

Expert

プラグイン開発





01	kintone プラグインについて	5
02	設定画面の作成	18
03	既存 JS のプラグイン化	35
04	情報の秘匿	42

01 kintone プラグインについて

- 02 設定画面の作成
- 03 既存 JS のプラグイン化
- 04 情報の秘匿

01

kintone プラグインについて



01 kintone プラグインについて

1. プラグインのメリット
2. ハンズオンについて

02 設定画面の作成

03 既存 JS のプラグイン化

04 情報の秘匿

01-1

プラグインのメリット



kintone プラグインとは

JavaScript や CSS ファイルを 1 つにパッケージング

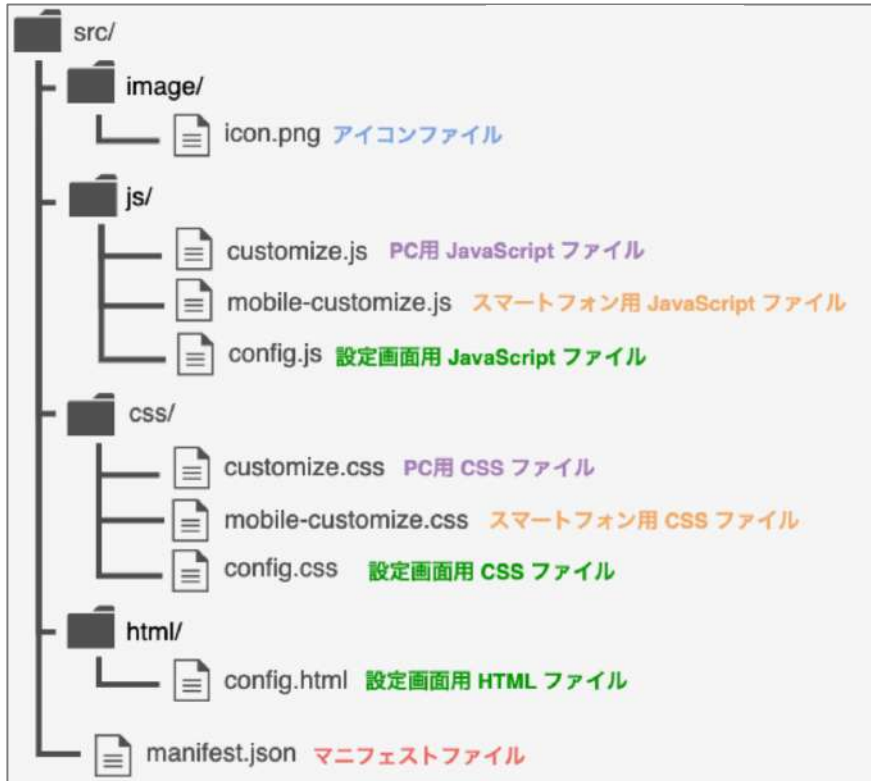
して

複数のアプリ / 環境で利用できるようにしたもの

プラグインに必要なファイル

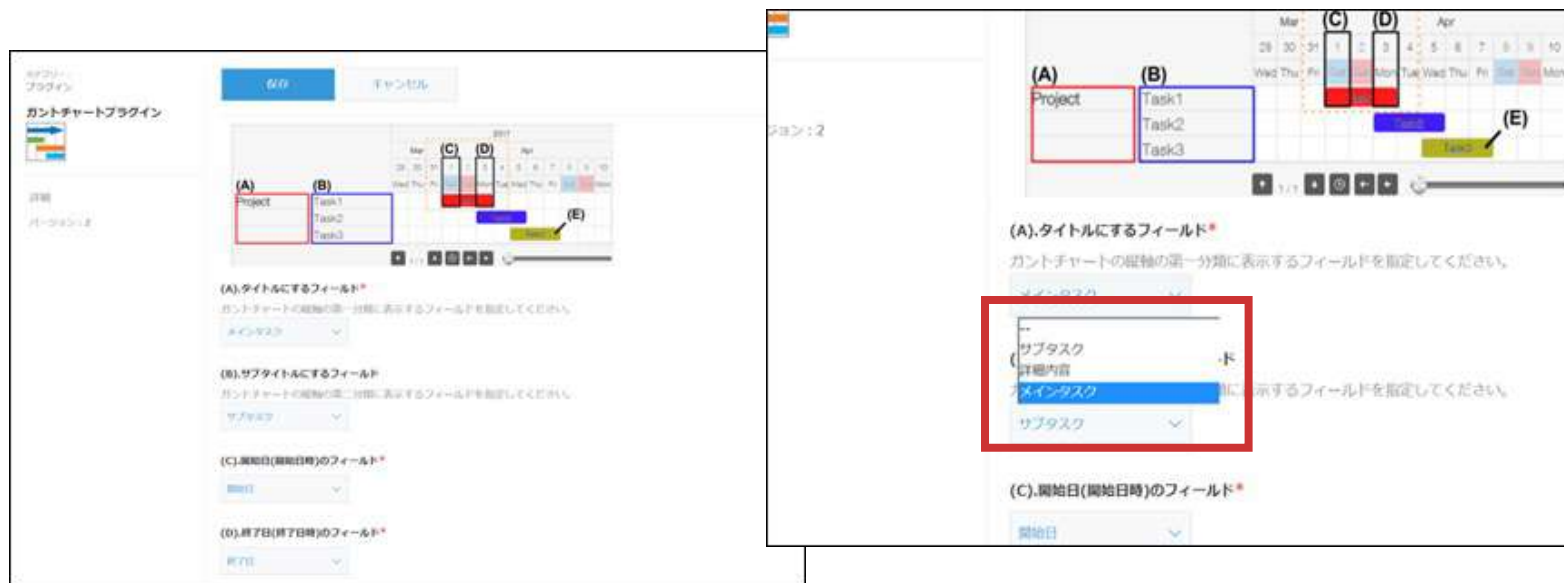
- パッケージングを行うにはマニフェストファイルで指定した構成である必要がある

➤ ファイル構成のサンプル



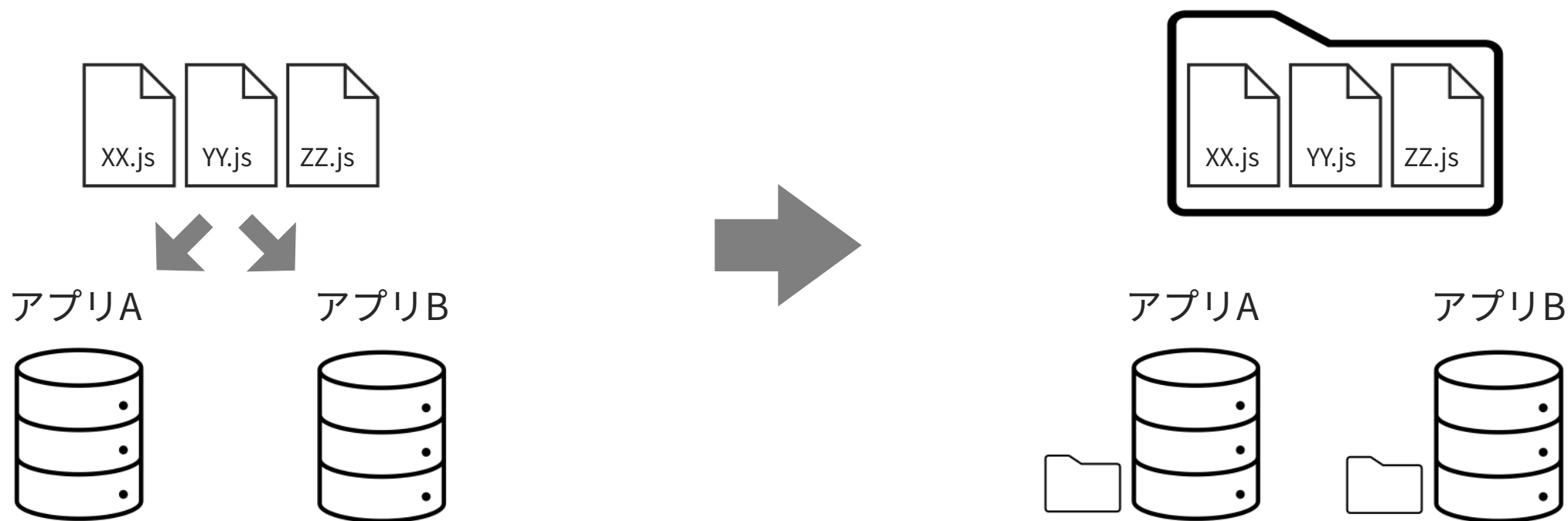
設定画面（GUI）で設定可能

- 通常はアプリに合わせて **JavaScript ファイルを編集する** 必要があるが、プラグインの場合は **画面から設定** することができる
- 個別のカスタマイズによる開発よりもフィールドの追加 / 変更など、設計変更の影響を最小限に抑えることができる



アプリへの適用が容易

