更新日：2019年12月11日
情報所有者：(株)ハレックス

2019年第3回WXBCセミナー

APIセミナー

ビジネスソリューション事業部　システム部
土川　正彦

“生活達人”応援企業
HALEX
HAPPY LIFE EXPERT

株式会社 ハレックス

ハレックス
ハンズオンセミナーの振り返り

ハンズオンセミナーの内容

プロダクト	第1回（7/12）	第2回（9/25）	第3回（12/11）
週間予報API（jpwx）	●		
世界版 API （jpwx）			●
降水ナウキャストAPI（６時間API）			●
７２時間API			
竜巻API			
気象特別警報・警報・注意報API			
流域雨量指数・表面雨量指数および			
洪水警報の危険度API			
過去データ(2012年1月～のデータ)		●	

本日ご参加の皆様に限り、ご連絡頂ければ、第1回、第2回用のお試しAPIキー払い出します

資料：APIセミナー（第1回） https://www.wxbc.jp/wp-content/uploads/2019/07/seminar_190712_05.zip
資料：APIセミナー（第2回） https://www.wxbc.jp/wp-content/uploads/2019/09/seminar_190925_05.zip

APIとは

APIとは、
Application Programming Interfaceの略称

サービス提供者が、サービスを利用してもらうために提供するインタフェースを指す。アプリケーション開発者（API利用者）が**APIを利用すれば、同じ機能を持ったサービスを開発する必要がない**ため、開発効率の向上や開発費用の低減が期待できる。

今回ご提供するAPIは「Web API」

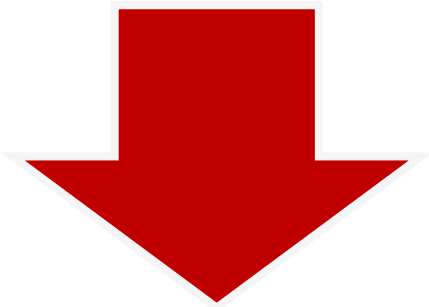
Web APIとは、API提供者とAPI利用者とのやりとりをHTTP／HTTPSベースで実現するAPIです。Web APIはHTTP／HTTPSベースのAPIであるため、Webサービスにも簡単に利用できるなど、汎用性が高い。

弊社のAPIは
Rest API設計

【参考】APIリクエスト
http://demo.halex.co.jp/wpast/hpd?sid=analysis-p-service&rem=all&lat=34.702485&lon=135.495951&gran=5&from=20170801&to=20170831&key=WX2000-1xRr4fGm

本日のコンセプト

APIを使って気象情報が
簡単に利用できることを体験してもらう



①サンプルプログラムの説明
②自分でプログラミングして気象情報を操作

【持ち物】

●WiFi接続可能なノートPC

※WiFi接続できないとAPIで気象情報が取得できない・・・

※使用ブラウザ：GoogleChrome･Microsoft Edge
　Internet Explorerは一部動作しない・・・

※文字コードは「UTF-8」　設定によっては文字化けします

●事前配布したライセンスキー

事前配布したライセンス
キーは使いません

※本日はセミナー用「WX3120-Gn3Zz9FH」を利用します

※セミナー用につき、**二次配布や商用利用NG**

【注意】本日中に
利用出来なくなります

【資料ダウンロード】

https://www.wxbc.jp/wp-content/uploads/2019/12/seminar_191211_05.zip

【その他】

- JavaScriptを利用してAPIで取得した気象情報を操作して頂きます。
- 時間の関係から、気象に関わるご質問は別途個別に打合せをさせて下さい。

※名刺を頂ければ、後日ご連絡させて頂きます

本日ご提供するAPI
HalexDream!（特許取得）

世界版API
雨雲画像API

ハレックスの主な気象系サービス

ハレックスでは様々な気象サービスを提供しています。

① **過去データ（HalexMemory!）** ← 気象との因果関係（傾向）を確認

② **ピンポイントでの気象予測（HalexDream!）** ← 気象データを利用した未来予測

③ **気象用Webサイト＋気象条件によるアラートメール（HalexSmile!）** ← すぐに気象を使いたい人向け

■ハレックスのサービス

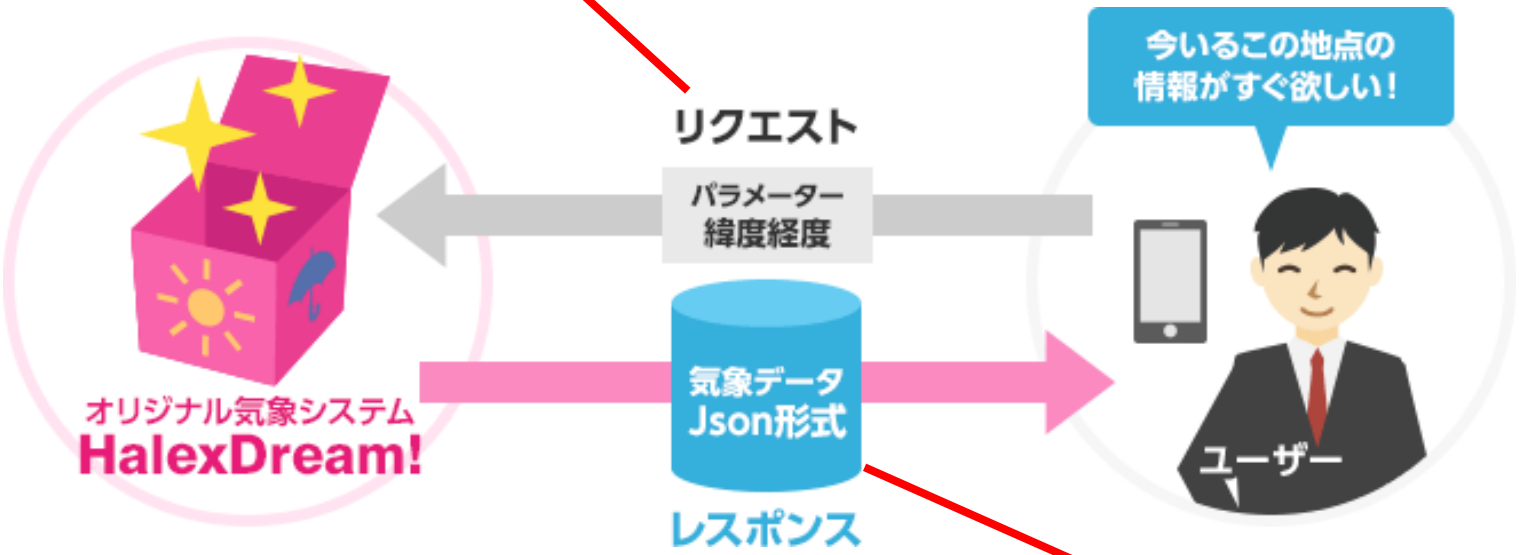
サービス		概要
気象	HalexDream! ※特許取得	気象庁から大量かつ高頻度で発行されるデータをオンラインで高速リアルタイム処理することで、利用される方々によって扱い易いデータに編集し、緯度経度によって取り扱えるように1kmメッシュでデータを管理・処理するシステム なお、編集したデータは次の３つの手段によって提供 ①DreamAPI 緯度経度でパラメータ指定されジャストポイントの時系列データをレスポンスするサービス ②DreamFiles 緯度経度で指定されたジャストポイントの時系列データをファイルに格納して提供するサ ③DreamAll 日本全国分の面展開したデータをファイルに格納して提供するサービス
	HalexMemory!	過去の気象データをご利用頂くサービス 日本全国1kmメッシュ×1時間間隔で管理したデータをCSVファイルによるオフライン提供、 スを実施 ※天気、温度、湿度、1時間降水量、風向風速・・・
	HalexSmile!	DreamAPIをフル活用した気象データ閲覧Webサイト 地図上で指定した任意地点のデータ閲覧が可能で常時閲覧する地点は固定地点として登録が可能 また条件設定することでアラートメール通知も可能

今日はコレ！！

ハレックスのAPIの特徴

HalexDream!APIとは

https://fspweb03.halex.co.jp/wimage/hpd?sid=jpwx-world&rem=wx&lat=[緯度]&lon=[経度]&key=[アクセスキー]




```
{
 "iniTime": "2019/07/11 16:00:00.000 JST",
 "param": {
          :
 },
 "sessionId": "F6DBF666691641A2E62C8BD0D5C9D014",
 "systemTime": "2019/07/11 15:34:55.813 JST",
 "weatherData": {
  "a": {
   "dy": {"mgtn": "0","mgtx": "-1","pb24": "60","pb6_3": "50","pb6_4": "60","tn": "19","tx": "23","wd": "5","ws": "2","wt": "300","wtt": "雨"},
   "h1-00": {"pr": "0.0","te": "19","wd": "2","ws": "2","wt": "100"},
   "h1-01": {"pr": "0.0","te": "19","wd": "2","ws": "2","wt": "100"},
   "h1-02": {"pr": "0.0","te": "19","wd": "2","ws": "2","wt": "100"},
     :
```

【URL】

●世界版API

https://fspweb03.halex.co.jp/wimage/hpd?sid=jpwx-world&lat=[緯度]&lon=[経度]&key=[アクセスキー]

●雨雲画像API（ファイル名取得）

https://fspweb01.halex.co.jp/wimage/hpd?sid=wimage-service&rem=all&lat=[緯度]&lon=[経度]&key=[アクセスキー]

●雨雲画像API（ファイル取得）

https://fspweb01.halex.co.jp/wimage/img?sid=wimage-image-service&key=[アクセスキー]&fileName=[ファイル名]

※[]内の詳細は仕様書を参照

JSONとはJavaScript Object Notationの略で、XMLなどと同様のテキストベースのデータフォーマットです。
その名前の由来の通りJSONはJavaScriptのオブジェクト表記構文のサブセットとなっており、XMLと比べると簡潔に構造化されたデータを記述することができるため、記述が容易で人間が理解しやすいデータフォーマットです。

※今回のセミナーはJSONPを利用して記述しています

【提供内容】

アクセスキー	取得範囲	取得要素	提供期間
本日利用キー WX3120-Gn3Zz9FH	制限なし	制限なし	セミナー時
各自配布したキー			**2020.1.7まで** ※4週間

ご提供資料の説明

編集用プログラムの動作を確認するには、メニューの「カスタマイズページを開く」をクリック

- img
- DreamAPI_6時間data簡易仕様書.pdf
- DreamAPI_jpwx(世界版)簡易仕様書.pdf
- index.css
- index.html
- Sample3-1.html
- Sample3-2.html
- Sample4-1.html
- Sample4-2.html
- Sample4-3.html
- Sample4-4.html
- Sample5-1.html
- Sample5-2.html
- カスタマイズ用Sample4-1.html
- カスタマイズ用Sample4-2.html
- カスタマイズ用Sample4-3.html
- カスタマイズ用Sample4-4.html
- カスタマイズ用Sample5-2.html

API仕様書　※重要

セミナーのメニュー画面

メニュー画面

サンプルプログラム

編集用プログラム

セミナーのメニュー画面を開きましょう！！

- img
- DreamAPI_6時間data簡易仕様書.pdf
- DreamAPI_jpwx(世界版)簡易仕様書.pdf
- index.css
- index.html
- Sample3-1.html
- Sample3-2.html
- Sample4-1.html
- Sample4-2.html
- Sample4-3.html
- Sample4-4.html
- Sample5-1.html
- Sample5-2.html
- カスタマイズ用Sample4-1.html
- カスタマイズ用Sample4-2.html
- カスタマイズ用Sample4-3.html
- カスタマイズ用Sample4-4.html
- カスタマイズ用Sample5-2.html

API仕様書　※重要

メニュー画面

セミナーのメニュー画面

コレを起動
ブラウザはGoogleChromeまたは
MicrosoftEdge！！

サンプルプログラム

【ヒント】GoogleChromeまたはMicrosoftEdge以外のブラウザで開いてしまう場合、
GoogleChromeまたはMicrosoftEdgeを起動後、「index.html」をドラッグ＆ドロップ！！

編集用プログラム

ハレックスAPIセミナー（2019年12月11日）

① HalexDream!について

https://www.halex.co.jp/service/api.html

② JSONについて

JSONとはJavaScript Object Notationの略で、軽量なデータ記述言語の1つです。
JSONはJavaScript専用のデータ形式では決してなく、様々なソフトウェアやプログラミング言語間におけるデータの受け渡しに使えるよう設計されています。
今回はWebブラウザを使用して情報を取得するためJSONPを使用します。
JSONPとは引数にJSONをとったコールバック関数のことで、主にWeb画面内のJavaScriptでJSONを使いたい場合に使用されます。

③ ハレックスのAPIを使ってみよう（世界版jpwx）

仕様書：DreamAPI_jpwx(世界版)簡易仕様書.pdf
リクエストURL：https://fspweb03.halex.co.jp/wimage/hpd?sid=jpwx-p-world&lat=[緯度]&lon=[経度]&key=[アクセスキー]
[アクセスキー]：別途案内されているアクセスキーを設定してください。
[緯度][経度]：取得したい地点の緯度経度を設定してください。
（http,httpsでのリクエストが可能）

ハレックスAPIセミナー（2019年12月11日）

弊社のサイト　API説明　※割愛

① HalexDream!について

https://www.halex.co.jp/service/api.html

APIレスポンスであるJSONについて
※割愛

② JSONについて

JSONとはJavaScript Object Notationの略で、軽量なデータ記述言語の1つです。
JSONはJavaScript専用のデータ形式では決してなく、様々なソフトウェアやプログラミング言語間におけるデータの受け渡しに使えるよう設計されています。
今回はWebブラウザを使用して情報を取得するためJSONPを使用します。
JSONPとは引数にJSONをとったコールバック関数のことで、主にWeb画面内のJavaScriptでJSONを使いたい場合に使用されます。

