Eseménykezelés

Bármilyen rendszer alatt dolgozunk is, legyen az X vagy Windows, a háttérben az egér egyetlen elmozdulását is események egész sora követi. Így van ez mindennel: esemény keletkezik, ha lenyomunk egy gombot, ha kattintunk valahol, sőt még akkor is, ha véletlenül meglökjük az egeret. A futó rendszer dönti el, hogy a pillanatnyilag zajló esemény az alkalmazásunkra tartozik-e vagy teljességgel lényegtelen. Az eseményeknek három fő típusa van: a billentyűzettel, az egérkezeléssel és az ablak helyével, illetve méretével kapcsolatosak. Ezek közül a billentyűzettel összefüggő eseményfajta az egyetlen, amelynek elfogásához az ablakunknak kell az aktív ablaknak, azaz a beviteli fókusszal bíró ablaknak lennie.
Egy eseményt kétféle módon lehet elfogni: vagy közvetlenül a `bind()` eljárással, vagy – amennyiben gombról van szó – a ***command*** tulajdonság megadásával.

```
btn.bind('<ButtonRelease-1>', eseménykezelő)
```

Ebben a példában gombunk objektumához egy eseményt rendelünk. Kikötjük, hogy amennyiben a gombunkon valaki a bal oldali egérgombot nyomja le, hajtsa végre az eseménykezelő által hivatkozott eljárást. Ennek legelső értéke kötelező érvényű, a `bind()` ezen keresztül tudatja az eljárással, hogy milyen esemény történt valójában. Ha függvényünket a ***command*** tulajdonságon keresztül hívatjuk vissza, erre a kötelező értékre nincs szükség.
Az alábbiakban felsorolunk néhány fontosabb eseményazonosítót:

ANY-ENTER	Az egérmutató az elem területére lépett.
BUTTON-1	Az egyes egérgombot lenyomták az objektum területén.
BUTTONRELEASE-2	Az objektum területén a kettes egérgombot felengedték.
KEYPRESS	Lenyomtak egy billentyűt.
CONTROL-SHIFT-F1	Az adott elemen lenyomták az adott billentyűkombinációt.
CONFIGURE	Az ablak mérete vagy helyzete megváltozott.
FOCUSIN	Ablakunk lett az aktív ablak.
DESTROY	Programunk bezárás alatt van.
A	Lenyomott A betű.

Ha gyakran végrehajtódó eseményhez rendelünk eseménykezelőt, ügyeljünk rá, hogy egy nagyobb függvény nagyon lefoglalhatja a processzorunkat, ezért igyekezzük elkerülni.
Az eseménykezelők egy alkalmazásban több szinten is beállíthatók. Alapértelmezésben a beállítás csak egy adott elemre vonatkozik. Ezenkívül még három szint létezik: az alkalmazásszint, amely egy adott alkalmazás minden ablakára és elemére vonatkozik; az osztályszint, ami egy osztály összes kezdeti példányára vonatkozik; illetve a héjszint, amely a szülő alkalmazásablakra vonatkozik.
Egy-egy létrehozott esemény legelőször azon az elemen jelentkezik, amelyen az esemény keletkezett, és attól halad lefelé egészen az alkalmazásszintig, kivéve, ha közben valamelyik szinten úgy rendelkezünk, hogy az eseményt nem engedjük tovább.

