Dylan Suzuki, Jackson Muller, Aidan Hodges, Isaiah Pajarillo
BJ Johnson
CMSI 4072
2/11/26

Homework 2

Problem 5.1, Stephens page 116

What's the difference between a component-based architecture and a service-oriented architecture?

A component-based architecture is a collection of loosely used components that provide services for each other. Meanwhile, a service-oriented architecture is similar to component-based architecture, however the pieces are implemented as services, where a service is a program that runs on its own and provides some service for its clients.

Problem 5.2, Stephens page 116

Suppose you're building a phone application that lets you play tic-tac-toe against a simple computer opponent. It will display high scores stored on the phone, not in an external database. Which architectures would be most appropriate and why?

A simple tic-tac-toe phone app that lets you play against a simple computer opponent, would be best suited by a monolithic architecture, combined with an event-driven user interface. The event-driven user interface is critical for the UI responding to the users' taps. Since the scores are stored on the phone, there's no need for client/server, no need for distributed (since it's on one phone), and the app is simply too small for service-oriented or component based applications. This means that the high scores can be stored locally on the phone, which fits the monolithic architecture.

Problem 5.4, Stephens page 116

Repeat question 3 [after thinking about it; it repeats question 2 for a chess game] assuming the chess program lets two users play against each other over an Internet connection.

A chess program that lets two users play against each other over an Internet connection is best suited by a client/server architecture, as well as an event-driven user interface. Each player runs a client that displays the board, while a server actually manages the game state and relays the player's moves over the internet. The event-driven user interface will allow the players to move

their pieces, which is critical for the UI in a chess game. A monolithic architecture doesn't work, since the program must operate across two devices on a network. Service-oriented and component-based architectures are designed for managing large systems, so these architectures would be overkill for a simple two-player chess game.

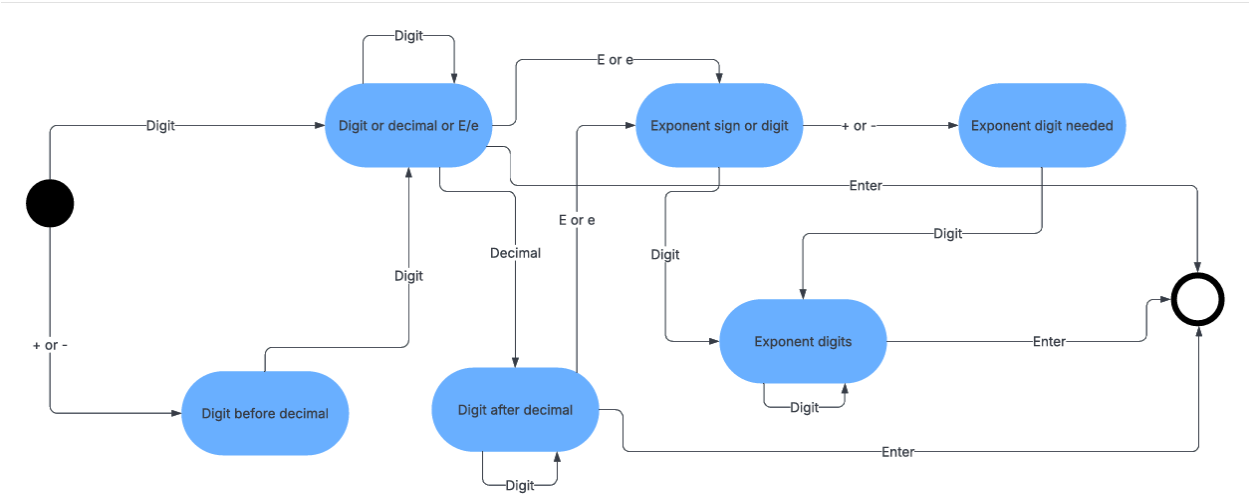
Problem 5.6, Stephens page 116

What kind of database structure and maintenance should the ClassyDraw application use?

ClassyDraw doesn't need a relational database. Instead, ClassyDraw should use a simple file-based system, storing drawings in text files or XML files. Each drawing can be stored in its own file, which contains the objects' properties such as position, size, and color. Since the drawings don't share data with other drawings, there simply is no need for a full database. In terms of maintenance, the ClassyDraw application should implement basic file operations, such as "Save", "Save-as", and "Open". Additionally, it would be nice to implement an auto-save feature into ClassyDraw, as there's nothing worse than losing an online drawing due to uncontrollable circumstances (laptop dies, computer crashes, etc).

Problem 5.8, Stephens page 116

Draw a state machine diagram to let a program read floating point numbers in scientific notation as in +37 or -12.3e+17 (which means -12.3 x 1017). Allow both E and e for the exponent symbol. [Jeez, is this like Dr. Dorin's DFAs, or what???]



Problem 6.1, Stephens page 138

Consider the ClassyDraw classes Line, Rectangle, Ellipse, Star, and Text.

1. What properties do these classes all share?
    a. Drawing objects that can be rendered
    b. They can be selected
    c. They can be moved
    d. They can be deleted
    e. They can be saved and loaded
    f. They have common state such as selected and visible
2. What properties do they NOT share?
    a. They do not share geometry representations
        i. Line uses endpoints
        ii. rectangle/ellipses use bounding boxes
        iii. Star uses bounding box and point/ratio
        iv. Text uses anchor point plus string and font
3. Are there any properties shared by some classes and not others?
    a. Rectangles, ellipse, star share FillStyle
    b. Line, rectangle, ellipse, star share stroke style
    c. Text has textstyle
    d. Closed shapes share interior hit-testing
4. Where should the shared and nonshared properties be implemented?
    a. Put universal behavior in an abstract base class
    b. Put partially shared features in subclasses
    c. Keep geometry specific data inside each concrete class
    d. Use composition for styles so only relevant classes include them

Problem 6.2, Stephens page 138

Draw an inheritance diagram showing the properties you identified for Exercise 6.1. [Create parent classes as needed, and don't forget the Drawable class at the top.]