

Chapter 2

Working in uPortal

Introduction to uPortal

Publishing New Channels

XHTML Design of uPortal

 The Layout

 Making a New Skin

 Cascading Style Sheets

 uPortal Graphics

Layout Fragments

Appendix A: The Default uPortal Cascading Style Sheet

 GUI Format

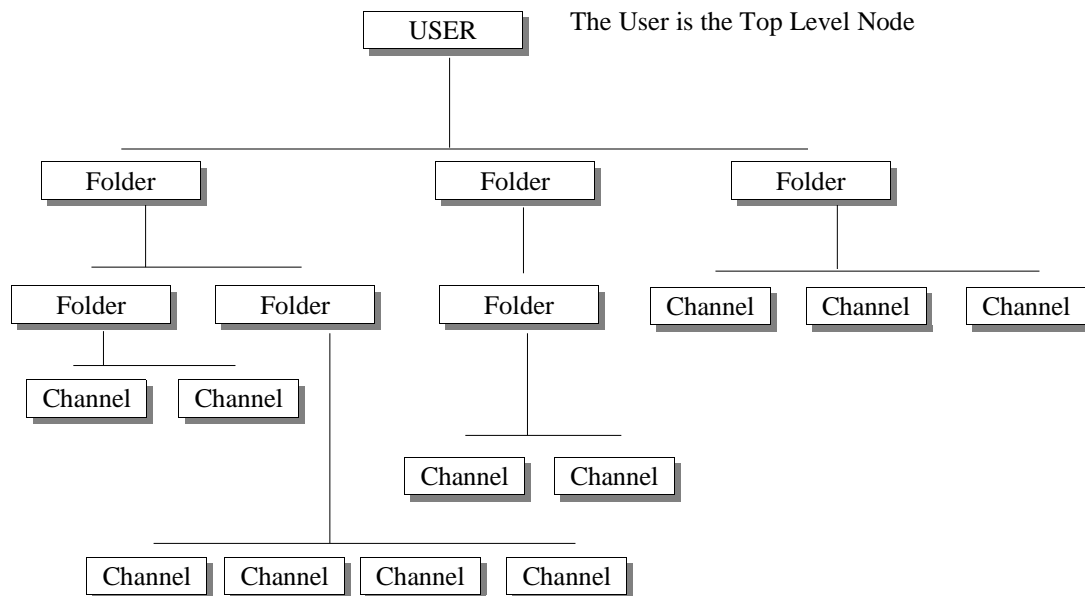
 Text Format

Note on the images in this document: Usually, the pictures that help someone understand how a program works will match exactly what that person will see on the screen of their computer. As they go from one screen to the next, the pictures in the book will move along with them so that they know that they are in the right place. A portal is very customizable in the way it looks and what options are made available for people using it. By this, each school or business can change the look and feel of their portal so that it matches their symbols and colors, as well as deciding to remove certain options and buttons. The pictures that are used in this manual were captured as uPortal was being created. It is almost certain that the look of the portal that you will be using will not match that of the one used during development. It may look different, but it will still work in the way described here.

Introduction to uPortal

uPortal is a framework for presenting aggregated content that is customizable by both the user and the administrators. It is built using a database to contain the information about each user, with XSL transformations and JAVA to take this abstract data and convert it into the final, structured layout. This process begins with the basic user layout. This layout is an XML document, with the USER as the root node. The data can be stored in a non-XML database, but must be converted into XML before transformation can continue.

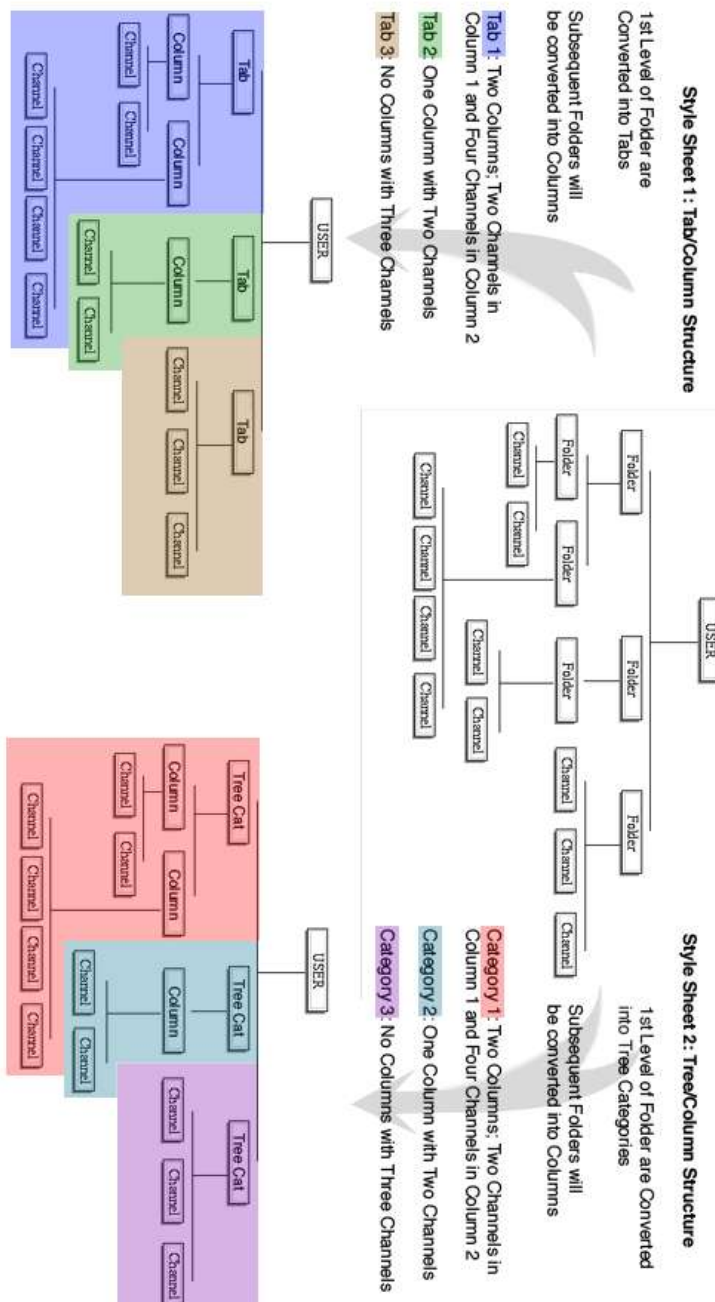
Sample User Data



*****Add Sample of User Data in XML Form*****

Structure the Data

The extracted XML data for the user is applied to a structure XSL style sheet. This style sheet creates a new XML document that organizes the data into a framework. This framework may arrange the channels within columns under and list of tabs, or it may use a vertical tree next to the cols and channels. The advantage of this step is that the data comes in as generic folders and channels and is then converted into the desired structure. One of the default style sheets can be made to convert this data into a tab/column or tree/column structure, or additional style sheets can be created to organize the data to fit whatever final structure is needed.



Apply the Theme and Render the Channels

The XML output from the structure style sheet is transformed through a second style sheet. This style sheet converts the structure into a final output mark up, such as XHTML for a web browser or WML for a cell phone, and attaches a “skin,” which contains any layout graphics, colors and text information. Parallel to this step, the channels are rendered and both pieces are combined together as the final output.

Picture of structure data having skin added

Separating the structure transform from the theme transform allows for an added level of variety. New themes can be applied to the same structure, ie an XHTML or WML version of the layout can be made from either the tab/column or the tree/column structure. Additionally, any number of skins can be made for each theme.

Publishing New Channels

Publishing a channel is the process of creating a new content channel for users to view in their layout. uPortal

HTML Design of uPortal

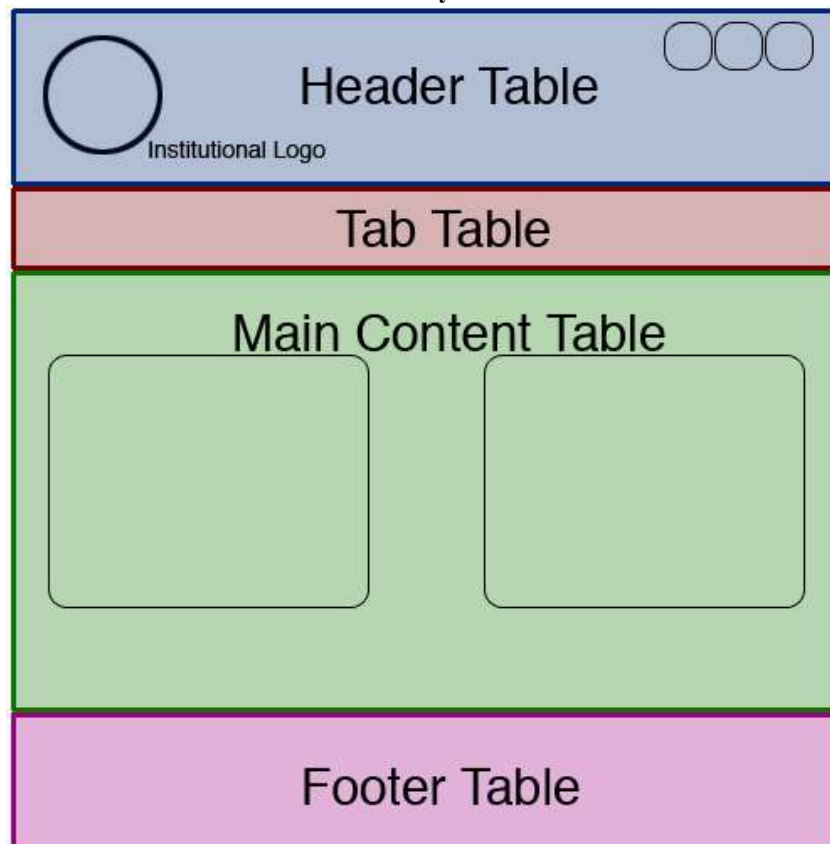
As described in the introduction, the structure of uPortal is independent of the imported data. This separation allows for the same information to be used for any number of output views, including an HTML page, PDA, cell phone, etc. The majority of uPortal users will be viewing their layout in a web browser. The following sections discuss how to modify the default look of the uPortal HTML design to match your institution or company, and to give your users extra options to personalize their view.


Layout





uPortal has been built using the XML family of mark up languages. For this reason, the HTML output from the theme style sheet is well formed XHTML. More information on XHTML can be found at the [W3C website](#), or compiled into PDF form at the [im+m eLibrary](#).

The XHTML uPortal layout consists of four tables, all set to 100% width and stacked on top of each other. These table separate the content for the **Header**, the **Tabs**, the **Channels** and the **Footer**.

Stylized Rendition of the uPortal XHTML Layout










Welcome student

EntertainmentNews




Number Guessing Game

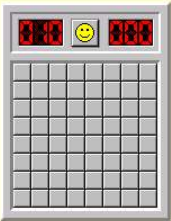


This is a number guessing game.
I am thinking of a number between 1 and 100.
What's your guess?

Submit

Minesweeper





Daily Business Cartoon



© 2004 Ted Goff www.newslettercartoons.com



"Your legacy system is a zombie,
your business plan is an apparition,
and your workers are a curse. We
recommend garlic necklaces."





Welcome student

EntertainmentNews


Number Guessing Game



This is a number guessing game.
I am thinking of a number between 1 and 100.
What's your guess?

Submit

Minesweeper





Daily Business Cartoon



© 2004 Ted Goff www.newslettercartoons.com



"Your legacy system is a zombie,
your business plan is an apparition,
and your workers are a curse. We
recommend garlic necklaces."

Making a New Skin

The skin is the last step in building each page in uPortal, wrapping the layout with the color scheme and text styles. The user has the ability to change the skin of their portal in the preferences section. This change will take place instantly, and will not effect their layout.