③ ハレックスのAPIを使ってみよう（世界版jpwx）

仕様書：DreamAPI_jpwx(世界版)簡易仕様書.pdf
リクエストURL：https://fspweb03.halex.co.jp/wimage/hpd?sid=jpwx-p-world&lat=[緯度]&lon=[経度]&key=[アクセスキー]
[アクセスキー]：別途案内されているアクセスキーを設定してください。
[緯度][経度]：取得したい地点の緯度経度を設定してください。
（http,httpsでのリクエストが可能）

API仕様書の説明
【世界版API】

ハレックスAPIセミナー（2019年12月11日）

① HalexDream!について

https://www.halex.co.jp/service/api.html

② JSONについて

JSONとはJavaScript Object Notationの略で、軽量なデータ記述言語の1つです。
JSONはJavaScript専用のデータ形式では決してなく、様々なソフトウェアやプログラミング言語間におけるデータの受け渡しに使えるよう設計されています。
今回はWebブラウザを使用して情報を取得するためJSONPを使用します。
JSONPとは引数にJSONをとったコールバック関数のことで、主にWeb画面内のJavaScriptでJSONを使いたい場合に使用されます。

③ ハレックスのAPIを使ってみよう（世界版jpwx）

仕様書：DreamAPI_jpwx(世界版)簡易仕様書.pdf
リクエストURL：https://[redacted]n=[経度]&key=[アクセスキー]
[アクセスキー]：別途案内されているアクセスキーを設定し[redacted]
[緯度][経度]：取得したい地点の緯度経度を設定してください。
（http,httpsでのリクエストが可能）

クリック

DreamAPI

HALEX DreamAPI　jpwx(世界版)　仕様書

株式会社　ハレックス
2019/11/25　ver1.0

1．概要

・1kmメッシュ（3次メッシュ）で管理された気象情報を、緯度・経度を指定することにより、該当メッシュの気象情報を時系列に、JSON形式または、JSONP形式でレスポンスします。
・1時間単位、3時間単位および（前日～2日先）、1日単位（前日～7日先）の気象情報をレスポンスします。
・日本国内および国外で異なる気象情報を取得します。

JMAデータ	北海道、本州、四国、九州とその周辺の有人島、沖縄、伊豆諸島および小笠原諸島の有人島
TWCデータ	日本国内を除く世界各地

・APIは常に最新の情報をレスポンスするので、更新タイミングを気にすることなく最新の気象情報を取得することができます。
※データは1時間毎に更新します。

※APIは、24時間365日アクセス可能です。定期メンテナンスはありませんが、元データの遅延により情報の更新が遅れる可能性があります。

リクエスト方法はP14参照

2．レスポンス

【レスポンスイメージ】

{"iniTime":"2019-11-14T12:00:00+0100", "param":
{"key":"", "lat":"41.384855", "lon":"2.172164", "sid":"xxxx-xxxx"},
"sessionId":"xxxxxxxx", "systemTime":"2019/11/14 19:38:56.271 JST", "xxxxx":
{"a":
{"dy":
{"mgtn":"-", "mgtx":"-", "pb24":"60", "pb6_1":"-", "pb6_2":"-", "pb6_3":"70", "pb6_4":"10", "tn":"7", "tx":"16", "wd":"7", "ws":"6", "wt":"103", "wtt":"晴時々雨"},
"h1-00":
{"hu":"93", "pr":"-", "te":"8", "wd":"8", "ws":"2", "wt":"100"},
"h1-01":
{"hu":"87", "pr":"-", "te":"9", "wd":"8", "ws":"3", "wt":"100"},
(省略)
"h1-22":
{"hu":"63", "pb":"0", "pr":"0.0", "te":"8", "wd":"8", "ws":"3", "wt":"100"},
"h1-23":
{"hu":"68", "pb":"0", "pr":"0.0", "te":"7", "wd":"8", "ws":"2", "wt":"100"},
"h3-00":
{"hu":"93", "pr":"-", "te":"8", "wd":"8", "ws":"2", "wt":"100"},
"h3-03":
{"hu":"81", "pr":"-", "te":"8", "wd":"1", "ws":"2", "wt":"100"},
"h3-06":
{"hu":"77", "pr":"-", "te":"16", "wd":"6", "ws":"9", "wt":"300"},
"h3-09":
{"hu":"87", "pr":"-", "te":"10", "wd":"8", "ws":"4", "wt":"100"},
"h3-12":
{"hu":"75", "pb":"70", "pr":"0.5", "te":"13", "wd":"7", "ws":"7", "wt":"300"},

←ヘッダ領域

日まとめの情報　1時間毎の情報　3時間毎の　当日の情報

←データ領域

レスポンスイメージ

APIの仕様書

【変数説明】

No	日時変数	要素変数	説明
1	iniTime		予測開始日時　日本時刻
2	a		当日の情報
	b		1日先の情報
	c		2日先の情報
	d		3日先の情報
	e		4日先の情報
	f		5日先の情報
	g		6日先の情報
	h		7日先の情報
	y		1日前の情報
3	h1		1時間毎の情報
3-1		-hh	該当時刻　00～23(時)
3-2		wt	天気　3桁のコード(※1)
3-3		te	気温　1℃刻み
3-4		hu	湿度　1%刻み
3-5		ws	風速　1m/s刻み
3-6		wd	風向　8方位のコード(※2)
3-7		pr	降水量　0.0, 0.5, 1, 以降1mm刻み
3-8		pb	降水確率　10%刻み　0～100(%)
4	h3		3時間毎の情報
4-1		-hh	該当時刻　00,03,06,09,12,15,18,21(時)
4-2		wt	天気　3桁のコード(※1)
4-3		te	気温　1℃刻み
4-4		hu	湿度　1%刻み
4-5		ws	風速　1m/s刻み
4-6		wd	風向　8方位のコード(※2)
4-7		pr	降水量　0.0, 0.5, 1, 以降1mm刻み
4-8		pb	降水確率　10%刻み　0～100(%)
5	dy		日まとめの情報
5-1		wt	天気　3桁のコード(※1)
5-2		wtt	天気文言　（例）晴れ時々くもり
5-3		tn	最低気温　1℃刻み
5-4		tx	最高気温　1℃刻み
5-5		mgtn	最低気温前日差　1℃刻み
5-6		mgtx	最高気温前日差　1℃刻み
5-7		pb6_1～pb6_4	6時間降水確率　10%刻み　0～100(%)
5-8		pb24	24時間降水確率　10%刻み　0～100(%)
5-9		ws	風速　1m/s刻み
5-10		wd	風向　8方位のコード(※2)

←データ部の変数の説明

←時間情報

←予測の時間

←1時間毎の予測
h1-XX（XX＝時間）

←3時間毎の予測
h3-XX（XX＝時間）

(※1)主な天気コード

天気コード	予報内容	天気コード	予報内容	天気コード	予報内容	天気コード	予報内容	天気コード	予報内容
100	晴	200	曇	300	雨	400	雪	500	みぞれ

←コード変換用

(※2)風向コード

風向コード	風向	風向コード	風向	風向コード	風向
0	静穏	3	東の風	6	南西の風
1	北の風	4	南東の風	7	西の風
2	北東の風	5	南の風	8	北西の風

【提供する情報】 ←データの提供内容

・1時間毎、3時間毎の情報

No	情報名	前日～当日実況		当日予測		1日先		2日先	
		国内	国外	国内	国外	国内	国外	国内	国外
1	天気	○	◇	○	◇	○	○	○	○
2	気温	○	◇	○	◇	○	○	○	○
3	湿度	-	◇	○	◇	○	○	○	○
4	風向	○	◇	○	◇	○	○	○	○
5	風速	○	◇	○	◇	○	○	○	○
6	降水量	○	◇	○	◇	○	○	○	○
7	降水確率	-	-	○	○	○	○	-	○

【注意】
国内と国外で仕様が異なる

・日まとめの情報

No	情報名	前日		当日		1日先		2日先		3~6日先		7日先	
		国内	国外	国内	国外	国内	国外	国内	国外	国内	国外	国内	国外
1	天気	○	-	○	○	○	○	○	○	○	○	◇	○
2	天気テキスト	○	-	○	○	○	○	○	○	○	○	◇	○
3	最低気温	○	-	○	○	○	○	○	○	○	○	◇	○
4	最高気温	○	-	○	○	○	○	○	○	○	○	◇	○
5	最低気温前日差	-	-	○	-	○	○	-	-	-	-	-	-
6	最高気温前日差	-	-	○	-	○	○	-	-	-	-	-	-
7	6時間降水確率	-	-	○	○	○	○	-	○	-	-	-	-
8	1日降水確率	-	-	○	○	○	○	○	○	○	○	◇	○
9	風向	○	-	○	○	○	○	○	○	○	○	◇	○
10	風速	○	-	○	○	○	○	○	○	○	○	◇	○

○：提供可能な要素・情報です。
◇：時刻帯、地点、情報提供元の仕様等の事情によって未格納となります。

※iniTimeより前の時刻にはデータは格納されません。

API仕様書の説明
【雨雲画像API】

⑤ ハレックスのAPIを使ってみよう（雨雲画像）

仕様書：DreamAPI_6時間data簡易仕様書.pdf

クリック

リクエストURL：https://fspweb01.halex.co.jp/wimage/hpt[illegible].key=[アクセスキー]&lat=[緯度]&lon=[経度]

[アクセスキー]：別途案内されているアクセスキーを設定し

[緯度][経度]：取得したい地点の緯度経度を設定してください。

（http,httpsでのリクエストが可能）

⑤-1 DreamAPIを使って雨雲画像を取得しよう

JavaScriptからDreamAPIを使って、最新の雨雲画像を地図上に表示します。

ajaxを使って、JSONPで画像を取得します。

サンプルページはこちら

⑤-2 6時間先の雨雲画像を取得しよう

DreamAPIは1時間毎に6時間先までの雨雲画像を取得することが可能です。

6時間先の雨雲画像を取得し、地図上に表示してください。

カスタマイズページを開く

回答例はこちら

⑥ 注意事項

⑥-1 ライセンスに関して

今回配布したAPIキーはセミナー用となります。

二次配布や商用利用は行わないでください。

DreamAPI

HALEX DreamAPI 6時間data 仕様書

株式会社 ハレックス
2019/11/18 ver1.0

1．概要

・1kmメッシュ（3次メッシュ）で管理された気象情報を、緯度・経度を指定することにより、該当メッシュの気象情報を時系列に、JSON形式または、JSONP形式でレスポンスします。
・また、降水量（強度）によってメッシュ毎に色分けした画像（png）をレスポンスします。
※メッシュ画像は正方格子、メルカトル図法の指定が可能です。
・メッシュ画像を地図上に重畳することにより、降水の状況を視覚的に時系列に確認することが可能です。
・APIは常に最新の情報をレスポンスするので、更新タイミングを気にすることなく最新の気象情報を取得することができます。

