

Sausalitoで遊ぶ Cobalt Appliance Programming入門

2002年9月8日
NTT DoCoMo Kansai, Inc.
Masahiro Watanabe

Windowsマシン、UnixマシンとCobaltの違い

	Windows	一般的なUnix	Cobalt
ディスプレイおよびキーボード	要	不要 (接続可)	不要 (LCDコンソールあり)
操作の手段	GUI (ウィンドウシステム)	CUI (コマンドラインインターフェース)	GUI (LCDコンソールおよびWebブラウザ)
遠隔メンテナンス	不可	可 (CUI)	可 (GUI)



Cobalt ServerはUNIXの省スペース性とWebブラウザを使ったGUIでの簡易な操作を兼ね備えたServer Applianceである。

Sausalitoとはなにか

Sausalitoとは、Cobalt Serverのソフトウェアに一貫した操作性を与えるために開発された、新アーキテクチャである。

	従来のCobalt	Sausalito
管理画面の構成	画面ごとにPerlのCGIで作成	XMLを使って自動生成されたメニューと、PHPのUIFCライブラリを利用して作成した画面から構成
管理画面の認証方法	HTTPのBASIC認証を利用	Sausalito内の独自認証を利用 (ユーザー権限ごとにメニュー構成を変更することが可能)
国際版対応	各国版毎にPerlのソースを書き換えてリリース	表示されるメッセージをStringファイルとして独立させ、Stringファイルを言語毎に用意することにより対応 (ソースは各国語版同一)
システムリソースへのアクセス	Perlにてアクセス	サーバ管理用デモン (CCE) 経由でアクセス

Sausalitoの特長

- 柔軟な機能拡張が可能
 - XMLを用いたメニュー自動生成処理 (Navigation Manager)により、メニューツリーへのメニュー追加が他の既存のメニューに影響を与えずに可能
- 各国語版パッチリリースに要する期間の短縮および信頼性の向上
 - 各国語版間でソースが共通であるため、シンプルなバージョン管理が可能
- ユーザインターフェースのLook & Feelの統一性の保持が可能
 - ユーザインターフェース用API (UIFC) が公開されているため、サードパーティのアプリケーションにおいてもCobaltオリジナル機能と同一のLook & Feelを実現可能

Sausalitoの特長 (つづき)

- サードパーティのアプリケーションにおいてもSausalitoのユーザ認証の利用が可能
 - サードパーティのアプリケーションにおいてもNavigation ManagerおよびUIFCを用いることによりSausalitoのユーザ認証を利用可能 (独自の認証機構を準備する必要がない)
- Sausalito対応機種間でのアプリケーションの移植が容易
 - ハードウェアやシステム情報へのアクセス用のAPI (CCE) が提供されていることにより、機種間のアプリケーションの移植が従来に比べ容易であると思われる。

Sausalito版Cobalt Server

Sausalitoアーキテクチャを採用しているCobalt Serverは以下のとおり

- Sun Cobalt Qube3
- NTT DoCoMo MMQUBE2 (OEM版Qube3)
- Sun Cobalt Qube3 Plus
- Sun Cobalt RaQ550
- Sun Cobalt Control Station

(2002年9月現在)

備考:

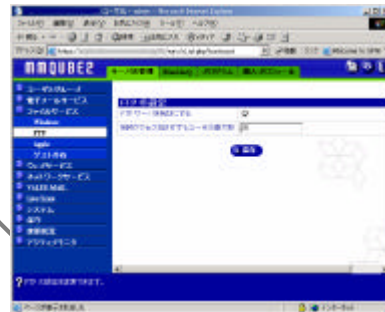
Sun Cobalt XTRIは部分的にSausalitoが使われている。

Sausalitoの画面構成 (ログイン画面)



この画面はMMQUBE2のものである。

Sausalitoの画面構成 (ログイン後)



