Virtual Development Environment in a Box

Students: Dylan McDougall (dmcdougall2019@my.fit.edu)

and Ian Orzel (iorzel2019@my.fit.edu)

Faculty: Ryan Stansifer (ryan@fit.edu)

Client: Ryan Stansifer (ryan@fit.edu)

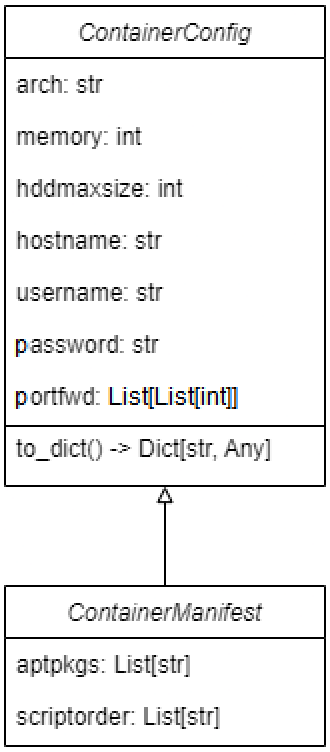
Progress of current Milestone:

Task	Completion	Ian	Dylan	Todo
Implement, test, and demo specifications for container creation	100%	0%	100%	
Implement, test, and demo container repositories	90%	90%	0%	Testing
Implement, test, and demo more intuitive file system interaction	100%	5%	95%	
Implement, test, and demo fixes found by students in the Compiler Theory course	100%	60%	40%	

Discussion:

- **Implement, test, and demo specifications for container creation**:

  In preparation for completing the container creation feature in the next milestones, the classes described by the diagram on the right were added. *ContainerConfig* describes an already created container, and *ContainerManifest* extends *ContainerConfig* and adds some members that will be used by the container creation wizard when implemented.



The creation of a container will require a specific directory structure created by the new *make_skeleton()* function. The directory structure, in reference to a root directory, can be described as follows:

- **resources**: Files or directories that will be copied into the new container.
- **scripts**: Scripts to be executed by the container post-installation. Execution order may be specified using the *scriptorder* member of the *ContainerManifest*.
- **packages**: Debian packages to be installed post-installation. (This is unrelated to *aptpkgs*, which specifies packages to install from the official Debian repositories.)
- **build/dist**: The directory where the final results of the build will be stored.
- **build/temp**: A temporary directory used during the creation process.
- **manifest.json**: A JSON file that will be parsed and used to initialize a *ContainerManifest* instance.

● **Implement, test, and demo container repositories**: We created the code for a repository, which allows users to download containers, get a list of containers on the repository, and upload containers after authenticating the user (which we left blank to be filled by the user). Then, we updated the container manager to be able to download containers from an archive, add a repository to the manager, update a repository in the manager, and upload a container to a repository. Then we had to implement an archive command that converted a container to an archive file.

● **Implement, test, and demo more intuitive file system interaction**:
The third argument of the get-file and send-file commands is now optional. If no destination file is provided, the destination file will inherit the name of the source file. send-file will default the container's default directory. get-file will default to the host's

current working directory. Also, if the destination file is a directory, the commands will assume the user wants a file with the same name as the source file copied into sid directory.

The files and sftp commands were added. The sftp command opens an sftp shell, implemented for those familiar with the protocol. The files command will open the container's internal virtual filesystem in a graphical file manager. This file manager is an embedded FileZilla program. We went with this because FileZilla is a popular cross-platform application that supports graphically browsing remote file systems via sftp. The program also has a relatively small footprint, so we can ship it with our program without significantly increasing the size of the installer.

- **Implement, test, and demo fixes found by students in the Compiler Theory course**: During this milestone, the Compiler Theory course had the opportunity to use our project to complete an assignment. During this time, these students were able to find some bugs and suggest some features in our project. For instance, there was a bug where file paths were treated as relative rather than absolute. Also, if the provided file path is a directory, we put the file in the directory and assume the name of the file. We also implemented an update script that allowed students to pull updates on the program from our GitHub repository.

