

Firefox 2の新APIの 拡張機能への応用

- Session Store API 編 -



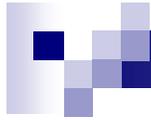
Firefox 2の拡張開発者向け新機能

- MDC (Mozilla Developer Center)
 - 開発者のためのFirefox 2の新機能
 - XUL と拡張の開発者向け
- <http://developer.mozilla.org/>



プレゼンの内容

- (1) セッション復元機能
 - 3つの復元パターン
 - 内部的な処理
- (2) セッションストアAPI
 - `nsISessionStartup`
 - `nsISessionStore`
- (3) 拡張機能への応用
 - セッションマネージャ機能
 - ScrapBook編集ツールバーのコメントを保存／復元
 - カウントダウン式のセッション回復プロンプト



(1) セッション復元機能

復元パターンその1

■ Firefox起動時、毎回セッションを復元する

□ オプション



□ browser.startup.page = 3

□ browser.sessionstore.resume_sessionは廃止

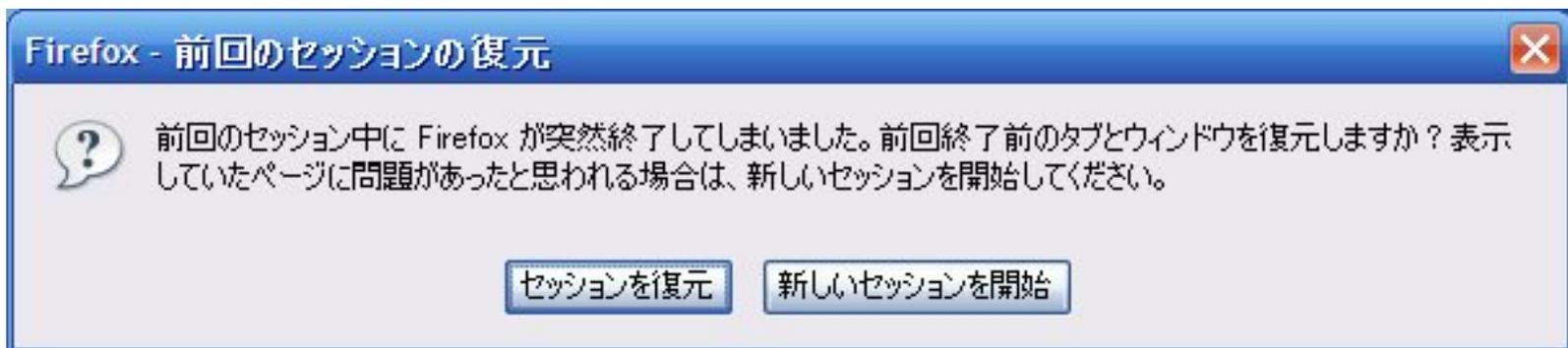
復元パターンその2

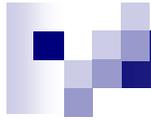
- 次回Firefox起動時、一度だけセッション復元
- 一度復元したら、それ以降は復元しない
 - `browser.sessionstore.resume_session_once = true`
 - セッション復元後は自動的に `false` へ戻る
 - 再起動直前に一時的に `true` へ変更される



復元パターンその3

- Firefoxクラッシュ後、再起動時セッション復元
 - デフォルトで有効
 - `browser.sessionstore.resume_from_crash = false` で無効





ここで、いくつかの疑問点



疑問1

そもそも何が保存／復元
されているのか？



その答え (1/3)

- すべてのウィンドウについて...
 - サイズ、位置、状態
 - サイドバー
 - 閉じたタブの履歴
 - Cookie
(有効期限がブラウザ終了時までのCookieも復元)
 - 拡張機能独自の値
(詳細は後ほど)



その答え (2/3)

■ すべてのタブについて...

