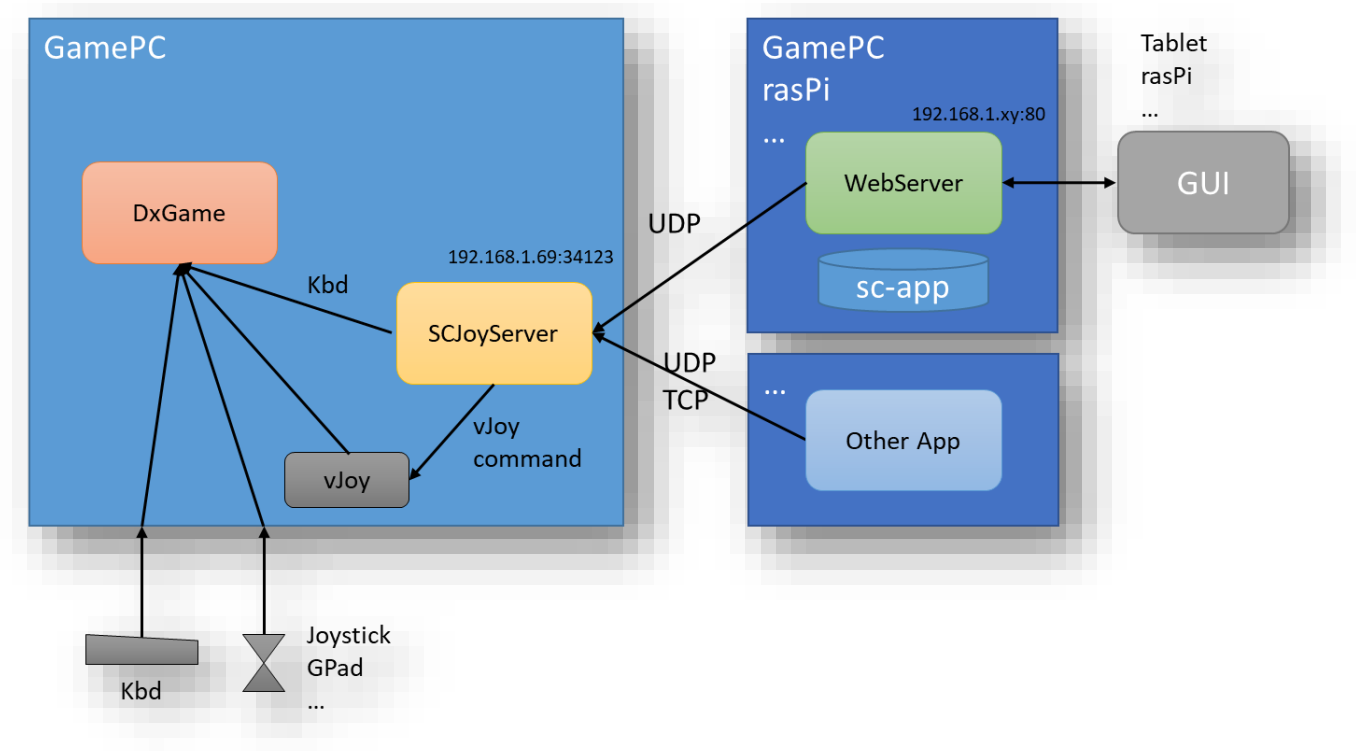
SC Remote Server – How To …

20181229/Cassini

General Overview



There are three Parts involved here:

- A game (application) that wants to get keyboard and/or Joystick commands to act on them.
- The SCJoyServer which provides keyboard and optionally vJoy commands to the receiving application e.g. a game and waits for triggers to do it on a network address/port.
- An application such as a Web Server that hosts the sc-app (SCRemoteServer) which sends commands to trigger the SCJoyServer via an UDP protocol.

