

※構想当時の資料です。

かなりぶちまけているので  
その慌てっぷりをご覧ください

## なにかを作ろう！（創刊号）

### 課題

- ・ 重要な特務機関(上級国民)にしか提供されない J-Alert や Em-Net まではいかなくとも信頼できるソースが欲しい
- ・ Elontter (エックタン) になった今、信頼できるソースからの Bot 情報に依存する時代は終わりを告げている
- ・ 一般庶民向けの携帯電話料金値下げ愚策による通信品質の低下で年々通信手段が限られてきている
- ・ 大げさに騒いで視聴率を狙うだけの報道もどきに正直うんざりしている

## 解決策

### みちびき(準天頂衛星)から絶えず送信される 災害・危機管理通報サービスを使えばいいのでは？

災害・危機管理通報サービスとは？

みちびき(準天頂衛星)から4秒おきに送出される気象庁の防災情報 XML や消防庁の J-Alert や Em-Net と同等の情報が得られるサービス

### 特徴

- ・ 初期費用は数万円以内(作り方によっては更に低価格化も可能)
- ・ 利用料は電気代のみ！(衛星の維持管理費自体は税金から出ているが NOK みたいに受信料がかからない)
- ・ 実質インターネット接続不要！
- ・ 日本国内で「空が見える屋外にアンテナが置ける環境」であればほぼどこでも受信できる！
- ・ 仕様書がある程度公開されているので機材をそろえれば誰でも扱える！（ただし、Q-ANPI などの秘匿化情報は除く）
- ・ 受信するだけなので無線技術士の免許が要らない(無線を用いて再配信する場合は必要)

### 目標

～自力で、苦しみ、低予算～

1. Oiita とか、 $\Delta it \Delta ub$  から丸パクリしてくるだけの「他人の成果横取り」は慎む
2. (Z 世代や AI 生成に) 負けない根気をもう一度
3. たとえ営利目的でなくても、収益性くらいは意識をした設計
4. ユーザビリティを意識した設計
5. 汎用性と拡張性を意識した設計

### 課題

- ・ 電源が必要なのに屋根のない環境でないと使えないという矛盾(※外付けアンテナで一応解決可能)
- ・ OO の大合併とかで市区町村が減んだり合併したりするとアップデートが必要
- ・ Web サーバー化するのであればセキュリティ上の懸念がある
- ・ 仮に商用化するとすると、各々のライブラリにかかるライセンスの扱いをどうするか悩む

## まずはざっくり仕様を決める

QZSS の L1S から送出される 250bit の災危通報 (MT43, MT44) を受信し、シリアルなどの形式で出力

↓ (ここまではハードウェアの基本的な仕様でできる)

シリアルなどから受信した 250bit の災危通報を解析し、コード内容を各 DB から抽出

↓ (ここで SQL などの DB を扱う知識が必要になった)

DB から抽出したコード内容を定型文に結合する

↓ (ここで出力方法を決めておかないとダメだね???)

結合した定型文を各々の形で出力

↓ (XML とか JSON とか)

何をどのように表示する???

こんな感じか・・・

## 「みちびき衛星から災害通報のシリアルを受信してシリアルなどの形で出力」する機材を決める

- u-blox MAX-M10S を使用して自作
  - 回路設計の勉強にはなるが、SoC が1個から入手できない制約があるため開発コストがギャンブル
  - 大量生産するのであれば相対的に安くなる(収益化する気があるならこれが最適解)
- GNSS 受信基板 GR-M10-B/S-B45 (PRIORIS)
  - 自ら自作開発して組み立てるよりは手軽ではあるが、人気の同人※ハードなので入手困難
  - ハード設計から実装までの部分を代行してくれているという意味では楽ではあるが
  - そもそも”同人活動”の趣旨とは全く異なってしまうため見送り！(ごめんなさい)
- Spresense (SONY)
  - 国内で最も手軽に入手でき、周辺機能もほとんど揃っているのも非常にお手軽である
  - ただし屋外設置不可の製品ため、アンテナ部に改造(保証不可)を施す必要がある
  - そして、一番高い。でも、機能面からするとコスパはよい(※ライセンス的に収益化は難しいが・・・)

とりあえずパパッと作るならくだらん意地なんか張ってないで  
おとなしく Spresense を選んでおけばいいんじゃないの????

