Chapter 5: Surveys and Experiments Using Forms

In the methods of preceding chapters, for you to get information from the person reading your Web pages, you had to wait for the reader to send you a letter, either by postal mail or by email. Chapter 3 described how to use a link to facilitate response by email. However, not everyone who reads you Web pages has email (e.g., a person in a public library); in addition, email is not anonymous.

In this chapter, the real action begins. You will learn the technique of *forms*, which allows the viewer to respond directly on the page and send data by clicking a button. Even a person without email can participate if he or she is using a computer connected to the Internet. The method of forms also allows people to remain anonymous. This chapter will illustrate these techniques with two studies, one on probability reasoning and the other on the St. Petersburg paradox.

A. Example Illustrating FORM and INPUT

The next example illustrates `FORM` and `INPUT` tags with a study of reasoning about probability. Consider the HTML in Figure 5.1, which you can load from the CD as *Ch5_exp1.htm*.

Insert Figure 5.1 about here.

Figure 5.1. Listing of *Ch5_exp1.htm*.

```
<HTML><HEAD><TITLE>A Problem in Probability</TITLE></HEAD>
<BODY>
<! Change the email address in the line below to your address >
<FORM METHOD="post" ACTION="mailto:user@address.domain" ENCTYPE="text/plain">
<H2>Find the Probability</H2><FONT SIZE=4>
<P>Joe tossed two fair pennies and peeked at them. He said,
<P><FONT COLOR=blue>"At least one is HEADS
 but I won't tell you which." <BR>
What is the probability that the other coin is also HEADS?
<P>Type your probability, expressed as a percentage,
in the box below:<BR>
<INPUT TYPE="text" NAME="00answer" SIZE="2" MAXLENGTH="3">%
<FONT color="black">
<P>Please check your answer. When you are done,
push the button below.
<P><INPUT TYPE="submit" VALUE="I'm done">
<H2>Thank You!</H2>
</FONT></FONT>
</FORM>
<A HREF="answer.htm">To see the answer, click here</A>
</BODY></HTML>
```

There are two new tags used in Figure 5.1 (*Ch5_exp1.htm*), `<FORM></FORM>` and `<INPUT>`. These two tags work together to facilitate two-way communication between you and the reader of the page. The `<FORM></FORM>` tags indicate that the material between will contain information that can be displayed in the form or entered by the reader of the page. The `INPUT` tags allow you to create different instruments for communication. There are two variations of `INPUT` tags in this example, making three lines to examine more closely.

First, consider the `<FORM>` tag in the example:

```
<FORM METHOD="post" ACTION="mailto:user@address.domain" ENCTYPE="text/plain">
```

The `METHOD` should be set to `POST` for the examples in this book. The `ACTION` in this case indicates that when the reader of the form clicks the submit button, then the answers will be sent by email to the address given. Change the email address to your own, save it, and try out the page in your browser. If your browser is configured to send email, then you will get an email message with the value that you typed into the form for the `INPUT`. The attribute, `ENCTYPE="text/plain"`, is used here to make the email easier to read.

Second, look at the `<INPUT TYPE="text">` tag, which creates a text box for the response:

```
<INPUT TYPE="text" NAME="00answer" SIZE="2" MAXLENGTH="3">%
```

The above tag creates a text box that is 2 characters in width, and it will accept 3 digits, which allows a person to type in "100". Note that there is a % sign after the box, put there to remind the user that you want a percentage (not a decimal), and also to suggest to the user that he or she need not type in the % sign. If some people enter "%" symbols in the response box, and others do not, you will need to edit them out of your data before

data analysis. (For that purpose, you might use the search and replace feature of NotePad to replace all % signs with a null string, i.e., nothing.)

The NAME is the name given to the variable, and this information will be sent along with the value to the destination in the ACTION of the form. The message sent, either from Netscape Navigator or Internet Explorer is the message, "00answer=33." If you entered 50, you would get the message "00answer=50."

You may wonder why the variable name begins with two leading zeros. The scripts used in this book create data files in which the data will be put in order according to these leading numbers in the variable names. Therefore, for this book you should get used to numbering variables in the order that you want them to appear in the data file, starting with 00, 01, 02, etc. The order in which the variables appear in the HTML file has no effect in this scheme.

Third, examine the INPUT tag that creates a submit button:

```
<INPUT TYPE="submit" VALUE="I'm done">
```

