

```
1    #!/usr/bin/perl
2    #
3
4    #
5    #A very basic tic-tac-toe program (the computer chooses randomly
)
6    #
7
8
9    use strict;
10   use CGI;
11   use Socket;
12
13   my ($rounds, $round_temp, $squares, $page, $x, $y, $z, $cell, $p
layer_move, @available_choices, $computer_move, @choices, $round, $winner, $
player_move_pretty, $computer_move_pretty);
14   my ($round_minus_one); #bug fix (was recording moves for round1
as round2
15
16   $page = CGI->new();
17
18   print $page->header;
19   print $page->start_html();
20
21   # print table beginnings
22
23   print("<table width=\"90%\" border=0 cellpadding=15>\n");
24   print("<tr valign=middle>\n");
25
26   # left cell is tic tac toe table
27
28   print("<td align=center>\n");
29
30   #
31   # find out which round it is so we know how to define $squares
32   #
33
34   unless ($page->param('round')) {
35       $round = 0;
36   }else {
37       $round = $page->param('round');
38   }
39
40   #
41   # set array of tic tac toe squares
42   #
43
44   if ($round > 0) {
45       $squares = [
46           [
47               $page->param('[0][0]'),
48               $page->param('[0][1]'),
49               $page->param('[0][2]')
50           ],
51           [
52               $page->param('[1][0]'),
53               $page->param('[1][1]'),
54               $page->param('[1][2]')
55           ],
56           [
57               $page->param('[2][0]'),
58               $page->param('[2][1]'),
59               $page->param('[2][2]')
60           ]
61       ];
62   }else {
```

```
63     $squares = [
64         ['?', '?', '?'],
65         ['?', '?', '?'],
66         ['?', '?', '?']
67     ];
68 }
69
70 #
71 # set array for determining history of moves (recorded by round)
72 #
73
74 if ($round > 0) {
75     $rounds = {
76         round1 => {
77             player => $page->param('round1_x'),
78             computer => $page->param('round1_o')
79         },
80         round2 => {
81             player => $page->param('round2_x'),
82             computer => $page->param('round2_o')
83         },
84         round3 => {
85             player => $page->param('round3_x'),
86             computer => $page->param('round3_o')
87         },
88         round4 => {
89             player => $page->param('round4_x'),
90             computer => $page->param('round4_o')
91         },
92         round5 => {
93             player => $page->param('round5_x'),
94             computer => $page->param('round5_o')
95         }
96     };
97 }else {
98     $rounds = {
99         round1 => {
100             player => '?',
101             computer => '?'
102         },
103         round2 => {
104             player => '?',
105             computer => '?'
106         },
107         round3 => {
108             player => '?',
109             computer => '?'
110         },
111         round4 => {
112             player => '?',
113             computer => '?'
114         },
115         round5 => {
116             player => '?',
117             computer => '?'
118         }
119     };
120 }
121
122 #
123 # increment $round to give it a new hidden value
124 #
125
126 $round = $round + 1;
127 $round_minus_one = $round - 1;
128 print ("Round is: $round<br>\n");
```

```

129
130     ##
131     ## get player move using subroutine (subroutine stores it to $sq
uares array)
132     ##
133
134     $player_move = $page->param('choice');
135     if ($player_move) {
136         $round_temp = "round" . $round_minus_one;
137         player_moves($player_move, $page, $squares);
138         $player_move_pretty = make_move_pretty($player_move);
139         $rounds->{$round_temp}->{'player'} = $player_move_pretty;
140
141         #
142         # evaluate for winner after player moves
143         #
144
145         $winner = evaluate_board($squares);
146         if ($winner eq "x") {
147             print ("<font color='blue'>Player Won!</font><p>\n");
148             print_final_table($squares, $page, $winner, $round, $round
s);
149             }else {
150
151             #
152             # check to see if player won. if player won, don't do the rest o
f this
153
154             #
155             # get available choices for computer choices
156             #
157
158             @available_choices = get_available_choices($squares);
159
160             #
161             # get computer move
162             #
163
164             $computer_move = get_computer_choice(@available_choices);
165             ($x, $y) = split(/:/, $computer_move);
166             #get coordinates for computer move and store in $x and
$y
167             $squares->[$x][$y] = "o";    ## change square to "o"
168             $round_temp = "round" . $round_minus_one;
169             $computer_move_pretty = make_move_pretty($computer_move);
170             $rounds->{$round_temp}->{'computer'} = $computer_move_pret
ty;
171
172             } #matches else {
173
174             } #matches if ($player_move)
175
176             #
177             # now that we have the array with computer and player choices, s
ee if there is a winner
178             #
179             $winner = evaluate_board($squares);
180             if ($winner eq "o") { #we already checked for x before
181                 print ("<font color = 'blue'>Computer Won!</font><p>\n");
182                 print_final_table($squares, $page, $winner, $round, $rounds);
183             }
184
185             if (($winner ne "o") and ($winner ne "x")) {
186                 print ("<font color='blue'>No winner yet</font><p>\n");
187
188

