AUTODESK. UNIVERSITY

Generative Design At Hogwarts: Using Tech Instead of Magic

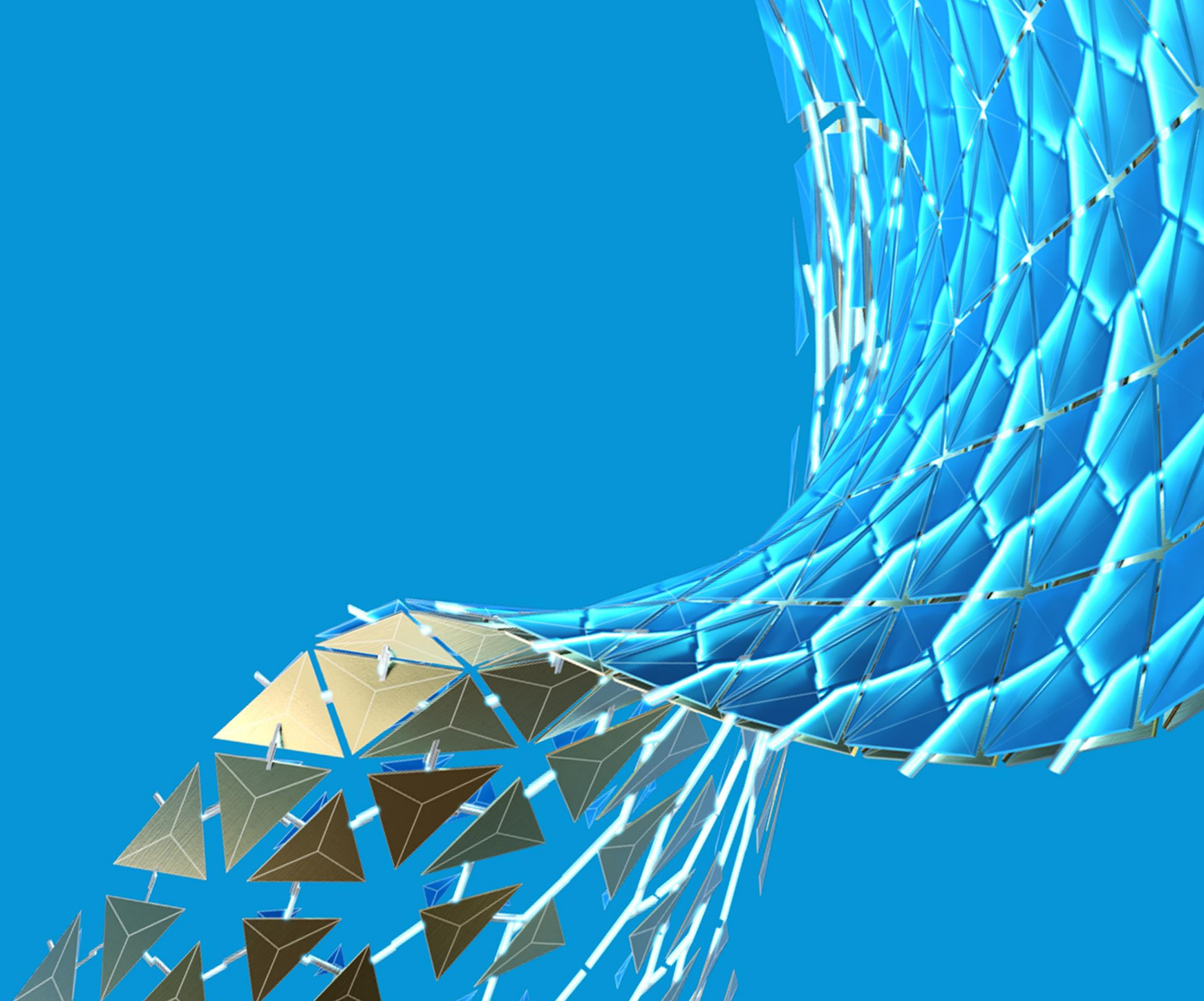
Jacob Small

Designated Support Specialist | Twitter: @JacobWSmall

Alexandra Nelson

Design Technology Specialist | Twitter: @AlexNelson_DT

Welcome!



About the speaker

Jacob Small

After attaining his Bachelor of Architecture degree from Wentworth Institute of Technology, Jacob Small began his work at a small firm on the north shore of Massachusetts, before joining CBT architects in Boston where he gained exposure to larger scale projects. In 2017 he joined Autodesk as a Designated Support Specialist, where he puts his 10+ years of experience in the AEC industry and expertise with Revit, AutoCAD, and Dynamo to help enterprise priority customers adopt computational design into their daily workflows.

About the speaker

Alexandra Nelson

Alexandra Nelson currently works as a member of the Design Technology team at Perkins Eastman where she acts as a firm-wide representative focused on advancing the efficacy of the firms design initiatives in the execution of its projects and global design technology strategy. She previously worked at Grimshaw Architects as one of two BIM specialists in the New York office. Her research is focused around automation and data collection from Revit models, with a current focus in space analytics and generative design.

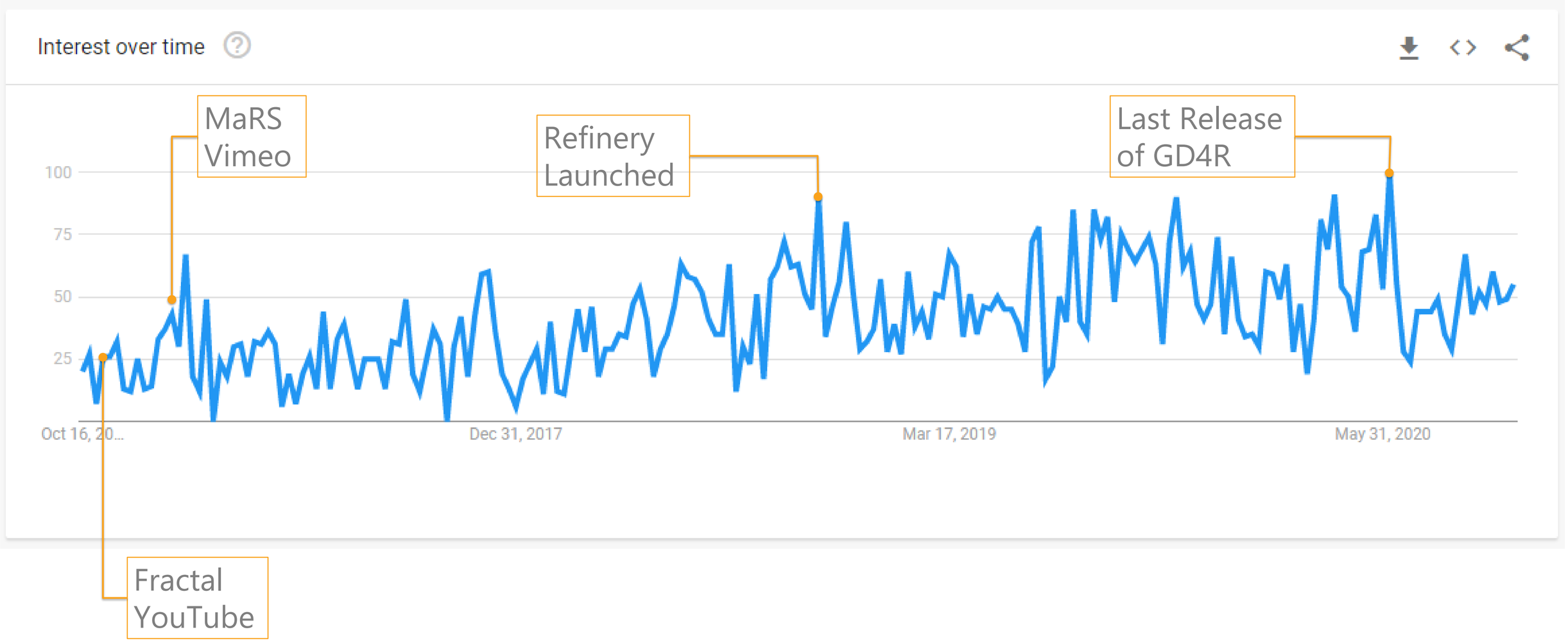
Safe Harbor

During the course of this presentation, we may make statements regarding future events and/or statements regarding planned or future development efforts for our existing or new products and services. We wish to caution you that such statements reflect our current expectations, estimates and assumptions based on factors currently known to us and that actual events or results could differ materially. Also, these statements are not intended to be a promise or guarantee of future delivery of products, services or features but merely reflect our current plans, which may change. Purchasing decisions should not be made based upon reliance on these statements. The statements made in this presentation are being made as of the time and date of its live presentation. We do not assume any obligation to update any statements we make to reflect events that occur or circumstances that exist after the date of this presentation.

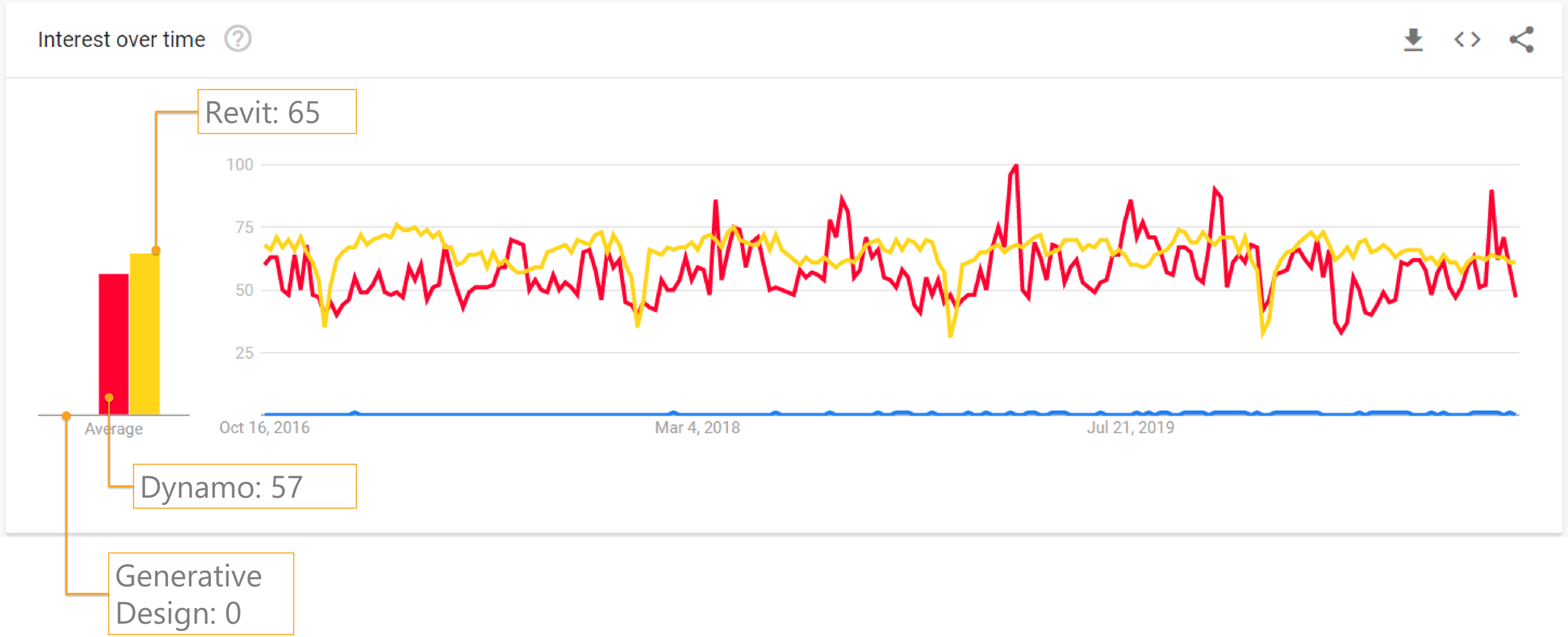
AGENDA

- **Welcome to Hogwarts**
  - Introductions, outline, goals, & notes
  - A Magic for Muggles
  - This is an O.W.L. Course
  - Learning Objectives
- **Taxonomy of Generative Design Graphs**
  - How the Magic Works
  - Graph Anatomy: Automation vs Generative
  - Remembering Static Values
  - Generation Systems
  - Evaluations & Results
- **Problem 1: Professor Snape's Seating Chart**
  - Assignment Studies
  - Permutations Explained
  - Graph Build
  - Generation & Results
- **Problem 2: Location for the Quidditch World Cup**
  - Fixed Option Studies
  - Pseudo Random Shuffle
  - Graph Build
  - Generation & Results
- **Problem 3: Layout of the Care of Magical Creatures OWL**
  - Flexible Option Studies
  - The Infinite Scale Problem
  - Graph Build
  - Generation & Results
- **Resources and Next Steps**
  - Notes and Review
  - Links
  - Development Team Notes
  - Office Hours
  - Special Thanks

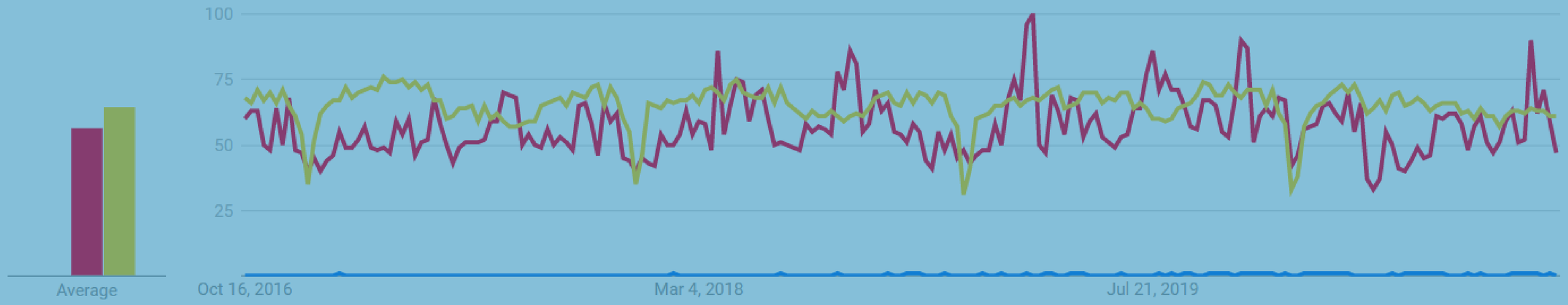
A Magic for Muggles



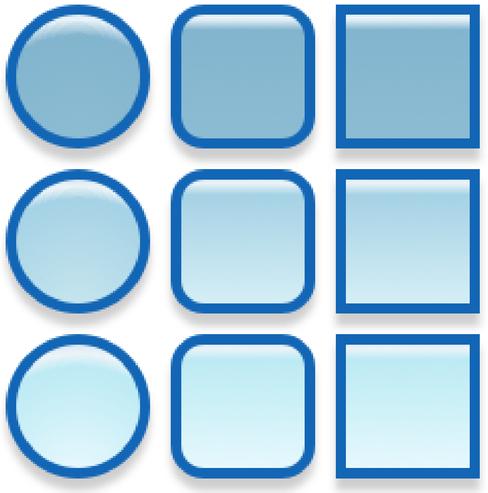
A Magic for Muggles?



WHY?



This is an O.W.L. Class!



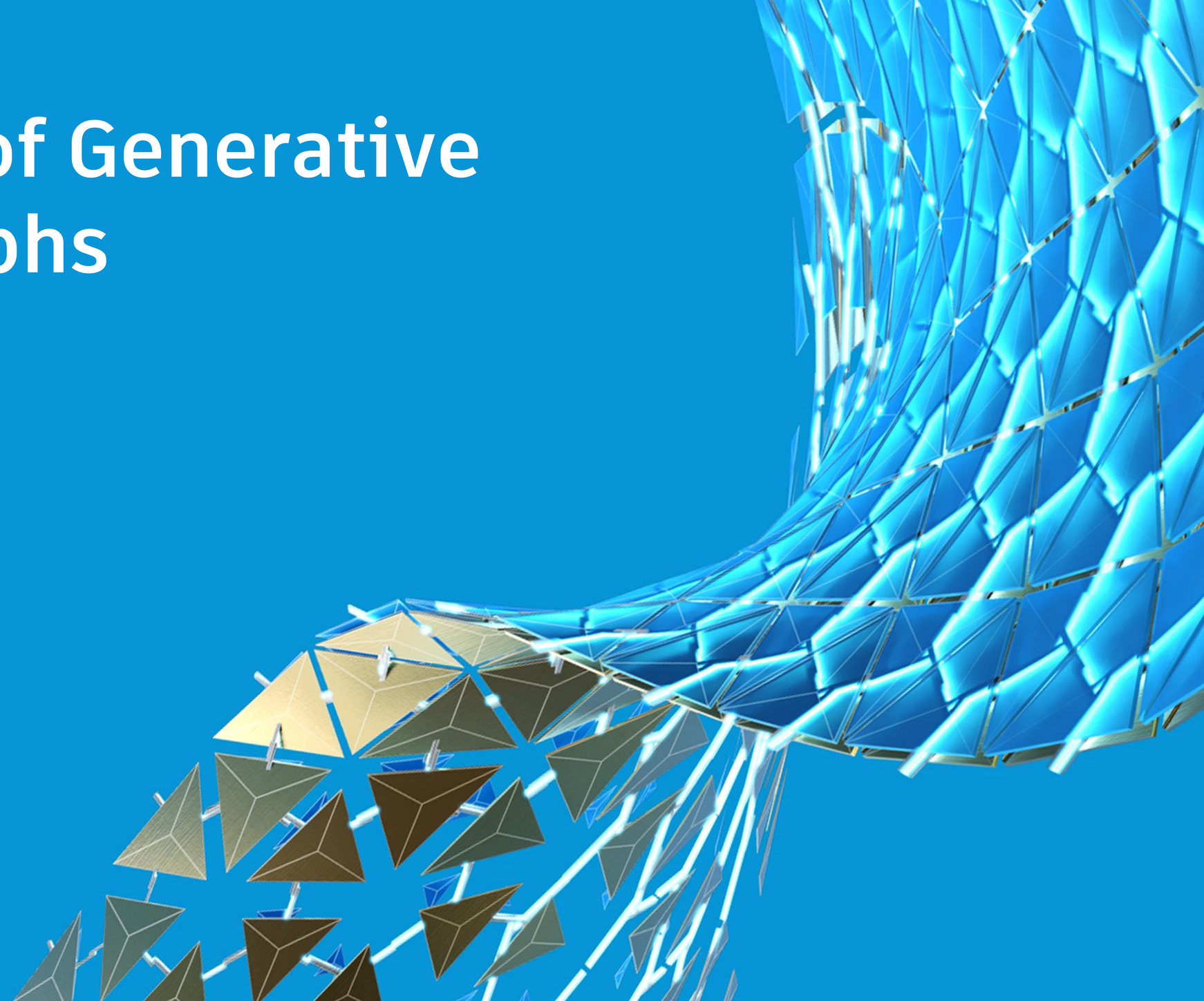
!=

Learning Objectives

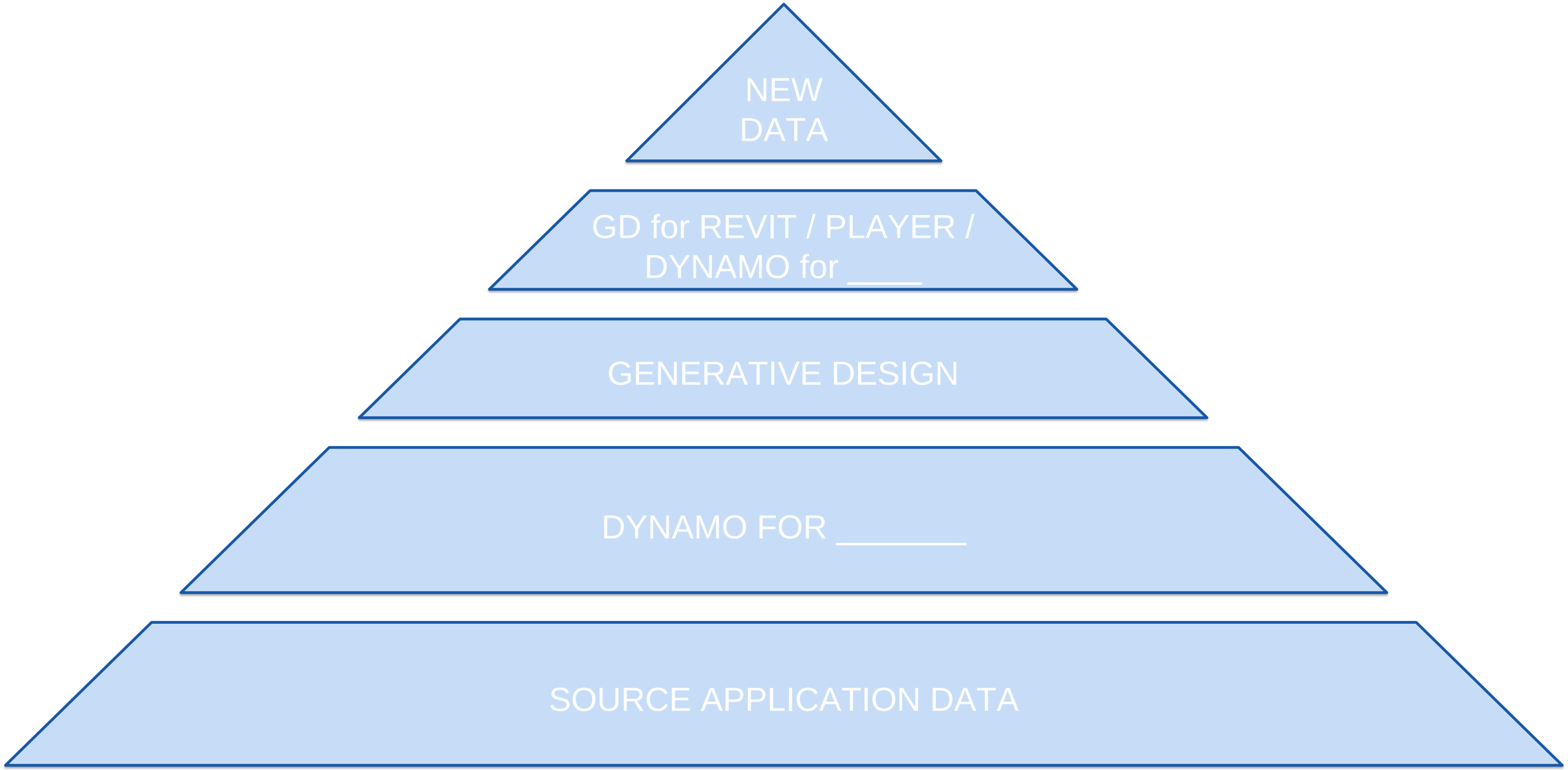
Upon completion of this class you should be able to:

1. Create optimal solutions when faced with competing goals and inputs by using Generative Design.
2. Build viable quantification systems in both relative and finite scope to define successful results.
3. Compare multiple viable solutions at various scales to ensure projects move forward with confidence.
4. Present a sampling of options for review/decision without requiring significant rework.
5. Find your way to Platform 9 ¾.

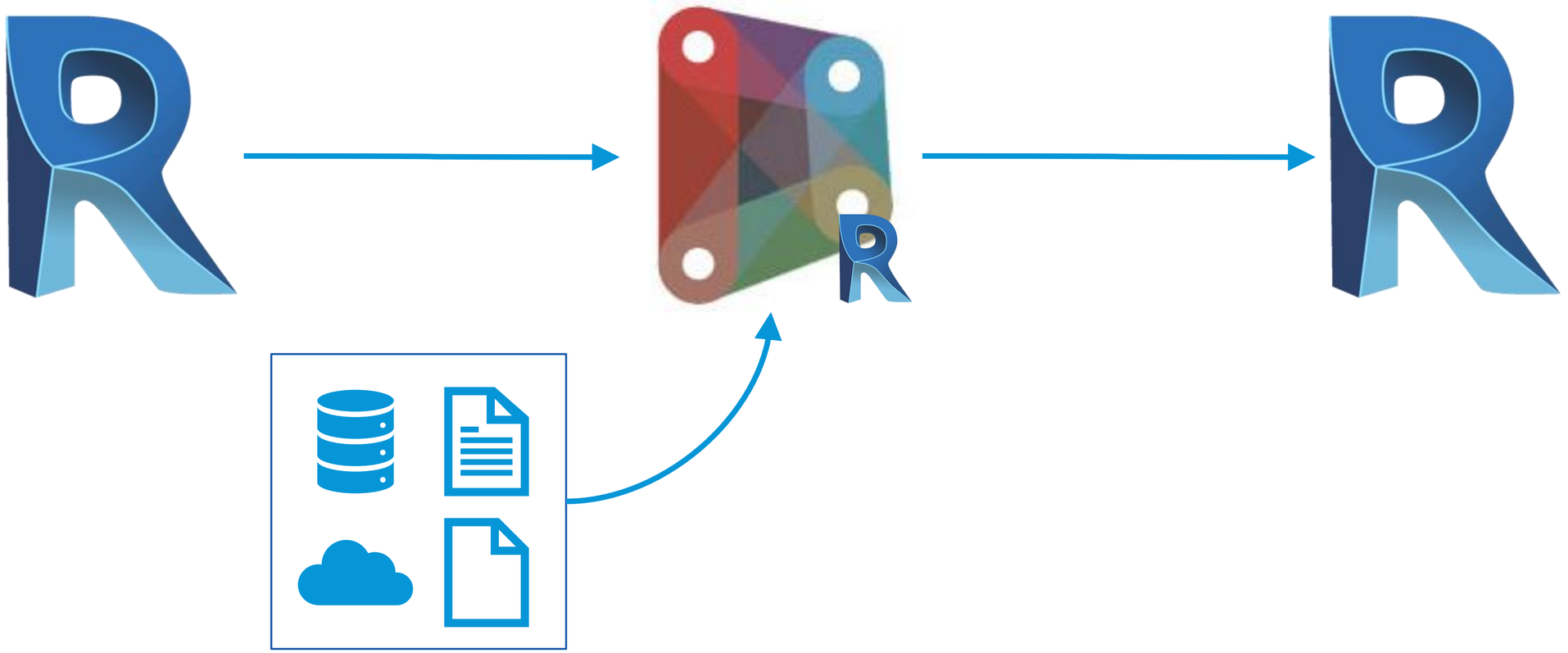
Taxonomy of Generative Design Graphs



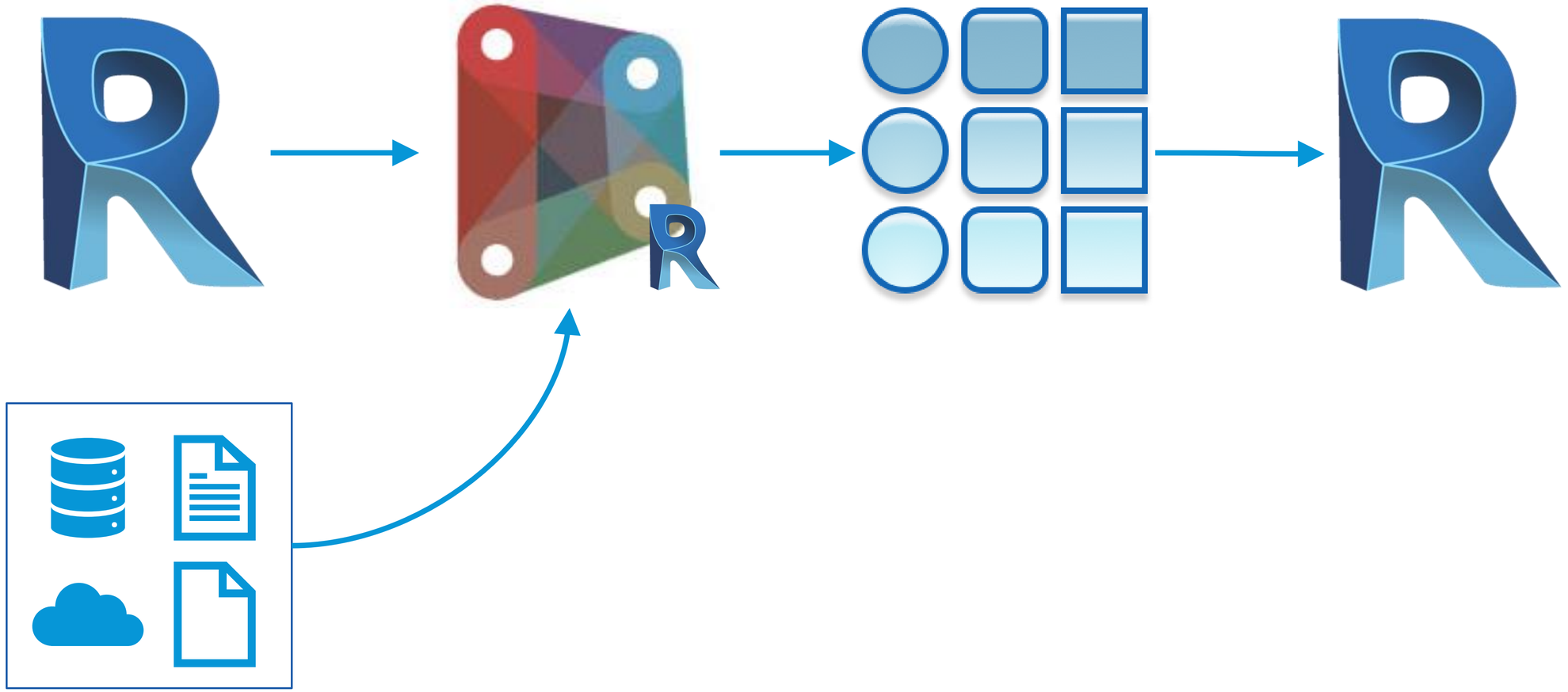
How the Magic Works



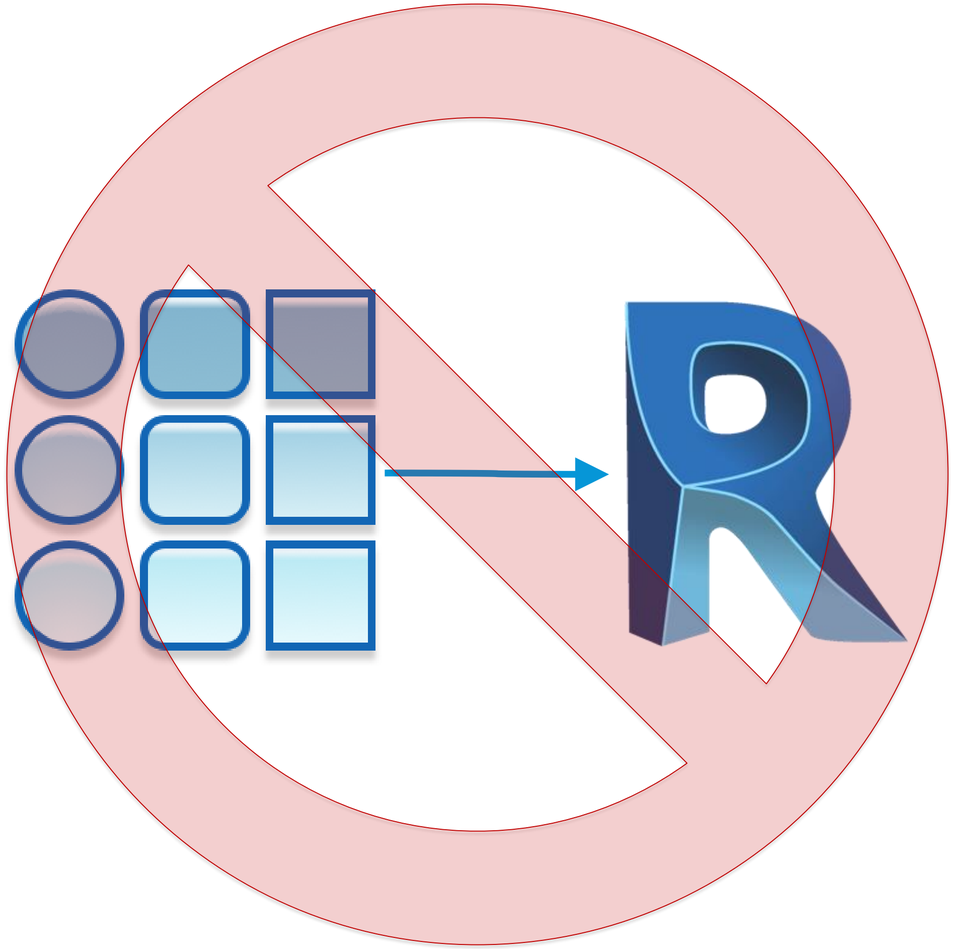
How the Magic Works



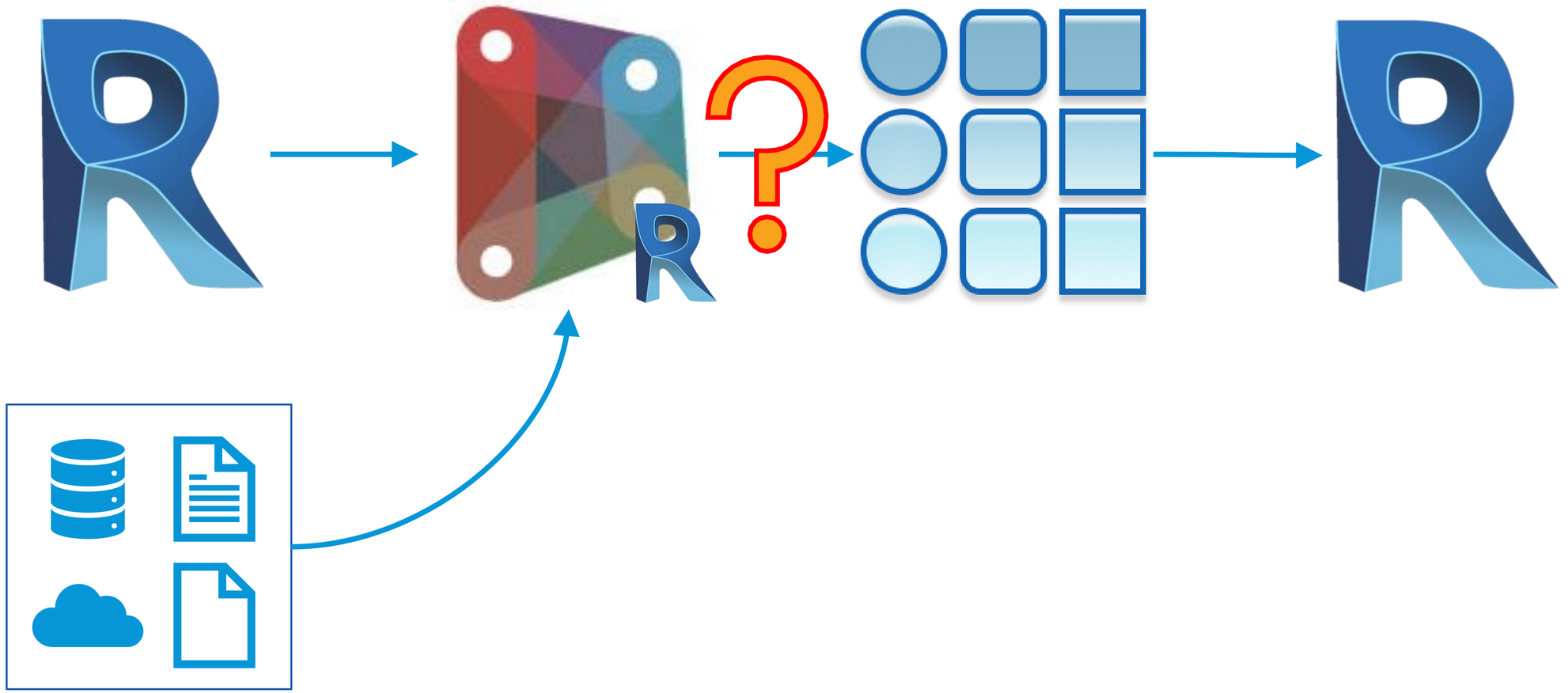
How the Magic Works



NOT How the Magic Works



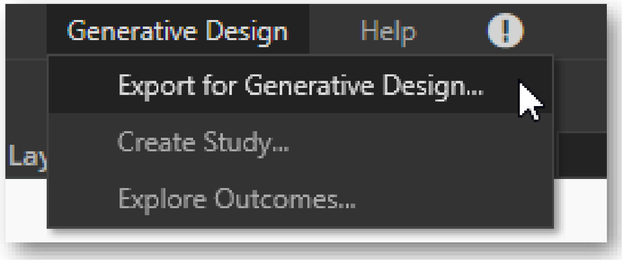
How the Magic Works



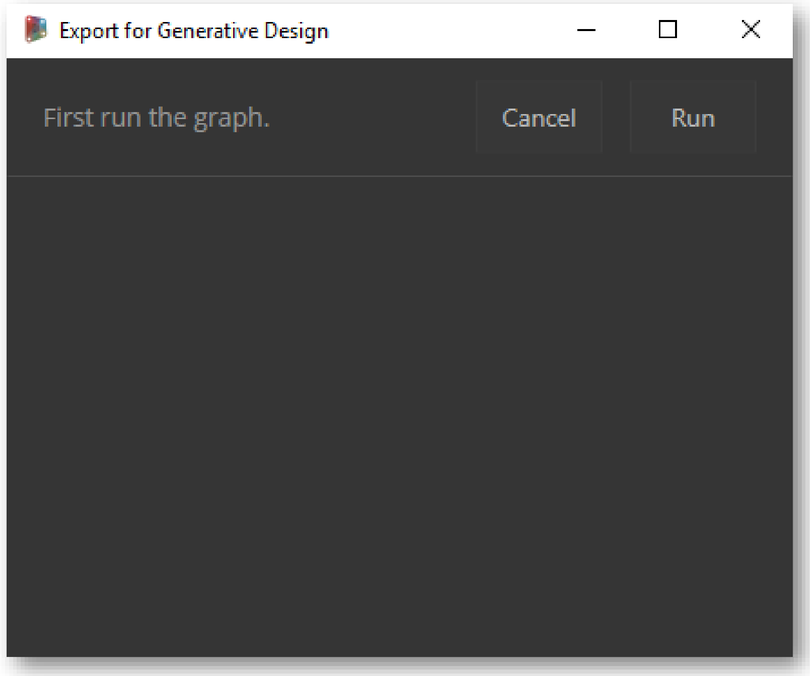
How the Magic Works

Launch GD Dynamo Script

Export for GD from Dynamo

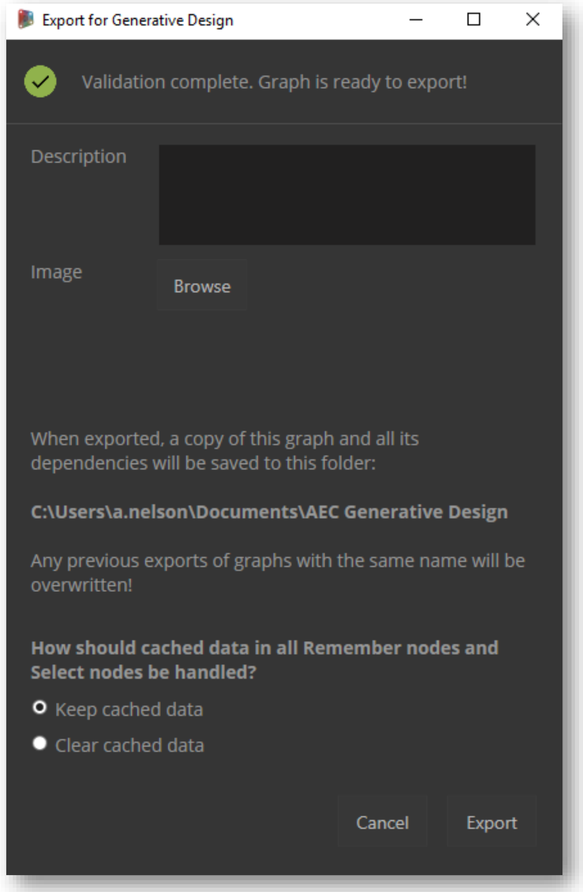


Click “Run” to Run the Graph for GD

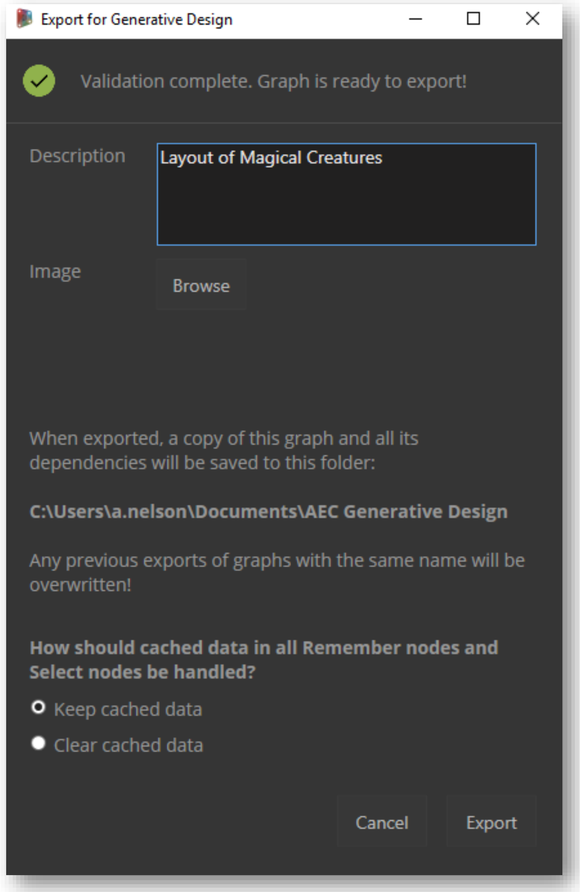


How the Magic Works

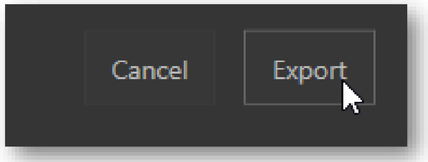
Graph is now ready to export!



Enter a Description for the Graph

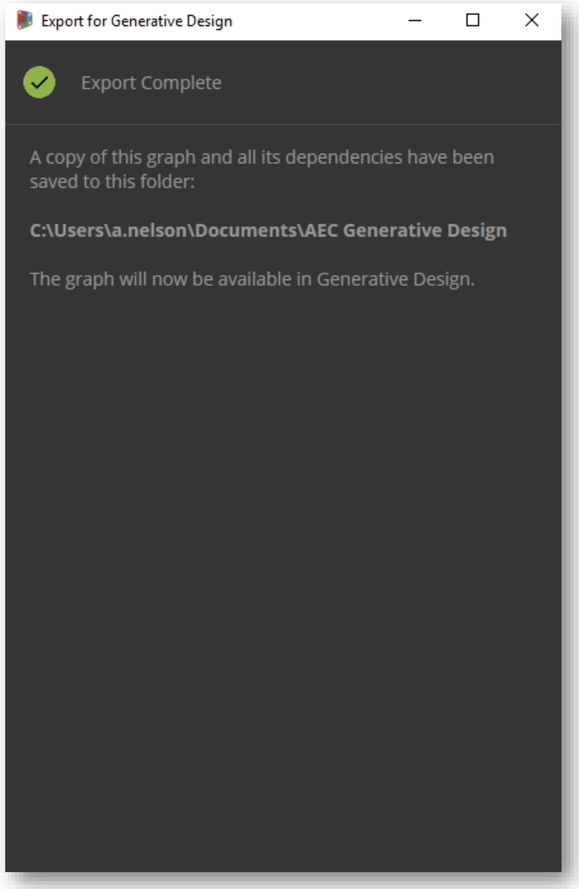


Click “Export”

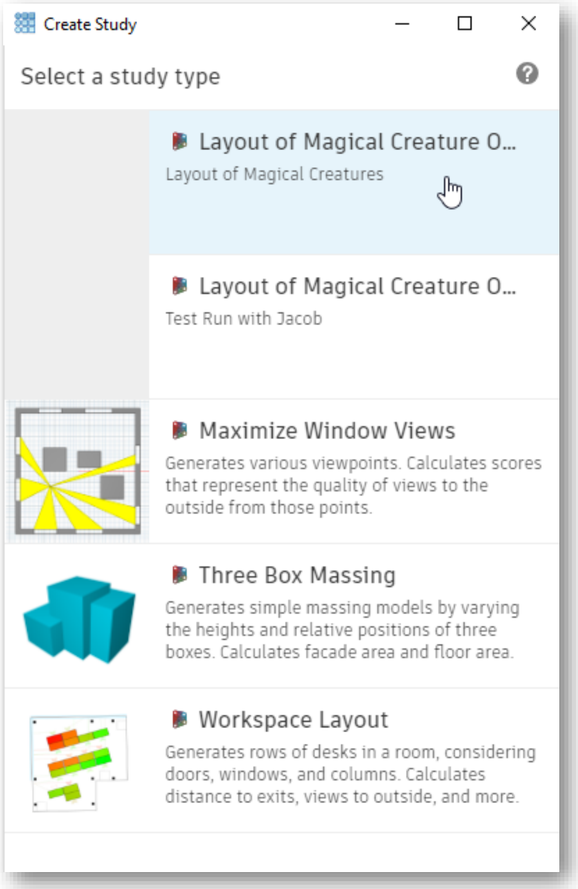


How the Magic Works

Export is now Complete!



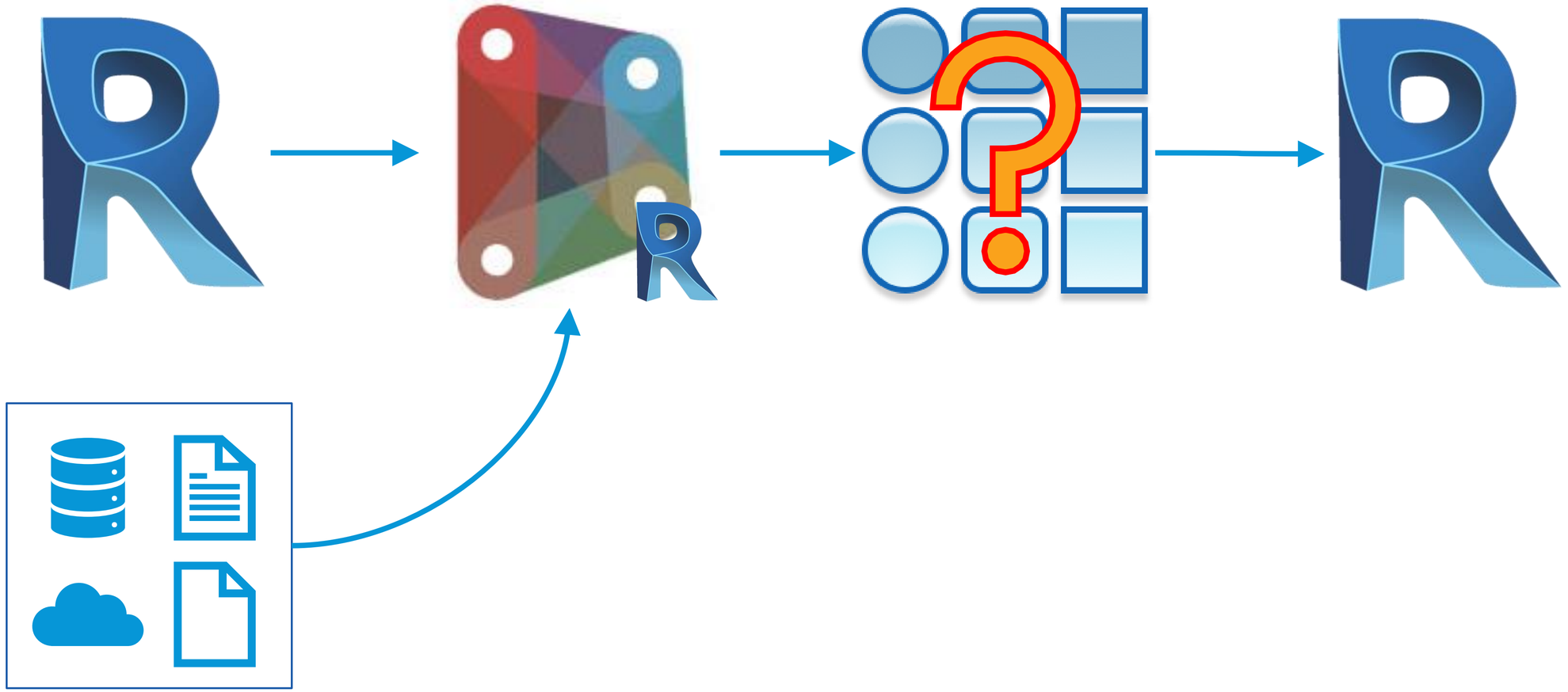
Your study is now available for GD!