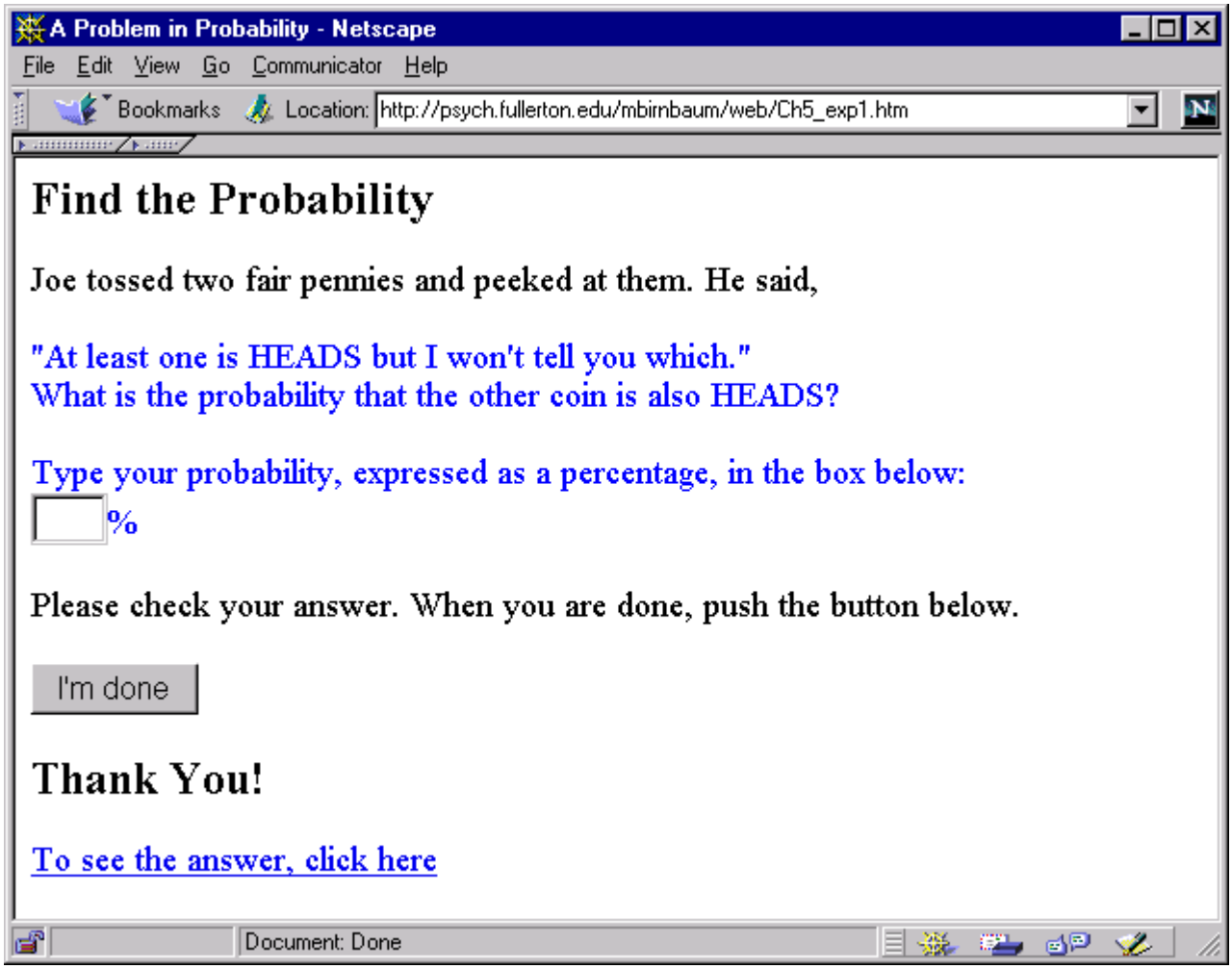
When the user clicks on a form's submit button, the variables defined in the form are sent. The TYPE="submit" designates this button as the button that sends the data, the VALUE is simply what is printed on the button; the VALUE has no effect on the action of the button. A similar button is the "reset" button, which will erase everything from the form when it is pushed. It has the following tag:

```
<INPUT TYPE="reset" VALUE="erase the form">
```

There are other ways to request input besides <INPUT TYPE="*text*">, and these will be covered later in this chapter. The appearance of the page in Netscape is shown in Figure 5.2.

Insert Figure 5.2 about here.

Figure 5.2. Appearance of the text box and submit button in Netscape for example, *Ch5_exp1.htm*.



When the person using this page clicks on the link at the bottom of the page, the link brings up a page that presents the correct solution, listed in Figure 5.3. There is nothing really new in Figure 5.3 besides the correct answer, which is that the correct probability is 1/3, or 33%. This example is sometimes considered a "trick" question in classes on reasoning and problem solving or statistics. The trick, if there is one, is a weakness among those who are unschooled in probability to jump to conclusions without working out the solution by counting the equally likely possibilities. As noted in Figure 5.3, 50% is the right answer to the wrong question.

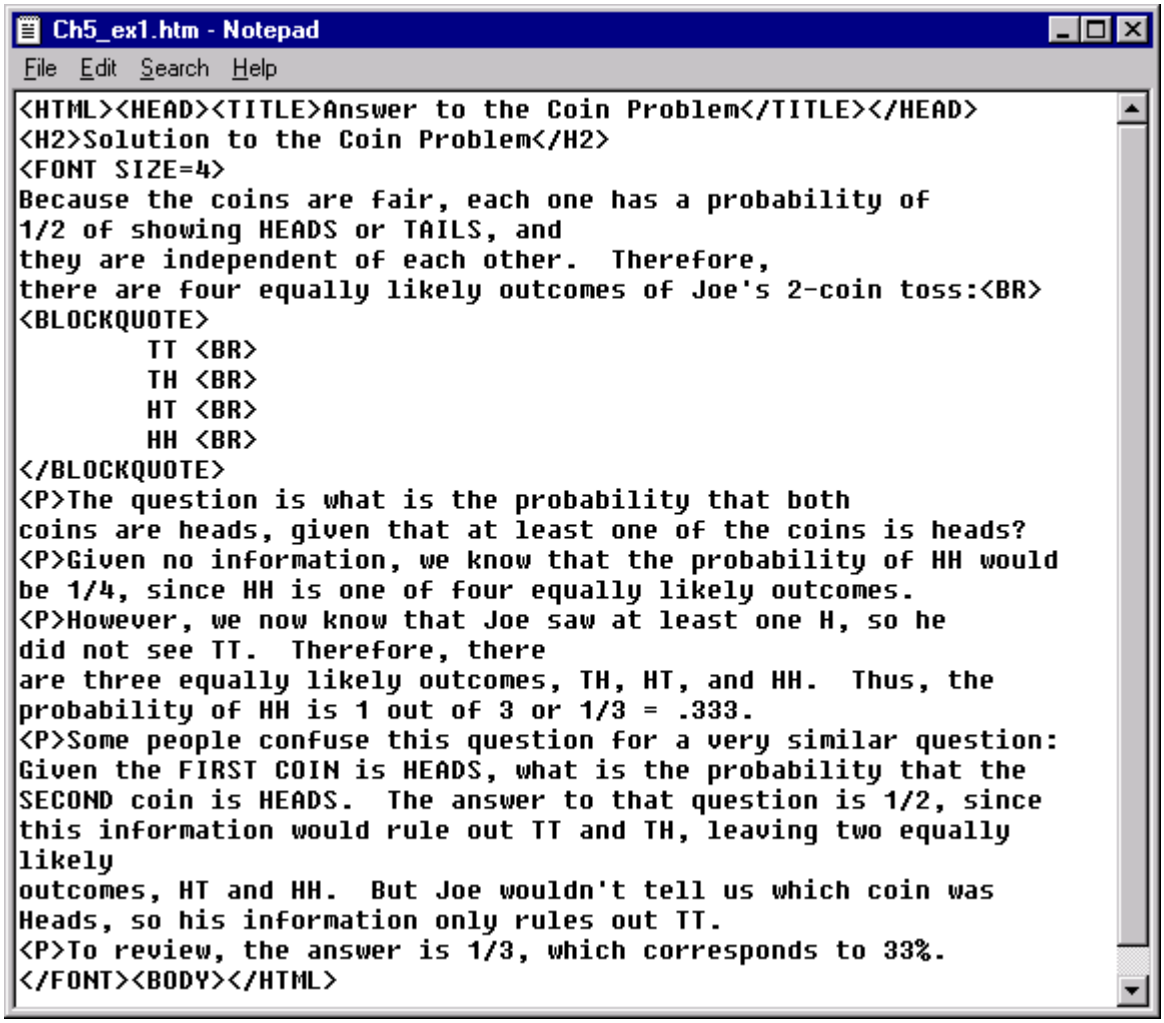
Insert Figure 5.3 about here.

