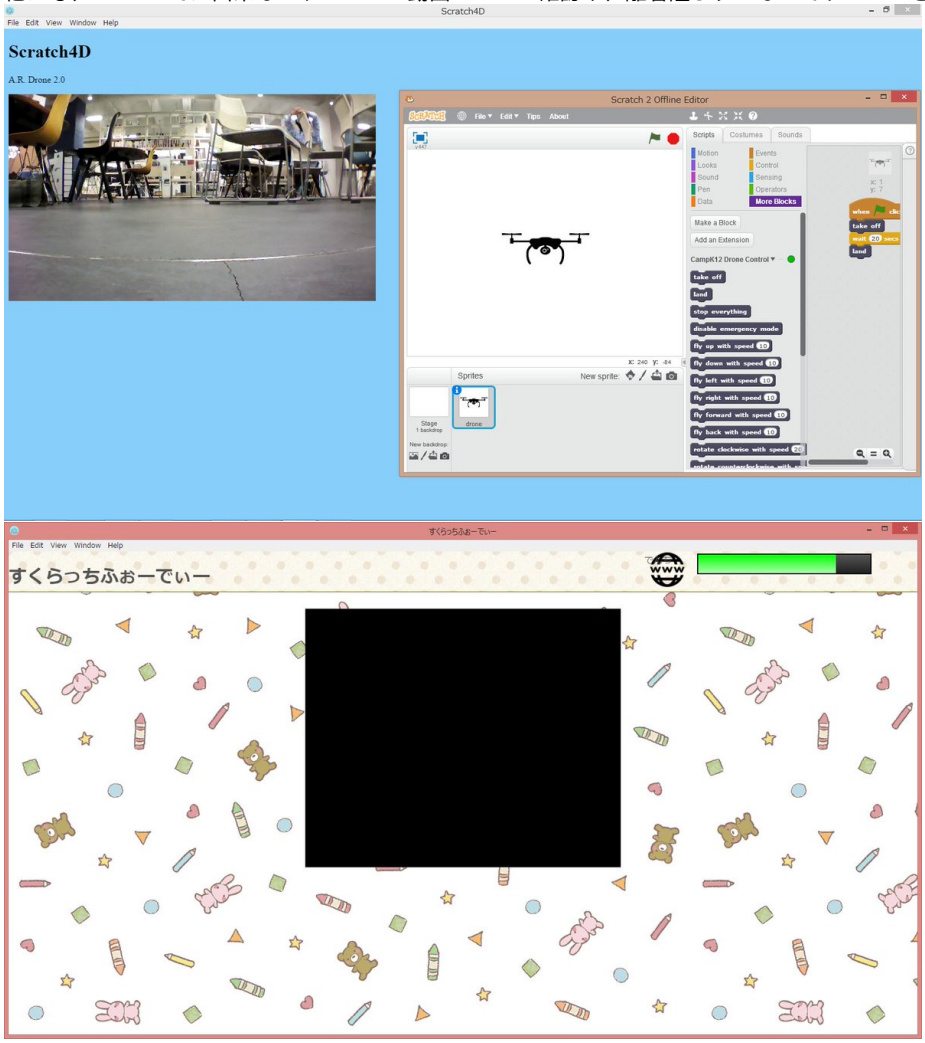
Contents

1 Scratch4D

Scratch4DはScratch HTTP Extensions & electronを使ってscratchでドローンを操縦できるようにしたソフトウエアです。
実行ファイルをダブルクリックするだけで起動できます。

他にも、Scratchでは出来ないドローンの動画ビューの確認や、離着陸ボタンなどでドローンを操作することもできます。



2　Scratch4Dの使い方

2.1　Windows

1. Scratch4Dをダウンロードして、実行ファイルをクリックして起動します。
   https://drive.google.com/file/d/0By3Sz9jl9bM3Z1l6eE1UbkhrdTQ/view?usp=sharing
2. Scratch4Dを解凍して、フォルダ内のScratch4D.exeをダブルクリックして実行してください。
3. Scratch Editorで.sb2プロジェクトファイルをHTTP Extensionsからロードしてください。

Scratchのドローンブロックが使用できるようになります。
Scratch Editorのランプが緑色になっていたらドローンへの接続に成功しています。
https://github.com/kendemu/Scratch4D/raw/master/project.sb2

2.2 Mac

1. Scratch4Dのgithubのレポジトリをクローンしてください。

   ```
   git clone https://github.com/kendemu/Scratch4D
   ```

2. NVMをインストールしてください。

   ```
   wget -qO- https://raw.githubusercontent.com/creationix/nvm/v0.33.0/install.sh | bash
   ```

3. NVMでNode.jsのバージョンをv6.9.0 LTSにしてください。

   ```
   nvm use v6.9.0
   ```

4. 動画エンコーディングライブラリffmpegをインストールしてください。

   ```
   brew install ffmpeg
   ```

5. Scratch4Dのディレクトリ下でelectronプロジェクトをビルドしてください。

   ```
   npm -g i asar
   npm run build
   ```

6. Scratch4Dのディレクトリ下でelectronプロジェクトをMacプラットフォーム用にデプロイしてください。

   ```
   npm -g i electron-packager
   npm run deploy:mac
   ```

7. Scratch4D-(プラットフォーム名)-(CPUアーキテクチャ名)という名前のディレクトリが生成されます。
   そのディレクトリ下でプログラムを実行してください。

8. Scratch Editorで.sb2プロジェクトファイルをHTTP Extensionsからロードしてください。
   Scratchのドローンブロックが使用できるようになります。
   Scratch Editorのランプが緑色になっていたらドローンへの接続に成功しています。
   https://github.com/kendemu/Scratch4D/raw/master/project.sb2

2.3 Linux

1. Scratch4Dのgithubのレポジトリをクローンしてください。

   ```
   git clone https://github.com/kendemu/Scratch4D
   ```

2. NVMをインストールしてください。

   ```
   wget -qO- https://raw.githubusercontent.com/creationix/nvm/v0.33.0/install.sh | bash
   ```