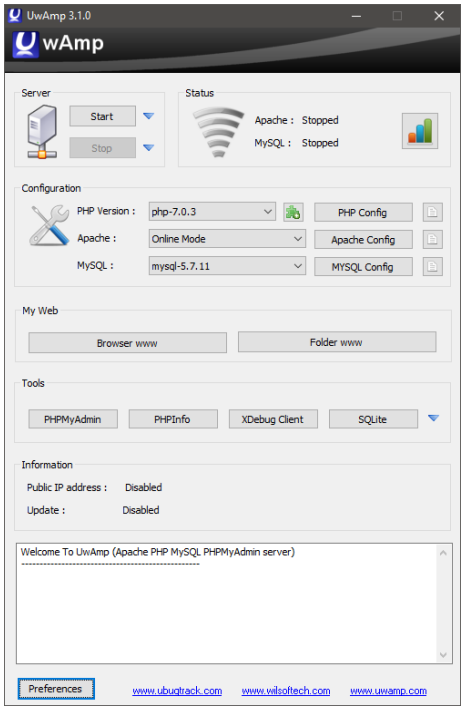
The SCJoyServer listens on its network address and port for incoming TCP or UDP connects.
Where UDB is just a one off message sent over the network that triggers a single command for the game.

The WebServer serves a GUI for a client such as a tablet or other device with a screen and input means. The clicks, touches are then initiating the command sent via network to SCJoyServer and from there it is injecting vJoy or keyboard input for the game application.

While this doc describes how to setup the SCJoyServer and WebServer on the GamePC and how to create your own sc-app; it would be possible to trigger the SCJoyServer from further applications running on either the GamePC or preferably any other computer that is able to connect to the GamePC and equipped to send meaningful triggers to the SCJoyServer on the GamePC.

Infrastructure Setup

WebServer



Any web server supporting PHP will do

I found UwAmp to be least intrusive and it was setup in a minute:

https://www.uwamp.com

Download the latest ZIP version.

Extract into an empty directory

Run UwAmp.exe …

Configure PHP 7 (may be 5 would do)

Configure 'Online Mode' if you want to access from the intranet (else it works on loopback only)

Configure a Web Server Port (default is :80)

START via blue arrow –> (Apache is enough, MySql is not required

Target your browser to `webServerIP:80` (or the port you defined)

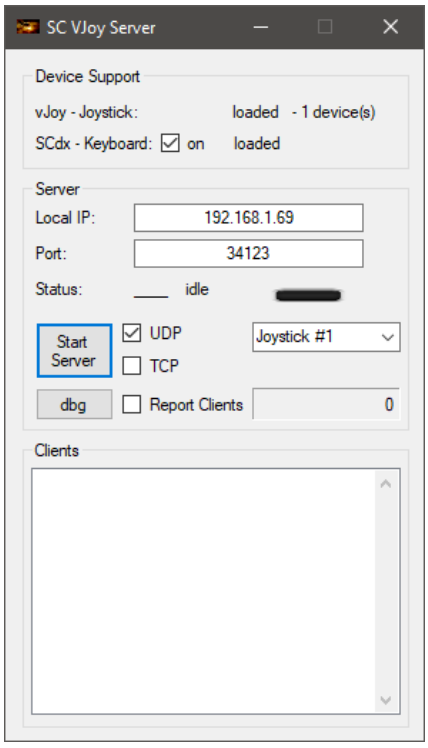
You find the default page that was supplied with the package.

Note: You may run the remote site on any web server with PHP anywhere e.g. also on a small rasPi.

SCJoyServer

Get the package here:

https://github.com/SCToolsfactory/SCJoyServer/releases



Extract into an empty directory on your Game PC.

Run `SCJoyServer.exe`

Configure your Game PC IP and the port to use (must be a free one)
IANA would suggest:
Dynamic and/or Private Ports are those from 49152 through 65535.
But for your in house use the default one is usually good enough.

You may see if a vJoy device is found – it is not required but then you can only use keyboard commands. Select one of the Joysticks if there are any.

Start the Server once you want to receive commands.

→ Beware keystroke commands are now supplied to your active window on that PC i.e. you may feed keys into the wrong application – some Window shortcuts such as Alt F4 (Close App) have unexpected results…

The Server reports commands received by incrementing the number at Report Clients.

dbg opens a window that may help to find issues.

