Introduction to Programming for Scientists

Lecture 11

Regular Expressions

Parsing

Prof. Steven Ludtke
N410, sludtke@bcm.edu

Web Scripting

Scripting, Server vs. Client

- Serverside scripting depends on the webserver you use
  - Many choices
  - May put load on server
- Clientside (with server support)
  - Java - often available, but many issues
  - Flash - Almost ubiquitous, but somewhat proprietary
  - Javascript built in to mostbrowsers
    - AJAX - Asynchronous Javascript And XML

Javascript - Button

```
<HTML><HEAD><TITLE>Hi there</TITLE></HEAD>
<BODY>
<h3>Here is a title</h3>
And some text
<p>
<input type="button" value="Push Me" onclick="alert('You pushed me too far')">
</p>
</body>
```

Javascript - mouseover

```
<HTML><HEAD><TITLE>Hi there</TITLE></HEAD>
<BODY>
<h3>Here is a title</h3>
And some text
<p>
<a href="index3.html" onmouseover="window.document.bgColor='red'">Red</a>
<a href="index3.html" onmouseover="window.document.bgColor='green'">Green</a>
<a href="index3.html" onmouseover="window.document.bgColor='blue'">Blue</a>
<a href="index3.html" onmouseover="window.document.bgColor='white'">White</a>
</p>
</body>
```

Javascript - Statements

- var name[=value],name[=value]
- function f(x,y) statement
- if (expression) statement; else statement;
- do statement while (expression)
- while (expression) statement
- for (var in array) statement
- for (init; update; test) statement
- switch (expr) {

case const:
statements
break
default:
statements }

Javascript - Events

- onclick
- onfocus, onblur
- onmousedown, up, move, over,out
- onkeydown, up, press
- onreset
- onsubmit
- onload, unload

Javascript Calculator

```
<HTML><HEAD><TITLE>Hi there</TITLE></HEAD>
<BODY>
<h3>Calculator</h3>
<form name=calc onsubmit=compute()>
<input type=text name=data></input>
</form>
<script>
document.calc.data.value=window.location.search.split("=")[1]
function compute() {
document.calc.data.value=eval(document.calc.data.value);
}
</script>
</body>
```

Javascript - Calculator #2

```
<HTML><HEAD><TITLE>Hi there</TITLE></HEAD>
<BODY>
<h3>Calculator</h3>
<form name=calc onsubmit=compute()>
<input type=text name=data value="0"></input>
<table><tr>
<td><input type="button" value="7" onclick="num('7')"></td>
<td><input type="button" value="8" onclick="num('8')"></td>
<td><input type="button" value="9" onclick="num('9')"></td>
<td><input type="button" value="X" onclick="fn('*')"></td></tr><tr>
<td><input type="button" value="4" onclick="num('4')"></td>
<td><input type="button" value="5" onclick="num('5')"></td>
<td><input type="button" value="6" onclick="num('6')"></td>
<td><input type="button" value="-" onclick="fn('-')"></td></tr><tr>
<td><input type="button" value="1" onclick="num('1')"></td>
<td><input type="button" value="2" onclick="num('2')"></td>
<td><input type="button" value="3" onclick="num('3')"></td>
<td><input type="button" value="+" onclick="fn('+')"></td></tr><tr>
<td colspan=3><input type="button" value="0" onclick="num('0')"></td>
<td><input type="button" value="=" onclick="eql()"></td>
</tr> </table> </form>
```

Javascript - Calculator #2

```
<script>
xpr=""
rst=1
function num(val) {
        xpr+=val
        if (rst) {
                rst=0
                document.calc.data.value=""
        }
        document.calc.data.value+=val
}

function fn(val) {
        xpr+=val
        rst=1
}

function eql() {
        document.calc.data.value=eval(xpr)
        xpr=""
        rst=1
}
</script>
</body>
```

References

- http://www.w3.org/TR/html4/
- http://www.w3.org/TR/html4/index/elements.html
- http://htmlhelp.com/reference/html40/olist.html

- http://www.javascriptkit.com/jsref
- http://www.w3schools.com/jsref/default.asp

Regular Expressions

- '.' - any character
- [abcd] - match any character in the list, may use '-' or '^'
- '\s' - any whitespace character [\t\n\r\f\v]
- '|' - or, match either of 2 expressions
- (...) - used to group parts of an expression
- (?P<name>...) - a 'named' group (see groupdict)
- '*' - 0 or more repetitions of the preceding element
- '+' - 1 or more repetitions of the preceding element
- '?' - 0 or 1 repetitions of the preceding element
- '*?','+?','??' - non greedy version of *, + and ?
- {m} - match exactly m copies of the preceding element
- '^' - start of the string
- '$' - end of the string
- ..... there are more

Regular Expressions

re functions

- re.search(pattern,string) - search the entire string for pattern
- re.match(pattern,string) - check the beginning of the string only
- re.split(pattern,string) - much like string.split()
- re.findall(pattern,string) - list of all non-overlapping instances
- re.finditer(pattern,string) - Match object for each match
- re.sub(pattern,repl,string) - replace matches with repl

Regular Expressions

Match objects

- group(n) - returns the matching part of the string in group n
- groups() - returns a tuple with all subgroups
- groupdict() - returns a dictionary of results based on <> names
- start(),end() - index of start or end of match

Testing Regular Expressions

- http://cthedot.de/retest/
- http://re-try.appspot.com/

What do you want to see next time ?

- Parsers
- Py-Qt
- Binary files, Proprietary data formats
- Databases
- Threads
- Debugging/Profiling
- Assembling a PC from component parts
- ???

- Work on your class projects !!!