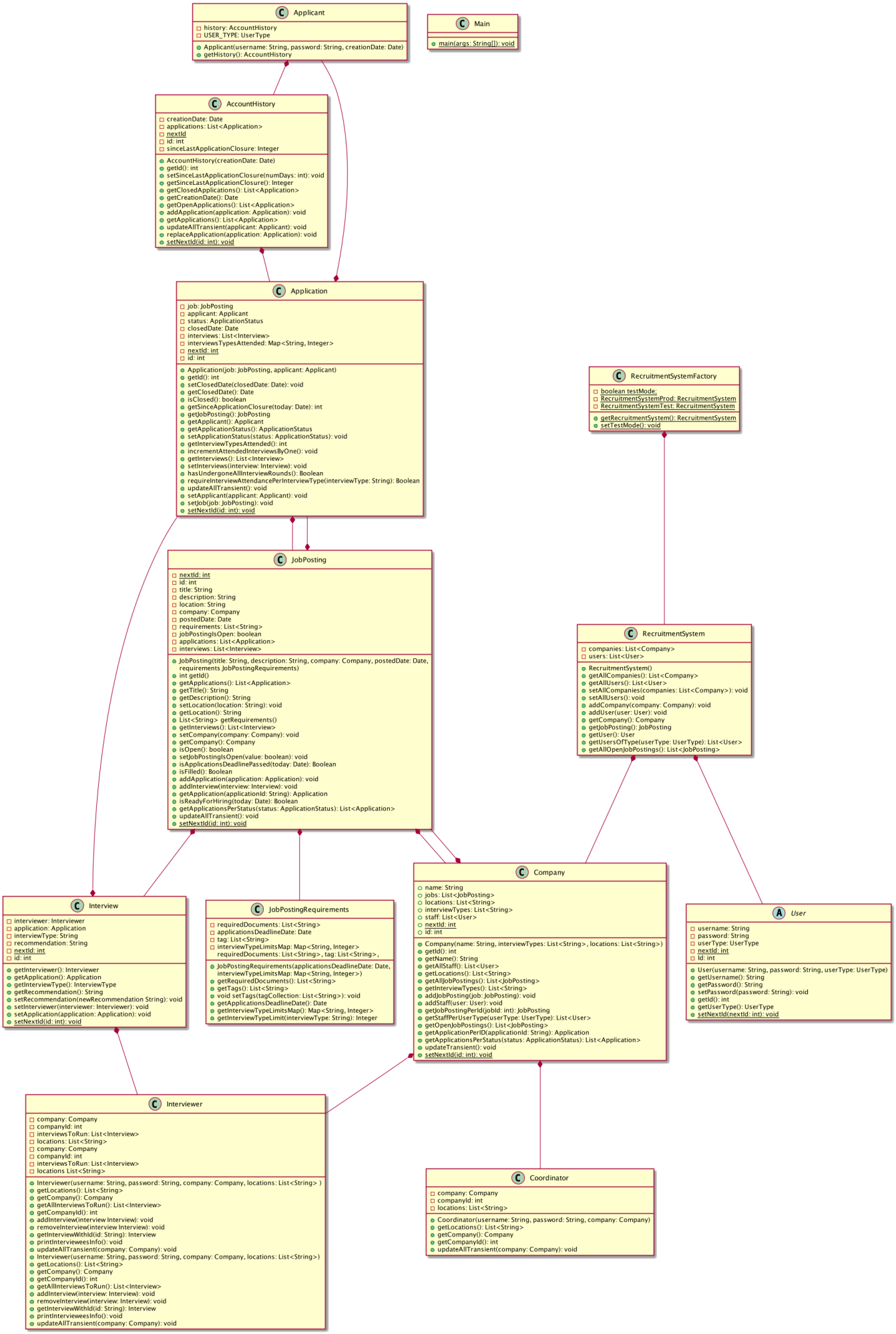
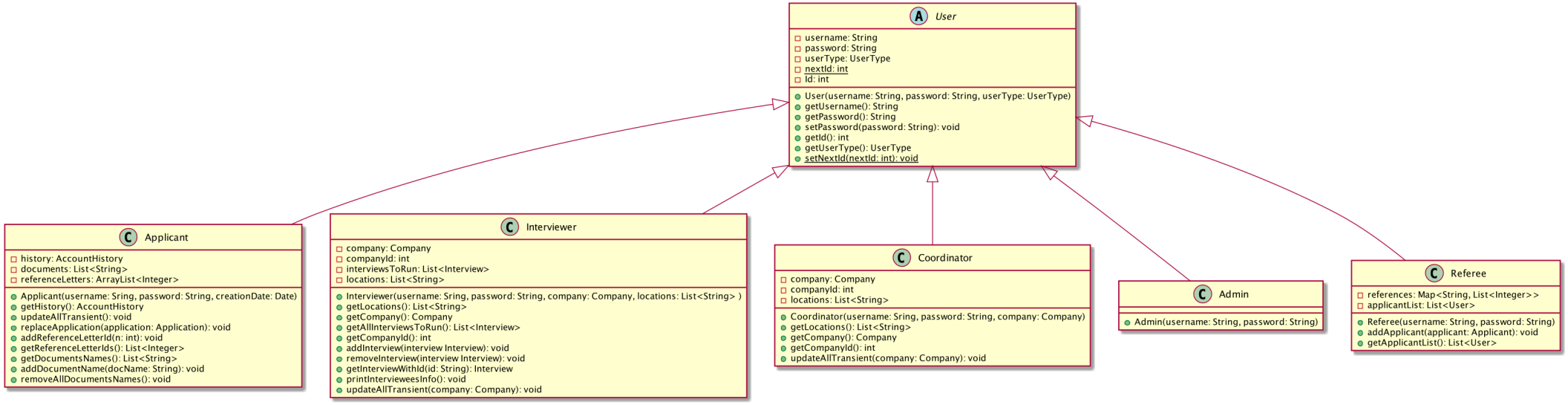