3. NVMでNode.jsのバージョンをv6.9.0 LTSにしてください。

   ```
   nvm use v6.9.0
   ```

4. 動画エンコーディングライブラリffmpegをインストールしてください。

   ```
   sudo apt-get install ffmpeg
   ```

5. Scratch4Dのディレクトリ下でelectronプロジェクトをビルドしてください。

   ```
   npm -g i asar
   npm run build
   ```

6. Scratch4Dのディレクトリ下でelectronプロジェクトをLinuxプラットフォーム用にデプロイしてください。

   ```
   npm -g i electron-packager
   npm run deploy:linux
   ```

7. Scratch4D-(プラットフォーム名)-(CPUアーキテクチャ名)という名前のディレクトリが生成されます。
   そのディレクトリ下でプログラムを実行してください。

8. Scratch Editorで.sb2プロジェクトファイルをHTTP Extensionsからロードしてください。
   Scratchのドローンブロックが使用できるようになります。
   Scratch Editorのランプが緑色になっていたらドローンへの接続に成功しています。
   https://github.com/kendemu/Scratch4D/raw/master/project.sb2

2.4 ARM Linux

1. Scratch4Dのgithubのレポジトリをクローンしてください。

2. NVMをインストールしてください。

   ```
   wget -qO- https://raw.githubusercontent.com/creationix/nvm/v0.33.0/install.sh | bash
   ```

3. NVMでNode.jsのバージョンをv6.9.0 LTSにしてください。

   ```
   nvm use v6.9.0
   ```

4. ffmpegをインストールしてください。ARM Linux用パッケージがaptレポジトリにないのでソースインストールしてください。
   http://www.jeffreythompson.org/blog/2014/11/13/installing-ffmpeg-for-raspberry-pi/

5. Scratch4Dのディレクトリ下でelectronプロジェクトをビルドしてください。

   ```
   npm -g i asar
   npm run build
   ```

6. Scratch4Dのディレクトリ下でelectronプロジェクトをARM Linuxプラットフォーム用にデプロイしてください。

   ```
   npm -g i electron-packager
   npm run deploy:arm
   ```

7. Scratch4D-(プラットフォーム名)-(CPUアーキテクチャ名)という名前のディレクトリが生成されます。
   そのディレクトリ下でプログラムを実行してください。

8. Scratch Editorで.sb2プロジェクトファイルをHTTP Extensionsからロードしてください。
   Scratchのドローンブロックが使用できるようになります。
   Scratch Editorのランプが緑色になっていたらドローンへの接続に成功しています。
   https://github.com/kendemu/Scratch4D/raw/master/project.sb2

3 Scratch4Dの仕組み

Scratch4Dは「初心者プログラマでも簡単にドローンプログラミングができる」ことを目的として作られたソフトウエアです。

3.1 初心者プログラマでも簡単にドローンプログラミングができるための工夫

1. Windowsでのサポートに最も力を入れました。
   初心者プログラマは多くの場合Windowsを使っています。

2. アイコンのダブルクリックでソフトウエアを起動できるようにしました。
   初心者プログラマは、環境構築が嫌いです。

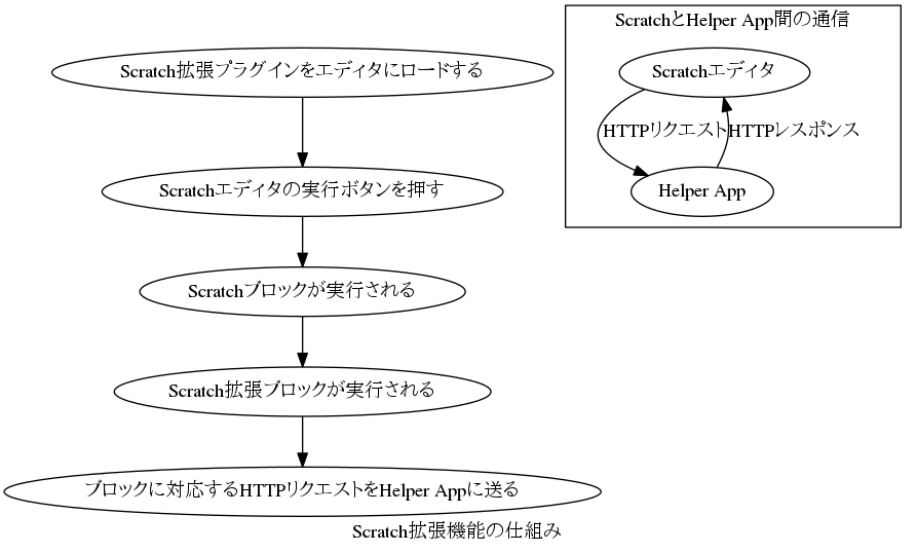
3. 動画ビューやドローンの操作がGUIでできるようにしました。
   初心者プログラマは、CUIやコマンドラインが嫌いです。GUIが大好きです。

3.2 Scratch Extensions

ScratchにはScratchXと呼ばれる、拡張機能があります。
この拡張機能を使うことによって,
ScratchでArduino, Kinect, Leap Motionなどのハードウエアと
連携させることができるようになります。

3.2.1　Scratch拡張機能の仕組み

以下のようなフローでScratch4Dは拡張機能を実現しています。



Scratch拡張機能の仕組み

3.2.2　Scratch拡張機能を実現する方法

拡張機能を実現するためには、2つのことを行う必要があります。
1. Scratchの拡張プラグインをJSON形式で作成する。
2. Scratch Helper Appと呼ばれるプログラムを作る。