通常は複数のカスタマイズファイルを扱う場合、複数アプリへの適用が煩雑であるが、プラグインの場合は**パッケージング**されているため容易に適用することができる

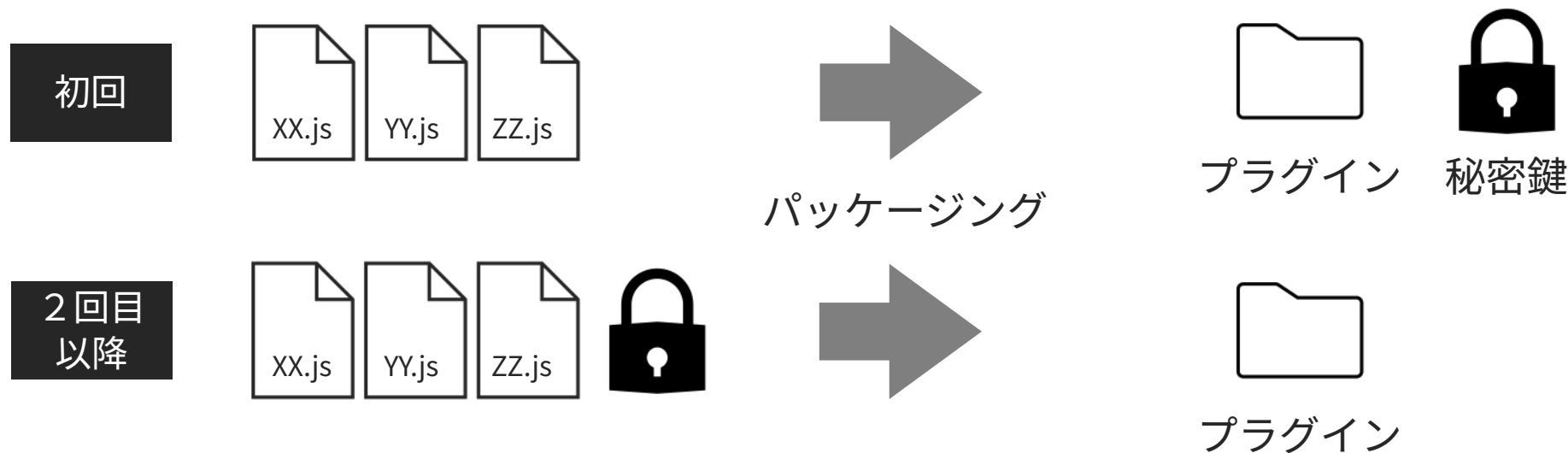


バージョンアップが楽

プラグインの初回パッケージング時に**秘密鍵**が生成される

改修後のパッケージング時に秘密鍵を指定できる

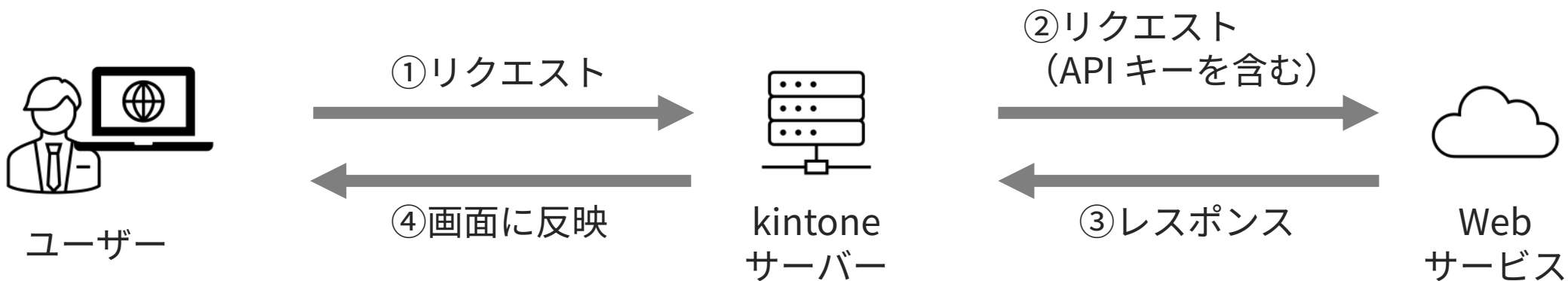
秘密鍵を使わないと、同じプログラムでも別プラグインとして認識される



秘密鍵を指定したプラグインを kintone に再登録すると自動的に同じ秘密鍵のプラグインに上書きされる

情報を秘匿できる

- 外部 API の実行に必要な API キーなどの情報は `kintone.plugin.app.setProxyConfig` を使うと **kintone サーバーに保存** できる
- プラグインに保存した情報は `kintone.plugin.app.proxy` を使って利用する
一般ユーザーからは参照できない（ソースから確認不可）



01 kintone プラグインについて

1. プラグインのメリット
2. ハンズオンについて

02 設定画面の作成

03 既存 JS のプラグイン化

04 情報の秘匿

01-2

ハンズオンについて



ハンズオン内容

➤ 前提

開発者ではないユーザーが設定画面で設定情報を変更できるプラグインを作成する

➤ 想定

テーマ：「有給休暇申請管理」

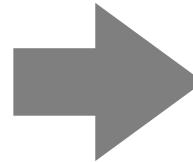
- ・ アプリ① 有給休暇マスタ
- ・ アプリ② 休暇申請 ← JS カスタマイズ適用アプリ

