AUTODESK

BIM GUIDEBOOK 2

Dynamo Part Ⅰ

BIM GUIDE BOOK 2

Dynamo Part Ⅰ

실무자에 의한

실무자를 위한

FREE

Upload & Download

"I wish to express appreciation to many colleagues at our Autodesk Korea the customers who have provided valuable feedback for putting together the 2nd release of the BIM Guidebook: Dynamo Handbook.

Since its collaborative release with Junglim Architecture in December 2017, over 35,000 customers have visited the BIM Guidebook homepage and over 5,000 customers have downloaded the BIM guidebook.

This was a great indication for the demand of creating a second guidebook catered to Dynamo. We would like for our customers to easily grasp Dynamo and use it efficiently in their design process since Autodesk is committed to being a great "customer company", and our AEC Team worked hard at improving the automation program to reduce time and documenting procedure. Also, the Dynamo 2D automation program and Dynamo Handbook are available to all customers free of charge and will be used by other countries across the pan Asia-Pacific.

In the very near future, digital technologies influenced by BIM will change the construct of our everyday lifestyle beyond our own imagination and I am fully determined that Autodesk will be in the leading forefront of this great evolution. Since noted publications across all spectrum have indicated that implementations of our transforming digital processes are changing rapidly on a global scale it is key to note that BIM is the centerpiece of it all and will play a significant role for innovating with other digital technologies.

I strongly expect that the Dynamo Handbook will contribute to the world class use of BIM for all Korean customers who long to embrace the capabilities and reap the benefits of BIM in their design, construction and facility management processes.

Once again, a big gratitude to Autodesk Korea for making this all possible!"

Patrick Williams

Senior Vice President, Autodesk Asia Pacific

Architecture

국내에 BIM이 소개되고 채택되어 나아가기 시작한 지 여러 해가 되었으나 아직도 설계, 엔지니어링 및 건설 산업 현장에서는 활용의 어려움을 겪고 있다는 피드백을 고객들로부터 적지 않게 받고 있습니다. 특히, BIM을 통해 전반적인 업무의 효율성을 높이면서도 새로운 차원의 프로세스 관리와 인사이트를 찾고자 하는 목적에 반하여 현업을 담당하는 실무자들의 BIM 경험이 풍부하지 못한 현실에서, 어떻게 하면 이 간극을 좁혀 해결할지 고민한 산물이 이 BIM 가이드북 2차 본이 되겠습니다.

특히, 바로 업무에 적용하여 실질적인 도움을 줄 수 있는 소프트웨어 툴 활용 방안이 추가되어 기존의 1차 본에서 제공한 가치 이상의 도움을 드릴 수 있으리라 봅니다.

지속적으로 변화 발전할 설계, 엔지니어링 및 건설 산업이 생산성과 부가가치를 더욱 증대하여 경쟁력을 제고하는데 미력이나마 보탬이 되기를 진심으로 희망하며 많은 활용을 부탁드립니다

최기영
Autodesk Korea 대표이사

It's extremely pleasing to see the progress and adoption of BIM in the Korean AEC industry and the opportunity that is presents to deliver benefits in design, productivity, sustainability and ultimately better outcomes for the people of Korea.

I'm proud of the part that Autodesk has played in this advancement of BIM and hope that with the release of this BIM Guidebook 2 we can further advance the knowledge, skills and capabilities of Korean architects, designers and engineers.

Andrew Cunningham
Autodesk APAC AEC

2016년 12월 1일, Autodesk와 정림건축이 공동으로 제작한 'Autodesk & 정림건축 BIM 가이드북'이 공개된 바 있습니다. 지난 1년간 5천여명이 넘는 건축 실무자들이 BIM 가이드북을 다운로드하였으며, 이를 통해 건설산업에서 BIM 도입 및 활용하려는 의지를 엿볼 수 있었습니다. 'Autodesk & 정림건축 BIM 가이드북'은 국내 대형 건축설계사무소와는 달리 중소규모 설계사무소의 BIM 도입에 대한 어려움을 지원하기 위해 제작되었습니다. 또한, BIM 가이드북을 통해 BIM 모델러와 건축설계자가 이원화되지 않고, 건축설계자가 직접 BIM 프로세스를 이용하여 설계가 가능하게 되었습니다.

'Autodesk & 정림건축 BIM 가이드북'은 총 3가지 파트로 나뉘어져 있습니다.

첫 번째, 모델링파트는 BIM 모델 작업시 조달청 시설사업 BIM 적용지침서에 부합하는 정보들을 모델에 기입하고, 모델 정보의 의미를 설명하였습니다. 두 번째, 도면화파트는 총 14개의 도면을 BIM 모델에서 추출하는 방법과 2D 주석을 표현하는 방법에 대해 설명하였습니다. 또한, 납품도면 작성 시 2D 도면에 포함되어야 하는 필수 건축요소를 설명하고 있습니다. 세 번째, 성과품 관리파트는 조달청 시설사업 BIM 적용지침서를 바탕으로 건축설계사무소에서 준비해야 하는 모델링요소, 파일체계 그리고 부재별 정보를 설명하였습니다.

오는 2018년 1월 10일, BIM 모델링작업 후 반복되는 2D 주석 작업을 단축시킬 수 있는Dynamo Code 가 담긴 'Autodesk BIM 가이드북2' 가 공개됩니다. Dynamo는 BIM 모델링프로그램인 Autodesk의 Revit에서 사용할 수 있는 무료 소프트웨어로서, 사용자가 직접 BIM 데이터를 가공할 수 있는 Open Programing 소프트웨어입니다. 'Autodesk BIM 가이드북2'에서 제안하는 Dynamo Code는 건축 설계실무자가 2D 도면화작업을 위해 소비되는 시간을 줄일 수 있고, BIM 모델에 기입된 정보를 보다 쉽고 효율적으로 활용할 수 있습니다. 'Autodesk BIM 가이드북2'에 소개된 Dynamo Code는 총 16개 이며, 3개의 도면파트로 분류하였습니다.

첫 번째, 일반도면 작성파트는 도면 작성 시 사용되는 표를 자동으로 그려주는 "Auto Table layout", 평/입/단면에서 Grid와 Level에 따라 자동으로 치수를 넣어주는 "Auto Dimension" 그리고, 도면의 Layout을 자동으로 배치하는 "Auto Sheet"를 활용하는 방법을 설명하고 있습니다. 두 번째, 건축도면 작성파트는 BIM 모델을 기반으로 단열/방수/방화구역을 표현해주는 "Auto Filled Region", BIM 모델에 표현된 창호를 기반으로 일람표를 작성하는 "Auto 창호일람표", BIM 모델 정보를 활용한 "Auto 재료명 작성", 모델정보를 가공한 "Auto 실내 재료 마감표"를 자동으로 작성하는 방법을 담고 있습니다. 세 번째, 구조도면 작성파트는 모델링된 구조 슬라브에 따른 "Auto 슬라브 범례 작성", 구조도면 작성을 위한 "Auto 슬라브 해치", 슬라브 높이에 따른 "Auto Level Tag"를 활용 방법을 제안하였습니다. 총 16개의 Dynamo Code는 10여 가지의 건축도면의 2D 요소를 자동화할 수 있는 방법에 대해 설명하고 있습니다. 'Autodesk BIM 가이드북2'는 16개의 Dynamo Code 생성뿐만 아니라, 이미 작성된 Dynamo Code를 활용하는 방법에 대한 내용 또한 담겨져 있습니다. 공개되는 16개의 Dynamo Code를 활용한 주석 자동화방식은 건축 설계실무자의 작업효율을 높여줄 것으로 기대합니다.