この画面はMMQUBE2のものである。

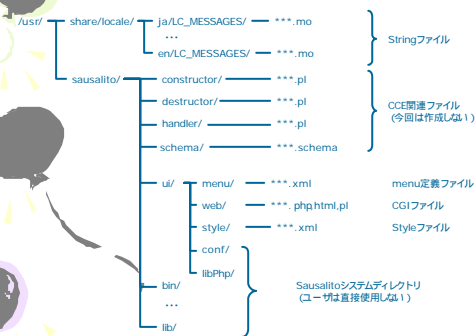
Sausalito対応アプリケーション作成の準備 用意するもの

- Sausalito アーキテクチャのサーバ
 - 管理者権限が必要
- IE5以上が動作するPC
- Telnetクライアント
 - TeraTerm (フリーウェア)がおすすめ
- FTPクライアントソフト
 - FFFTP(フリーウェア)がおすすめ
- tar, gzip, rpmに対応した圧縮・解凍ソフト
 - Explzh(シェアウェア)+tar32.dllがおすすめ
- テキストエディタ
 - 秀丸(シェアウェア)がおすすめ

Sausalito対応アプリケーション作成の準備

- あるとうれしい予備知識 (なくても可?)
 - TelnetおよびShellを使ったことがある
 - htmlのソースを見たり書いたことがある
 - xmlについて知っている
 - rpmのSPEC ファイルを書いたことがある
 - Perl, PHP, Awkのソースを見たことがある
 - EUC, SJISコードとは何のことが知っている
 - vi, moreを使ったことがある
 - '!'と書いてrootと読める

Sausalitoのディレクトリ構成 (後で使うよ)



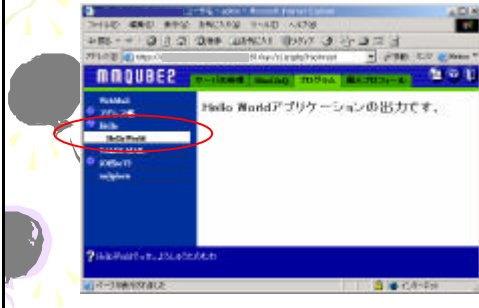
本日のお題

課題 1) Sausalitoメニューに "HelloWorld" アプリを追加する

課題 2) Sausalitoのメッセージを大阪弁に変更する

課題 1 :メニュー追加

Sausalitoメニューに "HelloWorld" アプリを追加しよう



STEP 1 :メニューの構造を探ってみる

/usr/sausalito/ui/menuの下を覗く

```
[admin admin]$ cd /usr/sausalito/ui/menu
[admin menu]$ ls -F
VM/ base/ ioffice/ live/ palette/

[admin menu]$ cd base
[admin base]$ ls -F
addressbook/ carmel/ firewall/ network/ telnet/ wizard/ am/
dhcpd/ ftp/ power/ time/ workgroup/ apache/ disk/
import/ quotastats/ user/ appshare/ dns/ ldap/ snmp/
webmail/ backup/ email/ maillist/ swupdate/ webstats/ cache/
fileshare/ multidrop/ system/ winshare/

[admin base]$ cd am
[admin am]$ ls -F
amSettings.xml amStatus.xml monitor.xml monitorLight.xml
```

STEP 2 :xm ファイルの構造を推測する

既存の xm ファイルを見てみる

ソース1 :monitor.xml

```
<item id="base_monitor"
label="[[base-am.activeMonitor]]"
description="[[base-am.activeMonitor_help]]" >
  <parent id="base_administration" order="80"/>
</item>
```

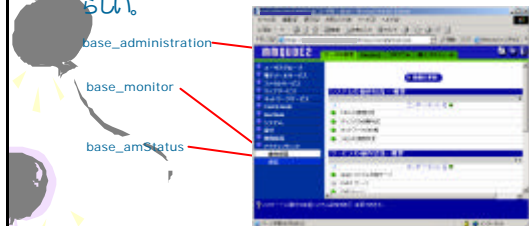
ソース2 :amStatus.xml

```
<item id="base_amStatus"
label="[[base-am.amStatusMenuName]]"
description="[[base-am.amStatusMenuDesc]]"
url="/base/am/amStatus.php" >
  <parent id="base_monitor" order="10"/>
</item>
```

