

# **IN4MATX 231:**

# **User Interface Design & Evaluation**

**Class 11:**  
**Inspection & Heuristics**

Daniel Epstein



# Today's goals

By the end of today, you should be able to...

- Describe why we use inspection-based methods
- Given Nielsen's heuristics, be able to:
  - Explain what each of them means
  - Apply them to identify usability failures in an interface
- Describe an effective heuristic evaluation process

# Improve your paper prototype



- Back in your original group, improve your paper prototype

# Inspection-based methods

- We have cut prototyping to its minimum
  - Sketches, storyboards, paper prototypes
  - Rapid exploration of potential ideas
- But we need evaluation to guide improvement
  - Can be relatively slow and expensive
  - Study participants can be scarce
  - Can waste participants on obvious problems

# Inspection-based methods

- Simulate study participants
  - Instead of actual participants, use inspection to quickly and cheaply identify likely problems
- Inspection methods are rational, not empirical

# Nielsen's 10 Heuristics

- Too few can be unhelpful, too many can be overwhelming
- Nielsen seeks to create a small set
  - Collects 249 usability problems
  - Collects 101 usability heuristics
  - Rates how well heuristics explain problems
  - Factor analysis to identify key heuristics

# Nielsen's 10 Heuristics

- Visibility of system status
- Match between system and the real world
- User control and freedom
- Consistency and standards
- Error prevention
- Recognition rather than recall
- Flexibility and efficiency of use
- Aesthetic and minimalist design
- Help recognize, diagnose, and recover from errors
- Help and documentation

# 1. Visibility

- Visibility of system status
  - The system should always **keep people informed** about what is going on, through appropriate **feedback** within reasonable time.
- Anytime a person is wondering what state the system is in, or the result of some action, this is a visibility violation.



## 2. Real World Match

- Match between system and the real world
  - The system should **speak a person's language**, with words, phrases, and concepts familiar to the person, rather than **system-oriented terms**. Follow real-world conventions, making information appear in a **natural and logical order**.
- Refers to word and language choice, mental model, metaphor

# 3. User in Control

- User control and freedom
  - People often choose system functions by mistake and will need a clearly marked “emergency exit” to **leave the unwanted state** without having to go through an extended dialog.
  - **Support undo and redo.**
- Not just for navigation exits, but for getting out of any situation or state

# 4. Consistency

- Consistency and standards
  - People should not have to wonder whether different words, situations, or actions **mean the same thing**. **Follow platform conventions**.
- Internal consistency is consistency throughout the same product. External consistency is consistency with other products in its class.

# 5. Error Prevention

- Error prevention
  - Even better than good error messages is a careful design which prevents a problem from occurring in the first place. Either **eliminate error-prone conditions** or check for them and **present people with a confirmation** option before they commit to the action.
- Try to commit errors and see how they are handled. Could they have been presented?

# 6. Recognition not Recall

- Recognition rather than recall
  - Minimize a person's memory load by making objects, actions, and options visible. A person should not have to remember information from one part of the dialog to another. Instructions for use of the system should be clearly visible or easily retrievable whenever appropriate.
- People should never carry a memory load
- Problems with affordances might go here
  - Hidden affordance: remember where to act

# 7. Flexibility and Efficiency

- Flexibility and efficiency of use
  - Accelerators, while unseen by novices, may often speed up the interaction for experts such that the system can cater to both the inexperienced and experienced use. Allow people to tailor frequent interactions.
- Concerns anywhere users have repetitive actions that must be done manually. Also concerns allowing multiple ways to do things.

# 8. Aesthetic Design

- Aesthetic and minimalist design
  - Dialogues should not contain information which is irrelevant or rarely needed. Every extra unit of information in a dialog competes with the relevant units of information and **diminishes their relative visibility**.
- Not just about “ugliness”. About clutter, overload of visual field, visual noise, distracting animations

# 9. Error Recovery

- Help users recognize, diagnose, and recover from errors
  - Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution.
- Error prevention is about preventing errors before they occur. This is about after they occur.



# 10. Help

- Help and documentation
  - Even though it is better if the system can be used without documentation, it may be necessary to **provide help and documentation**. Any such information should be **easy to search, focused on a person's task, list concrete steps** to be carried out, and not be too large.
- This does not mean that a person must be able to ask for help on every single item.

