TYPESETTER |

Jean Sébastien Beaulieu

Supervisor: Michael Longford
Concordia University

August 2004

TYPESETTER |

Thanks to:

Jean-François Côté pour la programmation,
CIAM pour les sous,
et a Mónica pour l’Espagne

TYPESETTER |intro

Modularity and variability 3
Giving control 4

TYPESETTER | design

Graphic Spec 7
  Screen Layout 7
    Control Panel 7
    Page Panel 9
  Color Palette 10
  Color Coding 11
  Interface Elements 11
  Iconography 14
The Controls & Functionality 16
  Font Style 16
  Font Size 18
  Line Spacing 19
  Column size 19
  Contrast 20
  Layouts 20
    One column content 21
    Content and link listing 21
    Content and images listing 21
    Content and outline listing 22
  Links and images visibility buttons 22
  Address Field 23
  Format 23
  Collapse Bar 23
  Visit Original Page 23

TYPESETTER | engin

The Big Picture 25
Technologies 25
  Flash 25
  PHP 25
  XHTML & CSS 25
The Structure 26
  display.swf 26
  index.php 26
  html.php 26
  htmlparser.inc 27
  css.php 27
The Process 27
  Overview 27
  onLoadPage() 27
  onNewSetting() 27
  onFormat() 28
The Modules 28
  DISPLAY.SWF 28
  INDEX.PHP 29
    URL Queries 29
    Printing Out The Page 30
  CSS.PHP 30
  HTML.PHP 30
    About HTML Tags 31
    The Tags We Want 31
    Building The Layout 32
    Creating An Outline Layout 33
    Dealing With Images 33
    The Filter 33
    Dealing With Links 34
  HTMLPARCER.INC 35

TYPESETTER |

The latest version of theTYPESETTER can be found at
http:://www.nonobots.com/typesetter

A copy of TYPESETTER is also included on the cd that came with this document.

In order for it to work properly,it need to be installed on a server running PHP 4.

TYPESETTER |

introduction

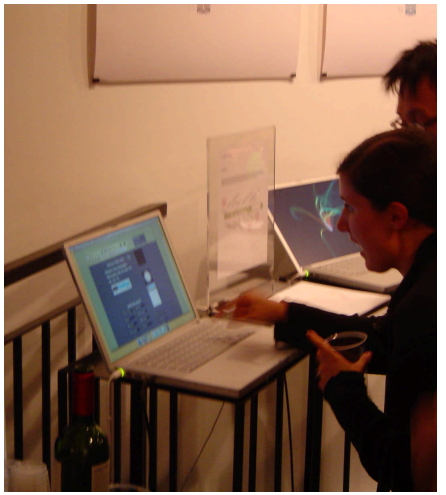
TYPESETTER | intro

To determine the best layout for a web-page is a classic design problem where you have to balance science and aesthetic.

To reformat and simplify the design of a website can easily create a really boring page. Like many other version[1.1] of this idea my first prototypes were just spitting out plain html. From the beginning, it was obvious that this over simplified page could not stay in this basic web formatting state. For example the default font of most browsers is Times, a serif font. That typeface doesn't come out that great on a computer screen. The little serifs were not design for the screen. If you use serif font to render a text on a computer screen you are just wasting pixels.

When you start looking a little harder, you find that one of the reasons that Times is the most readable font is because it's the one people are the most familiar with. It's familiar because it's the typeface publisher and editors of newspapers and book have been using for the last couple hundreds of years. People are use to seeing it and they have facility reading it. It's like a learned behaviour! But trouble is, a computer screen just doesn't have the resolution of the printing press.

The reason why it's used on the web is most probably link to the fact that to be displayed on the screen of a browser, a font must be available on the user's computer. Everybody has Times because it's the default type face for your average word processor. And the reason why it's the default is probably because that's what the average word processor user is used to read when he prints out a paper copy. Again, back to print. In the



1.1 One of the most awful example is Usablenet's Transcoder. There solution is deprived of any style or any aesthetic sense.

end, that's the fundamental problem with the Times type face, it wasn't design for the screen, it was design for print. It was created for legibility and economy of print space.

I could argue that there is no need to have Times as a choice for screen font. But technology is evolving, the screen are getting finer and sharper. The computer are now more powerful and have power to spare and can use part of there processing speed to display writings more intelligently[1.2].

This pseudo problem with the Times typeface is typical of my research. It seams that for every obvious wrong, ten small rights came to counter balance.

I have read many studies on how people read on the web and on what is the best type to use Turns out that there is no magical layout or optimal type face out there. I found some interesting studies from the Department of Psychology of Wichita State University. They have a special research department called Software Usability Research Lab were they test graphic layout for performance and efficiency. One might think that they would have found the ultimate font by now or the maximum of button you can put on a screen. But not really. Sure they have a whole set of recommendations, but it seems there's always a human factor that comes in and blurs the result. Their solutions for performance rarely matchs there subject's preference and what they perceive as efficient.

These study convinced me of the good of letting the user have it's say on how he want to see things. On my interface, I tried the give a broad enough range of controls. A range that goes from the efficient to the pleasing.

1.2 Opentype fonts have build in information on how to be displayed on screen. Microsoft and Apple operating systems have font smoothing capability.

This project is guided by the following principle: providing an easy to read layout and keeping in mind that there are many ways of reading. Not everybody feel the same way about reading on the web.

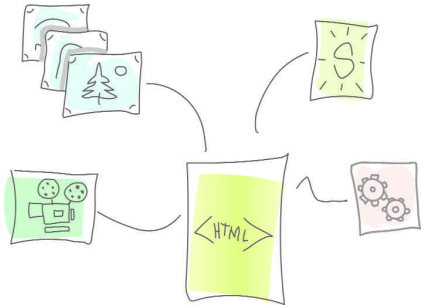
Modularity and variability

Web pages are digital object. They are variable by design. They are malleable in ways that people don't imagine. The way they are constructed give us the possibility to modify them[1.3].

Webpage are made of various parts that all comes together on the users browser. That is a major difference between web and the other media. A newspaper for example, is put together by the editor and sent to the presses to be fixed on paper. What the reader gets is a rigid object. Maybe he can cut out the letters and reassemble them to make a ransom letter but that's about as much variability you will get from a news paper. When you access a webpage, you are downloading a set of instruction that gives the complete description of how to build that page. The core html page has the text. Then you have the different addresses of were to find the images, another address where to find some dynamic content or some special graphic styling. Just the fact that they rely on fonts that are on the user computer tell us something about how malleable they are.

Ultimately, if you know how to read an HTML document (and it's not that hard) you can access any of its part individually.

The idea for the project came out of desire to explore the power of the styling component of web page. Styling of a web document is done trough Cascading Style Sheet (CSS). CSS are text docu-



The structure of an html document

1.3 Based on the concept of modularity and variability described by Manovish in “the Language of New Media”

ment instructing the browser on how to display its content. They can do much more than background colors and blinking links. With them you can modify any property of any object on a page.

As of the latest specification of HTML[1.4] instructions affecting the styling of the content of a page should to be move out of the body of the text and ideally in a separate document.

Modularity is a way of thinking that is probably going to influence a lot of the thing we are going to do in the future.

Having Individual component coming together to make one object and the possibility to change the property of any of the parts. An object that can be made into something else by switching one part for another. This is what digital object have to offer.

This project is about the customization of a digital object. I chose to act on the webpage first because it the stuff I work with every day. But also because I find it's one of the first modular digital object that is available and experienced by the general public. Although they are understand as finite object the possibility is there to act on the object's properties, to customise it at will.

