Just Xdo It!

Írta: critter

Néhány hónapja a gépem meghalt és nem engedhettem meg magamnak a javíttatását: más, sokkal sürgősebb dolgokra kellett a pénz, emlékeztetett a feleségem. Ott álltam egy ötéves, ugyan teljesen használható Lenovo laptoppal, ami hajlamos volt a melegedésre. KDE-t használtam, de most valami kicsit könnyebb, valami kevésbé forrásigényes és a szegény laptopomat nem nagyon terhelő kellett.

Kipróbálván az összes, leginkább támogatott asztali környezetet és ablakkezelőt; a PCLinuxOS által kínált 64 bites Mate mellett állapodtam meg. Legyalultam a merevlemezt, eltávolítottam a Windowst és a helyreállító partíciót (ezt a gépet használtam, amikor dolgoztam és most szabadon választhattam).

Ezt betöltve kiderült, hogy egy teljes operációs rendszerem van, ami valamivel több mint 200 MB RAM-ot használ és mindkét processzor 1 és 2% között fut, fél sebességen és a ventilátor még csak nem is fut – kiváló!

1. De, kell legyen egy „de”, a Mate nem rendelkezik a KDE összes tulajdonságával, amik közül néhányat nagyon hasznosnak találtam, ezért kezdtem a neten megoldást keresni. Keresgélésem során beleakadtam egy nagyon nagy tudású ember, Eric Zhiqiang MA cikkébe, ami egy Xdotool nevű eszközt használt (a PCLinuxOS tárolójában megtalálható) a Gnome terminál megjelenítésére egy adott billentyűparancs lenyomása esetén. A guake a KDE yakuake-jéhez hasonlóan működik, de ez a szkript lehetővé tette, hogy a mate-terminal-omat használjam, ami nem visz magával további függőségeket. Természetesen, bármilyen terminálemulátor alkalmazást választhattam volna.

A továbbiak arról szólnak, hogyan használtam ezt az eszközt Mate asztali környezetben, de ennek működnie kell majdnem minden asztalnál (talán az enlightement-tel lehet gond, de nem tudom, nem próbáltam ki). Az Xdotool-t telepítettem, majd letöltöttem a szkriptet és módosítottam a