※APIは、24時間365日アクセス可能です。定期メンテナンスはありませんが、元データの遅延により情報の更新が遅れる可能性があります。

リクエスト方法はP14参照

2．レスポンス

【レスポンスイメージ】


```
{"info":
 {"201807101700":
  {"cloud":"81.4", "humidity":"75.5", "temperature":"28.2", "weatherForecast":"200", "windDirection":"310.5", "windSpeed":"2.0"},
 "201807101800":
  {"cloud":"81.1", "humidity":"80.6", "temperature":"27.4", "weatherForecast":"200", "windDirection":"304.6", "windSpeed":"1.6"},
 "201807101900":
  {"cloud":"87.0", "humidity":"85.3", "temperature":"26.6", "weatherForecast":"200", "windDirection":"305.6", "windSpeed":"1.4"},
 "201807102000":
  {"cloud":"81.0", "humidity":"87.1", "temperature":"25.8", "weatherForecast":"200", "windDirection":"294.3", "windSpeed":"1.2"},
 "201807102100":
  {"cloud":"69.2", "humidity":"88.2", "temperature":"25.4", "weatherForecast":"200", "windDirection":"285.5", "windSpeed":"1.1"
 "201807102200":
  {"cloud":"77.2", "humidity":"87.8", "temperature":"24.9", "weatherForecast":"200", "windDirection":"290.7", "windSpeed":"1.1"}
 },
"lx":240, "ly":120, "param":
 {"key":"********", "lat":"35", "lon":"139", "rem":"all", "sid":"wimage-service"},
"precipitation":[
 {"dtf":"201807101600-201807101700", "east":142.0, "fileName":"precipitation/shortme/201807101600-00060/0/0120-0080/480-480/000-000/0003-0003.png", "north":36.0, "south":32.0, "value":"0.0", "west":136.0},
 {"dtf":"201807101700-201807101800", "east":142.0, "fileName":"precipitation/shortme/201807101600-00120/0/0120-0080/480-480/000-000/0003-0003.png", "north":36.0, "south":32.0, "value":"0.0", "west":136.0},
 {"dtf":"201807101800-201807101900", "east":142.0, "fileName":"precipitation/shortme/201807101600-00180/0/0120-0080/480-480/000-000/0003-0003.png", "north":36.0, "south":32.0, "value":"0.0", "west":136.0},
 {"dtf":"201807101900-201807102000", "east":142.0, "fileName":"precipitation/shortme/201807101600-00240/0/0120-0080/480-480/000-000/0003-0003.png", "north":36.0, "south":32.0, "value":"0.0", "west":136.0},
 {"dtf":"201807102000-201807102100", "east":142.0, "fileName":"precipitation/shortme/201807101600-00300/0/0120-0080/480-480/000-000/0003-0003.png", "north":36.0, "south":32.0, "value":"0.0", "west":136.0},
 {"dtf":"201807102100-201807102200", "east":142.0, "fileName":"precipitation/shortme/201807101600-00360/0/0120-0080/480-480/000-000/0003-0003.png", "north":36.0, "south":32.0, "value":"0.0", "west":136.0}
],
"precipitation5min":[
 {"dtf":"201807101625-201807101630", "east":142.0, "fileName":"precipitation/radar/201807101630-00000/0/0120-0080/480-480/000-000/0003-0003.png", "north":36.0, "south":32.0, "value":"0.0", "west":136.0},
 {"dtf":"201807101630-201807101635", "east":142.0, "fileName":"precipitation/nowcast/201807101630-00005/0/0120-0080/480-480/000-000/0003-0003.png", "north":36.0, "south":32.0, "value":"0.0", "west":136.0},
 {"dtf":"201807101635-201807101640", "east":142.0, "fileName":"precipitation/nowcast/201807101630-00010/0/0120-0080/480-480/000-000/0003-0003.png", "north":36.0, "south":32.0, "value":"0.0", "west":136.0},
 {"dtf":"201807101640-201807101645", "east":142.0, "fileName":"precipitation/nowcast/201807101630-00015/0/0120-0080/480-480/000-000/0003-0003.png", "north":36.0, "south":32.0, "value":"0.0", "west":136.0},
 {"dtf":"201807101645-201807101650", "east":142.0, "fileName":"precipitation/nowcast/201807101630-00020/0/0120-0080/480-480/000-000/0003-0003.png", "north":36.0, "south":32.0, "value":"0.0", "west":136.0},
 {"dtf":"201807101650-201807101655", "east":142.0, "fileName":"precipitation/nowcast/201807101630-00025/0/0120-0080/480-480/000-000/0003-0003.png", "north":36.0, "south":32.0, "value":"0.0", "west":136.0},
 {"dtf":"201807101655-201807101700", "east":142.0, "fileName":"precipitation/nowcast/201807101630-00030/0/0120-0080/480-480/000-000/0003-0003.png", "north":36.0, "south":32.0, "value":"0.0", "west":136.0},
 {"dtf":"201807101700-201807101705", "east":142.0, "fileName":"precipitation/nowcast/201807101630-00035/0/0120-0080/480-480/000-000/0003-0003.png", "north":36.0, "south":32.0, "value":"0.0", "west":136.0},
 {"dtf":"201807101705-201807101710", "east":142.0, "fileName":"precipitation/nowcast/201807101630-00040/0/0120-0080/480-480/000-000/0003-0003.png", "north":36.0, "south":32.0, "value":"0.0", "west":136.0},
 {"dtf":"201807101710-201807101715", "east":142.0, "fileName":"precipitation/nowcast/201807101630-00045/0/0120-0080/480-480/000-000/0003-0003.png", "north":36.0, "south":32.0, "value":"0.0", "west":136.0},
 {"dtf":"201807101715-201807101720", "east":142.0, "fileName":"precipitation/nowcast/201807101630-00050/0/0120-0080/480-480/000-000/0003-0003.png", "north":36.0, "south":32.0, "value":"0.0", "west":136.0},
 {"dtf":"201807101720-201807101725", "east":142.0, "fileName":"precipitation/nowcast/201807101630-00055/0/0120-0080/480-480/000-000/0003-0003.png", "north":36.0, "south":32.0, "value":"0.0", "west":136.0},
 {"dtf":"201807101725-201807101730", "east":142.0, "fileName":"precipitation/nowcast/201807101630-00060/0/0120-0080/480-480/000-000/0003-0003.png", "north":36.0, "south":32.0, "value":"0.0", "west":136.0}
],
"sessionId":"734684D704143AD44557168AFCCC23E7", "sid":"wimage-service", "systemTime":"2018/07/10 16:39:44.030 JST"
```


6時間先までの気象予報

6時間先までの降水量予測

60分先までの降水強度予測

レスポンスイメージ

APIの仕様書

【変数説明】 ←データ部の変数の説明

No	情報名	変数名	説明	データ更新間隔
1	1時間間隔、6時間先までの気象予報	-	1時間先～6時間先まで6コマ	60分 ←気象情報
2	天気	weatherForecast	3桁のコード(※1)	
3	気温	temperature	単位：℃ 0.1刻み	
4	風向	windDirection	単位：度 0.1刻み	
5	風速	windSpeed	単位：m/s 0.1刻み	
6	湿度	humidity	単位：% 0.1刻み	
7	雲量	cloud	単位：% 0.1刻み	
8	1時間間隔、6時間先までの降水量予測	-	1時間先～6時間先まで6コマ	30分 ←1時間毎の画像情報
9	予報期間	dtf	予報期間：YYYYMMDDhhmm-YYYYMMDDhhmm	
10	降水量	value	単位：mm/h 0.0、0.5、1、以降1刻み	
11	メッシュ画像情報(※2)	FileName	画像のパス	
12		north、south、west、east	貼り付け位置	
13	5分間隔、60分先までの降水強度予測	-	実況＋5分先～60分先まで13コマ	5分
14	予報期間	dtf	予報期間：YYYYMMDDhhmm-YYYYMMDDhhmm	
15	降水強度	value	単位：mm/h 0.1刻み 降水強度：瞬間的な降水の強さを1時間当たりに換算したもの	←5分毎の画像情報
16	メッシュ画像情報(※2)	FileName	画像のパス	
17		north、south、west、east	貼り付け位置	

(※1)主な天気コードは以下です。

天気コード	予報内容	天気コード	予報内容	天気コード	予報内容	天気コード	予報内容	天気コード	予報内容
100	晴	200	曇	300	雨	400	雪	500	みぞれ

(※2)メッシュ画像は、レスポンスされたメッシュ画像情報を元に取得いただきます。

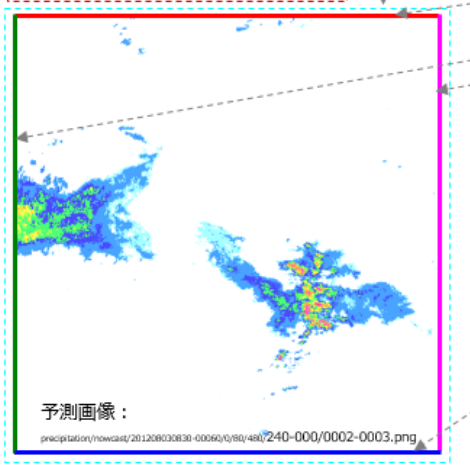
APIの仕様書

・地図へのメッシュ画像重畳イメージ

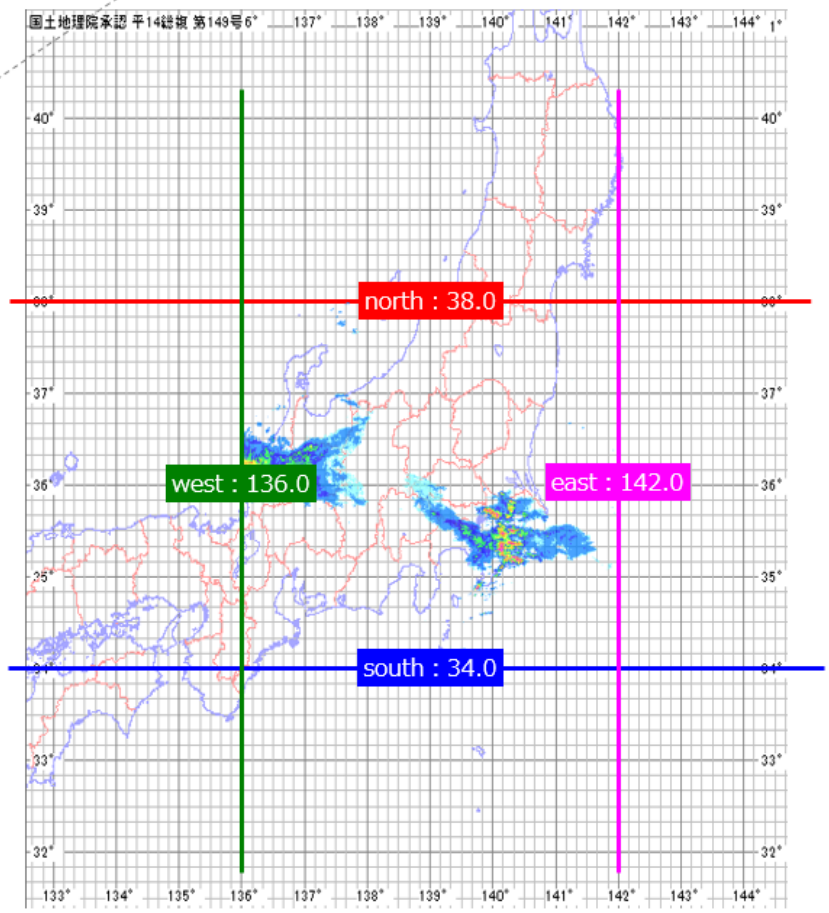
←APIレスポンスに地図へ重ねる緯度経度やファイルの格納位置が返ってくる

{"dtf":"201208030925-201208030930", "fileName":"precipitation/nowcast/201208030830-00060/0/80/480/240-000/0002-0003.png", "value":"0.0", "north":38.0, "south":34.0, "west":136.0, "east":142.0

2012年8月3日9時25分-9時30分の予測



地図に重畳すると



レスポンスデータ

HALEX HAPPY LIFE EXPERT

ハレックスAPIセミナー（2019年12月11日）

① HalexDream!について

https://www.halex.co.jp/service/api.html

② JSONについて

JSONとはJavaScript Object Notationの略で、軽量なデータ記述言語の1つです。
JSONはJavaScript専用のデータ形式では決してなく、様々なソフトウェアやプログラミング言語間におけるデータの受け渡しに使えるよう設計されています。
今回はWebブラウザを使用して情報を取得するためJSONPを使用します。
JSONPとは引数にJSONをとったコールバック関数のことで、主にWeb画面内のJavaScriptでJSONを使いたい場合に使用されます。

③ ハレックスのAPIを使ってみよう（世界版jpwx）

仕様書：DreamAPI_jpwx(世界版)簡易仕様書.pdf
リクエストURL：https://fspweb03.halex.co.jp/wimage/hpd?sid=jpwx-p-world&lat=[緯度]&lon=[経度]&key=[アクセスキー]
[アクセスキー]：別途案内されているアクセスキーを設定してください。
[緯度][経度]：取得したい地点の緯度経度を設定してください。
（http,httpsでのリクエストが可能）

③-1 DreamAPIを使ってJSONデータを取得しよう

JavaScriptからDreamAPIを使って、JSONを画面に表
ajaxを使って、JSONPでデータを取得します
サンプルページはこちら

クリック

（参考）主要都市の緯度経度

⑥-3 主要都市一覧

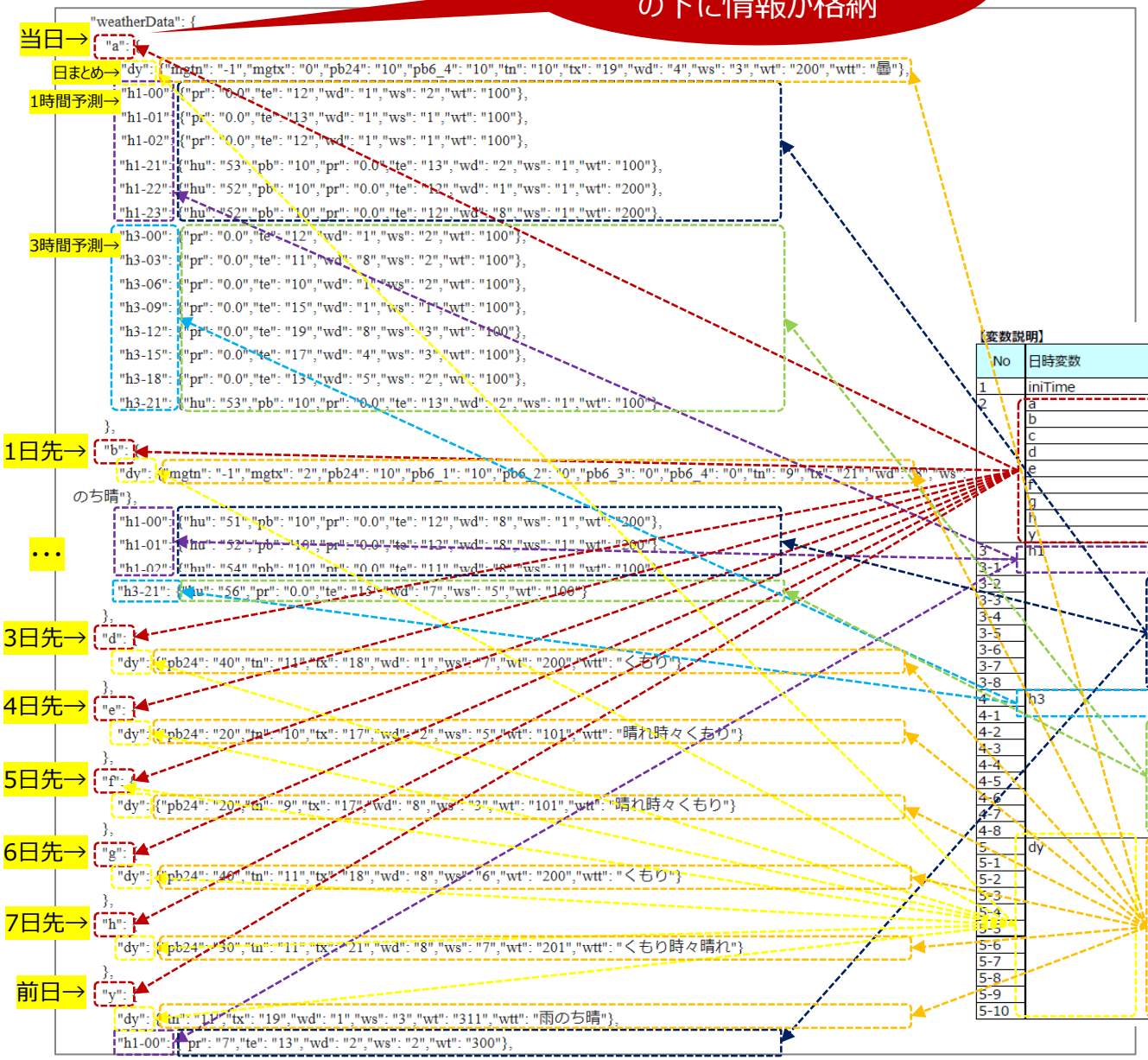
都市名	緯度	経度
札幌	43.06208	141.354361
仙台	38.268201	140.869426
東京	35.689486	139.691705
名古屋	35.181438	136.90642
大阪	34.693738	135.502165
福岡	33.590198	130.401719
沖縄	26.212523	127.680771

