Spring 2018 Research Project Report

Web Interface: Design for Everyone

UCLA LEMUR Shuting Qing

1. Big Picture

Nowadays technology has approached our everyday life. With the help of different apps and tools, most of the things can be done on electronic devices. Professional applications like CAD make mass customization possible for general audience. However, there are still many difficulties for everyone to develop their own ideal objects.

This research aims at developing a developer-friendly and user-friendly interface. On one hand, this online platform provides a visual interface that allow users to design and customize objects themselves. Unlike some professional developing tools which requires a lot of complex process and precise adjustments, this interface is easy to use. On the other hand, developers no long need to work on front end code once they change into another project with a different interface for that the interface is generated automatically based on backend.

1. Environment Configuration

Developing tool: DWCC 2018

Library: WebGL, Bootstrap,

Scripts: JS, HTML, jQuery

Others: Backend system, backend output

Tool Introduction:

The tool I use is DW. One advantage is that it allows developers to have real-time view of how your website will look like. It’s mostly the same as you it on browsers, only if it comes to compatibility differences on various browsers or other problems from some libraries. Honestly, it will save you a lot of time.

1. How to Run the Code

Step 1:

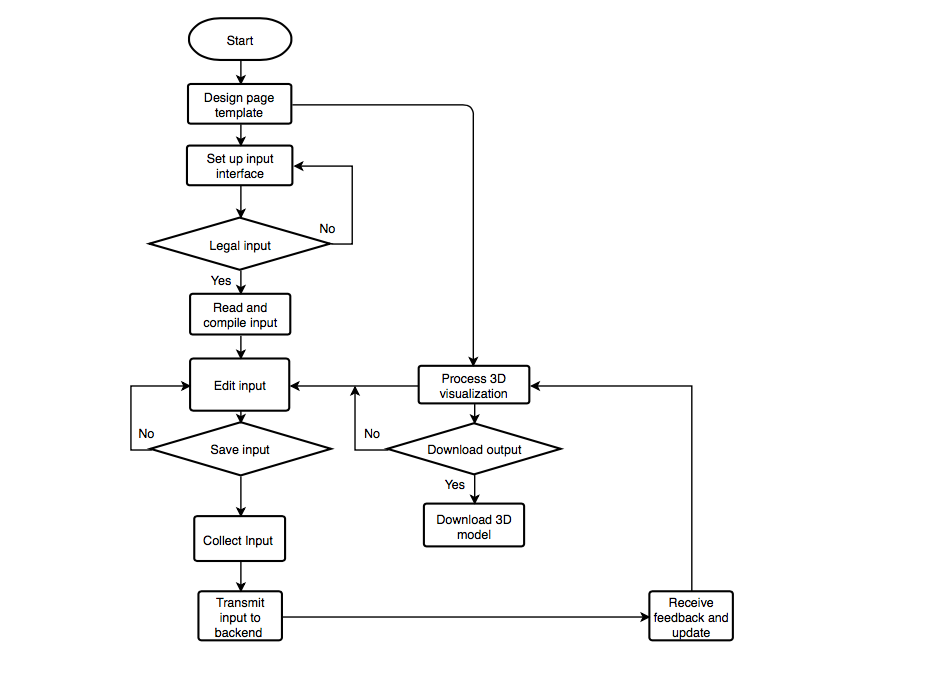
Make sure you’ve got a DW. My version is DWCC 2018 though I think most of other versions also work. And then you can download it directly from GitHub. All the extension libraries are already in the folder. So make sure you download them together.

Step 2:

Click the 15.html file directly in order to run it on a browser. You should have a view of 3D model on the right side of the website. If not, please follow:

* Check WebGL Compatibility. You can just click this link if you need. <https://threejs.org/docs/#manual/introduction/WebGL-compatibility-check> <https://caniuse.com/#feat=webgl>
* Check the folder static is complete.
* Check whether the backend links works. This link is written in 15js.js file. You may found : init(“[http://ayeaye.ee.ucla.edu/stool.stl?height=50&legs=3&radius=30”](http://ayeaye.ee.ucla.edu/stool.stl?height=50&legs=3&radius=30%E2%80%9D)). Since this link might be changed, but this statement will be still.
* Open 15.html in Google Chrome which allow users to debug to figure out where exactly the problem is. Sometimes it’s only because there some bugs make it stuck which leads to failure of other functions.

