Digitális képfeldolgozás
Python modulok használata

Horváth András, SZE GIVK FKT

v 1.0

Alapgondolat

A Python alaprendszere csak az elemi dolgokat ismeri.
Speciális tudás: „csomagokban”, úgynevezett „modul”-okban.

Alapgondolat

A Python alaprendszere csak az elemi dolgokat ismeri.
Speciális tudás: „csomagokban”, úgynevezett „modul”-okban.

Már használtunk ilyet eddig is, pl. az OpenCV csomag rutinjait az „`import cv2`” utasítással töltöttük be.

E prezentáció témája:

- Hogyan tölthetünk be mások által írt modulokat. (kötelező tudni)
- Hogyan tehetjük saját eljárásainkat modulokba. (hasznos tudni)

Mások által írt modulok használata

Többféleképp is behívhatjuk a modulokat. Példa: matematikai modul.

Mások által írt modulok használata

Többféleképp is behívhatjuk a modulokat. Példa: matematikai modul.

„`import math`”: A modul eljárásai és változói a „`math.`” előtaggal érhetők el.
Pl.: `math.sin()`, `math.e`, ...

Mások által írt modulok használata

Többféleképp is behívhatjuk a modulokat. Példa: matematikai modul.

„`import math`”: A modul eljárásai és változói a „`math.`” előtaggal érhetők el.
Pl.: `math.sin()`, `math.e`, ...

„`import math as m`”: A modul eljárásai és változói az „`m.`” előtaggal érhetők el.
Pl.: `m.sin()`, `m.e`, ...

Mások által írt modulok használata

Többféleképp is behívhatjuk a modulokat. Példa: matematikai modul.

„`import math`”: A modul eljárásai és változói a „`math.`” előtaggal érhetők el.
Pl.: `math.sin()`, `math.e`, ...

„`import math as m`”: A modul eljárásai és változói az „`m.`” előtaggal érhetők el.
Pl.: `m.sin()`, `m.e`, ...

„`from math import *`”: A modul eljárásai és változói a előtag nélkül érhetők el.
Pl.: `sin()`, `e`, ...

Mások által írt modulok használata

Többféleképp is behívhatjuk a modulokat. Példa: matematikai modul.

„`import math`”: A modul eljárásai és változói a „`math.`” előtaggal érhetők el.
Pl.: `math.sin()`, `math.e`, ...

„`import math as m`”: A modul eljárásai és változói az „`m.`” előtaggal érhetők el.
Pl.: `m.sin()`, `m.e`, ...

„`from math import *`”: A modul eljárásai és változói a előtag nélkül érhetők el.
Pl.: `sin()`, `e`, ...

„`from math import sin,cos`”: Csak a megadott eljárások érhetők el.
Pl.: `sin()`, `cos()`, de a többi nem.

Mire jó ez?

A modulok többféle betölthetősége segít elkerülni a névütközéseket.

„`from math import *`”: lesz egy „e” és egyből „`pi`” nevű változónk a nevezetes állandók értékével, de ezt akármikor felülírhatjuk, fontos információt elvesztve ezzel.

Az „`import math`”: a π-re mindig „`math.pi`”-ként hivatkozhatunk, ami csökkenti a véletlen felülírás esélyét.

Mire jó ez?

A modulok többféle betölthetősége segít elkerülni a névütközéseket.

„`from math import *`”: lesz egy „e” és egyből „pi” nevű változónk a nevezetes állandók értékével, de ezt akármikor felülírhatjuk, fontos információt elvesztve ezzel.

Az „`import math`”: a π-re mindig „`math.pi`”-ként hivatkozhatunk, ami csökkenti a véletlen felülírás esélyét.

„`import math as m`”: hasznos, mert kevesebbet kell gépelni. („`m.pi`”).
De ekkor vigyázzunk, nehogy más modult is „m” néven importáljunk!

Mire jó ez?

A modulok többféle betölthetősége segít elkerülni a névütközéseket.

„`from math import *`”: lesz egy „e” és egyből „`pi`” nevű változónk a nevezetes állandók értékével, de ezt akármikor felülírhatjuk, fontos információt elvesztve ezzel.

