more javascript
(adding logic and control)

Javascript

Conditional Statements

Javascript

Conditional Statements

allow your programs to react to different situations
and behave intelligently

a few examples include:
- form validation
- drag and drop
-evaluating user input

http://worldofsolitaire.com

Conditional Statement Basics

also called if/then statements because they perform a task only if the answer to a question is true.

basic structure looks like this:

```
if ( condition )
{
   // something happens here
}
```

if indicates that the programming that follows is a conditional statement

the parenthesis enclose the yes or no question called the condition

curly braces mark the beginning and end of the javascript that should execute if the condition is true

Conditional Statement Basics

in many cases the condition is a comparison between two values. For example, you may have a game that the player wins when the score is over 100...

basic structure looks like this:

```
if  (score > 100)
{
    alert('you won!');
}
```

the important part is *score > 100.* that phrase is the condition and tests whether the value stored in the score variable is greater than 100. if it is, then an alert dialog box appears with the text "you won!". If the player's score is less than or equal to 100, the javascript interpreter skips the alert and moves on.

Conditional Statement

Comparison Operators

==	**Equal to.** Compares to values to see if they are the same.
!=	**Not equal to.** Compares to values to see if they are not the same. Can be used to compare numbers or strings.
>	**Greater than.** Compares two numbers and checks if the number on the left side is greater than the right.
<	**Less than.** Compares two numbers and checks if the number on the left side is less than the right.
>=	**Greater than or equal to.** Compares two numbers and checks if the number on the left side is greater than or the same value as the number on the right.
<=	**Less than or equal to.** Compares two numbers and checks if the number on the left side is less than or the same value as the number on the right.

Conditional Statement Operators

The code that runs if the condition isn't limited to just a single line of code. You can have as many lines of Javascript between the opening and closing curly braces as you'd like.

For example on a quiz:

```
if  (answer == 12)
{
   alert('correct. that is how many eggs in a dozen.');
    numCorrect = numCorrect + 1;
}
```

Conditional Statement Else Clause

We know what happens if the condition is true. But what if the condition is false? The code below doesn't have a backup plan for a condition that turns out to be false.

```
if  (answer == 12)
{
   alert('correct. that is how many eggs in a dozen.');
    numCorrect = numCorrect + 1;
}
```

Conditional Statement Else Clause

The Backup Plan for Conditional Statements is called an else clause. This will execute if the statement is false.

For example on a quiz:

```
if  (answer == 12)
{
   alert('correct. that is how many eggs in a dozen.');
    numCorrect = numCorrect + 1;
} else {
    alert ('wrong! there are 12 eggs in a dozen.'
}
```

Conditional Statement

Testing more than one condition

Sometimes you will want to have more than two outcomes. Javascript allows you to do this with else if statements. The basic structure looks like this:

```
if  ( condition )  {
    // door #1
} else if  ( condition2 ) {
    // door #2
} else  ( condition3 ) {
    // door #3
}
```

notice that the last choice ends in "else" not "else if" because that is the fallback choice or backup plan.

http://www.csupomona.edu/~maflicker/spring12/art356/conditional1.html

Javascript

Functions

provide detailed instructions a single time.

a series of instructions that you set up at the beginning of your script that are stored by the browser and implemented when you call them back in javascript.

Function Basics

invaluable for performing multiple programming steps without writing the code multiple times.

basic structure looks like this:

```
function  fxName ( )
{
   // the javascript you
    want to run
}
```

keyword *function* lets the javascript know you are creating a function.

after the parentheses, the curly braces enclose one or more lines of javascript code. the curly braces mark the beginning and the end of the code that makes up the function

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.
dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<title>try a function</title>
<script type="text/javascript">




</script>
</head>
<body>

  <div>
  <h1>A Basic Function</h1>


</div>

</body>
</html>
```

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.
dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<title>try a function</title>
<script type="text/javascript">
function printToday() {
  var today = new Date();
  document.write(today.toDateString());
}
</script>
</head>
```

the function's name is printToday, and there are two lines of javascript code that retrieves the current date, converts it to a format we can understand and prints the results.

save this file and preview it in the web browser. what happens?

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.
dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<title>try a function</title>
<script type="text/javascript">
function printToday() {
  var today = new Date();
  document.write(today.toDateString());
}
</script>
</head>
<body>

  <div>
  <h1>A Basic Function</h1>
  <p>Today is <strong>
<script type="text/javascript">printToday();
</script>
</strong></p>
</div>

