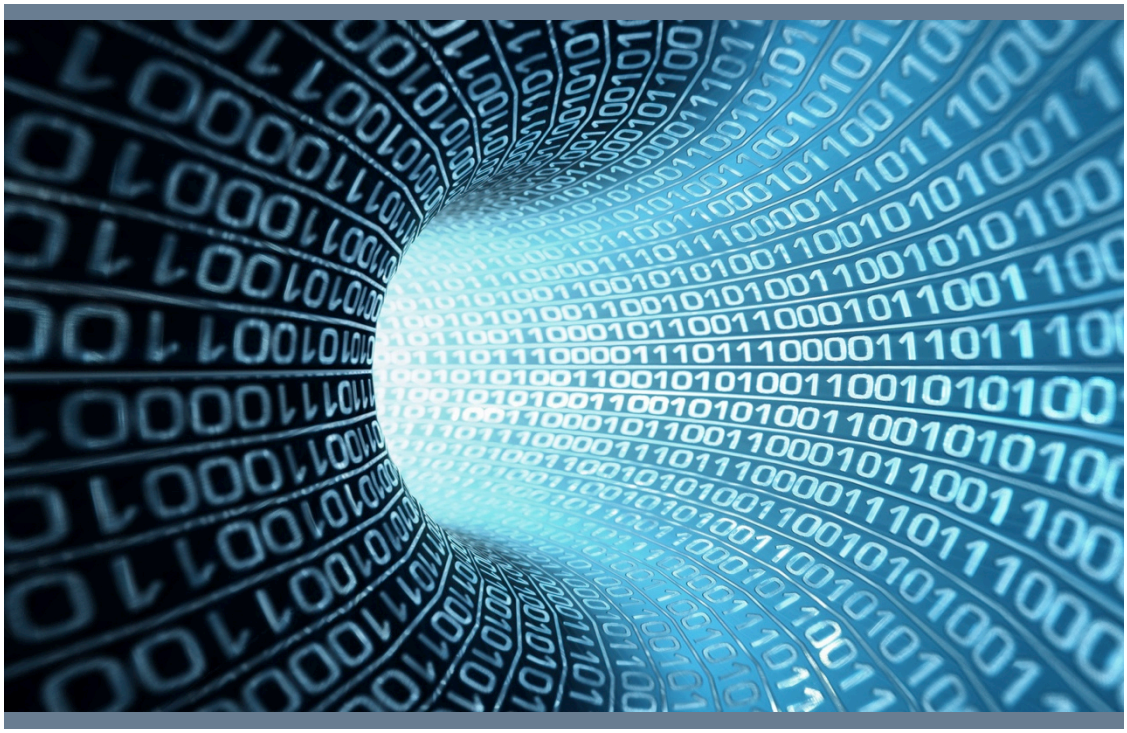

Dohaeris

Website zur Steuerung der Matrix

Nicola Knuchel, Raphaël de Picciotto
1.0, 28.10.2015

Berner Fachhochschule
Passerelle Informatik
TI

Inhaltsverzeichnis

1 Protokoll

Unser Ziel für diese vier Wochen war es eine elegante Lösung für das Routen von Input und Output bei einer 8x8 Matrix zu finden. Eine Lösung, welche uns auch von einem Handy aus den Zugriff auf die Matrix ermöglicht. Dies sollte in Form einer App oder einer Website geschehen.

1.1 Woche 1

Das Ziel der ersten Woche war das Kennenlernen der Matrix und die Möglichkeit, von einem Computer aus auf die Matrix zugreifen zu können.
Wir haben die Matrix, 4 Apple-TVs, einen Netgear-Switch und die Toolbox auf einem Rollregal aufgebaut und sowohl die Input-Kabel als auch die Output-Kabel beschriftet.
Unsere erste Vorstellung des Endprodukts war eine App, die man sich auf das Handy herunterladen kann, um damit die Matrix zu bedienen. Um eine Vorstellung zu haben, wie wir diese Idee umsetzen wollen oder könnten, haben wir erste Skizzierungen für eine App-Oberfläche gemacht. Wir haben auch im Internet nach Möglichkeiten recherchiert, eine App herzustellen, z.B. mit Hilfe des App-Maker, dem JQuery oder/und dem Phonegab. Entschieden haben wir uns schlussendlich aber für das Erstellen einer Website, von welcher aus man die Matrix bedienen kann.
Unter anderem haben wir auch mit der Matrix Funktionen getestet. Der Versuch, eine ssh-Verbindung zwischen Matrix und Computer herzustellen, ist fehlgeschlagen. Diese Verbindungsart funktioniert nicht.
Jedoch ist die zweite Verbindung via telnet geglückt. Der Port für die telnet-Verbindung ist 23.
Ausserdem haben wir gelernt, mit GitHub umzugehen.

