

Politechnika Krakowska
Kraków 2022

Konwerter danych EXIF

Projekt z przedmiotu
Automaty, Języki i Obliczenia

Aleksandra Rudy

Link do repozytorium: <https://github.com/aleksandrarudy/exifApp>

Spis treści

Spis treści	2
1. Wstęp	3
2. Cel projektu	3
3. Aplikacja	3
3.1 Działanie	3
3.2 Dane EXIF odczytywane przez aplikację	3
3.3 Plik wejściowy	4
3.4 Wygenerowany wyjściowy plik z danymi	4
3.5 Wygląd aplikacji	4
3.6 Zastosowana technologia	5
3.6.1 PHP	5
3.6.2 JavaScript	6
3.6.3 HTML	6
3.6.4 CSS	6
3.7 Implementacja	6
Skrypt preview.js	6
Skrypt download-data.js	7
Skrypt w pliku mainPage.php	8
3.8 Docker	10
4. Rozbudowa projektu	10
5. Bibliografia	11

1. Wstęp

Aplikacja zrealizowana w ramach projektu wydziela dane EXIF z pliku wejściowego jakim jest przesłane przez użytkownika zdjęcie o rozszerzeniu .jpg oraz .jpeg. Dane z niego wydzielone dostępne są do pobrania w formie pliku tekstowego. Aplikacja korzysta z biblioteki jQuery w wersji 3.6.1 oraz rozszerzenia jQuery.exif do tej biblioteki umożliwiającego odczyt danych EXIF ze zdjęć.

2. Cel projektu

Celem aplikacji jest umożliwienie wyodrębniania konkretnych danych EXIF ze zdjęć przesłanych przez użytkownika, automatyczne wprowadzenie ich do konkretnych pól oraz danie możliwości na pobranie pliku tekstowego z tymi danymi.

3. Aplikacja

3.1 Działanie

- Użytkownik naciska przycisk "CHOOSE FILE FROM YOUR COMPUTER".
 - Otwiera się lokalny folder na komputerze użytkownika.
- Użytkownik wybiera plik .jpg lub .jpeg.
 - Zdjęcie załadowuje się w podglądzie na stronie.
 - Wybrane dane EXIF automatycznie uzupełniają pola na stronie.
- Użytkownik naciska przycisk "DOWNLOAD FILE".
 - Plik z danymi exif pobiera się na komputer użytkownika.

3.2 Dane EXIF odczytywane przez aplikację

Dane EXIF odczytywane przez aplikację to:

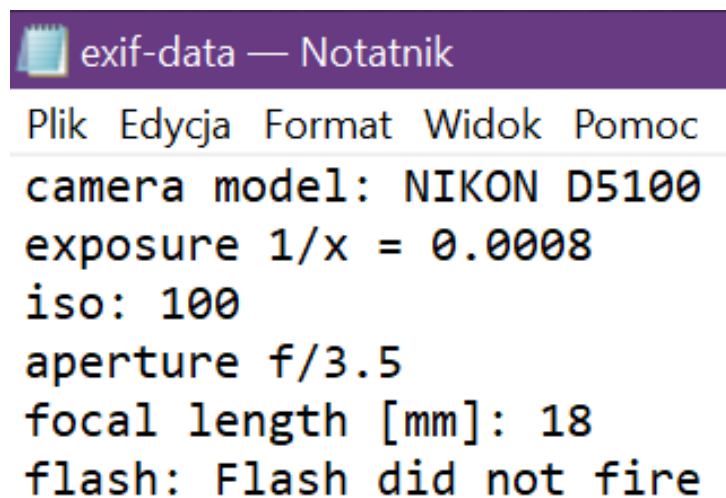
- **Camera model - Model aparatu** np. Nikon D90 itp.
- **Exposure - Czas naświetlania** wyrażany w postaci np. 1/200 s, w zakresie od 1/8000 sekundy do 30 lub nawet 60 sekund.
- **ISO** - wyrażany w postaci np. 100 w zakresie zaczynającym się od 50 w kompaktowych aparatach, od 100 w lustrzankach kolejne wartości to 200, 400, 800 i 1600 a nawet 1640000 w niektórych profesjonalnych aparatach.
- **Aperture - Wartość przysłony** w postaci np. f/1.0, f/1.4, f/2, f/2.8, f/4, f/5.6, f/8, f/11, f/16, f/22, f/32, f/45, f/64, f/90, f/128 itd.
- **Focal length - Ogniskowa** np. 18mm, 35mm, 50mm, 85mm, 135mm, 200mm itd.
- **Flash - Lampa błyskowa** np. 'Flash fired', 'Flash did not fire' itp.