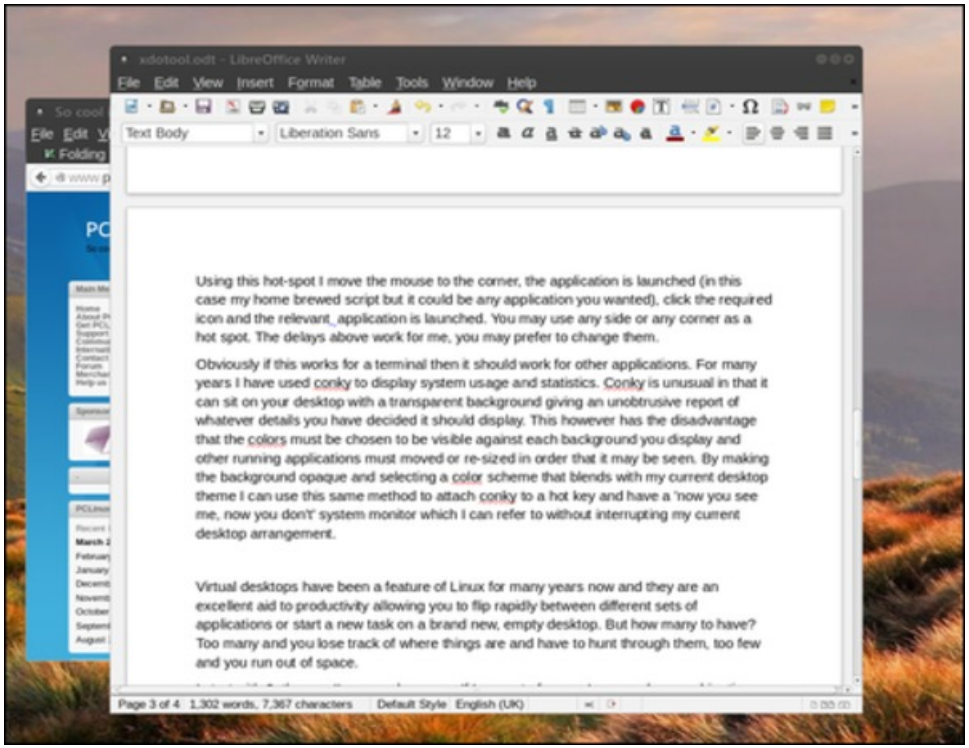
```
gnome-terminal --maximize &
```

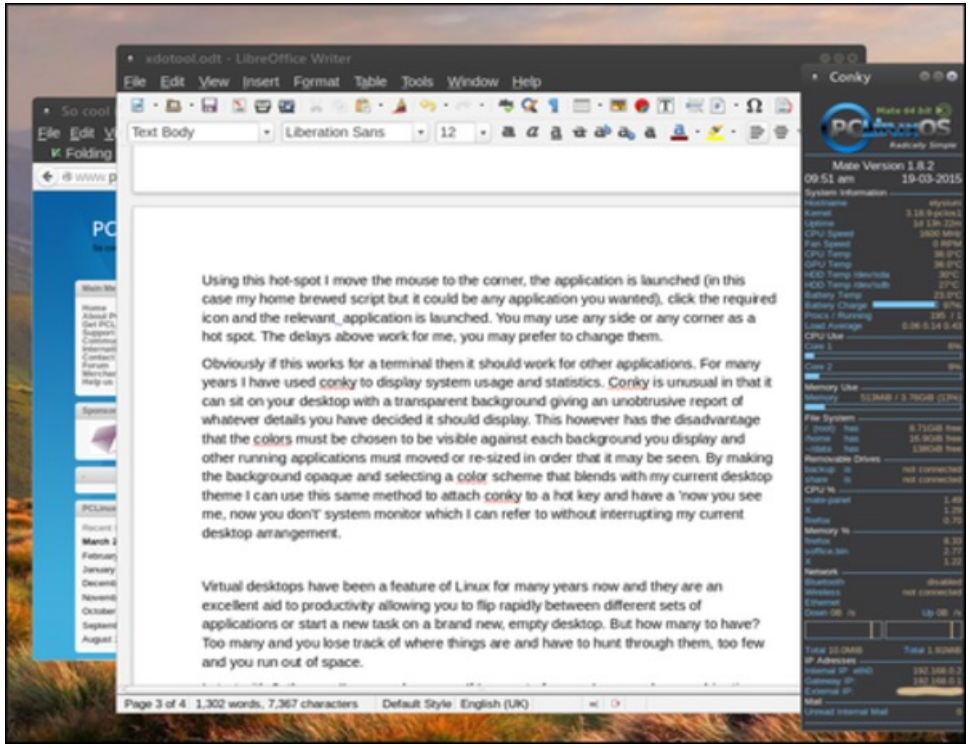
sort

```
mate-terminal --geometry 120x36+180+100 & -ra.
```

Úgy állítottam be, hogy F12 lenyomására fusson (System → Preferences → Keyboard Shortcuts). Hibátlanul működik. A működését a fent hivatkozott cikkben olvasható.

Természetes, ha ez működik terminállal, akkor más alkalmazással is. Sok éven át conky-val figyeltem és jelenítettem meg a rendszer használatát és statisztikáit. A conky attól szokatlan, hogy átlátszó háttérrel ülhet a képernyődön és diszkréten jelenti az általad megjeleníteni kiválasztott bármilyen részletet. Ugyanakkor hátránya, hogy a színeket úgy kell megválasztani, hogy az általad megjelenített valamennyi képernyőn látható legyen, illetve a többi futó alkalmazást el kell mozgatni, vagy át kell méretezni láthatósága érdekében. A hátteret áttetszővé téve és az aktuális asztaltémámhoz igazított szín-összeállítással ugyanezen módszert alkalmazhatom és a conky-hoz rendelhetek billentyűutasítást, amivel lesz egy „most látsz, most nem látsz” rendszerfigyelőm, amit az aktuális képernyő elrendezésem megzavarása nélkül előhívhatok, illetve mindig tökéletesen illeszkedik, bármilyen háttérképet is használnék.



