Today

- More Perl
  - Perl technical stuff
- Web Programming
  - Perl based
- Homework 1 ready
- Please think about acquiring the C book (Deitel & Deitel)

- reading:
  - perl object and packages
By the way

- Ranges are only in the positive direction

- \([ 5 .. 1]\)
  - null list returned

- So how to get the above list?
Here document

print <<something;
This will print
Everything on each line
Until here
something

- Case sensitive
- Helpful when printing out lots of information
- Interpolated strings
Something Interesting:

- Can have a perl program with the following variables in the same scope:
  - $name
  - @name
  - %name

- All in the same scope
- Perl will never mix them up (that is our job)
If you were in charge, any ideas on how to do it??

- How does perl do it?
Packages

- Think of a package as an area code for your variables
- Default package is main
- Each package has an associated data structure called a **symbol table** holding a list of variables

**Example:**

```java
package FOO;
```

- Sets the current symbol table till end of block or next package declaration
- Can have multiple package declaration within the code
Symbol Table

- maps variables to information needed by compiler to handle it
- Perl maps variables names to Glob type
- Glob type matches to each variable type
- Each namespace has own symbol table
- Will come back to this later when talking about object creation (will also play with it in the labs)
In short

- $package::variable to refer to specific variable
- $::variable  # assumes main
- In old perl:
  - $main’something  # old convention

- main ST hold global variables
- In old perl:
  - _variables used to be main only
  - now can have those variables anywhere
Little more on ST

- Symbol tables simple hashes
- All symbol tables linked through main (through parent)
- %main:: has reference to itself
- %main::main::main::main is ok 😊
- Values are type globs
sub dispSymbols {
    my($hashRef) = shift;
    my(%symbols);
    my(@symbols);
    %symbols = %{$hashRef};
    @symbols = sort(keys(%symbols));
    foreach (@symbols) {
        printf("%-10.10s| %s\n", $_, $symbols{$_});
    }
}

dispSymbols(
    package Foo;
        $bar = 2;
        sub baz {
            $bar++;
        }
    );
- Let's take a break from pure Perl
- Switch gears
- Let's talk about web-based programming
WWW

- global information space
- URI identify resources available
  - simple representation
  - simple references
  - simple access
- available over the internet
- Client server model
- Document Markup Language
Content types

- **Typical**
  - Request is served from a file formatted in html
  - Static file of what we would like to render on a web client.
  - Example:
    - Class syllabus webpage
    - Reload shows same thing

- What if we could tailor each user's web experience to what they want.
  - Design of protocol to handle this
  - Dynamic content of web page content

- Different than say AJAX tech
  - Interactive content on the fly
  - Breaks web idea
  - Can’t return to specific point in browse history
CGI

- If you want to be able to program across the web
- Need to know many different platforms
- Will need an international language

- Common Gateway Interface
  - protocol to allow software to interact with information sources
How does CGI work:

1. HTTP Request
2. Call CGI
3. CGI Responds
4. HTTP Response
Perl + cgi

- Remember:
  - Perl is only a tool here
  - Don’t just memorize, understand
    - Why
    - What
    - How
  - Don’t be afraid to experiment

- STDIN
  - Contents passed to perl script

- STDOUT
  - Will need HTTP headers before printing

- STDERR
  - Depends on server, sometimes just error logs, sometimes error reports on client
Working in CGI

- There are Perl modules for this
- Very easy to use

- WE WONT USE THEM

- Reason: want you to learn what is happening underneath
  - Make life easier if you need to do anything over cgi
  - Will know how to solve problems in this space
Important

- This will come back to haunt you if you miss this

- You might be on a windos platform

- Your perl script will be running on the web server
  - Which might be running a different operating system
  - Sometimes multiple machines running webservice so starting two copies of your code might be running on two different machines
So once we have a common language to allow clients and servers to talk

Need a common place to pass data

CGI hash!
This is will be your best friend
Used in getting information \textit{from} the client
Create content is way to pass back information \textit{to} the client
Unix background

- Unix permissions
  - user
  - group
  - other

- Need to set permissions:
  - chmod 0755 ???.cgi
  - -rwxr-xr-x

- Need to place script in correct place
  - Usually cgi-bin/ directory

- Naming
  - Usually need to end in something.pl.cgi
reminder

- When working with hash
  - `%hash`
    - Deals with entire hash at once
    - keys `%hash`
  - `$hash{somekey}`
    - Allows you to access individual elements in the hash
Sample test4.pl.cgi

#!/usr/local/bin/perl

use strict;

my $time = localtime;
my $remote_id = $ENV{REMOTE_ADDR};

print "Content-type: text/html\n\n";

print <<END_OF_PRINTING;
This is the time : $time
<P>
and your ip is $remote_id

END_OF_PRINTING
This is the time: Tue Jul 19 16:45:17 2005

and your name is 24.188.170.41
Some CGI Environmental Variables

- **CONTENT_LENGTH**
  - Length of data passed to cgi
- **CONTENT_TYPE**
- **QUERY_STRING**
- **REMOTE_ADDR**
  - Ip address of client
- **REQUEST_METHOD**
- **SCRIPT_NAME**
- **SERVER_PORT**
- **SERVER_NAME**
- **SERVER_SOFTWARE**
- **HTTP_FROM**
- **HTTP_USER_AGENT**
- **HTTP_REFERER**
- **HTTP_ACCEPT**
Problem

