画像加工アプリ

2024 年 2 月 5 日

K23024　小笠原悠太

1 目的

このアプリケーションは、画像を加工する Web アプリケーションである。ユーザーが保持している画像をエフェクトや変形、色相変更などを用いて加工することを目的とする。

2 機能

このアプリケーションには、加工した画像の保存機能をはじめとして、エフェクト・変形・彩度変更・画質変更機能がある。また、以下で紹介する機能は重ねがけすることができる。対応している画像形式は bmp, gif, jpeg, png, tiff である。各機能の画像は見やすさのためにアプリケーション画面ではなく、加工後保存した画像を表示する。

2.1 操作画面

操作方法は、まず左上のファイルを選択ボタンを押して、ローカルファイルから加工する画像を選択する。次に、その下にあるボタンやバーをいじることで画像の加工ができる。最後に、自分の好きなように加工ができたら右上の画像を保存するボタンを押すことで加工後の画像を保存できる。また、加工中に思うようにならず、画像を元に戻したいときは上の元の画像に戻すボタンを押すことで画像を元の状態に戻せる。画面の様子は図1に示す。

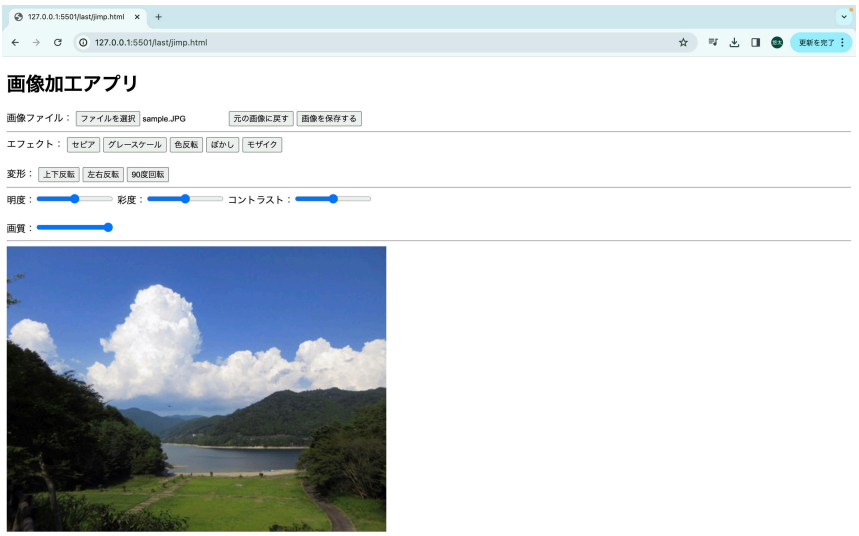

図 1: 操作画面

2.2 エフェクト機能

エフェクトには,セピア・グレースケール・色反転・ぼかし・モザイクがある。それぞれボタンを押すことで変化する。変化の様子はそれぞれ図 3 ~ 7 に示す。

図 2: 元の画像

図 3: セピア

図 4: グレースケール

図 5: 色反転

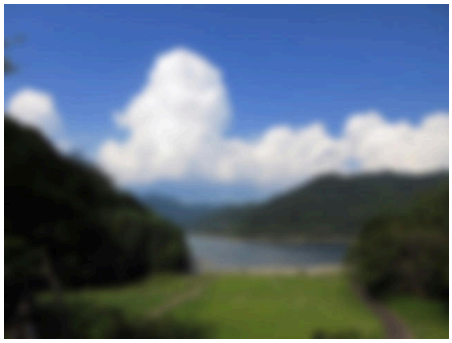

図 6: ぼかし

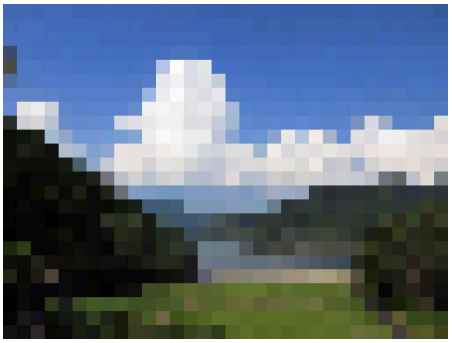

図 7: モザイク

2.3 変形機能

変形機能には,上下反転・左右反転・90 度回転がある。それぞれボタンを押すことで変化する。変化の様子はそれぞれ図 9 ~ 11 に示す。

図 8: 元の画像

図 9: 上下反転

図 10: 左右反転

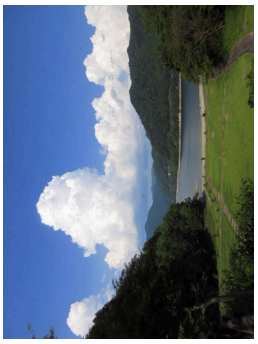

図 11: 90 度回転

2.4 色相変更機能

色相変更機能では,明度変更 · 彩度変更 · コントラスト変更ができる。これらの機能はスライドバーを使って細かな調整ができる。スライドバーのはじめ真ん中の位置にある。明度は、右に行くほど明るくなり、左に行くほど暗くなる。彩度は、右に行くほど彩度が上がり、左に行くほど彩度が下がる。コントラストは、右に行くほどコントラストが強くなり、左に行くほどコントラストが弱くなる。変化の様子を図 12 ~ 17 に示す。

図 12: 明度:　明るめ

図 13: 明度:　暗め

図 14: 彩度:　強め

図 15: 彩度:　弱め

図 16: コントラスト強め