という事で結局 Spresense にしました

# シリアルから 250bit 災危通報を解析、コード内容を各 DB から抽出し、DB から抽出したコード内容を定型文に結合する

## ワガママ

- ・複数の環境を経由せず、単独の環境で動作できるようなものがある
- ・地域リストの差し替えはテキストファイルを差し替えるくらいの簡単なものがある
- ・なるべく多くの環境で動かせるようにしたい(目標はジャンク箱の中に放置しているラズパイ Zero 上で動かす)

となると移植性では ODS(Excel)+SQLite+Python\*が楽かな??? ←あれ、趣旨がもう脱線している・・・

※Arduino を使うなら Processing でもできそうだが・・・まあうん。

## 表現方法の一例

- ・装置内で生成した映像をディスプレイ上に表示 (JR の Signadia がこの方式で、企画とまったく同じ)
- ・Web サーバーなどを使用してブラウザで表示 (理経の MAJA2000(J-Alert 専用受信機のセット)がこの方式)
- ・スマホに通知を飛ばす(これだと Twitter とやっている事はあまり変わらない)
- ・本体から自治体の防災無線のような TTS 機能は欲しい(J-Alert は AITalk (せいじ)なので全く同一はちょっと難しそう)

## プログラムに最低限必要な要素

- ・解析部と表示部はなるべく一緒にしたい
- ・HTML を表示する Web サーバー
- ・誰でも簡単に操作できる GUI
- ・汎用的な通信ができるシリアルポートや RS422 などの通信

ここで少し脱線して「Python ってマクロ的な文章書いて動かすアレでしょ？」なんて気持ちでググったら・・・

- ・シリアル通信できるよ (←うん。これはなんとなく知っている)
- ・Web サーバーも欲しい? もちろん Apache や IIS なんか無くても Python だけで動くよ (←えっ神すぎない?)
- ・GUI が欲しいの? それ、Python だけでできるよ! (←は?マジで???)
- ・埋め込みブラウザ? GUI に乗せればできるよ

## これ、Python で良くない??? (ほぼ採用)

### 受信した電文を喋らせたい

- ・gTTS(要は Google の TTS) : 喋れるけどオンライン環境じゃないとダメ(オフラインならば pyttsx3 というのがあるらしい)
- ・J-Alert の合成音声と同じエンジン: AITalk (せいじ)なので個人向けであれば 16,500 円(税込)で売っている(組み込みは 30 万円!)

利用方法	該当箇所	ライセンス形態
ストリーミングの一部として利用	1-1	個人ライセンス
音声ファイルだけをダウンロードさせる	2-2、5-5	禁止
Web ページの読み上げのみに使う	2-1	個人ライセンス
開発中にデバッグ(学習)目的で利用	2-5	個人ライセンス
放送設備に組み込んで使用する	5-7	個人商用ライセンス
音声を組み込んだアプリケーションを配布する	4-2	個人ライセンス
音声を組み込んだアプリケーションを販売する	3-2、4-1、4-3、4-4	個人商用または法人ライセンス

<https://aivoice.jp/usecase/> ユースケースより (要は音声だけを主目的で扱うのはダメという事らしい)

- ・ReadSpeaker (VoiceText) : 「おやおや?」の Show くんでおなじみのやつだけど、「価格はお問い合わせください」なのでボツ
- ・AquesTalk : ロイヤリティ込みのマイコンを売っているので手軽に手に入るし商用も OK! (聞き取りやすさでは難ありだけど)

音声合成 L S I A T P 3 0 1 1 F 1 - P U (ゆっくりな女性の音声)

<https://akizukidenshi.com/catalog/g/gI-06220/>

音声合成 L S I A T P 3 0 1 1 M 6 - P U (男性の音声)

<https://akizukidenshi.com/catalog/g/gI-06225/>

共通スペック	・ベースチップ: Atmel ATmega328P ・音声合成コア: AquesTalk pico ・入力データ: ローマ字音声記号列(ASCII) ・出力: 8KHzFS PCM 8bitPWM 出力 ・インターフェース: UART/I2C/SPI ・プリセットメッセージ: 15 メッセージ ・動作クロック: 内蔵
・3種類のシリアルインターフェース UART/I2C/SPI ・2種類のカスタマイズ可能なチャイム音付き ・端子の変化をトリガにプリセットメッセージを発生 ・15個のカスタマイズ可能なプリセットメッセージ ・話速やプリセットメッセージなどを実装後に設定可能 ・動作電圧: 2.5V~5.5V ・消費電流: 3.5mA(発声時) (3V 動作 typ): 1μA~0.5mA(待機時)	

これで商用ロイヤリティ込みなのに 1000 円弱とはちょっとすごい・・・!

