Remote TouchPad

Osnovna ideja

Osnovna ideja aplikacije je da se premik na ekranu prevede v ustrezni znak, potem ga z enostavno *client-server* vezo pošlje na računalnik. Server sprejeme znak, in ga pošlje na napravo */dev/mis* ki je dejansko *fake mouse*. Dalje ga ko zapišemo znak na našo "napravo", ga naš usterzni driver mis pogleda kateri znak je bil zapisan, I zaradi tega premakne kursor ali naredi kaj je potrebno.

Opomba: zaradi client-server veze čez WiFi miška včasih šteka, torej treba spremenit povezavo med Androidom in računalnikom. Jedro vse spreje znake obdela brez težav, kar je pričakovano.

Android del aplikacije

Naloga tega dela je da pogleda kako smo premikali prst po zaslonu, in pošlje ustrezen znak. Zaslon (full screen) je razdeljen na tri dele, dva gumba in en prostor.

V funkciji *protected void onCreate(Bundle savedInstanceState)* damo dva *OnClickListener-ja* na levi in desni gumb, ki pošljejo respektivno *'l'* in *'r'* na Server.

Fukncija *public boolean onTouchEvent(MotionEvent event)* poskrbi za različne premike. Prvič preveri *double touch,* ko so je sprejet *double touch* pogledamo v kateri smeri smo premaknuli oba prsta, ter pošljemo znak *(w,a,s,d). Za* normalen premik z enim prstom, pa imamo več možnosti. Prvo pregeldamo ali je narjen veliki premik in takrat pošljemo *W,A,S,D.* Dalje preverimo normalno smer premika in pošljemo *(1,2,3,4,5,6,7,8,9)* inspirisano z tipkovnico.

Server

Naloga servera je samo da sprejeme znak in ga dalje pošlje na */dev/mis.*

mis.c

Funkcije *static int __init mis_init(void) ,* naredi alociranje spomina za napravo, registracijo gonilnika. Preverimo ali smo dali naše fiknso major število in registrira spomin, drugače alocira spomin za nekatere prosto major število. V oba primera minor je 0. Dodamo napravo ter joj dodamo *fops.* Potem alociramo *mis_input_dev* postavimo "opisne" vrednosti te strukture in še z uporabo fukncije *set_bit* spremenimo vrednosti v registrima v strukturi *mis_input_dev.*

static void mis_exit(void) ja standardna, kjer samo počistimo spomin ko končamo.

Fukncija ki naredi največ dela je funkcija *static int __init mis_init(void)* . Prvič znak zapisan na napravo, kopiramo v *write_buffer* s pomočjo fuknicije *copy_from_user* nato pa še kopiramo iz *write_buffer*-ja v neko spremeljivko *m.* Nato funkcija *switch* spremeni vrednosti v posameznem registru glede na to kateri znak je zapisan. Ko pridemo v nek *case* prvič z uporamo *input_report_rel* enostavno spremenimo vrednost na ustreznem bitu. Vrednost se spremeni relativno v odnosu na pozicijo prej u neki smeri. Nato še s uporabo *input_sync()* povemo napravi da smo končali celoten prenos.