- 現在表示しているページのURL
(その内容は極力キャッシュから取り出される)
- 戻る／進むの履歴
- 文字の拡大／縮小状態
- 画像やスクリプトの許可状態
- 拡張機能独自の値
(詳細は後ほど)



その答え (3/3)

- すべてのページ／フレームについて...
 - スクロール位置
 - テキストボックスへの入力値
 - POSTされたデータ
(デフォルトでは保存されない)



疑問2

セッションはどこに
保存されているのか？



その答え

- プロファイルフォルダ内の「sessionstore.js」
- JSON形式で保存されている



疑問3

セッションはいつ
保存されているのか？



その答え(1/2)

- 以下のイベント発生後...
 - ウィンドウを開いた／閉じた
 - タブを開いた／閉じた
 - タブを選択した
 - ページをロードした
 - ページを遷移した(戻る／進む)
 - テキストボックスへ入力した
- 最大10秒以内に保存(更新)される



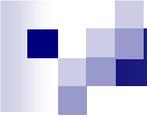
その答え(2/2)

- 以下のイベント発生後...
 - プライバシー情報を消去した
 - Firefoxを終了あるいは再起動させようとした
- 即座に保存(更新)される



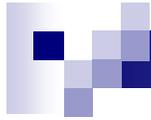
疑問4

どのようにクラッシュを
検知しているのか？



その答え

- セッションデータ内部の session.state の値
 - Firefox起動中 : session { state:"running" }
↓
 - Firefoxを正常終了 : データは削除される
 - Firefox再起動直前 : session { state:"stopped" }
↓
 - Firefox再起動時、前回保存時のセッションを調べ、session { state:"running" } のセッションが残っていれば、異常終了したと見なす。



(2) セッションストアAPI



2つのXPCOMサービス

- nsISessionStartup
- nsISessionStore



nsISessionStartup

Firefox起動時に呼び出され...

- 前回のクラッシュを検知する
- セッション復元プロンプトを表示する
- セッション復元すべきかどうかを判定する



nsISessionStore

- 10秒ごとに保存するタイマーを仕掛ける
- セッション状態の変化を監視する
- sessionstore.jsの読み書き
- 実際の保存／復元処理を行う
- 「閉じたタブを元に戻す」処理を行う



APIを拡張機能から利用する

- nsISessionStoreが持つ色々なメソッド
- 一部のメソッドをAPIとして拡張機能から利用
- セッションストア機能の潜在能力を引き出す

<http://developer.mozilla.org/en/docs/nsISessionStore>



getBrowserState()メソッド

- 現在のセッション状態を表すJSON文字列を取得する
- `AString getBrowserState();`



setBrowserState()メソッド

- セッション状態を表すJSON文字列からセッション復元を実行する
- ```
void setBrowserState(
 in AString aState
);
```



## setTabValue()メソッド

- 特定のタブに関連付けて  
拡張機能独自の値を保存する
- `void setTabValue(  
    in nsIDOMNode aTab,  
    in AString aKey,  
    in AString aStringValue  
);`



## getTabValue()メソッド

- setTabValueによって特定のタブに関連付けられた拡張機能独自の値を取得する
- `AString getTabValue(`  
    in nsIDOMNode aTab,  
    in AString aKey  
);



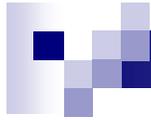
## deleteTabValue()メソッド

- setTabValueによって  
特定のタブに関連付けられた  
拡張機能独自の値を削除する
- void deleteTabValue(  
    in nsIDOMNode aTab,  
    in AString aKey  
);



## undoCloseTab()メソッド

- 特定のウィンドウについて  
特定の回数前に閉じたタブの  
開き直し処理を実行する
- `void undoCloseTab(  
    in nsIDOMWindow aWindow,  
    in unsigned long aIndex  
);`



