Web Browser for NORTi

ユーザーズガイド

2006年 5月14日 第4版

株式会社ミスポ

目次

1.概要

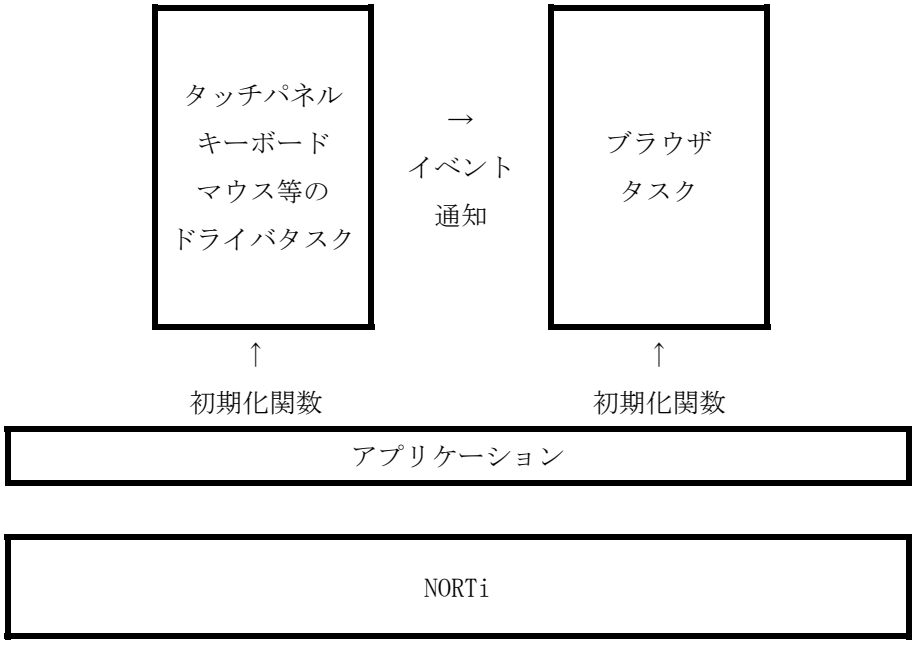
本書は、NORTi 上で動作する組込みシステム用ブラウザ Web Browser for NORTi を実装する為に必要な以下の情報について記述したものです。

・タスク構成
・ブラウザが使用するオブジェクト
・フォルダおよびファイル構成
・ブラウザ API の説明
・機能一覧
・サポートタグ

Web Browser for NORTi は、Unicoi Systems 社の組込みシステム用ブラウザエンジン Fusion Web Pilot を採用しています。

2.タスク構成

ブラウザを起動するには、ブラウザ初期化関数を実行して下さい。初期化関数は、ブラウザが必要とするオブジェクトを生成後、ブラウザタスクを開始します。キーボード・マウス等のイベントは、ブラウザ API を使用してブラウザタスクに通知して下さい。



3.オブジェクト

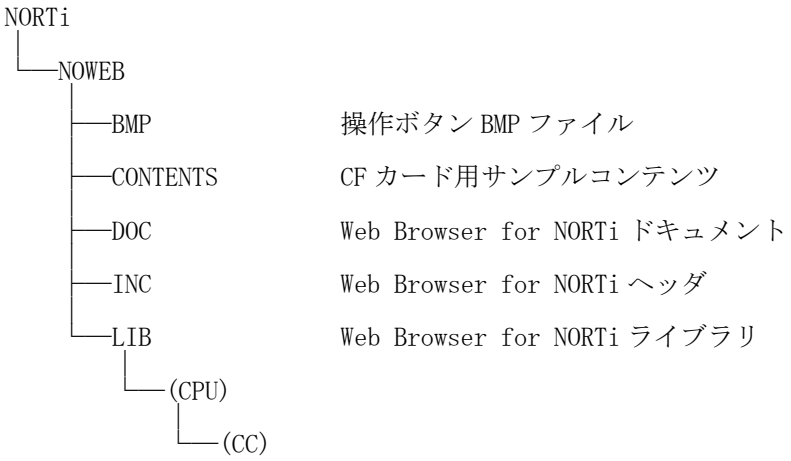
ブラウザは次のオブジェクトを使用します。

タスク	×１	(ブラウザタスク)
可変長メモリプール	×１	(ブラウザヒープ)
ミューテックス	×１	(イベントキューのタスク間同期用)
イベントフラグ	×１	(タイムアウト指定のある API をコールする度に１つ)