既存の JS カスタマイズをプラグイン化する

既存の JS カスタマイズの動作

プロセスを「未取得」から「取得済」に進める

有給休暇申請



「取得済日数」と「残日数」が更新される

有給休暇マスタ



プラグインの完成形



設定画面から設定情報を変更できる

三

🏠

🔔

★

ポータル > スペース: プラグイン開発勉強会 > アプリ: 有給休暇申請 > アプリの設定 > プラグイン > プラグインの設定

プラグインの設定

サンプルプラグイン

バージョン: 0.1

有給休暇申請プラグイン設定画面

有給休暇マスタのアプリIDを入力してください。

有給休暇マスタのAPIトークンを入力してください。

保存する

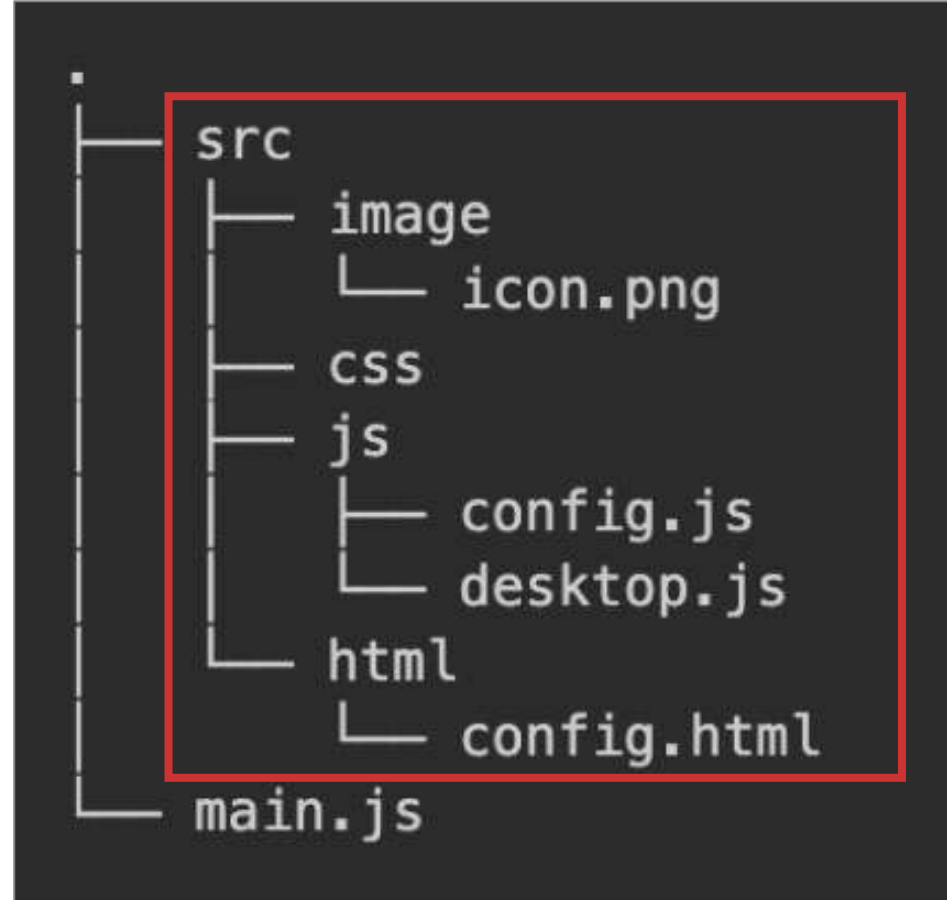
キャンセル

情報を秘匿する

×	ヘッダー	ペイロード	プレビュー	レスポンス	イニシエータ	タイミング	Cookie
▼	クエリ文字列パラメータ	ソースを表示	デコード済みを表示				
	_ref:	https%3A%2F%2F	cybozu.com%2Fk%2F499%2Fshow%23record%3D20%26mode%3Dshow				
▼	リクエストのペイロード	ソースを表示					
	{url: "https://	/k/v1/records.json", method: "PUT", ...}					
	body: {app: "500", ...}						
	headers: {Content-Type: "application/json", X-Cybozu-API-Token: "5eEKvthXqWYnUTlbGvDBSyXhUPcQ9tJGxJqKaF0x"}						
	Content-Type: "application/json"						
	X-Cybozu-API-Token: "5eEKvthXqWYnUTlbGvDBSyXhUPcQ9tJGxJqKaF0x"						
	method: "PUT"						
	url: "https://	cybozu.com/k/v1/records.json"					
	__REQUEST_TOKEN__:	"44d22dba-1a4e-4bb9-88bb-c1959264d794"					

ファイル構成を確認する

- 下記のファイル構成になっていることを確認する



01 kintone プラグインについて

02 設定画面の作成

03 既存 JS のプラグイン化

04 情報の秘匿

02

設定画面の作成



01 kintone プラグインについて

02 設定画面の作成

1. HTML の作成

2. JavaScript の作成

03 既存 JS のプラグイン化

04 情報の秘匿

02-1

HTML の作成



完成イメージ

- JavaScript で操作できるように `input` と `button` タグには `id` を付与する

h2		有給休暇申請プラグイン設定画面
p		有給休暇マスタのアプリIDを入力してください。
input	appld	<input type="text"/>
p		有給休暇マスタのAPIトークンを入力してください。
input	token	<input type="text"/>
button	submit	<input type="button" value="保存する"/>
	cancel	

HTML を作成する

- `./html/config.html` を開き、HTML を書く



`./src/html/config.html`

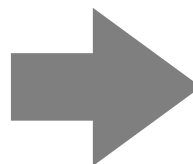
```
<h2>有給休暇申請プラグイン設定画面</h2>
<p>有給休暇マスタアプリのアプリIDを入力してください。</p>
<input type="text" id="appId" />
<p>有給休暇マスタアプリのAPIトークンを入力してください。</p>
<input type="password" id="token" />
<div>
  <button type="button" id="submit">保存する</button>
  <button type="button" id="cancel">キャンセル</button>
</div>
```

HTML の大枠は kintone が準備しているため **body タグの内側**だけで良い

【補足】51-modern-default

- [51-modern-default](#) - kintone のデザインと調和する CSS
上記の CSS ファイルで用意されている class を指定すると以下のような見た目になる

CSS 適用前



CSS 適用後





【補足】 kintone UI Component

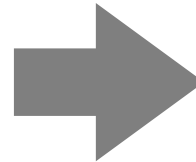
- [kintone UI Component](#) - kintone ライクな UI パーツを簡単に作ることができるライブラリ

通常のボタン



The screenshot shows a standard Kintone form interface. At the top, there is a header bar with a dropdown menu set to '本人', a share icon, a filter icon, a bar chart icon, and a button labeled '一覧のボタン' which is highlighted with a red rectangular border. Below the header is a table with two columns: 'ステータス' and '社員番号'. The table contains two rows of data, both with the status '取得済' and the employee number '002900'.


	ステータス	社員番号
	取得済	002900
	取得済	002900



KUC を使用したボタン



The screenshot shows a Kintone form interface using KUC (Kintone UI Component). The header bar is identical to the previous example, but the button labeled '一覧のボタン' is now a solid blue button with white text, also highlighted with a red rectangular border. The table below is identical to the one in the previous example, with two rows of data: '取得済' and '002900'.