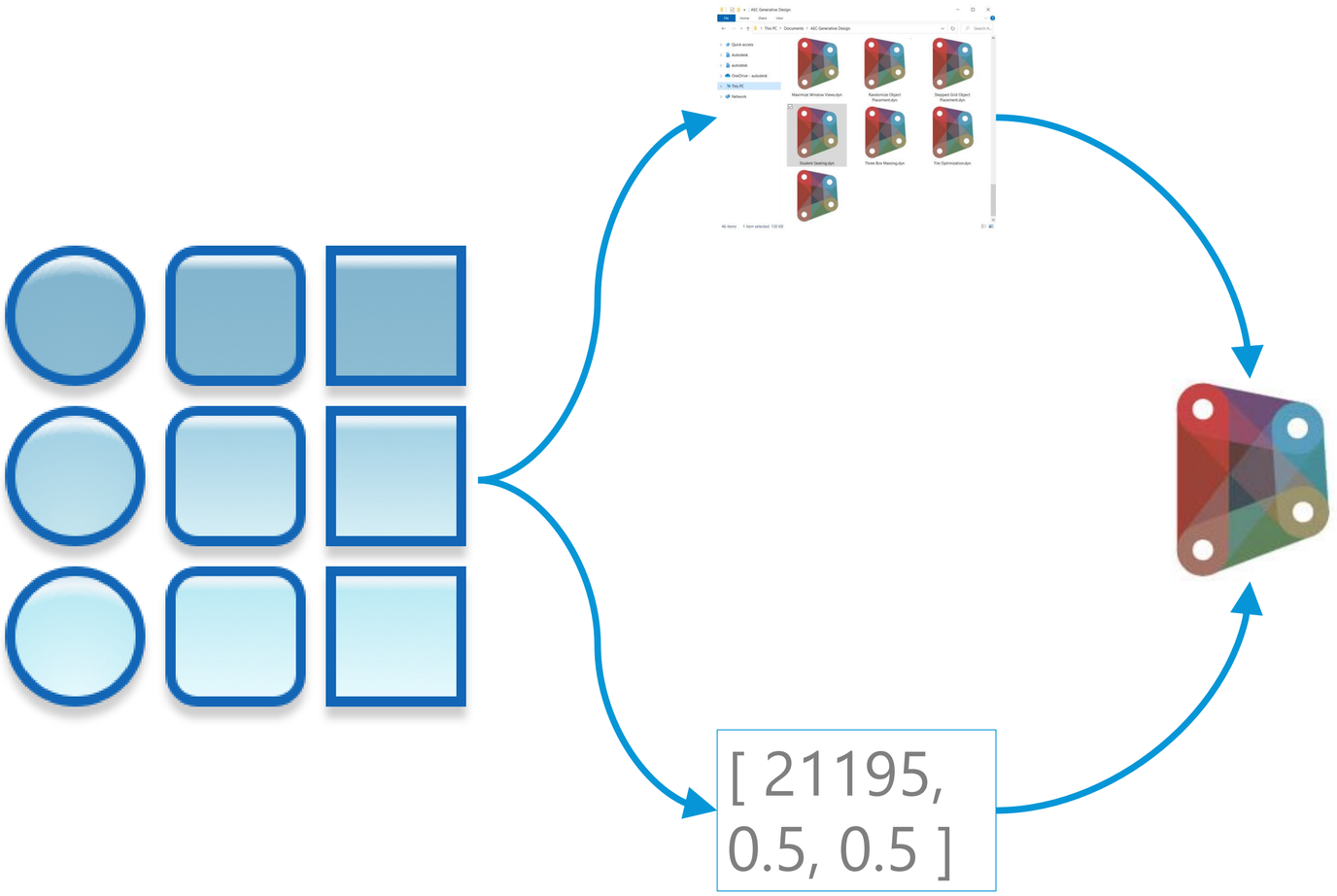
Custom Studies

Built-In Studies

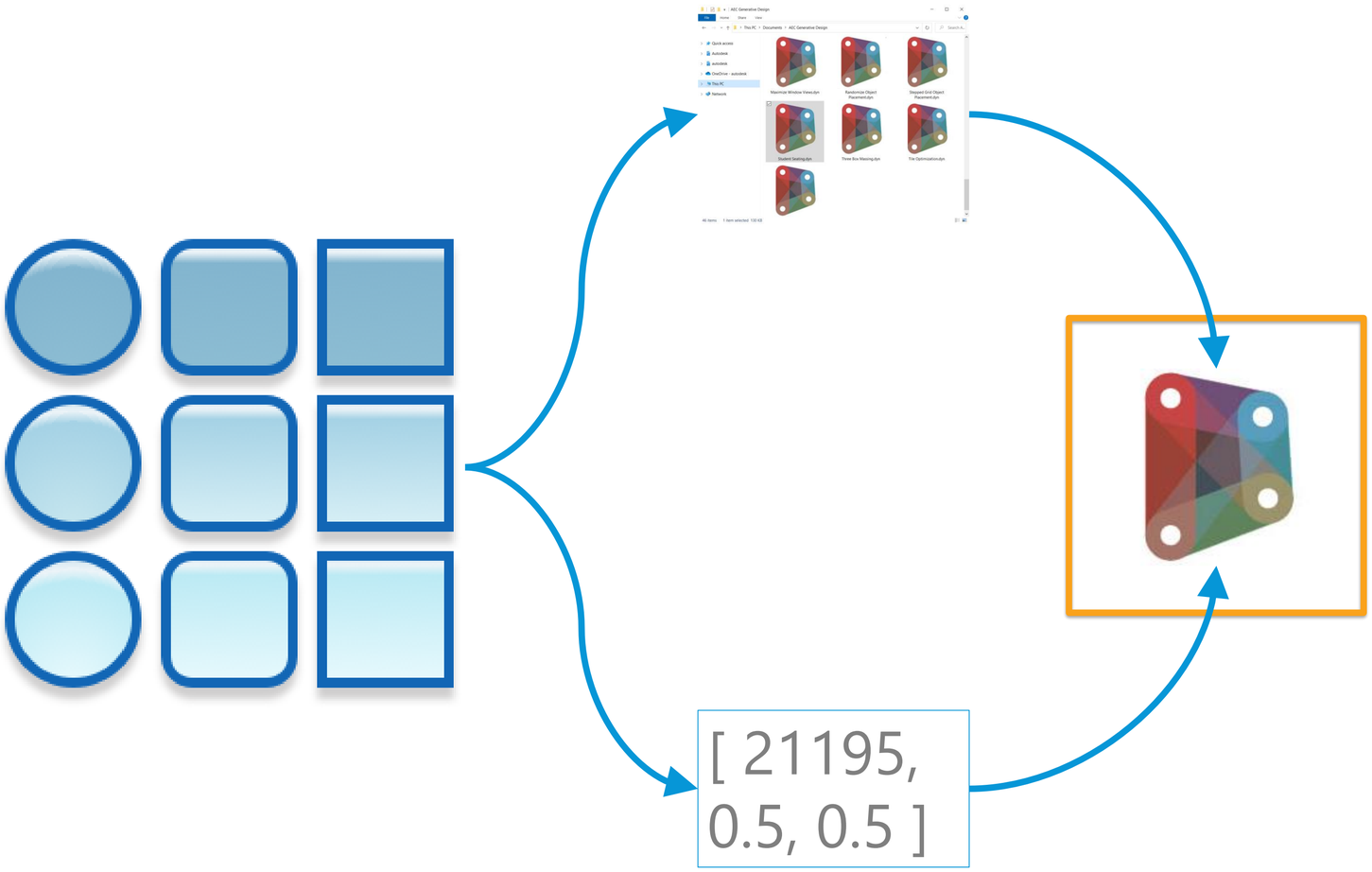
How the Magic Works



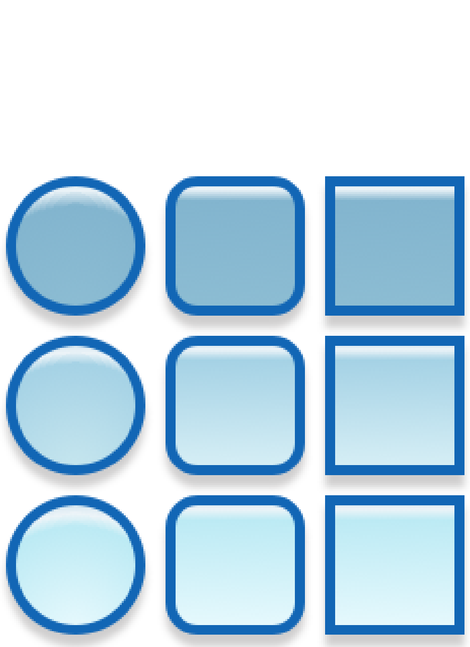
How the Magic Works

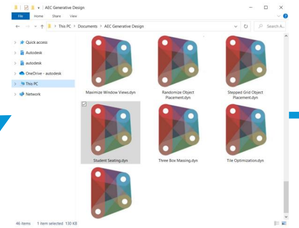


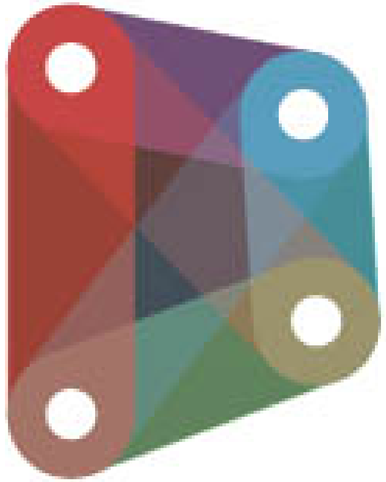
How the Magic Works

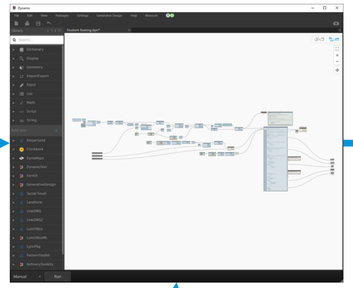


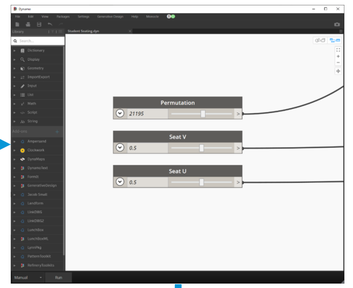
How the Magic Works

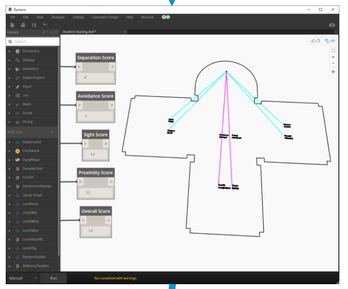








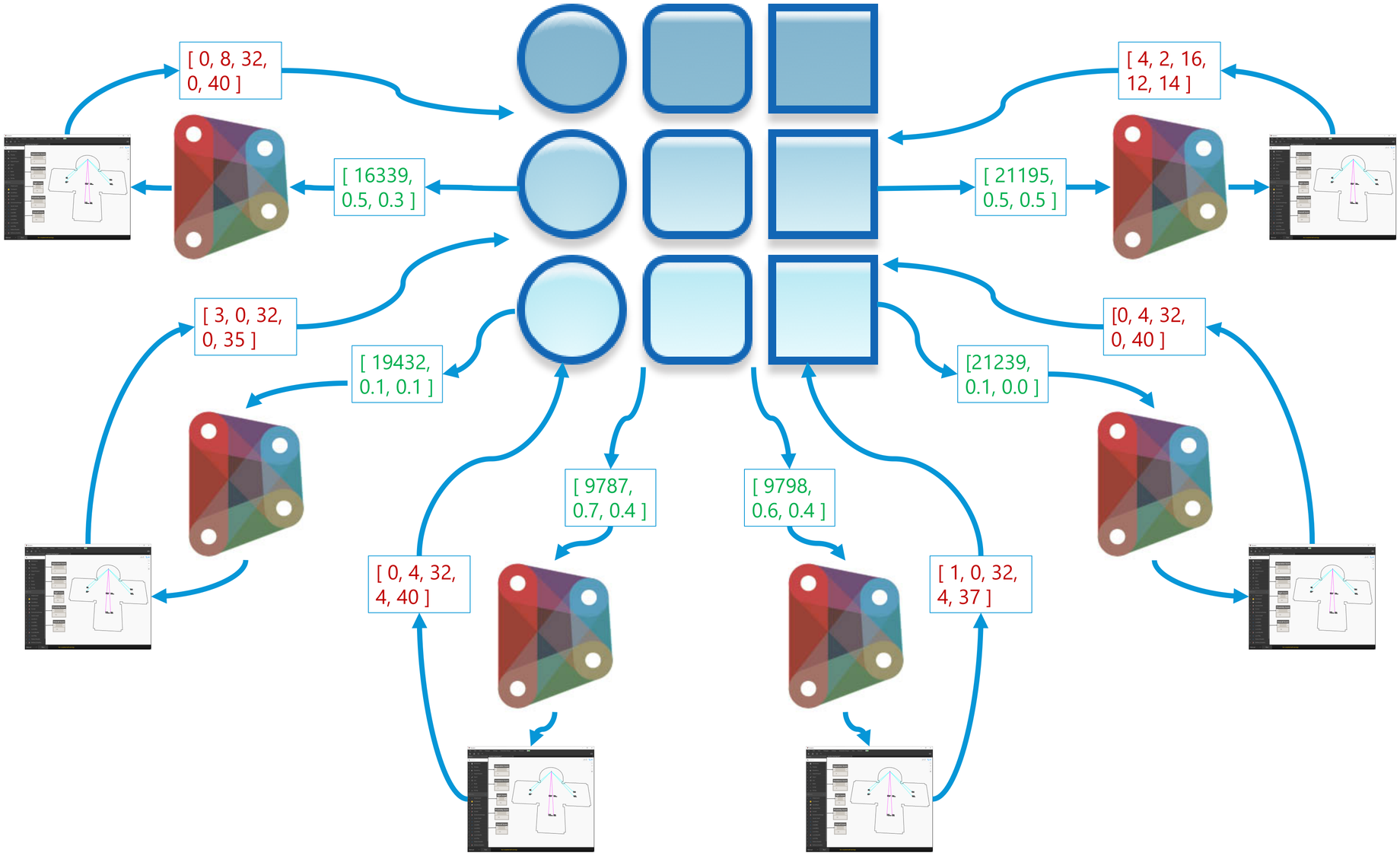




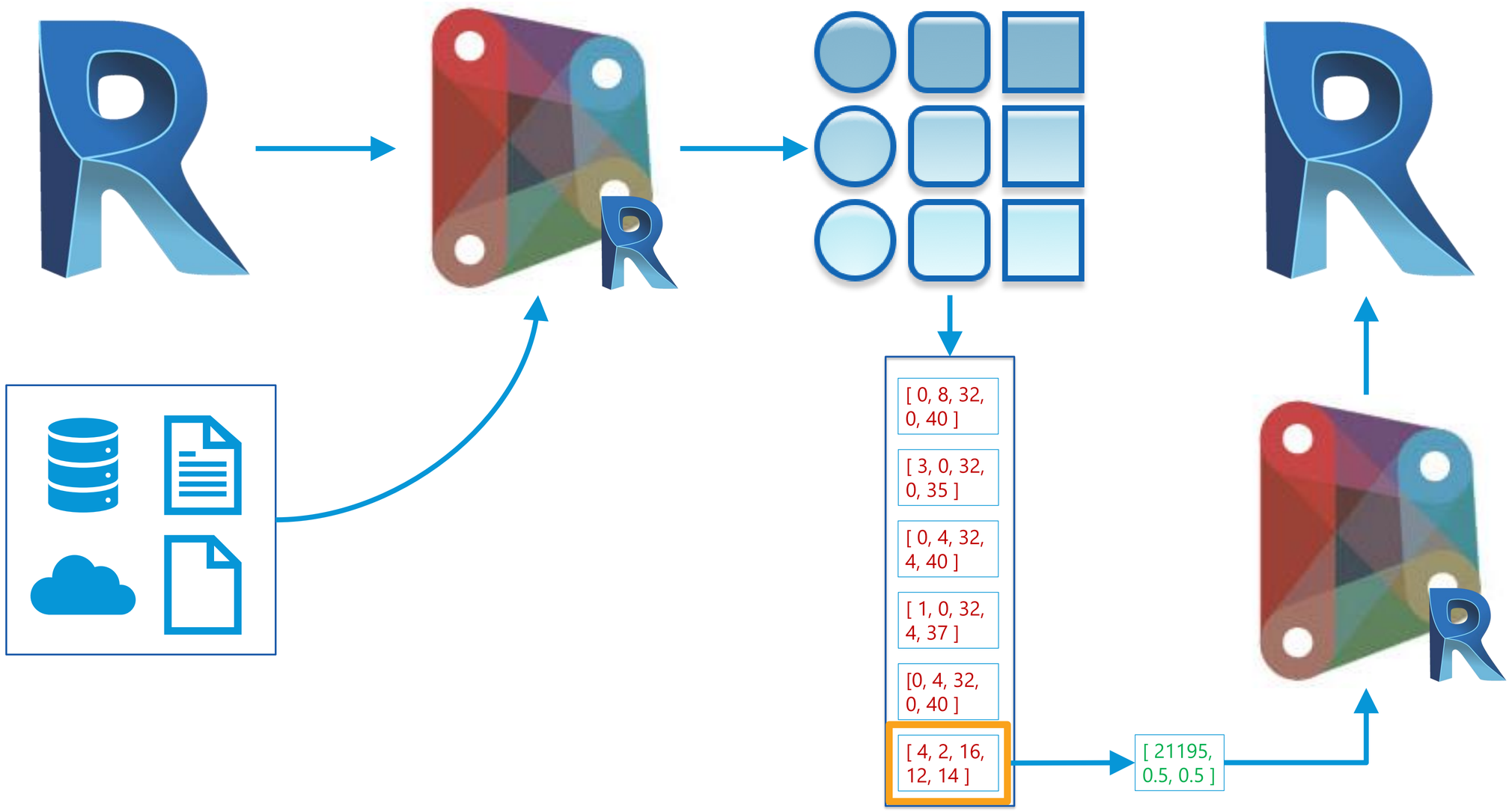
[21195, 0.5, 0.5]

[4, 2, 16, 12, 14]

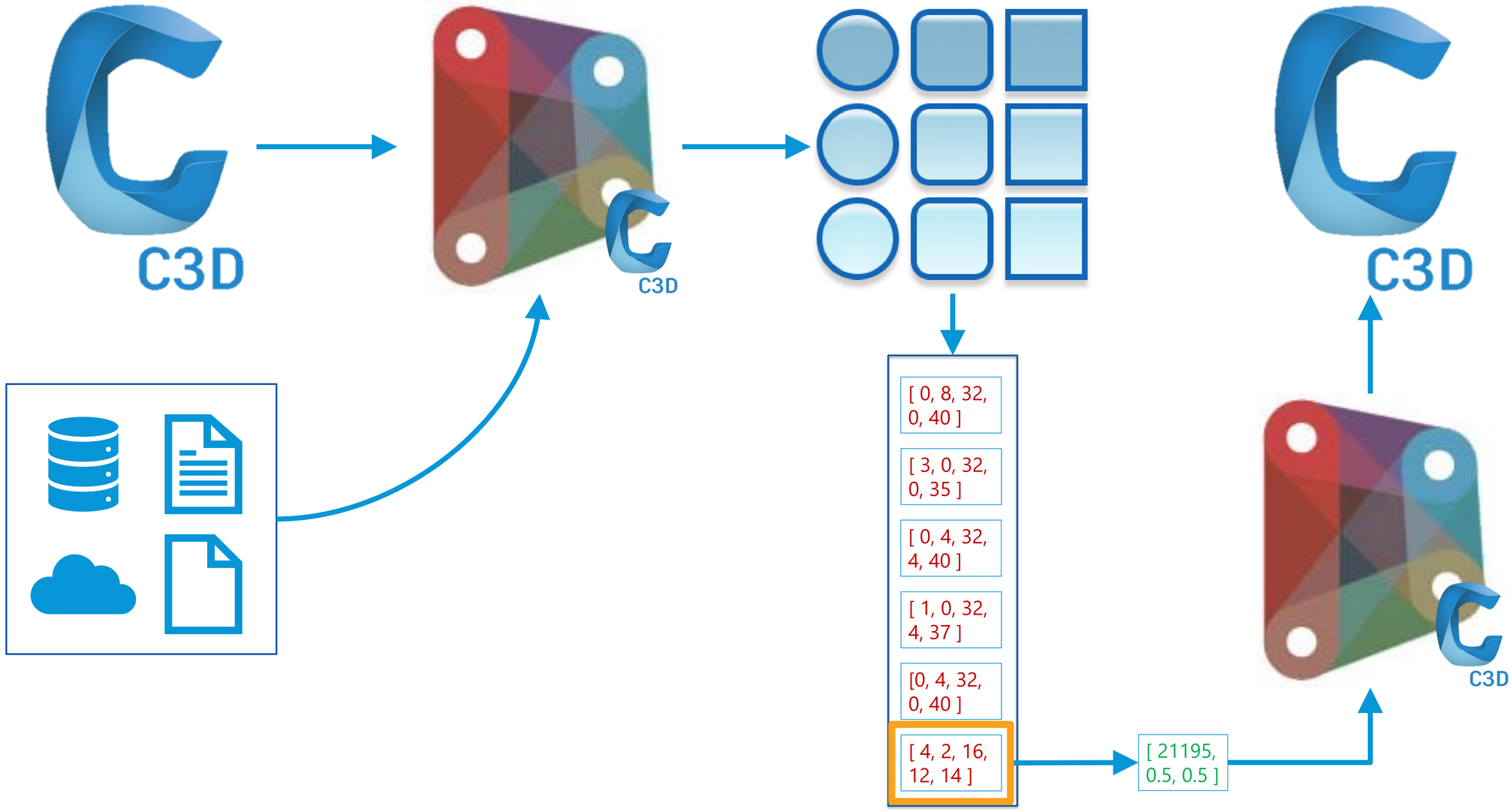
How the Magic Works



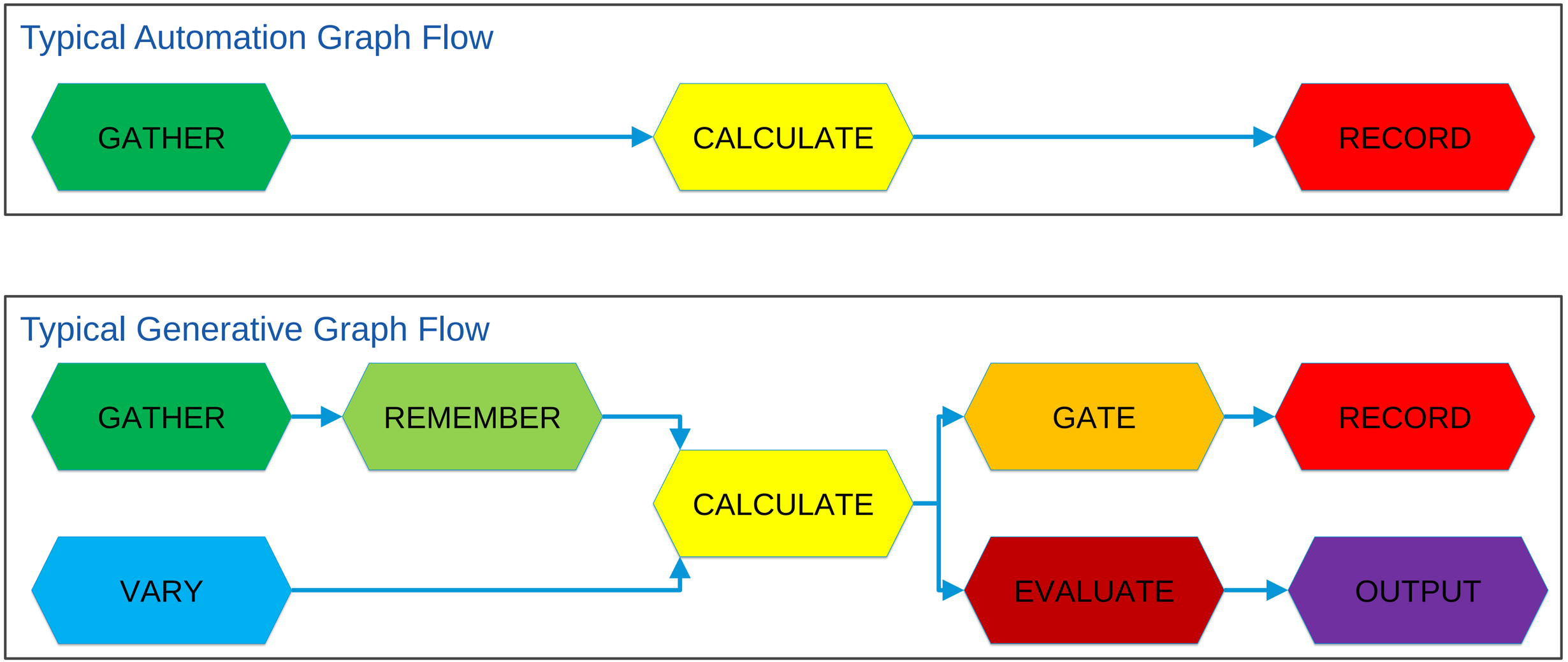
How the Magic Works



How the Magic Works



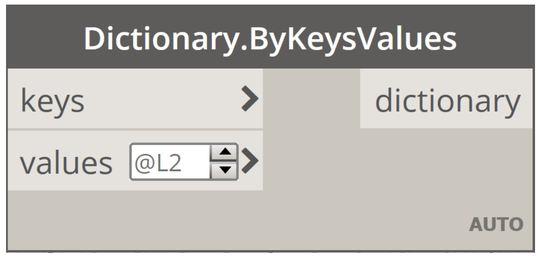
Graph Anatomy: Automation vs Generative



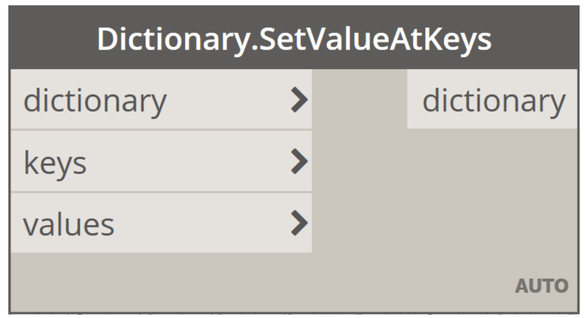
Dictionaries are your Friend

Data Set

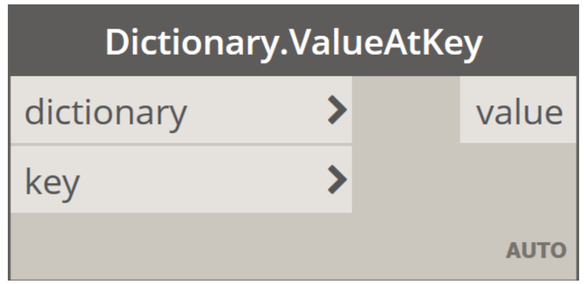
Create Dictionary



Add Keys + Values

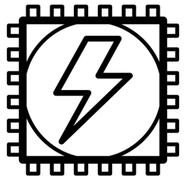


Get Value(s) at Key

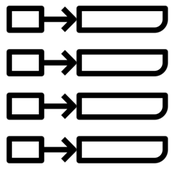


TIME SAVINGS

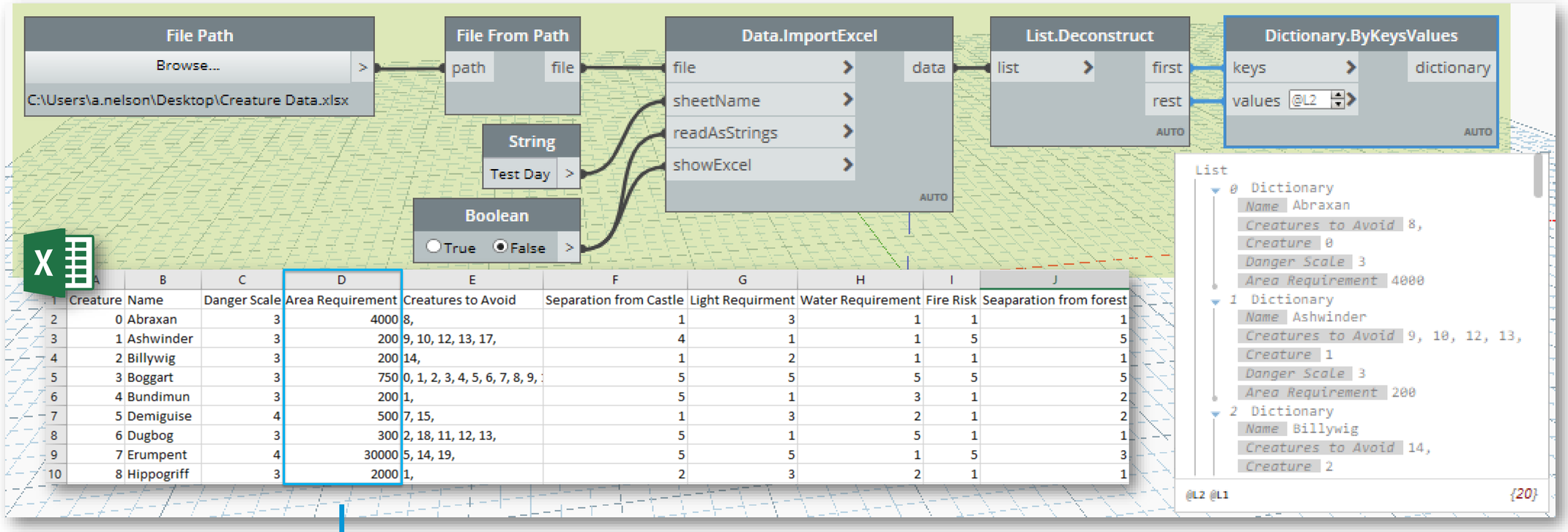
FASTER COMPUTATION



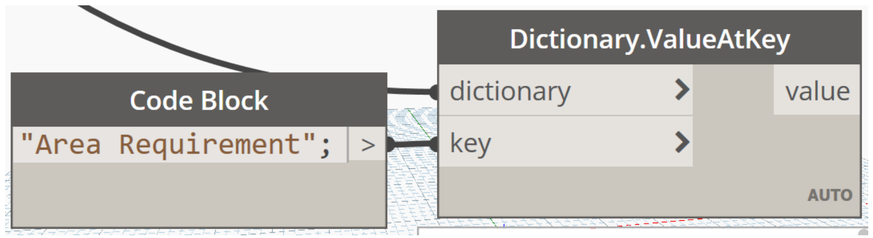
EASIER TO ACCESS DATA

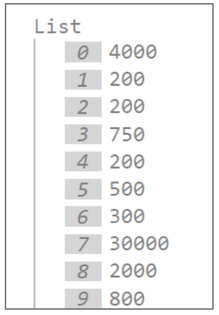


Dictionaries are your Friend

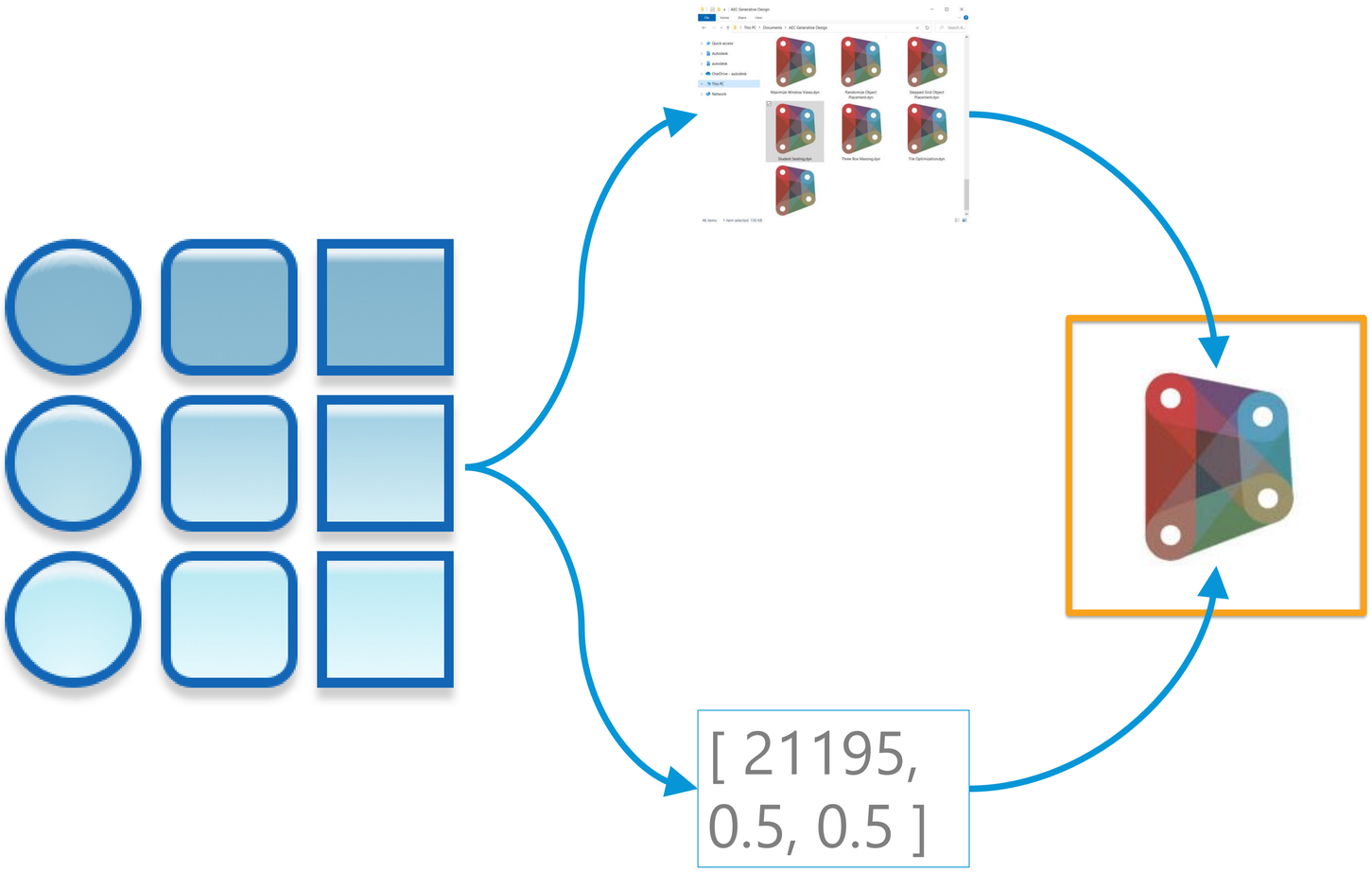


	A	B	C	D	E	F	G	H	I	J
1	Creature	Name	Danger Scale	Area Requirement	Creatures to Avoid	Separation from Castle	Light Requirment	Water Requirement	Fire Risk	Seaparation from forest
2	0	Abraxan	3	4000	8,	1	3	1	1	1
3	1	Ashwinder	3	200	9, 10, 12, 13, 17,	4	1	1	5	5
4	2	Billywig	3	200	14,	1	2	1	1	1
5	3	Boggart	3	750	0, 1, 2, 3, 4, 5, 6, 7, 8, 9,	5	5	5	5	5
6	4	Bundimun	3	200	1,	5	1	3	1	2
7	5	Demiguise	4	500	7, 15,	1	3	2	1	2
8	6	Dugbog	3	300	2, 18, 11, 12, 13,	5	1	5	1	1
9	7	Erumpent	4	30000	5, 14, 19,	5	5	1	5	3
10	8	Hippogriff	3	2000	1,	2	3	2	1	1

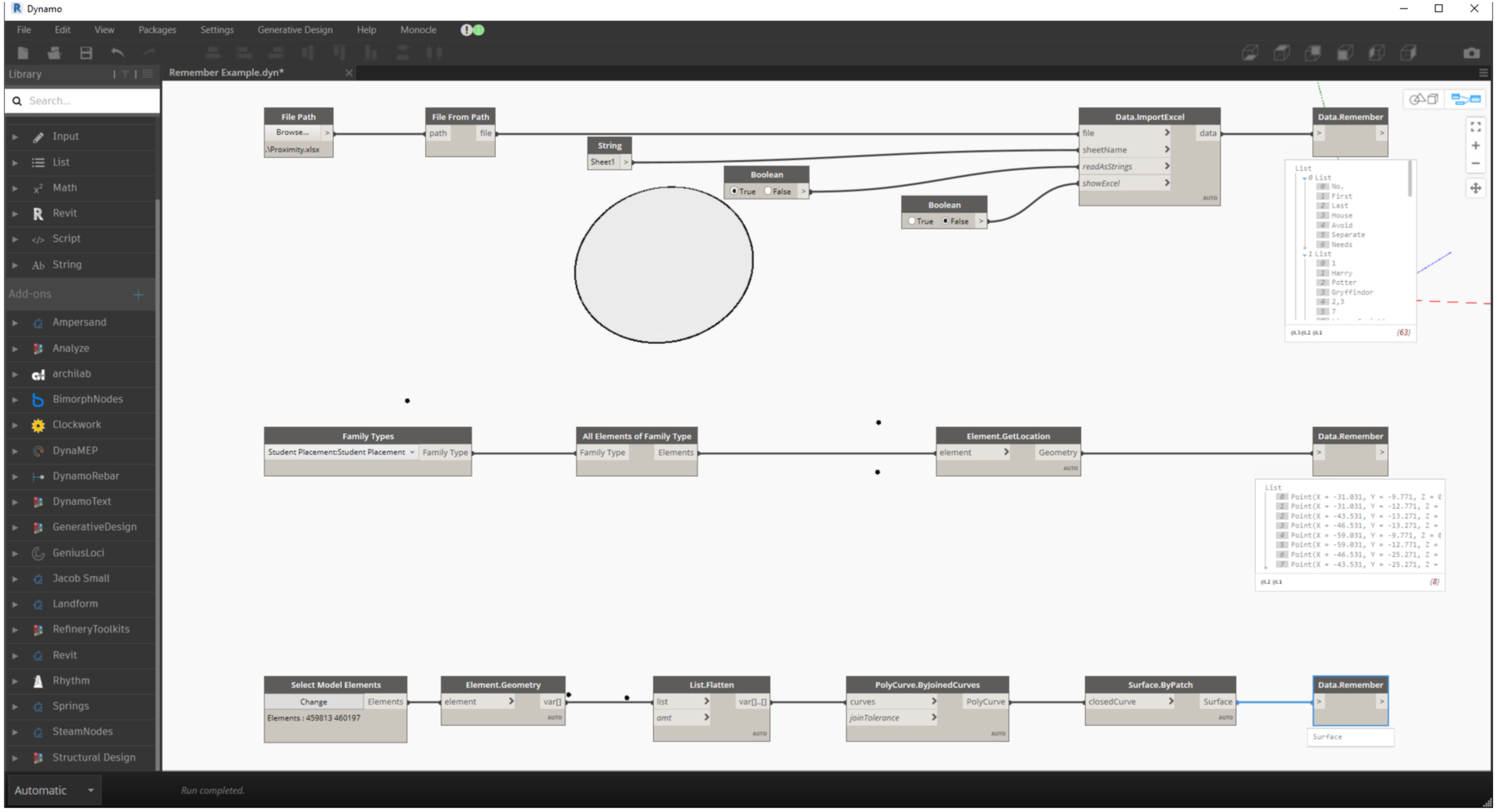




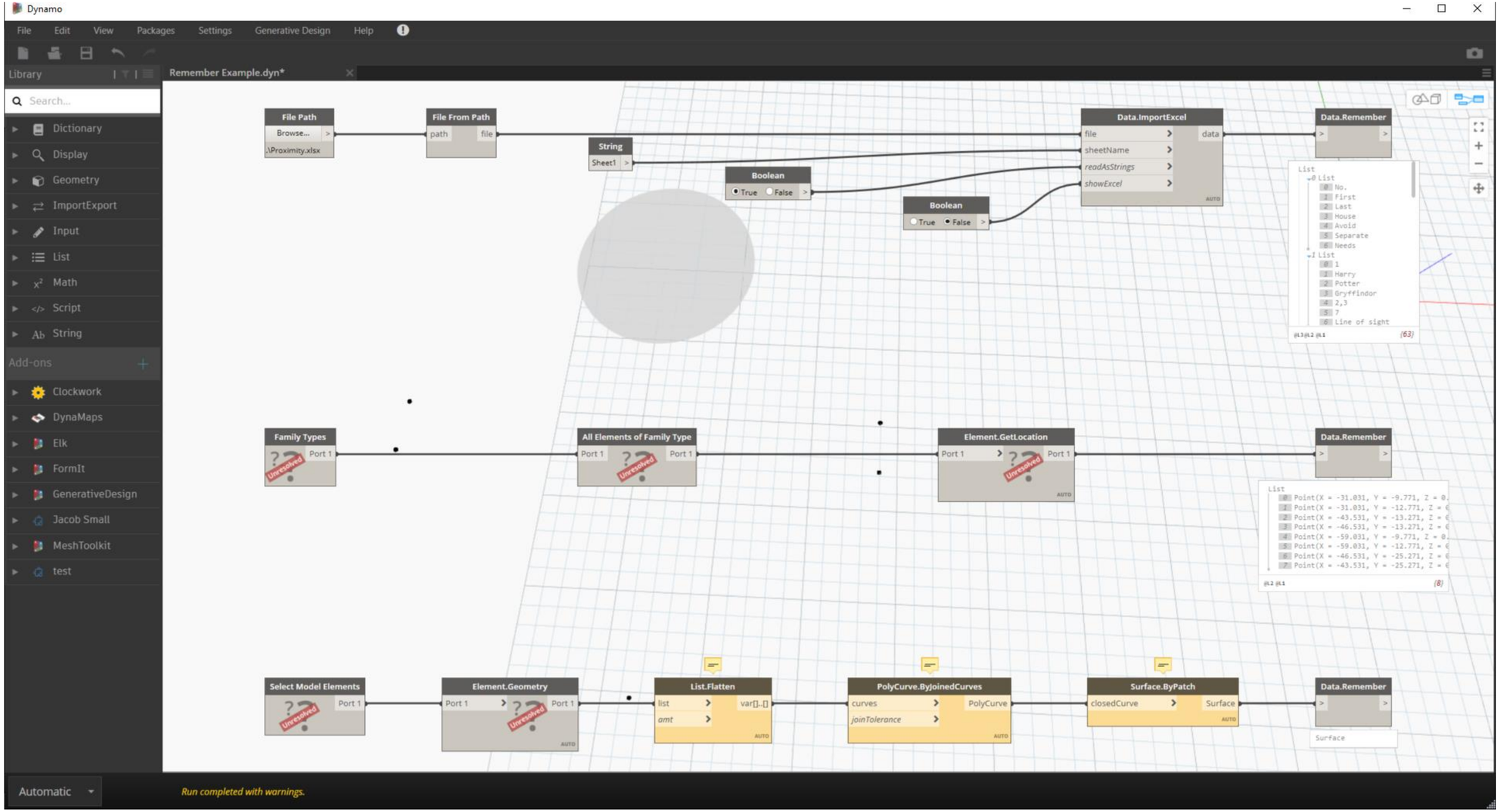
Remembering Static Values



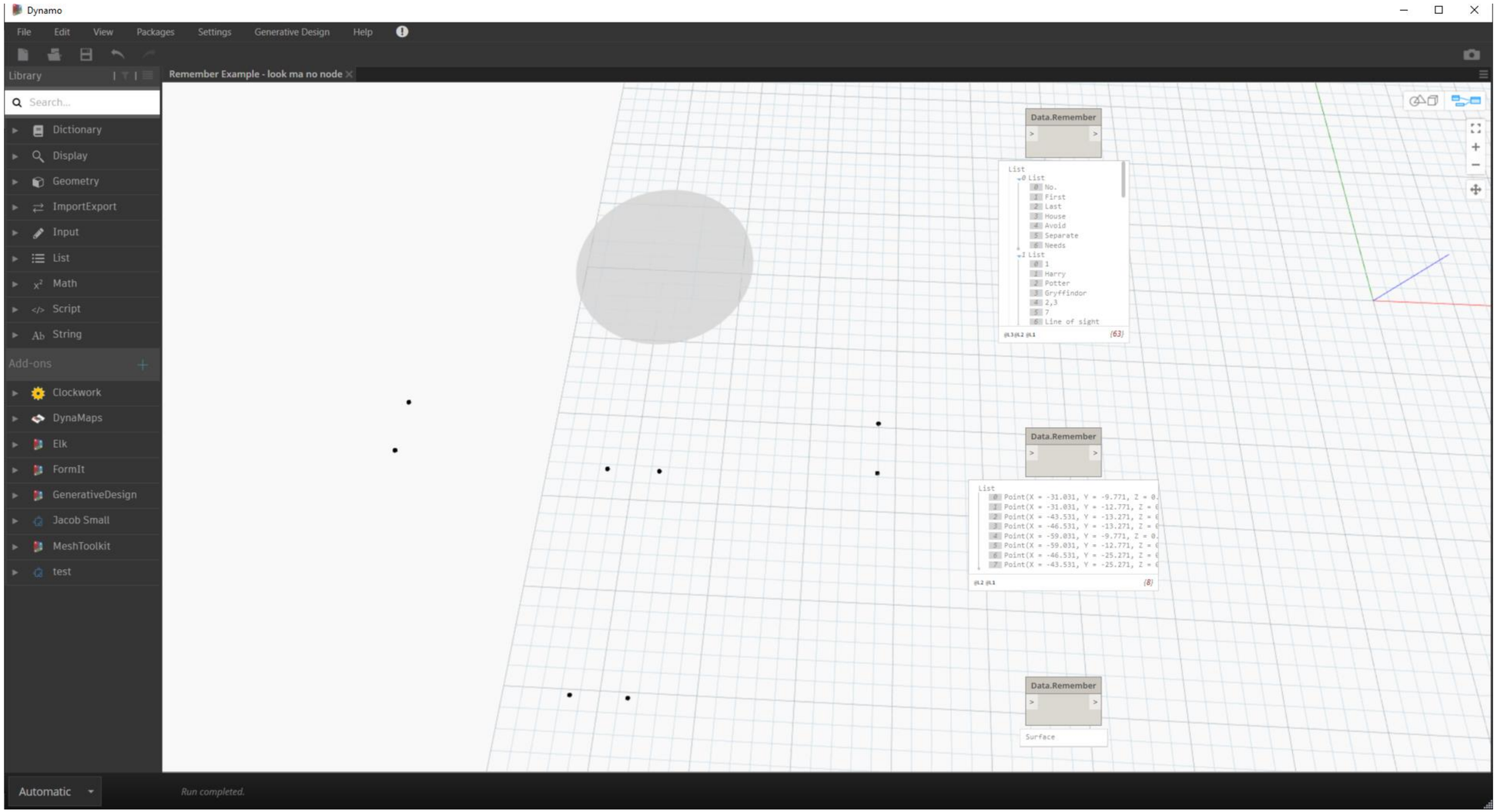
Remembering Static Values



Remembering Static Values

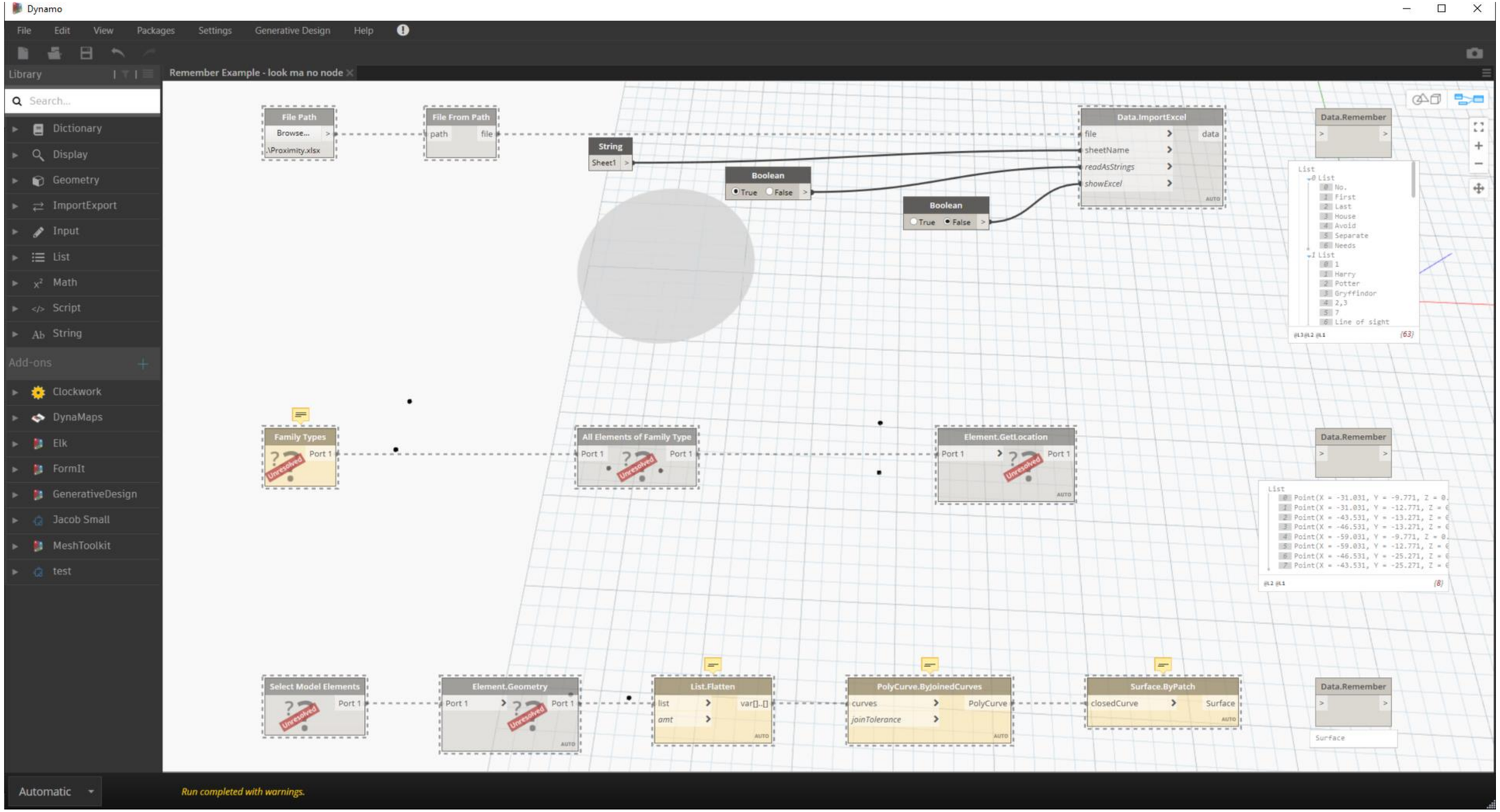


Remembering Static Values



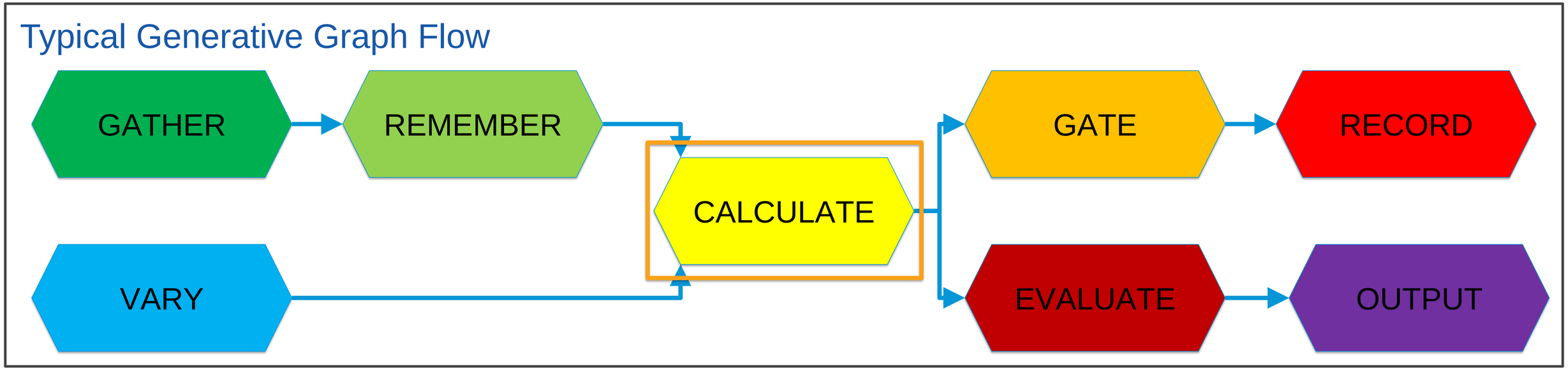
I don’t recommend this, but it’s certainly something you could do.

Remembering Static Values



Better! Faster execution (no need to spin up excel) and you can modify

Generation Systems



Generation Systems

Example #1

- **Assign to Option Set**
  - Gather option set
  - Gather values
  - Set value to option

- **Define Location & Orientation**
  - Coordinate system on curve
  - Coordinate system on surface
  - Coordinate system in space

- **Define Parti Diagram**
  - Polycurve by points
  - By offsets
  - NURBS by points

Example #2

- **Define & Set Points**
  - Point on curve
  - Point on surface
  - Point in space

Example #3

- **Define Shape / Primitive**
  - Slice up domain
  - Polygon by points
  - Primitives by inputs

- **Define by Other**
  - BE CREATIVE!

Evaluations & Results

“So… This is all just math.”

“I was told there would be no math!”

“Well, you were lied to.”

“Crap.”

“Dynamo will run the numbers for you.”

“Good, so no math.”

“But you have to show it how.”

“Don’t know if you realize this, but that’s still math.”

“Yep.”

Evaluations & Results – Common Methods

- **Properties of objects (geometry)**
  - Length
  - Area
  - Volume
  - Height, radius, etc.
- **Spatial conditions met**
  - Distance to /from
  - Path between
  - Length along curve network
  - Elevation
  - Orientation to ‘optimum’

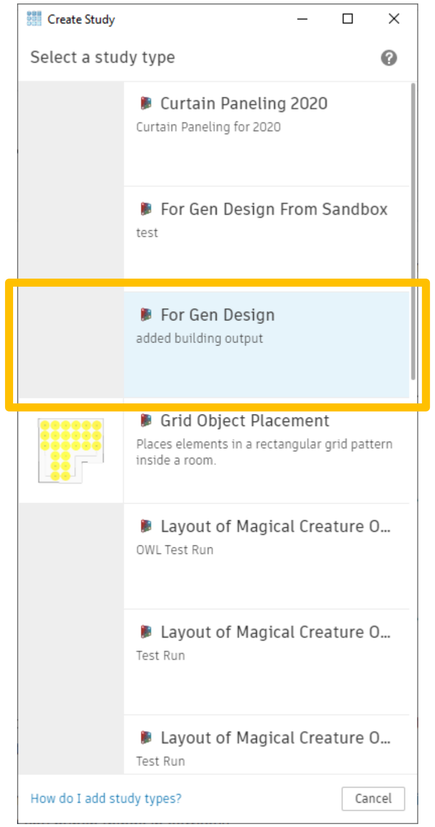
- **Penalize values based on ‘key’ outcome**
  - “Out of bounds”
  - “Strike 3”
- **Variety in solution set requires conflicting outcomes**
  - If outcome A reinforces B then there can be only one maximum
- **Try to always minimize**
  - Default setting for optimization
  - Multiplication by -1 helps!