NORTi のコンフィグレーションで各オブジェクトの使用数を設定する場合、これらの数値を考慮して下さい。オブジェクト ID 使用数は、ブラウザライブラリヘッダファイルに定義されています。

```
#define WEB_NTSK        1   /* タスク ID 使用数 */
#define WEB_NSEM        0   /* セマフォ ID 使用数 */
#define WEB_NFLG        1   /* イベントフラグ ID 使用数 */
#define WEB_NMBX        0   /* メイルボックス ID 使用数 */
#define WEB_NMBF        0   /* メッセージバッファ ID 使用数 */
#define WEB_NPOR        0   /* ランデブ用ポート ID 使用数 */
#define WEB_NMPL        1   /* 可変長メモリプール ID 使用数 */
#define WEB_NMPF        0   /* 固定長メモリプール ID 使用数 */
#define WEB_NDTQ        0   /* データキューID 使用数 */
#define WEB_NMTX        1   /* ミューテックス ID 使用数 */
#define WEB_NISR        0   /* 割り込みサブルーチン ID 使用数 */
#define WEB_NCYC        0   /* 周期起動ハンドラ ID 使用数 */
#define WEB_NALM        0   /* アラームハンドラ ID 使用数 */
```

4. フォルダおよびファイル構成



※(CPU)は CPU 名略称、(CC)はコンパイラ名略称です。

noweb.h
ブラウザを利用する全てのソースファイルで#include して下さい。

nowbxxx?.lib
ブラウザを利用するアプリケーションでリンクして下さい。xxx の部分は対応プロセッサ名に依存します。?の部分(l または b)はエンディアンを表します。

5.API

5.1 web_ini

機能
ブラウザの初期化

形式
ER web_ini(T_WEB_INI *pk_web_ini);
pk_web_ini　　初期化パラメータ構造体へのポインタ

解説
web_ini はブラウザが使用するオブジェクトを生成し、ブラウザタスクを起動します。
初期化パラメータ構造体の定義は次の通りです。

```
typedef struct t_web_ini
{
	UH width;			ブラウザウインドウ幅
	UH height;			ブラウザウインドウ高さ
	UB bpp;				bit per pixel
	const B *url;			デフォルト URL
	NOTIFY_CALLBACK callback;	通知コールバック
	PAINT_CALLBACK paintcb;		画面描画コールバック
	UB *fontarray[4];		フォントデータ先頭アドレス
	BOOL showmouse;			マウスポインタの表示・非表示
	SIZE mplsz;			メモリプールサイズ
} T_WEB_INI;
```

width、height にはブラウザウインドウの幅、高さを指定して下さい。
bpp には 1 ピクセルあたりのビット数 16 を指定して下さい。ブラウザが出力する画面データのカラーフォーマットは RGB565 です。
url にはブラウザ起動時、または、ホームボタンを押したときに遷移する URL 文字列を指定して下さい。NULL を設定した場合、起動時およびホームボタン押下時のページ遷移は起こりません。
callback には次の API をノンブロックキング（tmout = TMO_NBLK）で発行した時に、API の処理結果を通知する為にブラウザライブラリよりコールされるコールバック関数ポインタを設

定して下さい。

```
ER web_cal_hme(TMO tmout);
ER web_cal_rld(TMO tmout);
ER web_cal_bwd(TMO tmout);
ER web_cal_fwd(TMO tmout);
ER web_url_jmp(const B *url, TMO tmout);
```