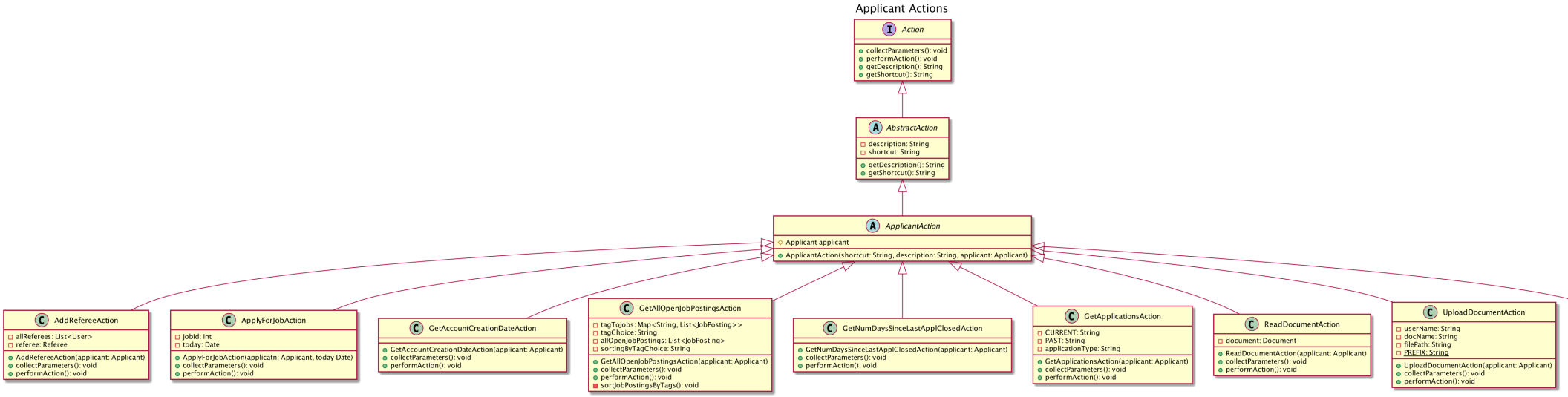
Composition