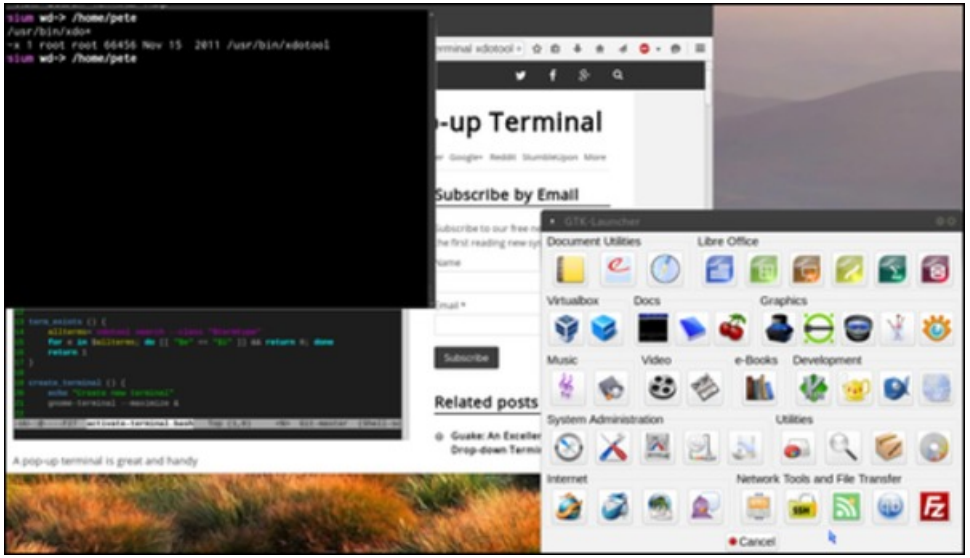


Nos, mi egyébre képes még az xdotool eszköz? Történetesen elég sokra. Billentyűleütéseket tud küldeni alkalmazásoknak, veszi és átalakítja a billentyűleütéseket, mielőtt elküldené. Képes írott üzeneteket bármely elérhető ablakra kiadni, még az egyes betűk megjelenítésének késleltetésével is, mintha valós időbe történne a gépelés (vagy, ha emlékeztek, régi géptávírók módján). Az egérmutatót bárhová el tudja mozgatni, bármely alkalmazásra, vagy bármilyen képernyőre abszolút, relatív, vagy akár polárkoordináták szerint. Az egérpozíció visszakeresheto, kattintás szimulálható és műveletek kapcsolhatók hozzá, elérve, vagy elhagyva egy adott ablakot vagy a kurzorral a képernyő szélt, vagy sarkot érve. Kiolvashatja egy ablak nevét, adatait és állapotát. Mozgathat, átméretezhet, fókuszálhat, mutathat, elrejthet, vagy kilőhet ablakot. Más munkaterületre is pozicionálhatsz, ablakot küldhetsz oda és létrehozhatsz, eltávolíthatsz munkaterületeket. Esemény bekövetkezésekor parancsot, vagy szkriptet futtathat. A parancsok tárolhatók és akár átirányíthatóak az xdotool-hoz,vagy bash-stílusban közvetlenül végrehajthatóak, amennyiben egy bash-fejlécet rakunk az első sorába:

```
#!/usr/bin/xdotool
```

Megjegyzés: csak akkor jó, ha a szkriptben csak xdotool parancsok vannak.

Vagyis, ha a náthát nem is gyógyítja, nagyon hasznos eszköznek tűnik, érdemes megismerni. Az érzékeny képernyőszéleket és -sarkokat a KDE-ből hiányoltam, de az xdotool tudja ezt és még egy lépéssel tovább is megy. Van egy kis, saját készítésű launcher.sh nevű alkalmazásom, amivel gyorsan választhatok az általam leggyakrabban használt alkalmazások közül.



Gondoltam, jó lenne, ha állandóan elérhetném egy adott pont egérkurzorral való aktiválásával. Ennek érdekében beállítottam egy xdotool parancsot induláskori futásra:

System → Preferences → Startup Applications

```
xdotool behave_screen_edge --delay 800 --quiesce 1500
bottom-right exec /bin/launcher.sh
```

A fentiek egyetlen sorba kellenek és a Startup Applications dialog 'command' négyzetbe kell beírni. Ez ezt jelenti:

xdotool az eszköz meghívása.

behave_screen_edge nézi, hogy az egérmutató képernyőszélet, vagy sarkot ér-e és a következő műveletet hajtja végre.