3.2.3　拡張プラグインを作る

プラグインは拡張子が.s2eのJSON形式ファイルになります。
プラグインの公式仕様書を読んで作ってください。
https://wiki.scratch.mit.edu/w/images/ExtensionsDoc.HTTP-9-11.pdf
サンプルとして、Arduino For Scratchのプラグインを貼ります。

```
{
"extensionName": "A4S (Arduino For Scratch)",
"extensionPort": 12345,
    "blockSpecs": [
        [" ", "set pin %n high", "pinHigh", 13],
        [" ", "set pin %n low", "pinLow", 13],
        [" ", "set pin %n as output", "pinOutput", 13],
        [" ", "set pin %n as input", "pinInput", 13],
        [" ", "set pin %n as %m.mode", "pinMode", 2, "Digital Input"],
        [" ", "digital write pin %n %m.highLow", "digitalWrite", 13, "high"],
```

```
        [" ", "analog write pin %n value %n", "analogWrite", 3, 255],
        [" ", "servo write pin %n degrees %n", "servoWrite", 5, 180],
        ["b", "digital read pin %n", "digitalRead", 2],
        ["r", "analog read pin %n", "analogRead", 0],
    ],
    "menus": {
        "mode": ["Digital Input", "Digital Output","Analog Input","Analog Output(PWM)","Serv
        "highLow": ["high", "low"],
    },
}
```

3.2.4 Scratch Helper Appを作る

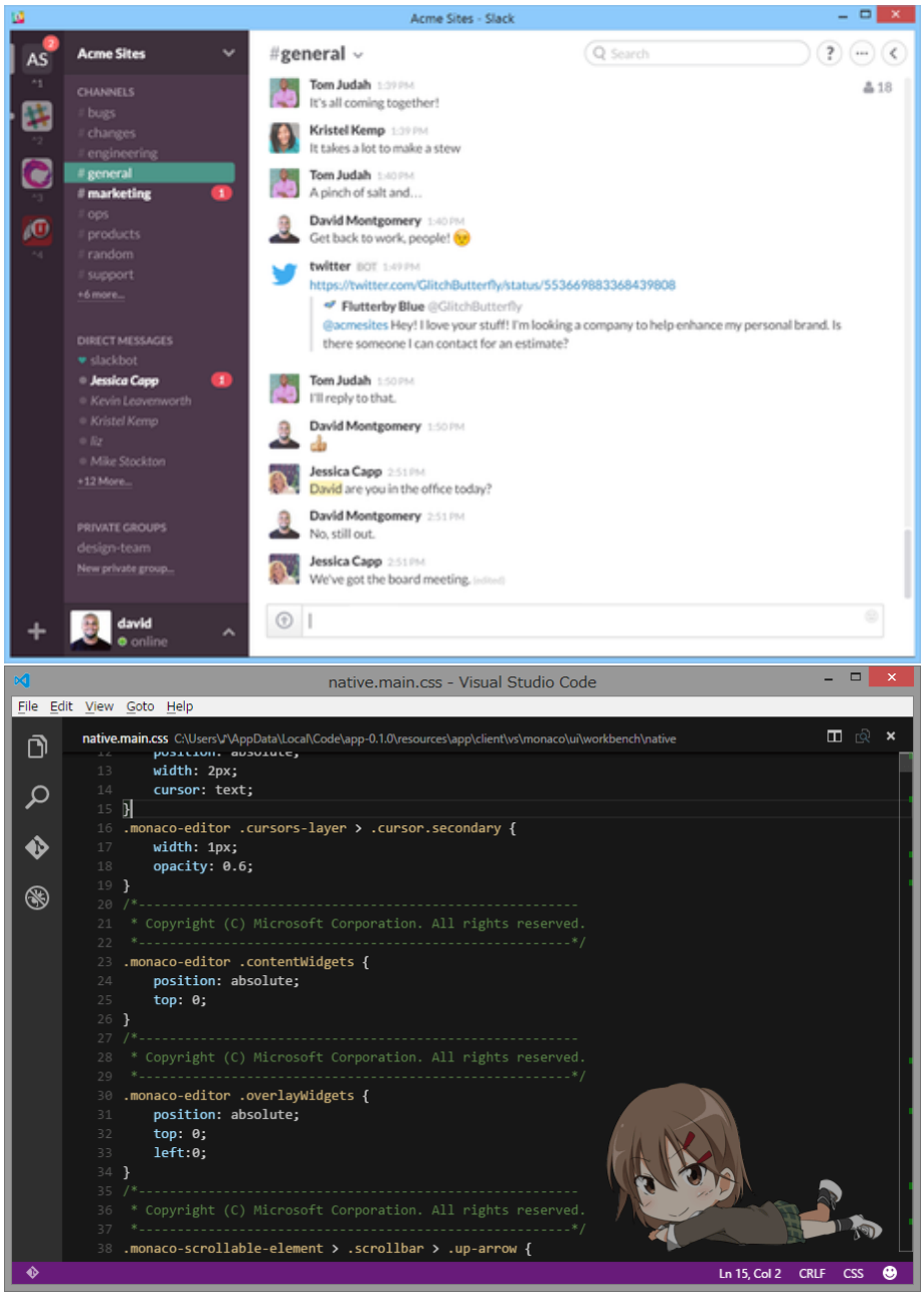
Scratch Helper Appは、Scratchとハードウエアのインターフェースの役目をします。
Scratch Editorから送信されたHTTPリクエストを元に、ハードウエアを操作します。
Scratch Helper Appは、HTTPリクエストを処理し、HTTPレスポンスを返すことができれば、
どのようなプログラミング言語でも実装可能です。
Javaで書かれていることが多いです。
（多分HTTPやGUIの実装がしやすく、プログラミングしやすいため）

4 Electron開発

今回はElectronでHelper Appを実装します。

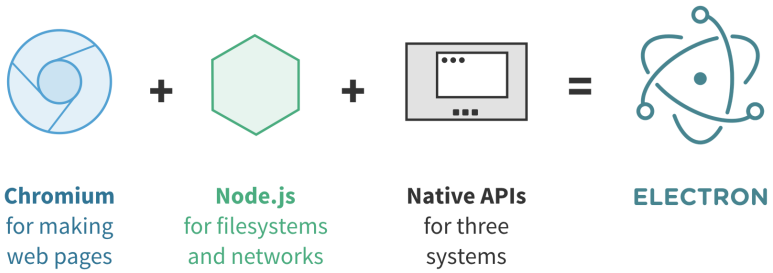
4.1 Electronとは

Electronはデスクトップアプリを開発するためのフレームワークです。
SlackのデスクトップアプリやMicrosoft Visual Studio CodeがElectronを使って作られています。