This "test" has a number of problems, however. Note that our student could have clicked the link to the answer, then used the BACK button to return to the page, and then entered the correct answer. This procedure might be fine for a computerized tutorial, but it would not be very good procedure for a test, since it allows people to "cheat." It would also not be good procedure for an experiment that was designed to find out the proportion in the population who can figure out the correct answer on their own.

Also, with any real experiment, with hundreds or even thousands of participants, it would be quite a nuisance to get the data in the form of so many separate email messages.

The solution to these problems is the CGI (Common Gateway Interface) script, which allows the server to process and save the data, which saves you the worry of dealing with all those emails.

Figure 5.3. HTML for a page of debriefing, which gives the correct answer.



B. Using a Script to Process Data

Change the FORM tag in *Ch5_exp1.htm* as follows:

```
<FORM METHOD="POST" ACTION="http://psych.fullerton.edu/cgi-win/polyform.exe/generic">
```

Now when the user clicks the submit button, the data are sent to a script that resides at the above address. This script processes the data, puts them in order of the numbers that precede the variable names, and puts them in a file called *data.csv*. The script then sends the user to a URL that delivers a generic "Thank you for serving in a student's experiment" message. You can test if you have correctly entered the script address by pushing the *Submit* button and seeing if you receive the *Thank you* message. This new variation of the FORM tag is used in *Ch5_exp2.htm*. The `ENCTYPE` attribute is not used with the scripts in this book.

The script residing at the address above sends data to a file, *data.csv*, on the psych.fullerton.edu server, from which you can retrieve the file by FTP, as described in Chapter 2. Therefore, you can practice running experiments with this script to put your data in a file that you can examine. Try it. Enter a “strange” number or some letters, download the file, and look for your “strange” data at the end of the file.

When you start doing serious experimentation, you will eventually want your own script(s) so that you will not be dependent on our server. With your own script, you can save data on your local server. You can get your scripts in one of several ways.

First, you can ask the person who runs your department's server for a script you can use. If you are a student in a lab, your instructor may arrange one or more scripts for students to use. If you have an account with an Internet service provider, your provider can supply a script for you and give you additional instructions on how to use it. Your

provider's script will probably support adding such variables as date and time to your data file.

Second, you can write a script yourself for your server in a CGI language such as *Perl*, Practical Extraction and Reporting Language, which can be downloaded free (http://www.perl.com/CPAN/). Schwartz (1998) has written a tutorial on Perl, describing its advantages for this purpose. A script in Perl that will work with the examples of this book is included on the CD (Appendix A).

Third, you can use a program to create scripts for you, such as PolyForm (http://software.ora.com/download/), or WWW Survey Assistant (Schmidt, 1997; URL http://or.psychology.dal.ca/~wcs/hidden/home.html).

The generic protocols used in this book were made with PolyForm, which creates scripts for a Windows NT server (http://polyform.ora.com/book/). Appendix A gives more information about scripts, including a discussion of how to use PolyForm. Until you get your own scripts, you are welcome to use the address above and download your data from psych.fullerton.edu.

C. Hidden Variables

Hidden variables can be added with the following tag,

```
<INPUT TYPE="hidden" NAME="00exp" VALUE="Ch5_exp3">
```

This variable will not be displayed to the viewer of the page (but it can be seen in the page source HTML). Hidden variables can be added to the data file to keep track of such things as the name of the experiment, and the condition that the subject received. For example, a hidden variable might identify “Condition A” or “Condition B” in a between-subjects design, in which different people send their data from different Web pages (that present different conditions) to the same script. As with any variable, it must have a name and a value. Hidden variables can also be used to hold the date, time, and other information (Appendix A). The `VALUE` is returned as datum.

The scripts used in this book support insertion of date, time, and the Remote Address (IP) from which the data are sent (this will be a number such as, 137.151.76.48). To insert the time and date, for example, use the following hidden variables:

```
<INPUT TYPE="hidden" NAME="02date" VALUE="pfDate">
<INPUT TYPE="hidden" NAME="03time" VALUE="pfTime">
```

The script will replace the values, `pfDate` and `pfTime,` with the actual date and time that the *submit* button was pressed, according to the clock on the server. For remote address, use pfRemoteAddress; both capitalization and spelling must be exactly as shown.

