemíchají** s jinými projekty
- Když se rozbije, **smažeš venv**
- Neovlivní globální Python



VIRTUALNÍ PROSTŘEDÍ = PÍSKOVIŠTĚ



PÍSKOVIŠTĚ

- Každé dítě má svůj **vlastní prostor**
- Hračky se **nemíchají** s ostatními
- Když se rozbije, **prostě ho zrušíš**
- Neovlivní ostatní děti



PYTHON VENV

- Každý projekt má **vlastní prostředí**
- Knihovny se **nemíchají** s jinými projekty
- Když se rozbije, **smažeš venv**
- Neovlivní globální Python

Andrej Pčelovodov



venv = bezpečné pískoviště pro experimenty i vývoj



PACKAGE MANAGERY

Andrej Pčelovodov



PACKAGE MANAGERY



ANACONDA®

Andrej Pčelovodov



PACKAGE MANAGERY



Andrej Pčelovodov



PACKAGE MANAGERY



Andrej Pčelovodov



PACKAGE MANAGERY



Andrej Pčelovodov



PACKAGE MANAGERY



Pro výuku a pochopení základů je důležitější **rozumět venv a pip** než používat nástroje, které všechno automatizují a schovávají.



PACKAGE MANAGERY



Pro výuku a pochopení základů je důležitější **rozumět venv a pip** než používat nástroje, které všechno automatizují a schovávají.



Nejdřív základy. Až pak nástroje.

PROČ NEPOUŽÍVÁME ANACONDU

Andrej Pčelovodov

PROČ NEPOUŽÍVÁME ANACONDU

-  Zbytečně obrovská

Anaconda má několik GB a obsahuje stovky balíků, které nikdy nepoužijeme.

✗ PROČ NEPOUŽÍVÁME ANACONDU

-  **Zbytečně obrovská**

Anaconda má několik GB a obsahuje stovky balíků, které nikdy nepoužijeme.

-  **Conda ≠ pip**

Používá vlastní balíčkovací systém → problémy s kompatibilitou a závislostmi.

✗ PROČ NEPOUŽÍVÁME ANACONDU

-  **Zbytečně obrovská**

Anaconda má několik GB a obsahuje stovky balíků, které nikdy nepoužijeme.

-  **Conda ≠ pip**

Používá vlastní balíčkovací systém → problémy s kompatibilitou a závislostmi.

-  **Uzavírá tě do vlastního ekosystému**

Mimo Anacondu je pak člověk ztracený (nezná venv, pip, pyproject.toml).

PROČ NEPOUŽÍVÁME ANACONDU

Andrej Pčelovodov

PROČ NEPOUŽÍVÁME ANACONDU

-  Špatná kompatibilita s praxí
 - CI/CD, Docker, produkční servery → všude se počítá s čistým Pythonem.

PROČ NEPOUŽÍVÁME ANACONDU

-  Špatná kompatibilita s praxí
 - CI/CD, Docker, produkční servery → všude se počítá s čistým Pythonem.
-  Nevhodná pro výuku základů
 - Skrývá, jak Python doopravdy funguje.

PROČ NEPOUŽÍVÁME ANACONDU

-  **Špatná kompatibilita s praxí**
CI/CD, Docker, produkční servery → všude se počítá s čistým Pythonem.
-  **Nevhodná pro výuku základů**
Skrývá, jak Python doopravdy funguje.
-  **Historický důvod, který už neplatí**
Dnes `pip install numpy` funguje bez problémů.

X PROČ NEPOUŽÍVÁME ANACONDU

-  **Špatná kompatibilita s praxí**
CI/CD, Docker, produkční servery → všude se počítá s čistým Pythonem.
 -  **Nevhodná pro výuku základů**
Skrývá, jak Python doopravdy funguje.
 -  **Historický důvod, který už neplatí**
Dnes `pip install numpy` funguje bez problémů.
-  **Anaconda dává smysl hlavně pro data science, ne pro vývojáře.**

Andrej Pčelovodov



VIRTUÁLNÍ PROSTŘEDÍ V PYTHONU

Virtuální prostředí (`venv`) je izolované prostředí pro Python projekt.