Acme Sites - Slack
Acme Sites
CHANNELS
bugs
changes
engineering
general
marketing
ops
products
random
support
DIRECT MESSAGES
slackbot
Jessica Capp
Kevin Leavenworth
Kristel Kemp
liz
Mike Stockton
PRIVATE GROUPS
design-team
#general
Tom Judah
It's all coming together!
Kristel Kemp
It takes a lot to make a stew
Tom Judah
A pinch of salt and…
David Montgomery
Get back to work, people!
twitter
https://twitter.com/GlitchButterfly/status/553669883368439808
Flutterby Blue @GlitchButterfly
@acmesites Hey! I love your stuff! I'm looking a company to help enhance my personal brand. Is there someone I can contact for an estimate?
Tom Judah
I'll reply to that.
David Montgomery
Jessica Capp
David are you in the office today?
David Montgomery
No, still out.
Jessica Capp
We've got the board meeting.
david
online
native.main.css - Visual Studio Code
File Edit View Goto Help
native.main.css C:\Users\r\AppData\Local\Code\app-0.1.0\resources\app\client\vs\monaco\ui\workbench\native
width: 2px;
cursor: text;
}
.monaco-editor .cursors-layer > .cursor.secondary {
width: 1px;
opacity: 0.6;
}
/*---
* Copyright (C) Microsoft Corporation. All rights reserved.
--/
.monaco-editor .contentWidgets {
position: absolute;
top: 0;
}
/*---
* Copyright (C) Microsoft Corporation. All rights reserved.
--/
.monaco-editor .overlayWidgets {
position: absolute;
top: 0;
left:0;
}
/*---
* Copyright (C) Microsoft Corporation. All rights reserved.
--/
.monaco-scrollable-element > .scrollbar > .up-arrow {
Ln 15, Col 2
CRLF
CSS

4.1.1　Electronの仕組み

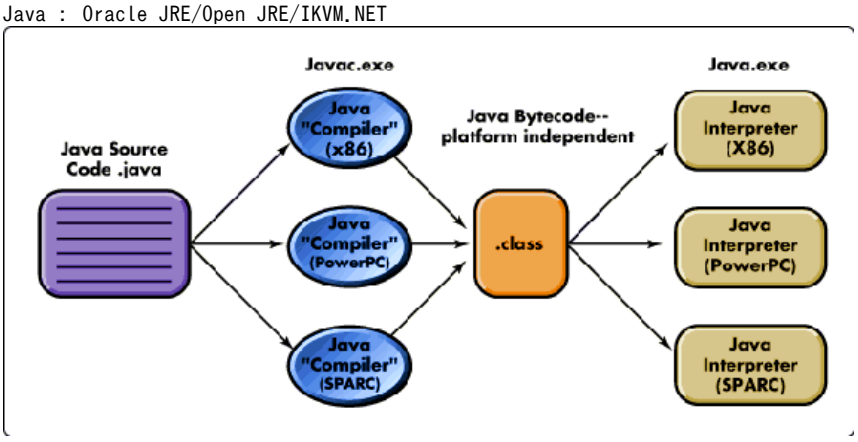
ElectronはchromiumブラウザとNode.jsを合体させたフレームワークです。
GUIはWEB技術のHTML5で設計でき、他のプログラムはNode.jsで作ります。



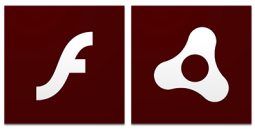
4.1.2　ランタイム/実行環境とは

ランタイムはソフトウエアを実行するために必要な部品のことです。実行環境とほぼ同義の意味です。
JavaはJRE(Java実行環境)が必要です。FlashはFlash PlayerかFlash AIRが必要です。
実行環境/ランタイムには複数の特徴があります。

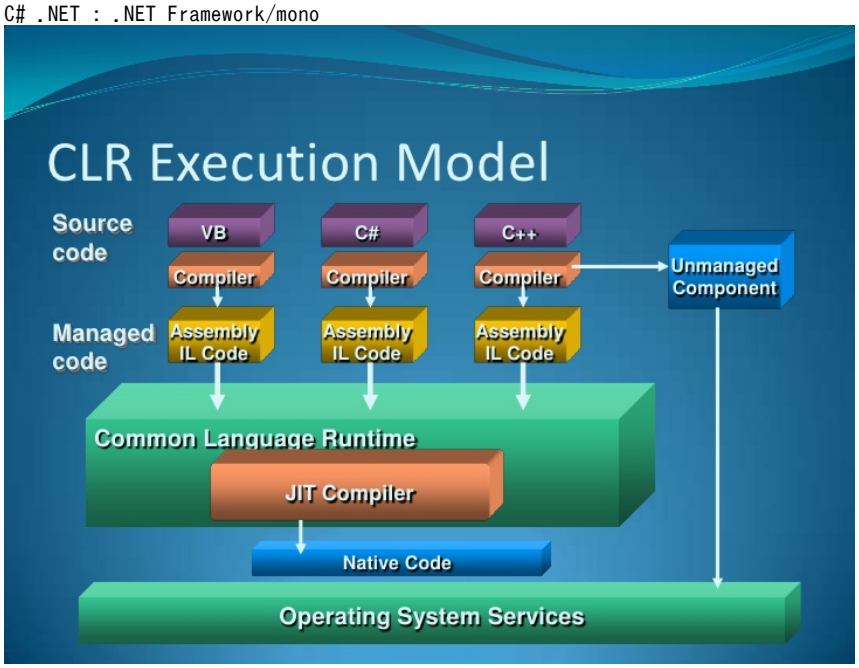
1. 複数の実装/種類のランタイムが存在する。 例:
   Java ： Oracle JRE/Open JRE/IKVM.NET



