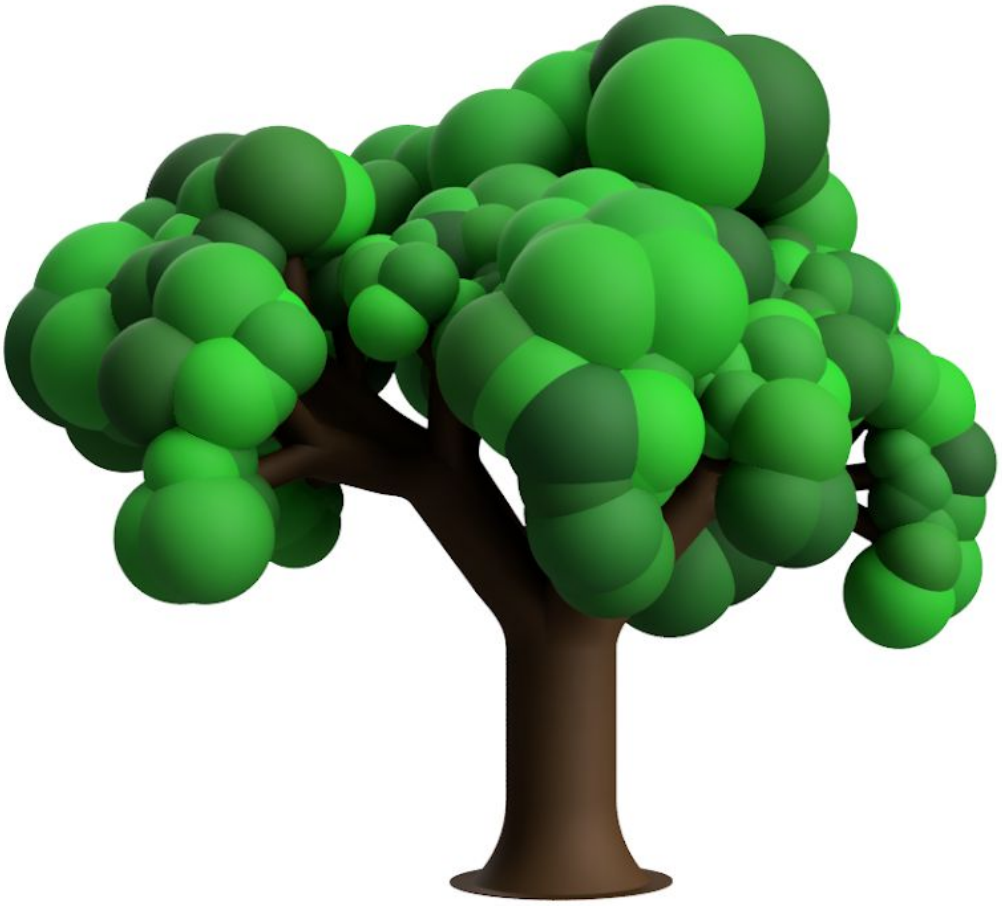
EnviroGen - Beta Plugin

Group 3
Simon Glimm 335266
Hasan Imran 407380



Description

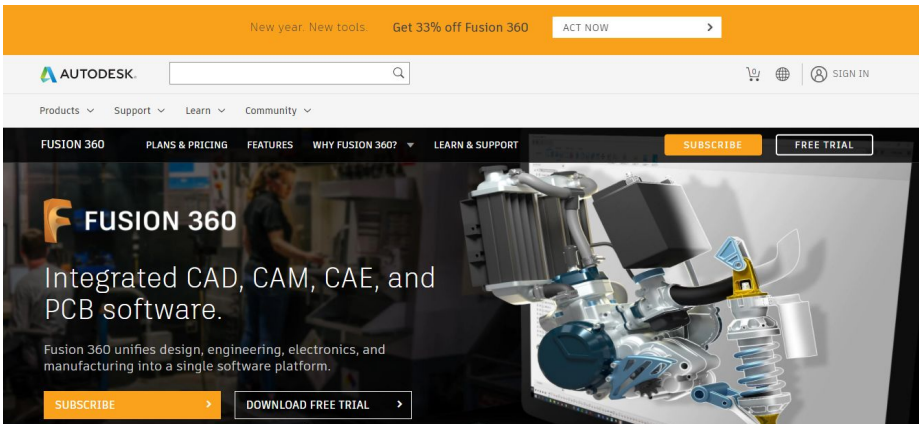
EnviroGen is a one-stop solution to add trees to the environment of your project. Be it a number of simple trees or a detailed tree, EnviroGen has you covered. Run our Add-In on your Fusion project, select the place where you want to have a tree, the size and done! Your very own tree, uniquely generated every time, is ready to make your project all the more beautiful. There are also customisation options available to fit the trees to your demands. Give it a go!