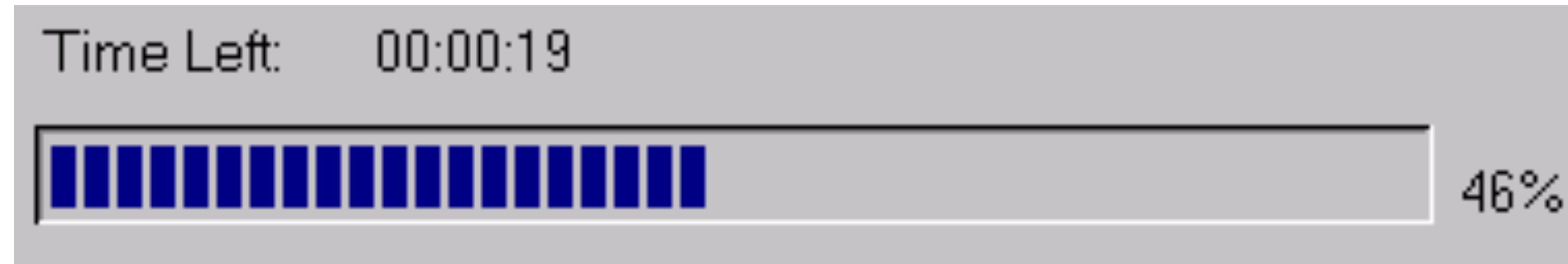
# Heuristic evaluation process

- Evaluators go through interface several times
  - Inspect various dialog events
  - Compare with list of usability principles
- Usability principles
  - Nielsen's heuristics
  - Supplemental list of category-specific heuristics (competitive analysis of testing existing products)
- Use violations to redesign/fix problems

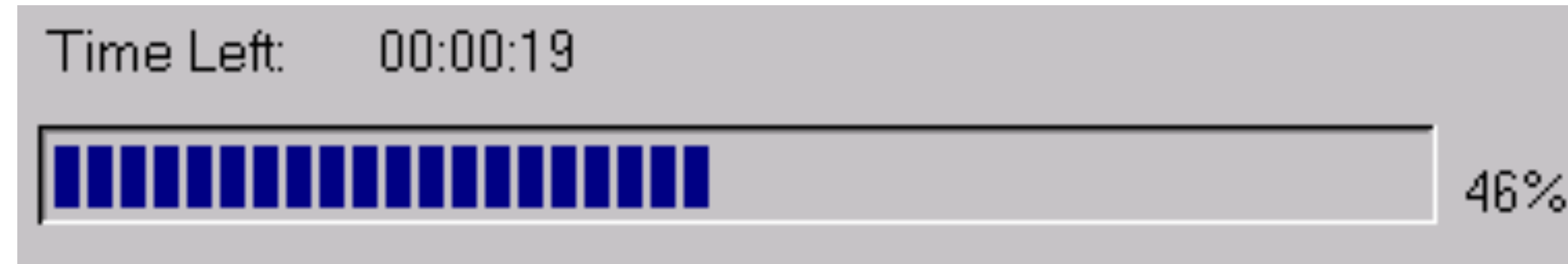
# Examples

- Can't copy information from one window to another
  - Violates “Minimize memory load” (H6)
  - Fix: allow copying
- Typography uses different fonts in 3 dialog boxes
  - Violates “Consistency and standards” (H4)
  - Slows users down, probably won't be found by usability testing
  - Fix: pick a single format for the entire interface

# Heuristics



# Heuristics



- Visibility of system status
  - Pay attention to response time
  - 0.1 sec: no special indicators needed (why?)
  - 1.0 sec: person tends to lose track of data
  - 10 sec: maximum duration if person to stay focused  
Longer delays require progress bars