通知コールバック関数の形式は次の通りです。

```
void callback(FN fncd, UINT status);
```

fncd	どの API に対する通知かを識別する機能コード
status	API の処理結果

fncd の値は次の通りです。

TFN_WEB_CAL_HME	web_cal_hme に対する通知
TFN_WEB_CAL_RLD	web_cal_rld に対する通知
TFN_WEB_CAL_BWD	web_cal_bwd に対する通知
TFN_WEB_CAL_FWD	web_cal_fwd に対する通知
TFN_WEB_URL_JMP	web_url_jmp に対する通知

status の値は次の通りです。

WEB_STA_DNE	正常終了
WEB_STA_ABT	読込み中止された
WEB_STA_CON	リモートホストに接続失敗
WEB_STA_NOT	処理は行われなかった
WEB_STA_INC	URL が不正または接続失敗
WEB_STA_IMG	画像の読込みに失敗

正常終了しなかった要因が複数ある場合、status の値は全ての要因の OR 値となります。
通知コールバック関数を使用しない場合は必ず callback に NULL を設定して下さい。

paintcb にはブラウザ画面描画コールバック関数ポインタを設定して下さい。
画面描画コールバック関数の形式は次の通りです。

```
void paintcb(UH x, UH y, UH width, UH height, VP img, UH widthbytes);
```

x	描画矩形左上 X 座標
y	描画矩形左上 Y 座標

width	描画矩形幅
height	描画矩形高さ
img	ブラウザ内部ディスプレイバッファ先頭アドレス
widthbytes	ブラウザ内部ディスプレイバッファ画面データ 1 行のバイト数

ブラウザライブラリには、初期化時に指定した幅と高さを持つディスプレイバッファがあり、ブラウザ内部の描画処理はディスプレイバッファに対して行われます。内部的な描画処理が完了した時点で、VRAM に再描画が必要な矩形領域を指定して本コールバック関数がコールされます。コールバック関数内に、ディスプレイバッファの描画矩形にあるデータを VRAM 空間の任意の位置にコピーする処理を実装して下さい。ディスプレイバッファ内データのカラーフォーマットは 1 ピクセル 16bit で RGB565 です。
fontarray は将来の拡張用です。必ず NULL を設定して下さい。
showmouse にはマウスポインタを表示する場合は TRUE を、表示しない場合は FALSE を設定して下さい。
mplsz にはブラウザが内部で動的なメモリ確保に使用する可変長メモリプール全体のサイズをバイト数で指定して下さい。0 を指定すると 0x800000(8MB)に設定されます。メモリプールサイズが小さすぎると、ディスプレイバッファや JPEG 等、画像データ処理領域が不足し、正常に動作しません。また、物理的メモリサイズを超える値を指定することはできません。ブラウザが動作するシステムに合わせて適切な値を設定して下さい。

戻値

E_OK	正常終了
E_NOID	タスク ID またはミューテックス ID または可変長メモリプール ID が不足
E_CTX	割込みハンドラから発行
E_SYS	管理ブロック用のメモリが確保できない
E_NOMEM	スタック用またはメモリプール用のメモリが確保できない
EV_DSP	描画系初期化処理に失敗
E_OBJ	タスクが既に起動されている

例

```
#include "noweb.h"
      :
void NotifyCallBack(FN fncd, UINT status);
void PaintCallBack(UH x, UH y, UH width, UH height, VP img, UH widthbytes);

const T_WEB_INI cweb_ini = {
```

```
        240, 320, 16,
        "http://www.mispo.co.jp/",
        NotifyCallBack,
        PaintCallBack,
        {NULL, NULL, NULL, NULL},
        FALSE,
        0
 };
        :
void NotifyCallBack(FN fncd, UINT status)
{
    switch(fncd)
    {
    case TFN_WEB_CAL_HME:
        if(status != WEB_STA_DNE)
                        :
        break;
    case TFN_WEB_CAL_RLD:
        if(status != WEB_STA_DNE)
                        :
        break;
    }
}

void PaintCallBack(UH sx, UH sy, UH sw, UH sh, VP img, UH wb)
{
    UH *dst = (UH *)VRAM_TOP;
    UH *src = (UH *)img;
    UH dst_w = 240;
    UH src_w = wb / sizeof(UH);
    UH x, y;

    for (y = sy; y < sy + sh; y++) {
        for (x = sx; x < sx + sw ; x++) {
            *(dst + (y * dst_w) + x) = *(src + (y * src_w) + x);
        }
```

```
    }
}

TASK MainTask(void)
{
        ER ercd;
        ercd = web_ini(&cweb_ini);
        if(ercd != E_OK)
                :
}
```