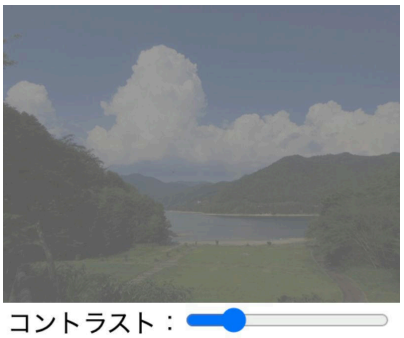

図 17: コントラスト弱め

2.5 画質調整機能

画質調整機能では、画質を元の状態よりも粗くすることができる。この機能はスライドバーを使って細かな調整ができる。スライドバーは初め右端にある。変化の様子を図 19 に示す。

図 18: 元の画像

図 19: 画質:　粗め

3 まとめ

このアプリは画像を加工することを目的としている。 その目的を達成するために変化のわかりやすい機能は実装できた。また、機能の種類別に列を変えて操作もしやすくなっている。

一方で、色相変更のスライドバーをずらす度に、変更後の状態が上書きではなく、追加される形になっているため毎回元の画像に戻して調整しなければいけなくなっている点が思ったようにいかなかった部分だ。

3.1 今後、改善・改良したい点

色相変更のスライドバー変更時に状態を上書きするように改善し、現在の change イベントから input イベントに変えて色相の変化の様子をわかりやすくしたい。

その他には、画像を切り抜く機能を今回実装できていないので、それを実装したい。また、切り抜く機能も実装すると画面が見づらくなってくると思うため、今回用いていない、CSS を用いて画面を見やすくしたい。

4 付録

この章では、本アプリケーションの開発における特徴を述べる。

4.1 ライブラリ

本アプリケーションでは、画像加工を行う上で画像処理ライブラリの JIMP を使用することで比較的短いコードでアプリケーションを実装した。JIMP は NPM インストールをして使用することが多いようだが、今回は CDN を利用することで利用者がわざわざインストールをしなくて済むようにした。

4.2 画像描画部分

このアプリでは加工した様子がすぐにその画面で見られるようにしている。 そのため、すべての機能の関数の中に画像を描画するプログラムを書く必要があり、関数化することでコードの短縮化をした。 画像描画をする関数を図 20 に示す。

```
44  // canvasに画像を描画する関数
45  function drawPicture() {
46    image.getBase64(Jimp.AUTO, function (err, data) {
47      if (err) throw err;
48      const imageElement = new Image();
49      imageElement.src = data;
50      imageElement.onload = function () {
51        ctx.drawImage(imageElement, 0, 0, imageWidth, imageHeight);
52      };
53    });
54  }
```

図 20: 画像描画関数