

Ploneの外観（スキン）を変える

—テーマプロダクトの作成—

2009/4/25

安田善一郎 (Ciel Serein / Plone研究会)





Ploneの外観（スキン）を変える ーテーマプロダクトの作成ー

はじめに



本日のゴール

- 素の状態のPlone3の見た目をカスタマイズします。
 - モックアップはこんなかんじです。（別ウインドウ）
これを素のPlone3に適用して行きます
- ZMI（管理インターフェース）で設定を変更するのではなく、
プロダクト（パッケージ）として開発します。
 - プロダクトはインストール／アンインストールが可能です。
 - ローカル環境で開発し、十分に作り込んでからこれを本番に適用する、というよ
うな手順がとれるようになります。
 - サーバー変更時などの移行作業が簡単になります。



イントロダクション

- 対象者
 - Ploneサイトのカスタマイズに興味のあるかた
- 前提知識など
 - XHTML/CSS
 - zopeのページテンプレート
 - tal 構文を見ても驚かない程度
 - MTなど他のテンプレートの経験があればなんとなくわかる



イントロダクション

- 環境など
 - Ploneがインストールできる環境
 - 今回はMac OS X 10.5.6で行います
 - ターミナルが使える
 - ところどころでコマンドを入れる必要があります
 - エディタ
 - ほとんどの作業はテキストファイルの編集です
 - ブラウザ
 - ZMIからパッケージをインストールしたり、動作検証に使用します

作業の流れ



- I. 開発環境の準備
- II. テーマプロダクトのスケルトンを作成する
- III. Viewlet コンポーネントの表示を制御する
- IV. Viewlet コンポーネントをカスタマイズする
- V. カスタムのスタイルシートを適用する



本日の資料

I. 発表資料「Ploneの外観（スキン）を変える」

http://ciel-serein.jp/study/wpd2009/wpd2009_plone3_theme.pdf/view

II. テーマプロダクト本体

<http://ciel-serein.jp/study/wpd2009/wpdj-theme.zip/view>

落とせる方はお手元で見ながらどうぞ。



Ploneの外観（スキン）を変える ーテーマプロダクトの作成ー

I. 開発環境の準備



I. 開発環境の準備 (1)

- Plone.orgからダウンロードしたインストーラを使用
<http://plone.org/products/plone>
 - buildoutベースのディレクトリ構造でセットアップ
 - Plone 3.2.2, Zope 2.10.7, Python 2.4.4
 - pasterなどのツールもセットでセットアップされる

※11月のときはMacでもUnified installerを使っていました。

※今回のデモは 3.2.1ベースです。



I. 開発環境の準備 (2)

デバッグモードの設定

```
$~/Plone-3.1/zinstance/buildout.cfg
```

```
[instance]  
debug-mode = on
```

```
$cd ~/Plone-3.1/zinstance
```

```
$ ./bin/buildout
```

これを実行することで反映される

※ 「\$~/Plone-3.1」 はインストールディレクトリ。環境に応じて読み替えてください。



I. 開発環境の準備 (3)

zopeの開始

```
$ ~/Plone-3.1/zinstance/bin/instance fg
```

メッセージがコンソールに逐一表示されるので安心
ctrl+Cで終了
.pyファイルを変更した場合は再起動

cssとjsがすぐに反映されるようにする

ZMI画面

```
[Ploneサイト名] > portal_css or portal_javascripts
```

「debug」をチェックする

Ploneの外観（スキン）を変える ーテーマプロダクトの作成ー

II. テーマプロダクトの スケルトンを作成する



II. テーマプロダクトのスケルトンを作成する

- 1) egg スケルトンを作る
- 2) できたeggをzopeにインストールする
- 3) Ploneサイトに適用する

egg … pythonのプロダクトパッケージの形式
(だと理解してます)



II. スケルトン作成

1) egg スケルトンを作る

```
$ cd ~/Plone-3.1/zinstance/src  
$ ../bin/paster create -t plone3_theme
```

```
Enter project name: wpdj.theme
```

```
Variables:
```

```
egg: wpdj.theme
```

```
package: wpdjtheme
```

```
project: wpdj.theme
```

```
Enter namespace_package (Namespace package  
(like plonetheme)) ['plonetheme']: wpdj
```

```
Enter package (The package contained namespace package  
(like example)) ['example']: theme
```

```
Enter skinname (The skin selection to be added to 'portal_skins'  
(like 'My Theme')) ['']: wpdjSkin
```

