

Grid Application テンプレートを 紐解く

ググるよりも ググられたい

注意!!

あくまでも Consumer Preview でのお話です。
製品版では変わる可能性があります。
というか、むしろ変わってほしい所もあります。

- ソース読む
- 源典あたる
- 中を漁る
- 試す
- 公開する

新しいプロジェクト

用したファイル

インストール済み

スタート

Visual Basic

Visual C#

Visual C++

Visual F#

ASP.NET Server

ASP.NET Switch

JavaScript

Windows Metro style

他のプロジェクトの種類

リンク プロジェクト

ト プロジェクト

JavaScript

Visual Basic

Visual C#

.NET Framework 4.5

並べ替え基準: 既定



Blank Application

JavaScript



Grid Application

JavaScript



Split Application

JavaScript



Fixed Layout Application

JavaScript



Navigation Application

JavaScript



Notifications + Access Control Application

JavaScript

インストール済み テンプレートの

種類: JavaScript

A multi-page project for Metro style app that navigates among groups of items. The pages display group and details.



Application1

C:\Work\Temp

新しいソリューションを作成する

Application1

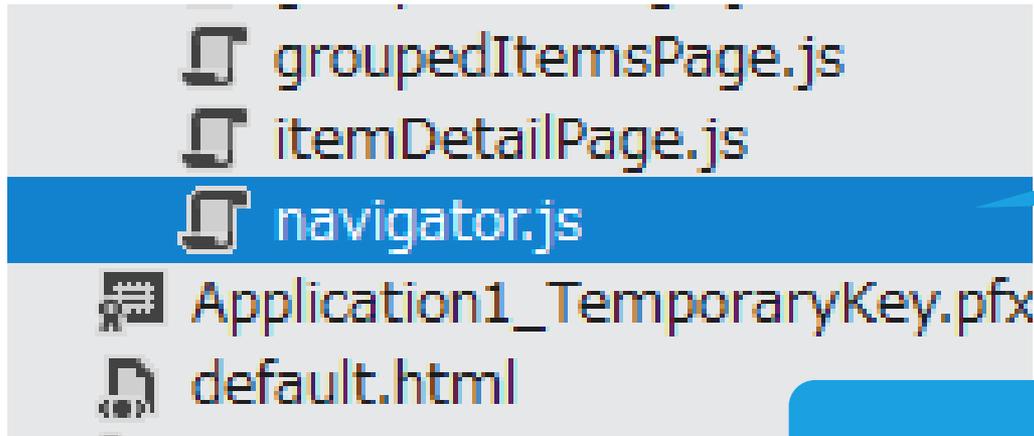
参照(B)...

ソリューションのディレクトリを作成

ソース管理に追加(U)

OK

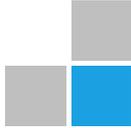
navigator.js?



なんかある...

Application1.PageControlNavigator

```
WinJS.Namespace.define("Application1", {
  PageControlNavigator: WinJS.Class.define(
    function (element, options) ..., {
      _createPageElement: function () ...,
      _keypressHandler: function (eventObject) ...,
      _keyupHandler: function (eventObject) ...,
    }
  )
});
```



PageControlNavigatorが呼ばれる場所は？

```
<div id="contenthost"  
  data-win-control="Application1.PageControlNavigator"  
  data-win-options="{home: '/html/groupedItemsPage.html'}"></div>
```

default.htmlで
使われてる

デバッグ実行してみると・・・

```
// TODO: This applicat  
// Restore application  
}  
WinJS.UI.processAll();  
}
```

default.js

navigator.js

```
WinJS.Namespace.define("Application1", {  
  PageControlNavigator: WinJS.Class.define(  
    function (element, options) {  
      this.element = element || document.createElement("div");  
      this.element.appendChild(this._createPageElement());  
    }  
  )  
});
```

ウオッチ 1

名前	値	型
element.id	"contenthost"	String
options	{...}	Object
[prototype]	{...}	Object
home	"/html/groupedItemsPage.html"	String

呼び出し階層 自動変数 ローカル ウオッチ 1

コントロールの作り方がなんとなく...

```
<div data-win-control="App.Test" data-win-options="{value: 10}"></div>
```