- How can we print out all the environment variables in CGI?
Example

#!/usr/local/bin/perl

use strict;

my $vars
print "Content-type: text/html\n\n"

foreach $vars (sort keys %ENV){
    print "<P><B>$vars</B><BR>");
    print $ENV{$_};
}

PATH
/usr/local/bin:/usr/local/sbin:/usr/bin:/bin

QUERY_STRING

REMOTE_ADDR
24.188.170.41

REMOTE_PORT
1674

REQUEST_METHOD
GET

REQUEST_URI
/~sh553/cgi-bin/test4.cgi

SCRIPT_FILENAME
/home/sh553/html/cgi-bin/test4.cgi

SCRIPT_NAME
/~sh553/cgi-bin/test4.cgi

SERVER_ADDR
128.59.16.101

SERVER_ADMIN
webmaster@cs.columbia.edu

SERVER_NAME
www1.cs.columbia.edu

SERVER_PORT
80

SERVER_PROTOCOL
HTTP/1.1

SERVER_SOFTWARE
Apache/1.3.33 (Unix) mod_ssl/2.8.22 OpenSSL/0.9.7e PHP/4.3.11
html

- Since clients we are dealing with here are going to be html clients

- Would like to format the output to make it easier to display

- Would like to print out things in html

- Anyone worked with html already ??
HTML

- Hyper Text Markup Language
- Standard by w3:  
  http://www.w3.org/MarkUp/
- Way of standardizing format of documents so that users can share information between different systems seamlessly
- Evolving to XHTML format
HTML

- Hypertext Transfer Protocol
- Language used between web servers and web clients
- http url’s

http://www.google.com:80/search?q=what
Google.com

http://www.google.com/search?q=shlomo
Very basics

- Html consists of matching tags
- `<something>` = opening tag
- `</something>` = close tags

- HTML DOC:
  - `<html> <body> ....... </body> </html>`
Web pages

- `<title> .... </title>` (before the body section)
- `<H1> .... </H1>` (header titles h1, h2, h3)
- `<P>` paragraphs
- `<BR>` line breaks
- `<b> ... </b>` bold
- `<i> ... </i>` italicize
- `<u> ... </u>` underline
More basics

- `<img src =“…..” width=“X” height=“Y”>`
- `<a href=“www.cnn.com”> something </a>`
- `<a name=“Anchor1”>`
  - Can be referred to by page.html#Anchor1
- `<hr> line`
- `<hr width=50%> half line`
Lists

- Unordered list
  `<ul> <li> </li> ...... </ul>`

- Ordered list
  `<ol> <li> </li> ..... </ol>`

- Nested lists
  - Lists themselves can be nested within another
# Tables

- `<table>
  <tr>
  <td>Hello</td>  
  <td>World</td>
  </tr>
</table>`
comments

<!--

anything you do

-->
More html

- Browsers allow you to see source code of html document
- Can get wysiwyg editors
- Word will allow you to save as html
  - Very complicated output
- This is not an html course so we will be just doing very basics
Browser Issues

- Although HTML should be universal, there are occasional differences between how Microsoft IE renders a webpage and Mozilla firefox.
  - Getting better with each new version
  - Should just be aware, at least test any real webpage against popular browsers
    - Also mac browsers 😊
So what?

- So easy to get a perl script to print out html and show up on browser

- Just need to include in url
    - Will be in the html/ directory (need to create if not there)
    - Runs on a sun os machine by the way

- So how do you interact with the users?
Interacting

- Forms allow you to display information for the user to enter
  - Login
  - Shipping info
  - etc

- GET
  - HTTP request directly to the cgi script by appending the URL
  - Value=key separated by &
  - Space replaced by +
  - URL conversion characters

- POST
  - HTTP request in content of message, i.e it is stdin to your script
Input Tag

- Each field in the form is in an input tag
- Type
  - Text
  - Radio button
  - Checkbox
  - Pull down menus
  - etc
- Name
  - Symbolic name (so can recognize it)
- Value
  - Default value, or what the user will end up typing
Note: Encoding

- Spaces are turned to +
- & separates field
- Special characters are turned into %?? (hex)
  - “(“ is %28

- So “class is great” = “class+is+great”
others

- Submit buttons
  - `<input type="submit">`

- Reset buttons
  - `<input type="reset">`

- Value will change the default name on the button

- try not to trick user with labels....
Decoding Form Input

1. $ENV{QUERY_STRING}$
2. If( $ENV{REQUEST_METHOD} eq POST) {
   read $ENV{CONTENT_LENGTH}"
}
3. Split pairs around &
4. Split keys and values
5. Decode URL
6. Remember key,values
Task...how would you?

1. Create a webpage counter (saying you are visitor x to this page)

2. Now create a graphical counter
MD5 Sum

- MD5 – uses a 128 bit hash value
- Designed in 1991
- Known problems with collision attacks
Bottom line

- Still in very wide use
- Allows authentication of files given a file and signature
- Visually authentication against tampering
- What obvious weakness??
Md5 of a file

- Can execute md5sum within perl
- Can use perl defined methods
  - Write yourself
  - Find someone else’s 😊

- perl libraries....will cover in labs