'Autodesk BIM 가이드북' 시리즈를 통하여 BIM을 활용하는 실무자의 증가, 국내 건설산업의 BIM 도입의 확산과 도면화작업에 어려움을 겪는 건축실무자에게 작은 도움이 되길 바라며, BIM을 통한 기술적 발전과 프로세스 개선에 도움이 되기를 기대합니다. 국내 건설시장의 BIM 확산과 건설 실무자를 위해 Autodesk는 연속되는 BIM 가이드북을 만들도록 노력할 것입니다.

2018.01.25

Autodesk BIM GUIDEBOOK2 제작팀 일동

C O N T E N T S

1 Dynamo 12

1.01 What is Dynamo? 14
1.02 설치 및 실행 하기 16
1.03 사용자 인터페이스 18
1.04 작업공간 22
1.05 시작하기 26
1.06 Nodes & Wires & Library 32
1.07 Package Node 40

2 Dynamo Code 44

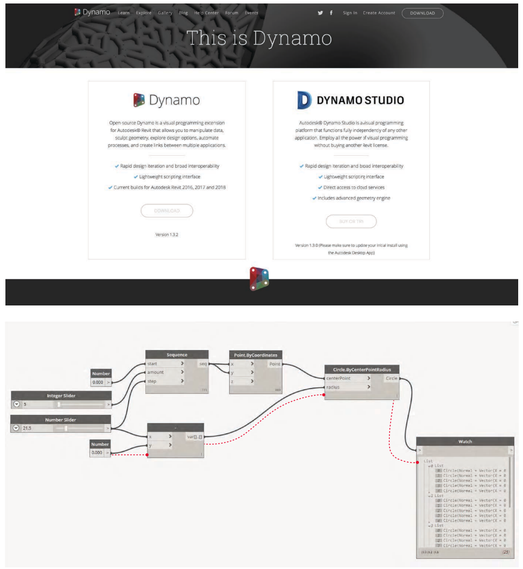
2.G1 Tablelayout 만들기 46
2.G2 Auto Dimension (평면) 54
Auto Dimension (입단면) 64
2.G3 Sheet에 배치된 View 크기 맞추기 76
2.G4 Sheet에 배치된 다중 View Layout 맞추기 86
2.A1 단열/방수/방화 Filled Region 만들기 100
2.A2 창호 일람표 만들기 110
2.A3 부재별 재료 정보 표기 124
2.A4 부재별 재료 Tag 배치 134
2.A5 실내 재료 마감표 144
바닥 146
벽 158
천장 166
2.S1 Slab Level 기준 범례 만들기 174
2.S2 Slab Level 범례기준 해치 패턴 입히기 184
2.S3 Spot Elevation Tag 배치하기 192
2.S4 Slab Level Gap Tag 배치 200

3 Custom Node 214

3.01 Custom Node 소개 216
3.02 Custom Node 만들기 218
3.03 Custom Node 공유 220

1. Dynamo

1.01 What is Dynamo?



The Application

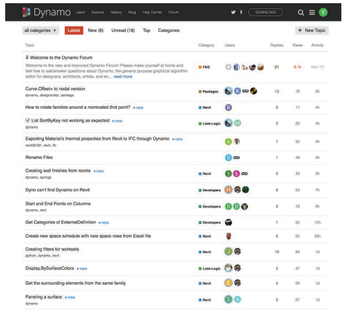
Dynamo는 www.dynamobim.org에서 다운로드하여 Revit에서 플러그인으로 실행할 수 있는 소프트웨어이다. 독립형 소프트웨어로 실행하려면 Dynamo Studio를 다운로드하면 된다. Dynamo는 Revit 이외에도 Maya와 Fusion360 등 Autodesk 프로그램과 연계하여 사용할 수 있다. Dynamo는 프로그래머가 아니어도 사용할 수 있는 비주얼 프로그래밍 툴이다. 비주얼 프로그래밍은 다양한 텍스트 프로그램 언어를 사용하여 프로그래밍의 동작을 시각적으로 확인할 수 있고, 사용자가 원하는 스크립트를 작성할 수도 있다.

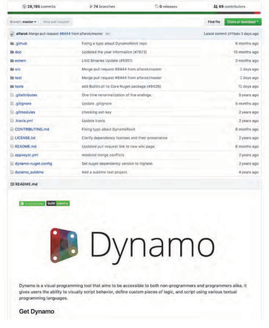
1. Revit의 플러그인 Dynamo 다운로드
2. 독립형 Dynamo Studio 다운로드

The Process

프로그램을 설치하면, Revit 안에서 Dynamo를 통해 비주얼 프로그래밍 프로세스를 작업할 수 있다. Dynamo에서 제공되는 node를 통하여 사용자가 원하는 알고리즘을 구성할 수 있으며, 알고리즘을 통한 Revit의 동작을 정의할 수도 있다.
사용자가 구성한 알고리즘을 통하여, 데이터 처리에서 Geometry 생성까지, Revit의 동작을 실시간 제어할 수 있다.

- Node를 추가하고, 연결하여, 비주얼 프로그램을 만들어 사용할 수 있다.





The Community

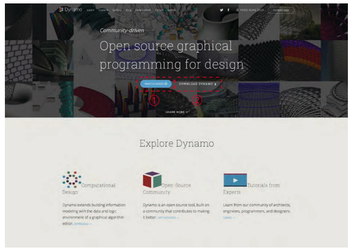
www.dynamobim.org 홈페이지에서 Dynamo를 사용하는 사용자들과 대화하고, 토론할 수 있다. Dynamo 공식 홈페이지는 영문으로 운영되어, 한국 사용자들을 위해 www.bimguidebook.co.kr 홈페이지 개설하였다. www.bimguidebook.co.kr의 Forum에서도 국내 사용자들과 대화하고, 토론할 수 있다.

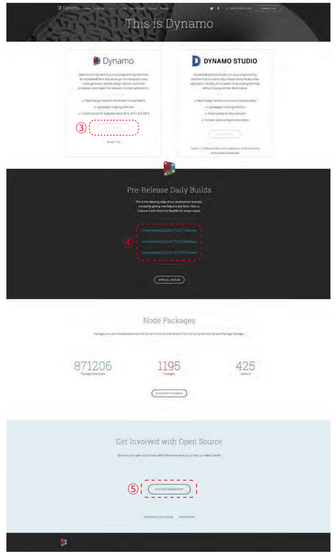
The Platform

Revit 안에서 구동되는 Dynamo는 Revit 사용자를 위한 시작적 프로그래밍 도구로 설계되었다. 대부분 제공되는 node 및 소스 코드는 오픈 소스이기 때문에 사용자의 알고리즘에 의해 Dynamo 콘텐츠를 확장될 수 있다. Dynamo로 만들어진 알고리즘은 사용자 API로 개발할 수도 있다.
제공되는 Dynamo node 이외에도 외부 라이브러리를 사용할 수 있도록 만들어졌다. 외부 라이브러리는 Package를 검색해서 다운로드할 수 있다. 또한 Github을 통해서 소스 코드의 업데이트와 문제점을 질의할 수 있다.

