

TDP003 Projekt: Egna datormiljön

Installationsmanual

Författare

Hadi Ansari, hadan326@student.liu.se

Nils Bark, nilba048@student.liu.se

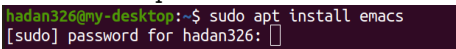
1 Revisionshistorik

Ver.	Revisionsbeskrivning	Datum
1.0	Första versionen av installationsmanualen	2020-09-17
1.1	Komplettering med Flask och Jinja2	2020-09-22

2 Skriva kod

För att skriva sin kod kan man använda en texteditor som heter Emacs. Ni kan läsa mer om hur man kan installera Emacs på sin linuxbaserade operativsystem här:

Installation av Emacs

1. Öppna terminalen genom att trycka på ALT + Ctrl + t på tangentbordet
2. Skriv `sudo apt install emacs`

3. Skriv in ditt lösenord
4. Vänta tills emacs installeras
5. För att köra emacs efter att det installerats kan man skriva `emacs` i terminalen
6. För att öppna en fil i emacs så skriver du `emacs <filnamn>`

Vanliga fel

- Om du vill ha tillgång till terminalfönstret samtidigt som du har öppnat emacs från det fönstret så kan du lösa det genom att lägga till `&` på slutet av ditt emacs-kommando
- Om du har en fil öppen som inte verkar ha färgat texten på rätt sätt, eller inte alls, så beror det troligen på att du har fel (eller inget) filformat på din fil. Åtgärda det och starta sedan om emacs så bör det automatiskt upptäcka formatet

3 Köra och felsöka kod

För att kunna köra och felsöka sin kod kan man använda sig av följande program.

Skal (terminal)

Terminalen kan användas som en plattform där vi kan köra vår kod och se resultatet av koden. I terminalen kan man även få några felmeddelanden som uppstår när något i koden inte fungerar bra. Ett exempel på detta kan vara om man försöker dela ett tal med noll. Att kontrollera och kunna tolka dessa räknas som en felsökningsmetod som är tillgänglig direkt i terminalen.

För att köra din pythonkod i terminalen så skriver du `python3 <filnamn>` (Glöm inte .py!). Om du vill köra koden i interaktivt läge, som gör att man stannar i pythonmiljön när koden har körts, så får du lägga till `-i` efter `python3`. Detta kan vara användbart om man vill t.ex testa specifika delar av sin kod.

Ett tips: Om du har råkat köra ett program som inte kan avslutas, eller som loopar oändligt, så kan du trycka Ctrl-C för att tvinga det att avslutas!

Pythontutor

VISUALIZE CODE EXECUTION

Learn Python, Java, C, C++, JavaScript, and Ruby

Python Tutor helps people overcome a fundamental barrier to learning programming: understanding what happens as the computer runs each line of code. You can use it to write Python, Java, C, C++, JavaScript, and Ruby code in your web browser and see its execution visualized step by step.

Related services: [Java Tutor](#), [C Tutor](#), [C++ Tutor](#), [JavaScript Tutor](#), [Ruby Tutor](#)

Over ten million people in more than 180 countries have used Python Tutor to visualize over 100 million pieces of code, often as a supplement to textbooks, lectures, and online tutorials. To our knowledge, it is the most widely-used program visualization tool for computing education. Research that references Python Tutor can cite this paper: *Online Python Tutor: Embeddable Web-Based Program Visualization for CS Education*. *ACM Technical Symposium on Computer Science Education (SIGCSE)*, 2013. [\[ACM DL\]](#)

Start visualizing your code now

You can also embed visualizations into any webpage. Here is a Python example:

The screenshot displays the Python Tutor interface. On the left, a code editor shows a Python 3.6 script:


```

1 def listSum(numbers):
2     if not numbers:
3         return 0
4     else:
5         (f, rest) = numbers
6         return f + listSum(rest)
7
8 myList = (1, (2, (3, None)))
9 total = listSum(myList)
    
```

 Below the code, a legend indicates that lines with '==>' are just executed and lines with '=>' are next to execute. At the bottom, navigation buttons for 'Prev' and 'Next' are visible, along with a step counter showing 'Step 11 of 22'.

On the right, a visual representation of the program state is shown. It includes a 'Frames' pane with a 'Global frame' containing 'listSum' and 'myList'. The 'listSum' frame is active, showing 'numbers' as '(1, 2, 3, None)' and 'f' as '1'. The 'Objects' pane shows the 'listSum' function object and three tuple objects: '(1, 2, 3, None)', '(2, 3, None)', and '(3, None)'. Arrows indicate the call stack and the current state of the 'numbers' list and the 'f' variable.