- - delay 800 800 milliszekundumot vár indítás előtt, a véletlen aktiválás elkerülésére.

- - quiesce 1500 további 1500 ms-ig ne hajtson végre más műveletet, elkerülendő a többpéldányos indítást, mielőtt az egeret elvinnéd az érzékeny területről.

bottom-right a figyelt képernyőterület.

exec művelet végrehajtása a billentyű lenyomására.

/bin/launcher.sh az amit végre kell hajtani.

Ezt a területet használva, amikor az egeret a jobb alsó sarokra húzom, az alkalmazás elindul (ez esetben a magam kreálta szkript, de bármilyen szükséges alkalmazás lehet), majd a megfelelő ikonra kattintva az adott alkalmazás elindul. Ez sokkal gyorsabb és kényelmesebb, mint mindenféle menüszintek között keveregni. Bármely oldalt, vagy sarkot kijelölheted érzékeny pontnak: bal, jobb, fent, lent, bal fent, jobb fent, bal lent, vagy jobb lent, ahogy én használtam itt. A késleltetés nálam bevált, de te megváltoztathatod azokat.

Virtuális asztalok és munkaterületek már évek óta jelen vannak a Linux-ban és a munkát kitűnő támogató kiegészítő eszközök, gyors váltást téve lehetővé különféle alkalmazás-összeállítások között, vagy új feladat indítását teljesen új asztalon. Ám mennyi asztalod kell legyen? Túl sok és nem tudod követni, hogy hol vannak a dolgaid és hogyan találd meg azokat; túl kevés és kifogysz a területekből.

Én kettővel kezdeném, azzal amin vagyok és a tartalékkal. Ha elfogy a helyem nyomok egy billentyűkombinációt és az xdotool készít nekem egyet. Ha eltávolítanám a kiegészítő asztalt, egy másik kombináció szól az xdotool-nak, hogy töröljön, egyszerre csak egyet. Bármilyen nyitott alkalmazás lenne a törölt asztalon, nem veszik el, a megmaradt asztalra átkerül.

A kód, ami ezt végrehajtja:

```
xdotool set_num_desktops $(( $(xdotool get_num_desktops) + 1 ))
```

és

```
xdotool set_num_desktops $(( $(xdotool get_num_desktops) - 1 ))
```

Minden alkalommal ezt bepötyögni természetesen nem jó, így én a kódot két szkript formájában tárolom egyszerűen egy bash-szkript fejlécet adva az egyes fájlok első sorába. Például:

```
#!/bin/bash
xdotool set_num_desktops $(( $(xdotool get_num_desktops) + 1 ))
```

A szkripteket mentettem addwkspc és rmvwkspc néven és végrehajthatóvá tettem ezzel:

```
chmod +x addwkspc
chmod +x rmvwkspc
```

A #!/bin/bash fejléc kell, sokkal inkább, mint a korábban említett xdotool fejléc, mivel az xdotool tartalmaz néhány bash jellemzőt (számtani kiterjesztés és alhéj), de a finomabb részletek megértése nélkül is használhatod a szkripteket.

Ezután az egyes szkriptekhez rendelhetsz gyorsbillentyűt (nálam CTRL+F7 és CTRL+F8), gondoskodván a teljes útvonal hozzárendeléséről, talán így /home/én/addwkspc, vagy hasonló.

Az xdotool lehetővé teszi a mozgást a különféle asztalok között, bármilyen módon, amit csak kitalálsz. Mivel a Mate-ban már biztosított az asztalok közötti előre- és hátralépés, szükségtelennek találtam ennek az eszköznek a használatát. Úgy érzem, inkább csak rontana, bonyolítaná a dolgokat. Ugyanakkor az opciók rendelkezésre állnak, ha szükséged lenne rá.

Az xdotool egyik nagy erőssége az ablakok manipulálása, ami lehetővé tette, hogy némi KDE-s funkcionalitással ruházzam fel a Mate-t. A KDE-ben képes voltam adott ablakot, vagy alkalmazást egy meghatározott asztal egy meghatározott helyén és méretben indítani, ami által mindig tudtam, hogy hol található. Az xdotool segítségével ez a Mate-ban is meg tudom csinálni, jóllehet egy kicsit több erőfeszítést igényel ennek a beállítása. Azonban ezt csak egyszer kell megcsinálni.