D. Radio Buttons

Radio buttons are a very useful method for collecting data. They are also convenient for the participant, because they only require a point and click, without typing. In the following example, the participant is asked to indicate his or her gender (M or F). Although gender has two levels, you should use three radio buttons, as follows:

```
<P><INPUT TYPE="radio" NAME="02sex" VALUE="" CHECKED>What is your gender?<BR>
<BLOCKQUOTE><INPUT TYPE="radio" NAME="02sex" VALUE="M">Male<BR>
            <INPUT TYPE="radio" NAME="02sex" VALUE="F">Female<BR></BLOCKQUOTE>
```

The reason to use three buttons instead of two is that some participants may not answer. If so, then you do not want to wrongly put down either male or female for them. Therefore, you should add an extra radio button that is initially checked, whose value will be null (i.e., "") unless the subject clicks either male or female.

In this example, the values M and F were assigned to male and female, but it would be just as clear to use 1 and 0. Think ahead to how the data will be analyzed. If you plan to compute a correlation coefficient between gender and a numerical variable, you should use numerical values, like 1 and 0 to code gender. If you plan to count frequencies of certain

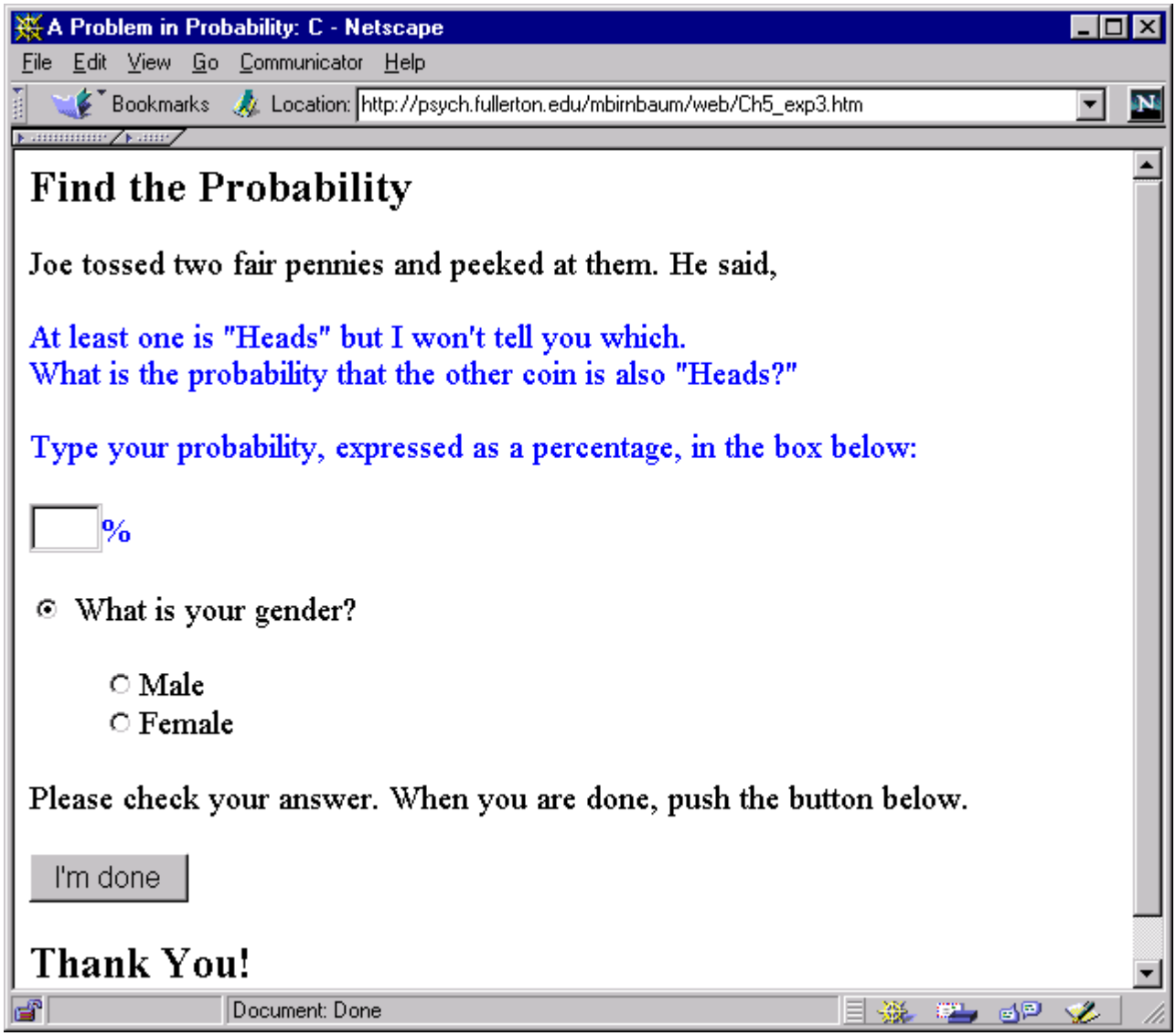
behaviors by males and by females in a table, however, then the table may be easier to read if you use verbal labels such as *Male* and *Female*, or M and F.