</body>
</html>
```

something actually did happen. you just didn't see it. the browser stored the function but never got to execute it. to make your function run, you need to call it back.

add the code to your page inside the <body> tags, save and preview.

now let's reuse that function and add a footer with the date in it.

Checkpoint:

Create a quiz :)

objectives are:

1) ask questions
you need a way to ask people questions. use prompt() command, and an array for storing your quiz questions.

2) user feedback
determine the correct answer and use a conditional statement. then to let the quiz taker know if they are right or wrong, you can use the alert() command.

3) print results
you need a way to track how well the quiz taker is doing. a variable to keep track of correct responses will work. then you can use the alert() command or the document.write method to announce the final results of the quiz.

objectives are:

1) ask questions
you need a way to ask people questions. use prompt() command, and an array for storing your quiz questions.

2) user feedback
determine the correct answer and use a conditional statement. then to let the quiz taker know if they are right or wrong, you can use the alert() command.

3) print results
you need a way to track how well the quiz taker is doing. a variable to keep track of correct responses will work. then you can use the alert() command or the document.write method to announce the final results of the quiz.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<title>my new quiz</title>
<!-- link to external css should go here-->
        <script type="text/javascript">

        </script>

</head>
<body>


<div>

        <h1>A Simple Quiz</h1>
        <p>
        <script type="text/javascript">

        </script>
        </p>
</div>


</body>
</html>
```

```
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<title>my new quiz</title>
<!-- link to external css should go here-->
        <script type="text/javascript">
                var score = 0;
        </script>

</head>
```

this variable stores the number of questions the quiz taker gets right. before the quiz is taken the default value is 0.

```
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<title>my new quiz</title>
<!-- link to external css should go here-->
        <script type="text/javascript">
                var score = 0;
                var questions = [


                ];
        </script>

</head>
```

you will be storing all of the questions inside an array, which is a variable that can hold multiple items

```
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<title>my new quiz</title>
<!-- link to external css should go here-->
        <script type="text/javascript">
                var score = 0;
                var questions = [
        ['how many legs do spiders have?' , 8],

                ];
        </script>

</head>
```

this is an example of a nested array (or multidimensional array) which means that you create an array that includes the question AND the answer, which is stored inside the 'questions' array.

the first item is a question, the second item is the answer. this array does not have a name since it is nested inside another array. the comma at the end of the line marks the end of the first question and indicates another array item will follow.

after the last item in the array, you do NOT type a comma.

```
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<title>my new quiz</title>
<!-- link to external css should go here-->
        <script type="text/javascript">
                var score = 0;
                var questions = [
        ['how many legs do spiders have?' , 8],

        //ADD THREE MORE QUESTIONS

                ];
        </script>

</head>
```

```
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<title>my new quiz</title>
<!-- link to external css should go here-->
        <script type="text/javascript">
                var score = 0;
                var questions = [
        ['how many legs do spiders have?' , 8],

        //ADD THREE MORE QUESTIONS

                ];

        //go through the list of questions and ask each one
        for (var i=0; i<questions.length; i++) {
        askQuestion(questions[i]);
                }
        </script>

</head>
```

this line is the first part of a for loop. it creates a new variable named i and stores the number 0 in it.

the second part is the condition which tests to see if the value in i is less than the number of questions in the array.

i++ changes the value of i each time through the loop, it adds 1 to the value of i.

```
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<title>my new quiz</title>
<!-- link to external css should go here-->
        <script type="text/javascript">
                var score = 0;
                var questions = [
        ['how many legs do spiders have?' , 8],

        //ADD THREE MORE QUESTIONS

                ];

        //go through the list of questions and ask each one
        for (var i=0; i<questions.length; i++) {
        askQuestion(questions[i]);
                }

//function for asking question
function askQuestion(question) {
  var answer = prompt(question[0],'');
        if (answer == question[1]) {
                alert('Correct!');
                score++;
        } else {
                alert('Sorry. The correct answer is ' + question[1]);
        }
}
</head>
```

this portion of code will execute the function called 'askQuestion'

by creating a function that asks the questions you will make a more flexible program

```
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<title>my new quiz</title>
<!-- link to external css should go here-->
        <script type="text/javascript">
                var score = 0;
                var questions = [
        ['how many legs do spiders have?' , 8],

        //ADD THREE MORE QUESTIONS

                ];

        //go through the list of questions and ask each one
        for (var i=0; i<questions.length; i++) {
        askQuestion(questions[i]);
                }

//function for asking question
function askQuestion(question) {
  var answer = prompt(question[0],'');
        if (answer == question[1]) {
                alert('Correct!');
                score++;
        } else {
                alert('Sorry. The correct answer is ' + question[1]);
        }
}
</head>
```

this code indicates the body of the function. the function receives a single argument and stores it in variable named 'question'.

THIS IS NOT THE SAME AS THE ARRAY NAMED 'QUESTIONS'

at this point the quiz is functional, if you uploaded it, you would be able to see it, BUT there is no way to see results yet...

```
<body>


<div>

        <h1>A Simple Quiz</h1>
        <p>
        <script type="text/javascript">
                var message = 'You got ' + score;
                message += ' out of ' + questions.length;
                message += ' questions correct.';
                document.write(message);
        </script>
        </p>
</div>


</body>
```

this chunk of code will create a new variable called "message" to store the string 'you got ' plus the quiz taker's score.

then it adds ' out of' and ' questions correct.' to the message string.

and finally it adds the document.write command to write the strings and variables to the page.
(you could also use alert() command)

Save your quiz and test it.

try adding additional questions to the questions array. you should have a total of five questions in all.

upload your completed quiz to your site and ask someone to complete it for fun.

this is checkpoint #1 :)