この名前はあとあと使います

/zinstance/src の下に
フォルダがたくさんできます

II. スケルトン作成

2009/4/25

2)eggをzopeにインストール



1. ~/Plone-3.1/zinstance/buildout.cfg を修正する

```
[buildout]
eggs = wpdj.theme
develop = src/wpdj.theme
[instance]
zcml = wpdj.theme
```

2. buildoutを実行して修正を反映する。

```
$cd ~/Plone-3.1/zinstance
$ ./bin/buildout
```

II. スケルトン作成

2009/4/25

3) Ploneサイトに適用する



再起動するとZMIの

/[Plone site]/portal_quickinstaller にでてくるので、インストールしてみます。

※この後も、.xmlや.zcmlを変更したら再インストールが必要。

installすると、見事にstylesheetがすべてはずれます。

=> [your theme package]/skins/wpdj_theme_styles/以下の
public.css.dtml portlets.css.dtml base.css.dtml
これらを削除すると、基本のスタイルシートが生きるようになる。



Ploneの外観（スキン）を変える ーテーマプロダクトの作成ー

III. Viewlet コンポーネントの 表示を制御する



III. Viewlet コンポーネントの表示を制御する

Viewlet (ビューレット)

Plone 3 has switched to use Zope 3 viewlet components instead of the old macro include approach.

Since Plone 3.0, `main_template.pt` calls viewlet managers instead of METAL macros

=> メインテンプレートの見通しがよくなりました

参考 : plone.org > Documentation > Tutorial > Customizing-main-template-viewlets > Introduction



III. Viewlet コンポーネントの表示を制御する

- 1) コンポーネントを非表示にする
 - Personal Bar, SearchBox, Colophonを非表示に
- 2) コンポーネントの表示順を入れ替える
 - ログ→global_sections→site_actionsの順にします

III. Viewlet コンポーネントの表示を制御する

1.) コンポーネントを非表示にする

Personal Bar, SearchBox, Colophonを非表示にする

(1) 表示を制御するxmlを編集します。

~/Plone-3.1/zinstance/src/wpdj.theme/wpdj/theme/profiles/default/viewlets.xml

```
<object>
```

```
<hidden manager="plone.portalheader" skinname="wpdjskin">  
  <viewlet name="plone.searchbox" />  
</hidden>
```

```
<hidden manager="plone.portaltop" skinname="wpdjskin">  
  <viewlet name="plone.personal_bar" />  
</hidden>
```

```
<hidden manager="plone.portalfooter" skinname="wpdjskin">  
  <viewlet name="plone.colophon" />  
</hidden>
```

```
</object>
```



III. Viewlet コンポーネントの表示を制御する

1.) コンポーネントを非表示にする

Personal Bar, SearchBox, Colophonを非表示にする



(2) ZMIからプロダクトをリインストールする。

=> 反映されました。

が、**アンインストールの対応をしていないので、アンインストールしても戻りません。**

viewlets.xmlなど、適宜バックアップを取りながら進めてください

参考：

viewletManager, viewletの名前がわからなかったら

URLの後ろに@@manage-viewletsつけると見ることができます。



III. Viewlet コンポーネントの表示を制御する

2) コンポーネントの表示順を入れ替える

ロゴ→global_sections→site_actionsの順にします

(1) ふたたび、表示を制御するxmlを編集します。

~/Plone-3.1/zinstance/src/wpdj.theme/wpdj/theme/profiles/default/viewlets.xml

<order>タグの中で対象となる <manager>を記述し、その中に <viewlet>を記述します。

```
<order manager="plone.portalheader" skinname="wpdjskin"
  based-on="Plone Default">
  <viewlet name="plone.logo" insert-before="*" />
  <viewlet name="plone.global_sections" insert-after="plone.logo" />
  <viewlet name="plone.site_actions"
    insert-after="plone.global_sections" />
</order>
```

#insert-before="*"とすると、なによりも最前列、という意味になります。

#記述に矛盾があると、いきなり表示されなくなることも. . .