3.3 Plik wejściowy

Plik wejściowy to zdjęcie w formacie JPEG lub JPG, najlepiej posiadający dane EXIF, dla nie posiadających takich danych w każdym polu zostanie wyświetlony tekst "undefined".

3.4 Wygenerowany wyjściowy plik z danymi

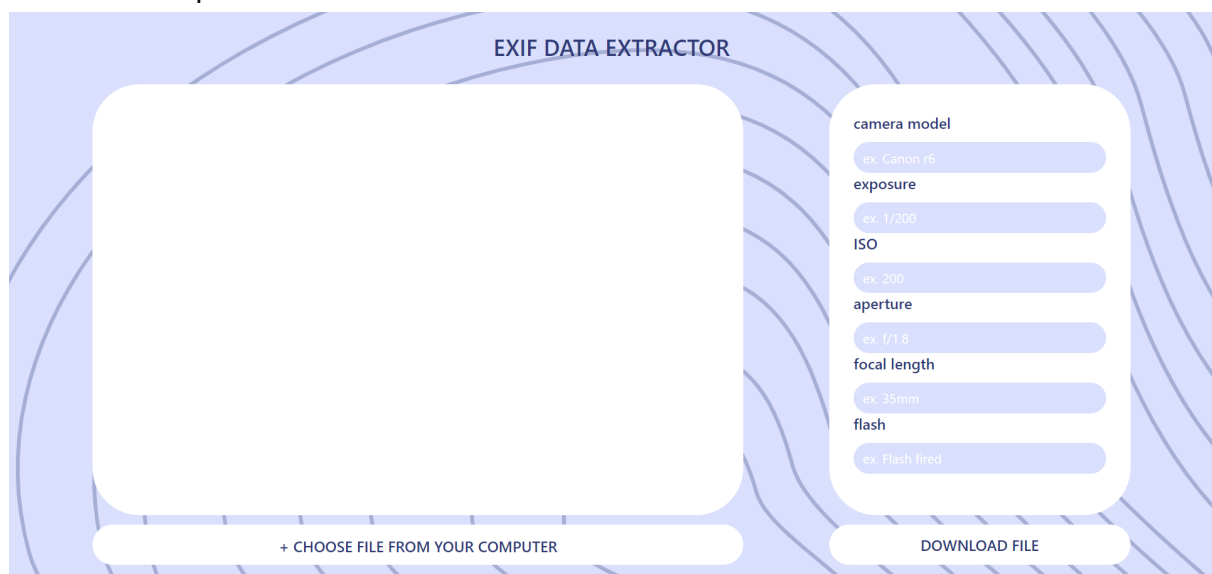
Plik tekstowy o nazwie exif-data.txt w każdej linijce zawiera wartość kolejnych pól ze strony.



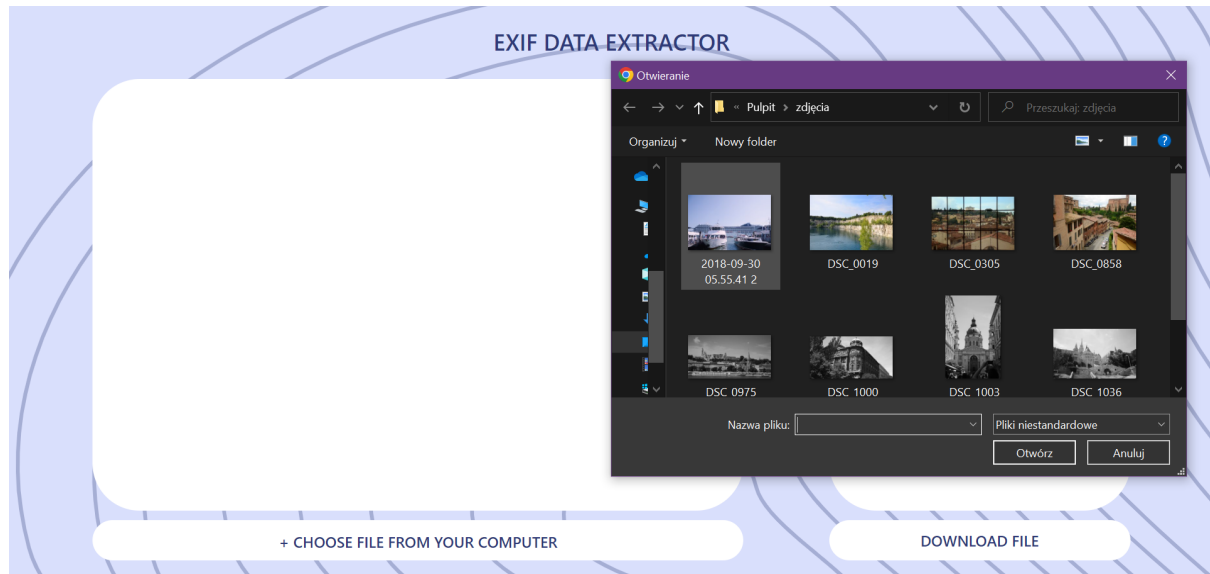
```
camera model: NIKON D5100
exposure 1/x = 0.0008
iso: 100
aperture f/3.5
focal length [mm]: 18
flash: Flash did not fire
```