STEP 2 続き

xm ファイルの構造を推測する(id属性)

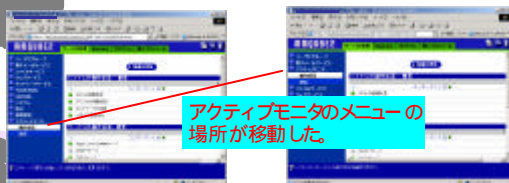
- item エLEMENTのid属性は自分自身のIDを指しているらしい。
- parent ELEMENTのid属性は自分の親のIDを指しているらしい。



STEP 2 続き

xm ファイルの構造を推測する(order属性)

- parent ELEMENTのorder属性は同じ親に属しているものの間の順序を指しているらしい
- monitor.xmlのparent ELEMENTのorder属性を80から25に変更してみる



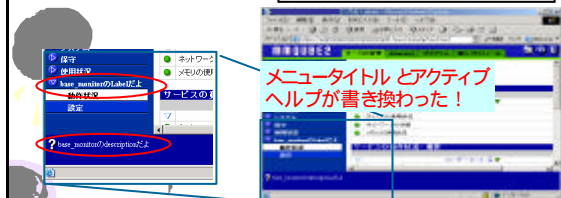
注意 自分で試す場合、ファイルを書き換えるときは、元のファイルを必ずバックアップしてから行うこと!

STEP 2 続き

xm ファイルの構造を推測する(label,description属性)

monitor.xmlのitem ELEMENTのlabel description属性を右のように書き換えてみる

```
<item id="base_monitor"
label="base_monitorのLabelだよ"
description="base_monitorのdescriptionだよ"
  <parent id="base_administration" order="80"/>
</item>
```



注意 自分で試す場合、ファイルを書き換えるときは、元のファイルを必ずバックアップしてから行うこと!

STEP2: 続き

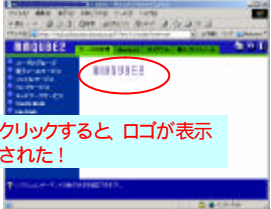
xmlファイルの構造を推測する(url属性)

amStatus.xml ファイルのitemエレメントのURL属性のURLを左上のロゴのビットマップのURLに書き換えてみる。
URLのドキュメントルートは /usr/sausalito/ui/webである。

```

amStatus.xml
<item id="base_amStatus"
label="[[base-am.amStatusMenuName]]"
description="[[base-am.amStatusMenuDesc]]"
url="/lib/image/topLogo.gif" >
<parent id="base_monitor" order="10"/>
</item>

```



クリックすると、ロゴが表示された!

注意: 自分で試す場合、ファイルを書き換えるときは、元のファイルを必ずバックアップしてから行うこと!

STEP2: まとめ

xmlファイルの構造

エレメント	属性	意味
item	id	自分自身のメニュー項目ID (すべてのメニュー項目でユニークである必要がある)
item	label	メニューに表示される文字列(日本語も使用可、S-JISで指定する必要がある)
item	description	アクティブヘルプに表示される文字列(日本語も使用可、S-JISで指定する必要がある)
item	url	メニュー項目クリック時に表示されるURL
parent	id	自分がぶら下がるメニュー項目ID
parent	order	メニューの並び(小さいほどまたは、右に表示される)

```

amStatus.xml
<item id="base_amStatus"
label="[[base-am.amStatusMenuName]]"
description="[[base-am.amStatusMenuDesc]]"
url="/base/am/amStatus.php" >
<parent id="base_monitor" order="10"/>
</item>