The new version of the experiment, *Ch5_exp3.htm*, illustrates hidden variables and radio buttons. Data for this version will be analyzed in Chapter 6.

```
<FORM METHOD="POST" ACTION="http://psych.fullerton.edu/cgi-
win/polyform.exe/generic">
<INPUT TYPE="hidden" NAME="00exp" VALUE="Ch5_exp3">
<INPUT TYPE="text" NAME="01answer" SIZE="2" MAXLENGTH="3">%
<FONT color="black">
<INPUT TYPE="hidden" NAME="02exp" VALUE="Prob Study 1">
<P><INPUT TYPE="radio" NAME="03sex" VALUE="" CHECKED>What is your gender?<BR>
<BLOCKQUOTE><INPUT TYPE="radio" NAME="03sex" VALUE="M">Male<BR>
           <INPUT TYPE="radio" NAME="03sex" VALUE="F">Female<BR>
</BLOCKQUOTE>
<P>Please check your answer. When you are done,
push the button below.
<P><INPUT TYPE="submit" VALUE="I'm done">
<H2>Thank You!</H2>
</FONT>
</FORM>
```

Each group of radio buttons is connected by the same `NAME`. Only one button in a group can be `CHECKED` at a time; clicking one will deselect the previously checked item. Radio buttons are easy for the subject to use for choices between two or more alternatives. They are the natural way to put multiple choice tests, including True/False, on the computer. Rating scales (e.g., Likert scales), used for attitude questionnaires or personality tests can also be well-implemented using radio buttons. Figure 5.4 shows the appearance of the set of three radio buttons. Note that the one in the margin next to the question has a black dot. When the reader clicks on one of the buttons, the dot will move to the selected button. Insert Figure 5.4 about here.

Figure 5.4. The appearance of radio buttons in the browser in *Ch5_exp3.htm*.


A Problem in Probability: C - Netscape
File Edit View Go Communicator Help
Bookmarks Location: http://psych.fullerton.edu/mbirnbaum/web/Ch5_exp3.htm
Find the Probability
Joe tossed two fair pennies and peeked at them. He said,
At least one is "Heads" but I won't tell you which.
What is the probability that the other coin is also "Heads?"
Type your probability, expressed as a percentage, in the box below:
%
What is your gender?
Male
Female
Please check your answer. When you are done, push the button below.
I'm done
Thank You!
Document: Done

HTML also supports a data input method known as the checkbox, which will be described below for completeness. I strongly advise you: Do not use checkboxes! Anything that you can do with a check box, you can do better by using two or three radio buttons. With two radio buttons, one radio button can represent non-selected ("no"), and the other radio button represents the "checked" alternative ("yes"). With three radio buttons, one can represent “yes”, one “no”, and the third can be “no response”. That way you can distinguish three possibilities by asking the subject to click once. Another problem with check boxes is that when they are not checked, they often send nothing. That will create havoc when you go to analyze the data with most data analysis packages because the variables will not be in the same columns. Take my word for it, and use radio buttons instead of check boxes in research.

E. Textareas, Passwords, and Checkboxes

For a short answer, consisting of a few words, the text box is ideal (`SIZE=60` or `100` characters). However, sometimes you may want to invite the participant to give a lengthy answer of a paragraph or more. Textareas are ideal for this situation. Here is an example that illustrates the textarea:

```
<TEXTAREA NAME="03Comments" ROWS=5 COLS=60 WRAP=virtual>text</TEXTAREA>
```

Note that there is a closing tag `</TEXTAREA>` that includes text that will be displayed in the box, which is usually something like, "Please type your answer in this box."

The password Input tag is useful for obtaining a response that a person would not want seen if someone were looking over their shoulder. A password is one example, and a sensitive or personal question might be another. The following example illustrates its use:

```
<INPUT TYPE="password" NAME="04Password" SIZE=10 MAXLENGTH=15>
```

The password input tag works just like text input, except that it displays asterisks or dots instead of characters in the box on the screen.

The checkbox can be made as follows:

```
<INPUT TYPE="checkbox" CHECKED>
```

The checkbox is either initially checked or not. With this response device, it is not possible to distinguish a failure to respond (because the person did not read the question) from a decision not to check the box. Also, this input device may send nothing (not even a null response) when it is left unchecked. The example, *Ch5_exp3b.htm*, illustrates this problem. Try leaving both checkboxes blank.

F. Pull-Down Selections

A selection list works like a pull-down menu. When the participant clicks on the list, the list appears, and the participant can drag the mouse to select an answer from the list of responses. Pull-down selections are interesting, because there is a lot of potential psychology in how an experimenter might bias the answers he or she gets by changing the choices on the list or the manner in which they are displayed. This response procedure is appropriate when there is a clear, finite set of all possible answers. This procedure can also be used for multiple choice items.

An illustration of the various methods for responding is in *Ch5_exp3b.htm*, on the CD. Play with this example to study how these response devices work. Note that in the selection lists, a clear non-answer is pre-selected. This experiment also illustrates how checkboxes can confuse a data file.

You may also find the psychology of response procedures interesting, and you may want to join psychologists who are studying the psychophysical and judgmental processes that affect how people respond when presented with different types of response devices. Schwarz (1999) reviews literature showing that how the question is asked, including the choices of responses, strongly affects the answers found in questionnaire studies.

G. St. Petersburg Paradox and Selection Lists

This section presents an experiment on the classic St. Petersburg paradox to illustrate how pull-down lists work, and also to investigate how they might bias the results in certain situations. The St. Petersburg paradox was originally posed as a mathematical problem, and was answered in 1738 in a classic paper by Daniel Bernoulli.

There is a gamble that has an infinite expected value, yet people prefer gambles with much lower expected value. Here is how the gamble works: a coin is tossed, and if it is heads, you win $2; if it is tails, however, then it is tossed again. Now if it is heads, you win $4, but if tails, then it is tossed again, and heads on the next toss would win $8. Each time tails occurs, the prize for heads on the next toss doubles, and so on forever. How much would you pay to play this gamble once? Would you rather play this gamble once, or have $15 for sure?

The expected value of the St. Petersburg gamble is one-half times $2 plus one-fourth times $4 plus one-eighth times $8 plus, and so on forever. Therefore, the expected value is the sum of an infinite number of 1s. Thus, if you value the gamble at its expected value, you should prefer playing this gamble once to receiving any finite amount of money.