```

```

189     # start form
190
191     print $page->startform(-method=> 'POST');
192
193     #print hidden values in form
194
195     print_hidden_values($page, $squares, $rounds);
196
197     # print hidden value for $round
198
199     print "<input type='hidden' name = 'round' value='$round'>\n"
;
200
201     # print hidden values for saving rounds
202
203
204     # start tic tac toe table
205
206     print ("<table border=1 cellpadding=10>\n<tr>");
207
208     #
209     # look through array elements ($squares) and find x's or o's
210     # for all squares with no value, print a checkbox with the coord
inates
211     # in the form of [0][0] as its name
212     #
213
214     foreach $x(0..2) {
215         if ($squares->[0][$x] eq "?") {
216             print_cell($squares, $page, $x, "0");
217         }else {
218             print ("<td>" . $squares->[0][$x] . "</td>\n");
219         }
220     }
221
222     print ("</tr><tr>\n");
223     foreach $x(0..2) {
224         if ($squares->[1][$x] eq "?") {
225             print_cell($squares, $page, $x, "1");
226         }else {
227             print ("<td>" . $squares->[1][$x] . "</td>\n");
228         }
229     }
230
231     print ("</tr><tr>\n");
232     foreach $x(0..2) {
233         if ($squares->[2][$x] eq "?") {
234             print_cell($squares, $page, $x, "2");
235         }else {
236             print ("<td>" . $squares->[2][$x] . "</td>\n");
237         }
238     }
239
240     print ("</tr></table>");
241
242     #
243     # print warning about picking multiple squares
244     #
245
246     print $page->submit();
247
248     print ("<p><font color='red'>Note: if you pick more than one
square, your choice will be the upper and leftmost square that you choose!!<
/font><p>\n");
249
250

```

```

251     print $page->endform();
252     }
253
254     # end table cell
255
256     print ("</td>\n");
257     print ("<td align=middle>\n");
258
259     # get printable versions of moves and print choices
260
261     if (($player_move) or ($computer_move)) {
262         foreach $x(1..$round_minus_one) {
263             $round_temp = "round" . $x;
264             print ("<p><b>Round $x:</b><br>\n");
265             print ("\tplayer: " . $rounds->{$round_temp}->{'player'} .
" <br>\n");
266             print ("\tcomputer: " . $rounds->{$round_temp}->{'computer
'} . " <br>\n");
267         }
268     }else {
269         print ("No moves yet.\n");
270     }
271
272     # end table
273
274     print ("</td></tr></table>\n");
275
276     print $page->end_html();
277
278
279     sub player_moves {
280         my $move = $_[0]; ##get move
281         my $page = $_[1]; ## import page object
282         my $squares = $_[2]; ## import array so we can change square
283         my ($x, $y, $z);
284
285         foreach $x(0..2) {
286             foreach $y(0..2) { ##test for each square
287                 $z = "$x[$y]";
288                 if ($move eq $z) {
289                     $squares->[$x][$y] = "x"; ## define array element
for choice
290                 }
291             }
292         }
293     }
294
295     sub print_cell {
296         my $squares = $_[0]; ##import semi-existant array of squares
297         my $page = $_[1]; ##import html stuffs
298         my $x = $_[2]; ##import the number that the foreach is
on
299         my $row = $_[3]; ##import the row we are on
300         my $cell = "$row[$x]"; ##get square coordinates for use in
naming checkbx
301
302         print ("<td>");
303         print ("<input type='checkbox' name='choice' value='$cell'>\n
");
304         print ("</td>");
305     }
306
307     sub get_available_choices {
308         my $squares = $_[0];
309         my ($x, $y, $z);
310         my @available_choices = ();