Member Discussion:

- **Ian Orzel**: During this milestone, my first focus was on troubleshooting for the Compiler Theory course. Students of the course had an assignment during the first couple of weeks that required the use of the program. So, we had to teach students how to use the program and provide support for them with it. During this, I helped fix a lot of the bugs that students ran into while they were working on the assignment. I also created an update script that would allow students to easily update the tool when bug fixes were released. For the second half of the milestone, my priority was to work on the repositories. I created the repository servers and added the functionality to the container manager to interact with them.

- **Dylan McDougall**:

  This was a very significant milestone for our project. The general stability of the program was increased significantly, and this was no small task. A significant portion of the code was rewritten, particularly in the ContainerManagerServer class, but the entire codebase was modified to some extent. The result is that we have a much more reliable product than the one we distributed to the Compiler Theory class at the beginning of that class, thanks a lot to the class's feedback. I put a lot of effort into changing the code to make the software more stable, where Ian worked more directly with the students to resolve their immediate issues. I also worked on improving the user experience for file interaction by improving the already existing get- and send-file commands, as well as adding a graphical option for managing files, which I think our users will really appreciate once they get their hands on it. Lastly I worked on laying the groundwork for the container creation wizard by defining container creation specifications. This is not a change that affects the end-user yet, but is significant nonetheless.

Next Milestone Matrix:

Task	Ian	Dylan
Implement, test, and demo the Container Creation wizard	1%	99%
Implement, test, and demo the ability to delete and rename containers	100%	0%
Implement, test, and demo Test Suite based on requirements	90%	10%
Conduct evaluation and analyze results	50%	50%
Create poster for Senior Design Showcase	50%	50%

Discussion of Planned Tasks for Next Milestone:

- **Implement, test, and demo the Container Creation wizard**: With the completion of this milestone, completing the creation wizard is now possible. It will allow users of the software to create their own customized container, specify the software packages to be included, and set up the environment with shell scripts which will be executed in the final stages of the build. The output of this wizard will be a distributable container archive.
- **Implement, test, and demo the ability to delete and rename containers**: Currently, there are two commands in the design document that have not yet been added to the tool: the delete command and the rename command. The delete command will remove a container from the user's file system. The rename command will change the name of a container for future use.
- **Implement, test, and demo Test Suite based on requirements**: Currently, our project does not have a way to ensure that it is fulfilling all requirements. We want to implement a test suite that ensures that the current code fulfills all of the requirements. This is important because it ensures that our project works how we want, especially as updates are added to it in the future.

- **Conduct evaluation and analyze results**: For our evaluation, we will first run the test suite that we are creating during this milestone. Then, we want to survey the students of the compiler theory course to receive their input on our tool. Finally, we want to study people using the tool to see how long it takes them to complete certain tasks. After this, we will bring all of this information together and analyze how well the tool is currently fulfilling our goals.
- **Create poster for Senior Design Showcase**: For the Senior Design Showcase, we have to have a poster that presents our project. During this milestone, we will create this poster that summarizes our project so that we can explain what we did to the judges.

Dates of Meeting with Client: (See Faculty Meeting Times)

Client Feedback for Milestone: (See Faculty Feedback)

Dates of Meeting with Faculty:

- January 18
- January 25
- February 8

Faculty Feedback for Milestone:

- **Implement, test, and demo specifications for container creation**:

- **Implement, test, and demo container repositories**:

- **Implement, test, and demo more intuitive file system interaction**:

- **Implement, test, and demo fixes found by students in the Compiler Theory course**:

Faculty Advisor Signature: ____ Date: _________

Score for each member:

Ian	0	0.5	1	1.5	2	2.5	3	3.5	4	4.5	5	5.5	6	6.5	7	7.5	8	8.5	9	9.5	10
Dylan	0	0.5	1	1.5	2	2.5	3	3.5	4	4.5	5	5.5	6	6.5	7	7.5	8	8.5	9	9.5	10

Faculty Advisor Signature: ____ Date: _________