Evaluations & Results – Frustrations

What I often hear at this phase of development:

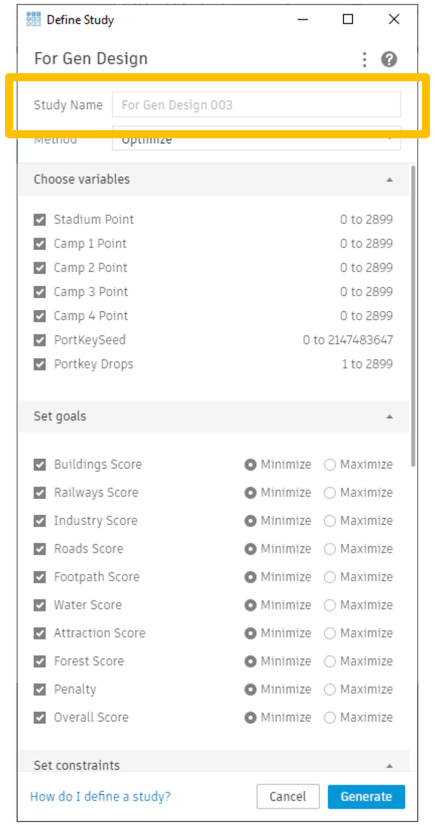
- **“I don’t know what to optimize for.”**
  - Listen to your client – they know what they want
  - Listen to your boss – they often know what’s important
  - Step back and ask why you started in the first place
- **“I don’t know how to measure what I want to optimize.”**
  - Find the ‘real world equivalent’ when possible
  - Look it up: https://dictionary.dynamobim.com/#/
  - Ask: https://forum.dynamobim.com/
- **“My evaluation calculation takes too long.”**
  - Simplify the calculation
    - Remember the static bits for reuse
    - Reduce the list complexity

UI and Selection



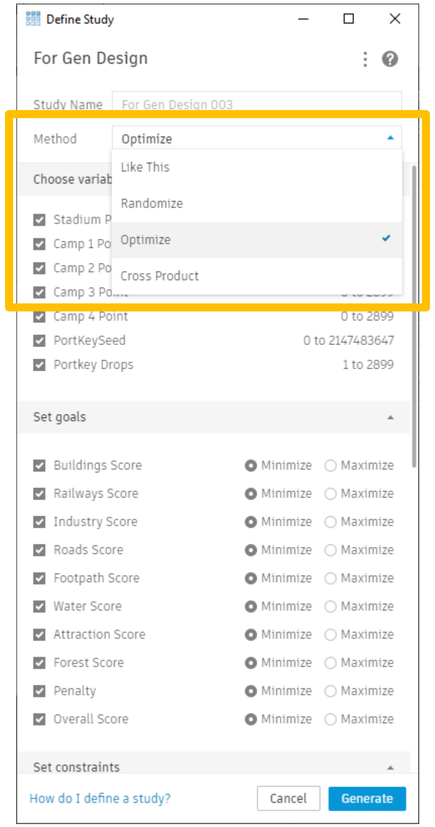
1. Select Study Type
2. Name Study
3. Select Method
4. Choose Variables
5. Configure Goals
6. Define Constraints
7. Set Generation Settings
8. Address Issues
9. Generate
10. Get a cup of coffee

UI and Selection



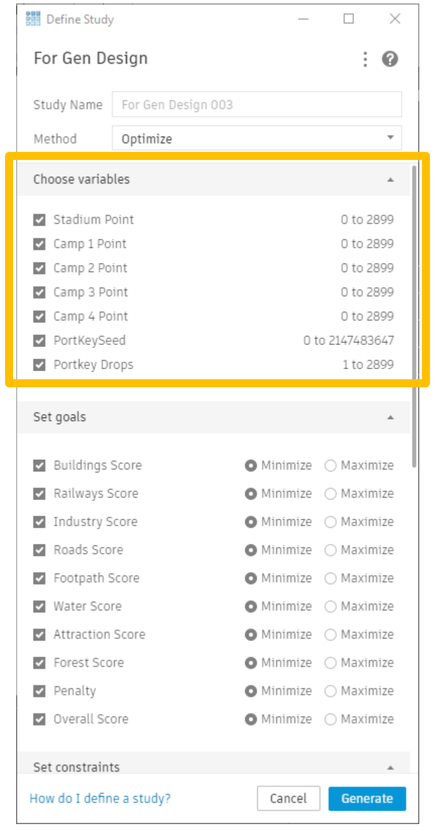
1. Select Study Type
2. Name Study
3. Select Method
4. Choose Variables
5. Configure Goals
6. Define Constraints
7. Set Generation Settings
8. Address Issues
9. Generate
10. Get a cup of coffee

UI and Selection



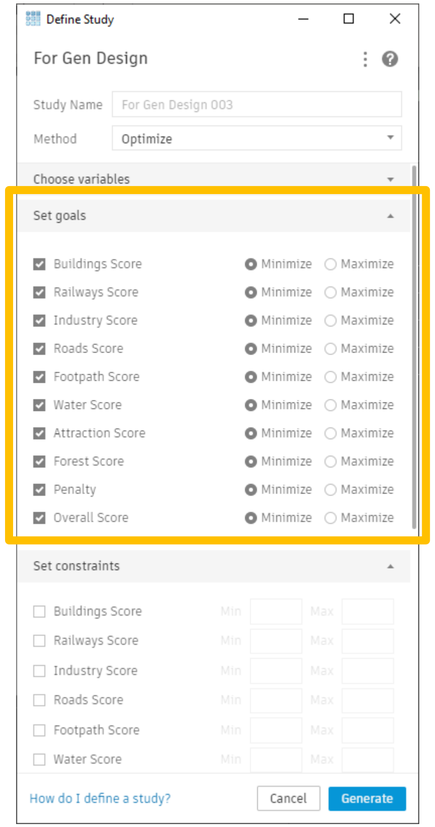
1. Select Study Type
2. Name Study
3. Select Method
4. Choose Variables
5. Configure Goals
6. Define Constraints
7. Set Generation Settings
8. Address Issues
9. Generate
10. Get a cup of coffee

UI and Selection



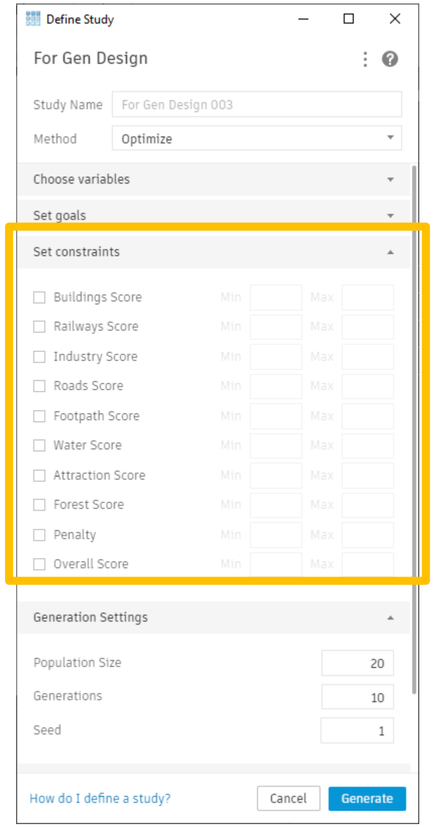
1. Select Study Type
2. Name Study
3. Select Method
4. Choose Variables
5. Configure Goals
6. Define Constraints
7. Set Generation Settings
8. Address Issues
9. Generate
10. Get a cup of coffee

UI and Selection



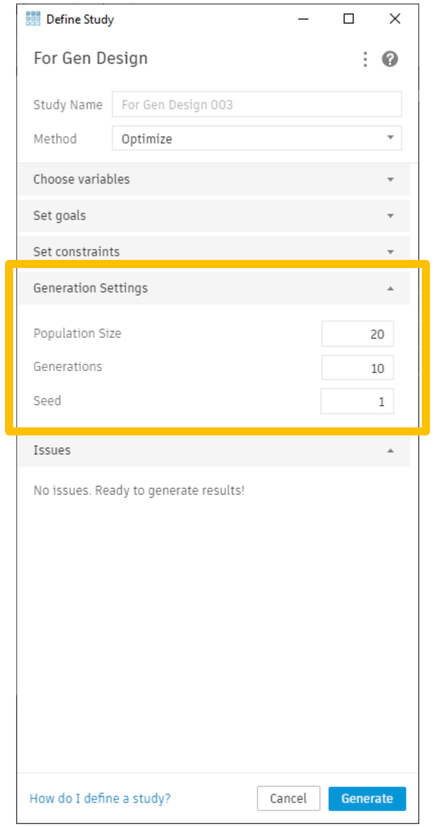
1. Select Study Type
2. Name Study
3. Select Method
4. Choose Variables
5. Configure Goals
6. Define Constraints
7. Set Generation Settings
8. Address Issues
9. Generate
10. Get a cup of coffee

UI and Selection



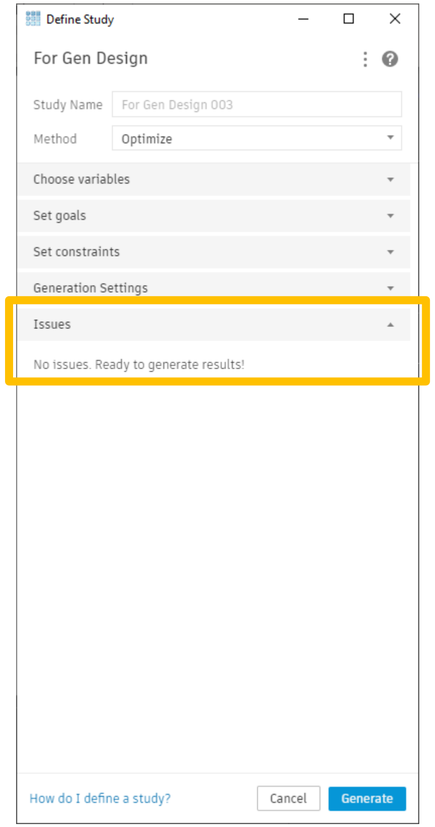
1. Select Study Type
2. Name Study
3. Select Method
4. Choose Variables
5. Configure Goals
6. Define Constraints
7. Set Generation Settings
8. Address Issues
9. Generate
10. Get a cup of coffee

UI and Selection



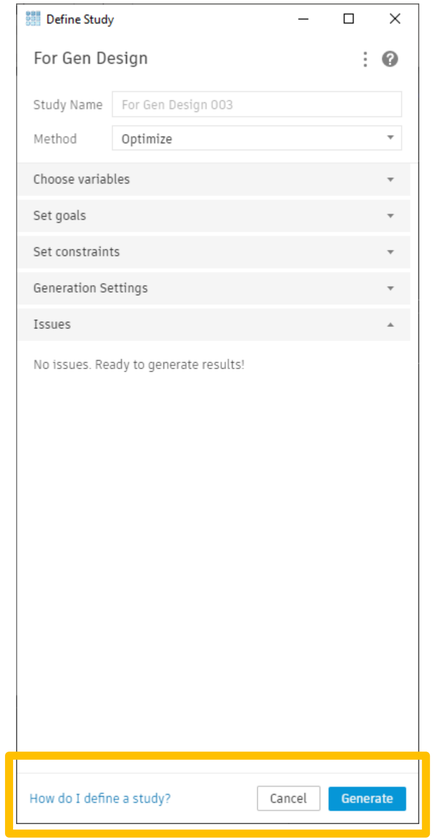
1. Select Study Type
2. Name Study
3. Select Method
4. Choose Variables
5. Configure Goals
6. Define Constraints
7. Set Generation Settings
8. Address Issues
9. Generate
10. Get a cup of coffee

UI and Selection



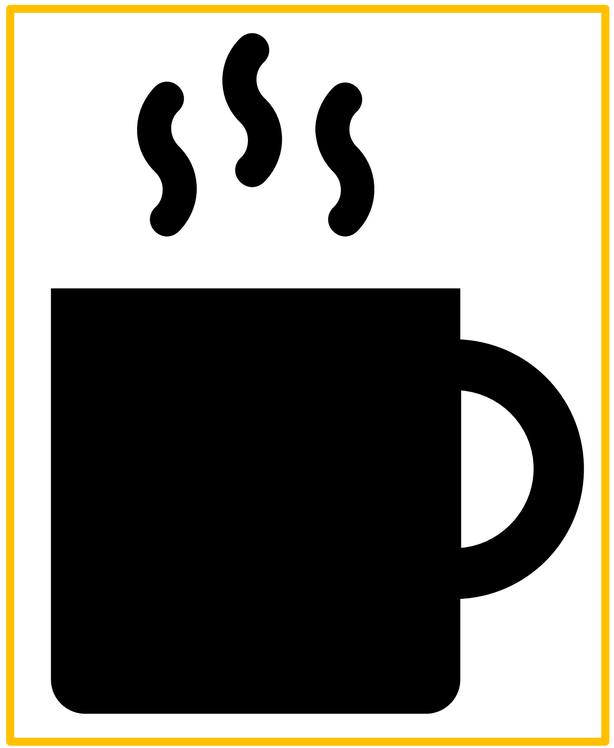
1. Select Study Type
2. Name Study
3. Select Method
4. Choose Variables
5. Configure Goals
6. Define Constraints
7. Set Generation Settings
8. Address Issues
9. Generate
10. Get a cup of coffee

UI and Selection



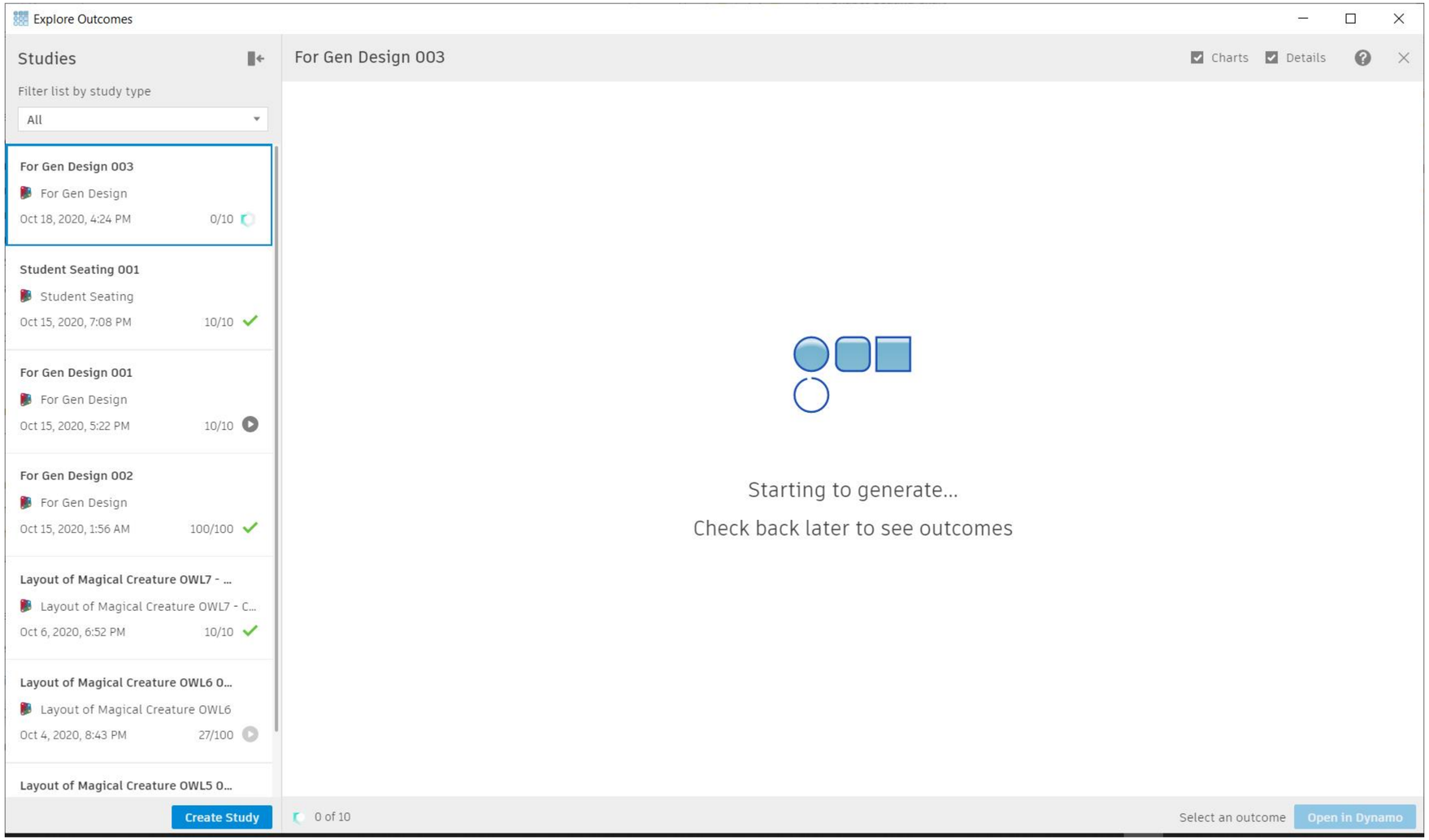
1. Select Study Type
2. Name Study
3. Select Method
4. Choose Variables
5. Configure Goals
6. Define Constraints
7. Set Generation Settings
8. Address Issues
9. Generate
10. Get a cup of coffee

UI and Selection

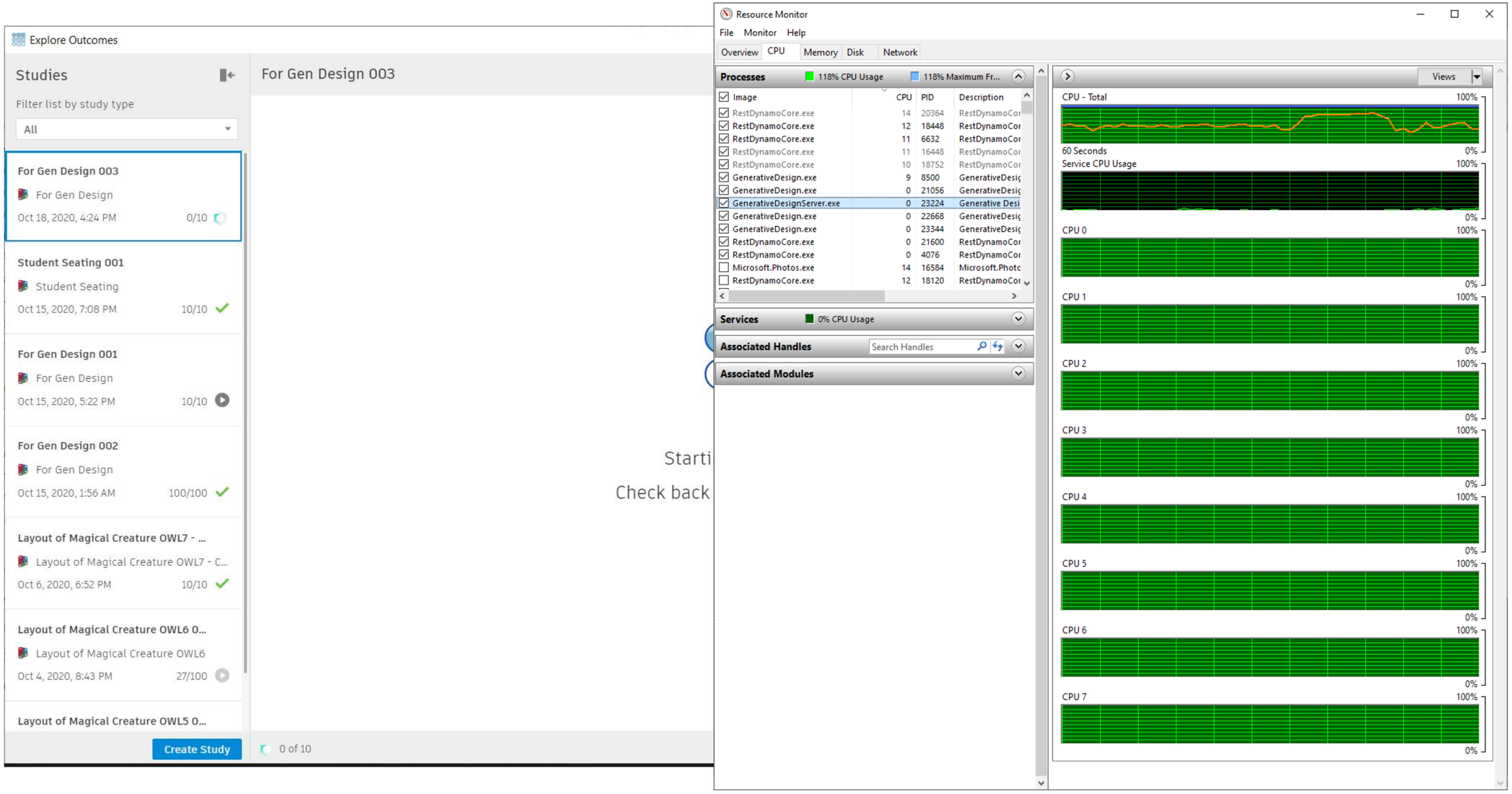


1. Select Study Type
2. Name Study
3. Select Method
4. Choose Variables
5. Configure Goals
6. Define Constraints
7. Set Generation Settings
8. Address Issues
9. Generate
10. Get a cup of coffee

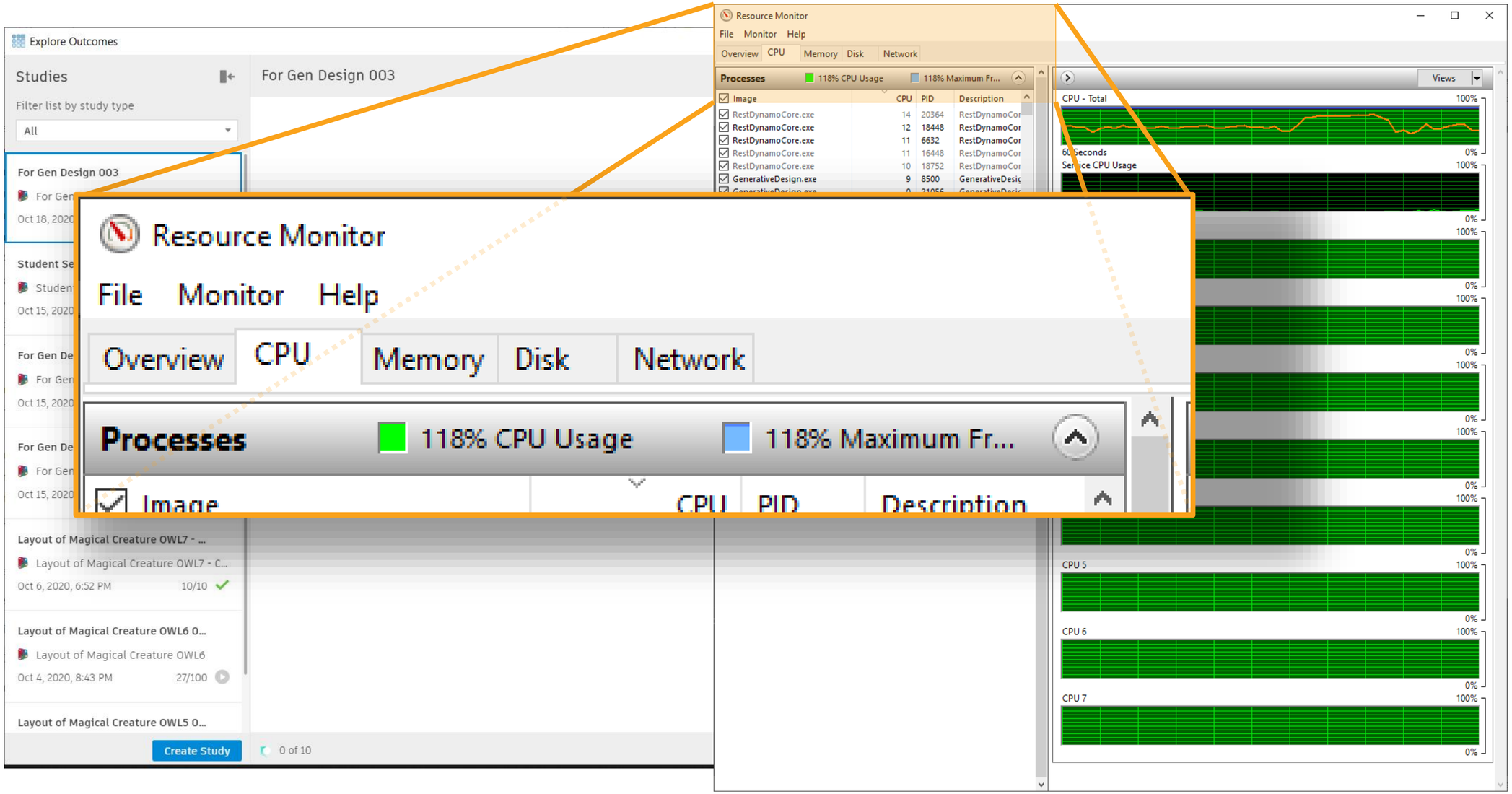
UI and Selection



UI and Selection



UI and Selection



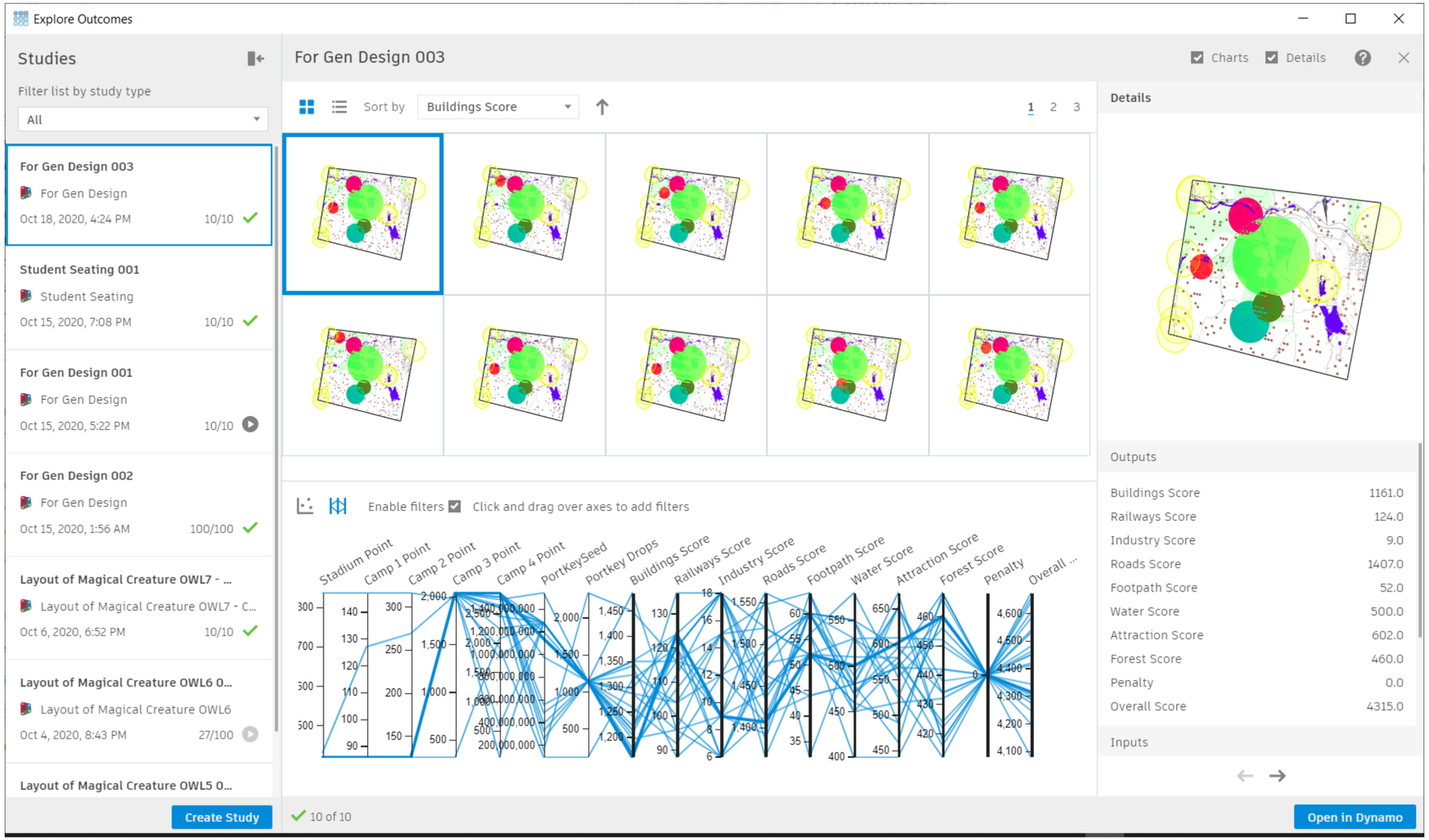
A Fika Break Later…

fika
|fi:ka| noun

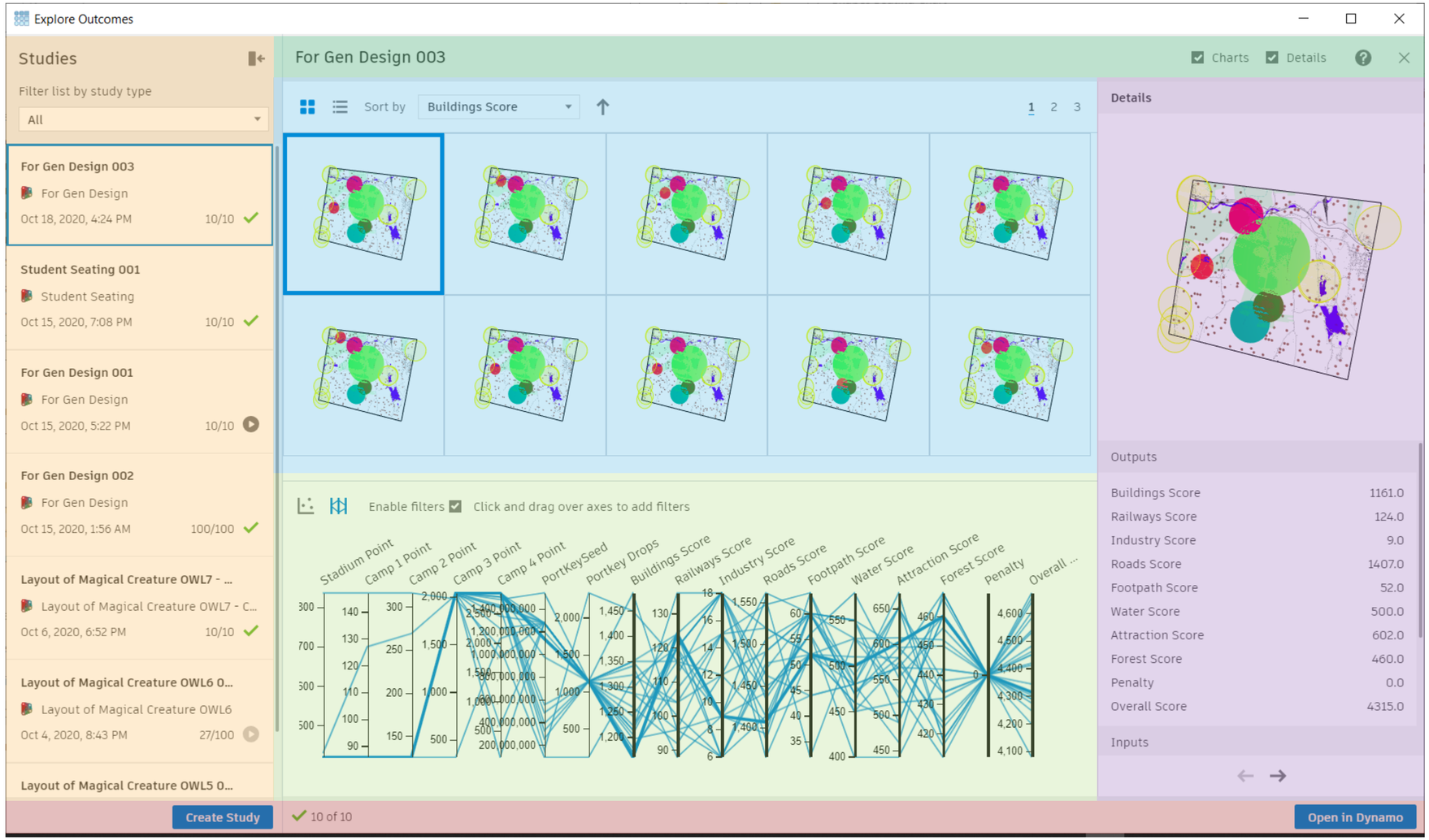
A Swedish concept meaning “to have coffee”, often accompanied with pastries and sweet treats.

Meant as a moment of quality time to savor with oneself or others.

UI and Selection



UI and Selection



Study Selection

Results and Config

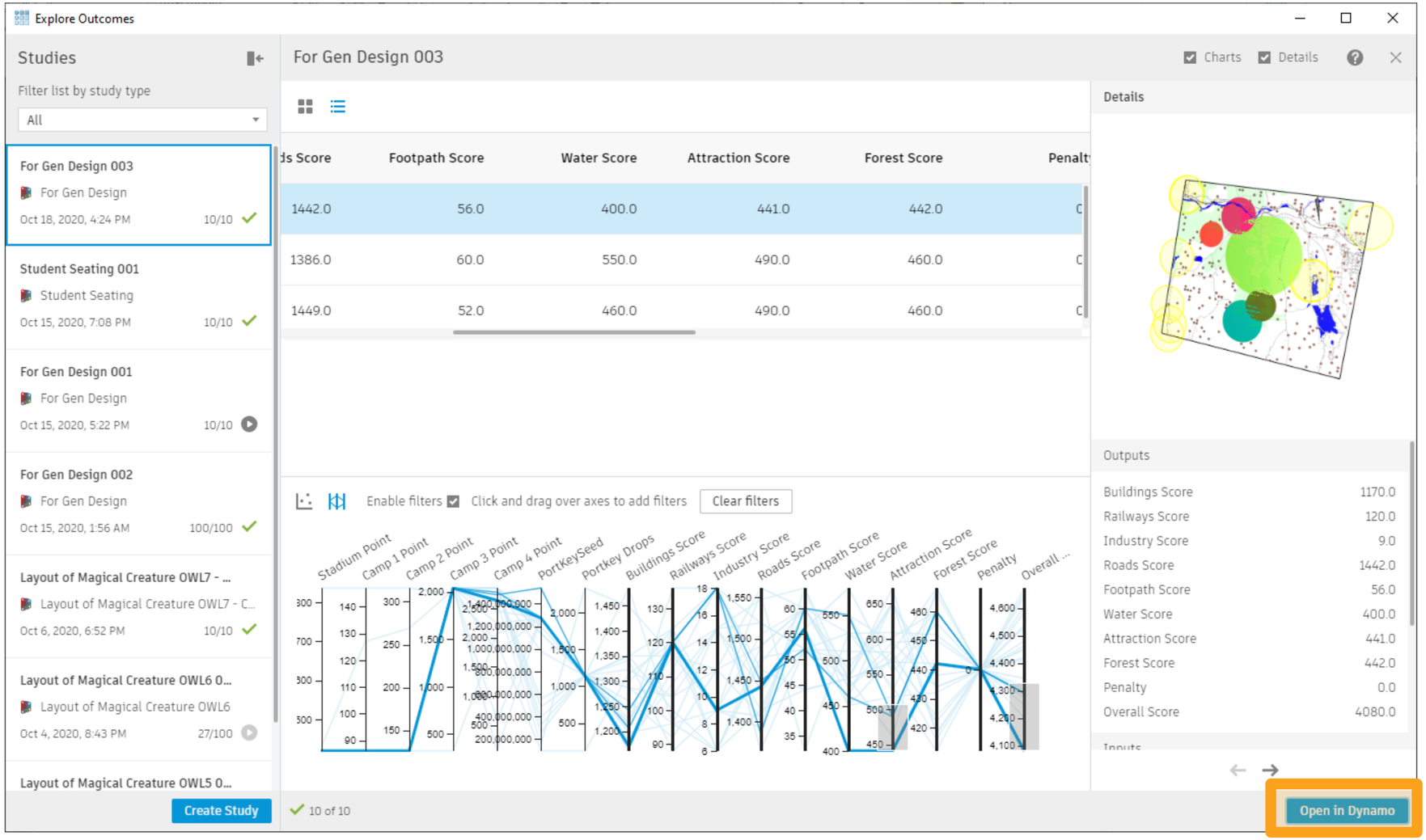
Grid or List

Charts

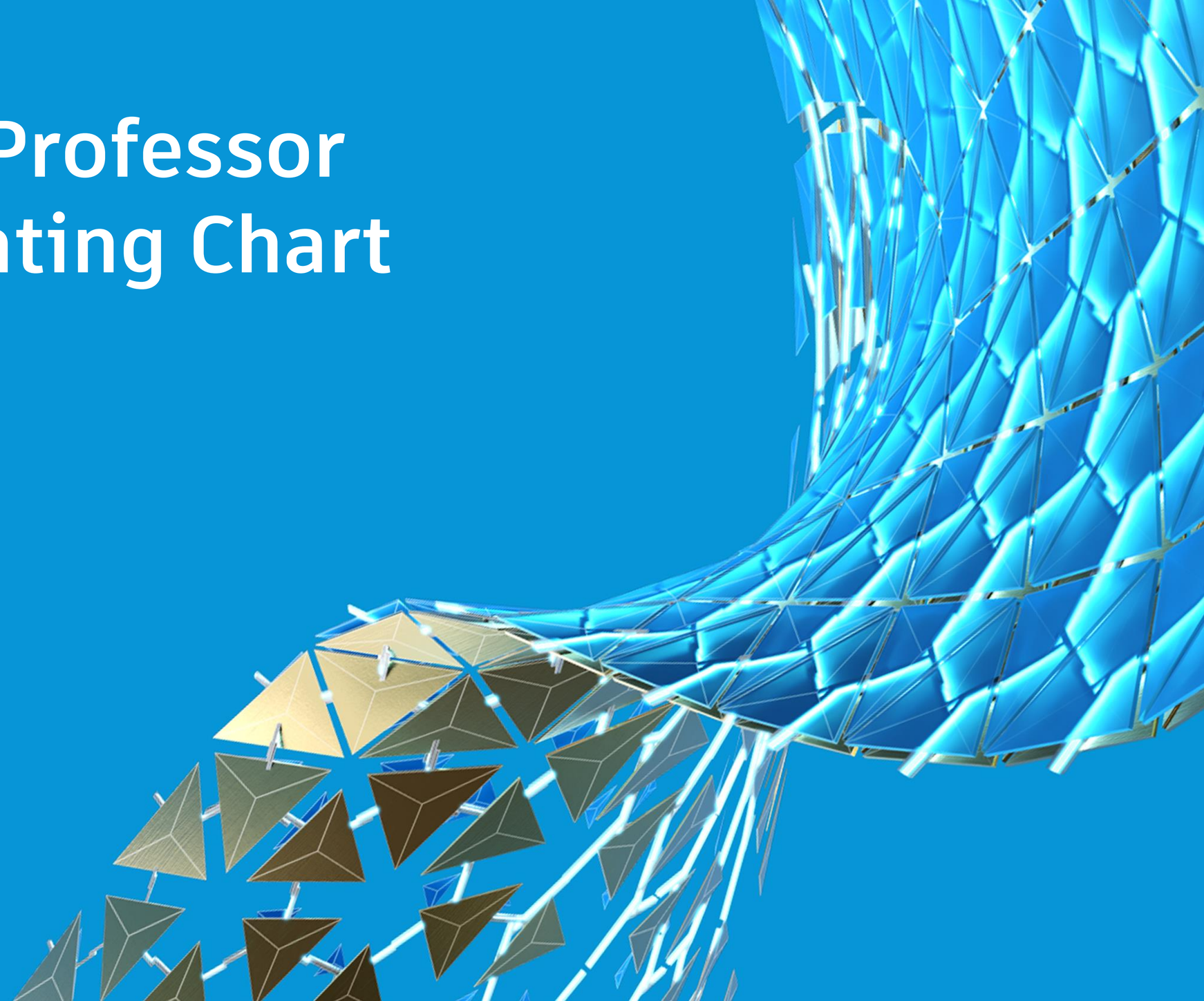
Details

Actions Bar

UI and Selection



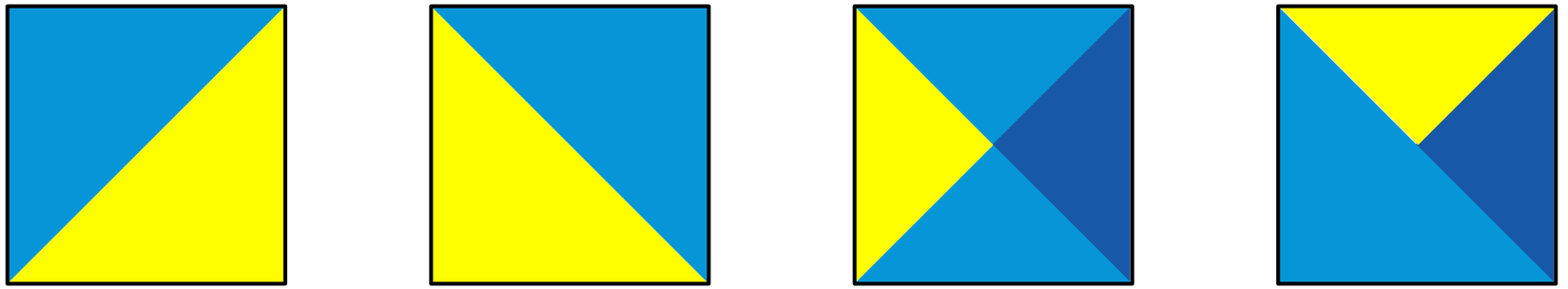
Problem 1: Professor Snape's Seating Chart

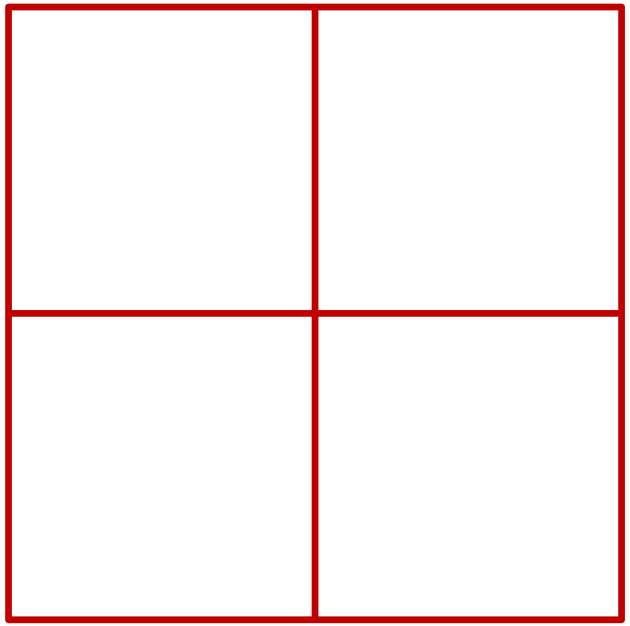


Assignment Studies

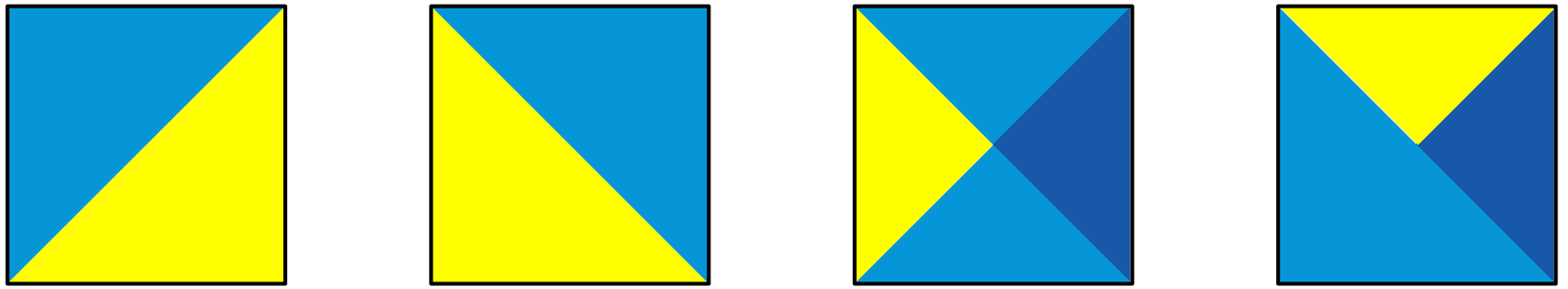
- Sometimes the design your working with is constrained so in terms of locations, but you need to optimize which program is assigned to each element.
- In this session, we will use the fixed seating in the potions classroom and assigning items from our data set to each.
- We'll be evaluating the design by reviewing each student's needs relative to the location of classmates, and to Professor Snape.

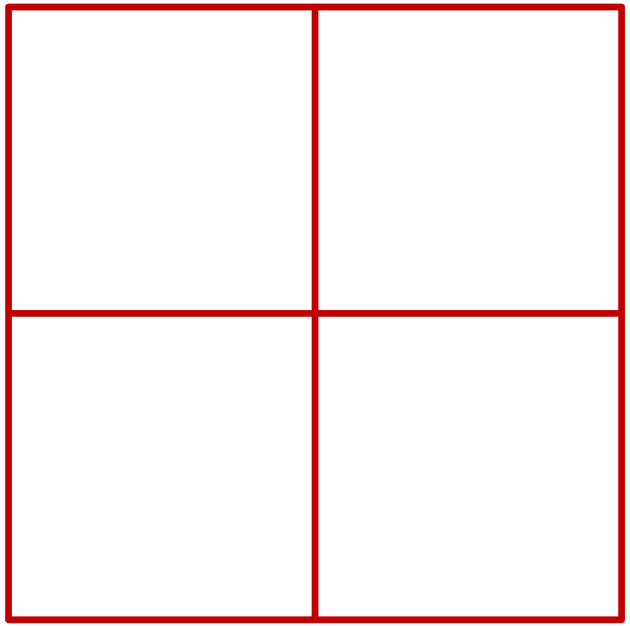
Generation Systems - Assign Values



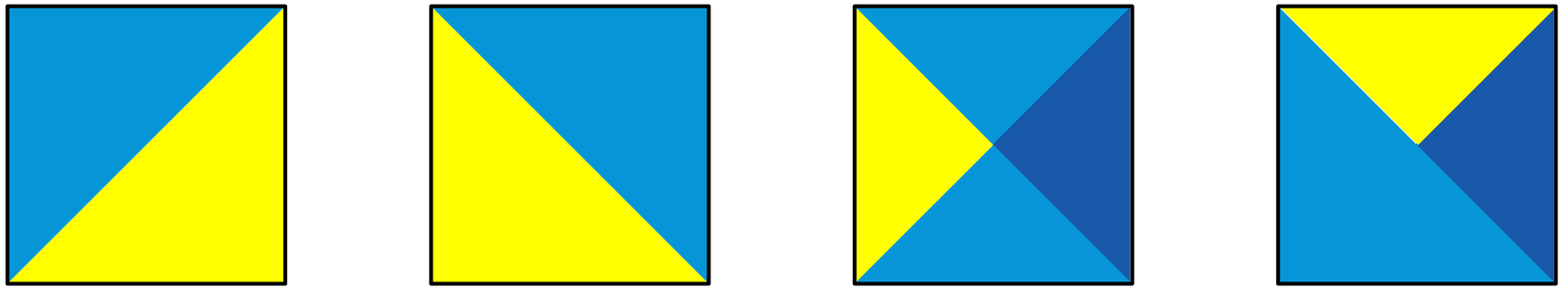


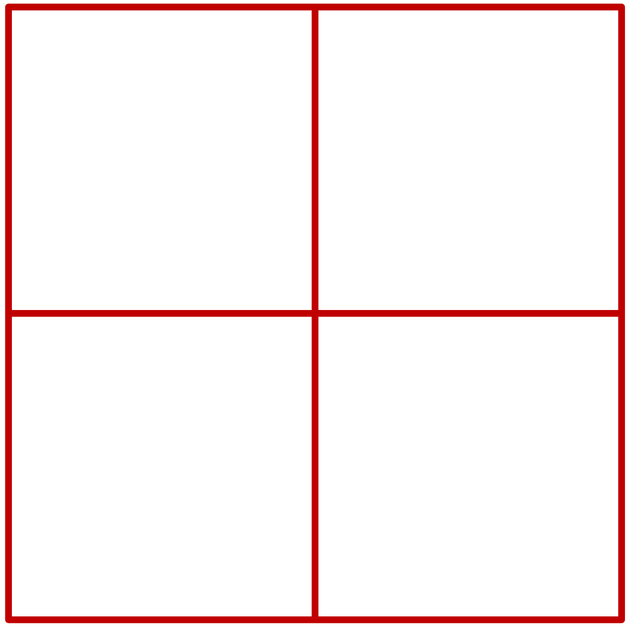
Generation Systems - Assign Values



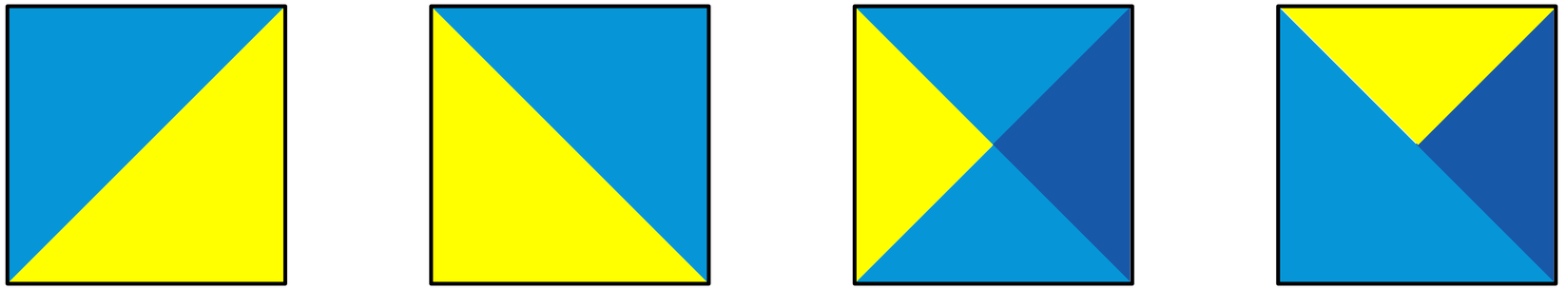


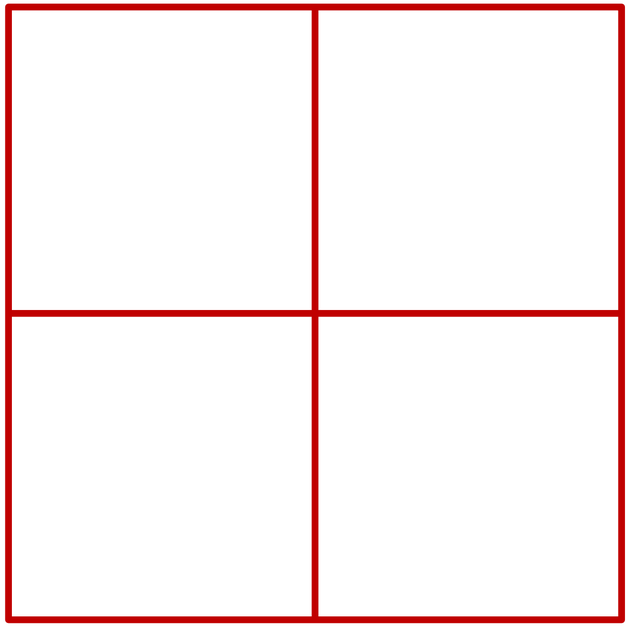
Generation Systems - Assign Values



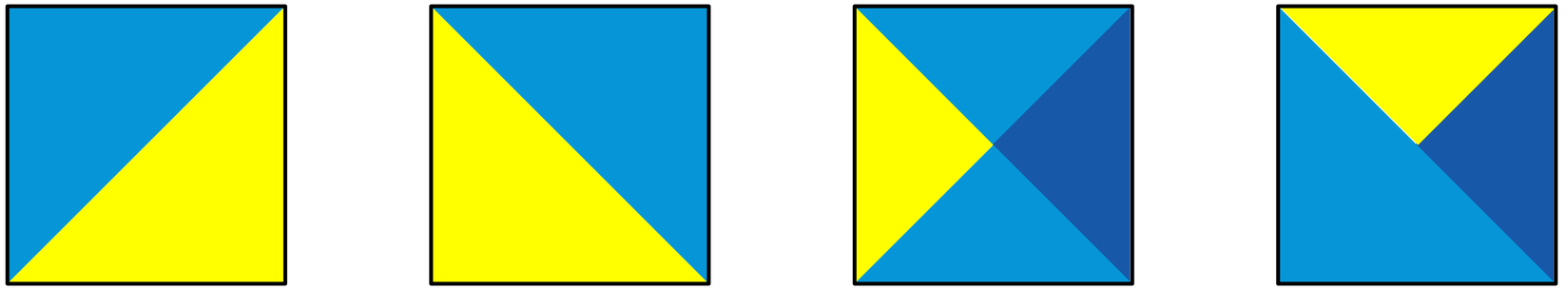


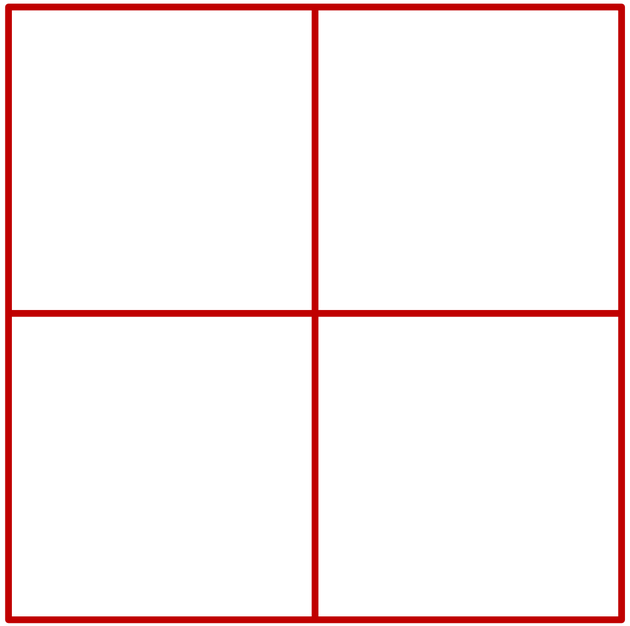
Generation Systems - Assign Values



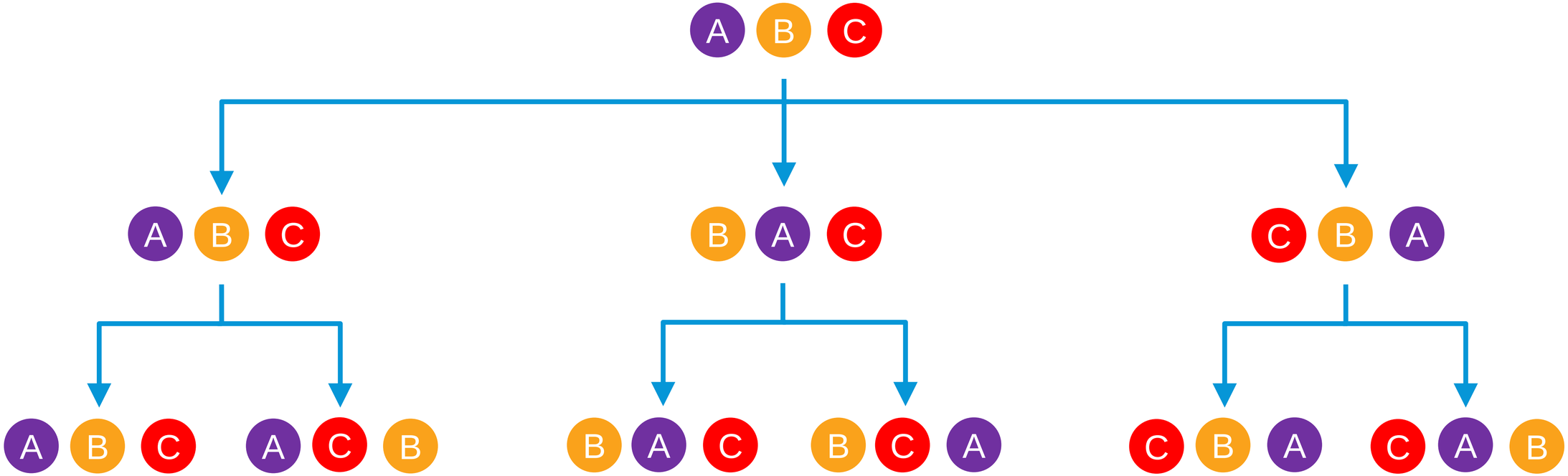


Generation Systems - Assign Values





Permutations Explained



Permutations Explained

720 items under (6!)