3.5 Wygląd aplikacji

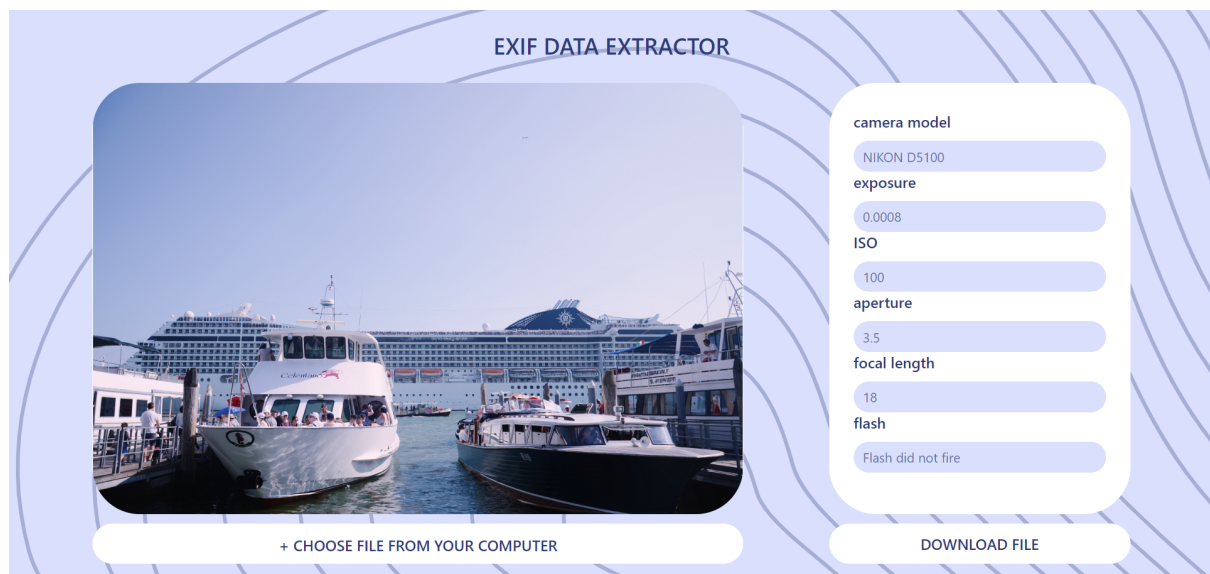
- Strona po uruchomieniu



- Strona po naciśnięciu “CHOOSE FILE FROM YOUR COMPUTER”.



- Strona po załadowaniu zdjęcia oraz wydzieleniu danych EXIF.



3.6 Zastosowana technologia

3.6.1 PHP

W języku PHP został napisany routing aplikacji czyli mechanizm obsługujący żądania HTTP, innymi słowy mechanizm który decyduje co ma się wydarzyć po odwiedzeniu strony w tym wypadku <http://localhost:8080/mainPage>.

3.6.2 JavaScript

W JS zostały zaimplementowane skrypty między innymi skrypt, który umożliwia wyświetlenie podglądu zdjęcia przesłanego za pomocą tagu HTML `<input type="file">`. Skrypt odpowiadający za pobieranie pliku na podstawie danych umieszczonych w wielu tagach HTML `<input type="text">`. W JavaScript została także dołączona biblioteka jQuery oraz rozszerzenie tej biblioteki o dodatek pozwalający na odczyt danych EXIF z plików JPEG i JPG.

