controller

ControllerMain
bp : BorderPane
controllerAdd : ControllerAdd
controllerGenerate :
ControllerGenerate
controllerRemove :
ControllerRemove
controllerSearch :
ControllerSearch
controllerUpdate :
ControllerUpdate
db : Database

ControllerMain()
addWindow(ActionEvent)
generateWindow(ActionEvent)
getBp()
getDb()
initialize(URL, ResourceBundle)
loadDB()
removeWindow(ActionEvent)
searchWindow(ActionEvent)
updateWindow(ActionEvent)

ControllerUpdate
birthday : DatePicker
cb : ComboBox<String>
cm : ControllerMain
country : TextField
firstName : TextField
gender : ComboBox<String>
height : TextField
id : TextField
lastName : TextField
obCb : ObservableList<String>
obGender :
ObservableList<String>
tf : TextField

ControllerUpdate(ControllerMain)
alert1()
initialize(URL, ResourceBundle)
update()

ControllerSearch
cb : ComboBox<String>
cm : ControllerMain
olCB : ObservableList<String>
tf : TextField
vb : VBox

ControllerSearch(ControllerMain)
convert(String)
initialize(URL, ResourceBundle)
search()

ControllerADD
birthday : DatePicker
cm : ControllerMain
country : TextField
firstName : TextField
gender : ComboBox<String>
height : TextField
id : TextField
lastName : TextField
olgender :
ObservableList<String>

ControllerAdd(ControllerMain)
addPerson(ActionEvent)
alert1()
alert2()
initialize(URL, ResourceBundle)

ControllerGenerate

ControllerGenerate(ControllerMain)
initialize(URL, ResourceBundle)

ControllerRemove

ControllerRemoveControllerMain)
initialize(URL, ResourceBundle)

ui

Main

main(String[])
start(Stage)

interfaz.fxml

model

Database

Database()
addPerson(Person)
globalSearch(String)
loadTextFile(String, String, String,
String)
searchByFirstName(String)
searchByFullName(String)
searchById(String)
searchByLastName(String)
update(String, Person)
updateFullName(String, Person)
updateId(String, Person)
updateLastName(String, Person)
updateName(String, Person)

Person

+MALE:int=1
+FEMALE:int=2
-firstName: String
-lastname: String
-id : String
-photo: String
-gender: int
-birthday: String
-height : double
country: String

Person(String, String, String,
String, String, String, int, double)
compareTo(Person)
getBirthday()
getCountry()
getFirstName()
getFullName()
getGender()
getHeight()
getId()
getLastName()
getPhoto()
setBirthday(String)
setCountry(String)
setFirstName(String)
setFullName(String)
setGender(int)
setHeight(double)
setId(String)
setLastName(String)
setPhoto(String)
toString()

name
<<Person>>

lastName
<<Person>

fullName
<<Person>>

id
<<Person>>

DataStructure

<<Interfaz>>
ABBInterface<k,v>

delete(Node<K, E>)
insert(K, E)
search(K)
update(K, E, K)

<<Interfaz>>
AVLInterface

deleteAVL(Node<K, E>, K)
insert(K, E)
search(K)
update(K, E, K)

<<Interfaz>>
RBTInterface

delete(K)
insert(K, E)
rotateLeft(K)
rotateRight(K)
swpColor(K, K)
update(K, E, K)

Node
balance : int
key : K
type : E

Node(K, E)
getBalance()
getKey()
getLeft()
getParent()
getRight()
getType()
setBalance(int)
setKey(K)
setLeft(Node<K, E>)
setParent(Node<K, E>)
setRight(Node<K, E>)
setType(E)
toString()

ABB<k,v>

ABB()
add(Node<K, E>)
delete(Node<K, E>)
deleteFromKey(K)
fullSearch(K)
getRoot()
insert(K, E)
minValue(Node<K, E>)
next(Node<K, E>)
search(K)
setRoot(Node<K, E>)
update(K, E, K)

AVL

balance(K)
deleteAVL(Node<K, E>, K)
getBalance(Node<K, E>)
height(Node<K, E>)
insert(K, E)
leftRotate(Node<K, E>)
max(int, int)
maximum()
minimum()
minValueAVL(Node<K, E>)
rightRotate(Node<K, E>)
rotateL(Node<K, E>)
rotateLeft(K)
rotateR(Node<K, E>)
rotateRight(K)
search(K)
update(K, E, K)

RBT

balance(K)
delete(K)
insert(K, E)
rotateL(RBNode<K, E>)
rotateLeft(K)
rotateR(RBNode<K, E>)
rotateRight(K)
swpColor(K, K)
update(K, E, K)