都市名	緯度	経度
アメリカ　ワシントンDC（ホワイトハウス）	38.897748	-77.03658
アメリカ　ニューヨーク（タイムズスクエア）	40.758219	-73.985547
ブラジル　サンパウロ（サンパウロ美術館）	23.557308	-46.65613
オーストラリア　シドニー（オペラハウス）	-33.856526	151.215232
中国　北京（天安門広場）	39.906193	116.397568
ロシア　モスクワ（赤の広場）	55.754136	37.620741
イギリス　ロンドン（ビッグ・ベン）	51.501143	-0.12469
南アフリカ　ケープタウン（V&A ウォーターフロント）	-33.899439	18.421451
インドネシア　ジャカルタ（モナス）	-6.175294	106.827139

※サンプルページの緯度経度の初期値はワシントンD.C（38.897748,-77.036580）となっています。

APIのレスポンスの解説

「weatherData」属性
の下に情報が格納



【変数説明】

No	日時変数	要素変数	説明
1	iniTime		予測開始日時　日本時刻
2	a		当日の情報
	b		1日先の情報
	c		2日先の情報
	d		3日先の情報
	e		4日先の情報
	f		5日先の情報
	g		6日先の情報
	h		7日先の情報
	y		1日前の情報
3	h1		1時間毎の情報
3-1		hh	該当時刻　00～23(時)
3-2		wt	天気　3桁のコード(※1)
3-3		te	気温　1℃刻み
3-4		hu	湿度　1%刻み
3-5		ws	風速　1m/s刻み
3-6		wd	風向　8方位のコード(※2)
3-7		pr	降水量　0.0, 0.5, 1, 以降1mm刻み
3-8		pb	降水確率　10%刻み　0～100(%)
4	h3		3時間毎の情報
4-1		hh	該当時刻　00,03,06,09,12,15,18,21(時)
4-2		wt	天気　3桁のコード(※1)
4-3		te	気温　1℃刻み
4-4		hu	湿度　1%刻み
4-5		ws	風速　1m/s刻み
4-6		wd	風向　8方位のコード(※2)
4-7		pr	降水量　0.0, 0.5, 1, 以降1mm刻み
4-8		pb	降水確率　10%刻み　0～100(%)
5	dy		日まとめの情報
5-1		wt	天気　3桁のコード(※1)
5-2		wtt	天気文言　(例) 晴れ時々くもり
5-3		tn	最低気温　1℃刻み
5-4		tx	最高気温　1℃刻み
5-5		mgtn	最低気温前日差　1℃刻み
5-6		mgtx	最高気温前日差　1℃刻み
5-7		pb6_1～pb6_4	6時間降水確率　10%刻み　0～100(%)
5-8		pb24	24時間降水確率　10%刻み　0～100(%)
5-9		ws	風速　1m/s刻み
5-10		wd	風向　8方位のコード(※2)

JSONの構造

【基本形】

{“項目1”:”値”}

キー　バリュー

【応用形①（2重構造）】

{“項目1”:{“項目2”:”値”}}

キー　バリュー

キー　バリュー

【応用形②（2重構造×複）】

{“項目1”:{“項目2”:”値”},
{“項目3”:”値”}}

キー　バリュー

キー　バリュー

サンプルプログラムを開きましょう

- img
- DreamAPI_6時間data簡易仕様書.pdf
- DreamAPI_jpwx(世界版)簡易仕様書.pdf
- index.css
- index.html
- Sample3-1.html
- Sample3-2.html
- Sample4-1.html
- Sample4-2.html
- Sample4-3.html
- Sample4-4.html
- Sample5-1.html
- Sample5-2.html
- カスタマイズ用Sample4-1.html
- カスタマイズ用Sample4-2.html
- カスタマイズ用Sample4-3.html
- カスタマイズ用Sample4-4.html
- カスタマイズ用Sample5-2.html

API仕様書　※重要

セミナーのメニュー画面

メニュー画面

コレをノートパッドで開く
※テキストエディタならOK

サンプルプログラム

編集用プログラム

③-1サンプルプログラムの説明

```
<!doctype html>
<html lang="ja">

<head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
    <meta http-equiv="Content-Style-Type" content="text/css" />
    <meta http-equiv="Content-Script-Type" content="text/javascript" />
    <title>JSONで気象情報を取得</title>
    <meta name="viewport" content="width=device-width" />
    <link rel="stylesheet" href="index.css" type="text/css" />
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
    <script type="text/javascript" language="javascript">
        //実行ボタン押下時の処理
        function onButtonClick() {
            //APIキーの設定 ※配布されたAPIキーを入力
            var api_key = "WX3120-Gn3Zz9FH";
            //緯度経度を取得
            var lat = document.forms.id_form1.lat.v
            var lon = document.forms.id_form1.lon.va
            //気象データの取得処理
            $.ajax({
                url: "https://fspweb03.halex.co.jp/wimage/hpd?sid=jpwx-p-world&lat=" + lat + "&lon=" + lon + "&key=" + api_key,
                dataType: "jsonp",//気象データの形式はJSONP
                jsonpCallback: "doJson"//JSONPのコールバック名はデフォルトdoJson
            })
            //正常に気象データを取得できた場合の処理
            .done(function(json) {
                //取得した気象データを見やすいように加工する
                var json_str = JSON.stringify(json, null, "□");
                var result = json_str.replace("¥n□□□□","").replace("¥n□□□]", "]");
                while(result !== json_str) {
                    json_str = json_str.replace("¥n□□□□","").replace("¥n□□□]", "]");
                    result = result.replace("¥n□□□□","").replace("¥n□□□]", "]");
                }
                //加工した気象データを画面に貼る
                var target = document.getElementById("output");
                target.innerText = json_str;
            })
            //気象データの取得に失敗した場合の処理
            .fail(function(json) {
                alert("気象データの取得に失敗しました。");
            });
        }
    </script>
</head>
```

「api_key」に配布した
APIキーを設定
※セミナー中は変更不要

サンプルプログラムの説明
③-2

③ ハレックスのAPIを使ってみよう（世界版jpwx）

仕様書：DreamAPI_jpwx(世界版)簡易仕様書.pdf
リクエストURL：https://fspweb03.halex.co.jp/wimage/hpd?sid=jpwx-p-world&lat=[緯度]&lon=[経度]&key=[アクセスキー]
[アクセスキー]：別途案内されているアクセスキーを設定してください。
[緯度][経度]：取得したい地点の緯度経度を設定してください。
（http,httpsでのリクエストが可能）

③-1 DreamAPIを使ってJSONデータを取得しよう

JavaScriptからDreamAPIを使って、JSONを画面に表示します。
ajaxを使って、JSONPでデータを取得します。
サンプルページはこちら

③-2 DreamAPIから本日18時の気温を取得しよう

JavaScriptからDreamAPIを使って、
取得したJSONから本日18時の気
サンプルページはこちら

クリック

③-2-1 DreamAPIから明日15時の予想気温を取得しよう

JavaScriptからDreamAPIを使って、JSONを取得します。
取得したJSONから明日15時の気温を表示します。
③-2で使用したhtmlファイルの中身を変更すると簡単です。

③-2-2 DreamAPIから明後日8時の予想風速を取得しよう

JavaScriptからDreamAPIを使って、JSONを取得します。
取得したJSONから明後日8時の予想風速を表示します。
③-2で使用したhtmlファイルの中身を変更すると簡単です。

③-2 DreamAPIから本日18時の気温を取得しよう

サンプルプログラムを開きましょう

- img
- DreamAPI_6時間data簡易仕様書.pdf
- DreamAPI_jpwx(世界版)簡易仕様書.pdf
- index.css
- index.html
- Sample3-1.html
- Sample3-2.html
- Sample4-1.html
- Sample4-2.html
- Sample4-3.html
- Sample4-4.html
- Sample5-1.html
- Sample5-2.html
- カスタマイズ用Sample4-1.html
- カスタマイズ用Sample4-2.html
- カスタマイズ用Sample4-3.html
- カスタマイズ用Sample4-4.html
- カスタマイズ用Sample5-2.html

DreamAPI_6時間data簡易仕様書.pdf / DreamAPI_jpwx(世界版)簡易仕様書.pdf → API仕様書　※重要

index.html → セミナーのメニュー画面

メニュー画面

Sample3-1.html / Sample3-2.html / Sample4-1.html / Sample4-2.html / Sample4-3.html / Sample4-4.html / Sample5-1.html / Sample5-2.html → サンプルプログラム

コレをノートパッドで開く
※テキストエディタならOK

カスタマイズ用Sample4-1.html / カスタマイズ用Sample4-2.html / カスタマイズ用Sample4-3.html / カスタマイズ用Sample4-4.html / カスタマイズ用Sample5-2.html → 編集用プログラム

③-2 本日18時の気温

```
<!doctype html>
<html lang="ja">

<head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
    <meta http-equiv="Content-Style-Type" content="text/css" />
    <meta http-equiv="Content-Script-Type" content="text/javascript" />
    <title>本日18時の気温を取得</title>
    <meta name="viewport" content="width=device-width" />
    <link rel="stylesheet" href="index.css" type="text/css" />
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
    <script type="text/javascript" language="javascript">
        //実行ボタン押下時の処理
        function onButtonClick() {←ボタン押下イベントで実行
            //APIキーの設定 ※配布されたAPIキーを入力してください
            var api_key = "WX3120-Gn3Zz9FH";←配布したAPIキーを設定
            //緯度経度を取得
            var lat = document.forms.id_form1.lat.value;
            var lon = document.forms.id_form1.lon.value;←気象情報を取得したい地点の緯度経度を設定
            var out_txt = "";
            //気象データの取得処理
            $.ajax({
                url: "https://fspweb03.halex.co.jp/wimage/hpd?sid=jpwx-p-world&lat=" + lat + "&lon=" + lon + "&key=" + api_key,←APIリクエストを設定
                dataType: "jsonp",//気象データの形式はJSONP
                jsonpCallback: "doJson"//JSONPのコールバック名はデフォルトdoJson
            })
            //正常に気象データを取得できた場合の処理
            .done(function(json) {
                //「a」：当日
                //「h1-18」：18時（1時間単位）
                //「te」：気温
                out_txt = "本日18時の気温：" + json["weatherData"]["a"]["h1-18"]["te"] + "℃";←要素を指定
                                                                                          [a]：当日
                                                                                          [h1-18]：18時
                                                                                          [te]：気温
            })
            //気象データの取得に失敗した場合の処理
            .fail(function(json) {
                alert("気象データの取得に失敗しました。");
                out_txt = "失敗";
            })
            //最終処理
            .always(function(json) {
                var target = document.getElementById("output");←変数「output」を表示
                target.innerHTML = out_txt;
            });
        }
    </script>
</head>
```

Ajaxとは、JavaScriptを
効率よく利用するためのフレームワーク

JavaScript

③-2 本日18時の気温

```
<body>
    <div id="outer">
        <div id="header">
            <a href="https://www.halex.co.jp/" target="_blank">
                <img src="https://www.halex.co.jp/wp-content/themes/halex/images/common/logo_header.png" >
            </a>
            <h1>DreamAPIから本日18時の気温を取得</h1>
        </div>
        <ul class="breadcrumb">
            <li>
                <a href="index.html" itemprop="url">
                    <span itemprop="title">トップ</span>
                </a>
            </li>
            <li>
                <span itemprop="title">DreamAPIから本日18時の気温を取得</span>
            </li>
        </ul>

        <div id="main">
            <div class="content">
                <h2>DreamAPIから本日18時の気温を取得</h2>
                <p>
                    JavaScriptからDreamAPIを使って、JSONを取得します。<br> 取得したJSONから本日18時の気温を表示します。<br>
                     本日の情報は「a」、18時の情報は「h1-18」、気温は「te」となります。<br>
                     <a href="img/koumoku.gif" target="_blank">サンプルはこちら</a>
                </p>
                <h3>緯度経度の設定</h3>
                <div>
                    <form name="form1" id="id_form1" action="#">
                        <label>緯度：
                            <input type="text" name="lat" id="lat" size="15" maxlength="15" value="38.897748" />経度：
                            <input type="text" name="lon" id="lon" size="15" maxlength="15" value="-77.036580" />
                        </label>
                        <input type="button" value="実行" onclick="onButtonClick();" />
                        <div id="output"></div>
                    </form>
                </div>
            </div>
        </div>
        <div id="footer">
            <div class="copy">株式会社ハレックス 問い合わせ先：03-5420-4311</div>
        </div>
    </div>
</body>

</html>[EOF]
```