1.2 Woche 2

Das Ziel dieser Woche bestand darin, eine Verbindung zwischen der Matrix und einer PHP-Datei herzustellen. Ausserdem sollten wir Presets festlegen.
Auch wenn wir PHP kurz im Unterricht angeschaut haben, mussten wir einiges lernen. Wie z.B. einen TCP/IP-Client mit Hilfe eines sockets. Um die Verbindung von PHP und Matrix zu überprüfen, haben wir den XAMPP Apache Server heruntergeladen und installiert.
Zudem haben wir uns erkundigt, wie man einen Full-Screen-Modus einrichten könnte. Dies ist für die Matrix leider nicht möglich.
Zur PHP-Datei haben wir neben der Rout-Methode die Synchronized-Methode hinzugefügt. Diese bewirkt, dass ein Input an alle Outputs geroutet wird.
Inzwischen haben wir auch die PHP-Datei soweit bearbeitet, dass sie funktionstüchtig ist. Wir haben einige Schritte aus dem Internet herauskopiert und diese dann auch kommentiert, um sie unter anderem auch für uns verständlich zu machen. Gegen Ende der Woche teilten wir die PHP-Datei in 2 Teile auf. Die Datei *commander.class.php* für Funktionen und die Datei *commander.php* für Kommandos. Zudem haben wir eine Switch-Funktion eingeführt, welche die Rückmeldung der Matrix User-freundlicher macht. Ursprünglich dachten wir, dass die *commander.php* Datei als Bindeglied zwischen der Matrix und der Source-Datei fungieren könnte. (Diese Annahme war inkorrekt.)
Da wir nicht immer mit der Matrix direkt arbeiten konnten, haben wir auch schon mit dem Aufbau und dem Gestalten einer groben Weboberfläche angefangen.

1.3 Woche 3

Das Ziel der Woche war, über die Weboberfläche die Matrix zu kontrollieren.
Begonnen haben wir die Woche mit der Entdeckung, dass PHP eine Objekt-orientierte Sprache ist. Des Weiteren leiteten wir eine Recherche zu den Verbindungsmöglichkeiten der PHP- und der html-Datei ein. Wir suchten nach einem Weg, die *index.html* Datei in die *commander.php* zu integrieren. Doch wir haben herausgefunden, dass die beiden Dateien via JavaScript miteinander kommunizieren können. Demzufolge haben wir eine weitere Datei erstellt: *sendCommand.js*.
Diese Datei erfüllt die Aufgabe, „Type, Input und Output" von der *index.html* Datei entgegen zu nehmen und an die *commander.php* Datei weiterzuleiten.

In den PHP-Dateien haben wir 4 verschiedene Presets eingeführt und sie mit Buttons verbunden, um ihre Funktionstüchtigkeit zu testen.
Unsere Weboberfläche hat sich von einfachen Onclick-Buttons für jeden einzelnen Input zu Dropdown-Menus für alle Outputs mit individueller Inputwahl entwickelt. Man konnte am Ende der Woche auf einem Bildschirm-Menu einen beliebigen Input auswählen. Die Website war da ca. zu 80% einsatzbereit. Die kleine Matrix mit dem Input ClickShare ist weiterhin widerwillig.

1.4 Woche 4

Das letzte Wochenziel lautet, die Website und die Dokumentation zu 100% zu beenden.
An der *commander.php* haben wir das Problem mit dem Umschalten der kleinen Matrix gelöst und die Website Mobile-freundlich gemacht.
Die Dokumentation hat einen Grossteil dieser Woche in Anspruch genommen. Zudem Haben wir versucht, die Website noch etwas Ästhetischer zu machen.
Für Donnerstag haben wir eine Präsentation über unser 4-Wöchiges Projekt vorbereitet.

2 Schema

2.1 Schema 1

Dieses Schema zeigt auf, wie die Matrix mit den Inputs und Outputs verkabelt ist.