Step 3:

Upload a input file. 1. Only .txt file is allowed to uploaded. 2. If it doesn’t generate controls successfully, please check whether the input format is correct. Format grammar will be introduced more detailed later. 3. If there is a error reminder after you upload the file, just fix the format error and refresh the website again before you upload a correct txt file.

1. Developing Process
2. Technical Analysis

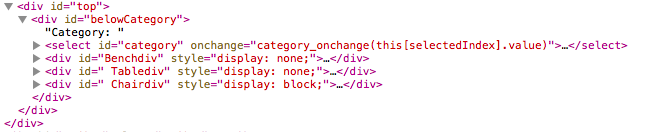
Summary:

This project aims at developing a automatically generated web interface. All the components shown in this interface may change with back-end development without coding on front-end. Besides, it applies a real-time 3D visualization based on the parameter changes of the components it generated before. So this is not only a user-friendly interface, but also a developer-friendly interface.

Technical implementation:

* **Template**: Since most of divs the website is about to create are unknown, a clear design solving problems like where to create and how to create is a must. A easy and clear design.

1. Div ‘top’ is a general one used to put all the automatically generated labels and controls.
2. Div ‘belowCategory’ is used to put the category select and visible and invisible layers.
3. For each category, there is a div created to put corresponding controls as to realize switching between categories.

* **Input interface**: It’s is a bridge between back-end and front-end. Most of the controls and labels shown on the front-end is generated from a back-end output file which is supposed to be uploaded here.

1. File type limitation. So far, txt file is the only legal input type. Once users click upload, they can only choose txt files as input.
2. File information: File name will be shown after it is successfully upload.

* **Input file**: There is a special grammar of this input file in order to help the front-end distinguish what and where to create controls.

1. General grammar:

Category1;

\*Inner Content\*

~

Category2;

\*Inner Content\*

~

Category3;

\*Inner Content\*

~

... ...

1. Grammar for inner content:

Format of \*Inner Content\*:

\* textbox, label for this textbox: lower bound, upper bound;

\* dropdown, label for this dropdown, num n of all choices, choice 1, choice 2, choice 3,...., choice n;

1. Blank space is permitted. File compiler will filter blank space if is needed.

* **Online file compiler:** This is no doubt the core of this project. In order to generate front-end automatically, file compiler needs to read input content, distinguish, process and translate the content into html language.

1. Classify: Faced with the whole input content, the first step of the compiler is to separate content by category in terms of classifying information belongs to each category orderly.
2. Distinguish: Once the compiler goes into the loop of each category, the next step is to distinguish the type, name, and other special attributes of a control.
3. ID allocation: In order to call these controls later, each one should be allocated a exclusive ID. For the uncertainty of a dynamic control, there is a format for id allocation.

Format:

Its category + control function + row in the content of this category.

Examples: (Refer to given input example in GitHub）

Legs: Benchdropdown3

Seat width: Chairtextbox5

Connections(under Table): Tabledropdown9

Connections(under Chair): Chairdropdown10

1. Error reminder: For input errors, the compiler will detect and give precise error messages to users.

* **Front-end changes listener:** It’s used to listen user’s changes of the parameters the made in the front-end. It needs to locate which category is user browsing and making changes, confirm controls of this page, extract data and save them.

1. Locate the category: Check the value of category select control so as to locate the which part is user making changes on.
2. Confirm related controls: According to the category user selected, confirm id of corresponding controls instead of listening and traversing all the controls which may help generate information precisely and decrease time complexity as well.
3. Calculate id : For every control, there’s a unique format of id generation. To extract information, the first thing is to get the id.
4. Extract information: Distinguish the type of a control and then correspondingly choose a proper way to extract information.

* **ID generalization:**

The id of a control:

Its category + control function + row in the content of this category

For example:

Legs: Benchdropdown3

Seat width: Chairtextbox5

Connections(under Table): Tabledropdown9

Connections(under Chair): Chairdropdown10

* **3D model visualization:**

1. Tool: WebGL
2. Bug Analysis
3. Future Development
4. Personal and Academic Growth