Webcam JavaScript

Config.js

```
let isFrontCamera = true; // Biến lưu trạng
thái camera (trước/sau)

//Mở camera
function ready() {
    if (document.readyState == 'complete')
{
        Webcam.set({
            width: 320,
            height: 240,
            image_format: 'jpeg',
            jpeg_quality: 90,
            facingMode: isFrontCamera ? 'us
er' : 'environment'
        });
        try {
            Webcam.attach('#camera');
        } catch (e) {
            alert(e);
        }
    }
}

//Đóng camera
function stopCamera() {
    Webcam.reset();
}


//Chụp hình
async function take_snapshot() {
    return new Promise((resolve, reject) =>
```

webcam.js

```
(function (window) {
    var _userMedia;

    // https://stackoverflow.com/questions/
783818/how-do-i-create-a-custom-error-in-ja
vascript
    function FlashError() {
        var temp = Error.apply(this, argume
nts);
        temp.name = this.name = "FlashErro
r";
        this.stack = temp.stack;
        this.message = temp.message;
    }

    function WebcamError() {
        var temp = Error.apply(this, argume
nts);
        temp.name = this.name = "WebcamErro
r";
        this.stack = temp.stack;
        this.message = temp.message;
    }

    var IntermediateInheritor = function ()
{ };
    IntermediateInheritor.prototype = Erro
r.prototype;

    FlashError.prototype = new Intermediate
Inheritor();
    WebcamError.prototype = new Intermediat
{
        Webcam.snap(function (data_uri) {
            resolve(data_uri);
        });
    });
}

//Chuyển camera
function switchCamera() {
    stopCamera(); // Dừng camera hiện tại

    isFrontCamera = !isFrontCamera; // Chuy
ển đổi giá trị của biến isFrontCamera

    // Cấu hình Webcam với camera mới
    Webcam.set({
        width: 320,
        height: 240,
        image_format: 'jpeg',
        jpeg_quality: 90,
        facingMode: isFrontCamera ? 'user'
: 'environment' // 'user' cho camera trước,
'environment' cho camera sau
    });

    try {
        Webcam.attach('#camera'); // Kết nố
i với camera mới
    } catch (e) {
        alert(e);
    }
}

//Xem thiết bị đang kết nối
navigator.mediaDevices.enumerateDevices()
    .then(function (devices) {
        devices.forEach(function (device) {
            console.log('Device ID: ', devi
ce.deviceId);
            console.log('Kind: ', device.ki
nd);
            console.log('Label: ', device.l
abel);
            console.log('------------------
------');
        });
    })
    .catch(function (err) {
        console.error('Error: ', err);
    });
```

Blazor.razor:

```
- UI:
@************************** Button Upload -
Camera  **********************@
@if (uploadedFileIds.Count < 5)
{
```

```
eInheritor();

    var Webcam = {
        version: '1.0.26',

        // globals
        protocol: location.protocol.match(/
https/i) ? 'https' : 'http',
        loaded: false,   // true when webca
m movie finishes loading
        live: false,     // true when webca
m is initialized and ready to snap
        userMedia: true, // true when getUs
erMedia is supported natively

        iOS: /iPad|iPhone|iPod/.test(naviga
tor.userAgent) && !window.MSStream,

        params: {
            width: 0,
            height: 0,
            dest_width: 0,         // size
of captured image
            dest_height: 0,        // these
default to width/height
            image_format: 'jpeg',  // image
format (may be jpeg or png)
            jpeg_quality: 90,      // jpeg
image quality from 0 (worst) to 100 (best)
            enable_flash: true,    // enabl
e flash fallback,
            force_flash: false,    // force
flash mode,
            flip_horiz: false,     // flip
image horiz (mirror mode)
            fps: 30,               // camer
a frames per second
            upload_name: 'webcam', // name
of file in upload post data
            constraints: null,     // custo
m user media constraints,
            swfURL: '',            // URI t
o webcam.swf movie (defaults to the js loca
tion)
            flashNotDetectedText: 'ERROR: N
o Adobe Flash Player detected.  Webcam.js r
elies on Flash for browsers that do not sup
port getUserMedia (like yours).',
            noInterfaceFoundText: 'No suppo
rted webcam interface found.',
            unfreeze_snap: true,    // Whet
her to unfreeze the camera after snap (defa
ults to true)
            iosPlaceholderText: 'Click here
to open camera.',
            user_callback: null,    // call
back function for snapshot (used if no user
_callback parameter given to snap function)
```

```
    <div style="padding: 0.5rem 0.1rem;">
        <Button Clicked="OpenUploadModalAsync">Attachments File <Icon Name="IconName.PlusCircle"></Icon></Button>
    </div>
}

@**************************** Modal Upload ***************************@
<Modal @ref="EditUploadModal" Closing="@EditUploadModal.CancelClosingModalWhenFocusLost">
    <ModalContent Centered="true">
        <Form id="EditImageForm">
            <ModalHeader>
                <ModalTitle>Upload file from</ModalTitle>
                <CloseButton Clicked="CloseEditUploadModalAsync" Style="font-size: medium;" />
            </ModalHeader>
            <ModalBody>
                <Row Style="margin-right: 0;
                            margin-left: 0;">
                    <Column ColumnSize="ColumnSize.Is6" Style="padding: 1.3rem;">
                        <div style="padding: 0.5rem 0.1rem;">
                            <InputFile class="custom-input-file" OnChange="@OnFileSelection"></InputFile>
                        </div>
                    </Column>
                    <Column ColumnSize="ColumnSize.Is6" Style="padding: 1.3rem;">
                        <Button Class="button-camera" Clicked="OpenCameraModalAsync">Camera <Icon Name="IconName.Camera"></Icon></Button>
                    </Column>
                </Row>
            </ModalBody>
        </Form>
    </ModalContent>
</Modal>

@******************************* Modal Camera *********************@
<Modal @ref="EditCameraModal" Closing="@EditCameraModal.CancelClosingModalWhenFocusLost" ScrollToTop="false">
    <ModalContent Centered="true">
        <Form id="EditImageForm">
            <ModalHeader>
                <ModalTitle>@nameFile</ModalTitle>
```

```
            user_canvas: null       // user provided canvas for snapshot (used if no user_canvas parameter given to snap function)
        },

        errors: {
            FlashError: FlashError,
            WebcamError: WebcamError
        },

        hooks: {}, // callback hook functions

        init: function () {
            // initialize, check for getUserMedia support
            var self = this;

            // Setup getUserMedia, with polyfill for older browsers
            // Adapted from: https://developer.mozilla.org/en-US/docs/Web/API/MediaDevices/getUserMedia
            this.mediaDevices = (navigator.mediaDevices && navigator.mediaDevices.getUserMedia) ?