Giving control

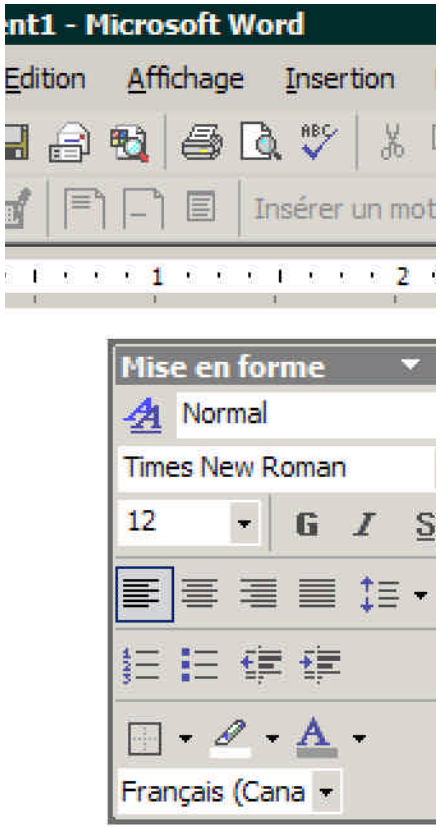
I want to give people some control on the HTML they read.

Yes it's very nice to give control and everyone wants more control. Some others would say it's a good way to make sure you don't have to take any decision. But it’s not that easy. How do you decide how much and what control to give is not that simple.

4 XHTML™ 1.1 - Module-based XHTML W3C Recommendation 31 May 2001

People have access to HTML code and HTML is quite simple. But it really doesn't mean anything to the average user. What they see when they are in front of a website is digital text. For them to know that a paragraph is marked by two <p> tags is of no use what so ever. Furthermore, to base the controls of my project on HTML would require the user to learn something new on top of the interface itself.

I choose to follow a simple software developers guideline. “Give control to the extent of what your user understand“[1.5]. I take for granted the fact that the vast majority of people using computers are familiar with word processing software. What I propose is that the user acts on the contentment of web page as if he was in Word Perfect or Microsoft Word. Appropriating the content and adjusting it to his liking by operating a set of controls similar to the one he knows from his word processor.



The style panel of Microsoft Word.

1.5 Microsoft software developement guideline

TYPESETTER |

design

TYPESETTER | design

The TYPESETTER is an environment the lives inside a browser window. All the browser controls are still available and functional. The core component of its interface is its control panel. The control panel is in line with the other toolbar of the browser and even share some of it's functionality but it's drastically different in style.

Graphically, I decided to pursue the style I have been developing all year for the seminars. Clean simple design that combines hand drawn lines and subtle gradient.

Graphic Spec

The TYPESETTER application takes place in one single web page generated by PHP. Its design to fit inside a browser window expanded to full screen on a computer monitor set to a resolution of 1024 pixels by 768 pixels. On a 17 inch monitor, witch are commonly use, this resolution offers a good balance between screen real estate and image quality[2.1] Also, the vector graphic based control panel needs a good resolution in order to render its fine details.

The TYPESETTER interface is 960pixel wide and centered to the middle of the browser window. It's divided in two distinct parts. On top is the control panel and below, is where the resulting re-formatted html page is displayed.

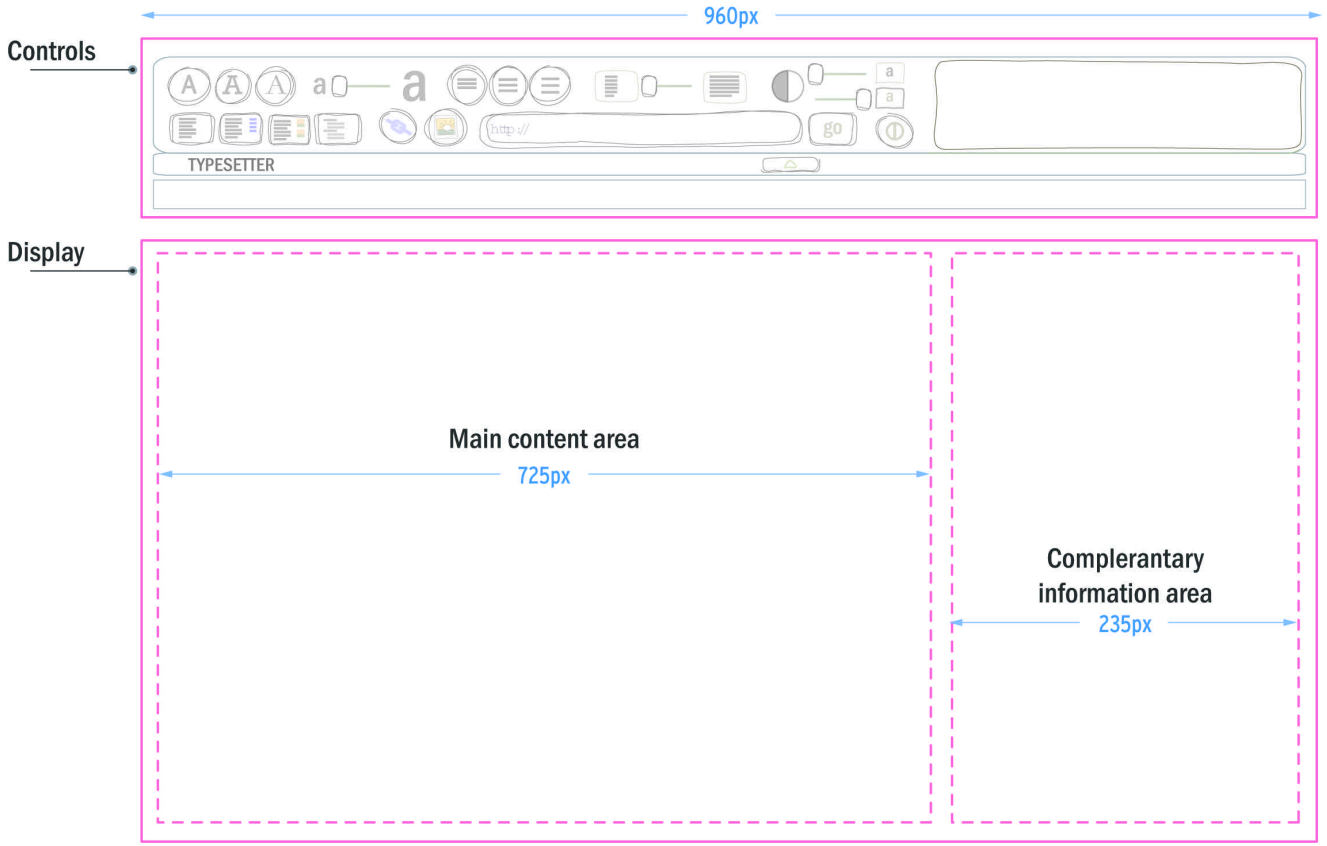
Screen Layout

Control Panel

The control panel area is 102 pixels high and occupies the top part of the browser window. The main part of the control panel is a flash movie/application of 82

2.1 As of september 2003 more than 51% of web user had their monitor's resolution set to at lease 1024 x 768. http://www.dreamink.com/design5.shtml

TYPESETTER | Screen layout



pixels high by 960 pixels. The width of the flash was set considering the possibility of other object existing in the user browser windows. Objects like vertical toolbars, scroll bars, side tabs or other plug-in elements that a user could have added to his browser. These elements take up screen real estate. Further more, we can't expect that the user will turn off any of is browser setting. This app only requirement is that the resolution be set a 1024.