Mindez a gyakorlatban

Most, amikor az elmélettel már nagyjából tisztában vagyunk, vessünk néhány pillantást számológépünk forráskódjára.
Ha a kódsorokat begépeljük és .PY kiterjesztéssel mentjük, a `python` paranccsal meghívva fárasztó gépelésünk eredményében már gyönyörködhetünk is.
Amennyiben közelebbi pillantást vetünk a programra, látható, hogy a gerincét a ***Calculator*** osztály alkotja, amelyben a lényeg igazából az `__init__()` eljárásba van sűrítve. Azt már tudjuk, hogy ez az az eljárás, amely objektumpéldányunk létrehozásakor önmagától lefut, ennek megfelelően a benne rejlők már az alkalmazás indulásakor végrehajtódnak. Az első ismeretlenbe a 12. sorban ütközünk, itt hozzuk létre alkalmazásunk kijelzőjét, amely valójában egy egyszerű szövegbeviteli mező, és a következő sorban található `pack()` eljárással tesszük ki a képernyőre. A `relief` értékkel a kijelző stílusát adhatjuk meg, ami ebben az esetben SUNKEN, azaz süllyesztett.
A `textvariable` értékkel azt a változót jelöljük ki, amelyet összekötünk a kijelzővel. Amennyiben a változón módosítunk, változik a kijelző tartalma, ami fordítva is igaznak fog bizonyulni.
A 15. sortól kezdődően alkalmazásunk gombjait rajzoljuk ki, a 17. sorban pedig a ***key*** karaktersorozatot bontjuk szét karakterekre, amelyeket kirajzolásuk után egy lambda kifejezésen keresztül a `setscreen()` eljárásra csatolunk vissza, így ha bármelyiket lenyomjuk, az adott érték a kijelzőn jelenik meg.
A 18. sorban meghívott `button()` eljárás csak hivatkozás egy alább bevezetett függvényre, amelyet nem szabad a ***Button*** osztállyal összekevernünk, utóbbi ugyanis egy objektumpéldánnyal térne vissza.
A 16. és 20. sorokban megint csak egy saját függvényen keresztül hozunk létre egy keretet. Mivel a Packer felületkezelőt használjuk, erre azért van szükség, hogy a Packer egyértelműen eldönthesse, hova kell az adott elemet helyeznie.
A 24. és 25. sorokban a `bind()` függvényt hívjuk meg, amellyel az egérgomblenyomás eseményéhez rendelünk hozzá egy `lambda` eljárást. Az eljárás első értéke (`e`) látszólag használaton kívüli, valójában viszont azért szükséges, mert a `bind()` ezen keresztül küldi el az `event` eseményváltozót.
Az 51. sorban az `eval()` függvényen keresztül a kijelző tartalmát kiértékeljük és az eredményt kiíratjuk.

Összegzés

Ebben a részben egy számológépes példán keresztül a Tkinter alapvető lehetőségeivel ismerkedtünk meg, amelyek felhasználásával egyszerű alkalmazások készítésére leszünk képesek. A példaprogram részeit begépelés után ajánlatos módosítgatni vagy akár bővíteni. Így biztosak lehetünk benne, hogy a frissen elsajátított ismereteket nem felejtjük el azonnal, és később is fel tudjuk használni.

Gludovátz Gábor
(ggabor@sopron.hu)
1996 óta foglalkozik Linux-rendszerekkel. Egyik kedvenc időtöltése a programozás, jelenleg éppen egy C++-ban írt KDE-s játékon dolgozik, de szívesen kódol Pythonban és PHP-ben is. Honlapja ➲ http://www.sopron.hu/~ggabor/

Kapcsolódó címek

A Python hivatalos honlapja ➲ http://www.python.org/
Tkinter- tanfolyam
➲ http://www.pythonware.com/library/tkinter/introduction/

Az ablakkezelők és a Linux

Telepítéskor a legtöbb Linux-változat két ablakkezelőt helyez előtérbe: a KDE-t és a Gnome-t. Az összes többi ablakkezelő általában csak lehetőségként választható, amely sajnálatos módon azt eredményezi, hogy a felhasználók többsége – főleg a kezdők – nem, vagy csak kis mértékben próbálkozik megismerkedni a többivel. Néhány éve, amikor Linuxszal kezdtem foglalkozni (RedHat 5.2-sel), ha jól emlékszem, a KDE és a Gnome még egyáltalán nem volt üzembiztosnak nevezhető, ezért nem is választhattam őket. Akkoriban a manapság szinte ismeretlen fvwm ablakkezelő és az AfterStep, valamint a WindowMaker képviselték az elfogadható választást. Azóta több, a témával kapcsolatos fordítást is készítettem a levelezőlistákon megismert sorstársakkal. A legtöbb Linux-változat esetén néhány egyszerűbb ablakkezelő külön is választható. Jelenleg a Debian SID-ben található alapablakkezelőket szeretném bemutatni, terjedelmi és egyéb okokból kifolyólag azonban csak röviden. A cikksorozat első lépésben az igen egyszerű felépítésű, eszközkészlet nélküli ablakkezelők ismertetését veszi célba. Ehhez a Debian SID-változata szolgáltatta az alapot, amely a ***táblázatban*** látható ablakkezelőket tartalmazza.
A szóban forgó ablakkezelőkről, mint említettem, helyenként csak meglehetősen szűkszavúan fogok nyilatkozni, mint látni fogjuk részben azért, mert csekély mozgásteret engednek.

Az alapszintű ablakkezelők

Ebbe a csoportba azokat a grafikus felületeket soroltam, amelyek kinézet, tudás, valamint kezelhetőség tekintetében is igen puritánok.