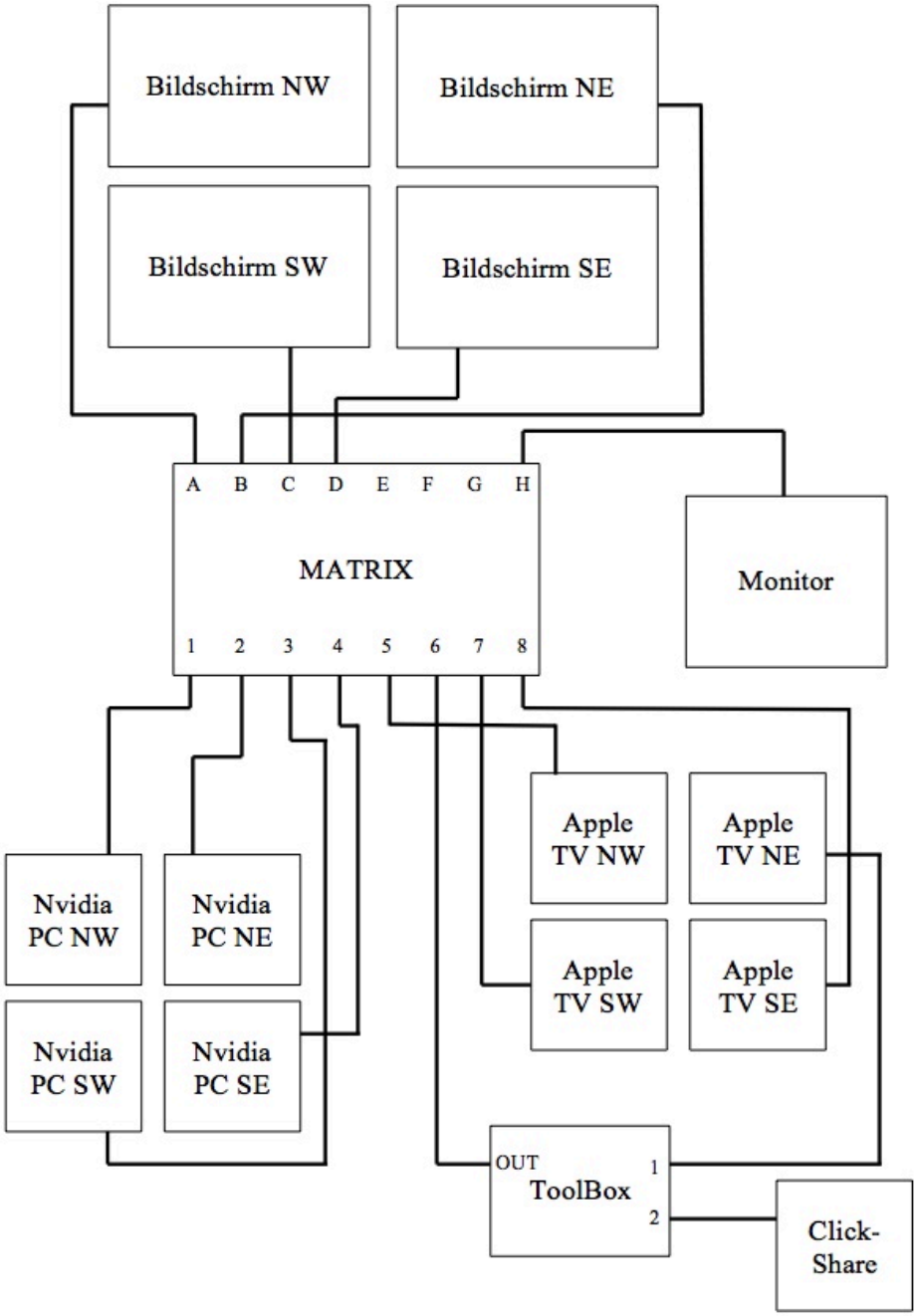

Abb. 1 : Schema 1, Verkabelung der Matrix

2.2 Schema 2

Das zweite Schema beschreibt und veranschaulicht den Verbindungsvorgang anhand eines Beispiels.