When the browser window is maximised, the control panel is visible in its entirety and is not touching the edge of the windows.

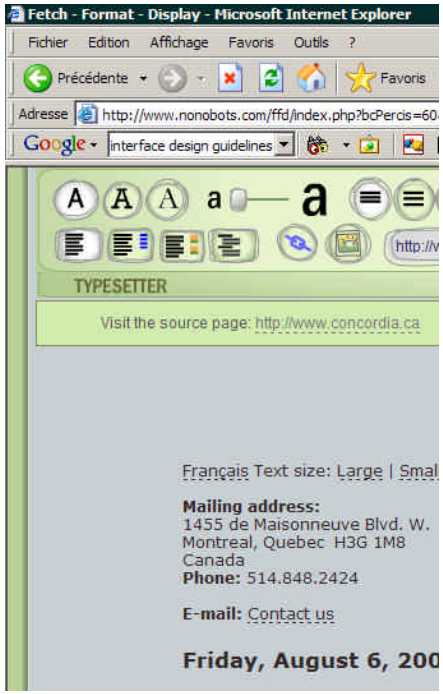
Since fine vector graphic elements were use in the design of the panel, it was decided that the panel would not resize automatically. It was observed that when the flash movie is reduced to a smaller size, fine graphical element became blurry.

Second part of the control panel is the title / collapse bar. This 20pixel high jpeg graphic is one big button. It's used to hide the control panel when it's not in use.

Page Panel

The reformatted page area takes up the rest of the screen. The actual active area or "the page" is as wide as the control panel above. And for the same reason: when the window is maximized and all information should be visible and horizontal scrolling must be avoided.

Two gradient shadows can be seen on each side of the page. A long slim graphic is use in the background to create the effect of an elevated page. The image is a long transparent area with to small gradient a both ends. This optical effect gives the illusion of a shadow and helps to break the overall flatness of the page.



The gradient edges on the side of the page.

Color palette for TYPESETTER

This panel is composed of two vertical areas: the main content column and the optional complementary information column. The visibility of the second column depends on user choice of layout.

The content panel column's width is 725 pixels. The position of the text in that column is set relatively according to the user setting. The smaller the column the bigger the space will be from the edge of the window. One of the principles guiding this design was that there should always be ample of space to rest the eye. Element should be well isolated from each other and are given a lot of breathing space for the reader. Margins are set relatively from the center of the section.

The complementary column as a set width of 235 pixels. It is visible when the user chooses either the "link listing", "image listing" or the "outline" layout.

Color Palette

The color that were chosen for this environment are Light green; Silver; Dark Olive; Magenta, Blue, Orange and Purple.

Green is the dominant color in this interface. Green is known to be a calm relaxing color. This particular light dull green is inspired by the old desktop cover use by teacher. Almost the same green as the rubber covers of drawing table. I find it to be a color that relate well with the world of study, reading, libraries.

The Control panel is filled with a subtle gradient of dull green going to a lighter green. This gives it a soft, not to shiny, reflective surface. The same gradient color is used on the button, but as a radial gradient.

The interface elements don't have hard edges. Soft hand drawn lines are used

to indicate the edges of the components. Combination of radial gradient with free floating hand drawn line creates a panel that is in between object and drawing.

Buttons are filled with a lighter green radial gradient and have a silver line around them. On roll over they turn to a white – silver gradient and the outline turns to magenta.

Sliders are always active, so they keep their silver white background. When a slider is moving the outline turns magenta.

Color Coding

Silver is used to outline buttons and any dynamic object when in their idle state.

Olive use on symbols and non clickable iconography. It is used as a second color (the first one being black) on various labels.

Magenta is use to indicate an active state.

Blue, is related to hyper links,

Orange is related to images

Purple is the color related to headings.

Interface Elements

The types of interface component that have been chosen to activate the controls of the TYPESETTER interface are the followings: Toggle buttons, radio buttons, Sliders, Text entry field and Text area.

It was decided to limit the extent of the controls available to the reader to those he would find in a basic word processor[1.2]. The point is not to re-stylize the page to make it pretty. The goal is to re-

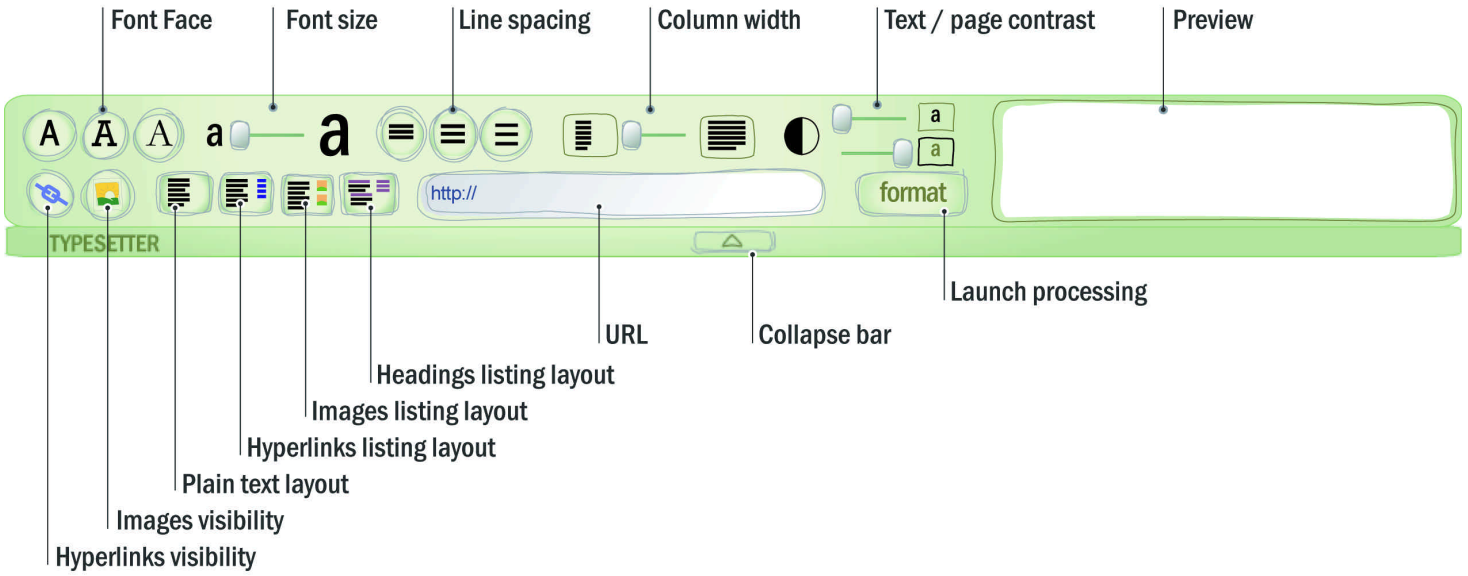
Radio buttons

toggle button

Slider

1.2 See chapter1, “Giving control”

TYPESETTER | Control panel



format the page so it's easy and pleasant to read.

Radio buttons force you to choose between a limited set of option of which one as got to be selected. This type of control is used to select the font style, the line height and the layout style.