## (3) 拡張機能への応用



## 応用例その1

- 好きなタイミングで何度でもセッション保存／復元ができる拡張機能
- いわゆるセッションマネージャ

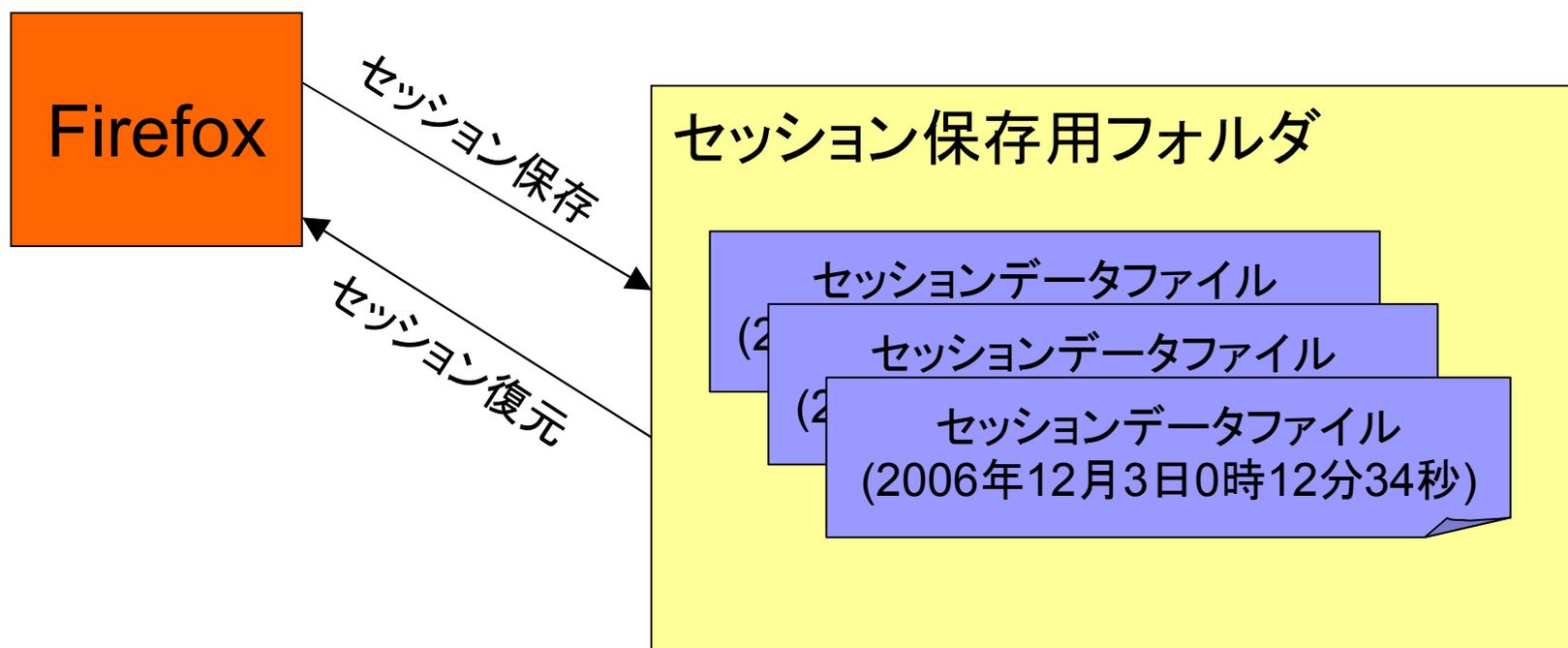
# 完成品のイメージ (1/2)

The image shows a screenshot of a web browser's developer tools menu. The menu is open, and the 'セッションマネージャ' (Session Manager) option is selected and highlighted in blue. A mouse cursor is pointing at the right arrow of this option. To the right of the main menu, a sub-menu is displayed, listing the following items:

- 現在のセッションを保存
- 2006年12月8日 17:38:07
- 2006年12月6日 1:23:56
- 2006年12月6日 0:52:24
- クリア

The background shows a browser window with the Google Japan search engine visible in the address bar.