(大手電機メーカーの電話機に採用されたのもよくわかる)

ということで、ゆっくりしていったね!(採用)

## 報知音をどうするか

### ○本物の音源を用いる場合

- ・ ETWS 音 : Android 8x 以降から ETWS (CBS) 用のソース  
「/packages/apps/CellBroadcastReceiver/res/raw」内に ogg ファイルで入っている  
※元々の音源は「ボーーーー」とか「ビギャー」という電子音だが、日本の各キャリアはこの ogg ファイルを「ビュッ! ビュッ! ビュッ! 地震です。」「テロリッ! テロリッ! 津波です。」に差し替えているはず。  
(※日本語版の音源はもちろん公開も配布もされていない)
- ・ 緊急地震速報チャイム音  
NHK 版 : 一般個人向けには配布していない(商用利用であれば審査の上、3,000 円+税で販売する模様)  
住所なんか教えたら「絵の動く板や箱あんだろ?金払えやゴラ」と刺客(職員)や定期振込用紙を送ってきそうだけど  
REIC 版 : 規定は NHK より細かくて規制が多いが、非会員でも 10,000 円+税払えばライセンスを発行する模様
- ・ 国民保護サイレン音  
利用許諾に関する内容が見当たらない。まあ、実装は当分先だからまあいいや  
(音の構造自体は正弦波の組み合わせで難しい物ではないので最悪自分で作るのはいいかも)

### ○疑似音源を用いる場合

- ・ 電子音  
マイコン上 : シンプルな物ならプログラム上で作れるが、音のバリエーションは少ない  
音源 IC : 設計の手間が省けるが、希少性が高く入手性が悪い(NPC 製などは特に)
- ・ ロイヤリティフリー音源
  - ・ NML 音源 : 放送業界などでおなじみの音源が証書付きのライセンスにて販売している最大手。  
リーズナブルな価格(1 曲 : 500 円~)も良い  
例 : [NSE-S082-37\(妖精あいさつ\)](#) [NSE-S071-0802\(Pin Pon 2\)](#) [NSC-SG-0825\(爽やかな音\(3\)\)](#) など

#### 使えそうな音源の候補

NSE-S481-098	非常ブザー	→各種警報に使えそう(ブザー断続音 1 1 秒)
SE-SE1-514	8bit アラート	→J アラートに使えそう(高めの 8bit 断続音 4 秒)
SE-SE1-520	8bit ジャンプ	→J アラートに使えそう(低めの 8bit 断続音 4 秒)
NSE-S302-56	アラーム(1)	→緊急地震速報に使えそう(スイープ音断続で日本版 ETWS 音に近い鳥肌感)
NSE-S231-53	銃 3	→地震情報 5 弱以上とか津波情報受信時に使えそう(短めのブブ音)
NSE-S173-69	Music Accent 69	→その他情報受信時(共同通信の新着音に似てる鐘の音)
NSE-S081-0201	サイレン 1	→津波情報受信時とか(低めのダダ音)
NSE-S071-0801	Pin Pon 1	→情報受信時(低めのピンポン)
NSE-S071-0802	Pin Pon 2	→情報受信時(高めのピンポン)
NSE-S071-0805	PipoPipoPipon	→警報解除(よく聞く定番の正解音)
NSE-S071-0806	BUU	→訂正時やエラー音(よく聞く定番の不正解音)
MN-S052-6101	Comical 5	→予報内容訂正時(ピピピピン)

- ・ STUDIO COM 音源 : 首都圏の鉄道や放送機器で聞きなじみのある音源が多数ある  
NML 音源と同じくリーズナブルな価格(1 曲 : 660 円)も良い

#### 使えそうな音源の候補

No. 135656	電子音系サイレン	→J アラート
No. 135655	低めの音質のサイレン音	→J アラート
No. 134434	ピンポンパンポン(注目、ベル、お知らせ)	→その他情報受信時
No. 134385	軽やか木琴打楽器音(アニメ、ジングル)	→地震情報 5 弱以上とか津波情報受信時
No. 131633	ピーピーピーピーピー(高音のお知らせ音)	→緊急地震速報
No. 134366	ファンファンファン(お知らせ、合図)	→訂正時やエラー音、通知
No. 131530	デジタルシンセのキラキラジングル	→訂正時やエラー音、通知

本題(準備編)

「シリアルから 250bit 災危通報を解析、コード内容を各 DB から抽出し、DB から抽出したコード内容を定型文に結合する」を Python で実現するには？