                navigator.mediaDevices :
((navigator.mozGetUserMedia || navigator.webkitGetUserMedia) ? {
                    getUserMedia: function (c) {
                        return new Promise(function (y, n) {
                            (navigator.mozGetUserMedia ||
                                navigator.webkitGetUserMedia).call(navigator, c, y, n);
                        });
                    }
                } : null);

            window.URL = window.URL || window.webkitURL || window.mozURL || window.msURL;
            this.userMedia = this.userMedia && !!this.mediaDevices && !!window.URL;

            // Older versions of firefox (< 21) apparently claim support but user media does not actually work
            if (navigator.userAgent.match(/Firefox\D+(\d+)/)) {
                if (parseInt(RegExp.$1, 10) < 21) this.userMedia = null;
            }
            if (this.userMedia) {
                window.addEventListener('be
                <CloseButton Clicked="Close
EditCameraModalAsync" Style="font-size: med
ium;" />
            </ModalHeader>
            <ModalBody>
                <Column Style="padding: 1.3
rem;">
                    <HQSOFTCamera Capture
="Capture"></HQSOFTCamera>
                    <Text TextAlignment="Te
xtAlignment.Center">Total: @cameraList.Coun
t() </Text>
                </Column>
                <Column>
                    <div id="results">
                        <Button Size="Size.
Medium" Color="Color.Danger" @onclick="Clea
rImage">Clear</Button>
                        @if (cameraList.Cou
nt > 0)
                        {
                            @foreach (var i
tem in cameraList)
                            {
                                <img src="@
item" alt="Alternate Text" width="30" heigh
t="30"
                                    style
="padding: 2px; width: fit-content;" />
                            }
                        }
                    </div>
                </Column>
                <Column>
                    <p>Status Camera: @GetC
ameraState()</p>
                    <Button Size="Size.Medi
um" Color="Color.Warning" @onclick="SwitchC
amera">Switch Camera</Button>
                    @* <Button Size="Size.M
edium" Color="Color.Secondary">Preview</But
ton> *@
                    <Button Size="Size.Medi
um" Color="Color.Success" @onclick="Captur
e">Take Photo</Button>
                    <Button Size="Size.Medi
um" Color="Color.Primary" @onclick="SaveToD
atabase">Save & Submit</Button>
                </Column>
            </ModalBody>
        </Form>
    </ModalContent>
</Modal>


- Backend:
#region Open & Close Modal
        private async Task OpenShowImageMod
```

```
foreunload', function (event) {
                    self.reset();
                });
            }
        },

        exifOrientation: function (binFile)
{
            var dataView = new DataView(bin
File);
            if ((dataView.getUint8(0) != 0x
FF) || (dataView.getUint8(1) != 0xD8)) {
                console.log('Not a valid JP
EG file');
                return 0;
            }
            var offset = 2;
            var marker = null;
            while (offset < binFile.byteLen
gth) {
                // find 0xFFE1 (225 marker)
                if (dataView.getUint8(offse
t) != 0xFF) {
                    console.log('Not a vali
d marker at offset ' + offset + ', found: '
+ dataView.getUint8(offset));
                    return 0;
                }
                marker = dataView.getUint8
(offset + 1);
                if (marker == 225) {
                    offset += 4;
                    var str = "";
                    for (n = 0; n < 4; n++)
{
                        str += String.fromC
harCode(dataView.getUint8(offset + n));
                    }
                    if (str != 'Exif') {
                        console.log('Not va
lid EXIF data found');
                        return 0;
                    }

                    offset += 6; // tiffOff
set
                    var bigEnd = null;

                    // test for TIFF validi
ty and endianness
                    if (dataView.getUint16
(offset) == 0x4949) {
                        bigEnd = false;
                    } else if (dataView.get
Uint16(offset) == 0x4D4D) {
                        bigEnd = true;
                    } else {
                        console.log("Not va
alAsync(string fileName)
        {
            nameFile = fileName;
            base64Image = ""; // Reset base
64Image to empty string
            base64Txt = ""; // Reset base64
Image to empty string
            fileExtension = Path.GetExtensi
on(nameFile); // Get the file extension
            await GetContents();
            await GetImages();
            await EditImageModal.Show();
        }
        private async Task OpenUploadModalA
sync()
        {
            await EditUploadModal.Show();
        }
        private async Task OpenCameraModalA
sync()
        {
            await JSRuntime.InvokeVoidAsync
("ready");
            await EditCameraModal.Show();
        }

        private async Task CloseEditImageMo
dalAsync()
        {
            await EditImageModal.Hide();
        }
        private async Task CloseEditUploadM
odalAsync()
        {
            await EditUploadModal.Hide();
        }
        private async Task CloseEditCameraM
odalAsync()
        {
            await JSRuntime.InvokeVoidAsync
("stopCamera");
            await EditCameraModal.Hide();
        }
        #endregion

        #region Camera
        private async Task Capture()
        {
            var maxFilesToShow = 5;
            var numImagesToSave = maxFilesT
oShow - cameraList.Count;

            if (numImagesToSave > 0)
            {
                var capturedImage = await J
SRuntime.InvokeAsync<string>("take_snapsho
t");
                cameraList.Add(capturedImag
```

```
lid TIFF data! (no 0x4949 or 0x4D4D)");
                        return 0;
                    }

                    if (dataView.getUint16
(offset + 2, !bigEnd) != 0x002A) {
                        console.log("Not va
lid TIFF data! (no 0x002A)");
                        return 0;
                    }

                    var firstIFDOffset = da
taView.getUint32(offset + 4, !bigEnd);
                    if (firstIFDOffset < 0x
00000008) {
                        console.log("Not va
lid TIFF data! (First offset less than 8)",
dataView.getUint32(offset + 4, !bigEnd));
                        return 0;
                    }

                    // extract orientation
data
                    var dataStart = offset
+ firstIFDOffset;
                    var entries = dataView.
getUint16(dataStart, !bigEnd);
                    for (var i = 0; i < ent
ries; i++) {
                        var entryOffset = d
ataStart + i * 12 + 2;
                        if (dataView.getUin
t16(entryOffset, !bigEnd) == 0x0112) {
                            var valueType =
dataView.getUint16(entryOffset + 2, !bigEn
d);
                            var numValues =
dataView.getUint32(entryOffset + 4, !bigEn
d);
                            if (valueType !