	ステータス	社員番号
	取得済	002900
	取得済	002900

01 kintone プラグインについて

02 設定画面の作成

1. HTML の作成

2. JavaScript の作成

03 既存 JS のプラグイン化

04 情報の秘匿

02-2

JavaScript の作成



設定画面用 JS ファイルを作成する

- `./js/config.js` を開き、JavaScript を書く



`./src/js/config.js`

```
((PLUGIN_ID) => {  
  'use strict';  
  // ここから処理を書いていく  
})(kintone.$PLUGIN_ID);
```

即時関数に `PLUGIN_ID`,
`kintone.$PLUGIN_ID` を入れる

有給休暇申請プラグイン設定画面

有給休暇マスタのアプリIDを入力してください。

有給休暇マスタのAPIトークンを入力してください。

保存する

キャンセル

「保存する」ボタンをクリックしたときに
kintone に設定情報を保存する

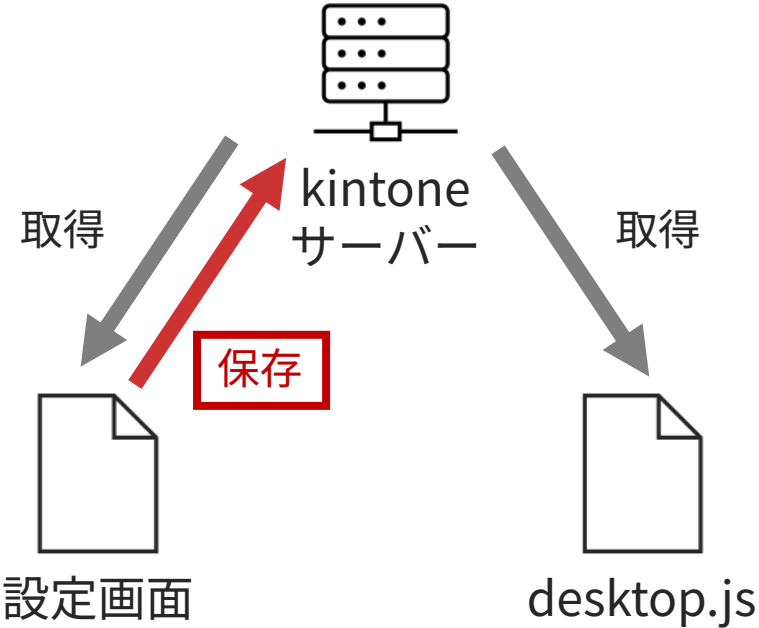
「キャンセル」ボタンをクリックしたときに
前のページに遷移する

プラグインの設定情報を保存する

設定情報を保存する API

```
kintone.plugin.app.setConfig(config, callback)
```

引数	型	値
config	オブジェクト	プラグインに保存する設定 例： <pre>{ "key1": "value1", "key2": "value2" }</pre>
callback	関数	設定の保存が完了した後に実行する関数



ボタンの処理を記述する

- 「保存する」 ボタンをクリックしたときに kintone に設定情報を保存する
- 「キャンセル」 ボタンをクリックしたときに前のページに遷移する

```
./src/js/config.js
```

```
((PLUGIN_ID) => {  
  'use strict';  
  
  // 「保存する」 ボタンをクリックしたときの処理  
  document.getElementById('submit').addEventListener('click', () => {  
    kintone.plugin.app.setConfig({  
      appId: document.getElementById('appId').value  
    });  
  });  
  
  // 「キャンセル」 ボタンをクリックしたときの処理  
  document.getElementById('cancel').addEventListener('click', () => history.back());  
})(kintone.$PLUGIN_ID);
```

既存の設定情報を表示する

- すでに設定されている情報があれば設定画面読み込み時に表示する

有給休暇申請プラグイン設定画面

有給休暇マスタのアプリIDを入力してください。

12345

有給休暇マスタのAPIトークンを入力してください。

保存する

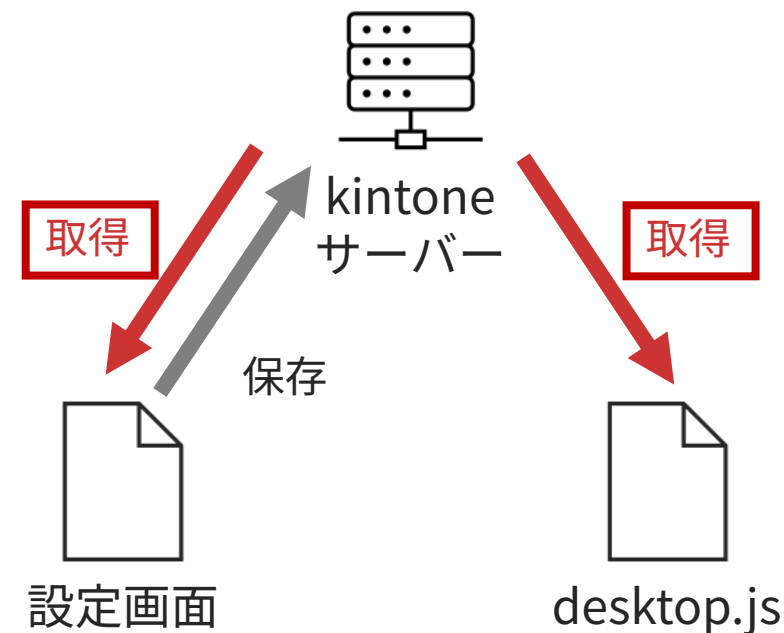
キャンセル

プラグインの設定情報を取得する

設定情報を取得する API

```
kintone.plugin.app.getConfig(pluginId)
```

引数	型	値
pluginId	文字列	設定情報を取得する プラグインの プラグイン ID
戻り値	キーと値を対にしたオブジェクトの形式で プラグインの設定情報	



既存の設定情報を表示する

- `kintone.plugin.app.getConfig` で設定情報を取得する

```
./src/js/config.js
```

```
((PLUGIN_ID) => {  
  'use strict';  
  
  const config = kintone.plugin.app.getConfig(PLUGIN_ID);  
  if (!config) {  
    window.alert('プラグインの設定の読み込みに失敗しました。');  
    window.location.href = `/k/admin/app/${kintone.app.getId()}/plugin/`;  
  }  
  document.getElementById('appId').value = config.appId || '';  
  
  . . .  
})(kintone.$PLUGIN_ID);
```

処理の詳細

- `PLUGIN_ID` に基づき、プラグインの設定情報を取得する

```
const config = kintone.plugin.app.getConfig(PLUGIN_ID);
```

- プラグインの設定情報の取得が失敗した時の処理を書く

```
if (!config) {  
  window.alert('プラグインの設定の読み込みに失敗しました。');  
  window.location.href = `/k/admin/app/${kintone.app.getId()}/plugin/`;  
}
```

- `config` に `appId` が存在する場合は、設定値を画面に設定する

```
document.getElementById('appId').value = config.appId || '';
```

➤ HTML ファイルの作成

- 設定画面を作るために HTML ファイルを作成する

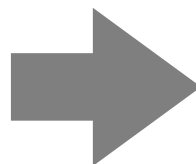
➤ JavaScript ファイルの作成

- 設定画面で入力された情報を読み取るために JavaScript ファイルを作成する
- `kintone.plugin.app.setConfig` で kintone へ保存する
- `kintone.plugin.app.getConfig` で保存された設定情報を取得する

[発展] 設定画面を kintone ライクなデザインにする

- [51-modern-default](#) - kintone のデザインと調和する UI パーツのスタイルシート (CSS)
GitHub から [51-modern-default.css](#) をダウンロードして利用できる

CSS 適用前



CSS 適用前



[発展] 設定画面を kintone ライクなデザインにする

- 51-modern-default 適用後のコード

```
1  <div class="block">
2    <label class="kintoneplugin-label">
3      <span class="container_label">有給申請プラグイン設定画面</span>
4    </label>
5    <div class="kintoneplugin-row"></div>
6    <div class="kintoneplugin-row">有給休暇マスタアプリのアプリID入力してください。</div>
7    <div class="kintoneplugin-input-outer">
8      <input id="appId" class="kintoneplugin-input-text" type="text" />
9    </div>
10   <div class="kintoneplugin-row">有給休暇マスタアプリのAPIトークンを入力してください。</div>
11   <div class="kintoneplugin-input-outer">
12     <input id="token" class="kintoneplugin-input-text" type="text" />
13   </div>
14   <div class="kintoneplugin-row"></div>
15 </div>
16 <div class="block">
17   <button type="button" id="submit" class="kintoneplugin-button-dialog-ok">保存する</button>
18   <button type="button" id="cancel" class="kintoneplugin-button-dialog-cancel">キャンセル</button>
19 </div>
```