～Python を使う上での約束～

人気のあるオープンソース開発プロジェクトがそうであるように、Python には貢献者たちとユーザたちの活発なサポートコミュニティがあり、またこれらはほかの Python 開発者たちに、彼らのソフトウェアのオープンソースライセンスのもとでの利用も可能にしてくれています。

これはほかの人が既に挙げた共通(あるいは時折極めて稀有な)問題や、彼ら自身の解法による潜在的な貢献が共通の場所に蓄えられることによる恩恵によって、Python ユーザに共有と協調を効果的に行なうことの助けとなっています。

This guide covers the installation part of the process. For a guide to creating and sharing your own Python projects, refer to the [Python packaging user guide](#).

**注釈:** あなたが企業や組織のユーザであれば、多くの組織がオープンソースソフトウェアの利用と貢献に関する彼ら独自のポリシーを持っていることに気をつけてください。Python によって提供される配布とインストールのツールを利用する際には、そのようなポリシーを考慮に入れてください。

Python モジュールのインストール <https://docs.python.org/ja/3/installing/index.html> から引用

○準備

昔は落としても PATH しなきゃいけなかったのが、今では便利なストアアプリがあってワンクリックさえすれば何もする事はないので省略 Python にエディタは無いので各々で用意すると書きやすい模様 (Notepad++とか VSCode とか)

Microsoft ストア版 Python 「PATH を設定すると言ったな」  
僕「そうだ！さっさと動けよ！」

Microsoft ストア版 Python 「アレは嘘だ。」  
僕「Ahhhhhhhhh(時間返せ！)」

という事になって、ローカルアクセスができない事があるから  
インストールする時はポンポン押さないように気を付けようね！

とある日の発狂

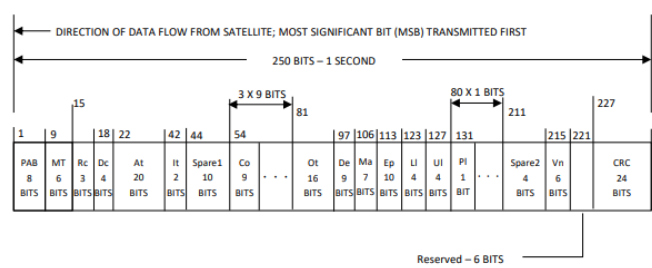
# とりあえず要件仕様を決めてコードを書いてみる

## 必須要件

- ・シリアルから受け取った 250bit を 2 進、10 進にそれぞれ変換して 2 回分の受信内容を表示
- ・各項目 (MT や Dc など) の値をログに表示する
- ・履歴を保存する機能 (最低 1 0 件)
- ・特定の bit を抽出する事を「ビット演算」というらしい(あれ？記憶の片隅に・・・)
- ・送られてくる 250bit は 16 進数なので、2 進数や 10 進数で書かれた DB を参照する場合はそれぞれ相互に変換をしないといけない
- ・仕様上、4 秒に 1 回送出されるが重複する事が多いので (CRC を除いた) 解析後のデータをキャッシュして差分比較する必要がある

シリアルを受信したい  
pyserial という物が一般的

250bit を分割？抽出？したい  
・まず、↓の構造を理解しないと始まらない



↑は緊急地震速報用のフォーマット(他もほとんど変わらないが 18 ビット以降の It や Li (L1L2) の使う値がそれぞれ変わる)



つまずき

Python リファレンスを見たら、かえって頭が痛くなった  
 "Python チュートリアル - Python 3.11.3 ドキュメント"を見たらちょっとわかった

例

### 配信例①

気象庁発表：2019/02/21 21:22(JST)の緊急地震速報

配信された250bit の災危通報データ

【9AAC89558B0003240000AB160F3A2499B40000000000002000000010C93712C】

発表時刻：2019年2月21日21時22分

震央地名：胆振地方中東部

深さ(km) : 30    マグニチュード : 5.8

震度(下限):震度5弱 震度(上限):震度5弱

府県予報区及び地方予報区：北海道道央、北海道道南、北海道道東、北海道

配信 21:22:58~21:24:34 回数24回 (2020年1月末時点では5分間配信される)

災害・危機管理通報サービスアプリケーションノート(2020年10月 第1.0版)

## P51 データ配信例（緊急地震速報）

変換は

[illegible]

↑のように 250bit になるはず

これを Python プログラム上で 250bit の bin にするには????