A twm

A `twm` az egyik legegyszerűbb ablakkezelő, hiszen indítás után csupán egyetlen menüt tudunk megjeleníteni, jó esetben talán egy ikonkezelőt is. Az ablakkezelő igényeinknek megfelelő átformálását csakis valamilyen szövegszerkesztő program segítségével végezhetjük el. A beállítóállományok nálam két helyen találhatók: a */etc/X11/twm/system.twmrc*-ben (ez rendszerszintű fájl, amelyben a menü központilag frissül), valamint a *~/.twmrc* (ez pedig a saját beállításaimat tartalmazó fájl).
A helyi fájl csak a felhasználó számára használható beállításokat tartalmazza. Amennyiben a *~/.twmrc* fájl létezik, a rendszerszintű beállításokat tartalmazó fájl nem kerül feldolgozásra. Következzék néhány fontosabb twmrc-beállítás:

- a fájl elején található sorok a menü és az ablakfejlécek betűkészleteinek típusát tartalmazzák:
  ```
  TitleFont "-adobe-helvetica-bold-r-normal
  ➥--*-140-*-*-*-*-*-*"
  ```
- a fájl tartalmazza a menüket. Fontos, hogy először a menü könyvtárszerkezetét kell meghatároznunk, és csak ezután térhetünk rá az egyszerű menüpontok megadására:
  ```
  "Xcalc" f.exec "xcalc &"
  ```

A *twmrc* fájl felépítéséről további adatokat a `man twm` parancs kiadásával kaphatunk. A `twm` megbízható, gyors, üzembiztos ablakkezelő, amely az eddigiek során számtalan ablakkezelő alapjául szolgált.

9wm

Ha lehet az egyszerűséget tovább fokozni, a `9wm` még az előzőnél is egyszerűbb ablakkezelő. A grafikus felület indítása után a menüt az egér jobb gombjával érhetjük el, és öt elemet tartalmaz.
New – új X-terminálablakot nyit meg.
Reshape – feladata az ablakok átméretezése (az egér jobb gombjával használhatjuk, az első kattintással jelöljük ki az átméretezendő ablakot, a másodikkal adjuk meg az egyik sarkát, majd egy újabb kattintással az adott ablak átlós sarkát).
Move – az ablakok mozgatására szolgál, szintén az egér jobb gombjával használható. Az első kattintás és az egérgomb nyomva tartása az ablak kijelölésére szolgál, amit az elmozgatása után az egérgombot elengedve elhelyezhetünk.
Delete – bezárja az ablakot.
Hide – az ablak elrejtésére szolgál. Az elrejtett ablakok e jobbgombos menü további pontjaiként fognak megjelenni.
Az ablakkezelő bezárása a CTRL+ALT+BACKSPACE billentyűkkel vagy a `9wm exit` parancs X-terminálon történő kiadásával zajlik. Szintén megbízható és üzembiztos ablakkezelő.

aewm

Egyszerű ablakkezelő. Indítás után csak egy X-terminálablak jelenik meg, amelyről több alkalmazást is el tudunk indítani, például az alábbi paranccsal:

```
rxvt &
```

Az `&` jel visszaadja a parancsértelmezőnek a vezérlést, így több parancs is elindítható egy terminálról.
A fő ablakra kattintva a jobb egérgombbal újabb X-terminálok indíthatók el. A középső gombbal az asztalra kattintva egy ablaklistát hívhatunk elő. A megnyitott ablakokat az ablakok fejlécének jobb szélén lévő négyzetre kattintva a bal egérgombbal zárhatjuk be, és a jobb egérgombbal ugyanoda kattintva rejthetjük el.
A bal egérgombbal a fejlécre kattintva előtérbe hozhatjuk az ablakot, a jobbal pedig a háttérbe küldhetjük, a középső egérgomb folyamatos lenyomásával ugyanott mozgatni is tudjuk.
A CTRL+ALT+BACKSPACE billentyűkkel léphetünk ki az `aewm`-ből. Szintén megbízható ablakkezelő.

pwm

A `pwm` a `twm`-hez hasonlóan képes a rendszermenü használatára, és a beállítási lehetőségei is hasonlóak hozzá.
Indításkor lehetőségként megadható a pillanatnyi beállításokat tartalmazó fájl (`-pwm -cfgfile beÆll t fÆjl`).

larswm

Egyszerű felépítésű ablakkezelő, ennek megfelelően a kezelőparancsai is meglehetősen egyértelműek.
Fontos megjegyeznem, hogy mielőtt ebben az ablakkezelőben bárminek nekikezdenénk, másoljuk át a leírásban található *.larswm* fájlt a saját könyvtárunkba, és alaposan tanulmányozzuk át a **.ps.gz* fájlokat (szintén a leírásban lelhetők fel), amelyek az egérhasználatról és a billentyűkombinációkról nyújtanak tájékoztatást.
Amennyiben ezt a „műveletsorozatot” kihagyjuk, igen kevés