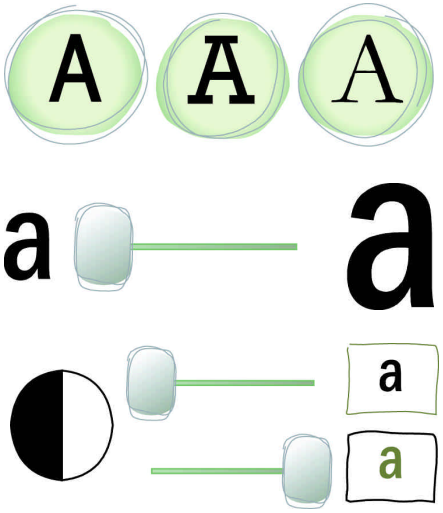
Options needing some sort of value input use a slider. For this interface, since any value of size is ultimately relative to the size of the user screen, there is no real need for precise measurements, the choice of a slider type of control was chosen for option requiring a quantity or scale value. This includes font size, column width, background color and text color

Toggle buttons. The two option left are the visibility setting of links and images. These two options are independent and are either on or off. They are activated by regular toggle buttons.

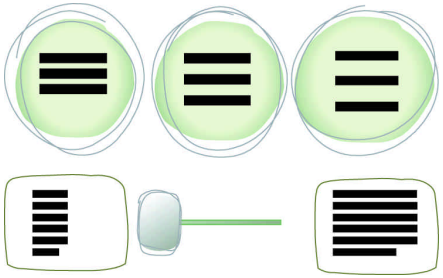
A preview window situated at the right side of the control panel. It gives immediate feedback to the user on the change he is making on the page layout. This device prevents the user from having to wait for the whole page to be processed before seeing his changes. The processing of the webpage is done from the server and there is always a delay before the new page is displayed.

Address field is a text input area where the user types the address of the webpage he wants to access. Just like a regular browser, if the user chooses to navigate using the links on the page, this field gives the current address of the page.

The Controls are laid out on two rows. From left to right, we start with the letters controls then controls affecting the text. On the second row, we have the choice of layout, the address field and finally the format button which activates



The letter “a” is used on controls affecting the text.



Controls affecting the display of the body of the text. : Line Spacing. Column Width.

the processing. This is loosely set on the order in which the user would set his parametres

Iconography

Most of the symbols used on the control panel are aligned on the controls that are found on most word processing software. These symbols have proven to be the most efficient in representing the element of a page. At first glance, this interface is not what a regular computer user would encounter very often. Having recognizable symbols also speed up the learning process.

I chose to use the letter “a” to identify all control applying to the text (text color, text size, and text font). In order not to confuse this control with the styling controls, I chose a small cap “a” therefore keeping the relation between the two similar control. And staying stays inline with the established visual language.

A set of three evenly spaced lines is used to suggest the line spacing. This is similar as what most word processor use.

To suggest the column size, I chose to use the representation of a paragraph found on the commonly used “alignment” symbol. This symbol can also be found in numerous word processing software. The symbol is a set of 5 evenly spaced lines the first 4 being of the same width and the last one is a little shorter.

This paragraph symbol is used in a slightly different context here. To suggest a small column the symbol as been compressed horizontally, for a large column it has been stretch a little more then its commonly known proportions. The symbols are bordered by in a rectangular outline. This can be interpreted as the screen of the computer or the browser window. It gives a reference to

the user about the space around the text. A slider controller was put in-between. Sliding toward the slim column symbol reduce the column size and vice versa.

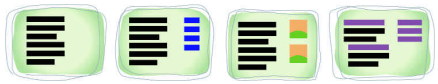
The layout icons are simplified visual representation of the actual layout.

Layout icons reuse the paragraph symbols and are also displayed inside rectangular shape. Unlike the column size icons, which are only visual label, these one also act as buttons. That is why they are outlined by a double silver line and have a roll over state. The first icon describes a single column of text in a screen.

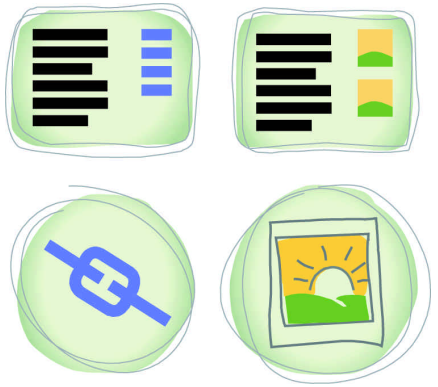
The second layout button is the content and links listing layout. It is represented again by the column of text symbol and, on its right three blue lines to represent links. The color blue was chosen because of its historical relation to the hyper links. By default, hyperlinks have always been displayed as blue underlined text. Since text is already represented by a line, just having the blue coloured lines seams to be enough. It didn't make much sense to have an underline line on this button. It would be a bit redundant also.

Third layout is the content and image listing. For consistency, it uses the same column icon as the previous button. Instead of blue links, it has two little orange and green squares as images symbol.

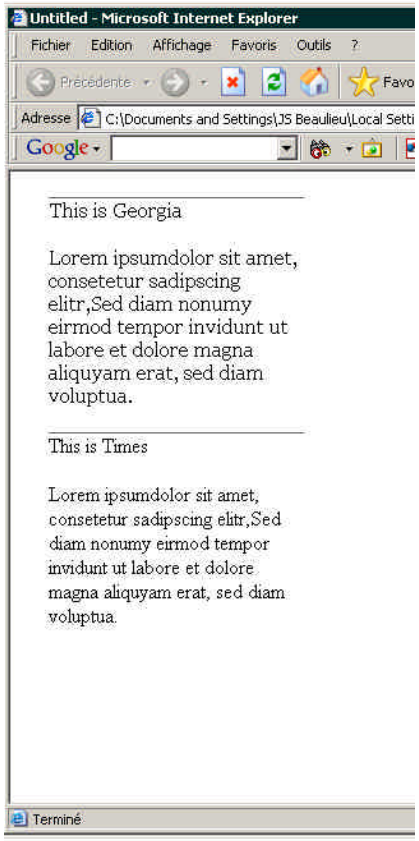
Images are represented in various ways in the digital world. On most operating system, it's represented by a photograph. But the subject of the picture varies. Picture frames with house, Polaroid style plus landscape. Sail boat, geometric shape etc. Representing a picture is hard because it's a support for



The 4 layout options.



Same color relationship between layout and visibility buttons.



Two serif font as seen in a web browser window.

something else. Whatever is on the photo can be mistaken for the object.

Another problem encountered is the extremely small space available on the button to represent the picture. The solution was to create a visual connection with another, more explicit button. The image visibility button is right next to the layout button and it offers four times the real estate. On the big button we can have a more elaborate picture symbol. The representation of a sunset was chosen mainly for its dominant colors: orange. So by putting a small orange square on the layout button we create a color relationship between the two.

The same color relationship strategies are also used for the links visibility button. The chain link symbol is use in many WYSIWYG HTML editors and in some word processing software to represent hyper link. It's the same shade of blue as the three small lines in the second layout toggle button.

The last layout button is the heading listing layout. Purple is the color we chose to associate with headings. On the column of text symbol we added some purple lines. The lines represent the headings found at the beginning of an article. On the right column, three purple lines represent the listing. We use only three in order to differentiate it a little more from the links listing which has four lines.

These colors are carried over in the actual output of the formatted web page. Links are blue, and images are framed with an orange outline.

The Controls & Functionality

Font Style

There are 3 font styles available: Georgia, Verdana and Courier. The choice of the font are based on studies conducted by the Software Usability Research Laboratory (SURL) of Wichita State University. They have made many test on the different fonts available in web browser. They tested them for reading speed and perception of legibility. Their study reveals no clear winner as of readability quality, but some font faces were found to be perceived as being more legible[2.3].