# 完成品のイメージ (2/2)





# ポイント

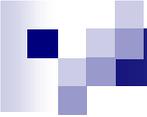
- 以下のAPIを利用

- `nsISessionStore::getBrowserState()`

- `nsISessionStore::setBrowserState()`

# 実装① browser.xulへのオーバーレイ

```
<menupopup id="menu_ToolsPopup">
 <menu label="セッションマネージャ" insertbefore="sanitizeSeparator">
 <menupopup onpopupshowing="exSessionManager.onPopupShowing(event);"
 oncommand="exSessionManager.restore(event);">
 <menuitem label="現在のセッションを保存"
 oncommand="exSessionManager.save();" />
 <menuseparator />
 <menuseparator />
 <menuitem label="クリア"
 oncommand="exSessionManager.clear();" />
 </menupopup>
 </menu>
</menupopup>
```



## 実装② JavaScriptの概要

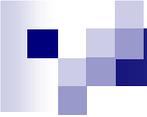
```
var exSessionManager = {
 // セッション保存処理を実行
 save: function() {},
 // セッション復元処理を実行
 restore: function(event) {},
 // 保存されたすべてのセッションデータを削除する
 clear: function() {},
 // 「セッションマネージャ」メニューのポップアップ時のイベントハンドラ
 onPopupShowing: function(event) {},
 // プロファイルフォルダの「sessionmanager」フォルダからファイルを取得
 _getFile: function(aFileName) {},
 // ファイルを読み込む
 _readFile: function(aFile) {},
 // ファイルへ書き込む
 _writeFile: function(aFile, aData) {},
};
```

## 実装③ save関数の詳細

```
save: function()
{
 // XPCOMサービスを呼び出し
 var ss = Components.classes['@mozilla.org/browser/sessionstore;1']
 .getService(Components.interfaces.nsISessionStore);

 // セッション状態を表すJSON文字列を取得
 var state = ss.getBrowserState();

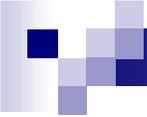
 // ファイルへ書き込み
 var timeStamp = (new Date().getTime()).toString();
 var file = this._getFile("session_" + timeStamp + ".js");
 this._writeFile(file, state);
},
```



## 実装④ restore関数の詳細

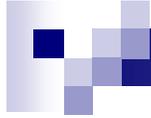
```
restore: function(event)
{
 // クリックしたmenuItem要素のIDからファイル名を取得
 var fileName = event.target.id;
 // ファイルからセッション状態を表すJSON文字列を読み出し
 var file = this._getFile(fileName);
 var state = this._readFile(file);

 // XPCOMサービスを呼び出し
 var ss = Components.classes['@mozilla.org/browser/sessionstore;1']
 .getService(Components.interfaces.nsISessionStore);
 // JSON文字列からセッション復元を実行
 ss.setBrowserState(state);
},
```



## 実装⑤ その他の関数

- clear()
  - セッション保存用フォルダを丸ごと削除するだけ
- onPopupShowing(event)
  - セッション保存用フォルダ内のファイル一覧を取得してメニューポップアップを生成
- \_getFile(aFileName)
  - セッション保存用フォルダからファイルを取得
- \_readFile(aFile)
- \_writeFile(aFile, aData)
  - ファイルの読み書き



完成したセッションマネージャを  
動かしてみよう...



## 応用例その2

- ScrapBookの編集ツールバー
- 入力中(保存前)のコメントは、タブを切り替えるたびに消えてしまう

そこで...

