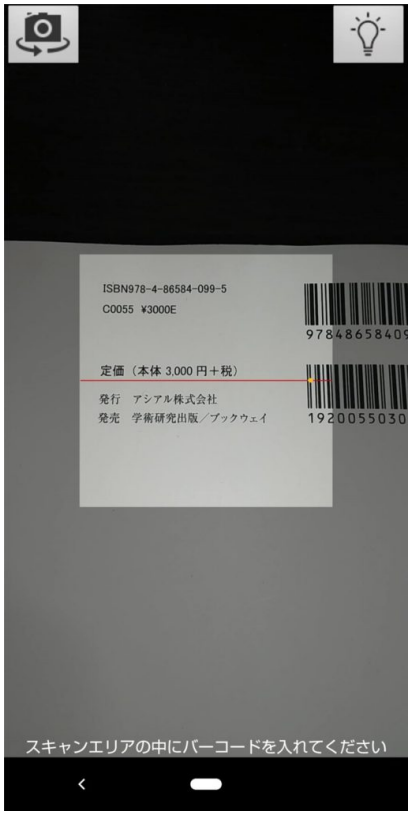
Monaca

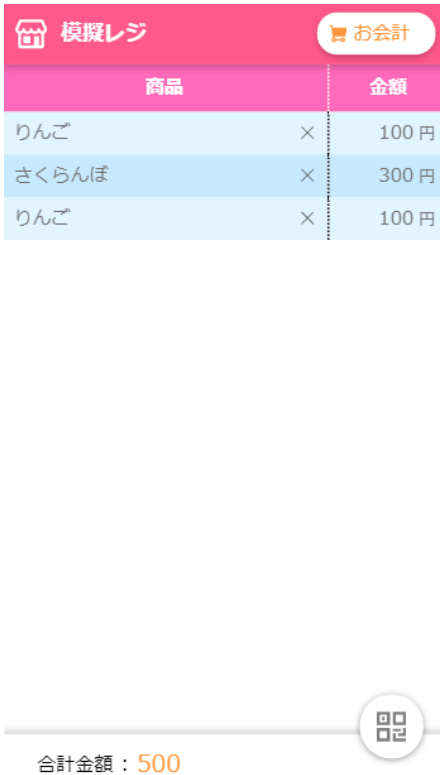
模擬レジアプリ編

アシアル株式会社
アシアル情報教育研究所
岡本 雄樹

■ 模擬レジアプリの紹介①

- 購入商品の合計金額を計算するアプリです
- バーコードを読み取ります
- 商品名と金額が表示されます





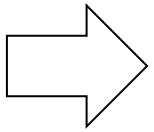
■ 模擬レジアプリの紹介②

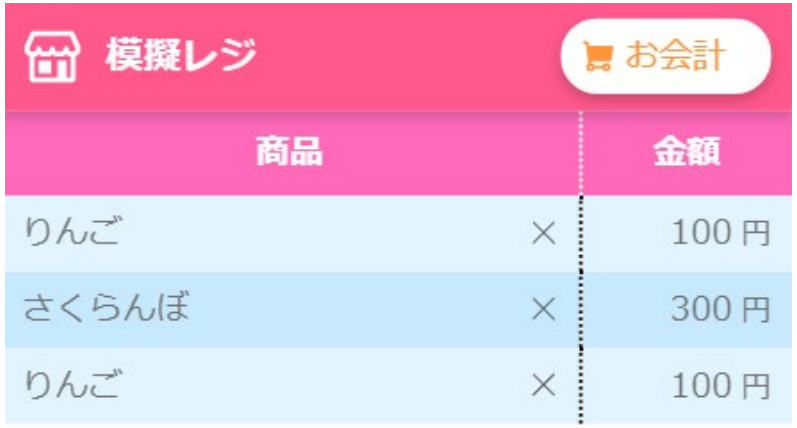
- 右下の読み取りボタンでバーコードを読み取ります
- 読み取るとcart変数(配列)に商品情報が『追記』されます
- 右上の『お会計ボタン』が押されるとお会計して終了です

■ 模擬レジアプリの紹介③

- 配列には商品のコードが格納されます
- 配列の値を元に画面を生成しています

キー	バリュー
0	1001
1	1002
2	1001





■ 模擬レジアプリの紹介④

- コードと商品の対応表はconfig.js内に記載
- バーコードのサンプルはWebサイト上に掲載
- 詳細はあんこエデュケーションの記事を参照
  - https://anko.education/apps/register

```
let itemMaster = {
    1001:{name:"りんご",price:100},
    1002:{name:"さくらんぼ",price:300},
    1003:{name:"梨",price:150},
    1004:{name:"イチゴ",price:550},
    1005:{name:"ブル一ベリ一",price:350},
    1006:{name:"ぶどう",price:600},
    1007:{name:"もも",price:200},
    1008:{name:"みかん",price:300},
    1009:{name:"すいか",price:1500},
    1010:{name:"マンゴ一",price:800},
    1011:{name:"メロン",price:3000},
}
```

1001:りんごのバーコード

1002:さくらんぼのバーコード

ソースコード解説

ソースコード解説

■ HTML側

```
<body>
    <header>
        <h1 onclick="reset()">模擬レジ</h1>
        <button type="button" onclick="pay()">お会計</button>
    </header>
    <table id="itemList">
        <thead>
            <tr>
                <th>商品</th>
                <th>金額</th>
            </tr>
        </thead>
        <tbody id="itemListBody"></tbody>
    </table>
    <div id="barcode" onclick="scan()">
        <image src="img/iconCode.svg" alt="コード読み取りボタン">
    </div>
    <footer>
        <div id="title">合計金額</div>
        <div id="totalPrice">0</div>
    </footer>
</body>
```