サンプルプログラムの説明
③-2-1

サンプルプログラムを開きましょう

- img
- DreamAPI_6時間data簡易仕様書.pdf
- DreamAPI_jpwx(世界版)簡易仕様書.pdf
- index.css
- index.html
- Sample3-1.html
- Sample3-2.html
- Sample4-1.html
- Sample4-2.html
- Sample4-3.html
- Sample4-4.html
- Sample5-1.html
- Sample5-2.html
- カスタマイズ用Sample4-1.html
- カスタマイズ用Sample4-2.html
- カスタマイズ用Sample4-3.html
- カスタマイズ用Sample4-4.html
- カスタマイズ用Sample5-2.html

DreamAPI_6時間data簡易仕様書.pdf / DreamAPI_jpwx(世界版)簡易仕様書.pdf → API仕様書　※重要

index.html → セミナーのメニュー画面

メニュー画面

Sample3-1.html / Sample3-2.html / Sample4-1.html / Sample4-2.html / Sample4-3.html / Sample4-4.html / Sample5-1.html / Sample5-2.html → サンプルプログラム

コレをノートパッドで開く
※テキストエディタならOK

カスタマイズ用Sample4-1.html / カスタマイズ用Sample4-2.html / カスタマイズ用Sample4-3.html / カスタマイズ用Sample4-4.html / カスタマイズ用Sample5-2.html → 編集用プログラム

③ ハレックスのAPIを使ってみよう（世界版jpwx）

仕様書：DreamAPI_jpwx(世界版)簡易仕様書.pdf
リクエストURL：https://fspweb03.halex.co.jp/wimage/hpd?sid=jpwx-p-world&lat=[緯度]&lon=[経度]&key=[アクセスキー]
[アクセスキー]：別途案内されているアクセスキーを設定してください。
[緯度][経度]：取得したい地点の緯度経度を設定してください。
（http,httpsでのリクエストが可能）

③-1 DreamAPIを使ってJSONデータを取得しよう

JavaScriptからDreamAPIを使って、JSONを画面に表示します。
ajaxを使って、JSONPでデータを取得します。
サンプルページはこちら

③-2 DreamAPIから本日18時の気温を取得しよう

JavaScriptからDreamAPIを使って、
取得したJSONから本日18時の気
サンプルページはこちら

クリック

③-2-1 DreamAPIから明日15時の予想気温を取得しよう

JavaScriptからDreamAPIを使って、JSONを取得します。
取得したJSONから明日15時の気温を表示します。
③-2で使用したhtmlファイルの中身を変更すると簡単です。

③-2-2 DreamAPIから明後日8時の予想風速を取得しよう

JavaScriptからDreamAPIを使って、JSONを取得します。
取得したJSONから明後日8時の予想風速を表示します。
③-2で使用したhtmlファイルの中身を変更すると簡単です。

③-2-1 明日の15時の気温　※③-2からのカスタマイズ内容

```
<!doctype html>
<html lang="ja">

<head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
    <meta http-equiv="Content-Style-Type" content="text/css" />
    <meta http-equiv="Content-Script-Type" content="text/javascript" />
    <title>本日18時の気温を取得</title>
    <meta name="viewport" content="width=device-width" />
    <link rel="stylesheet" href="index.css" type="text/css" />
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
    <script type="text/javascript" language="javascript">
        //実行ボタン押下時の処理
        function onButtonClick() {←ボタン押下イベントで実行
            //APIキーの設定 ※配布されたAPIキーを入力してください
            var api_key = "WX3120-Gn3Zz9FH";←配布したAPIキーを設定
            //緯度経度を取得
            var lat = document.forms.id_form1.lat.value;
            var lon = document.forms.id_form1.lon.value;←気象情報を取得したい地点の緯度経度を設定
            var out_txt = "";
            //気象データの取得処理
            $.ajax({
                url: "https://fspweb03.halex.co.jp/wimage/hpd?sid=jpwx-p-world&lat=" + lat + "&lon=" + lon + "&key=" + api_key,←APIリクエストを設定
                dataType: "jsonp",//気象データの形式はJSONP
                jsonpCallback: "doJson"//JSONPのコールバック名はデフォルトdoJson
            })
            //正常に気象データを取得できた場合の処理
            .done(function(json) {
                //「a」：当日
                //「h1-18」：18時（1時間単位）
                //「te」：気温
                out_txt = "本日18時の気温：" + json["weatherData"]["a"]["h1-18"]["te"] + "℃";←要素を指定
            })                                                                          [b]：明日
            //気象データの取得に失敗した場合の処理                                        [h1-15]：15時
            .fail(function(json) {                                                      [te]：気温
                alert("気象データの取得に失敗しました。");
                out_txt = "失敗";
            })
            //最終処理
            .always(function(json) {
                var target = document.getElementById("output");←変数「output」を表示
                target.innerHTML = out_txt;
            });
        }
    </script>
</head>
```

Ajaxとは、JavaScriptを
効率よく利用するためのフレームワーク

JavaScript

サンプルプログラムの説明
③-2-2

サンプルプログラムを開きましょう

- img
- DreamAPI_6時間data簡易仕様書.pdf
- DreamAPI_jpwx(世界版)簡易仕様書.pdf
- index.css
- index.html
- Sample3-1.html
- Sample3-2.html
- Sample4-1.html
- Sample4-2.html
- Sample4-3.html
- Sample4-4.html
- Sample5-1.html
- Sample5-2.html
- カスタマイズ用Sample4-1.html
- カスタマイズ用Sample4-2.html
- カスタマイズ用Sample4-3.html
- カスタマイズ用Sample4-4.html
- カスタマイズ用Sample5-2.html

DreamAPI_6時間data簡易仕様書.pdf / DreamAPI_jpwx(世界版)簡易仕様書.pdf → API仕様書　※重要

index.html → セミナーのメニュー画面

メニュー画面

Sample3-1.html / Sample3-2.html / Sample4-1.html / Sample4-2.html / Sample4-3.html / Sample4-4.html / Sample5-1.html / Sample5-2.html → サンプルプログラム

コレをノートパッドで開く
※テキストエディタならOK

カスタマイズ用Sample4-1.html / カスタマイズ用Sample4-2.html / カスタマイズ用Sample4-3.html / カスタマイズ用Sample4-4.html / カスタマイズ用Sample5-2.html → 編集用プログラム

③ ハレックスのAPIを使ってみよう（世界版jpwx）

仕様書：DreamAPI_jpwx(世界版)簡易仕様書.pdf
リクエストURL：https://fspweb03.halex.co.jp/wimage/hpd?sid=jpwx-p-world&lat=[緯度]&lon=[経度]&key=[アクセスキー]
[アクセスキー]：別途案内されているアクセスキーを設定してください。
[緯度][経度]：取得したい地点の緯度経度を設定してください。
（http,httpsでのリクエストが可能）

③-1 DreamAPIを使ってJSONデータを取得しよう

JavaScriptからDreamAPIを使って、JSONを画面に表示します。
ajaxを使って、JSONPでデータを取得します。
サンプルページはこちら

③-2 DreamAPIから本日18時の気温を取得しよう

JavaScriptからDreamAPIを使って、
取得したJSONから本日18時の気
サンプルページはこちら

クリック

③-2-1 DreamAPIから明日15時の予想気温を取得しよう

JavaScriptからDreamAPIを使って、JSONを取得します。
取得したJSONから明日15時の気温を表示します。
③-2で使用したhtmlファイルの中身を変更すると簡単です。

③-2-2 DreamAPIから明後日8時の予想風速を取得しよう

JavaScriptからDreamAPIを使って、JSONを取得します。
取得したJSONから明後日8時の予想風速を表示します。
③-2で使用したhtmlファイルの中身を変更すると簡単です。

③-2-2 明後日8時の風速　※③-2からのカスタマイズ内容

```
<!doctype html>
<html lang="ja">

<head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
    <meta http-equiv="Content-Style-Type" content="text/css" />
    <meta http-equiv="Content-Script-Type" content="text/javascript" />
    <title>本日18時の気温を取得</title>
    <meta name="viewport" content="width=device-width" />
    <link rel="stylesheet" href="index.css" type="text/css" />
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
    <script type="text/javascript" language="javascript">
        //実行ボタン押下時の処理
        function onButtonClick() {←ボタン押下イベントで実行
            //APIキーの設定 ※配布されたAPIキーを入力してください
            var api_key = "WX3120-Gn3Zz9FH";←配布したAPIキーを設定
            //緯度経度を取得
            var lat = document.forms.id_form1.lat.value;
            var lon = document.forms.id_form1.lon.value;←気象情報を取得したい地点の緯度経度を設定
            var out_txt = "";
            //気象データの取得処理
            $.ajax({
                url: "https://fspweb03.halex.co.jp/wimage/hpd?sid=jpwx-p-world&lat=" + lat + "&lon=" + lon + "&key=" + api_key,←APIリクエストを設定
                dataType: "jsonp",//気象データの形式はJSONP
                jsonpCallback: "doJson"//JSONPのコールバック名はデフォルトdoJson
            })
            //正常に気象データを取得できた場合の処理
            .done(function(json) {
                //「a」：当日
                //「h1-18」：18時（1時間単位）
                //「te」：気温
                out_txt = "本日18時の気温：" + json["weatherData"]["a"]["h1-18"]["te"] + "℃";
            })
            //気象データの取得に失敗した場合の処理
            .fail(function(json) {
                alert("気象データの取得に失敗しました。");
                out_txt = "失敗";
            })
            //最終処理
            .always(function(json) {
                var target = document.getElementById("output");
                target.innerHTML = out_txt;←変数「output」を表示
            });
        }
    </script>
</head>
```

Ajaxとは、JavaScriptを効率よく利用するためのフレームワーク

単位に注意！！ ℃→m

←要素を指定
[c]：明後日
[h1-08]：8時
[ws]：風速

JavaScript

例題をやってみよう

HALEX HAPPY LIFE EXPERT

セミナーで編集してもらうプログラムです。メモ帳などで開いて編集して下さい。

- img
- DreamAPI_6時間data簡易仕様書.pdf
- DreamAPI_jpwx(世界版)簡易仕様書.pdf
- index.css
- index.html
- Sample3-1.html
- Sample3-2.html
- Sample4-1.html
- Sample4-2.html
- Sample4-3.html
- Sample4-4.html
- Sample5-1.html
- Sample5-2.html
- カスタマイズ用Sample4-1.html
- カスタマイズ用Sample4-2.html
- カスタマイズ用Sample4-3.html
- カスタマイズ用Sample4-4.html
- カスタマイズ用Sample5-2.html

カスタマイズ用ファイルはこのファイルをコピーしたもの

```
<!doctype html>
<html lang="ja">

<head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
    <meta http-equiv="Content-Style-Type" content="text/css" />
    <meta http-equiv="Content-Script-Type" content="text/javascript" />
    <title>本日18時の気温を取得</title>
    <meta name="viewport" content="width=device-width" />
    <link rel="stylesheet" href="index.css" type="text/css" />
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
    <script type="text/javascript" language="javascript">
        //実行ボタン押下時の処理
        function onButtonClick() {
            //APIキーの設定 ※配布されたAPIキーを入力してください
            var api_key = "WX3120-Gn3Zz9FH";
            //緯度経度を取得
            var lat = document.forms.id_form1.lat.value;
            var lon = document.forms.id_form1.lon.value;
            var out_txt = "";
            //気象データの取得処理
            $.ajax({
                url: "https://fspweb03.halex.co.jp/wimage/hpd?sid=jpwx-p-world&lat=" + lat + "&lon=" + lon + "&key=" + api_key,
                dataType: "jsonp",//気象データの形式はJSONP
                jsonpCallback: "doJson"//JSONPのコールバック名はデフォルトdoJson
            })
            //正常に気象データを取得できた場合の処理
            .done(function(json) {
                //「a」：当日
                //「h1-18」：18時（1時間単位）
                //「te」：気温
                out_txt = "本日18時の気温：" + json["weatherData"]["a"]["h1-18"]["te"] + "℃";
            `
                象データの取得に失敗した場合の処理
                (function(json) {
                lert("気象データの取得に失敗しました。");
                ut_txt = "失敗";

                最終処理
            .always(function(json) {
                var target = document.getElementById("output");
                target.innerHTML = out_txt;
            });
        }
    </script>
</head>
```

