アプリ開発PBL 第３回

TA：石関，高橋，林，中井

昨年度の成果物　2年編

RPG班が作ってくれたものしか
残ってないですが……(以下反省点らしい)

・bgmがループしない
・マップの切り替えが存在するが,
マップが未完成
・敵の強さやレベル上げの条件などの
ゲームバランスが調整されていない

昨年度の成果物　1年編

- 発表の時間は10分
  (準備1分，発表7分，質疑応答2分)

- 去年と今とで違うことやってる班も
  いるかも

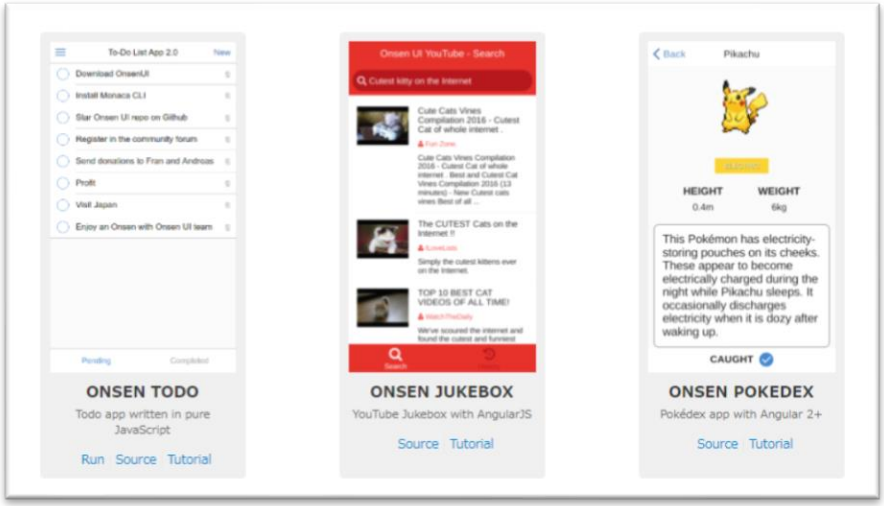
次回は2年次のアプリ計画です

- 今回はonsenのサンプルを見てみましょう
  - https://ja.onsen.io/samples/
  - Source -> Tutorial でサンプルアプリの
    ソースコードの解説を見ることができる

- 時間が余ったら最終発表用のアプリ計画の
  時間にします

あるある

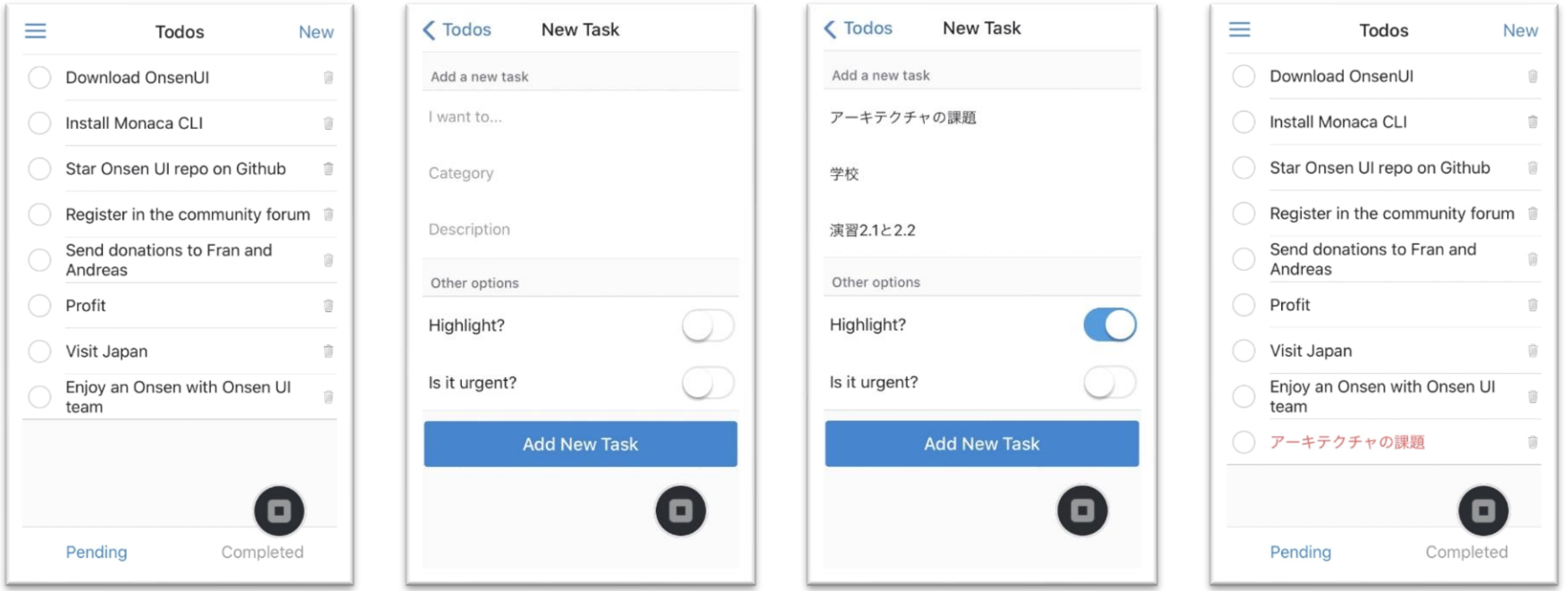
- プログラムを作るとき，他のプログラムを参考にすることは多い
- 解説が理解できないと移植することも難しい
- https://medium.com/the-web-tub/style-your-app-automatically-with-onsen-ui-79fc403a378e