Ploneの外観（スキン）を変える ーテーマプロダクトの作成ー

IV. Viewlet コンポーネントを カスタマイズする



IV. Viewlet コンポーネントをカスタマイズする

- 1. パンくず(path_bar) をコンテンツの下に移動する
- 2. ロゴ (logo) を変更する
- 3. site_actions をフッタに移動

いずれも、新たにviewletを作成し、
それを前出のviewlets.xmlを使って表示させます。
(オリジナルのviewletは隠します)。



IV. Viewlet コンポーネントをカスタマイズする

(1) 新たにviewletを作成するためにclassを記述

[my package]/browser/viewlets.py にclass を追加する。

(2) 新viewlet表示のためのテンプレート(ptファイル)を準備

(3) 新viewletを /browser/configure.zcml に登録。

(4) 表示の制御のために /profiles/default/viewlets.xmlを変更

(1) オリジナルのパンくずのviewlet "plone.path_bar" を隠す。

(2) 新しく作成した viewletを表示する。

IV. Viewlet コンポーネントをカスタマイズする

1) パンくず(path_bar) をコンテンツの下に移動する

(1) 新たにviewletを作成するためにclassを記述

/profiles/default/viewlets.xml にある“plone.path_bar”をおきかえます。

#念のため、pythonのclassです...



[my package]/browser/viewlets.py にclass を追加する。

pathViewlet という名前にする。

ここでは、path_barの機能を実装しなければならないので、以下ページのサンプルを参考にして元のコードをそのまま移植します。

6.8.7. Path Bar (Portal Breadcrumbs)

http://plone.org/documentation/manual/theme-reference/elements/visibleelements/plone.path_bar

#必要であればなんらかのロジックもここに書くことができる。

IV. Viewlet コンポーネントをカスタマイズする

1) パンくず(path_bar) をコンテンツの下に移動する

サンプルコード (抜粋) [my package]/browser/viewlets.py



```
# 以下は ptをレンダリングするために必要
from Products.Five.browser.pagetemplatefile import ViewPageTemplateFile

# カスタマイズしたviewで利用する (継承する) classをインポートする。
from plone.app.layout.viewlets.common import PathBarViewlet

# viewletを定義する。
# browser/configure.zcmlに登録した名前とclassの名前が一致していることが必要
# PathBarViewletの関数をそのまま利用して、wpdpath_bar.ptをレンダリングする
class pathViewlet(PathBarViewlet):
    render = ViewPageTemplateFile('wpdpath_bar.pt')
```

IV. Viewlet コンポーネントをカスタマイズする

1) パンくず(path_bar) をコンテンツの下に移動する

(2) 新viewlet表示のためのテンプレート(ptファイル)を準備します。



ZMI で

[site root] > portal_view_customizations から、path_barを探す。

plone.app.layout.viewlets/path_bar.ptをそのまま

[my package]/browser/wpdpath_bar.pt として保存する

#見た目の変更のためにhtml上のclassを変えたいなどの場合、これを変更する。

IV. Viewlet コンポーネントをカスタマイズする

1) パンくず(path_bar) をコンテンツの下に移動する

サンプルコード [my package]/browser/wpdpath_bar.pt



```
<div id="portal-breadcrumbs"
  i18n:domain="plone">

  <span id="breadcrumbs-you-are-here" i18n:translate="you_are_here">You
are here:</span>
  <a i18n:translate="tabs_home" tal:attributes="href view/navigation_root_url">Home</a>
  <span tal:condition="view/breadcrumbs" class="breadcrumbSeparator">
    <tal:ltr condition="not: view/is_rtl">&rarr;</tal:ltr>
    <tal:rtl condition="view/is_rtl">&raquo;</tal:rtl>
  </span>
  <span tal:repeat="crumb view/breadcrumbs"
    tal:attributes="dir python:view.is_rtl and 'rtl' or 'ltr'">
  <tal:last tal:define="is_last repeat/crumb/end">
    <a href="#"
      tal:omit-tag="not: crumb/absolute_url"
      tal:condition="python:not is_last"
      tal:attributes="href crumb/absolute_url"
      tal:content="crumb/Title">
      crumb
    </a>
    <span class="breadcrumbSeparator" tal:condition="not: is_last">
      <tal:ltr condition="not: view/is_rtl">&rarr;</tal:ltr>
      <tal:rtl condition="view/is_rtl">&raquo;</tal:rtl>
    </span>
    <span tal:condition="is_last"
      tal:content="crumb/Title">crumb</span>
  </tal:last>
</span>

</div>
```

小さくてすみません。

サンプルといっても、元と同じなので...