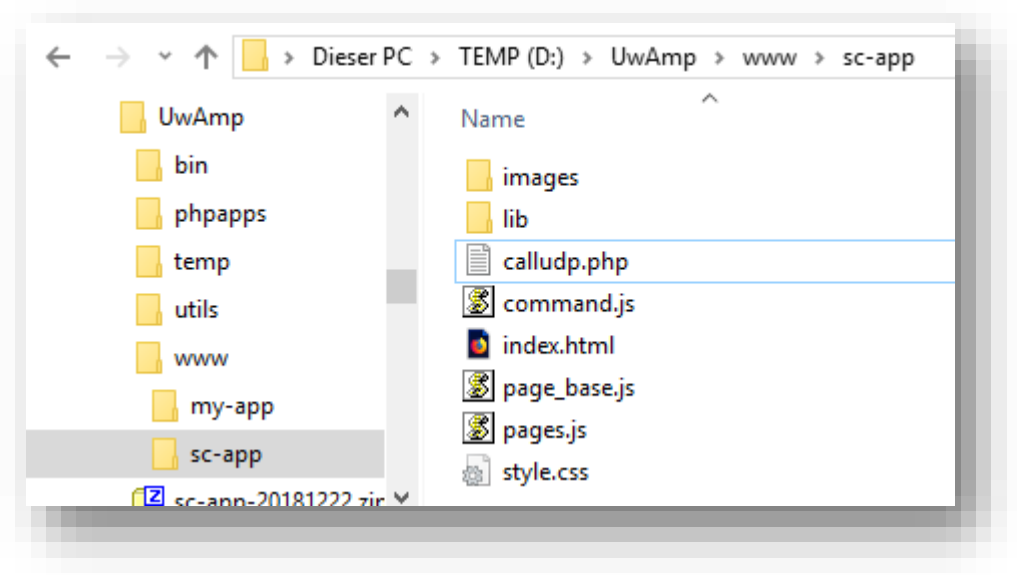
vJoy Driver

is located here:

http://vjoystick.sourceforge.net/site/index.php/download-a-install/download

Setup of the SCRemoteServer Web Site

Copy all of `sc-app` to the UwAmp Web directory (`www`)



Should then look as above.

Any directory added to www is exposed by the web server and can be accessed using:

`http://webServerIP:80/directory/`

e.g. our new site as `http://webServerIP:80/sc-app/`

Note `webServerIP` is something like: 192.168.1.68 i.e. the IP address of the machine where the web server is running.

If you now start your UwAmp Apache server it will serve `index.html` with the default setup.

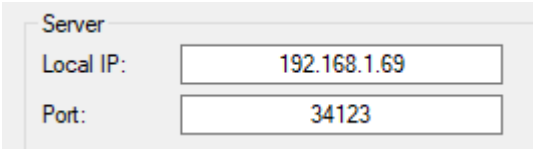
To test the site you may need to change the SCJoyServer address and port to your game PC.

Edit `pages.js` with Notepad

Find:

```
// the vJoy Command Server IP
const IP = '192.168.1.69';
// the vJoy Command Server PORT (UDP protocol)
const PORT = 34123;
```

Change it to the values you set in the SCJoyServer.



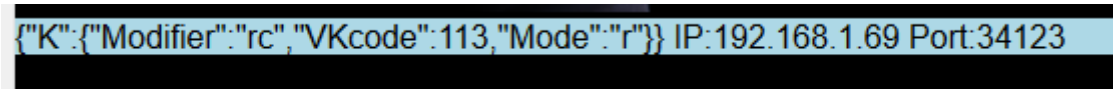
Now you should be able to navigate to `http://webServerIP:80/sc-app/` and your browser should show the following (see next page for an image from the iPad2 – Safari browser)

Click the Tabs (top row thumbnails) to change the page.

The DEBUG information is showing what command was issued later.

If you click and hold the mouse on the buttons it should issue a Press command, Releasing issues the release to the key or button. i.e. it presses as long as you click or touch the item.

Clicking ‘Key’ shows:



You may see Mode: “p” while pressing and “r” once released

The Keycode was 113 – which is F2

Modifier is “rc” which is Right Ctrl – so the key sent was RCtrl+F2.

Now you are ready to play with your own customization..

Customization

The workflow is:

- Design one up to 5 pages (images) size is currently fixed to 1366x768 pixels
- Edit `pages.js` to setup your hit targets.

That's it.

The site consist of 5 pages (tabs) where the background image contains all drawn visuals.