3.6.3 HTML

W tym języku został zaimplementowany widok strony w formie surowej warstwy graficznej oraz wszystkie inputy dzięki którym mamy możliwość umieszczania zdjęcia oraz miejsce na wyświetlenie danych EXIF odczytanych z tego zdjęcia.

3.6.4 CSS

Ten język posłużył mi do wystylizowania tagów HTML które tworzą układ strony.

3.7 Implementacja

Skrypt preview.js

```
1      const chooseFile = document.getElementById( elementId: "file");
2      const imgPreview = document.getElementById( elementId: "image-placeholder");
3
4      chooseFile.addEventListener( type: "change", listener: function () {
5          getImgData();
6      });
7
8      function getImgData() {
9          const files = chooseFile.files[0];
10         if (files) {
11             const fileReader = new FileReader();
12             fileReader.readAsDataURL(files);
13             fileReader.addEventListener( type: "load", listener: function () {
14                 imgPreview.style.display = "flex";
15                 imgPreview.innerHTML = '';
16             });
17         }
18     }
```

Zmienna `chooseFile` odnosi się do inputa którego naciśnięcie powoduje wykonanie funkcji `getImgData()` co zaimplementowane jest w liniach kodu 4,5,6. Zmienna `imgPreview` odnosi się do kontenera w którym umieszczony zostanie podgląd zdjęcia

Funkcja `getImgData()` za pomocą API `FileReader` a konkretnie z metody `readAsDataURL`, która odczytuje dane zakodowane jako ciąg znaków wybranego zdjęcia oraz interpretuje je. Wyświetla podgląd zdjęcia i umieszcza go w konkretnym tagu w kontenerze do którego odnosi się zmienna `imgPreview`

API `FileReader` jest wykorzystywane również w dodatku do biblioteki `jQuery` który znajduje się w pliku `jquery.exif.js`. Służy ona właśnie do odczytu danych z plików formatu JPEG i JPG.

Skrypt `download-data.js`

```
1 let separator = "\n";
2 let names = ["camera model: ", "exposure 1/x = ", "iso: ", "aperture f/", "focus length [mm]: ", "flash: "];
3 let textInputs = ["cameraModel", "exposure", "iso", "aperture", "focus", "flash"];
4
5 function saveTextAsFile() {
6     let textToSave = "";
7     for (let a = 0; a < textInputs.length; a++) {
8         textToSave += a < textInputs.length - 1 ?
9             names[a] + document.getElementById(textInputs[a]).value + separator :
10            names[a] + document.getElementById(textInputs[a]).value;
11     }
12     let textToSaveAsBlob = new Blob( blobParts: [textToSave], options: {
13         type: "text/plain"
14     });
15     let textToSaveAsURL = window.URL.createObjectURL(textToSaveAsBlob);
16     let fileNameToSaveAs = "exif-data";
17
18     let downloadLink = document.createElement( tagName: "a");
19     downloadLink.download = fileNameToSaveAs;
20     downloadLink.href = textToSaveAsURL;
21     downloadLink.style.display = "none";
22     document.body.appendChild(downloadLink);
23     downloadLink.click();
24
25     downloadLink.onclick = destroyClickedElement;
26 }
27
28 function destroyClickedElement(event) {
29     document.body.removeChild(event.target);
30 }
```

Na początku określona jest zmienna `separator`, która jak sama nazwa wskazuje oddziela nam informacje, pobrane z inputów przenosząc je do nowej linii, ponieważ pożądanym wyglądem pliku powinien być zgodny z tym co widzimy na stronie. Zmienna `names` jest tablicą w której przechowywane są nazwy interesujących nas danych. Natomiast zmienna `texInputs` odnosi się do ID inputów w których znajdują się informacje.

Funkcja `saveTextAsFile()` pozwala na utworzenie adresu URL którego naciśnięcie spowoduje pobranie pliku.

Do zmiennej `textToSave` przypisujemy nazwy z tabeli `names` zawartości inputów do których odnosimy się poprzez ID oraz separator czyli znak nowej linii.

