Breaking a Problem into Classes

CSCI 2300

Problem Statement

- I want an electronic catalog of all the books I have. As the first step, I want a program that I can use to add basic book information. I also want to be able to list the books I have entered so far, and search my "catalog" by author or by title.
- Break the problem statement into objects
  - Look for nouns
- Identify actions
  - Look for verbs
- Determine which actions/behaviors belong to which object

I want an electronic catalog of all the books I have. As the first step, I want a program that I can use to add basic book information. I also want to be able to list the books I have entered so far, and search my "catalog" by author or by title.

- Objects (nouns)

Catalog

Book

Author

Title

- Actions (verbs)

Add

List

Search

Which object is responsible for each action?

Combine objects

- Which objects are "attributes" of other objects?

- Catalog has Book(s)
- Book has Title
- Book has Author

Putting it all together – define interfaces

```
Book
String title;
Author author;
Book(String t, Author a)
```

```
Author
String firstName
String lastName
Author(String f, String l)
```

```
Catalog
ArrayList<Book> books
Book[] list()
void add(Book b)
Book[] searchByTitle(String title)
Book[] searchByAuthor(String name)
```

Interface of a class the list of its methods and variables

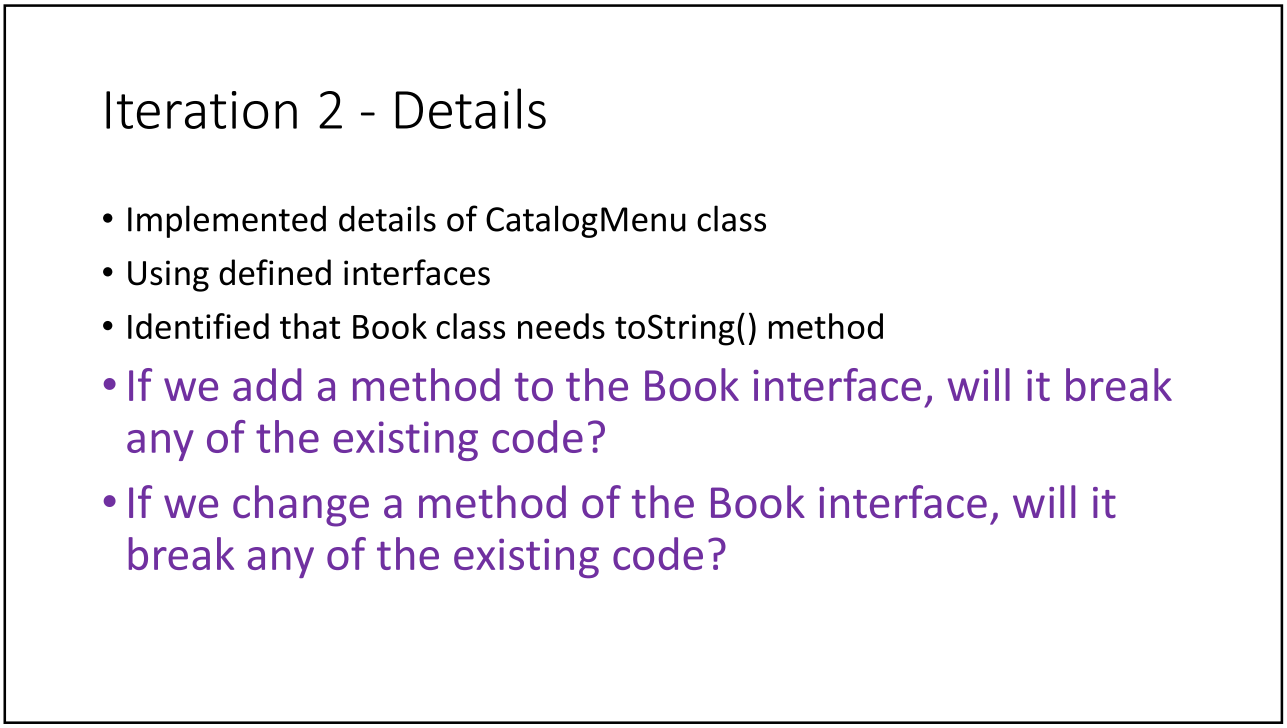
Sketch of the solution using defined interfaces

- User needs ability to
  - Enter a book
  - Search by author
  - Search by title
- The main() method:
  - Displays a basic "menu"
  - Reads input from user
  - Performs actions based on input

