### **Session 3.4**

### **Dynamo Master Class While Playing Monopoly**

**Sean Fruin, IMEG**

# A cartoon of a person with a cane Description automatically generated with medium confidence****Class Description****

A qr code with a cartoon character

Description automatically generated with medium confidenceA screenshot of a computer

Description automatically generatedLearning how Dynamo can interact with Revit can be confusing for nonprogrammers, and mastering all Dynamo offers can seem ridiculous. Still, this class has the whole spectrum covered by teaching Dynamo using a topic everyone can understand, the game of Monopoly. This master class covers various topics discussed by the three Dynamo experts. Gian an understanding of critical concepts and invaluable tips and tricks as they walk through the code and share their development and collaborative best practices. These topics include manipulating Revit with Dynamo, creating rotating Revit families that can be controlled with Dynamo, building geometric systems for placing Revit elements, and how to best author code, to name a few. Whether you are a beginner or an expert, this class has practical examples for you. Be sure to stop by and check out this fantastic demonstration of just how far Revit and Dynamo can go and get your hands on the latest version of America’s favourite board game, Monopoly.

# ****About the Speakers:****



Sean Fruin is a Mechanical Engineer and Mechanical Applications Product Owner at IMEG, a full-service engineering firm with over 60 offices throughout the US. He is fascinated with automation and the exploration of computational design solutions for MEP design. He has had the opportunity to learn many aspects of the design industry, working in manufacturing as an MEP designer and consulting for General Contracting around the globe specializing in BIM Management and Autodesk Revit development. Sean is living his dream, playing with the latest technologies, acquiring the knowledge to innovate, improving efficiency, and sharing his insights with the AEC community.

**Sean Fruin**

IMEG



@SeanFruin

[sigma-aec.com](file:///C:\Users\Sean%20Fruin\AppData\Roaming\Microsoft\Word\sigma-aec.com)



He knows modelling, coordination, and improving processes; he’s been doing it for over 20 years. And above all, he knows how to teach and inspire others to adopt BIM/VDC in their own workflows. He loves talking shop about BIM and ConTech; seriously, get him started and he won’t stop! Or, if you’re not in the mood to talk BIM/VDC, Carl is always on the lookout for any excuse to chat about his newest passion: learning to code. Carl doesn’t bite, so reach out to him directly on Twitter @theBIMsider, on his Blog thebimsider.com, or listen to his smooth voice on pretty much any AECO podcast you can find.



@theBIMsider

[thebimsider.com](https://thebimsider.com/)