New skins can be made and added to uPortal. Creating a new skin requires the following steps:

1. Making a new Cascading Style Sheet (CSS)
2. Modifying the skin graphics
3. Moving the new files into their appropriate folders and redeploying the portal

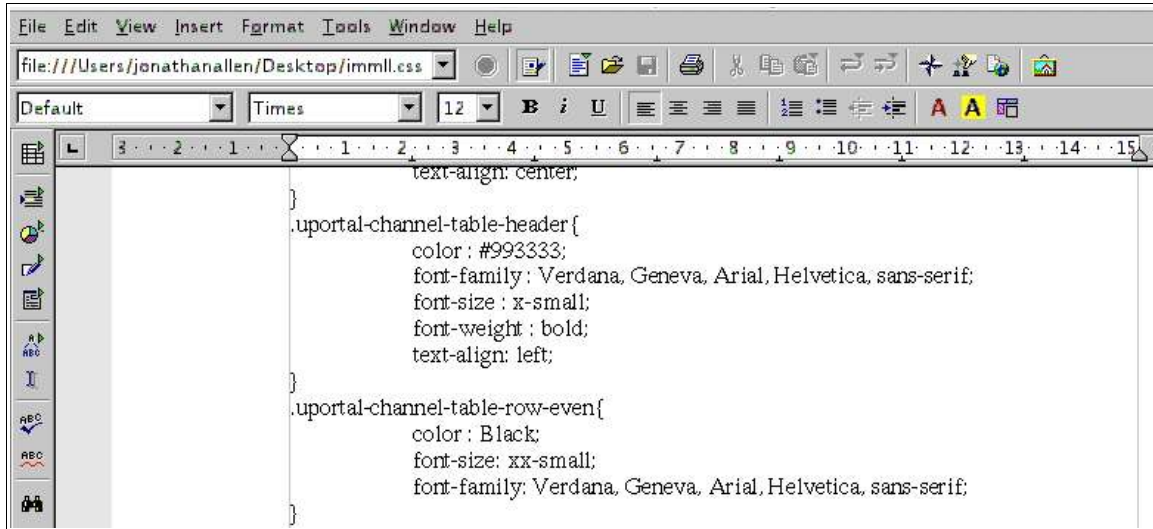
Technically, the CSS does not need to be made before the skin graphics are created. Since the CSS is used to define the color scheme of the skin, however, it will be very easy to make the new images after you have used the work on the CSS to define what the color changes need to be.

Making a new Cascading Style Sheet

uPortal uses a cascading style sheet to control the text fonts and styles, as well as the background color scheme information. Each skin uses a single style sheet for all of its styles. The styles defined in the new CSS will be translated into the look of the uPortal layout when that skin is chosen.

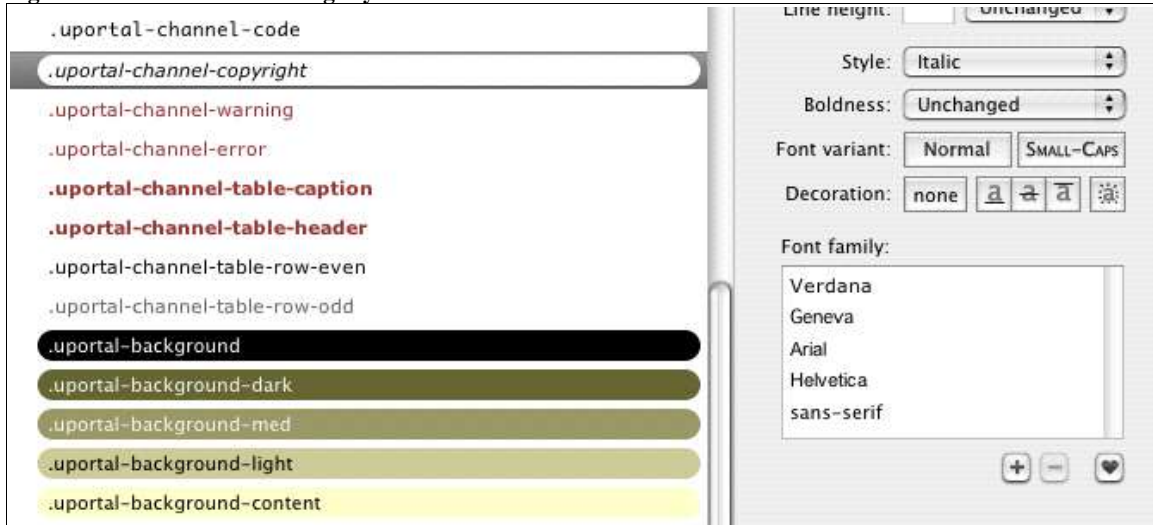
A stylesheet is a list of text and color information that can be viewed and modified in a word processor:

Fig 2.3.1: A Cascading Style Sheet in Text Format



or in a special program designed to view a CSS file:

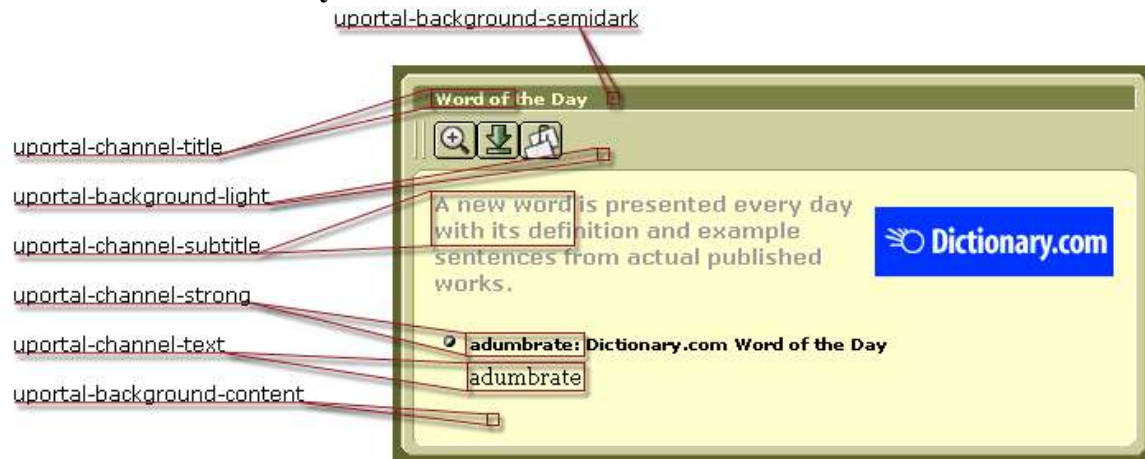
Fig 2.3.2: The Same Cascading Style Sheet in WYSIWYG Format



In addition to these two viewing options, the **CSS Viewer** channel will show the uPortal CSS in a channel on your layout.

The images below show the more prominent uPortal styles:

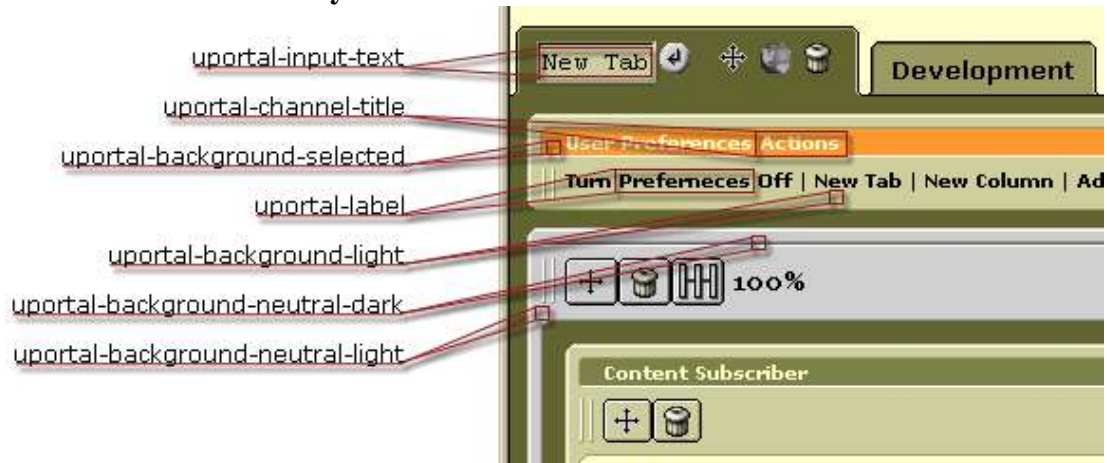
Channel Styles



Layout Styles



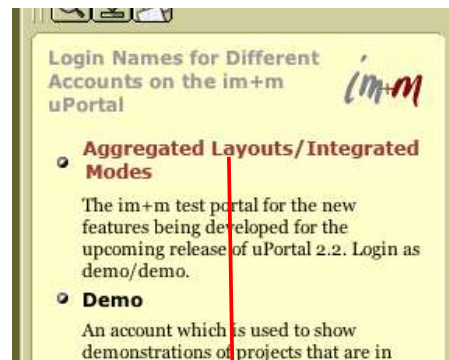
Preferences Styles



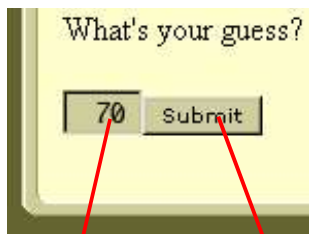
Some other important styles that are not illustrated above:



uportal-channel-code



hover



uportal-input-text

uportal-button

LEC #	LECTURER	TOPIC
1	RAW	Introducti
2	RAW	Biochemis I
3	RAW	Biochemis II
4	RAW	Biochemis

uportal-channel-table-row-odd

uportal-channel-table-row-even

The entire default, uPortal skin CSS is displayed in Appendix A. Some of the styles in this CSS appear to be duplicates, such as “uportal-channel-emphasis” and “uportal-channel-strong.” Other styles may also appear to be unused, such as “uportal-crumtrail.” These lesser used styles are included for one of two reasons. In some cases, they are legacy styles that were used in earlier version of uPortal and have remained for backwards compatibility. In other cases, extra options have been included for times when an institution may wish to have styles that are similar, yet still distinct. In all cases, no style is absolutely necessary for the portal to function. If a style is called for by the portal that is not defined in the CSS, it will be ignored and the browser default will be used in its stead. Therefore, each designer has the option to define any, all or none of the styles in the CSS when creating a new skin. It is not recommended to leave any undefined, however, as it may cause unpredictable behavior when the skin is used by uPortal.

Modifying the Skin Graphics

All of the graphic elements (Tabs, Channels, Columns and Alert Box) in uPortal have been designed to expand or contract to exactly fit the size of the content.

uPortal Tab: Browser Text Size Set to 100%



uPortal Tab: Browser Text Size Set to 500%



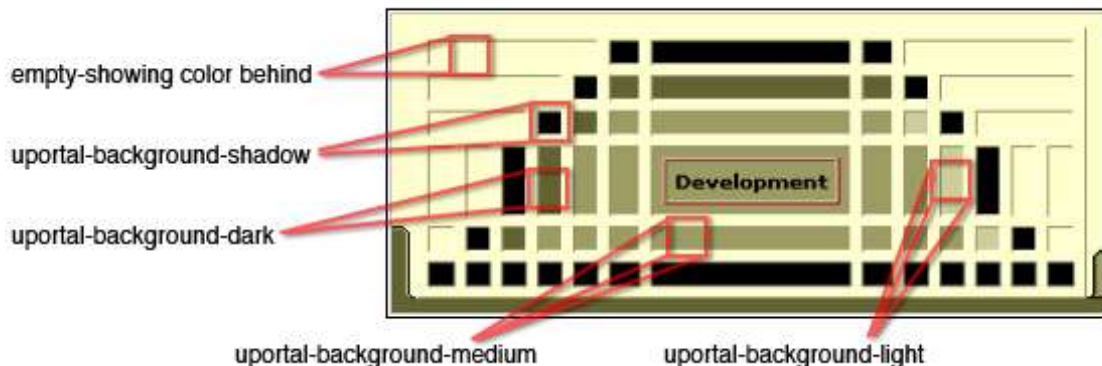
Each element is described in more detail below, along with the graphics it uses:

Tabs: The tabs in uPortal v2.2 actually do not use any graphics at all. They have been made entirely from table cells with background colors, and transparent gifs providing the correct spacing.

uPortal Tab: As Viewed in Browser



uPortal Tab: Table Spacing Set to 5 and Borders Turned On



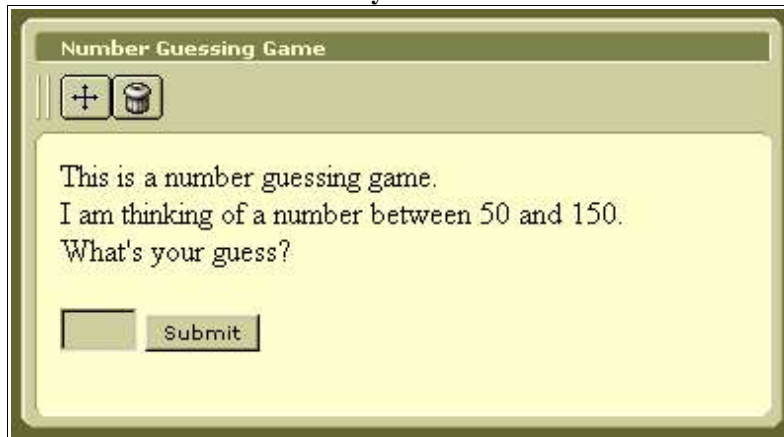
The above graphic (Fig 2.x) shows how the tab in Fig 2.x would look with the table spacing and padding set to 5, and with the borders turned on. Viewing the tabs in this way shows the design using painted table rows. All of these colors are defined by the CSS. Therefore, once the CSS has been made, the tabs will instantly updated with the

new colors.