とはいえまったくわからずという事とで結局ググった。

でも基礎すらできておらず全くわからなかったので、結局 3000 円くらいの分厚い鈍器を未来の借金マイナポイントで買ってしまった。

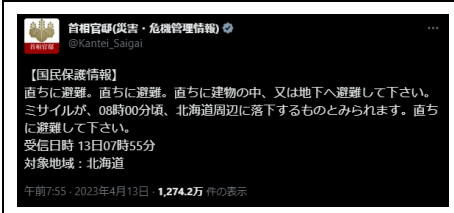
とりあえずデコードはできた！

## 表現方法

データから配列→テキストへデコードできたので、これをどのように出力するかという話

少し脱線

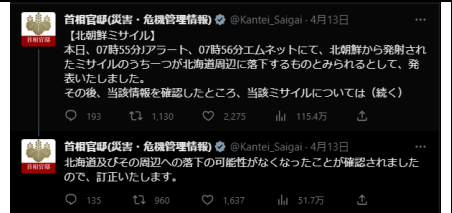
2023 年 4 月 13 日に「北朝鮮からミサイルが発射され、北海道に落下する恐れがある」というなかなかクリティカルな J アラートが発出された(※後に「落下の恐れはなくなった」と訂正された)



首相官邸(災害・危機管理情報) @Kantei\_Saigai

【国民保護情報】  
直ちに避難。直ちに避難。直ちに建物の中、又は地下へ避難して下さい。  
ミサイルが、08時00分頃、北海道周辺に落下するものとみられます。直ちに避難して下さい。  
受信日時 13日07時55分  
対象地域：北海道  
午前7:55 · 2023年4月13日 · 1,274.2万 件の表示

[https://twitter.com/Kantei\\_Saigai/status/1646285895165960192?s=20](https://twitter.com/Kantei_Saigai/status/1646285895165960192?s=20)


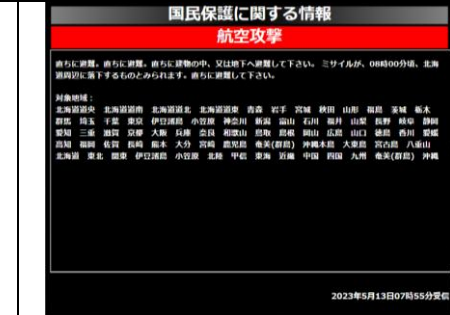
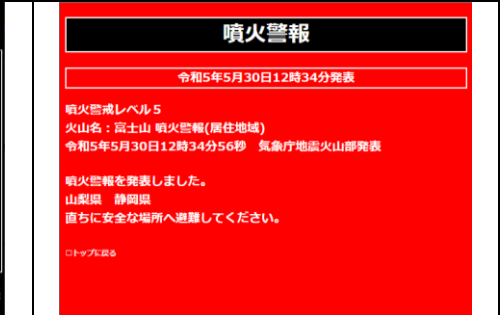


首相官邸(災害・危機管理情報) @Kantei\_Saigai · 4月13日

【北朝鮮ミサイル】  
本日、07時55分アラート、07時56分エムネットにて、北朝鮮から発射されたミサイルのうち一つが北海道周辺に落下するものとみられるとして、発表いたしました。  
その後、当該情報を確認したところ、当該ミサイルについては(続く)  
首相官邸(災害・危機管理情報) @Kantei\_Saigai · 4月13日  
北海道及びその周辺への落下の可能性がなくなったことが確認されましたので、訂正いたします。

[https://twitter.com/Kantei\\_Saigai/status/1646303057079332866?s=20](https://twitter.com/Kantei_Saigai/status/1646303057079332866?s=20)