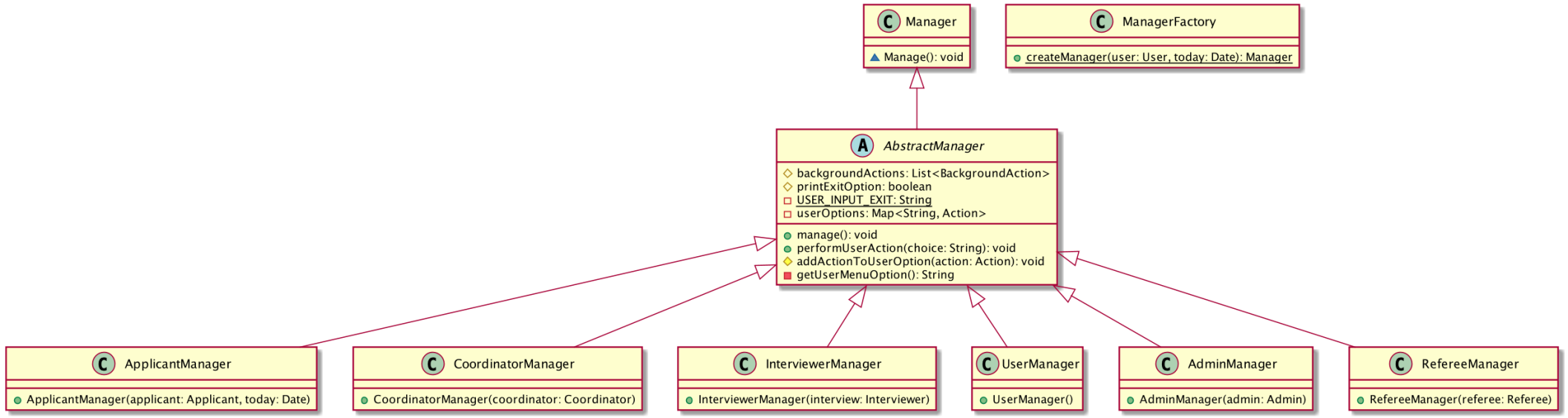


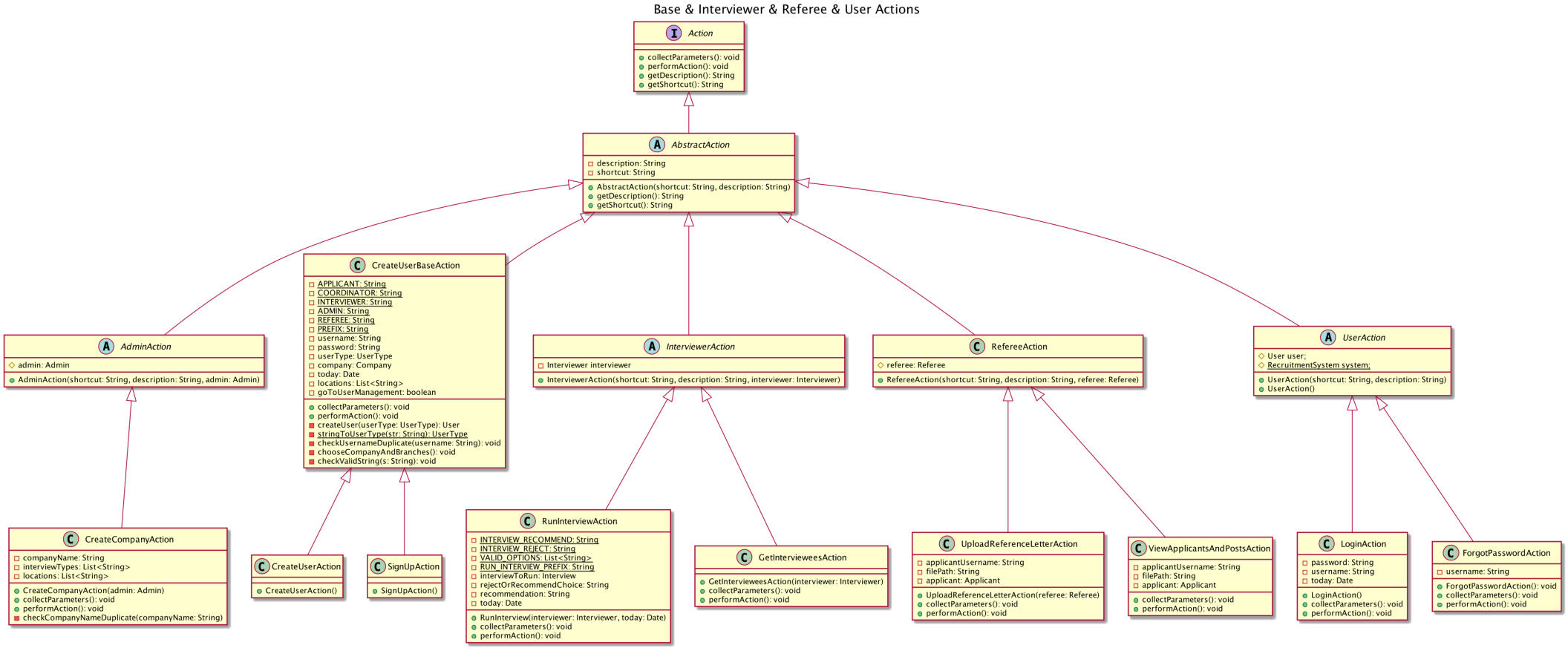
Users

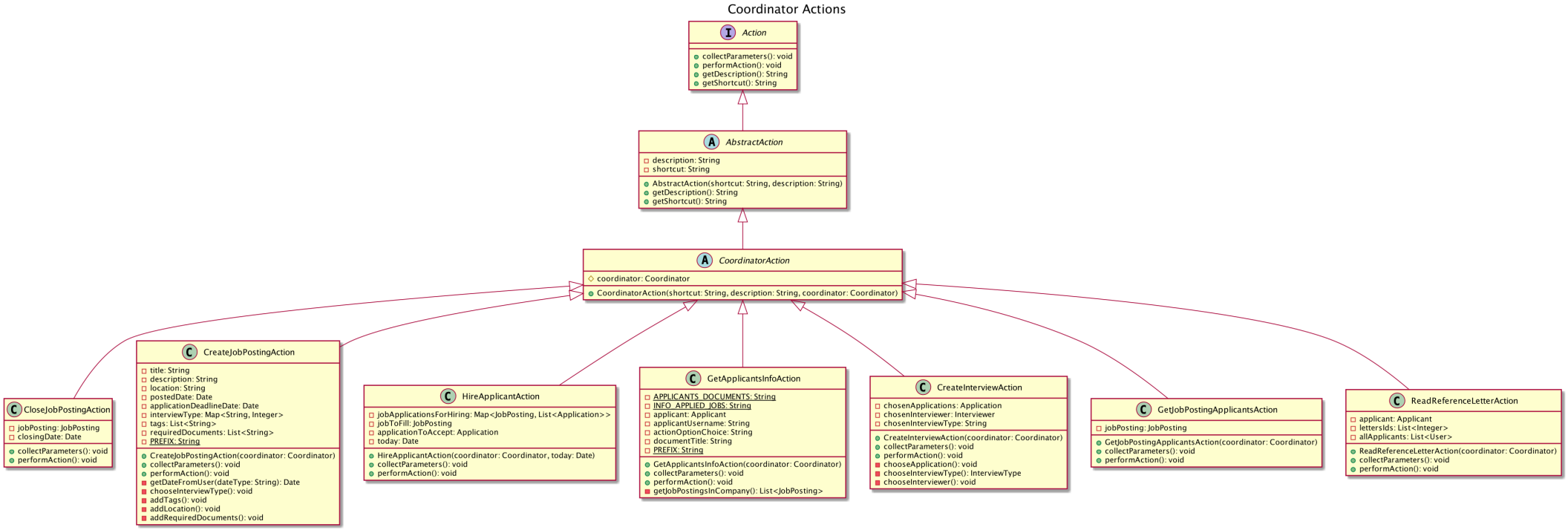

User
- username: String
- password: String
- userType: UserType
- nextId: int
- Id: int
+ User(username: String, password: String, userType: UserType)
+ getUsername(): String
+ getPassword(): String
+ setPassword(password: String): void
+ getId(): int
+ getUserType(): UserType
+ setNextId(nextId: int): void
Applicant
- history: AccountHistory
- documents: List<String>
- referenceLetters: ArrayList<Integer>
+ Applicant(username: Sring, password: String, creationDate: Date)
+ getHistory(): AccountHistory
+ updateAllTransient(): void
+ replaceApplication(application: Application): void
+ addReferenceLetterId(n: int): void
+ getReferenceLetterIds(): List<Integer>
+ getDocumentsNames(): List<String>
+ addDocumentName(docName: String): void
+ removeAllDocumentsNames(): void
Interviewer
- company: Company
- companyId: int
- interviewsToRun: List<Interview>
- locations: List<String>
+ Interviewer(username: Sring, password: String, company: Company, locations: List<String>)
+ getLocations(): List<String>
+ getCompany(): Company
+ getAllInterviewsToRun(): List<Interview>
+ getCompanyId(): int
+ addInterview(interview Interview): void
+ removeInterview(interview Interview): void
+ getInterviewWithId(id: String): Interview
+ printIntervieweesInfo(): void
+ updateAllTransient(company: Company): void
Coordinator
- company: Company
- companyId: int
- locations: List<String>
+ Coordinator(username: Sring, password: String, company: Company)
+ getLocations(): List<String>
+ getCompany(): Company
+ getCompanyId(): int
+ updateAllTransient(company: Company): void
Admin
+ Admin(username: String, password: String)
Referee
- references: Map<String, List<Integer>>
- applicantList: List<User>
+ Referee(username: String, password: String)
+ addApplicant(applicant: Applicant): void
+ getApplicantList(): List<User>


