



Adobe Common Extensibility Platform(CEPエクステンション)につい 7

Created by 10 A on Nov 9, 2017 10:35 PM. Last modified by 10 A on Nov 12, 2017 3:32 PM.

Web開発スキルを活用してAdobe CC アプリケーションを拡張する

Adobe Common Extensibility Platform(CEP)は、HTML5、CSS3、およびJavaScriptの標準的なWeb技術を使 用して、Adobe CCアプリケーションを拡張するエクステンションを開発することが出来ます。もう、API にアクセスするためにC++やプラットフォーム固有のスクリプティングツールを学ぶ必要はありません。 また、CEPと統合されたAdobe Exchange(https://exchange.adobe.com/addons 🗗)ポータルサイトを利用し て、CCアプリケーション用エクステンションを販売し、配布することも可能です。

CEF (Chrome Embeded Framework) とNode.jsが組み込まれているため、CEPでは一般的なWebプラット フォームを活用して、各アプリのスクリプティング・オブジェクトモデルにアクセスし、CCア プリケー ション用エクステンションを構築できます。 詳しい情報及びサンプル・エクステンションは以下の公式 gitを参照して下さい。

Adobe CEP公式リポジトリ(英語)

Adobe CEP · GitHub

CEP概要 (英語)

Creative Cloud Extension SDK - CC extension resources | Adobe I/O 2

また、以下のフォーラム(英語)も役に立つでしょう。

② Extensions / Add-ons Development

CEPエクステンションの特徴

まずCEPエクステンションの特徴を見てみます。

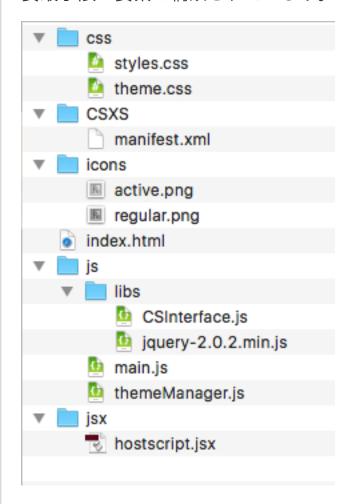
- HTML5+CSS3+Javascriptでユーザーインターフェース(エクステンションパネル)を構成できる。
- UIにはCEFが利用されている為、jQueryやAngularJS等の各種JavaScriptフレームワークに加え、ブラウ ザー側の実装であるCookieやLocal Storageといったものも利用できる。
- CC 2014以降のバージョンではnode.jsが搭載されている為、NPMモジュールが利用出来る。
- アプリケーションのオブジェクトモデルを直接操作するのはExtendscriptで、2つの異なった JavascriptエンジンをつなぐのはCSInterfaceクラスに内蔵されたev alScriptメソッドだけである。

エクステンションの種類

一般的な形態はパネルと呼ばれるもので、CCアプリケーション上で、他のパネルとドッキングしたりア イコン化して収納しておける機能を有します。その他モーダルダイアログ、モードレスダイアログ、イ ンビジブル、エンベッデッドがあります。各CCアプリケーションによってサポートする種類が異なりま すので前述の公式リソース文書にて確認して下さい。

エクステンションの構造

以下に示すのはテンプレートプロジェクトのファイルツリーです。このテンプレートプロジェクトは必要最小限の要素で構成されています。



パネルのUIを構成するのはindex.htmlであり、このファイルがmainPathとしてCSXSフォルダ内のmanifest.xml内で指定されます。index.htmlは通常のHTMLファイルの構成と同様で、CEPインフラ上では内蔵するCEFがHTMLのレンダリングを担当します。利用されるJavascriptエンジンはV8であり動作的にはChro meだと考えて差し支えありません。通常のブラウザで利用可能なCookieやLocal Storage、そしてIndexed DB等も正常に動作します。また、CSXSインターフェースクラスにいくつかのイベントが設定されています。これらで幾つかのアプリケーションの状態を監視することが出来ます。

しかし、基本的にCEP側のインフラからは直接アプリケーション・オブジェクトモデルへのアクセスは出来ません。CEPインフラから各アプリケーション・オブジェクトモデルへのアクセスは evalScriptという CSInterface クラスに用意されたメソッドを介してのアクセスのみが利用可能だという事に注意が必要です。

多くのCCアプリケーションでは、アプリケーション・オブジェクトモデル側はExtendScriptを利用しますので、従来から利用されていたスクリプト関連のリソースをそのまま利用することが出来ます。また、プラグイン経由でイベントを発行させるなど、ネイティブ系からダイレクトにCEPへとアクセスさせることも可能です。