```

STEP3: メニューツリーを確認する

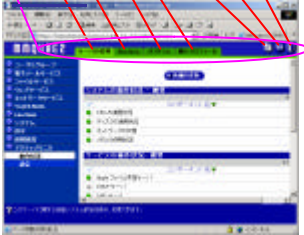
管理者ログイン時のメニューツリーを知る

- メニューツリーのTOPはrootである。

```

root
├── base_administration
├── base_software
├── base_programs
├── base_personalProfile
├── base_updateLight
├── base_monitorLight
└── base_logout

```



STEP4: 課題のHelloメニューをチェックする

- Helloメニューは プログラム タブの中にある。
- base_programsの子メニューとなる
- HelloWorldメニューは Helloメニューの中にある
- HelloメニューのIDの子メニューとなる。
- HelloWorldアプリケーションは HelloWorldメニューから呼び出される。

```

base_programs
├── base_hello
└── base_helloWorld

```



STEP5: Helloメニューを作成する

これまでの情報を用いて、Helloメニューを作成する

```

hello.xml
<item id="base_hello"
label="Hello"
description="Helloメニューです。" >
<parent id="base_administration" order="21"/>
</item>

```

```

helloworld.xml
<item id="base_helloWorld"
label="HelloWorld"
description="HelloWorldアプリケーションです。"
url="hello/hello.htm" >
<parent id="base_hello" order="10"/>
</item>

```

STEP5: 続き

どこにXMLファイルを置くか?

- Sausalitoは/usr/sausalito/ui/menuの下フォルダを再帰的に探索する。



アプリごとにxmlファイル配置のためのディレクトリを分けることが望ましい。

今回は、ui/menuの下に、helloというディレクトリを作成し、その中に配置する。

注意: Sausalitoは、シンボリックリンクのディレクトリは探索しないため、実体のディレクトリ内に入れる必要がある

STEP6: hello.htmを作成する

- メニューからリンクされるHTMLファイルを作成する

```
hello.htm
<HTML>
<body>
Hello World アプリケーションの出力です。
</body>
</html>
```

HTMLファイルの配置場所は/usr/sausalito/ui/web/hello/

課題 1 :完成 !

The screenshot shows a web browser window with the URL 'http://localhost:8080/'. The page content is 'Hello World アプリケーションの出力です。' (Output of the Hello World application).

課題 2 :メッセージの書き換え

課題 2) Sausalitoのメッセージを大阪弁に変更しよう
- 電源切断手順画面を大阪弁に変更する

The screenshot shows the application's power-off screen. The text is in Osaka dialect, such as '電源を切るには、この画面から電源ボタンを押してください。' (To turn off the power, please press the power button from this screen).

STEP1: メッセージのありかを探す

- /usr/share/locale/ja/LC_MESSAGES/の下を覗く

```
admin admin] $ cd /usr/share/locale/ja/LC_MESSAGES/
(admin LC_MESSAGES) $ ls
base-fileshare.mo base-sys.mo base-firewall.mo base-system.mo base-ftp.mo base-teinet.mo
base-import.mo base-lime.mo base-lcd.mo base-user.mo base-ldap.mo base-webmail.mo
base-mailbox.mo base-maillist.mo base-webstats.mo base-addressbook.mo base-memory.mo
base-wineywork.mo base-am.mo base-modem.mo base-winsare.mo base-apache.mo
base-multioptop.mo base-wizard.mo base-appleshare.mo base-network.mo base-workgroup.mo
base-backup.mo group.mo base-cache.mo base-power.mo palette.mo base-carmel.mo
base-power.mo.org sharutils.mo base-cce.mo base-quotastats.mo swatch.mo base-dhclient.mo
base-sauce-basic.mo tar.mo base-dhcpd.mo base-services.mo textutils.mo base-disk.mo
base-snm.mo trueBlue.mo base-dns.mo base-ssl.mo util-linux.mo base-email.mo base-swupdate.mo
(admin LC_MESSAGES) $
```