- セッションストア機能を利用して、入力中のコメントを保存／復元したい



## まずは実際の動作を比較

- ビフォー

- ScrapBookの現在のバージョン

- アフター

- セッションストア機能を利用したバージョン



# ポイント

- 以下のAPIを利用

- `nsISessionStore::setTabValue()`

- `nsISessionStore::getTabValue()`

- `nsISessionStore::deleteTabValue()`

## 実装① コメントをセッションデータへ保存

- 編集ツールバーにコメントを入力した時...
- 現在のタブへ関連付けてコメントの値を保存する

### XUL

```
<textbox id="ScrapBookEditComment"
 oninput="sbPageEditor.onInputComment(this.value);" />
```

### JavaScript

```
onInputComment: function(aValue)
{
 var ss = Components.classes['@mozilla.org/browser/sessionstore;1']
 .getService(Components.interfaces.nsISessionStore);
 ss.setTabValue(gBrowser.mCurrentTab, "scrapbook-comment", aValue);
},
```

- 
- このとき、sessionstore.jsには以下のように  
拡張機能独自の値が保存される

```
extData: {'scrapbook-comment': "hogehoge"},
```



## 実装② セッションデータからコメントを復元

- タブを切り替えて編集ツールバーを表示した時、
- 現在のタブに関連付けられたコメントの値を、
- 編集ツールバーのテキストボックスへセット

### JavaScript

```
var ss = Components.classes['@mozilla.org/browser/sessionstore;1']
 .getService(Components.interfaces.nsISessionStore);
var val = ss.getTabValue(gBrowser.mCurrentTab, "scrapbook-comment");
if (val)
 document.getElementById("ScrapBookEditComment").value = val;
```



## 実装③ セッションデータのコメントを削除

- 保存ボタン押下時、
- 現在のタブに関連付けられたコメントの値を、
- キーごと削除

JavaScript

```
var ss = Components.classes['@mozilla.org/browser/sessionstore;1']
 .getService(Components.interfaces.nsISessionStore);
ss.deleteTabValue(gBrowser.mCurrentTab, "scrapbook-comment");
```

