# Quick Start Guide – Mobile Phones

2023-02-20-A
STRIKLY CONFIDENTIAL

If you want to start within 5 Minutes, then skip the following pages and start reading at “Making the Mobile Phones Running in 5 Minutes” or at “Making the Camera and Pointers Running in 5 Minutes”.

Introduction:

We have developed a small camera and pointers which you can see on our homepage (www.ilstec.com). Up to 80 pointers can simultaneously point and click onto a projection screen and the camera reports X and Y, the state of the 2 buttons and 2 flags, if the pointers are on the screen and if the button state is valid. The pointers can receive commands like different vibration sequences. These commands are necessary as user feedback to clicks the user has made. Although we talk about only 1 camera in this document, we support 16 cameras and therefore 16 screens simultaneously.

In the moment we have all this working on Windows 10.

We decided to communicate with a game or LibreOffice Impress or any other APP which should be controlled using our pointers using a shared memory. A shared memory is a block of physical memory which is mapped into the address space of each participating process. Mapped means, that all processes access the same physical memory bytes, 16-bit words, 32-bit words or 64-bit words. But since they all use the memory simultaneously there must be a strict definition which process is allowed to write which memory address at which time while reading for multiple processes is non critical. For easier handling, we have defined all memory variables as 32-bit DWORDs (UInt32, uint32_t) because they are written and read on the Intel/AMD processors in a so called atomic way (even on 64-bit processors). This way of interchanging data is very fast, writing the data of all 80 pointers into the FIFO needs much less than 1 µsec, reading needs the same time.

The advantage for the game / Impress / AnyApp:

They do never have to wait for any hardware or even worse, they do not have to talk to hardware drivers or handle USB packets etc. They can access the FIFOs at any time they like and it is pretty fast!

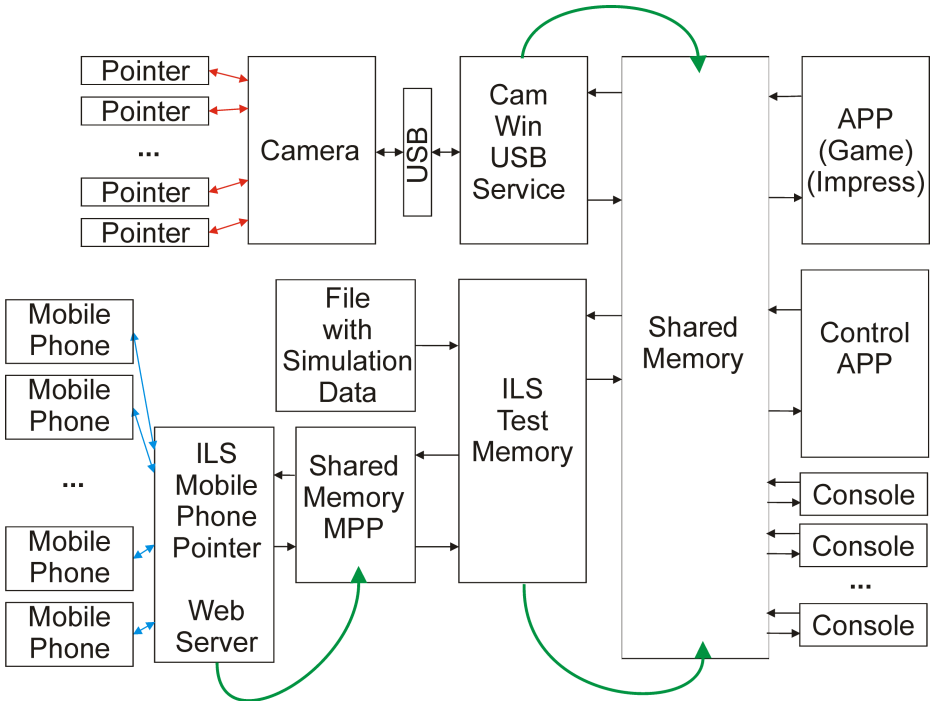
To be able to use our system also without having our pointers or camera available, we also developed a mobile phone interface where the mobile phones act like our pointers.

# Quick Start Guide – Mobile Phones

2023-02-20-A
STRIKLY CONFIDENTIAL

Both, pointers and mobile phones allow the detection of X and Y and the state of the 2 buttons. And both allow reception and execution of commands like short, normal and long vibration and double short, double normal and double long vibration. These are single commands, the game / Impress / AnyApp do not have to care about the sequence and execution times. This is needed as a user feedback (Gamification).

