Unit 04

Composition

CMPS 251, Fall 2020, Dr. Abdulaziz Al-Ali

Composition

- Sometimes you want an attribute of one class to be an object
- Embedding one class inside another is called composition

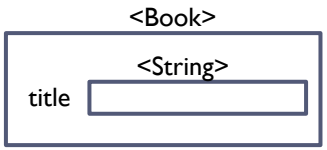
- We call this the *has a* relationship

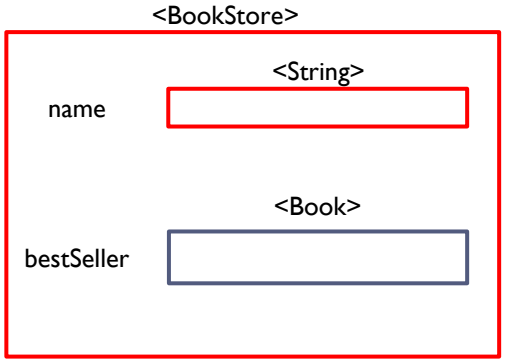
- Examples:
  - Car has an Engine
  - LectureHall has a Projector
  - BookStore has a Book (or books)

Object References and Composition

- Let's look at how composed objects are stored in memory

Composition Class Definition






```
public class Book {
        private String title;

        public Book(String title) {
                this.title = title;
        }

        // getters and setters
        public String getTitle() {
                return title;
        }
        public void setTitle(String title) {
                this.title = title;
        }
}
```

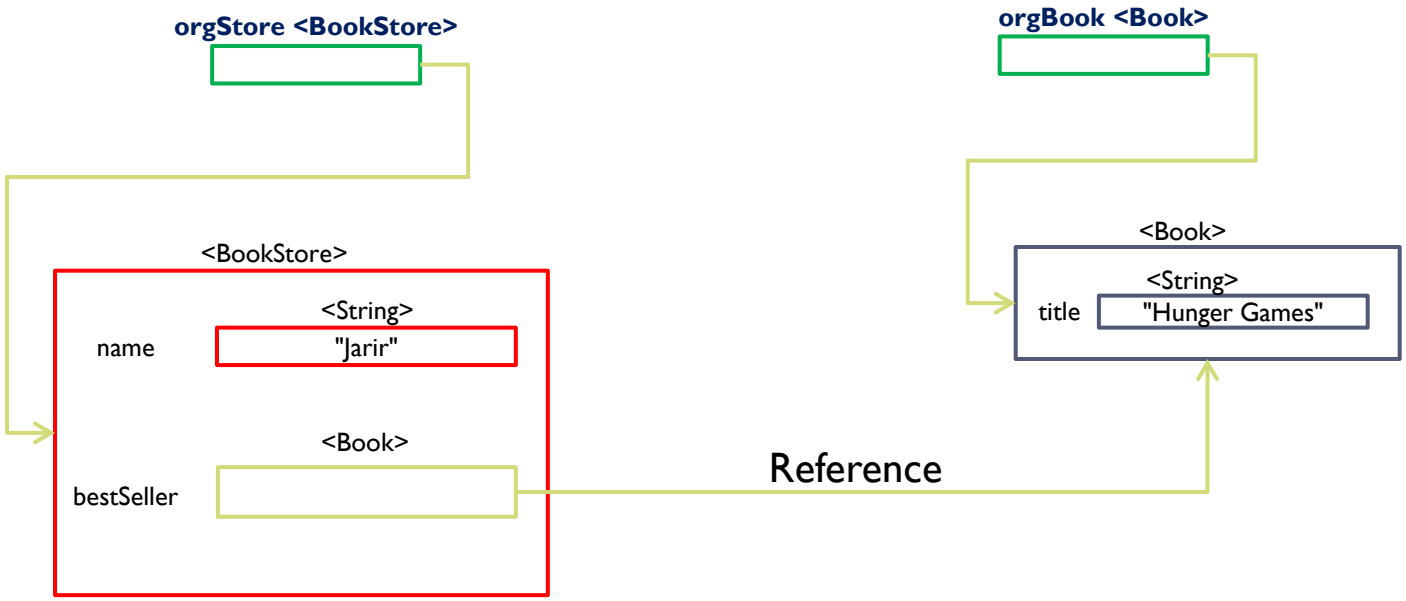
Composition!

```
public class BookStore {
        String name;
        private Book bestSeller;

        public BookStore(String name, Book bestSeller) {
                this.name = name;
                this.bestSeller = bestSeller;
        }
}
```

