IT452 Advanced Web and Internet Systems

Set 4: Perl and Database Connections

Assumptions

• You know Perl
  – We’ll review
  – Can use PHP instead

• (maybe??) You know how to use SQL
Perl Basics

use CGI qw( :standard );
print( header() );

$x = 2 + 3;
$y = $x * 4;

if ($x == 5.0) {
    print ("x is five");
}

for ($i = 0; $i < 3; $i++) {
    $squared = $i * $i;
    print ("<br> $i = $i, squared is $squared");
}

$pet1 = "dog";
$pet2 = "llama";

# Single quotes vs. double quotes
print ('<br/>I have a $pet1 and a $pet2.');
print ('<br/>I have a $pet1 and a $pet2.');

$comp1 = ($pet1 eq "dog");
print ("<br/> comp1: $comp1");

Perl Stuff

"Scalar" variables:
 $x = 3;
 $y = "Hello";

"Array" variables:
 @list = (3, 7, "dog", "cat");
 @list2 = @list1;    # copies whole array!

A single element of an array is a “scalar:
    print "Second item is: @list[1]";   # Don’t use @

Get array length by treating whole array as scalar:
    $lengthOfList2 = @list2;

File operations
    open ( MYFILE, "input.txt" );
    open ( MYFILE, "output.txt" );
    open ( MYFILE, "LOG.txt" );
Perl Function Calls ("subroutines")

```perl
use CGI qw(:standard);
print( header() );

# Prints "hello", takes no arguments
sub hello {
    print "\n<br/> Hello.";
}

# Takes two arguments, return their product
sub multiply {
    my($valA, $valB) = @_; 
    return $valA * $valB;
}

my($x) = 2;
&hello;
print "\n<br/> $x * 7 = " . &multiply($x,7);
&hello();
&hello(72145);
print(end_html());
```

Function Calls and Arrays

```perl
# Takes an array as argument, returns minimum value
sub findMin {
    my(@array) = @_; 
    my $min = $array[0];
    my $ii;
    my $len = @array;
    for ($ii=0; $ii < $len; $ii++) {
        if ($array[$ii] < $min) {
            $min = $array[$ii];
        }
    }
    return $min;
}

# Defines new global array, @array1 
# AND returns a new array with 4 elements.
sub makeArray() {
    @array1 = (89, 23, 90);
    my @array2 = (34, 5.4, 123, 2.01);
    return @array2;
}

@test1 = makeArray();
@test2 = (89, 23, 40, -17);
print "\nMin1 is: " . &findMin(@test1);
print "\nMin2 is: " . &findMin(@test2);
print "\nMin3 is: " . &findMin(@array1);
print "\nMin4 is: " . &findMin(@array2);
```
Example – Get from DB, output ALL to HTML

```perl
#!/usr/local/bin/perl
use strict;                            # Include this!
use CGI::Carp qw( fatalsToBrowser );   # And this!
use CGI qw(:standard );
use DBI;
use DBD::mysql;

my $dtd = "--W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1-transitional.dtd";
print( "Content-type: text/html; charset=UTF-8\n"
);
print(start_html( { dtd => $dtd, title => "Read Test", style=>{'src'=>'styles.css'}} ));

my $databaseHandle = DBI->connect( "STUFF" );
my $query = "SELECT * FROM comments";
my $statementHandle = $databaseHandle->prepare($query);
$statementHandle->execute;

# put results in a table
print "<table border='1'> <thead >
";
print "<th> ID </th> <th> User </th> <th> Comment </th> </thead> <tbody >
";
while (my @row = $statementHandle->fetchrow_array) {
";
}
print "</tbody> </table> <br />
";
$databaseHandle->disconnect();
$statementHandle->finish();
print(end_html());
```

Example – Get from DB, output SOME to HTML

```perl
#!/usr/local/bin/perl
use strict;                            # Include this!
use CGI::Carp qw( fatalsToBrowser );   # And this!
use CGI qw(:standard );
use DBI;
use DBD::mysql;

my $c_id = param("comment_id");
my $dtd = "--W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1-transitional.dtd";
print( "Content-type: text/html; charset=UTF-8\n"
);
print(start_html( { dtd => $dtd, title => "Read Test", style=>{'src'=>'styles.css'}} ));

my $databaseHandle = DBI->connect( "STUFF" );
my $query = "SELECT * FROM comments WHERE id=$c_id";
my $statementHandle = $databaseHandle->prepare($query);
$statementHandle->execute;

# put results in a table
print "<table border='1'> <thead >
";
print "<th> ID </th> <th> User </th> <th> Comment </th> </thead> <tbody >
";
while (my @row = $statementHandle->fetchrow_array) {
";
}
print "</tbody> </table> <br />
";
$databaseHandle->disconnect();
$statementHandle->finish();
print(end_html());
```
Example – Get from DB, output MATCHING to TEXT