VIRTUÁLNÍ PROSTŘEDÍ V PYTHONU

Virtuální prostředí (`venv`) je izolované prostředí pro Python projekt.

- Každý projekt má vlastní balíčky



VIRTUÁLNÍ PROSTŘEDÍ V PYTHONU

Virtuální prostředí (`venv`) je izolované prostředí pro Python projekt.

- Každý projekt má vlastní balíčky
- Žádné konflikty verzí



VIRTUÁLNÍ PROSTŘEDÍ V PYTHONU

Virtuální prostředí (`venv`) je izolované prostředí pro Python projekt.

- Každý projekt má vlastní balíčky
- Žádné konflikty verzí
- Čistý a reprodukovatelný setup



PROČ POUŽÍVAT VENV?

Andrej Pčelovodov



PROČ POUŽÍVAT VENV?

- Různé projekty = různé verze knihoven



PROČ POUŽÍVAT VENV?

- Různé projekty = různé verze knihoven
- Co funguje dnes, musí fungovat i za rok



PROČ POUŽÍVAT VENV?

- Různé projekty = různé verze knihoven
- Co funguje dnes, musí fungovat i za rok
- Bez rozbití globální instalace Pythonu



PROČ POUŽÍVAT VENV?

- Různé projekty = různé verze knihoven
- Co funguje dnes, musí fungovat i za rok
- Bez rozbití globální instalace Pythonu



Jedno pravidlo: jeden projekt = jedno virtuální prostředí



BEZ VIRTUÁLNÍHO PROSTŘEDÍ

```
1 pip install numpy
2 pip install tensorflow
3 pip install flask
```

Andrej Pčelovodov



BEZ VIRTUÁLNÍHO PROSTŘEDÍ

```
1 pip install numpy
2 pip install tensorflow
3 pip install flask
```

Andrej Pčelovodov



BEZ VIRTUÁLNÍHO PROSTŘEDÍ

```
1 pip install numpy
2 pip install tensorflow
3 pip install flask
```



BEZ VIRTUÁLNÍHO PROSTŘEDÍ

```
1 pip install numpy
2 pip install tensorflow
3 pip install flask
```

Andrej Pčelovodov



BEZ VIRTUÁLNÍHO PROSTŘEDÍ

```
1 pip install numpy
2 pip install tensorflow
3 pip install flask
```

- Všechno se instaluje globálně



BEZ VIRTUÁLNÍHO PROSTŘEDÍ

```
1 pip install numpy
2 pip install tensorflow
3 pip install flask
```

- Všechno se instaluje globálně
- Konflikty verzí



BEZ VIRTUÁLNÍHO PROSTŘEDÍ

```
1 pip install numpy
2 pip install tensorflow
3 pip install flask
```

- Všechno se instaluje globálně
- Konflikty verzí
- „Na mém počítači to funguje“

Andrej Pčelovodov



KROK 1: VYTVOŘENÍ VENV

Otevřte PowerShell ve Windows (Terminal v MacOS/Linux)

```
1 cd D:/Path/To/Project  
2 python -m venv .venv # .venv - název složky
```



KROK 1: VYTVOŘENÍ VENV

Otevřte PowerShell ve Windows (Terminal v MacOS/Linux)

```
1 cd D:/Path/To/Project  
2 python -m venv .venv # .venv - název složky
```



KROK 1: VYTVOŘENÍ VENV

Otevřte PowerShell ve Windows (Terminal v MacOS/Linux)

```
1 cd D:/Path/To/Project  
2 python -m venv .venv # .venv - název složky
```



KROK 1: VYTVOŘENÍ VENV

Otevřte PowerShell ve Windows (Terminal v MacOS/Linux)

```
1 cd D:/Path/To/Project  
2 python -m venv .venv # .venv - název složky
```