開発環境

CEPエクステンションの開発は一般的なWeb開発に準じたものなので様々な開発ツールが利用可能ですが、CEPエクステンション開発に便利な機能拡張が提供されている **Brackets**がよく利用されています。CC Extension Builder for BracketsとPackage Current Projectというエクステンション開発用機能拡張を組み込んだ例を以下に示します。

```
🐞 Brackets ファイル 編集 検索 表示 ナビゲート Emmet ウィンドウ デバッグ ヘルプ Package Current Project CC Extension Builder
                                                                   ar csInterface = new CSInterface();
                                                                 function init(){
   //alert("init panel...");
                                                                        //alert("init panel...");
csInterface.addEventListener("com.adobe.csxs.events.Switchworkspace", Switchworkspace);
if (window.localStorage){
   if (localStorage.getItem('wsSwitcher')==undefined||localStorage.getItem('wsSwitcher')=="undefined") {
        localStorage.setItem('wsSwitcher', true);
        csInterface.evalScript('pref()');
        .
                                                                        var menuXML
                                                                                               = '<Menu><MenuItem Id="menuItemId1" Label="preference" Enabled="true" Checked="false"/></Menu>';
                                                                      csInterface.addEventListener("com.adobe.csxs.events.flyoutMenuClicked", setPanelCallback);
csInterface.addEventListener("getPref", function(evt) {
    if (evt.data!=""){
                                                                                       var wsn = y1
var wsn = evt.data.split(",");
for (var i=0;i<wsn.length;i++) localStorage.setItem('wsSwitcher_wsn'+i, wsn[i]);</pre>
                                                       function setPanelCallback() {
    var str = [];
    for (var i=0;i<9;i++) str.push(localStorage.getItem('wsSwitcher_wsn'+i));
    csInterface.evalScript("pref('" + str.join(',') + "')");</pre>
                                                              aiwsName = "";
                                                        33 ▼ function Switchworkspace(ev) {
                                                               workspace1 art work edit
                                                               workspace2 text edit
                                                              //alert(ev.data);
//alert($("input#registerEvent").prop("checked"));
                                                                       if (!document.getElementById("registerEvent").checked) return;
                                                                     if (!document.getElementById("registerEvent").c
var toolName = [];
toolName["Adobe Type Tool"] = 2;
toolName["Adobe Area Type Tool"] = 2;
toolName["Adobe Path Type Tool"] = 2;
toolName["Adobe Vertical Type Tool"] = 2;
toolName["Adobe Vertical Area Type Tool"] = 2
toolName["Adobe Vertical Path Type Tool"] = 2;
toolName["Adobe Touch Type Tool"] = 2;
                                                                       toolName["Adobe Right-To-Left Type Tool"] = 2;
toolName["Adobe Right-To-Left Area Type Tool"] = 2;
toolName["Adobe Right-To-Left Path Type Tool"] = 2;
                                                                       toolName["Adobe Rectangle Tool"] = 3;
toolName["Adobe Rounded Rectangle Tool"] = 3;
toolName["Adobe Ellipse Tool"] = 3;
                                                                      toolName["Adobe Select Tool"] = 0;
toolName["Adobe Toggle Select Tool"] = 0
toolName["Adobe Direct Select Tool"] = 0;
                                                    行1,列1-185行
                                                                                                                                                                                                                                                                         INS
                                                                                                                                                                                                                                                                                   JavaScript ▼ 🔼
```

デバッグモード

CCアプリケーションでは通常、署名されていないエクステンションの動作は許可されていません。しかし、エクステンション開発時のデバッグ作業の為にこの制限を回避することが出来ます。これは PlayerDebugModeと呼ばれるものです。

設定方法

- ●Macの場合/ターミナルから以下のコマンドを実行(CC2018リリースの場合) defaults write ~/Library/Preferences/com.adobe.CSXS.8.plist PlayerDebugMode 1
- ●Windowsの場合/コマンドプロンプトから以下のコマンドを実行(CC2018リリースの場合) reg add HKEY_CURRENT_USER\SOFTWARE\Adobe\CSXS.8 /t REG_SZ /v PlayerDebugMode /d 1 /f

*設定する必要のある値はターゲットとするCCアプリケーションのバージョンによって異なります。必要なCSXSバージョンを下に示すリンクを参照し適宜書き換えて実行して下さい。

■ Quick reference of CSSDK and CEP supported application version

2515 Views 🔲 Categories: 🐙 Tags : cep, エクステンション, adobe exchange, playerdebugmode, adobe addons