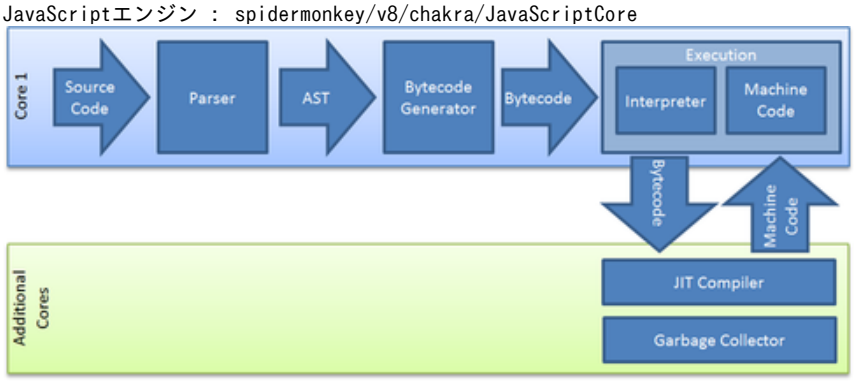
   Python実行環境 ： CPython/IronPython/PyPy/Jython
   Flash ： Flash player/Flash AIR



C# .NET : .NET Framework/mono



JavaScriptエンジン : spidermonkey/v8/chakra/JavaScriptCore



JavaScriptサーバーサイド実行環境 : Node.js/Jxcore/io.js

2. ランタイムが必要なソフトウエアはクロスプラットフォームである場合が多い。
   Java : 30億のデバイスで走るJava(組み込み/Windows/Linux/Mac)
   C# : 組み込み, Windows, mac, Linux
   Python : Windows, mac, Linux
   ブラウザJavaScript : Windows, mac, Linux, IE, Chrome, Safari, Opera
3. ランタイムが必要なソフトウエアはランタイムがないと動かない。

Flash ： Flash Playerがないと動かない
Java ： JREがないと動かない
C# ： .NETがないと動かない
JavaScript ： ブラウザかNode.jsがないと動かない

4.1.3 Electronを使うとうれしいこと

Scratchを使う初心者プログラマの環境構築したくない・GUI以外使いたくない、
デベロッパーのアプリケーションを簡単にプログラミングしたい気持ちが、
Elecronでマッチングできます。
Electronを使用すると、Node.jsとchromiumブラウザが合体したアプリケーションを,
Node.jsなどのランタイムなしで、アイコンのダブルクリックだけで起動することができます。
Electronにはクロスコンパイルのような仕組みがあるため、
mac, windows, linuxネイティブなアプリケーションをJavaScriptコードから生成できます。
Javaはユーザーが動かすためにJREが必要ですが、
Electronはユーザーは何もインストールする必要がありません。ダブルクリックだけです。

4.2 ElectronでHTTPサーバー、メインロジックをプログラミングする

Node.jsでプログラミングします。
Node.jsとは
Node.jsはサーバーサイドで JavaScriptを実行できるようにした実行環境です。 本来-
JavaScriptはWEBブラウザ上でしか動きませんでしたが、
Node.jsを使うとJavaScriptでPythonやJavaと同じように汎用的なプログラムが作れます。
コーディングされたJavaScriptがシングルスレッドで非同期に実行されることが特徴です。
少ないコンピューター資源でサーバーが作れることで有名です。
C++でコーディングされたJIT VMのGoogle V8 Engineを使用しているため、
高速でJavaScriptを実行することができます。
最近は組み込み機器とNode.jsを組み合わせてIoTにも使われています。
主にWEBサーバー開発やWEBフロントエンド開発に使用されます。

4.2.1 Node.jsの仕組み

Node.jsには3つの仕組みがあります。
1. 簡単に軽量なサーバープログラムが作れる
2. NPMパッケージマネージャーを使って豊富なライブラリをすぐに使用することができる
3. npm scriptで開発工程が自動化・効率化できる

4.2.2 サンプル：Node.jsでHTTPサーバーを作る

```
const http = require('http');
http.createServer(function (req, res) {
res.writeHead(200, {'Content-Type': 'text/plain'});
```

```
    res.end('Hello World\n');
}).listen(8080);
console.log('Server running at http://127.0.0.1:8080/');
```

4.3 Electron UI開発

HTML5でプログラミングします。
Electronは同じコードで複数のプラットフォーム用にデスクトップアプリを配布できます。
WEB UIの知識が必要です。

4.3.1 コンピューターのUIの種類

4.3.1.1 CUI

CUIは文字列を入力として、文字列を出力として表示するUIです。
GUIが一般的になる前は、一般人は嫌々ながらも使っていたそうです。
キャラクターベースのため、出力は1行ずつになります。
一般人が大嫌いなUIです。

4.3.1.2 TUI

TUIはターミナル上で1度に画面全体に色付きで出力できるインターフェースです。
BIOSやMS-DOSゲーム(ロックマンなど)がTUIソフトウエアとして有名です。

4.3.1.3 GUI

GUIはCGとポインティングデバイスを使用して、直感的な操作を提供するUIです。
CUI/TUIとは比較にならない可視化機能を持っています。
タスクの可視化 ： タスクウィンドウ
コマンドの可視化 ： メニュー、ボタン
データの可視化 ： アイコン
一般人が頭で考えると難しいタスク、コマンド、データなどの概念を、
直感的に使用できるようにしている非常に優れたUIです。
GUIを実現するには、2つのソフトウエアが必要です。
1. ウィンドウシステムと呼ばれる、タスクをウィンドウとして管理し、
ポインティングデバイスの入力を処理するシステム
2. デスクトップ環境と呼ばれる、
アイコン、ウィンドウ、ツールバーなどのUI(見た目)を司るシステム

4.3.2 GUIプログラミングとプラットフォーム依存

GUIプログラミングは非常にプラットフォームへの依存性が高いです。
なぜなら、描画のAPIがOSに依存しているからです。

WindowsはXAML/WPF, MacはCocoa, LinuxはGTK・Qt, AndroidはSurfaceなどの別々なフレームワークで実装する必要があります。
そのため、通常はGUIアプリを複数のプラットフォームでローンチするためには、
複数のGUIフレームワークを覚えてプログラミングする必要があります。
非常に面倒です。