01 kintone プラグインについて

02 設定画面の作成

03 既存 JS のプラグイン化

04 情報の秘匿

03

既存 JS のプラグイン化



設定情報の利用

- `main.js` のコードを `desktop.js` に貼り付ける
- 即時関数に `PLUGIN_ID`, `kintone.$PLUGIN_ID` を入れる
- `config.js` で設定した情報を、設定情報を取得する API を利用して `desktop.js` で受け取る



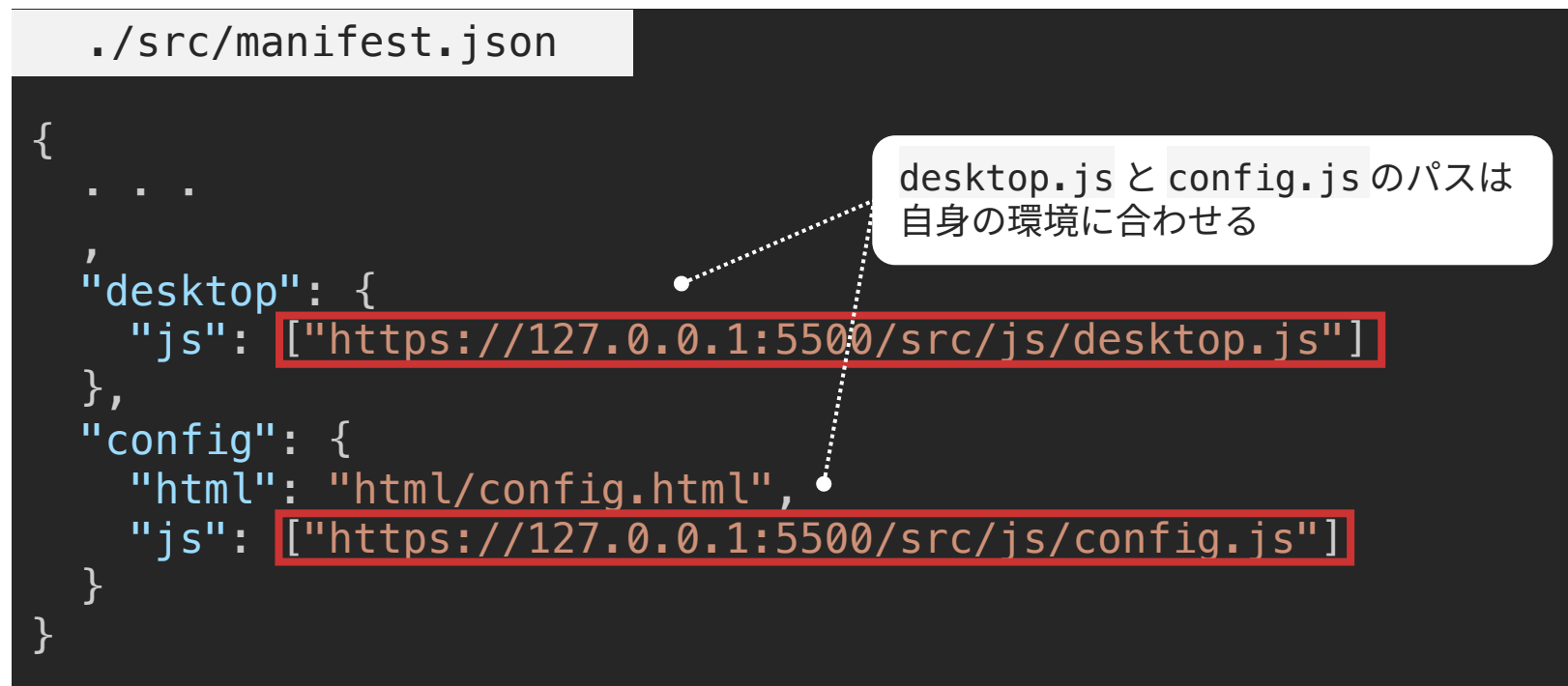
```
./src/js/desktop.js
```

```
((PLUGIN_ID) => {  
  'use strict';  
  
  const config = kintone.plugin.app.getConfig(PLUGIN_ID);  
  const HOLIDAY_MANAGEMENT_APP_ID = config.appId;  
  . . .  
})(kintone.$PLUGIN_ID);
```

マニフェストの作成

➤ マニフェストファイルとは

- プラグイン作成に必要なファイル情報をまとめた設定ファイルで kintone プラグイン作成に必須
- マニフェストファイル (manifest.json) をダウンロードして src フォルダ直下に移動する



パッケージング

- パッケージングとは、プラグインに必要なファイルを ZIP ファイルにまとめること
- 今回は **Web 版の plugin-packer** でパッケージングする

src フォルダを左側のエリアへドラッグアンドドロップする



「作成する」をクリックする



※ 2回目以降のパッケージングの場合は、秘密鍵ファイルを右側のエリアへドラッグアンドドロップする
<https://plugin-packer.kintone.dev/index-ja.html>

kintone への適用

- プラグインを kintone に適用し、プラグインの設定画面でアプリ ID を入力する
- JS カスタマイズの動作と同じ処理がプラグインでできていることを確認する

プロセスを「未取得」から「取得済」に進める



有給休暇申請

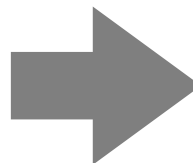


種類	取得日	状態	取得日数
有給休暇	2024-03-28	終日休	1 日

残有給日数: 20

休暇日数合計: 1 日

申請者: guest



「取得済日数」と「残日数」が更新される



有給休暇マスタ



社員番号	氏名	ふりがな	付与日数	取得済日数	残日数
0002	テスト	てすと	20	0	20

残日数: 19

03 既存 JS のプラグイン化 > まとめ



➤ 設定情報の利用

- `kintone.plugin.app.setConfig` で設定した情報を `kintone.plugin.app.getConfig` で受け取る

➤ パッケージング

- Web 版の plugin-packer で zip 化する

➤ kintone への適用

- zip 化したファイルを kintone へアップロードし、設定画面でアプリ ID を入力して適用
- JS カスタマイズの動作と同じ処理がプラグインでできていることを確認

01 kintone プラグインについて

02 設定画面の作成

03 既存 JS のプラグイン化

04 情報の秘匿

04

情報の秘匿



01 kintone プラグインについて

02 設定画面の作成

03 既存 JS のプラグイン化

04 情報の秘匿

1. アクセス権の設定
2. 秘匿情報の利用
3. 設定済み秘匿情報の表示

04-1

アクセス権の設定



ここまでの課題

- 申請者が「有給休暇マスタ」アプリの編集権限を持っており、休暇の付与日数を編集できてしまう

「有給休暇マスタ」アプリのアクセス権

アプリのアクセス権

[?ヘルプ](#)[📖 便利に使うガイドブック vol.08 アクセス権編](#)