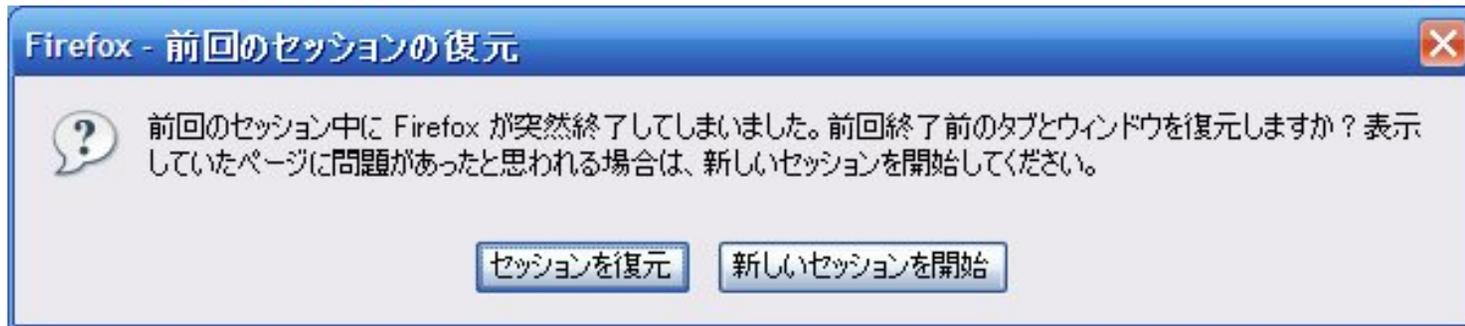


セッションデータへ保存されるということは、  
Firefoxを再起動しても、  
入力中のコメントは復元されるはず。

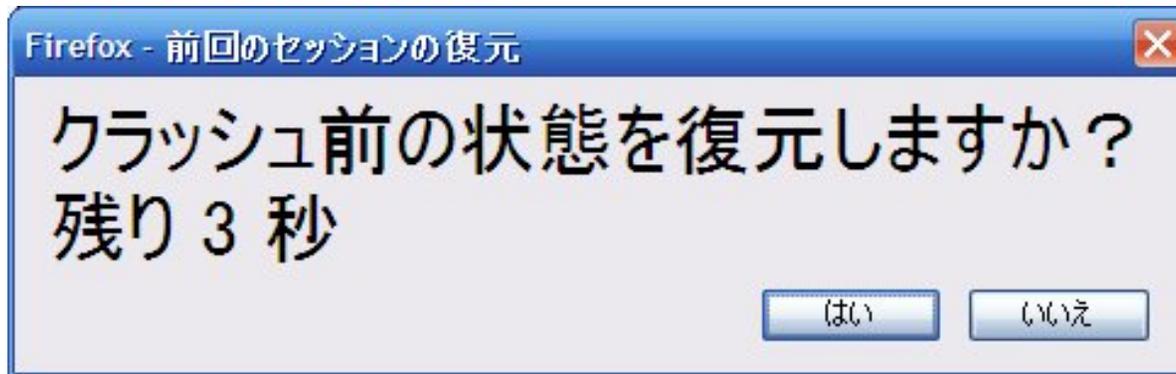
実際に試してみよう...

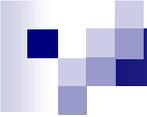
## 応用例その3

### ■ Firefoxクラッシュ後のセッション復元プロンプト



### ■ カウントダウン式にする





# ポイント

- 以下の設定値を変更すると...

- `browser.sessionstore.restore_prompt_uri`  
=「chrome://xuldev/content/sessionrestore.xul」

- 標準プロンプトを独自のものに置き換え可能  
(by zenikoさん)

- SessionStartupクラスの `_doRecoverSession()`

```
// allow extensions to hook in a more elaborate restore prompt
// XXXzeniko drop this when we're using our own dialog
// instead of a standard prompt
var dialogURI = this._getPref("sessionstore.restore_prompt_uri");
```

# 実装① XUL

```
<?xml version="1.0" ?>
```

```
<?xml-stylesheet href="chrome://global/skin/" type="text/css" ?>
```

```
<dialog title="Firefox - 前回のセッションの復元"
 xmlns="http://www.mozilla.org/keymaster/gatekeeper/there.is.only.xul"
 buttonlabelaccept="はい" buttonlabelcancel="いいえ"
 onload="exSessionRestore.onLoad();"
 ondialogcancel="exSessionRestore.onCancel();"
 style="font-size: xx-large;">
```

```
 <script type="application/x-javascript"
 src="chrome://xuldev/content/sessionrestore.js" />
```

```
 <label value="クラッシュ前の状態を復元しますか?" />
```

```
 <label value="残り 5 秒" id="message" />
```

```
</dialog>
```



## 実装② JavaScriptの概要

```
var exSessionRestore = {

 // 自動的にプロンプトを閉じるまでの残り時間
 _seconds: 5,

 // プロンプトが表示されたときのイベントハンドラ
 onLoad: function() {},

 // 「いいえ」ボタンを押したときのイベントハンドラ
 onCancel: function() {}

};
```

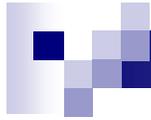
## 実装③ onLoad関数の詳細

```
// コールバック関数を定義
var callback = function(self)
{
 return function()
 {
 if (--self._seconds > 0)
 {
 // 残り1秒以上であればカウントダウンする
 var val = "残り " + self._seconds + " 秒";
 document.getElementById("message").value = val;
 }
 else
 // 残り0秒ならウィンドウを閉じる
 window.close();
 };
};
// タイマー発動
setInterval(callback(this), 1000);
```



## 実装④ onCancel関数の詳細

```
// window.arguments[0]を0以外の値に変更する
var params = window.arguments[0].
 QueryInterface(Components.interfaces.nsIDialogParamBlock);
params.SetInt(0, 1);
```



完成したセッション復元  
プロンプトを試してみよう...