= 3 && numValues != 1) {
                                console.log
('Invalid EXIF orientation value type (' +
valueType + ') or count (' + numValues +
')');
                                return 0;
                            }
                            var value = dat
aView.getUint16(entryOffset + 8, !bigEnd);
                            if (value < 1 |
| value > 8) {
                                console.log
('Invalid EXIF orientation value (' + value
+ ')');
                                return 0;
                            }
                            return value;
                        }
```

```
e);
                StateHasChanged();
            }
            else
            {
                await _uiMessageService.Err
or(L["Notification:MaximumFile"]);
            }
        }
        private async Task SwitchCamera()
        {
            await JSRuntime.InvokeVoidAsync
("switchCamera");
            isFrontCamera = !isFrontCamera;
            StateHasChanged();
        }
        private string GetCameraState()
        {
            return isFrontCamera ? "Front C
amera" : "Back Camera";
        }
        public async Task ClearImage()
        {
            cameraList.Clear();
            StateHasChanged();
        }
        private bool IsBase64String(string
value)
        {
            try
            {
                Convert.FromBase64String(va
lue);
                return true;
            }
            catch (FormatException)
            {
                return false;
            }
        }
        private async Task SaveToDatabase()
        {
            var maxFilesToShow = 5;
            var numImagesToSave = maxFilesT
oShow - cameraList.Count;

            if (cameraList.Count <= numImag
esToSave)
            {
                foreach (var image in camer
aList)
                {
                    try
                    {
                        // Extract the Base
64 string from the image data
                        string base64String
= image.Substring(image.IndexOf(',') + 1);
```

```
                    }
                } else {
                    offset += 2 + dataView.
getUint16(offset + 2);
                }
            }
            return 0;
        },

        fixOrientation: function (origObjUR
L, orientation, targetImg) {
            // fix image orientation based
on exif orientation data
            // exif orientation information
            //    http://www.impulseadventu
re.com/photo/exif-orientation.html
            //    link source wikipedia (ht
tps://en.wikipedia.org/wiki/Exif#cite_note-
20)
            var img = new Image();
            img.addEventListener('load', fu
nction (event) {
                var canvas = document.creat
eElement('canvas');
                var ctx = canvas.getContext
('2d');

                // switch width height if o
rientation needed
                if (orientation < 5) {
                    canvas.width = img.widt
h;
                    canvas.height = img.hei
ght;
                } else {
                    canvas.width = img.heig
ht;
                    canvas.height = img.wid
th;
                }

                // transform (rotate) image
- see link at beginning this method
                switch (orientation) {
                    case 2: ctx.transform(-
1, 0, 0, 1, img.width, 0); break;
                    case 3: ctx.transform(-
1, 0, 0, -1, img.width, img.height); break;
                    case 4: ctx.transform
(1, 0, 0, -1, 0, img.height); break;
                    case 5: ctx.transform
(0, 1, 1, 0, 0, 0); break;
                    case 6: ctx.transform
(0, 1, -1, 0, img.height, 0); break;
                    case 7: ctx.transform
(0, -1, -1, 0, img.height, img.width); brea
k;
                    case 8: ctx.transform
```

```
                        // Check if the cap
tured image string is a valid Base64 string
                        if (IsBase64String
(base64String))
                        {
                            // Convert the
captured image to a byte array
                            byte[] imageByt
es = Convert.FromBase64String(base64Strin
g);
                            string randomFi
leName = GenerateRandomFileName();
                            filePath = rand
omFileName + ".jpeg";
                            using (var stre
am = new MemoryStream(imageBytes))
                            {
                                var editing
File = new CreateFileInputWithStream
                                {
                                    Name =
filePath,
                                    File =
new RemoteStreamContent(
                                        str
eam: stream,
                                        fil
eName: filePath,
                                        con
tentType: "image/jpeg",
                                        dis
poseStream: true
                                        )
                                };
                                editingFil
e.SetProperty("Url", DocUrl);
                                editingFil
e.SetProperty("DocId", DocId);
                                var uploade
dFile = await FileDescriptorAppService.Crea
teAsync(directoryId, editingFile);
                                await GetFi
leContent();
                                await Notif
y.Success(L["Notification:Save"]);
                                await Close
EditCameraModalAsync();
                                await Close
EditUploadModalAsync();
                                StateHasCha
nged();
                            }
                        }
                        else
                        {
                            Console.WriteLi
ne("Invalid Base64 string");
```

```
(0, -1, 1, 0, 0, img.width); break;
                }

                ctx.drawImage(img, 0, 0);
                // pass rotated image data
to the target image container
                targetImg.src = canvas.toDa
taURL();
            }, false);
            // start transformation by load
event
            img.src = origObjURL;
        },

        attach: function (elem) {
            // create webcam preview and at
tach to DOM element
            // pass in actual DOM referenc
e, ID, or CSS selector
            if (typeof (elem) == 'string')
{
                elem = document.getElementB
yId(elem) || document.querySelector(elem);
            }
            if (!elem) {
                return this.dispatch('erro
r', new WebcamError("Could not locate DOM e
lement to attach to."));
            }
            this.container = elem;
            elem.innerHTML = ''; // start w
ith empty element

            // insert "peg" so we can inser
t our preview canvas adjacent to it later o
n
            var peg = document.createElemen
t('div');
            elem.appendChild(peg);
            this.peg = peg;

            // set width/height if not alre
ady set
            if (!this.params.width) this.pa
rams.width = elem.offsetWidth;
            if (!this.params.height) this.p
arams.height = elem.offsetHeight;

            // make sure we have a nonzero