Następnie do zmiennej `textToSaveAsBlob` przypisujemy konstruktor `Blob` który tworzy bloba ze zmiennej `textToSave`, `Blob` jest plikopodobnym obiektem, a zapisane do niego dane mogą być odczytywane między innymi jako tekst.

Kolejno do zmiennej `textToSaveAsURL` zapisujemy obiekt utworzony za pomocą metody `createObjectURL` która tworzy z podanego argumentu `textToSaveAsBlob` adres URL.

Przy pomocy zmiennej `fileNameToSaveAs` nadajemy nazwę pliku który zostanie utworzony.

Do zmiennej `downloadLink` przypisujemy utworzony obiekt na którym wykonujemy czynności przypisane do zmiennych `fileNameToSaveAs` i `textToSaveAsURL`. Metoda `click()` wywołuje zdarzenie kliknięcia elementu w tym wypadku przycisku w którym wywołujemy funkcję `saveTextAsFile()` dzięki temu przycisk wie jak ma się zachować.

```
<div class="exifdata-download-button">  
  <button class="download-btn" onclick="saveTextAsFile()">DOWNLOAD FILE</button>  
</div>
```

(Przycisk który uruchamia funkcję `saveTextAsFile()`)

Wywołanie funkcji `destroyClickedElement`, która usuwa utworzony obiekt po naciśnięciu w adres URL.

Skrypt w pliku `mainPage.php`

To skrypt umieszczający konkretne dane EXIF w odpowiednich inputach.


```

64 <script>
65     let someCallback = function(exifObject) {
66         document.getElementById('cameraModel').value = exifObject.Model
67         document.getElementById('aperture').value = exifObject.FNumber
68         document.getElementById('iso').value = exifObject.ISOSpeedRatings
69         document.getElementById('exposure').value = exifObject.ExposureTime
70         document.getElementById('focus').value = exifObject.FocalLength
71         document.getElementById('flash').value = exifObject.Flash
72     }
73
74     try {
75         document.getElementById('file').onchange = function() {
76             $(this).fileExif(someCallback);
77         };
78     }
79     catch (e) {
80         alert(e);
81     }
82 </script>

```

Zmienna someCallback przypisana jest do funkcji której argumentem jest obiekt exifObject.

W funkcji tej do każdego elementu(inputa) o konkretnym ID przypisana jest wartość odczytana przy pomocy obiektów funkcji znajdującej się dodatku do biblioteki jQuery, który znajduje się w pliku jquery.exif.js. wartości te są odczytywane z pliku JPEG lub JPG, który zaciągnięty jest z komputera użytkownika przy pomocy inputa typu file o ID = file.

```

<label for="cameraModel">camera model</label><br>
<input type="text" name="cameraModel" id="cameraModel" placeholder="ex. Canon ró"/> </br>

<label for="exposure">exposure</label><br>
<input type="text" name="exposure" id="exposure" placeholder="ex. 1/200"/></br>

<label for="iso">ISO</label><br>
<input type="text" name="iso" id="iso" placeholder="ex. 200" /> </br>

<label for="aperture">aperture</label><br>
<input type="text" name="aperture" id="aperture" placeholder="ex. f/1.8" /> </br>

<label for="focus">focus length</label><br>
<input type="text" name="focus" id="focus" placeholder="ex. 35mm"/> </br>

<label for="flash">flash</label><br>
<input type="text" name="flash" id="flash" placeholder="ex. Flash fired"/> </br>

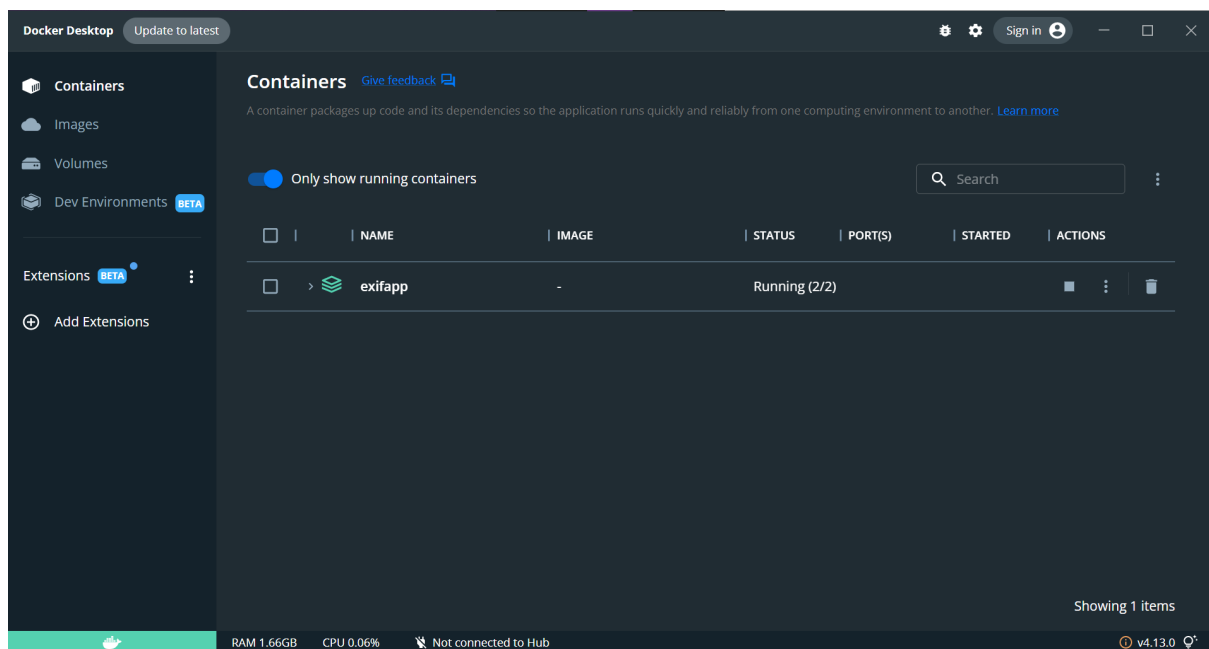
```