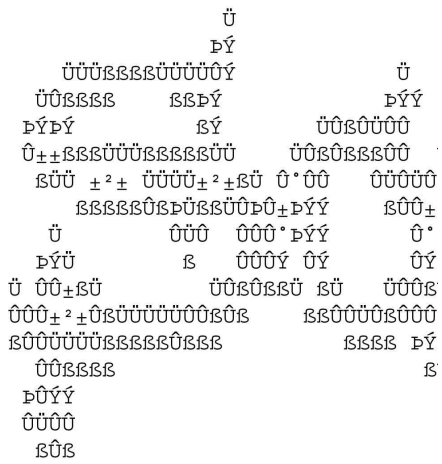
Georgia is a serif font. This type of font is known to be the most easy to read font. This reputation is partly based on the fact that it's the most common type used. It may be true on the printed world but it's not completely true when it comes to screen type. Bad screen resolution makes this kind of font crunchy and not very pleasant to look at. But with new graphic cards and powerful computer and new font format like clear type, recent computer can smooth the edges of the letters.

Georgia was designed specifically for computer-display. To make it more legible for computer-screen viewing, its uppercase characters were lightened and the height of some lowercase letters was increased. Research by Boyarski, Neuwirth, Forlizzi, and Regli (1998) examining Times, Georgia, and Verdana fonts on computer screens has found that Georgia was significantly perceived to be easier to read, sharper, and more legible than Times.

Verdana is a Sans serif font like Helvetica and Arial. It a font face developed by Microsoft Corporation for optimal readability on screens. It has a wider letter spacing and letters like i j and l are more distinctive then in other

Comparison between the letters i j and capital I from the Verdana font and the Arial font.

2.3 Bernard, M.L., Lida, B., Riley, S., Hackler, T., & Janzen, K. (2002).



ASCII signature by Team S.C.O.T.C.H.

similar font type. This type of font is usually the font of choice of graphic interfaces and screen medium. They are good for medium resolution screen like the on we have been using for the last 10 years.

Courier is a typewriter-like font. It's a monospace font which means all of its letters occupies the same amount of space. This kind of type face is actually the first kind of digital font that was invented. It was used on back when computers were named "terminal" and monitors only displayed text. It's still the preferred font of programmer everywhere. With this kind of font you can organize text visually and align blocks of numbers in columns. It's also the only way to enjoy ASCII art.

The fonts displayed in a web browser always rely on the font installed on the computer. In the event that the user doesn't have the fonts mentioned above, a set of default fonts is used. Georgia is replaced by the users default serif font, most likely Times. Verdana will be substituted by Arial or Helvetica. Courier is usually the default monospace font on most PC. It might be replaced by the Monaco font on Macintosh

Font Size

In word processing applications, font size is associated with a numerical value. As mention earlier, this application does not use any specific value for its settings. A numerical value as a font size may be useful if your final output is a physical object like paper. But not here. In an environment like a computer screen, relative measurements are much more meaningful. Some of the latest browsers already give you the possibility the change size of font. And they refer to the font size in relative terms. They use terms like big, bigger, smaller,

smallest or they use plus and minus sign.

The size of the text is control by a slider. Apart from the reasons mention above about the use of specific measurements, slider controls have been chosen because in most cases more that one font size are used on a single page. Headers are bigger than paragraphs and paragraphs are usually written using a bigger font than footer notes. A slider relates better to the concept of increasing and reducing the size of the whole body of the text. Therefore, we keep the proportion and relation implied by the different sizes of the text elements.

The text size of the page elements is set according to the size of the text body. The smallest font size this interface will render is 8 pixels and the biggest is 24 pixels.

Line Spacing

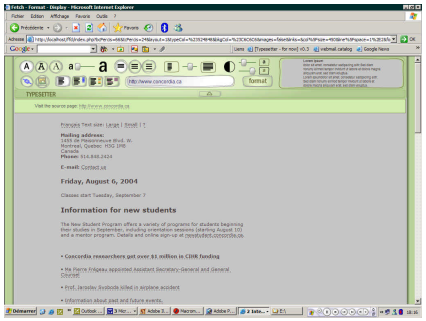
Just like in a word processor or on a typewriter, this interface give you to possibility to choose between 3 different line height I believe that line spacing is a simple option that all web browsers should have. Where increasing font size is an existing feature in almost all the latest browser, line spacing is not.

Like for the choice of fonts, performance test have revealed no dramatic differences between text with no line spacing and text with line spacing[2.4]. But most of the subject felt or perceived the text with line spacing to be more legible. Of course, a well design font should already have optimal leading. But this interface is about comfort and choice, so the choice as been given to the user.

Column size

We are using a slider to adjust the width of the column of text. The user can

2.4 Chaparro, B., Baker,R.J., Shaikh, A.D., S. Hull, L., Brady, (2004)



Layout no.1 : One column content

choose from a narrow, newspaper like, column to a full screen wide layout.

In HTML, text flows from left to right until it reaches the edge of the windows. On simple web pages, there is no limits to the width of a page. If someone use a large monitor and expand a browsers window full screen, he will and up with a document composed of long paragraphs of one or two lines. Long strings of word are difficult to read.

Putting text in a column also create a margin. In another studies[2.5] made at the SURL, the subjects found that text with margin were easier and more pleasant to read than text without margin, but interestingly, the performance on reading speed was faster on text with no margin.

Contrast

Two sliders enable the user to adjust the contrast between the text and the background without having to change the monitor calibration. This is useful because you apply it to one single window and not to all your system.

These control also give you the possibility to have a dark background and light text. A lot of programmers like to work on (dark) background. Programmers work on computer screen for long hours and they find it less tiring. Is it nostalgia of monochromatic cathode ray tube screen or is it really useful?

For now the settings only allow to choose shades of grey going from pure black to pure white. In future version, a color palette will be introduce allowing the user to change background and text colors.

2.5 Chaparro, B., Baker,R.J., Shaikh, A.D., S. Hull, L., Brady, (2004)

Once all the text modification options have been chosen, the user can choose how the content of the page will be display.

Layouts

The different layout options give the user the opportunity to rapidly scan trough the content of a page. Three layout have been pre define for this version. A one column content layout, a two column content and link listing layout and a two columns content plus images listing layout.

One column content

This is the simplest layout. It displays the content of the page as is in one straight column in the content area. The size of that column has been set using the column size slider. The complementary column is left empty.

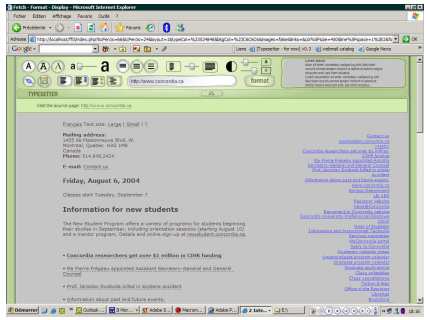
Two column content and link listing

In this layout, the left column displays the content of the page. The complementary area, on the right, gives the user a listing of all the hyper links used in the page. The links are displayed in the classic styling of hyper links: underline blue text.