**CARL STORMS**

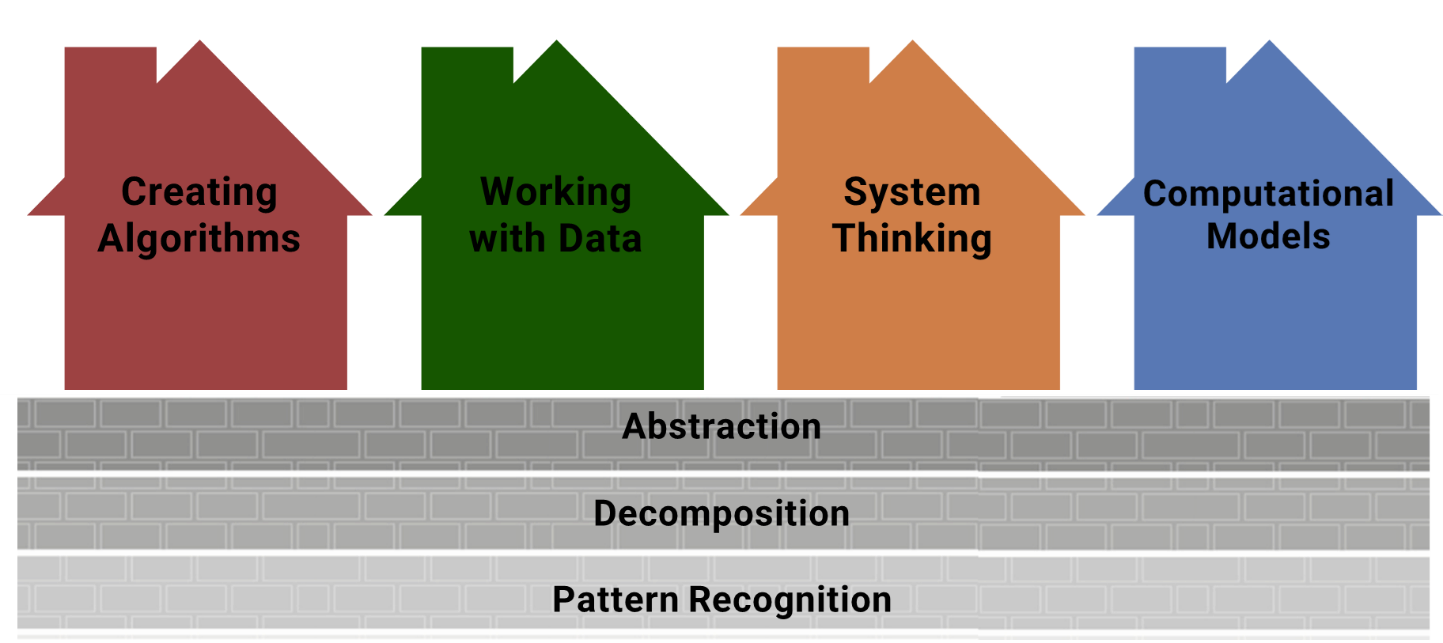
BIM TRACK

# Intro

Mastering the art of coding can be daunting, and I can personally attest to my disdain for programming during my engineering school days. At the time, I failed to see the significance of it all since the problems we were solving seemed like something that could be quickly done by hand. However, upon entering the real world, I quickly realized that programming was essential for eliminating the mundane. If you're someone who didn't take programming seriously in school, learning to program can be a confusing and frustrating experience. But fear not, for I have discovered the keys to mastering the art of programming is setting a clear fun target.

My journey to becoming a proficient programmer started with learning a few Dynamo workflows for transferring data between Excel and Revit and automating Revit project setup. These simple accomplishments served as the spark that ignited my passion for programming. From there, I set myself a lofty challenge - to build the game of Monopoly using Revit and Dynamo. Without clearly defined goals, it's hard to see the point of programming and even harder to derive the satisfaction of accomplishing something. Goals give you something to aim for and achieving them releases a flood of dopamine that keeps you motivated to continue your journey. In other words, if you want to enjoy learning to code, set clear and ambitious goals.

Building a game of Monopoly using Revit and Dynamo can be a fantastic learning experience for those interested in integrating Building Information Modling (BIM) and programming. This is because it sets a clear goal with human language before you script and the ultimate goal is clear.



By combining Revit's visual representation with Dynamo's programming interactivity, you will have the opportunity to explore the use of Dynamo in AEC workflows, automate repetitive tasks, create interactive dialog boxes, and streamline workflows. This hands-on project will allow you to apply your BIM and programming skills practically, providing a fun and engaging learning experience. Additionally, by creating a game of Monopoly, you can explore the integration of BIM and programming, while developing valuable skills highly relevant to AEC professionals. So, whether you're a beginner or an experienced professional, building a game of Monopoly using Revit and Dynamo is an excellent opportunity to improve your skills, have fun, and learn something new.

**Learning Objectives**

* Deconstruct and program the game of Monopoly.
* Discover the key Dynamo functionalities.
* Provide an excellent resource for reference.

# What is Dynamo?

**A picture containing screenshot, design

Description automatically generated**Defining Dynamo can be daunting as it is a versatile program that can be utilized in various ways. The program's vast capabilities, intuitive interface, flexibility, and easy integrations with Revit make it a powerful tool. It's a visual programming tool that can create algorithms and define relationships to generate geometry and process data, but it's so much more than that. It's a canvas for creativity, a portal to endless possibilities, a tool that empowers designers to bring their wildest dreams to life, and it is a community of passionate people. So, what is Dynamo? It's a glimpse into the future of design, a spark of inspiration that ignites the imagination. A formal description is:

**“A visual programming tool that aims to be accessible to both non-programmers and programmers alike. It gives users the ability to visually script behaviour, define custom pieces of logic, and script using various textual programming languages.”**

**What is GitHub?**

GitHub is a web-based hosting service for version control using Git. It is primarily used for code but also supports other types of files. It provides distributed version control and source code management (SCM) functionality of Git, plus its features. GitHub is a crucial tool for many software developers and teams because it allows them to work together and keep track of changes in their projects over time. It also includes access control and collaboration features such as bug tracking, feature requests, task management, and wikis for every project.

GitHub has been used to host all the necessary files required for the Dynopoly, including Dynamo scripts, Dialog box images, and Revit files.

For a deep dive into the code or to play the game, "clone" (download) the repository to your local system and explore the files at your own pace. The GitHub wiki as specific step on the process. Found a bug? You can also make changes and "commit" (save) them to your local repository copy. If you want to share your changes with others, you can "push" (upload) their changes back to the GitHub repository. Similarly, you can "pull" (download) changes made by others from the repository to your local system.