アプリを利用できるユーザーを指定・制限することや、所属するアプリグループを変更することができます。

レコードの操作（閲覧、追加、編集、削除など）の制限だけでなく、アプリの設定を変更可能なユーザーの指定（アプリ管理権限の付与）もこの画面から行います。

アプリグループ

Public

ユーザー/組織/グループとアクセス権



ユーザー/組織/グループ	許可する操作	
 アプリ作成者（徳永剛）	<input checked="" type="checkbox"/> レコード閲覧 <input checked="" type="checkbox"/> レコード追加 <input checked="" type="checkbox"/> レコード編集 <input checked="" type="checkbox"/> レコード削除 <input checked="" type="checkbox"/> アプリ管理 <input checked="" type="checkbox"/> ファイル読み込み <input checked="" type="checkbox"/> ファイル書き出し	
 Everyone	<input checked="" type="checkbox"/> レコード閲覧 <input checked="" type="checkbox"/> レコード追加 <input checked="" type="checkbox"/> レコード編集 <input checked="" type="checkbox"/> レコード削除 <input type="checkbox"/> アプリ管理 <input type="checkbox"/> ファイル読み込み <input type="checkbox"/> ファイル書き出し	

プラグイン化と同時に適切な権限を設定する

アプリのアクセス権を変更する

- 「有給休暇マスタ」アプリの 設定 > 「アクセス権」 > 「アプリ」 から次の設定を変更する

「アプリ作成者」と「Everyone」の不要な権限を外す

アプリのアクセス権 [?ヘルプ](#) [便利に使うガイドブック vol.08 アクセス権](#)

アプリを利用できるユーザーを指定・制限することや、所属するアプリグループを変更することができます。
レコードの操作（閲覧、追加、編集、削除など）の制限だけでなく、アプリの設定を変更可能なユーザーの指定（アプリ管理権限の付与）もこの画面から行います。

アプリグループ

Public

ユーザー/組織/グループとアクセス権 [?](#)

ユーザーを追加

ユーザー/組織/グループ	許可する操作						
 アプリ作成者	<input checked="" type="checkbox"/> レコード閲覧	<input checked="" type="checkbox"/> レコード追加	<input type="checkbox"/> レコード編集	<input checked="" type="checkbox"/> レコード削除	<input checked="" type="checkbox"/> アプリ管理	<input checked="" type="checkbox"/> ファイル読み込み	<input checked="" type="checkbox"/> ファイル書き出し
 Everyone	<input checked="" type="checkbox"/> レコード閲覧	<input type="checkbox"/> レコード追加	<input type="checkbox"/> レコード編集	<input type="checkbox"/> レコード削除	<input type="checkbox"/> アプリ管理	<input type="checkbox"/> ファイル読み込み	<input type="checkbox"/> ファイル書き出し

アクセス権が適切に設定されたことを確認する

- Everyone に編集権限がない状態でプロセスを進めてみる
- 申請者は「有給休暇マスタ」アプリの編集権限がないためエラーになる

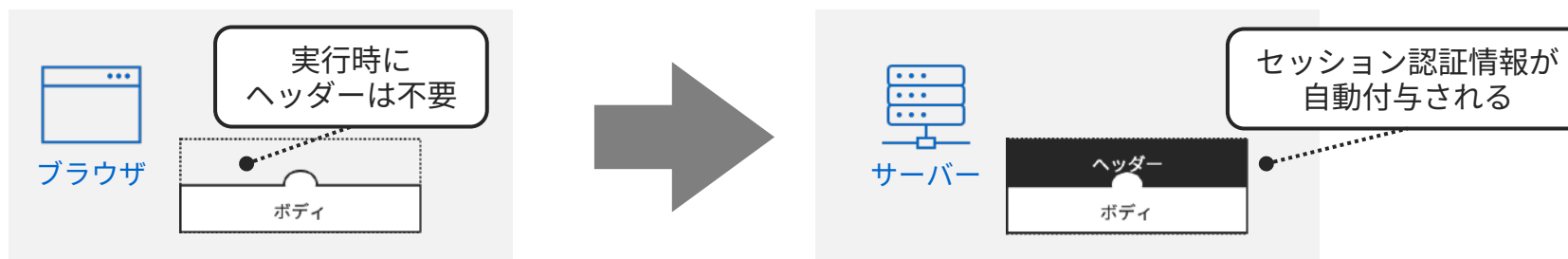
コンソールに権限エラーが出る

```
✖ ▶ PUT https://gon.cybozu.com/k/v1/records.json show.js:1128 ⓘ  
403 (Forbidden)  
  
download.do?app=371&...ffa0780d2748263f:42  
▼ {code: 'CB_NO02', id: '2c7GR8pTgflDBz4jNVmx', message: '権限が  
ありません。'} ⓘ  
  code: "CB_NO02"  
  id: "2c7GR8pTgflDBz4jNVmx"  
  message: "権限がありません."  
  ▶ [[Prototype]]: Object
```

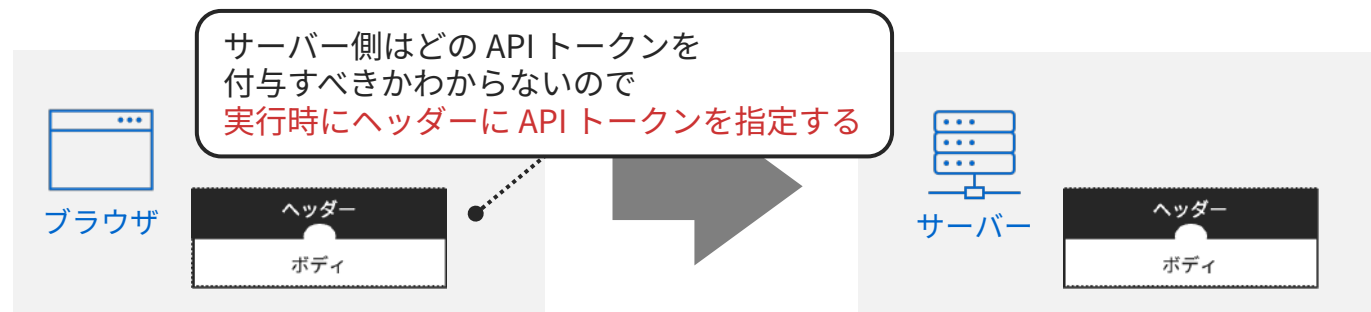
ユーザー権限に依存しない API トークン認証

- セッション認証はユーザーの権限に依存するため、API トークン認証を利用する必要がある

➤ セッション認証を利用したリクエスト ユーザーの権限に基づいて実行の可否が決まる



➤ API トークン認証を利用したリクエスト ユーザーの権限に依存せずに REST API を実行できる



API トークン認証の注意点

- API トークンをコードに記載するとブラウザ上で確認できてしまいセキュリティ上の問題がある



今回必要なヘッダーのうち、Content-Type は見えても良いが X-Cybozu-API-Token は見えると困る

リクエストにボディが必要な場合は Content-Type が必須
参考：kintone REST API の共通仕様
<https://cybozu.dev/ja/kintone/docs/rest-api/overview/kintone-rest-api-overview/>

プラグインの **情報を秘匿する機能** で API トークンを隠して REST API を実行する

01 kintone プラグインについて

02 設定画面の作成

03 既存 JS のプラグイン化

04 情報の秘匿

1. アクセス権の設定
2. 秘匿情報の利用
3. 設定済み秘匿情報の表示

04-2

秘匿情報の利用



- 秘匿したい情報を保存するには `setProxyConfig`、
- 保存した情報を利用して REST API を実行するには `proxy`

➤ kintone.plugin.app.proxy を実行したときの流れ





秘匿情報を保存する

秘匿情報を保存する API

kintone.plugin.app.setProxyConfig(url, method, headers, data, successCallback)