5.2 web_ext

機能
ブラウザの終了

形式
```
ER web_ext(void);
```

解説
web_extはブラウザが使用するオブジェクトを削除し、ブラウザタスクを終了します。

戻値

E_OK	正常終了
E_NOEXS	ブラウザタスクが生成されていない

例
```
#include ″noweb.h″
       :
web_ext();
```

5.3 web_inp_key

機能

キーボードイベントの通知

形式

ER web_inp_key(UB ch, UW status);

ch	ASCII コードまたは仮想キーコード
status	キーステータスコード

解説

web_inp_key はキーボードイベントをブラウザタスクに通知します。

仮想キーコードは次の通りです。

WEB_KEY_INS	INSERT キー
WEB_KEY_DEL	DELETE キー
WEB_KEY_PGU	PAGE UP キー
WEB_KEY_MVU	↑キー
WEB_KEY_MVD	↓キー
WEB_KEY_MVL	←キー
WEB_KEY_MVR	→キー
WEB_KEY_PGD	PAGE DOWN キー
WEB_KEY_HME	HOME キー
WEB_KEY_END	END キー
WEB_KEY_F1	F1 キー
WEB_KEY_F2	F2 キー
WEB_KEY_F3	F3 キー
WEB_KEY_F4	F4 キー
WEB_KEY_F5	F5 キー
WEB_KEY_F6	F6 キー
WEB_KEY_F7	F7 キー
WEB_KEY_F8	F8 キー
WEB_KEY_F9	F9 キー
WEB_KEY_F10	F10 キー
WEB_KEY_F11	F11 キー

WEB_KEY_F12	F12 キー

キーステータスコードは次の通りです。

WEB_KEY_STA_NON	(通常はこの値を設定)
WEB_KEY_STA_RST	右シフトキー押下中
WEB_KEY_STA_LST	左シフトキー押下中
WEB_KEY_STA_CAP	CAPS LOCK オン
WEB_KEY_STA_NUM	NUM LOCK オン
WEB_KEY_STA_RCT	右 CTRL キー押下中
WEB_KEY_STA_LCT	左 CTRL キー押下中
WEB_KEY_STA_RAL	右 ALT キー押下中
WEB_KEY_STA_LAL	左 ALT キー押下中
WEB_KEY_STA_RMS	右マウスボタン押下中
WEB_KEY_STA_LMS	左マウスボタン押下中
WEB_KEY_STA_MMS	中央マウスボタン押下中

戻値

E_OK	正常終了
E_NOEXS	ブラウザタスクが生成されていない

例

```
#include "noweb.h"
      :
UB ch = …;
UW status = …;
      :
web_inp_key (ch, status);
```

5.4 web_mse_mov

機能

マウスムーブイベントの通知

形式

```
ER web_mse_mov(INT x, INT y, UW status);
```

x	マウス X 座標
y	マウス Y 座標
status	キーステータスコード

解説

web_mse_mov はマウスムーブイベントをブラウザタスクに通知します。

戻値

E_OK	正常終了
E_NOEXS	ブラウザタスクが生成されていない

例

```
#include "noweb.h"
      :
INT x = …,y = …;
UW status = …;
      :
web_mse_mov(x, y, status);
```

5.5 web_mse_lbd

機能

マウス左ボタンダウンイベントの通知

形式

```
ER web_mse_lbd(INT x, INT y, UW status);
```

x	マウス X 座標
y	マウス Y 座標
status	キーステータスコード

解説

web_mse_lbd はマウス左ボタンダウンイベントをブラウザタスクに通知します。
キーステータスコード status には WEB_KEY_STA_LMS を or で設定して下さい。

戻値

E_OK	正常終了
E_NOEXS	ブラウザタスクが生成されていない

例

```
#include "noweb.h"
      :
INT x = …,y = …;
UW status = WEB_KEY_STA_LST | WEB_KEY_STA_LMS;

web_mse_lbd(x, y, status);
```