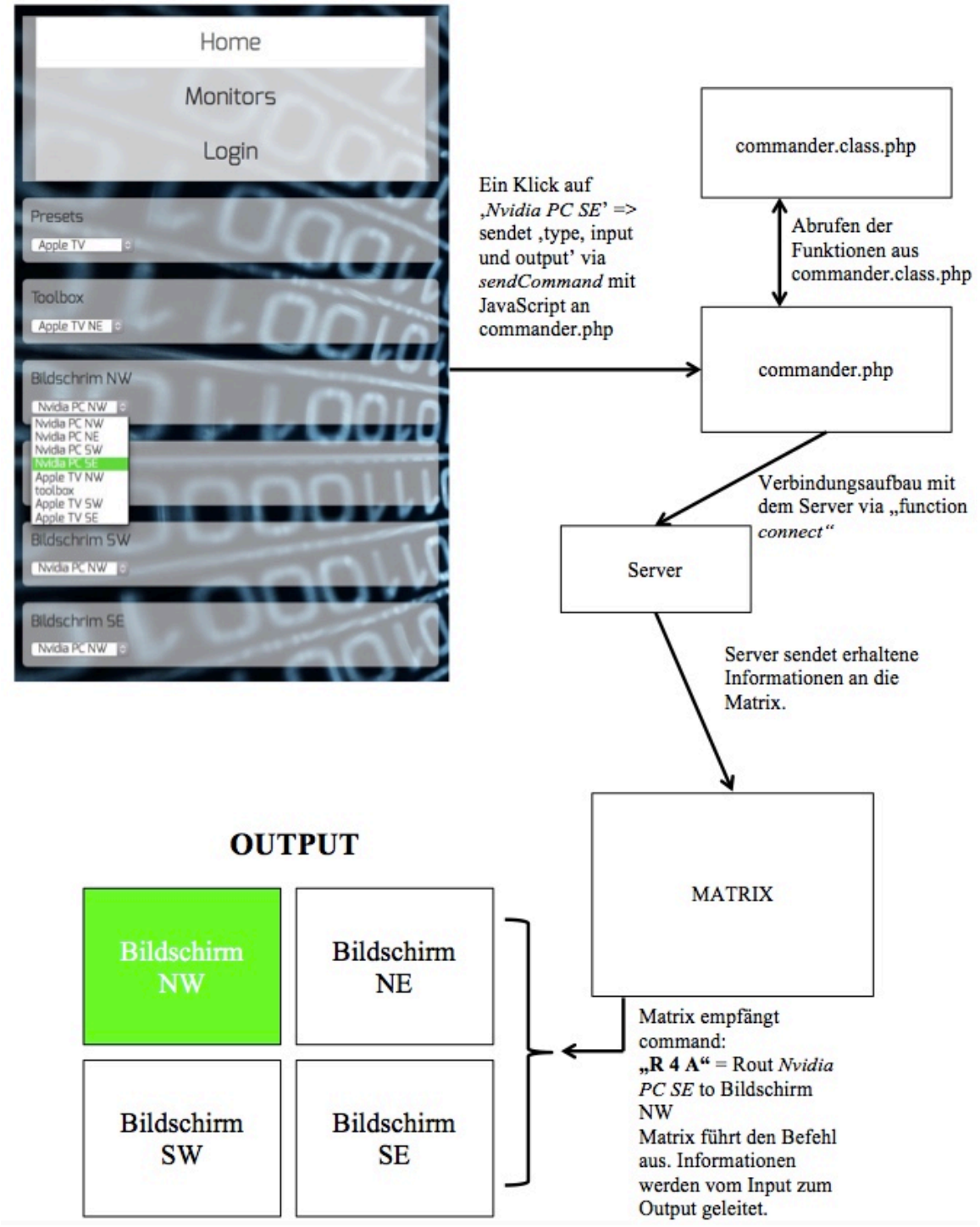

Abb. 2 : Schema 2, Ablauf eines Rout-Vorgang

3 Code

In diesem Abschnitt erklären wir kurz einige Codes, die wir verwendet haben.

3.1 PHP

Das PHP ist hier in zwei Teile aufgeteilt. Einerseits haben wir die Funktionen in der *commander.class.php*, andererseits nimmt die *commander.php* die Befehle der html Datei entgegen, sortiert sie und legt den Ablauf fest.

3.1.1 commander.class.php

```php
function connect() {
    $this->socket = socket_create(AF_INET, SOCK_STREAM, SOL_TCP);
    if ($this->socket === false) {
        //echo "socket_create() fehlgeschlagen: Grund: " . socket_strerror(socket_last_error()) . "\n";
        $this->connected = false;
        return;
    }

    if (socket_connect($this->socket, $this->ip, $this->port) === false) {
        //echo "socket_connect() fehlgeschlagen.\nGrund: " . socket_strerror(socket_last_error($this->socket))
        $this->connected = false;
    } else {
        // wait the welcome message - need sonnst timeout
        while($resp = socket_read($this->socket, 1000)) {
            $out = $resp;
            if (strpos($out, "\n") !== false) break;
        }

        $this->connected = true;
    }
}
```

Abb. 3 : Verbindungs-Funktion

socket_create ist eine Funktion, welche im PHP integriert ist.
Das erste Parameter *AF_INET* legt legt in diesem falle die Protokollfamilie IPv4 fest. *SOCK_STREAM* steht für den Kommunikationstyp, den dieser Socket verwendet und *SOL_TCP* bestimmt das Protokoll.