TODOアプリを作ろう①

- 解説ページは全編英語
  - 世の中のまともなドキュメントは大体英語
- 前回「解説不足」というコメントが多数だったので，解説メインです
- プロジェクトのzipファイルをインポートしよう

TODOアプリを作ろう②



- シンプルながらも洗練されたデザイン
  - ぜひとも参考にしたい！

TODOアプリ解説序文

- このTODOアプリは基本的な要素で実装されており，アプリ開発の参考になる
- ToDoリスト作成の方法はたくさんあり，これはそのうちの1つでしかない
- 自動スタイリング，Navigator + Tabbar + Splitterの組み合わせなど多くのonsenUIの機能を使う

プロジェクトの構成

```
www/
  ├ css/
  │  └ style.css
  ├ js/
  │  ├ app.js
  │  ├ controllers.js
  │  └ services.js
  ├ html/
  │  ├ completed_tasks.html
  │  ├ details_task.html
  │  ├ menu.html
  │  ├ new_task.html
  │  └ pending_tasks.html
  ├ lib/
  │  └ onsen/
  └ index.html
```

プロジェクトの構成

```
www/
  ├ css/
  │  └ style.css
  ├ js/
  │  ├ app.js
  │  ├ controllers.js
  │  └ services.js
  ├ html/
  │  ├ completed_tasks.html
  │  ├ details_task.html
  │  ├ menu.html
  │  ├ new_task.html
  │  └ pending_tasks.html
  ├ lib/
  │  └ onsen/
  └ index.html
```

App structure:HTML①

- index.html内のすべてのナビゲーションコンポーネントの要素はons-templateで定義されている
  - ons-navigator
  - ons-splitter
  - ons-tabbar

App structure:HTML②

- TODOアプリに必要なもの
  - タブバーで仕切られたpending tasks と completed tasksのリスト
  - タスクのフィルタリングに使用するカテゴリのリスト(メニューに配置)
  - 新しいタスクを追加できるページ
  - 既存のタスクを変更し詳細を確認できるページ

Tabbar

- 最下層のナビゲーションコンポーネント
- 保留リストと完了リストを分離

```
<ons-template id="tabbar.html">
  <ons-page id="tabbarPage">
    ...

    <ons-tabbar id="myTabbar" position="auto">
      <ons-tab page="html/pending_tasks.html" label="Pending" active>
      </ons-tab>
      <ons-tab page="html/completed_tasks.html" label="Completed">
      </ons-tab>
    </ons-tabbar>
  </ons-page>
</ons-template>
```

Splitter (menu)

- 上記2リストの共通メニュー
- ons-splitterはons-tabbarの親
- 2つのSplitterを作成

```
<ons-template id="splitter.html">
  <ons-splitter id="mySplitter">
    <ons-splitter-side page="html/menu.html" swipeable
width="250px" collapse swipe-target-width="60px">
    </ons-splitter-side>
    <ons-splitter-content page="tabbar.html">
    </ons-splitter-content>
  </ons-splitter>
</ons-template>
```

Navigator

- 「新しいタスク」または「タスクの変更」のページに変更
- ons-navigatorは2つの親コンポーネント
- タスクを作成/変更するときにメニューとタブバーを非表示にする

```
<ons-navigator id="myNavigator" page="splitter.html">
</ons-navigator>
```

プロジェクトの構成

```
www/
  ├ css/
  │  └ style.css
  ├ js/
  │  ├ app.js
  │  ├ controllers.js
  │  └ services.js
  ├ html/
  │  ├ completed_tasks.html
  │  ├ details_task.html
  │  ├ menu.html
  │  ├ new_task.html
  │  └ pending_tasks.html
  ├ lib/
  │  └ onsen/
  └ index.html
```

