

Announcements

- Extra credit (40 points each)
 - Labs 8/9
 - Upload to Catalyst Collect It (new dropbox)
 - Living Computer Museum
 - 1-page reflection paper OR
 - 1-minute podcast
- Do either, both, or none (optional)

11/12/2009 FIT100, D.A. Clements, UW iSchool 1

Announcements

- Project 2A and Labs 8/9 due Wednesday at 10pm

11/12/2009 FIT100, D.A. Clements, UW iSchool 2

Announcement

- Labs 8/9—Last page
 - Added some instructions
 - It is 2310:00am...

North American Time Zones

At :00 in the Pacific Time Zone,
it is 2310:00am in the following time zone:

Hawaii
 Alaska
 Pacific
 Mountain
 Central
 Eastern
 Atlantic

11/12/2009 FIT100, D.A. Clements, UW iSchool 3

Project 2A

- MadLib Examples
 - http://www.eduplace.com/tales/content/wwt_003.html
 - http://courses.washington.edu/fit100/projects/key/storytellerForm_2A_KEY.html
- Your ten replacement words
 - Nine text boxes
 - One dropdown menu / select box (gender)

11/12/2009 FIT100, D.A. Clements, UW iSchool 4

Conditionals, Branches, or Tests

Adding logic to an algorithm

D.A. Clements

11/12/2009 FIT100, D.A. Clements, UW iSchool 5

Conditional Statement Syntax

```
if ( <Boolean expression > )
  <then-statement>;
```

- Boolean expression is a relational expression
- then-statement is any JavaScript statement

11/12/2009 FIT100, D.A. Clements, UW iSchool 6

Example

```
if (today == "Friday")
{
    reading = "Fluency, chapter 21";
}
else if (today == "Monday")
{
    reading = "Fluency, chapter 19 and JavaScript
    Phrasebook, ch 8 Forms";
}
```

11/12/2009

FIT100, D.A. Clements, UW iSchool

7

If Statements Control Flow

- The Boolean statement is evaluated, producing a true or false outcome
- If the outcome is true, the then-statement is performed
- If the outcome is false, the then-statement is skipped
- Then-statement can be written on the same line as the Boolean or on the next line

11/12/2009

FIT100, D.A. Clements, UW iSchool

8

Compound If Statements

- Sometimes we need to perform more than one statement on a true outcome of the predicate test
- Group these statements using curly braces { }
- { and } are delimiters just like
 - < and > in HTML
 - " and " for strings

11/12/2009

FIT100, D.A. Clements, UW iSchool

9

Example Compound If-Else

```
if (age <=19 && age > 12)
{
    group = teenagers;
}
```

11/12/2009

FIT100, D.A. Clements, UW iSchool

10

if/else Statements

- To execute statements if a condition is false

```
if ( <Boolean expression> )
{
    <then-statements>;
}
else
{
    <else-statements>;
}
```
- The Boolean expression is evaluated first
 - If the outcome is true, the then-statements are executed and the else-statements are skipped
 - If the outcome is false, the then-statements are skipped and the else-statements are executed

11/12/2009

FIT100, D.A. Clements, UW iSchool

11

Nested if/else Statements

- The then-statement and the else-statement can contain an if/else
- The else is associated with the immediately preceding if
- Correct use of curly braces ensures that the else matches with its if

11/12/2009

FIT100, D.A. Clements, UW iSchool

12

Nested if/else Statements

```
if (<Boolean exp1>) {  
  if (< Boolean exp2>) {  
    <then-stmts for exp2>;  
  } else {  
    <else-stmts for exp2>;  
  }  
}
```

```
if (<Boolean exp1>) {  
  if (< Boolean exp2>) {  
    <then-stmts for exp2>;  
  } else {  
    <else-stmts for exp1>;  
  }  
}
```

11/12/2009

FIT100, D.A. Clements, UW iSchool

13

Nested if/else Statements

```
if (<Boolean exp1>  
{  
  if (< Boolean exp2>)  
  {  
    <then-stmts for exp2>;  
  }  
  else  
  {  
    <else-stmts for exp2>;  
  }  
}
```

```
if (<Boolean exp1>)  
{  
  if (< Boolean exp2>)  
  {  
    <then-stmts for exp2>;  
  }  
  else  
  {  
    <else-stmts for exp1>;  
  }  
}
```

11/12/2009

FIT100, D.A. Clements, UW iSchool

14

Demo—Nested If-Else

```
if (genderList == "Male")  
{  
  gender = malePronouns;  
  if (age > 19)  
  {  
    group = adults;  
  } else if (age <= 19 && age > 12)  
  {  
    group = teens;  
  } else {  
    group = kids;  
  }  
}
```

11/12/2009

FIT100, D.A. Clements, UW iSchool

15

Next week's quiz topics

11/12/2009

FIT100, D.A. Clements, UW iSchool

16

Arrays

Indexing a Collection of Items
D.A. Clements

11/12/2009

FIT100, D.A. Clements, UW iSchool

17

Arrays

- Indexing
 - Creating and using lists, or arrays
- Processing an array
 - Element by element
- Array methods
 - Quick work with lists

11/12/2009

FIT100, D.A. Clements, UW iSchool

18

Creating and using lists, or arrays

INDEXING

11/12/2009 FIT100, D.A. Clements, UW iSchool 19

What is an Array?