引数	型	説明
url	文字列	実行する API の URL
method	文字列	HTTP メソッド (GET, POST, PUT, DELETE)
headers	オブジェクト	API のリクエストヘッダーに加えるパラメーター (※ 条件あり)
data	オブジェクト	API のリクエストデータに加えるリクエストボディ (※ 条件あり)
successCallback	関数	保存が完了したときに実行するコールバック関数

秘匿情報を保存する

- 設定画面で動作する config.js で、setProxyConfig を使って情報を保存する
- コールバック関数でアプリ ID を保存する setConfig を実行する

```
./src/js/config.js
```

```
// 「保存する」ボタンをクリックしたときの処理
document.getElementById('submit').addEventListener('click', () => {
  kintone.plugin.app.setProxyConfig(
    kintone.api.url('/k/v1/record.json'),
    'PUT',
    { 'X-Cybozu-API-Token': document.getElementById('token').value },
    {},
    () => {
      kintone.plugin.app.setConfig({
        appId: document.getElementById('appId').value,
      });
    }
  );
});
```

秘匿情報を利用する



秘匿情報を利用する API

kintone.plugin.app.proxy(pluginId, url, method, headers, data, successCallback, failureCallback)

引数	型	説明
pluginId	文字列	API を実行するプラグインの プラグイン ID
url	文字列	実行する API の URL
method	文字列	HTTP メソッド (GET, POST, PUT, DELETE)
headers	オブジェクト	リクエストヘッダー
data	オブジェクト	リクエストボディ
successCallback	関数	リクエストが完了したあとに実行するコールバック関数
failureCallback	関数	リクエストが失敗したときに実行するコールバック関数

秘匿情報を利用する

- desktop.js で proxy を使って、情報を秘匿したまま REST API を実行する

```
./src/js/desktop.js
```

```
await kintone.plugin.app.proxy(  
  PLUGIN_ID,  
  kintone.api.url('/k/v1/record.json'),  
  'PUT',  
  { 'Content-Type': 'application/json' }, // APIトークンを書かない  
  {  
    app: HOLIDAY_MANAGEMENT_APP_ID,  
    updateKey: {  
      // 変更はないため省略  
    },  
    record: {  
      // 変更はないため省略  
    },  
  }  
);
```

設定画面で API トークンを設定する

- API トークンを生成してプラグイン設定画面で保存する

レコード閲覧・編集権限を持つ API トークンを生成する



有給休暇マスタ

ポータル > スペース: プラグイン開発勉強会 > アプリ: 有給休暇マスタ > アプリの設定 > APIトークン

APIトークン [ヘルプ](#)

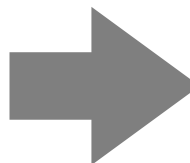
外部のプログラムからREST APIを呼び出す際に使用する認証情報（APIトークン）を管理します。
この画面でAPIトークンを生成し、REST APIのリクエストヘッダに付与することで、REST APIを実行することができます。
例: curl -H "X-Cybozu-API-Token: YOUR_TOKEN" "https://cy-002900.cybozu.com/k/v1/record.json?app=118&id=1"

生成する

APIトークン	アクセス権
K3V7IKZ9VcQYNqrFq1U0wS1lQrj0V6gi80qxZ79u	<input checked="" type="checkbox"/> レコード閲覧 <input type="checkbox"/> レコード追加 <input checked="" type="checkbox"/> レコード編集 <input type="checkbox"/> レコード削除 <input type="checkbox"/> アプリ管理

kintone開発者向けドキュメント

kintone開発者向けサイト



プラグインの設定画面で API トークンを入力する



有給休暇申請

ポータル > スペース: プラグイン開発勉強会 > アプリ: 有給休暇申請 > アプリの設定 > プラグイン > プラグインの設定

プラグインの設定

サンプルプラグイン

有給休暇申請プラグイン設定画面

有給休暇マスタのアプリIDを入力してください。

有給休暇マスタのAPIトークンを入力してください。

保存する キャンセル

動作確認

- これまでと同じ処理ができていることを確認する

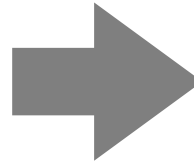
プロセスを「未取得」から「取得済」に進める

有給休暇申請



種別	取得日	動員	取得日数
有給休暇	2024-03-28	終日休	1 日

休暇日数合計: 1 日



「取得済日数」と「残日数」が更新される

有給休暇マスタ



社員番号	氏名	ふりがな
0002	テスト	てすと

付与日数	取得済日数	残日数
20	1	19

動作確認

- 開発者ツールの「ネットワーク」でヘッダーに API トークンが含まれていないことを確認できる



01 kintone プラグインについて

02 設定画面の作成

03 既存 JS のプラグイン化

04 情報の秘匿

1. アクセス権の設定
2. 秘匿情報の利用
3. 設定済み秘匿情報の表示

04-3

設定済み秘匿情報の表示



現状の課題

- 設定画面を開いたときに設定済みの API トークンが表示されず設定済みかどうか分からない

有給申請プラグイン設定画面

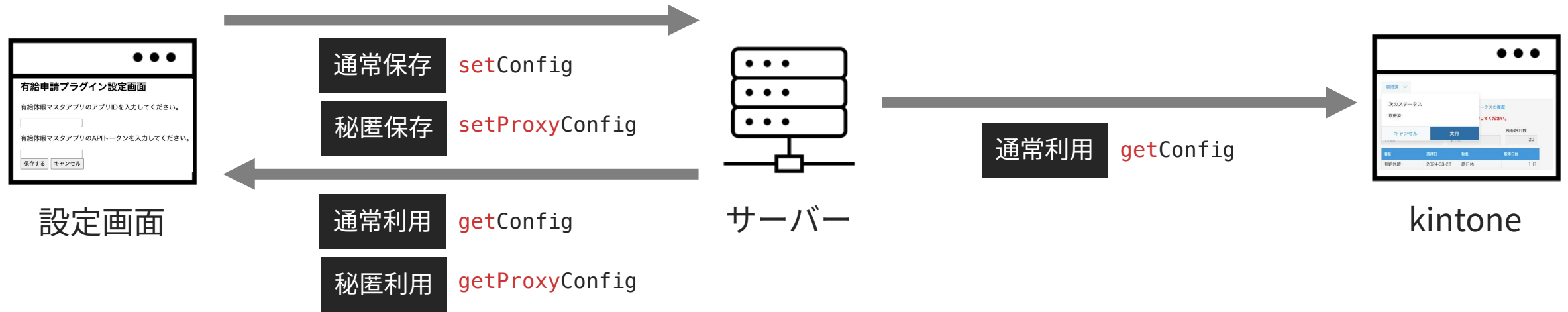
有給休暇マスタアプリのアプリIDを入力してください。

有給休暇マスタアプリのAPIトークンを入力してください。

保存するキャンセル

`kintone.plugin.app.getProxyConfig` を利用して設定済み情報を取得する

設定情報を保存・利用するメソッド



画面によって使えるメソッドが異なる



秘匿情報を取得する

秘匿情報を取得する API `kintone.plugin.app.getProxyConfig(url, method)`

引数	型	説明
url	文字列	実行する API の URL
method	文字列	HTTP メソッド (GET, POST, PUT, DELETE)