width and height at this point
            if (!this.params.width || !thi
s.params.height) {
                return this.dispatch('erro
r', new WebcamError("No width and/or height
for webcam.  Please call set() first, or at
tach to a visible element."));
            }
```

```
                        }
                    }
                    catch (FormatException
ex)
                    {
                        Console.WriteLine
($"Error decoding Base64 string: {ex.Messag
e}");
                    }
                }
                cameraList.Clear();
                StateHasChanged();
            }
            else
            {
                await _uiMessageService.Err
or(L["Notification:MaximumFile"]);
            }
        }
```

```
            // set defaults for dest_width
/ dest_height if not set
            if (!this.params.dest_width) th
is.params.dest_width = this.params.width;
            if (!this.params.dest_height) t
his.params.dest_height = this.params.heigh
t;

            this.userMedia = _userMedia ===
undefined ? this.userMedia : _userMedia;
            // if force_flash is set, disab
le userMedia
            if (this.params.force_flash) {
                _userMedia = this.userMedi
a;
                this.userMedia = null;
            }

            // check for default fps
            if (typeof this.params.fps !==
"number") this.params.fps = 30;

            // adjust scale if dest_width o
r dest_height is different
            var scaleX = this.params.width
/ this.params.dest_width;
            var scaleY = this.params.height
/ this.params.dest_height;

            if (this.userMedia) {
                // setup webcam video conta
iner
                var video = document.create
Element('video');
                video.setAttribute('autopla
y', 'autoplay');
                video.setAttribute('playsin
line', 'playsinline');
                video.style.width = '' + th
is.params.dest_width + 'px';
                video.style.height = '' + t
his.params.dest_height + 'px';

                if ((scaleX != 1.0) || (sca
leY != 1.0)) {
                    elem.style.overflow =
'hidden';
                    video.style.webkitTrans
formOrigin = '0px 0px';
                    video.style.mozTransfor
mOrigin = '0px 0px';
                    video.style.msTransform
Origin = '0px 0px';
                    video.style.oTransformO
rigin = '0px 0px';
                    video.style.transformOr
igin = '0px 0px';
                    video.style.webkitTrans
```

```
form = 'scaleX(' + scaleX + ') scaleY(' + s
caleY + ')';
                    video.style.mozTransfor
m = 'scaleX(' + scaleX + ') scaleY(' + scal
eY + ')';
                    video.style.msTransform
= 'scaleX(' + scaleX + ') scaleY(' + scaleY
+ ')';
                    video.style.oTransform
= 'scaleX(' + scaleX + ') scaleY(' + scaleY
+ ')';
                    video.style.transform =
'scaleX(' + scaleX + ') scaleY(' + scaleY +
')';
                }

                // add video element to dom
                elem.appendChild(video);
                this.video = video;

                // ask user for access to t
heir camera
                var self = this;
                this.mediaDevices.getUserMe
dia({
                    "audio": false,
                    "video": this.params.co
nstraints || {
                        mandatory: {
                            minWidth: this.
params.dest_width,
                            minHeight: thi
s.params.dest_height
                        }
                    }
                })
                    .then(function (stream)
{
                        // got access, atta
ch stream to video
                        video.onloadedmetad
ata = function (e) {
                            self.stream = s
tream;
                            self.loaded = t
rue;
                            self.live = tru
e;
                            self.dispatch
('load');
                            self.dispatch
('live');
                            self.flip();
                        };
                        // as window.URL.cr
eateObjectURL() is deprecated, adding a che
ck so that it works in Safari.
                        // older browsers m
ay not have srcObject
                        if ("srcObject" in
video) {
                            video.srcObject
= stream;
                        }
                        else {
                            // using URL.cr
eateObjectURL() as fallback for old browser
s
                            video.src = win
dow.URL.createObjectURL(stream);
                        }
                    })
                    .catch(function (err) {
                        // JH 2016-07-31 In
stead of dispatching error, now falling bac
k to Flash if userMedia fails (thx @john201
4)
                        // JH 2016-08-07 Bu
t only if flash is actually installed -- if
not, dispatch error here and now.
                        if (self.params.ena
ble_flash && self.detectFlash()) {
                            setTimeout(func
tion () { self.params.force_flash = 1; sel
f.attach(elem); }, 1);
                        }
                        else {
                            self.dispatch
('error', err);
                        }
                    });
            }
            else if (this.iOS) {
                // prepare HTML elements
                var div = document.createEl
ement('div');
                div.id = this.container.id
+ '-ios_div';
                div.className = 'webcamjs-i
os-placeholder';
                div.style.width = '' + thi
s.params.width + 'px';
                div.style.height = '' + thi
s.params.height + 'px';
                div.style.textAlign = 'cent
er';
                div.style.display = 'table-
cell';
                div.style.verticalAlign =
'middle';
                div.style.backgroundRepeat
= 'no-repeat';
                div.style.backgroundSize =
'contain';
                div.style.backgroundPositio
n = 'center';
```

```
                var span = document.createE
lement('span');
                span.className = 'webcamjs-
ios-text';
                span.innerHTML = this.param
s.iosPlaceholderText;
                div.appendChild(span);
                var img = document.createEl
ement('img');
                img.id = this.container.id 
+ '-ios_img';
                img.style.width = '' + thi
s.params.dest_width + 'px';
                img.style.height = '' + thi
s.params.dest_height + 'px';
                img.style.display = 'none';
                div.appendChild(img);
                var input = document.create
Element('input');
                input.id = this.container.i
d + '-ios_input';
                input.setAttribute('type', 
'file');
                input.setAttribute('accep
t', 'image/*');
                input.setAttribute('captur
e', 'camera');

                var self = this;
                var params = this.params;
                // add input listener to lo
ad the selected image
                input.addEventListener('cha
nge', function (event) {
                    if (event.target.files.