Im zweiten Absatz werden die Verbindungsparameter, die der Socket entgegennehmen soll, definiert. Im *while-Loop* befindet sich das *socket_read*, welches die Antwort der Matrix in der Variable *$resp* speichert.

```php
class Commander {
    private $pInputp;
    private $port;
    private $socket;
    private $connected = false;

    public function __construct( $pIp = NULL, $pPort = NULL ){
        $this->ip = $pIp;
        $this->port = $pPort;
        $this->connect();
    }
```

Abb. 4 : Class Commander Attribute

class Commander ist die Source Datei. Die Attribute haben oft gleiche Namen. Um sie besser auseinanderhalten zu können, werden sie in einer Funktion mit dem Präfix *p* versehen.

```php
function route( $pInput, $pOutput = null ){
    if(!$this->connected)
        return false;

    $possibleInputValue = array(1, 2, 3, 4, 5, 6, 7, 8, 9);
    $possibleOutputValue = array('a', 'b', 'c', 'd', 'h', null);

    if( !in_array($pInput, $possibleInputValue) )
            return false;

    if( !in_array($pOutput, $possibleOutputValue) )
            return false;

    if( is_null( $pOutput ) )
        $command = "r ".$pInput."\r\n";
    else
        $command = "r ".$pInput." ".$pOutput."\r\n";

    socket_write($this->socket ,$command, strlen($command));

    while($resp = socket_read($this->socket, 1000)) {
       if (strpos($resp, "\n") !== false) break;
    }
```

Abb. 5 : Route-Funktion

In dieser Funktion wird festgelegt, welche Funktionen entgegen genommen werden können. *$pOutput = null* ist so deklariert, weil diese Funktion sowohl für die 8x8 Matrix als auch für die Toolbox anwendbar sein soll.

Socket_write ist die Funktion, welche mit der Matrix kommuniziert.

3.1.2 commander.php

```php
if(isset($_GET['type'])){
    switch($_GET['type']){
        case 'io':
            $myCommander = new Commander( '147.87.117.33', 23 );
            $myCommander->route( $_GET['in'], $_GET['out'] );
            echo "nice: " . $_GET['in']." to ".$_GET['out'];
            if( $_GET['in'] == 9){
                $myCommander = new Commander( '147.87.117.33', 23 );
                $myCommander->route( '6', $_GET['out'] );
                $myCommander = new Commander( '147.87.117.34', 23 );
                $myCommander->route( 2 );
                echo "very nice: " . $_GET['in'];
            }

            if( $_GET['in'] == 6){
                $myCommander = new Commander( '147.87.117.33', 23 );
                $myCommander->route( $_GET['in'], $_GET['out'] );
                $myCommander = new Commander( '147.87.117.34', 23 );
                $myCommander->route( 1 );
                echo "nice: " . $_GET['in'];
            }
```

Abb. 6 : TypeRecognition Protokoll

Diese Datei nimmt Parameter vom *sendCommand.js* entgegen und bestimmt den Ablauf des Vorgangs. Mit dem *case* rufen wir eine String-Erkennung auf, welches zur Auswahl des Vorgehens verwendet wird.
In der Variable *$myCommander* wird neben IP-Adresse und Port der Matrix die gesamte *commander.class.php* aufgerufen und eingetragen

3.2 JavaScript

```
function sendCommand( type, input, output) {

  var xhttp = new XMLHttpRequest();
  /*xhttp.onreadystatechange = function() {
    if (xhttp.readyState == 4 && xhttp.status == 200) {
       alert(xhttp.responseText);
    }
  }*/
  xhttp.open("GET", "commander.php?type="+type+"&in="+input+"&out="+output, true);
  xhttp.send();
}
```

Abb. 7 : Sende-Funktion

sendCommand sendet die angewählten Parameter in der Website an die *commander.php*.

