Query

- queryString: String
- filter: FilterState

+ Query
+ toString(): String
+ <u>fromString(str: String): Query</u>
+ getSearch(): String
+ setSearch(search: String): void
+ getFilter(): FilterState
+ setFilter(filter: FilterState): void

Controller

- searcher: CodeSearcher
- database: Database

+ InputViewAdapter
+ doQuery(): List
- changePage(url): void
+ doExport(): File
+ doImport(file: File): void
+ getCurrentQuery(): Query
+ setCurrentQuery(qry: Query): void
+ getSearcher(): CodeSearcher
+ setSearcher(searcher: CodeSearcher): void
+ getUser(): User
+ setUser(user: User): void
+ validate(uname: String, pw: String): User
- showResults(): void
- displayVisualization(): void
- displayLogFile(user: User): void
+ createExport(): String
+ parseImport(String): void
+ toggleSound(): void

creates

CodeSearcher

- database: Database

+ CodeSearcher
+ search(query: Query): List

has a

User

- id:int
- username:String
- password:String
- logFile: List

+ User
+ addQueryToLog(qry: Query): boolean
+ clearLog(): boolean

has a

FilterState

- repoName:String
- committer:String
- type:enum
- startDate: Calendar
- endDate: Calendar

+ FilterState
+ getRepoName(): String
+ setRepoName(name: String): void
+ getCommitter(): String
+ setCommitter(commiter: String): void
+ getType(): enum
+ setType(type: enum): void
+ getStartDate(): Calendar
+ setStartDate(startDate: Calendar): void
+ getEndDate(): Calendar
+ setEndDate(endDate: Calendar): void

can update

View

+ query: TextField
+ repo: TextField
+ searchCommit: CheckBox
+ searchCode: CheckBox
+ importButton: Button
+ exportButton: Button
+ visualization: Canvas
+ resultsList: TextView
+ timeFilter: DropDown

+ toggleSound: Toggle

has a

Searches and Manipulates

stores a

Database

- currentQuery: Query
- currentUser: User
- currentResults: List
- userList: List

+ Database
+ getCurrentQuery(): Query
+ setCurrentQuery(query: Query): void
+ getCurrentUser(): User
+ setCurrentUser(user: User): void
+ getCurrentResults(): List
+ setCurrentResults(results: List): void

Controller belongs to the controller portion of our architecture diagram, coordinating between the view and the model (database). CodeSearcher also belongs to the controller portion, handling the actual searching of queries and returning of results (interacts with Github, which is not an entity we create thus we did not put it explicitly in the diagram).

User, Query, and FilterState all belong to the model portion of our architecture diagram, working together to store information about the current query and the active user. They are stored in Database, which is the "front end" of the model and interacts with the controller classes.

View is in the view portion of our architecture diagram, and represents our program's UI.