

Declaro por minha honra que este diagrama foi realizado apenas pelos elementos que constituem o grupo do projeto
Triagá

Declaro por minha honra que este diagrama foi realizado apenas pelos elementos que constituem o grupo do projeto.

Rodrigo Baptista

Library

- creators: Collection < Creator >
- works: Collection < Work >
- users: Collection < User >
- requests: Collection < Request >
- currentRate: Rate

- + registerCreator (name: String): Creator
- + getAllCreators(): Collection < Creator >
- + getCreator (name: String): Creator
- + registerBook (IDwork: int, available: int, stock: int, title: String, price: int, type: Type, creators: Collection < String >, ISBN: String): Book
- + registerDVD (IDwork: int, available: int, stock: int, title: String, price: int, type: Type, creator: String, IGAC: String): DVD
- + getAllWorks(): Collection < Work >
- + getWork (IDwork: int): Work
- + registerUser (IDuser: int, name: String, email: String, history: Collection < Book >): User
- + getAllUsers(): Collection < User >
- + getUser (IDuser: int): User
- + requestWork (IDuser: int, IDwork: int): Request
- + getRequest (IDuser: int): Collection < Request >
- + getCurrentRate(): Rate
- + advanceRate (days: int): void

bci



Digitalizada com CamScanner

Digitalizada com CamScanner

- Library : Library
 - Filename : String

+ getLibrary(): Library
 + save(): void
 + search(filename: String): void
 + load(filename: String): void
 + importFile(filename: String): void
 + advanceDate(days: int): void
 + displayDate(): int
 + registerCreator(name: String): Creator
 + getAllCreators(): Collection < Creator >
 + getCreator(name: String): Creator
 + registerBook(IDwork: int, available: int, stock: int, title: String, price: int, type: Type, Creators: Collection < String >, ISBN: String): Book
 + registerDVD(IDwork: int, available: int, stock: int, title: String, price: int, type: Type, Creator: String, I & A C: String): DVD
 + getAllWorks(): Collection < Work >
 + getWork(IDwork: int): Work
 + searchWorks(Criteria: SearchCriteria): Collection < Work >
 + ChangeInventory(IDwork: int, change: int): void
 + registerUser(IDuser: int, name: String, email: String, history: Collection < Boolean >): User
 + getAllUsers(): Collection < User >
 + getUser(IDuser: int): User
 + requestWork(IDuser: int, IDwork: int): Request
 + returnWork(IDwork: int): void
 + getRequest(IDuser: int): Collection < Request >
 + getAllRequests(): Collection < Request >
 + getUserFines(IDuser: int): int
 + payFines(IDuser: int): void
 + getUserNotifications(IDuser: int): Collection < Notification >
 + clearUserNotifications(IDuser: int): void

loci

Creator

- -name: String
+ Creator(name: String)
+ removeCreator(): void

<<abstract>> Work

- -IDWork: int
- stock: int
- -title: String
- price: int
- type: Type

Work(input: String)

<<enumeration>> Type

+ FICTION
+ SCITECH
+ REFERENCE

Book

- -creators: Collection<String>
- -ISBN: String
+ Book(input: String)

DVD

- -creator: String
- -IGAC: String
+ DVD(input: String)

Date

- -current-day: int
+ Date(actual: int)
+ advanceDate(days: int): void

bci.date

bci.work

User
- IDUser: int
- name: String
- email: String
- behaviour: UserBehaviour
- state: UserState
+ User (IDUser: int, name: String, email: String)
+ remove User(): void

UserBehaviour
- maxWork: int
- deadline: int
+ UserBehaviour (max: int, deadline: int)

Normal Behaviour
+ NormalBehaviour (max: int, deadline: int)

bei. user

Good Behaviour
+ GoodBehaviour (max: int, deadline: int)

Bad Behaviour
+ BadBehaviour (max: int, deadline: int)

UserState
+ canRequest (u: user, w: work): boolean
+ return (u: user, w: work): void
+ suspend (u: user): void
+ reactivate (u: user): void

Active State
Suspended State

Request
+ work: Work
- work: work
- deadline: int
- valid: boolean
- IDRules: int
- fine: int
+ Request (user: User, work: Work, deadline: int)

Rule Verification
- R1: String
- R2: String
- R3: String
- R4: String
- R5: String
- R6: String
+ RuleVerification (R1: String, R2: String, R3: String, R4: String, R5: String, R6: String)
+ verify (i: int)

bei. request

bci.exceptions

ImportFileException

MissingFileException

UnavailableFileException

UnrecognisedBinaryException

Notification

-- type: NotificationType

-- msg: String

+ Notification (type: NotificationType,
msg: String, IDUser: Int)

+ deliverNotification (IDUser: Int)

<< enumeration >>
NotificationTypes

+ AVAILABLE

+ REQUEST

<< Search & Receiver >>

+ search(): Collection<object>

Searcher

+ search(): User

SearchWork

+ search(): Work

bci-search