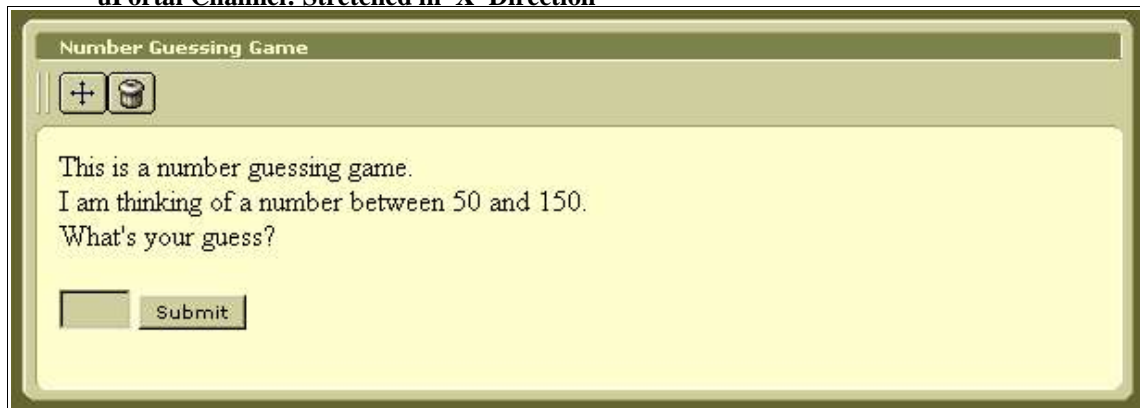
The tabs are being discussed here, in the graphics section, not as a reference to the tabs, but rather why graphics are used in uPortal. The same technique of using table cells and transparent gifs could not be used for the other three graphic elements. The reason for this is due to the need for compatibility with older web browsers, Netscape 4.7 in particular. NS 4.7 can have a maximum of 4 nested tables before behavior of the browser becomes unpredictable. All of the tabs consist of just one table for each tab, containing its name, nested within the main tab table. The channels are much more complicated in their design, especially when the columns are added in preferences mode, and often include many more than 4 nested tables. The only way to create them and keep compatibility with NS 4.7, then, was to use the technique described below.

Channels: The channel design is made from a 7 row by 3 column table, as shown below. Each of the **corner** and **transition** cells contains a single image, and is filled completely by that image. These cells are set to 0% width. Each of the **edge** and **trim** cells contains a single transparent gif and has a background image, one pixel in height or width, that repeats in the x or y direction. These cells are set to 100% width to ensure that they will be the only ones that expand with whatever content is added into middle cell of the channel table.

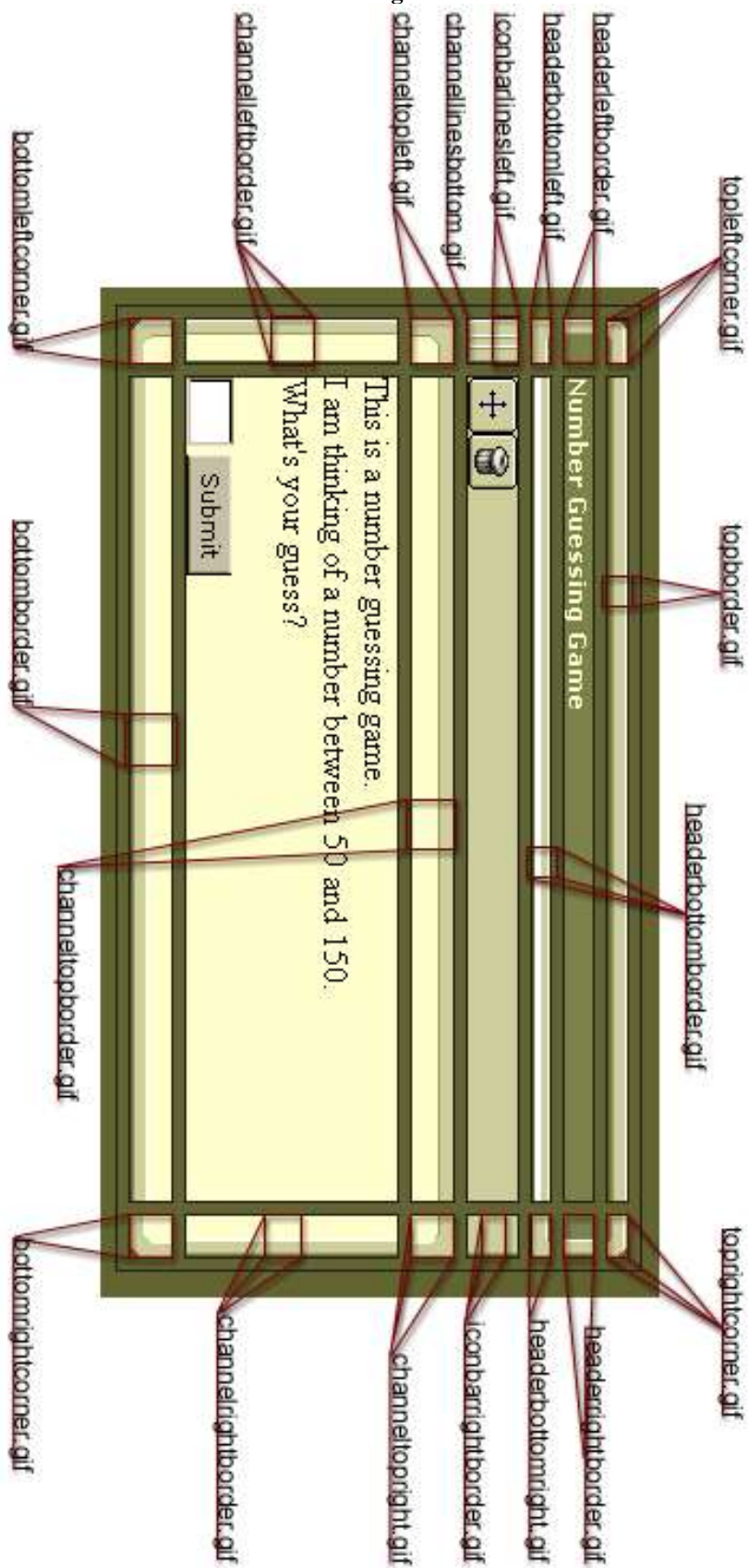
uPortal Channel: As Normally Viewed in Browser



uPortal Channel: Stretched in 'X' Direction



uPortal Channel: Table Border and Cell Padding On

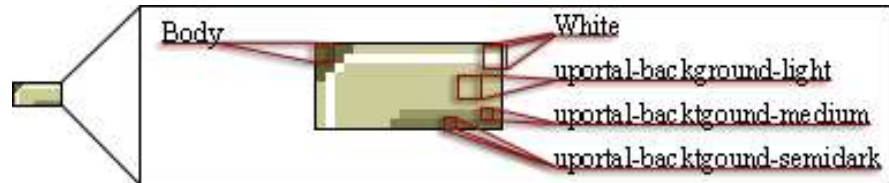


Channel Image Colors

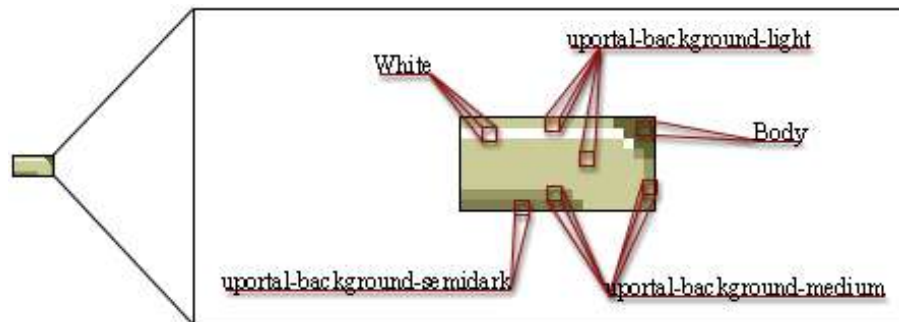
Below is a list of the images used for creating the channel interface. Each graphic is detailed with the names of the CSS styles that match the colors used. When a color is used by name and not by CSS style, it was chosen for its use in matching the scheme as an appropriate highlight or shadowing color.

Corner Images – Full Sized Images

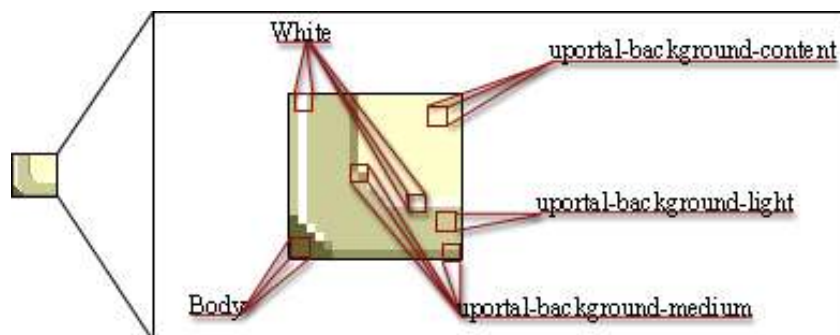
toleftcorner.gif – CSS colors listed below