length > 0 && event.target.files[0].type.in
dexOf('image/') == 0) {
                        var objURL = URL.cr
eateObjectURL(event.target.files[0]);

                        // load image with 
auto scale and crop
                        var image = new Ima
ge();
                        image.addEventListe
ner('load', function (event) {
                            var canvas = do
cument.createElement('canvas');
                            canvas.width = 
params.dest_width;
                            canvas.height =
params.dest_height;
                            var ctx = canva
s.getContext('2d');

                            // crop and sca
le image for final size
                            ratio = Math.mi
```

```
n(image.width / params.dest_width, image.he
ight / params.dest_height);
                            var sw = param
s.dest_width * ratio;
                            var sh = param
s.dest_height * ratio;
                            var sx = (imag
e.width - sw) / 2;
                            var sy = (imag
e.height - sh) / 2;
                            ctx.drawImage(i
mage, sx, sy, sw, sh, 0, 0, params.dest_wid
th, params.dest_height);

                            var dataURL = c
anvas.toDataURL();
                            img.src = dataU
RL;
                            div.style.backg
roundImage = "url('" + dataURL + "')";
                        }, false);

                        // read EXIF data
                        var fileReader = ne
w FileReader();
                        fileReader.addEvent
Listener('load', function (e) {
                            var orientation
= self.exifOrientation(e.target.result);
                            if (orientation
> 1) {
                                // image ne
ed to rotate (see comments on fixOrientatio
n method for more information)
                                // transfor
m image and load to image object
                                self.fixOri
entation(objURL, orientation, image);
                            } else {
                                // load ima
ge data to image object
                                image.src =
objURL;
                            }
                        }, false);

                        // Convert image da
ta to blob format
                        var http = new XMLH
ttpRequest();
                        http.open("GET", ob
jURL, true);
                        http.responseType =
"blob";
                        http.onload = funct
ion (e) {
                            if (this.status
== 200 || this.status === 0) {
```

```
                                fileReader.
readAsArrayBuffer(this.response);
                            }
                        };
                        http.send();

                    }
                }, false);
                input.style.display = 'non
e';
                elem.appendChild(input);
                // make div clickable for o
pen camera interface
                div.addEventListener('clic
k', function (event) {
                    if (params.user_callbac
k) {
                        // global user_call
back defined - create the snapshot
                        self.snap(params.us
er_callback, params.user_canvas);
                    } else {
                        // no global callba
ck definied for snapshot, load image and wa
it for external snap method call
                        input.style.display
= 'block';
                        input.focus();
                        input.click();
                        input.style.display
= 'none';
                    }
                }, false);
                elem.appendChild(div);
                this.loaded = true;
                this.live = true;
            }
            else if (this.params.enable_fla
sh && this.detectFlash()) {
                // flash fallback
                window.Webcam = Webcam; //
needed for flash-to-js interface
                var div = document.createEl
ement('div');
                div.innerHTML = this.getSWF
HTML();
                elem.appendChild(div);
            }
            else {
                this.dispatch('error', new
WebcamError(this.params.noInterfaceFoundTex
t));
            }

            // setup final crop for live pr
eview
            if (this.params.crop_width && t
his.params.crop_height) {
```

```
                var scaled_crop_width = Mat
h.floor(this.params.crop_width * scaleX);
                var scaled_crop_height = Ma
th.floor(this.params.crop_height * scaleY);

                elem.style.width = '' + sca
led_crop_width + 'px';
                elem.style.height = '' + sc
aled_crop_height + 'px';
                elem.style.overflow = 'hidd
en';

                elem.scrollLeft = Math.floo
r((this.params.width / 2) - (scaled_crop_wi
dth / 2));
                elem.scrollTop = Math.floor
((this.params.height / 2) - (scaled_crop_he
ight / 2));
            }
            else {
                // no crop, set size to des
ired
                elem.style.width = '' + thi
s.params.width + 'px';
                elem.style.height = '' + th
is.params.height + 'px';
            }
        },

        reset: function () {
            // shutdown camera, reset to po
tentially attach again
            if (this.preview_active) this.u
nfreeze();

            // attempt to fix issue #64
            this.unflip();

            if (this.userMedia) {
                if (this.stream) {
                    if (this.stream.getVide
oTracks) {
                        // get video track
to call stop on it
                        var tracks = this.s
tream.getVideoTracks();
                        if (tracks && track
s[0] && tracks[0].stop) tracks[0].stop();
                    }
                    else if (this.stream.st
op) {
                        // deprecated, may
be removed in future
                        this.stream.stop();
                    }
                }
                delete this.stream;
                delete this.video;
            }

            if ((this.userMedia !== true) &
& this.loaded && !this.iOS) {
                // call for turn off camera
in flash
                var movie = this.getMovie
();
                if (movie && movie._release
Camera) movie._releaseCamera();
            }

            if (this.container) {
                this.container.innerHTML =
'';
                delete this.container;
            }

            this.loaded = false;
            this.live = false;
        },

        set: function () {
            // set one or more params
            // variable argument list: 1 pa
ram = hash, 2 params = key, value
            if (arguments.length == 1) {
                for (var key in arguments
[0]) {
                    this.params[key] = argu
ments[0][key];
                }
            }
            else {
                this.params[arguments[0]] =
arguments[1];
            }
        },

        on: function (name, callback) {
            // set callback hook
            name = name.replace(/^on/i,
'').toLowerCase();
            if (!this.hooks[name]) this.hoo
ks[name] = [];
            this.hooks[name].push(callbac
k);
        },

        off: function (name, callback) {
            // remove callback hook
            name = name.replace(/^on/i,
'').toLowerCase();
            if (this.hooks[name]) {
                if (callback) {
                    // remove one selected
callback from list
                    var idx = this.hooks[na
me].indexOf(callback);
                    if (idx > -1) this.hook
s[name].splice(idx, 1);
                }
                else {
                    // no callback specifie
d, so clear all
                    this.hooks[name] = [];
                }
            }
        },

        dispatch: function () {
            // fire hook callback, passing
optional value to it
            var name = arguments[0].replace
(/^on/i, '').toLowerCase();
            var args = Array.prototype.slic
e.call(arguments, 1);

            if (this.hooks[name] && this.ho
oks[name].length) {
                for (var idx = 0, len = thi
s.hooks[name].length; idx < len; idx++) {
                    var hook = this.hooks[n
ame][idx];

                    if (typeof (hook) == 'f
unction') {
                        // callback is func
tion reference, call directly
                        hook.apply(this, ar
gs);
                    }
                    else if ((typeof (hook)
== 'object') && (hook.length == 2)) {
                        // callback is PHP-
style object instance method
                        hook[0][hook[1]].ap
ply(hook[0], args);
                    }
                    else if (window[hook])
{
                        // callback is glob
al function name
                        window[hook].apply
(window, args);
                    }
                } // loop
                return true;
            }
            else if (name == 'error') {
                var message;
                if ((args[0] instanceof Fla
shError) || (args[0] instanceof WebcamErro
r)) {
                    message = args[0].messa
ge;
```

```
                } else {
                    message = "Could not ac
cess webcam: " + args[0].name + ": " +
                        args[0].message + "
" + args[0].toString();
                }

                // default error handler if
no custom one specified
                alert("Webcam.js Error: " +
message);
            }

            return false; // no hook define
d
        },

        setSWFLocation: function (value) {
            // for backward compatibility.
