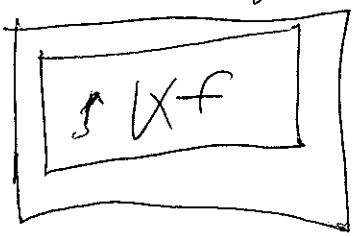
Loading sequence:

goto page; $\boxed{\boxed{\text{sXf}}}$ user_id, room_id
con_id, token,



con_id - cccaf replay mode → rep, ~~[illegible]~~,
req_url_API

~~[illegible]~~ if (replay mode) {

user_id, room_id - ~~[illegible]~~ from

flash vars - needed for authorisation;

After ... ~~[illegible]~~ get replay by id

old get from replay().

} else { // not replay

get modDATA

get userDATA

connectFMS();

prepareModules();

run(); already

Tutorion C:

Board: paint; Load presentation & paint

Save ...

Edit existed ...

Chat

~~Video~~ camera

Main panel

User list:

Hand up icon

Typing icon

Load presentations: Load & preview

Translator

Recorder

1. Allocation

struct libusb_transfer {
*handle, ..., to, status, al, ll,
callback, *userdata, *buf, ... }

libusb_alloc_transfer (∅) // returns *tr or NULL
libusb_free_transfer (*tr)

2. Filling

libusb_fill_control_transfer (*tr, *dev, *buf, callback, *ud, to)

Только для control transfer:

(!) Перед вызовом надо сделать *buf.
Первые 8 байт в *buf – это Setup packet. (для control transfer)
Для заполнения вызвать.
libusb_fill_control_setup (*buf, bmRqType, bRequest, wValue, wIndex, wLength)
Заполнять, как в синхронном вызове.
При операции host → device нужно еще засунуть в *buf (после 8-го байта) отсылаемые данные.
Размер этих данных → в wLength.

3. Submission

libusb_submit_transfer (*tr)

4. Completion handling

a). Success
b). TO
c). Err
d). transfer is cancelled: libusb_cancel_transfer (*tr)

Replay DATA

Structure:

operations: array

name: string

version SVF: (if (!=) {msg("press to convert")

user name: ~~[illegible]~~ — it is user id

roomname: ~~[illegible]~~ id room

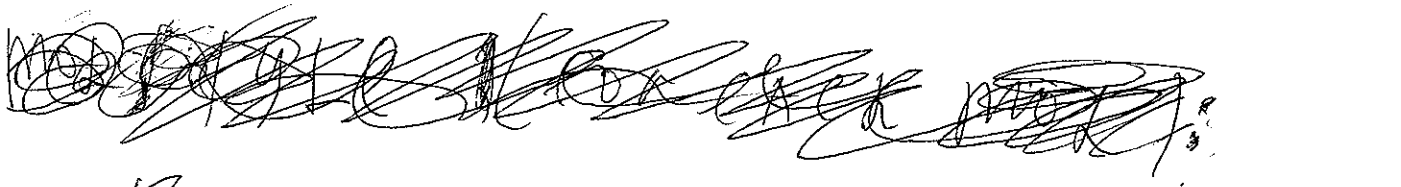
mod XML: STATIC about Layouts

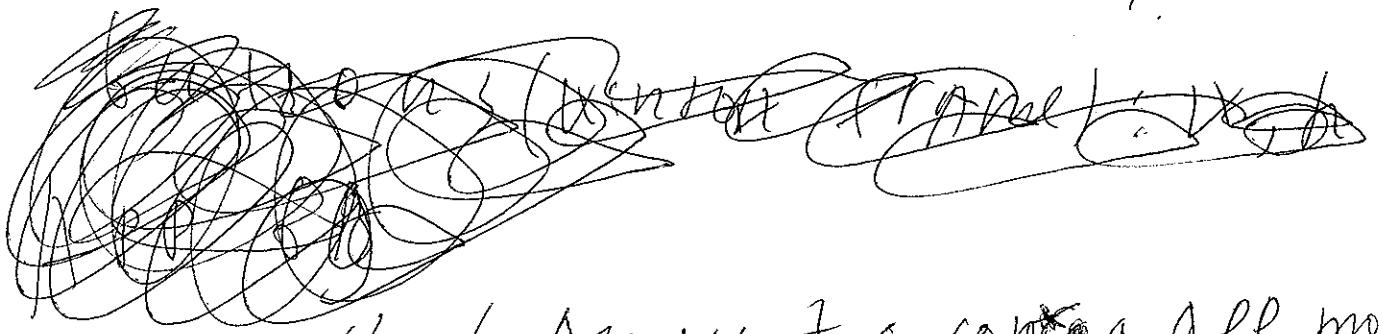
(needed for init)

~~[illegible]~~ Also localisation (TEXT)

~~[illegible]~~

~~[illegible]~~



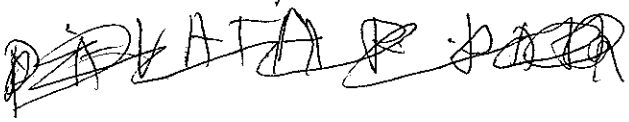


// Layout: config app modules

and their position

user XML: needed for init

~~[illegible]~~



Window H & H: (when recorded)

end.

Details:

Operations:

time: int

window ID: int

Code: string (type for operation name)

Left: *