Das Auskommentierte kann man aktivieren um festzustellen, ob das Kommando funktioniert.

```
<script type="text/javascript">
    var presetSelect = document.getElementById("availablePreset");
    var sourceSelectNW = document.getElementById("sourceSelectNW");
    var sourceSelectNE = document.getElementById("sourceSelectNE");
    var sourceSelectSW = document.getElementById("sourceSelectSW");
    var sourceSelectSE = document.getElementById("sourceSelectSE");

    presetSelect.onchange = function() {
        var value = presetSelect.options[presetSelect.selectedIndex].value;
        if (value != "") {
            sendCommand('preset', value);
            sourceSelectNW.value = "";
            sourceSelectNE.value = "";
            sourceSelectSW.value = "";
            sourceSelectSE.value = "";
            console.log('preset: ' + value);
        }
    };

    sourceSelectNW.onchange = function() {
        var value = sourceSelectNW.options[sourceSelectNW.selectedIndex].value;
        if (value != "") {
            sendCommand('io', value, 'a');
            presetSelect.value = "";
```

Abb. 8 : Reselect Bugfix

Wenn ein Preset ausgewählt wurde und die Bildschirme nicht von einer anderen Preset-Funktion ersetzt, sondern einzeln geroutet wurden, konnte man die vorherige Preset-Auswahl nicht mehr auswählen. Dieser js Code setzt die Auswahl nach jedem Preset-Aufruf wieder auf neutral oder “ “.

3.3 html

```
<div class="container">
    <div class="well well-lg">
        <center><h1>Dohaeris</h1></center>
    </div>

    <div class="well well-lg"> <!--Presets-->
            <center><p> Presets</p>
        <select id="availablePreset">
            <option value="">Please select...</option>
            <option value="1">Apple TV</option>
            <option value="4">PC Full-screen</option>
            <option value="3">Click-Share</option>
        </select></center><br>

    </div>
```

Abb. 9 : Dropdown-Menu Presets

<select id="availablePreset"> erstellt eine Liste mit den Namen, die unten als *option* aufgeführt sind.

```
<div class="well well-lg"> <!--NW-->
    <center>
        <table>
            <tr>
                <td class="table-responsive">
                    <div class="well well-lg">
                        <p> Bildschrim NW</p>
                        <select id="sourceSelectNW" type= 'io'>
                            <option value="">Please select...</option>
                            <option value="1"> Nvidia PC NW</option>
                            <option value="2"> Nvidia PC NE</option>
                            <option value="3"> Nvidia PC SW</option>
                            <option value="4"> Nvidia PC SE</option>
                            <option value="5"> Apple TV NW</option>
                            <option value="6"> Apple TV NE</option>
                            <option value="7"> Apple TV SW</option>
                            <option value="8"> Apple TV SE</option>
                            <option value="9"> Click-Share</option>
                        </select>
                    </div>
                </td>
```

Abb. 10 : Dropdown-Menu Bildschirm NW

Wichtig ist, dass die *id* in *<select...* ein Unikat ist, damit man die verschiedenen Listen ansprechen kann.

4 Probleme/Lösungen

1. Verbindung zur Matrix:
Wir hatten unglaublich lange dafür gebraucht, eine einfache telnet-Verbindung zur Matrix aufzubauen. Unsere Versuche endeten damit, dass wir uns aus der Matrix ausgeschlossen haben, in dem wir den Port geändert haben. Das Problem konnten wir nur durch Hilfe von Aussen lösen.

2. Matrix benötigt ein *socket_read*:
Wir hatten schon einen mehr oder weniger funktionierenden Code. Unser Problem bestand darin, dass die Matrix immer wieder bis zum Timeout gewartet hat. Die Lösung zu diesem Problem war das *socket_read (Siehe Seite 7)*.

3. PHP kann nicht ohne js mit html kommunizieren:
Zu Beginn haben wir versucht, die PHP-Datei in der html-Datei zu referenzieren. Dies war in dieser Form nicht möglich. Hier war js eine mögliche Lösung (Siehe Abb.7 Sende-Funktion).

4. Kopieren von Code:
Um auch die Toolbox mit einem Klick verwenden zu können, mussten wir die Toolbox mit einem if-Statement in die Route-Funktion integriert. Dafür haben wir einen bereits bestehenden Code kopiert. Theoretisch hätte diese Lösung funktionieren sollen. Unser Fehler war, dass wir einen Parameter falsch gekennzeichnet haben. Dies wäre nicht so einfach passiert, wenn der Code von Grund auf neu entstanden wäre. Deshalb ist es immer wichtig jeden Teil eines Codes zu verstehen und im Zweifelsfall ihn nachzuschreiben, anstatt ihn zu kopieren.