#!/usr/local/bin/perl
use strict;              # Include this!
use CGI::Carp qw{ fatalsToBrowser }; # And this!
use CGI qw( :standard );
use DBI;
use DBD::mysql;

print( "Content-type: text/plain; charset=UTF-8\n\n" );
my $search = param("q");
my $databaseHandle = DBI->connect( "STUFF" );
my $query = "SELECT * FROM comments WHERE user LIKE '%$search%'";
my $statementHandle = $databaseHandle->prepare($query);
$statementHandle->execute;
# Output plain results. Separate lines with vertical bar
while (my @row = $statementHandle->fetchrow_array) {
    print "$row[0],$row[1],$row[2]|\n";
}
$databaseHandle->disconnect();
$statementHandle->finish();

Example – Get from DB, output ALL to XML

#!/usr/local/bin/perl
use strict;              # Include this!
use CGI::Carp qw{ fatalsToBrowser }; # And this!
use CGI qw( :standard );
use DBI;
use DBD::mysql;

print( "Content-type: text/xml; charset=UTF-8\n\n" );
my $databaseHandle = DBI->connect( "STUFF" );
my $query = "SELECT * FROM comments";
my $statementHandle = $databaseHandle->prepare($query);
$statementHandle->execute;
# Output plain results. Separate lines with vertical bar
print "<results>\n";
while (my @row = $statementHandle->fetchrow_array) {
    print "<result>\n";
    print "<id>$row[0]<id>\n";
    print "<user>$row[1]<user>\n";
    print "<comment>$row[2]<comment>\n";
    print "</result>\n";
}
print"</results>\n";
$databaseHandle->disconnect();
$statementHandle->finish();
Example – Simple INSERT

```
#!/usr/local/bin/perl
use strict;                            # Include this!
use CGI::Carp qw( fatsToBrowser );     # And this!
use CGI qw(:standard );
use DBI;
use DBD::mysql;
my $user    = param("user");
my $comment = param("comment");
my $dtd = "--W3C/STDF XHTML 1.0 Transitional/DTD/""http://www.w3.org/TR/xhtml1-transitional.dtd"; print( "Content-type: text/html; charset=UTF-8\n\n" ); print(start_html( { dtd => $dtd, title => "Read Test", style=>{"src":"styles.css"} } ));
my $databaseHandle = DBI->connect( "STUFF" );

# Do the SQL insert
my $query = "INSERT INTO comments (user, blurb) VALUES ('$user', '$comment')";
my $statementHandle = $databaseHandle->prepare($query);
$statementHandle->execute;
print( "<h2> SUCCESS -- inserted into the DB! </h2>" );

# If the SQL fails, we won't necessarily know. Check here.
# Any errors? Print them here
print("Errors, if any: $DBI::errstr\n\n" );

# Close up
$databaseHandle->disconnect();
$statementHandle->finish();
print(end_html());
```

Perl “strict” mode

- This forces variables to be declared, and a few other things:
  my $ii = 0;
  my @someArray = (1,2,3);
- Required for all IT452 Perl scripts
- Will save you pain!
- Also use “Carp” mode shown in examples
Example from before – what needs to change?

```javascript
// Make synchronous call to server to get data for a new row
function handleQuery() {
  xhr = window.ActiveXObject
      ? new ActiveXObject("Microsoft.XMLHTTP")
      : new XMLHttpRequest();

  // Get data from server
  xhr.open("GET", "dummy_data1.csv", false);
  xhr.send(null);  // GET, so no "data" part

  // Deal with results
  if (xhr.status != 200) {
    alert("Error contacting server! Status: "+xhr.status);
  } else {
    // Get comma-separated data and make into an array
    var data  = xhr.responseText;
    var elems = data.split(",");

    // Make new row with this data
    insertRow(elems);
  }
  return false;  // false prevents the form from actually submitting
}
```

How does AJAX really work?  OR DB work?

- Client / Server