5.6 web_mse_lbu

機能

マウス左ボタンアップイベントの通知

形式

```
ER web_mse_lbu (INT x, INT y, UW status);
```

x	マウス X 座標
y	マウス Y 座標
status	キーステータスコード

解説

web_mse_lbu はマウス左ボタンアップイベントをブラウザタスクに通知します。

戻値

E_OK	正常終了
E_NOEXS	ブラウザタスクが生成されていない

例

```
#include "noweb.h"
      :
INT x = …,y = …;
UW status = …;

web_mse_lbu(x, y, status);
```

5.7 web_chg_pri

機能

ブラウザタスクベース優先度変更

形式

```
ER web_chg_pri(PRI tskpri);
tskpri          優先度
```

解説

web_chg_priはブラウザタスクのベース優先度(初期値7)をtskpriの値に変更します。この関数はNORTiのシステムコールchg_pri()をブラウザタスクに対して実行します。従って、tskpriにTPRI_INIを指定することにより、初期優先度に戻すことができます。ベース優先度を変更することによるタスクの挙動は『NORTi Version4 カーネル編 ユーザーズガイド』のchg_pri()の解説を参照して下さい。

戻値

```
E_OK            正常終了
E_NOEXS         ブラウザタスクが生成されていない
E_PAR           優先度が範囲外
E_OBJ           タスクが起動されていない
```

例

```
#include “noweb.h”
        :
ER ercd;
        :
ercd = web_chg_pri(5);
if(ercd != E_OK)
        :
web_chg_pri(TPRI_INI);
if(ercd != E_OK)
        :
```

5.8 web_set_fcs

機能

ツールバーのフォーカス設定

形式

ER web_set_fcs(ID id)

id　　ツールバーコンポーネント ID

解説

web_set_fcs は id で指定したツールバー上のボタンまたは URL 入力エディットボックスにフォーカスを設定します。

ツールバーコンポーネント ID の値は次の通りです。

WEB_TBRID_HME	HOME ボタン
WEB_TBRID_RLD	RELOAD ボタン
WEB_TBRID_BWD	BACK ボタン
WEB_TBRID_FWD	FORWARD ボタン
WEB_TBRID_STP	STOP ボタン
WEB_TBRID_URL	URL 入力エディットボックス
WEB_TBRID_EXT	EXIT ボタン

戻値

E_OK	正常終了
E_NOEXS	ブラウザタスクが生成されていない
E_ID	ツールバーコンポーネント ID が不正

例

```
#include “noweb.h”
      :
ER ercd;
      :
ercd = web_set_fcs(WEB_TBRID_HME);
if(ercd != E_OK)
      :
```

5.9 web_cal_hme

機能
初期URLへページ遷移

形式
ER web_cal_hme(TMO tmout);
tmout　　タイムアウト値

解説
web_cal_hmeはT_WEB_INI構造体のurlメンバに設定したURLへページ遷移し、HOMEボタンにフォーカスを設定します。

本APIの処理結果の通知方法はtmoutの値により次のようになります。
タイムアウトなし（tmout = TMO_FEVR）で本APIを発行した場合、発行元のタスクは、処理が完了するまで待ち状態となります。この場合正の戻値として処理結果が通知されます。
タイムアウトあり（tmout = 1 ～ 0x7fffffff）で本APIを発行した場合、指定した時間が経過しても処理が完了しなければ、E_TMOUT エラーが返ります。tmoutは、NORTiのMSECマクロを用いてミリ秒単位で時間指定することもできます。
ノンブロッキング（tmout = TMO_NBLK）で本APIを発行した場合、APIはすぐにリターンし、処理の完了はコールバック関数で通知されます。コールバック関数については、web_iniの説明を参照して下さい。

本APIをタイムアウトあり、またはなしで実行したときの処理結果の値は次の通りです。

WEB_STA_DNE	正常終了
WEB_STA_ABT	読込み中止された
WEB_STA_CON	リモートホストに接続失敗
WEB_STA_NOT	処理は行われなかった
WEB_STA_INC	URLが不正または接続失敗
WEB_STA_IMG	画像の読込みに失敗

正常終了しなかった要因が複数ある場合、処理結果の値は全ての要因のOR値となります。

戻値
正の値　　処理結果(tmout = TMO_FEVRまたはtmout = 1 ～ 0x7fffffffのとき)