- 사용자에 의한 소스 코드 찾기, 사용자의 소스 코드를 통한 Dynamo의 확장

1.02 설치 및 실행 하기

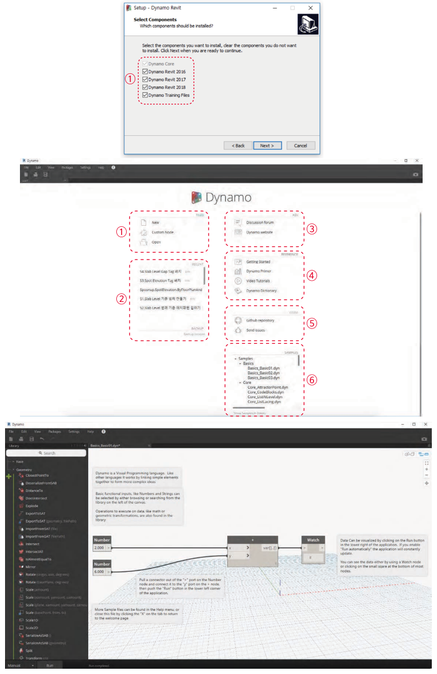




Downloading

공식 버전과 업데이트 버전의 Dynamo를 다운로드하려면, www.dynamobim.org에 방문하여 다운로드할 수 있다. Revit 2017 버전부터는 Revit 설치 시 자동으로 설치되지만, 업데이트 버전은 따로 설치하여야 한다.

1. Dynamo Sample 영상 보기
2. 다운로드 페이지 이동
3. 공식 설치 프로그램 다운로드
4. Daily 업데이트 버전 다운로드
5. Dynamo 개발을 위한 Github



Installing

다운로드한 설치 프로그램을 찾아서 파일을 실행한다. 설치 프로세스 중에 설치할 Revit의 버전을 사용자 정의할 수 있다

1. 설치할 Revit 버전 선택

Launching

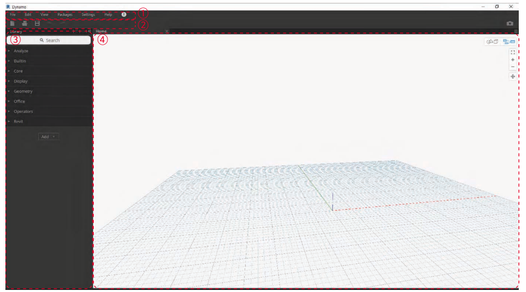
Revit 안의 Dynamo를 실행하려면, Revit의 Manage 탭 〉 Visual Programming 〉Dynamo를 클릭하여 실행한다. Dynamo의 시작 페이지에서 파일 열기 이외에도 Ask,Reference, Code, Sample 파일에 접근할 수 있다.

1. Files : 새 파일을 시작하거나 기존 파일을 선택
2. Recent : 최근 파일을 선택
3. Ask: 사용자 포럼 또는 Dynamo 웹사이트에 액세스
4. Reference: 추가 학습 자료 참조
5. Code: 오픈 소스 개발 프로젝트에 참여
6. Samples: Dynamo 예제 파일

첫번째 샘플 파일을 열고 Dynamo가 올바르게 작동하는지 확인 한다.
Samples 〉 Basics 〉Basics_Basic01.dyn.

- 하단의 실행 표시줄에 “Automatic” 또는 ”Manual”로 설정 되어있는지 확인 후 “Manual”로 설정 되어 있다면, “Run”을 클릭 하여 실행 한다.
- 두번째 Number node의 Output을 클릭하고, + node의 y에 연결 한다.
- “Run”을 클릭 한 뒤, Watch node에서 결과 값이 표시 되는지 확인 한다.

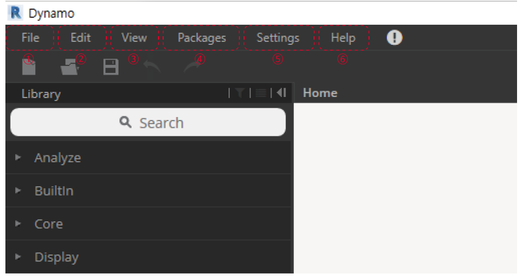
1.03 사용자 인터페이스



The Dynamo User Interface

Dynamo의 사용자 인터페이스 (UI)는 5개의 주요 영역으로 구성된다.

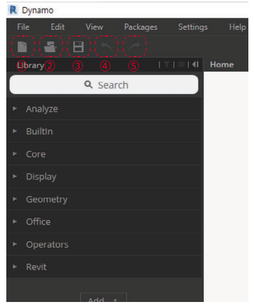
1. 메뉴
2. 툴바
3. 라이브러리
4. 작업 공간
5. 실행 바

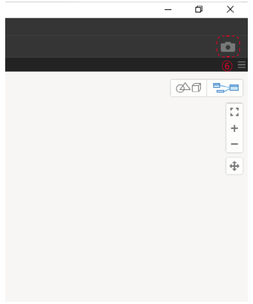


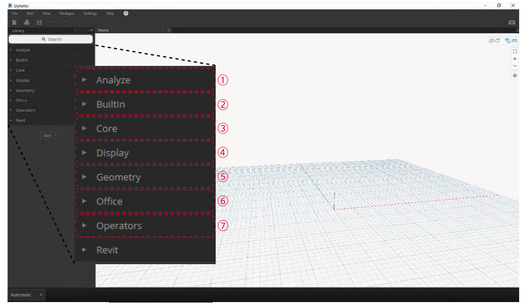
The Dynamo User Interface_ Menus

메뉴를 드롭 다운 하면 대부분의 소프트웨어처럼 파일 관리, 선택 및 편집 작업의 메뉴를 찾을 수 있다.

1. File
2. Edit
3. View
4. Packages
5. Settings
6. Help







The Dynamo User Interface_ Toolbar

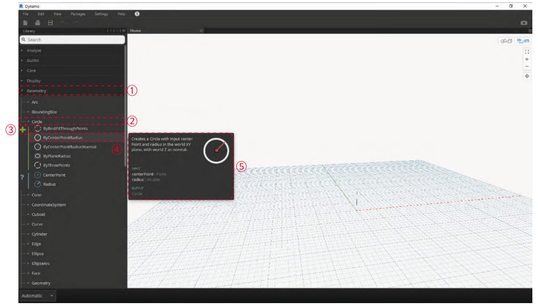
Dynamo의 툴바는 파일 작업에 빠르게 액세스 할 수 있는 Quick 버튼과 작업 실행 취소와 다시 실행 버튼이 구성되어 있다. 맨 오른쪽 버튼으로 작업 공간을 스냅샷으로 내보내어, 작업 공간을 문서화 및 공유할 수 있다.

1. New : .dyn 파일을 새로 작성
2. Open : 기존 .dyn (작업 영역) 또는 .dyf (사용자 정의 노드) 파일 열기
3. Save/Save as : 활성 .dyn 또는 .dyf 파일 저장
4. Undo : 마지막 작업 취소
5. Redo : 다음 작업 실행
6. Export Workspace as Image : 보이는 작업 영역을 PNG 파일로 내보내기

The Dynamo User Interface_ Library

Library에는 Dynamo 설치와 함께 제공되는 기본 Node와 사용자가 설치한 Node를 포함한 모든 Node가 포함된다.
Node들은 Library의 위계에 의해 Category와 정의된 Node 특성에 맞게 List 형식으로 정리된다.

