End User Programming

- Customization of application performance by end users: regular (typically “non programmer”) users
- Emphasis on application behavior that is conventionally accomplished by programming new or modified “commands”
End User Programming Approaches

- Preferences
  - User chooses among alternative behaviors
  - Choices are limited and predefined
  - Often used for defining layout (e.g., windows in an IDE)
- Scripting languages
  - User writes code in special-purpose language customized for the application (e.g., shell script, elisp)
  - User needs to program

End User Programming Approaches

- Macro recorders
  - User turns on macro recording, interacts with the application, and stops recording
  - System records interactions and makes it possible to reexecute them (e.g., GNU Emacs macros, Microsoft Visual Basic macros)
  - Too literal: Use exact positions, keystrokes
  - Generalizing or fixing macros requires “real” programming
- Programming by demonstration
  - Generalizes macro recorders
  - www.acypher.com/wwid

“Before you begin a VBA project, ensure that you have the time to work with VBA. Programming requires focus and can be unpredictable. Especially as a beginner, never turn to programming unless you have time to work carefully. Trying to write a “quick script” to solve a problem when a deadline looms can result in a very stressful situation. If you are in a rush, you might want to use conventional methods, even if they are monotonous and repetitive.”

—Getting Started with VBA in Office 2010
Programming by Demonstration Terminology

- **PBE**: Programming by example (AKA Example-based programming)
  - User provides one or more concrete examples of the behavior or effect of a more general program
  - System infers intent
- **PBD**: Programming by demonstration—D.C. Smith (AKA Demonstrational programming—B. Myers)
  - PBE in which the user demonstrates actions on example data
- **PITUI**: Programming in the User Interface—D. Halbert
  - PBD, emphasizing the use of existing UI commands

- Inferencing
  - Determining appropriate generalizations from examples

Pygmalion  
**D.C. Smith, 1975**

- Introduced icons to computer UIs
- First PBD system
- Designed to work as “blackboard”
- *Programmer* demonstrates algorithm with an example
- No inferencing
- Many relationships not visible
SmallStar 84  D. Halbert

- Reimplementation of Xerox Star subset to support PBD of desktop operations
- Approach
  - User performs a specific set of tasks
  - System records tasks to create a program (at this point, a macro)
  - System determines data description of any object selected by user (picks one arbitrarily if ambiguous), reuses it throughout program (i.e., no inferencing)
  - User can view data description of any object with “properties”
  - User must edit program to
    - Modify data descriptions, including generalization
    - Create flow of control

(Xerox Star icons created by DC Smith)