Az „`import math`”: a π-re mindig „`math.pi`”-ként hivatkozhatunk, ami csökkenti a véletlen felülírás esélyét.

„`import math as m`”: hasznos, mert kevesebbet kell gépelni. („`m.pi`”).
De ekkor vigyázzunk, nehogy más modult is „m” néven importáljunk!

A „`from math import *`” változat nagyon veszélyes! Próbáljuk elkerülni, ha csak lehet.

Elnevezés: névtér

Az előzőeket úgy szokás mondani, hogy:
„`from math import *`”: a math modul objektumait beemeltük a közös névtérbe.
„`import math`”: a math modul objektumait saját névtérben érjük el.
„`import math as m`”: a math modul objektumait az „m.” névtérben érjük el.

Elnevezés: névtér

Az előzőeket úgy szokás mondani, hogy:
„`from math import *`”: a math modul objektumait beemeltük a közös névtérbe.
„`import math`”: a math modul objektumait saját névtérben érjük el.
„`import math as m`”: a math modul objektumait az „m.” névtérben érjük el.

Többi modul: ugyanígy működik.

„cv2” = „OpenCV”
Szokás: „`import cv2`” (vagy az újabb divat: „`import cv2 as cv`”)

„numpy” = „Numerical Python”
Szokás: „`import numpy as np`” (de van aki szeret gépelni: „`import numpy`”)

Saját modulok írása

Saját Python változóinkat és függvényeinket célszerű modulokba szervezni.
Fő lépések:

1. Megszerkesztjük a modul tartalmát és elmentjük a merevlemezre mondjuk „`mymodul.py`” néven.
   (IDLE-ben új ablakban, de akár a Notepad-del is.)
2. Ahonnan hívni akarjuk a modult, ott kiadjuk az „`import mymodul`” parancsot.
   (vagy más névtérben: „`import mymodul as mymo`”)
3. Használjuk a modul változóit, eljárásait.

De honnan tudja a rendszer, hol keresse a mi modulunkat?

A modulok elérése

Nem lenne jó, ha minden „`import`” parancs mindenütt keresné a modult a merevlemezen.

Python: `sys` modul `path` nevű listája tartalmazza a felkeresett könyvtárneveket.

`import sys` : hívjuk be a modult saját névtérbe.
`print(sys.path)` : nézzük meg, mi van most a listán.
`sys.path+=[r"C:...."]` : kibővítjük a listát azzal a könyvtárral, ahol a mi modulunk található.
(Vagy: `sys.path.append(r"C:...")`)

Megjegyzés: a legtöbb Python telepítéskor a munkakönyvtár belekerül a sys.path-ba:
⇒ Ha a hívó program és a modul ugyanabban a könyvtárban van, nem kell sys.path-t állítani.

A modulok újra betöltése interaktív módban

Saját modult gyakran sokszor átszerkesztünk.
Az új tartalom érvényessé tétele: „`reload`” parancs:

```
reload(mymodul)
```

A modulok újra betöltése interaktív módban

Saját modult gyakran sokszor átszerkesztünk.
Az új tartalom érvényessé tétele: „`reload`” parancs:

```
reload(mymodul)
```

(Az utasítás arra is jó, hogy ha véletlen felülírtunk egy modul változót, akkor ez kijavítja.)

Ennek csak interaktív módban van értelme.

Megjegyzések

1. A rendszerrel érkező modulok olyan helyre kerülnek, melyek eleve a „`sys.path`” részei.
2. Saját modult rendszerterületre téve nem kell bajlódni a „`sys.path`”-szal. (Akkor célszerű, ha egy ideig nem nyúlunk a modulhoz.)
3. Célszerű a kicsit is bonyolultabb dolgokat modulba írni és az interaktív felületről ezeket csak használni.
4. Ha a saját modul függvényeibe a „def” utáni sorba egy stringet teszünk, az a függvény rövid dokumentációja és ezt az IDLE buboréksúgóban megmutatja nekünk.
5. Lehet modult írni nemcsak Pythonban, de mi ekkor is Python utasításokkal érhetjük el. (Pl. az OpenCV rutinjai C++-ban vannak megírva.)