4.3.3 WEBアプリ, ネイティブアプリとハイブリッドアプリ

スマホの到来によって、スマホアプリの開発が盛んになりました。
そこで、アプリ開発の新しい手法が注目されるようになりました。

4.3.3.1 WEBアプリ

WEBページでネイティブアプリケーションのような複雑な動作やUIを実現しているソフトウエアです。
HTML5の登場により、注目されるようになりました。
WEB用のコードだけで、ブラウザがあるすべてのデバイスで動作します。
bootstrapなどのフレームワークを使用すると、UIデザインもすべてのデバイスで共通化できます。

4.3.3.2 ネイティブアプリ

デバイスにインストールして使用するアプリです。
プラットフォーム依存のアプリです。
ネイティブアプリをiOSとAndroid両方にローンチしたい場合は、
iOS版とAndroid版のアプリのコードを両方書かなければいけません。

4.3.3.3 ハイブリッドアプリ

WEBアプリとネイティブアプリ両方の特徴を持ったアプリです。
デザインとHTTPサーバーの通信はWEBアプリで共通化し、
ハードウエアやプラットフォーム特有の機能を使用したい部分はネイティブアプリのコードで作れます。
アプリ開発のコストを軽減しながら、アプリの性能も追求することができます。

4.3.4 WEBとUIの歴史

WEB UIはノウハウ、フレームワークやドキュメントがネイティブUIに比べて充実しています。
もともとWEBは研究者が文書を共有するために生まれました。
そのため、WEBが生まれた当初は文字しか表示できなかったのですが、
時間が経つにつれて画像の表示や、動的なWEBサイトの作成ができるようになりました。

4.3.4.1 初期のWWW

もともとWWWは研究者が文書を共有するために生まれました。
そのインターネットで共有する文書のフォーマットがHTMLです。
そのため、当初のHTMLは文字情報だけしか使用できませんでした。

4.3.4.2　初期のブラウザ

最も初期のブラウザはHTTPではないネットワーク上にあるテキストファイルを
観覧するためのCUIソフトウエアでした。
1992年にWWW対応のブラウザが生まれました。
WWW対応のViolaWWWやLynxブラウザ（現在もあります）はGUIではなく、
TUIで作られていました。

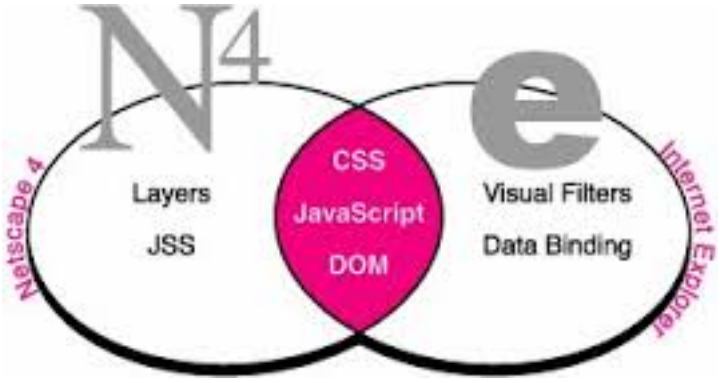
4.3.4.3　マルチメディアとGUIブラウザの台頭

1992年にWindowsで動く世界初のGUI WEBブラウザCelloが発表されました。
そして、1993年に文書の中に画像を表示できるブラウザMosaicが発表されたことで、
WEBブラウザアプリケーションが爆発的に一般人に使用されるようになりました。
今までのブラウザは画像のリンクをクリックすると、画像をダウンロードして、
その画像を別のアプリケーションで見るという仕組みになっていました。
その当時の他のブラウザは、
ブラウザ上で表やチャートなどの論文に必要なものをを表示する方針でしたが、
Mosaicは一般人にマルチメディアコンテンツを提供することに注目していました。
1994年にMosaicチームのリーダーがNetscape社を設立、Navigatorを発表しました。
1995年にブラウザのデファクトスタンダードになりました。

4.3.4.4　ブラウザ戦争と拡張HTML

MicrosoftがSpyglass社からInternet Explorerを買収し、発表したことで勃発しました。
ブラウザ同士の競争が盛んになり、ブラウザごとの独自HTML拡張の実装が盛んになりました。
1996年にIEは世界で初めてブラウザにCSSを実装。
Netscapeはブラウザにcookie, frame, JavaScriptを導入し,
W3C(WWWの標準化団体)やECMAで標準化されたことによって,
WEBアプリケーションの元となる技術の一つ「DHTML」が誕生しました。