            this.set('swfURL', value);
        },

        detectFlash: function () {
            // return true if browser suppo
rts flash, false otherwise
            // Code snippet borrowed from:
https://github.com/swfobject/swfobject
            var SHOCKWAVE_FLASH = "Shockwav
e Flash",
                SHOCKWAVE_FLASH_AX = "Shock
waveFlash.ShockwaveFlash",
                FLASH_MIME_TYPE = "applicat
ion/x-shockwave-flash",
                win = window,
                nav = navigator,
                hasFlash = false;

            if (typeof nav.plugins !== "und
efined" && typeof nav.plugins[SHOCKWAVE_FLA
SH] === "object") {
                var desc = nav.plugins[SHOC
KWAVE_FLASH].description;
                if (desc && (typeof nav.mim
eTypes !== "undefined" && nav.mimeTypes[FLA
SH_MIME_TYPE] && nav.mimeTypes[FLASH_MIME_T
YPE].enabledPlugin)) {
                    hasFlash = true;
                }
            }
            else if (typeof win.ActiveXObje
ct !== "undefined") {
                try {
                    var ax = new ActiveXObj
ect(SHOCKWAVE_FLASH_AX);
                    if (ax) {
                        var ver = ax.GetVar
iable("$version");
                        if (ver) hasFlash =
true;
                    }
                }
                catch (e) { ; }
            }

            return hasFlash;
        },

        getSWFHTML: function () {
            // Return HTML for embedding fl
ash based webcam capture movie
            var html = '',
                swfURL = this.params.swfUR
L;

            // make sure we aren't running
locally (flash doesn't work)
            if (location.protocol.match(/fi
le/)) {
                this.dispatch('error', new
FlashError("Flash does not work from local
disk.  Please run from a web server."));
                return '<h3 style="color:re
d">ERROR: the Webcam.js Flash fallback does
not work from local disk.  Please run it fr
om a web server.</h3>';
            }

            // make sure we have flash
            if (!this.detectFlash()) {
                this.dispatch('error', new
FlashError("Adobe Flash Player not found.
Please install from get.adobe.com/flashplay
er and try again."));
                return '<h3 style="color:re
d">' + this.params.flashNotDetectedText +
'</h3>';
            }

            // set default swfURL if not ex
plicitly set
            if (!swfURL) {
                // find our script tag, and
use that base URL
                var base_url = '';
                var scpts = document.getEle
mentsByTagName('script');
                for (var idx = 0, len = scp
ts.length; idx < len; idx++) {
                    var src = scpts[idx].ge
tAttribute('src');
                    if (src && src.match(/
\/webcam(\.min)?\.js/)) {
                        base_url = src.repl
ace(/\/webcam(\.min)?\.js.*$/, '');
                        idx = len;
                    }
                }
                if (base_url) swfURL = base_url + '/webcam.swf';
                else swfURL = 'webcam.swf';
            }

            // if this is the user's first visit, set flashvar so flash privacy settings panel is shown first
            if (window.localStorage && !localStorage.getItem('visited')) {
                this.params.new_user = 1;
                localStorage.setItem('visited', 1);
            }

            // construct flashvars string
            var flashvars = '';
            for (var key in this.params) {
                if (flashvars) flashvars += '&';
                flashvars += key + '=' + escape(this.params[key]);
            }

            // construct object/embed tag
            html += '<object classid="clsid:d27cdb6e-ae6d-11cf-96b8-444553540000" type="application/x-shockwave-flash" codebase="' + this.protocol + '://download.macromedia.com/pub/shockwave/cabs/flash/swflash.cab#version=9,0,0,0" width="' + this.params.width + '" height="' + this.params.height + '" id="webcam_movie_obj" align="middle"><param name="wmode" value="opaque" /><param name="allowScriptAccess" value="always" /><param name="allowFullScreen" value="false" /><param name="movie" value="' + swfURL + '" /><param name="loop" value="false" /><param name="menu" value="false" /><param name="quality" value="best" /><param name="bgcolor" value="#ffffff" /><param name="flashvars" value="' + flashvars + '"/><embed id="webcam_movie_embed" src="' + swfURL + '" wmode="opaque" loop="false" menu="false" quality="best" bgcolor="#ffffff" width="' + this.params.width + '" height="' + this.params.height + '" name="webcam_movie_embed" align="middle" allowScriptAccess="always" allowFullScreen="false" type="application/x-shockwave-flash" pluginspage="http://www.macromedia.com/go/getflashplayer" flashvars="' + flashvars + '"></embed></object>';

            return html;
        },

        getMovie: function () {
            // get reference to movie objec
t/embed in DOM
            if (!this.loaded) return this.d
ispatch('error', new FlashError("Flash Movi
e is not loaded yet"));
            var movie = document.getElement
ById('webcam_movie_obj');
            if (!movie || !movie._snap) mov
ie = document.getElementById('webcam_movie_
embed');
            if (!movie) this.dispatch('erro
r', new FlashError("Cannot locate Flash mov
ie in DOM"));
            return movie;
        },

        freeze: function () {
            // show preview, freeze camera
            var self = this;
            var params = this.params;

            // kill preview if already acti
ve
            if (this.preview_active) this.u
nfreeze();

            // determine scale factor
            var scaleX = this.params.width
/ this.params.dest_width;
            var scaleY = this.params.height
/ this.params.dest_height;

            // must unflip container as pre
view canvas will be pre-flipped
            this.unflip();

            // calc final size of image
            var final_width = params.crop_w
idth || params.dest_width;
            var final_height = params.crop_
height || params.dest_height;

            // create canvas for holding pr
eview
            var preview_canvas = document.c
reateElement('canvas');
            preview_canvas.width = final_wi
dth;
            preview_canvas.height = final_h
eight;
            var preview_context = preview_c
anvas.getContext('2d');

            // save for later use
            this.preview_canvas = preview_c
anvas;