System Overview (the picture only shows 1 camera, but 16 are supported):




The pointers send out short invisible light flashes to the screen which the camera detects (X/Y). Within the light beam also the button states are encoded and in a special case also the pointer unique serial number. The camera can also send commands to the pointers using an RF transmission (2.4GHz). The camera is connected via USB to the PC.

Within the PC our program "**`CamWinUSBService.EXE`**" is running. It picks-up data from the camera and also sends data to the camera. To be able to send data to the game / Impress / AnyApp and receive commands from the game / Impress / AnyApp, it creates a shared memory block. In this memory are several FIFOs (First-In First Out Circular Buffers) which allow real-time multi threaded multi core data communication in both directions.

On the other side of the shared memory is the game / Impress / AnyApp which opens the already created shared memory. Thus, "**CamWinUSBService.EXE**" must run before the game / Impress / AnyApp is started, best at boot time.

To be able to develop such applications it is necessary to have an alternative if our camera and pointers are not available at the place of creating the contents.

So we have developed a ultra fast small webserver "**ILSMobilePhonePointer.EXE**" to which mobile phones connect using W-LAN. We have tested mobile phones running Android and the chrome browser. We did not test iPhones yet!

The webserver creates a small private shared memory. "**ILSMemoryTest.EXE**" (a program which we developed for sending simulation data into a game) opens the small private shared memory created by "**ILSMobilePhonePointer.EXE**" and then it also creates a shared memory block like our program "**CamWinUSBService.EXE**". So, the webserver "**ILSMobilePhonePointer.EXE**" must be started before "**ILSMemoryTest.EXE**"!

!!!!!
It is therefore not allowed that
"**CamWinUSBService.EXE**" and "**ILSMemoryTest.EXE**" run simultaneously
!!!!!

We intend to change "**CamWinUSBService.EXE**" to also support the small private shared memory so that pointers and mobile phones can be used simultaneously.

The green lines in the above picture show which process creates which shared memory!

There exist 2 special FIFO structures identical to the 2 which transfer X and Y etc. and the commands and these are used for the "**MasterPointer**" or "**MasterPhone**". "**CamWinUSBService.EXE**" and "**ILSMemoryTest.EXE**" know the master pointers and all the X and Y and button states and flags are copied into this additional FIFO.

The reason is, that a FIFO may be read only by a single process because once an item is read from the FIFO it is no more available there.

# Quick Start Guide – Mobile Phones

2023-02-20-A
STRIKLY CONFIDENTIAL

We want to have the ability to have a “Control App” which e.g. selects between games, might control the Windows cursor (allowing to control normal Windows applications and Windows itself using the master pointer) but if this “Control App” reads the normal FIFO then the data is no more available for the game / Impress / AnyApp and the other way round because we wanted the master pointer to also participate in the game / Impress / AnyApp.

Additionally there are some FIFOs to which special console apps might connect and we use them for logging and there are some managing FIFOs for data interchange which new pointers arrived, for authentication and licensing purpose which are out of the scope of this document.

Usage of the Mobile Phones:

On the mobile phones you simply start Chrome (no APP needs to be installed!) and open the start page of our webserver and then you can move your finger in the blue areas (best is to keep it on the touch) and with a finger of the other hand you click or hold down the green (it is the pointer’s lower button and the Windows left mouse button) or the red (pointer top/ Windows right) button.