5. Zeitmanagement
Ein weiteres Problem war, dass wir uns bei gewissen Problemen zu lange aufgehalten haben. Dies hat uns viel Zeit gekostet. Eine Möglichkeit wäre: man setzt sich eine Zeitspanne von vielleicht 1h in welcher man versucht das Problem zu beheben und recherchieren. Nach Ablauf dieser Zeit, sollte man sich Hilfe holen gehen.

5 To Do

1. Einen Button integrieren, mit dem man auf die Konfigurationsseite von Genfen kommt.

2. Einen Weg finden, geänderte Konfigurationen in dem Quellcode automatisch zu ersetzen.

3. Eine Kommandozeile einführen, welche Strings entgegen nimmt und die Antwort der Matrix 1 zu 1 in einem Textfeld zurückgibt.

4. Eine Funktion finden, mit welcher man unbenutzte Bildschirme ausschalten oder abdunkeln kann.

5. Ein einfaches Login, damit nicht jeder X-Beliebige auf die Website zugreifen kann.

6 Konfiguration Matrix Stand 26.10.2015

```
<?xml version="1.0" encoding="utf-8"?>
<Gefen product="EXT-HDFST-848CPN">
	<main>
		<routing command="R" param1="1" param2="3"/>
		<routing command="R" param1="2" param2="2"/>
		<routing command="R" param1="3" param2="1"/>
		<routing command="R" param1="4" param2="2"/>
		<routing command="R" param1="5" param2="1"/>
		<routing command="R" param1="6" param2="1"/>
		<routing command="R" param1="7" param2="7"/>
		<routing command="R" param1="8" param2="1"/>
		<mask command="#MASKSTATUS" param1="00000000"/>
		<savePreset command="#SAVE_PRESET" param1="1" param2="56781171" param3="ff"/>
		<savePreset command="#SAVE_PRESET" param1="2" param2="12341171" param3="ff"/>
		<savePreset command="#SAVE_PRESET" param1="3" param2="66661171" param3="ff"/>
		<savePreset command="#SAVE_PRESET" param1="4" param2="21435678" param3="ff"/>
		<savePreset command="#SAVE_PRESET" param1="5" param2="77775678" param3="ff"/>
		<savePreset command="#SAVE_PRESET" param1="6" param2="12345678" param3="ff"/>
		<savePreset command="#SAVE_PRESET" param1="7" param2="12345678" param3="ff"/>
		<savePreset command="#SAVE_PRESET" param1="8" param2="12345678" param3="ff"/>
		<lockMatrix command="#LOCK_MATRIX" param1="0"/>
	</main>
	<iosetup>
		<presetName command="#SET_PRESET_NAME" param1="1" param2="apple_tv"/>
		<presetName command="#SET_PRESET_NAME" param1="2" param2="pc"/>
		<presetName command="#SET_PRESET_NAME" param1="3" param2="click_sh"/>
		<presetName command="#SET_PRESET_NAME" param1="4" param2="PC_Ful"/>
		<presetName command="#SET_PRESET_NAME" param1="5" param2="Preset_5"/>
		<presetName command="#SET_PRESET_NAME" param1="6" param2="Preset_6"/>
		<presetName command="#SET_PRESET_NAME" param1="7" param2="Preset_7"/>
		<presetName command="#SET_PRESET_NAME" param1="8" param2="Preset_8"/>
		<inputName command="#SET_INPUT_NAME" param1="1" param2="MIS-NW"/>
		<inputName command="#SET_INPUT_NAME" param1="2" param2="MIS-NE"/>
		<inputName command="#SET_INPUT_NAME" param1="3" param2="MIS-SW"/>
		<inputName command="#SET_INPUT_NAME" param1="4" param2="MIS-SE"/>
		<inputName command="#SET_INPUT_NAME" param1="5" param2="ATV-NW"/>
		<inputName command="#SET_INPUT_NAME" param1="6" param2="ToolBox"/>
		<inputName command="#SET_INPUT_NAME" param1="7" param2="ATV-SW"/>
		<inputName command="#SET_INPUT_NAME" param1="8" param2="ATV-SE"/>
		<outputName command="#SET_OUTPUT_NAME" param1="1" param2="TV-NW"/>
		<outputName command="#SET_OUTPUT_NAME" param1="2" param2="TV-NE"/>
		<outputName command="#SET_OUTPUT_NAME" param1="3" param2="TV-SW"/>