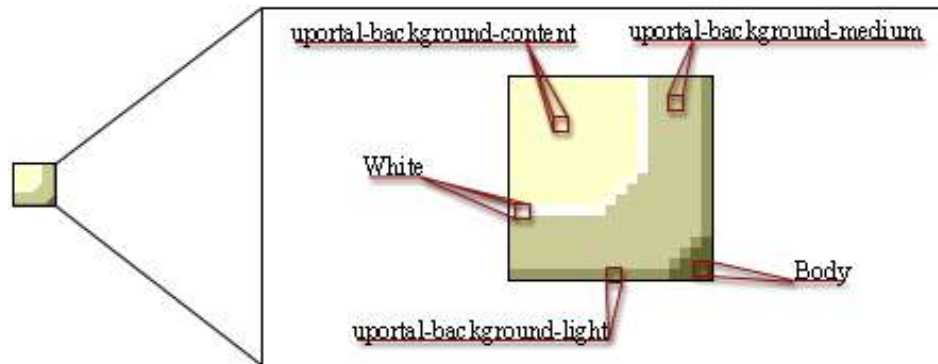
toprightcorner.gif



bottomleftcorner.gif

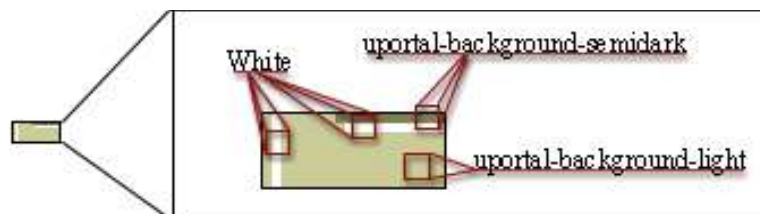


bottomrightcorner.gif

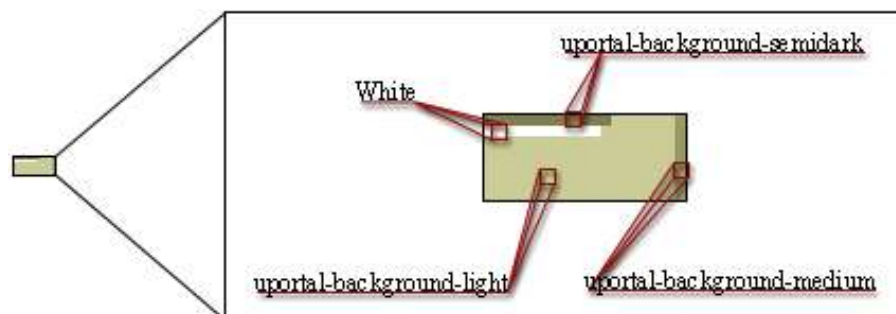


Transition Images – Full Sized Images Between Edge Cells

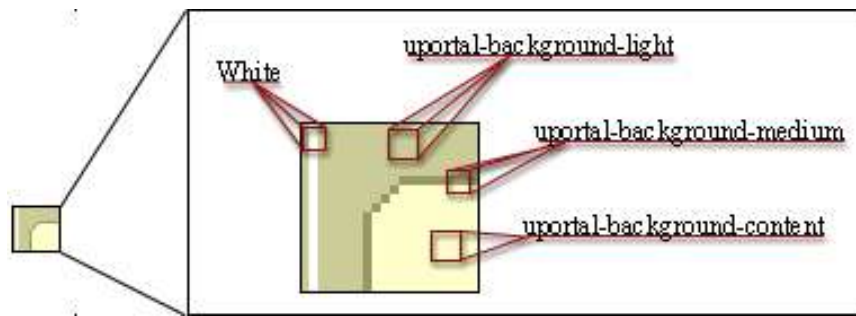
headerbottomleft.gif



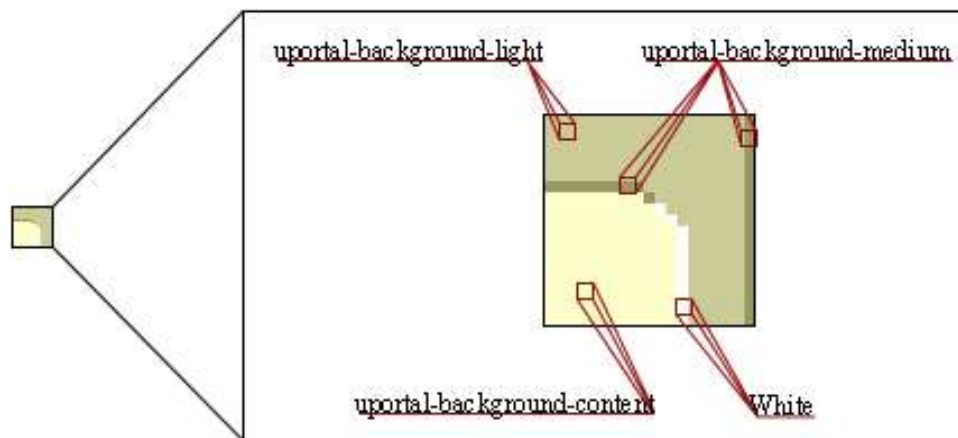
headerbottomright.gif



channeltopleft.gif



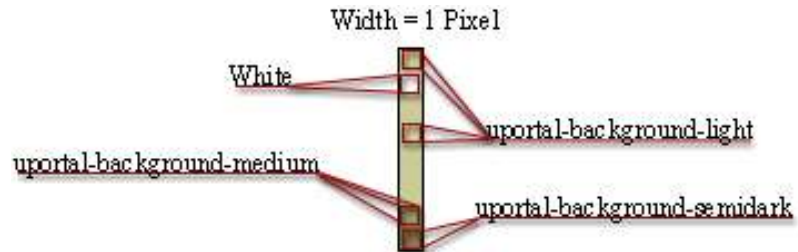
channeltopright.gif



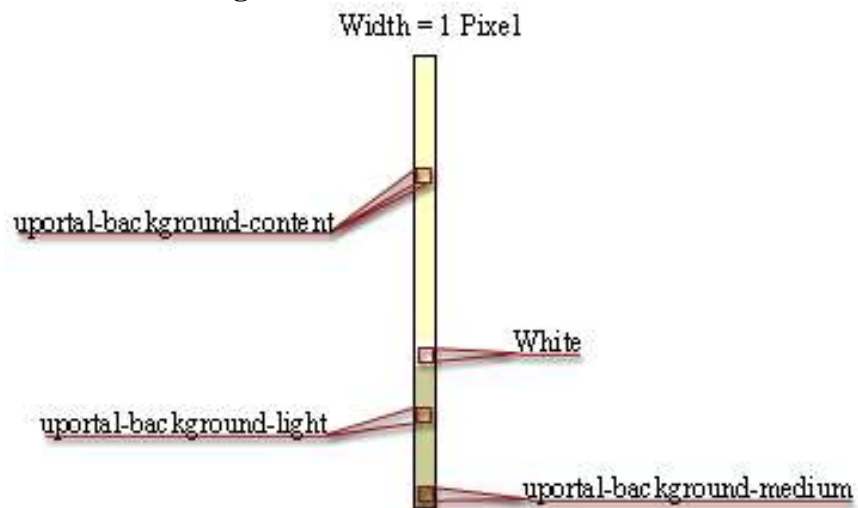
Edge Images – Single Pixel Images

Repeat 'X' Direction – One Pixel Wide

topborder.gif



bottomborder.gif

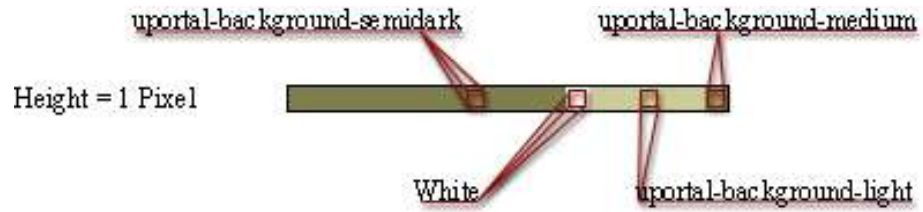


Repeat 'Y' Direction – One Pixel High

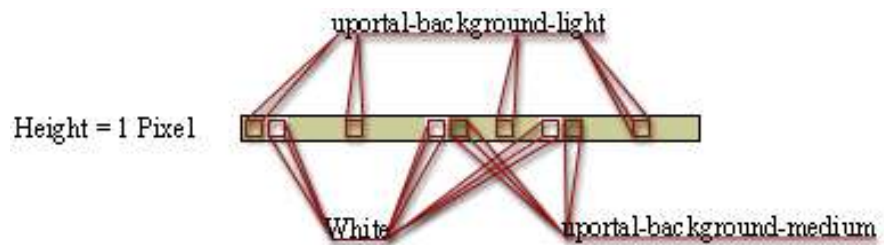
headerleftborder.gif



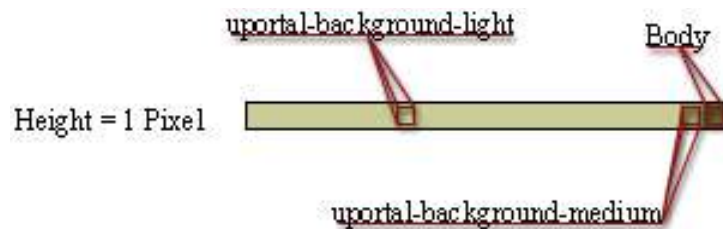
headerrightborder.gif



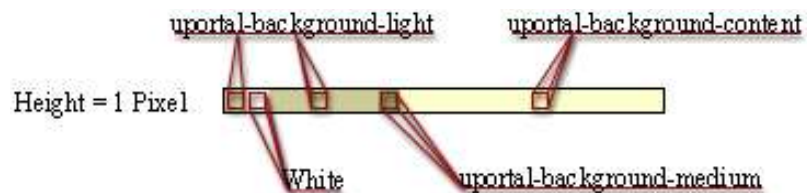
iconbarlinesleft.gif



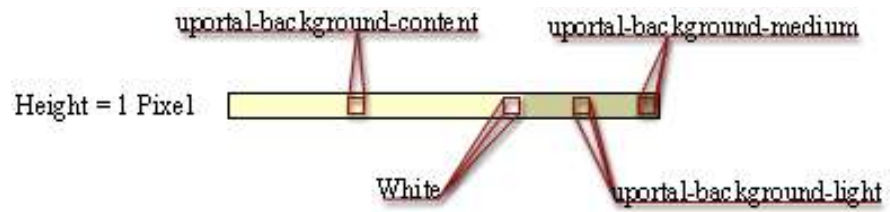
iconbarrightborder.gif



channelleftborder.gif

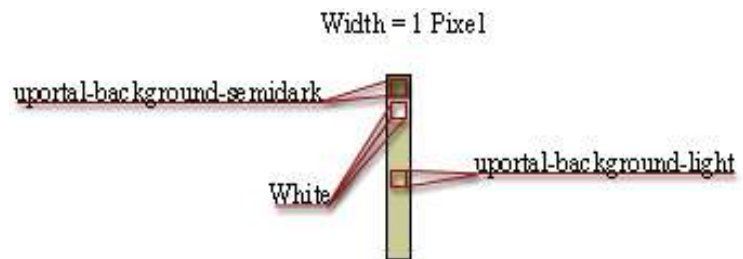


channelrightborder.gif

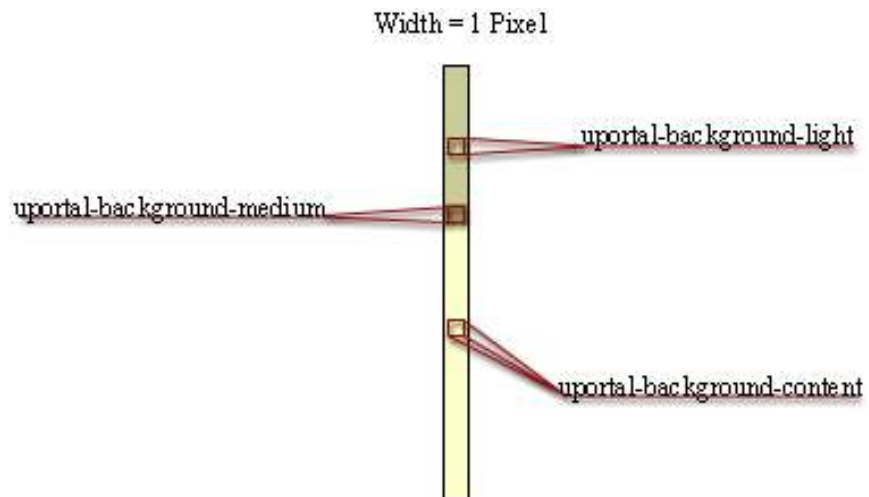


Trim Images – Single Pixel Wide Images, Repeat in 'X' Direction

headerbottomborder.gif

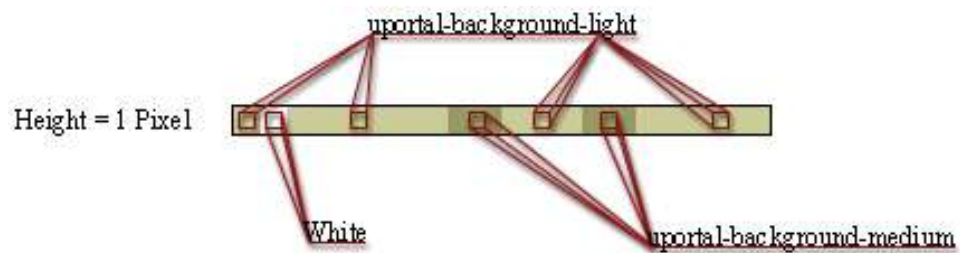


channeltopborder.gif



Other – Single Pixel High Image that Does NOT Repeat

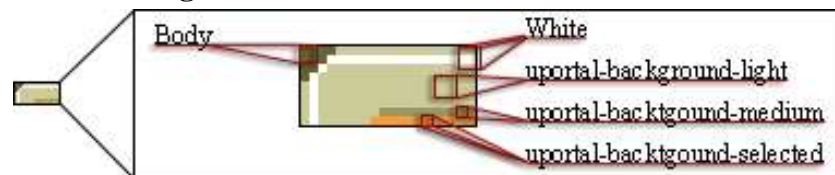
channellinesbottom.gif



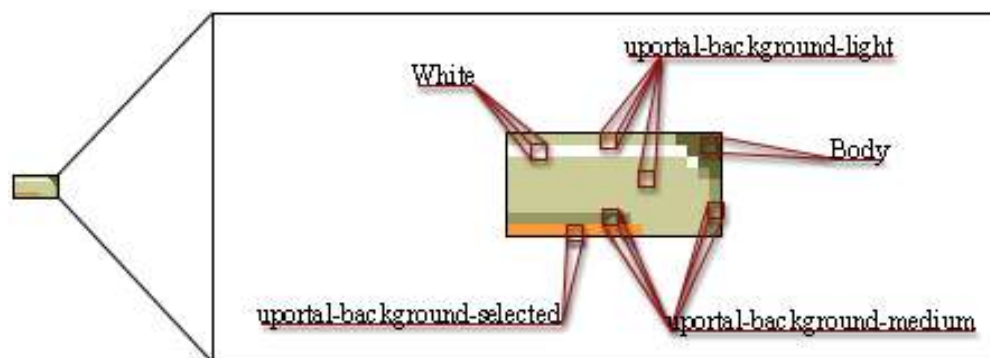
This graphic has been added to “cap off” the icon bar lines, giving them a sense of completion.