Applicant Actions
Action
+ collectParameters(): void
+ performAction(): void
+ getDescription(): String
+ getShortcut(): String
AbstractAction
- description: String
- shortcut: String
+ getDescription(): String
+ getShortcut(): String
ApplicantAction
◇ Applicant applicant
+ ApplicantAction(shortcut: String, description: String, applicant: Applicant)
AddRefereeAction
- allReferees: List<User>
- referee: Referee
+ AddRefereeAction(applicant: Applicant)
+ collectParameters(): void
+ performAction(): void
ApplyForJobAction
- jobId: int
- today: Date
+ ApplyForJobAction(applicatn: Applicant, today Date)
+ collectParameters(): void
+ performAction(): void
GetAccountCreationDateAction
+ GetAccountCreationDateAction(applicant: Applicant)
+ collectParameters(): void
+ performAction(): void
GetAllOpenJobPostingsAction
- tagToJobs: Map<String, List<JobPosting>>
- tagChoice: String
- allOpenJobPostings: List<JobPosting>
- sortingByTagChoice: String
+ GetAllOpenJobPostingsAction(applicant: Applicant)
+ collectParameters(): void
+ performAction(): void
- sortJobPostingsByTags(): void
GetNumDaysSinceLastApplClosedAction
+ GetNumDaysSinceLastApplClosedAction(applicant: Applicant)
+ collectParameters(): void
+ performAction(): void
GetApplicationsAction
- CURRENT: String
- PAST: String
- applicationType: String
+ GetApplicationsAction(applicant: Applicant)
+ collectParameters(): void
+ performAction(): void
ReadDocumentAction
- document: Document
+ ReadDocumentAction(applicant: Applicant)
+ collectParameters(): void
+ performAction(): void
UploadDocumentAction
- userName: String
- docName: String
- filePath: String
- PREFIX: String
+ UploadDocumentAction(applicant: Applicant)
+ collectParameters(): void
+ performAction(): void

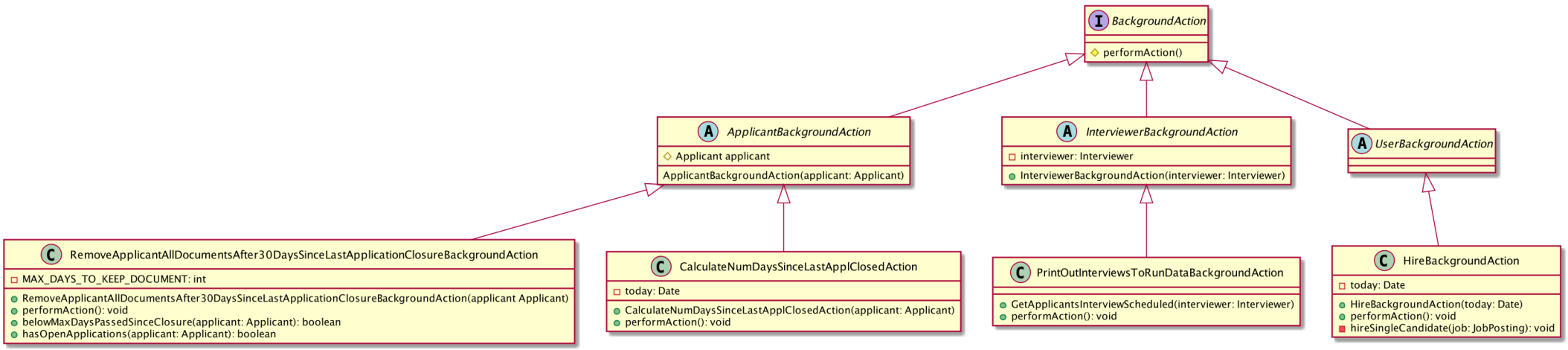
Managers


Manager
Manage(): void
ManagerFactory
createManager(user: User, today: Date): Manager
AbstractManager
backgroundActions: List<BackgroundAction>
printExitOption: boolean
USER_INPUT_EXIT: String
userOptions: Map<String, Action>
manage(): void
performUserAction(choice: String): void
addActionToUserOption(action: Action): void
getUserMenuOption(): String
ApplicantManager
ApplicantManager(applicant: Applicant, today: Date)
CoordinatorManager
CoordinatorManager(coordinator: Coordinator)
InterviewerManager
InterviewerManager(interview: Interviewer)
UserManager
UserManager()
AdminManager
AdminManager(admin: Admin)
RefereeManager
RefereeManager(referee: Referee)