みんなでやってみよう！！

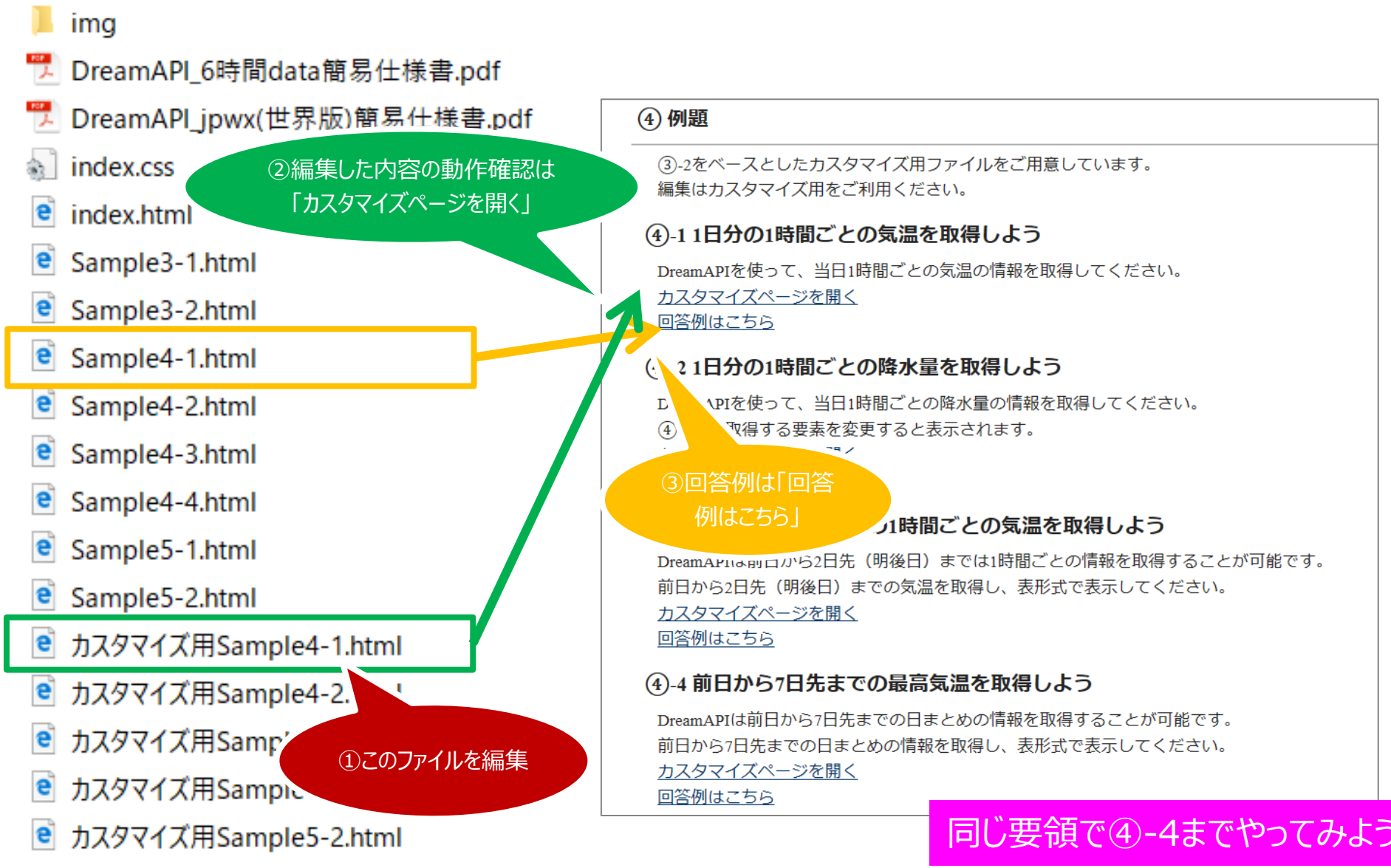
④-1～④-4の例題で編集するファイル

編集用プログラムの動作を確認するには、メニューの「カスタマイズページを開く」をクリック

- img
- DreamAPI_6時間data簡易仕様書.pdf
- DreamAPI_jpwx(世界版)簡易仕様書.pdf
- index.css
- index.html
- Sample3-1.html
- Sample3-2.html
- Sample4-1.html
- Sample4-2.html
- Sample4-3.html
- Sample4-4.html
- Sample5-1.html
- Sample5-2.html
- カスタマイズ用Sample4-1.html
- カスタマイズ用Sample4-2.html
- カスタマイズ用Sample4-3.html
- カスタマイズ用Sample4-4.html
- カスタマイズ用Sample5-2.html

まずは例題④-1

編集したプログラムの動作を確認するには、メニューの「カスタマイズページを開く」をクリック



同じ要領で④-4までやってみよう！！

（ここからは事前配布しない）

例題④-1解説

④-1　1日分の1時間ごとの気温を取得しよう

```
<!doctype html>
<html lang="ja">

<head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
    <meta http-equiv="Content-Style-Type" content="text/css" />
    <meta http-equiv="Content-Script-Type" content="text/javascript" />
    <title>1日分の気温を取得</title>
    <meta name="viewport" content="width=device-width" />
    <link rel="stylesheet" href="index.css" type="text/css" />
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
    <script type="text/javascript" language="javascript">
        //実行ボタン押下時の処理
        function onButtonClick() {
            //APIキーの設定 ※配布されたAPIキーを入力してください
            var api_key = "WX312O-Gn3Zz9FH";
            //緯度経度を取得
            var lat = document.forms.id_form1.lat.value;
            var lon = document.forms.id_form1.lon.value;;
            var out_txt = "";
            var out_th = "";
            var out_td = "";
            //気象データの取得処理
            $.ajax({
                url: "https://fspweb03.halex.co.jp/wimage/hpd?sid=jpwx-p-world&lat=" + lat + "&lon=" + lon + "&key=" + api_key,
                dataType: "jsonp",
                jsonpCallback: "doJson"
            })
            //正常に気象データを取得できた場合の処理
            .done(function(json) {
                //keyの数だけループする
                for (let key of Object.keys(json["weatherData"]["a"])) {
                    //h1-〇〇のデータのみ使用する
                    if (key.indexOf("h1") == 0) {
                        out_th = out_th + "<th>" + key.replace("h1-", "") + "</th>";
                        // 「te」：気温
                        out_td = out_td + "<td>" + json["weatherData"]["a"][key]["te"] + "</td>";
                    }
                }
                out_txt = "<table><tr>" + out_th + "</tr><tr>" + out_td + "</tr></table>";
            })
            //気象データの取得に失敗した場合の処理
            .fail(function(json) {
                alert("気象データの取得に失敗しました。");
                out_txt = "失敗";
            })
            //最終処理
            .always(function(json) {
                var target = document.getElementById("output");
                target.innerHTML = out_txt;
            });
        }
    </script>
</head>
```

Ajaxとは、JavaScriptを
効率よく利用するためのフレームワーク

←ボタン押下イベントで実行

←配布したAPIキーを設定

←気象情報を取得したい地点の緯度経度を設定

←APIリクエストを設定

←時間分だけループ　※[“a”]配下の時間毎に処理

ループ処理

←要素を指定
[a]：当日
[key]：日付←APIで取得した日付
[te]：気温

↑ループ内でテーブル表示用に編集

←変数「output」を表示

JavaScript

④-1　1日分の1時間ごとの気温を取得しよう

```
<body>↵
    <div id="outer">↵
        <div id="header">↵
            <a href="https://www.halex.co.jp/" target="_blank">↵
                <img src="https://www.halex.co.jp/wp-content/themes/halex/images/common/logo_header.png" >↵
            </a>↵
            <h1>1日分の1時間ごとの気温を取得</h1>↵
        </div>↵
        <ul class="breadcrumb">↵
            <li>↵
                <a href="index.html" itemprop="url">↵
                    <span itemprop="title">トップ</span>↵
                </a>↵
            </li>↵
            <li>↵
                <span itemprop="title">1日分の1時間ごとの気温を取得</span>↵
            </li>↵
        </ul>↵
↵
        <div id="main">↵
            <div class="content">↵
                <h2>1日分の1時間ごとの気温を取得</h2>↵
                <p>DreamAPIを使って、当日1時間ごとの気温の情報を取得してください。</p>↵
                <h3>緯度経度の設定</h3>↵
                <div>↵
                    <form name="form1" id="id_form1" action="#">↵
                        <label>緯度：↵
                        <input type="text" name="lat" id="lat" size="15" maxlength="15" value="38.897748" />経度：↵
                        <input type="text" name="lon" id="lon" size="15" maxlength="15" value="-77.036580" />↵
                        </label>↵
                        <input type="button" value="実行" onclick="onButtonClick();" />↵
                    </form>↵
                    <div id="output"></div>↵
                </div>↵
            </div>↵
        </div>↵
        <div id="footer">↵
            <div class="copy">株式会社ハレックス 問い合わせ先：03-5420-4311</div>↵
        </div>↵
    </div>↵
</body>↵
↵
</html>[EOF]
```

例題④-2解説

ただ、④-2を実現するには

④-1を１か所修正するだけ！！

修正箇所を考えてみよう。

④-2　1日分の1時間ごとの降水量を取得しよう

```
<!doctype html>
<html lang="ja">

<head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
    <meta http-equiv="Content-Style-Type" content="text/css" />
    <meta http-equiv="Content-Script-Type" content="text/javascript" />
    <title>1日分の気温を取得</title>
    <meta name="viewport" content="width=device-width" />
    <link rel="stylesheet" href="index.css" type="text/css" />
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
    <script type="text/javascript" language="javascript">
        //実行ボタン押下時の処理
        function onButtonClick() {
            //APIキーの設定 ※配布されたAPIキーを入力してください
            var api_key = "WX3120-Gn3Zz9FH";
            //緯度経度を取得
            var lat = document.forms.id_form1.lat.value;
            var lon = document.forms.id_form1.lon.value;;
            var out_txt = "";
            var out_th = "";
            var out_td = "";
            //気象データの取得処理
            $.ajax({
                url: "https://fspweb03.halex.co.jp/wimage/hpd?sid=jpwx-p-world&lat=" + lat + "&lon=" + lon + "&
                dataType: "jsonp",
                jsonpCallback: "doJson"
            })
            //正常に気象データを取得できた場合の処理
            .done(function(json) {
                //keyの数だけループする
                for (let key of Object.keys(json["weatherData"]["a"])) {
                    //h1-〇〇のデータのみ使用する
                    if (key.indexOf("h1") == 0) {
                        out_th = out_th + "<th>" + key.replace("h1-", "") + "</th>";
                        // 「te」：気温
                        out_td = out_td + "<td>" + json["weatherData"]["a"][key]["te"] + "</td>";
                    }
                }
                out_txt = "<table><tr>" + out_th + "</tr><tr>" + out_td + "</tr></table>";
            })
            //気象データの取得に失敗した場合の処理
            .fail(function(json) {
                alert("気象データの取得に失敗しました。");
                out_txt = "失敗";
            })
            //最終処理
            .always(function(json) {
                var target = document.getElementById("output");
                target.innerHTML = out_txt;
            });
        }
    </script>
</head>
```

Ajaxとは、JavaScriptを
効率よく利用するためのフレームワーク

←ボタン押下イベントで実行

←配布したAPIキーを設定

←気象情報を取得したい地点の緯度経度を設定

←時間分だけループ　※["a"]

ループ処理

修正はココだけ
["te"]気温
→["pr"]降水量

←要素を指定
[a]：当日
[key]：日付←APIで取得した日付
[te]：気温

↑ループ内でテーブル表示用に編集

←変数「output」を表示

JavaScript

④-2　1日分の1時間ごとの降水量を取得しよう

以下は④-1のサンプルプログラム

```
56
57
58
59              <a href="https://www.halex.co.jp/" target="_blank">↵
60                  <img src="https://www.halex.co.jp/wp-content/themes/halex/images/common/logo_header.png" >↵
61              </a>↵
62              <h1>1日分の1時間ごとの気温を取得</h1>↵
63          </div>↵
64          <ul class="breadcrumb">↵
65              <li>↵
66                  <a href="index.html" itemprop="url">↵
67                      <span itemprop="title">トップ</span>↵
68                  </a>↵
69              </li>↵
70              <li>↵
71                  <span itemprop="title">1日分の1時間ごとの気温を取得</span>↵
72              </li>↵
73          </ul>↵
74  ↵
75          <div id="main">↵
76              <div class="content">↵
77                  <h2>1日分の1時間ごとの気温を取得</h2>↵
78                  <p>DreamAPIを使って、当日1時間ごとの気温の情報を取得してください。</p>↵
79                  <h3>緯度経度の設定</h3>↵
80                  <div>↵
81                      <form name="form1" id="id_form1" action="#">↵
82                          <label>緯度：↵
83                          <input type="text" name="lat" id="lat" size="15" maxlength="15" value="38.897748" />経度：↵
84                          <input type="text" name="lon" id="lon" size="15" maxlength="15" value="-77.036580" />↵
85                          </label>↵
86                          <input type="button" value="実行" onclick="onButtonClick();" />↵
87                      </form>↵
88                      <div id="output"></div>↵
89                  </div>↵
90              </div>↵
91          </div>↵
92          <div id="footer">↵
93              <div class="copy">株式会社ハレックス 問い合わせ先：03-5420-4311</div>↵
94          </div>↵
95      </div>↵
96  </body>↵
97  ↵
98  </html>[EOF]
```