IV. Viewlet コンポーネントをカスタマイズする

1) パンくず(path_bar) をコンテンツの下に移動する

(3) 新viewletを /browser/configure.zcml に登録。



[my package]/browser/configure.zcml に新たに追加するviewletを定義する

viewletの名称は "wpdjskin.path_bar"

classは [my package]/browser/viewlets.py に追加したclass名とあわせる。

サンプルコード (抜粋) [my package]/browser/configure.zcml

```
<!-- パンくずリスト -->
<browser:viewlet
    name="wpdjskin.path_bar"
    manager="plone.app.layout.viewlets.interfaces.IPortalFooter"
    class=".viewlets.pathViewlet"
    permission="zope2.View"
/>
```



IV. Viewlet コンポーネントをカスタマイズする

1) パンくず(path_bar) をコンテンツの下に移動する

(4) 表示の制御のために profiles/default/viewlets.xmlを変更する

(1) オリジナルのパンくずのviewlet "plone.path_bar" を隠す。

viewlet manager "plone.portaltop" に対して、
"plone.path_bar" をhidden にする。

サンプルコード (抜粋) [my package]/profiles/default/viewlets.xml

```
<hidden manager="plone.portaltop" skinname="wpdjskin">
  <viewlet name="plone.app.i18n.locales.languageselector" />
  <viewlet name="plone.personal_bar" />
  <viewlet name="plone.path_bar" />
</hidden>
```

IV. Viewlet コンポーネントをカスタマイズする

1) パンくず(path_bar) をコンテンツの下に移動する

(4) 表示の制御のために profiles/default/viewlets.xmlを変更する



(2) 新しく作成した viewlet "wpdjskin.path_bar" をフッタの上に表示する。

viewlet manager "plone.portalfooter" に
"wpdjskin.path_bar" を追加する。

サンプルコード (抜粋) [my package]/profiles/default/viewlets.xml

```
<order manager="plone.portalfooter" skinname="wpdjskin"
  based-on="Plone Default">
  <viewlet name="wpdjskin.path_bar" insert-before="*" />
  <viewlet name="wpdjskin.footer" />
</order>
```

#zopeをrestartし、portal_quickinstallerからパッケージを再インストールすると反映されます

IV. Viewlet コンポーネントをカスタマイズする

2) ロゴ (logo) を変更する

(1) 新たにviewletを作成するためにclassを記述します。
“plone.logo”をおきかえます。



[my package]/browser/viewlets.py にclass を追加する。

LogoViewletという名前にする。

以下ページのサンプルを参考にします。

6.8.4. Logo

<http://plone.org/documentation/manual/theme-reference/elements/visibleelements/plone.logo>

IV. Viewlet コンポーネントをカスタマイズする

2) ロゴ (logo) を変更する

サンプルコード (抜粋) [my package]/browser/viewlets.py



```
# 以下は ptをレンダリングするために必要
from Products.Five.browser.pagetemplatefile import ViewPageTemplateFile

# カスタマイズしたviewで利用する (継承する) classをインポートする。
from plone.app.layout.viewlets.common import LogoViewlet

# viewletを定義する。
# browser/configure.zcmlに登録した名前とclassの名前が一致していることが必要
# ロゴ
class logoViewlet(LogoViewlet):
    render = ViewPageTemplateFile('wpdlogo.pt')
```

IV. Viewlet コンポーネントをカスタマイズする

2) ロゴ (logo) を変更する

(2) 新viewlet表示のためのテンプレート(ptファイル)を準備します。



ZMI で

[site root] > portal_view_customizations から、logoを探す。

plone.app.layout.viewlets/logo.ptをそのまま

[my package]/browser/wpdlogo.pt として保存し、これを修正。

IV. Viewlet コンポーネントをカスタマイズする

2) ロゴ (logo) を変更する

サンプルコード [my package]/browser/wpdlogo.pt



```
<div id="logo-wrapper">
<a metal:define-macro="portal_logo"
  id="portal-logo"
  accesskey="1"
  tal:attributes="href view/navigation_root_url"
  i18n:domain="plone">
  
</a>
</div>
```

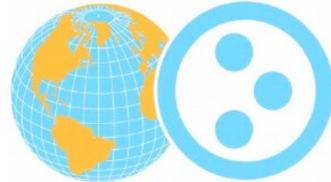
“++resource++wpdj.theme.images/[画像ファイル名]” と記述することで