戻り値	型	説明
headers	オブジェクト	setProxyConfig の headers に指定したリクエストヘッダー
data	オブジェクト	setProxyConfig の data に指定したリクエストボディ

キーとバリューを対にしたオブジェクトの形式で返却される

秘匿情報を取得する

- 設定画面で動作する config.js で、getProxyConfig を使って情報を取得する
- 設定情報がある場合は API トークンを表示する

```
./src/js/config.js
```

```
document.getElementById('appId').value = config.appId || '';  
  
const proxyConfig = kintone.plugin.app.getProxyConfig(  
  kintone.api.url('/k/v1/record.json'),  
  'PUT'  
);  
  
document.getElementById('token').value = proxyConfig  
  ? proxyConfig.headers['X-Cybozu-API-Token']  
  : '';  
  
document.getElementById('submit').addEventListener('click', async () => { ... });
```

➤ 情報を秘匿する

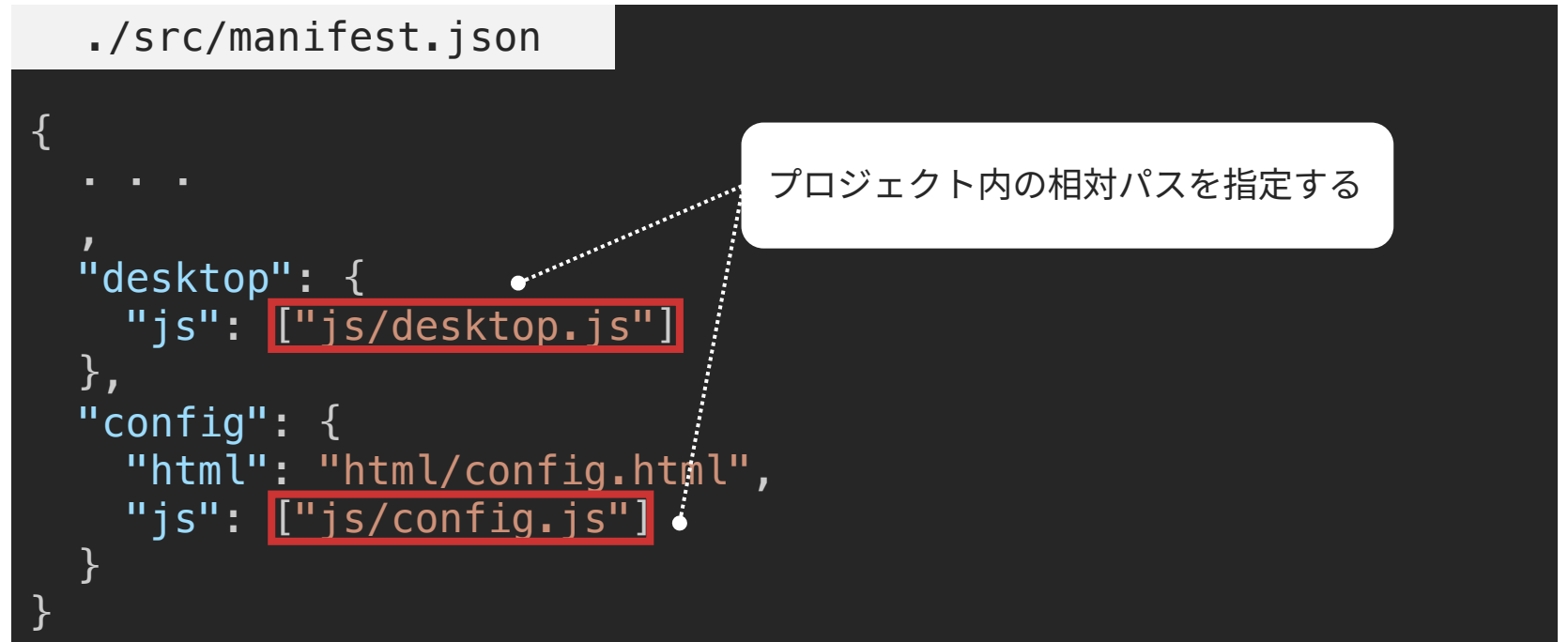
- `setProxyConfig` を使うことで API トークンなどの重要情報を秘匿することができる

➤ 情報を秘匿したまま利用する

- `proxy` を使うことで API トークンなどの重要情報を秘匿したまま REST API を実行できる

[発展] 本番用プラグインを kintone に適用する

- 本番用のプラグインには Live Server のパスではなく
パッケージ内のファイルパスを指定する必要がある
- `./manifest.json` の `config.js` と `desktop.js` のパスを書き換える



[発展] 本番用プラグインを kintone に適用する

- `manifest.json` を書き換えたため再度パッケージングが必要となる
- パッケージングしたプラグインを kintone に再度適用する

秘密鍵を利用して再度パッケージングを行う

Package your kintone plug-in!

プラグインディレクトリをアップロードしてください。
Chrome, Firefox以外の場合は、Zipで固めてアップロードしてください。 ファイルが選択されていません
秘密鍵と署名済みのプラグインファイルを生成します。
2回目以降の場合は、秘密鍵も添付してください。

🟢 ☐ プラグインをドラッグアンドドロップするか、ここをクリックして選択してください

📁 plugin

🔑 秘密鍵(.ppk)ファイルをドラッグアンドドロップするか、ここをクリックして選択してください (オプション)

📁 ...

作成するリセットする

kintone に適用する

プラグイン [? ヘルプ](#)

プラグインの読み込みや管理を行うことができます。
プラグインを利用すると、kintoneで「できること」が広がります。
ここでプラグインを読み込んだ後、各アプリのアプリ設定にある「プラグイン」画面から

プラグインの一覧 [CSV形式でダウンロードする](#)

読み込んだプラグイン

プラグイン名	説明
 サンプルプラグイン バージョン: 0.1	devcampのサンプルプラグインです。

05

おわりに



- create-plugin
 - プラグインの雛形を作るツール（開発未経験者/1から作りたい人向け）
 - 下記の plugin-packer と plugin-uploader も含まれる
- plugin-packer ← 今回はこれの [Web](#) 版を利用
 - zip ファイルにパッケージングするツール（開発経験者/既存のファイルがある人向け）
- plugin-uploader
 - zip ファイルを環境にアップロードするツール（アップロードを自動化したい人向け）
- webpack-plugin-kintone-plugin
 - Webpack を利用したプラグイン開発をサポートするツール（Webpack を使ってプラグインファイルをパッケージングしたい人向け）

参考 URL : <https://developer.cybozu.io/hc/ja/articles/360000975763>

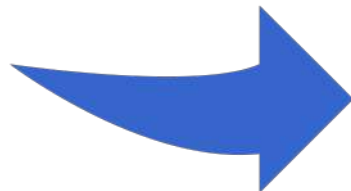
- プラグイン開発 Tips
 - <https://cybozu.dev/ja/kintone/tips/development/plugins/development-plugin/>
- kintone ライクな UI
 - 51-modern-default.css
 - <https://cybozu.dev/ja/kintone/sdk/library/plugin-stylesheet-guide/>
 - kintone UI Component
 - <https://ui-component.kintone.dev/ja/>



kintone CERTIFIED
Customization SP 2024

これって実現できる？

調べないと
答えられない…



その場ですぐに
答えられる！



**カスタマイズ
スペシャリスト**

に合格して

案件対応力 をアップしよう！

✓ 対策動画 年内公開予定！