Ablakon műveletet végezni akkor tudsz csak, ha tudsz arról valamit: az ablak azonosítóját (id), az ablak nevét (ami a címsorban jelenik meg), az ablak osztályát, vagy az ablakosztály nevét például. Ezeket használja az ablakkezelőd az egyes ablakok hivatkozásaként. Az xdotool eszköz prezentálhatja neked ezeket az adatokat, de én egy másik, xprop nevű eszközt használok erre, ami ha nem lenne telepítve, a PCLinuxOS tárolóiban megtalálható.

Ha szeretném, hogy a Mate szövegszerkesztője a pluma mindig egy adott helyen és méretben nyíljon meg, akkor tudnom kell, hogyan hivatkozzak arra az ablakra. Ablaknak id-t az ablakkezelő ad és nem állandó jelleggel. Amikor először nyílik meg a pluma, az ablaknév „Unsaved Document 1 – pluma” lesz, és ha bezárom az üres dokumentumot, akkor simán „pluma”-ra vált, de ha nyitok, vagy mentek egy dokumentumot, átvált az adott dokumentum nevére. Azonban ablakosztály ugyanaz marad és ennek azonosítására pluma-t nyitok és futtatom xprop parancsot, átcsövezve grep -en, hogy csak a nekem szükséges információ jelenjen meg, mivel az xprop elég bőbeszédű:

```
xprop | grep CLASS
```

A parancs vár, hogy rákattintsak annak az ablaknak belsejére, aminek a tulajdonságai kellenek, majd ezt adja vissza:

WM_CLASS(STRING) = "pluma", "Pluma"

A pluma ablakára akár a pluma-, akár Pluma-ként hivatkozhatok, mindkettő működik.

Alternatívaként az osztály felülírható a pluma GTK+ - -class opcióval tröténő meghívásával.

Az ablak mozgatásához és átméretezéséhez szólnom kell az xdotool-nak, hogy keressen egy pluma osztályt, majd azon hajtsa végre a műveletet. A bal felső sarokba mozgatás (0 0) és az ablak 700 x 500-ra átméretezés kódja a következő (mindent egy sorba kell írni):

```
xdotool search --class Pluma windowsize %@ 700 500
windowmove %@ 0 0
```

Ugyanakkor ez csak akkor működik, ha már fut a program. Az, hogy elindítsuk az alkalmazást és ezután az xdotool parancsot, némi plusz munka szükséges. Az alkalmazást bash-szkripttel indítsd, amit egy követő & segítségével helyezd a háttérbe. Ezután add ki a disown parancsot, ami a bash-szkript és az alkalmazás közötti függőséget megszünteti és lehetővé teszi mindkettőnek az önálló futás folytatását. Végül add ki az xdotool - -sync opciós utasítást az xdotool keresés parancsnak. Az opció közli az xdotool-la, hogy várjon addig, amíg az alkalmazás ablaka láthatóvá nem válik, mielőtt a méretezési és mozgatási parancsot végrehajtaná. Íme a szkript:

```
#!/bin/bash
/usr/bin/pluma &
disown
/usr/bin/xdotool search --sync --class Pluma windowsize %@
700 500 windowmove %@ 0 0
```

Az utolsó egyetlen sorba kell kerüljön. A szkript nem a legtisztábban éri el az eredményt, mivel rövid ideig megjeleníti az alkalmazást, mielőtt átméretezné, de ez kis ár, amit az egyszerűség megtartásáért fizetünk.

Mivel az ablakkezelőd sokféleképpen hivatkozhat egy alkalmazásra, kell a %@ ahhoz, hogy biztosítsuk, az xdotool az általad szükségeset éri el.