For every page one has to define hit targets. Those serve as interaction spots where a click or touch is recognized and then the according command is sent to the SCjoyServer.

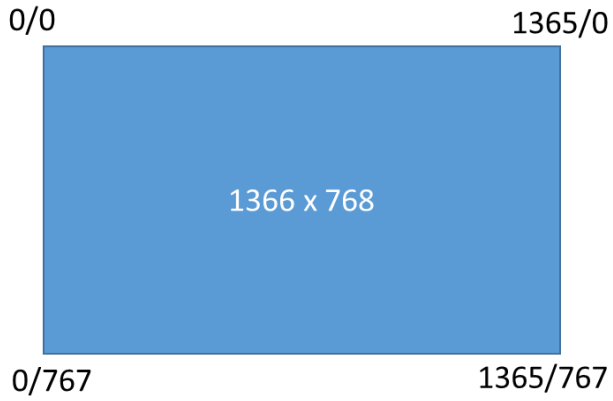
Targets can be of different modes and shapes. The target layer will respond with visual clues depending on the mode of activation. There is either a visual feedback on a hit, a toggle state or an analog slider setting.

Drawing background images

Format can be either PNG or JPG, Size is 1366x768 pixels.

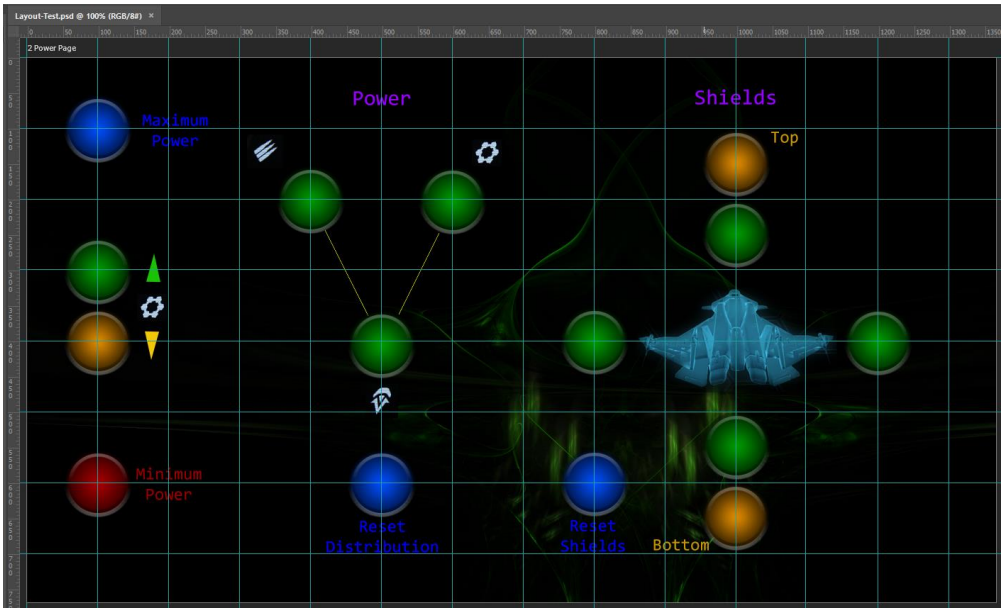
Images go into the `images` directory.

To identify hit targets you must record the X and Y pixel location and the size of the hit target. Locations are top – down and left right oriented. I.e. top left corner is 0/0.



The example buttons are about 100x100 pixels which is an acceptable touch target size on an iPad 2.

Using guides and place targets on the grid makes it rather easy – below a 100x100 grid is used.



Editing pages.js

- ➔ Use only notepad or a similar code editor (never Wordpad or Word or any other word processor) Make a backup copy before you edit…
- ➔ Once edited and saved you have to reload the page in your browser to make it active (F5)

The file consist of 5 similar parts for each of the 5 supported pages:

```
// PAGE 1 Construction
const page_1_obj = new Page_Base_obj(
  "Test Page 1",
  'images/page_1.png',
  [
    new Target("my1", 200, 100, 90, 0, ItemTypeKey, ItemModePR, VK_F2, ItemKModRCtrl),
    new Target("my2", 200, 300, 90, 0, ItemTypeButton, ItemModePR, 3, ItemKModNone),
    new Target("my3", 500, 200, 90, 0, ItemTypeKey, ItemModePR, VK_Y, ItemKModNone),
    …
  ]
);
```

The page name – it is also shown below the Tab for easy navigation.
The image path and filename – it refers to the image to load as background image.