Selected Images – These images replace those in the column header when the column is selected to perform an action on it, such as moving it.

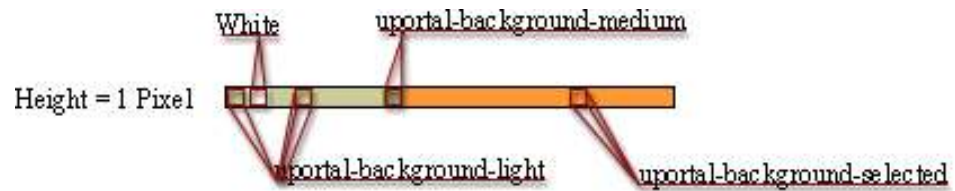
toleftcornerselected.gif



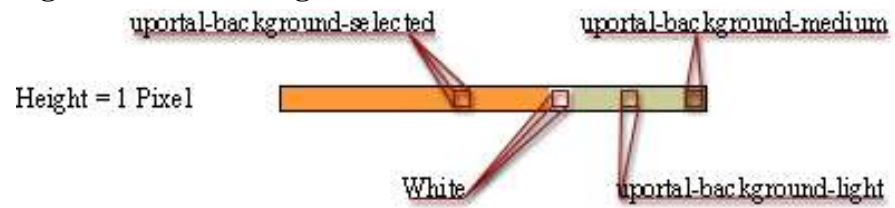
toprightcornerselected.gif



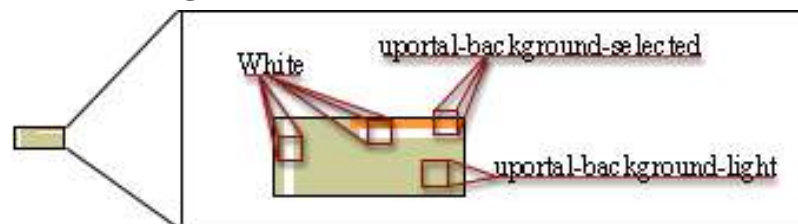
headerleftborderselected.gif



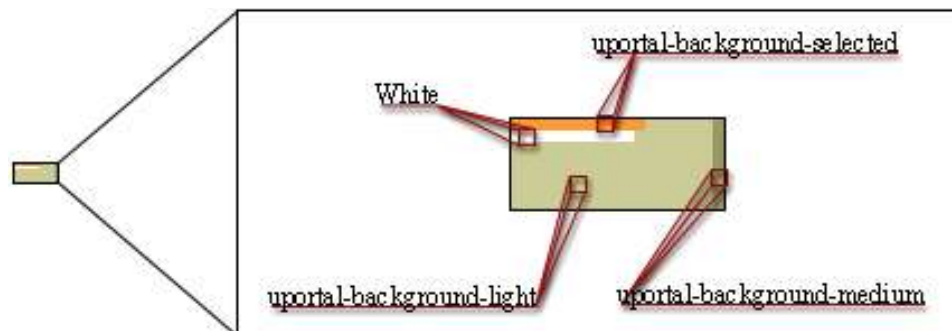
headerrightborderselected.gif



headerbottomleftselected.gif



headerbottomrightselected.gif



Content Cells – The three remaining table cells

Channel Title Cell is filled with 'uportal-background-semidark'

Number Guessing Game

Icon Bar Cell is filled with 'uportal-background-light'



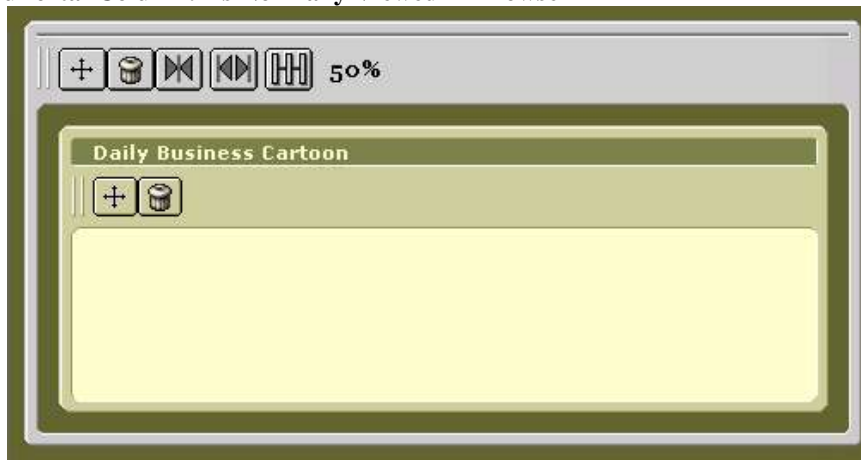
Channel Content Cell is filled with 'uportal-background-content'

This is a number guessing game.
I am thinking of a number between 50 and 150.
What's your guess?

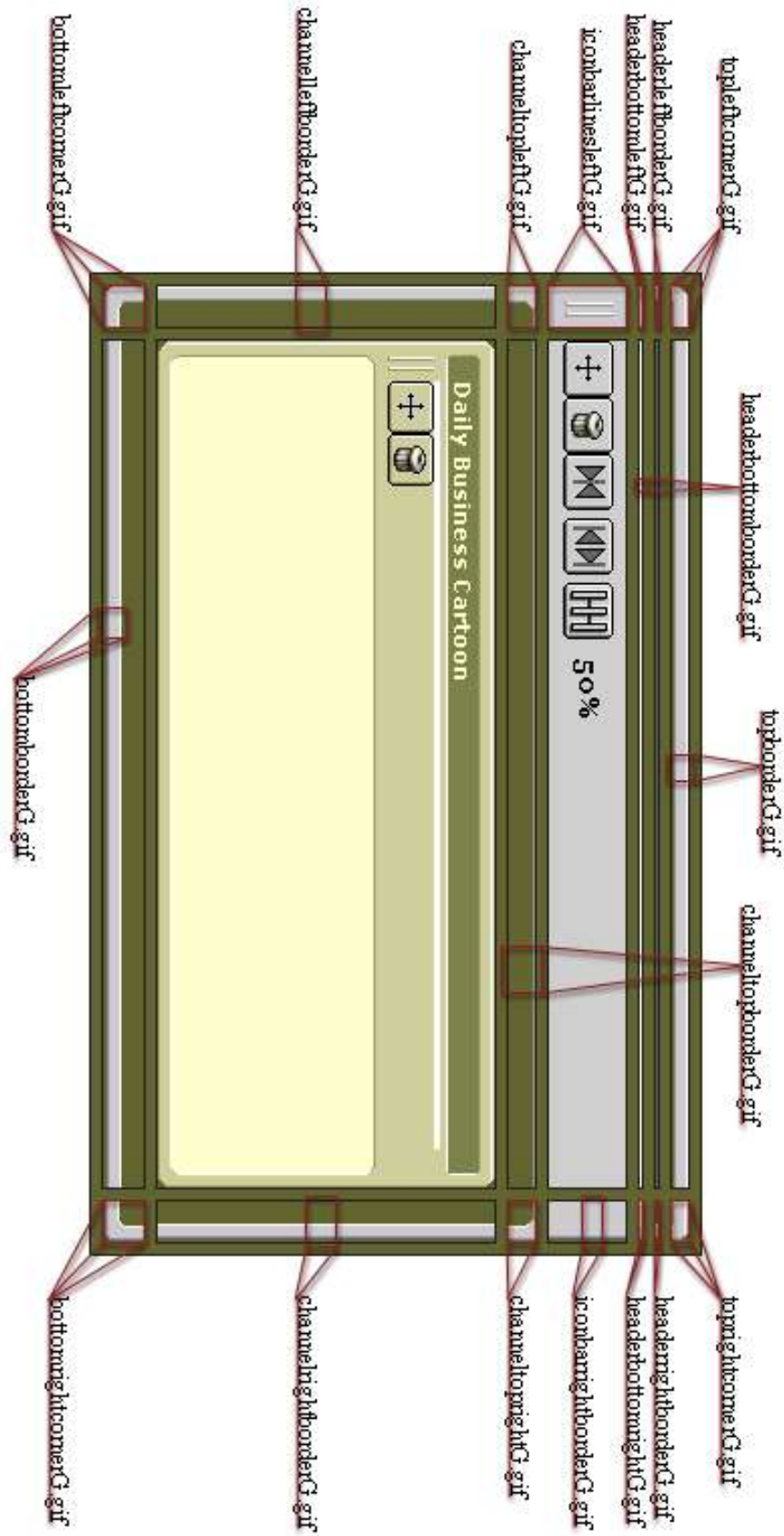
Columns: The columns are designed in exactly the same way as the channels, using a 7x3 table to expand around the content in the middle column of cells. The only real difference between the design of the channel and the column is the lack of a title cell in the column. The space is still there for it, but it is left as a one pixel high, dark grey line instead of filling it with text. As a more subtle difference in the column design, the padding around the icon bar is built into the icon bar itself on the column, whereas it is part of the graphics above and below the icon bar in the channel.

The graphics used for the columns are shown below. The names of the images used for the column are often the same as those in the channels, with a “G” (which originally stood for “grey,” when all of the graphics were in one directory) tacked on the end. Therefore, the channel graphic “**bottomborder.gif**” would be similar to the column graphic “**bottomborderG.gif**.” Later, the column border and the channel border images were separated into different directories, but the “G” remained as legacy.

uPortal Column: As Normally Viewed in Browser



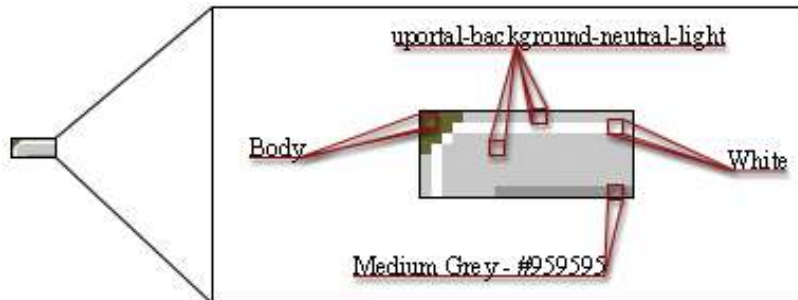
uPortal Column: Table Border and Cell Padding On