- ".venv" je název složky



KROK 1: VYTVOŘENÍ VENV

Otevřte PowerShell ve Windows (Terminal v MacOS/Linux)

```
1 cd D:/Path/To/Project  
2 python -m venv .venv # .venv - název složky
```

- ".venv" je název složky
- Vznikne izolované prostředí



KROK 1: VYTVOŘENÍ VENV

Otevřte PowerShell ve Windows (Terminal v MacOS/Linux)

```
1 cd D:/Path/To/Project  
2 python -m venv .venv # .venv - název složky
```

- ".venv" je název složky
- Vznikne izolované prostředí
- Žádné balíčky kromě základních



KROK 2: AKTIVACE

WINDOWS:

```
1 .venv\Scripts\activate
```

LINUX / MACOS:

```
1 source .venv/bin/activate
```



KROK 2: AKTIVACE

WINDOWS:

```
1 .venv\Scripts\activate
```

LINUX / MACOS:

```
1 source .venv/bin/activate
```

Po aktivaci vidíš (**.venv**) v terminálu



KROK 3: INSTALACE A OVĚŘENÍ BALÍČKŮ

```
1 pip install requests  
2 pip install numpy  
3 pip list
```



KROK 3: INSTALACE A OVĚŘENÍ BALÍČKŮ

```
1 pip install requests  
2 pip install numpy  
3 pip list
```

Andrej Pčelovodov



KROK 3: INSTALACE A OVĚŘENÍ BALÍČKŮ

```
1 pip install requests  
2 pip install numpy  
3 pip list
```



KROK 3: INSTALACE A OVĚŘENÍ BALÍČKŮ

```
1 pip install requests  
2 pip install numpy  
3 pip list
```

- Instaluje se jen do .venv



KROK 3: INSTALACE A OVĚŘENÍ BALÍČKŮ

```
1 pip install requests  
2 pip install numpy  
3 pip list
```

- Instaluje se jen do .venv
- Globální Python zůstává čistý!



JEDNODUCHÝ PŘÍKLAD

Projekt A potřebuje:

```
1 numpy==1.21
```

Projekt B potřebuje:

```
1 numpy==2.0
```



JEDNODUCHÝ PŘÍKLAD

Projekt A potřebuje:

```
1 numpy==1.21
```

Projekt B potřebuje:

```
1 numpy==2.0
```



Bez venv problém, s venv žádný stres



REQUIREMENTS.TXT

```
1 pip freeze > requirements.txt
```

```
1 pip install -r requirements.txt
```



REQUIREMENTS.TXT

```
1 pip freeze > requirements.txt
```

```
1 pip install -r requirements.txt
```

- Přesně stejné prostředí



REQUIREMENTS.TXT

```
1 pip freeze > requirements.txt
```

```
1 pip install -r requirements.txt
```

- Přesně stejné prostředí
- Ideální pro tým i produkci



DEAKTIVACE

1 deactivate



DEAKTIVACE

1 deactivate

Vracíš se zpět do globálního Pythonu



SHRNUTÍ

Andrej Pčelovodov



SHRNUTÍ

- Virtuální prostředí je základ



SHRNUTÍ

- Virtuální prostředí je základ
- Používá se vždy, bez výjimek



SHRNUTÍ

- Virtuální prostředí je základ
- Používá se vždy, bez výjimek
- Šetří nervy, čas a rozbité projekty



SHRNUTÍ

- Virtuální prostředí je základ
- Používá se vždy, bez výjimek
- Šetří nervy, čas a rozbité projekty