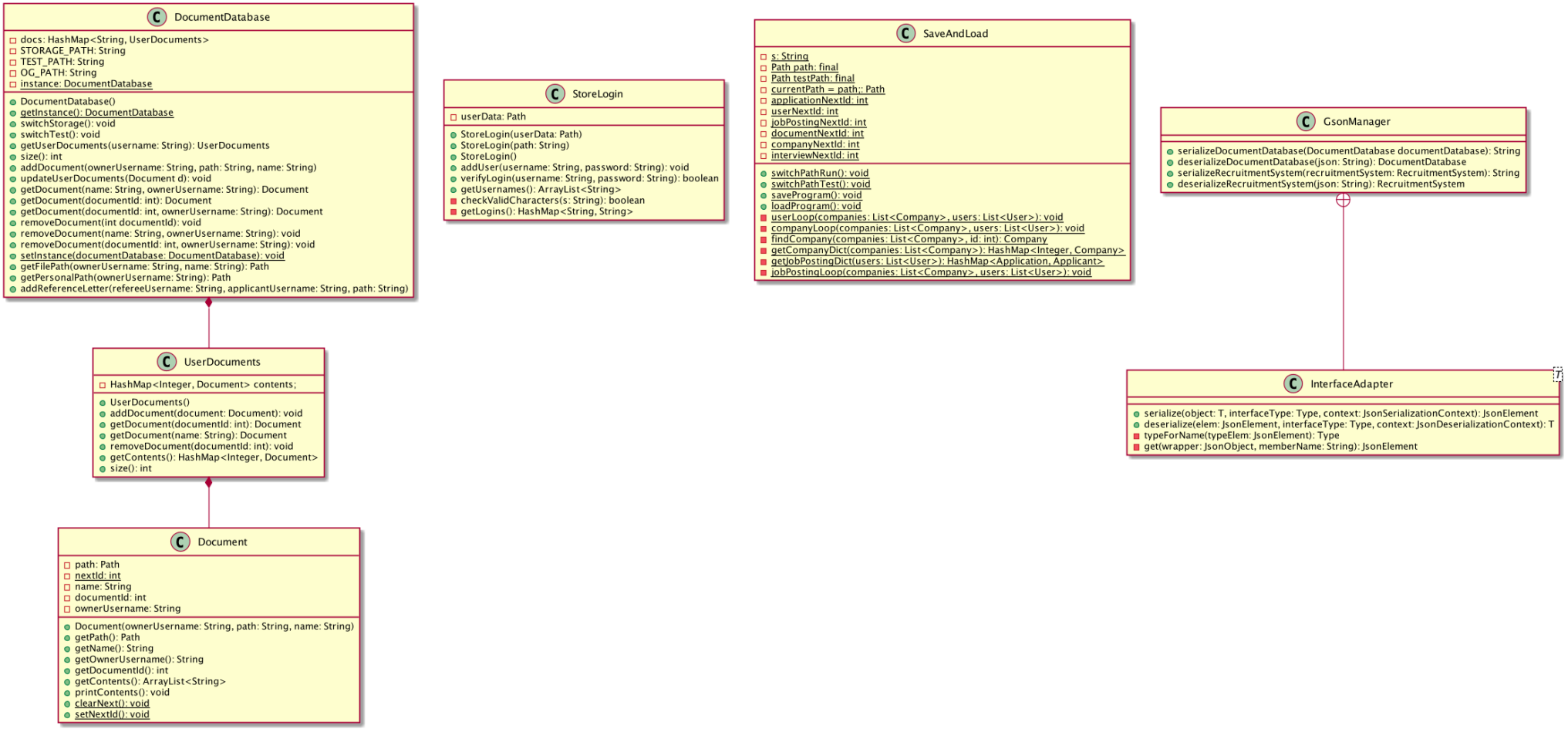

Base & Interviewer & Referee & User Actions
Action
collectParameters(): void
performAction(): void
getDescription(): String
getShortcut(): String
AbstractAction
description: String
shortcut: String
AbstractAction(shortcut: String, description: String)
getDescription(): String
getShortcut(): String
CreateUserBaseAction
APPLICANT: String
COORDINATOR: String
INTERVIEWER: String
ADMIN: String
REFEREE: String
PREFIX: String
username: String
password: String
userType: UserType
company: Company
today: Date
locations: List<String>
goToUserManagement: boolean
collectParameters(): void
performAction(): void
createUser(userType: UserType): User
stringToUserType(str: String): UserType
checkUsernameDuplicate(username: String): void
chooseCompanyAndBranches(): void
checkValidString(s: String): void
AdminAction
admin: Admin
AdminAction(shortcut: String, description: String, admin: Admin)
InterviewerAction
Interviewer interviewer
InterviewerAction(shortcut: String, description: String, interviewer: Interviewer)
RefereeAction
referee: Referee
RefereeAction(shortcut: String, description: String, referee: Referee)
UserAction
User user;
RecruitmentSystem system;
UserAction(shortcut: String, description: String)
UserAction()
CreateCompanyAction
companyName: String
interviewTypes: List<String>
locations: List<String>
CreateCompanyAction(admin: Admin)
collectParameters(): void
performAction(): void
checkCompanyNameDuplicate(companyName: String)
CreateUserAction
CreateUserAction()
SignUpAction
SignUpAction()
RunInterviewAction
INTERVIEW_RECOMMEND: String
INTERVIEW_REJECT: String
VALID_OPTIONS: List<String>
RUN_INTERVIEW_PREFIX: String
interviewToRun: Interview
rejectOrRecommendChoice: String
recommendation: String
today: Date
RunInterview(interviewer: Interviewer, today: Date)
collectParameters(): void
performAction(): void
GetIntervieweesAction
GetIntervieweesAction(interviewer: Interviewer)
collectParameters(): void
performAction(): void
UploadReferenceLetterAction
applicantUsername: String
filePath: String
applicant: Applicant
UploadReferenceLetterAction(referee: Referee)
collectParameters(): void
performAction(): void
ViewApplicantsAndPostsAction
applicantUsername: String
filePath: String
applicant: Applicant
collectParameters(): void
performAction(): void
LoginAction
password: String
username: String
today: Date