In addition, you can use GitHub's "Issues" feature to report problems or suggest improvements to the files in the repository and the "Pull Requests" feature to propose specific changes to the repository. These features support comments, allowing for conversation before acceptance (or rejection).

A qr code with a cartoon character

Description automatically generated with medium confidence

In this way, GitHub provides not only a place for you to explore and dive deep into the materials of your class but also a platform for collaboration and share your learning journey. The QR code is a link to the url below.

<https://github.com/SigmaAEC/Dynopoly>

**Authoring & Best Practices**

A cartoon of a person with a mustache

Description automatically generated with low confidenceThis section introduces participants to the effective use of Dynamo in their workflows. It covers managing version control using GitHub, creating Dynamo templates for consistent and efficient scripting, annotating graphs for readability and future reference, and mastering new import/export settings to facilitate data management. Best practices aim to enhance the efficiency and reliability of scripts, leading to more productive and error-free results. These tips and tricks are littered about the scripts to make navigating easy.

**Script Development Process**

A picture containing text, screenshot, diagram, graphic design

Description automatically generatedA cartoon of a person running

Description automatically generated with low confidenceThe class navigates the journey from a conceptual idea to a functioning Dynamo script in this section. It includes understanding requirements, designing the script architecture, drafting pseudocode, translating it into Dynamo nodes and connectors, testing, and refining the script. Allow these tips and tricks to help approach problem-solving programmatically and turn their ideas into functional scripts in a systematic manner.

**Knowledge**

Collecting knowledge in Monopoly, much like in Architecture, Engineering, and Construction (AEC), is an evolving process that extends beyond just reading the rules or instructions. First those rules need to be interpreted into logic steps just like building code.

A picture containing text, screenshot, font, businesscard

Description automatically generatedSecond the rules are not static but constantly changing. Just as Monopoly has seen multiple versions, rule changes, and adaptations over time, the AEC industry is perpetually evolving with advancements in technology, new regulations, and changing market demands.

Third, geography plays a significant role in both Monopoly and AEC. Monopoly's properties are based on real-world locations, and the game's strategy can change based on the version of the game you're playing - much like how the practices and regulations in AEC can differ from one location to another. Understanding these geographical nuances can significantly impact your strategy in Monopoly and your approach to AEC projects.

Lastly, personal interpretations and opinions greatly influence both fields. In Monopoly, house rules can significantly change the gameplay. Similarly, in AEC, individual creativity, interpretation of codes, and personal design philosophies greatly impact the final output. Recognizing and navigating these varying opinions is a crucial part of being successful in both Monopoly and the AEC industry.

**Data**

Once the rules of the Monopoly game have been well defined, the next step is to create the database. This involves translating the game's components into Revit elements and setting up the right parameters. Each property, chance card, community chest card, and game piece should be a separate element with its own unique properties.

These properties can be stored as parameters in Revit, and each parameter must have the correct data type, whether it's a text, integer, boolean, or other data type. For instance, a property in Monopoly might have parameters for its name (text), cost (integer), rent (integer), and whether it's currently owned or not (boolean).

In addition to this, you'll need to collect and organize images for each game component. These images could be used to represent the game components visually in your Dynamo script or even within Revit itself.

Once you've created your Revit elements and collected your images, you'll need to think about how to structure your relational database. A relational database is organized into tables, each of which represents a different type of entity in the game. For example, you might have a table for properties, another for chance cards, and another for players. Each row in these tables represents a specific instance of that entity (e.g., a specific property or player), and each column represents a different attribute or parameter of that entity (e.g., the property's name or cost).

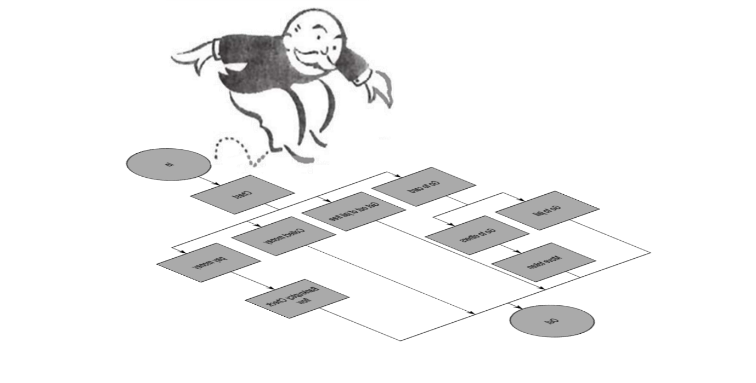
The structure of your relational database should be designed to efficiently store and retrieve the game's data. This might involve creating relationships between different tables. For instance, you might create a relationship between the properties table and the players table to track which properties each player currently owns.