クリックされたら『リセット』

クリックされたら『お会計』

table要素で商品一覧を表示

クリックされたら『スキャン』

フッターには合計金額を表示

■ アプリ内の関数一覧

関数	処理
estimate()	商品一覧や合計金額を表示
cartDelete()	指定されたキーの商品をカートから取り除く
pay()	支払金額を受け取り、お釣りを表示
reset()	変数cartを空にする
onSuccess()	スキャン成功時の処理
onError()	スキャン失敗時の処理
scan()	スキャンを実行

■ config.js

```
/**
 * 商品の名前と価格はここで定義
 */
let itemMaster = {
    1001:{name:"りんご",price:100},
    1002:{name:"さくらんぼ",price:300},
    1003:{name:"梨",price:150},
    1004:{name:"イチゴ",price:550},
    1005:{name:"ブル□ベリ□",price:350},
    1006:{name:"ぶどう",price:600},
    1007:{name:"もも",price:200},
    1008:{name:"みかん",price:300},
    1009:{name:"すいか",price:1500},
    1010:{name:"マンゴ□",price:800},
    1011:{name:"メロン",price:3000},
}
/**
 * バ□コ□ドを読み取るための関数のオプション設定
 */
let options = {
    preferFrontCamera : false,
    showFlipCameraButton : true,
    showTorchButton : true,
    torchOn: true,
    saveHistory: false,
    prompt : "スキャンエリアの中にバ□コ□ドを入れてください",
    resultDisplayDuration: 500,
    formats : "QR_CODE,EAN_13,CODE_39",
    orientation : "unset",
    disableAnimations : true,
    disableSuccessBeep: false
}
```

連想配列itemMasterに商品をコードで管理

optionsにバーコードを読み取る際の設定を記述

■ main.js

```
/**
 * スクリプト全体で共有する変数を定義
 */
let cart = [];
let totalPrice = 0
```

アプリ全体で使う変数を定義
変数cartには配列形式で商品のコードを格納する
totalPriceは合計金額を格納する

■ estimate()

```
function estimate() {
    let totalPrice = 0;
    let table = document.getElementById("itemListBody");
    table.innerHTML = "";
    for (let i = 0; i < cart.length; i++) {
        // 商品マスタ一の情報を元に合計金額を計算
        let item = itemMaster[cart[i]];
        totalPrice += item.price;
        // 商品情報を表に反映
        let tr = document.createElement("tr");
        tr.innerHTML = "<td>"
                     + item.name
                     + '<span class="delete"  onclick="cartDelete('
                     +  i + ')"></span>'
                     + "</td>"
                     + "<td>" + item.price + "</td>";
        table.appendChild(tr);
    }
    document.getElementById("totalPrice").innerText = totalPrice.toLocaleString();
    console.log(cart);
}
```

変数cartを元にループ処理して商品を画面表示

■ cartDelete()

```
function cartDelete(key) {
    cart.splice(key,1);
    estimate();
}
```

指定されたキーに基づき、商品をcartから削除

■ cartReset ()

```
function reset() {
    if(confirm("リセットしますか？")){
        cart = [];
        totalPrice = 0;
        estimate();
    }
}
```

変数cartをリセット

■ pay ()

```
function pay() {
    payment = promptInt("お支払い金額を入力してください");
    let change = payment - totalPrice;

    if (change == 0) {
        alert("お釣りはありません");
    } else if (change > 0) {
        alert("お釣りは" + change + "円です");
    } else {
        alert("お預かり金額が" + Math.abs(change) + "円不足です");
    }
}
```

支払金額を受け取り、お釣りを計算

■ scan ()

```
function scan() {
    if (typeof cordova === "undefined") {
        let int = promptInt("数値を入力してください。");
        cart.push(int);
        estimate();
        return false;
    }
    cordova.plugins.barcodeScanner.scan(
        onSuccess,
        onError,
        options
    );
}
```

スキャンを実行する。

cordovaの使えない『ブラウザ』の場合には、プロンプトで商品番号を入力させる。

cordovaが使える『アプリ』の場合はcordovaのスキャン機能を呼び出す。

■ onSuccess()とonError()

```
function onSuccess (result) {
    let str = result.text;
    let int = parseInt(str);
    cart.push(int);
    estimate();
}

function onError (error) {
    alert("スキャン失敗: " + error);
}
```

スキャンが成功したら、商品番号を変数cartに追記する。

スキャンが失敗したら、『スキャン失敗』とアラート表示する

カスタマイズ

オリジナルの商品を追加しよう

■ オリジナルの商品を追加しよう

- config.jsのitemMaster変数に商品を追加すれば可能
- バーコードも作成して読み取ってみよう

■ 商品の追加

- config.jsのitemMaster変数に商品を追加
- 商品番号は連番でなくてもOK
- 文字列の場合はシングルクォートかダブルクォートで囲む

```
let itemMaster = {
    1001:{name:"りんご",price:100},
    1002:{name:"さくらんぼ",price:300},
    1003:{name:"梨",price:150},
    1004:{name:"イチゴ",price:550},
    1005:{name:"ブルーベリー",price:350},
    1006:{name:"ぶどう",price:600},
    1007:{name:"もも",price:200},
    1008:{name:"みかん",price:300},
    1009:{name:"すいか",price:1500},
    1010:{name:"マンゴー",price:800},
    1011:{name:"メロン",price:3000},
    2001:{name:"メダル",price:5000},
}
```

■ バーコードの作成

└ バーコード生成サイト＆アプリもご用意しておきました

└ https://anko.education/tool/qr_generator