Table of Contents

Installation

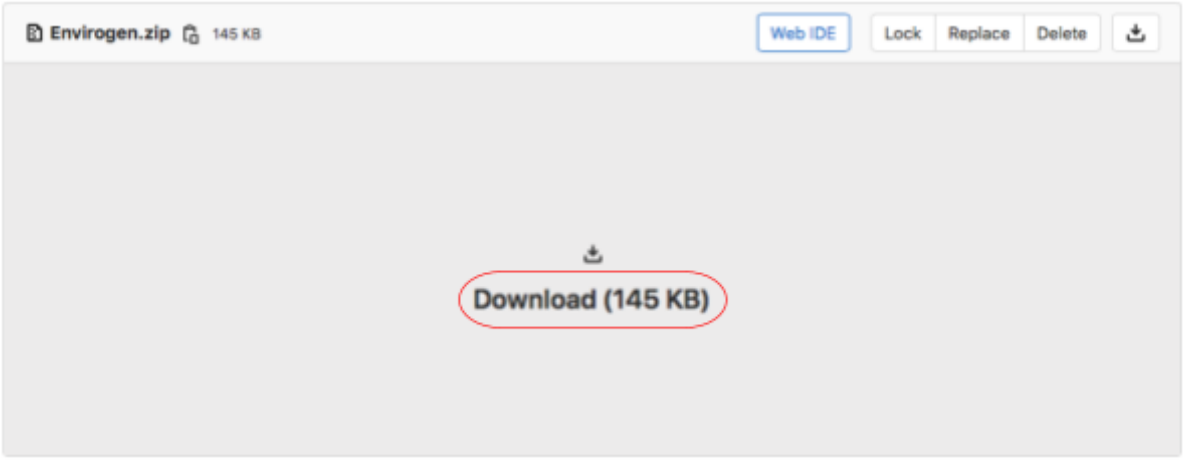
1. Download Fusion

If Fusion is not already installed, download and install Fusion from:
https://www.autodesk.com/products/fusion-360/overview



2. Download the EnviroGen plugin

Download the Plug-In from
https://git.rwth-aachen.de/simon.glimm/envirogen/-/blob/master/Envirogen.zip
and unzip the folder.