Bernoulli (1738) gave a solution to the St. Petersburg paradox. He argued that although the expected value is infinite, this gamble is worth only $4, for psychological reasons. Bernoulli theorized that if the subjective value of money ("utility" of money) is logarithmically related to objective money, then the gamble has a finite worth. This idea influenced Gustav Fechner, whose work on psychophysical measurement led directly to the creation of the field of Psychology. The idea of diminishing marginal "utility" of money also had considerable impact in the field of Economics as well. You will learn more about Bernoulli's expected utility theory of decision making in Chapter 8, and more about Fechner in Chapter 14.

Figure 5.5. HTML for a pull-down selection list, in Condition 512.

```
<SELECT NAME="01amount">
      <OPTION VALUE="0" SELECTED>0
      <OPTION VALUE="1">1
      <OPTION VALUE="2">2
      <OPTION VALUE="4">4
      <OPTION VALUE="8">8
      <OPTION VALUE="16">16
      <OPTION VALUE="32">32
      <OPTION VALUE="64">64
      <OPTION VALUE="128">128
      <OPTION VALUE="256">256
      <OPTION VALUE="512">512
</SELECT>
```

The experiment in Figure 5.5 shows an example of one condition (Condition 512) of an experiment to investigate both the St. Petersburg paradox and the effect of pull-down menus. The new HTML, which defines a pull down list, is shown in Figure 5.5.

Insert Figure 5.5 about here.

Another condition of the experiment is listed in Figure 5.6. In contrast with the previous version of the experiment, this variation (Ch5_exp4.htm) uses a different selection list, with 11 equally spaced values from $0 to $10. Insert Figure 5.6 about here.

Selection lists can also appear as a box with a scroll bar, which you control with the SIZE attribute. Try changing the <SELECT> tag in the above example as follows:

```
$<SELECT NAME="01amount" SIZE=3>
```

SIZE will determine the number of options showing in the scroll window. Try changing the value of SIZE in the example of Figure 5.6 to see its effect. The appearance of a pull-down selection list is shown in Figure 5.7.

Insert Figure 5.7 about here.

Different values of SIZE might interact with the options given in a selection list to produce different responses from participants. Perhaps options that show without scrolling are more likely to be selected than options that require a considerable time to reach by scrolling. As a psychologist, you should be concerned that the results might be affected by biases produced by response formats presented to participants. One type of bias to avoid is the use of a reasonable answer as a pre-selected option. Always code the default option as a non-response, so that your participant must act to make a response. Otherwise, the study will be biased to confirm the choices set as defaults in the HTML.

Figure 5.6. A portion of the HTML for *Ch5_exp4.htm*.