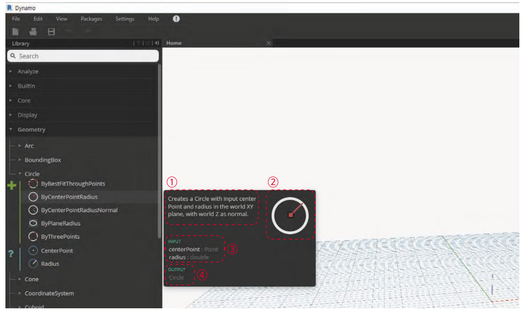
1. Analyze
2. Built-in Functions
3. Core
4. Geometry
5. Migration
6. Office
7. Operators



The Dynamo User Interface_ Library

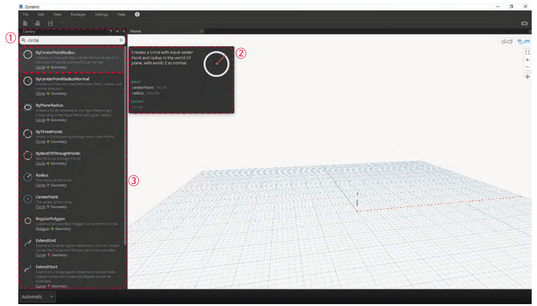
Library의 List에서 Geometry 〉 Circle의 List를 클릭한다.

1. Library
2. Category
3. Subcategory: Create/Actions/Query
4. Node
5. Node 설명 및 특성 : Node 아이콘 위로 마우스를 가져가면 표시 된다.



Circle List의 ByCenterPointRadius Node 위에 마우스를 올리고, Node의 정보를 확인한다.
Node의 정보에는 Node 이름, 기능, 입력 데이터, 출력 데이터를 담고 있다.

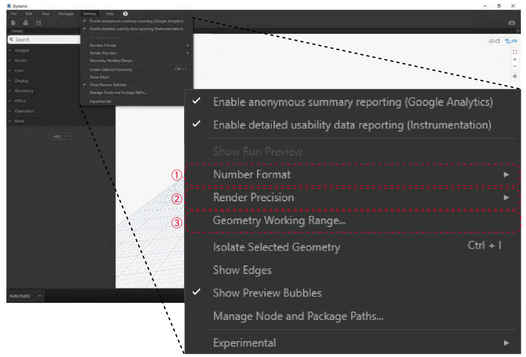
1. Description : Node의 설명
2. Icon
3. Input(s) : 이름, 데이터 유형 및 구조
4. Output(s) : 데이터 유형 및 구조



The Dynamo User Interface_ Searching

Node를 선택하여, 작업 공간에 배치할 경우 검색을 이용하면 빠르게 Node를 배치할 수 있다.

1. 검색 창
2. 검색 결과 및 선택
3. 검색 결과 목록

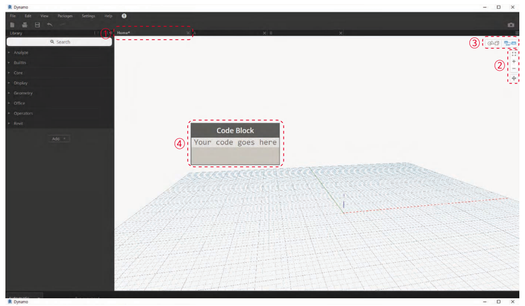


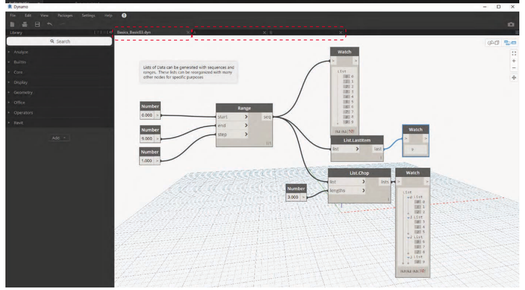
The Dynamo User Interface_ Setting

작업 공간 설정에서 사용자 설정까지 Dynamo에서 가능한 설정 메뉴를 찾을 수 있다.
Dynamo 설정을 통하여, 프로그램의 연산 정밀도 및 렌더링 품질을 높일 수 있다.

1. Number Format : Dynamo에서 표현되는 단위 설정
2. Render Precision : Dynamo에서 보이는 형상의 정밀도
3. Geometry Working Range : Dynamo에서 작업할 형상의 크기

1.04 작업공간





The Workspace

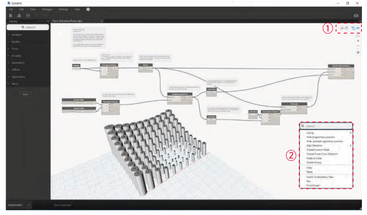
Dynamo의 화면은 기본적으로 비주얼 프로그래밍을 작성 하지만, 프로그래밍에 따른 형태도 미리 볼 수 있다. 오른쪽 상단의 모드를 클릭하여 작업공간의 특성을 전환하여 형태를 확인 한다. 또는 3D Watch Node에서 쉽게 형태를 확인 할 수 있다.

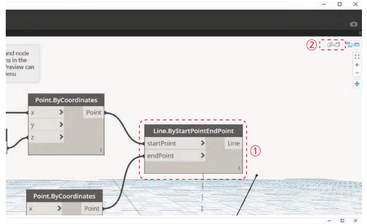
1. Tabs
2. Zoom/Pan Buttons
3. Preview Mode
4. Double Clicking on the Workspace

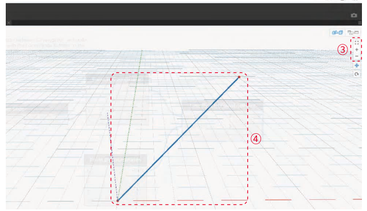
Tabs

활성화된 작업 공간의 탭을 사용하면, 열려있는 .dyn과 .dyf를 편집할 수 있다. 새 파일을 열면 기본적으로 새로운 작업 공간을 볼 수 있으며, 사용자 Node를 만든 작업 공간도 열 수 있다.

- 한 번에 하나의 .dyn파일을 열 수 있다. 그러나 사용자 Node인 .dyf는 여러 개의 파일을 열 수 있다.







Graph versus 3D Preview Navigation

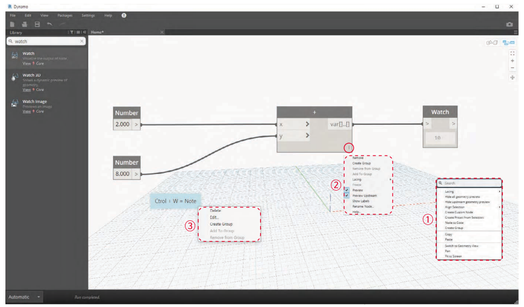
Dynamo에서 작성된 Geometry는 작업 공간에서 렌더링 되어 확인할 수 있다. 전환된 작업 공간에서 탐색 버튼 또는 마우스의 가운데 버튼을 사용하여 이동, 확대/축소할 수 있다.

1. 작업공간 전환
2. 마우스 오른쪽 클릭 후 작업공간 전화 선택
3. 키보드 단축키 : Ctrl +B

Zoom to Recente

전환된 작업 공간인 3D 미리 보기 모드에서 모델을 자유롭게 이동, 확대/축소할 수 있다. 특정 Geometry를 확대하려면, Geometry를 생성한 Node를 선택 한 상태에서 확대 아이콘을 사용하여 Geometry를 확대할 수 있다.