**A screenshot of a computer

Description automatically generated with medium confidence**In all, setting up the database is a crucial step in building your Monopoly game in Dynamo and Revit, as it lays the foundation for how the game's data will be stored, manipulated, and visualized.

**Logic**

Before diving into the coding aspect of the project, it's indeed best to map out the process using flow charts. Flow charts offer a visual way to represent and understand complex processes, making them particularly useful when planning out the coding structure for a game like Monopoly.

Creating a flow chart for the Monopoly game will involve defining the main actions or events that occur during the game, such as rolling the dice, moving a game piece, buying a property, or drawing a Chance card, and then determining the sequence and conditions under which these events occur. Each action can be represented as a node in the flow chart, and the connections between nodes can represent the sequence or conditions.

For instance, you might start with a node for rolling the dice, followed by a node for moving the game piece. From there, different paths could diverge based on the result of the move. If the player lands on an unowned property, one path might lead to the option of buying the property. If the player lands on a Chance space, another path might lead to drawing a Chance card. Each potential event in the game gets mapped out in this manner.

Additionally, flow charts can be beneficial when considering the logic of your script. They allow you to visualize how data will flow and be manipulated, helping to identify potential challenges or inefficiencies in your script's design.

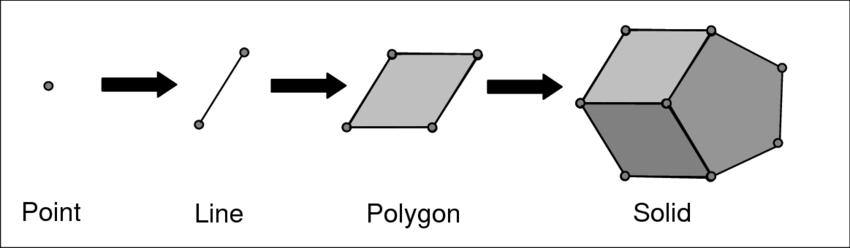
Once your flow chart is complete, you'll have a clear roadmap for writing your Dynamo script. This structured approach to planning helps to ensure that the coding process is more organized, efficient, and less prone to errors. It's a method that emphasizes the importance of strategic planning in scripting and computational design, promoting a more thoughtful and efficient approach to problem-solving.

**Computational Geometry**

A cartoon of a person holding a kite

Description automatically generated with low confidenceFocusing on the creation of 3D objects in Dynamo and their representation in Revit, this part of the course explores the principles of computational geometry. Dynamo scripts are available to create the game board, house geometries, and shuffle community chest cards. We also cover essential geometry functions for randomizing dice numbers and locations and placing tokens and houses on property blocks. This hands-on approach ensures attendees understand the intricacies of geometry creation and manipulation in Dynamo and Revit.

In the process of developing a Monopoly game using Dynamo and Revit, several key concepts in computational design come into play, providing practical applications of these principles.

1. Building Geometry: Each property on the Monopoly board can be represented as a 3D object or building in Revit. For example, you might model each property as a small building, using different parameters to represent the property's characteristics, such as color or height. The process of creating these geometries computationally involves defining the shape, size, and location of each object, which can be controlled using Dynamo scripts.
2. Rotating Geometry: In Monopoly, the game board is a square with properties laid out along its edges. This means that the property 'buildings' need to be rotated to face inwards towards the center of the board. This involves rotating each geometry about a specific axis, a key concept in computational design. Dynamo provides nodes for rotating geometries, allowing you to define the angle and axis of rotation.
3. Vectors: Vectors play a crucial role in defining the position and orientation of objects in computational design. In the context of Monopoly, vectors can be used to determine the location of each property along the game board, to define the direction that property 'buildings' face, and even to move the game pieces along the board. By controlling the magnitude and direction of these vectors, you can precisely control the layout and movement within the game.

By understanding and implementing these computational design concepts, you can create a detailed and interactive 3D representation of a Monopoly game. It serves as a fun and engaging application of computational design principles that are directly relevant to tasks in the AEC industry, from designing building layouts to scripting design modifications.

**Dynamo and Revit**

While Dynamo is a flexible environment, designed to port into a wide range of programs, it was originally created for use with Revit. Thanks Ian. A visual program creates robust options for a Building Information Model (BIM). Dynamo offers a whole suite of nodes specifically designed for Revit, as well as third-party libraries from a thriving AEC community.