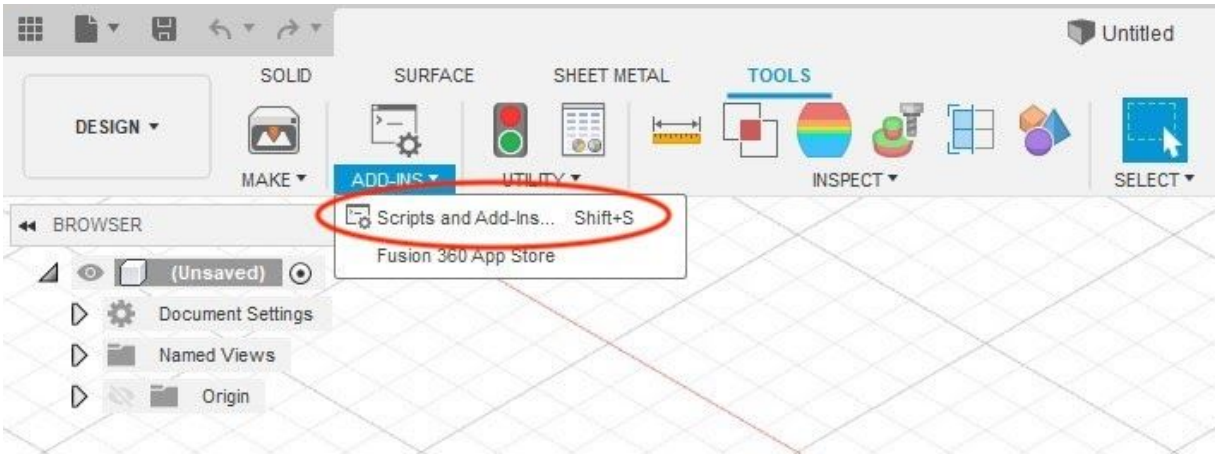


3. Enable the Add-In

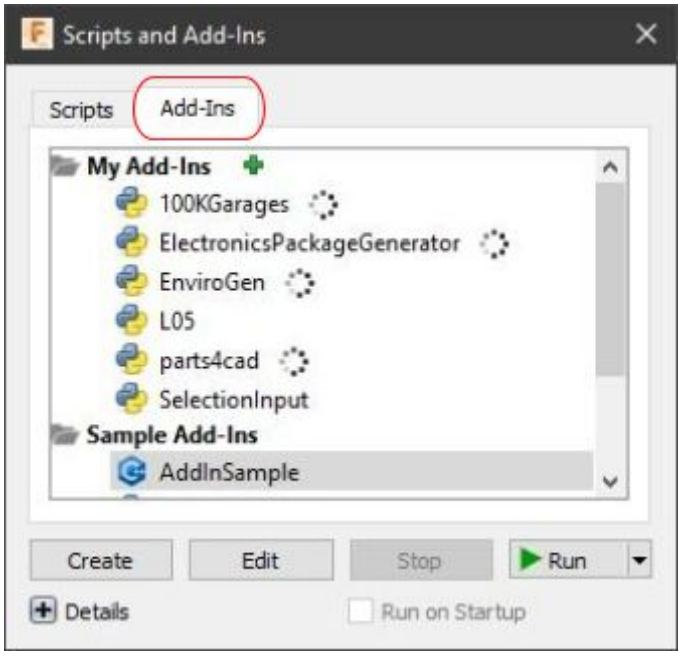
The screenshots were taken on Windows but the process is the same on MacOS

- by going to “TOOLS” in the top bar

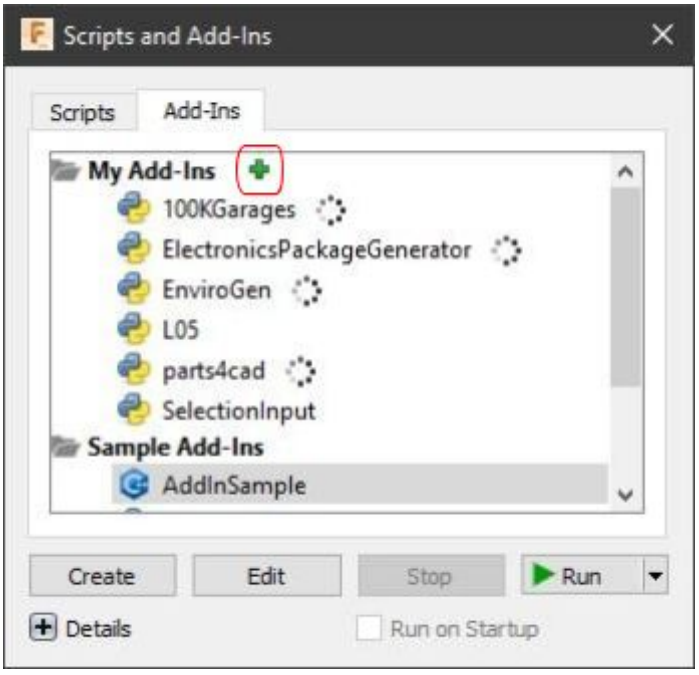
- then “Scripts and Add-Ins…” in the ADD-INS Menu