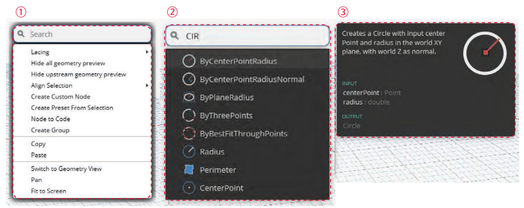
1. 작업 공간 중앙에 배치할 지오메트리를 생성한 Node 선택
2. 작업 공간 전환
3. 오른쪽 상단의 확대/축소 아이콘 클릭
4. 선택한 지오메트리 확대



Hello Mouse!

어떤 작업 공간 모드가 활성화되어 있는지에 따라서 마우스 버튼이 다르게 작동한다.
일반적으로 마우스 왼쪽 버튼을 클릭하면 Node를 선택할 수 있다.
마우스 오른쪽 버튼을 클릭하면 옵션에 액세스할 수 있고, 마우스 가운데 버튼을 클릭하면 작업 영역을 탐색할 수 있다. Node 위에서 마우스 오른쪽 버튼을 클릭하면 Node를 설정할 수 있다.

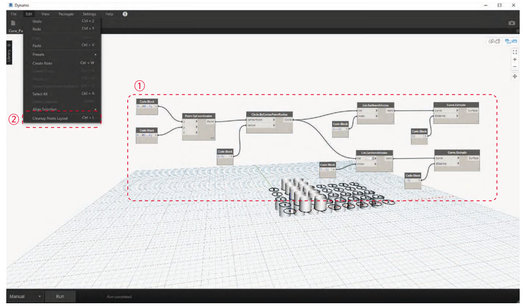
1. 작업 공간 위에서 마우스 오른쪽 클릭
2. Node 위에서 마우스 오른쪽 클릭
3. Note 위에서 마우스 오른쪽 클릭

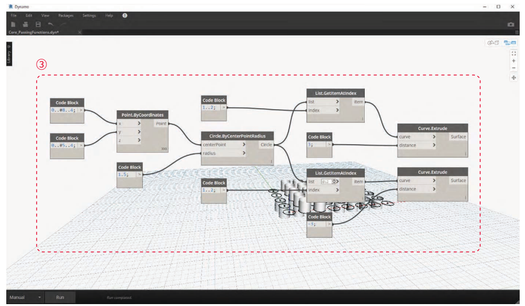


In-Canvas Search

마우스 오른쪽을 클릭하면 볼 수 있는 검색 창을 "In-Canvas Search"라고 한다.
"In-Canvas Search"는 작업 공간에서 사용할 Node를 검색할 수 있다.

1. 작업 공간에서 마우스 오른쪽 버튼으로 클릭하면 검색 기능 확인.
2. 검색 창에 입력할 때 드롭 다운 메뉴가 가장 관련성이 높은 검색 결과가 표시
3. 검색 결과 위로 마우스를 이동하여 해당 설명과 도구 설명 표시 확인





Clean Up Node Layout

Dynamo에서 비주얼 프로그래밍이 진행될수록 작업 공간에 많은 Node들이 배열되어 복잡한 모습을 나타나게 된다. Node를 효율적으로 배치하고, 작업 내용을 쉽게 파악하기 위해서는 작업 공간의 정리가 중요하다. Align Selection 도구를 이용하여 선택된 Node들을 일정하게 정렬하고, Clean up 도구를 활용하여 Node의 배열을 손쉽게 정리할 수 있다.

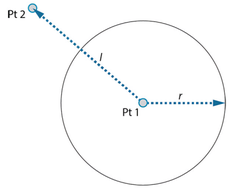
Clean up 실행 전

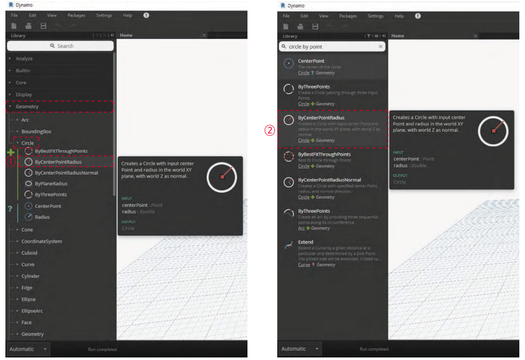
1. 정리한 Node를 선택할 경우 선택된 Node만 정리되고, 전체 Node 정리시에는 노드를 선택하지 않은 상태에서 Clean up을 실행한다.
2. Edit) Cleanup Node Layout (단축키 : Ctrl + L)

Clean up 실행 후

3. Node는 자동으로 재 배치되고, 인접한 Node와 정렬한다. 겹치는 Node나 엇갈린 Node는 자동으로 정리된다.

1.05 시작하기





Defining Objectives and Relationships

Dynamo에서 Node를 추가하기 전에 어떤 작업을 하고자 하는지 확실하게 설정하는 것이 중요하다. 형태를 만들 것인지, Revit의 Parameter Data를 가공할 것인지를 고려하여 Node를 배치한다. Dynamo에서 사용되는 Node들을 연결할 때마다 가공되는 Data는 달라지게 된다. 때문에 Node를 연결하는 Node 간의 관계를 잘 고려하여, 연결해야 한다.

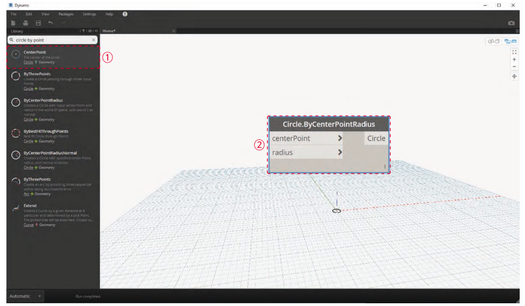
1. 목표: 원을 만든다.
2. 조건: 반경을 입력하고 반경에서 가장 가까운 거리 값으로 원을 만든다.

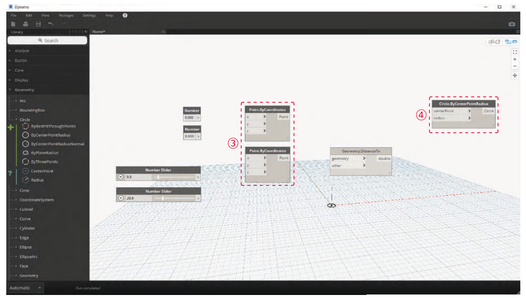
위와 같은 목표와 조건을 설정하고, Dynamo에서 형태를 구축한다.

Adding Nodes to the Workspace

위에서 설정한 목표와 조건을 통해 Dynamo에서 지오메트리를 만든다.
첫 번째로 원을 만들 수 있는 Node를 필드에서 검색하거나, Library 리스트에서 선택한다

1. Browse to Geometry 〉 Circle 〉 Circle.ByPointRadius
2. Search 〉 "Circle by Point..."





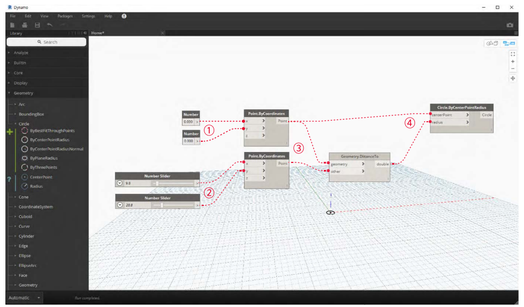
Adding Nodes to the Workspace

Library에서 [Circle.ByPointRadius] Node를 클릭하여 작업 영역에 추가한다.
Node는 작업 영역의 중심에 추가 된다.