[theme directry]/browser/imagesの画像にアクセスできる

※[theme directry]/browser/configure.zcmlに resource directory として登録されていることが必要で、これはセットアップ時に行われている。

IV. Viewlet コンポーネントをカスタマイズする

2) ロゴ (logo) を変更する



(3) 新viewletを /browser/configure.zcml に登録。

[my package]/browser/configure.zcml に新たに追加するviewletを定義する
viewletの名称は "wpdjskin.logo"

classは [my package]/browser/viewlets.py に追加したclass名とあわせる。

サンプルコード (抜粋) [my package]/browser/configure.zcml

```
<!-- ロゴ -->
<browser:viewlet
    name="wpdjskin.logo"
    manager="plone.app.layout.viewlets.interfaces.IPortalHeader"
    class=".viewlets.logoViewlet"
    permission="zope2.View"
/>
```

IV. Viewlet コンポーネントをカスタマイズする

2) ロゴ (logo) を変更する

(4) 表示の制御のために profiles/default/viewlets.xmlを変更する



- (1) オリジナルのロゴのviewlet "plone.logo" を隠す。
viewlet manager "plone.portalheader" に対して、
"plone.logo" をhidden にする。

サンプルコード (抜粋) [my package]/profiles/default/viewlets.xml

```
<hidden manager="plone.portalheader" skinname="wpdjskin">  
  <viewlet name="plone.site_actions" />  
  <viewlet name="plone.searchbox" />  
  <viewlet name="plone.logo" />  
</hidden>
```

IV. Viewlet コンポーネントをカスタマイズする

2009/4/25

2) ロゴ (logo) を変更する

(4) 表示の制御のために profiles/default/viewlets.xmlを変更する



(2) 新しく作成した viewlet "wpdjskin.logo" をヘッダに表示する。

viewlet manager "plone.portalheader" に
"wpdjskin.logo" を追加する。

サンプルコード (抜粋) [my package]/profiles/default/viewlets.xml

```
<order manager="plone.portalheader" skinname="wpdjskin"  
    based-on="Plone Default">  
    <viewlet name="wpdjskin.logo"  
        insert-before="plone.global_sections" />  
</order>
```

#zopeをrestartし、portal_quickinstallerからパッケージを再インストールすると反映されます

IV. Viewlet コンポーネントをカスタマイズする

3. site_actions をフッタに移動

(1) classを記述します。

site_actions をフッタに移動するために、"plone.footer"をおきかえます。



[my package]/browser/viewlets.py にclass を追加する。

footerViewletという名前にする。

SiteActionsViewletの関数をそのまま利用して、wpdfooter.ptをレンダリングする。(ちょっとイレギュラー)

参考:

6.8.2. Site Actions

http://plone.org/documentation/manual/theme-reference/elements/visibleelements/plone.site_actions

IV. Viewlet コンポーネントをカスタマイズする

3. site_actions をフッタに移動

サンプルコード (抜粋) [my package]/browser/viewlets.py



```
# 以下は ptをレンダリングするために必要
from Products.Five.browser.pagetemplatefile import ViewPageTemplateFile

# カスタマイズしたviewで利用する (継承する) classをインポートする。
# 元はヘッダにあるsite_actions
from plone.app.layout.viewlets.common import SiteActionsViewlet

# viewletを定義する。
# フッター
class footerViewlet(SiteActionsViewlet):
    render = ViewPageTemplateFile('wpdfooter.pt')
```

IV. Viewlet コンポーネントをカスタマイズする

3. site_actions をフッタに移動

(2) 新viewlet表示のためのテンプレート(ptファイル)を準備します。



ZMI で [site root] > portal_view_customizations から、
footerと site_actions を探す。

plone.app.layout.viewlets/footer.pt を外枠にして、

plone.app.layout.viewlets/site_actions.pt の内容を埋め込む。

適宜、不要な表示を消すなど調整。

[my package]/browser/wpdfooter.pt として保存。

IV. Viewlet コンポーネントをカスタマイズする

2009/4/25

3. site_actions をフッタに移動

サンプルコード [my package]/browser/wpdfooter.pt (前半)



```
<div id="portal-footer" metal:define-macro="portal_footer"
i18n:domain="plone">
<ul id="portal-siteactions"
    tal:define="accesskeys python: {'sitemap' : '3', 'accessibility' : '0',
'contact' : '9'};"
    tal:condition="view/site_actions"
    i18n:domain="plone">

<li tal:repeat="saction view/site_actions"
    tal:attributes="id string:siteaction-`${saction/id}`"><a
    href=""
    tal:define="title saction/title;
                id saction/id;
                accesskey python: accesskeys.get(id, '');"
```