            this.preview_context = preview_
context;
```

```
            // scale for preview size
            if ((scaleX != 1.0) || (scaleY
!= 1.0)) {
                preview_canvas.style.webkit
TransformOrigin = '0px 0px';
                preview_canvas.style.mozTra
nsformOrigin = '0px 0px';
                preview_canvas.style.msTran
sformOrigin = '0px 0px';
                preview_canvas.style.oTrans
formOrigin = '0px 0px';
                preview_canvas.style.transf
ormOrigin = '0px 0px';
                preview_canvas.style.webkit
Transform = 'scaleX(' + scaleX + ') scaleY
(' + scaleY + ')';
                preview_canvas.style.mozTra
nsform = 'scaleX(' + scaleX + ') scaleY(' +
scaleY + ')';
                preview_canvas.style.msTran
sform = 'scaleX(' + scaleX + ') scaleY(' +
scaleY + ')';
                preview_canvas.style.oTrans
form = 'scaleX(' + scaleX + ') scaleY(' + s
caleY + ')';
                preview_canvas.style.transf
orm = 'scaleX(' + scaleX + ') scaleY(' + sc
aleY + ')';
            }

            // take snapshot, but fire our
own callback
            this.snap(function () {
                // add preview image to do
m, adjust for crop
                preview_canvas.style.positi
on = 'relative';
                preview_canvas.style.left =
'' + self.container.scrollLeft + 'px';
                preview_canvas.style.top =
'' + self.container.scrollTop + 'px';

                self.container.insertBefore
(preview_canvas, self.peg);
                self.container.style.overfl
ow = 'hidden';

                // set flag for user captur
e (use preview)
                self.preview_active = true;

            }, preview_canvas);
        },

        unfreeze: function () {
            // cancel preview and resume li
ve video feed
```

```
            if (this.preview_active) {
                // remove preview canvas
                this.container.removeChild
(this.preview_canvas);
                delete this.preview_contex
t;
                delete this.preview_canvas;

                // unflag
                this.preview_active = fals
e;

                // re-flip if we unflipped
before
                this.flip();
            }
        },

        flip: function () {
            // flip container horiz (mirror
mode) if desired
            if (this.params.flip_horiz) {
                var sty = this.container.st
yle;
                sty.webkitTransform = 'scal
eX(-1)';
                sty.mozTransform = 'scaleX
(-1)';
                sty.msTransform = 'scaleX(-
1)';
                sty.oTransform = 'scaleX(-
1)';
                sty.transform = 'scaleX(-
1)';
                sty.filter = 'FlipH';
                sty.msFilter = 'FlipH';
            }
        },

        unflip: function () {
            // unflip container horiz (mirr
or mode) if desired
            if (this.params.flip_horiz) {
                var sty = this.container.st
yle;
                sty.webkitTransform = 'scal
eX(1)';
                sty.mozTransform = 'scaleX
(1)';
                sty.msTransform = 'scaleX
(1)';
                sty.oTransform = 'scaleX
(1)';
                sty.transform = 'scaleX
(1)';
                sty.filter = '';
                sty.msFilter = '';
            }
        },

        savePreview: function (user_callbac
k, user_canvas) {
            // save preview freeze and fire
user callback
            var params = this.params;
            var canvas = this.preview_canva
s;
            var context = this.preview_cont
ext;

            // render to user canvas if des
ired
            if (user_canvas) {
                var user_context = user_can
vas.getContext('2d');
                user_context.drawImage(canv
as, 0, 0);
            }

            // fire user callback if desire
d
            user_callback(
                user_canvas ? null : canva
s.toDataURL('image/' + params.image_format,
params.jpeg_quality / 100),
                canvas,
                context
            );

            // remove preview
            if (this.params.unfreeze_snap)
this.unfreeze();
        },

        snap: function (user_callback, user
_canvas) {
            // use global callback and canv
as if not defined as parameter
            if (!user_callback) user_callba
ck = this.params.user_callback;
            if (!user_canvas) user_canvas =
this.params.user_canvas;

            // take snapshot and return ima
ge data uri
            var self = this;
            var params = this.params;

            if (!this.loaded) return this.d
ispatch('error', new WebcamError("Webcam is
not loaded yet"));
            // if (!this.live) return this.
dispatch('error', new WebcamError("Webcam i
s not live yet"));
            if (!user_callback) return thi
s.dispatch('error', new WebcamError("Please
provide a callback function or canvas to sn
ap()"));

            // if we have an active preview
freeze, use that
            if (this.preview_active) {
                this.savePreview(user_callb
ack, user_canvas);
                return null;
            }

            // create offscreen canvas elem
ent to hold pixels
            var canvas = document.createEle
ment('canvas');
            canvas.width = this.params.dest
_width;
            canvas.height = this.params.des
t_height;
            var context = canvas.getContext
('2d');

            // flip canvas horizontally if
desired
            if (this.params.flip_horiz) {
                context.translate(params.de
st_width, 0);
                context.scale(-1, 1);
            }

            // create inline function, call
ed after image load (flash) or immediately
(native)
            var func = function () {
                // render image if needed
(flash)
                if (this.src && this.width
&& this.height) {
                    context.drawImage(this,
0, 0, params.dest_width, params.dest_heigh
t);
                }

                // crop if desired
                if (params.crop_width && pa
rams.crop_height) {
                    var crop_canvas = docum
ent.createElement('canvas');
                    crop_canvas.width = par
ams.crop_width;
                    crop_canvas.height = pa
rams.crop_height;
                    var crop_context = crop
_canvas.getContext('2d');

                    crop_context.drawImage
(canvas,
                        Math.floor((params.
dest_width / 2) - (params.crop_width / 2)),
                        Math.floor((params.