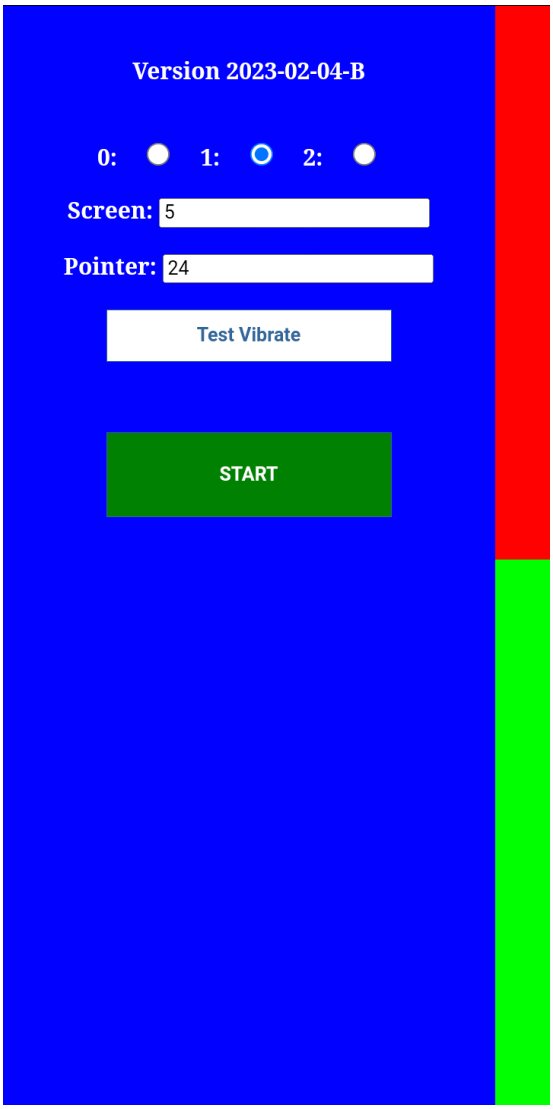
The blue area is an absolute pointing device not like a mouse, a trackball or a mouse pad which are relative pointing devices. So the blue area directly corresponds to X and Y that are reported to the game / Impress / AnyApp. Since all mobile phones have different resolutions, we transfer normalized co-ordinates always ranging from 0 to 65535 (2^16), exactly the same way as our camera does.

The white bar in the moment is red colored and will be replaced to a terminate image to allow the mobile phones to return from full screen mode. Text is not allowed on this display because of there is some text, then the browser may show a copy/paste dialog if the finger is put onto this text.

# Quick Start Guide – Mobile Phones

2023-02-20-A
STRIKLY CONFIDENTIAL

When the mobile phone is started, it shows a start screen. The following picture shows the old one dated 2023-01-25).




The upper line shows the current version.

Below that, the user can select the layout of the screen where in the moment only layout 0 and 1 are supported. We will add new ones later. Layout 0 has the buttons on the left side, layout 1 has the buttons on the right side.

The next field allows to select the screen with which to interact. With our pointers, the camera in front of each screen (0-15) know the screen they are looking to and report the screen number. But with mobile phones, we do not know in front of which screen the user is.

Then the pointer number 0-79 must be entered.

The button “Test Vibrate” immediately generates a 500msec vibration and writes the Morse sequence SOS into the vibration FIFO.

And the button “START” does what it says.

What happens when “START” is clicked:

The current selection and values are written to a phone internal non volatile memory so there is no need to enter them again.

The display turns to full screen locked landscape mode. Locked means that the display no more changes back to portrait if you turn the phone. So, the display then looks like in the previous 2 pictures.

Then an internal 50msec timer is started which was blocked before. The timer has several tasks but 2 of them are:

1. Report the position of the finger in the blue area as normalized X and Y values and the 2 button states as an HTTP Get Request to our mobile phone server and receive the response of the mobile phone server and write commands if there are any into the command FIFO.
2. Read and execute vibration commands from the command FIFO if the previous vibration commands have finished.

If the user pressed “Test Vibrate” before pressing “START”, then the SOS sequence is executed because there are 9 commands in the command FIFO:

SHORT - SHORT - SHORT - LONG - LONG - LONG - SHORT - SHORT - SHORT

The timer sends a get request every 50msec to our webserver which feeds the data into the small private shared memory and “**`ILSMemoryTest.EXE`**” forwards the data to the shared memory from which the game / Impress / AnyApp reads the data.

If the game / Impress / AnyApp has send commands into the shared memory, then “**`ILSMemoryTest.EXE`**” forwards the data to the small private shared memory and the webserver “**`ILSMobilePhonePointer.EXE`**” reads the FIFO and if a command for one or more mobile phones is there then it adds these commands to the response of the get request towards the mobile phones.

The mobile phones can do nearly everything our pointers can do:

1. They report X and Y and the button states and the valid flags
2. They are able to receive vibration commands from the application as user feedback

# Quick Start Guide – Mobile Phones

2023-02-20-A
STRIKLY CONFIDENTIAL

We will add functionality to the webserver and our PointerNet:

1. A QR code for accessing the access point
2. A QR code for opening the start page of the webserver
3. A QR code for opening the start page of the webserver with predefined screen
   This QR code can be placed next to the screen thus avoiding entering a screen number
4. A QR code for opening the login page of our content creating partners or our PointerNet:
   This page may take browser based user pictures or user avatars
   Associate a mobile phone number with the user account
   Associate a specific pointer with their internal unique serial number with the user account
   etc.
5. A web API which allows our content creating partners to set specific values which will login new pointers and mobile phones to the game and screen

For the mobile phones there must be added some features:

The start page which is sent by our webserver must (!!!) contain an encrypted session ID generated by our PointerNet and each further GET REQUEST of the mobile phone must be preceded by a further always changing (derived from the previous session ID) session ID.

