Assignment 3 – REFACTOR

Main changes:

Old design	New design	Reason
EmailGenerator Provides getEmail(IMember member) to generate email for the member (replaces placeholders and returns the email as a string)	**Template** Finds placeholders; Replaces placeholders with info in Evaluator and returns a new Email **Evaluator (interface)** Returns the value corresponding to the placeholder **Email** Stores the content of email. Provides getters to get the main text and email address to which this email will be sent.	To follow the one class, one responsibility principle. Implemented Evaluator to encapsulate the flight information and passenger information.
Member (a class that stores info of one passenger)	Removes this class, all passenger infos are stored in *PassengerInfo*	
MemberDatebase Parses the csv file; stores passenger info as a list of Member objects	**PassengerInfo** Uses a *CSVParser* to parse the csv file. Provides getters to obtain headers and passenger info for each passenger. Also provides hasNextPassenger() and nextPassenger() **CSVParser** Parses the csv file. Returns List<List<String>>, in which each List<String> represents one line in csv file	
ArgumentsParser Besides parsing the info like template file name, csv file, event, and output directory, also parses the cities from csv file name and returns a map that contains this info.	**CMDHandler** Only parses the cmd arguments. Leaves the parsing of cities to *FlightInfo*. **FlightInfo** Parses departure and destination cities from csv file name. Also stores the event details. Provides getters to obtain these infos.	
NA	Classes implemented *Evaluator* **DateEvaluator** **KeyValueEvaluator** Class implemented *IEvaluatorContainer* (extends *Evaluator*) **EvaluatorContainer**	
NA	**IOLibrary** Reads files and returns a string that contains the content of file; Writes a string to a *File*	Puts all io operations in one class to reduce replicate codes. This is also for the safety.

What remains the same as before:

EmailAutomationTool is the main controller of this program.
1) It accepts the command-line arguments from user and uses a CMDHandler to parse the arguments.
2) Creates a EvaluatorContainer that contains all sources of info.
3) For each passenger in PassengerInfo, generates a new Email by passing the EvaluatorContainer to Template.
4) Depending on the mode that user selects, either saves the emails to an output directory, or sends the eamils to passengers.

EmailAutomationTool throws an InvalidInputFormatException if the cmd-line arguments miss some necessary info.

Template throws an InvaliPlaceholderException if the placeholder information cannot be determined.

(see UML diagram on next page)

Powered by yFiles