.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 object (.obj) file for a personalized ring that the user can then use with 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 .obj 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 OBJ file. This makes the process of obtaining an OBJ file much simpler, rather than having to create and modify the file from scratch.

We believe that this challenge can be solved through creating this programming language because 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 for the ring, which is represented by changes within the OBJ 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 .obj file created using Fusion360 and is then exported to an OBJ file. 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.obj”
Expected output: an OBJ file named ”ring.obj” representing the 3D object of a gold band ring in size 7, as specified by the user. Note: for all examples, we currently output a text file that contains information about the specifications provided by the user to confirm that our parsing works. The text file contains a dictionary of the ring specifications:

```
{ color = Gold "gold"
  design = Band "band"
  size = 7.0
  file = "ring.txt" }
```

dotnet run ”black cat ears in size 5 catring.obj”
Expected output: an OBJ file named ”catrings.obj” representing the 3D object of a black ring with little cat ears on it, in size 5.
Expected output for minimally working version:

```
{ color = Black "black"
  design = CatEars "cat ears"
  size = 5.0
  file = "catring.txt" }
```

dotnet run ”silver moon and stars in size 6.5 moon.obj”
Expected output: an OBJ file named ”moon.obj” representing the 3D object of a silver ring with moon and stars on it, in size 6.5
Expected output for minimally working version:

```
{ color = Silver "silver"
  design = MoonAndStars "moon and stars"
  size = 6.0
  file = "moon.txt" }
```

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 (float or 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 .obj 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 .obj 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>   ::= CatEars | Band
<size>     ::= 3 | 3.5 | 4 | 4.5 | ... | 13.5
<sizespec> ::= "in size"
<file>     ::= any string ending in ".txt"
<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	float	possible ring sizes including half sizes
Details	{ string,string,float,string }	a dictionary containing specifications for Color, Design, Size, and File
File	string	a file entered is a string ending in ”.obj”. 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 the band design modeled in Fusion360, but can create as many design templates as our time and imagination allows.

We are still working on the internal calculations to create the ring models. We have the backbone of the language, but now we must save the template OBJ files within the language that our language will make changes to. We need to make a scaling function that will edit the template OBJ for the design by a specific scale factor that is correlated with the ring size. We will have a scale value saved for each ring size, with the smallest size (3) as 1.0, and each other size will be the scale relative to that.

We also plan to create a special instructions file to be outputted alongside the OBJ file. This file will contain instructions for how to use the 3D printer in the Makerspace and will be personalized by instructing the user to load in the colored material that they specified.

Currently, our minimally working version writes to a .txt file. We are going to make a slight change to the parser and user input, in that we will be writing to and outputting an .obj file instead of a .txt file.