Otherwise anybody knowing the servers IP address can login into the game and even control/disturb other mobile phone users while gaming, clicking, interacting.

We will do that for you in our PointerNet!

# Quick Start Guide – Mobile Phones

2023-02-20-A
STRIKLY CONFIDENTIAL

Making the Mobile Phones Running in 5 Minutes:

1. Extract the "**ILSSoftware.ZIP**" with directory structure to a folder
2. Change all paths within the INI files of the programs to the selected folder
   All program INI files have the same name as the application. Only exception is are the Unity based games, which INI files are always called "**!MYINI.INI**"
3. Set the port number the webserver uses in "**ILSMobilePhonePointer.INI**"
4. Set the firewall to allow "**ILSMobilePhonePointer.EXE**" to receive and send HTTP packets on the given port of step 3
5. Start "**ILSMobilePhonePointer.EXE**"
6. If everything is done carefully, open a browser and open http://localhost:XXXX where XXXX is the port of step 3 (https not supported!!!)
7. Find out the IP address of the PC on which "**ILSMobilePhonePointer.EXE**" is running by using "ipconfig"
8. Give the mobile phone access to your (Guest) W-LAN (can be done using an QR code)
9. Open the chrome browser on the android mobile phone and open http://y.y.y.y:xxxx where "y.y.y.y" is the IP address of step 7 and "xxxx" is the port number of step 3 (https not supported!!!)
10. Now, the above viewed start page should be visible (in portrait mode)
11. Start "**ILSMemoryTest.EXE**" and click OK on the message box which appears (you may click it also later)
12. Edit "Apps\ProofOfTechnologyGame\!MYINI.INI":
    [Control] GameMode=1
13. Start our Unity demo game "**ProofOfTechnologyGame.exe**"
    It will show 3 circles (red, green blue) and depending on the button you click, these 3 circles send 6 different vibration patterns to the mobile phones
    You can stop the game either by the windows mouse or the master pointer (also mobile phone) clicking onto the button "Quit the Game".
14. Edit "Apps\ProofOfTechnologyGame\!MYINI.INI":
    [Control] GameMode=2
15. Start our Unity demo game "**ProofOfTechnologyGame.exe**"
    It will show 4 corner points which you must click with the master pointer (or with the mobile phone). With this you can see the simple camera calibration which naturally is not necessary if you use mobile phones (without camera) only!
    You can stop the game either by the windows mouse or the master pointer (also mobile phone) clicking onto the button "Quit the Game".
16. Happy Mobile Phone Pointing and Clicking

Troubleshooting:

1. Ping the mobile phones IP address from the PC where "**`ILSMobilePhonePointer.EXE`**" is running
2. Try a different PC which is connected to your W-LAN and/or LAN and try opening the above HTTP URLs there
3. If http://localhost:XXXX is working but no external access it is a firewall problem
4. It also might be, that your PC has more than one IP address and we have unfortunately no idea to which IP address our webserver then binds.
5. If the "Test Vibration" button does not immediately vibrated for 500msec, then the Vibration API is simply disabled in your mobile phone or JavaScript is disabled or you are using an **`iPhone`** or an not supported browser.

There are LOG files for each of our processes:

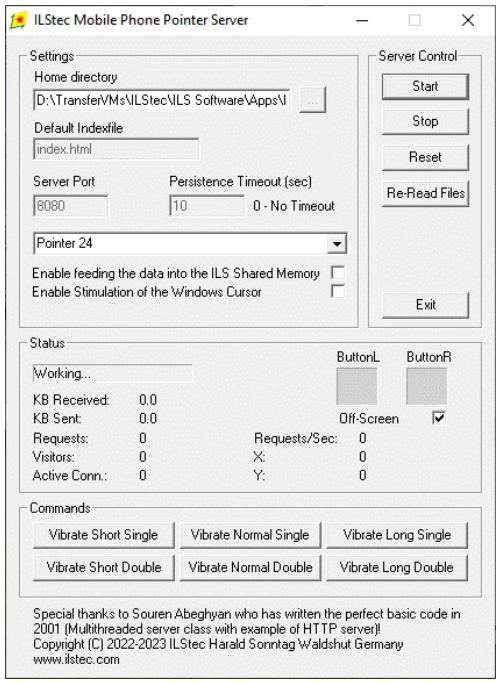
```
ILSMemoryTest.LOG
ILSMobilePhonePointer.LOG
```

But keep in mind that logging must be enabled in the INI file of the process:

```
[General]
Debug=FALSE
##Log=TRUE
Log=FALSE
LogLevel=1 .. . 4
```

And logging naturally slows down the speed of the programs because they log to a file in the moment and not to the shared memory in the future.