Iterative solutions

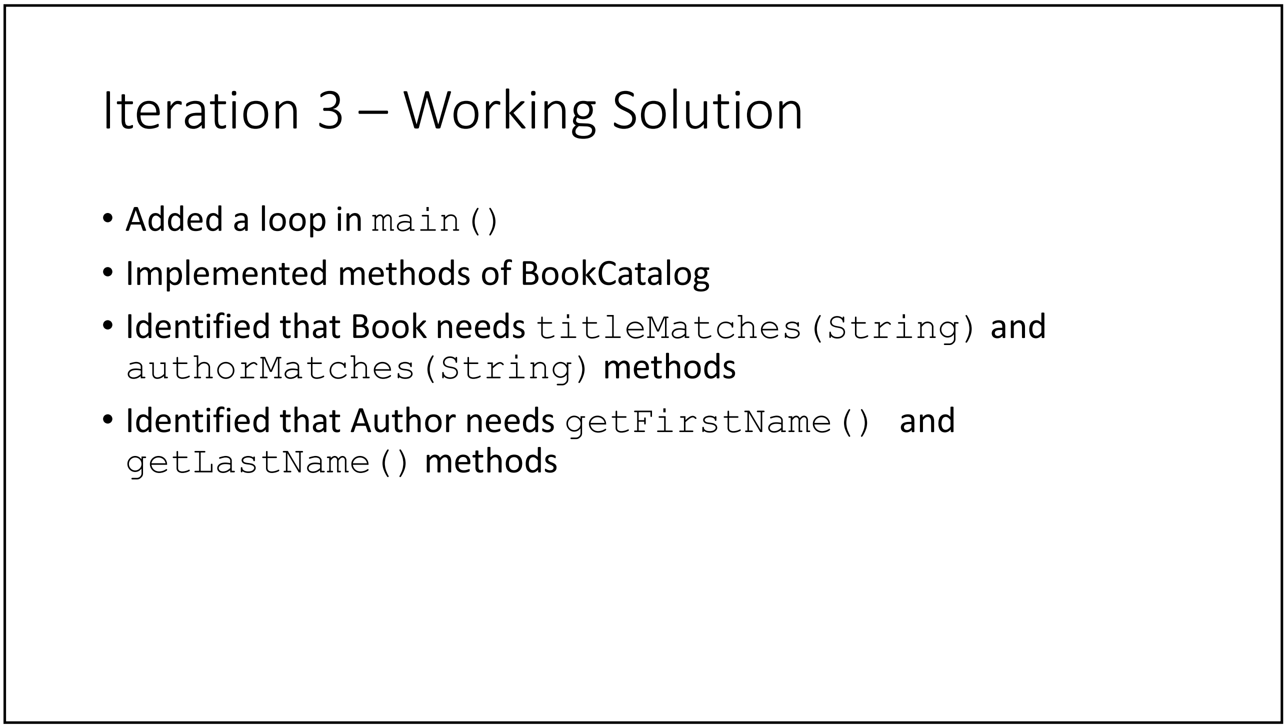
Iteration 1 - Sketch

- Defined class interfaces
- Basic structure of the main() method, with implementation for:
  - Printing menu
  - Getting user's selection
  - Reacting to user's selection
  - Fully implemented 'add' functionality
- In BookCatalog
  - `searchByTitle()` and `searchByAuthor()` are not yet fully implemented
  - These methods are used by CatalogMenu
  - **We don't need to know implementation details in order to use these methods!**
  - What do we need to know in order to use these methods?



Iteration 2 - Details

- Implemented details of CatalogMenu class
- Using defined interfaces
- Identified that Book class needs toString() method
- If we add a method to the Book interface, will it break any of the existing code?
- If we change a method of the Book interface, will it break any of the existing code?



Iteration 3 – Working Solution

- Added a loop in `main()`
- Implemented methods of BookCatalog
- Identified that Book needs `titleMatches(String)` and `authorMatches(String)` methods
- Identified that Author needs `getFirstName()` and `getLastName()` methods

Iteration 4 – Improved Solution

- Added feature: load books from CSV
- Modified main menu
- Added methods to BookCatalog class
    - `public void fromCSV(String)`
    - `private void fromScanner(Scanner)`
    - `fromCSV() uses fromScanner()`

Summary

- Start with a high level break down of objects and actions
- Design class interfaces
- Put together `main()` structure
- Implement the methods used by `main()`
- Implement interfaces
- Add features

Your CSCI 2300 git repos are ready

Submit your labs to csci2300 git repos by Friday, Jan 25

Only submit .java files (do not submit .class)

Your ‘git’ repositories

- Everyone has an account on git.cs.slu.edu
- You have a repository for Spring 2019 CSCI 2300
- Log in to git.cs.slu.edu using your git credentials
- Open a linux terminal and create a directory where you’ll store you git repositories
- Clone your git repository:
  git clone git@git.cs.slu.edu:courses/spring19/csci_2300/<LOGIN>.git
  Example:
  git clone git@git.cs.slu.edu:courses/spring19/csci_2300/aguayoj.git