You may change them according to your need and image file naming.

Then there are as many Target definitions as hit areas you want to use on that page.

The Test Page has 4 buttons, 4 toggles and 5 axis parts (blue bubbles) and hence 4 + 4 + 5 entries for the hit targets.

A Target is defined with:

Name e.g. “my1” this must be a unique name within the page but serves only internal purposes.

X, Y, DW, H Center of the hit target (X/Y) and for circles diameter (DW) and height=0 (H), for rectangles use width (DW) and height (H) of the shape.
The hit area of my 100x100 buttons get a circle with a diameter of 90 pixels in order to show properly. Hit targets are marked with a semi-transparent circle that gets whiter when pressed (alpha 0.05 vs. 0.3 – defined in page_base.js), or a black shape for toggle mode items (they black out if OFF).

Type – the type of control (key, button, axis, rotAxis, slider)

Mode – the mode of activation (Press/Release, Toggle, Tap, Value, Analog)

Value – a value or index related to the control

KMod – for keystrokes only a key modifier such as Left or Right - Ctrl, Alt, Shift

Key and Button ‘Press – Release’ activation

```
new Target("my1", 200, 100, 90, 0, ItemTypeKey, ItemModePR, VK_F2, ItemKModRCtrl),
new Target("my2", 200, 300, 90, 0, ItemTypeButton, ItemModePR, 3, ItemKModNone),
```

A **Type** whether a Key or a Button is triggered (`ItemTypeKey, ItemTypeButton` are valid here)
The **Mode** is `ItemModePR` (press, release)
→ The key/button remains down as long as you click/touch.
The **Value** as Key Code or button index (1.. max button).
The last one is a **Key modifier** such as RightCtrl + Key.

For **keys** you may use the symbols defined in the file command.js. Those are `VK_something`. E.g. `VK_A` is an A pressed, `VK_3` is the 3 key one on the main keyboard, 3 on the num pad would be `VK_NUMPAD3`.

Note: if you are using a non US (QWERTY) keyboard/language setting you are punished with the same issues that the game gets the input from the key location rather than the imprint on the key. I.e. QWERTZ keyboards issue a Y Key but the game sees it as Z input. Here you may use the same letter key assignment as on the keyboard imprint.
For special chars use the ones you really need e.g. VK_SEMICOLON issues really a semicolon.
You may need to check what arrives in e.g. notepad as receiving active window.

Valid key modifiers are:
`ItemKModNone, ItemKModLCtrl, ItemKModRCtrl, ItemKModLAlt, ItemKModRAlt, ItemKModLShift, ItemKModRShift`

Once edited and saved you have to reload the page in your browser to make it active (F5)

Key and Button 'Toggle' activation

Toggles might get it sometimes wrong either because an input is lost by the game or you pressed the keyboard and the App would not know about that one.

```
new Target("tg1", 200, 500, 90, 0, ItemTypeKey, ItemModeTogOn,   VK_V, ItemKModNone),
new Target("tg2", 300, 500, 90, 0, ItemTypeButton, ItemModeTogOn,   4, ItemKModNone),
new Target("tg3", 400, 500, 90, 0, ItemTypeKey, ItemModeTogOff,  VK_W, ItemKModNone),
new Target("tg4", 500, 500, 90, 0, ItemTypeButton, ItemModeTogOff,  5, ItemKModNone),
```

A **Type** selector whether a Key or a Button is triggered (`ItemTypeKey, ItemTypeButton` are valid here)
The **Mode** is either `ItemModeTogOn` (ON at startup) or `ItemModeTogOff` (OFF at startup).
→ It issues a single stroke when clicked/touched and changes the visual toggle state from one to the other.
The **Value** as Key Code or button index (1.. max button).
The last one is a **Key modifier** such as RightCtrl + Key.

