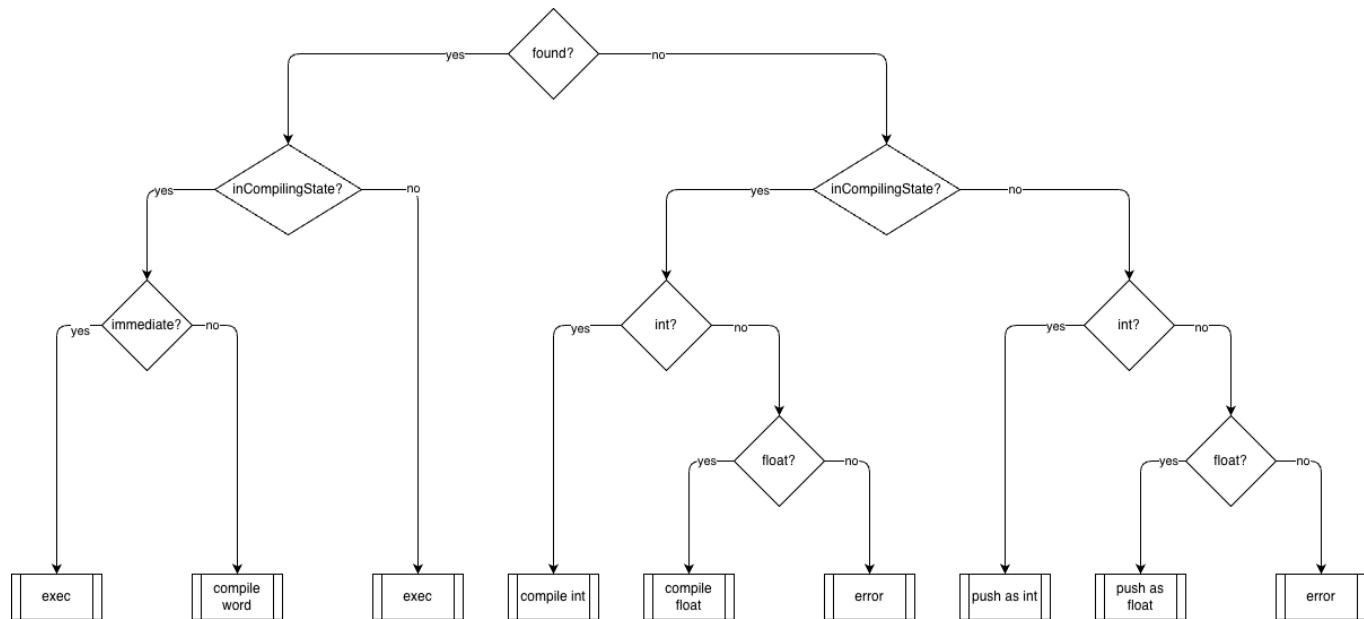


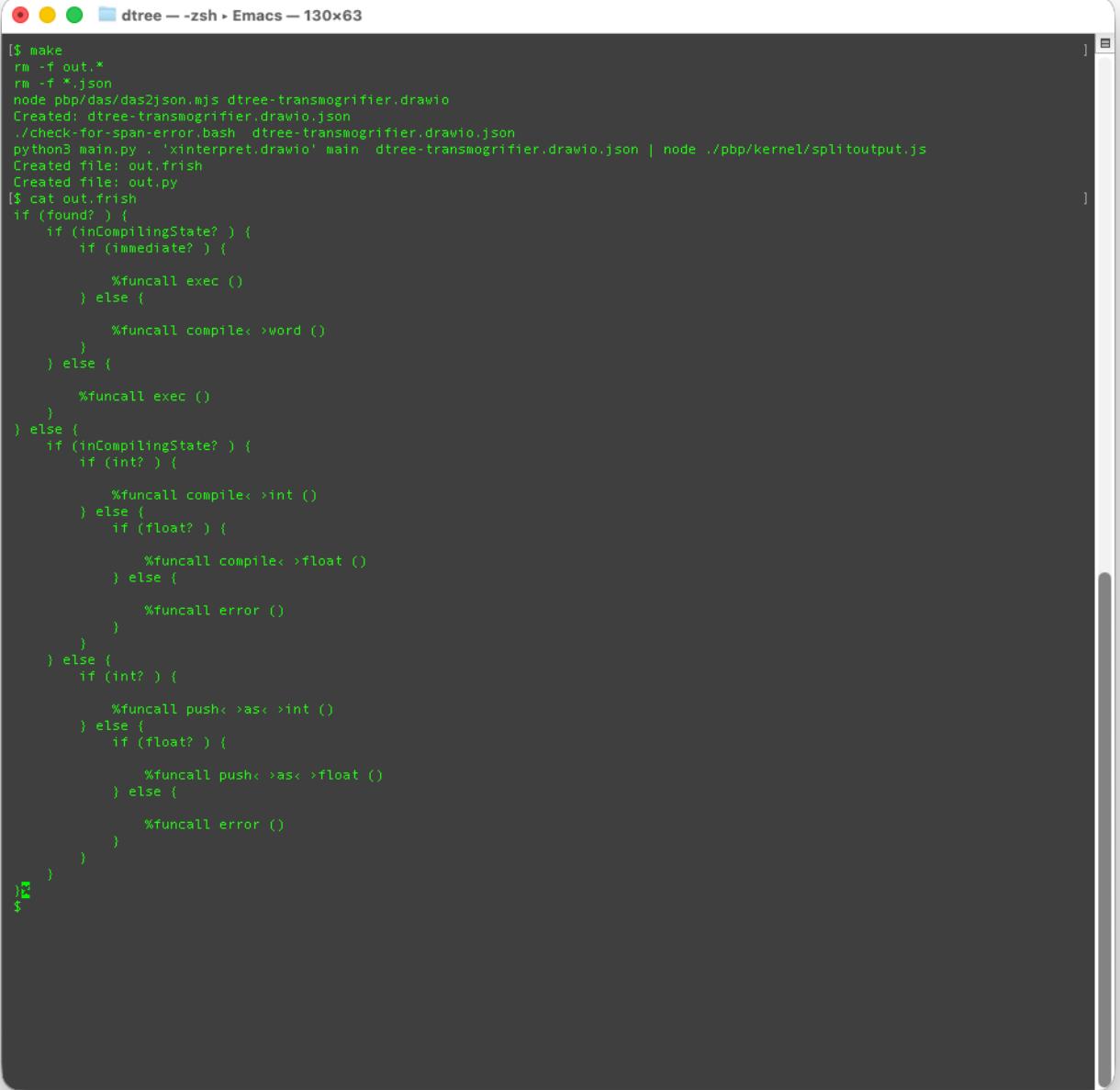
Final

Final

This diagram:



is transmogrified to this code:



The screenshot shows a terminal window titled "dtree -- zsh - Emacs - 130x63". The terminal displays a series of commands and their output. It starts with "\$ make", followed by several file operations like "rm -f" and "node" calls. Then it runs ".check-for-span-error.bash" and "python3 main.py". The output includes "Created: dtree-transmogrifier.drawio.json", "Created file: out.frisch", and "Created file: out.py". Finally, it shows the content of "out.frisch" which contains a complex sequence of conditional statements and function calls, including "%funcall exec ()" and "%funcall compile: >word ()".

```
$ make
rm -f out.*
rm -f *.json
node pbp/das/das2json.mjs dtree-transmogrifier.drawio
Created: dtree-transmogrifier.drawio.json
./check-for-span-error.bash dtree-transmogrifier.drawio.json
python3 main.py . 'xinterpret.drawio' main dtree-transmogrifier.drawio.json | node ./pbp/kernel/splitoutput.js
Created file: out.frisch
Created file: out.py
$ cat out.frisch
if (found? ) {
  if (inCompilingState? ) {
    if (immediate? ) {

      %funcall exec ()
    } else {

      %funcall compile: >word ()
    }
  } else {
    %funcall exec ()
  }
} else {
  if (inCompilingState? ) {
    if (int? ) {

      %funcall compile: >int ()
    } else {
      if (float? ) {

        %funcall compile: >float ()
      } else {

        %funcall error ()
      }
    }
  } else {
    if (int? ) {

      %funcall pushc >as< >int ()
    } else {
      if (float? ) {

        %funcall pushc >as< >float ()
      } else {

        %funcall error ()
      }
    }
  }
}
$
```

In the process, I invented a new way to represent spaces within identifiers and function names. Names that contain spaces are emitted with the spaces surrounded by `< >` unicode characters.

```
push< >as< >float
```