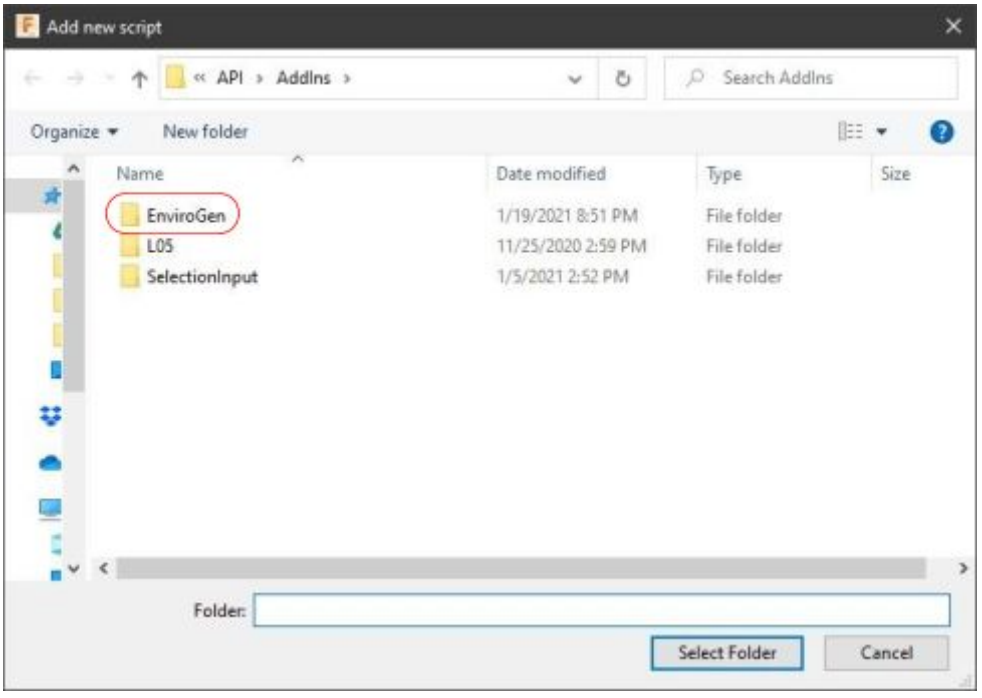
- Klick on the rider Add-Ins



- Select the plus sign next to the My Add-Ins heading

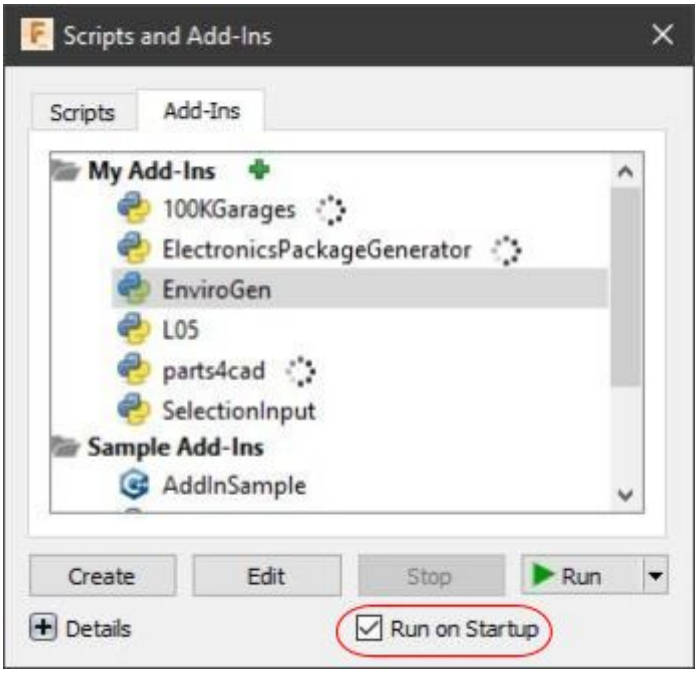


- Navigate to where you saved the EnviroGen folder and select the folder in the file explorer



- Select “EnviroGen” from the folder “My Add-Ins”