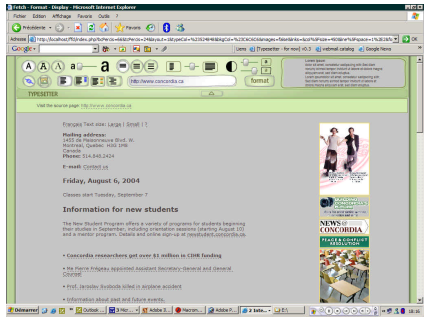
Two columns content and images listing

Again, left column displays the content of the page. The right one give the user a list of the images used in the current page. The images are proportionally resized to fit the width of the column and are framed by an orange border.

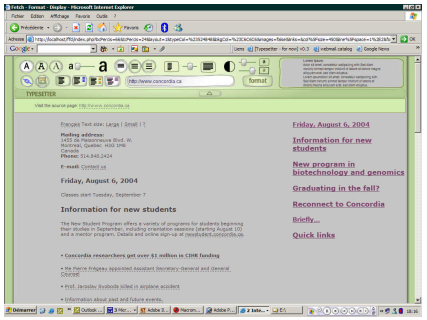
Not Absolutely all images are displayed. Images smaller then 15 pixels of width or height are considered as being of “cosmetic purposes” and are not consider as being relevant to the understanding of the text. One of the most commonly encountered semantically empty



Layout no.2 : Two columns content and link listing:



Layout no.3 : Two columns content plus images listing:



Layout no.4 : Two columns content and headings listing:

image is a 1 x 1 pixel transparent image use as (placeholder / spacer) to help fix the layout of a web page.

Other unwanted images include graphic menu bars and banner ad. In order to discard these images from our listing we used to following strategy[2.6].

Given that the vast majority of these kinds of images have long rectangular shape, we calculated the ratio of the width of the image over its height. Any image having a ration superior to 2.5:1 is discarded. This method was proven successful in 75% of the case. For example, a regular banner ad size is 60 pixels by 468 pixels which giving it a ratio of 7.8:1.

Two columns content and outline listing

This layout gives the user the possibility to rapidly scan the content of a web page by going trough all its headings.

According to J.Nieson[2.7], people don't read on the web, they scan quickly in search of what would interest them. The headings from the listing are all link to their occurrences in the body of the text. So the user just has to click a heading and the window will scroll to the corresponding paragraphs.

Links and images visibility buttons

These two buttons enable or disable the visibility of in line images and link in the main content column

The purpose of this feature is to reduce to a minimum the visual distraction and the temptation to click on a link and go somewhere else. If we take the example of the hyperlinks, visually having a word underline gives it emphasis which it may not necessarily deserve. A word with a hyperlink in a given phrase is not necessarily the subject of that phrase. In a context were people are scanning trough(Neison 1997) page to get was

2.6 See Chapter 3 "Dealing with Images" for a more detail on that process.
2.7 Jacob Neison's Alertbox for October 1, 1997

they are looking for, attracting the attention on the hyperlinks may not be the best way to help the reader understand a page.

When hyper links are mask, the user can still access them; they are still active and functional. When the user brings the mouse pointer over a link, the word change color to indicate it is active and clickable.

When set to visible, hyper links are in indicated with a thin dotted line. It makes the link less intrusive and less likely to disturb the reader.

Address Field

This is where the user type in the address of the web page he wants to see re-formatted.

It also displays the current web address just like any browser would do.

Format

This button launches the application. On click, the user's preference and the address of the page he want to consult are sent to the server to be processed.

Collapse Bar

Just below the control panel, we have a collapse bar. This device allows the user to hide the control panel once he has chosen his display preference. This makes the whole thing less obstructive.

Visit Original Page

At any time, the user can go out of the TYPESETTER and go to the original page he as reformatted. The webpage open in a new window.

TYPESETTER |

engin

TYPESETTER | engin

The Big Picture

The TYPESETTER is a web application that is made of many distinct parts that interact with each other to create the re-formatted pages. The interface is on the client side (the user's browser). It is the only "visible" component. Its role is to collects the data from the user. On the server side, one main PHP file calls upon other secondary processes to collect information, dissect html code and create the layout.

Technologies

Flash

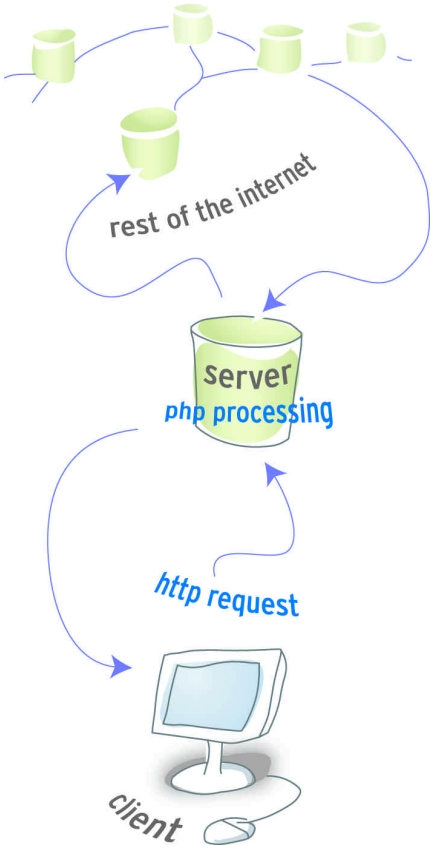
The control panel was built using Macromedia Flash and its ActionScript language. They were chosen for both the ease of use of the scripting language and it's capability of rendering great hi-definition and dynamic vector graphic.

PHP

The bulk of the software is done on the server[3.1] were the site is hosted. All heavy processing is done trough PHP scripts. PHP is a general purpose scripting language used to create dynamic, data-driven websites.

XHTML & CSS

The pages generated by the TYPESETTER follow the guide lines of the latest version of hypertext markup language: XHTML. As it is recommended, all the styling and the layout is done trough Cascading Style Sheet (CSS). CSS allows separation of content and presentation. It controls the visual property of the objects that make up the web page.



The path of the HTTP request

3.1 The current running version of the Typesetter runs on an Apache server running PHP4.

The Structure

The TYPESETTER is made up of 5 distinctive modules that all have a role to play is creating the pages.

display.swf

Display.swf is the interface of this application it's the only part visible to the user. The control panel was built using Macromedia Flash and its ActionScript scripting language. It registers the user setting and transmits them to the server. Main reasons for this choice were the stability of the flash player on different platforms and browsers. It was also chosen for the simplicity of its scripting language.

index.php

Index.php is the core of the application. It's both where the process begins, and where it ends.

When index.php receives the values from the interface, it dispatches them to the different processes used in making the new page. It sends the URL data to html.php and sends the style values to display.php. Once html.php is done with the parsing and the style sheet is generated, index.php puts it all together and delivers the new page to the user.

html.php

It's in this file that all the real processing takes place.

The functions in this file analyse the tags that have been detected by html-parser.inc and extract all the pertinent content. Then they feed it to index.php. At the same time, it classifies objects and properties and builds listings of all the elements needed to create the different layout[3.2].

3.2 Four different layout are available: plain text, image listing, link listing and outline view

htmlparser.inc

The htmlparser.inc[3.3] is a php class that is used to loop through all the tags and content composing a html document.

css.php

Using the data collected by the control panel, css.php generates the style sheet that will set the general appearance of the page.

The Process

Overview

First the user as to enter his preference using the control panel at the top of the page. The user set the appearance of the text and chooses one of the available layouts and of course the address of the page he wants to see reformatted. These settings are then sent to the server to be processed. The server read the data, writes a new html page according to the user's choices and sends it back to the user.