1. Library 리스트에서 [Circle.ByPointandRadius] Node 확인
2. 선택한 Node 클릭하여 작업 영역에 추가

설정한 조건에 따라 원의 반경을 정할 수 있도록 Input 할 Node를 선택한다.
([Point.ByCoordinates], [Number and Number Slider Nodes])

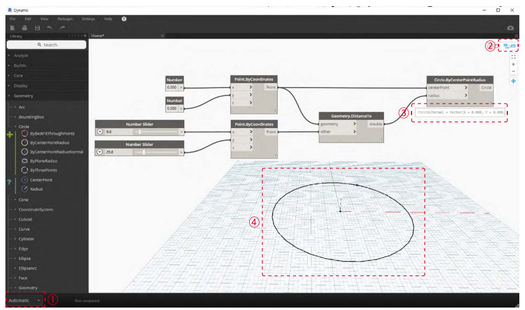
3. Geometry 〉 Point 〉 Point.ByCoordinates
4. Geometry



Connecting Nodes with Wires

작업 영역에 Node를 배치하였다면, Node의 포트를 와이어를 통해 연결한다.

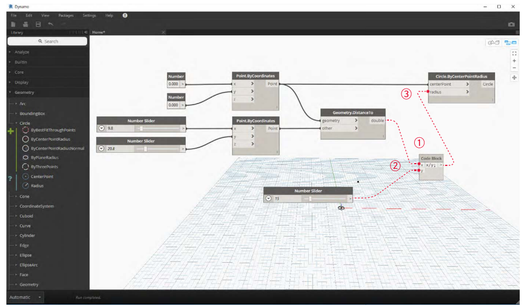
1. Number의 Output 포트를 [Point.ByCoordinates]의 Input 포트에 연결한다.
2. Number Sliders 의 Output 포트를 [Point.ByCoordinates] 의 Input 포트에 연결한다.
3. [Point.ByCoordinates] (2) 의 Output 포트를 Distance 의 Input 포트에 연결한다.
4. [Point.ByCoordinates] 와 DistanceTo 의 Output 포트를 [Circle.ByCenterPointRadius] 의 Input 포트에 연결한다.



Executing the Program

Node들의 연결의 통해 Dynamo 작업 흐름을 정의 한 뒤, 왼쪽 하단의 Run 버튼을 클릭하여, Dynamo를 실행한다. 완료된 Data들을 작업 공간에서 확인한다.

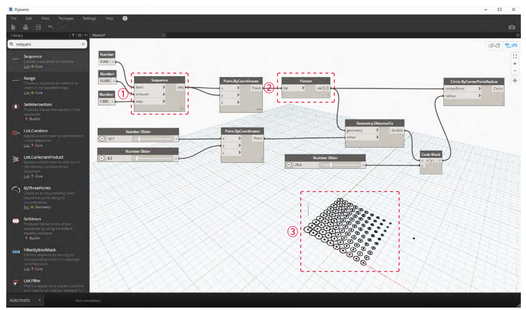
1. Run 클릭
2. Node 미리 보기 : Node의 오른쪽 하단에 있는 상자 위로 마우스를 가져가면 나타나는 Data 값 확인
3. 3D 미리 보기 : Node를 이용해 형상을 만들어 3D watch Node 또는 작업 공간을 전환하여 3D 미리 보기에서 확인



Adding Detail

Dynamo를 통해 작성된 원 Geometry에 세부 사항을 추가하여 컨트롤할 수 있다.
원의 형태를 교정할 수 있고, Input 값에 따라 형태가 변화되게 할 수도 있다.

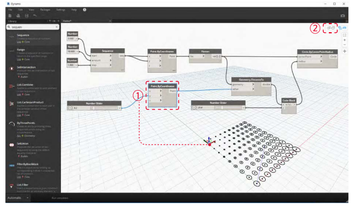
1. Number Slider와 Code Block Node를 추가 후 Code Block Node에 "X/Y" 수식 입력
2. DistanceTo 와 Number Slider 의 Output 포트를 Code Block 의 Input 포트에 연결한다.
3. Code Block 의 Output 포트를 [Circle.ByCenterPointRadius] 의 Input 포트에 연결한다.

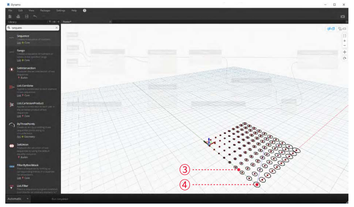


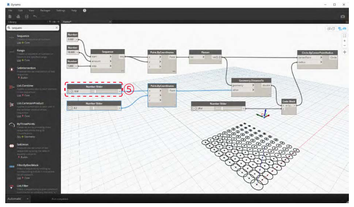
Adding complexity

단순하게 만들어진 원에 몇 개의 Node를 추가하여, 복합적인 형태로 변경한다.
하나의 원을 여러 개의 원으로 만들고, 원에 규칙을 갖는 반경 값을 주어 배치한다.

1. Number Sequence Node를 추가하고 [Point.ByCoordinates]에 다시 연결한다. [Point.ByCoordinates Node]를 마우스 오른쪽 클릭하여, 연산 방법을 Cross product로 변경한다. (Lacing) Cross Product)
2. Flatten Node를 추가하고 [Point.ByCoordinates]의 Output 포트를 Flatten Node 의 Input 포트에 연결한다.
3. "Run"을 클릭하여 실행 후 3D Preview에서 업데이트된 부분을 확인한다.





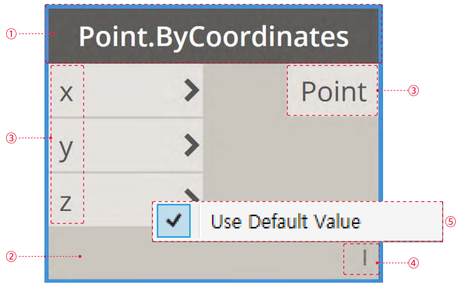


Adjusting with Direct Manipulation

Dynamo에서 원점 Point를 이동시키는 방법은 두 가지가 있다.
첫 번째는 [Point.ByCoordinates]에 연결된 Number Slider 값을 변경하여, 원점을 이동시킨다.
두 번째는 3D Preview에서 [Point.ByCoordinates]를 통해 표현되는 좌표 아이콘을 클릭하여 이동시킨다.
첫 번째와 두 번째 방법을 통해 사용자가 원하는 형태의 기준 또는 형태를 제어할 수 있다.

1. 이동시킬 Geometry를 생성한 Node를 클릭한다.
2. 작업 공간을 3D Preview Navigation mode로 전환한다.
3. Point 위로 마우스를 가져가 X, Y, Z 축을 확인한다.
4. 아이콘을 클릭하고 드래그하여 해당 축을 이동시킨다. 이동과 동시에 Silder 값이 실시간으로 업데이트 된다.
5. 만일 Number Slider를 연결하지 않은 축을 이동시킬 경우 자동으로 Number Slider Node가 생성되어 연결된다.