E_OK	正常終了(tmout = TMO_NBLKのとき)
E_NOEXS	ブラウザタスクが生成されていない
E_NOID	イベントフラグID が不足
E_CTX	割込みハンドラから発行、または、ディスパッチ禁止状態で待ち実行
E_SYS	管理ブロック用のメモリが確保できない
E_TMOUT	タイムアウト

例

```
#include “noweb.h”
       :
ER ercd;
       :
ercd = web_cal_hme(5000/MSEC);
if(ercd != WEB_STA_DNE)
       :
```

5.10 web_cal_rld

機能
表示中のページを再読込み

形式
ER web_cal_rld (TMO tmout);
tmout　　タイムアウト値

解説
web_cal_rld は表示中のページを再読込みし、RELOAD ボタンにフォーカスを設定します。

本 API の処理結果の通知方法は tmout の値により次のようになります。
タイムアウトなし（tmout = TMO_FEVR）で本 API を発行した場合、発行元のタスクは、処理が完了するまで待ち状態となります。この場合正の戻値として処理結果が通知されます。
タイムアウトあり（tmout = 1 ～ 0x7fffffff）で本 API を発行した場合、指定した時間が経過しても処理が完了しなければ、E_TMOUT エラーが返ります。tmout は、NORTi の MSEC マクロを用いてミリ秒単位で時間指定することもできます。
ノンブロッキング（tmout = TMO_NBLK）で本 API を発行した場合、API はすぐにリターンし、処理の完了はコールバック関数で通知されます。コールバック関数については、web_ini の説明を参照して下さい。

本 API をタイムアウトあり、またはなしで実行したときの処理結果の値は次の通りです。

WEB_STA_DNE	正常終了
WEB_STA_ABT	読込み中止された
WEB_STA_CON	リモートホストに接続失敗
WEB_STA_NOT	処理は行われなかった
WEB_STA_INC	URL が不正または接続失敗
WEB_STA_IMG	画像の読込みに失敗

正常終了しなかった要因が複数ある場合、処理結果の値は全ての要因の OR 値となります。

戻値

正の値	処理結果(tmout = TMO_FEVR または tmout = 1 ～ 0x7fffffff のとき)
E_OK	正常終了(tmout = TMO_NBLK のとき)

E_NOEXS	ブラウザタスクが生成されていない
E_NOID	イベントフラグ ID が不足
E_CTX	割込みハンドラから発行、または、ディスパッチ禁止状態で待ち実行
E_SYS	管理ブロック用のメモリが確保できない
E_TMOUT	タイムアウト

例

```
#include “noweb.h”
      :
ER ercd;
      :
ercd = web_cal_rld (5000/MSEC);
if(ercd != WEB_STA_DNE)
      :
```

5.11 web_cal_bwd

機能

１つ前のページに戻る

形式

ER web_cal_bwd(TMO tmout);

tmout　　タイムアウト値

解説

web_cal_bwd は１つ前のページに戻り、BACK ボタンにフォーカスを設定します。URL ヒストリに履歴が無い場合、ページ遷移は発生しません。

本 API の処理結果の通知方法は tmout の値により次のようになります。
タイムアウトなし（tmout = TMO_FEVR）で本 API を発行した場合、発行元のタスクは、処理が完了するまで待ち状態となります。この場合正の戻値として処理結果が通知されます。
タイムアウトあり（tmout = 1 ～ 0x7fffffff）で本 API を発行した場合、指定した時間が経過しても処理が完了しなければ、E_TMOUT エラーが返ります。tmout は、NORTi の MSEC マクロを用いてミリ秒単位で時間指定することもできます。
ノンブロッキング（tmout = TMO_NBLK）で本 API を発行した場合、API はすぐにリターンし、処理の完了はコールバック関数で通知されます。コールバック関数については、web_ini の説明を参照して下さい。

本 API をタイムアウトあり、またはなしで実行したときの処理結果の値は次の通りです。

WEB_STA_DNE	正常終了
WEB_STA_ABT	読込み中止された
WEB_STA_CON	リモートホストに接続失敗
WEB_STA_NOT	処理は行われなかった
WEB_STA_INC	URL が不正または接続失敗
WEB_STA_IMG	画像の読込みに失敗