- An indexed list of items, or elements
 - Indexed means each element in the list has a number, or index

1. George Washington	16. Abraham Lincoln	31. Herbert Hoover
2. John Adams	17. Andrew Johnson	32. Franklin D. Roosevelt
3. Thomas Jefferson	18. Ulysses S. Grant	33. Harry S. Truman
4. James Madison	19. Rutherford B Hayes	34. Dwight Eisenhower
5. James Monroe	20. James Garfield	35. John Kennedy
6. John Quincy Adams	21. Chester Arthur	36. Lyndon Johnson
7. Andrew Jackson	22. Grover Cleveland	37. Richard Nixon
8. Martin Van Buren	23. Benjamin Franklin	38. Gerald Ford
9. William Harrison	24. Grover Cleveland	39. James Carter
10. John Tyler	25. William McKinley	40. Ronald Reagan
11. James Polk	26. Theodore Roosevelt	41. George H. W. Bush
12. Zachary Taylor	27. William H. Taft	42. William Clinton
13. Millard Fillmore	28. Woodrow Wilson	43. George W. Bush
14. Franklin Pierce	29. Warren Harding	44. Barack Obama
15. James Buchanan	30. Calvin Coolidge	

11/12/2009 FIT100, D.A. Clements, UW iSchool 20

Indexing

- Process of creating a sequence of names by associating a base name with a number (like Apollo 13 or Henry VIII)
 - Each indexed item is called an element of the base-named sequence
- Index Syntax
 - index number is enclosed in square brackets []
- Iterations can be used to refer to all elements of a name
 - `A[j]` for successive iterations over `j` referring to different elements of `A`

11/12/2009 FIT100, D.A. Clements, UW iSchool 21

Indexing (cont'd)

- Index Origin*
 - The point at which indexing begins (the least index)
 - In life, the first element may begin with 1, or have no number (Queen Elizabeth)
 - JavaScript *always* uses index origin 0

11/12/2009 FIT100, D.A. Clements, UW iSchool 22

Rules for Arrays

- Arrays are variables initialized by `new Array (<number of elements>);`
- <number of elements> is number of items in ;
- Array indexing begins at 0
- Greatest index is <number of elements> - 1
- Number of elements is array length
- Index values range from 0 to (length - 1)

11/12/2009 FIT100, D.A. Clements, UW iSchool 23

Syntax for Arrays

- Initialize array
 - with name and # elements


```
books = new Array (6);
```
 - with name and elements


```
books = new Array ('War and Peace', 'Tom Sawyer', 'Jane Eyre');
```
- Add elements


```
books[3] = 'Pride and Prejudice';
books[4] = 'Moby Dick';
books[5] = 'Captain Horatio Hornblower';
```

11/12/2009 FIT100, D.A. Clements, UW iSchool 24

Syntax for Arrays

- Reference an element of the array:
 - Index must be a non-negative integer or expression or variable that resolves to non-negative integer

`array[i]`

11/12/2009

FIT100, D.A. Clements, UW iSchool

25



Element by element

PROCESSING AN ARRAY

11/12/2009

FIT100, D.A. Clements, UW iSchool

26

Array Reference Syntax

- The World-Famous Iteration, or 0-origin loop iteration, is perfect for looping through arrays
 - Start at 0
 - Increment by 1 to process every element in the array
 - Use the incrementing variable as the index for the array element
 - End when you reach the last element in the array

11/12/2009

FIT100, D.A. Clements, UW iSchool

27



for Loops Rule

- The World-Famous Iteration for looping through an array:

```
for ( i = 0; i < fruits.length; i++ )
{
    alert(fruits[i]);
}
```

- `.length` is a built-in JavaScript property that always gives you the length of an array
 - Length of an array is the number of elements

11/12/2009

FIT100, D.A. Clements, UW iSchool

28

Demonstration

- Looping through the fruits array

11/12/2009

FIT100, D.A. Clements, UW iSchool

29



Processing elements in an array

```
var i, text=""; //declare iteration and other variables
var fruits = new Array(
    'lemons','apples','mangoes','tangerines','kumquats',
    'cantaloupe','peaches','grapefruit','raspberries');
alert("Total number of fruits is " + fruits.length);
for (i=0; i<fruits.length; i++)
{
    text += i + '. ' + fruits[i] + '<br />';
}
document.write("<h1>Elements of Fruits Array:</h1><p>" + text +
"</p>");
```