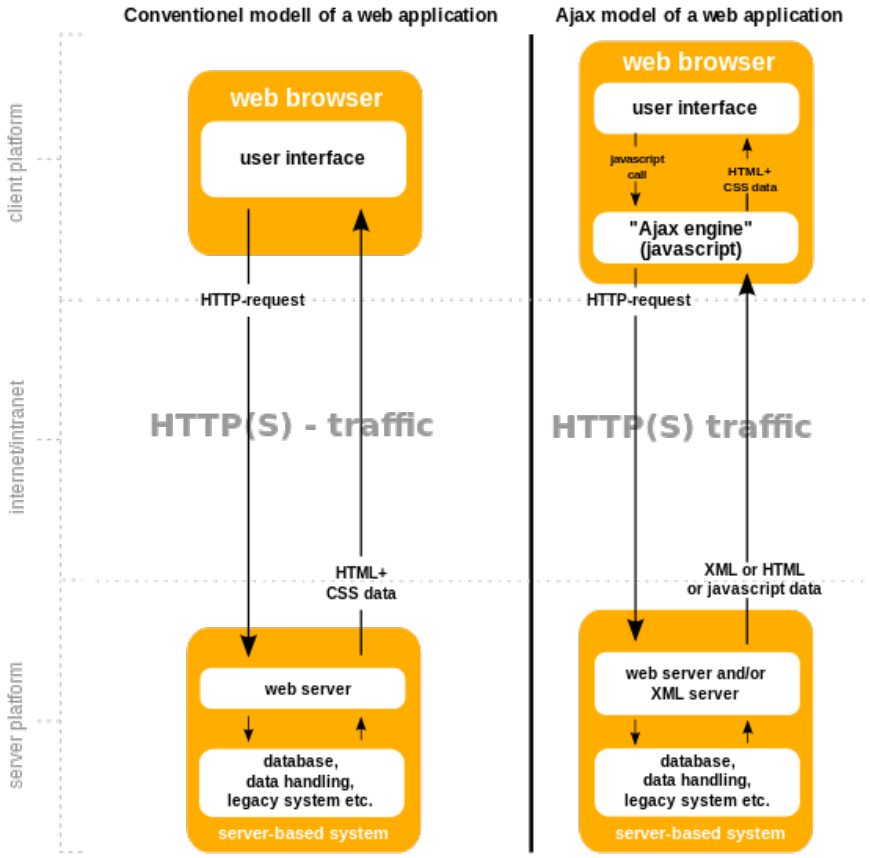
4.3.4.5　DHTMLとインタラクティブWEBページ



DHTMLはブラウザ戦争、ブラウザの機能拡張競争によって生まれた技術です。
HTMLだけでは静的なWEBページしか作成できません。

http://abehiroshi.la.coocan.jp/
DHTMLは、JavaScriptやCSSを使ってHTMLを動的に書き換えます。
WEB上でアニメーションやゲーム、ボタンをクリックして遷移するアプリケーションを
作ることを可能にしました。 http://vincentgarreau.com/particles.js/

4.3.4.6 Ajaxとは



Ajaxは2005年から注目された、ウェブブラウザ内でHTTPの非同期通信を行いながらUIの変更を行う技術です。
Google Mapや初期のFacebookがAjaxを使ったWEBアプリケーションとして有名です。
Ajaxを使用すると、ページの再読み込み無しでインターネットから
新しいコンテンツをページが表示することができます。
Google Mapがもしajaxを使用していなかったら、
場所を移動するときにいちいちページを再読み込みする必要がありました。
Ajaxを使うと再検索や繰り返してサーバーと通信するときにページを再読み込みする必要がありません。

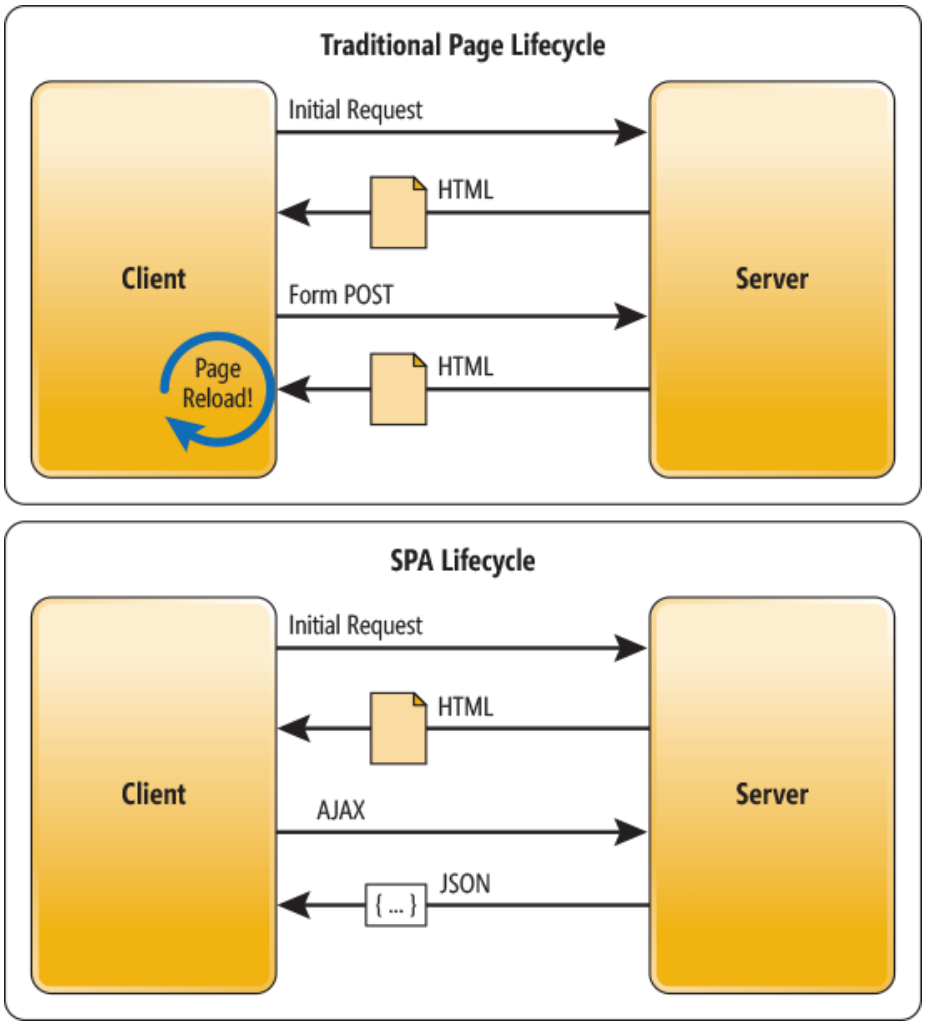
例
1. google mapで場所を移動したときにページを再読み込み無しでマップを読みこむ

2. google検索でリアルタイムで検索結果を表示する
3. Facebookで下にスクロールすると自動で投稿を読み込む
Ajaxによって、 WEBアプリケーションが爆発的に増えました。
DHTML + AjaxによるインタラクティブWEBページを作るためのフレームワークでは、
jQueryが有名です。

4.3.4.7 レスポンシブデザインとは

レスポンシブデザインは2011年に登場した、
様々な種類のスマホ・PC・タブレットに対して、
WEBページの見た目や操作方法が最適化されたサイトを作るWEBデザインの手法です。
レスポンシブデザインフレームワークでは、bootstrapが有名です。
SPAと組み合わせることで、デザインレベルで1つのWEBアプリケーションだけで
様々なデバイスに対応することができます。
Scratch4Dの次期バージョンはレスポンシブデザインに対応します。