Once everything is running you will see the webserver:




You can select a specific mobile phone (named “Pointer 24” in the above picture), naturally the same number must have been entered on the mobile phone.

Then you will see that the **`ButtonL`** and **`ButtonR`** squares change their color depending on which button you click/press, the normalized X and Y co-ordinates are shown, the Off-Screen checkbox changes, if you close your mobile phone browser or if the mobile phones enters stand-by.

You can send vibration commands that mobile phone (the lower 6 buttons).

And you may enable the stimulation of the Windows cursor including the buttons.

Once you have set this checkbox, it is hard to uncheck it because the mobile phone permanently overwrites your standard mouse. Hold down the keys CTRL and SHIFT on the PC and the checkbox is cleared and you get back control with your standard mouse.

You can open e.g. Paint and draw with the mobile phone and use it like you standard mouse.

# Quick Start Guide – Mobile Phones

2023-02-20-A
STRIKLY CONFIDENTIAL

Attention:

If you have a virtual machine running and it get the focus then the SHIFT-CRTL mechanism is no more working on the host, because the virtual machine then is the owner of the keyboard.

The only way to get back control with your standard mouse is to kill "**`ILSMobilePhonePointer.EXE`**" from a task manager opened with CTRL-ALT-DEL!

We have measured the overall speed of our webserver and opened 12 browsers sending 20 requests/sec and the webserver showed 240 requests/sec (12*20) and still only needed 3% of the CPU time on a low end 3GHz PC (not a gaming one).

And please keep in mind that the whole mobile phone system depends on the through-put of your access point, the speed of the LAN connection to the PC running our webserver, the speed of the firewall, the number of applications running on that PC simultaneously, number of devices accessing the same W-LAN etc.

Especially if there are clients on the W-LAN which stream videos etc., the through-put of the W-LAN goes down dramatically!

Improvements might be:

Use a "D-Link WLAN 300MBit Nano USB Dongle". It generates a new access point directly connected to your PC. The distance of the mobile phones is a problem then (bad antenna within the dongle) but 4-5 meters should work. We have got some of them and give them a try.

It is also possible to use a "T-Dongle ESP32-S2 Wireless WIFI Module" which has an integrated 240MHz ARM core and generates an access point and an integrated webserver. We have got some of them and them a try.

Use a Raspberry Pi with integrated W-LAN. It also can generate an access point and an internal webserver and transfer the raw data to the PC using a low speed USB connection. We have some of them here and give them a try.

# Quick Start Guide – Mobile Phones

2023-02-20-A
STRIKLY CONFIDENTIAL

Making the Camera and Pointers Running in 5 Minutes:

This step requires a camera and 1-10 pointers to start.

This chapter does not yet exist but it will be added within the next 5-7 days!

Then you will find some more programs like "**`CamWinUSBService.EXE`**" and a tool named "**`QRCodeGenerator.exe`**" etc.

You will also find "**`ILS Camera Calibration.exe`**" which is a Unity based "game" and allows to click onto the 4 corners of the screen with the master pointer. Then the co-ordinates are transferred to "**`CamWinUSBService.EXE`**" which then makes sure, that the camera switches to normalized co-ordinates with trapezoid correction. If the camera is at the horizontal center of the screen but not at the vertical center, then it will "see" a trapeze with the lower and upper lines being parallel. But if it is not at the horizontal and also not at the vertical center it will "see" a trapezoid which is a rectangle where no line is parallel to any other line. We do an automatic trapezoid correction of all points to a flat screen.

Unity based game does not mean you have to install Unity! It is a package compiled by Unity for a Windows based platform and executes there without Unity.

We also have programs like "**`PointerConfig.EXE`**" and "**`PointerTester.EXE`**" which allows setting of pointer internal data, firmware update, read out serial numbers and set and read user specific data. You will also get "**`ILSConsole.EXE`**" which allows to connect to any of the integrated logging FIFOs.

We did not yet decide if all this can be covered by a quick installation guide!