- Tick “Run on Startup” if you don’t want to repeat this process in the future



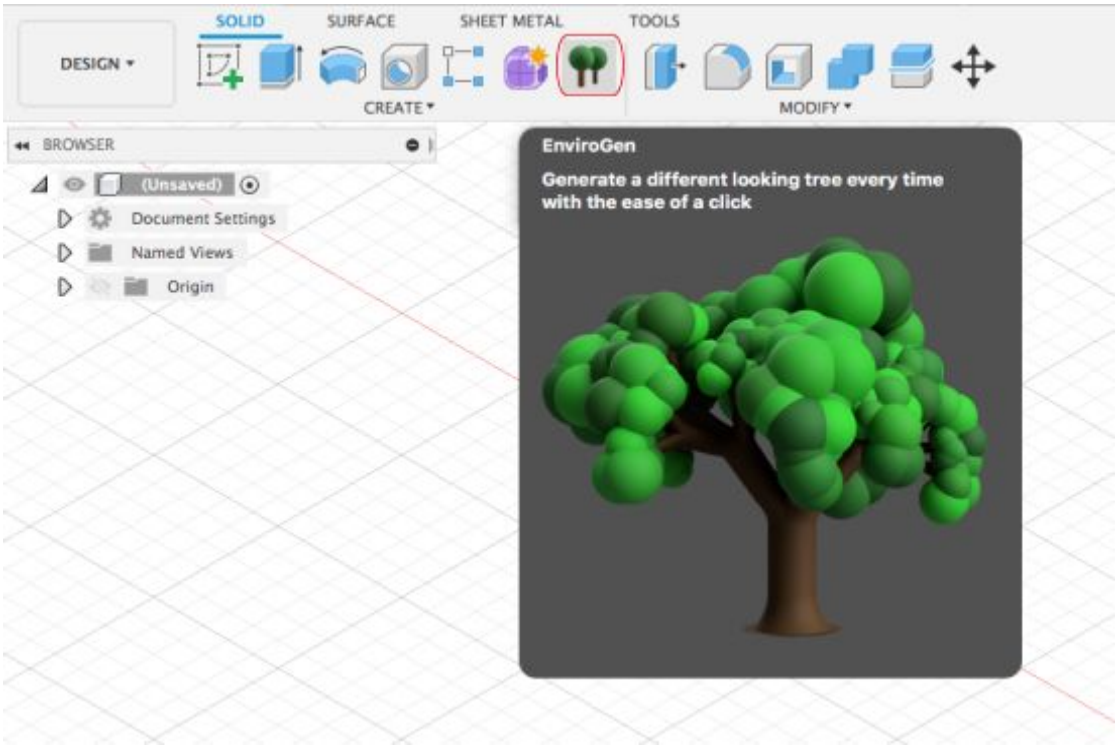
- Hit run.

4. Run the plugin

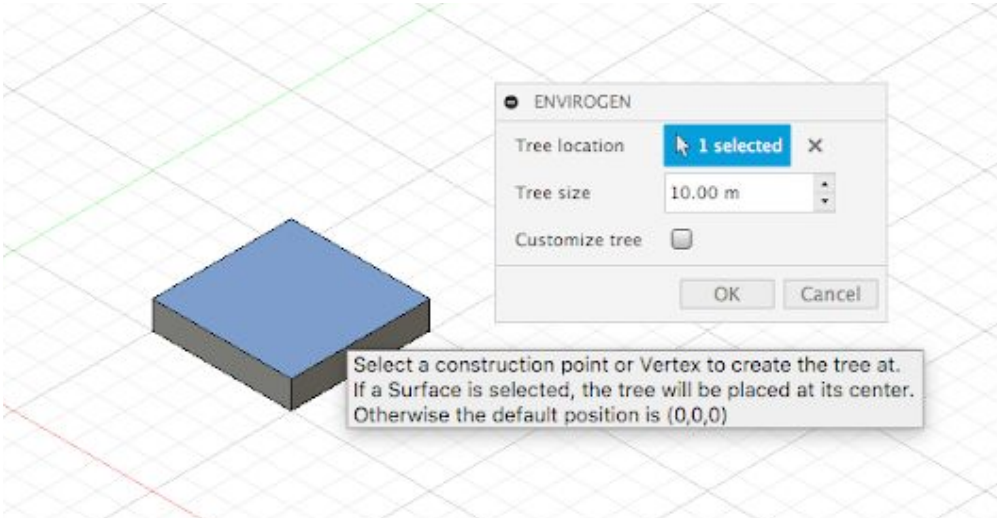
You will find the Plug-In under the “SOLID” menu in the category “CREATE”

Usage

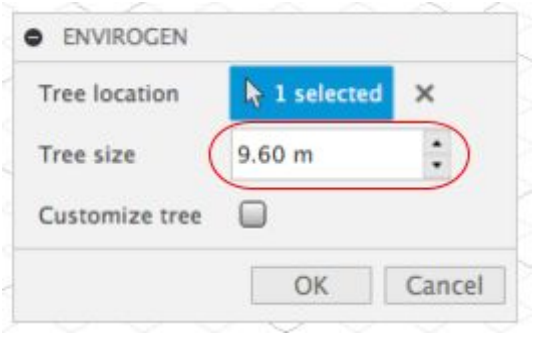
1. Start the EnviroGen Plug-In from the category “CREATE” in the “SOLID” menu.