*“Co není izolované,
to se dřív nebo později rozbije.”*



PRAKTICKÉ CVIČENÍ: TURTLE

Andrej Pčelovodov



PRAKTICKÉ CVIČENÍ: TURTLE

- Vytvoříme vizuální obrázek



PŘÍPRAVA PROJEKTU

```
1 mkdir turtle_art  
2 cd turtle_art
```



PŘÍPRAVA PROJEKTU

```
1 mkdir turtle_art  
2 cd turtle_art
```



PŘÍPRAVA PROJEKTU

```
1 mkdir turtle_art  
2 cd turtle_art
```



PŘÍPRAVA PROJEKTU

```
1 mkdir turtle_art  
2 cd turtle_art
```

turtle je součást standardní knihovny → nic neinstalujeme



PŘÍKLAD 1: BAREVNÁ SPIRÁLA



Andrej Pčelovodov



PŘÍKLAD 1: BAREVNÁ SPIRÁLA

```
1 import turtle
2 import random
3
4 t = turtle.Turtle()
5 t.speed(0)
6 t.pensize(20)
7
8 for i in range(100):
9     t.color(random.random(), random.random(), random.random()) # [0,1]
10    t.forward(i * 4)
11    t.right(45)
12
13 turtle.done()
```



PŘÍKLAD 1: BAREVNÁ SPIRÁLA

```
1 import turtle
2 import random
3
4 t = turtle.Turtle()
5 t.speed(0)
6 t.pensize(20)
7
8 for i in range(100):
9     t.color(random.random(), random.random(), random.random()) # [0,1]
10    t.forward(i * 4)
11    t.right(45)
12
13 turtle.done()
```



PŘÍKLAD 1: BAREVNÁ SPIRÁLA

```
1 import turtle  
2 import random  
3  
4 t = turtle.Turtle()  
5 t.speed(0)  
6 t.pensize(20)  
7  
8 for i in range(100):  
9     t.color(random.random(), random.random(), random.random()) # [0, 1]  
10    t.forward(i * 4)  
11    t.right(45)  
12  
13 turtle.done()
```



PŘÍKLAD 1: BAREVNÁ SPIRÁLA

```
1 import turtle
2 import random
3
4 t = turtle.Turtle()
5 t.speed(0)
6 t.pensize(20)
7
8 for i in range(100):
9     t.color(random.random(), random.random(), random.random()) # [0,1]
10    t.forward(i * 4)
11    t.right(45)
12
13 turtle.done()
```



PŘÍKLAD 1: BAREVNÁ SPIRÁLA

```
1 import turtle
2 import random
3
4 t = turtle.Turtle()
5 t.speed(0)
6 t.pensize(20)
7
8 for i in range(100):
9     t.color(random.random(), random.random(), random.random()) # [0,1]
10    t.forward(i * 4)
11    t.right(45)
12
13 turtle.done()
```



PŘÍKLAD 1: BAREVNÁ SPIRÁLA

```
1 import turtle
2 import random
3
4 t = turtle.Turtle()
5 t.speed(0)
6 t.pensize(20)
7
8 for i in range(100):
9     t.color(random.random(), random.random(), random.random()) # [0,1]
10    t.forward(i * 4)
11    t.right(45)
12
13 turtle.done()
```

Andrej Pčelovodov



PŘÍKLAD 1: BAREVNÁ SPIRÁLA

```
1 import turtle
2 import random
3
4 t = turtle.Turtle()
5 t.speed(0)
6 t.pensize(20)
7
8 for i in range(100):
9     t.color(random.random(), random.random(), random.random()) # [0,1]
10    t.forward(i * 4)
11    t.right(45)
12
13 turtle.done()
```



PŘÍKLAD 1: BAREVNÁ SPIRÁLA

```
1 import turtle  
2 import random  
3  
4 t = turtle.Turtle()  
5 t.speed(0)  
6 t.pensize(20)  
7  
8 for i in range(100):  
9     t.color(random.random(), random.random(), random.random()) # [0, 1]  
10    t.forward(i * 4)  
11    t.right(45)  
12  
13 turtle.done()
```



PŘÍKLAD 1: BAREVNÁ SPIRÁLA

```
1 import turtle
2 import random
3
4 t = turtle.Turtle()
5 t.speed(0)
6 t.pensize(20)
7
8 for i in range(100):
9     t.color(random.random(), random.random(), random.random()) # [0,1]
10    t.forward(i * 4)
11    t.right(45)
12
13 turtle.done()
```



