

Advanced Application Development

Final Project

Integrate everything you've done this semester into a masterpiece of Windows Application development. Write a fully-functional, extremely well-tested Adventure Game Engine.

Background

Consider classic text adventure games like Adventure, Zork (<http://en.wikipedia.org/wiki/Zork>), and The Legend of Zelda without the pictures. There is one player navigating through a map/dungeon/castle/whatever. Each location in the game is described in text to the player along with any items that may be found there and any gateways to other locations. The player can try to take any items present in any location. (Not all items need to be "takable".) The player can exit from any available gateway if the exit conditions are met. (Exit conditions in our game are simply a list of items the player must be carrying to use the gateway.) Our game is over when the player gets to a certain location carrying certain items.

Goal

Rather than hard-code **your** particular game, develop a game engine that reads individual game specifications from an XML document and runs that game. In this way your game engine can run an infinite number of these types of games so long as they are expressed in the proper XML format.

Details

Write a program for the .Net environment that reads an XML document of the format specified on our web site and "runs" the game described in that document for a single player. The player has two commands: look and take.

- Look – Describe to the player the location, its contents, the items present.
- Take – Prompt the player to take an item from a location if that item is takable.
- Gateways – List all the gateways for each location and allow the player to select one. Before changing the player's location to wherever that gateway leads, check that the player is holding the items needed to use that gateway. If so, move them to the destination location and display the gateway success message. If not, keep them there and display the failure message.

For further details, see the XML example. Additional details may follow.

Deliverables

- You will present your project to your classmates and me at our class meeting during final exam week. Be prepared to show all aspects of your project.
- E-mail me your XML, source code, and executable.
- Develop a class diagram documenting your design. (Visual Studio 2005 has a nice facility for this. You can also use Visio or some other tool, including drawing it by hand.)
- Annotate your printed source code and turn that in as well. Annotate? Huh? This means taking a pen (feel free to use red if you like; it's fun) and writing notes on your printout pointing all "good" (and, if necessary, "bad") elements of programming style and practice. Use the checklists throughout our text book (Code Complete, 2nd edition) for inspiration. Be very thorough!
- Add a cover sheet and staples to complete the package.

Advanced Application Development

Evaluation Criteria

When evaluating your project, I will ask myself the following questions about your program:

- Is it correct; i.e., free from faults in its specification, design, and implementation?
- Is it usable?
- About the interface...
 - Is it “clean” and well-organized? Would Mr. Monk be proud?
 - Does it obey the laws of least astonishment?
 - How accurate is it?
 - Is it easy to use?
 - Is it reliable and robust; i.e., able to perform its functions without breaking down, even given unexpected data or other circumstances?
- About the readability and maintainability of the source code...
 - Are the comments plentiful, clear, meaningful, and helpful?
 - Are the identifier names accurate, clear, and meaningful?
 - How is its object-oriented architecture?
 - Does it compile cleanly?
- About the Design...
 - Is the solution well-designed?
 - Is it re-usable?
 - Does it illustrate the points made or principles discussed in class?
 - Have any “lazy shortcuts” been taken?
 - Can the program be unit and system tested?
- About the results, are they accurate; i.e., are the qualitative outputs free of error?
 - Does it perform as assigned?
 - Is the output well-formatted and meaningful?
- Overall, does it reflect all the time we spent this semester on good programming practice, development techniques, and code strategies? Would Steve McConnell be proud? Most importantly, would **I** be proud of your code?