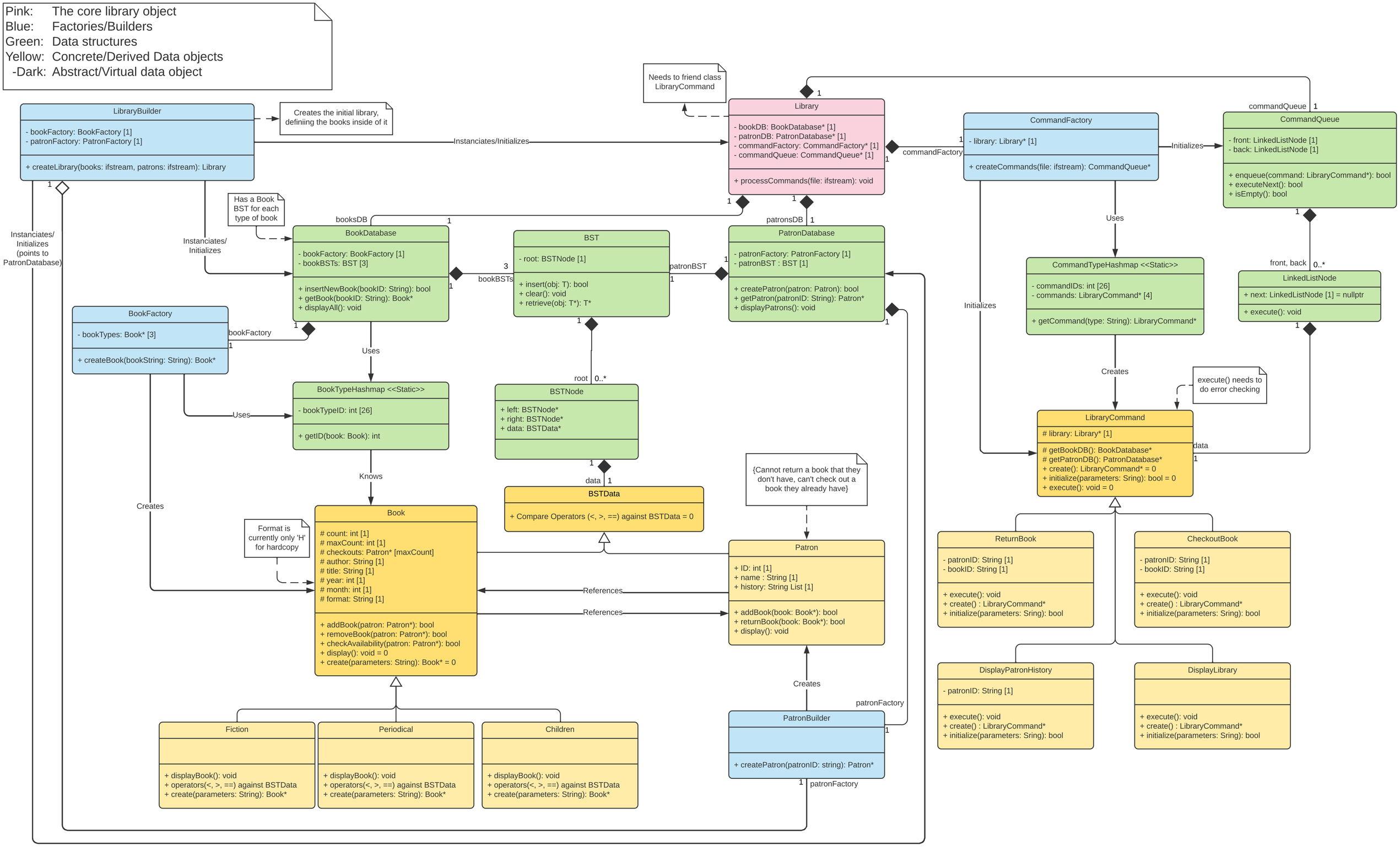
Pink: The core library object
Blue: Factories/Builders
Green: Data structures
Yellow: Concrete/Derived Data objects
-Dark: Abstract/Virtual data object
LibraryBuilder
- bookFactory: BookFactory [1]
- patronFactory: PatronFactory [1]
+ createLibrary(books: ifstream, patrons: ifstream): Library
Creates the initial library, definiing the books inside of it
Instanciates/Initializes
Needs to friend class LibraryCommand
Library
- bookDB: BookDatabase* [1]
- patronDB: PatronDatabase* [1]
- commandFactory: CommandFactory* [1]
- commandQueue: CommandQueue* [1]
+ processCommands(file: ifstream): void
commandFactory
CommandFactory
- library: Library* [1]
+ createCommands(file: ifstream): CommandQueue*
Initializes
commandQueue
CommandQueue
- front: LinkedListNode [1]
- back: LinkedListNode [1]
+ enqueue(command: LibraryCommand*): bool
+ executeNext(): bool
+ isEmpty(): bool
Uses
CommandTypeHashmap <<Static>>
- commandIDs: int [26]
- commands: LibraryCommand* [4]
+ getCommand(type: String): LibraryCommand*
front, back
0..*
LinkedListNode
+ next: LinkedListNode [1] = nullptr
+ execute(): void
Initializes
Creates
execute() needs to do error checking
LibraryCommand
library: Library* [1]
getBookDB(): BookDatabase*
getPatronDB(): PatronDatabase*
+ create(): LibraryCommand* = 0
+ initialize(parameters: Sring): bool = 0
+ execute(): void = 0
data
ReturnBook
- patronID: String [1]
- bookID: String [1]
+ execute(): void
+ create() : LibraryCommand*
+ initialize(parameters: Sring): bool
CheckoutBook
- patronID: String [1]
- bookID: String [1]
+ execute(): void
+ create() : LibraryCommand*
+ initialize(parameters: Sring): bool
DisplayPatronHistory
- patronID: String [1]
+ execute(): void
+ create() : LibraryCommand*
+ initialize(parameters: Sring): bool
DisplayLibrary
+ execute(): void
+ create() : LibraryCommand*
+ initialize(parameters: Sring): bool