PŘÍKLAD 1: BAREVNÁ SPIRÁLA

```
1 import turtle
2 import random
3
4 t = turtle.Turtle()
5 t.speed(0)
6 t.pensize(20)
7
8 for i in range(100):
9     t.color(random.random(), random.random(), random.random()) # [0,1]
10    t.forward(i * 4)
11    t.right(45)
12
13 turtle.done()
```

Andrej Pčelovodov



PŘÍKLAD 1: BAREVNÁ SPIRÁLA

```
1 import turtle  
2 import random  
3  
4 t = turtle.Turtle()  
5 t.speed(0)  
6 t.pensize(20)  
7  
8 for i in range(100):  
9     t.color(random.random(), random.random(), random.random()) # [0, 1]  
10    t.forward(i * 4)  
11    t.right(45)  
12  
13 turtle.done()
```



PŘÍKLAD 1: BAREVNÁ SPIRÁLA

```
1 import turtle  
2 import random  
3  
4 t = turtle.Turtle()  
5 t.speed(0)  
6 t.pensize(20)  
7  
8 for i in range(100):  
9     t.color(random.random(), random.random(), random.random()) # [0, 1]  
10    t.forward(i * 4)  
11    t.right(45)  
12  
13 turtle.done()
```

Notepad++ -> Run -> Run.. (F5)



PŘÍKLAD 1: BAREVNÁ SPIRÁLA

```
1 import turtle  
2 import random  
3  
4 t = turtle.Turtle()  
5 t.speed(0)  
6 t.pensize(20)  
7  
8 for i in range(100):  
9     t.color(random.random(), random.random(), random.random()) # [0, 1]  
10    t.forward(i * 4)  
11    t.right(45)  
12  
13 turtle.done()
```

Notepad++ -> Run -> Run.. (F5)

```
1 python.exe "$(FULL_CURRENT_PATH)"
```

Andrej Pčelovodov



PŘÍKLAD 1: BAREVNÁ SPIRÁLA

```
1 import turtle  
2 import random  
3  
4 t = turtle.Turtle()  
5 t.speed(0)  
6 t.pensize(20)  
7  
8 for i in range(100):  
9     t.color(random.random(), random.random(), random.random()) # [0,1]  
10    t.forward(i * 4)  
11    t.right(45)  
12  
13 turtle.done()
```

Notepad++ -> Run -> Run.. (F5)

```
1 python.exe "$(FULL_CURRENT_PATH)"
```



PŘÍKLAD 2: KVĚTINA

Andrej Pčelovodov



PŘÍKLAD 2: KVĚTINA

```
1 import turtle  
2 import random  
3  
4 t = turtle.Turtle()  
5 t.speed(0)  
6  
7 for i in range(36):  
8     t.color(random.choice(["red", "purple", "blue", "orange"]))  
9     t.circle(100)  
10    t.right(10)  
11  
12 turtle.done()
```



PŘÍKLAD 2: KVĚTINA

```
1 import turtle  
2 import random  
3  
4 t = turtle.Turtle()  
5 t.speed(0)  
6  
7 for i in range(36):  
8     t.color(random.choice(["red", "purple", "blue", "orange"]))  
9     t.circle(100)  
10    t.right(10)  
11  
12 turtle.done()
```

Notepad++ -> Run -> Run.. (F5)



PŘÍKLAD 2: KVĚTINA

```
1 import turtle  
2 import random  
3  
4 t = turtle.Turtle()  
5 t.speed(0)  
6  
7 for i in range(36):  
8     t.color(random.choice(["red", "purple", "blue", "orange"]))  
9     t.circle(100)  
10    t.right(10)  
11  
12 turtle.done()
```

Notepad++ -> Run -> Run.. (F5)