HTMLにdata-win-controlとdata-win-optionsを定義

```
WinJS.Namespace.define("App", {
  Test: WinJS.Class.define(
    function (element, options) {
      var leftButton = document.createElement("button");
      leftButton.textContent = "<";
      var numberLabel = document.createElement("span");
      var rightButton = document.createElement("button");
      rightButton.textContent = ">";

      element.appendChild(leftButton);
      element.appendChild(numberLabel);
      element.appendChild(rightButton);

      this._leftButton = leftButton;
      this._numberLabel = numberLabel;
      this._rightButton = rightButton;
      this._value = 0;
      this.value = "value" in options ? options.value : 0;
    },
    {
      value: {
        get: function () {
          return this._value;
        },
        set: function (value) {
          this._numberLabel.textContent = value;
          this._value = value;
        }
      }
    }
  )
});
```

コンストラクタの第1引数に指定されている要素、第2引数にオプションが渡される模様

詳細はまだまだ調査が必要

- WinJS.UI.setOptionsとの兼ね合い
- プロパティが更新された時の画面更新方法
- CSSクラスの指定方法
- etc...

Ratingコントロールなどシンプルなコントロールが参考になりそう



遷移部分をもう少し

PageControlNavigatorのコンストラクタ

```
this.home = options.home;  
nav.onnavigated = this._navigated.bind(this);  
appView.getForCurrentView().onviewstatechanged = thi  
document.body.onkeyup = this._keyupHandler.bind(this)  
document.body.onkeypress = this._keypressHandler.bin  
nav.navigate(this.home);
```

先頭ページのURL保持して

画面遷移完了イベントひっかけて

画面遷移に画面遷移

これで画面遷移完了？



遷移イベントを見てみる

```
_createPageElement: function () {  
    var element = document.createElement("div");  
    element.style.width = "100%";  
    element.style.height = "100%";  
    return element;  
},  
  
_navigated: function (eventObject) {  
    var newElement = this._createPageElement();  
var parentedComplete;  
    var parented = new WinJS.Promise(function (c) { parentedComplete = c; });  
  
    var that = this;  
    WinJS.UI.Pages.render(eventObject.detail.location, newElement,  
        eventObject.detail.state, parented).  
    then(function (control) {  
        that.element.appendChild(newElement);  
        that.element.removeChild(that.pageElement);  
parentedComplete();  
        document.body.focus();  
        that.navigated();  
    });  
},
```

新しい要素を作って

Pages.render?
URLを渡してるが...

結果を自要素に設定して
自要素からなんか削除してる...



ここから読み取れること

Navigator

- 「遷移依頼されたよ」ってことを通知して、指定された値を渡すだけ
 - 実際の遷移は行ってない
- ってことは、画面遷移より抽象的な遷移の橋渡し役
 - 例えば、Ajaxでよくあった次のデータとってきた時に、urlパラメーター書き換えて履歴に放り込むようなのとか
- 履歴も管理してる模様

Pages

- 指定されたURLのHTMLを取得
- 指定された要素に取得した要素を設定
 - CSSやJSの解決もやってる模様

その他

- 画面表示するしないは自前で実装
- 結構、柔軟に遷移(画面遷移ではない)を制御できる

最低限必要な部分を取り出してみる

```
var navigated = function (e) {  
  var container = document.getElementById("contentHost");  
  container.innerHTML = "";  
  
  var parentedComplete;  
  var parented = new WinJS.Promise(function (c) { parentedComplete = c; });  
  
  WinJS.UI.Pages.render(e.detail.location, container, e.detail.state, parented)  
    .then(function (e) {  
  
      parentedComplete();  
  
    });  
};  
  
WinJS.Navigation.addEventListener("navigated", navigated);  
WinJS.Navigation.navigate("/html/home.html");
```

これがないと動かない
実装の都合っぽいので
変わりそうな気がする

たったこれだけ！

遷移後の動きをみる

groupedItemsPage.js

```
ui.Pages.define("/html/groupedItemsPage.html", {
```

URLをキーにしてPagesオブジェクトを定義しているらしい

```
// This function is called whenever a user navigates to this page. It  
// populates the page elements with the app's data.  
ready: function (element, options) {
```

```
    var listView = element.querySelector(".groupeditemslist").winControl;  
  
    ui.setOptions(listView, {  
        groupHeaderTemplate: element.querySelector("#groupHeaderTemplate"),  
        itemTemplate: element.querySelector("#itemTemplate"),  
        oniteminvoked: this.itemInvoked.bind(this)  
    });  
  
    this.updateLayout(element, appView.value);  
},
```

readyが呼ばれるらしい
ってことでこれはPagesのイベント(っぽいもの)だろう



再び ready 関数

```
// This function is called whenever a user navigates to this page. It
// populates the page elements with the app's data.
ready: function (element, options) {

    var listView = element.querySelector(".groupedititemslist").winControl;

    ui.setOptions(listView, {
        groupHeaderTemplate: element.querySelector(".headerTemplate"),
        itemTemplate: element.querySelector(".itemtemplate"),
        oniteminvoked: this.itemInvoked.bind(this)
    });

    this.updateLayout(element, appView.value);
},
```