Ez kicsit zavaros lehet, ezért magyarázatul az xdotool saját dokumentációjának egy példájára hivatkozom. Alapból a pluma-nak csak egy példánya jelenhet meg. Ha egy újabb pluma példányt akarsz indítani, az egy új fület nyit. Más alkalmazások lehetővé teszik, hogy teleszemeteld a képernyődet tetszőleges számú példánnyal. A mate-terminal egy ilyen alkalmazás. Az egyes példányokra az ablak id-k listájában egy pozícióval hivatkozik, az ablakkezelő így tudja azonosítani az egyes ablakokat. Az első ablak hivatkozása a mate-terminal ablakosztályban %1, a másodiké %2 és így tovább, a %@ mindegyikre hivatkozik. Hasonlóképp, a szkripteknek átadott hivatkozás paraméter is $1, $2 … . A szkript (fourterms-nek neveztem el) csak xdotool parancsokat tartalmaz, így bash fejléc helyett xdotool fejléccel kezdődik. Sokkal hatékonyabb, mivel nem kell bash hozzá.

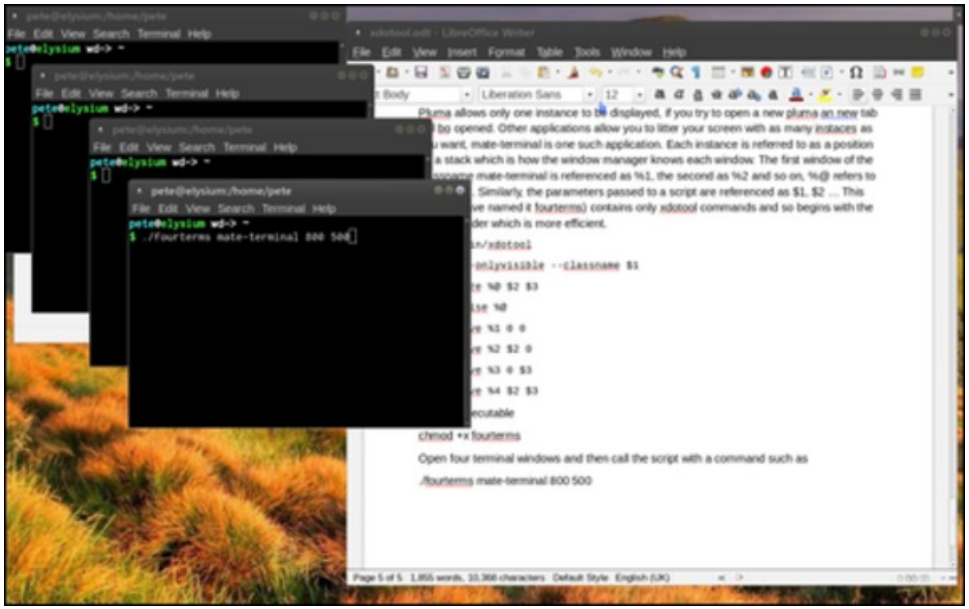
```
#!/usr/bin/xdotool
search --onlyvisible --classname $1
windowsize %@ $2 $3
windowmove %1 0 0
windowmove %2 $2 0
windowmove %3 0 $3
windowmove %4 $2 $3
```

Tedd végrehajthatóvá.

```
chmod +x fourterms
```

Nyiss négy terminál-ablakot és hívd meg a szkriptet ilyesmi paranccsal

```
./fourterms mate-terminal 800 500
```



Csak a látható és 'mate-terminal' (a szkriptnek átadott első paraméter) osztálynevű ablakokra hajtja végre.