2. Select the location where the tree will be generated. It can be:
    - A construction point / vertex
    - Or a surface (the tree will be placed on its centroid)
    - Or none (the tree will be generated at the origin [0,0,0] of the Design)

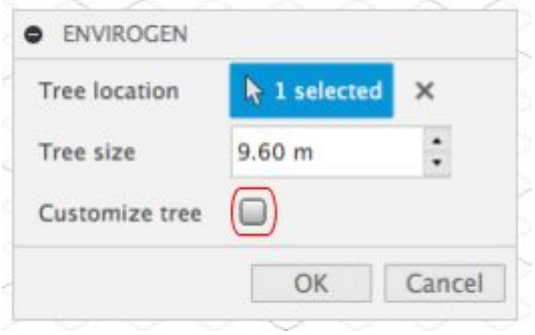


3. Choose a height for the tree.
    - Due to the random generation of the trees, the final height might deviate.

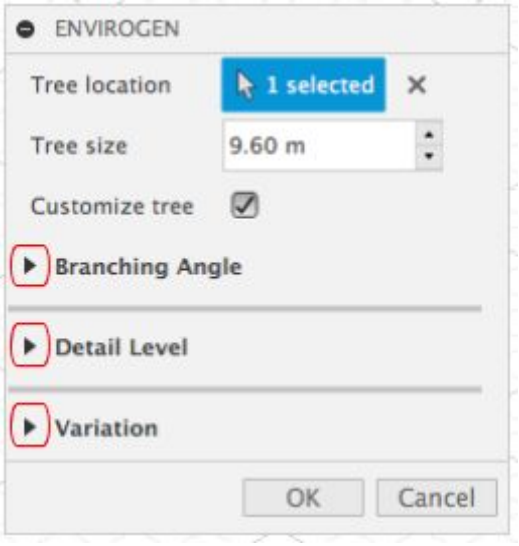


4. If you do not wish to customize the tree further, skip to step 8.

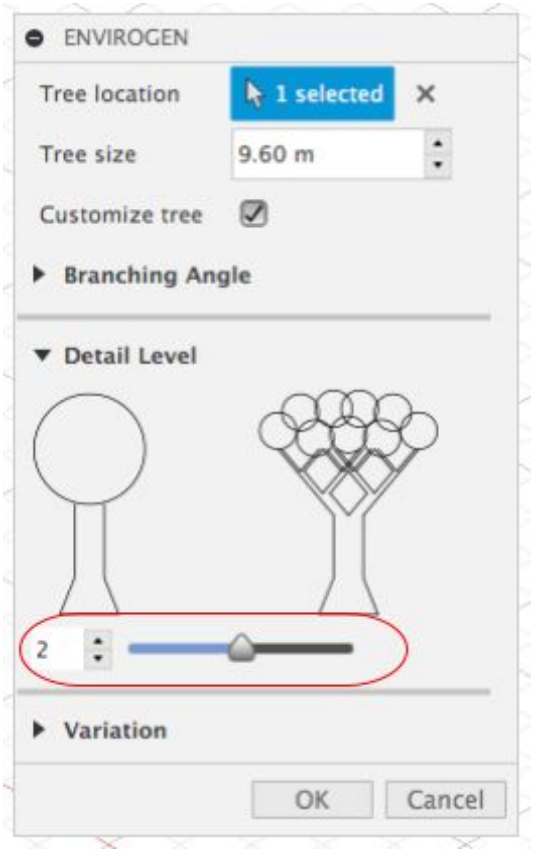
5. Tick the box “Customize tree”. More options will show up in the menu.



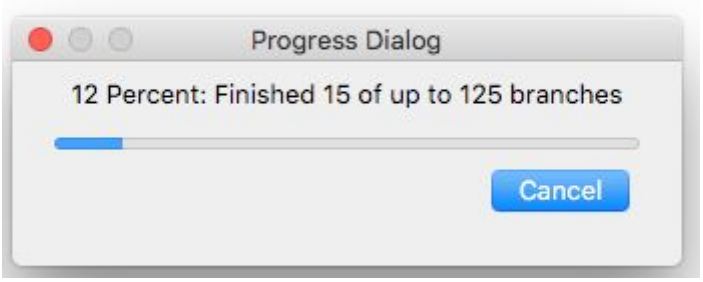
6. To change a parameter, click on the arrow next to its name to expand the view