		<outputName command="#SET_OUTPUT_NAME" param1="4" param2="TV-SE"/>
		<outputName command="#SET_OUTPUT_NAME" param1="5" param2="OUTPUT5"/>
		<outputName command="#SET_OUTPUT_NAME" param1="6" param2="OUTPUT6"/>
		<outputName command="#SET_OUTPUT_NAME" param1="7" param2="OUTPUT7"/>
		<outputName command="#SET_OUTPUT_NAME" param1="8" param2="Console"/>
		<fst command="#HDCP" param1="1"/>
		<fst command="#HDCP" param1="2"/>
		<fst command="#HDCP" param1="3"/>
		<fst command="#HDCP" param1="4"/>
		<fst command="#HDCP" param1="5"/>
		<fst command="#HDCP" param1="6"/>
		<fst command="#HDCP" param1="7"/>
		<fst command="#HDCP" param1="8"/>
		<hdcp command="#SET_BANK_NAME" param1="00000000"/>
	</iosetup>
	<manageEDID>
		<bankName command="#LOCK_EDID" param1="1" param2="Bank 1"/>
		<bankName command="#LOCK_EDID" param1="2" param2="Bank 2"/>
		<bankName command="#LOCK_EDID" param1="3" param2="Bank 3"/>
		<bankName command="#LOCK_EDID" param1="4" param2="Bank 4"/>
		<bankName command="#LOCK_EDID" param1="5" param2="Bank 5"/>
		<bankName command="#LOCK_EDID" param1="6" param2="Bank 6"/>
		<bankName command="#LOCK_EDID" param1="7" param2="Bank 7"/>
		<bankName command="#LOCK_EDID" param1="8" param2="Bank 8"/>
		<edidLock command="#BANKEDID" param1="0"/>
		<bankEdid command="#SIPADD" param1="1" param2="ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff"/>
		<bankEdid command="#SIPADD" param1="2" param2="ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff"/>
		<bankEdid command="#SIPADD" param1="3" param2="ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff"/>
		<bankEdid command="#SIPADD" param1="4" param2="ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff"/>
		<bankEdid command="#SIPADD" param1="5" param2="ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff"/>
		<bankEdid command="#SIPADD" param1="6" param2="ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff"/>
		<bankEdid command="#SIPADD" param1="7" param2="ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff"/>
		<bankEdid command="#SIPADD" param1="8" param2="ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff"/>
	</manageEDID>
	<configuration>
		<ip command="#SNETMASK" param1="147.87.117.33"/>
		<subnet command="#SGATEWAY" param1="255.255.255.0"/>
		<gateway command="#SET_HTTP_PORT" param1="147.87.117.254"/>
		<port command="#SET_TELNET_PORT" param1="80"/>
		<telnetPort command="#SET_UDP_PORT" param1="23"/>
		<udpPort command="#SET_TELNET_PASS" param1="50007"/>
		<telnetPass command="#USE_TELNET_PASS" param1="Admin"/>
		<forcePassOnConnect command="#DISPLAY_TELNET_WELCOME" param1="0"/>
		<showLoginMessageOnConnect command="#SET_UDP_REMOTE_IP" param1="1"/>
		<remoteUdpIp command="#SET_UDP_REMOTE_PORT" param1="192.168.1.80"/>
		<remoteUdpPort command="#USE_UDP_ENABLE" param1="50008"/>
		<remoteUdpEnable command="#SET_WEBUI_OP_PASS" param1="1"/>
		<OPwebPass command="#SET_WEBUI_AD_PASS" param1="Admin"/>
		<ADwebPass command="command=" param1="Admin"/>
	</configuration>
</Gefen>
```

7 Abbildungsverzeichnis

8 Literaturverzeichnis

Hrsg. www.stackoverflow.com
Hrsg. www.php.net
Hrsg. www.w3schools.com
Hrsg. www.getbootstrap.com

9 Versionskontrolle

Version	Datum	Beschreibung	Autor
0.1	21.10.2015	Dokument erstellt	Nicola Knuchel
0.5	22.10.2015	Dokument bearbeitet	Nicola Kncuhel, Raphaël de Picciotto
0.7	27.10.2015	Dokument bearbeitet	Nicola Knuchel, Raphaël de Picciotto
1.0	28.10.2015	Dokument beendet	Nicola Knuchel, Raphaël de Picciotto