WinJS.UI.processAllしてないのにいきなりコントロール化されてる！！

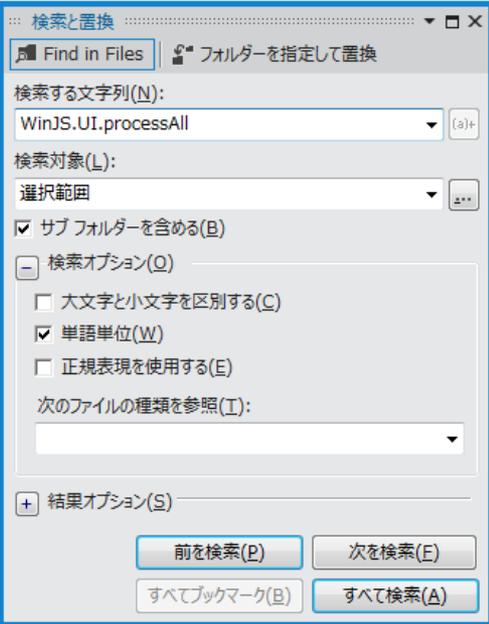
考えられること

- default.jsで最初にWinJS.UI.processAllしてるので、追加された要素は良しなに自動でやってくれる
- Pagesがロード後に内部的に行っている

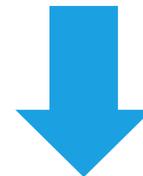
Pagesの中を探ってみる

base.js

```
(function (global) {  
  // not supported in WebWorker  
  if (!global.document) {  
    return;  
  }  
  
  function abs(uri) {  
    var a = document.createElement("a");  
    a.href = uri;  
    return a.href.toLowerCase();  
  }  
  
  var viewMap = {};  
  
  function selfhost(uri) {  
    return document.location.href.toLowerCase();  
  }  
  
  WinJS.Namespace.define("WinJS.UI.Pages", {  
    _mixin: {  
      load: function (uri) {  
        /// <summary locid="WinJS.UI.P...>  
        /// Creates a copy of the DOM...>  
        /// </summary>  
        /// <param name="uri" locid="W...>  
        /// The URI from which to copy...>  
        /// </param>  
        /// <returns locid="WinJS.UI.P...>  
        /// A promise whose fulfilled...>  
        /// </returns>  
        if (!this.selfhost) {  
          return WinJS.UI.Promise.resolve(f);  
        }  
      }  
    }  
  });  
})(global);
```



Pagesの定義部分を全選択して
検索！！！！



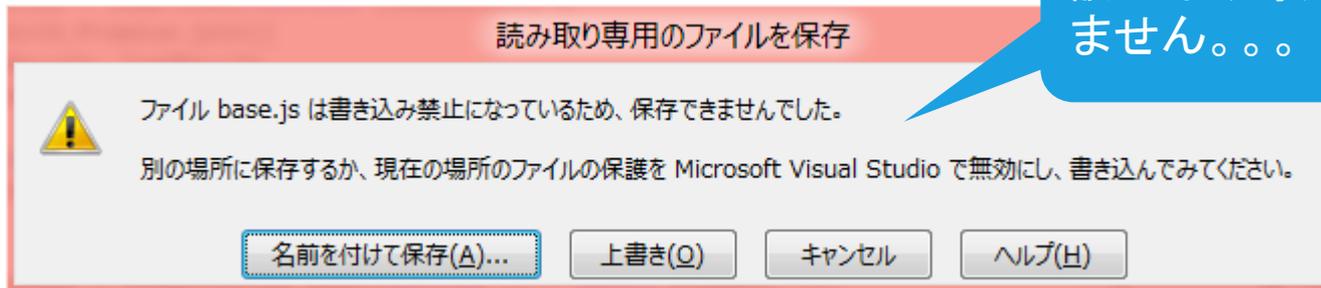
defineメソッドの中
に発見

```
...this.renderComplete = renderCalled.  
  then(function (f) {  
    return WinJS.UI.processAll(f).then(function () { return f; });  
  }).then(function (f) {  
    return that.processed(element, options);  
  }) then(  
    ...
```