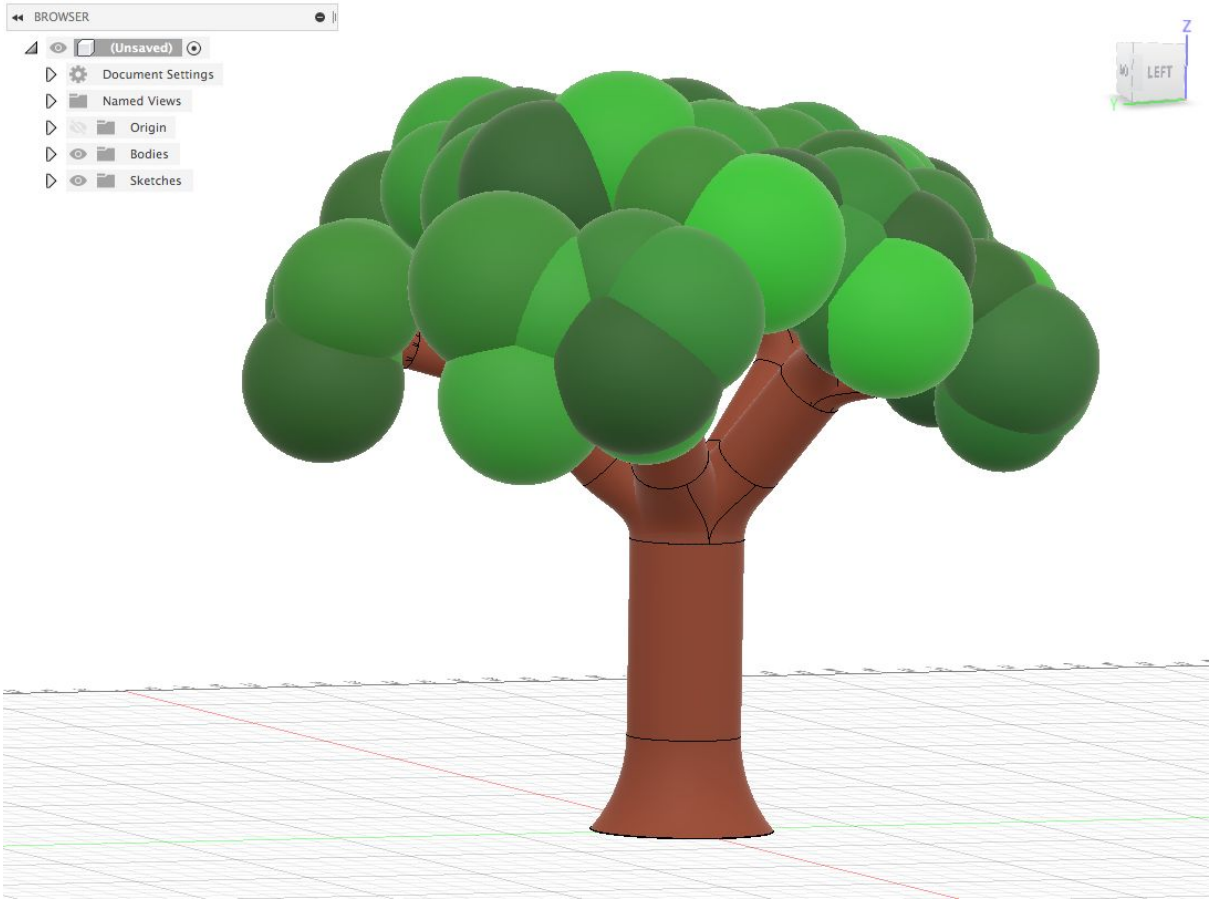
7. Adjust the sliders according to your preferences. Tooltips and Graphics will explain the effect on the generated trees.



8. Click “OK” to generate a tree or cancel to abort. A progress dialog will show up. Due to the random generation, this is an estimation and the process might finish before reaching 100%. This is intended and does not mean the tree was not finished.



9. The tree is generated.



10. If you don’t like that particular tree, just hit undo and generate a new one. Every tree is unique and will look different.

Limitations

1. Efficiency

Generating more and more bodies takes longer time which increases exponentially. This means that each new object takes more time to create than the previous object, so creating 100 objects takes significantly more than double the time it takes to create 50 objects. Too many bodies will lead to Fusion crashing. Combining all the individual bodies from the branches sometimes leads to errors because, as other reports suggest, sometimes bodies are not recognized as touching but barely apart. Since it was unreliable and sometimes throwing errors, we did not use it.

2. Images in the UI

The images are of fixed size, so they need to be manually configured to fit the UI box. Since they don't scale automatically, depending on the size of window, or resolution of display, they don't line up exactly with the sliders.

3. Selecting location

It is only possible to select predefined Fusion objects, such as faces, edges and concentration points etc. It would have been nice to be able to select a point anywhere on the plane or on a surface for the tree. However the API doesn't offer this functionality.