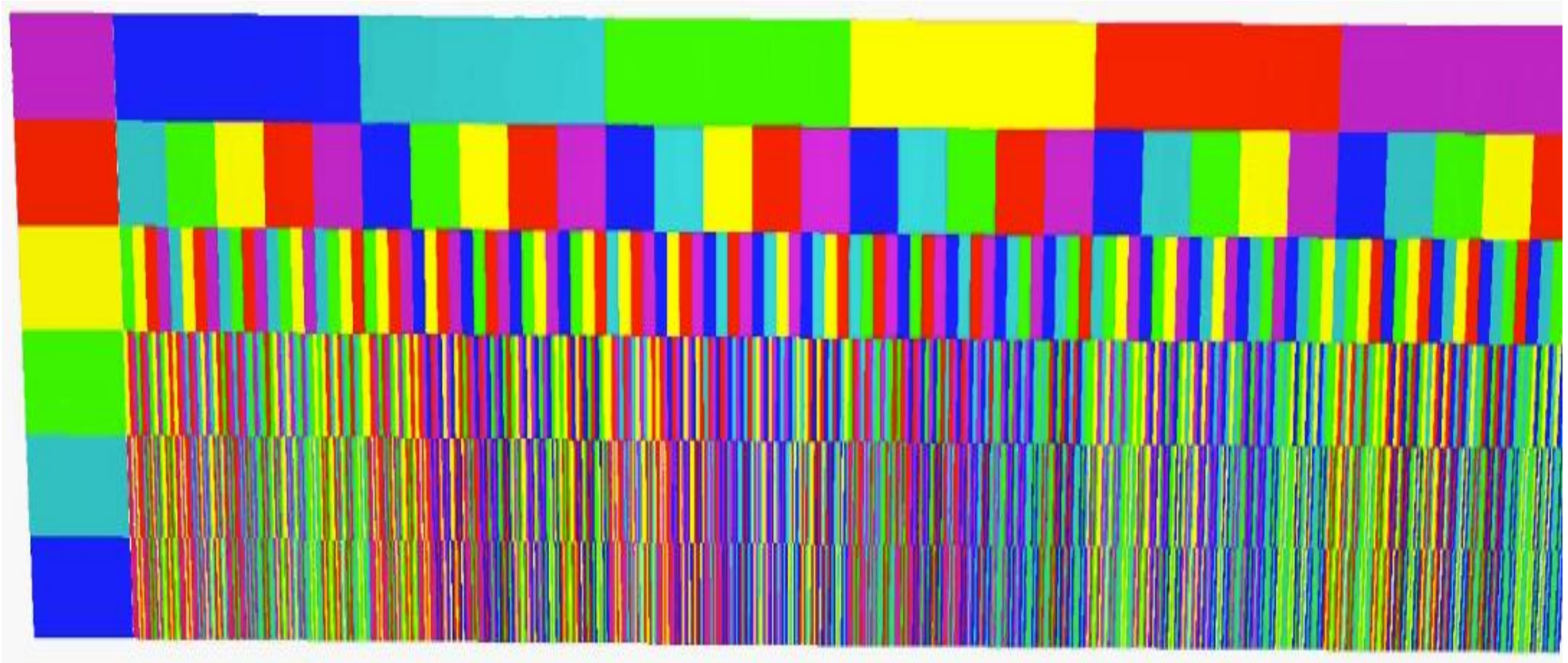
120 items under (5!)

24 items under (4!)

6 items under (3!)

2 items under (2!)

1 item under (1!)



Evaluations & Results – Student Seating

The design results are a list of dictionaries

For each dictionary in the list of layouts:

1. Get the location point
2. Get any students to avoid
3. For each student to avoid, get the associated dictionary
4. For each associated dictionary, get the location point
5. For each avoidance point, get the distance to the associated location point
6. For the distances, if the value is less than the accepted min, return 1, otherwise zero
7. Sum the result
8. Sum the value from all the dictionaries

Evaluations & Results – Student Seating

The design results are a list of dictionaries

1

For each dictionary in the list of layouts:

1. Get the location point
2. Get any students to avoid
3. For each student to avoid, get the associated dictionary
4. For each associated dictionary, get the location point
5. For each avoidance point, get the distance to the associated location point
6. For the distances, if the value is less than the accepted min, return 1, otherwise zero
7. Sum the result
8. Sum the value from all the dictionaries

Evaluations & Results – Student Seating

The design results are a list of dictionaries

3,4,8

1

For each dictionary in the list of layouts:

1. Get the location point
2. Get any students to avoid
3. For each student to avoid, get the associated dictionary
4. For each associated dictionary, get the location point
5. For each avoidance point, get the distance to the associated location point
6. For the distances, if the value is less than the accepted min, return 1, otherwise zero
7. Sum the result
8. Sum the value from all the dictionaries

Evaluations & Results – Student Seating

The design results are a list of dictionaries

3,4,8

1

For each dictionary in the list of layouts:

1. Get the location point
2. Get any students to avoid
3. For each student to avoid, get the associated dictionary
4. For each associated dictionary, get the location point
5. For each avoidance point, get the distance to the associated location point
6. For the distances, if the value is less than the accepted min, return 1, otherwise zero
7. Sum the result
8. Sum the value from all the dictionaries

Evaluations & Results – Student Seating

The design results are a list of dictionaries

3,4,8

1

3

4

8

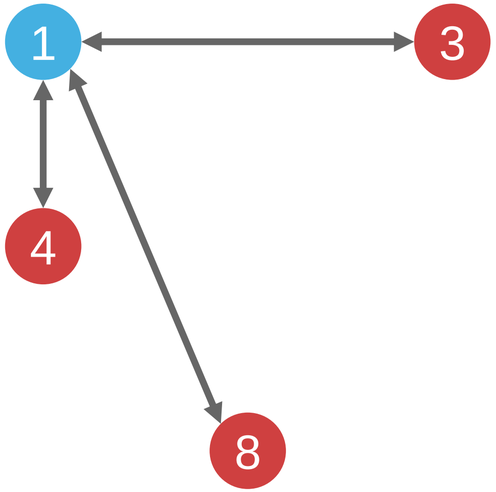
For each dictionary in the list of layouts:

1. Get the location point
2. Get any students to avoid
3. For each student to avoid, get the associated dictionary
4. For each associated dictionary, get the location point
5. For each avoidance point, get the distance to the associated location point
6. For the distances, if the value is less than the accepted min, return 1, otherwise zero
7. Sum the result
8. Sum the value from all the dictionaries

Evaluations & Results – Student Seating

The design results are a list of dictionaries

3,4,8



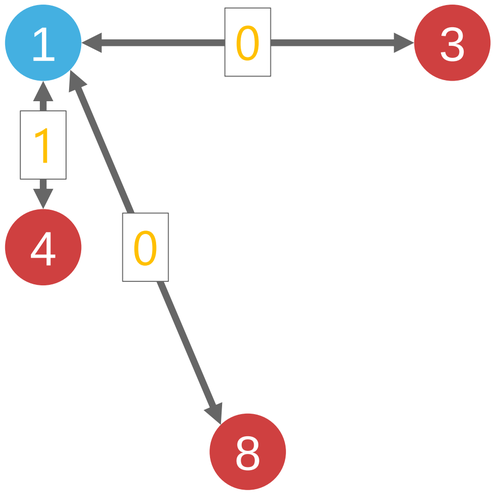
For each dictionary in the list of layouts:

1. Get the location point
2. Get any students to avoid
3. For each student to avoid, get the associated dictionary
4. For each associated dictionary, get the location point
5. For each avoidance point, get the distance to the associated location point
6. For the distances, if the value is less than the accepted min, return 1, otherwise zero
7. Sum the result
8. Sum the value from all the dictionaries

Evaluations & Results – Student Seating

The design results are a list of dictionaries

3,4,8



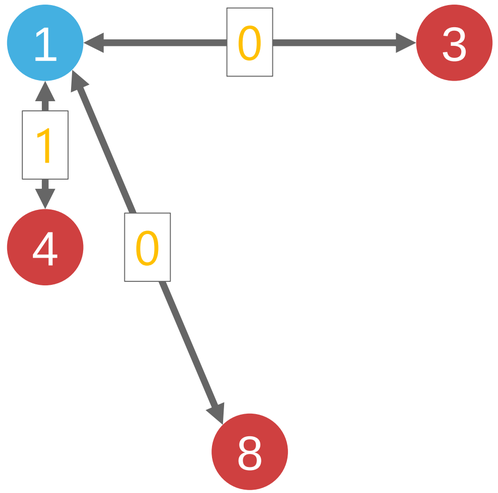
For each dictionary in the list of layouts:

1. Get the location point
2. Get any students to avoid
3. For each student to avoid, get the associated dictionary
4. For each associated dictionary, get the location point
5. For each avoidance point, get the distance to the associated location point
6. For the distances, if the value is less than the accepted min, return 1, otherwise zero
7. Sum the result
8. Sum the value from all the dictionaries

Evaluations & Results – Student Seating

The design results are a list of dictionaries

3,4,8



0+1+0 = 1

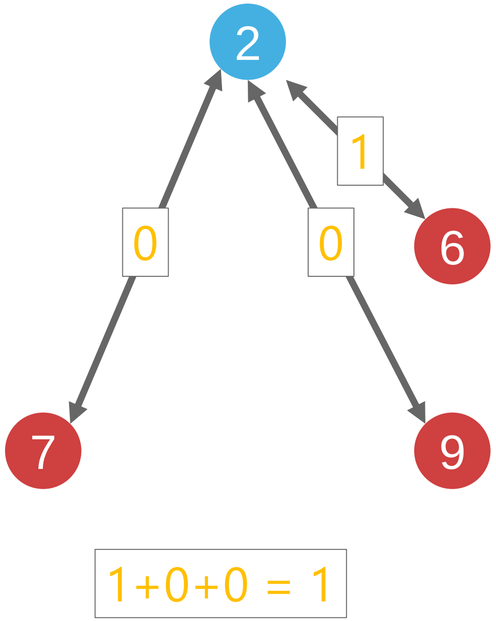
For each dictionary in the list of layouts:

1. Get the location point
2. Get any students to avoid
3. For each student to avoid, get the associated dictionary
4. For each associated dictionary, get the location point
5. For each avoidance point, get the distance to the associated location point
6. For the distances, if the value is less than the accepted min, return 1, otherwise zero
7. Sum the result
8. Sum the value from all the dictionaries

Evaluations & Results – Student Seating

The design results are a list of dictionaries

6,7,9



1+0+0 = 1

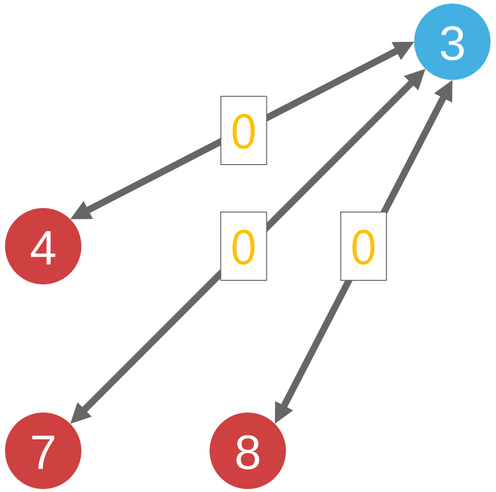
For each dictionary in the list of layouts:

1. Get the location point
2. Get any students to avoid
3. For each student to avoid, get the associated dictionary
4. For each associated dictionary, get the location point
5. For each avoidance point, get the distance to the associated location point
6. For the distances, if the value is less than the accepted min, return 1, otherwise zero
7. Sum the result
8. Sum the value from all the dictionaries

Evaluations & Results – Student Seating

The design results are a list of dictionaries

4,7,9



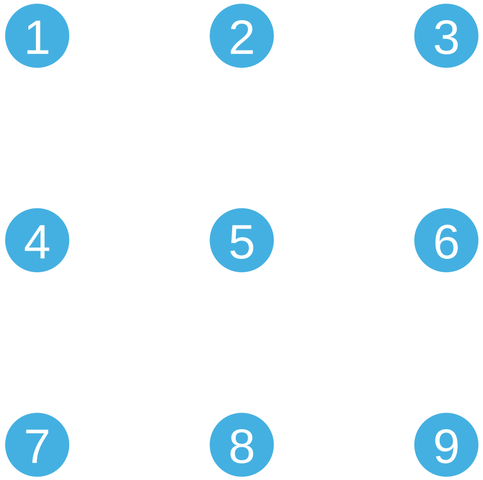
0+0+0 =0

For each dictionary in the list of layouts:

1. Get the location point
2. Get any students to avoid
3. For each student to avoid, get the associated dictionary
4. For each associated dictionary, get the location point
5. For each avoidance point, get the distance to the associated location point
6. For the distances, if the value is less than the accepted min, return 1, otherwise zero
7. Sum the result
8. Sum the value from all the dictionaries

Evaluations & Results – Student Seating

The design results are a list of dictionaries



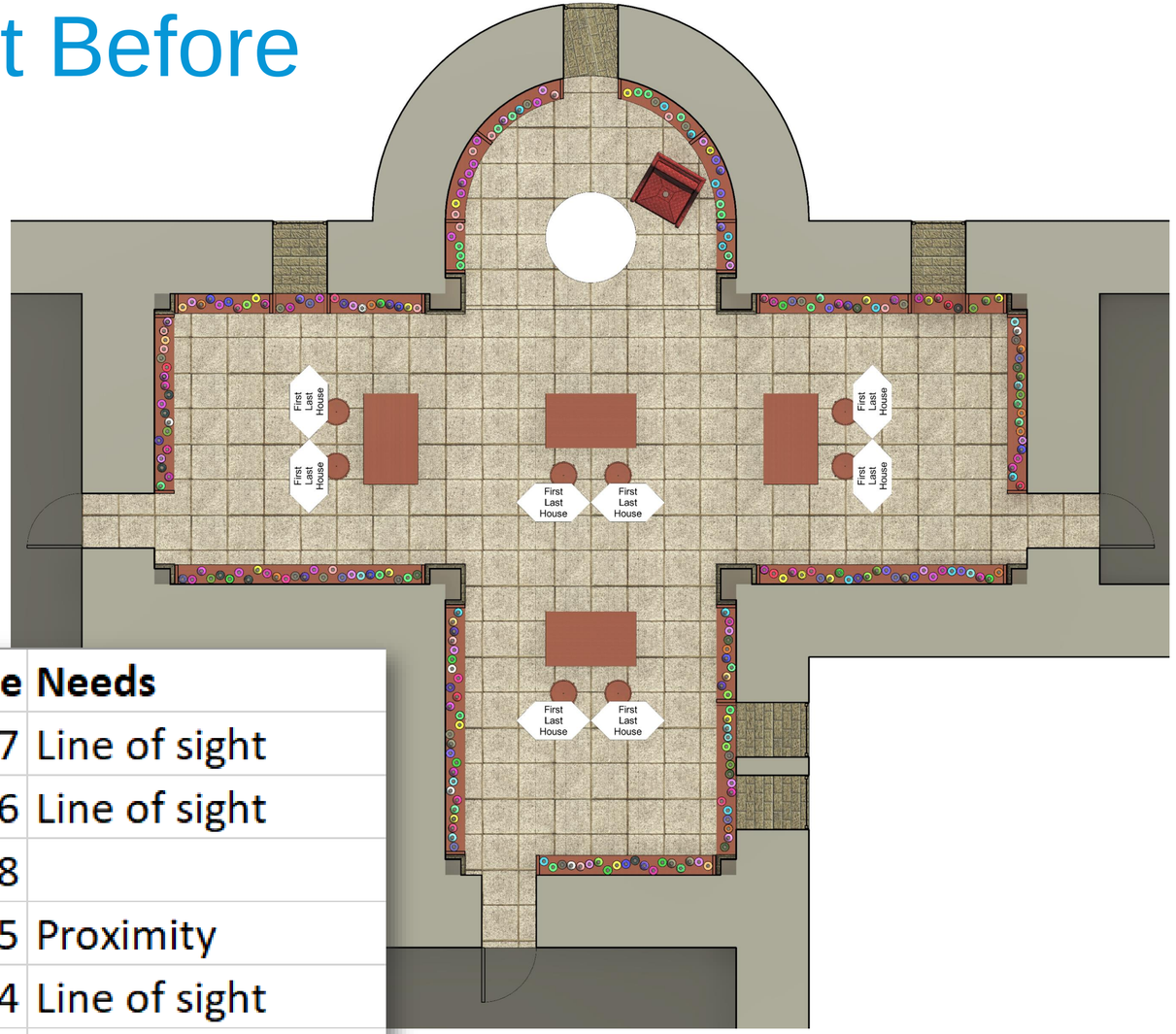
1 + 1 + 0 + 3 + 3 + 1 + 2 + 0 + 1 = 12

For each dictionary in the list of layouts:

1. Get the location point
2. Get any students to avoid
3. For each student to avoid, get the associated dictionary
4. For each associated dictionary, get the location point
5. For each avoidance point, get the distance to the associated location point
6. For the distances, if the value is less than the accepted min, return 1, otherwise zero
7. Sum the result
8. Sum the value from all the dictionaries

Data Set Before

- **Model** of Potions Dungeon showing seat locations and tags.
- **Student data**: reference number, first name; last name; house; who they shouldn't sit near; who they MUSTEN sit near; and what they need to be safe or learn optimally.

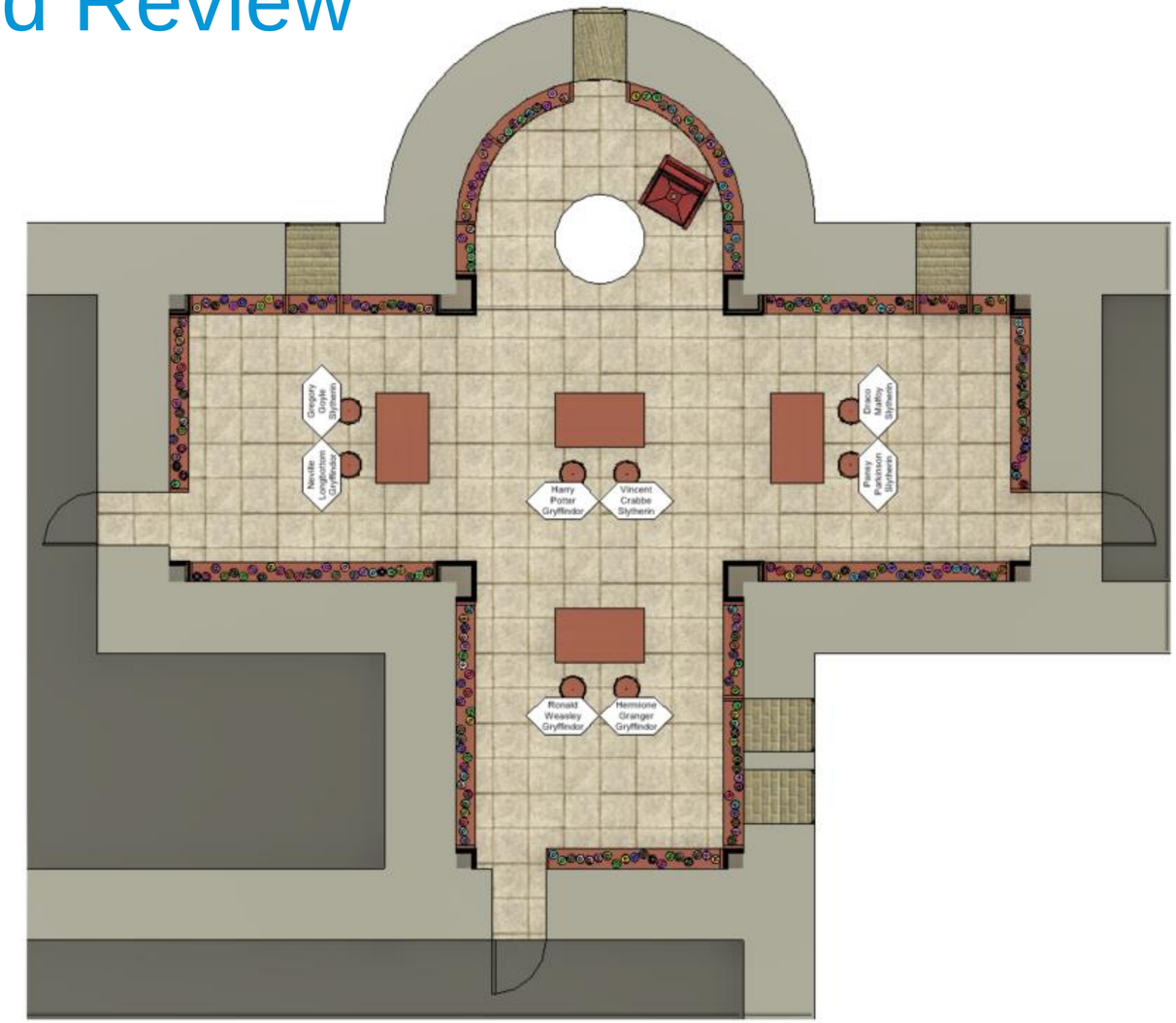


No.	First	Last	House	Avoid	Separate	Needs
1	Harry	Potter	Gryffindor	2,3	7	Line of sight
2	Ronald	Weasley	Gryffindor	1,3	6	Line of sight
3	Hermione	Granger	Gryffindor	1,2	8	
4	Neville	Longbottom	Gryffindor	6,7	5	Proximity
5	Vincent	Crabbe	Slytherin	6,7	4	Line of sight
6	Gregory	Goyle	Slytherin	5,7	2	Line of sight
7	Draco	Malfoy	Slytherin	2,4	1	Proximity
8	Pansy	Parkinson	Slytherin		3	

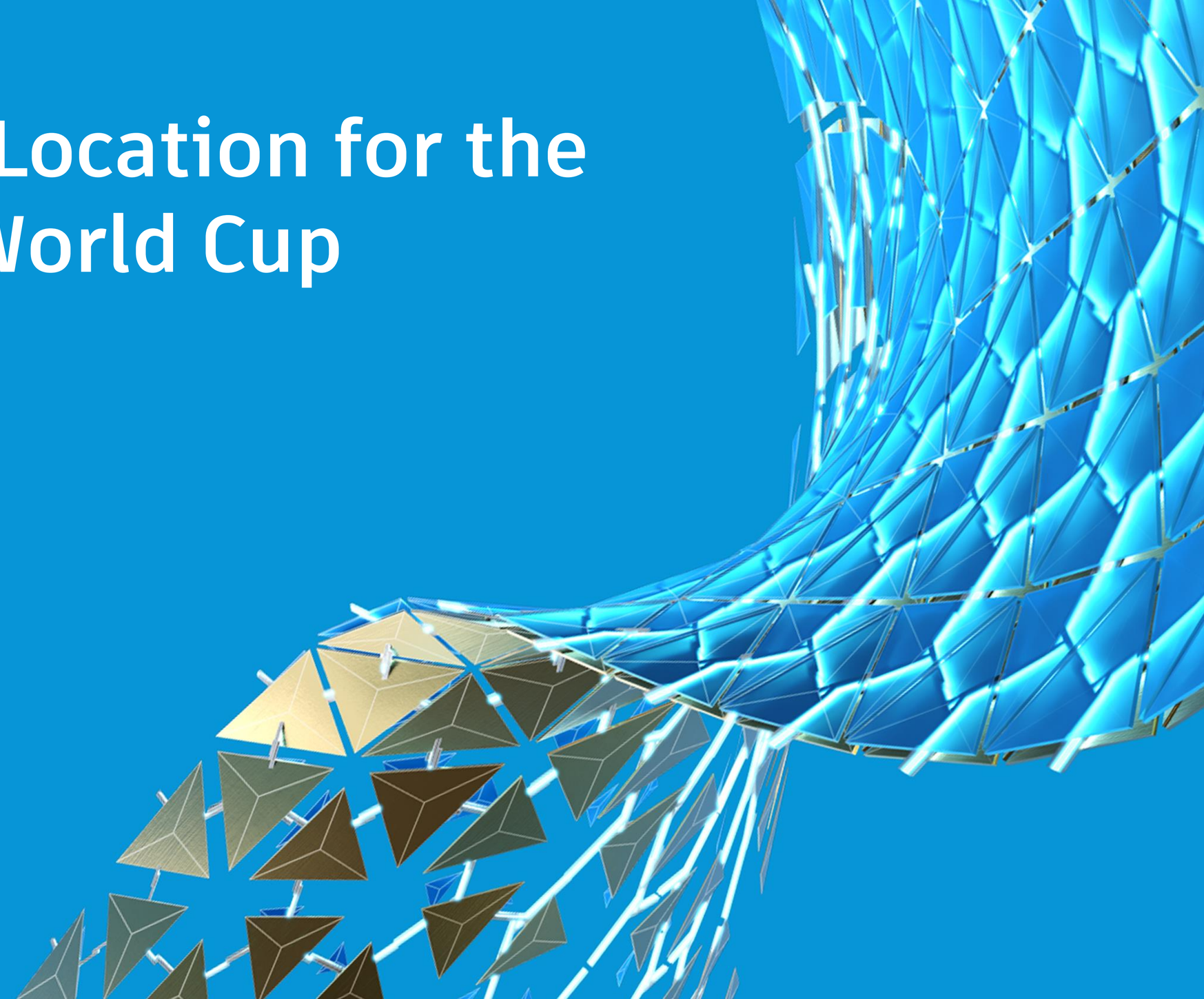
Let’s Build the Graph!

Notes and Review

- Small set ‘assignment’ studies can be accomplished quickly with permutations
- Results can be filtered on multiple axes
- Small changes in location can have BIG changes in outcomes
- An ‘optimum’ result does not ensure ALL optimum results are found.

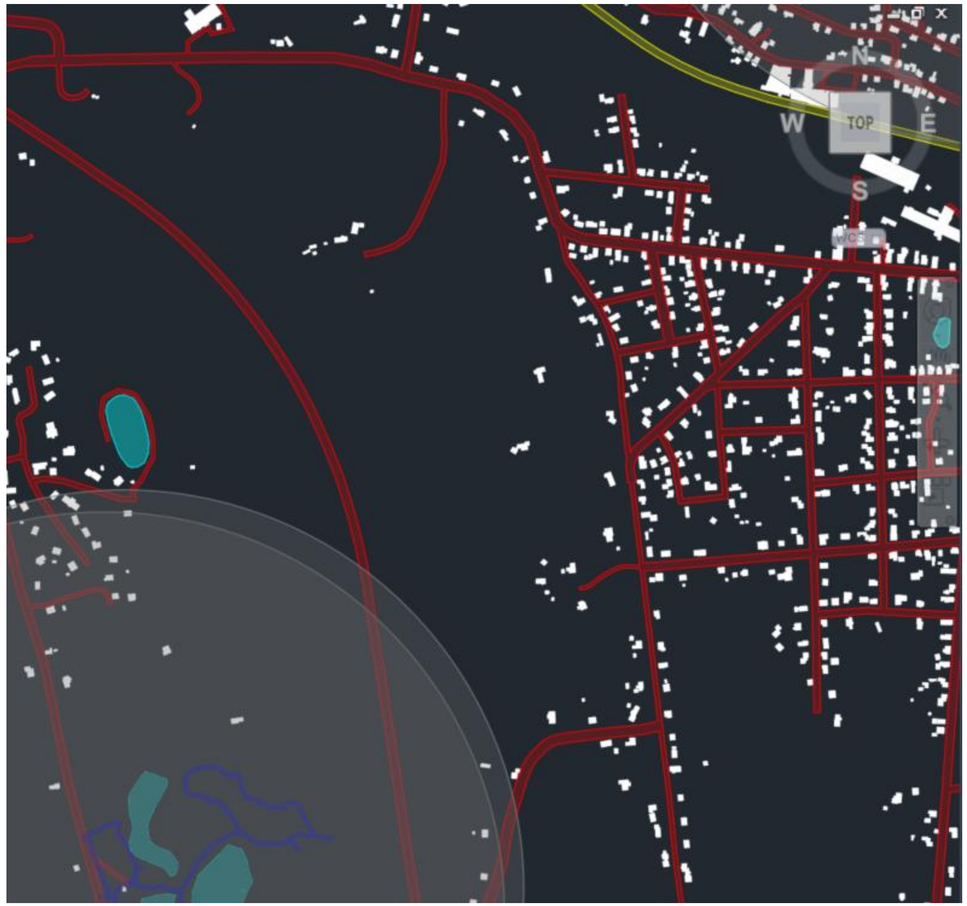


Problem 2: Location for the Quidditch World Cup

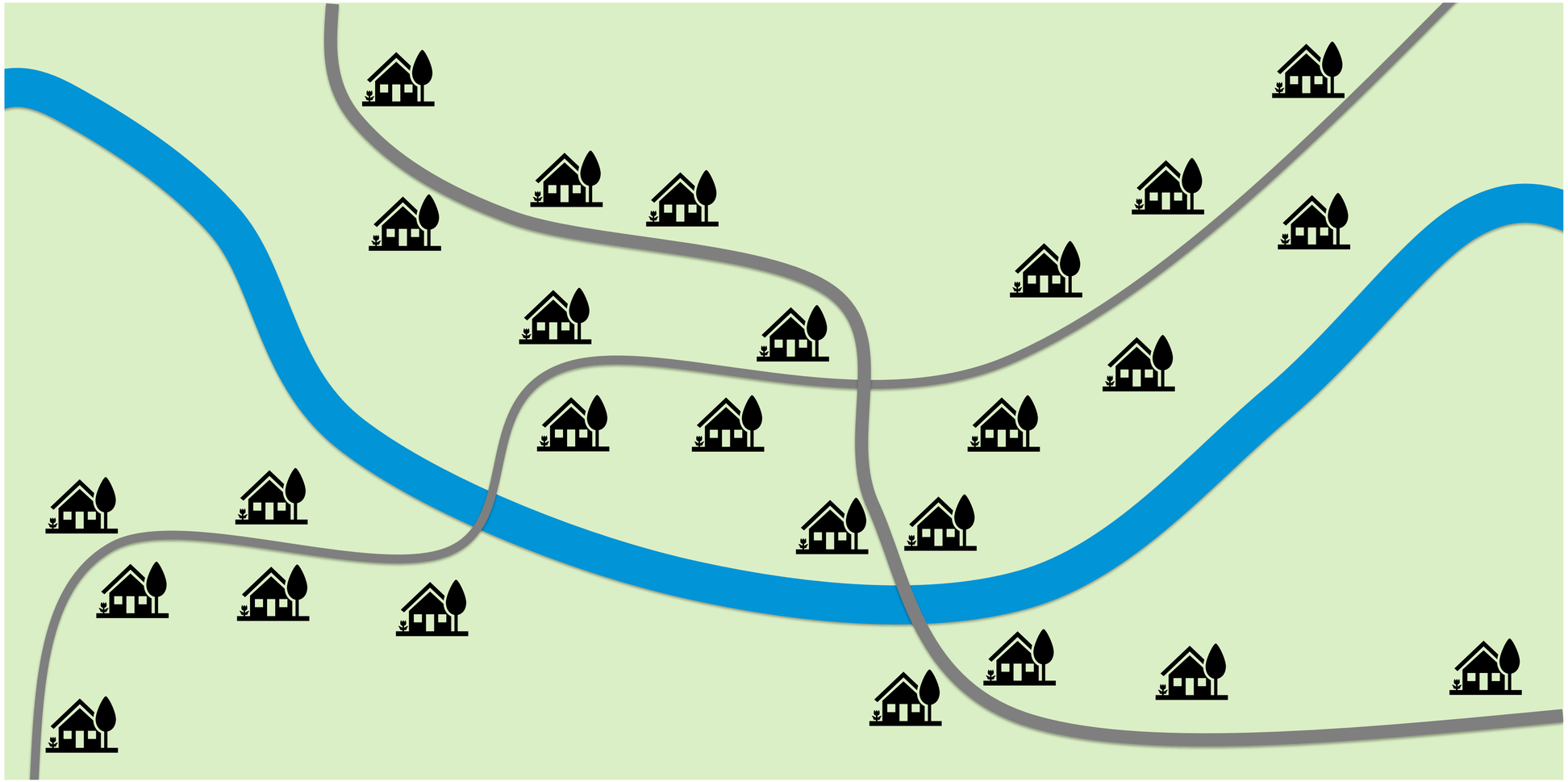


Fixed Option Studies

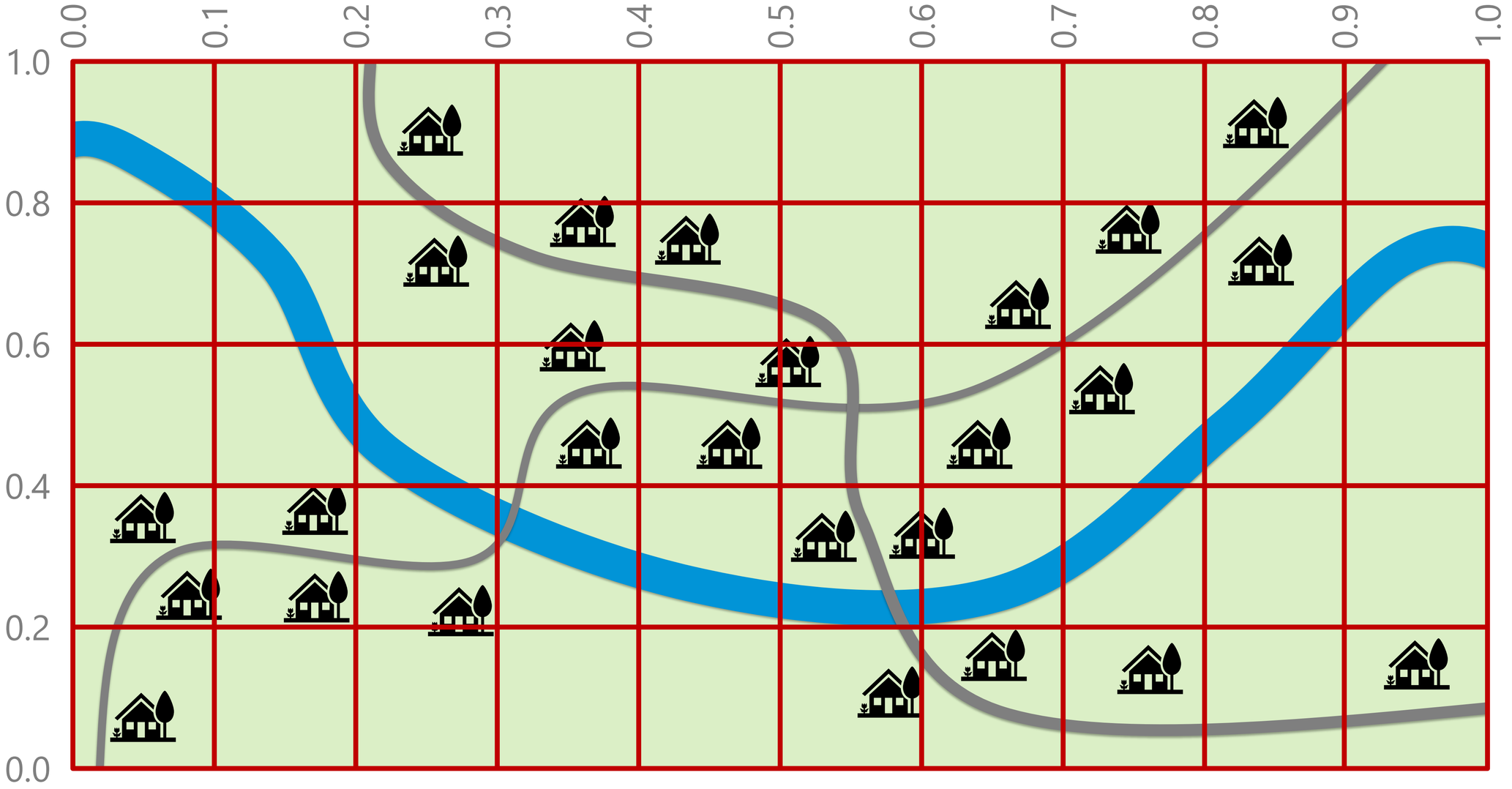
- Sometimes the number of options for your design is vast and won't be completely used, but provides a scalable solution set to work with.
- In this session, we will be using a known and quantified context to evaluate where in a sleepy town the Quidditch World Cup stadium & grounds should be located.
- We'll be evaluating each program element against the adjacent context.



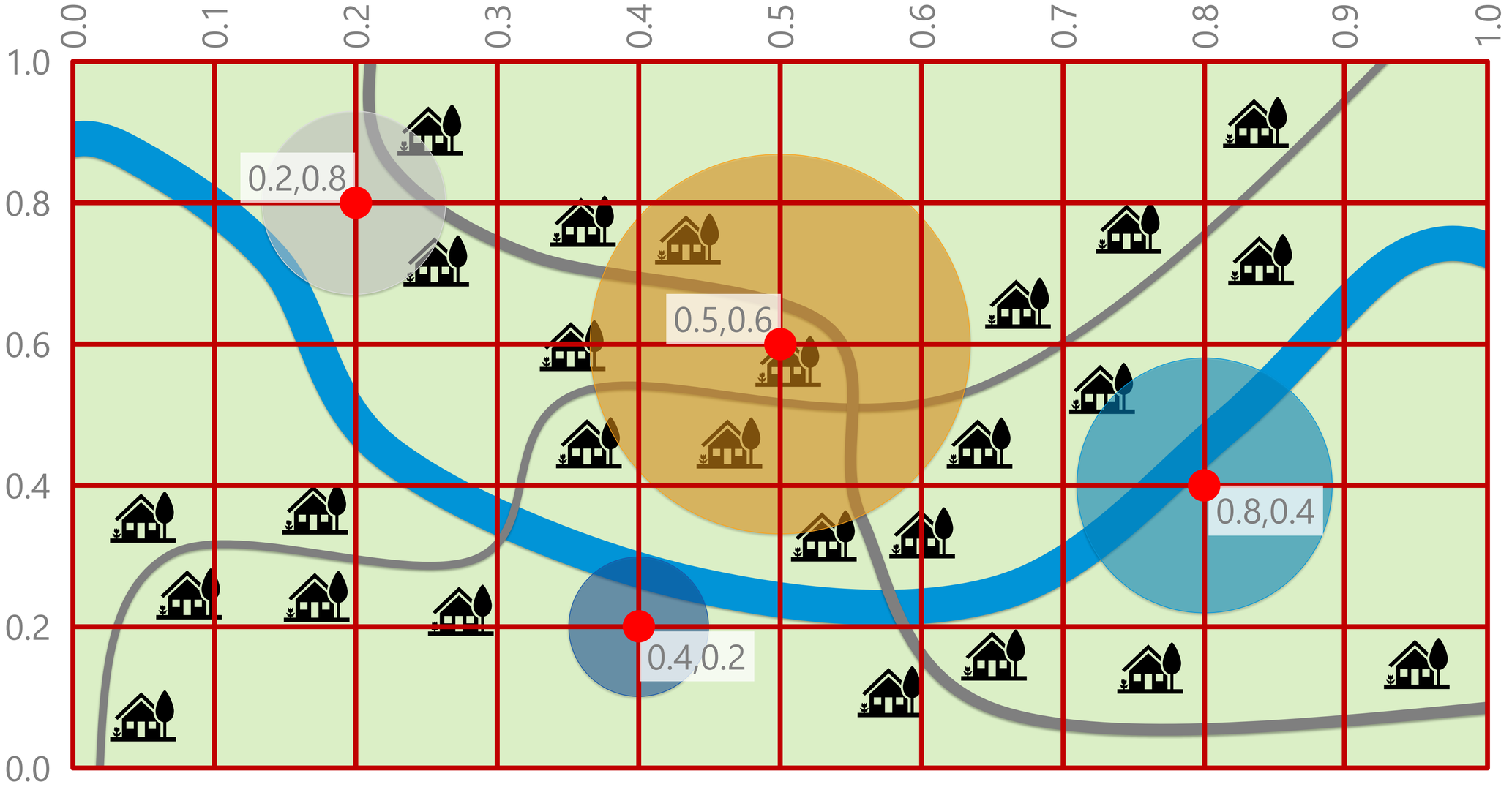
Generation Systems - Set Points



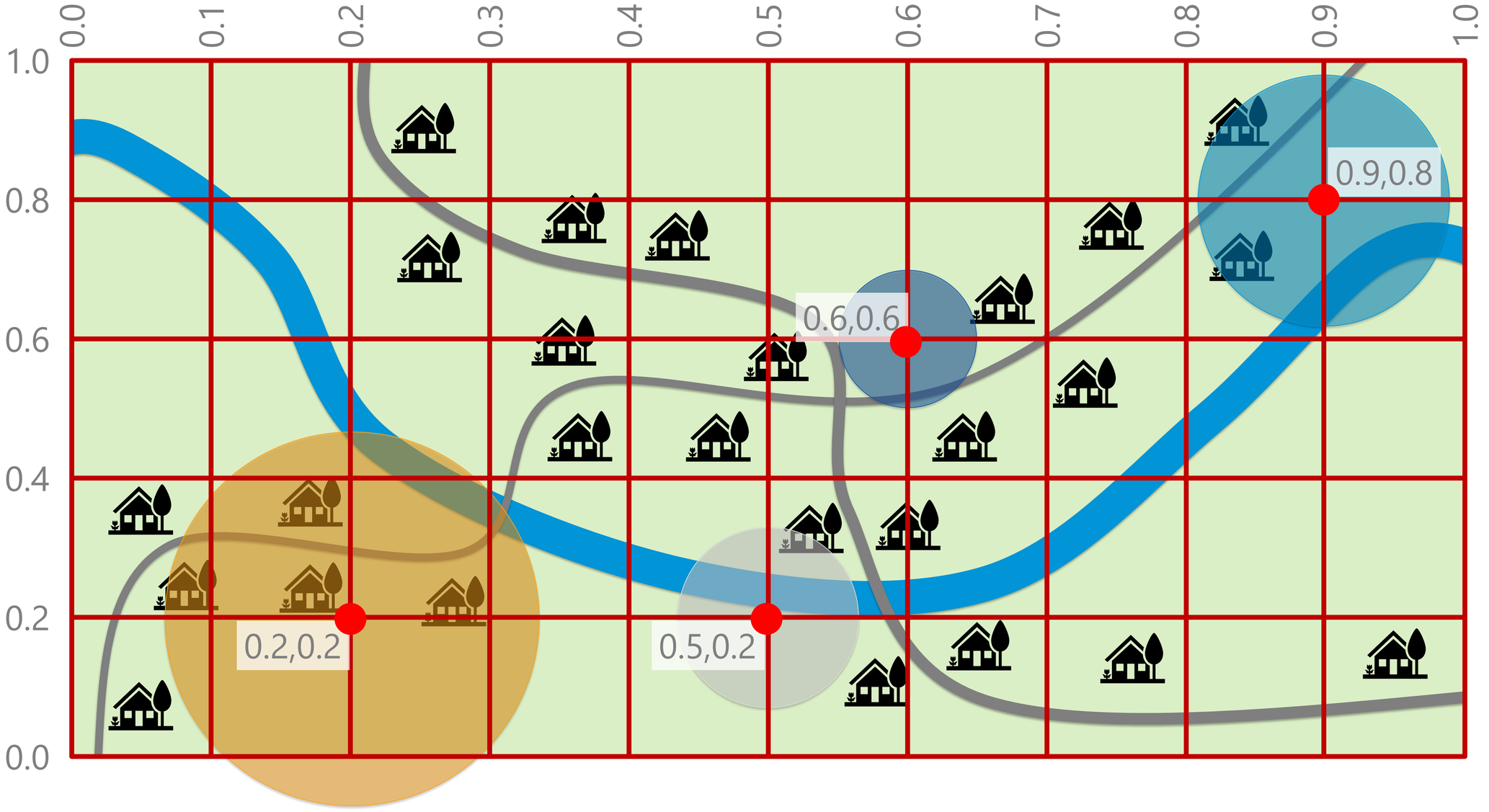
Generation Systems - Set Points



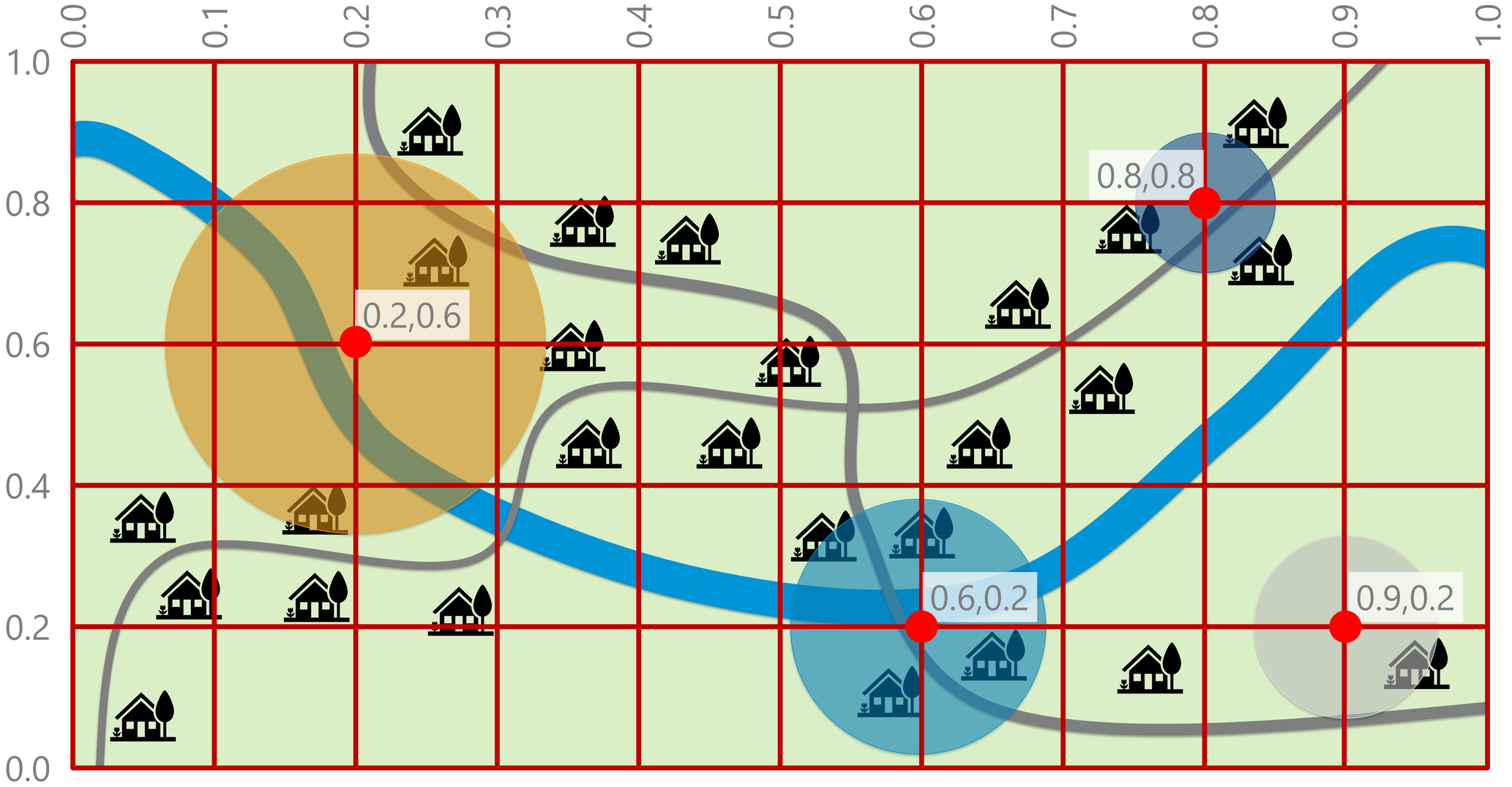
Generation Systems - Set Points



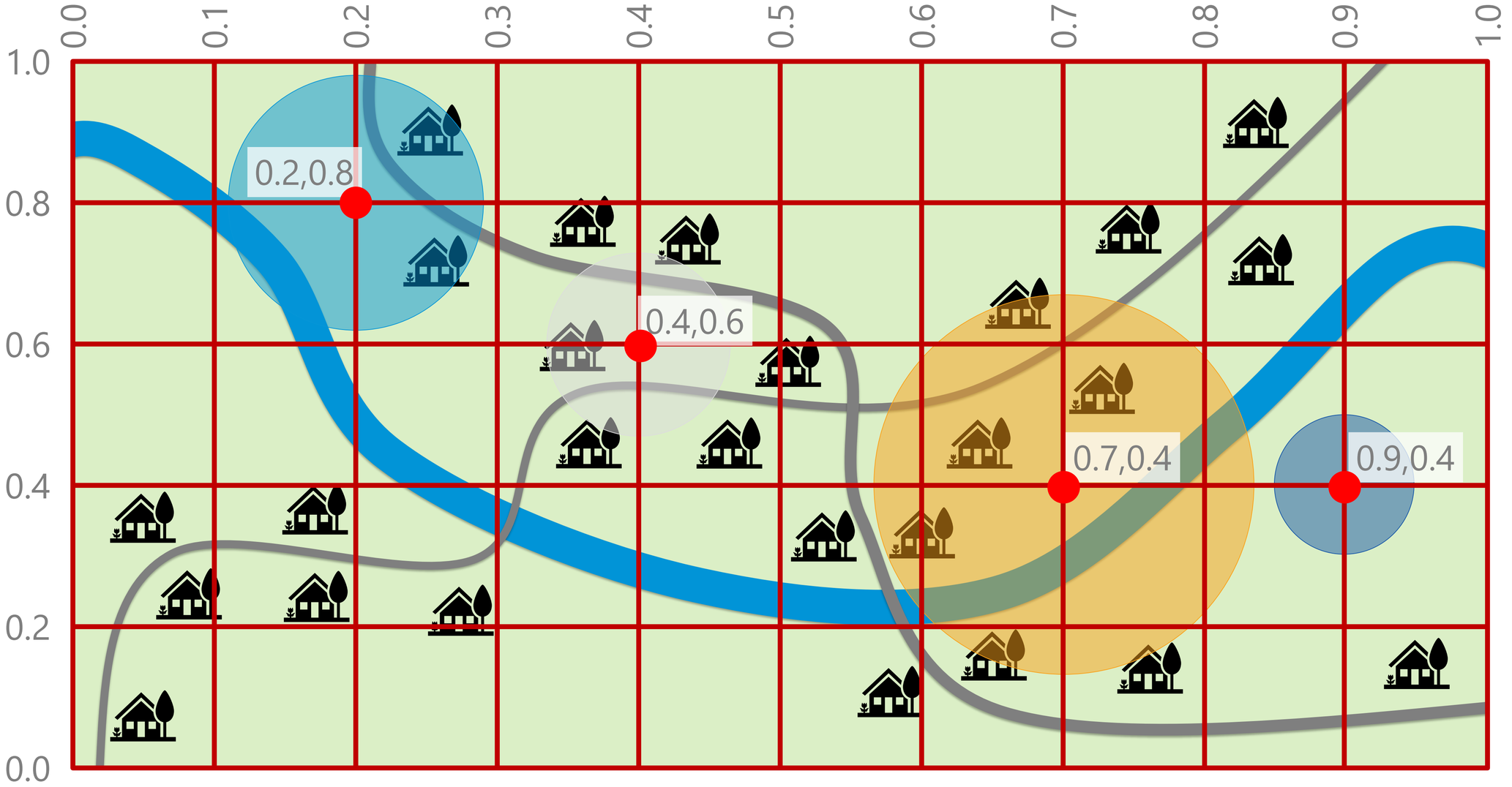
Generation Systems - Set Points



Generation Systems - Set Points

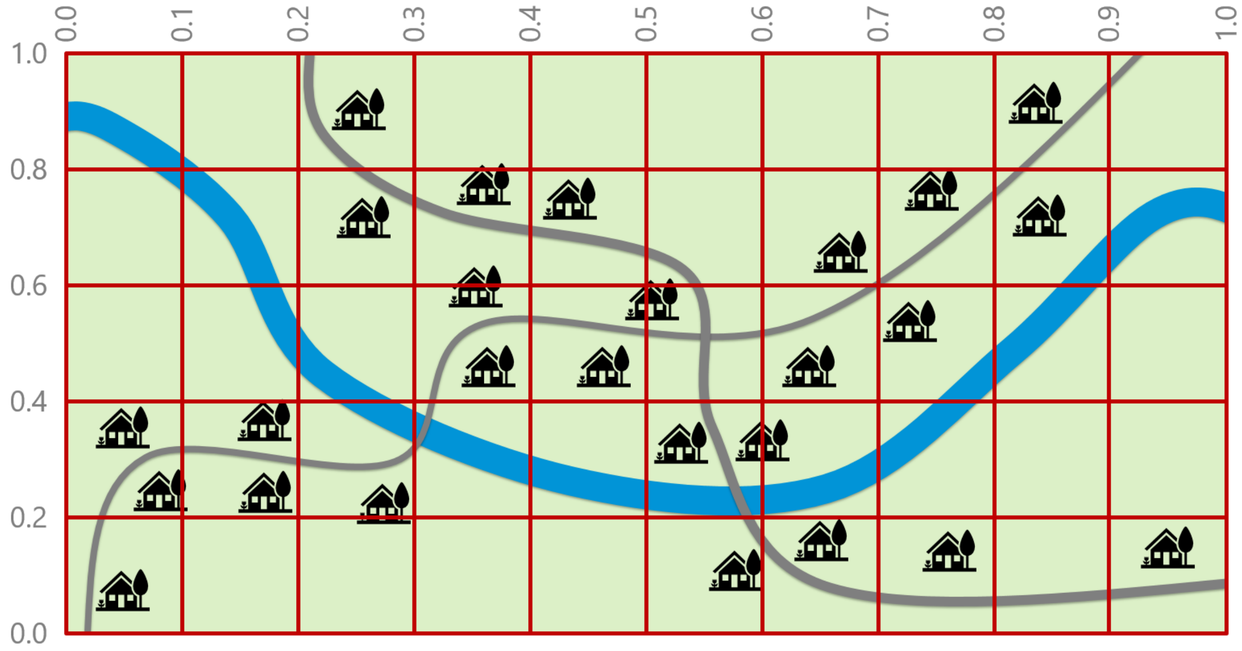


Generation Systems - Set Points



Evaluations & Results – Stadium Location

The site ‘cells’ have associated dictionaries, and the designs are surfaces

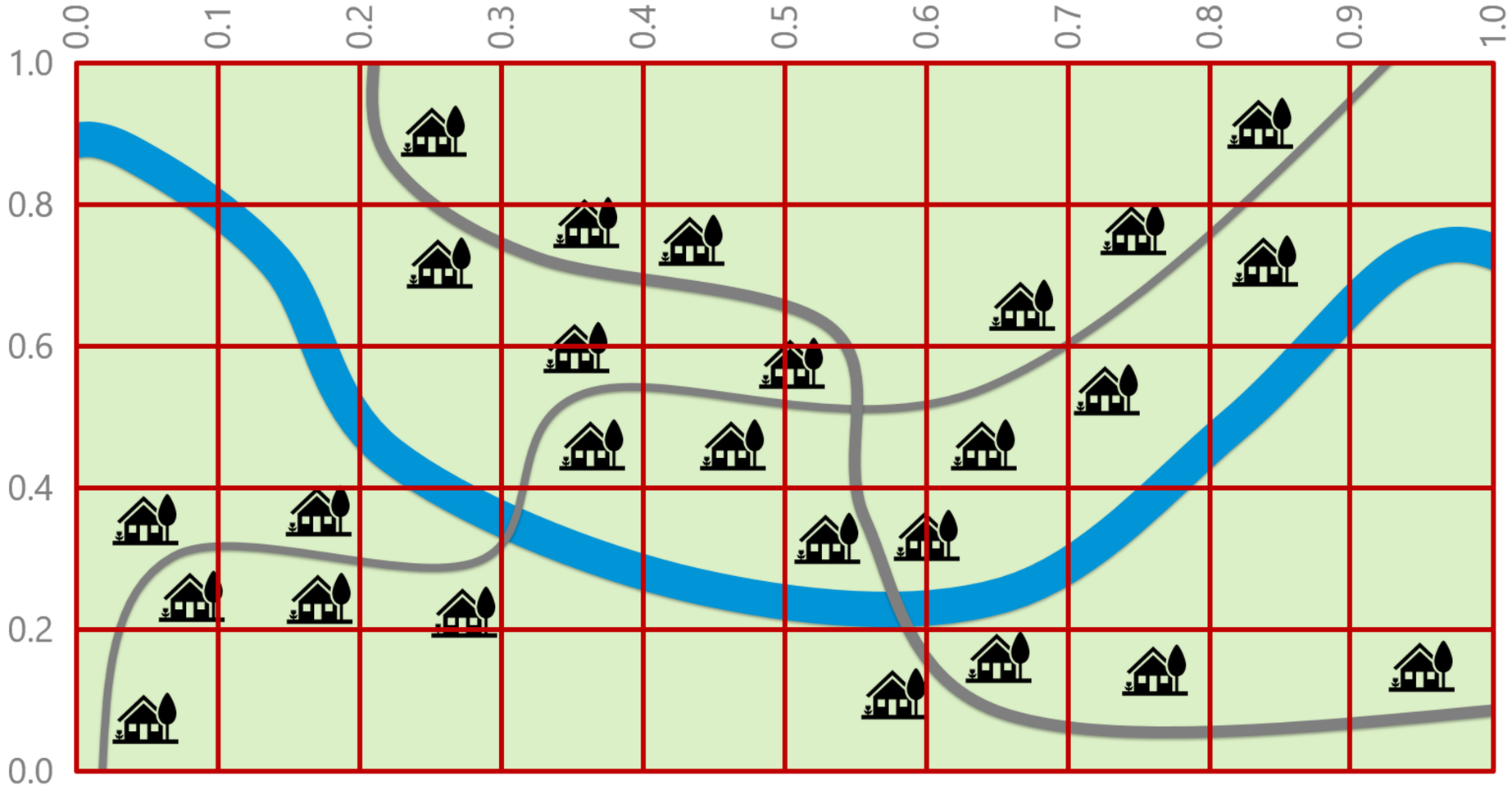


For the zones of influence:

1. Take the difference of each design surface and the extruded site bounds
2. Define the penalty by taking the remaining area of each design surface, summing the results and multiply by 100
3. Join the original design surfaces into a polysurface
4. Test all site cells for intersection with the polysurface
5. Filter out the list of dictionaries by the intersection test
6. For the remaining dictionaries, pull and sum the values for each metric and add the penalty

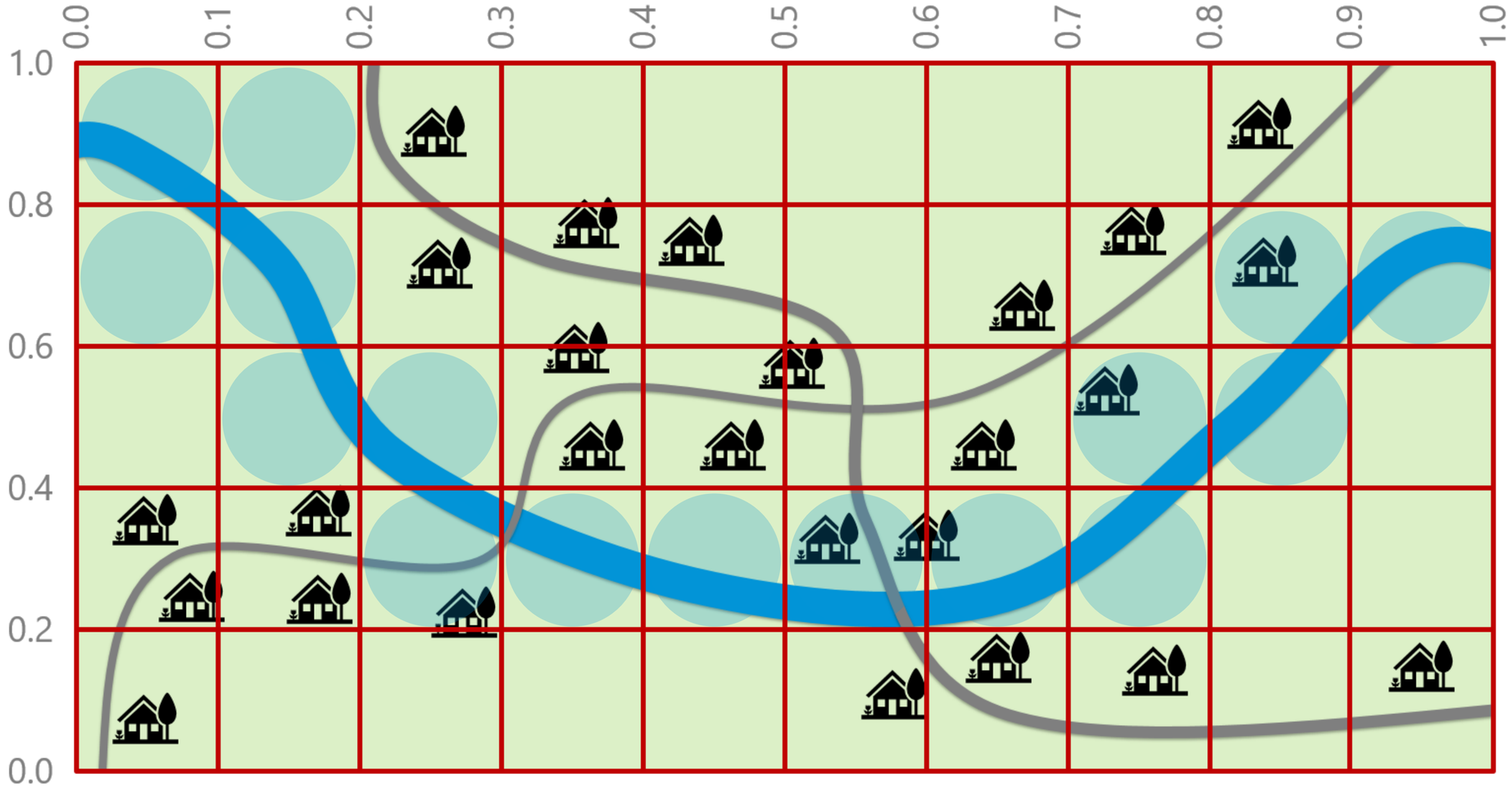
Evaluations & Results – Stadium Location

How to build the site dictionary



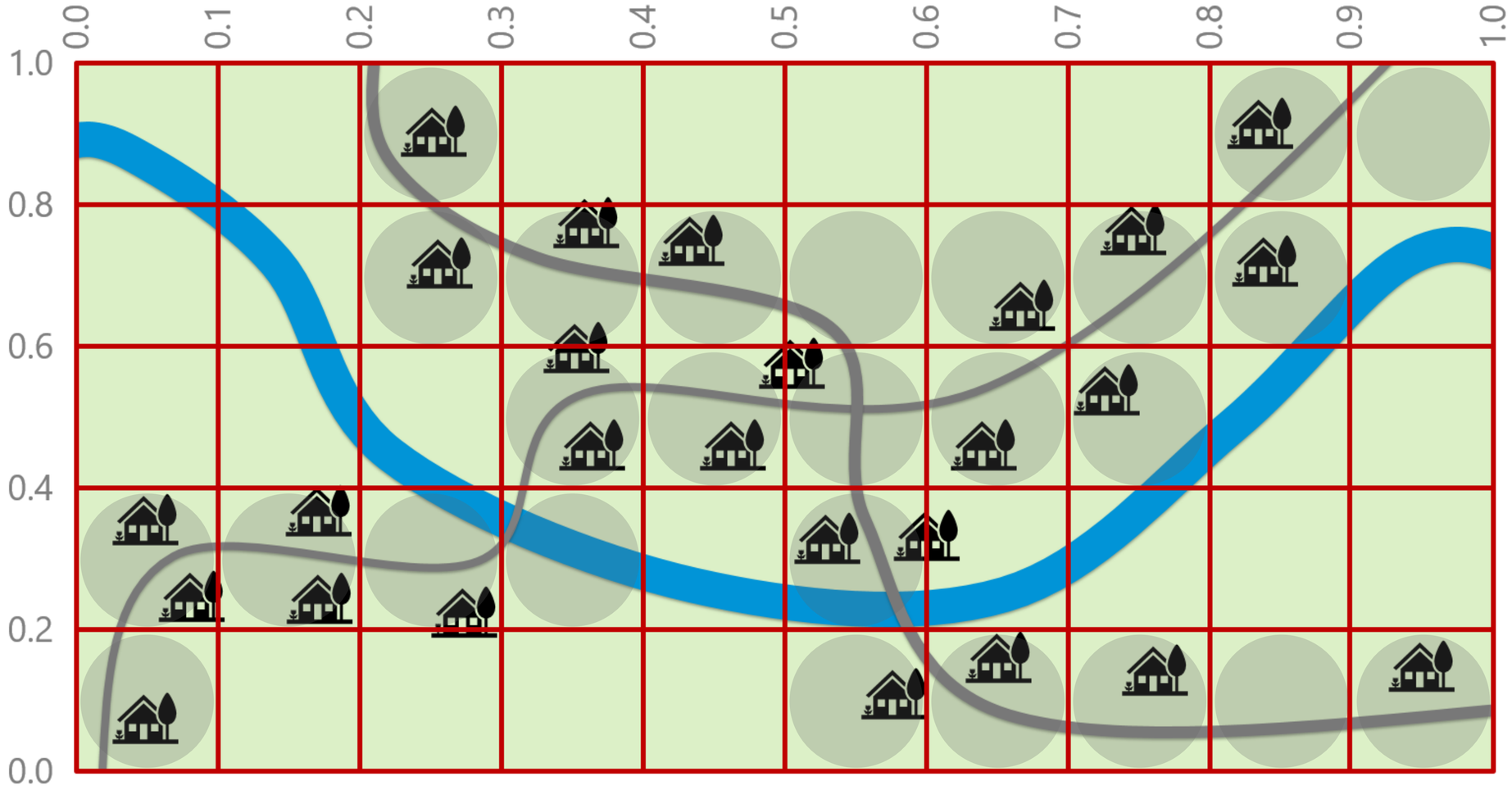
Evaluations & Results – Stadium Location

How to build the site dictionary



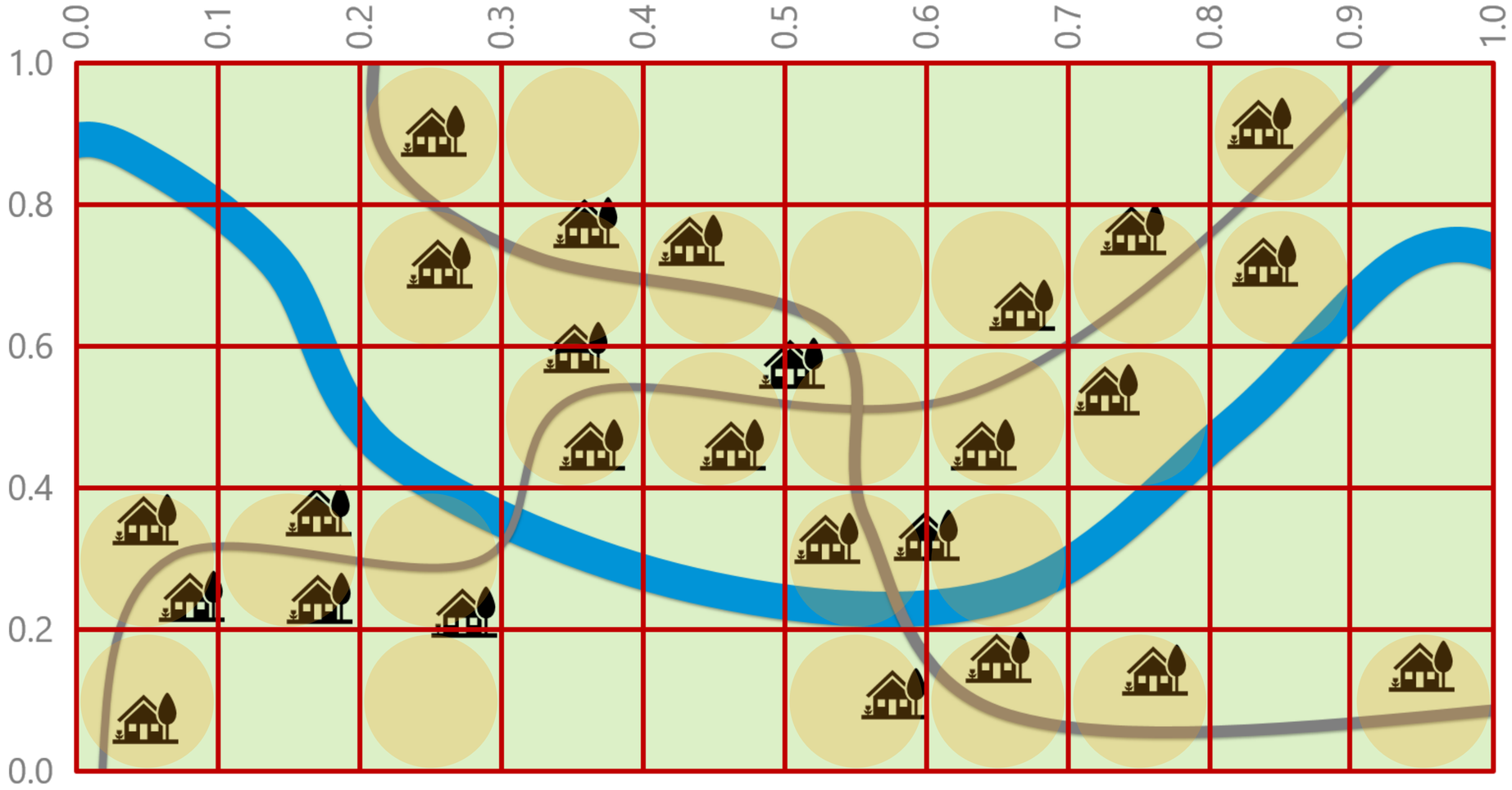
Evaluations & Results – Stadium Location

How to build the site dictionary



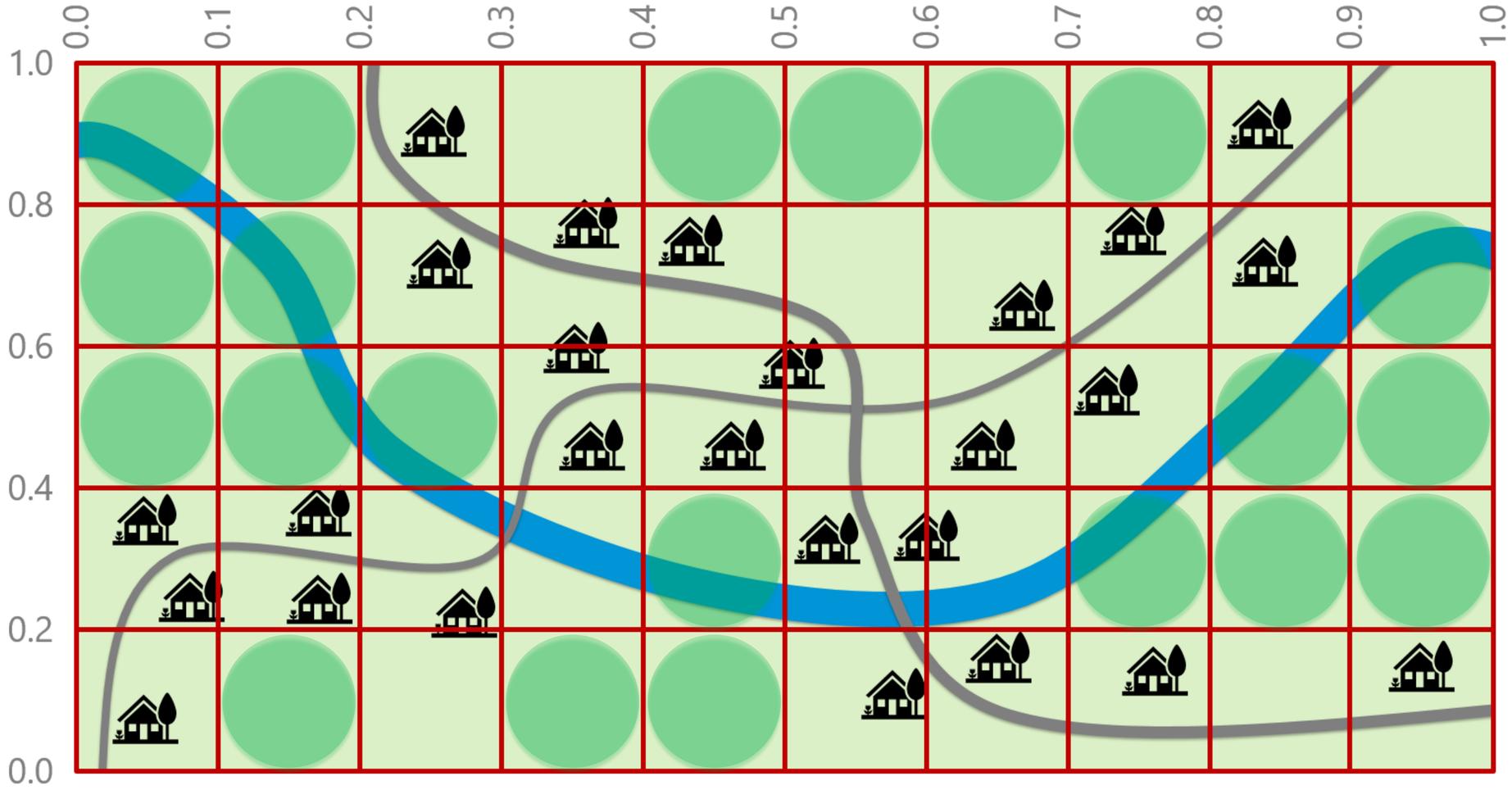
Evaluations & Results – Stadium Location

How to build the site dictionary



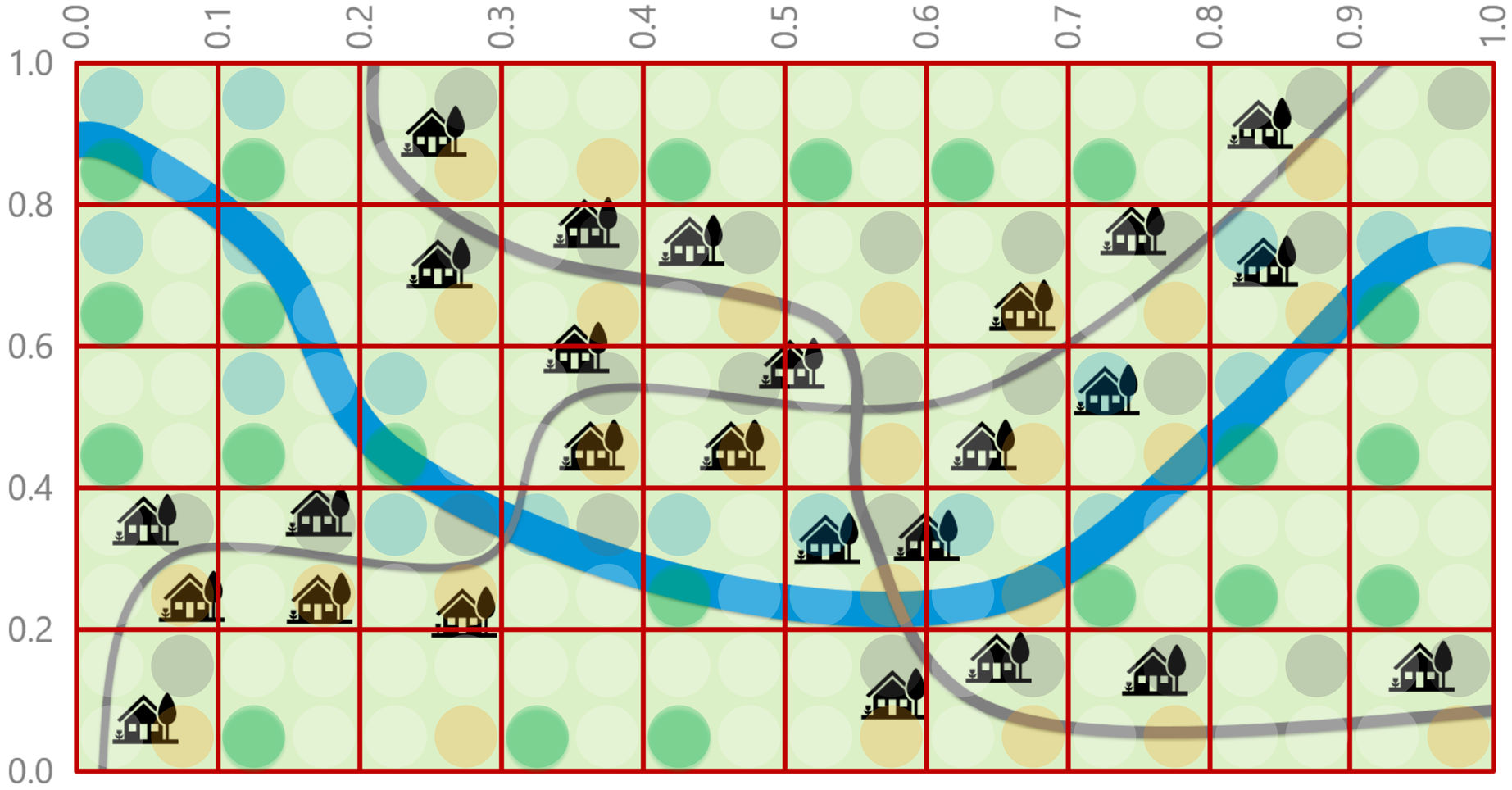
Evaluations & Results – Stadium Location

How to build the site dictionary



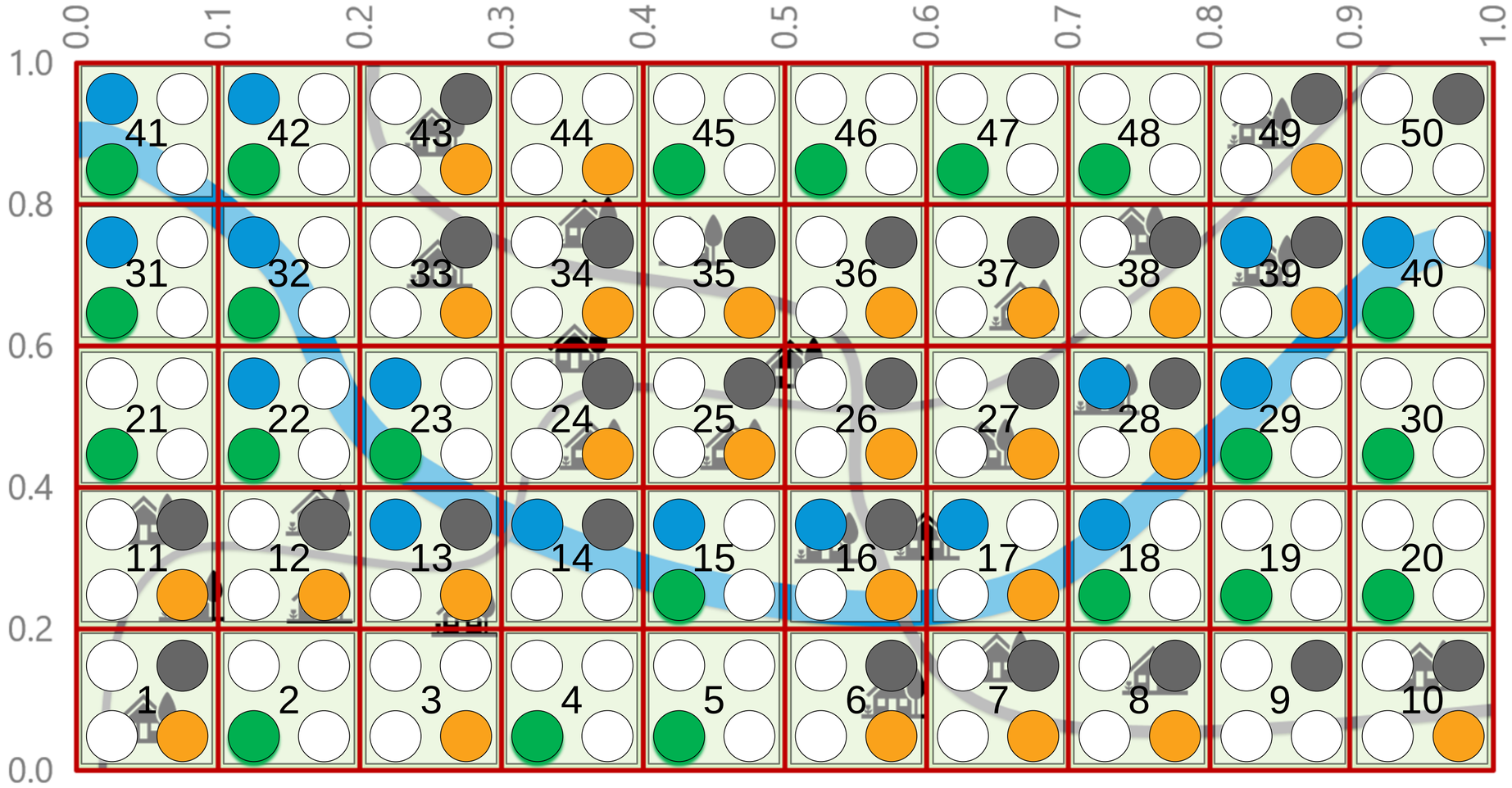
Evaluations & Results – Stadium Location

How to build the site dictionary



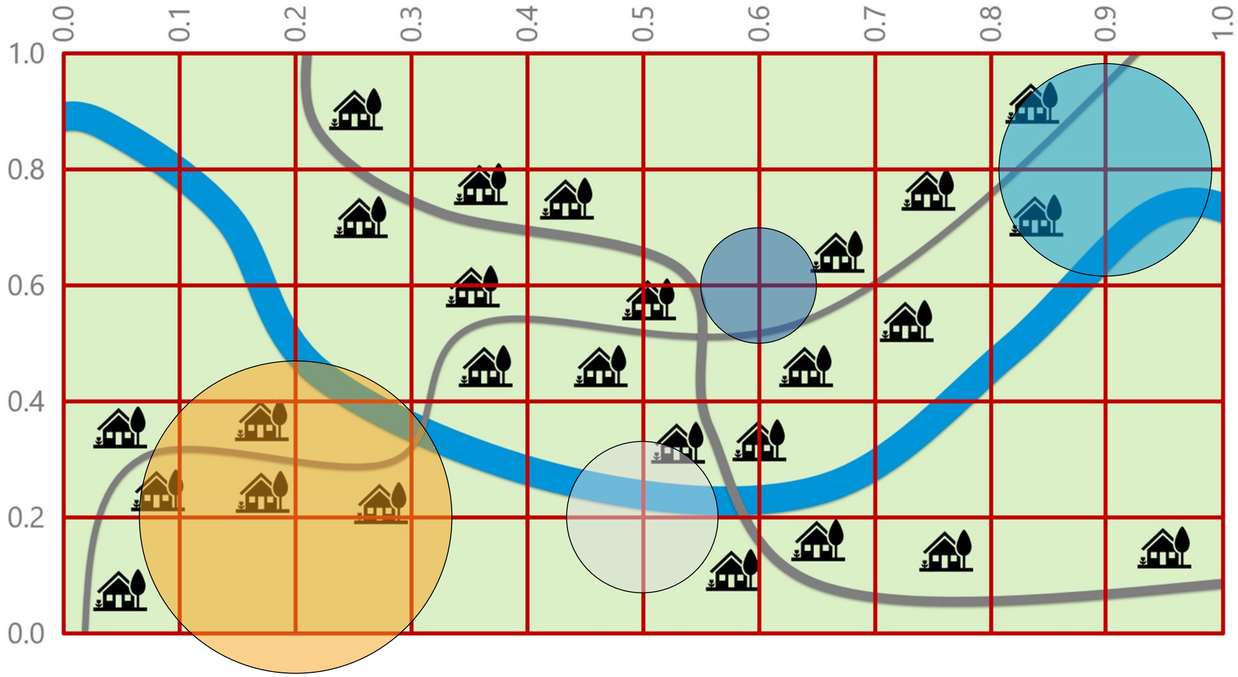
Evaluations & Results – Stadium Location

How to build the site dictionary



Evaluations & Results – Stadium Location

The site ‘cells’ have associated dictionaries, and the designs are surfaces

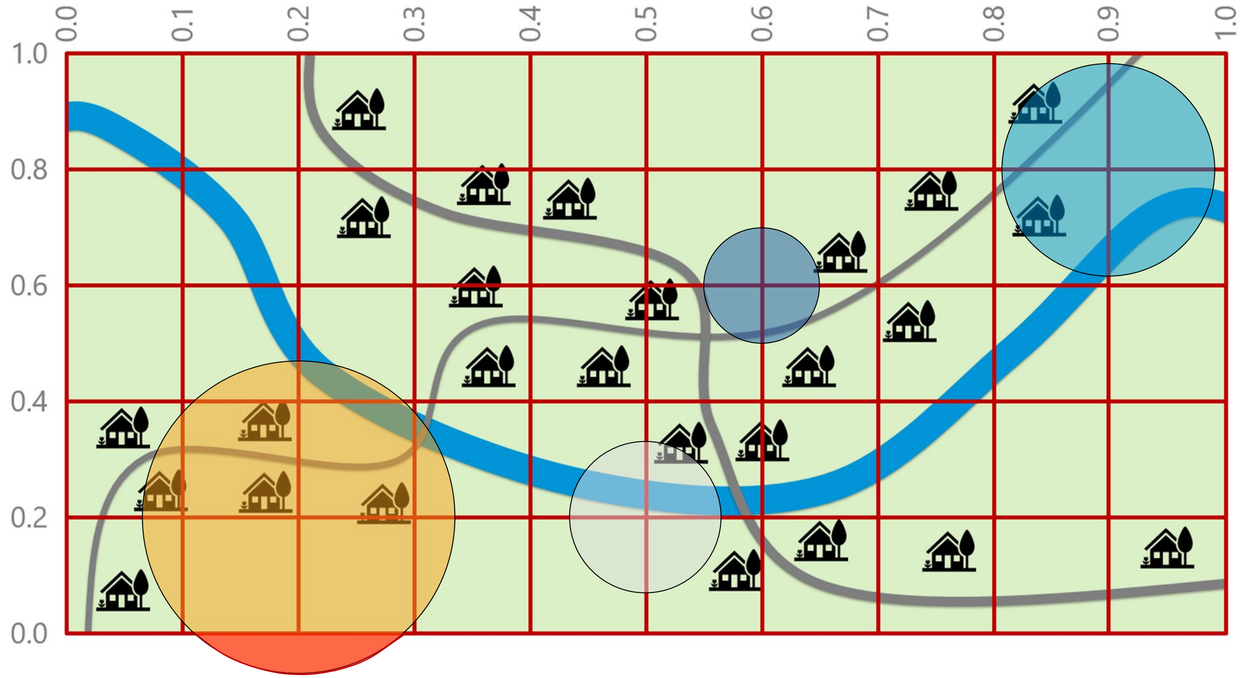


For the zones of influence:

1. Take the difference of each design surface and the extruded site bounds
2. Define the penalty by taking the remaining area of each design surface, summing the results and multiply by 100
3. Join the original design surfaces into a polysurface
4. Test all site cells for intersection with the polysurface
5. Filter out the list of dictionaries by the intersection test
6. For the remaining dictionaries, pull the values for each metric
7. Sum the results and add the penalty

Evaluations & Results – Stadium Location

The site ‘cells’ have associated dictionaries, and the designs are surfaces



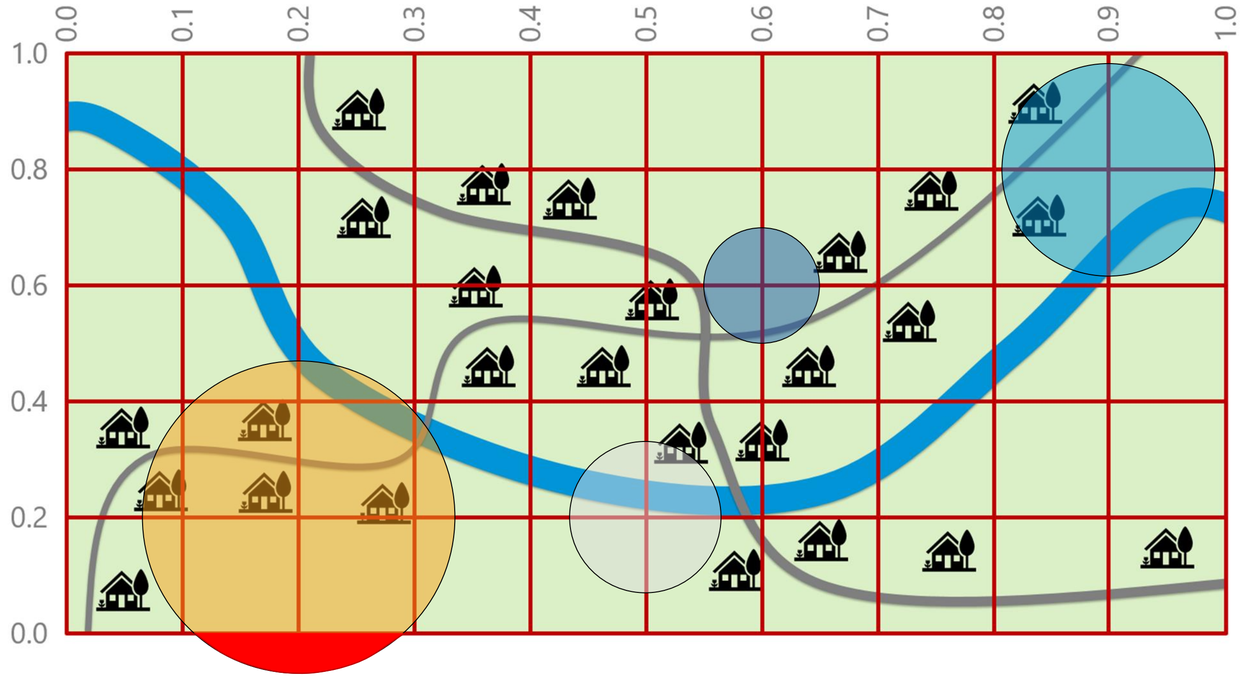
PENALTY = 4*100 = 400

For the zones of influence:

1. Take the difference of each design surface and the extruded site bounds
2. Define the penalty by taking the remaining area of each design surface, summing the results and multiply by 100
3. Join the original design surfaces into a polysurface
4. Test all site cells for intersection with the polysurface
5. Filter out the list of dictionaries by the intersection test
6. For the remaining dictionaries, pull the values for each metric
7. Sum the results and add the penalty

Evaluations & Results – Stadium Location

The site ‘cells’ have associated dictionaries, and the designs are surfaces



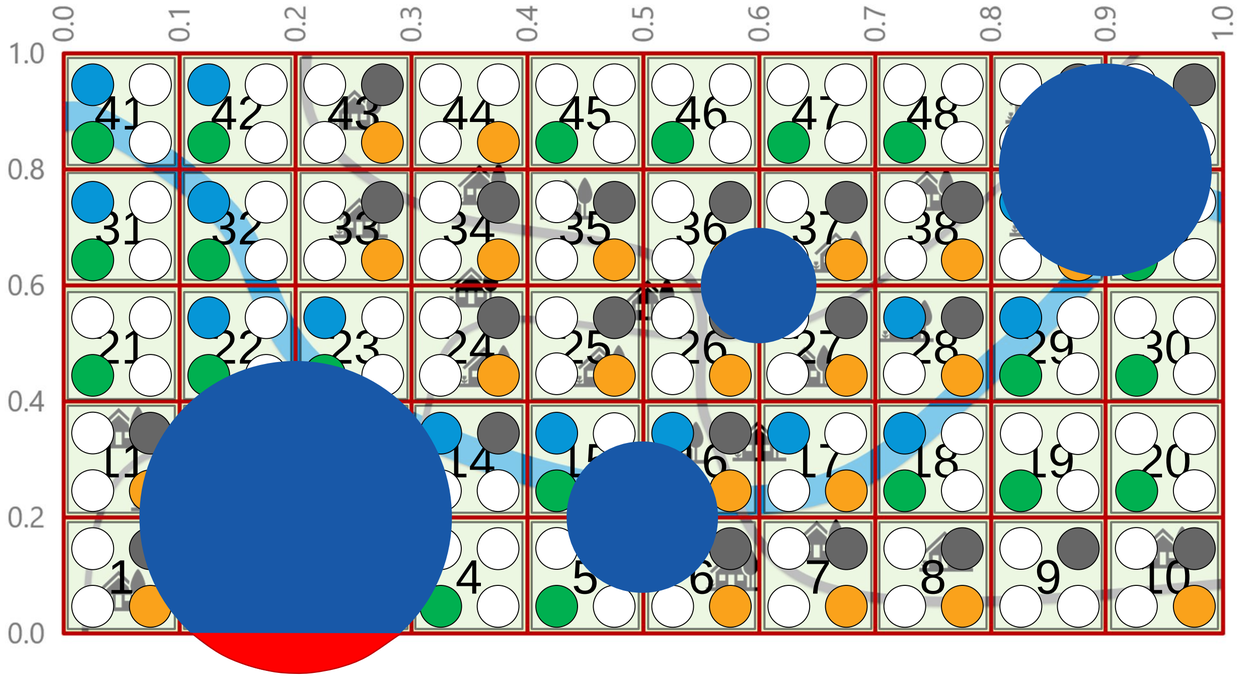
PENALTY = 4*100 = 400

For the zones of influence:

1. Take the difference of each design surface and the extruded site bounds
2. Define the penalty by taking the remaining area of each design surface, summing the results and multiply by 100
3. Join the original design surfaces into a polysurface
4. Test all site cells for intersection with the polysurface
5. Filter out the list of dictionaries by the intersection test
6. For the remaining dictionaries, pull the values for each metric
7. Sum the results and add the penalty

Evaluations & Results – Stadium Location

The site ‘cells’ have associated dictionaries, and the designs are surfaces



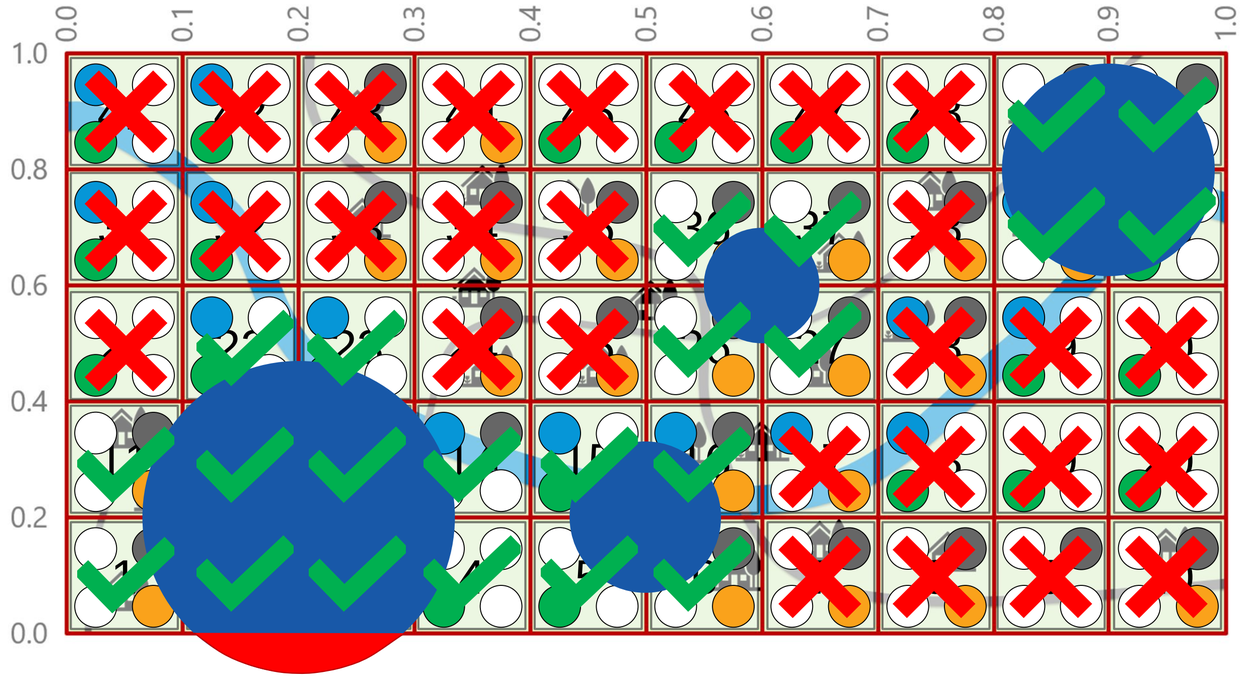
PENALTY = 4*100 = 400

For the zones of influence:

1. Take the difference of each design surface and the extruded site bounds
2. Define the penalty by taking the remaining area of each design surface, summing the results and multiply by 100
3. Join the original design surfaces into a polysurface
4. Test all site cells for intersection with the polysurface
5. Filter out the list of dictionaries by the intersection test
6. For the remaining dictionaries, pull the values for each metric
7. Sum the results and add the penalty

Evaluations & Results – Stadium Location

The site ‘cells’ have associated dictionaries, and the designs are surfaces



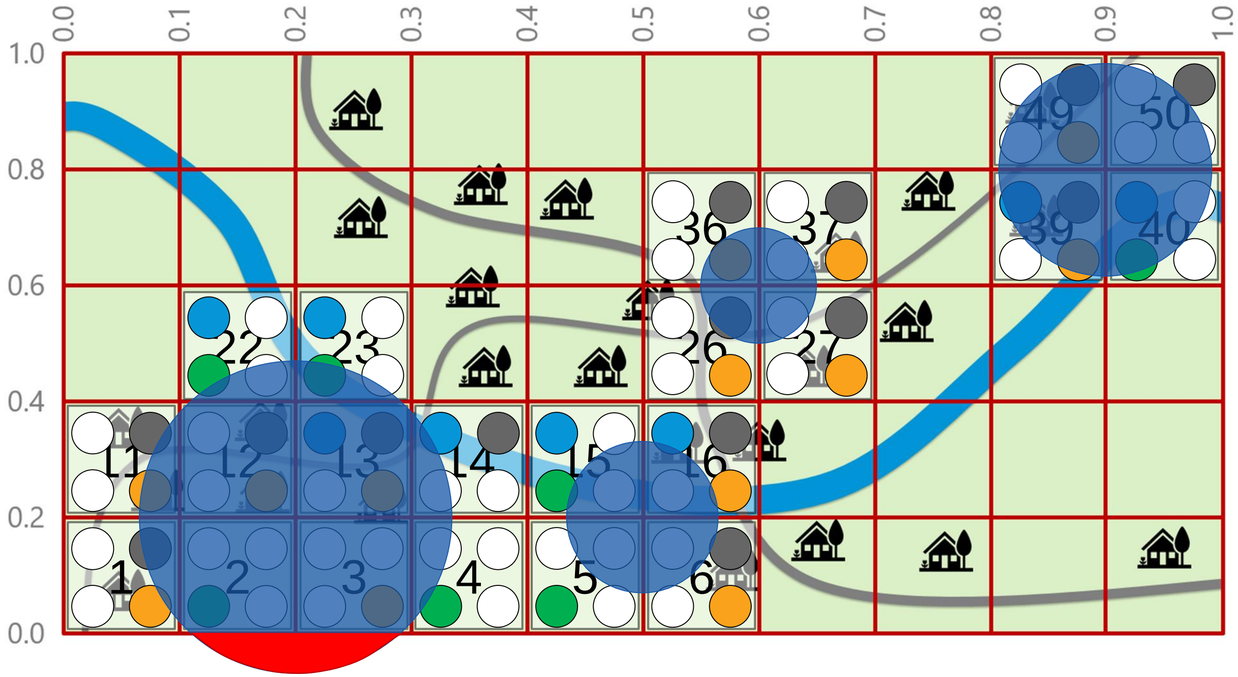
PENALTY = 4*100 = 400

For the zones of influence:

1. Take the difference of each design surface and the extruded site bounds
2. Define the penalty by taking the remaining area of each design surface, summing the results and multiply by 100
3. Join the original design surfaces into a polysurface
4. Test all site cells for intersection with the polysurface
5. Filter out the list of dictionaries by the intersection test
6. For the remaining dictionaries, pull the values for each metric
7. Sum the results and add the penalty

Evaluations & Results – Stadium Location

The site ‘cells’ have associated dictionaries, and the designs are surfaces



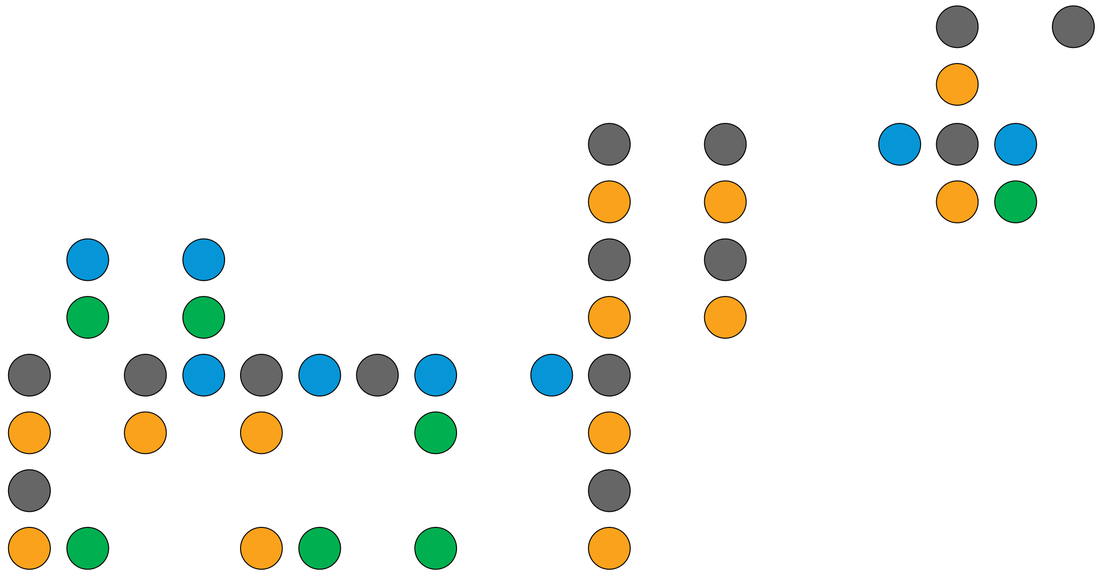
PENALTY = 4*100 = 400

For the zones of influence:

1. Take the difference of each design surface and the extruded site bounds
2. Define the penalty by taking the remaining area of each design surface, summing the results and multiply by 100
3. Join the original design surfaces into a polysurface
4. Test all site cells for intersection with the polysurface
5. Filter out the list of dictionaries by the intersection test
6. For the remaining dictionaries, pull the values for each metric
7. Sum the results and add the penalty

Evaluations & Results – Stadium Location

The site ‘cells’ have associated dictionaries, and the designs are surfaces



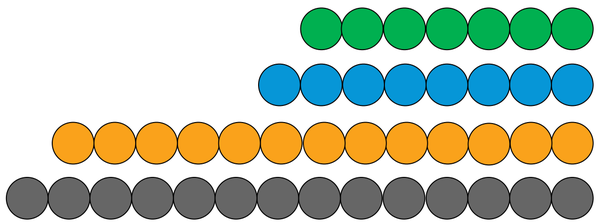
PENALTY = 4*100 = 400

For the zones of influence:

1. Take the difference of each design surface and the extruded site bounds
2. Define the penalty by taking the remaining area of each design surface, summing the results and multiply by 100
3. Join the original design surfaces into a polysurface
4. Test all site cells for intersection with the polysurface
5. Filter out the list of dictionaries by the intersection test
6. For the remaining dictionaries, pull the values for each metric
7. Sum the results and add the penalty

Evaluations & Results – Stadium Location

The site ‘cells’ have associated dictionaries, and the designs are surfaces

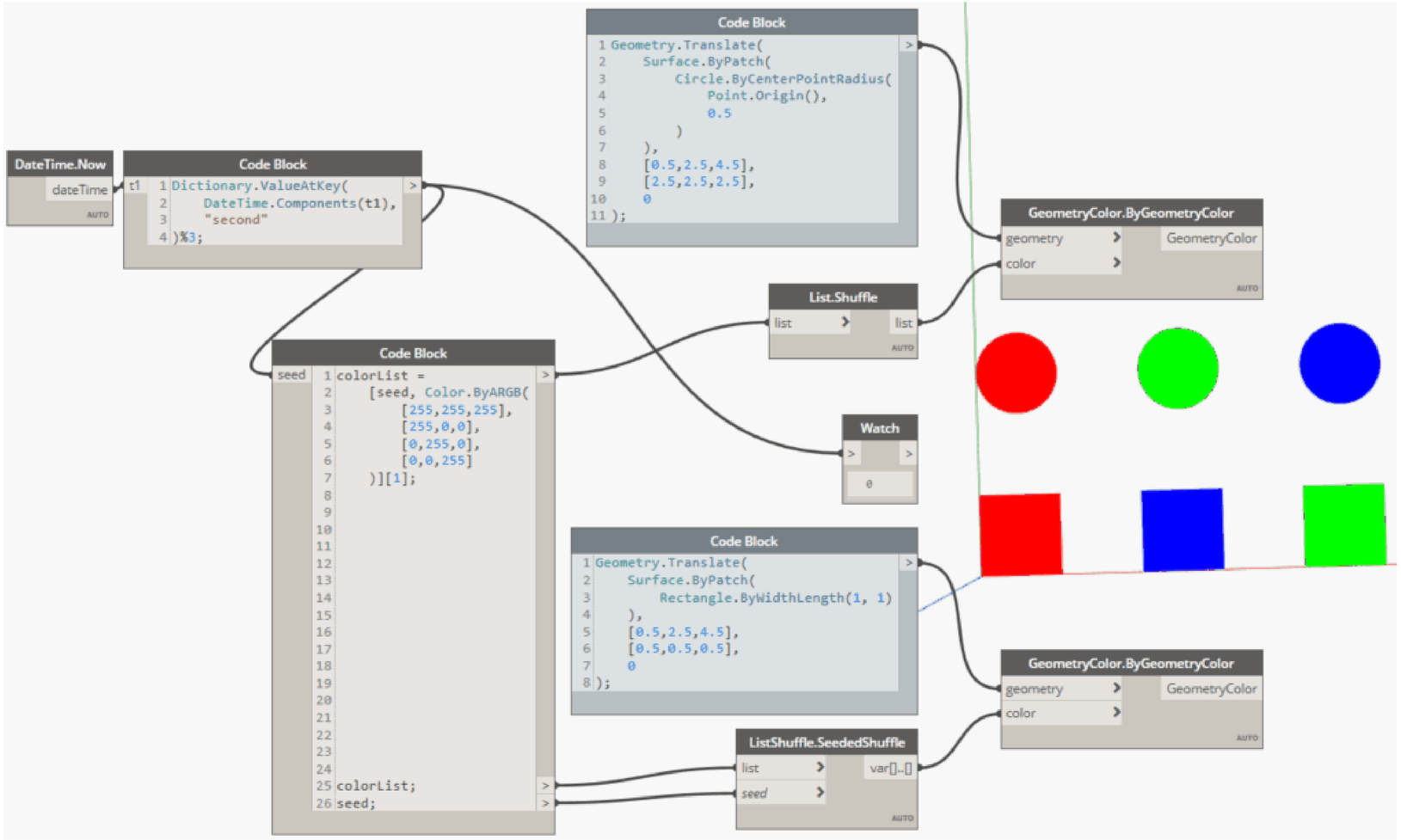


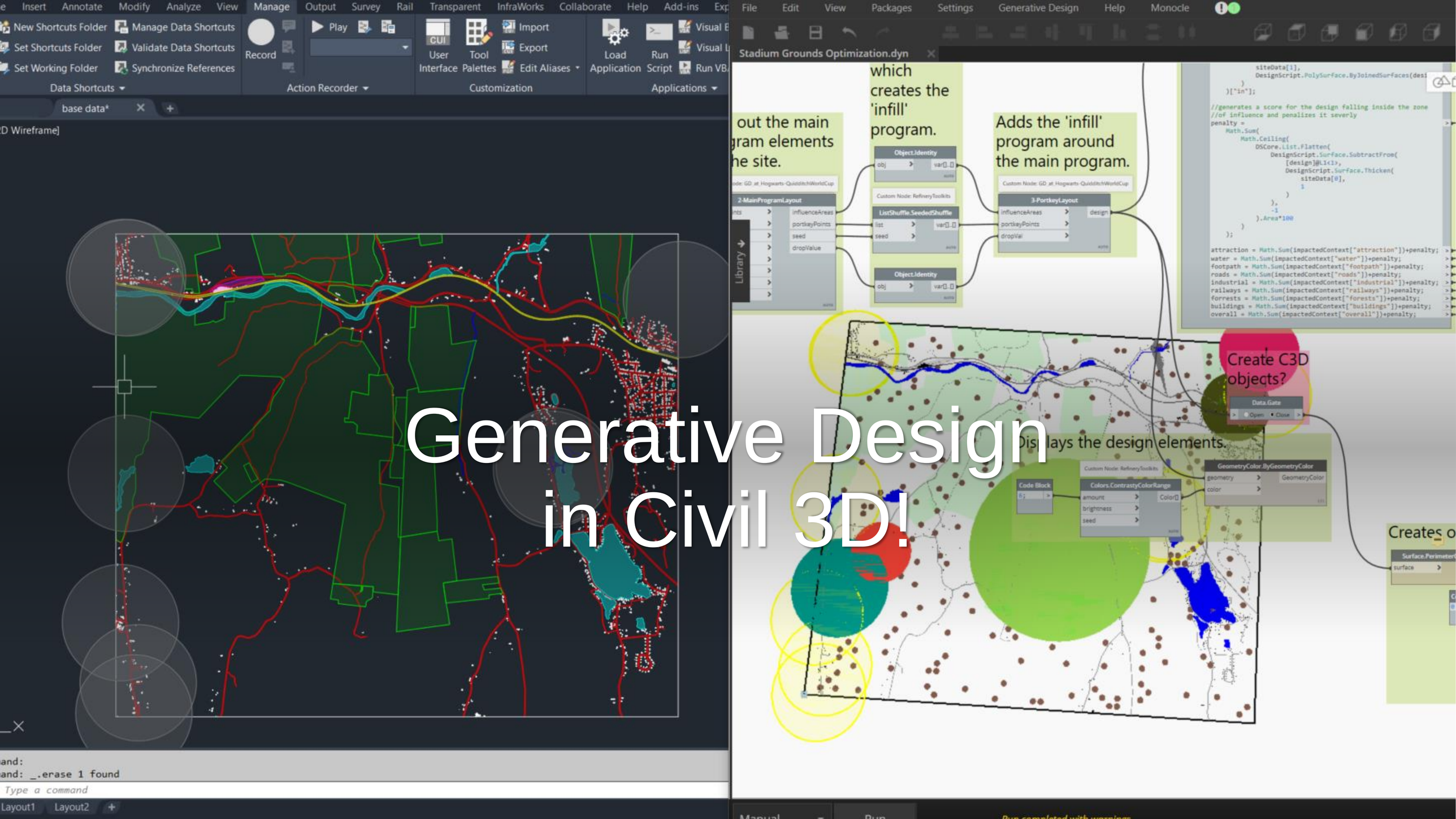
FOREST:	7
WATER:	8
BUILDING:	13
ROAD:	14
PENALTY = 4*100 =	400
FOREST SCORE:	**407**
WATER SCORE:	**408**
BUILDING SCORE:	**413**
ROAD SCORE:	**414**