LoginAction()
collectParameters(): void
performAction(): void
ForgotPasswordAction
username: String
ForgotPasswordAction(): void
collectParameters(): void
performAction(): void


Coordinator Actions
Action
collectParameters(): void
performAction(): void
getDescription(): String
getShortcut(): String
AbstractAction
description: String
shortcut: String
AbstractAction(shortcut: String, description: String)
getDescription(): String
getShortcut(): String
CoordinatorAction
coordinator: Coordinator
CoordinatorAction(shortcut: String, description: String, coordinator: Coordinator)
CloseJobPostingAction
jobPosting: JobPosting
closingDate: Date
collectParameters(): void
performAction(): void
CreateJobPostingAction
title: String
description: String
location: String
postedDate: Date
applicationDeadlineDate: Date
interviewType: Map<String, Integer>
tags: List<String>
requiredDocuments: List<String>
PREFIX: String
CreateJobPostingAction(coordinator: Coordinator)
collectParameters(): void
performAction(): void
getDateFromUser(dateType: String): Date
chooseInterviewType(): void
addTags(): void
addLocation(): void
addRequiredDocuments(): void
HireApplicantAction
jobApplicationsForHiring: Map<JobPosting, List<Application>>
jobToFill: JobPosting
applicationToAccept: Application
today: Date
HireApplicantAction(coordinator: Coordinator, today: Date)
collectParameters(): void
performAction(): void
GetApplicantsInfoAction
APPLICANTS_DOCUMENTS: String
INFO_APPLIED_JOBS: String
applicant: Applicant
applicantUsername: String
actionOptionChoice: String
documentTitle: String
PREFIX: String
GetApplicantsInfoAction(coordinator: Coordinator)
collectParameters(): void
performAction(): void
getJobPostingsInCompany(): List<JobPosting>
CreateInterviewAction
chosenApplications: Application
chosenInterviewer: Interviewer
chosenInterviewType: String
CreateInterviewAction(coordinator: Coordinator)
collectParameters(): void
performAction(): void
chooseApplication(): void
chooseInterviewType(): InterviewType
chooseInterviewer(): void
GetJobPostingApplicantsAction
jobPosting: JobPosting
GetJobPostingApplicantsAction(coordinator: Coordinator)
collectParameters(): void
performAction(): void
ReadReferenceLetterAction
applicant: Applicant
lettersIds: List<Integer>
allApplicants: List<User>
ReadReferenceLetterAction(coordinator: Coordinator)
collectParameters(): void
performAction(): void