```

```

311
312     foreach $x(0..2) {
313         foreach $y(0..2) {
314             unless (($squares->[$x][$y] eq "x") or ($squares->[$x][
$y] eq "o")) {
315                 $z = "$x:$y";    ## return in this form for later u
se (so we can split it by the colon)
316                 push (@available_choices, $z);
317             }
318         }
319     }
320     return @available_choices;
321 }
322
323 sub get_computer_choice {
324     my @available_choices = @_;
325     my $length = @available_choices;
326     my $number = int(rand() * ($length - 1));
327     my $choice = $available_choices[$number];
328     return $choice; ## this will be a coordinate, in the form of
$x:$y from $z above
329 }
330
331
332 sub print_hidden_values {
333     my $page = $_[0];
334     my $squares = $_[1];
335     my $rounds = $_[2];
336     my ($x, $y, $cell, $round);
337
338     #print hidden values for cells
339     foreach $x(0..2) {
340         foreach $y(0..2) {
341             $cell = "$x[$y]";
342             print("<input type='hidden' name='$cell' value='" .
$squares->[$x][$y] . "'>");
343             print("\n");
344         }
345     }
346
347     #print hidden values for rounds (history)
348     foreach $x(1..5) {
349         $round = "round" . $x;
350         print("<input type='hidden' name='$round" . "_x' value ='"
. $rounds->{$round}->{'player'} . "'>\n");
351         print("<input type='hidden' name='$round" . "_o' value='"
. $rounds->{$round}->{'computer'} . "'>\n");
352     }
353 }
354
355
356 sub evaluate_board {
357     my ($squares) = $_[0];
358     my ($x, $y, $winner);
359
360     foreach $x (0..2) {
361         if (
362             ($squares->[$x][0] eq $squares->[$x][1]) and
363             ($squares->[$x][1] eq $squares->[$x][2])
364         ) {
365             $winner = $squares->[$x][0];
366             return $winner;
367         }
368     }
369     foreach $y (0..2) {
370         if (

```

```

371             ($squares->[0][$y] eq $squares->[1][$y]) and
372             ($squares->[1][$y] eq $squares->[2][$y])
373             ) {
374                 $winner = $squares->[0][$y];
375                 return $winner;
376             }
377         }
378     if (
379         (($squares->[1][1] eq "x") or ($squares->[1][1] eq "o"))

380         and
381         (
382             (($squares->[0][0] eq $squares->[1][1]) and
383             ($squares->[1][1] eq $squares->[2][2]))
384             or
385             (($squares->[0][2] eq $squares->[1][1]) and
386             ($squares->[1][1] eq $squares->[2][0]))
387         )
388         ) {
389             $winner = $squares->[1][1];
390             return $winner;
391         }
392     }
393
394     sub print_final_table {
395         my $squares = $_[0];
396         my $page = $_[1];
397         my $winner = $_[2];
398         my $round = $_[3];
399         my $rounds = $_[4];
400         my ($visitor, $visitor_name, $time);
401
402         ## print ending table
403
404         print $page->startform(action=>'tic_tac.cgi',
405                               method=>'POST'
406                               );
407
408         print_hidden_values($page,$squares,$rounds);
409
410         print ("<input type='hidden' name='round', value='$round'>\n"
411 );
412         print ("<table border=1 cellpadding=10>\n");
413         print ("<tr valign=middle>\n");
414         foreach $x(0..2) {
415             print ("<td align=center>" . $squares->[0][$x] . "</td>\n"
416 );
417         }
418         print ("</tr><tr valign=middle>\n");
419         foreach $x(0..2) {
420             print ("<td align=center>" . $squares->[1][$x] . "</td>\n"
421 );
422         }
423         print ("</tr><tr valign=middle>\n");
424         foreach $x(0..2) {
425             print ("<td align=center>" . $squares->[2][$x] . "</td>\n"
426 );
427         }
428         print ("</tr></table><p>\n");
429         print ("<a href=\"http://soya.serve.com/cgi-bin/tic_tac.cgi\"
430 >Play Again!</a>");
431     }

```

```
431     #print log of play
432
433     $visitor = $page->remote_host();
434     if ($visitor =~ /\d*\.\d*\.\d*\.\d*/) {
435         $visitor_name = gethostbyaddr(inet_aton($visitor), AF_INET);
436     }
437
438     $time = localtime(time());
439
440     open (MAIL, "| /usr/sbin/sendmail -t");
441     print MAIL "To: author@example.com\n";
442     print MAIL "Subject: tic tac toe results\n";
443     print MAIL "\n$visitor, $visitor_name: $time: $winner on roun
d $round";
444     close MAIL;
445 }
446
447 sub make_move_pretty {
448     my ($move) = $_[0]; #get move (either $player_move or $compu
ter_move
449     my (%squares_names, $pretty_move);
450
451     if ($move =~ /:/) {
452         $move =~ s/^\(\d\):(\d)/$1$2/;
453     }
454     else {
455         $move =~ s/^\[(\d)\]\[(\d)\]/$1$2/;
456     }
457
458     %squares_names = ("00" => "top left",
459                     "01" => "top center",
460                     "02" => "top right",
461                     "10" => "center left",
462                     "11" => "center",
463                     "12" => "center right",
464                     "20" => "lower left",
465                     "21" => "lower center",
466                     "22" => "lower right"
467                 );
468
469     $pretty_move = $squares_names{$move};
470     return $pretty_move;
471 }
472
473
```