.objewelry: Draft Project Specification

Morgan Steckler & Alisha Naidu

1 Introduction

We plan to build a language that parses specifications from the user to create an STL file that models a personalized ring. Then, the user can export the file to a 3D printer to create a ring. The language will also return a text file with easy to follow instructions for how to use this .stl file to print their ring in the Williams Makerspace. This can help make using 3D printers more accessible to people who have less experience in the 3D printing process, since users can input specifications they would like for their ring (such as size, color, and shape) and be provided with an STL file. This makes the process of obtaining an STL file much simpler, rather than having to create and modify the file from scratch. It also does not require purchasing expensive modeling software.

We believe that our vision of allowing users to easily 3D print custom rings can be achieved using a programming language. Becoming fluent in the 3D printing process takes much more time and energy than entering certain keywords to specify changes to be made to the file. This also comes with limitations for the user, as they do not have as much freedom in creating objects to be 3D printed as they would by learning the process themselves, but we hope that the simplicity of using our language will allow for more people to experiment with 3D printing. We hope that through building this language, we can both learn more about 3D printing and inspire others to try using 3D printers!

2 Design Principles

Our language will require both artistic and technical design. Our main focus is allowing simple input from a user who is not experienced in the 3D printing process. For this reason, the language will have a lot of stored information in preset programs, with the parameters allowing for small customizations to the ring, which is represented by changes within the STL file. However, our language could be expanded by those fluent in F Sharp to make entirely new designs. The language is designed around the 3D printing process in place at the Makerspace, which begins with a .stl file, which is typically created using Fusion360, or other CAD suite software. We designed this language to be simple for the user, but will output a complex file.

3 Examples

dotnet run ”gold band in size 7 ring.stl”
Expected output: an STL file named ”ring.stl” representing the 3D object of a gold band ring in size 7, as specified by the user, along with instructions in a .txt file called ”instructionmanual.txt”.

dotnet run ”black heart in size 5 heartring.stl”
Expected output: an STL file named ”heart.stl” representing the 3D object of a black ring with a heart design, in size 5, along with instructions in a .txt file called ”instructionmanual.txt”.

dotnet run ”silver moon and stars in size 6 moon.stl”
Expected output: an STL file named ”moon.stl” representing the 3D object of a silver ring with moon and stars on it, in size 6.5, along with instructions in a .txt file called ”instructionmanual.txt”.

When the user gets their STL file, they can preview it here: https://www.meshlabjs.net/ or follow the instructions in the instruction manual to print their new ring out!

4 Language Concepts

The user needs to understand a few concepts to write a successful program, including the range of the ring sizes, the design options, and the color options. The order in which the user writes these are also important for the program to be parsed successfully. As demonstrated in the examples section, the color of the ring comes first, then the design, then the word ”in size” and the number (int) with the size of the ring. For this reason, we give a specific usage message, which gives the format of the parameters, along with the options for each parameter.
There are a few concepts from the internal working of the language that could be useful for the user to know. One is that color, design, and size are the primitive types - they are independent of each other, which is what allows the customization. Within the language, these factors are combined into the ”details” combining form internally in the language, in order to create the model of the specific ring. The user also needs to know that the .stl file they are given needs to be imported into slicing software (ex: Cura, PrusaSlicer, MatterControl), which slices the model into layers, that will be stacked level by level by the 3D printer to make the object. The slicer will convert the .stl file to a G-Code file, which is the final set of instructions that the printer uses directly. Depending on the 3D printer, the G-Code can be put on a USB which is then connected to the printer, or their computer can be connected to the printer, then allowing them to choose the model that is now downloaded on their computer.
The language will give the user a text file with directions for using a slicer and finally printing their ring, giving the greatest ease of use.

5 Formal Syntax

```
<details>  ::= <ws> <color> <ws> <design> <ws> <sizespec> <ws> <size> <ws> <file>
<color>    ::= Gold | Silver | Black
<design>   ::= MoonAndStars | Band | Heart
<size>     ::= 5 | 6 | 7 | 8 | ... | 13
<sizespec> ::= "in size"
<file>     ::= any string ending in ".stl"
<ws>       ::= ␣ | epsilon
```

6 Semantics

Syntax	Type	Meaning
Color	string	a primitive type representing the color setting for the ring - chosen from options in the grammar
Design	string	a primitive type representing the design setting of the ring - also restricted to those available in the grammar
Size	int	possible ring sizes 5 - 13
Details	{ string,string,int,string }	a dictionary containing specifications for Color, Design, Size, and File
File	string	a file entered is a string ending in ”.stl”. This file will be written to by our language with the information specified by the user.

7 Remaining Work

We still need to create more ring template models. We currently only have three band designs modeled in Fusion360, but can create as many design templates as our time and imagination allows.
We also would like to adjust our evaluator slightly to make the band width around the finger standardized, even when scaling the rest of the ring