Instanciates/ Initializes (points to PatronDatabase)
Instanciates/ Initializes
Has a Book BST for each type of book
booksDB
BookDatabase
- bookFactory: BookFactory [1]
- bookBSTs: BST [3]
+ insertNewBook(bookID: String): bool
+ getBook(bookID: String): Book*
+ displayAll(): void
bookBSTs
BST
- root: BSTNode [1]
+ insert(obj: T): bool
+ clear(): void
+ retrieve(obj: T*): T*
patronBST
patronsDB
PatronDatabase
- patronFactory: PatronFactory [1]
- patronBST : BST [1]
+ createPatron(patron: Patron): bool
+ getPatron(patronID: String): Patron*
+ displayPatrons(): void
BookFactory
- bookTypes: Book* [3]
+ createBook(bookString: String): Book*
bookFactory
Uses
BookTypeHashmap <<Static>>
- bookTypeID: int [26]
+ getID(book: Book): int
Uses
root
0..*
BSTNode
+ left: BSTNode*
+ right: BSTNode*
+ data: BSTData*
data
BSTData
+ Compare Operators (<, >, ==) against BSTData = 0
Knows
Creates
Format is currently only 'H' for hardcopy
Book
count: int [1]
maxCount: int [1]
checkouts: Patron* [maxCount]
author: String [1]
title: String [1]
year: int [1]
month: int [1]
format: String [1]
+ addBook(patron: Patron*): bool
+ removeBook(patron: Patron*): bool
+ checkAvailability(patron: Patron*): bool
+ display(): void = 0
+ create(parameters: String): Book* = 0
{Cannot return a book that they don't have, can't check out a book they already have}
Patron
+ ID: int [1]
+ name : String [1]
+ history: String List [1]
+ addBook(book: Book*): bool
+ returnBook(book: Book*): bool
+ display(): void
References
References
Creates
patronFactory
PatronBuilder
+ createPatron(patronID: string): Patron*
patronFactory
Fiction
+ displayBook(): void
+ operators(<, >, ==) against BSTData
+ create(parameters: String): Book*
Periodical
+ displayBook(): void
+ operators(<, >, ==) against BSTData
+ create(parameters: String): Book*
Children
+ displayBook(): void
+ operators(<, >, ==) against BSTData
+ create(parameters: String): Book*