For the zones of influence:

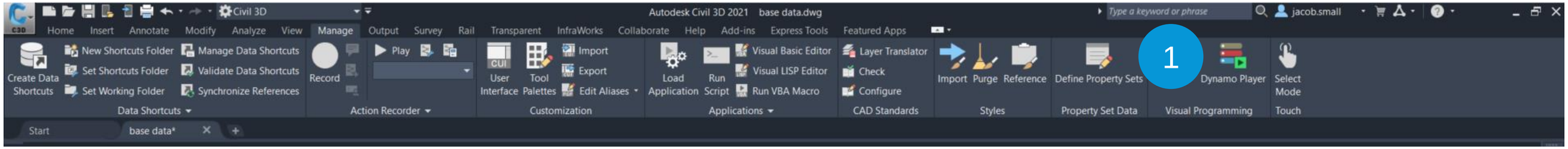
1. Take the difference of each design surface and the extruded site bounds
2. Define the penalty by taking the remaining area of each design surface, summing the results and multiply by 100
3. Join the original design surfaces into a polysurface
4. Test all site cells for intersection with the polysurface
5. Filter out the list of dictionaries by the intersection test
6. For the remaining dictionaries, pull the values for each metric
7. Sum the results and add the penalty

Pseudo Random Shuffle

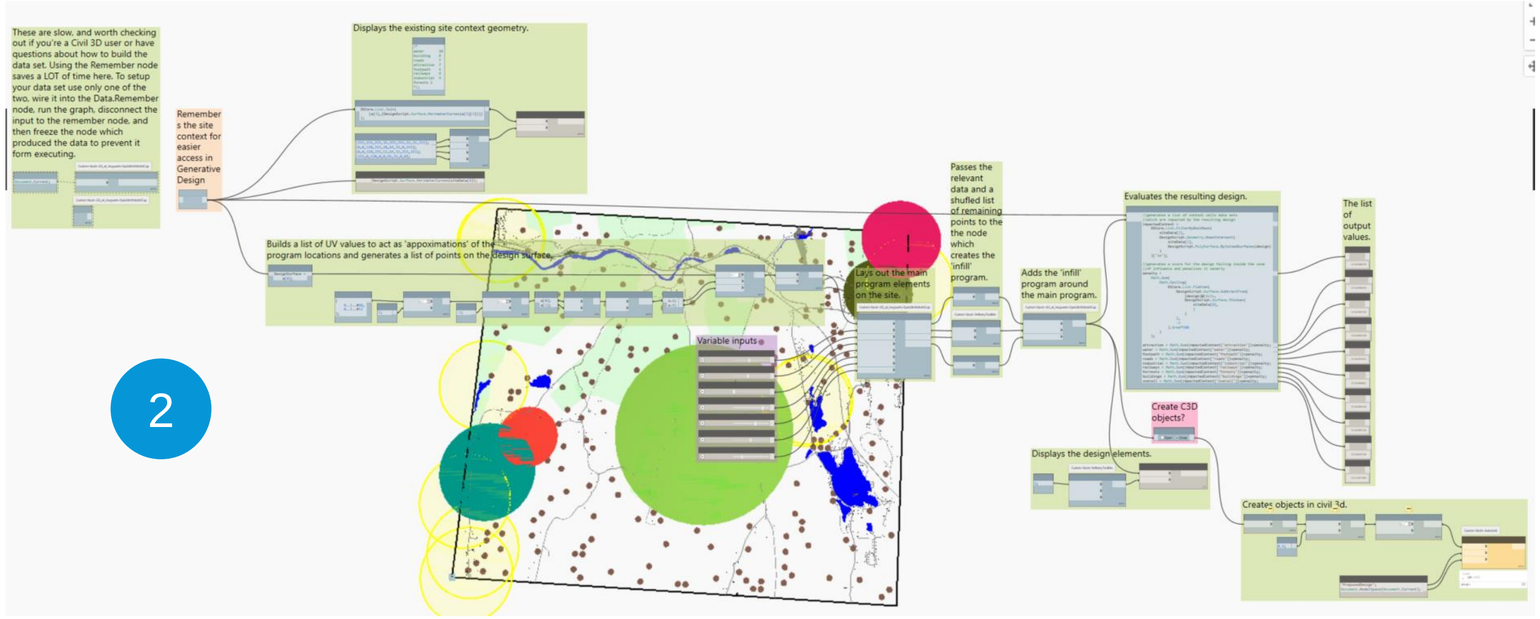



Insert Annotate Modify Analyze View Manage Output Survey Rail Transparent InfraWorks Collaborate Help Add-ins
New Shortcuts Folder
Set Shortcuts Folder
Set Working Folder
Manage Data Shortcuts
Validate Data Shortcuts
Synchronize References
Data Shortcuts
Record
Play
Action Recorder
User Interface
Tool Palettes
Import
Export
Edit Aliases
Customization
Load Application
Run Script
Applications
base data*
D Wireframe]
Generative Design
in Civil 3D!
File Edit View Packages Settings Generative Design Help Monocle
Stadium Grounds Optimization.dyn
which creates the 'infill' program.
Adds the 'infill' program around the main program.
Create C3D objects?
Displays the design elements.
Creates o
Type a command
Layout1
Layout2

Getting Generative Design in Civil 3D

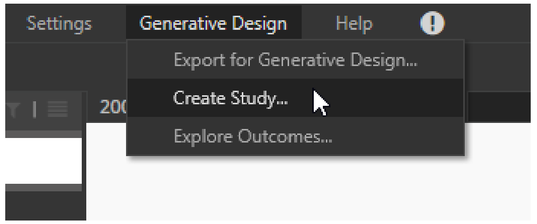


Launch Dynamo



Generative Design Script

3

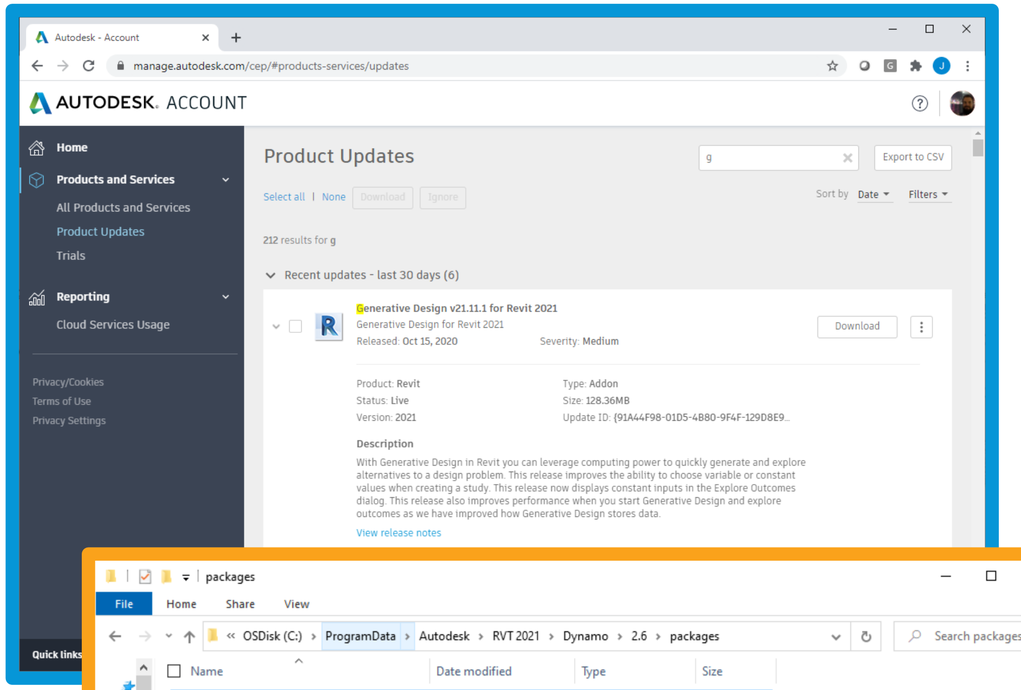


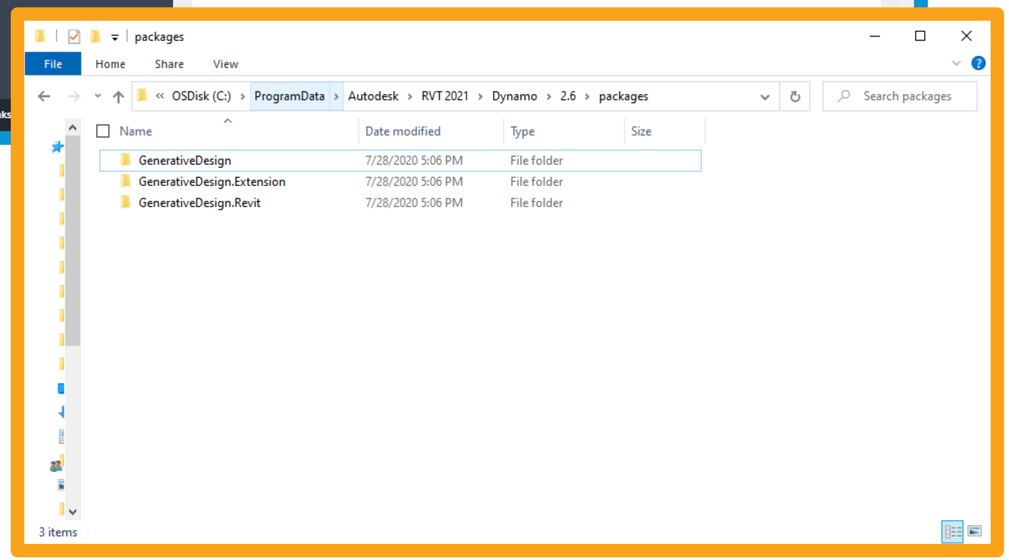
“Export" to GD and then create study.

“How do I get the Generative Design Package though?”

The 4 easyest steps in this entire session!

1. Install Revit 2021 and all updates.
2. Install any Generative Design updates from **Manage.Autodesk.com**.
3. Copy the contents of the Generative Design package from this directory to your Civil 3D package directory:
   C:\ProgramData\Autodesk\RVT 2021\Dynamo\2.6\packages
4. Make cool things and have fun doing it.



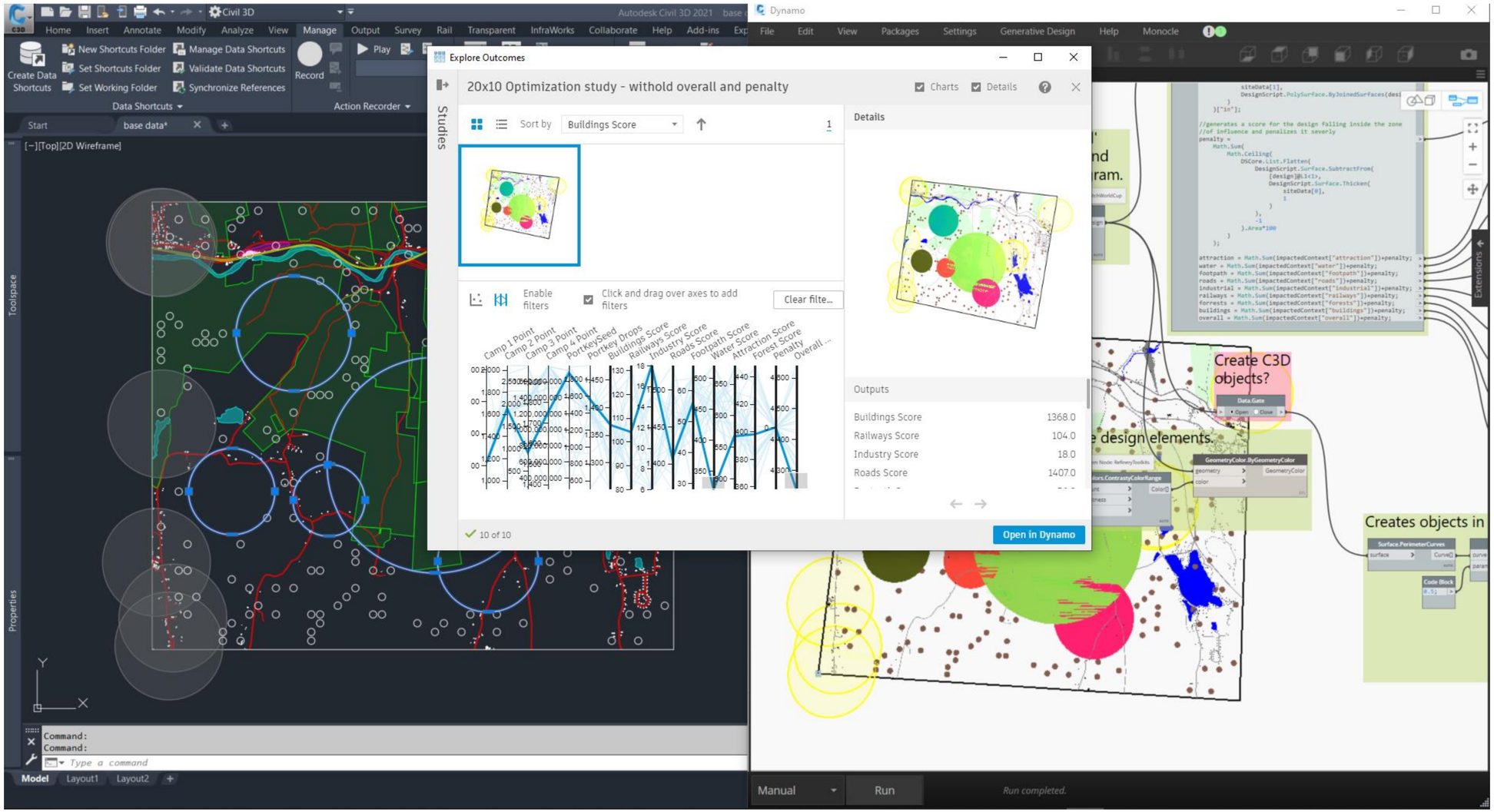


Data Set Before

- **Site Plan** of city showing buildings, roads, water features, public forests, industrial areas, footpaths, railways and local attractions.
- **Note:** This data set was entirely built with the Elk Package from OSM data. Special thanks to **Timothy Logan** for the excellent contribution to the Dynamo community. More info here: https://github.com/logant/Elk

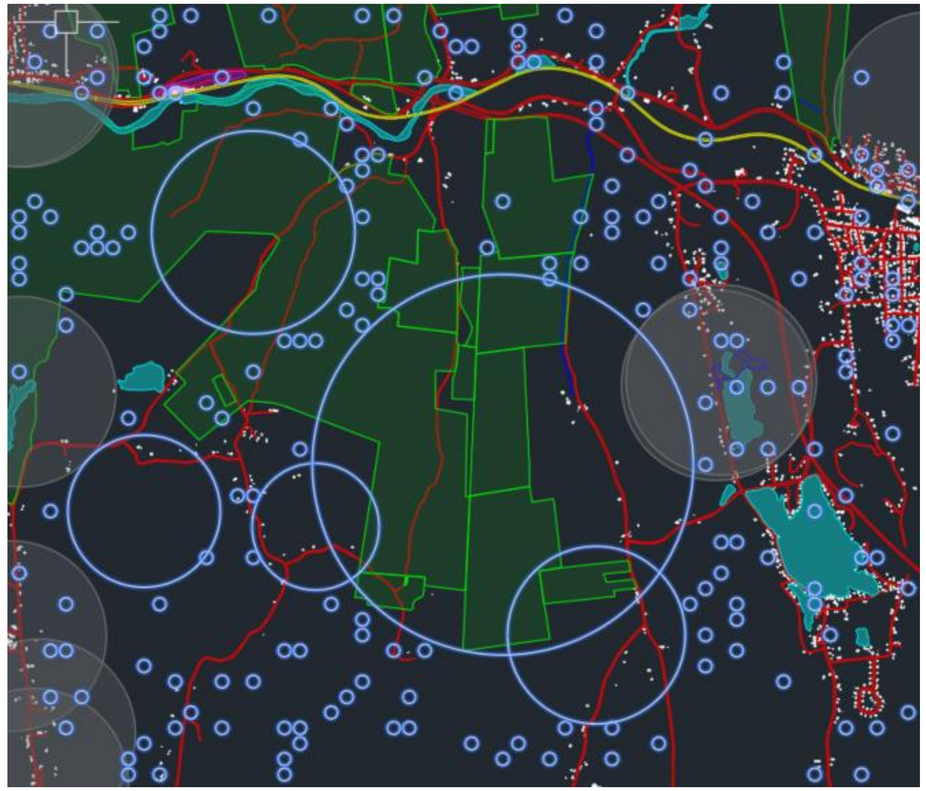
Let’s build the graph!

Generation and Results

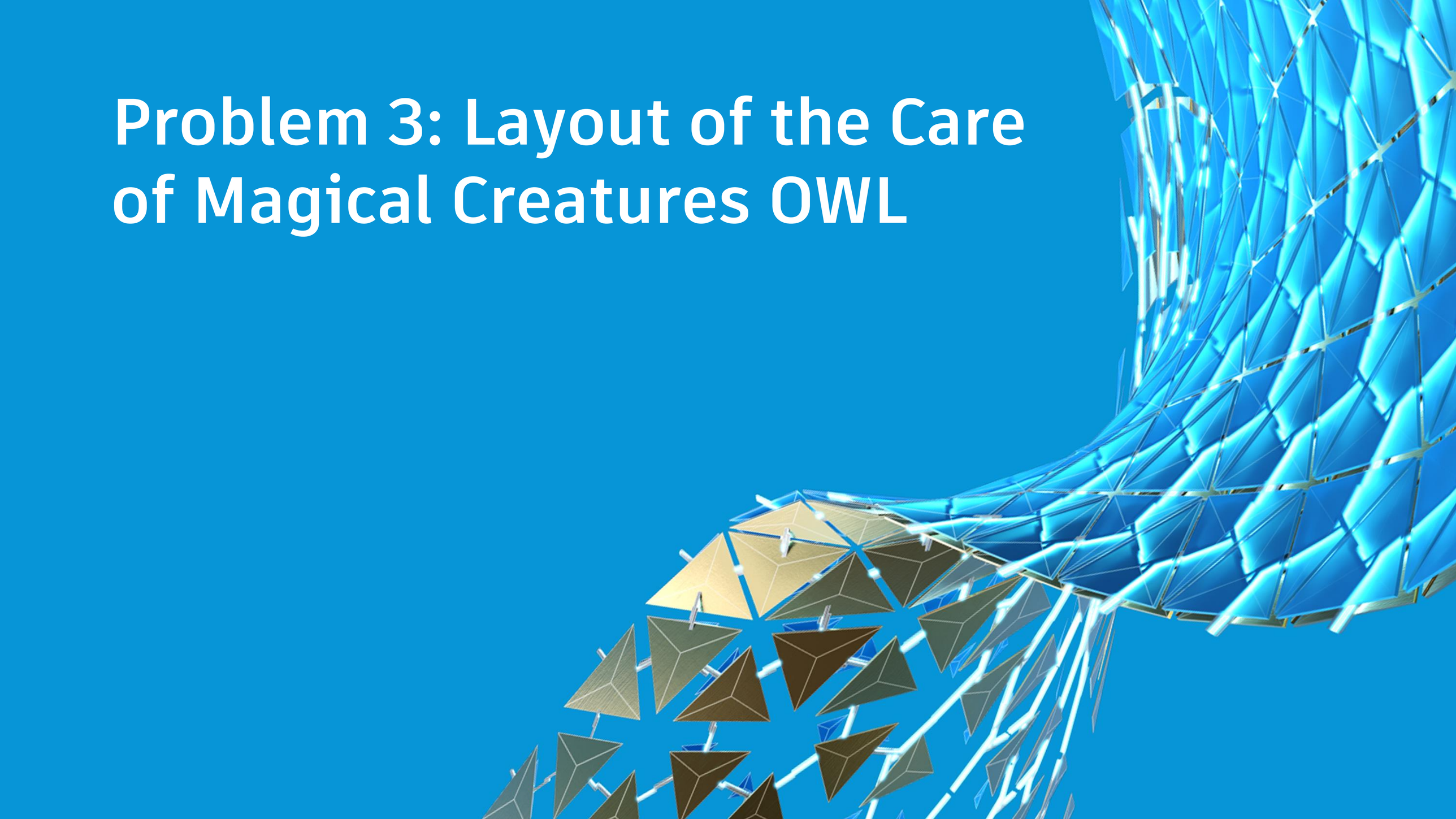


Notes and Review

- Sometimes multiple small studies will get the best answer faster than one big study: set and confirm what you learn works!
- The size of problems can be reduced to a reasonable scale drastically by abstracting and simplifying the problem.
- Finding the ‘exact’ location isn’t always a must – learning what we can from generalizations will allow you to focus in on a particular area of the design.

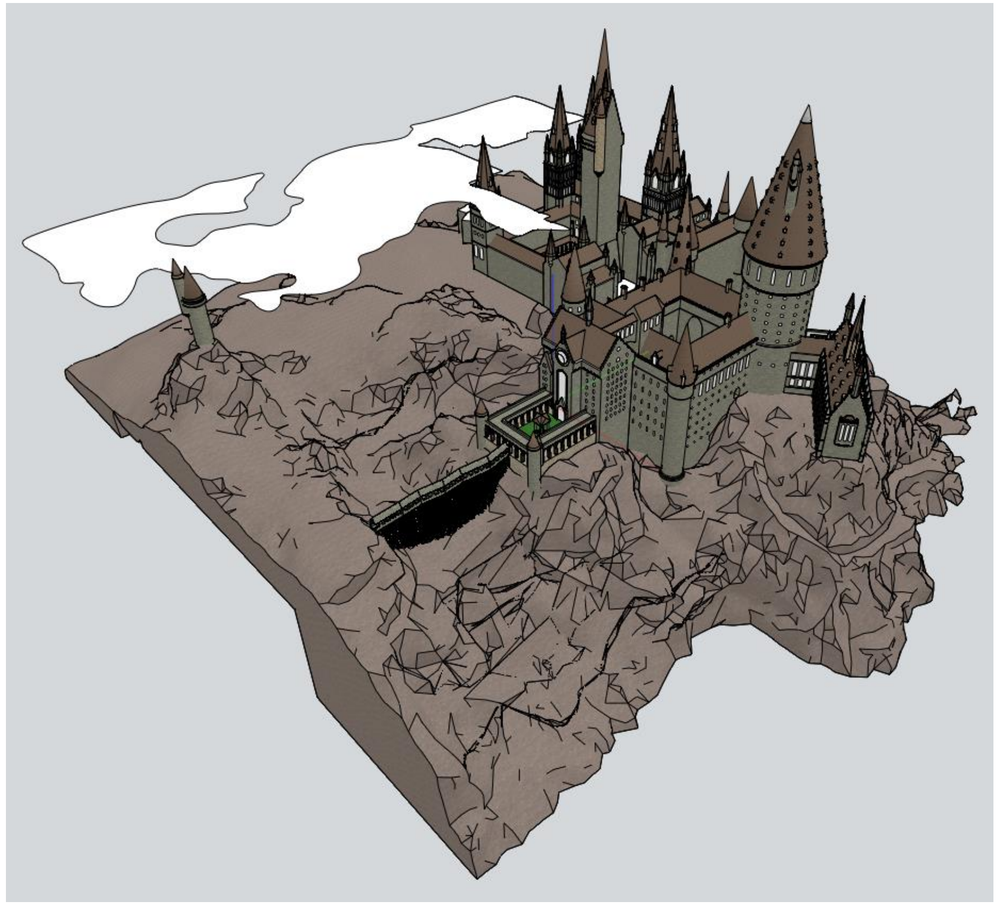


Problem 3: Layout of the Care of Magical Creatures OWL

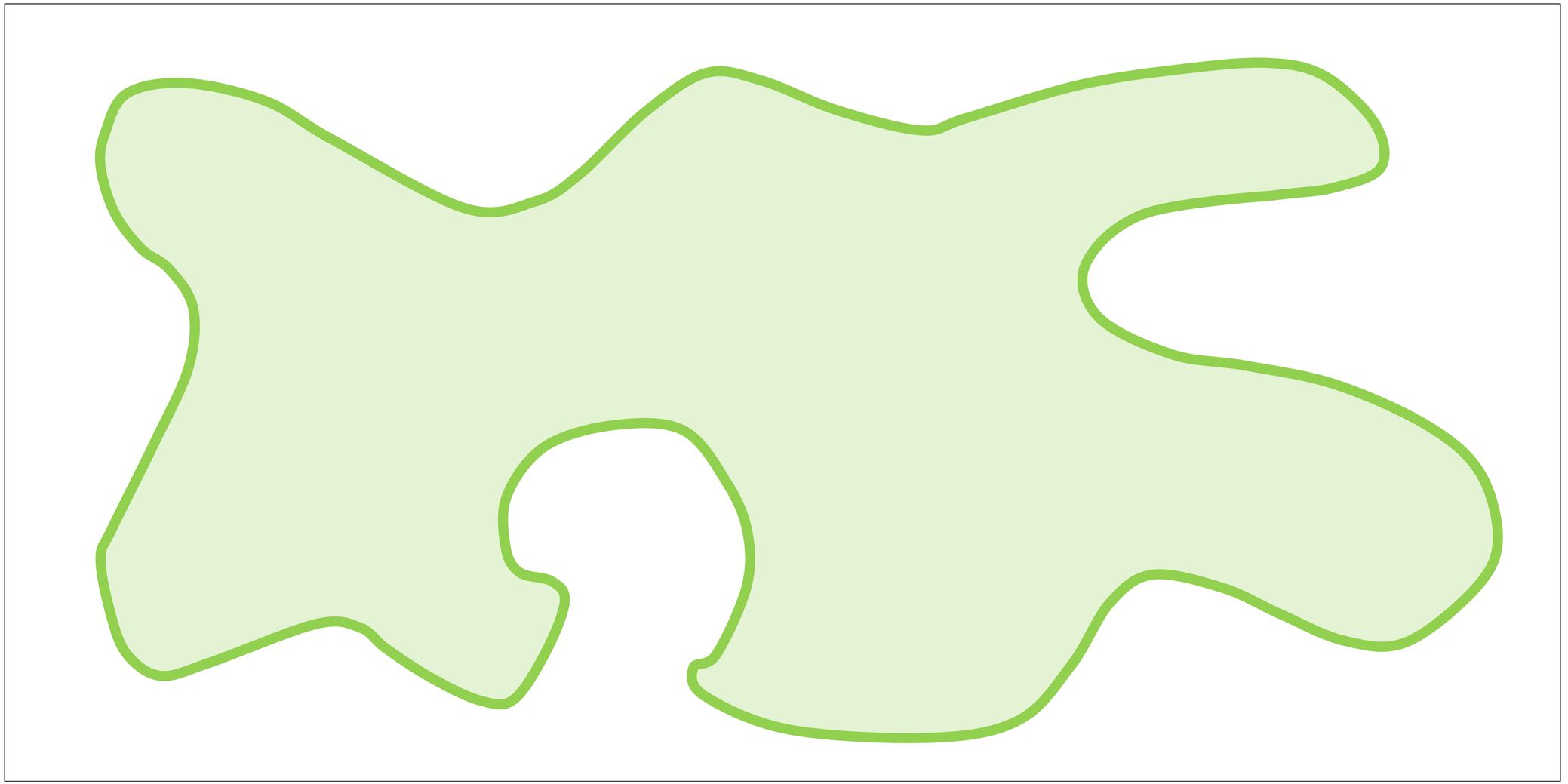


Flexible Option Studies

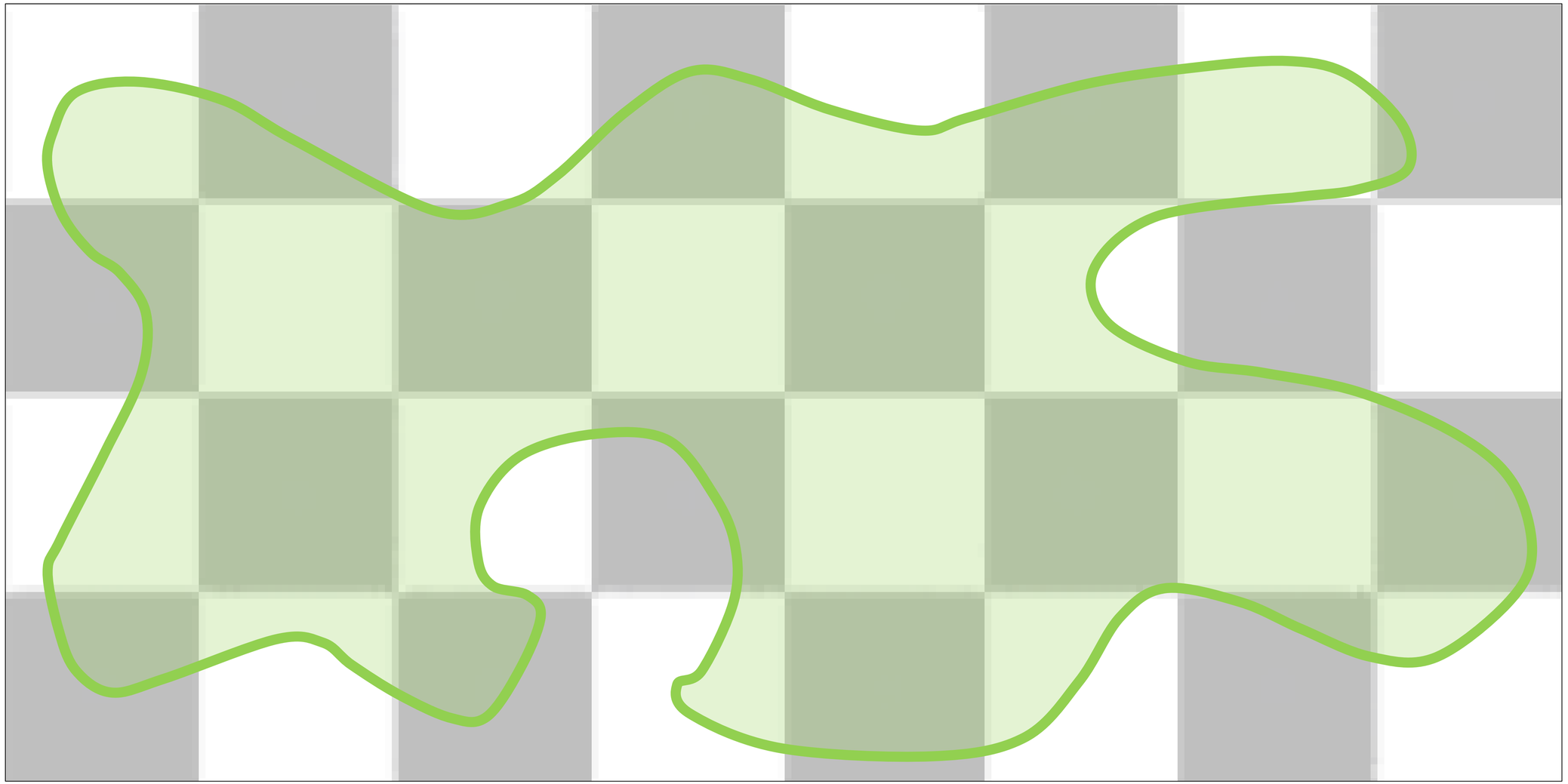
- Sometimes the design space is wide open and there are no limits or restrictions which quickly come to light. In these cases you have to design the system to be flexible and work with what you know
- In this session, we will lay out the care of magical creatures OWL by using the known qualities of magical creatures to size and locate a pen for each.
- The outcomes will be evaluated by each creature’s relative and contextual location.



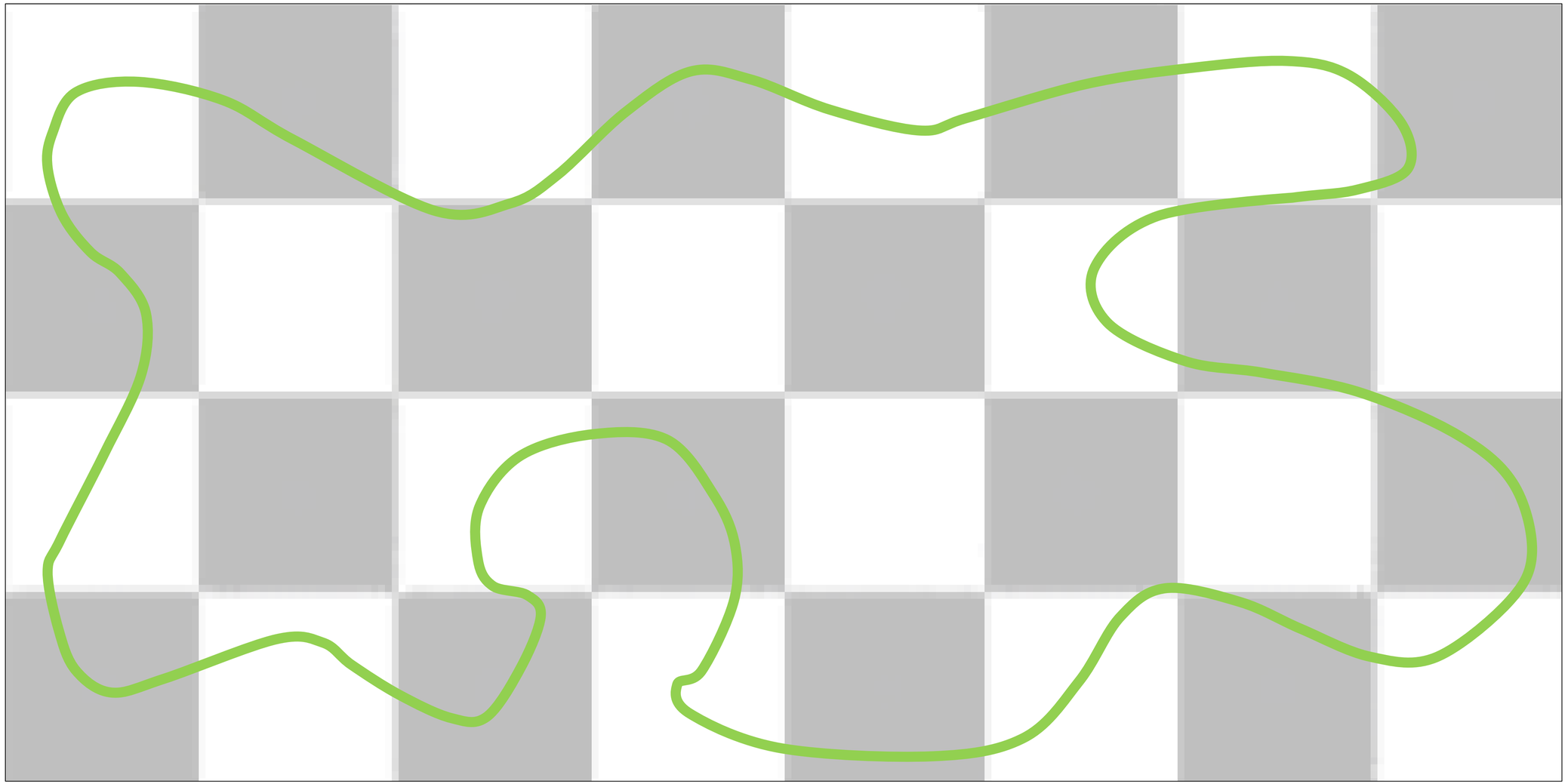
Generation Systems – Define Shape Values