# Heuristics

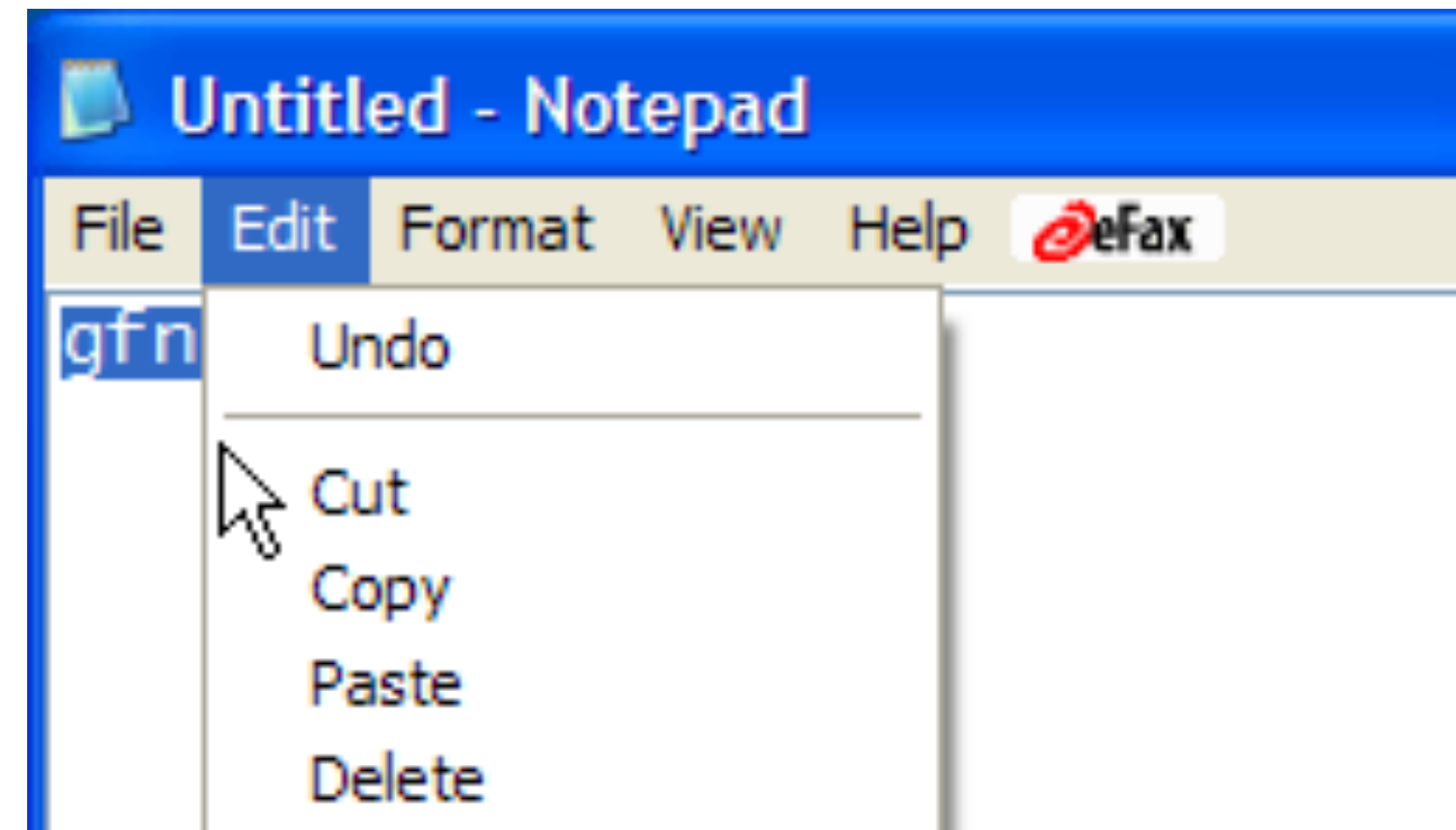


# Heuristics



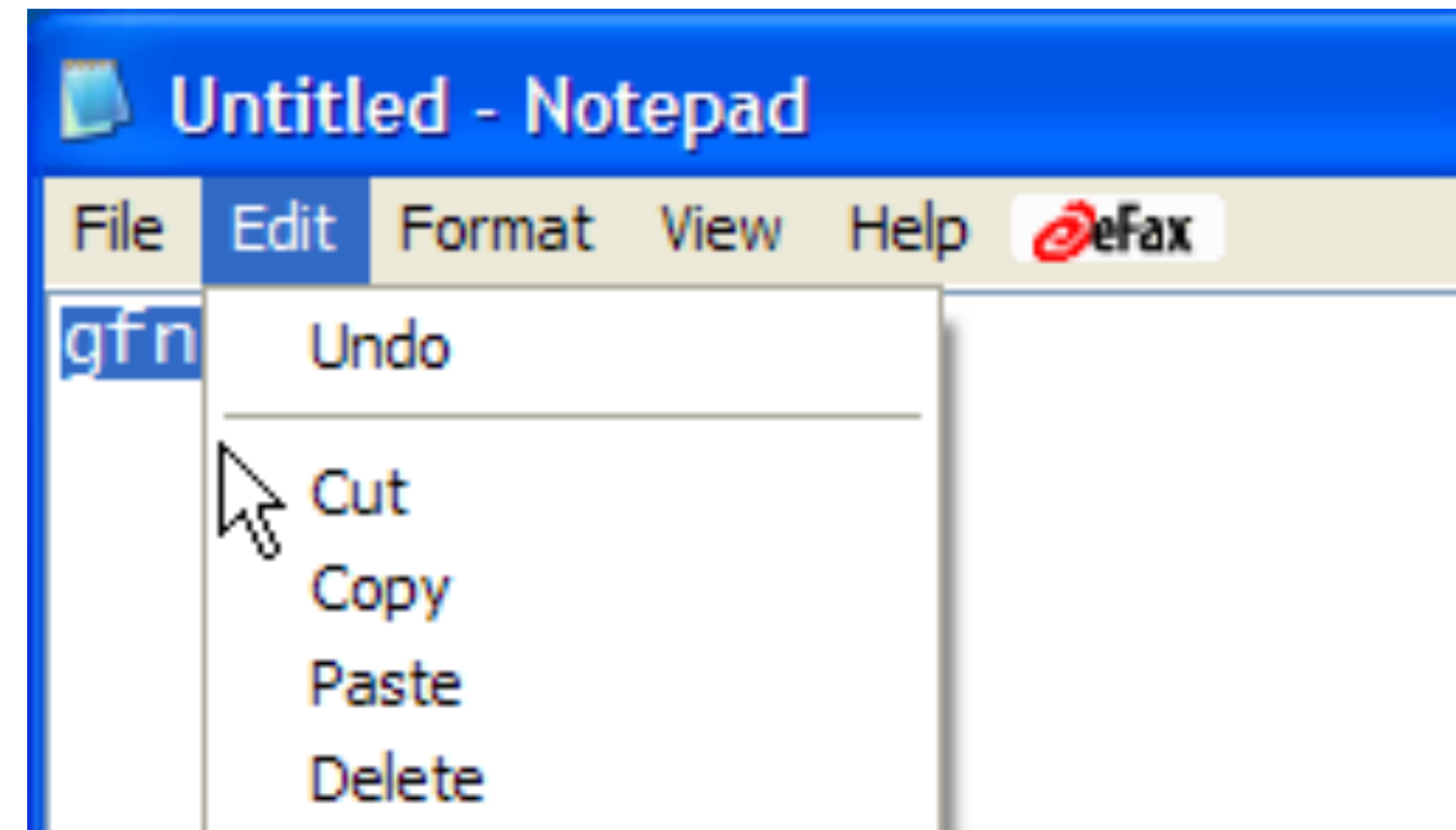
- Mailto? protocol?
- Match system to real world
- Speak the person's language