A sor, ami azt modja, hogy

```
windowmove %4 $2 $3
```

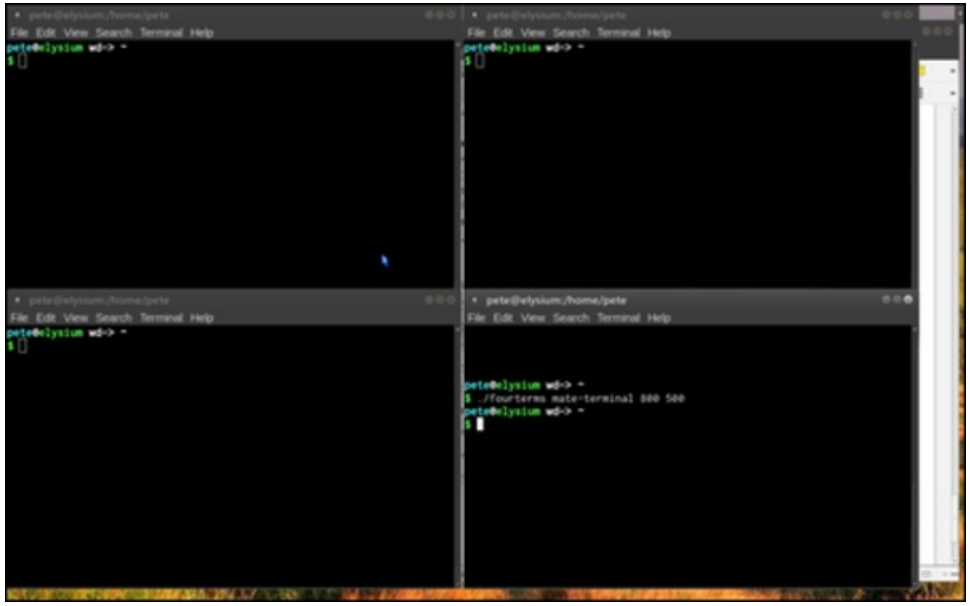
mozgatja a negyedik ablakot (%4).

A $2 elmozgatása (második paraméter – 800) x irányába és a $3 elmozdítása (a harmadik paraméter – 500) y irányába. A 0 0 érték jelenti a képernyő bal felső sarkát.

A terminálok egyidejű mozgatásához a desktop 3-ra (legyen desktop 3 és azt is tudd, hogy az ablakok számozása 0-tól indul, tehát ez a negyedik asztalt jelenti) a következő sorokat add hozzá a szkripthez.

```
set_desktop_for_window %1 3
set_desktop_for_window %2 3
set_desktop_for_window %3 3
set_desktop_for_window %4 3
```

Végül kapsz egy szép terminál-láncot azon az asztalon.



Ha a feladatok végrehajtásához egy adott csokor alkalmazást gyakran használsz több asztalon, akkor az xdotool beállítható úgy, hogy egyetlen billentyűkombinációra végrehajtsa azok megnyitását, átméretezését és mozgatását csakúgy, mint a megfelelő ablakszám létrehozását.

Néha szükség van arra, hogy valamit egy akkor kiválasztott ablakkal végezz és nem egy adott, előreválasztott ablakkal. Esetleg szükség lehet egy olyan billentyűutasításra, hogy időszakosan eltávolítsd egy az asztalt zavaró ablakot, amivel még nem végeztél. Mondjuk mozgasd át a negyedik asztalra (3-as számú), ahová később átléphetsz és tovább használhatod. Ez esetben nem tudod az ablak ID-jét, osztályát, vagy bármi hasonlót előre és erre van az xdotool selectwindow parancsa. Amikor az xdotool megkapja ezt a parancsot tudja, hogy várnia kell, amíg rá nem kattintasz egy ablakra, majd végrehajtja a további parancsot azon az ablakon.

```
xdotool selectwindow set_desktop_for_window 3
```

Hozzáadva ezt a sort egy billentyűparancshoz pl. wingomb+4, az elküldi a negyedik asztalra azt az ablakot, amire legközelebb rákattintasz.

Gyakran van, hogy valahol nyitva van a Firefox és hirtelen el kell érnem egy weblapot. A következő kódsor megkeresi a Firefoxot, átléptet arra az asztalra, ahová raktam, ráviszi a fókuszt (aktivvá teszi), a kurzort a címsorra viszi (crtl+L) és törli az abban található szöveget (BackSpace) lehetővé téve, hogy azonnal elkezdhessem a kívánt weblap címét gépelni.

```
xdotool search "Mozilla Firefox" windowactivate -sync key
ctrl+l key BackSpace
```

Az egésznek egy sorban kell lennie és hozzákapcsolható egy billentyűkombinációhoz.

Nagyjából ennyit az xdotool alapjairól. A kézikönyvoldal leírja az összes elérhető parancsot és opciót, és ennek a néhány példának a segítségével képes vagy a téged érdeklőket használni. Az, hogy mennyit profitálsz ebből az eszközből, azt saját igényeid határozzák meg és nem a komplex programozási képességed. Nekem jó munkát végzett abban, hogy áthidaljam a nagy képességű ám nagy igényű KDE környezet és a sima, egyszerű Mate-gép közötti szakadékot.