Designing for Error

People will make mistakes
+ Major catastrophes
  - Three Mile Island
  - Chernobyl
  - Plane crashes
+ Minor catastrophes
  - Accidentally cutting power, quitting applications, closing windows, deleting files.
  - Annoying, time-wasting, disrupt the goal-completion cycle, distract from task completion
  - All too often these problems are dismissed as “operator error.”
  - Many errors can be avoided.

How to design for human error
+ Prevent them to begin with.
  + Design an easy-to-use system
  + Keep it simple, just what is needed for the task.
    - Compare an unnecessarily complex word processor to an ATM - which is more likely to lead to errors?
    - Apply Norman’s guidelines.
    - Test with real people, real tasks.
  + Require confirmation for potentially destructive actions.
+ When possible, allow users to recover from destructive actions.
  - How are these last two built into interfaces?

Require confirmation for potentially destructive actions
+ Avoid errors in the first place with forcing functions

Convey the right level of severity
+ Don’t just tell the user that something is wrong, but also offer a suggestion as to how to fix it. Explain:
  - What happened?
  - Why did it happen?
  - What can I do about it?
  - (Apple Macintosh Human Interface Guidelines)
+ Be sure the alert is necessary.

Be sure the alert is necessary: Do not over-warn
+ Advertisements, tool tips, unsolicited help alerts, and other unnecessary extra alerts will lead to more errors.
+ Signal detection theory has demonstrated that the harder it is to distinguish a signal (a warning that should be heeded) from noise (routine computer output, including warnings to be ignored), the more likely a person is to miss a signal.
+ People’s responses in a signal detection task are affected by their perceptual sensitivity (d’) and their response bias (β).
Possible responses in a signal detection task

<table>
<thead>
<tr>
<th>User's response</th>
<th>Noise alone</th>
<th>Noise plus signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes, signal</td>
<td>False alarm (error)</td>
<td>Hit</td>
</tr>
<tr>
<td>No signal</td>
<td>Correct rejection</td>
<td>Miss (error)</td>
</tr>
</tbody>
</table>

Possible responses in a signal detection task

What the device presents

### Error Recovery
- **Undo and Redo**
  - But undo how many steps?
  - People are going to make mistakes.
- Build "forgiveness" into your systems.
  - "You can encourage people to explore your application by building in forgiveness. Forgiveness means that actions on the computer are generally reversible. People need to feel that they can try things without damaging the system."
  - (Apple Macintosh Human Interface Guidelines)
- Java Swing builds support for undo and redo support right into the class hierarchy.

### There is a risk of over-reliance on the error-correction capabilities of the system
- Airplane ground avoidance systems
- Anti-lock brakes
- Hard drives not backed up
- Spellcheckers

### Another way to design for error: Design for the extra-ordinary user
- In extra-ordinary circumstances, the ordinary user becomes the extra-ordinary user.
- The "normal" users often suffer from these impairments when...
  - In a crowded, noisy office.
  - Driving a car.
  - Doing too many things at once.
  - Under intense pressure to meet a deadline.

### Human error and system design
- Be suspicious when catastrophes are categorized as "operator error."
- Many errors can be avoided.
- Your users are human, have limited capabilities, and are going to make mistakes.
- Build your systems to avoid errors and irrecoverable disasters.