Skolan för Datavetenskap och kommunikation

Programmeringsteknik

Föreläsning 16

Grafiskt användargränssnitt (GUI)

Använd modulen **tkinter**, som har klasser för *komponenter*.

Se "Referenser" på kursens webbsida: GUI-länkar

Button	Menu
Canvas	Menubutton
Checkbutton	Message
Entry	Radiobutton
Frame	Scale
Label	Scrollbar
Listbox	Text

Hur gör man?

```
from tkinter import *
roten = Tk()
knapp = Button(roten, text="Tryck")
knapp.pack()
roten.mainloop()
```

Hämtar alla klasser i modulen Tkinter.

Tk-konstruktorn - skapar rotfönstret.

Button-konstruktorn - skapar en knapp.

Knappen placeras.

Startar en slinga som väntar på inmatning från användaren.

Komponenter

- Knappar och annat kallas *komponenter* och är objekt.
- Varje komponent har en konstruktor med många defaultparametrar.
- Anropa bara med det som behövs:

```
knapp = Button(roten,text="Handla")
```

- Första parameter ska vara roten

Ändra attribut

- Attributen kan ändras ett i taget:

```
knapp["text"] = "Klart"
```

- Med metoden *config* kan man ändra flera attribut åt gången:

```
knapp.config(bg = "lightblue",
     height = 3, width = 9,
     font = ('times', 20, 'italic'))
```

- Här ändrar vi knappens färg, storlek, och textfont.

Anropa funktion med knapptryck!

- Ett attribut som alla komponenter har är `command`
- Där anger man vilken metod/funktion som ska anropas när komponenten används.
- Om vi skriver en funktion addera() som ska anropas när nån trycker på knapp så kan vi koppla ihop funktion med knapp så här:

```
knapp["command"] = addera
```

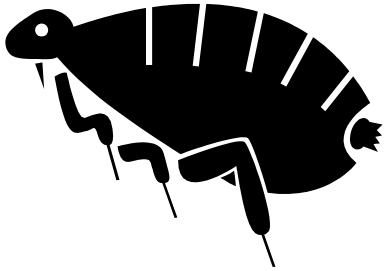
- Man kan också koppla ihop knappen med en funktion när den skapas:

```
from tkinter import *

def byttext():
    knapp["text"] = "Aj!"

roten = Tk()
knapp = Button(roten,
               text = "Tryck inte",
               command = byttext)
knapp.pack()
roten.mainloop()
```

GUI-exempel

- Exemplen finns på KTH Social under "GUI-exempel"

Olika typer av fel



- Felavbrott (Exception) när programet körs
- Inget händer när man kör programmet
- Massor av text rinner över skärmen
- Programmet gör något annat än det man ville
- Programmet gör rätt för vissa indata, men inte för andra

Felavbrott

Lär dig tolka felutskrifter!

```
Traceback (most recent call last):
  File ”filmer.py", line 124, in <module>
    titta(listan)
  File ”filmer.py", line 102, in titta
    film.ny_visning(1)
TypeError: ny_visning() takes exactly 1
 argument (2 given)
```

Tolkning

- Sista raden förklarar felet!

```
TypeError: ny_visning() takes exactly 1 argument (2 given)
```

- Raderna ovanför visar anropskedjan. Läs nerifrån och uppåt! Felet uppstod på rad 102, i funktionen titta som anropades på rad 124.

Kontrollutskrifter

- Använd kontrollutskrifter för att hittar var i programmet felet uppstår.
- En kontrullutskrift är en vanlig print-sats, till exempel:

```
print "Klar med inläsningen"
```

- Eller stanna upp så här:

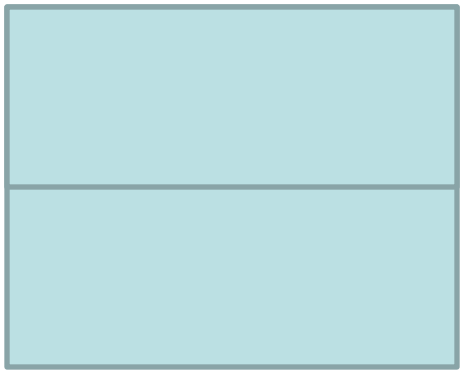
```
raw_input("Tryck Enter")
```

- Du kan också skriva ut variabelvärden för att se hur dom ändras under körning.

Mittiprick-metoden

Anta att programmet hänger sig, men vi vet inte var i programmet det inträffar.

1. Lägg en kontrollutskrift i början och en i slutet. Blev det fel däremellan?
2. Lägg då in en kontrollutskrift mitt i. Om den kommer ut som den ska finns felet i andra halvan, annars i första.
3. Fortsätt tills du hittat felet!



Testning

- Skriv upp (i en textfil) hur du testar programmet. Missa inte specialfallen.
- Efter att du har fixat ett fel – testa att programmet fortfarande fungerar för *alla* testfallen.

Hjälpmedel (överkurs)

- Är du van att använda en debugger?

  Titta på modulen pdb
- Vill du infoga automatisk testning i programmet?

  Titta på modulen doctest

På torsdag

- Felhantering (ingår ofta i C-uppgiften)
- Mer GUI