dest_height / 2) - (params.crop_height /
2)),
                        params.crop_width,
                        params.crop_height,
                        0,
                        0,
                        params.crop_width,
                        params.crop_height
                    );

                    // swap canvases
                    context = crop_context;
                    canvas = crop_canvas;
                }

                // render to user canvas if
desired
                if (user_canvas) {
                    var user_context = user
_canvas.getContext('2d');
                    user_context.drawImage
(canvas, 0, 0);
                }

                // fire user callback if de
sired
                user_callback(
                    user_canvas ? null : ca
nvas.toDataURL('image/' + params.image_form
at, params.jpeg_quality / 100),
                    canvas,
                    context
                );
            };

            // grab image frame from userMe
dia or flash movie
            if (this.userMedia) {
                // native implementation
                context.drawImage(this.vide
o, 0, 0, this.params.dest_width, this.param
s.dest_height);

                // fire callback right away
                func();
            }
            else if (this.iOS) {
                var div = document.getEleme
ntById(this.container.id + '-ios_div');
                var img = document.getEleme
ntById(this.container.id + '-ios_img');
                var input = document.getEle
mentById(this.container.id + '-ios_input');
                // function for handle snap
shot event (call user_callback and reset th
e interface)
```

```
                iFunc = function (event) {
                    func.call(img);
                    img.removeEventListener
('load', iFunc);
                    div.style.backgroundIma
ge = 'none';
                    img.removeAttribute('sr
c');
                    input.value = null;
                };
                if (!input.value) {
                    // No image selected ye
t, activate input field
                    img.addEventListener('l
oad', iFunc);
                    input.style.display =
'block';
                    input.focus();
                    input.click();
                    input.style.display =
'none';
                } else {
                    // Image already select
ed
                    iFunc(null);
                }
            }
            else {
                // flash fallback
                var raw_data = this.getMovi
e()._snap();

                // render to image, fire ca
llback when complete
                var img = new Image();
                img.onload = func;
                img.src = 'data:image/' + t
his.params.image_format + ';base64,' + raw_
data;
            }

            return null;
        },

        configure: function (panel) {
            // open flash configuration pan
el -- specify tab name:
            // "camera", "privacy", "defaul
t", "localStorage", "microphone", "settings
Manager"
            if (!panel) panel = "camera";
            this.getMovie()._configure(pane
l);
        },

        flashNotify: function (type, msg) {
            // receive notification from fl
ash about event
```

```
            switch (type) {
                case 'flashLoadComplete':
                    // movie loaded success
fully
                    this.loaded = true;
                    this.dispatch('load');
                    break;

                case 'cameraLive':
                    // camera is live and r
eady to snap
                    this.live = true;
                    this.dispatch('live');
                    break;

                case 'error':
                    // Flash error
                    this.dispatch('error',
new FlashError(msg));
                    break;

                default:
                    // catch-all event, jus
t in case
                    // console.log("webcam
flash_notify: " + type + ": " + msg);
                    break;
            }
        },

        b64ToUint6: function (nChr) {
            // convert base64 encoded chara
cter to 6-bit integer
            // from: https://developer.mozi
lla.org/en-US/docs/Web/JavaScript/Base64_en
coding_and_decoding
            return nChr > 64 && nChr < 91 ?
nChr - 65
                : nChr > 96 && nChr < 123 ?
nChr - 71
                    : nChr > 47 && nChr < 5
8 ? nChr + 4
                        : nChr === 43 ? 62
: nChr === 47 ? 63 : 0;
        },

        base64DecToArr: function (sBase64,
nBlocksSize) {
            // convert base64 encoded strin
g to Uintarray
            // from: https://developer.mozi
lla.org/en-US/docs/Web/JavaScript/Base64_en
coding_and_decoding
            var sB64Enc = sBase64.replace(/
[^A-Za-z0-9\+\/]/g, ""), nInLen = sB64Enc.l
ength,
                nOutLen = nBlocksSize ? Mat
h.ceil((nInLen * 3 + 1 >> 2) / nBlocksSize)
```

```
* nBlocksSize : nInLen * 3 + 1 >> 2,
                taBytes = new Uint8Array(nO
utLen);

            for (var nMod3, nMod4, nUint24
= 0, nOutIdx = 0, nInIdx = 0; nInIdx < nInL
en; nInIdx++) {
                nMod4 = nInIdx & 3;
                nUint24 |= this.b64ToUint6
(sB64Enc.charCodeAt(nInIdx)) << 18 - 6 * nM
od4;
                if (nMod4 === 3 || nInLen -
nInIdx === 1) {
                    for (nMod3 = 0; nMod3 <
3 && nOutIdx < nOutLen; nMod3++, nOutIdx++)
{
                        taBytes[nOutIdx] =
nUint24 >>> (16 >>> nMod3 & 24) & 255;
                    }
                    nUint24 = 0;
                }
            }
            return taBytes;
        },

        upload: function (image_data_uri, t
arget_url, callback) {
            // submit image data to server
using binary AJAX
            var form_elem_name = this.param
s.upload_name || 'webcam';

            // detect image format from wit
hin image_data_uri
            var image_fmt = '';
            if (image_data_uri.match(/^data
\:image\/(\w+)/))
                image_fmt = RegExp.$1;
            else
                throw "Cannot locate image
format in Data URI";

            // extract raw base64 data from
Data URI
            var raw_image_data = image_data
_uri.replace(/^data\:image\/\w+\;base64\,/,
'');

            // contruct use AJAX object
            var http = new XMLHttpRequest
();
            http.open("POST", target_url, t
rue);

            // setup progress events
            if (http.upload && http.upload.
addEventListener) {
                http.upload.addEventListene
r('progress', function (e) {
                    if (e.lengthComputable)
{
                        var progress = e.lo
aded / e.total;
                        Webcam.dispatch('up
loadProgress', progress, e);
                    }
                }, false);
            }

            // completion handler
            var self = this;
            http.onload = function () {
                if (callback) callback.appl
y(self, [http.status, http.responseText, ht
tp.statusText]);
                Webcam.dispatch('uploadComp
lete', http.status, http.responseText, htt
p.statusText);
            };

            // create a blob and decode our
base64 to binary
            var blob = new Blob([this.base6
4DecToArr(raw_image_data)], { type: 'imag
e/' + image_fmt });

            // stuff into a form, so server
s can easily receive it as a standard file
upload
            var form = new FormData();
            form.append(form_elem_name, blo
b, form_elem_name + "." + image_fmt.replace
(/e/, ''));

            // send data to server
            http.send(form);
        }

    };

    Webcam.init();

    if (typeof define === 'function' && def
ine.amd) {
        define(function () { return Webcam;
});
    }
    else if (typeof module === 'object' &&
module.exports) {
        module.exports = Webcam;
    }
    else {
        window.Webcam = Webcam;
    }
```

```
}(window));
```