- * .mo というファイルが多数存在する。
- これらはStringファイルと呼ばれるものである。

STEP2: Stringファイルとは

- Stringファイルとは、文字列にタグ(ID)をつけて格納したもの。
- Stringソースファイル(*.po)から生成される。
- 国際化ライブラリにて使用される。

```

Stringソースファイル(*.po)
  | encode
  v
Stringファイル(*.mo)
  | decode
  v
Stringソースファイル(*.po)

```

msgfmt コマンド (encode) / msgunfmt コマンド (decode)

STEP3: Stringソースファイルの書式

```
msgid "ユニークなID1"
msgstr "メッセージ1"
(空行)
msgid "ユニークなID2"
msgstr "メッセージ2"
(空行)
.....
msgid "ユニークなIDn"
msgstr "メッセージn"
```

msgid: メッセージのIDをあらわす
msgstr: メッセージの文字列。日本語可。文字コードはS-JISを使用する。

STEP4: msgfmt, msgunfmt の使い方

- msgfmtコマンド
 - 書式 :msgfmt xxx.po [-o xxx.mo]
 - Stringソースファイル xxx.po から Stringファイル xxx.mo を生成する。-o オプションが無指定の場合は message というファイル名のファイルに出力する。
- msgunfmtコマンド
 - 書式 msgunfmt xxx.mo [-o xxx.po]
 - Stringファイル xxx.moから Stringファイル xxx.po を生成する。-o オプションが無指定の場合は、標準出力に出力する。

STEP5: *mo ファイルから*poファイルを作成

- Stringファイルから Stringソースファイルを作成する

```
[root LC_MESSAGES]# msgunfmt base-power.mo -o /tmp/base-power.po
[root LC_MESSAGES]# head /tmp/base-power.po
msgid "askRebootConfirmation"
msgstr "サーバを再起動してよろしいですか?"

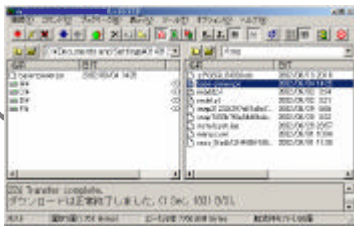
msgid "power"
msgstr "電源"

msgid "powerHelp"
msgstr "サーバを再起動できます。"

msgid "reboot"
.....
```

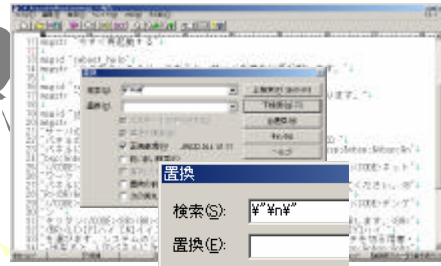
STEP6: *poファイルをPCに転送する

- STEP5で生成したbase-power.poをFFFTPを利用してPCに転送する。
 - ASCIIモードで転送する。
 - 文字コードは、変換しない。



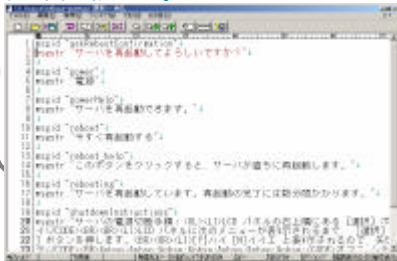
STEP7: *.poファイルを編集する

- msgunfmtコマンドで生成された *.poファイルは、文字列が長い場合80文字で改行されているため、文字列を一本にする。



STEP7: 続き (関西人用)

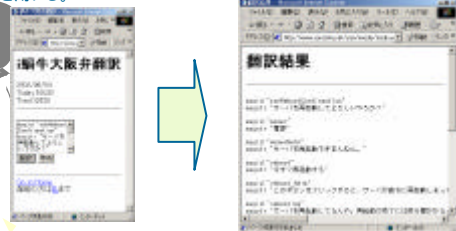
- 文字列をチェックし、変な日本語をすべて大阪弁に書き換える。



編集が完了したら、base-power-osaka.po というファイル名で保存する。

STEP7: 続き (関西人以外用)

