



---

# Debugging and Troubleshooting

INFO/CSE 100, Spring 2006

Fluency in Information Technology

<http://www.cs.washington.edu/100>



# Readings and References

---

- Reading
  - » *Fluency with Information Technology*
    - Chapter 7, To Err is Human
      - “To err is human, but it takes a computer to really foul things up”
- References
  - » *World Wide Web Consortium*
    - [http://www.w3schools.com/html/html\\_reference.asp](http://www.w3schools.com/html/html_reference.asp)
    - <http://validator.w3.org>
  - » Jedit (java-based editor)
    - <http://www.jedit.org>

# Advanced HTML

---

`<hr />` a horizontal rule

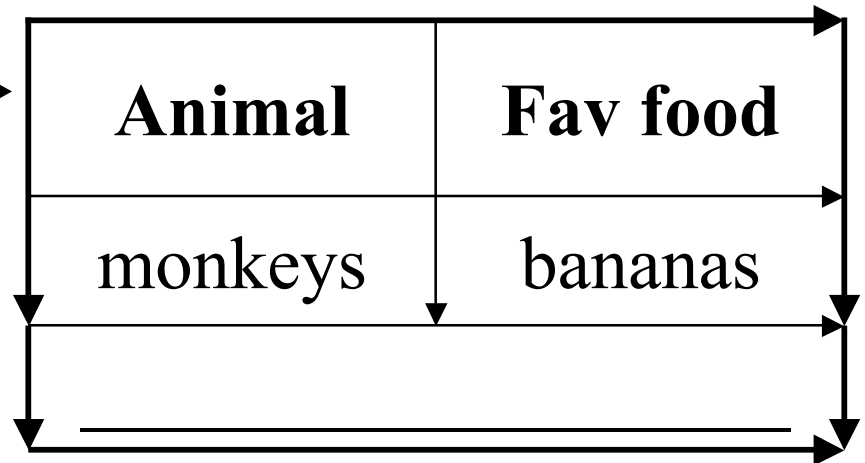
---

`<p>`This paragraph has space  
around it.`</p>`

This is a line`<br />`  
break carried onto the next  
line

# HTML Tables

```
<table border="1">  
  <tr>  
    <th>Animal</th>  
    <th>Fav food</th>  
  </tr>  
  <tr>  
    <td>monkeys</td>  
    <td>bananas</td>  
  </tr>  
  <tr><td colspan="2"><hr /></td></tr>  
</table>
```



Animal	Fav food
monkeys	bananas
<hr/>	



# List Items

---

```
<ul>
  <li>List item 1</li>
  <li>
    <ol>
      <li>Another list</li>
      <li>first item</li>
      <li>second item</li>
    </ol><li>
</ul>
```

- List item 1
- Another list
  1. first item
  2. second item
-



# Character Entities

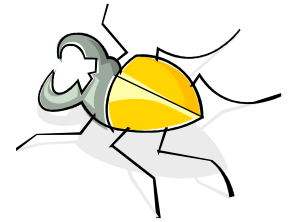
---

- Some characters are special in HTML
  - » `<`, `>`, `&`, `"`, `'`
- They are interpreted by the web browser
- To get them to display properly we have to encode them specially
  - » `&lt;` = `<`
  - » `&gt;` = `>`
  - » `&amp;` = `&`
  - » `&quot;` = `"`
  - » `&apos;` = `'` (note: Not in IE)

# Using Computers...

---

- In IT, stuff goes wrong ... debugging is the process of finding the error
  - » Term coined by Grace Murray Hopper
- Best solution ... make no mistakes!
  - » Be accurate ... get it right the 1<sup>st</sup> time
  - » Follow a process that makes it easier to get it right
  - » Computers can't make "common sense" decisions about what we really meant. They do what we say, not what we mean.



# Cost of Debugging

---

- Debugging may consume 60-70% of your development time
- 80% of overruns may be due to debugging
  - » Keep this in mind when you are budgeting time for your projects!



# Common Bug Types

---

- Compilation/Syntax errors
  - » Program doesn't won't run due to problems with what you typed in
    - HTML tags must be entered precisely
    - Required attributes must be present
- Logic errors
  - » Program runs, but output/behavior is wrong

# When You Debug...

Debugging is not algorithmic: no guaranteed process

- There are guidelines for debugging...

Rather than trying things aimlessly and becoming frustrated, think of yourself as solving a mystery



- Be objective: What are my clues? What is my hypothesis? Do I need more data?
- Consciously ‘watch’ yourself debug -- its an out-of-body experience
- When stumped, don’t become frustrated, but ask, “What am I misunderstanding?”