4.3.4.8 SPAとは



SPAは1つのページだけで構成されるWEBアプリケーションです。
デスクトップアプリケーションに「ページ」というものは存在しませんよね。
SPAはデスクトップアプリケーションのような振る舞いを持ちます。
Ajaxが生まれたため、SPAが実現できるようになりました。
画面遷移はDHTMLで、サーバーとの通信はAjaxで行います。
モバイルWEBアプリなどはSPAで実装します。
SPAフレームワークはReact.jsが有名です。

5　Scratch4Dの設計と実装

5.1　システム図

このシステム図には6つの役割が存在します。

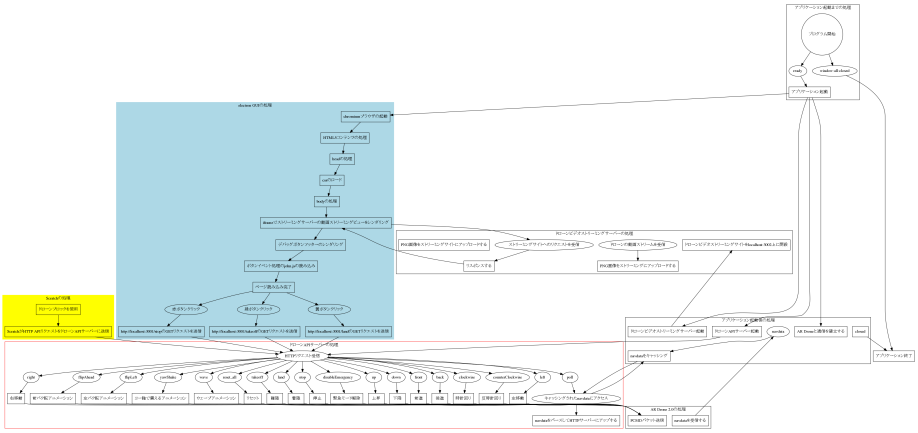


Figure 1: scratch4d

1. Scratch Editorクライアント
2. Electron起動プロセス
3. Electron BrowserWindow(chromiumブラウザ)
4. ドローン動画ストリーミングサーバー
5. ドローンHTTP API制御サーバー
6. ドローン

5.2　起動プロセス

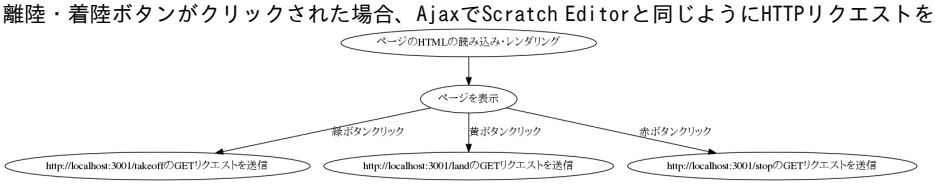
Electronメインプロセスが UI, ストリーミングサーバーとHTTP APIサーバーを同時に起動します。

5.3　Scratch Editor

Scratch Editorが拡張ブロックを実行するたびにScratch4Dに対してHTTPリクエストを送信します。

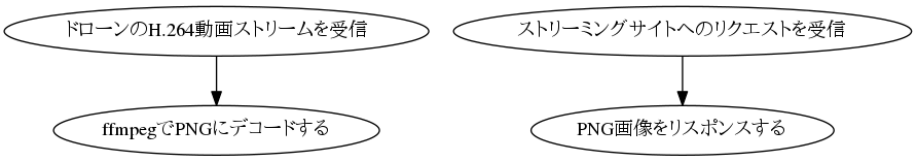
5.4　Scratch4D UIの処理

HTML5を使用します。
ドローン動画ストリーミングサーバーの動画をiframeで読み込み、CSSなどで装飾して表示します。
離陸・着陸ボタンがクリックされた場合、AjaxでScratch Editorと同じようにHTTPリクエストを送信します。



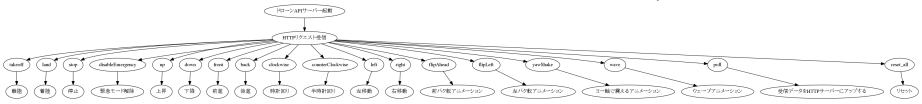
5.5　Scratch4D 動画ストリーミングサーバーの処理

ドローンの動画ストリームを受信したときに画像を保存します。
ストリーミングサーバーへのHTTPリクエストを受信すると、HTTPレスポンスとして画像を送信します。
動画のエンコーディング・デコーディングはffmpegを使っています。
A.R. DroneのH.264動画ストリームをPNGフォーマットに変えてストリーミングしています。



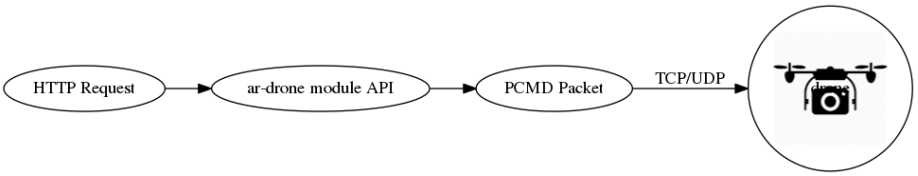
5.6　Scratch4D HTTP APIサーバーの処理

Ajaxを使ったHTTPリクエストをサーバーサイドで処理するAPIのことをHTTP APIと呼びます。
今回はScratchのドローンブロック用のHTTP APIを実装しました。
ドローンのセンサ値をサーバーからクライアントに送信するために, HTTP Pollingを使用しました。



5.7　ドローンの操縦

TCP/UDPでA.R. Droneは操作できます。
A.R. DroneはPCMDパケットと呼ばれるパケットのデータグラムをTCP/UDPパケットに格納すれば操縦できます。
今回はNode.jsのar-droneモジュールを使って、コード上ではdrone.land()というAPIで
パケット作成＆パケット送信を実現しました。


HTTP Request
ar-drone module API
PCMD Packet
TCP/UDP