11/12/2009

FIT100, D.A. Clements, UW iSchool

30

Array Methods: `.push`

- ```
var i, text=""; //declare iteration and other variables
var fruits = new Array(
 'lemons','apples','mangoes','tangerines','kumquats','cantaloupe',
 'peaches','grapefruit','raspberries');
fruits.push('bananas','oranges','pears');
alert("Total number of fruits is " + fruits.length);
for (i=0; i<fruits.length; i++)
{
 text += i + '. ' + fruits[i] + '
';
}
document.write("<h1>Elements of Fruits Array:</h1><p>" + text + "</p>");
```

11/12/2009

FIT100, D.A. Clements, UW iSchool

31

## for Loops Rule!

- Now we've added more elements to our array. Do we need to change anything in our for loop?

```
for (i = 0; i < fruits.length; i++)
{
 alert(fruits[i]);
}
```

- No! `fruits.length` still takes us to the end of the fruits array—whatever its length.

11/12/2009

FIT100, D.A. Clements, UW iSchool

32

## Array Methods: `push`

- Verify it by looping through the expanded fruits array

11/12/2009

FIT100, D.A. Clements, UW iSchool

33



Quick work with lists

## ARRAY METHODS

11/12/2009

FIT100, D.A. Clements, UW iSchool

34

## Array Methods = Possibilities!

- **push**
  - adds elements to the array  
`fruits.push('bananas','nectarines','apples');`
- **pop**
  - pulls the last element off of the array  
`fruits.pop();`
- **concat**
  - combines several arrays into one
  - Note: copies of the arrays are used
  - The original arrays remain and are unaffected  
`fruits.concat(citrus,stoneFruit,berries);`

11/12/2009

FIT100, D.A. Clements, UW iSchool

35

## Array Methods = More Possibilities!

- **join**
  - combines all elements into a string, separated by commas or as specified  
`fruits.join(';');`
- **sort**
  - sorts the elements in the array  
`fruits.sort(); //always ascending`
- **reverse**
  - reverses the elements in an array
  - Used with `sort` to sort descending  
`fruits.sort(); //sorts into ascending order`  
`fruits.reverse(); //reverses to descending`

11/12/2009

FIT100, D.A. Clements, UW iSchool

36

## Array Methods = More Possibilities!

- **toString**
  - converts the array to a string  
fruits.toString();

11/12/2009 FIT100, D.A. Clements, UW iSchool 37

## Array Method: **sort**

- Sort with **.sort**
  - Ascending only (A-Z, 0-9)

```
var i, text=""; //declare iteration and other variables
var fruits = new Array(
 'lemons','apples','mangoes','tangerines','kumquats','cantaloupe',
 'peaches','grapefruit','raspberries');
fruits.push('bananas','oranges','pears');
fruits.sort();
alert("Total number of fruits is " + fruits.length);
for (i=0; i<fruits.length; i++)
{
 text += i + '. ' + fruits[i] + '
';
}
document.write("<h1>Elements of Fruits Array:</h1><p>" + text + "</p>");
```

11/12/2009 FIT100, D.A. Clements, UW iSchool 38

## Array Sort

- Demonstration

11/12/2009 FIT100, D.A. Clements, UW iSchool 39

## Sort the Array in Descending Order

- Reverse the sort with **.reverse**

```
var i, text=""; //declare iteration and other variables
var fruits = new Array(
 'lemons','apples','mangoes','tangerines','kumquats','cantaloupe',
 'peaches','grapefruit','raspberries');
fruits.push('bananas','oranges','pears');
fruits.sort();
fruits.reverse();
alert("Total number of fruits is " + fruits.length);
for (i=0; i<fruits.length; i++)
{
 text += i + '. ' + fruits[i] + '
';
}
document.write("<h1>Elements of Fruits Array:</h1><p>" + text + "</p>");
```

11/12/2009 FIT100, D.A. Clements, UW iSchool 40

## Array Method: **reverse**

- Demonstration

11/12/2009 FIT100, D.A. Clements, UW iSchool 41

## End papers...

- Why is programming fun?
  - Second is the pleasure of making things that are useful to other people. Deep within, we want others to use our work and to find it helpful. In this respect the programming system is not essentially different from the child's first clay pencil holder "for Daddy's office."

Source: Frederick P. Brooks, Jr. *The Mythical Man-Month: Essays on Software Engineering*

11/12/2009 FIT100, D.A. Clements, UW iSchool 42