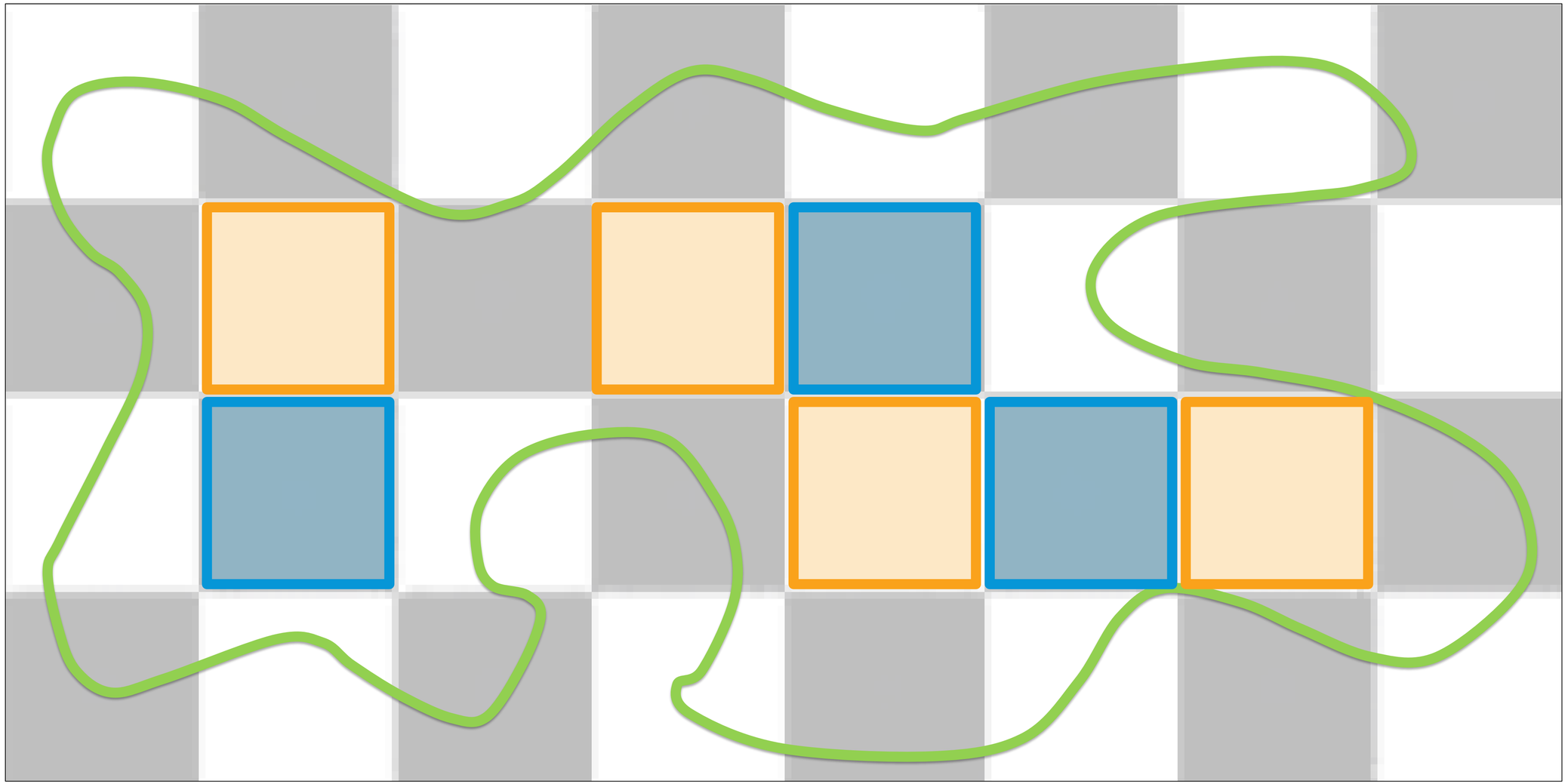
Generation Systems - Assign Values



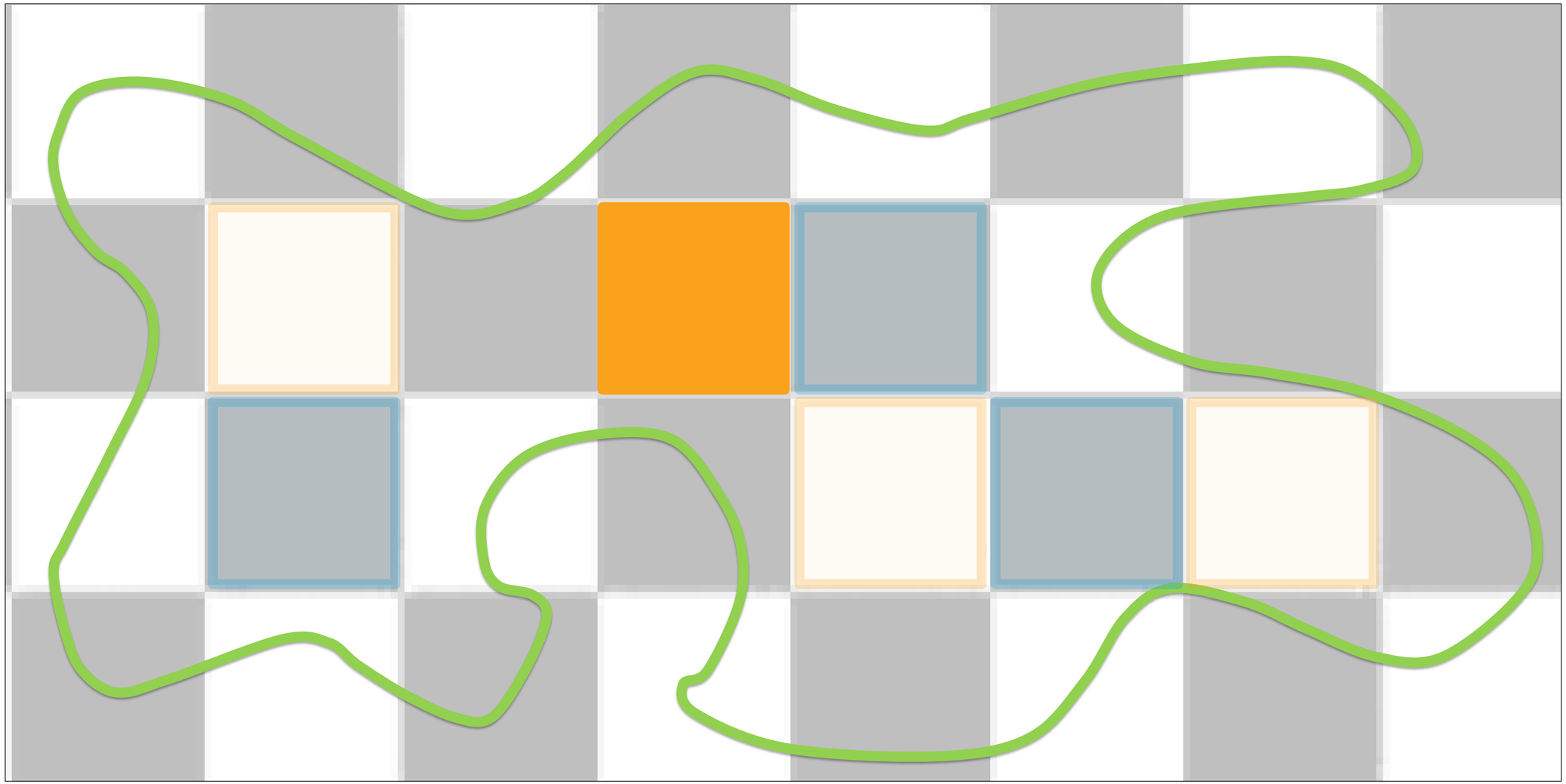
Generation Systems - Assign Values



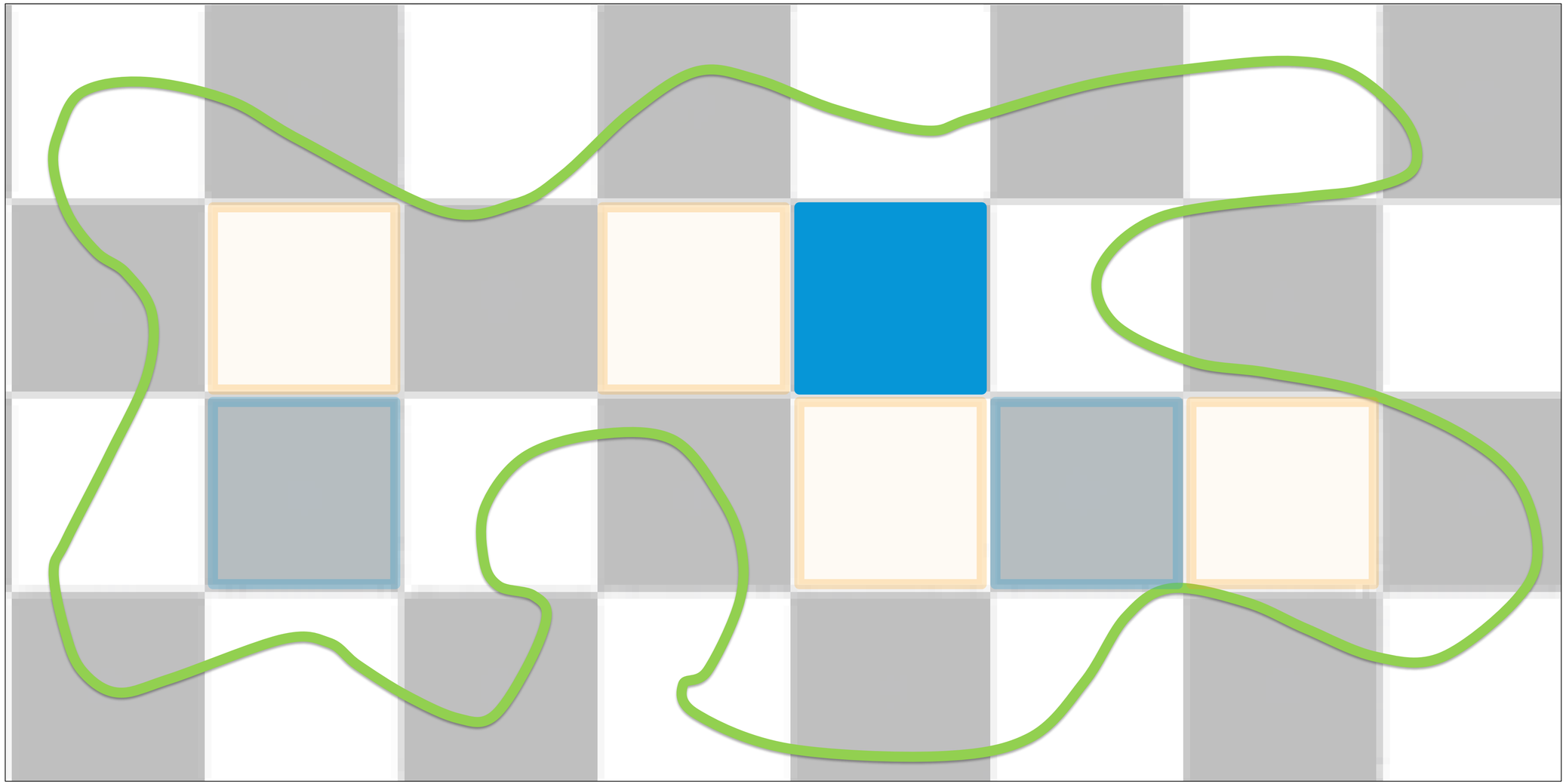
Generation Systems - Assign Values



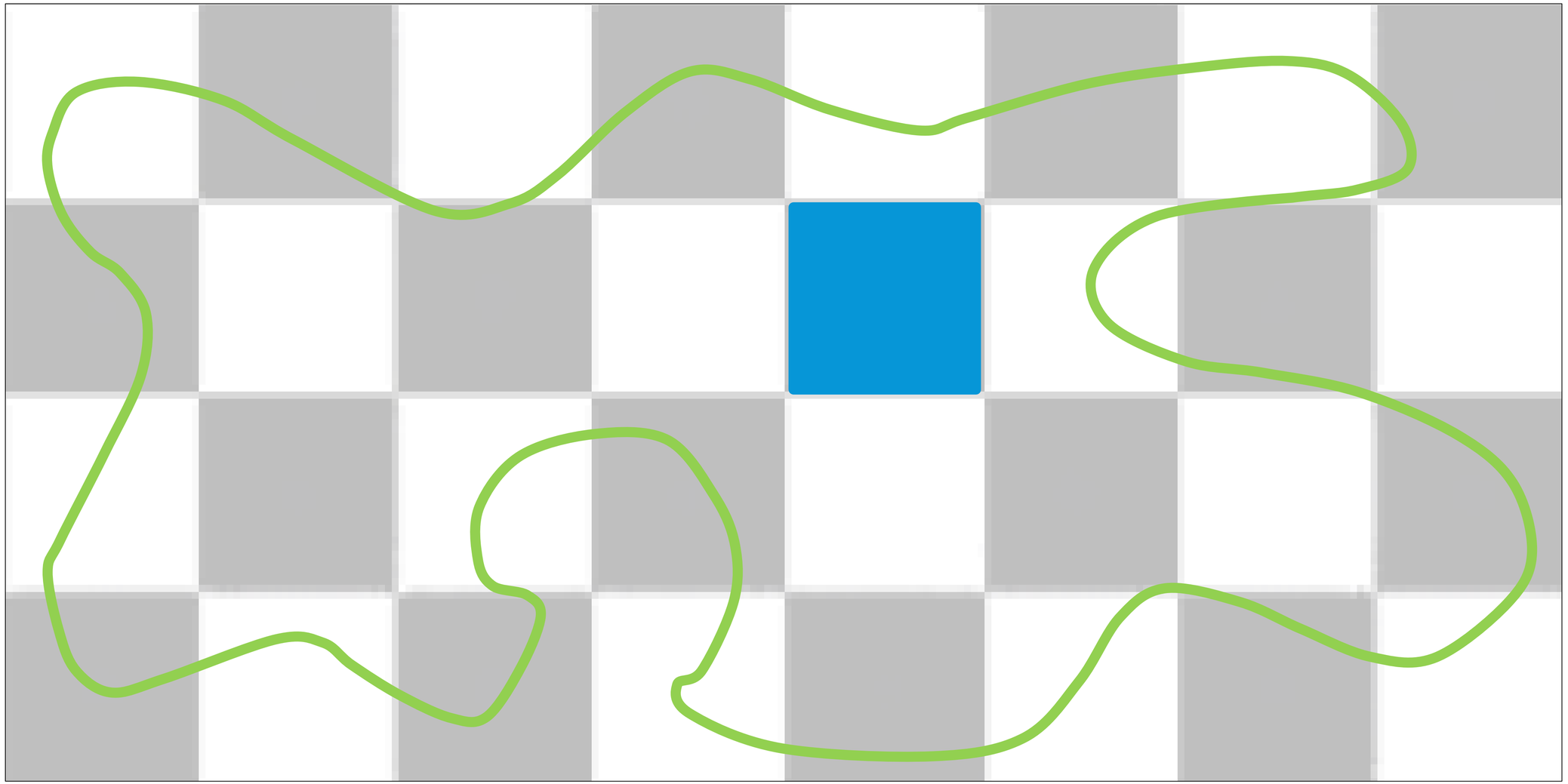
Generation Systems - Assign Values



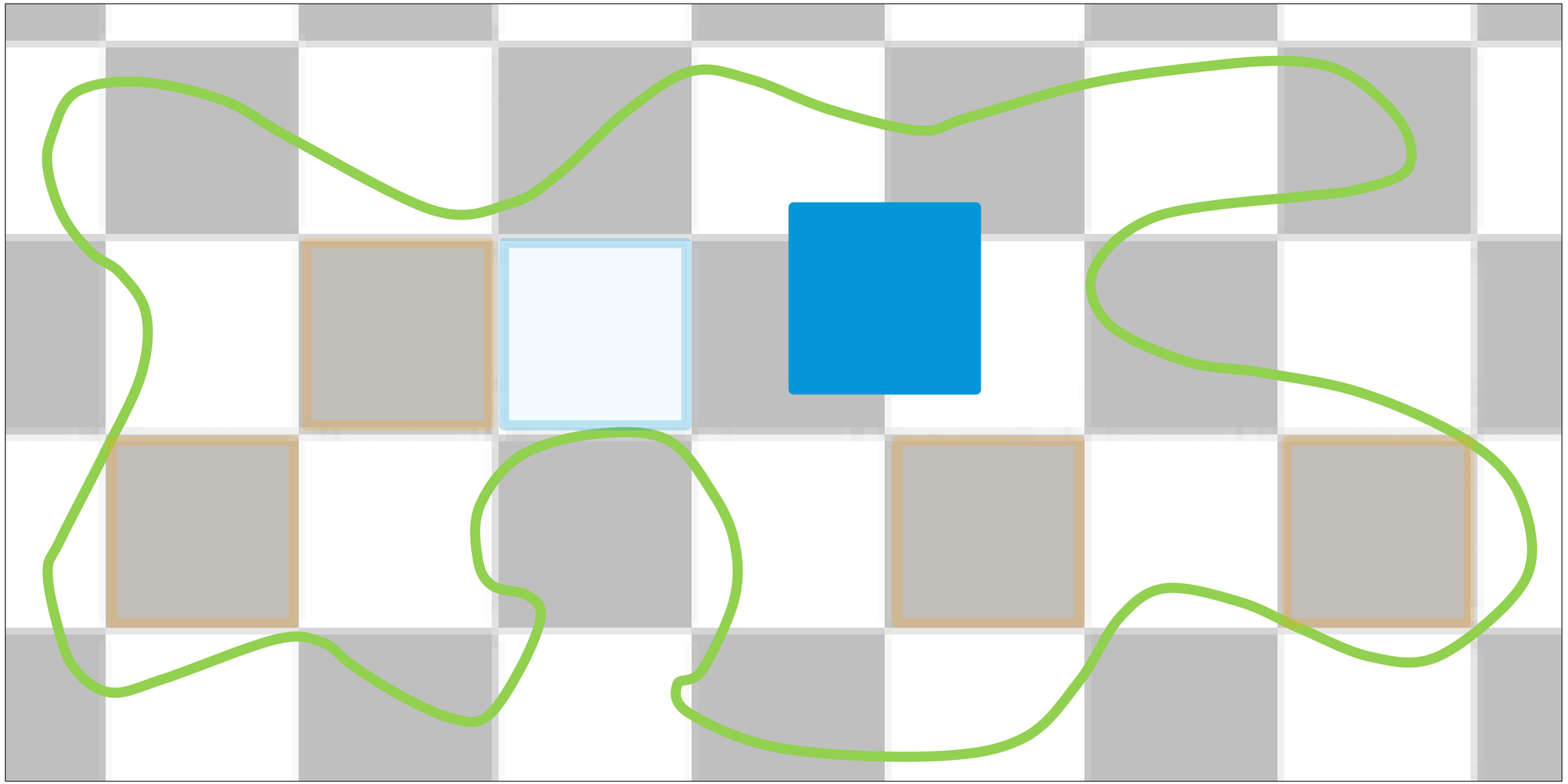
Generation Systems - Assign Values



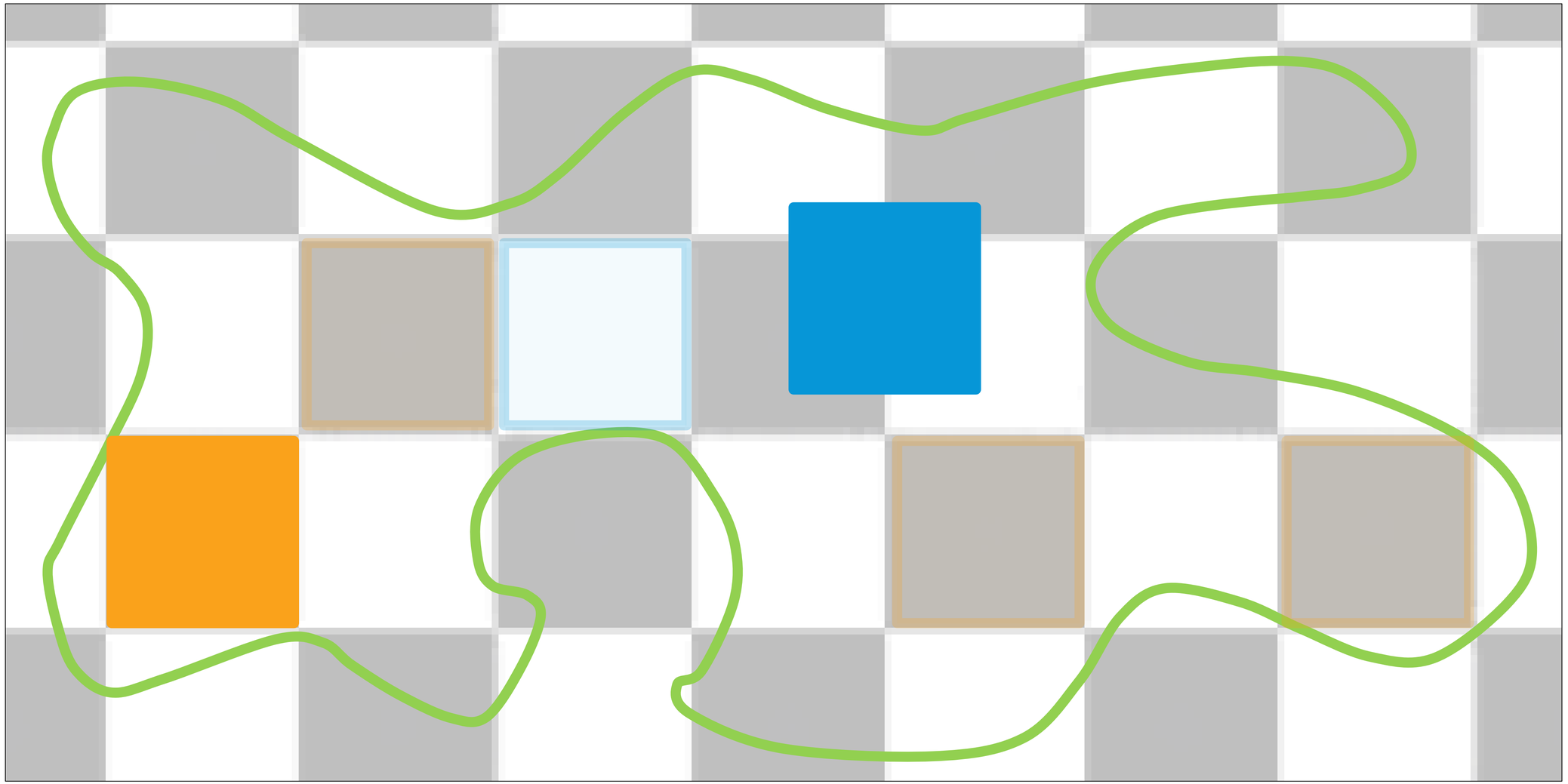
Generation Systems - Assign Values



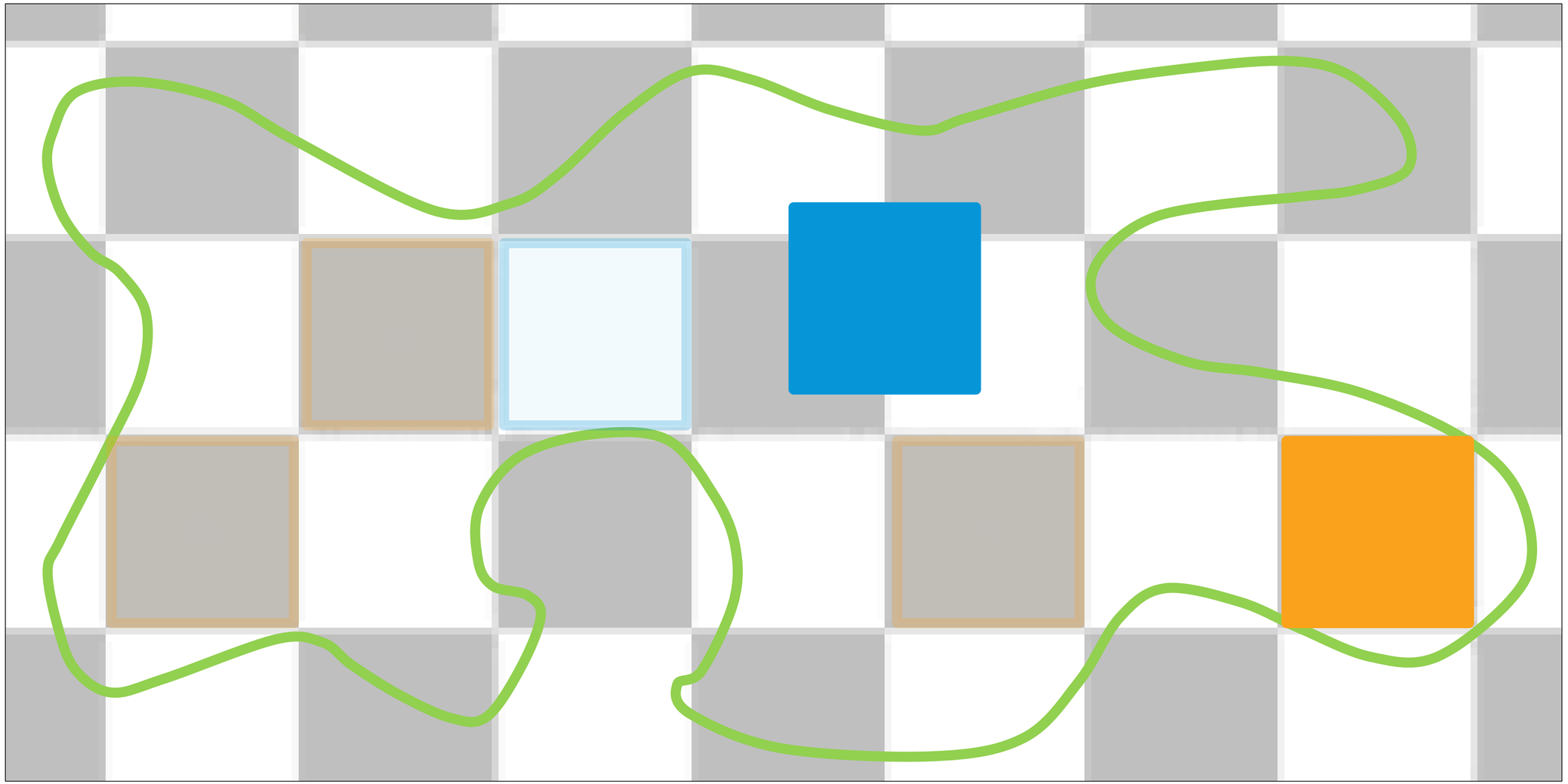
Generation Systems - Assign Values



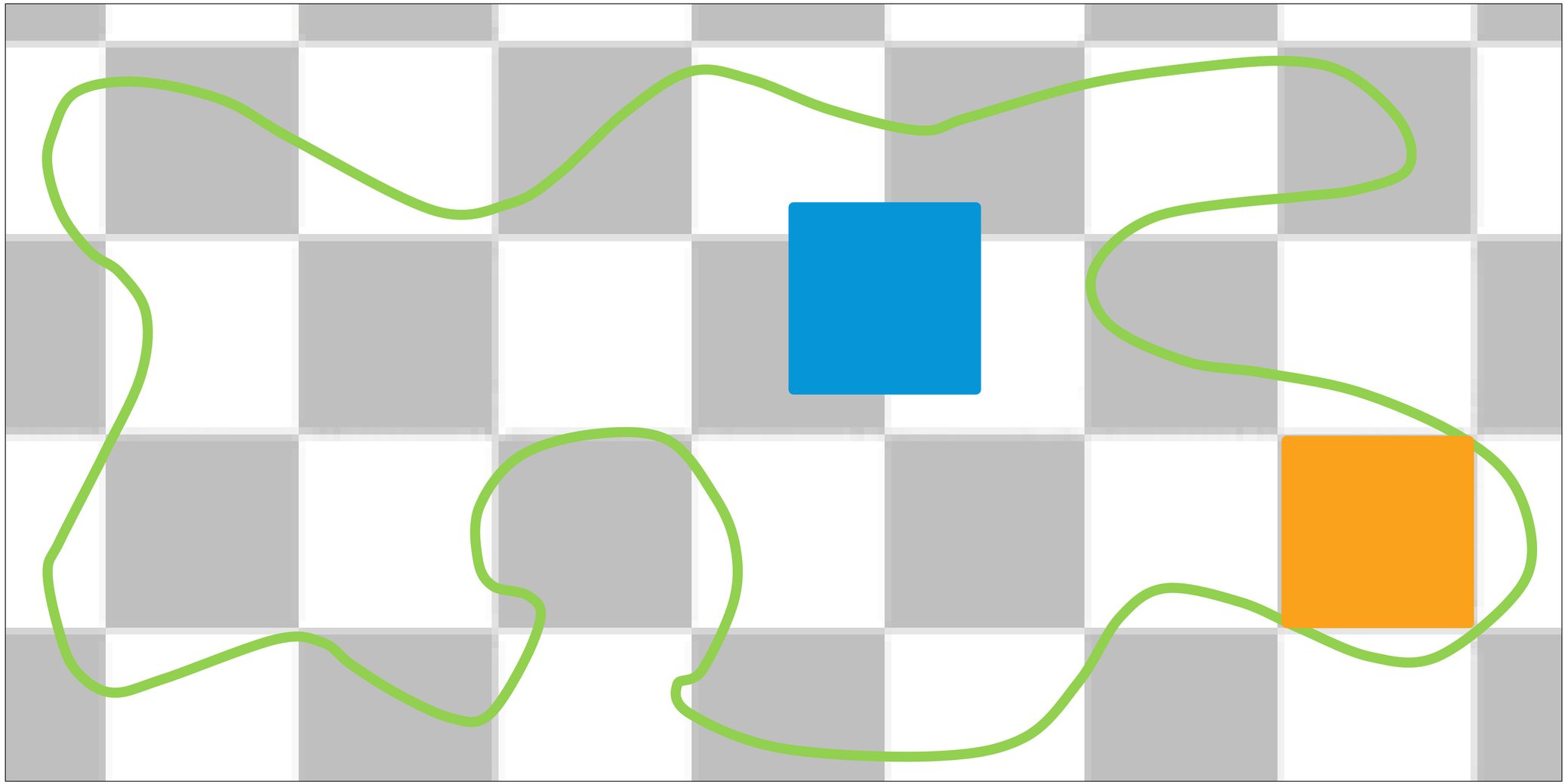
Generation Systems - Assign Values



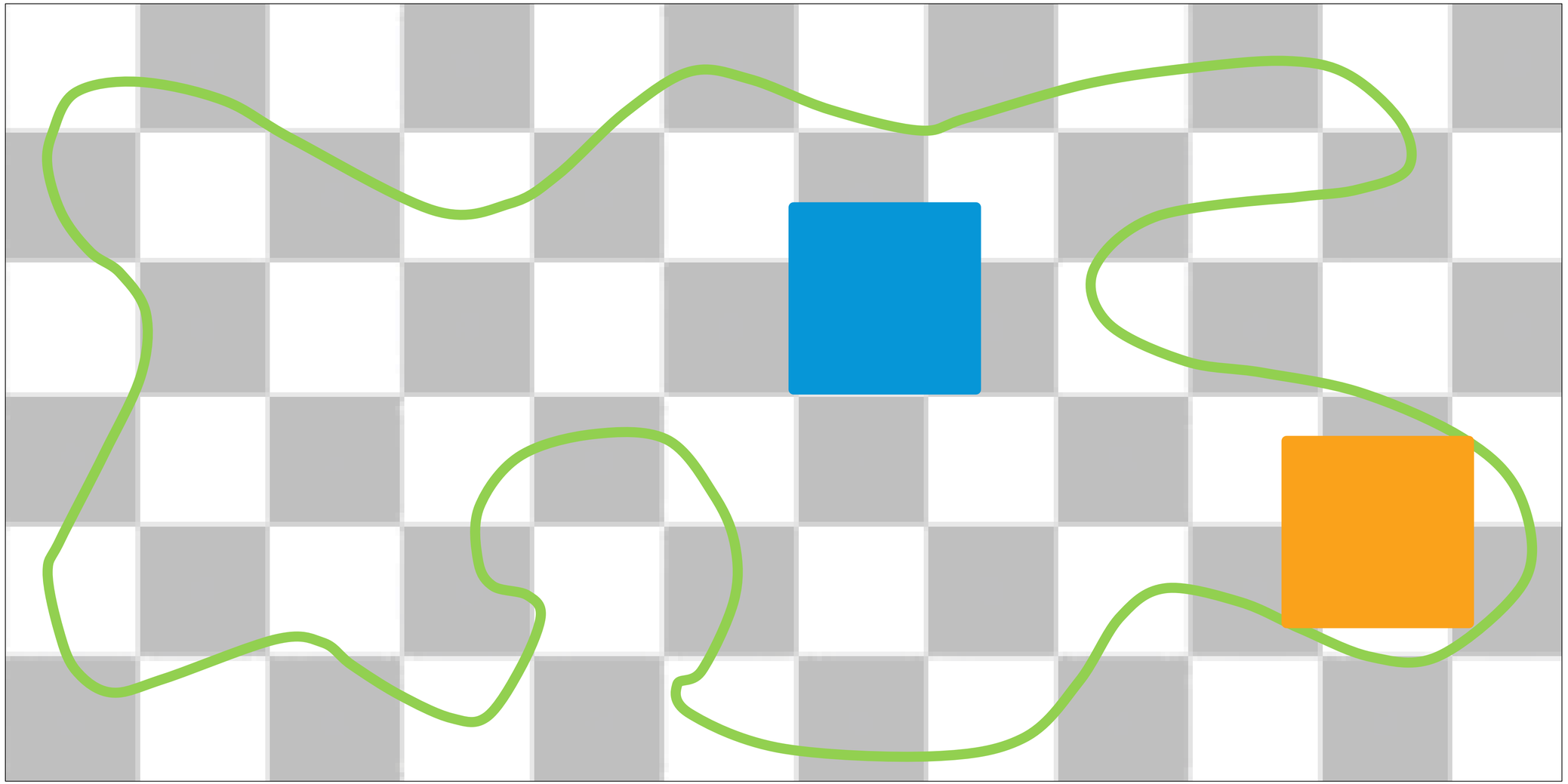
Generation Systems - Assign Values



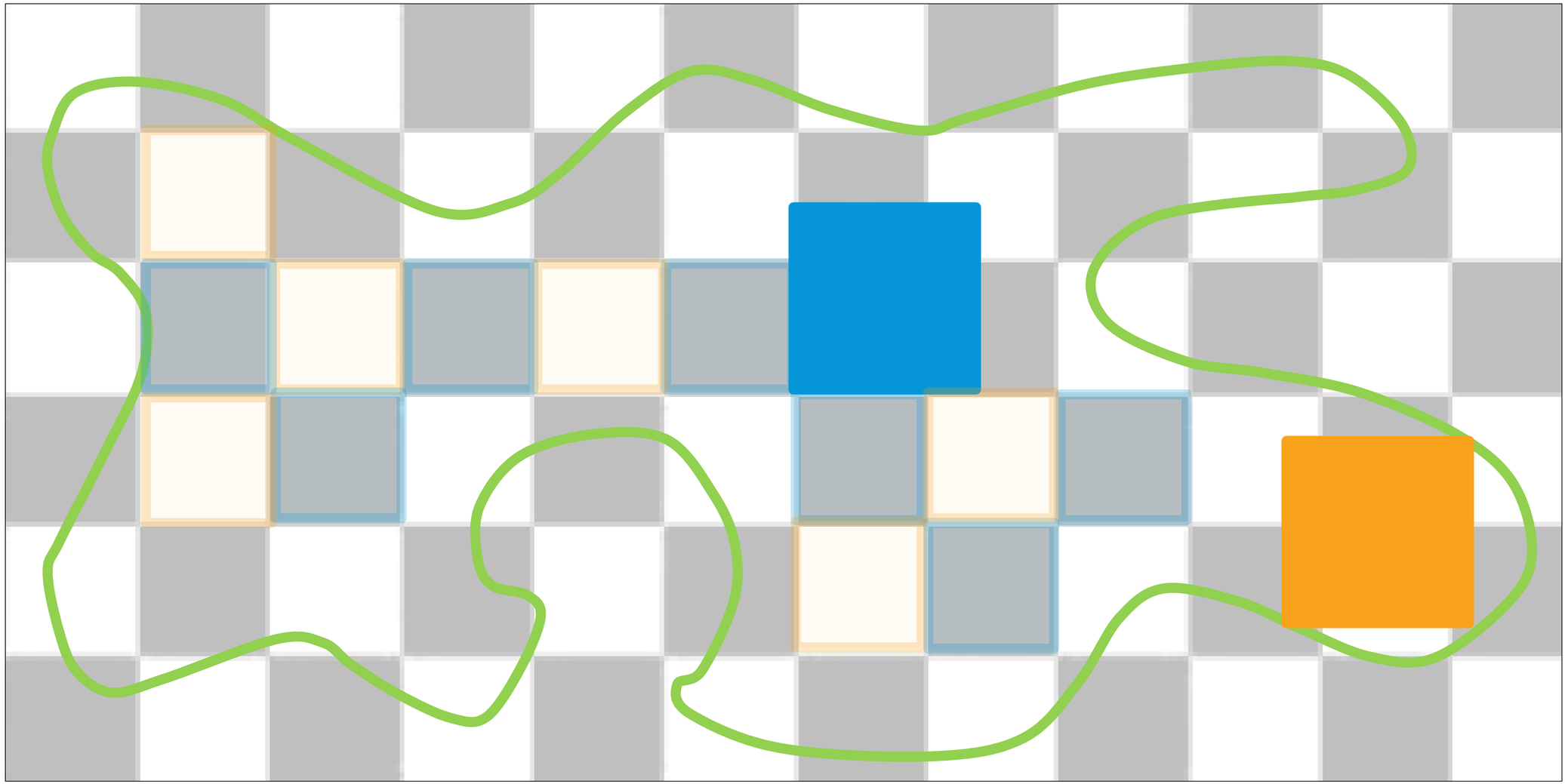
Generation Systems - Assign Values



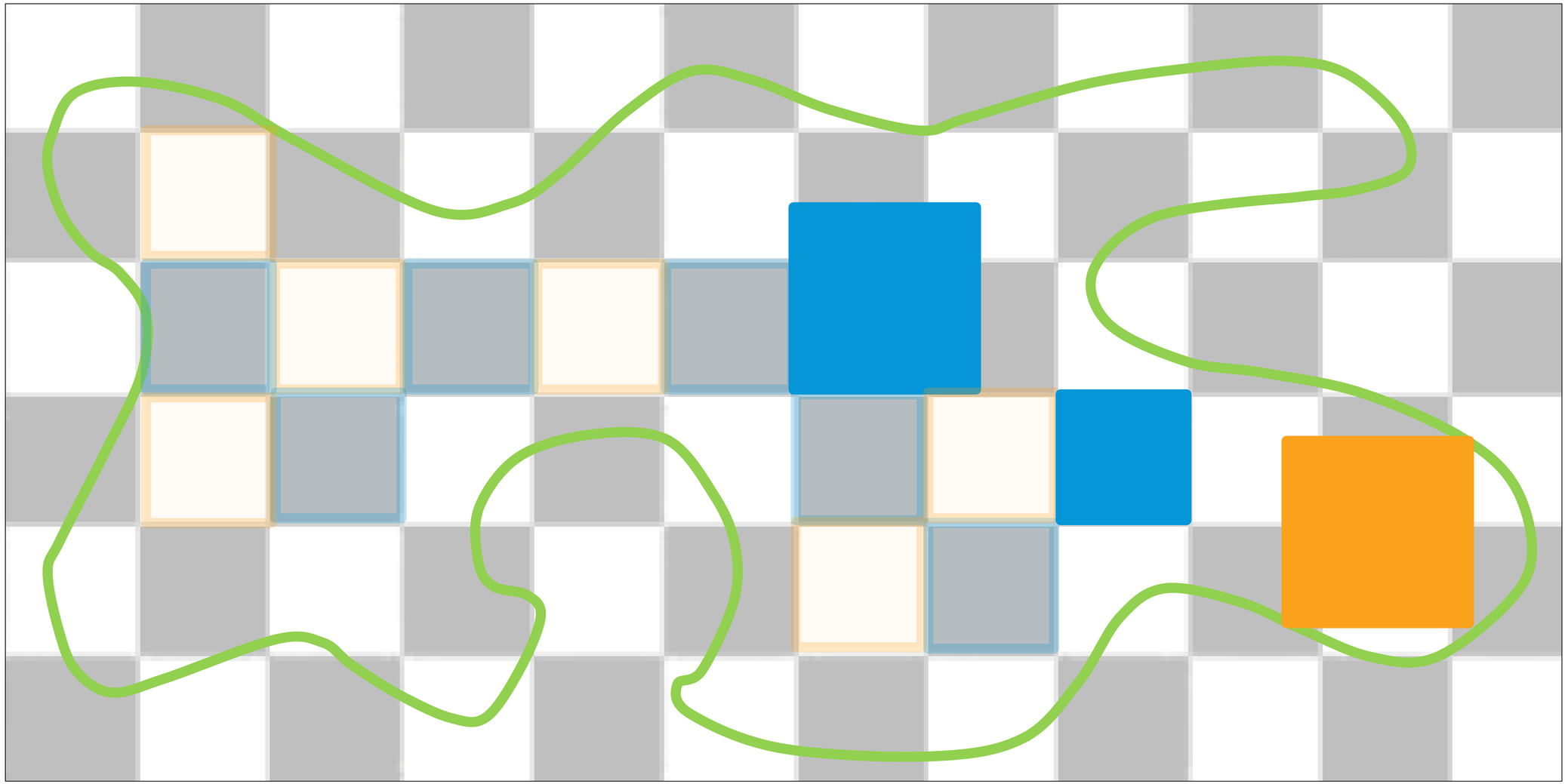
Generation Systems - Assign Values



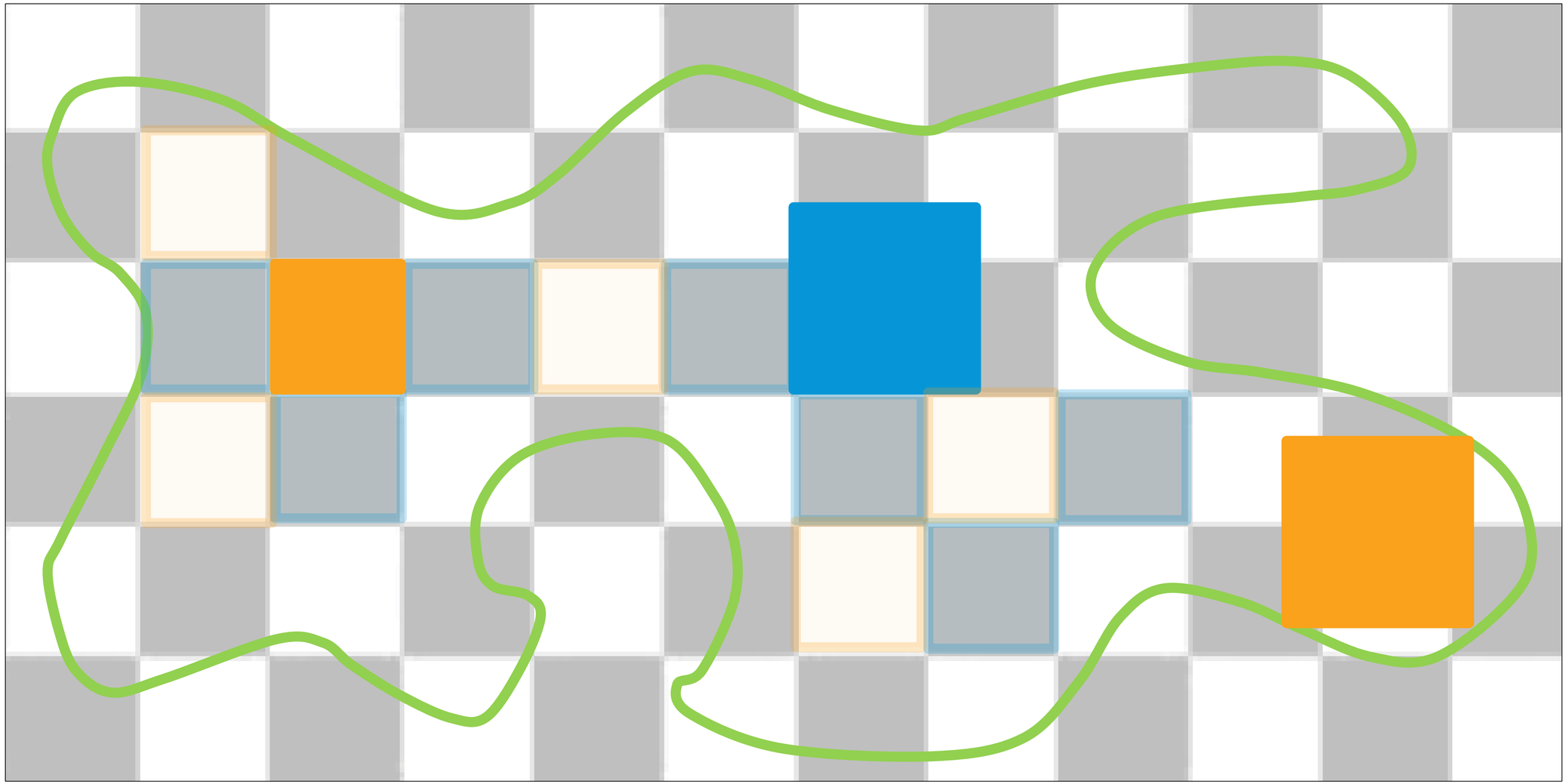
Generation Systems - Assign Values



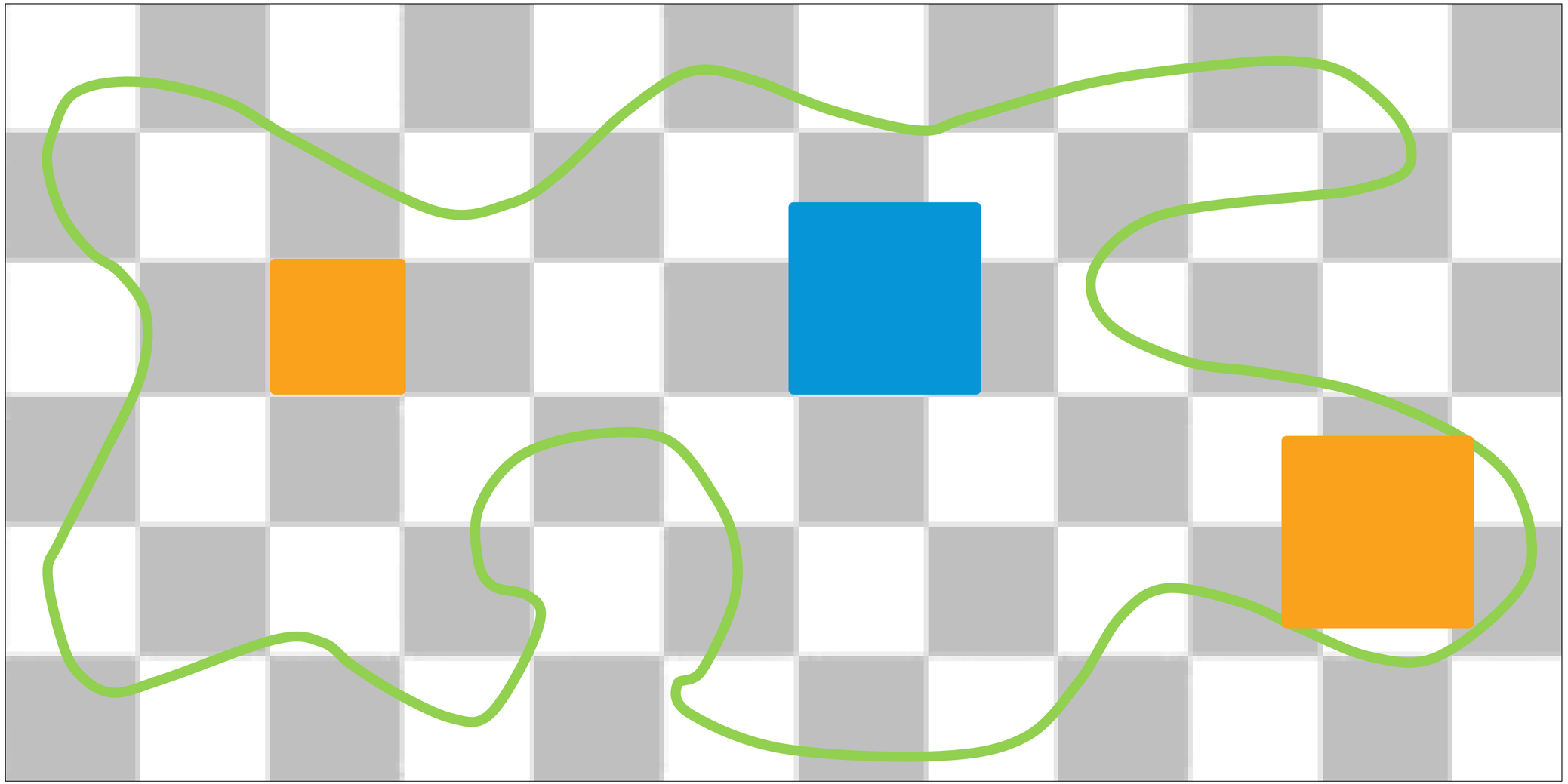
Generation Systems - Assign Values



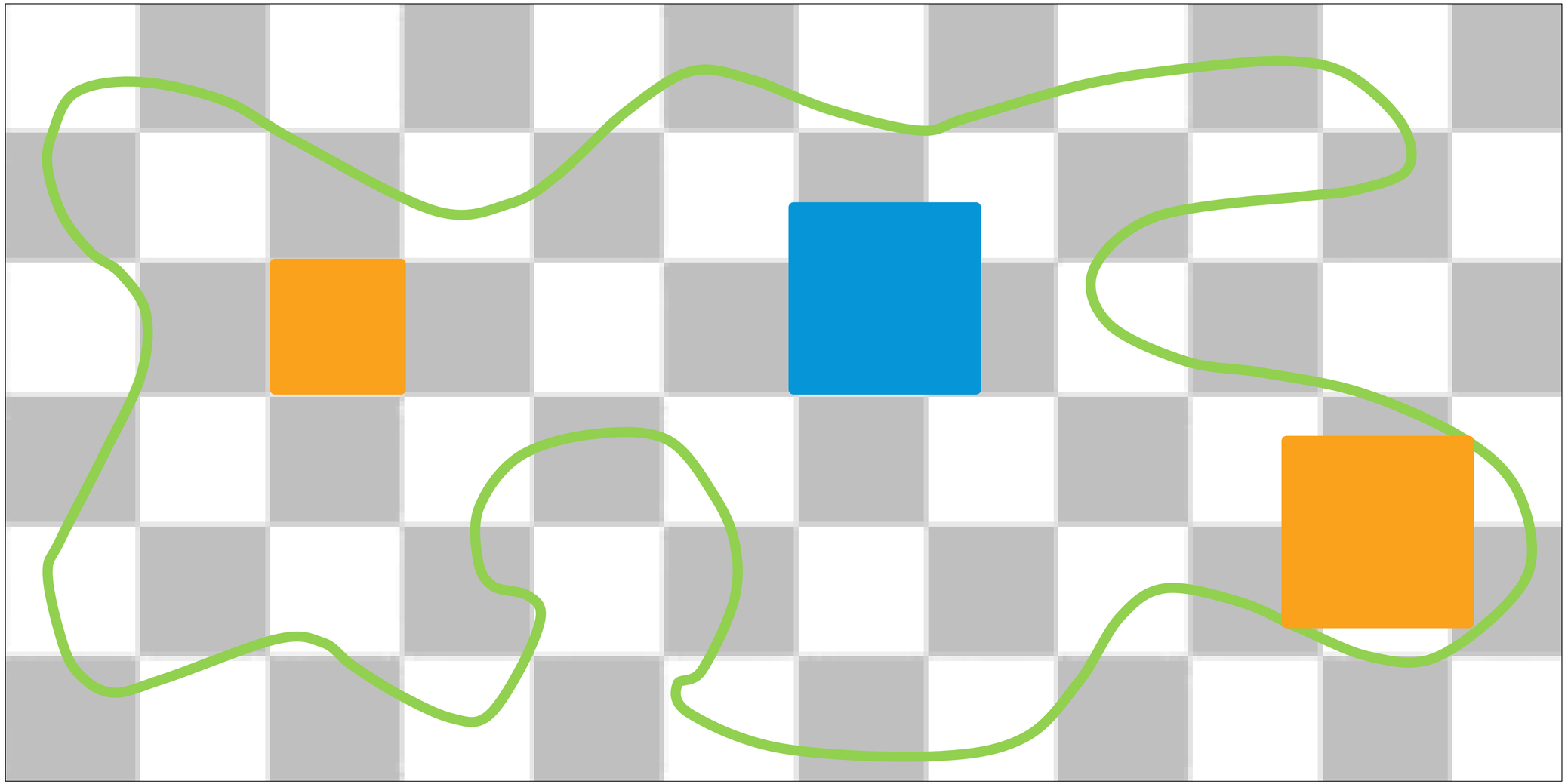
Generation Systems - Assign Values



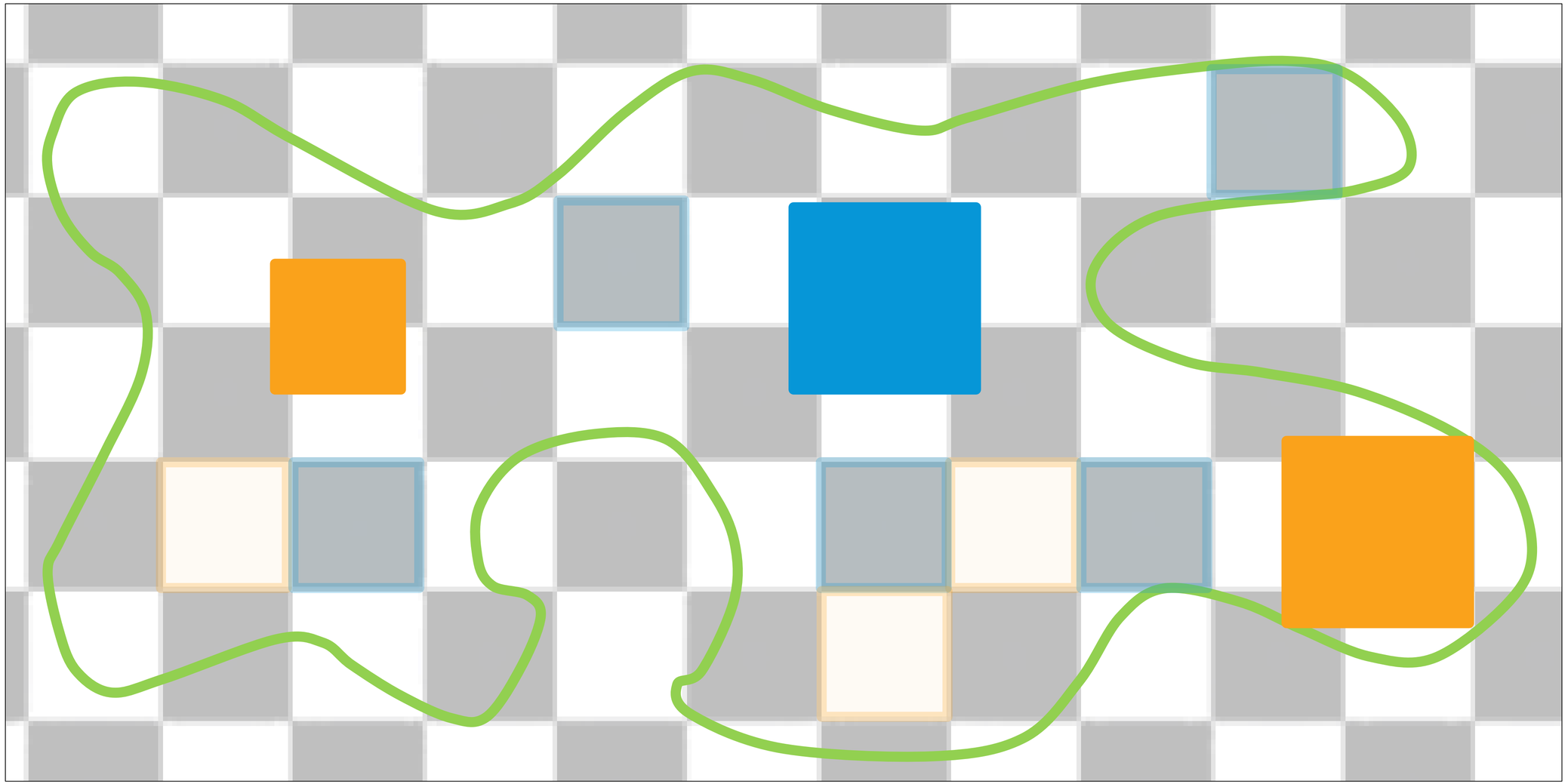
Generation Systems - Assign Values



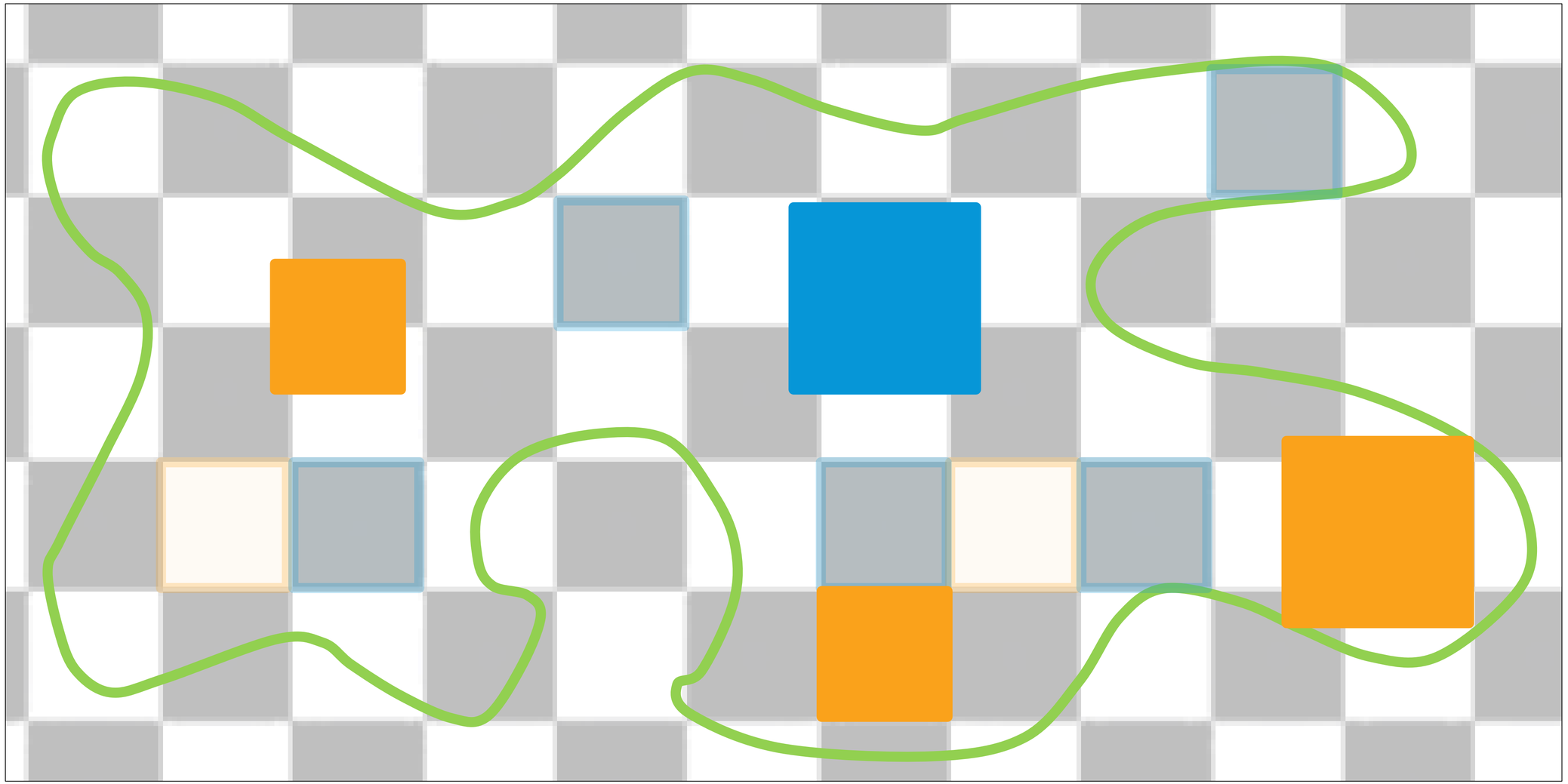
Generation Systems - Assign Values



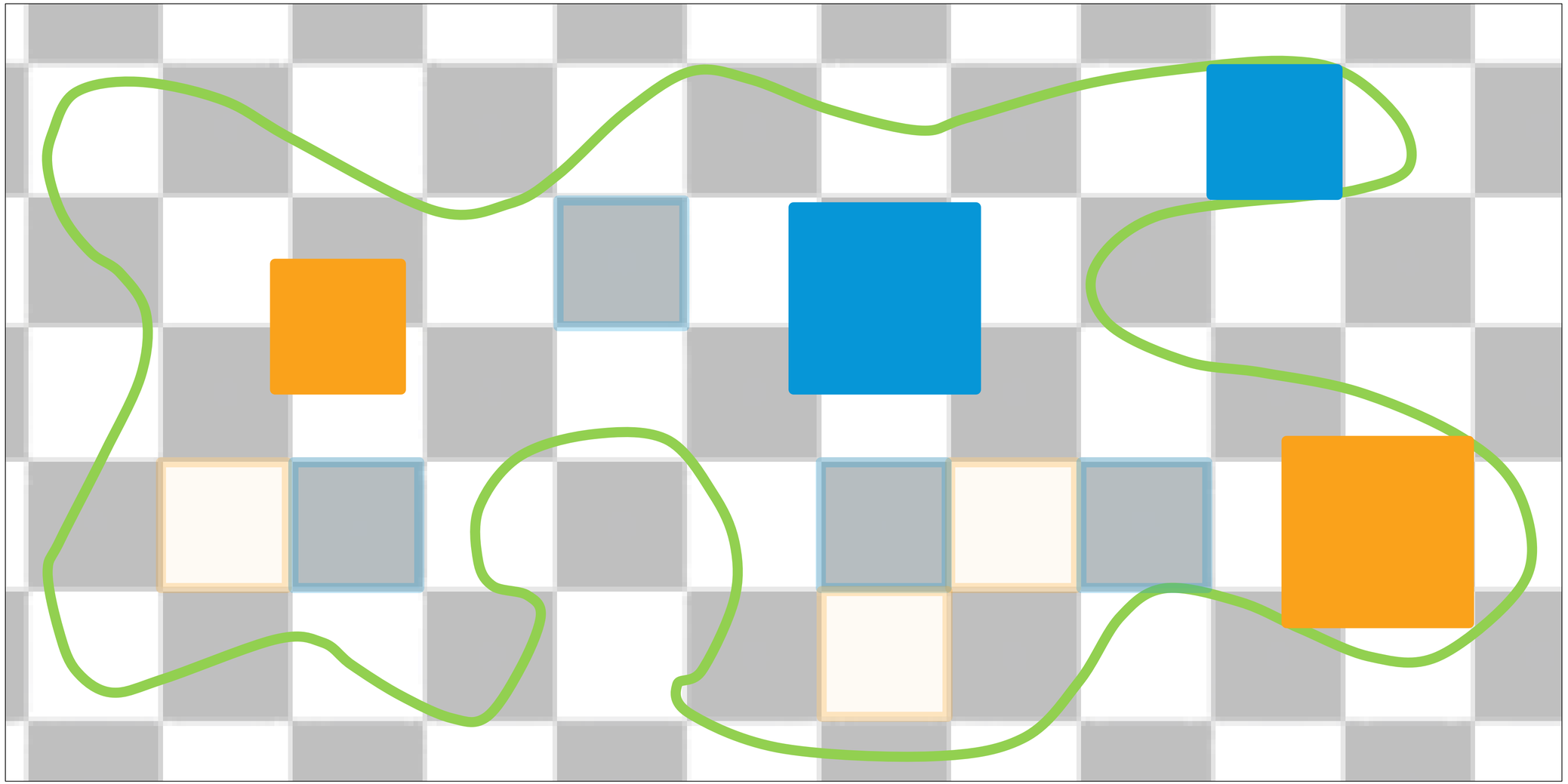
Generation Systems - Assign Values



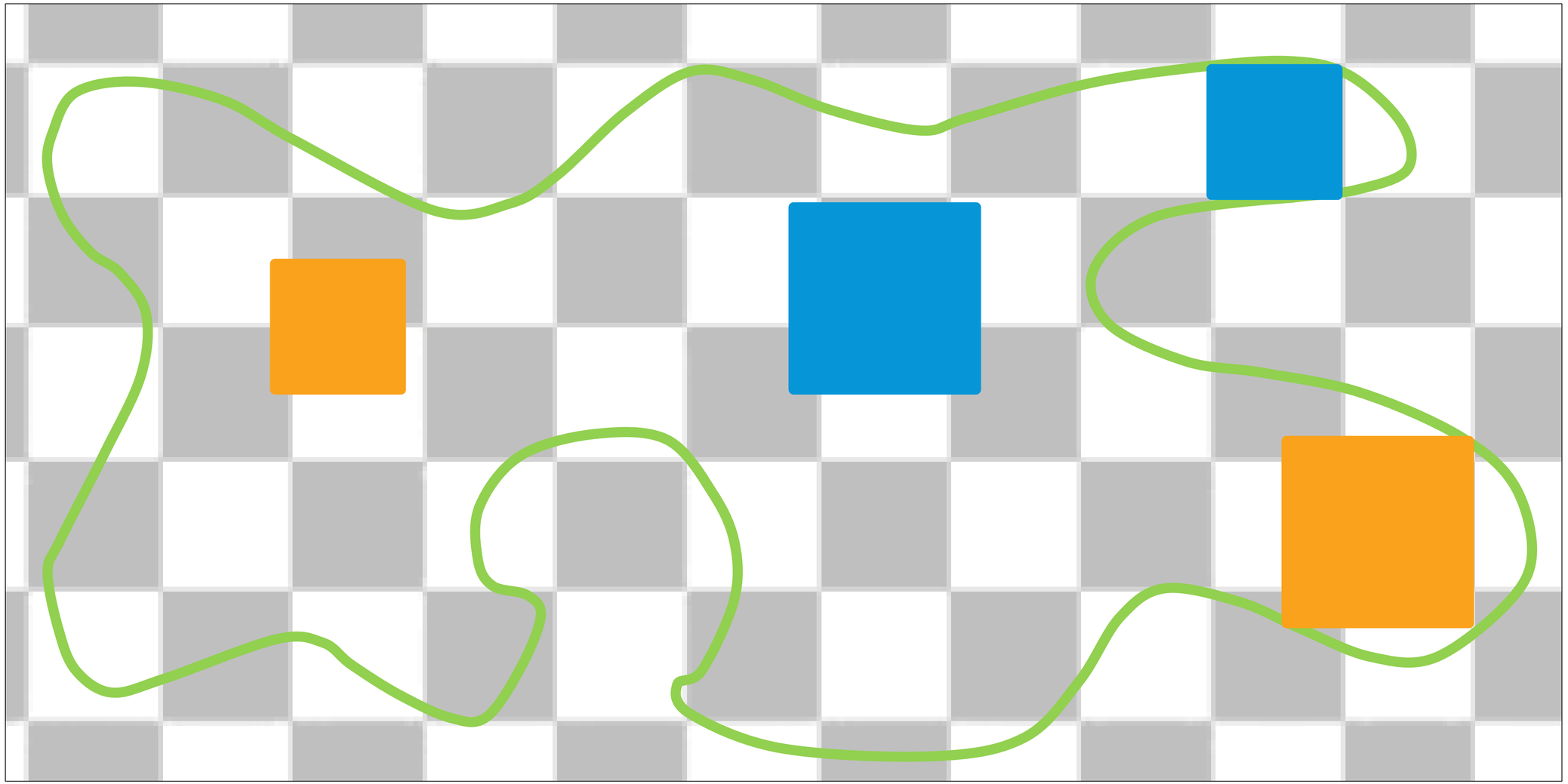
Generation Systems - Assign Values



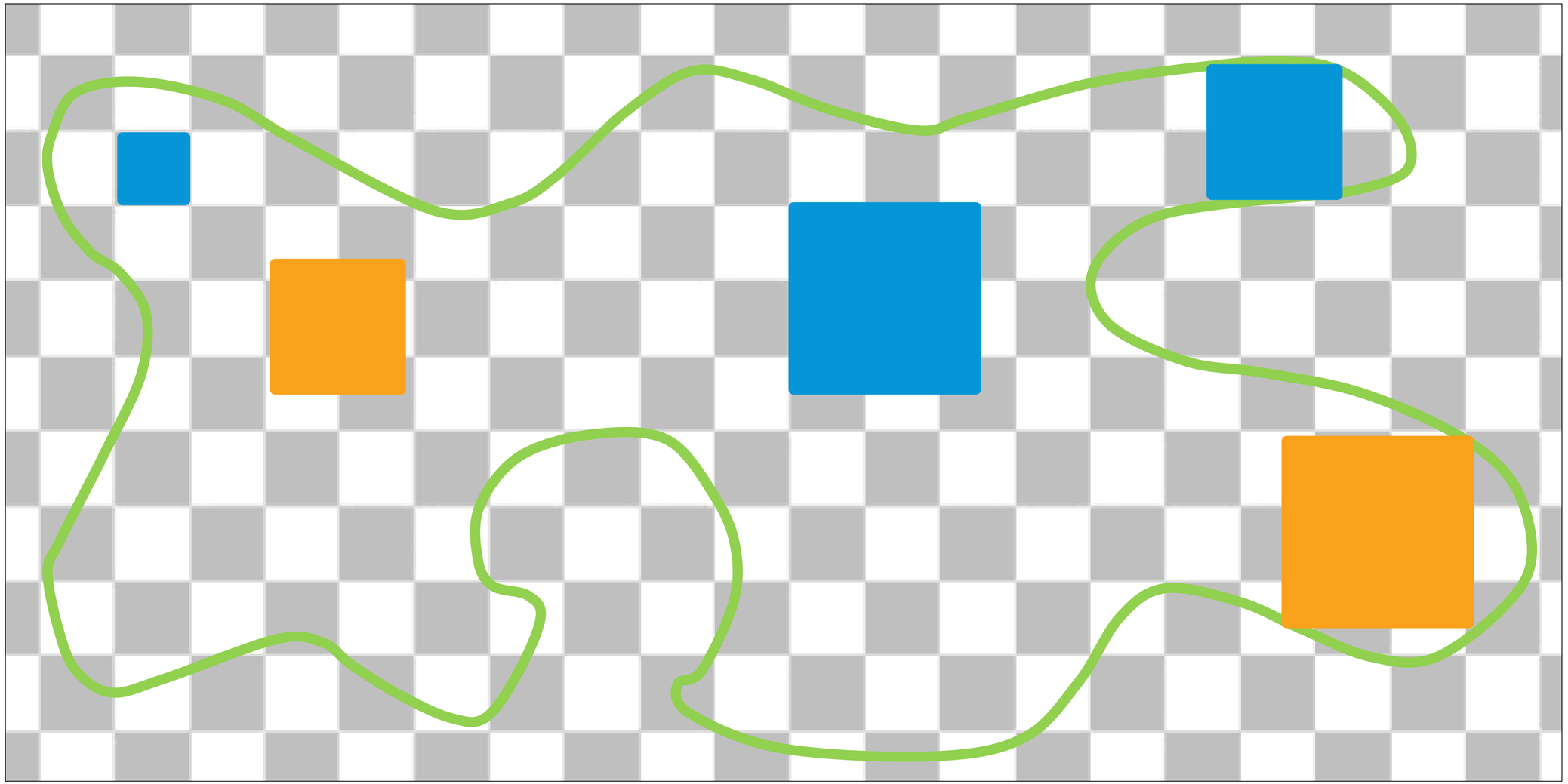
Generation Systems - Assign Values



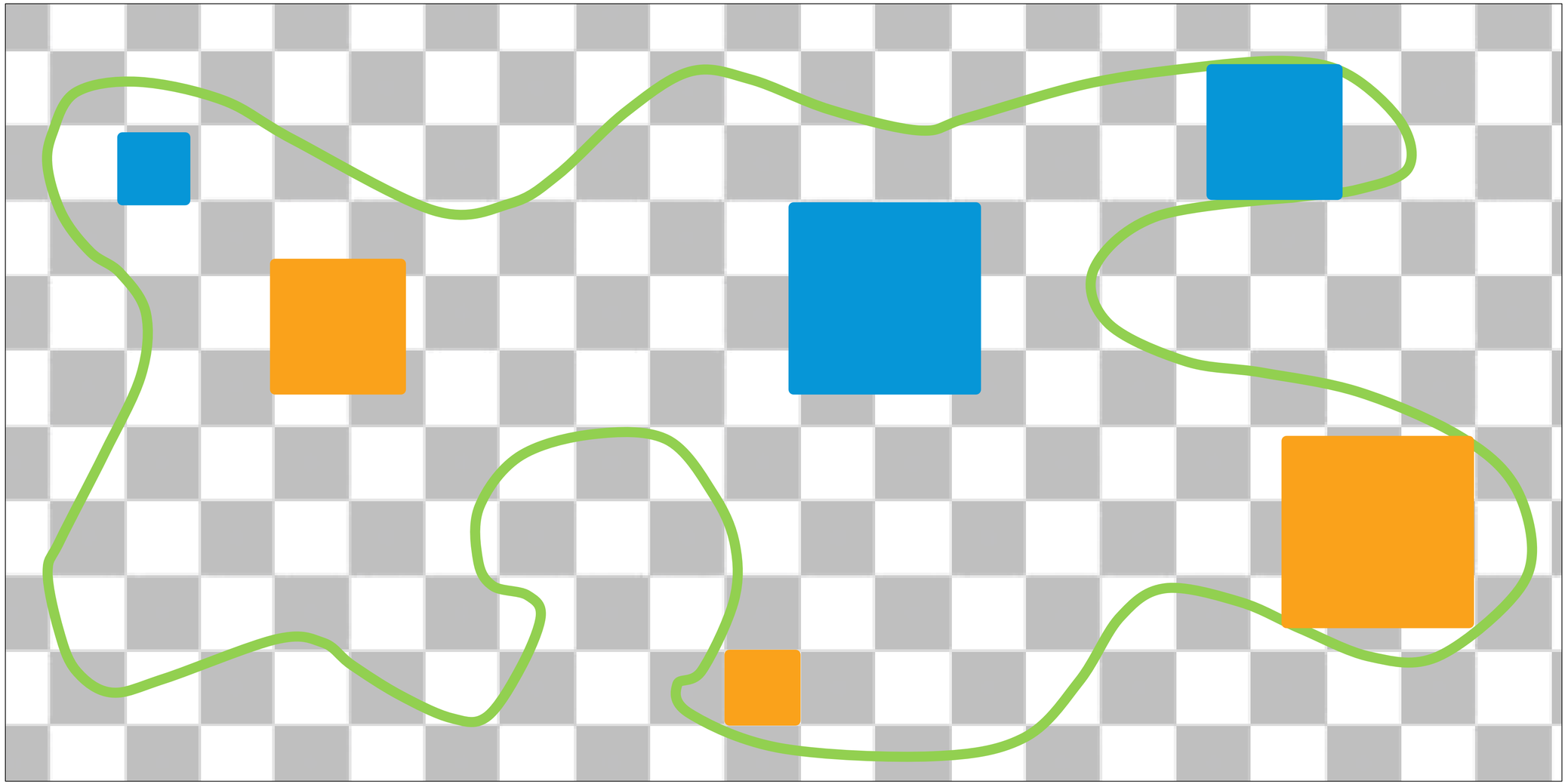
Generation Systems - Assign Values



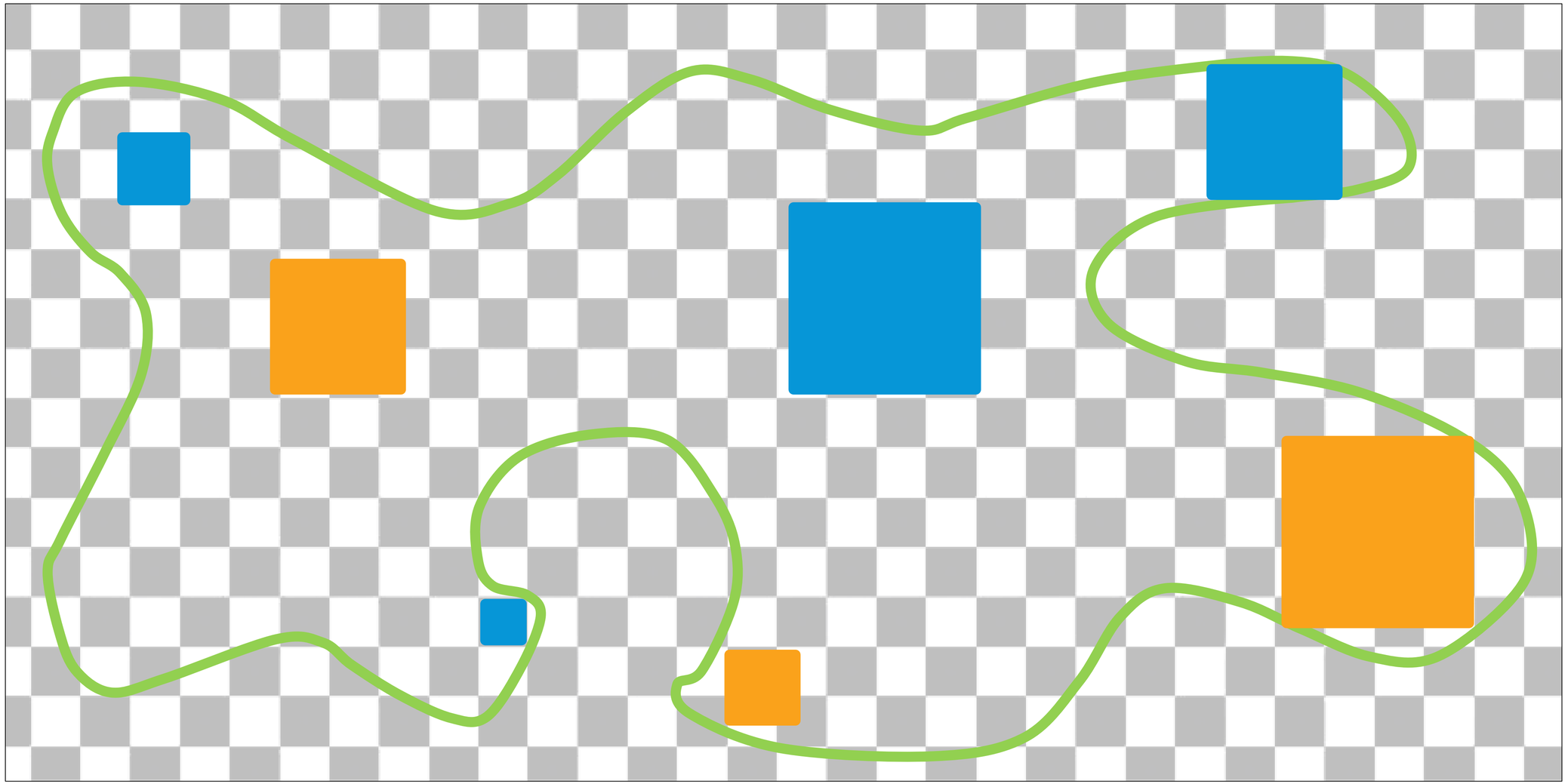
Generation Systems - Assign Values



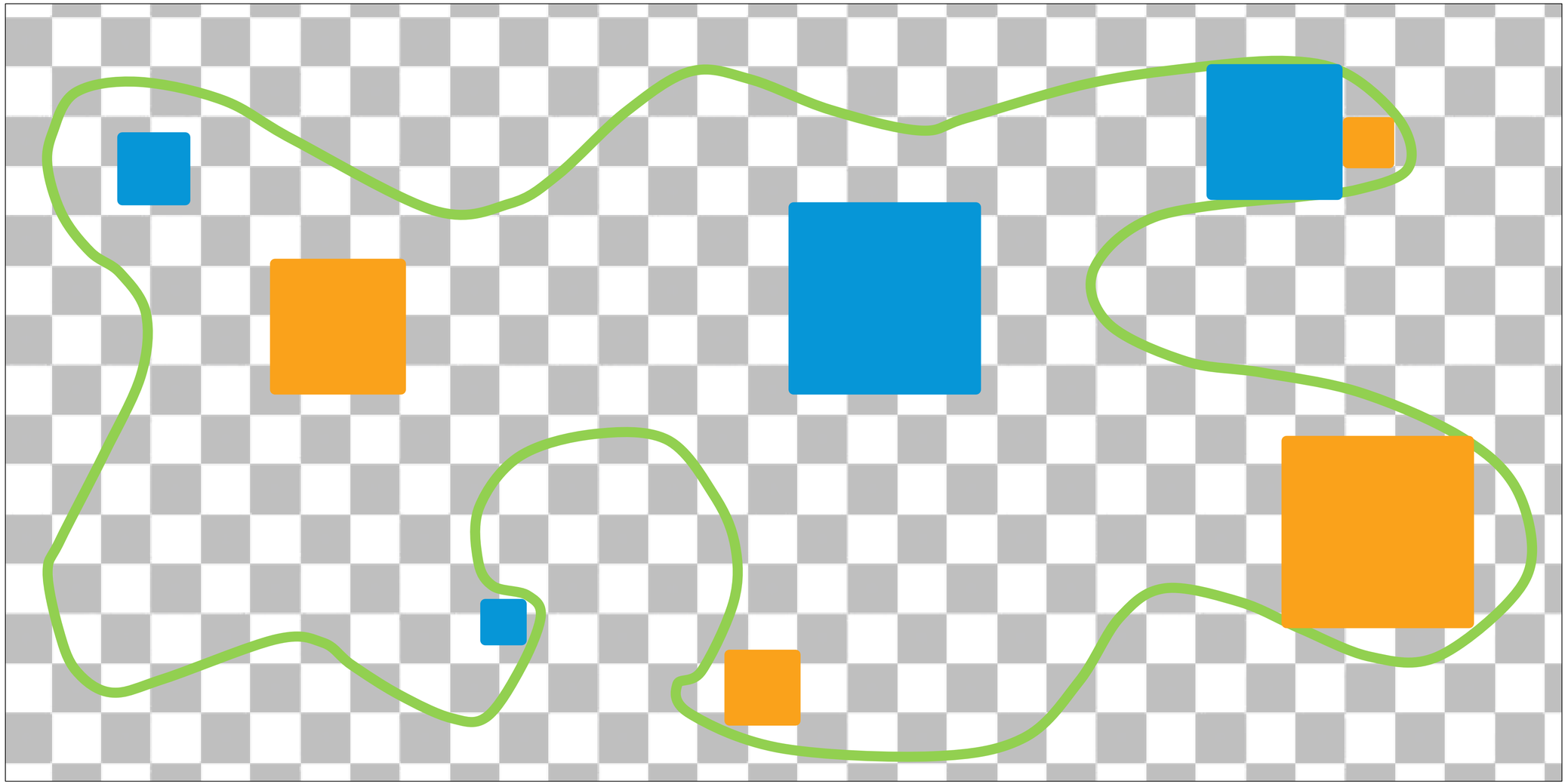
Generation Systems - Assign Values



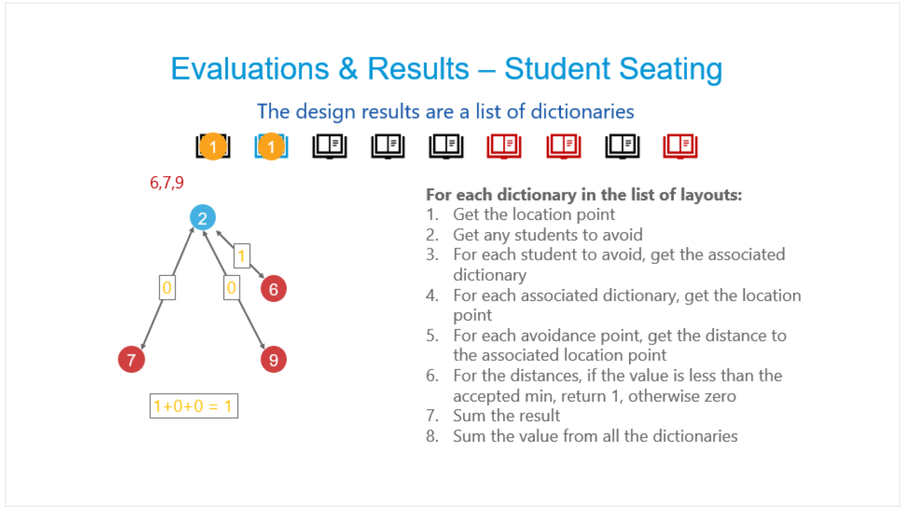
Generation Systems - Assign Values



Generation Systems - Assign Values



Evaluations & Results – Creature Layout



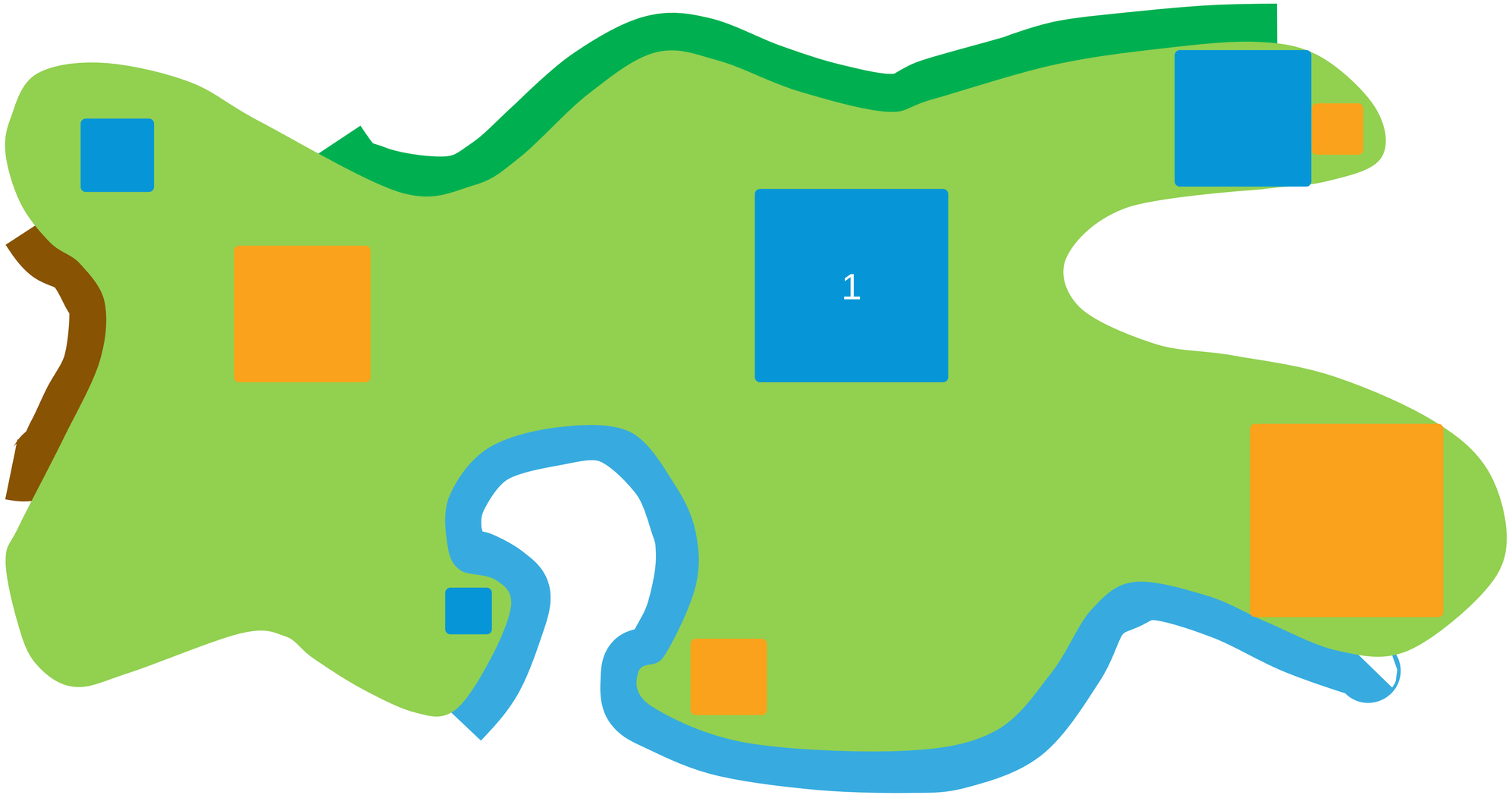
Remember this?

For each dictionary in the list of layouts:

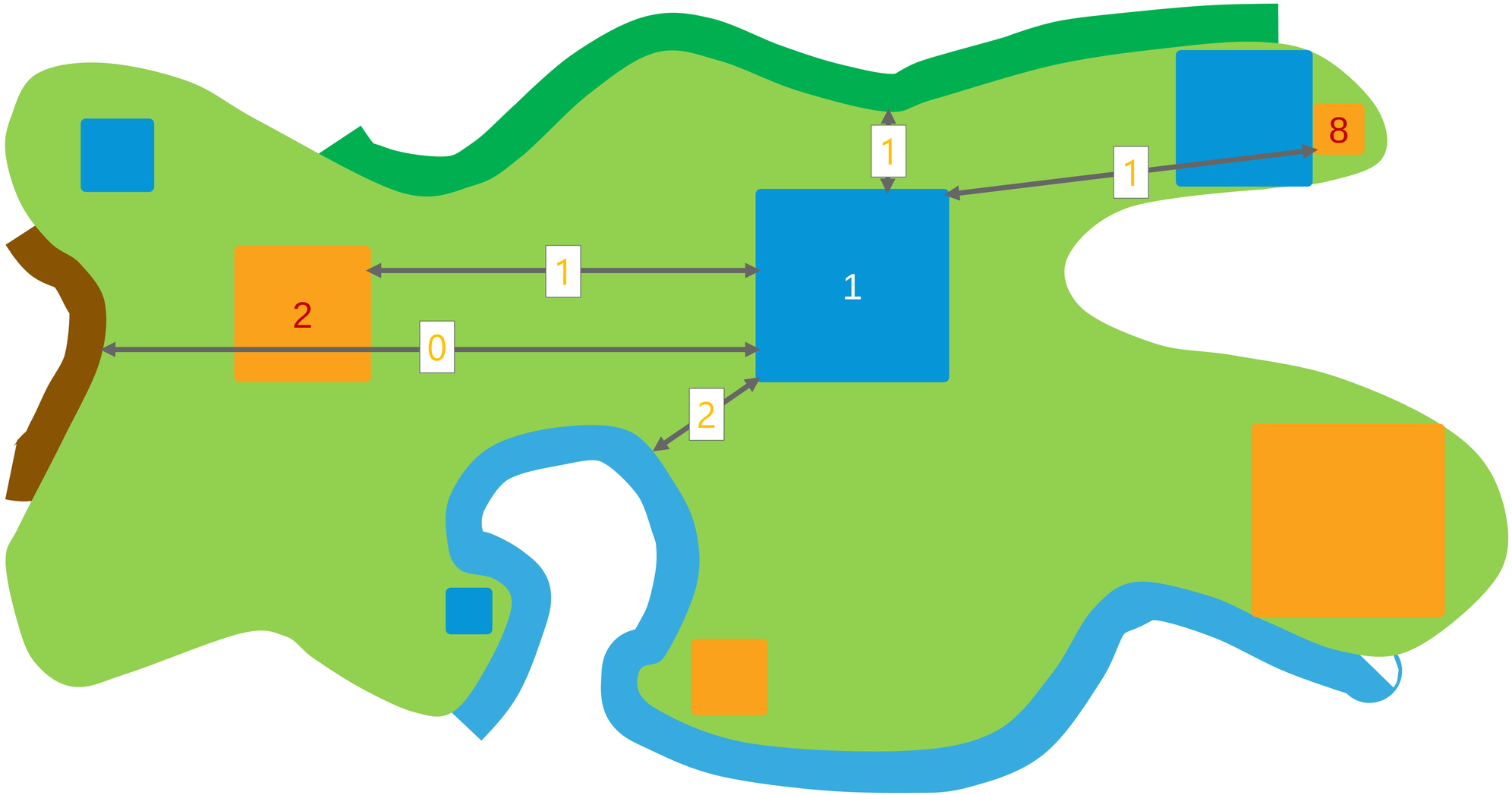
1. Get the space
2. Calculate the adjacency scores:
    A. Get any creatures to avoid, and pull the associated dictionary
    B. For each associated dictionary, get the space
    C. For each avoidance space, get the distance to the creature space
    D. For the distances, if the value is less than the accepted min, return 1, otherwise zero
    E. Sum the result
    F. Sum the value from all the dictionaries
3. Calculate the contextual separation scores
    A. For each context item, get the associated geometry
    B. Get the distance from the context geometry to the space
    C. For the distances, if the value is less than the accepted min, return 1, otherwise zero
    D. Sum the results
    E. Sum the value from all the dictionaries

Evaluations & Results – Creature Layout

Evaluations & Results – Creature Layout



Evaluations & Results – Creature Layout



The world is an infinitely big, and infinitely small place.

That is Terrifying.

As a result, some problems cannot be simplified to a ‘selection of options’ without getting well beyond the scope of seeded randomization or permutations.

The “Infinite” Scale Problem

Bad news: https://docs.python.org/3.0/library/random.html

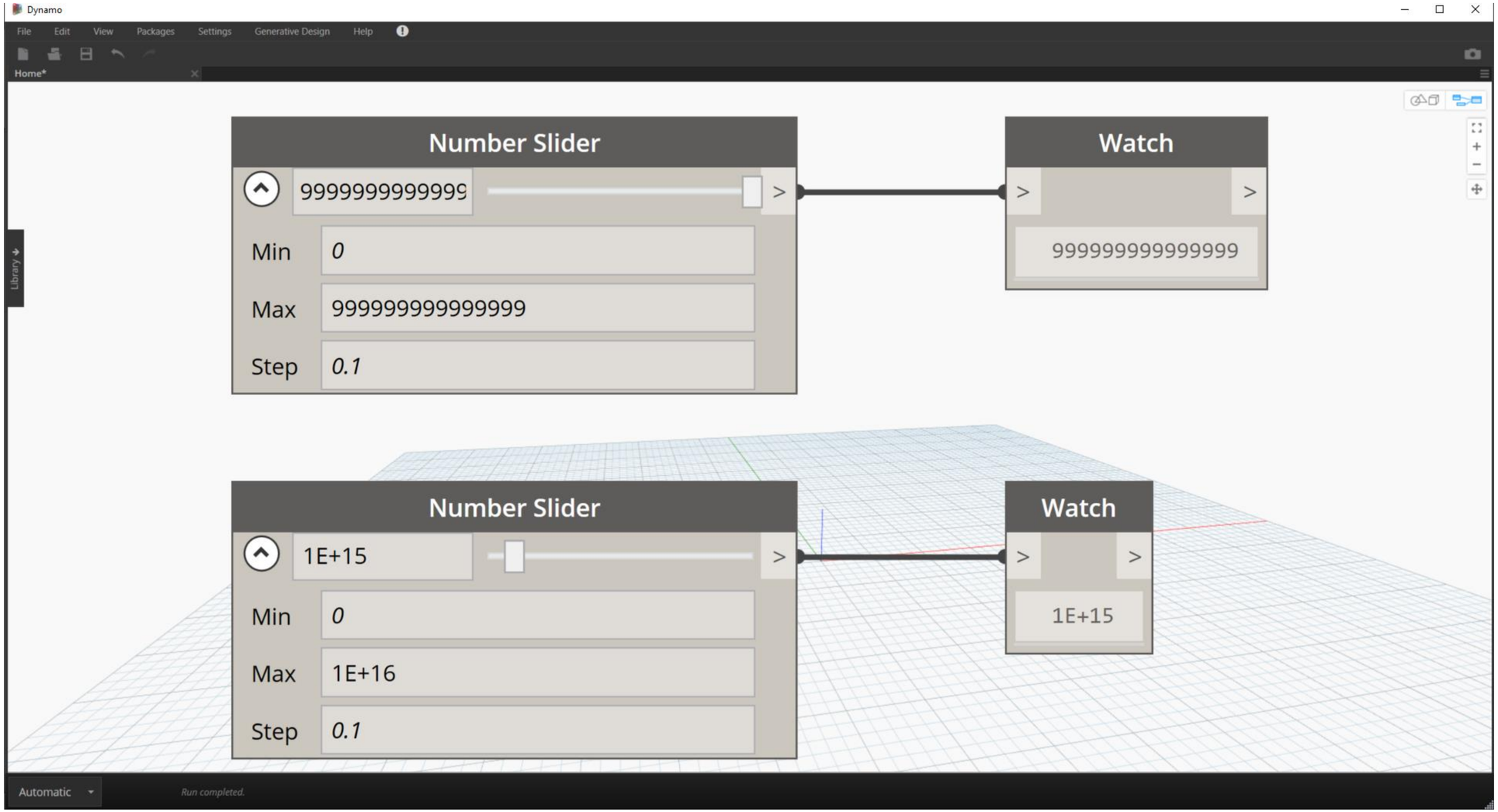
> **random.shuffle(*x*[, *random*])**: Shuffles the sequence *x* in place. The optional argument *random* is a 0-argument function returning a random float in [0.0, 1.0); by default, this is the function random().
>
> Note that for even rather small len(x), the total number of permutations of *x* is larger than the period of most random number generators; this implies that most permutations of a long sequence can never be generated.

But don’t worry: https://stackoverflow.com/a/21284969

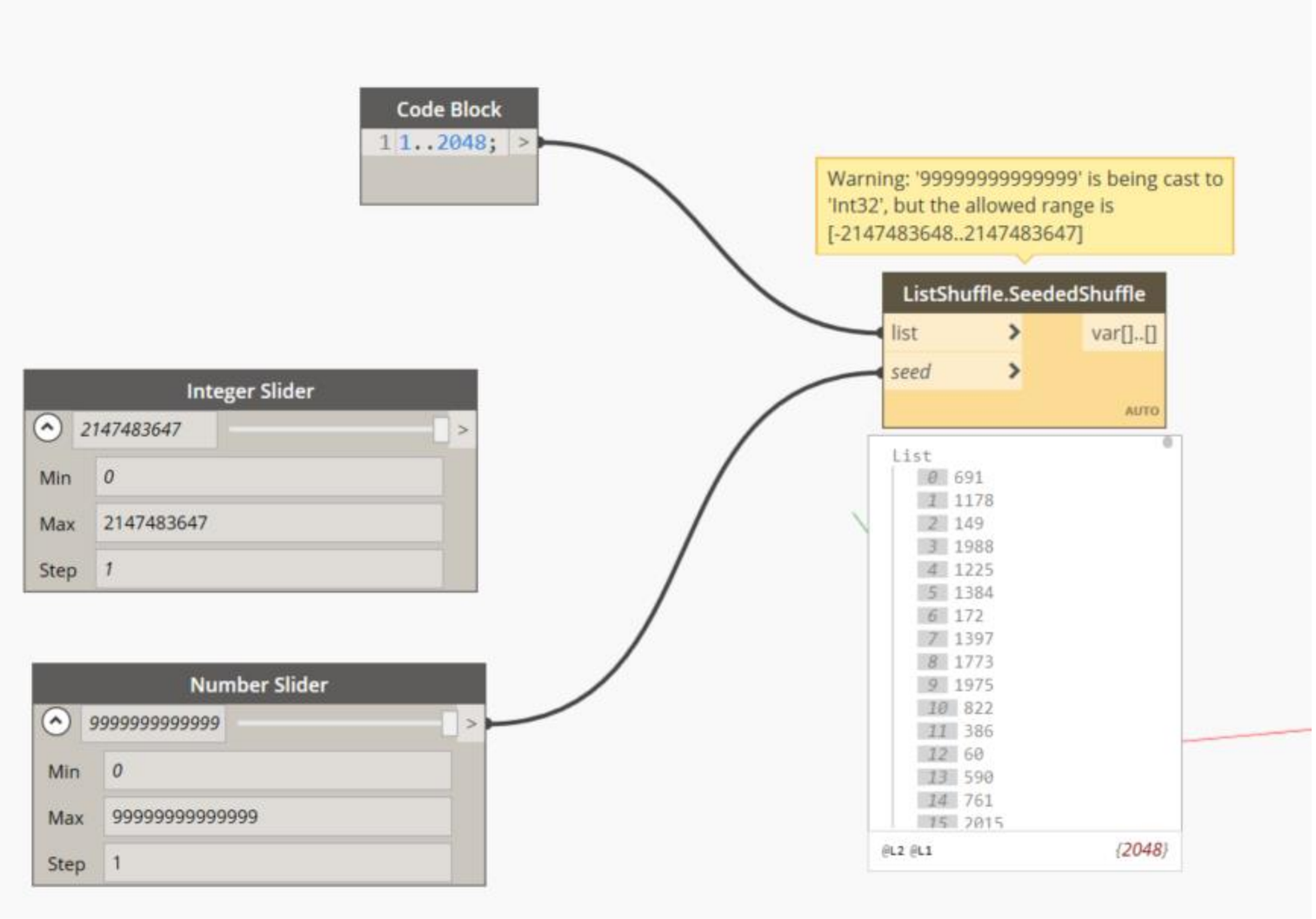
The default random.random() function uses the Mersenne Twister algorithm, which has a period of 2**19937-1. That means that you'd need a list of length 2081 before you will see this behavior occur.

But…

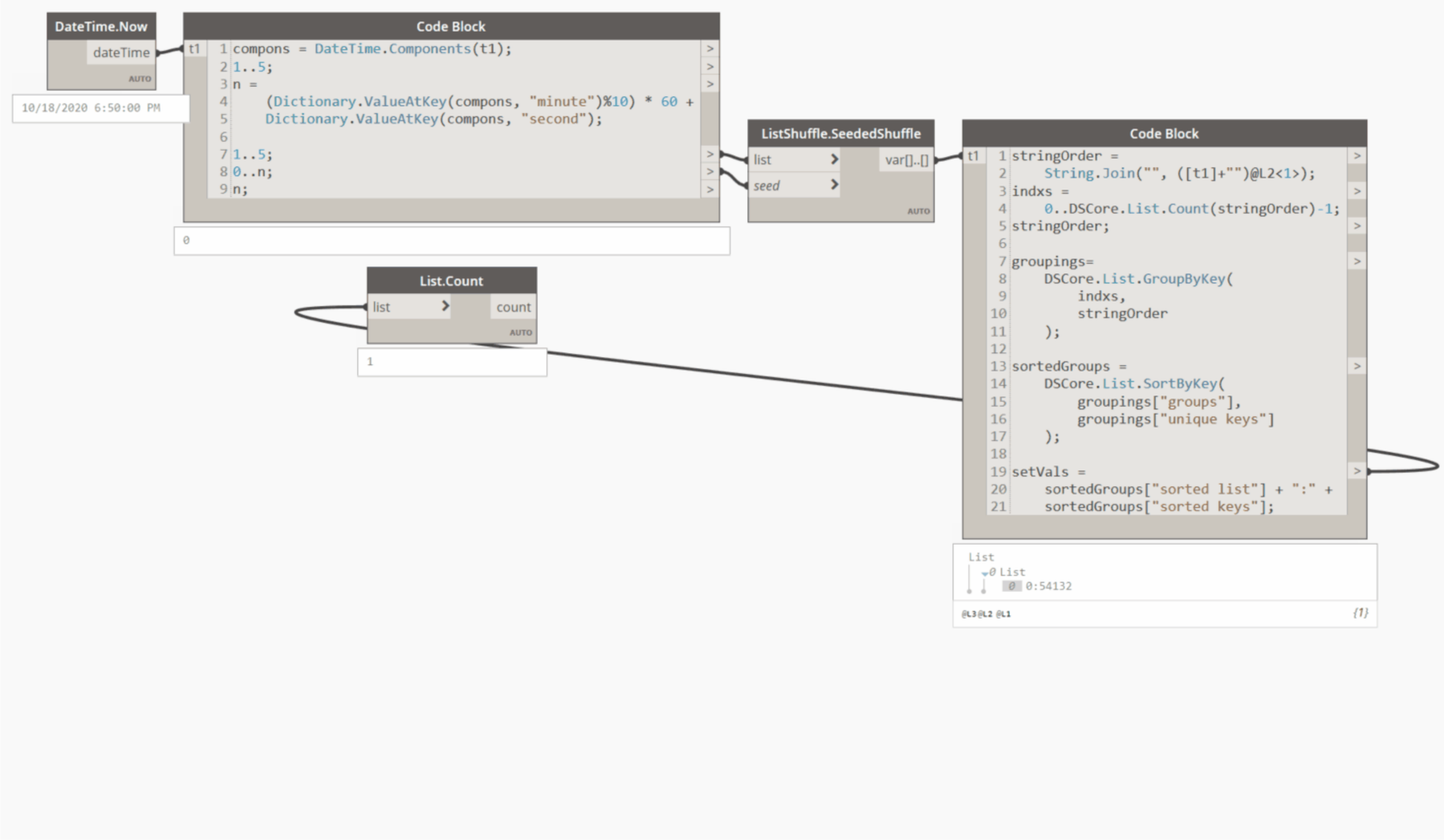
The “Infinite” Scale Problem



The “Infinite” Scale Problem



The “Infinite” Scale Problem



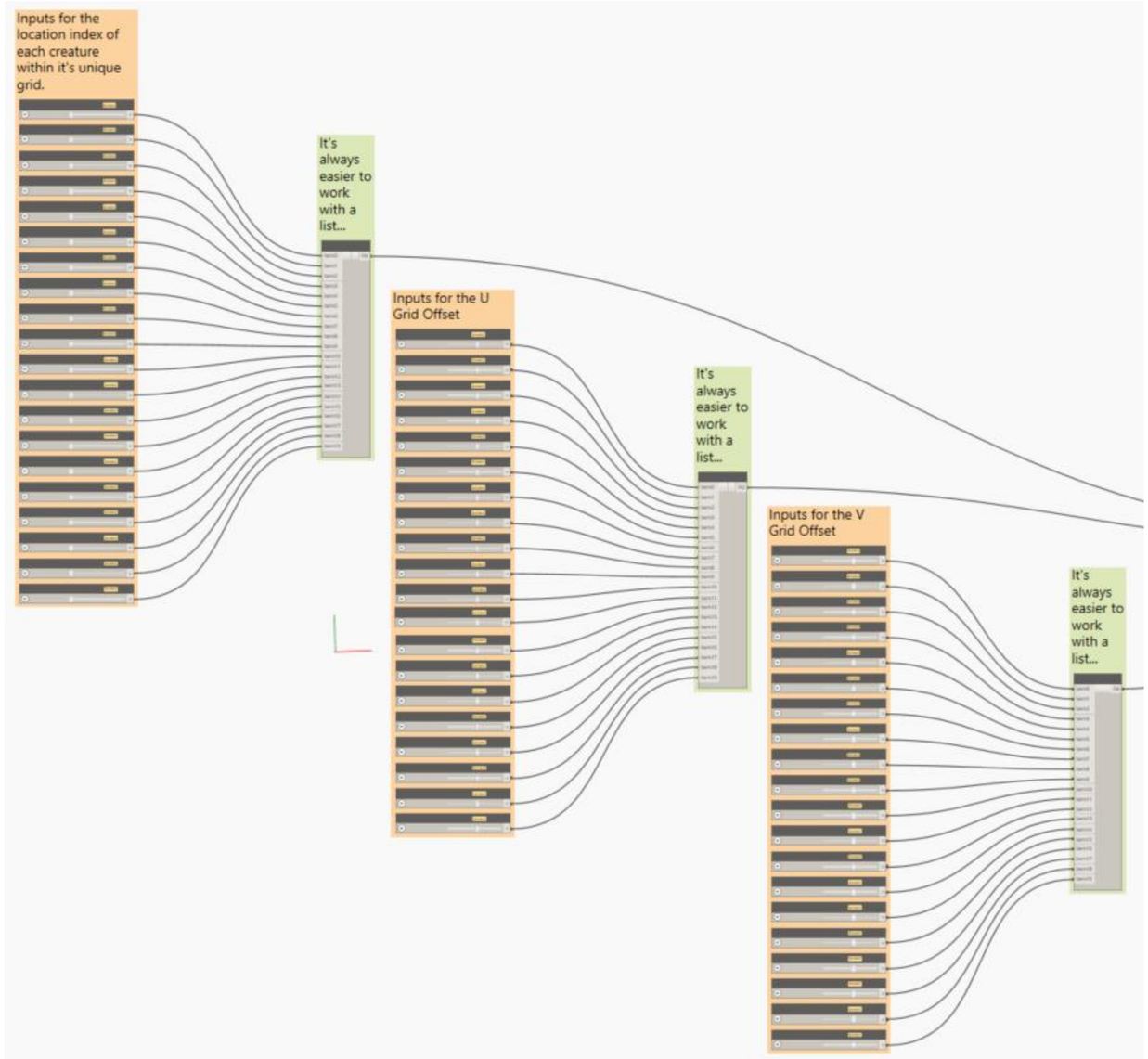
473 searches to find all 120 items!

The “Infinite” Scale Problem

If his forces are united, separate them.

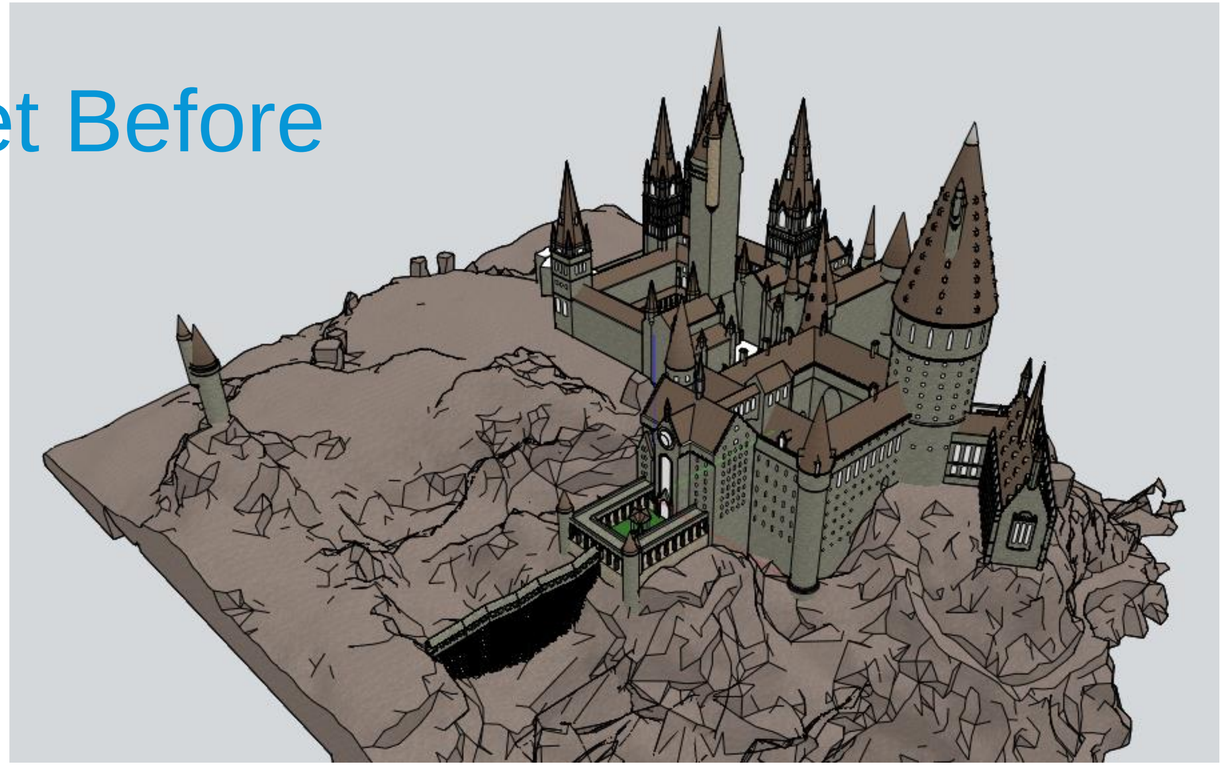
Sun Tzu, The Art of War

The “Infinite” Scale Problem



Data Set Before

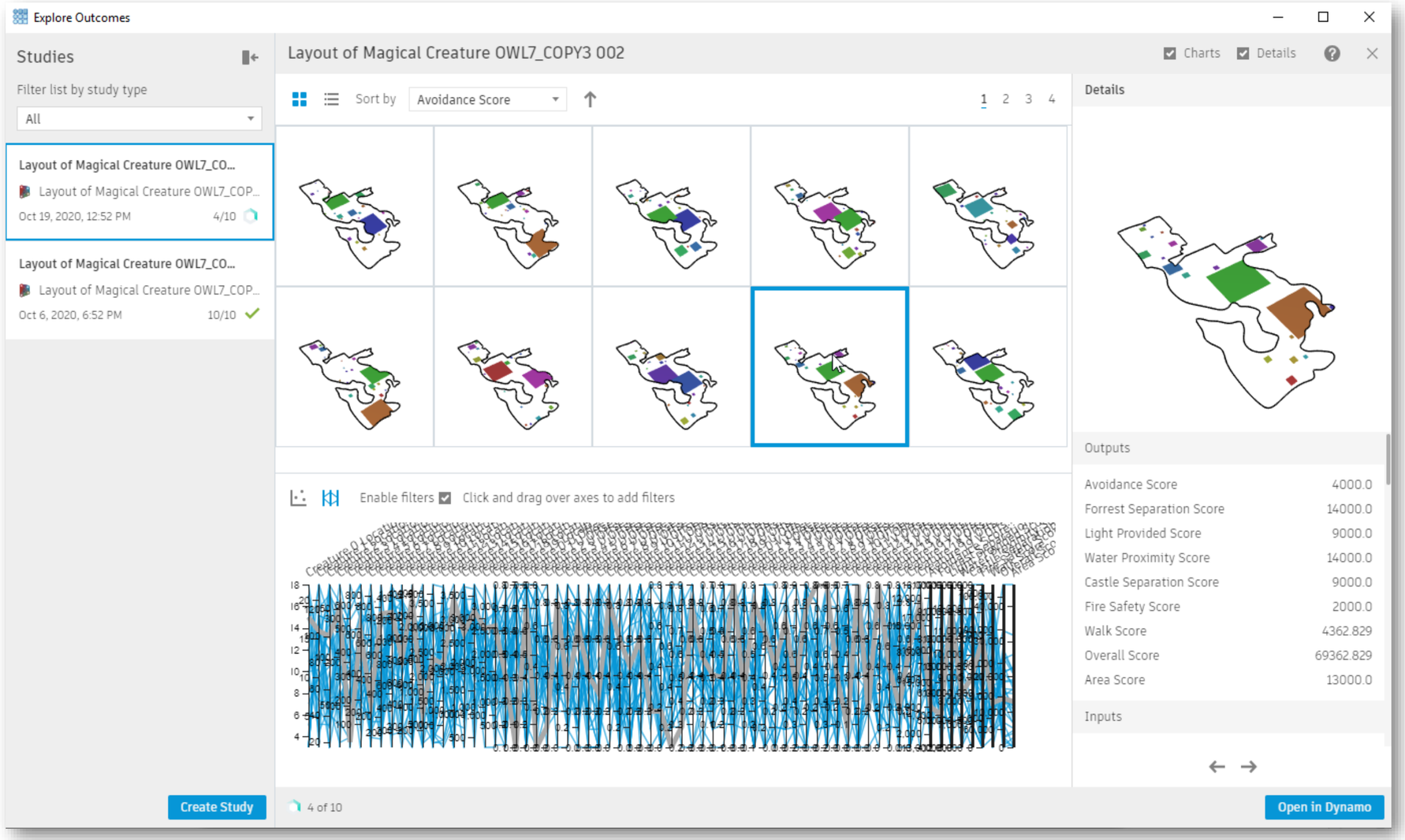
- **Model** of Hogwarts Castle and Grounds showing 2D site for creature layout.
- **Creature data**: creature number, name; danger scale; area requirement; creatures to avoid; separation from Castle; light requirement; water requirement; fire risk; separation from forest; and notes!



	A	B	C	D	E	F	G	H	I	J	K
1	Creature	Name	Danger Scale	Area Requirement	Creatures to Avoid	Separation from Castle	Light Requirment	Water Requirement	Fire Risk	Seaparation from forest	Notes
