Programming

• Why is programming fun?
  • First is the sheer joy of making things. As the child delights in his mud pie, so the adult enjoys building things, especially things of his own design. I think this delight must be an image of God's delight in making things, a delight shown in the distinctness and newness of each leaf and each snowflake.
  
  Source: Frederick P. Brooks, Jr. The Mythical Man-Month: Essays on Software Engineering

Grading

• Quiz 3 drew from your labs as well as the questions in each chapter.
  • Unfair since I didn't tell you in advance.
  • Last week was hard with 2 quizzes and a project due.
  • We won't do that again!
  • How to make it fair:
    • We will drop your lowest quiz score for the quarter.

Homework

• By today you should have read
  • Fluency
    • Chapters 20, 21, 22
  • QuickStart
    • Chapters 1 and 2, and pp. 108-113

Prepare

This week’s quiz
  • Review the questions at the end of each chapter:
    • Fluency chapters 20, 21, and 22
    • QuickStart chapters 1 and 2
  
Expect lots of questions on JavaScript!

JavaScript topics will include:
  • Variables
  • Values
  • Assignment statements
  • Conditionals
  • Functions
  • Curly brackets
  • Relationship to HTML

Schedule Changes

Monday and Tuesday:
  • Keep working on Lab 7
  • Due at your Wednesday or Thursday lab this week

Deadline for next project is postponed
  • Will introduce the next project next Monday
  • Student As Teacher—Creating an Online Quiz using JavaScript

Functions & Abstraction

A function is a package for an algorithm; once written, it can be use over and over.
The Package

Functions have a specific syntax

```javascript
function <name> ( <parameter list> ) {<function definition>
  • <name> names are identifiers; start w/letter
  • <parameter list> is the input variables, a list separated by commas
  • <function definition> is just the program to do the work
}
```

A Sample Function

Compute the Body Mass Index when the input data is in metric

```javascript
function bmiM ( weightKg, heightM ) {
  // Figure Body Mass Index in metric units
  return weightKg / (heightM * heightM);
}
```

Writing Functions

Most programming is done by writing functions, so learning the form is key

```javascript
function bmiE ( weightLBS, heightIn ) {
  // Figure Body Mass Index in English units
  var heightFt = heightIn / 12; // Change to feet
  return 4.89 * weightLBS / (heightFt * heightFt);
}
```

Declarations

A function is declared by writing down the “package” ... the function is used when it is called

```javascript
function BMI (units, height, weight) {
  // Compute BMI in either metric or English
  if (units == "English")
    return bmiE(weight, height);
  else
    return bmiM(weight, height);
}
```

Summarizing

Declaration: the function “package,” says what happens when the function runs

Call: the function use, asks for the computation to be run

• There is only one function declaration
• There can be many calls ... functions are reusable
• In JS, functions tend to be grouped together but the calls go where they are needed

Assignment Statements

Review

Variable is replaced by Expression
Gold Function

Suppose we compute “weight in Au”

\[
\text{Worth} = (\text{Weight} \times 12) \times 368.4
\]

Begin with the form ...

\begin{verbatim}
function worthau ( weight ) {
  // Compute the dollar value
  // of weight at $368.40/tz
  return weight * 12 * 368.4;
}
\end{verbatim}

Pick a Name

Pick the Parameter

Define the Computation

Testing

No one writes perfect programs the first time … smart programmers check!

To test, have a standard page handy

\begin{verbatim}
<html><head><title>My Test Page</title></head>
<body>
<script type="text/javascript">
  function worthau ( weight ) {
    // Compute the dollar value
    // of weight at $368.40/troy oz
    return weight * 12 * 368.4;
  }
  alert(worthau(1/12));
</script>
</body>
</html>
\end{verbatim}
Unquestionably, the best practice is to test everything

Try The Function

Function Features

Reviewing properties of functions

- Selecting names... don't use alert()
- Parameter variables... don't need to be declared, but work like local variables

```javascript
function bmiE(weightLBS, heightIn) {
  // Figure BMI in English
  var heightFt = heightIn / 12; // Change to feet
  return 4.89 * weightLBS / (heightFt * heightFt);
}
```

```javascript
function bmiE(weightLBS, height) {
  // Figure BMI in English (height in in)
  height = height / 12; // Change to feet
  return 4.89 * weightLBS / (height * height);
}
```

Function Features (cont.)

- Scope of Reference... refers to where in the program a variable is "known," i.e. where its value can be referenced and/or assigned

```javascript
function bmiE(weight, heightIn) {
  // Figure BMI in English
  var heightFt = heightIn / 12; // Change to feet
  return 4.89 * weight / (heightFt * heightFt);
}
```

```javascript
function BMI(units, height, weight) {
  // Compute BMI
  if (units == "English")
    return bmiE(weight, height);
  else
    return bmiM(weight, height);
}
```

Function Features (cont.)

- Global... declared outside of functions

```javascript
var priceperoz = 368.40;
...

function worthau(weight) {
  // Compute the dollar value of weight at priceperoz
  return weight * 12 * priceperoz;
}
...
```

Function Features (cont.)

- Parameters vs Arguments... parameters are the "formal" variables in the function declaration; arguments are the same thing in the call

```javascript
function bmiE(units, height, weight) {
  // Compute BMI
  if (units == "English")
    return bmiE(weight, height);
  else
    return bmiM(weight, height);
}
```

Assignments

- Lab 7 today and tomorrow
- Read QuickStart
  - chapter 3 for Wednesday
- Prepare for Quiz 4
Summary

Functions are packages for algorithms

- They follow a series of rules, that quickly become intuitive
- Functions have both a declaration and a call
- Functions have both parameters (in the declaration) and arguments (in the call)
- Scope refers to the region of a program where a variable is “known”

Functions are the secret to building complex systems