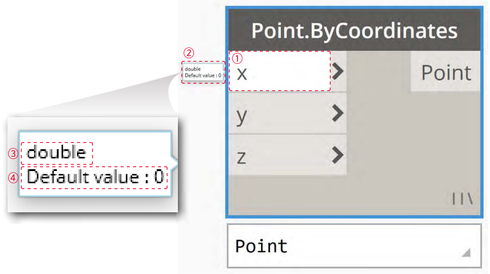
1.06 Nodes & Wires & Library



Anatomy of a Node

Dynamo의 Node는 5가지 부분으로 구성된다.
Python 코드를 활용하는 Node가 아니라면 각 Node의 구조는 다음과 같이 설명할 수 있다.

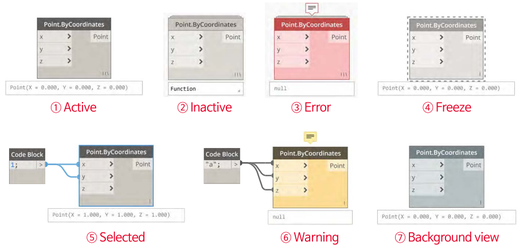
1. Name : Category + Name을 규칙으로 구성된 Node의 이름
2. Main : Node의 본체
3. Ports (Input & Output) : Node와 Node의 연결 또는 입력 Data 연결
4. Lacing : 설정한 연산 방법에 대한 설명
5. Default Value : Input에 Node 연결이 없을 경우 사용되는 설정된 기본값

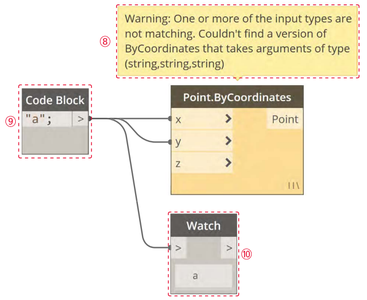


Ports

Node의 입력 및 출력을 하는 부분을 Port라고 하며 와이어를 통해 Node와 Node를 연결한다.
Node의 작업 순서는 연결되는 Port를 기준으로 왼쪽에서 오른쪽으로 진행된다.
Node의 Input Port에 연결되는 값의 종류는 Node에 따라 다르기 때문에, Node가 인식할 수 있는 값을 Port에 연결하여야 한다.
예를 들면 [Point.ByCoordinates]에 숫자 형식의 값을 Port에 연결하면 Node가 제대로 작동하지만 Text 또는 Geometry형식을 연결하면 오류가 발생한다.

1. Port Label
2. Tool Tip
3. Data Type
4. Default Value





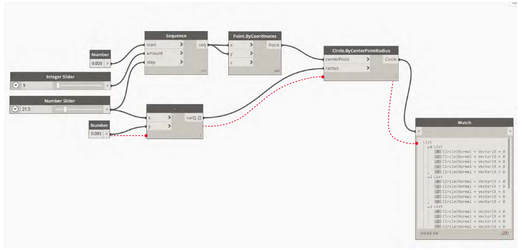
States

Dynamo를 실행 한 뒤, Node에 표현된 색으로 Node의 오류 및 실행 상태를 확인할 수 있다. 오류가 발생한 Node 일 경우 제공되는 Notice를 읽고 문제를 수정한다.

1. Active : 이름의 배경 색이 진한 회색으로 변경된다. Node의 연결 및 연산이 성공적으로 진행되었다는 의미이다.
2. Inactive : 이름의 배경색이 회색이고, 아직 Node가 활성화되지 않았다. Input Data의 연결을 확인한다.
3. Error State : 빨간색은 Node가 오류 상태임을 알려준다. 제공되는 Notice를 읽고 문제를 해결한다.
4. Freeze : 노드 실행을 일시 중지한다. 선택된 Node와 연결된 Node 전체가 일시 중지한다.
5. Selected : 현재 선택된 Node를 말하며, Node의 테두리 색이 변경된다.
6. Warning : 노란색은 Node가 잘못된 데이터 유형으로 연산의 오류 상태를 알려준다. 제공되는 Notice를 읽고 문제를 해결한다.
7. Background Preview : 어두운 회색으로 표현된 Node는 작업 공간에서 미리 보기가 꺼져 있음을 나타낸다.

Node에 경고 또는 오류가 발생하였으면 Dynamo는 해당 문제에 관련한 Notice를 제공한다. 노란색 또는 빨간색의 Notice를 확장하여 확인한다..

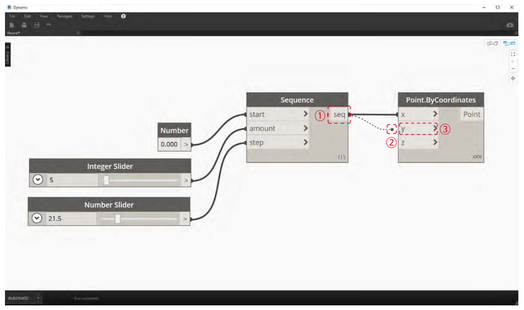
8. Warning Tooltip : 연결된 Data 값이 잘 연결되어 있다.
9. [Watch] Node를 이용하여 입력 Data를 확인한다.
10. Input Node인 Number에 숫자가 아닌 Text: Red가 기입되었다.

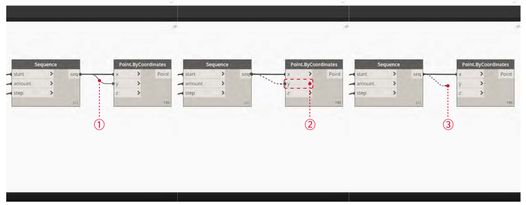


Creating Wires

Input할 Data를 가지고 있는 Node의 Output Port에서 마우스 왼쪽을 클릭하여 연결할 Node의 Input Port에 연결한다. 연결 과정에서 Wire은 점선으로 나타나고 연결이 완료되면 실선으로 표현된다. Dynamo에서 Data는 Wire를 통해 이동된다. Wire 연결이 잘못되어 잘못된 형상 또는 Data가공이 안되는 경우가 많아, 연결시 유의해야 한다.

1. Number Sequence Node의 seq 출력 Port를 클릭
2. 마우스를 이동하여 점선 Wire 확인
3. [Point.ByCoordiantes]의 y 입력 Port를 클릭하여 연결

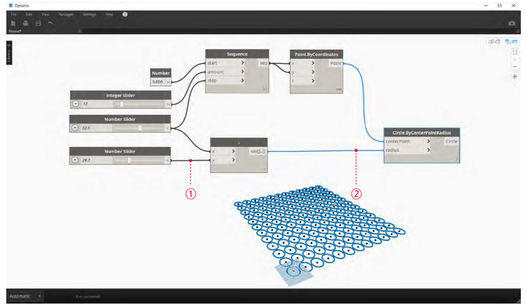




Editing Wires

연결된 Wire를 편집하여 Data의 흐름을 조정할 수 있다.
Wire를 수정하려면 이미 연결되어 있는 Node의 입력 Port를 클릭하고, 연결한 Port에 클릭한다.
Wire 연결을 하지 않을 경우 빈 작업 공간을 클릭한다.