Ukázka rotace a opakování

PŘÍKLAD 3: GEOMETRICKÝ CHAOS

```
1 import turtle  
2 import random  
3  
4 t = turtle.Turtle()  
5 t.speed(0)  
6  
7 for _ in range(50):  
8     t.penup()  
9     t.goto(random.randint(-200, 200), random.randint(-200, 200))  
10    t.pendown()  
11  
12    t.color(random.random(), random.random(), random.random())  
13    sides = random.randint(3, 8)  
14  
15    for i in range(sides):
```

PŘÍKLAD 3: GEOMETRICKÝ CHAOS

```
1 import turtle  
2 import random  
3  
4 t = turtle.Turtle()  
5 t.speed(0)  
6  
7 for _ in range(50):  
8     t.penup()  
9     t.goto(random.randint(-200, 200), random.randint(-200, 200))  
10    t.pendown()  
11  
12    t.color(random.random(), random.random(), random.random())  
13    sides = random.randint(3, 8)  
14  
15    for i in range(sides):
```

Každý má unikátní chaos

Andrej Pčelovodov



DOBROVOLNÝ ÚKOL

Andrej Pčelovodov



DOBROVOLNÝ ÚKOL

Dokumentace Python Turtle:

<https://docs.python.org/3/library/turtle.html>



DOBROVOLNÝ ÚKOL

Dokumentace Python Turtle:

<https://docs.python.org/3/library/turtle.html>

- Přidej podpis (jméno) do rohu



DOBROVOLNÝ ÚKOL

Dokumentace Python Turtle:

<https://docs.python.org/3/library/turtle.html>

- Přidej podpis (jméno) do rohu
- Změň velikost okna a pozadí



PLANTUML V NOTE PAD++

Andrej Pčelovodov



PLANTUML V NOTEPAD++

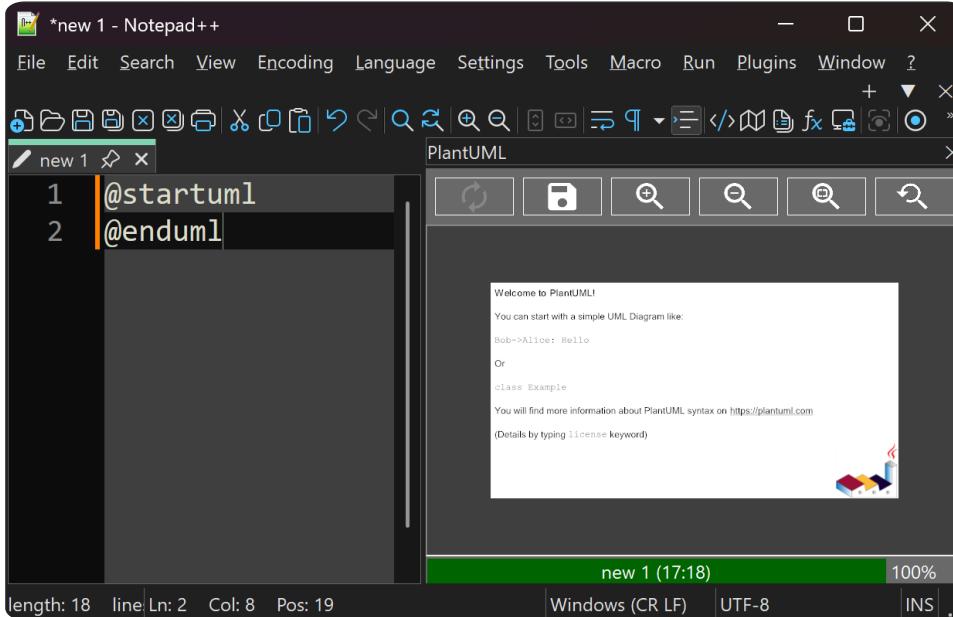
Plugins -> Plugins Admin.. -> PlantUML -> Install

Andrej Pčelovodov



PLANTUML V NOTEPAD++

Plugins -> Plugins Admin.. -> PlantUML -> Install



Andrej Pčelovodov