正常終了しなかった要因が複数ある場合、処理結果の値は全ての要因の OR 値となります。

戻値

正の値　　処理結果(tmout = TMO_FEVR または tmout = 1 ～ 0x7fffffff のとき)

E_OK	正常終了(tmout = TMO_NBLK のとき)
E_NOEXS	ブラウザタスクが生成されていない
E_NOID	イベントフラグ ID が不足
E_CTX	割込みハンドラから発行、または、ディスパッチ禁止状態で待ち実行
E_SYS	管理ブロック用のメモリが確保できない
E_TMOUT	タイムアウト

例

```
#include “noweb.h”
      :
ER ercd;
      :
ercd = web_cal_bwd (5000/MSEC);
if(ercd != WEB_STA_DNE)
      :
```

5.12 web_cal_fwd

機能

１つ先のページに進む

形式

ER web_cal_fwd(TMO tmout);

tmout　　タイムアウト値

解説

web_cal_fwd は１つ先のページに進み、FORWARD ボタンにフォーカスを設定します。URL ヒストリに履歴が無い場合、ページ遷移は発生しません。

本 API の処理結果の通知方法は tmout の値により次のようになります。
タイムアウトなし（tmout = TMO_FEVR）で本 API を発行した場合、発行元のタスクは、処理が完了するまで待ち状態となります。この場合正の戻値として処理結果が通知されます。
タイムアウトあり（tmout = 1 ～ 0x7fffffff）で本 API を発行した場合、指定した時間が経過しても処理が完了しなければ、E_TMOUT エラーが返ります。tmout は、NORTi の MSEC マクロを用いてミリ秒単位で時間指定することもできます。
ノンブロッキング（tmout = TMO_NBLK）で本 API を発行した場合、API はすぐにリターンし、処理の完了はコールバック関数で通知されます。コールバック関数については、web_ini の説明を参照して下さい。

本 API をタイムアウトあり、またはなしで実行したときの処理結果の値は次の通りです。

WEB_STA_DNE	正常終了
WEB_STA_ABT	読込み中止された
WEB_STA_CON	リモートホストに接続失敗
WEB_STA_NOT	処理は行われなかった
WEB_STA_INC	URL が不正または接続失敗
WEB_STA_IMG	画像の読込みに失敗

正常終了しなかった要因が複数ある場合、処理結果の値は全ての要因の OR 値となります。

戻値

正の値　　処理結果(tmout = TMO_FEVR または tmout = 1 ～ 0x7fffffff のとき)

E_OK	正常終了(tmout = TMO_NBLK のとき)
E_NOEXS	ブラウザタスクが生成されていない
E_NOID	イベントフラグ ID が不足
E_CTX	割込みハンドラから発行、または、ディスパッチ禁止状態で待ち実行
E_SYS	管理ブロック用のメモリが確保できない
E_TMOUT	タイムアウト

例

```
#include “noweb.h”
      :
ER ercd;
      :
ercd = web_cal_fwd (5000/MSEC);
if(ercd != WEB_STA_DNE)
      :
```

5.13 web_cal_stp

機能

ページ読込み中止

形式

```
ER web_cal_stp(void);
```

解説

web_cal_stp はページ読込みを中止し、STOP ボタンにフォーカスを設定します。

戻値

E_OK	正常終了
E_NOEXS	ブラウザタスクが生成されていない

例

```
#include “noweb.h”
      :
ER ercd;
      :
ercd = web_cal_stp ();
if(ercd != E_OK)
      :
```

5.14 web_url_jmp

機能
指定 URL へページ遷移

形式
ER web_url_jmp(const B *url, TMO tmout)
url　　URL 文字列
tmout　　タイムアウト値

解説
web_url_jmp は url で指定した URL へページ遷移し、URL 入力エディットボックスにフォーカスを設定します。

本 API の処理結果の通知方法は tmout の値により次のようになります。
タイムアウトなし（tmout = TMO_FEVR）で本 API を発行した場合、発行元のタスクは、処理が完了するまで待ち状態となります。この場合正の戻値として処理結果が通知されます。
タイムアウトあり（tmout = 1 ～ 0x7fffffff）で本 API を発行した場合、指定した時間が経過しても処理が完了しなければ、E_TMOUT エラーが返ります。tmout は、NORTi の MSEC マクロを用いてミリ秒単位で時間指定することもできます。
ノンブロッキング（tmout = TMO_NBLK）で本 API を発行した場合、API はすぐにリターンし、処理の完了はコールバック関数で通知されます。コールバック関数については、web_ini の説明を参照して下さい。