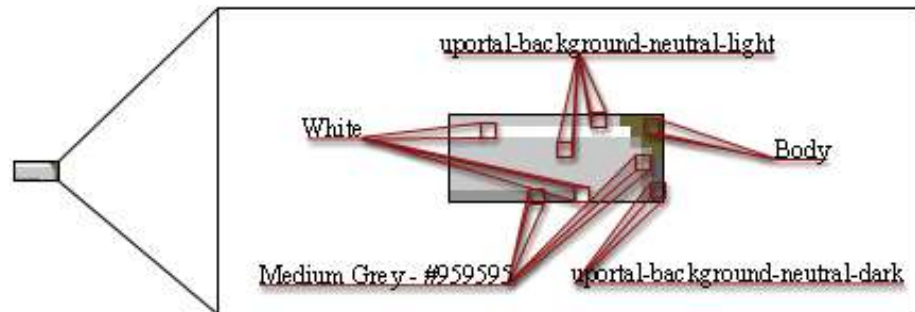
Column Images

Corner Images – Full Sized Images

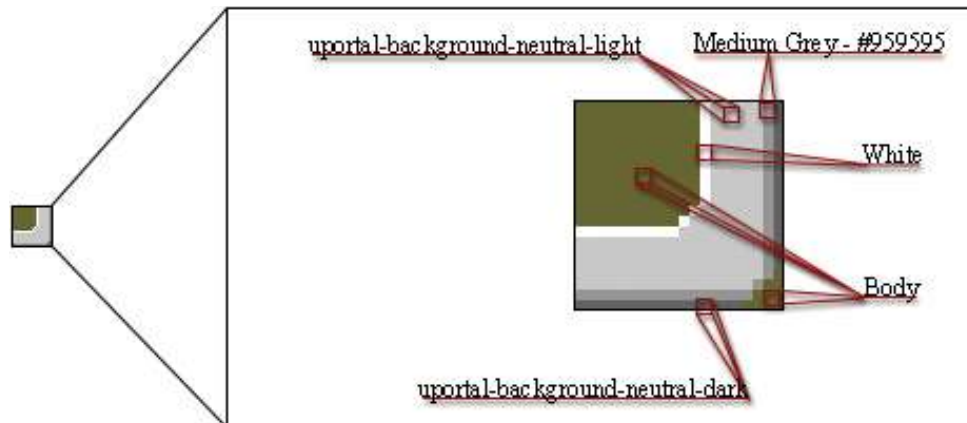
toleftcornerG.gif



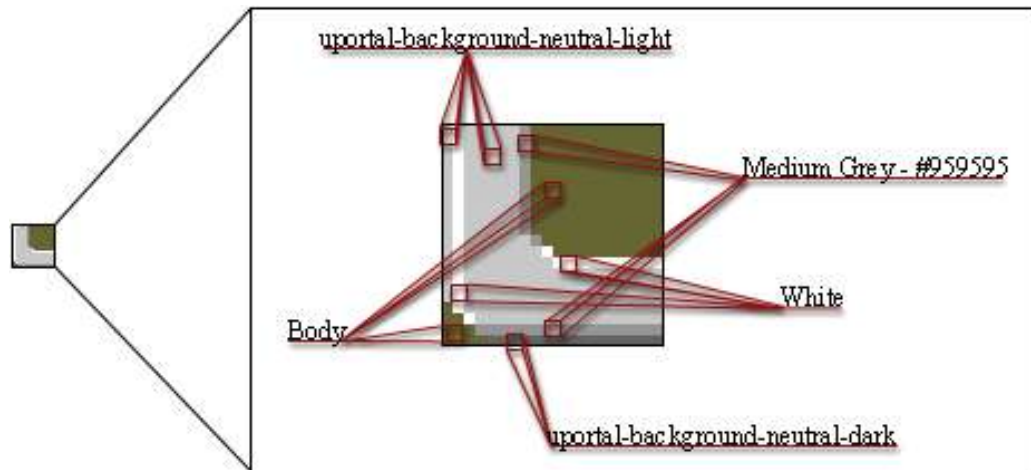
toprightcornerG.gif



bottomrightcornerG.gif

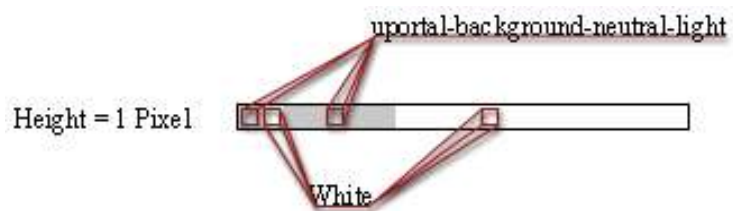


bottomleftcornerG.gif

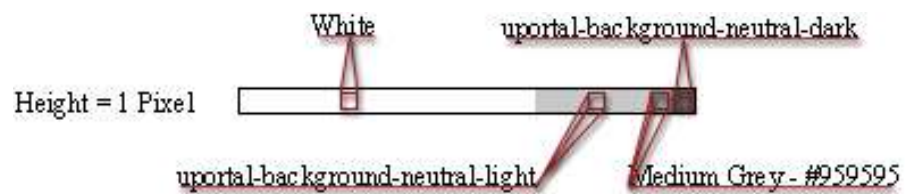


Transition Images – Full Sized Images Between Edge Cells

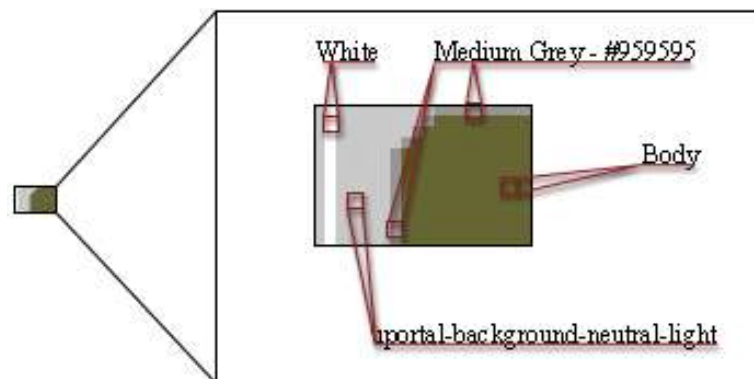
headerbottomleftG.gif – Note that it is full size at 1 Pixel



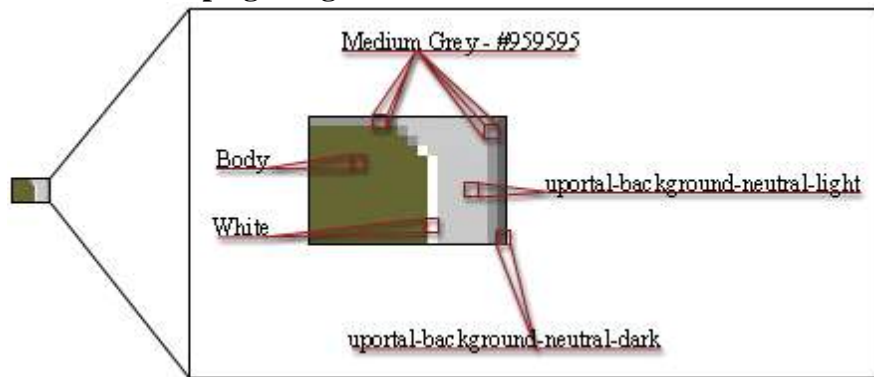
headerbottomrightG.gif - Note that it is full size at 1 Pixel



channeltopleftG.gif



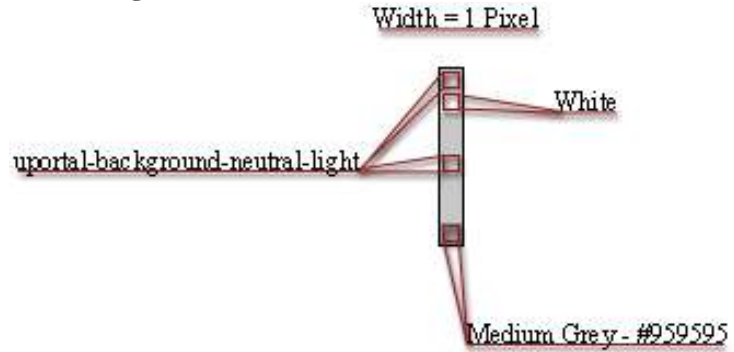
channeltoprightG.gif



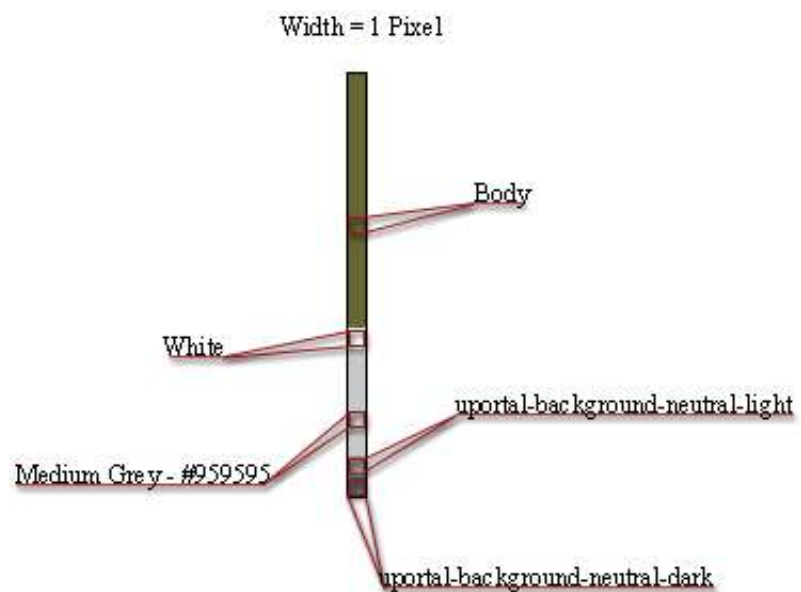
Edge Images – Single Pixel Images

Repeat 'X' Direction – One Pixel Wide

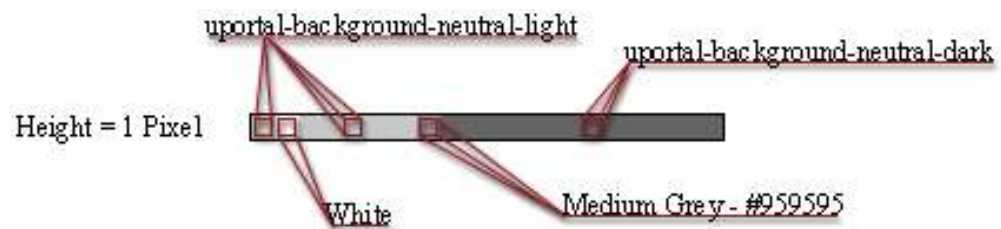
topborderG.gif



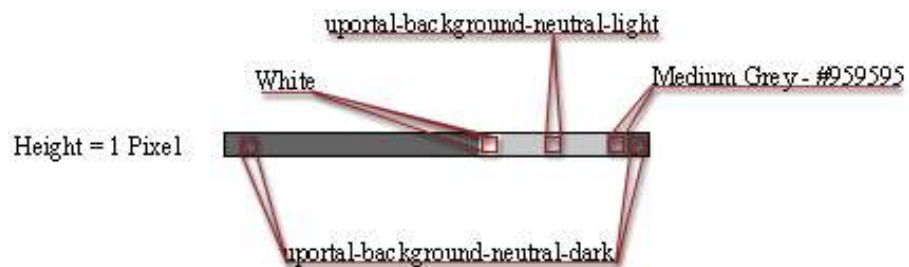
bottomborderG.gif



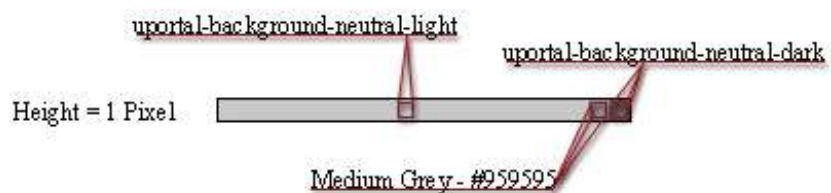
Repeat 'Y' Direction – One Pixel High
headerleftborderG.gif