2	0	Abraxan	3	4000	8,	1	3	1	1	1	Wingged horse the size of an elephant. Needs to fly.
3	1	Ashwinder	3	200	9, 10, 12, 13, 17,	4	1	1	5	5	Snake born in magic flame. Eggs burn down the house if not found and hexed in time.
4	2	Billywig	3	200	14,	1	2	1	1	1	Small insect. Fast. Causes flaoting
5	3	Boggart	3	750	0, 1, 2, 3, 4, 5, 6, 7, 8, 9,	5	5	5	5	5	Takes shape of what student fears. Unpredictable results but stable when supervised
6	4	Bundimun	3	200	1,	5	1	3	1	2	Greenish fungus with eyes - causes structures to rot.
7	5	Demiguise	4	500	7, 15,	1	3	2	1	2	Apelike creature which can turn invisible and predict the future.
8	6	Dugbog	3	300	2, 18, 11, 12, 13,	5	1	5	1	1	Marsh dweller whicih looks like a piece of wood, but has finned paws. Feeds on small animals and bites ankles. Eats Mandrakes.
9	7	Erumpent	4	30000	5, 14, 19,	5	5	1	5	3	Similar to rhino in size. Horn injects a fluid which explodes.
10	8	Hippogriff	3	2000	1,	2	3	2	1	1	Needs to fly. Bow on approach. Eats birds, insects, other small mammals.
11	9	Kappa	4	800	2, 18,	5	1	5	1	1	Water dwelling creature. Strangles humans who enter their area for food. Tricked by cucumbers or causing it to 'spill' the water
12	10	Kelpie	4	800	2, 18,	5	1	5	1	1	Water dwelling shape shifter, usually takes the form of a horse which lures travelers onto it's back before swimming to the bott
13	11	Kneazle	3	200	6,	1	3	2	1	1	Cat like, highly intelligent. Sometimes aggressive. Capable of guaarding homes.
14	12	Leprechaun	3	200	1, 17, 6,	4	2	3	1	1	Green skin, about 6" high. Mischievous in nature. Herbivorous.
15	13	Niffler	3	200	6,	5	2	2	1	1	Long snouted mammal, about the size of a house cat. Collects all the shiney things they can get their hands on.
16	14	Occamy	4	1000	7, 2,	3	2	1	1	1	Serpant like creature with wings. Grows (or shrinks) to the size of it's habitat. Feeds primarily on ratsa nd birds, but has been kn
17	15	Red Cap	3	500	5,	3	3	2	1	3	Small dwarf like creature, about 3'-6" tall. Bludgeons unwary to death.
18	16	Re'em	4	30000	19,	1	4	2	1	1	Resembles an Ox. Fairly large ground required.
19	17	Runespoor	4	500	1, 9, 10, 12, 13,	5	2	3	1	5	3 headed snake, up to 7' long.
20	18	Salamander	3	200	6, 9, 10,	5	4	1	5	4	Born in salamander fire, and live as long as the flame does. Need to return to the flame to keep living.
21	19	Tebo	4	5000	7, 16,	4	5	1	1	5	Warthog which can turn invisble.

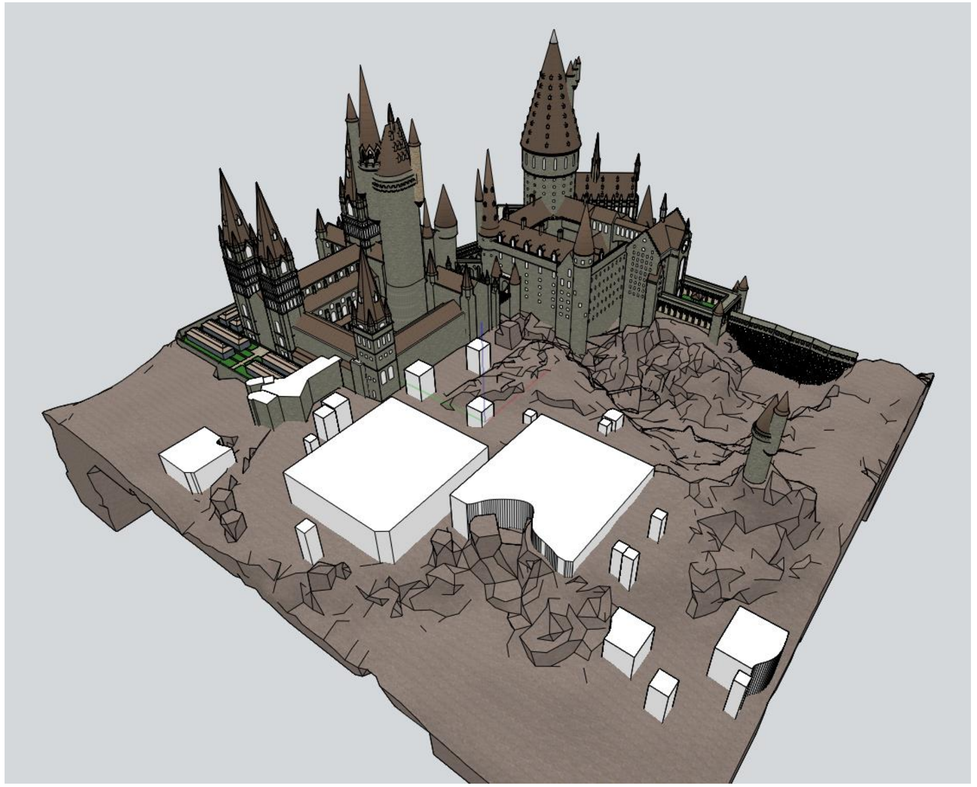
Let’s build the graph!

Generation & Results



Notes and Review

- Beware of competing goals which may work against one another as you attempt generate an optimized design solution.
- The most important result metric should drive your decision - set them as the ‘first’ filter.
- When working with a large set of parameters, use the "Parallel Coordinates" viewer to help in making a final selection. The "Scatter Plot" viewer limits you only to 4 parameters: an X-Axis, Y-Axis, Color, and Size.



Resources and Next Steps

Notes and Review

1. **Simplify the problem whenever possible**
   - Abstract the problem, then scale it down, and reduce it to the ‘core parts’
2. **Do not plan on eating the entire erumphant in one sitting**
   - Many battles may produce a more complete victory than one
3. **Getting a result does not mean that is your design**
   - Partner with the machine, don’t follow it
4. **Use your compute cycles wisely**
   - Just because you’re not at your desk doesn’t mean you can’t be looking for the solution
5. **Crushing with the side of the blade can release juices better than cutting**
   - Tools often have more than one use learn as many as you can
6. **Remember to remember**
   - Generative Design works faster if it is all ‘in the file’, but don’t overuse the node!
7. **Gates are good, but freezing can be better**
   - No warnings in Dynamo, no execution in Generative Design
8. **Dictionaries are your friend**
   - Save time, work with more data, get better answers faster

Links

- **Dynamo Resources**
  - Dynamo Primer: http://primer.dynamobim.org
  - Dynamo Forums: https://forum.dynamobim.com/
  - Dynamo Dictionary: https://dictionary.dynamobim.com

- **Generative Design**
  - Info: https://www.autodesk.com/campaigns/refinery-beta
  - Primer: https://www.generativedesign.org/
  - Beta Site: https://feedback.autodesk.com/key/RefineryLanding

Special Thanks

Jacob would like to thank:

- Sofie for supporting and inspiring me to not hold back
- The Dynamo and Refinery Development teams for providing such amazing tools
- The Dynamo Forum community for teaching me and letting me grow with you.
- My colleagues in Autodesk product support for helping our customers make anything

Alex would like to thank:

- Jacob Small for including me in this great lab/presentation and teaching me the ropes along the way.
- Iv and my family for their help, love, and support in everything I do
- Perkins Eastman and our Design Technology/IT Team for always pushing the boundaries and supporting new research.

And we'd all like to thank J. K. Rowling for bringing so much magic into the world.