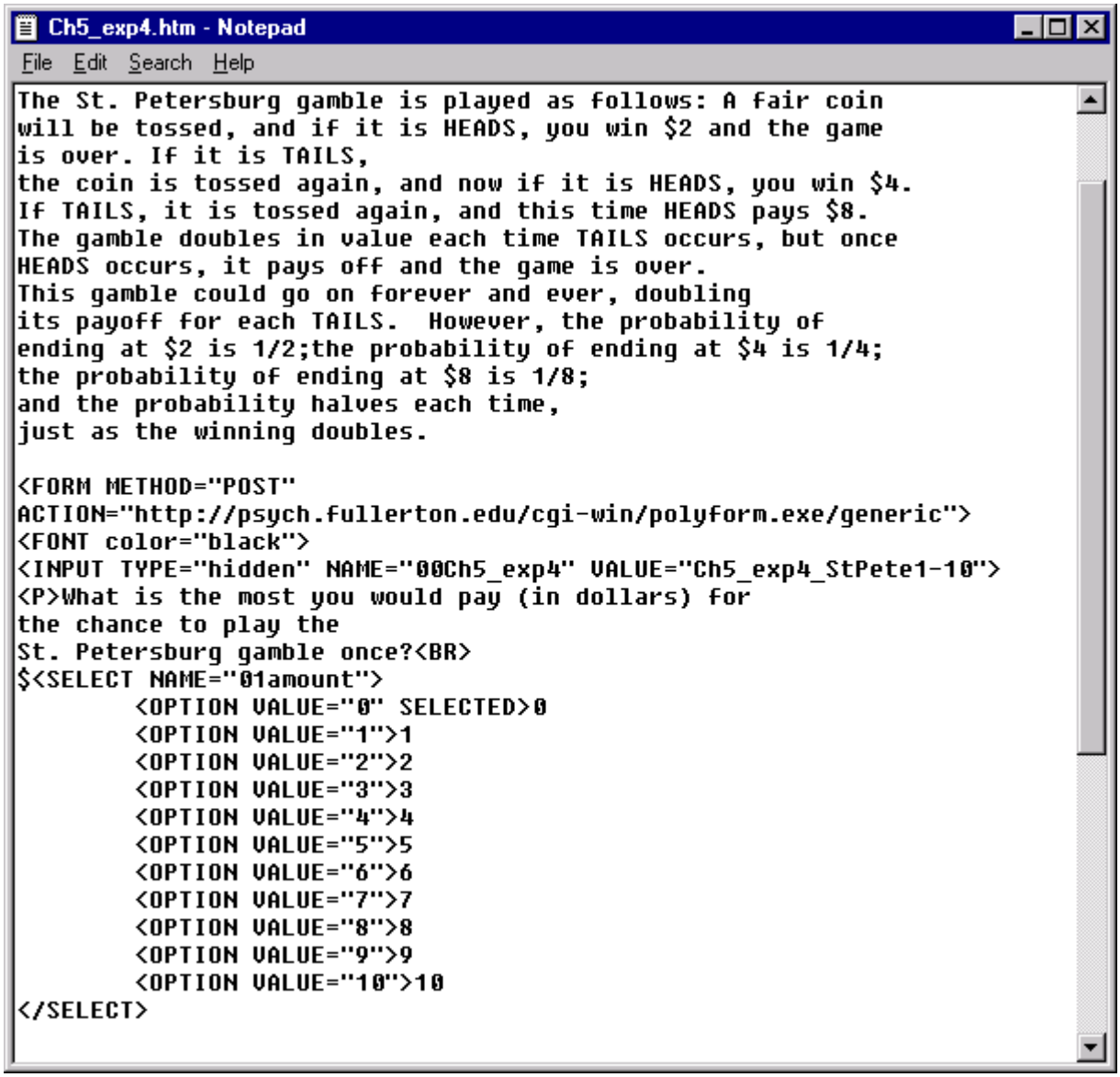
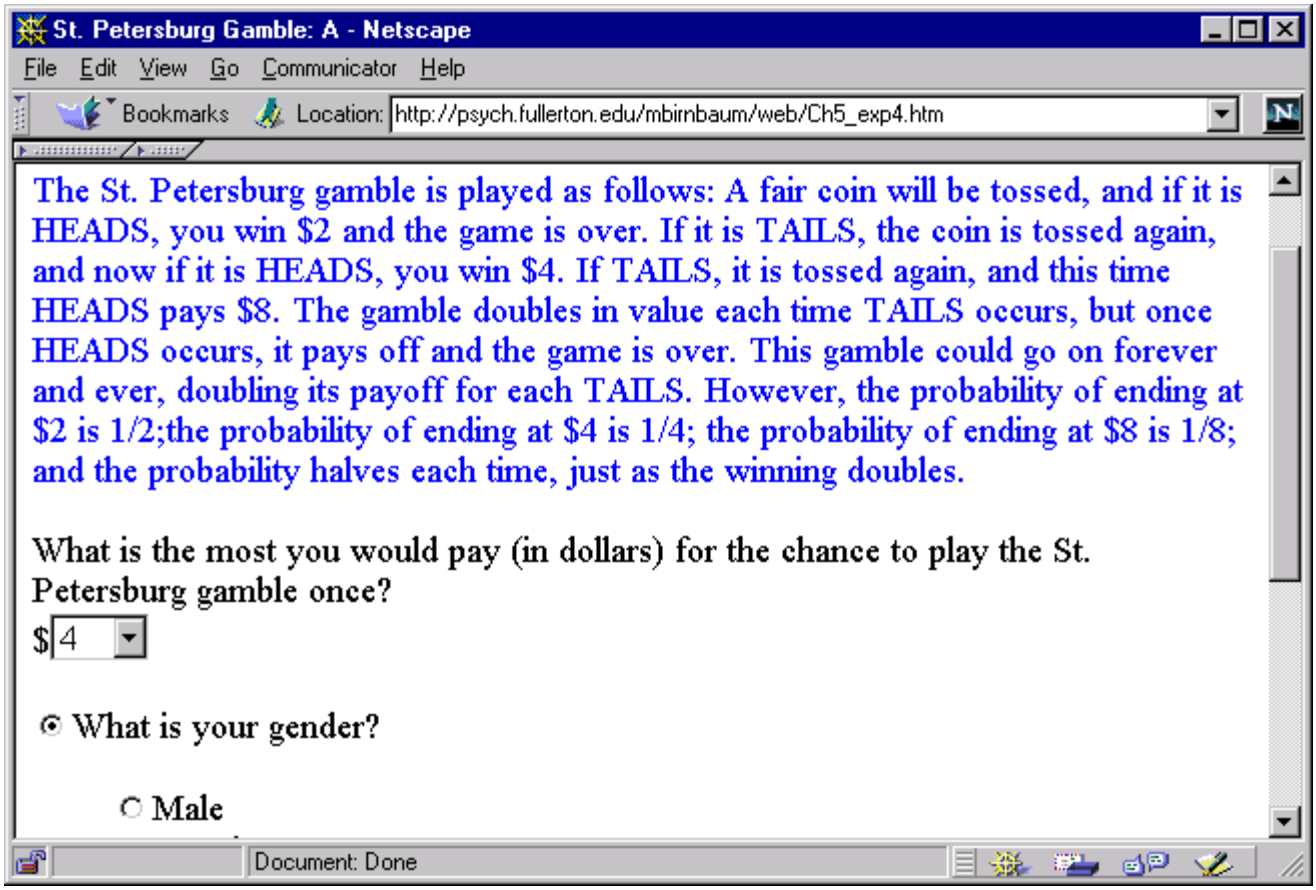


Figure 5.7. Appearance of a pull down menu, in *Ch5_exp4.htm* or *Ch5_exp5.htm*. Clicking on the arrow brings up a finite list of responses. Such a list might bias the response.



Here is the question: Do you think that Condition 10 (equally spaced options from $0 to $10) and Condition 512 (geometrically spaced options from $1 to $512 would produce the same results? Notice that Condition 10 has a maximum of $10, but Condition 512 has a maximum of $512. It seems reasonable to predict that the mean judgment of value will be greater in Condition 512 than in Condition 10. Any participants who might have been willing to say the gamble is worth more than $10 cannot express it in *Ch5_exp4.htm,* but they can go as high as $512 in *Ch5_exp5.htm*.

If anyone wanted to say the gamble was worth say, $6400, that person would not be able to express this response with *either* of the option lists of these experiments. Do you think the experiment would yield different results, if instead of pull-down selections, you used a text box to ask how much a person would pay to play the gamble? You might think it would not introduce any bias to just ask in a text box what the gamble is worth. But even a text box will have a size (in columns) that gives the participant an idea of how big a number they are expected to enter.