App structure:JS

- ページごとに初期化関数(controller)を定義し，そこから関数を設定している

- 他は内容的に今の知識では難しい

app.js

- 基本的なアプリのセットアップ
- ページの準備ができたら，対応する初期化コントローラーを呼び出し，タスクリストに初期データを入力

```
window.myApp = {};

document.addEventListener('init', function(event) {
  var page = event.target;

  if (myApp.controllers.hasOwnProperty(page.id)) {
    myApp.controllers[page.id](page);
  }
```

controllers.js

- すべての初期化関数を格納
- ひとつひとつを確認

services.js①

- 一番大変な部分
- myApp.services.tasks.create(...)を呼び出すと……

```
myApp.services = {
  tasks: {
    create: function() {},
    update: function() {},
    ...
  },
  categories: {
    create: function() {},
    ...
  },
  animators: {
```

services.js②

- Dataオブジェクトで定義された新しいons-list-itemを生成する
  - document.createElement('ons-list-item’)
- 空のdiv要素を作成し，innerHTMLプロパティを使用してカスタムコンポーネントを内部に挿入
- 各タスクのデータをons-list-item内部配列ではなく，それ自体の中に格納している(taskItem.data = data;)

services.js③

- タスクを完了または元に戻すための機能
- チェックボックスのchangeイベントのためにリスナーを追加
- completion機能は保留を完了に追加

```
// Add 'completion' functionality when the checkbox changes.
  taskItem.data.onCheckboxChange = function(event) {
    myApp.services.animators.swipe(taskItem, function() {
      var listId = (taskItem.parentElement.id === 'pending-list'
&& event.target.checked) ? '#completed-list' : '#pending-list';
      document.querySelector(listId).appendChild(taskItem);
    });
  };
```

services.js④

- キーフレームやタイムアウトを持つカスタムcssクラスを追加

- 特定パラメータときアニメーションが終了するコールバックを実行

services.js⑤

- アイテムを保留中のタスクリストに追加
- insertBeforeがnullの場合項目はリストの最後に追加される

```
var pendingList = document.querySelector('#pending-list');
pendingList.insertBefore(taskItem, taskItem.data.urgent ?
pendingList.firstChild : null);
```

Automatic Styling

- iOSとandroidで自動的にスタイルを変更できる

- 様々な方法がある

プロジェクトの構成

```
www/
  ├ css/
  │  └ style.css
  ├ js/
  │  ├ app.js
  │  ├ controllers.js
  │  └ services.js
  ├ html/
  │  ├ completed_tasks.html
  │  ├ details_task.html
  │  ├ menu.html
  │  ├ new_task.html
  │  └ pending_tasks.html
  ├ lib/
  │  └ onsen/
  └ index.html
```

Tabbar position

- positionの指定にはtopやbottomといったものがあり，autoにすると端末に合うものになる

```
<ons-tabbar id="myTabbar" position="auto">
    ...
</ons-tabbar>
```

```
<ons-tabbar id="myTabbar" position=”top">
    ...
</ons-tabbar>
```

ons-if

```
<ons-toolbar>
  ...
  <div class="right">
    <!-- Toolbar-button  only visible for iOS/other. -->
    <ons-if platform="ios other">
      <ons-toolbar-button component="button/new-task">New</ons-
toolbar-button>
    </ons-if>
  </div>
</ons-toolbar>

<!-- Floating Action Button only visible for Android. -->
<ons-if platform="android">
  <ons-fab position="right bottom" component="button/new-task">
    <ons-icon icon="md-edit"></ons-icon>
  </ons-fab>
</ons-if>

...
```

プロジェクトの構成

```
www/
  ├ css/
  │  └ style.css
  ├ js/
  │  ├ app.js
  │  ├ controllers.js
  │  └ services.js
  ├ html/
  │  ├ completed_tasks.html
  │  ├ details_task.html
  │  ├ menu.html
  │  ├ new_task.html
  │  └ pending_tasks.html
  ├ lib/
  │  └ onsen/
  └ index.html
```

ons.platform

- 現在のプラットフォームをチェックするための一連のメソッドが含まれている

- そのうちの2つはisAndroid()とisIOS()

```
... + '<ons-icon ... icon="' + (ons.platform.isAndroid() ? 'md-delete' : 'ion-ios-trash-outline') + '"></ons-icon>' + ...
```

Animations

- 現在のons-navigatorのデフォルトアニメーションはプラットフォームで異なる

- Androidでのみタブバーアニメーションを「スライド」に変更

```
document.querySelector('#myTabbar').setAttribute('animation',
ons.platform.isAndroid() ? 'slide' : 'none');
```

Promises for async methods

- すべての非同期メソッドはメソッドがプッシュ，ポップなどしている要素に解決されるPromiseを返すようになった

- このアプリでは，ダイアログを簡単に作成し，新しいPromiseでユーザー入力を処理するためにons.notificationを使用します

```
ons.notification.confirm(
  {
    title: 'Save changes?',
    message: 'Previous data will be overwritten.',
    buttonLabels: ['Discard', 'Save']
  }
).then(function(buttonIndex) {

  if (buttonIndex === 1) {
    // If 'Save' button was pressed, overwrite the task.
    myApp.services.tasks.update(element,
      {
        title: ...,
        category: ...,
        ...
      }
    );

    ...
  }
});
```

課題

- 「ドキュメントを読んでサンプルを変更」をできるようにしよう

- ons-navigatorで例を行う

課題（要提出）

- PBL日報の提出
- 前回同様の方法で班のリーダーがまとめて提出してください
- 提出先:
  nakai@yamamoto.ist.aichi-pu.ac.jp