本 API をタイムアウトあり、またはなしで実行したときの処理結果の値は次の通りです。
WEB_STA_DNE　　正常終了
WEB_STA_ABT　　読込み中止された
WEB_STA_CON　　リモートホストに接続失敗
WEB_STA_NOT　　処理は行われなかった
WEB_STA_INC　　URL が不正または接続失敗
WEB_STA_IMG　　画像の読込みに失敗

正常終了しなかった要因が複数ある場合、処理結果の値は全ての要因の OR 値となります。

戻値

正の値	処理結果(tmout = TMO_FEVR または tmout = 1 ～ 0x7fffffff のとき)
E_OK	正常終了(tmout = TMO_NBLK のとき)
E_NOEXS	ブラウザタスクが生成されていない
E_NOID	イベントフラグ ID が不足
E_CTX	割込みハンドラから発行、または、ディスパッチ禁止状態で待ち実行
E_SYS	管理ブロック用のメモリが確保できない
E_TMOUT	タイムアウト

例

```
#include "noweb.h"
      :
ER ercd;
      :
ercd = web_url_jmp ("http://www.mispo.co.jp/", 5000/MSEC);
if(ercd != WEB_STA_DNE)
      :
```

5.15 web_shw_mse

機能

マウスポインタの表示・非表示

形式

```
ER web_shw_mse(BOOL flag)
flag  = TRUE  マウスポインタ表示
      = FALSE マウスポインタ非表示
```

解説

web_shw_mse はT_WEB_INI構造体のshowmouseメンバにFALSEを設定してブラウザを起動し、マウスポインタが非表示状態のときに flag = TRUE で実行するとマウスポインタを表示します。逆に、マウスポインタが表示状態のときにflag = FALSEで実行するとマウスポインタを消去します。

戻値

E_OK	正常終了
E_NOEXS	ブラウザタスクが生成されていない

例

```
#include “noweb.h”
      :
ER ercd;
      :
ercd = web_shw_mse(TRUE);
if(ercd != E_OK)
      :
```

6.機能一覧

Web Browser for NORTi は以下の機能をサポートします。

・HTTP1.0, 1.1
・HTML3.2, 4.0 の一部
・14×14 ドット日本語フォントによる日本語表示
・GIF, JPEG, BMP ファイルの表示
・Cookie
・コンテンツ Cache
・URL ヒストリ
・オンスクリーンキーボード

7. サポートタグ

Web Browser for NORTi は、HTML3.2 および HTML4.0 の一部をサポートします。
サポートするタグの一覧は以下の通りです。

HTML3.2

タグ名	備考
A	
ADDRESS	
AREA	
B	
BASE	
BASEFONT	
BIG	
BLOCKQUOTE	
BODY	
BR	
CAPTION	
CENTER	
CITE	
CODE	
DD	
DFN	
DIR	
DIV	
DL	
DT	
EM	
FONT	
FORM	
H1	
H2	
H3	
H4	

H5	
H6	
HR	
I	
IMG	GIF、JPEG、BMP
INPUT	
KBD	
LI	
MAP	
MENU	
META	
OL	
OPTION	
P	
PRE	
SAMP	
SELECT	
SMALL	
STRONG	
SUB	
SUP	
TABLE	
TD	
TEXTAREA	
TITLE	
TR	
TT	
UL	
VAR	

HTML4.0

タグ名	備考
ACRONYM	
DEL	
FRAME	
FRAMESET	
INS	
NOFRAMES	
NOSCRIPT	
Q	

拡張

タグ名	備考
NOBR	
NOEMBED	

Web Browser for NORTi ユーザーズガイド

株式会社ミスポ	http://www.mispo.co.jp/
一般的なお問い合わせ	sales@mispo.co.jp
技術サポートご依頼	norti@mispo.co.jp