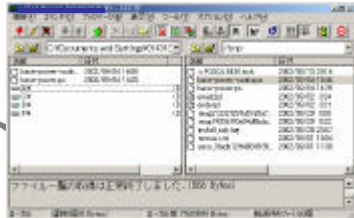
- インターネット上の大阪弁変換フィルタ等を用いて、メッセージを大阪弁に変換する。
例 <http://www.yorosiku.net/yan/imode/osaka.html> を用いる。



編集が完了したら、base-power-osaka.po というファイル名で保存する。

STEP8: 編集後の*.poファイルをアップする。

- 編集後のbase-power-osaka.poファイルをサーバにFTPする。
- ASCIIモードで転送する。
- 文字コードは、変換しない!



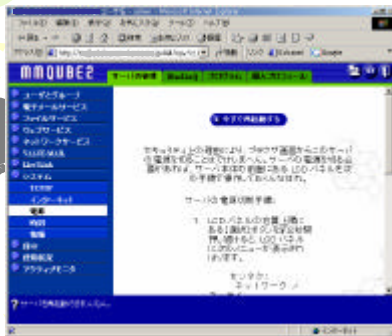
STEP9: *.moファイルを作成し入替える

- 書き換えた、base-power-osaka.poファイルを元に、base-power.moファイルを作成し、元のファイルと入替えを行う

```
[root /tmp]# msgfmt base-power-osaka.po -o base-power.mo
[root /tmp]# cd /usr/share/locale/ja/LC_MESSAGES/
[root LC_MESSAGES]# mv base-power.mo base-power.mo.org
[root LC_MESSAGES]# cp /tmp/base-power.mo .
[root LC_MESSAGES]#
```

注意 自分で試す場合、ファイルを書き換えるときは、元のファイルを必ずバックアップしてから行うこと!

課題 2 :完成 !



おまけ 1 :メニュー内でStringファイルを使う

hello.po

```
msgid "helloworld"
msgstr "Hello だぜ"
```

→ /usr/share/locale/ja/LC_MESSAGES/hello.mo

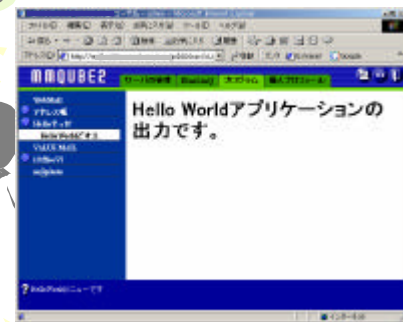
hello.xml

```
<item id="base_hello" label="[hello.helloworld]"
description="HelloWorldメニュー です type="monitorOn" >
<parent id="base_programs" order="21"/>
</item>
```

helloworld.xml

```
<item id="base_hello_helloApp" label="[hello.helloworldgroup]"
description="HelloWorld だぜ。よろしゅうなだぜ"
url="/hello/hello.htm" >
<parent id="base_hello" order="21"/>
</item>
```

おまけ 1 :出力結果



おまけ 2 :日本語文字が化ける場合

- S-JIS漢字中に含まれる0x5c(¥)キャラクターコードをエスケープすることにより回避する。

? enadd.pl :漢字中に0x5cが含まれている場合、その前に0x5cを挿入する。(msgfmtの前に使用)

? endel.pl :漢字中に0x5cが2回連続して含まれている場合、0x5cをひとつ削る。(msgunfmtの後に使用)

enadd.pl

```
#!/usr/bin/perl
while(<>){
s/%% /%% /g;
print;
}
```

endel.pl

```
#!/usr/bin/perl
while(<>){
s/%% /%% /g;
print;
}
```

おまけ3:メニューツリーのrootを変える

- ログイン後特定のメニューをrootにして、Navigation Managerを立ち上げる

`http://サーバ:444/login.php?target =
/nav/cList.php%3Froot=メニュー
ID %3FCommFrame=追加スクリプト`

例: `http://192.168.0.30:444/login.php?
target=/nav/cList.php%3Froot=base
_administration`

おまけ3:結果