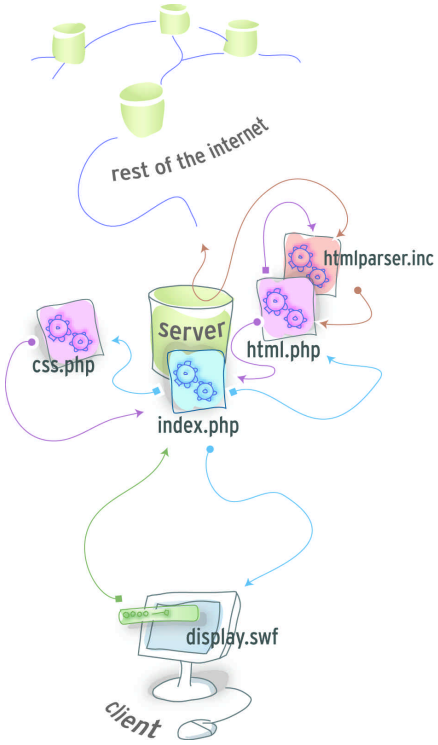
Of course we are not doing anything to the original page, it stays as it is on its server. The TYPESETTER read the page and copies the info that is pertinent to a new page that is generated by the server.

onLoadPage()

When the TYPESETTER is loaded for the first time, index.php is generated using default value. The Flash control panel read the value, adjust its controls accordingly to reflect the settings. Since no URL have been requested at this point, html.php has noting to parse. Instead, it generates the default instruction page.

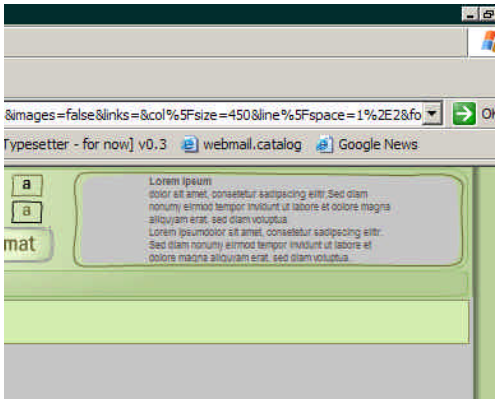
onNewSetting()

By activating the various controls on the TYPESETTER interface, the user sets



Data flow chart of the TYPESETTER.

3.3 htmlparser.inc is an open source document that was developed by Jose Solorzano (http://jexpert.us) with contributions by Leo West.



The preview window of the TYPSETTER's control panel.

new values to the style variables. These variables will be sent to the server.

Whenever a setting is change, a call is sent to the preview window at the right end of the panel. The dummy text in that window react accordingly to reflect the output the user will get.

onFormat()

The control panel calls upon index.php to reload itself using the newly set style value. The values are transferred to index.php trough an http request. On receiving the new values, index.php sends the style variable to css.php to create the style sheet.

Then he sends the URL to html.php to be parsed.

With the help of htmlparser.inc, Html.php analyse the URL and start going trough its tags. When it finds an open tag, it checks to see if it's a required tag. If positive, the tag is collected and sent to index.php to be display.

While going trough the code, it also collect information needed to generate the special layout available to the user. Link listing, Image listing and outline listing

When the closing tag is encountered, html.php continues to the next tag in the source page, until the end of the document is reached.

When the entire source page has been processed, the new html page is displayed in the user's browser.

The Modules

DISPLAY.SWF

The user's preferences are first processed by the control panel using ActionScript. For example, In the case of a toggle button like font face, a numeric value is set to a variable named font_face which will be later sent to the server. In the case of the slider type controls, the position of the nod is evaluated and transformed into a value that will be meaningful for the CSS processes. A "meaningful" value is for example the name of a font or the hexadecimal value of an RGB color. When the user hit the format button, display.swf sends an http request to index.php and sends along the users preferences.

The only real dynamic part of the control panel is the preview window. The preview windows read the value of the controls as they are change by the user.

INDEX.PHP

Index.php commands all the processing. Here is a summary description of some the functions that are taken care of by index.php.

URL Queries

To make sure the software user proof, the basic behavior of a regular browser had to be programmed in. The address field of the control panel is the only input the user has to type. The text he entered need to be analysed and completed to make sure it's a valid http url.

First it checks for a complete URL address. If http is not there it will be added.

There is two type of URL you can enter in a browser, you can type the address of a document (most likely ending with the famous .html file format) or you can type the address of a folder (which

```
$font_face = "sans-serif";
$font_size = "10";
$line_space = "1.2";
$col_size = "450";
$links = "false";
$images = "false";
$bkgCol = "#bad19a";
$typeCol = "#000";
$url = "Please enter a location";
$layout = "1";
```

List of the variables and their default values

would end by a slash or just the name of the folder). If the address does leads to a folder, then the application as to look for a file named index.html.

Printing Out The Page

Index.php is in charge of rendering the reformatted page.

Using the data that css.php as processed, index.php print the CSS style sheet. Then it inserts the control panel and set it up with the new value.

Then it calls html.php to start working. Index.php printout the content html.php produces from the page being processed,

Once the page as been completely “squeezed” out of its tags, a second call is place to html.php to deliver the content of the layouts. By default, the four layouts are generated regardless of the user’s settings

CSS.PHP

Css.php receives the style information from index.php and generates a style sheet. This style sheet commands the appearance of all the elements included in index.php.

Although only one is visible, all the layouts are present on each page being processed. It is trough CSS that the visibility of the layouts is controlled. The same logic is applied to the presentation of links and images through the page content, they are always present, but css.php outputs a CSS that will show or hide them depending on what is set in the control panel.

HTML.PHP

Html.php loops through the source document's content using the

htmlparser.inc class. It will change mode, or state, depending on what content is being processed and decide if the content and tags are displayed in the final web page. When html.php receives the code of the requested page it starts to dissect its html. For this prototype it was decided that only a limited sets of html tags would be collected. In most cases, tags that mark text and text attribute.

All the source content that isn't meant for viewing[3.4]is skipped altogether. All the textual content is processed and displayed as is: texts, headers, lists.

Links and images are also displayed. A lot of data from those two type of tags also has to be collected for the link layout and the image layout. html.php takes care of that, as it loops through the content, by collecting all the relevant data in lists

About HTML Tags

HTML is an embedded language. The instruction and the content are fused into the same document the reader loads into its browser. The browser reads the tag and decides how to display the content. All the content of a web page is bracketed inside pairs of less-than (<) and grater-than (>) character. This is what we refer as a “tag”. The first word in a tag is its name. This is what this software use to decide if the content is worth collecting. Tags can also have special attribute that further define its actions. Attributes are listed after the tag name in the opening tag.

The Tags We Want

When html.php “collects” an html page, it collects the text and its tags. This script is looking for the textual content of the page it’s analysing. The tags it is collecting are the so called physical style tags; these include the paragraph tags, the headers tags and any tags that con-

3.4 Javascript, css styles, forms and metadata (information about the page.)

vey meanings. Other meaningful tags include bold, italic, underline. Html text can also appear in the form of a list. When a list elements tag is encountered it is also collected.

With this set of tag we can collect the textual content of most web pages. We noticed that it worked better on the most recent pages, the pages that follow a stricter XHTML model.

For older web pages, the main problems encountered are tables. Since the first years of the graphic browser, tables have been used to do page layouts instead of their proper function witch is to display tabular data.

Tables are built using a minimum of 3 different tags. First the top table tag, then the table row tag and the last children in the chain the table cell tag. Our strategy is to treat tables as layout tags. The software is set to ignore all table tag except for the table cell. We strip out the table cell tag and replace it by a paragraph tag.