例題④-3解説
（処理部に限定して解説）

④-3 前日から2日先までの1時間ごとの気温を取得しよう

処理部に限定して解説

```
        //気象データの取得処理
        $.ajax({
            url: "https://fspweb03.halex.co.jp/wimage/hpd?sid=jpwx-p-world&lat=" + lat + "&lon=" + lon + "&key=" + api_key,
            dataType: "jsonp",
            jsonpCallback: "doJson"
        })
        //正常に気象データを取得できた場合の処理
        .done(function(json) {
            iniTime_HH = Number(moment(json["iniTime"].replace(" JST", ""), "YYYY/MM/DD HH:mm").format('HH'));

            //前日分データの取得処理
            for (let key of Object.keys(json["weatherData"]["y"])) {
                //h1-〇〇のデータのみ使用する
                if (key.indexOf("h1") == 0) {
                    out_th_yes = out_th_yes + "<th>" + key.replace("h1-", "") + "</th>";
                    out_td_yes = out_td_yes + "<td class=¥"passed¥">" + json["weatherData"]["y"][key]["te"] + "</td>";
                }
            }
            //当日分データの取得処理
            for (let key of Object.keys(json["weatherData"]["a"])) {
                //h1-〇〇のデータのみ使用する
                if (key.indexOf("h1") == 0) {
                    out_th_tod = out_th_tod + "<th>" + key.replace("h1-", "") + "</th>";
                    if (iniTime_HH > Number(key.replace("h1-", ""))) {
                        out_td_tod = out_td_tod + "<td class=¥"passed¥">";
                    } else {
                        out_td_tod = out_td_tod + "<td>";
                    }
                    out_td_tod = out_td_tod + json["weatherData"]["a"][key]["te"] + "</td>";
                }
            }
            //2日目分データの取得処理
            for (let key of Object.keys(json["weatherData"]["b"])) {
                //h1-〇〇のデータのみ使用する
                if (key.indexOf("h1") == 0) {
                    out_th_dat = out_th_dat + "<th>" + key.replace("h1-", "") + "</th>";
                    out_td_dat = out_td_dat + "<td>" + json["weatherData"]["b"][key]["te"] + "</td>";
                }
            }
            //3日目分データの取得処理
            for (let key of Object.keys(json["weatherData"]["c"])) {
                //h1-〇〇のデータのみ使用する
                if (key.indexOf("h1") == 0) {
                    out_th_tdat = out_th_tdat + "<th>" + key.replace("h1-", "") + "</th>";
                    out_td_tdat = out_td_tdat + "<td>" + json["weatherData"]["c"][key]["te"] + "</td>";
                }
            }
            out_txt = "<p>前日のデータ</p><table><tr>" + out_th_yes + "</tr><tr>" + out_td_yes + "</tr></table>";
            out_txt = out_txt + "<p>当日のデータ</p><table><tr>" + out_th_tod + "</tr><tr>" + out_td_tod + "</tr></table>"
            out_txt = out_txt + "<p>明後日のデータ</p><table><tr>" + out_th_dat + "</tr><tr>" + out_td_dat + "</tr></table>"
            out_txt = out_txt + "<p>明々後日のデータ</p><table><tr>" + out_th_tdat + "</tr><tr>" + out_td_tdat + "</tr></table>"
```

ループ処理

←時間分だけループ　※["y"]配下の時間毎に処理

←要素を指定
[a]：当日
[key]：日付←APIで取得した日付
[te]：気温

←APIリクエストを設定

↑テーブル表示用に編集

←テーブル表示用に編集

例題④-4解説
（処理部に限定して解説）

④-4 前日から7日先までの最高気温を取得しよう

処理部に限定して解説

```
    <script type="text/javascript" language="javascript">
        function onButtonClick() {
            //APIキーの設定 ※配布されたAPIキーを入力してください
            var api_key = "WX312O-Gn3Zz9FH";
            //緯度経度を取得
            var lat = document.forms.id_form1.lat.value;
            var lon = document.forms.id_form1.lon.value;;

            var out_txt = "";
            var out_th = "";
            var out_td = "";
            var iniTime;

            //気象データの取得処理
            $.ajax({
                url: "https://fspweb03.halex.co.jp/wimage/hpd?sid=jpwx-p-world&lat=" + lat + "&lon=" + lon + "&key=" + api_key,
                dataType: "jsonp",
                jsonpCallback: "doJson"
            })
            //正常に気象データを取得できた場合の処理
            .done(function(json) {
                out_th = "<th>前 日</th><th>当 日</th><th>1日先</th><th>2日先</th><th>3日先</th><th>4日先</th><th>5日先</th><th>6日先</th><th
                out_td = "<td>" + json["weatherData"]["y"]["dy"]["tx"] + "</td>";
                out_td = out_td + "<td>" + json["weatherData"]["a"]["dy"]["tx"] + "</td>";
                out_td = out_td + "<td>" + json["weatherData"]["b"]["dy"]["tx"] + "</td>";
                out_td = out_td + "<td>" + json["weatherData"]["c"]["dy"]["tx"] + "</td>";
                out_td = out_td + "<td>" + json["weatherData"]["d"]["dy"]["tx"] + "</td>";
                out_td = out_td + "<td>" + json["weatherData"]["e"]["dy"]["tx"] + "</td>";
                out_td = out_td + "<td>" + json["weatherData"]["f"]["dy"]["tx"] + "</td>";
                out_td = out_td + "<td>" + json["weatherData"]["g"]["dy"]["tx"] + "</td>";

                //iniTimeが12:00より前は7日目のデータが入っていないので、「-（ハイフン）」を設定する
                iniTime = new Date(json["iniTime"].replace(" JST","")).getHours();
                if (iniTime < 12) {
                    out_td = out_td + "<td>-</td>";
                } else {
                    out_td = out_td + "<td>" + json["weatherData"]["y"]["dy"]["tx"] + "</td>";
                }
                out_txt = "<p>前日から7日先までの最高気温</p><table><tr>" + out_th + "</tr><tr>" + out_td + "</tr></table>";
            })
            //気象データの取得に失敗した場合の処理
            .fail(function(json) {
                console.log(json);
                out_txt = "失敗";
            })
            //最終処理
            .always(function(json) {
                var target = document.getElementById("output");
                target.innerHTML = out_txt;
            });
        }
    </script>
```

←APIリクエストを設定

↓テーブル表示用に編集

←要素を指定
[y-g]：前日～7日先
[dy]：日まとめ
[tx]：最高気温

←時間によって7日目のデータが取得出来ないための処理

←画面表示用の処理

サンプルプログラムの説明

⑤-1

⑤-1 DreamAPIを使って雨雲画像を取得しよう

⑤ ハレックスのAPIを使ってみよう（雨雲画像）

仕様書：DreamAPI_6時間data簡易仕様書.pdf
リクエストURL：https://fspweb01.halex.co.jp/wimage/hpd?sid=wimage-p-service&rem=all&key=[アクセスキー]&lat=[緯度]&lon=[経度]
[アクセスキー]：別途案内されているアクセスキーを設定してください。
[緯度][経度]：取得したい地点の緯度経度を設定してください。
（http,httpsでのリクエストが可能）

⑤-1 DreamAPIを使って雨雲画像を取得しよう

JavaScriptからDreamAPIを使って、 表示します。
ajaxを使って、JSONPで画像を取
サンプルページはこちら

クリック

⑤-2 6時間先の雨雲画像を取得しよう

DreamAPIは1時間毎に6時間先までの雨雲画像を取得することが可能です。
6時間先の雨雲画像を取得し、地図上に表示してください。
カスタマイズページを開く
回答例はこちら

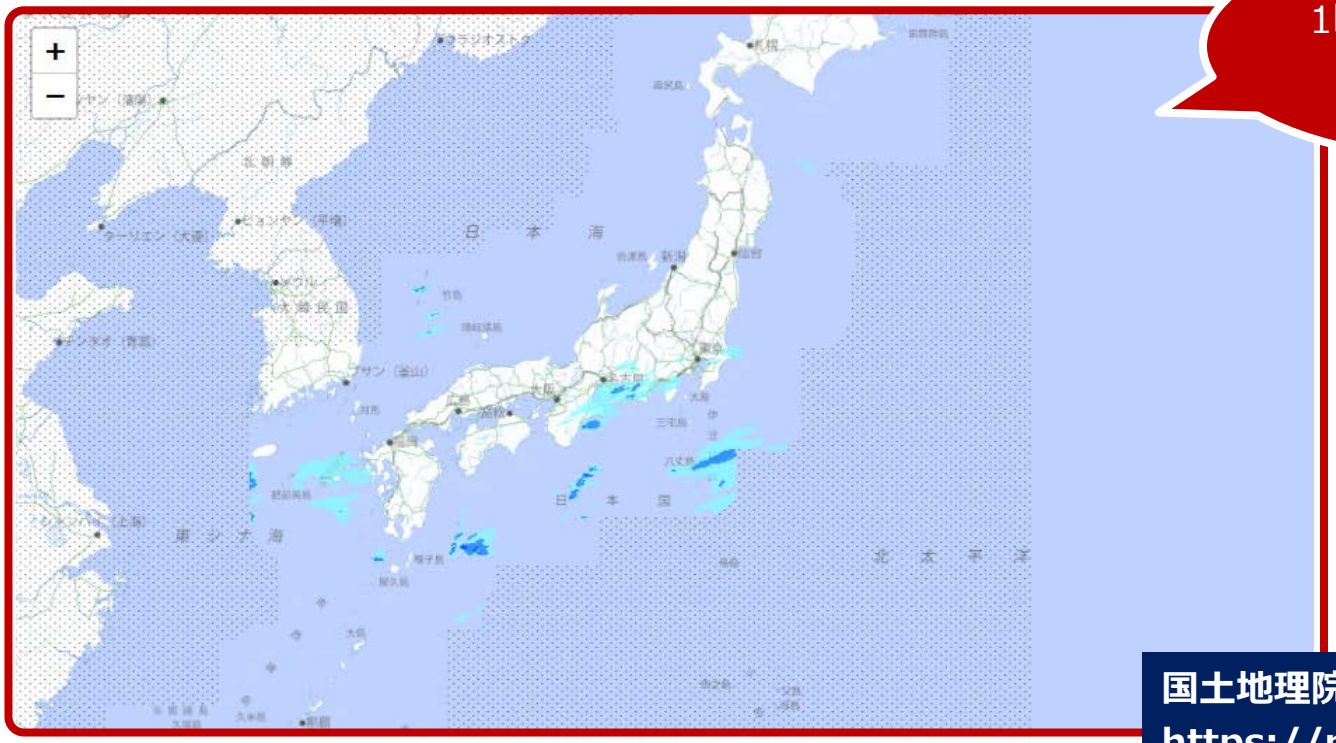
⑤-1 DreamAPIを使って雨雲画像を取得しよう

DreamAPIからJSONで雨雲画像を取得

トップ > DreamAPIからJSONで雨雲画像を取得

DreamAPIからJSONで雨雲画像を取得

JavaScriptからDreamAPIを使って、JSONを取得します。
取得したJSONから雨雲画像を取得して地図に表示します。
雨雲画像は、JSONの「precipitation」の「fileName」から取得します。
地図は、国土地理院の地理院タイルを使用しています。



1時間先の雨雲画像
を表示

株式会社ハレックス 問い合わせ先：03-5420-4311

国土地理院の地図を利用しています。
https://maps.gsi.go.jp/help/index.html

サンプルプログラムを開きましょう

- img
- DreamAPI_6時間data簡易仕様書.pdf
- DreamAPI_jpwx(世界版)簡易仕様書.pdf
- index.css
- index.html
- Sample3-1.html
- Sample3-2.html
- Sample4-1.html
- Sample4-2.html
- Sample4-3.html
- Sample4-4.html
- Sample5-1.html
- Sample5-2.html
- カスタマイズ用Sample4-1.ht…
- カスタマイズ用Sample4-2.html
- カスタマイズ用Sample4-3.html
- カスタマイズ用Sample4-4.html
- カスタマイズ用Sample5-2.html

コレをノートパッドで開く
※テキストエディタならOK

API仕様書　※重要

セミナーのメニュー画面

メニュー画面

サンプルプログラム

編集用プログラム

⑤-1 DreamAPIからJSONで雨雲画像を取得

```
<!doctype html>
<html lang="ja">

<head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
    <meta http-equiv="Content-Style-Type" content="text/css" />
    <meta http-equiv="Content-Script-Type" content="text/javascript" />
    <title>DreamAPIからJSONで雨雲画像を取得</title>
    <meta name="viewport" content="width=device-width" />
    <title></title>
    <link rel="stylesheet" href="https://unpkg.com/leaflet@1.2.0/dist/leaflet.css" /><!--地図関連-->
    <link rel="stylesheet" href="index.css" type="text/css" />
    <script src="https://unpkg.com/leaflet@1.2.0/dist/leaflet.js"></script><!--地図関連-->
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
    <script type="text/javascript" language="javascript">
        //ロード時の処理
        function map_load(){
            //APIキーの設定 ※配布されたAPIキーを入力してください
            var api_key = "WX3120-Gn3Zz9FH";

            //地図の生成
            var map = L.map('map');
            L.tileLayer('https://cyberjapandata.gsi.go.jp/xyz/pale/{z}/{x}/{y}.png', {
                attribution: "<a href='https://maps.gsi.go.jp/development/ichiran.html' target='_blank'>地理院タイル</a>"
            }).addTo(map);
            map.setView([35.3622222, 138.7313889], 5);

            //ajax通信でAPIからデータを取得
            $.ajax({
                url: "https://fspweb01.halex.co.jp/wimage/hpd?sid=wimage-p-service&rem=all&lat=35&lon=139&scalex=20&scaley=30&bsx=640&bsy=840&proj=3&key=" + api_key
                dataType: "jsonp",//気象データの形式はJSONP
                jsonpCallback: "doJson"//JSONPのコールバック名はデフォルトdoJson
            })
            //正常に気象データを取得できた場合の処理
            .done(function(json) {
                //「fileName」：メッシュ画像ファイル名
                fileName = json["precipitation"][0]["fileName"];
                //画像貼り付け位置
                north = json["precipitation"][0]["north"];
                south = json["precipitation"][0]["south"];
                east = json["precipitation"][0]["east"];
                west = json["precipitation"][0]["west"];

                var imageUrl = "https://fspweb01.halex.co.jp/wimage/img?sid=wimage-image-service&key=" + api_key + "&fileName=" + fileName
                var imageBounds = [[north, west], [south, east]];
                var tegaki = L.imageOverlay(imageUrl, imageBounds).addTo(map);
            })
            //気象データの取得に失敗した場合の処理
            .fail(function(json) {
                alert("気象データの取得に失敗しました。");
            });
        }
    </script>

    <style>
        #map {height: 500px; width: 100%; margin-top:10px;}
    </style>
</head>
```