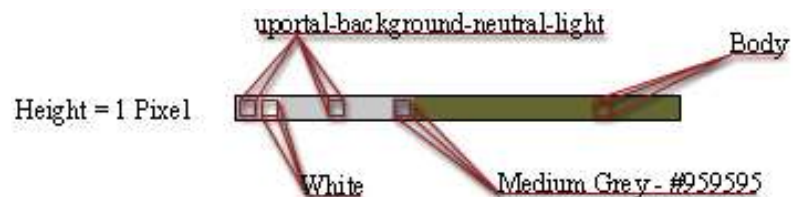
headerrightborderG.gif



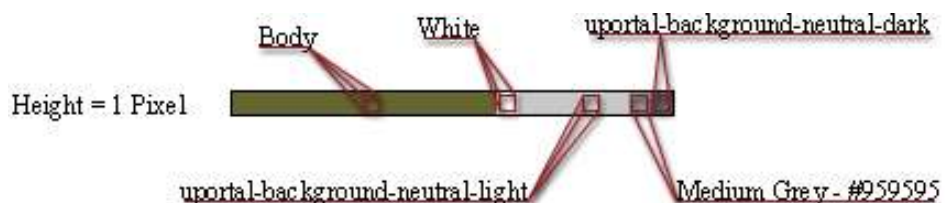
iconbarrightborderG.gif



channellleftborderG.gif



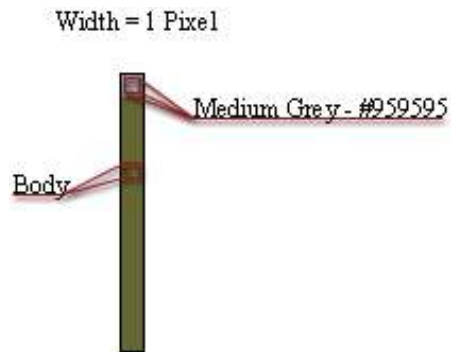
channelrightborderG.gif



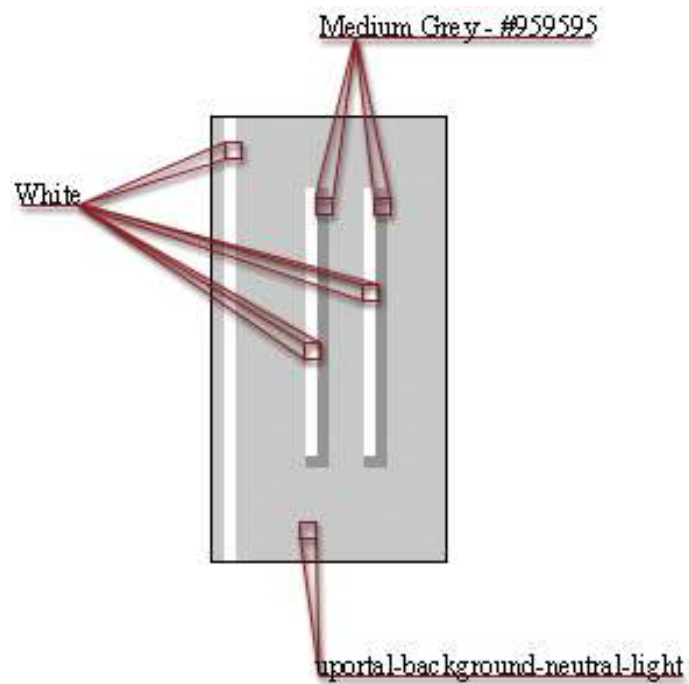
Trim Images – Single Pixel Wide Images, Repeat in 'X' Direction
headerbottomborderG.gif

White
1 Pixel Square 

channeltopborderG.gif

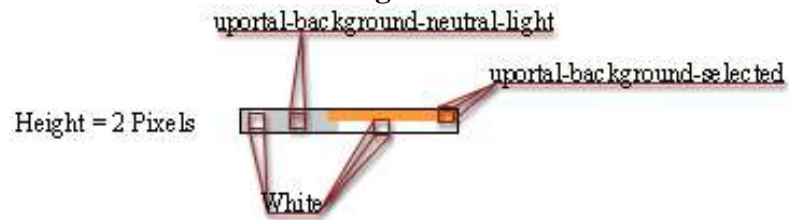


Other – Icon Bar Lines Image that Does Not Repeat
iconbarlinesleftG.gif

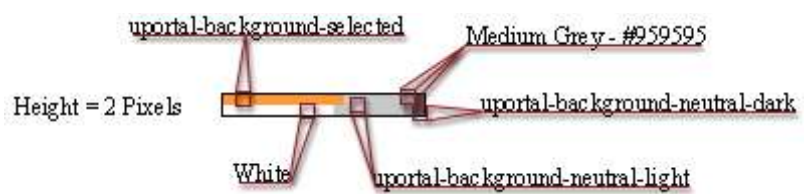


Selected Images – These images replace those in the column header when the column is selected to perform an action on it, such as moving it.

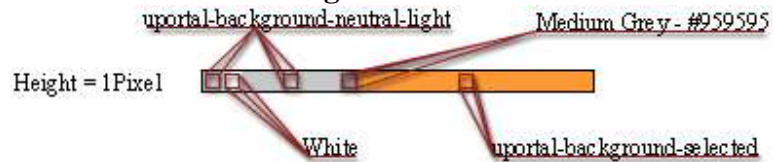
headerbottomleftGselected.gif



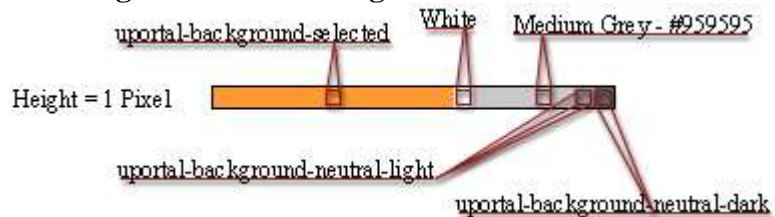
headerbottomrightGselected.gif



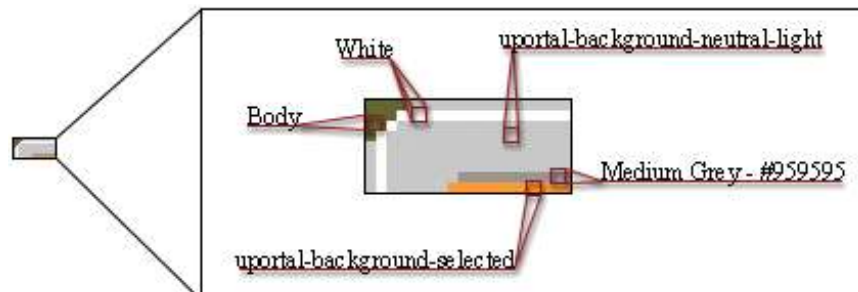
headerleftborderselected.gif



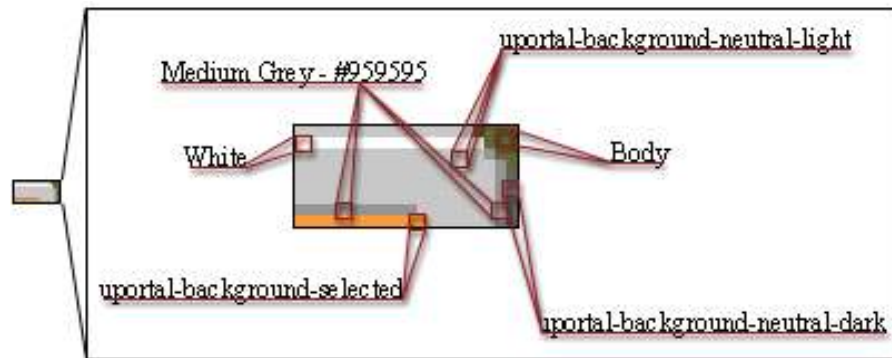
headerrightborderselected.gif



topleftcornerGselected.gif



toprightcornerGselected.gif



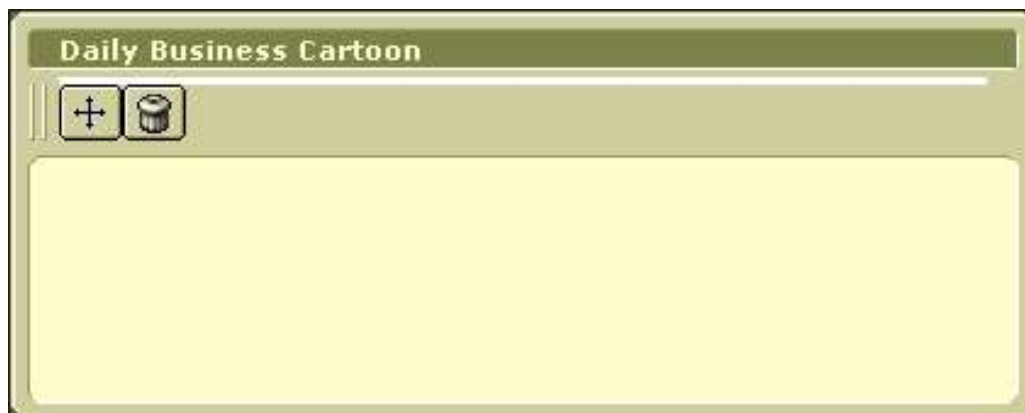
Content Cells – The three remaining table cells

Column Header Cell is one pixel in height, and filled with 'Medium Grey - #959595'

Icon Bar Cell is filled with 'uportal-background-neutral-light'

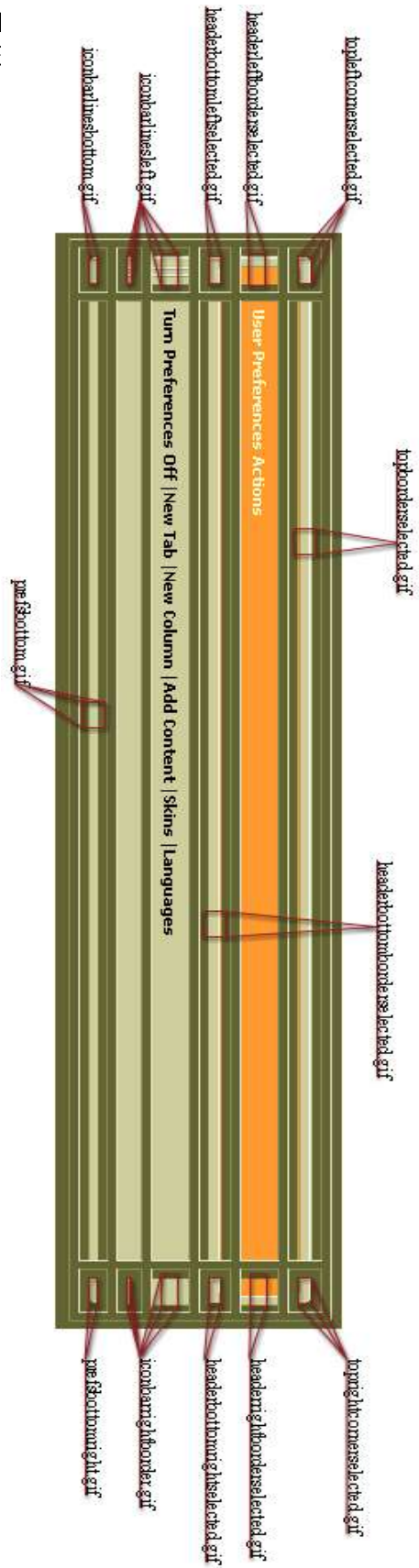


Column Content Cell is filled with 'Body'



ire to the channels and
cell

Alert Box: The `table` element can have a maximum of 6 columns. It only uses a 6x6 grid.



Icons: The icon graphics are fixed in size, though new ones can be made to replace the default versions.

Header Icons

Preferences Icons

Layout Fragments

An Introduction to Layout Fragments

A layout fragment is a block of uPortal content, including framework elements such as tabs and columns or tree elements as well as channels. In the purest sense, a fragment can be a single channel, or they can be more complicated, consisting of several channels dispersed through out multiple columns and tabs. Fragments can be “pushed” out to a user, in that an author such as a professor forces the fragment to appear on the layout of each of their students, or they may be “pulled” by the user, in that a user subscribes to a large block of content all at one time. Fragments are a very useful feature for uPortal, though there are some very difficult problems to solve as development of uPortal continues.

On the lowest level, the attachment point of a layout fragment becomes an issue. As described above, the USER is the root element, with “Tab” folders listed below that. How exactly does the fragment get added into the rest of the user’s layout? What if a developer wants to have a fragment that is channels without columns or tabs? What if they want to have one that involves multiple tabs? How do we represent a tab/column fragment in a tree structure? There are several issues that disrupt how a fragment can work in a uPortal layout. The solution at this time is to create a base level structure for a fragment, known as the “foundation.” All fragments must start with this foundation. As development of uPortal and fragments continues, more options for the layout fragment foundation can be included. At the current level of uPortal development, this foundation will need to be defined as a single “Tab.” This will give the portal a place that it understands to attach the fragment.

On the content level, layout fragments will run into issues with permissions of the channels that are included in the fragment and the users of that fragment. For a pushed fragment, it is more probable that an author will know who will receive the fragment. For example, a Biology professor who makes a fragment for a 301 class will know who has enrolled in the class. However, schedule changes make it impossible to ever be certain of this 100%. This issue is compounded for pulled fragments, as the fragment author has very little information about who will be subscribing to a fragment. What happens, then, in the inevitable condition when a layout fragment subscriber is pushed or pulling content to which they do not have permission to include in their layout? Once again, this is a case where continued uPortal and fragment development can change how this will be handled. For the current release of uPortal, however, the fragment will be displayed to the user, although any channels that conflict with a user due to permissions will be empty.

Creating Fragments

As discussed in the introduction, fragments are a new addition to uPortal. For this reason, much of the development for fragment management will continue through the 2.2 release of uPortal. As this development continues, new options for creating fragments will be added and they will be discussed in subsequent versions of this document. At the time of this writing, however, the only method for creating a new fragment is to build the necessary documents by hand.