Anchor and image tag are also collected but need a little more attention. For these two tags the software needs to record some of their attributes[3.5].

To keep the links active, they need to be re-interpreted in absolute form. This way, the user will be able to continue browsing using the page's hyperlinks. It also needs to encode the links with the current formatting settings so that every following page will be re-formatted according to the user settings.

Images like links also need some intelligent processing. In order to display properly, reference to images have to be rewritten in absolute form. This means using the complete internet address of the image.

3.5 The ALT attribute of an image specifies alternative text use by the browser when it's unable to display an image.

Building The Layout

As it is looping trough the tags, html.php is also building lists of information that will be needed in the end to generate the different layout.

Creating An Outline Layout

The first list is record of all headers. When it encounters a <H> tag[5], it sends a copy to the list of headings that will be used to generate the outline listing lay-out. All heading tags are copied as is so their hierarchical relationship is kept[3.6]

The script also paste in an anchor alongside each header in the main content. That way, the headings in the right column can link to their occurrences in the text. To make the scanning of the text even more efficient , a “back to top” link as been added to the body of the text to bring the user back to the listing.

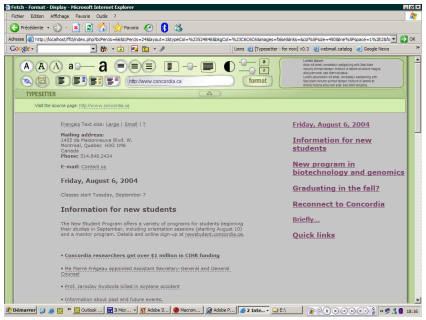
Dealing With Images

On the vast majority of web site, the images are referenced relatively from the page that calls them. The TYPESETTER is only building a new html page. The images it’s using stays on their respective server so the script need to transform the reference to an absolute link

The Filter

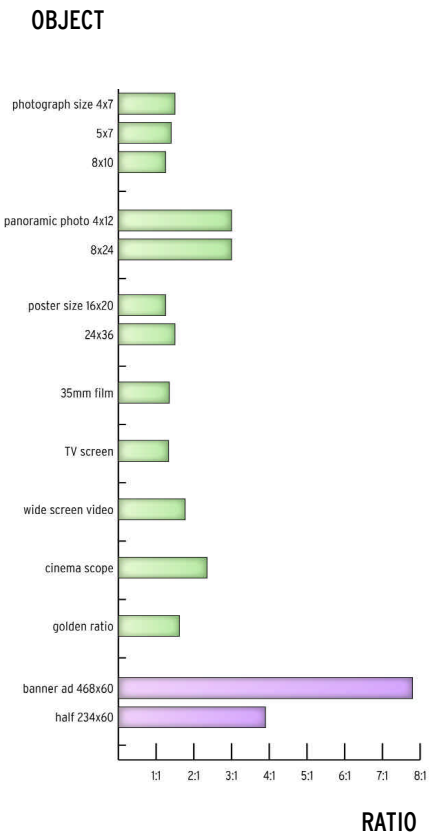
Web page can be filled with tons of graphic. Buttons, banner ad and other garniture are rarely relevant to what the page is about. For the image listing, we tried to filter out these graphics and only keep the images that are pertinent.

The easy part is to get ride of the cosmetics. Spacers, bitmap tiles and icons can be eliminated by discarding any graphic smaller than 15 pixels.



Re-formatted page with the outline layout option

3.6 Header tags are mark H1 H2 H3 etc



Ratio of various graphic display

The software can't really know what a picture is about. It's not that intelligent. Sure we can detect things like its name or the alt[3.5] attribute but we don't have the processing power or the intelligence to analyse a picture's name against the whole content of the page. One thing we can assume tough, is that a meaningful image, an image placed in to support the subject, would have a minimum size.

The strategy is to check the aspect ration of each image. Most of the visual message we experience everyday have rectangular shapes. Further more, for the vast majority of these rectangles, the length of their longest side rarely surpass the other one by more than 3. A TV screen has a ration of 1.33 to 1. The average 4x6 photograph is 1.5 to 1. But the average banner ad rates as high as 7,8:1. So, from a quick study of the visual information we find all around us, it was concluded that any image with a ratio over 3:1 would not be consider as supporting the content. This strategy is successful in about 75% of cases.

When an image tag has both its width and height attributes available, it is only put it in the image layout if it's big enough, and its proportional ratio is between the set values

Dealing With Links

As it was explained before, all the links in the pages generated by the TYPESETTER are functional. The user can use them to continue navigate the website he has re-formatted. And all the pages he will access will be re-formatted and will follow his setting.

To do this, the parser need to transform all the links to absolute links just like it does for the images. In order to enable the navigation trough re-formatted pages, the parser also needs to include in the link all the style value it needs to

reformat the next page like the one being looked at.

At runtime, the html.php also collects information to build the link layout. When it encounter a link tag <A>the parser collects the information include between the open and closing tag and puts the reprocessed address as its link.

That information may not necessarily be on a word. If a link is on an image, the parser will look for the alt[3.5] tag of that image and use it to identify the link in the listing. If no information is available, the word “image” will be use.

HTMLPARCER.INC

htmlparser.inc can load a web page and transform it into an object that can be called to loop through all its content.

The looping occurs in the order the web-page content is written, i.e. from the first element in a page, to the last element of the page.

Looping through the content, we can test what each part is. It is either an opening html tag, text content, a closing html tag, or an html comment.

TYPESETTER |

bibliography

Bibliography

BERNARD, M.L., LIDA, B., RILEY, S., HACKLER, T., & JANZEN, K. (2002). *A comparison of popular online fonts: Which size and type is best?* Usability News 4.1

BERNARD, M.L., MILLS, M.M., PETERSON, M., & STORRER, K. (2001). *A comparison of popular online fonts: Which is best and when?* Usability News 3.2

CHAPARRO, B., BAKER,R.J., SHAIKH, A.D., S. HULL, L., BRADY, (2004) *Online Text: A Comparison of Four White Space Layouts*. Usability News 4.1

ITTEN, J. *L'art de la couleur (edition abrégée)*, Dessain et Tolra, Paris, 1973, 98p.

MALO, M. (1996) *Guide de la communication écrite*, Editions Québec/Amérique, 322p.WELLER, D. (2004). *The Effects of Contrast and Density on Visual Web Search*. Usability News 3.2

MAYER, E.A., *Cascading Style Sheets, The Definitive Guide*, O'Reily Press, 2000, 453p.

MICROSOFT CORPORATION (2004) *MSDN Library* http://msdn.microsoft.com/library/

MUSCIANO, C., KENNEDY, B. *HTML & XHTML The Definitive Guide, Fourth Edition* O'Reily Press, 2000, 653p

NIELSEN, J. (1997) *How Users Read on the Web*. Jakob Nielsen's Alertbox October 1, 1997

TUFTE, E.R., *Envisioning Information*, Graphic Press, Cheshire, Connecticut, 1990, 121p.

WIKIPEDIA, "Typeface" the free encyclopedia, http://en.wikipedia.org/wiki/Typeface

The World Wide Web Consortium (W3C). "XHTML™ 1.0 The Extensible HyperText Markup Language (Second Edition)" http://www.w3.org/TR/xhtml1.

ZELDMAN, J., *Designing with web standards*, New Riders Press, Indianapolis, 2003, 436p.