When Trouble Comes:
The Basics of Debugging

Or as I often call it:
"What the (bleep!) did I just do?!?"

Nobody gets it right the first time. Part of being fluent is the ability to identify the problems of the program. Although debugging is very case-specific, there are some principles.

Bugs vs. Faults

- When the car doesn't start because of a dead battery, figuring out the problem uses debugging skills … however, finding the dead battery is not technically debugging – it's "fault identification".
  - When the error is a failing component of a correct design, it is a fault … when the battery is fixed, the car runs
  - When the error is a failure of the design, it is a bug
- When dealing with complex computer software and technologies, the chances are extremely high that the error is a bug
  - In other words, you've most likely made a reasoning error

To Debug is to Think Abstractly

- Debugging is a process that improves with practice.
- Helps you trace what is going wrong with the program at hand
- An effective way to proceed is to…
  - Think about what you know … the facts
  - Consider what should be true … the assumptions
  - Formulate a test hypothesis … gather evidence
  - Work intelligently … assess if you're making progress
- Think about how great it feels to find the problem that stumped everyone else!

Guidelines for Debugging

- There is no one sure way to debug. Every situation is different…but there are some guidelines you can follow

1. Make sure the error is reproducible – in other words, make it happen again
   - "Transient errors" can occur
   - The error may have been caused by a state or configuration that was unknowingly set .. Get a "clean" instance of the bug
   - When reproducing the error, try to work with or create a minimal version of the system or program with the bug
     - Copy a chunk of HTML and look at it by itself

Guidelines : Check the obvious!

2. Check for obvious problems
   - Make sure that what you entered is what is required
     - Are there substitution mistakes? O- 0 or 1- I or 1-l
   - If there are multiple components or files in the system with bugs, make sure they are properly connected
     - HTML files and the pictures/images that are referenced
   - Has anything been changed recently?
     - Or, do you just THINK you changed something?
   - When there are multiple inputs, does the order matter?
   - The chances are small that the problem is obvious – but always start with this as a process of elimination

Guidelines : Isolate the error

3. Isolate the problem – Most likely the error is in a specific place in the system/program, so sections that are "correct" should be removed from consideration
   - Isolating the problem to a specific procedure is best
     - Your HTML displays up to a point, then nothing – you know where you should start looking
   - Verifying that parts you think are correct really ARE correct is essential
     - Are you SURE you don't have to end a tag, or enclose a value in quotes?

Guidelines : Step through the process

4. Ok, you've isolated the error – now what? Reason through the process start-to-finish, predicting what should be computed and then verifying that is has been
   - If your prediction doesn't match an observation, then move inwards and further isolate the problem
     - The process was OK prior to this step
     - The process was incorrect after this step
   - Look at the inputs and reason through the step
   - If the bug isn't found, continue applying the guidelines

Guidelines : Assess Objectively

5. It often will happen that you check everything out and find it to be OK, but the bug is still there

   DON'T become frustrated!!!! Instead, evaluate your progress objectively
   - Are you making a wrong assumption
   - Are you misinterpreting the data input or output?
   - Have you made a wrong prediction/deduction?

FIT 100

Debugging Example:
Building an HTML Table

FIT 100

Tables in HTML

- The basic 2 x 2 table in HTML has the following scheme:

```
<TABLE>
   <TR>
        <TD>This is Row 1, Cell 1</TD>
        <TD>This is Row 1, Cell 2</TD>        Row 1 spec
   </TR>

   <TR>
        <TD>This is Row 2, Cell 1</TD>
        <TD>This is Row 2, Cell 2</TD>        Row 2 spec
   </TR>
</TABLE>
```

This is Row 1, Cell1	This is Row 1, Cell 2
This is Row 2, Cell1	This is Row 2, Cell 2

FIT 100

NBA Players Table: First attempt

nba.html

```
<TABLE WIDTH="80%" CELLPADDING="3" BORDER="2">
   <TR BGCOLOR="#33CCFF">
        <TD>Name</TD>
        <TD>Team</TD>
        <TD>Photo</TD></TR>

        <TD>Michael Jordan</TD>
        <TD>Chicago Bulls</TD>
        <TD<IMG SRC="jordan></TD>

        <TD>Larry Bird</TD>
        <TD>Boston Celtics</TD>
        <TD<IMG SRC="bird"></TD>

        <TR><TD>Dennis Rodman</TD>
        <TD>Chicago Bulls</TD>
        <TD align="center"<IMG SRC="worm jpg"></TD>
   </TR>
</TABLE>
```

FIT 100

Steps

- Is the bug reproducible? …reconstruct web page
- Check the “obvious” stuff … locate the NBA photos
- Isolate the problem … analyze the page –what’s wrong?
- Reason through the process
  - □ Think about what should be happening (what you should see)
  - □ Make predictions and check if they occur
- Assess your progress objectively (don’t freak out!!!!)
  - □ What do you need to know or find out?
  - □ Are there other things you can do?
  - □ Don’t get frustrated (I know it’s easy to do!)

Coming up…

- Read through Lab 6
  - □ There are readings AND helpful web pages in the lab
- For Friday Lecture
  - □ Read Chapter 9 in the FIT course pack
- And of course, for Friday….
  - □ Project 1, Part 1 due
  - □ Read the Project description for turn-in instructions
  - □ A suggestion:
    Don’t use your freebie on this one- you’ll need it later
- Office Hours all this week to help you out!!!!!!!
  - □ See web page