A fragment is an XML document. A sample XML fragment and it resulting XHTML rendering in uPortal is shown below, with an explanation of the tags following.

```
<?xml version="1.0" encoding="utf-8"?>
<fragments>
  <fragment name="pfragment1.0">
    <description>The push fragment example</description>
    <groups>
      <group>Students</group>
      <group>Developers</group>
    </groups>
    <restrictions>
      <restriction path="local" name="priority" value="0-20000"/>
      <restriction path="local" name="depth" value="1"/>
      <restriction path="parent" name="priority" value="1-10"/>
    </restrictions>
    <folder name="Sample Fragment" immutable="Y" unremovable="Y" hidden="N">
      <folder name="column1" immutable="Y" unremovable="Y" hidden="N">
        <channel fname="word-of-the-day" immutable="Y" unremovable="Y" hidden="N"/>
        <channel fname="salon.com" immutable="Y" unremovable="Y" hidden="N"/>
      </folder>
      <folder name="column2" immutable="Y" unremovable="Y" hidden="N">
        <channel fname="motley-fool" immutable="Y" unremovable="Y" hidden="N"/>
      </folder>
    </folder>
  </fragment>
</fragments>
```



Explanation of XML Fragment Tags

<?xml version="1.0" encoding="utf-8"?>: Opening tag for all XML documents. It is not an XML element, and does not require a closing tag.

<fragments>: Root tag for the fragment. One and only one “fragments” tag is required.

<fragment name="pfragment1.0">: Opening tag for this particular fragment, including an attribute, “name,” that contains the name of the fragment.

<description>: Metadata about the fragment that will be shown by uPortal to a user when they are subscribing to new content.

<groups>: Opening tag for the list of groups that have permissions to subscribe to this fragment.

<group>: One “group” tag is needed for each group that has permission to subscribe to this fragment.

<restrictions>: Opening tag for the list of restrictions that will be applied to the use of this fragment.

<restriction path="local" name="priority" value="0-20000"/>: Each restriction requires one tag, with its values defined in the attributes. These restrictions are:

Local Priority: defines any restriction that the fragment author decides to put on the position of the fragment root in the layout. Can contain any numeric string range from 0-20000.

Depth: defines any restrictions that the fragment author may place on where in the user layout node tree the fragment can be attached. The USER node is depth level 1, the first folder level is depth level 2, etc. Since the fragment foundation is currently restricted to a Tab, which is attached to the ROOT, this must be set to a value of “1.” The fragment author will be given the ability to set this restriction once the foundation is more open. Can contain a single string number, the range of which is dependent on the number of levels in the layout.

Parent Priority: defines any restriction that the fragment author decides to put on the position of the parent to which the fragment will be attached or moved to. Can contain a numeric string range from 0-20000.

<folder name="Sample Fragment" immutable="Y" unremovable="Y" hidden="N">
The opening element for the actual content of the fragment. The attributes for this tag define its:

name: the name of the fragment. Can carry any string value.

immutable: defines whether changes can be made to the fragment by the user.
Can carry the value of “Y” or “N.”

unremovable: defines whether the fragment can be deleted by the user. Can carry the value of “Y” or “N.”

hidden: defines whether the fragment appears in the rendered layout. Can carry the value of “Y” or “N.”

<folder name="column1" immutable="Y" unremovable="Y" hidden="N">
Opening tag for the framework elements. Uses the same attribute types as the “fragment” tag, though the values can be different.

<channel fname="word-of-the-day" immutable="Y" unremovable="Y" hidden="N"/> Opening tag for a content channel. Also uses the same attribute types as the “fragment” tag, and the values may also be different, except “**fname**” is used in the place of “**name**.” Fname is a value used by the database to locate the requested channel.

Publishing Fragments

This is something that I do not fully understand, and need to have Michael explain to me...as it must be done manually at this time.

Appendix A: Default uPortal Cascading Style Sheet

Styles
BODY
A
A:VISITED
A:HOVER
A.navigation
A.navigation:visited
A.navigation:hover
.navigation-selected
A.uportal-navigation-category
A.uportal-navigation-category:visited
A.uportal-navigation-category:hover
.uportal-navigation-category-selected
A.uportal-navigation-channel
A.uportal-navigation-channel:visited
A.uportal-navigation-channel:hover
.uportal-navigation-channel-selected
.uportal-text
.uportal-text-small
.uportal-button
.uportal-label
.uportal-input-text
.uportal-text-reversed
.uportal-crumbtrail
<i>.uportal-copyright</i>
.uportal-channel-text
.uportal-channel-title
.uportal-channel-title-reversed
.uportal-channel-subtitle
.uportal-channel-subtitle-reversed
.uportal-channel-emphasis
.uportal-channel-strong
.uportal-channel-code
<i>.uportal-channel-copyright</i>
.uportal-channel-warning
.uportal-channel-error
.uportal-channel-table-caption

Appendix A: Default uPortal Cascading Style Sheet

.uportal-channel-table-caption

.uportal-channel-table-header

.uportal-channel-table-row-even

.uportal-channel-table-row-odd

.uportal-background

.uportal-background-dark

.uportal-background-med

.uportal-background-light

.uportal-background-content

.uportal-background-highlight

.uportal-background-shadow

.uportal-background-page

.uportal-background-semidark

.uportal-background-selected

.uportal-background-neutral-dark

.uportal-background-neutral-light

Appendix A: Default uPortal Cascading Style Sheet - Text Format

```
BODY{
    background: #666633;
}
A {
    text-decoration : none;
    color : Black;
}
A:VISITED {
    color : Black;
}
A:HOVER {
    color : #993333;
    text-decoration: underline;
}
A.navigation {
    color : #FFFFCC;
    font-size : x-small;
    font-weight : bold;
    font-family : Verdana, Geneva, Arial, Helvetica, sans-serif;
}
A.navigation:visited {
    color : #FFFFCC;
    font-size : x-small;
    font-weight : bold;
    font-family : Verdana, Geneva, Arial, Helvetica, sans-serif;
}
A.navigation:hover {
    color : #993333;
    font-size : x-small;
    font-weight : bold;
    font-family : Verdana, Geneva, Arial, Helvetica, sans-serif;
    text-decoration: underline;
}
.navigation-selected{
    font-size : x-small;
    font-weight : bold;
    font-family : Verdana, Geneva, Arial, Helvetica, sans-serif;
    color : #FFFFCC;
}
A.uportal-navigation-category{
    color : Black;
    font-size : x-small;
    font-weight : bold;
    font-family : Verdana, Geneva, Arial, Helvetica, sans-serif;
```

```
}
A.uportal-navigation-category:visited{
    color : Black;
    font-size : x-small;
    font-weight : bold;
    font-family : Verdana, Geneva, Arial, Helvetica, sans-serif;
}
A.uportal-navigation-category:hover{
    color : #993333;
    font-size : x-small;
    font-weight : bold;
    font-family : Verdana, Geneva, Arial, Helvetica, sans-serif;
    text-decoration: underline;
}
.uportal-navigation-category-selected{
    font-size : x-small;
    font-weight : bold;
    font-family : Verdana, Geneva, Arial, Helvetica, sans-serif;
    color : #000000;
}
A.uportal-navigation-channel{
    color : Black;
    font-family : Verdana, Geneva, Arial, Helvetica, sans-serif;
    font-size : x-small;
}
A.uportal-navigation-channel:visited{
    color : Black;
    font-family : Verdana, Geneva, Arial, Helvetica, sans-serif;
    font-size : x-small;
}
A.uportal-navigation-channel:hover{
    color : #993333;
    font-family : Verdana, Geneva, Arial, Helvetica, sans-serif;
    font-size : x-small;
    text-decoration: underline;
}
.uportal-navigation-channel-selected{
    background : Black;
    font-family : Verdana, Geneva, Arial, Helvetica, sans-serif;
    font-size : x-small;
    color : White;
}
.uportal-text{
    color : Black;
    font-family : Georgia, "Times New Roman", Times, serif;
    font-size : x-small;
}
```

```
.uportal-text-small{
    color : Black;
    font-family : Verdana, Geneva, Arial, Helvetica, sans-serif;
    font-size : xx-small;
}
.uportal-button{
    color : Black;
    font-family : Verdana, Geneva, Arial, Helvetica, sans-serif;
    font-size : xx-small;
    background : #CCCC99;
}
.uportal-label{
    color : Black;
    font-family : Verdana, Geneva, Arial, Helvetica, sans-serif;
    font-size : xx-small;
    font-weight : bold;
}
.uportal-input-text{
    color : Black;
    font-size: x-small;
    font-family: Monaco, Andale Mono, monospace;
    background : #CCCC99;
}
.uportal-text-reversed{
    color : White;
    font-family : Georgia, "Times New Roman", Times, serif;
    font-size : x-small;
}
.uportal-crumbtrail{
    font-family : Verdana, Geneva, Arial, Helvetica, sans-serif;
    font-size : xx-small;
    color : #666666;
}
.uportal-copyright{
    font-family : Verdana, Geneva, Arial, Helvetica, sans-serif;
    font-size : xx-small;
    color : Black;
    font-style : italic;
}
.uportal-channel-text{
    color : Black;
    font-family : Georgia, "Times New Roman", Times, serif;
    font-size : x-small;
}
.uportal-channel-title{
    color : #FFFFCC;
    font-family : Verdana, Geneva, Arial, Helvetica, sans-serif;
```

```
        font-size : xx-small;
        font-weight : bold;
    }
    .uportal-channel-title-reversed{
        color : #663333;
        font-family : Verdana, Geneva, Arial, Helvetica, sans-serif;
        font-size : xx-small;
        font-weight : bold;
    }
    .uportal-channel-subtitle{
        color : #999999;
        font-family : Verdana, Geneva, Arial, Helvetica, sans-serif;
        font-size : x-small;
        font-weight : bold;
    }
    .uportal-channel-subtitle-reversed{
        color : #333333;
        font-family : Verdana, Geneva, Arial, Helvetica, sans-serif;
        font-size : xx-small;
        font-weight : bold;
    }
    .uportal-channel-emphasis{
        font-weight : bold;
        font-size : x-small;
        font-family : Georgia, "Times New Roman", Times, serif;
    }
    .uportal-channel-strong{
        font-weight : bold;
        font-size : x-small;
        font-family : Georgia, "Times New Roman", Times, serif;
    }
    .uportal-channel-code{
        color : Black;
        font-size: x-small;
        font-family: Monaco, Andale Mono, monospace;
    }
    .uportal-channel-copyright{
        font-family : Verdana, Geneva, Arial, Helvetica, sans-serif;
        font-size : xx-small;
        color : Black;
        font-style : italic;
    }
    .uportal-channel-warning{
        font-size: xx-small;
        font-family: Verdana, Geneva, Arial, Helvetica, sans-serif;
        color : #993333;
    }
}
```

```
.uportal-channel-error{
    color : #993333;
    font-size: xx-small;
    font-family: Verdana, Geneva, Arial, Helvetica, sans-serif;
}
.uportal-channel-table-caption{
    color : #993333;
    font-family : Verdana, Geneva, Arial, Helvetica, sans-serif;
    font-size : x-small;
    font-weight : bold;
    text-align: center;
}
.uportal-channel-table-header{
    color : #993333;
    font-family : Verdana, Geneva, Arial, Helvetica, sans-serif;
    font-size : x-small;
    font-weight : bold;
    text-align: left;
}
.uportal-channel-table-row-even{
    color : Black;
    font-size: xx-small;
    font-family: Verdana, Geneva, Arial, Helvetica, sans-serif;
}
.uportal-channel-table-row-odd{
    color : #666666;
    font-size: xx-small;
    font-family: Verdana, Geneva, Arial, Helvetica, sans-serif;
}
.uportal-background{
    background : #000000;
}
.uportal-background-dark{
    background : #666633;
}
.uportal-background-med{
    background : #999966;
}
.uportal-background-light{
    background : #CCCC99;
}
.uportal-background-content{
    background : #FFFFCC;
}
.uportal-background-highlight{
    background : #FFFF99;
}
```

```
.uportal-background-shadow{
    background : #000000;
}
.uportal-background-page{
    background : #FFFFFF;
}
.uportal-background-semidark{
    background: #7F804D;
}
.uportal-background-selected{
    background : #FF9933;
}
.uportal-background-neutral-dark{
    background: #666666;
}
.uportal-background-neutral-light{
    background: #CCCCCC;
}
```