地図活用のフレームワーク

Ajaxとは、JavaScriptを
効率よく利用するためのフレームワーク

←画面読み込み時に実行

←配布したAPIキーを設定

←地図の種類を設定

←Leafletを設定

←地図の中心位置を設定

↑APIリクエストを設定

JavaScript

←雨雲画像のファイルパスを取得

←雨雲画像の四隅の緯度経度を取得

←雨雲画像取得URIを設定

←地図に雨雲画像をセット

←地図の表示サイズを設定

⑤-1 DreamAPIからJSONで雨雲画像を取得

```
<body onLoad="map_load()">
    <div id="outer">
        <div id="header">
            <a href="https://www.halex.co.jp/" target="_blank">
                <img src="https://www.halex.co.jp/wp-content/themes/halex/images/common/logo_header.png" >
            </a>
            <h1>DreamAPIからJSONで雨雲画像を取得</h1>
        </div>
        <ul class="breadcrumb">
            <li>
                <a href="index.html" itemprop="url">
                    <span itemprop="title">トップ</span>
                </a>
            </li>
            <li>
                <span itemprop="title">DreamAPIからJSONで雨雲画像を取得</span>
            </li>
        </ul>
        <div id="main">
            <div class="content">
                <h2>DreamAPIからJSONで雨雲画像を取得</h2>
                <p>
                    JavaScriptからDreamAPIを使って、JSONを取得します。<br>
                    取得したJSONから雨雲画像を取得して地図に表示します。<br>
                    雨雲画像は、JSONの「precipitation」の「fileName」から取得します。<br>
                    地図は、国土地理院の地理院タイルを使用しています。<br>
                </p>
                <div id="map"></div>  ←地図表示
            </div>
        </div>
        <div id="footer">
            <div class="copy">株式会社ハレックス 問い合わせ先：03-5420-4311</div>
        </div>
    </div>
</body>

</html>[EOF]
```

サンプルプログラムの説明
⑤-2

⑤-1からどこを修正すれば良いか
考えてみよう！！

セミナーで編集してもらうプログラムです。メモ帳などで開いて編集して下さい。

- img
- DreamAPI_6時間data簡易仕様書.pdf
- DreamAPI_jpwx(世界版)簡易仕様書.pdf
- index.css
- index.html
- Sample3-1.html
- Sample3-2.html
- Sample4-1.html
- Sample4-2.html
- Sample4-3.html
- Sample4-4.html
- Sample5-1.html
- Sample5-2.html
- カスタマイズ用Sample4-1.html
- カスタマイズ用Sample4-2.html
- カスタマイズ用Sample4-3.html
- カスタマイズ用Sample4-4.html
- カスタマイズ用Sample5-2.html

カスタマイズ用ファイルはこのファイルをコピーしたもの

⑤-2の例題で編集するファイル

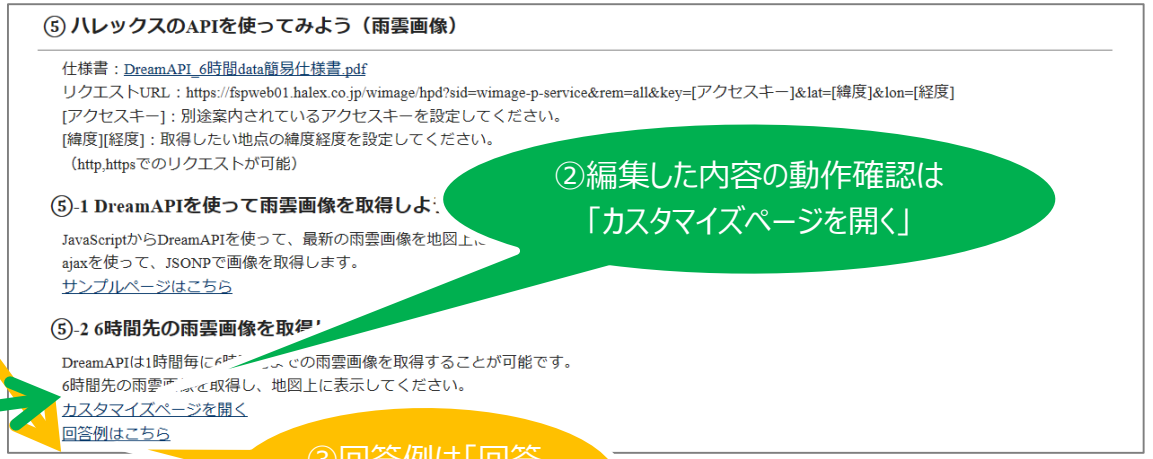
```
<!doctype html>
<html lang="ja">

<head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
    <meta http-equiv="Content-Style-Type" content="text/css" />
    <meta http-equiv="Content-Script-Type" content="text/javascript" />
    <title>本日18時の気温を取得</title>
    <meta name="viewport" content="width=device-width" />
    <link rel="stylesheet" href="index.css" type="text/css" />
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
    <script type="text/javascript" language="javascript">
        //実行ボタン押下時の処理
        function onButtonClick() {
            //APIキーの設定 ※配布されたAPIキーを入力してください
            var api_key = "WX3120-Gn3Zz9FH";
            //緯度経度を取得
            var lat = document.forms.id_form1.lat.value;
            var lon = document.forms.id_form1.lon.value;
            var out_txt = "";
            //気象データの取得処理
            $.ajax({
                url: "https://fspweb03.halex.co.jp/wimage/hpd?sid=jpwx-p-world&lat=" + lat + "&lon=" + lon + "&key=" + api_key,
                dataType: "jsonp",//気象データの形式はJSONP
                jsonpCallback: "doJson"//JSONPのコールバック名はデフォルトdoJson
            })
            //正常に気象データを取得できた場合の処理
            .done(function(json) {
                //「a」：当日
                //「h1-18」：18時（1時間単位）
                //「te」：気温
                out_txt = "本日18時の気温：" + json["weatherData"]["a"]["h1-18"]["te"] + "℃";
            })
            //気象データの取得に失敗した場合の処理
            .fail(function(json) {
                alert("気象データの取得に失敗しました。");
                out_txt = "失敗";
            })
            //.終処理
            .always(function(json) {
                var target = document.getElementById("output");
                target.innerHTML = out_txt;
            });
        }
    </script>
</head>
```

編集用プログラムの動作を確認するには、メニューの「カスタマイズページを開く」をクリック

- img
- DreamAPI_6時間data簡易仕様書.pdf
- DreamAPI_jpwx(世界版)簡易仕様書.pdf
- index.css
- index.html
- Sample3-1.html
- Sample3-2.html
- Sample4-1.html
- Sample4-2.html
- Sample4-3.html
- Sample4-4.html
- Sample5-1.html
- Sample5-2.html
- カスタマイズ用Sample4-1.html
- ...4-2.html
- ...4-3.html
- カスタマ...用Sample4-4.html
- カスタマイズ用Sample5-2.html

①このファイルを編集



②編集した内容の動作確認は「カスタマイズページを開く」

③回答例は「回答例はこちら」

⑤-2 6時間先の雨雲画像を取得

⑤-1サンプルファイル

```
<!doctype html>
<html lang="ja">

<head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
    <meta http-equiv="Content-Style-Type" content="text/css" />
    <meta http-equiv="Content-Script-Type" content="text/javascript" />
    <title>DreamAPIからJSONで雨雲画像を取得</title>
    <meta name="viewport" content="width=device-width" />
    <title></title>
    <link rel="stylesheet" href="https://unpkg.com/leaflet@1.2.0/dist/leaflet.css" /><!--地図関連-->
    <link rel="stylesheet" href="index.css" type="text/css" />
    <script src="https://unpkg.com/leaflet@1.2.0/dist/leaflet.js"></script><!--地図関連-->
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
    <script type="text/javascript" language="javascript">
        //ロード時の処理
        function map_load(){
            //APIキーの設定 ※配布されたAPIキーを入力してください
            var api_key = "WX3120-Gn3Zz9FH";

            //地図の生成
            var map = L.map('map');
            L.tileLayer('https://cyberjapandata.gsi.go.jp/xyz/pale/{z}/{x}/{y}.png', {
                attribution: "<a href='https://maps.gsi.go.jp/development/ichiran.html' target='_blank'>地理院タイル</a>"
            }).addTo(map);
            map.setView([35.3622222, 138.7313889], 5);

            //ajax通信でAPIからデータを取得
            $.ajax({
                url: "https://fspweb01.halex.co.jp/wimage/hpd
                dataType: "jsonp",//気象データの形式はJSONP
                jsonpCallback: "doJson"//JSONPのコールバック名は
            })
            //正常に気象データを取得できた場合の処理
            .done(function(json) {
                // 「fileName」：メッシュ画像ファイル名
                fileName = json["precipitation"][0]["fileName"];
                //画像貼り付け位置
                north = json["precipitation"][0]["north"];
                south = json["precipitation"][0]["south"];
                east = json["precipitation"][0]["east"];
                west = json["precipitation"][0]["west"];

                var imageUrl = "https://fspweb01.halex.co.jp/wimage/img?sid=wimage-image-service&key=" + api_key + "&fileName=" + fileName
                var imageBounds = [[north, west], [south, east]];
                var tegaki = L.imageOverlay(imageUrl, imageBounds).addTo(map);
            })
            //気象データの取得に失敗した場合の処理
            .fail(function(json) {
                alert("気象データの取得に失敗しました。");
            });
        }
    </script>

    <style>
        #map {height: 500px; width: 100%; margin-top:10px;}
    </style>
</head>
```

地図活用のフレームワーク

Ajaxとは、JavaScriptを
効率よく利用するためのフレームワーク

←画面読み込み時に実行

←配布したAPIキーを設定

←地図の種類を設定

←Leafletを設定

←地図の中

修正はココだけ

[0]1時間先
→[5]6時間先

aley=30&bsx=640&bsy=840&proj=3&key=" + api_key

↑APIリクエストを設定

JavaScript

←雨雲画像のファイルパスを取得

←雨雲画像の四隅の緯度経度を取得

←雨雲画像取得URIを設定

←地図に雨雲画像をセット

←地図の表示サイズを設定

APIレスポンス


```
{"info":
 {"201911272100":
  {"cloud":"95.1", "humidity":"82.9", "temperature":"11.9", "weatherForecast":"200", "windDirection":"100.9", "windSpeed":"1.3"},
 "201911272200":
  {"cloud":"96.5", "humidity":"82.9", "temperature":"11.8", "weatherForecast":"200", "windDirection":"103.7", "windSpeed":"1.0"},
 "201911272300":
  {"cloud":"99.1", "humidity":"83.4", "temperature":"11.6", "weatherForecast":"200", "windDirection":"95.3", "windSpeed":"1.4"},
 "201911280000":
  {"cloud":"98.0", "humidity":"88.1", "temperature":"11.4", "weatherForecast":"200", "windDirection":"99.9", "windSpeed":"2.0"},
 "201911280100":
  {"cloud":"96.8", "humidity":"89.0", "temperature":"11.0", "weatherForecast":"300", "windDirection":"95.0", "windSpeed":"2.3"},
 "201911280200":
  {"cloud":"95.5", "humidity":"88.8", "temperature":"10.4", "weatherForecast":"300", "windDirection":"88.6", "windSpeed":"2.5"}
 },
"lx":240, "ly":120, "param":
 {"key":"WX2120-C-3Z-9EU", "lat":"35", "lon":"139", "name":"all", "id":"wimage_service"},
"precipitation":[
  {"dtf":"201911272000-201911272100", "east":142.0, "fileName":"precipitation/shortme/201911272000-00060/0/0120-0080/480-480/000-000/0003-0003.png", "north":36.0, "south":32.0, "value":"0.0", "west":136
  {"dtf":"201911272100-201911272200", "east":142.0, "fileName":"precipitation/shortme/201911272000-00120/0/0120-0080/480-480/000-000/0003-0003.png", "north":36.0, "south":32.0, "value":"0.0", "west":136
  {"dtf":"201911272200-201911272300", "east":142.0, "fileName":"precipitation/shortme/201911272000-00180/0/0120-0080/480-480/000-000/0003-0003.png", "north":36.0, "south":32.0, "value":"0.0", "west":136
  {"dtf":"201911272300-201911280000", "east":142.0, "fileName":"precipitation/shortme/201911272000-00240/0/0120-0080/480-480/000-000/0003-0003.png", "north":36.0, "south":32.0, "value":"0.0", "west":136
  {"dtf":"201911280000-201911280100", "east":142.0, "fileName":"precipitation/shortme/201911272000-00300/0/0120-0080/480-480/000-000/0003-0003.png", "north":36.0, "south":32.0, "value":"0.5", "west":136
  {"dtf":"201911280100-201911280200", "east":142.0, "fileName":"precipitation/shortme/201911272000-00360/0/0120-0080/480-480/000-000/0003-0003.png", "north":36.0, "south":32.0, "value":"1.0", "west":136
 ],
"precipitationEmpiric":[
  {"dtf":"201911272030-201911272035", "east":142.0, "fileName":"precipitation/radar/201911272035-00000/0/0120-0080/480-480/000-000/0003-0003.png", "north":36.0, "south":32.0, "value":"0.0", "west":136.0},
  {"dtf":"201911272035-201911272040", "east":142.0, "fileName":"precipitation/nowcast/201911272035-00005/0/0120-0080/480-480/000-000/0003-0003.png", "north":36.0, "south":32.0, "value":"0.0", "west":136.0},
```


6時間先までの画像情報など
※precipitation属性

←[0] 1時間先
←[1] 2時間先
←[2] 3時間先
←[3] 4時間先
←[4] 5時間先
←[5] 6時間先