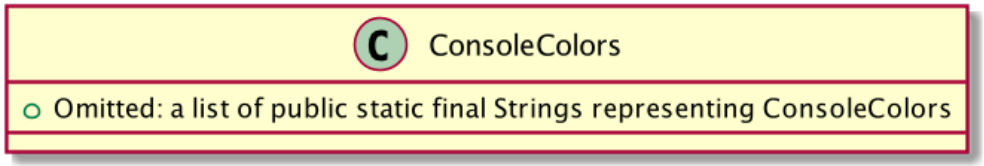
Background Actions

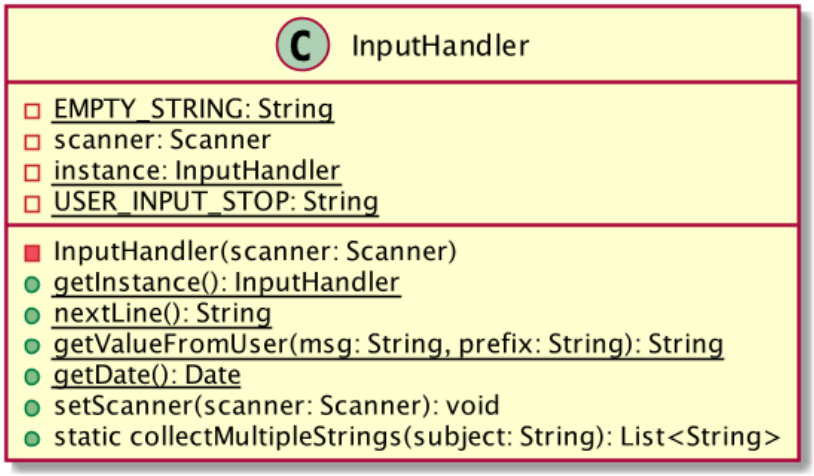


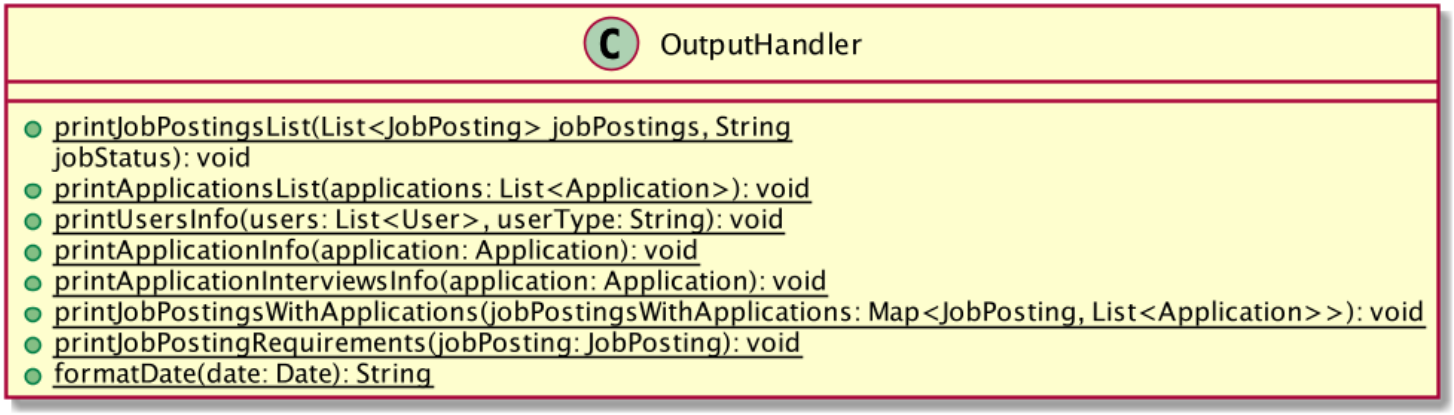
Documents & Storage



Tools


ConsoleColors
Omitted: a list of public static final Strings representing ConsoleColors


InputHandler
EMPTY_STRING: String
scanner: Scanner
instance: InputHandler
USER_INPUT_STOP: String
InputHandler(scanner: Scanner)
getInstance(): InputHandler
nextLine(): String
getValueFromUser(msg: String, prefix: String): String
getDate(): Date
setScanner(scanner: Scanner): void
static collectMultipleStrings(subject: String): List<String>


OutputHandler
printJobPostingsList(List<JobPosting> jobPostings, String jobStatus): void
printApplicationsList(applications: List<Application>): void
printUsersInfo(users: List<User>, userType: String): void
printApplicationInfo(application: Application): void
printApplicationInterviewsInfo(application: Application): void
printJobPostingsWithApplications(jobPostingsWithApplications: Map<JobPosting, List<Application>>): void
printJobPostingRequirements(jobPosting: JobPosting): void
formatDate(date: Date): String