Rendered by [Python Tutor](#)
[Customize your settings](#) (NEW)

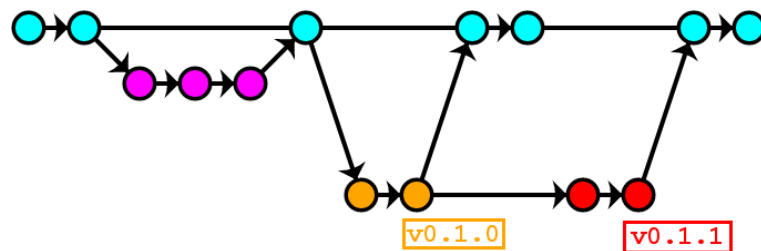
Pythontutor är ett bra verktyg för att kunna köra pythonkod och se steg för steg hur programmet körs. Det kan vara mycket användbart när man vill felsöka några delar av sin kod som inte fungerar.

Pythontutor är en webbsida som man kan få tillgång till genom att skriva adressen <http://www.pythontutor.com> i webbläsaren. Klicka på “Start visualising your code now” För att börja. Sen är det bara att kopiera sin kod och klicka på “Visualize execution” för att se hur din kod körs. Se till att välja den senaste versionen av python, och att du har den senaste versionen av python installerad på ditt system.

4 Versionhantering

För att kunna hålla koll på sin utveckling av koden och ha olika versioner av koden sparade på en och samma plats så kan man använda sig av git tillsammans med sitt konto på GitLab. På så sätt kan man ha olika versioner av sin kod sparade vid olika tillfällen och man har möjlighet att gå tillbaka till en specifik version av koden.

Git och Gitlab



Så här gör du för att skapa och länka ett GitLab repository till en mapp så att du kan använda dig av Git:

- Först så går du till GitLabs hemsida (www.gitlab.com) och loggar in.
- Klicka på “create project” och ge det ett namn.
- Gå in i “ssh-key” i inställningarna och kopiera in din dators ssh-nyckel.

- Du genererar en nyckel genom `ssh-keygen -t rsa -C "Gitlab" -b 4096` och hittar nyckeln i “`~/ssh/id_rsa.pub`”
- Navigera till mappen du vill lägga ditt projekt i
- Kopiera och kör kommandot som börjar med `git clone` från projektets hemsida
- Du kan sedan spara dina ändringar lokalt med `git commit` och sedan spara dem på ditt repository med `git push`
- Använd `git pull` för att hämta hem den senaste versionen av dina filer

Vanliga fel

- Om du har problem med att köra ditt `git clone` kommando så beror det troligen på att du inte har länkat rätt SSH-nyckel eller att du har fel URL i `git clone` kommandot. Dubbelkolla att du har tagit hela ssh-nyckeln från textfilen, och att du kopierar hela `git clone` kommandot som finns på hemsidan till ditt nya och tomma projekt.
- Om git vill att du väljer att namn/email innan du kan använda `commit/push/pull` så gör du det med följande två kommandon: `git config --global user.name "<Ditt namn>"` och `git config --global user.email "<Din email-adress>"`

Flask och Jinja

Flask är ett python-ramverk som används inom webbutveckling för flera olika uppgifter. Ett av de packet som följer med Flask är Jinja som bl.a används för att skapa mallar av html-sidor som kan underlätta utvecklingen av en hemsida rejält. T.ex kan man genom att använda Jinja se till att en navbar och footer automatiskt dyker upp på alla sidor, utan att man på egen hand måste kopiera över koden.

Installera Flask (och Jinja)

Flaskutvecklarna rekommenderar att man skapar ett “virtual environment” (vi kommer kalla det för en virtuell miljö) för python för att få bättre kompatibilitet när du jobbar med flask, vilket vi tänker göra här:

- Skapa först en ny mapp på valfri plats med passande namn
- Gå in i mappen med `cd`
- Skriv `python3 -m venv venv`
- Gå igenom installationen
- Skriv `. venv/bin/activate` för att aktivera miljön
- Sedan installerar vi flask med `pip install Flask`
- Gå även igenom denna installationen som vanligt
- Om allt har gått om det ska så får du detta meddelandet. Notera att Jinja installerades tillsammans med flask:

```
Successfully installed Flask-1.1.2 Jinja2-2.11.2 MarkupSafe-1.1.1 Werkzeug-1.0.1 click-7.1.2 itsdangerous-1.1.0
```
- Testa om allt fungerar som det ska genom att göra en ny .py fil där du skriver `from flask import Flask`. Se till att den virtuella miljön är aktiverad (Det ska stå “(venv)” innan ditt namn i terminalen) och kör sedan filen. Om du inte får några felmeddelanden så fungerar det!

Vanliga fel

- Om du får ett felmeddelande när du försöker skapa din virtuella miljö så är det troligt att du saknas ett pythonpaket. Installera det med `sudo apt-get install python3-venv` och skriv in ditt lösenord.
- För att kunna importera och använda flask i dina pythonfiler kommer du behöva aktivera din virtuella miljö innan du kör dem (`. venv/bin/activate`). För att tillbaka till den vanliga linuxmiljön skriver du bara `deactivate` i terminalen