Referencing After Instantiation

```
public class App {
    public App() {
        Book orgBook = new Book("Hunger Games");
        BookStore orgStore = new BookStore("Jarir", orgBook);
    }
}
```



Objects and References

- Once a class is defined, you can declare variables (object reference) of that type

```
Book book1, book2;
BookStore store1;
Author author1;
```

- Object references are initially **`null`**
  - The **`null`** is a special value in Java indicating that the object is NOT created yet

- The ***`new`*** operator is required to create the object

```
ClassName variableName = new ClassName();
```

Check point

- What is composition?

- How can we tell by looking at a given class code that it has composition?

Demo

- See the *courses* package in unit 4 sample code.
  - See TODO items 1 to 5

Check point

- If each Course has an Instructor, and each Instructor has a Car. How many times do we need to call **new** to create a Course?

- Can you make the new Mall below but in one line?

```
Store s2 = new Store("Qatar optics");
Mall m2 = new Mall("Gulf mall", s2);
```

- If each Mall has a Store (bestStore), and each Store has an Employee (bestEmployee), and each Employee has a *name*. How can you print the employee's name of the mall m2 above?

Comparing objects (using ==)

```
public class App {
    public App() {
        Book orgBook = new Book("Hunger Games");
        BookStore orgStore = new BookStore("Jarir", orgBook);

        Book copybook = new Book(“Hunger Games”);
        BookStore copyStore = new BookStore(“Jarir”, copybook);

        if (orgStore == copyStore)
        {
                System.out.println(“They have the same values”);
        }
    }
}
```

Will this line work?

Demo

- See the *bookstores* package in unit 4 sample code.
  - See TODO items 6 to 8

The solution to the == problem?

Define an `equals` method

```
public class Book {
…..
    public boolean equals(Book that) {

        return this.title.equals(that.title);

    }
}
```

Two Books are equal if their *title* values are the same.

Code Deconstructed
<equals method>

```
Book book1 = new Book(“Alf Laila wa Laila”);

Book book2 = new Book(“Alf Laila wa Laila”);


if (book1 == book2)
   System.out.println("Both variables refer to the same object");
else
   System.out.println("Each variable refers to a different object");


if ( book1.equals(book2) )
   System.out.println("Both objects have the same value");
else
   System.out.println("Each object has a different value");
```

Can we use book2.equals(book1) instead?

Demo

- See the *bookstores* package in unit 4 sample code.
  - See TODO items 9 to 13

Question

- What if we want to compare BookStores?

```
public class BookStore {
…..
    public boolean equals(BookStore that) {


       return   this.name.equals(that.name) &&
                this.bestseller.equals(that.bestSeller);

    }
}
```

Question

- What if we want to store 100 Books in the BookStore?

Using arrays in composition

```
public class BookStore {
    public String name;
    public Book bestSeller;

    public Book[] books = {
        new Book("The Kite Runner"),
        new Book("Kalila Wa Domna"),
        new Book("A Thousand Night and a Night")
        };

    public BookStore() {
    }

    public BookStore(String name, Book bestSeller) {
        this.name = name;
        this.bestSeller = bestSeller;
    }

}
```

Use an array!

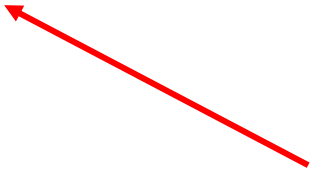
Demo

- See the *bookstores* package in unit 4 sample code.
  - See TODO items 14 to 16

Example: Print all books of a BookStore

```
public class App {
        public App() {
                Book orgBook = new Book("Hunger Games");
                BookStore orgStore = new BookStore("Jarir", orgBook);

                // print all books of orgStore:
                for (int i = 0 ; i < 3 ; i++)
                    System.out.println( orgStore.books[i].getTitle() );
        }
}
```



What if the array *books* is private?