4. Leaves

We did not include more realistic leaves as a result of body limitations. Since Fusion is already struggling with the complexity of the generated trees as is, adding many leaf shaped objects instead of a sphere would increase the delay in the generation of the tree even further.

5. User facing limitations

a. The time needed to create detailed trees can become very long.
b. Exact placement of the tree is not as straightforward as we would like it to be. Adding a construction point can be used as a workaround to select an exact position.
c. If we set the branching depth to greater than 4 Fusion crashes due to the complexity of the model. Therefore creating extremely detailed trees is not possible.
d. When the generation process is canceled, the user is left with a half finished tree and has to manually click undo to delete it.

Design Decisions

1. Trunk of the tree

To simulate a real tree, the trunk is designed in three parts. The first is a base which is a flat disc on the plane of construction. The second part is the cylinder which is the trunk on which the branches are connected. To join the disc and the cylinder a funnel is created on top of the base and connects the lower face of the cylinder.

2. Fractal algorithm

To implement the fractal tree approach, we used recursion to create the branches for the tree. Each branch generates between 3 to 5 random splits for the branches. Then each split calls the recursion again until the required depth is reached. This approach enables us to have a consistent behaviour between all stages of the tree, which is defined by parameters that get passed on.

3. Tree leaves

For the tree leaves, we decided to abstract them using a sphere. This left us more time to focus on achieving a realistic shape for the rest of the tree and most importantly helps keeping the processing time and performance of Fusion in check. We expect our Plug-In to be used mainly to generate trees that add to an existing environment instead of creating a tree to watch in detail. Therefore we feel like this is an acceptable compromise.

4. User Interface

When deciding which customisation options to offer to the user, we tried not to overwhelm with too many and confusion parameters, so we decided to stay with a few parameters to adjust the size, detail and general shapes of the trees. Since we introduce a lot of (variable) randomness to the trees to create a unique looking tree every time, too detailed or fine controls would not necessarily reflect in the end result anyways. For similar reasons we decided not to use precise values in some parameters, since they don’t represent actual units and offer a slider with intuitive descriptions instead.

5. Progress Bar

Since the creation of the tree is random, we don’t know for sure how many bodies will be created in total. Therefore we have to use an estimation. Since it’s not advisable to make the user wait longer than expected and he might think the program has crashed, we considered the highest possible number as a base for our estimation. This way the program might finish faster than expected, not slower which is a better experience.

Working Procedure

After running the add-in, a dialog box window pops up with the options of selecting the tree location, the tree size and a check box for customizability. The location of the tree can be selected by a simple mouse click for convenience. The size of the tree is currently in meters and also supports float values so the user can accurately model the tree size to the scale of his project. If the user ticks the box for customizability, the UI reacts to the changed input, and the dialogue box drops down with more options of setting the branching depth, the variations in the branches and the angle of the branches. Once all the parameters have been set and the user has pressed OK the dialogue box closes.

The location to place the tree is determined, depending on the selected input. Values are pulled from the inputs and processed or converted to fit to the next steps of the program and passed on to the function that will start the actual creation of the tree.

A progress bar is displayed showing the progress of the creation of the tree.

As the first step in the tree generation, we create the trunk of the tree. This is done by creating a circle and extruding it to a cylinder, which is the trunk itself. The cylinder is colored with a randomly generated shade of brown. A wider base is then added on the bottom, on which first a chamfer and then a fillet are added to create a flared base to add more realism to the trunk shape.

In case of the simplest trees, the cylinder is elongated further and a big sphere placed on top of it to represent the canopy. Otherwise the branching process is started.

This is done using two functions working in a recursion where one is calling and defining the splits, and the other function is creating the actual branches.
The first one, callSplits, decides on how many branches to create, defines the necessary angles, new sizes etc and passes the parameters necessary to create the branches and calls for their creation.
If the desired recursion depth has been reached, it instead calls for the creation of a sphere to represent the leaves at the end of the branch via addLeaves.
The second one, createBranch, creates a cylinder with an offset that will be the next branch. It then rotates the branch into the desired position and adds a loft body in between this new branch and the one it originates from to create a smooth transition. It then uses callSplits to add new branches to its end.

The addLeaves function is called to create a sphere and color it in a new random shade of green at the end of each branch to depict the leaves. It also updates the progress bar, as each call represents one finished branch.