**Become Sherlock Holmes**

# Debugging Guidelines

---

- » Verify that the error is reproducible
- » Determine exactly what the actual failure is
- » Eliminate the “obvious” causes by checking
- » Divide process into working/faulty parts
- » On reaching a dead end, reassess the information you have, trying to identify the mistake you are making
- » Work through process making predictions and checking they’re fulfilled

# Reproducibility

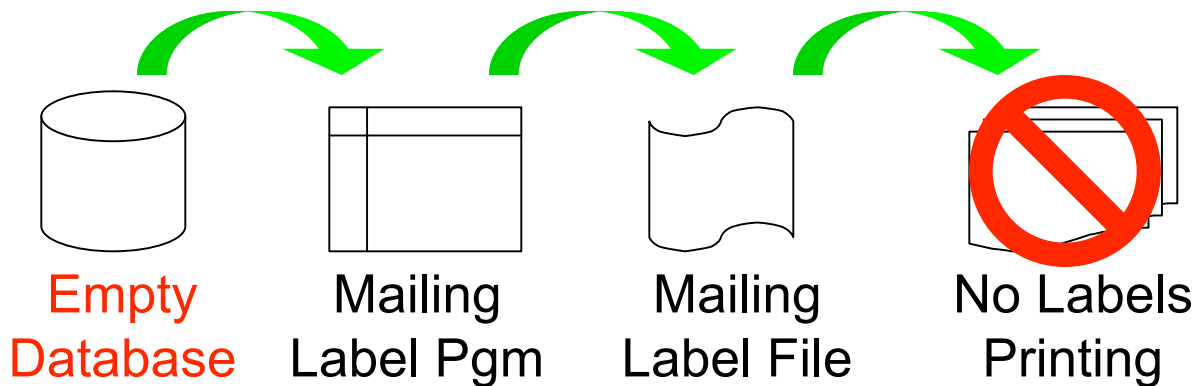
- First step: verify the error is reproducible
  - » You can't find something that you can't reproduce
  - » Get out and get back in. Does it still happen?
    - Restart the application.
    - Try a different application
    - Reboot the operating system. Sometimes this is appropriate, especially for errors involving peripheral devices (printers, modems)



Getting Out and Getting Back In

# Determine the Problem

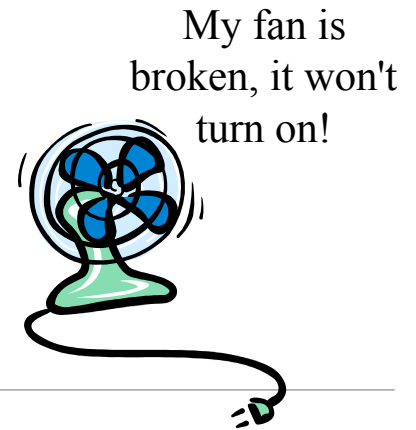
- Second step: figure out what's wrong
  - » Often there is a sequence of steps following an error and propagating it ... work backwards looking to see where the error first occurred



**This is not a printer problem!**

# Eliminate the Obvious

- Third step: eliminate obvious causes
  - “If the cause were obvious, the problem would have been fixed!” - Yeah, right.
  - » There are standard things to check:
    - Inputs
    - Connections
    - “Permissions”
    - Physical connectivity
    - Requirements



# Isolate the Problem

---

- Try to “partition” the situation into working and non-working parts
  - Form a hypothesis of what’s wrong
  - Make as few assumptions as possible
  - Take nothing for granted

The goal is to eliminate as many things from consideration as possible

## At a Dead End, Reassess

---

- When everything seems to check out, don't get frustrated
- Instead, ask yourself “What am I overlooking or misunderstanding?”
  - » Your goal is to see the situation as it is, not as you think it should be
    - Am I assuming too much?
    - Am I misreading the clues?
    - What can I eliminate or simplify?
- Explain the situation to a friend



# Make Predication/Check

---

- Beginning with the isolated part, step through the process, predicting the outcome and verifying it
  - » A prediction that is not fulfilled shows...
    - A possible bug
    - A possible misunderstanding
    - A chance to narrow the search

'Sleeping on it' often helps!

# Summary

---

- Debugging is not algorithmic, but there are guidelines to follow
  - » Stay calm - high blood pressure clouds your brain
  - » Be organized as you investigate and fix things
  - » Recognize that you may feel a little embarrassed when you finally figure out the problem.
    - If we were perfect, we would never make mistakes ...
    - A little humility is a good thing for all of us
  - » Watch yourself debug -- assess how you are doing, what you need to know
  - » Only try to fix one bug at a time!