# Heuristics





# Heuristics



- Flexibility and efficiency of use
  - Accelerators for experts (e.g., keyboard shortcuts)
  - Allow tailoring of frequent actions (e.g., macros)

# How to perform heuristic evaluation

- At least two passes for each evaluator
  - First to get feel for flow and scope of system
  - Second to focus on specific elements
- If system is walk-up-and-use or evaluators are domain experts, no assistance needed
  - Otherwise might supply evaluators with scenarios
- Each evaluator produces list of problems
  - Explain why with reference to heuristic
  - Be specific & list each problem separately

# Example heuristic violation

- 1. [H4 consistency] [Severity 3] [Fix 4]
- The interface used the string “Save” on the first screen for saving the person’s file, but used the string “Write File” on the second screen. People may be confused by this different terminology for the same function.
- Fix: Change the second screen to “Save”.

# Phases of heuristic evaluation

- Pre-evaluation and training
  - Give expert evaluators needed domain knowledge & information on the scenario
- Evaluation
  - Individuals evaluate interface and make lists of problems
- Severity rating
  - Determine how severe each problem is
- Aggregation
  - Group meets and aggregates problems (w/ ratings)
- Debriefing
  - Discuss the outcome with the design team

# Performing a Heuristic Evaluation



- Follow Nielsen's heuristics to evaluate your paper prototype

# Cognitive Walkthrough

- Evaluation method based on:
  - A person works through an interface in an exploratory manner
  - A person has goals
  - The person is applying means-ends reasoning to work out how to accomplish these goals
- Evaluation by an expert, who goes through a task while simulating this cognitive process

# Preparation: need four things

- Person description, including level of experience and any assumptions made by the designer
- System description (e.g., paper prototype)
- Task description, specifying the task the expert has to carry out, from a person's point of view
- Action sequence describing the system display and the actions needed to complete the task. One system display and one action together are one step.

# Cognitive walkthrough process

- Designer/developer prepare the required documents described
- Give these documents to the usability expert
- Expert reads the descriptions, carries out the task by following the action list
- At each step in the action list, asks four question
- Record problems similar to heuristic evaluation



# Believability

- Will the person be trying to produce whatever effect the action has?
- Will the person be able to notice that the correct action is available?
- Once the person finds the correct action at the interface, will they know that it is the right one for the effect they are trying to produce?
- After the action is taken, will the person understand the feedback given?

# Inspection vs. Usability Testing

- Inspection
  - Is much faster
  - Does not require interpreting participant actions
  - May miss problems or find false positives
- Usability testing
  - More accurate, by definition
  - Account for actual people and tasks

# Inspection vs. Usability Testing

- One approach is to alternate between them
  - Find different problems, conserve participants

# Today's goals

By the end of today, you should be able to...

- Describe why we use inspection-based methods
- Given Nielsen's heuristics, be able to:
  - Explain what each of them means
  - Apply them to identify usability failures in an interface
- Describe an effective heuristic evaluation process

# **IN4MATX 231:**

# **User Interface Design & Evaluation**

**Class 10:**  
**Paper Prototyping &**  
**Usability Evaluation**

Daniel Epstein