

情報デザイン応用演習I 6.アプリ開発入門

目次

1. 初めに
2. アプリ開発入門
 - i. アプリとは?
 - ii. JavaScript
 - iii. Electron について
 - iv. Electronの導入について

初めに

「ECMAScript入門」「Canvas入門」「Canvas応用」「ThreeJS入門」やってから取り掛かりましょう。

ThreeJsFirstStepの説明

軽くおさらいしてみましょう。

アプリ開発入門

アプリとは？

人がやらせたい作業をしてくれるソフトウェアがアプリ・アプリケーションと言えるでしょう。

どのように作られるのか？

アプリケーションは、最終的に CPU が認識できるような言語 (機械語) にならないと CPU が実行することはできません。そのため、様々な言語がありますが以下の二つに分けることができます。

コンパイラ型言語

プログラムを言語で記載した後、あらかじめ機械語 に翻訳してから実行する方法

インタープリタ型言語

プログラムを実行すると、言語をリアルタイム に機械語に翻訳しながら実行する方法

コンパイラ型/インタープリンタ型メリット・デメリット

コンパイラ型の方は「コンパイル」という作業をするため作成には一手間かかりますが、最初に翻訳を済ませているため、動作が早くなる、という利点があります。

アプリケーションを開発する言語は...

たくさんありすぎて書ききれません。

いろんな言語で書くことができますのですが、向き不向きがあったりします。最近でもよく使われているのは

- Python
- JavaScript
- Java
- C++
- Ruby
- C#
- Swift

JavaScript

JavaScript

これまでずっと JavaScript にてプログラミングを行ってきましたが、これはインタープリタ型です。

コンパイル、ってしてないでしょ？

ちょっとこの歴史について触れておきたいと思います。

JavaScript の歴史

- 1995 年、JavaScript の前身となる LiveScript がネットスケープナビゲーターというブラウザに実装され、「Java」の勢いに便乗しようと「JavaScript」と改名
- 1996 年、IE には JavaScript に似た「JScript」が搭載。ブラウザによって異なる言語を利用することに
- 1997 年、ブラウザに依存しないように ECMA という標準機関ができるが難航
- 2000 年代前半、Flash が普及したため、JavaScript は要らないのではという扱い
- 2005 年、JavaScript の Ajax という技術を使った GoogleMap で世の中を驚かす
- 2000 年代後半 ライブラリの普及や V8 エンジンが開発され、Web になくてはならないものに
- 2009 年、それまでクライアントでしか動かなかったが、サーバサイドで JavaScript が利用できる Node.js が登場。

言語の確固たる地位を築いて現在に至る。

JavaとJavaScript

全く違う言語ですので気をつけましょう。

Electron について

ソフトウェアフレームワーク

ソフトウェアフレームワークとはアプリの開発環境です。

JavaScript を用いたソフトウェアフレームワークとして

- Electron
- NW.js

これらを利用することによって、デスクトップアプリケーションを作成することができます。どちらも

- Node.js
- WebKit
- という技術を利用しています。

Electronで開発されているアプリケーション

- Atom
- Visual Studio Code
- Microsoft Teams
- Slack
- Facebook Messenger
- え？と思うようなアプリまでElectronで開発されています。

Electron とは?

GitHub 社が Atom というエディタを作成するため作成したクロスプラットフォームデスクトップアプリケーションエンジンです。

クロスプラットフォームというのは、Mac でも Windows, Linux 用のアプリケーションを作成できることを意味します。

Electronの導入について

パッケージ

通常のアプリとは異なり、こういう開発とかになると、ターミナルを操作することが必要になることが多いです。ターミナルでは GUI レベルではなく、コマンドレベル (UNIX) のツールを使います。コマンドレベルのツールを管理するものに「パッケージ管理システム」を利用します。

各種のソフトウェアの導入と削除、そしてソフトウェア同士や ライブラリとの依存関係を管理するシステムである。(Wiki)

ちょっとややこしいですが

1. Electron を使うには npm と呼ばれる Node.js パッケージ管理システムが必要で
2. Node.js の利用には HomeBrew と呼ばれる Mac のパッケージ管理システムが必要となります。

Electronのインストール

1. HomeBrew のインストール
2. Node.js のインストール
3. Electron のインストール

この 3 つで Electron の導入が可能となります。

1.HomeBrew のインストール

<https://brew.sh/>

にアクセスして、日本語にして、インストールと書いてある行をコピーします。
ターミナル開いて、ペーストしてリターンを押してください。

最初の\$はいらないです。

2.Node.js のインストール

引き続き、ターミナルにて

```
$ brew install node
```

とします。確認するには

```
$ which node  
/usr/local/bin/node  
$ which npm  
/usr/local/bin/npm
```

\$の行を入力して、リターンを押して、表示されることを確認してください。

3.Electron のインストール

```
$ npm install electron -g
```

としてから、同じく

```
$ which electron  
/usr/local/bin/electron
```

\$の行を入力して、リターンを押して、表示されることを確認してください。

Electronの導入ができたね。

それではElectronFirstStepに行きましょう。

※：AppleSiliconとかバージョンアップとかで、buildまでエラー出たら、できる限り対処します。