ということで、早速こんなのを作ってみましたよ

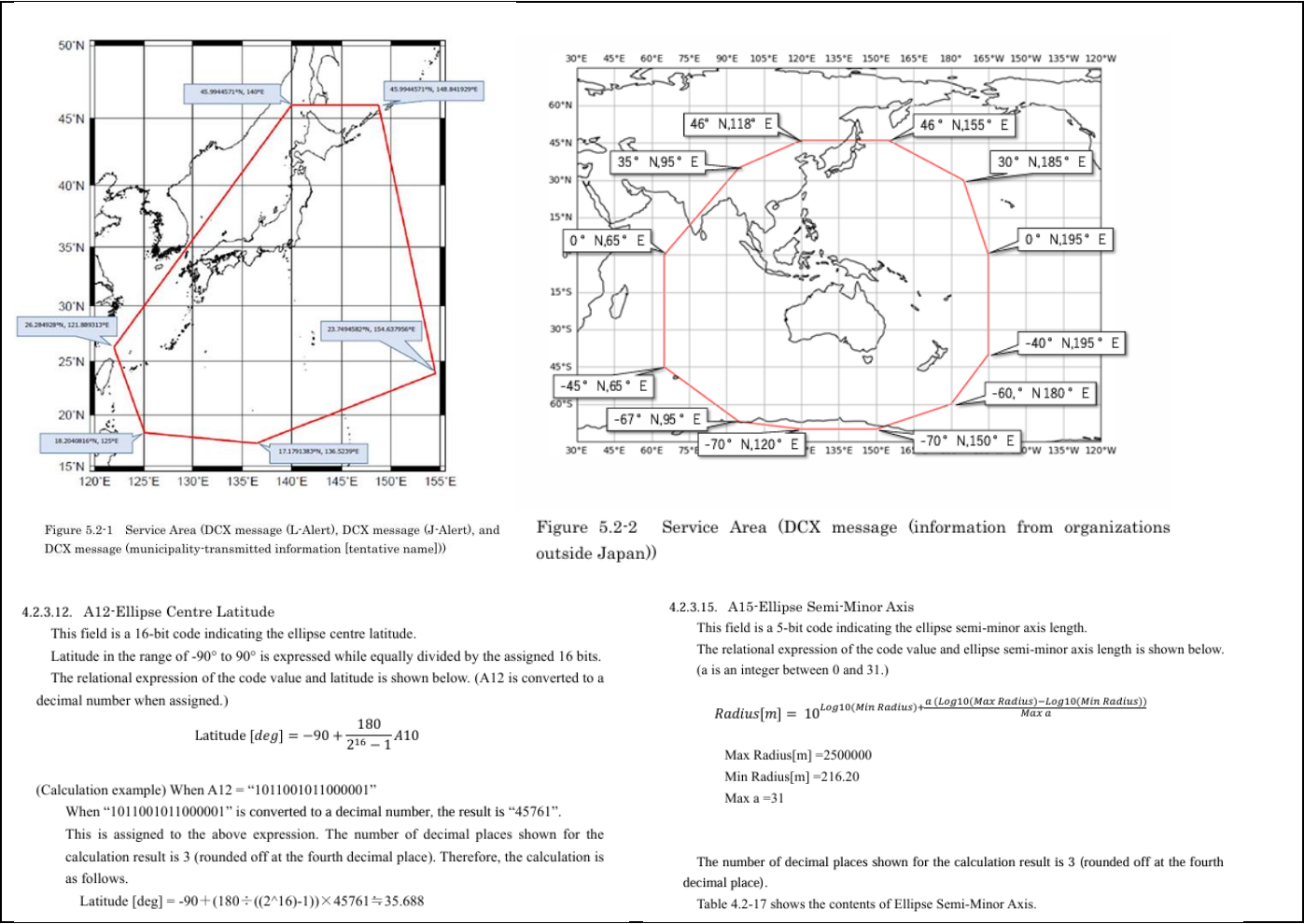
		
メイン画面 (試作品)	J アラートならぬ M アラート (おそらく、消防庁の要件に準拠した画面)	噴火警報 (内容はもちろん適当)

※これらは実際に受信した物ではなくベタ書きのイメージ図で、データベース参照なんて高度な事はしてない

課題

- ・ 画像は重いし、環境によって大きさが変わる (CSS 理解した)  
→SVG にすれば解決しそう？
- ・ 表とかが画面の解像度で崩れるので環境依存が非常に大きい
- ・ 色覚異常の方に優しい色使いを心掛けたい (NERV の石森さんはこれも意識してアプリを作っている)

この後、  
HTML ベタ書きの見た目だけ作ってみて満足したので  
本公開版のソースコードにこんな表示機能はありません！  
(みんなこの程度の画面ならホイと簡単に連携まで作れるでしょ・・・？)