(Inputy w których umieszczone zostaną dane EXIF)

```
<div class="image-upload-button">
  <input type="file" name="file" id="file" class="image-upload-input" accept="image/jpg, image/jpeg">
  <label for="file">+ CHOOSE FILE FROM YOUR COMPUTER</label>
</div>
```

(Input który służy do zaciągania pliku JPEG lub JPG z komputera użytkownika)

3.8 Docker



Serwer aplikacji znajduje się na kontenerze Dockera w celu przyspieszenia procesu wytwarzania oprogramowania. W celu uruchomienia serwera oraz otwarcia aplikacji w terminalu znajdując się w folderze projektu należy wpisać komendę **docker-compose up**.

```
PS C:\Users\AleksandraRudy\PhpstormProjects\exifApp> docker-compose up
[+] Running 2/0
 - Container exifapp-php-1 Created                                0.0s
 - Container exifapp-web-1 Created                                0.0s
Attaching to exifapp-php-1, exifapp-web-1
exifapp-php-1 | [14-Dec-2022 10:50:27] NOTICE: fpm is running, pid 1
exifapp-php-1 | [14-Dec-2022 10:50:27] NOTICE: ready to handle connections
exifapp-php-1 | 172.19.0.3 - 14/Dec/2022:10:50:36 +0000 "GET /index.php" 200
exifapp-web-1 | 172.19.0.1 - - [14/Dec/2022:10:50:37 +0000] "GET /mainPage HTTP/1.1" 200 3909 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64
) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/108.0.0.0 Safari/537.36" "-"
exifapp-php-1 | 172.19.0.3 - 14/Dec/2022:10:50:37 +0000 "GET /index.php" 200
exifapp-web-1 | 172.19.0.1 - - [14/Dec/2022:10:50:37 +0000] "GET /favicon.ico HTTP/1.1" 200 20 "http://localhost:8080/mainPage" "Mozilla/5.
0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/108.0.0.0 Safari/537.36" "-"
[]
```

4. Rozbudowa projektu

Aplikację tę można rozbudować o możliwość wyboru przez użytkownika jakie dane EXIF mają być odczytane z pliku, poprzez np. wybranie ich z listy. Formatowanie danych tak aby wyświetlały się one w bardziej przystępnej formie np. czas naświetlania w formie ułamka, lub automatyczne dodanie jednostek przy wartościach.

5. Bibliografia

<https://developer.mozilla.org/en-US/docs/Web/JavaScript>
<https://www.youtube.com/watch?v=3uxilPhD0HU&t=58s>
<https://developer.mozilla.org/en-US/docs/Web/API/FileReader>
<https://developer.mozilla.org/en-US/docs/Web/API/Blob>
<https://developer.mozilla.org/en-US/docs/Web/API/URL/URL>