To allow people to enter a large number, you can display 30 columns in the text box and allow 40, as follows:

```
<INPUT TYPE="text" NAME="01amount" SIZE=30 MAXLENGTH=40>
```

This INPUT tag is used instead of the pull-down lists in *Ch5_exp6.htm*. The next section will describe a mechanism for assigning people to different versions of the experiment, Ch5_exp4--Ch5_exp6.

H. Between-Subjects Experiment

The file, *Ch5_exp7.htm,* contains a simple method for assignment to conditions in a between-subjects experiment. People are invited to click the month of their birth and they are assigned to *Ch5_exp4.htm, Ch5_exp5.htm,* or *Ch5_exp6.htm,* depending on which month they click. To counterbalance any effect of astrological horoscope on the data, one should reverse the

relationship between months and experiments during the experiment, so that an equal number of people from each birth month eventually end up in each condition (see Chapter 1). One can then separate the effect of horoscope from the effects of the conditions.

At one time, people believed that mechanisms such as the outcome of rolling dice or time and date and time of one's birth were caused by the same forces that ruled the universe, so dice rolls or birth dates could be used to predict future events. Based on considerable evidence with these mechanisms; however, scientists now accept such mechanisms as definitions of randomness. The HTML for this method of assignment is in Figure 5.8.

Insert Figure 5.8 about here.

Figure 5.8. Web page to assign people to conditions to investigate the effect of response procedures. The connection between birth months and conditions is counterbalanced over the run of the experiment. A method of random assignment using JavaScript is presented in Chapter 17.

```
<HTML><HEAD>
<TITLE>St. Petersburg Gamble</TITLE></HEAD>
<H2>Please click on the month of your birth below</H2>
<B><PRE>
</B><A HREF="Ch5_exp4.htm"><B>Jan          April       July         Oct</B></A>
<A HREF="Ch5_exp5.htm"><B>    Feb         May         Aug        Nov</B></A>
<A HREF="Ch5_exp6.htm"><B>        Mar        June         Sept       Dec
</B></A></PRE></BODY></HTML>
```

I. Summary.

In this chapter, you learned that by placing a form within a Web page, you can collect data via the Web either by email or by sending the data to a CGI script that can save the data to the server. With a script to organize and save the data, the Web experiment or survey eliminates the need for an experimenter to be present, or for a researcher to code and enter data.

This chapter described different methods for eliciting responses from subjects, including text boxes (which can be used for either numerical or short answers), radio buttons (which can be used for Likert scales, multiple choice, etc.), textareas, and selection lists. It was recommended that you use radio buttons instead of checkboxes, and that you use TEXT boxes when there is a large set of possible answers. The selection list is appropriate when there is a finite list of possibilities, such as a multiple-choice test, or choosing one's nationality from a finite list of nations. You will learn in the next chapter that the choices in a selection list do influence the results.

Table 5.1 summarizes the tags in this chapter. Insert Table 5.1 about here.

Table 5.1. Summary of Tags in Chapter 5.

Tag	Description
<FORM></FORM>	Material within these tags can contain fields for two-way communication.
ACTION=*URL*	Data are sent to URL to be processed by the script residing at the address.
ACTION=mailto:*user@address.ext*	Data from the FORM will be sent by email when the participant clicks the "submit" button. ENCTYPE="text/plain" improves the appearance of the email.
<INPUT TYPE=*text* NAME=*varname* SIZE=*3* MAXLENGTH=*5*>	Defines an instrument for communicating with a text box of size 3 with room for 5 characters.
<INPUT TYPE=radio NAME=*varname* VALUE=*x*>	Creates a radio button whose value is *x* if the button is clicked.
<SELECT> <OPTION VALUE="*x*">*Option 1 label* <OPTION VALUE="*y*">*Option 2 label* </SELECT>	Defines a selection list. Each option has a value and a description.
<TEXTAREA NAME=*varname* ROWS=*r* COLS=*c* WRAP=virtual> *Anytext* </TEXTAREA>	Creates a textarea that is r rows by c columns. *Anytext* appears in the textarea.
<INPUT TYPE=reset VALUE="*erase*">	Creates a button that will clear the form.
<INPUT TYPE=submit VALUE="*done*">	Creates a submit button that says, "done."
<INPUT TYPE=password SIZE=*8* MAXLENGTH=*8* NAME=*varname*>	Creates a text box that does not display the response.

J. Exercises

1. Use your text editor to modify *Ch5_exp1.htm*, in Figure 5.1. Put your own email address in the `ACTION="mailto:user@address.domain"`. See if your browser will send you the data by email. If it does, then you can use this method to check your experiments before you really need a script. `ENCTYPE="text/plain"` is not necessary, but makes the email look neater. Repeat with *Ch5_exp3.htm* and *Ch5_exp3b.htm*.

2. Now change the ACTION to the URL of the generic script in *Ch5_exp2.htm*. Delete `ENCTYPE="text/plain"`. Load the HTML page into your browser and click the "I'm done" button. If you get a “Thank You” message, then everything has worked, and your data should be available for viewing, along with much other data from a mix of experiments, in the file, *data.csv*, which you can download from psych.fullerton.edu, using your FTP program. Instructions are given in Chapter 2. Your data will be added to the end of the file.

3. Write the HTML to use radio buttons to collect the poll question of Chapter 3, Exercise 1. Include a third button so that you will know if the participant failed to respond.

4. Perform the exercise in *Ch5_exp3b.htm* and study the example. Next, make your own form that has at least one example of each of the following input devices: text box, radio buttons, textarea, password, and pull-down selection list. Include hidden variables of survey’s name (name it, 00exp), date, and time. Include submit and reset buttons. Check that your form works with the generic script and that your data are properly returned to the file. Check to see if the time and date are correct. Remember, the clock is set by Pacific Time.

5. Check with your Internet service providers or your local server administrator to see if you can get help with scripts to process and save your data. In the meantime, you can use the generic script to test the examples in this book and to test pilot studies. (see Appendix A).