IV. Viewlet コンポーネントをカスタマイズする

2009/4/25

3. site_actions をフッタに移動

サンプルコード [my package]/browser/wpdfooter.pt (後半)



```
        i18n:attributes="title"
        i18n:translate=""
        tal:content="title"
        tal:attributes="href saction/url;
                        title title;
                        accesskey accesskey;"
    >Site action</a></li>

</ul>
<!-- 表記を変更 -->
<p>World Plone Day 2008 Tokyo Theme &copy;<span
    tal:omit-tag=""
    tal:define="now modules/DateTime/DateTime"
    tal:content="now/year" /> Plone研究会</p>

</div>
```

IV. Viewlet コンポーネントをカスタマイズする

3. site_actions をフッタに移動



(3) 新viewletを /browser/configure.zcml に登録。

[my package]/browser/configure.zcml に新たに追加するviewletを定義する
viewletの名称は "wpdjskin.footer"

classは [my package]/browser/viewlets.py に追加したclass名とあわせる。

サンプルコード (抜粋) [my package]/browser/configure.zcml

```
<!-- フッター -->
<browser:viewlet
  name="wpdjskin.footer"
  manager="plone.app.layout.viewlets.interfaces.IPortalFooter"
  class=".viewlets.footerViewlet"
  permission="zope2.View"
/>
```

IV. Viewlet コンポーネントをカスタマイズする

2009/4/25

3. site_actions をフッタに移動

(4) 表示の制御のために profiles/default/viewlets.xmlを変更する



- (1) オリジナルのロゴのviewlet "plone.logo" を隠す。
viewlet manager "plone.portalfooter" に対して、
"plone.footer" をhidden にする。

サンプルコード (抜粋) [my package]/profiles/default/viewlets.xml

```
<hidden manager="plone.portalfooter" skinname="wpdjskin">  
  <viewlet name="plone.footer" />  
  <viewlet name="plone.colophon" />  
</hidden>
```

IV. Viewlet コンポーネントをカスタマイズする

3. site_actions をフッタに移動

(4) 表示の制御のために profiles/default/viewlets.xmlを変更する



(2) 新しく作成した viewlet "wpdjskin.footer" をフッタに表示する。

viewlet manager "plone.portalfooter" に
"wpdjskin.footer" を追加する。

サンプルコード (抜粋) [my package]/profiles/default/viewlets.xml

```
<order manager="plone.portalfooter" skinname="wpdjskin"  
  based-on="Plone Default">  
  <viewlet name="wpdjskin.path_bar" insert-before="*" />  
  <viewlet name="wpdjskin.footer" />  
</order>
```

#zopeをrestartし、portal_quickinstallerからパッケージを再インストールすると反映されます



Ploneの外観（スキン）を変える ーテーマプロダクトの作成ー

V. カスタムのスタイルシートを 適用する



V. カスタムのスタイルシートを適用する

[your package]/profiles/default/cssregistry.xml

ここにスタイルシートを登録します。

```
<stylesheet title=""  
  id="++resource++wpdj.theme.stylesheets/main.css"  
  media="screen" rel="stylesheet" rendering="import"  
  cacheable="True" compression="safe" cookable="True"  
  enabled="1" expression=""/>
```

#今回はスタイルシートの調整はこちらのモックアップで行い、固まったものをmain.cssにまとめました。



Resources

Plone Theme Reference

<http://plone.org/documentation/manual/theme-reference>

Create new eggs and packages quickly with paster

<http://plone.org/documentation/how-to/use-paster>

pasterを使って素早くeggとパッケージを作る (上記サイトの日本語訳)

<http://nagosui.org:8080/Nagosui/Docs/how-to/create-new-eggs-and-packages-quickly-with-paster>

Customizing the viewlets in main_template

<http://plone.org/documentation/tutorial/customizing-main-template-viewlets>

Where is What in Plone 3

<https://weblion.psu.edu/trac/weblion/wiki/PloneThreeWhereIsWhat>