なんじゃこりゃ・・・ Latitude [deg] = ってなに・・・

参考文献（QZSS 災害危機資料ソースリスト.txt より）

<p>○衛星仕様関連</p> <p>災害・危機管理通報サービス アプリケーションノート(2020年10月 第1.0版) <a href="https://qzss.go.jp/en/technical/download/do-report/users-manual_ver1.pdf">https://qzss.go.jp/en/technical/download/do-report/users-manual_ver1.pdf</a></p> <p>パフォーマンス スタンダード Quasi-Zenith Satellite System Performance Standard(PS-QZSS-003) (March 17, 2022) <a href="https://qzss.go.jp/en/technical/download/pdf/ps-is-qzss/ps-qzss-003.pdf">https://qzss.go.jp/en/technical/download/pdf/ps-is-qzss/ps-qzss-003.pdf</a></p> <p>ユーザインタフェース仕様書 IS-QZSS-DCR-010 (Jan. 24, 2022/ 4.5MB) <a href="https://qzss.go.jp/en/technical/download/pdf/ps-is-qzss/is-qzss-dcr-010.pdf">https://qzss.go.jp/en/technical/download/pdf/ps-is-qzss/is-qzss-dcr-010.pdf</a></p> <p>システム概要 <a href="https://qzss.go.jp/technical/system/dcr.html">https://qzss.go.jp/technical/system/dcr.html</a></p> <p>「準天頂衛星システム ユーザインタフェース仕様書（IS-QZSS）について <a href="https://qzss.go.jp/technical/download/isos7j0000000bhx-att/03_is-qzss-lis_com_20141021.pdf">https://qzss.go.jp/technical/download/isos7j0000000bhx-att/03_is-qzss-lis_com_20141021.pdf</a></p> <p>○ハードウェア関連</p> <p>Spresense <a href="https://developer.sony.com/ja/develop/spresense/">https://developer.sony.com/ja/develop/spresense/</a></p> <p>ATOM GPS Kit (M8030-KT)でお手軽Bluetooth GPS レシーバーを作った - 知的好奇心 for IoT <a href="https://intellectualcuriosity.hatenablog.com/entry/2020/07/01/041715">https://intellectualcuriosity.hatenablog.com/entry/2020/07/01/041715</a> ※端末設定方法の解説だけで M8030 が防災通報に対応しているわけではない</p>	<p>○アプリケーション関係</p> <p>★Python Python 3.11.3 ドキュメント <a href="https://docs.python.org/ja/3/">https://docs.python.org/ja/3/</a></p> <p>Python チュートリアル - Python 3.11.3 ドキュメント <a href="https://docs.python.org/ja/3/tutorial/index.html">https://docs.python.org/ja/3/tutorial/index.html</a></p> <p>Python Documentation contents - Python 3.11.3 ドキュメント <a href="https://docs.python.org/ja/3/contents.html">https://docs.python.org/ja/3/contents.html</a></p> <p>Python 言語リファレンス - Python 3.11.3 ドキュメント <a href="https://docs.python.org/ja/3/reference/index.html">https://docs.python.org/ja/3/reference/index.html</a></p> <p>Python 標準ライブラリ &amp;#8212; Python 3.11.3 ドキュメント <a href="https://docs.python.org/ja/3/library/index.html">https://docs.python.org/ja/3/library/index.html</a></p> <p>What's New In Python 3.2 - Python 3.11.3 ドキュメント (sqlite3) <a href="https://docs.python.org/ja/3/whatsnew/3.2.html#sqlite3">https://docs.python.org/ja/3/whatsnew/3.2.html#sqlite3</a></p> <p>http.server --- HTTP サーバ - Python 3.11.3 ドキュメント <a href="https://docs.python.org/ja/3/library/http.server.html?highlight=python%20web%20server#module-http.server">https://docs.python.org/ja/3/library/http.server.html?highlight=python%20web%20server#module-http.server</a></p> <p>pySerial - pySerial 3.4 documentation <a href="https://pyserial.readthedocs.io/en/latest/pyserial.html">https://pyserial.readthedocs.io/en/latest/pyserial.html</a></p> <p>ゼロからは始めるPython(71) Excelで読めないSQLiteデータをPythonで読んでExcelに差し込みたい   TECH+ (テックプラス) <a href="https://news.mynavi.jp/techplus/article/zeropython-71/">https://news.mynavi.jp/techplus/article/zeropython-71/</a></p> <p>Pythonで2進数、8進数、16進数の数値・文字列を相互に変換   note.nkmk.me <a href="https://note.nkmk.me/python-bin-oct-hex-int-format/">https://note.nkmk.me/python-bin-oct-hex-int-format/</a></p> <p>Pythonで浮動小数点数floatと16進数表現の文字列を相互に変換   note.nkmk.me <a href="https://note.nkmk.me/python-float-hex/">https://note.nkmk.me/python-float-hex/</a></p> <p>Python, format で書式変換 (0 埋め、指数表記、16 進数など)   note.nkmk.me <a href="https://note.nkmk.me/python-format-zero-hex/">https://note.nkmk.me/python-format-zero-hex/</a></p> <p>Pythonで文字列・数値をゼロ埋め（ゼロパディング）   note.nkmk.me <a href="https://note.nkmk.me/python-zero-padding/">https://note.nkmk.me/python-zero-padding/</a></p>