Valid key modifiers are again:
`ItemKModNone, ItemKModLCtrl, ItemKModRCtrl, ItemKModLAlt, ItemKModRAlt, ItemKModLShift, ItemKModRShift`

NOTE: if you use it on your PC where the SCJoyServer runs you just issue the keys into your active window – **the browser** – and you may see strange effects when using Function keys.. or other browser short cuts.
Alternatively use a tablet to test the web client and make the active window on your PC an application that does not bother with keys sent – e.g. an empty notepad window.

Key and Button 'Short Tap' activation

Sometimes you may want to issue just a short tap.

```
new Target("my4", 700, 200, 90, 0, ItemTypeKey, ItemModeTap, VK_Z, ItemKModNone),
```

The items here are as above:

Name, CenterX, CenterY, DiameterOrWidth, Height, **Type**, **Mode**, Value, kMod

This is for **Type** key or button, `ItemTypeKey, ItemTypeButton` are valid here.

The **Mode** is then `ItemModeTap` which equals a single short key or button strike independent of the length of the key/button click or touch.

Axis and Slider activation by Value

You may also issue axis and slider actions with a value.
Value range is always 0…1000 (min … max), 500 is center.

An X axis single click/touch command with a value of 500 (center point) looks like:

```
new Target("ax2", 1100, 300, 90, 0, ItemTypeXaxis, ItemModeVal,  500, ItemKModNone),
```

The items here are as above:

Name, CenterX, CenterY, DiameterOrWidth, Height, **Type**, **Mode**, **Value**, kMod (where kMod is ignored)

The **Mode** is `ItemModeVal` as you give a Value to set in the command.

The **Type** for the axis and slider are:

```
ItemTypeXaxis, ItemTypeYaxis, ItemTypeZaxis,
ItemTypeRXaxis, ItemTypeRYaxis, ItemTypeRZaxis,
ItemTypeSL1, ItemTypeSL2
```

Value is the set point 0...1000

You may use e.g. 5 targets to cover 0, 250, 500, 750, 1000 as in the example page 1.

Axis and Slider activation by analog control

If you wish to use a slider with analog behavior use the following

```
new Target("al1", 1000, 300, 100, 500, ItemTypeXaxis, ItemModeAnalog,   500, ItemKModNone),
```

The items here are:

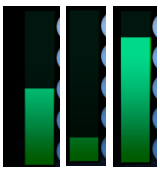
Name, CenterX, CenterY, **Width**, **Height**, **Type**, **Mode**, **Value**, kMod (where kMod is ignored)

The **Mode** is `ItemModeAnalog`. **Value** is the initial set value 0..1000.

The **Type** for the axis and slider are:

```
ItemTypeXaxis, ItemTypeYaxis, ItemTypeZaxis,
ItemTypeRXaxis, ItemTypeRYaxis, ItemTypeRZaxis,
ItemTypeSL1, ItemTypeSL2
```

The orientation of the slider is determined by the larger extent of dw or h.
In the above example **dw**=100 and **h**=500 → **h** > **dw** defines a vertical orientation of the sliding effect.


Slider at different positions. Zero is left or bottom, max is right or top.

The slider target should match the area that is to be hidden or revealed depending on the control set point. The effect covers and therefore hides the area right or above the current set point with a black mask of almost zero transparency. The color and alpha used for this effect is in page_base.js defined (`A_SliderCol, A_Alpha`).

Note: as there is no feedback from the receiving application the control just shows what you clicked or touched but not what the device setting is. I.e. two of the same controls e.g. Y-axis on the same or different tabs are not tracking each other.

Circle and Rectangle Shapes and PR and Toggle items


```
new Target("hli", 400, 100, 80, 75, ItemTypeKey, ItemModeTogOff, VK_3, ItemKModNone),
```

Above is the HeadLights target definition, a square with center of 400/100 and a size of 80x75.

For analog sliders only a rectangle is supported, it needs both width and height.

For toggles you may even use red and green logic – but again sometimes toggles might get it wrong either because an input is lost by the game or you pressed the keyboard and the App would not know about that one.

The green ones are meant to be OFF when dark i.e. Lights are off, Scanning Mode is ON, Mining OFF.
The red ones are meant to be OFF when light i.e. COMSTAB is off and G-Force is ON (hence dark because it's safe – red would be unsafe)

→ But it is up to you to make this as helpful as possible…