A hand holding a puppet

Description automatically generated with low confidenceThis section delves deeper into the integration of Dynamo with Revit, focusing on methods for creating, selecting, and manipulating Revit elements using Dynamo. By showcasing the symbiotic relationship between Dynamo and Revit, the course highlights how the visual programming language can streamline and automate many tasks in Revit, leading to more efficient and accurate modeling processes.

There are a range of methods for selecting elements in Revit The monopoly scripts uses a bunch of different ones. Once selected we can query the elements by their parameters a filter the list down looking out for those error like mentioned above or grab a spicific token based on a conditional like player turn. All so geometry from Revit can be dynamically link to Dynamo geometry to preforem those parametric operations listed in the computational section.

A powerful feature of Dynamo is that you can edit parameters on a parametric level. For example, a generative algorithm or the results of a simulation can be used to drive the parameters of an array of elements. In this way we can up date the Revit model after each run of a Dynamo script to relex the changes like token location, money and setting property ownership.

You can create an array of Revit elements in Dynamo with full parametric control. The Revit nodes in Dynamo offer the ability to import elements from generic geometries to specific category types (like walls and floors), add schedules and custom filters to reflex the new player at the start of a game. components.



**UX/UI with Dynamo:**

A drawing of a person holding a bouquet

Description automatically generated with medium confidenceHere, the course explores the user experience and user interface design capabilities of Dynamo. Attendees will learn to create interactive dialog boxes and other interface elements to enhance user interactions with their scripts. They'll understand how to make their hands more user-friendly, providing a smoother user journey and more effective toolsets for the end-users of their writings.

**Data Analytics:**

A cartoon of a person with a hat and a computer

Description automatically generated with low confidenceIn this final module, the class will discover how to leverage data analytics in Dynamo, including data export to Excel and CSV, game logging, and Power BI game dashboard creation. Participants will learn how to collect, analyze, and visualize data from their scripts, deepening their understanding of the game mechanics, player behavior, and other aspects. It will

In conclusion, this Masterclass offers a unique and engaging approach to learning and teaching Dynamo and Revit, integrating the concepts of Building Information Modeling (BIM) and programming through the fun lens of building a game of Monopoly.

We've covered several fundamental aspects of Autodesk's Dynamo and Revit, beginning with best practices for using Dynamo. We started with Dynamo's best practices to optimize your scripts and enhance collaboration. Next, we demystified the process of transforming ideas into functional scripts. We practiced creating and manipulating 3D objects in Dynamo and Revit, highlighting the close integration of the two. We also focused on user interface and experience, emphasizing the need for user-friendly scripts. Finally, we explored pully data out of Revit for data analytics, showing how it can provide valuable insights about project data.

The skills learned in this Masterclass extend far beyond creating a game of Monopoly; they are directly applicable to real-world AEC projects. For instance, Dynamo's best practices can help manage complex scripts, make them easier to share and ensure fewer errors. Understanding the script development process equips you to automate repetitive tasks, improving accuracy and saving valuable time. Computational geometry skills are vital in AEC automation and building geometry systems like the one used here will be an ever more valuable skill. Integrating Dynamo and Revit enhances the manipulation of BIM elements, fostering better design control and management. The emphasis on UX/UI design can be used in creating intuitive interfaces for custom scripts or plugins, easing the adoption of these tools among team members. Lastly, data analytics skills are essential in tracking project progress, optimizing design elements, and informing better decision-making through actionable insights. These skills contribute to a more streamlined, collaborative, and efficient approach to handling AEC projects, ultimately leading to superior outcomes.

As you dive deeper into these topics, remember that all the necessary files, including Dynamo scripts, Dialog box images, and Revit files, are available in our GitHub repository. Feel free to clone, explore, modify, and share these resources. GitHub offers a platform for further collaboration, discussion, and learning, enabling you to get the most out of this Masterclass.

By creating a game of Monopoly using Revit and Dynamo, we hope we improved your technical and creative problem-solving skills and, hopefully, had fun along the way. This experience underlines the power and potential of programming in the AEC industry, promising exciting opportunities for future exploration and innovation. Feel free to pass the topic and materials along to help the industry to think more computationally.

# Helpful Links

<https://primer.dynamobim.org/>

<https://forum.dynamobim.com/>

<https://www.sigma-aec.com/post/what-is-dynamo>

<https://github.com/SigmaAEC/Dynopoly>