<p>○音源関係</p> <p>AITalk® micro   製品   音声合成ソフト、読み上げ、人工・電子音声の「株式会社 エーアイ (AI)」 <a href="https://www.ai-j.jp/products/micro/">https://www.ai-j.jp/products/micro/</a></p> <p>ReadSpeaker とは - 【公式】ReadSpeaker   AI 音声合成・音声読み上げ (WEB・テキスト) ソフトのリードスピーカー <a href="https://readspeaker.jp/feature/">https://readspeaker.jp/feature/</a></p> <p>AquesTalk - 組み込み用 規則音声合成エンジン <a href="https://www.ai-j.jp/consumer/">https://www.ai-j.jp/consumer/</a></p> <p>緊急地震速報利用者協議会   緊急地震速報の受信時の報知音の音源提供 <a href="http://www.eewrk.org/eewrk_hochi-on/eewrk_hochi-on.html">http://www.eewrk.org/eewrk_hochi-on/eewrk_hochi-on.html</a></p> <p>緊急地震速報 チャイム音の利用について <a href="https://www.nhk.or.jp/nijiriyu/howtochaim.html">https://www.nhk.or.jp/nijiriyu/howtochaim.html</a></p> <p>REIC の緊急地震速報サイン音をご活用ください。【REIC リアルタイム地震・防災情報利用協議会】 <a href="http://www.real-time.jp/?page_id=465">http://www.real-time.jp/?page_id=465</a></p> <p>REIC サイン音利用規約書 <a href="http://www.real-time.jp/wp/wp-content/uploads/2020/04/200401_2017signagreement.pdf">http://www.real-time.jp/wp/wp-content/uploads/2020/04/200401_2017signagreement.pdf</a></p> <p>Android のオープンソース (AOSP) を見る - Qiita <a href="https://qiita.com/okey01/items/89e575fc8e8ebab0e2bc">https://qiita.com/okey01/items/89e575fc8e8ebab0e2bc</a></p> <p>Cross Reference: /packages/apps/CellBroadcastReceiver/res/raw <a href="https://search.siprof.org/android-8.0.0_r1.0/xref/packages/apps/CellBroadcastReceiver/res/raw/">https://search.siprof.org/android-8.0.0_r1.0/xref/packages/apps/CellBroadcastReceiver/res/raw/</a></p> <p>Developer Collaboration Project - Android ソースコード検索サービス <a href="https://sites.google.com/site/devcollaboration/codesearch?pli=1">https://sites.google.com/site/devcollaboration/codesearch?pli=1</a></p>	<p>○その他関連(J-Alert など)</p> <p>Jアラート   株式会社 理経 <a href="https://www.rieki.co.jp/j-alert/">https://www.rieki.co.jp/j-alert/</a></p> <p>J-ALERT 専用小型受信機 JARS-2000 センチュリーシステムズ <a href="http://www.centurysys.co.jp/products/jalert/jars2000.html">http://www.centurysys.co.jp/products/jalert/jars2000.html</a></p> <p>全国瞬時警報システム (Jアラート) の概要   消防庁の組織および所掌業務   総務省消防庁 <a href="https://www.fdma.go.jp/about/organization/post-18.html">https://www.fdma.go.jp/about/organization/post-18.html</a></p> <p>Jアラートと連携する情報伝達手段の多重化について   総務省消防庁 <a href="https://www.fdma.go.jp/mission/protection/item/protection001_13_J-ALERT_300112_2.pdf">https://www.fdma.go.jp/mission/protection/item/protection001_13_J-ALERT_300112_2.pdf</a></p> <p>[学生向け] レポート・論文に使える参考文献の書き方 - USACO <a href="https://www.usaco.co.jp/endnote/reference_report.html">https://www.usaco.co.jp/endnote/reference_report.html</a></p> <p>参考文献の書き方 - 新潟大学 <a href="https://www.lib.niigata-u.ac.jp/learning_support/doc/20210709-3.pdf">https://www.lib.niigata-u.ac.jp/learning_support/doc/20210709-3.pdf</a></p> <p>書籍 松浦 健一郎/司 ゆき, わかるPython. 決定版. SBクリエイティブ, 2022, 383p. ISBN:978-4-7973-9544-0</p>