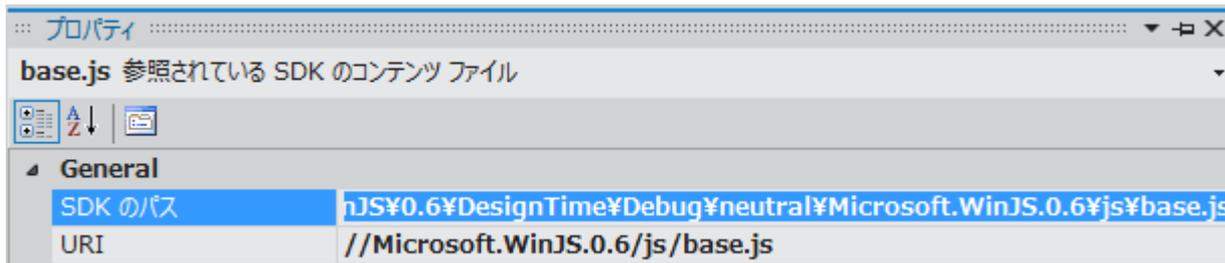
とりあえず消してみる W

1. 保存！

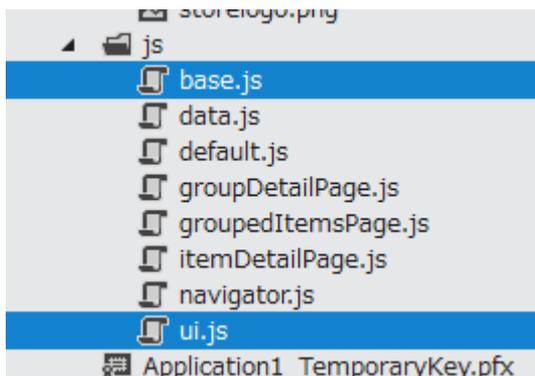


読み取り専用のため保存できません。。。

2. ファイルのプロパティから実態のありかは分かる



3. コピーして読み取り専用はずして、プロジェクトに含める(ui.jsもついでに入れといた)



とりあえず消してみる W その2

4. 書き換え成功(戻り値は適当に合わせておく)

```
then(function (f) {  
  
    //return WinJS.UI.processAll(f).then(function () { return f; });  
    return WinJS.Promise.as(function () { });  
});
```

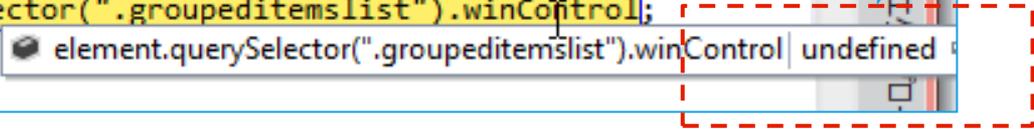
5. htmlでbase.jsの参照先を変える

```
<script src="/js/base.js"></script>  
<script src="/js/ui.js"></script>  
<!--<script src="//Microsoft.WinJS.0.6/js/base.js"></script>  
<script src="//Microsoft.WinJS.0.6/js/ui.js"></script-->
```

取れない!!!

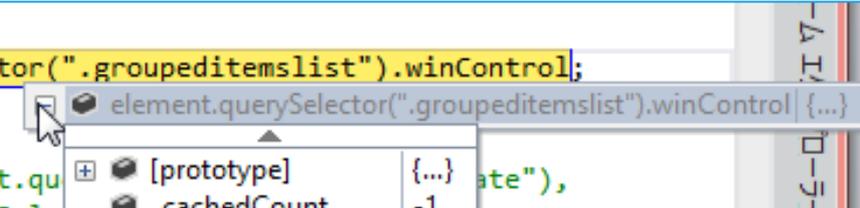
6. 動かして.winControlプロパティが取れないか確かめる

```
var listView = element.querySelector(".groupeditemslist").winControl;  
  
//ui.setOptions(listView, {
```



7. もちろんもとに戻して取れるかも確かめる

```
var listView = element.querySelector(".groupeditemslist").winControl;  
  
//ui.setOptions(listView, {  
//    groupHeaderTemplate: element.qu
```



- ソース読む
- 源典あたる
- 中を漁る
- 試す
- 公開する