Can Advanced Type Systems Be Usable? An Empirical Study of Ownership, Assets, and Typestate in Obsidian

Getting Started

The artifact consists of three parts:

1. The Obsidian compiler
2. A VSCode plugin to support editing and compiling Obsidian code within VSCode
3. Replication materials for the user study

Claims supported:
1. The Obsidian compiler works to typecheck basic Obsidian programs, such as those used in the user study.
2. The user study could be replicated by other researchers.

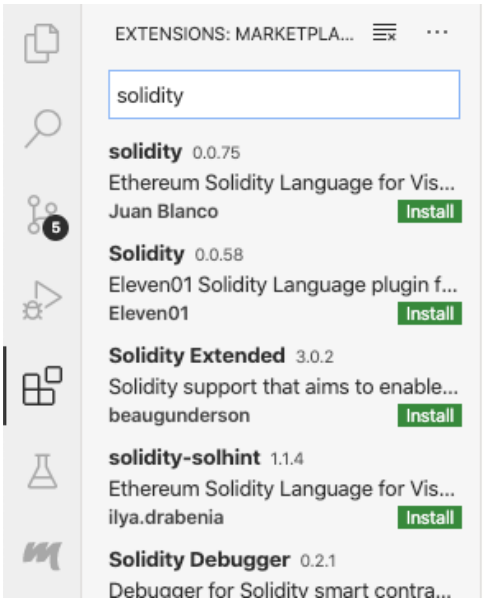
The Obsidian tools have been tested on macOS 10.15.6. Our collaborators have had success in the past in using the compiler on Linux. We have encountered problems on Windows, and although those should be solvable, we do not recommend use on Windows at this time. However, the tools are not explicitly platform-dependent, so there should not be any serious barriers to getting the tools working on other platforms.

To build the compiler, you will need to first install:
- sbt (https://www.scala-sbt.org)
  - Use one of the minor updates under 1.3, such as 1.3.10.
- Oracle Java 11 (https://www.oracle.com/java/technologies/javase-jdk11-downloads.html)
- Gradle (https://gradle.org/install/), e.g., 6.6.
- Protoc (https://github.com/protocolbuffers/protobuf/releases), e.g., 3.13.0.

We have included a copy of the Obsidian codebase in the replication package so that the artifact stands alone (without depending on GitHub). If you want to check it out independently for whatever reason, you can:

```
1. git clone https://github.com/mcoblenz/Obsidian.git
2. cd Obsidian
3. git checkout b87ddb00b340010342dbc3f7b6495e1746f34966
```

Please ensure that the artifact is unpacked in a directory path that does NOT contain spaces.



Then, inside the Obsidian directory, run `make`. This will run all of the unit tests; if they pass, you have a working Obsidian compiler. The compiler translates from Obsidian to Java.

To make the Obsidian compiler accessible from VSCode, you should add Obsidian/bin to your $PATH. If you do not do this, you will likely get errors when you try to use VSCode to compile your Obsidian code.

To actually run the generated code on a blockchain, you would need to install Hyperledger Fabric. Because the user studies did not require participants to execute their compiled code, this step is outside the scope of this replication package. However, if you are curious, you can look at Obsidian/README.md for instructions.

The final step is to build and install the Visual Studio Code extension.

1. Install Code: https://code.visualstudio.com.
2. Change to the “obs-vscode-extension” directory inside the Obsidian directory.
3. Follow the instructions in README.md in that directory. Namely:
    1. Install Node.js: http://nodejs.org
    2. npm install -g typescript
    3. npm install -g vsce
    4. npm install
    5. npm run compile
    6. vsce package
    7. code --install-extension obsidian-*.vsix

Now everything is installed. You can test to make sure the VSCode extension is working. From the Obsidian directory:

1. `code evaluation/obsidian_participant_template/auction-exercises`
   (where “Obsidian” is the top-level directory you checked out at the beginning)
2. In the list of files in the explorer, select “auction.obs”.
3. Bring up the Command Palette. The keyboard shortcut for this is platform-dependent. On macOS, it is Command-Shift-P. On Windows and Linux, it is Ctrl-Shift-P.

4. Select “Obsidian: Compile Smart Contract.” Start typing the command, and it should appear.
5. Wait a few moments. In the Terminal portion of the window, you should see an error message (because this particular starter code does not typecheck yet):

```
auction.obs 27.9: Variable 'money' is incompatibly typed as both
'Money@Unowned' and 'Money@Owned' after branch.
```

You can also see the output in the Problems tab.

The “Compilation succeeded” pop-up message only indicates that the compiler executed, not that there were no errors.

To configure Code to support Solidity, you need to install the Solidity extension in Code. To do this, click the Extensions icon in the panel at the left. Then, type “solidity” in the search box. It should look like the figure below. Click the “install” link corresponding to the version by Juan Blanco. To test that this works, close Code. Then, from the Obsidian directory:

1. `code evaluation/solidity_participant_template/auction-exercises/`
2. In the list of files, select “auction.sol”.
3. In the Command Palette (again, Cmd-Shift-P or Ctrl-Shift-P), select “Solidity: Compile contract.”

If everything worked, you should see that compilation was successful in the Output section of the window. You can also see the output in the Problems tab.

Step-by-Step

Once you have the tools working (above), you can act as if you were a participant in the user study. The starter materials from the studies are in the solidity_participant_template and obsidian_participant_template directories that you saw above.

To act as if you were a participant in the Obsidian condition in the study:

1. Open obsidian-guide/index.html in your web browser.
2. In a second window, open “Obsidian tutorial.pdf” and follow the instructions. When the tutorial tells you to read a document, e.g. “"Ownership -- Introduction,” it is referring to a particular section of the web page you opened in step 1. When the tutorial asks you to do a task “in the VSCode window,” specifying a particular file, this means that you should open that file with Code. The tutorial files can be found in Obsidian/evaluation/ obsidian_participant_template/tutorial-exercises/.

The participants had an interactive, online version of the tutorial; we have included a PDF here so that the replication package stands alone, independent of the service that we previously relied on.

Once you have completed the tutorial, you can move on to the three programming tasks, or you can experiment on your own. The starter code is located in the same obsidian_participant_template directory as before. The participants were first given Auction, then Prescription, and finally Casino, but there are no actual code dependencies and you can explore them in any order you choose. Ignore any “client” files you find; they were not used in the study. The task instructions for Auction and Prescription are included as comments in the

source files. The Casino instructions are located at Obsidian/evaluation/casino-exercises/casino_instructions.html and apply to both experimental conditions.

To act as a Solidity participant, do the same as above except that you should use “Solidity tutorial.pdf” and “solidity-guide/index.html”. The tutorial files and tasks are in Obsidian/evaluation/solidity_participant_template/.