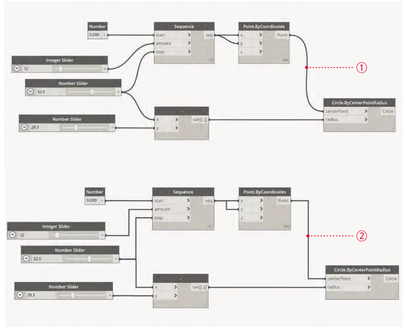
1. 기존 Wire
2. 입력 Port에 대한 연결을 변경하려면 다른 입력 Port를 클릭
3. Wire를 제거하려면 연결된 Port를 클릭하고, 빈 작업 공간 클릭



Wire Previews

기본적으로 Wire는 회색선으로 표현되지만, Node를 선택하면 Node에 연결된 Wire 선은 파란색으로 강조된다.

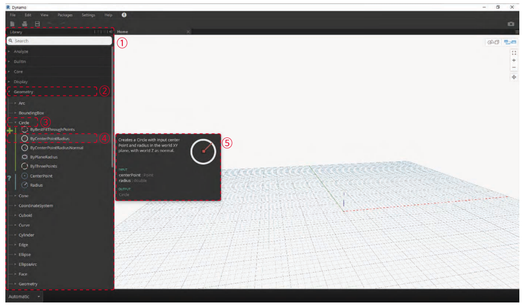
1. 기본 Wire
2. 선택된 Wire



Wire Previews

Dynamo에서 표현되는 Wire의 형태를 선택할 수 있다. View 〉 Connectors에서 Curve 또는 Poly Line를 선택해 표현될 Wire의 형태를 설정한다.

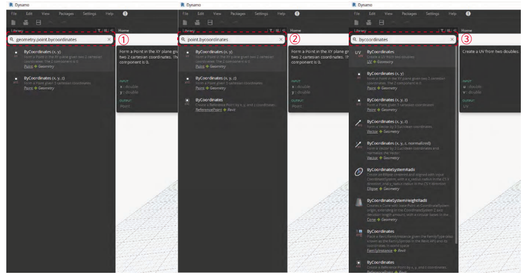
1. Connector Type: Curves
2. Connector Type: Polylines



The Organizational Scheme

Dynamo UI의 Library List는 Category의 종류와 순서로 정리되어 있다.
List 순서는 Library, Category, Sub-category 로 되어 있다.

1. Library : Dynamo에서 사용 할 수 있는 Node 메뉴
2. Library Name : Node의 기능
3. Category : 비슷한 연산을 하는 Node의 모음
4. Sub-Category
5. Node



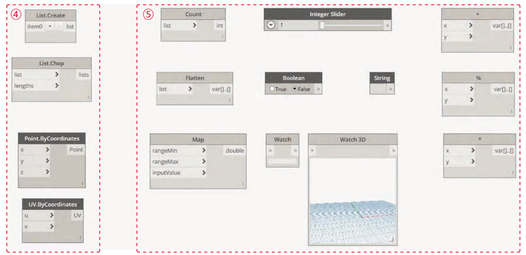
Naming Conventions

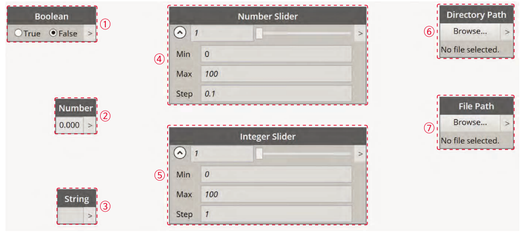
Node를 선택할 경우 Library에서 List를 확인하여 Node를 선택하는 방법과 검색 필드를 활용하여 Node를 선택하는 방법이 있다. 또는 Code Block을 이용하여 Node를 만들어 사용할 수도 있다. Node의 이름은 Library + Category + Node Name으로 구성되어 있다.

1. library.category.nodeName
2. category.nodeName
3. nodeName or keyword

일반적으로 작업 공간에 있는 Node Name은 category.nodeName으로 표현되지만, 입력 기준에 따라서, 이름은 비슷하지만 다른 Category로 분류된 Node도 있다.

4. [Point.ByCoordinates]와 [UV.ByCoordinates]는 같은 이름이지만 다른 카테고리이다
5. 자주 사용하는 기본적인 연산 Node 들이다.

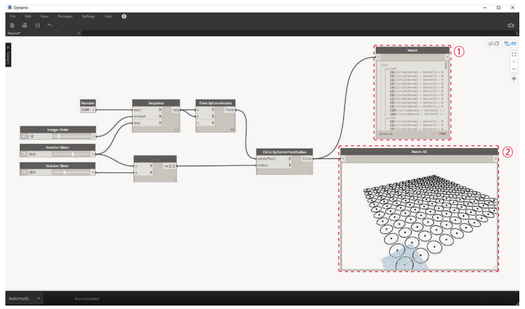




Input

Input Node는 Dynamo에서 가장 기본이 되는 Node이다.

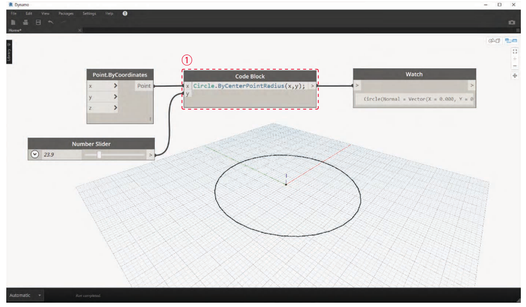
1. Boolean
2. Number
3. String
4. Number Slider
5. Integer Slider
6. Directory Path
7. File Path



Watch

Node를 출력 값을 확인하기 위해 Watch Node를 활용한다. Watch Node는 2D와 3D Watch Node로 제공된다. 2D watch Node는 출력된 값을 List 형태로 확인할 수 있고, 3D Watch Node는 미리 보기 형태로 제공된다.

1. Watch
2. Watch3D

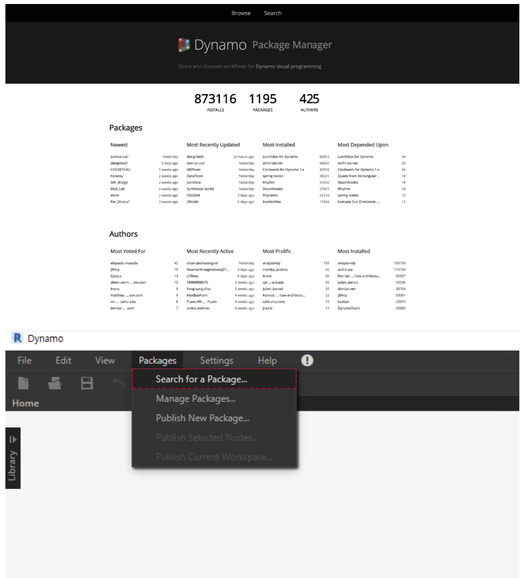


Code Block

Code Block Node는 수식 뒤에 ;를 붙여 Code를 작성한다. Code Block을 이용하여 사칙연산을 간단하게 할 수 있고, 프로그래밍 언어를 사용하여 Node를 만들 수도 있다. Dynamo의 Node를 이용하여 Data를 연산하는 것도 좋은 방법이지만, Node의 개수가 많아져, Code가 복잡해 보일 수 있다. 간단한 연산은 Code Block을 이용하는 것을 추천한다.

1. 빈 작업 공간을 더블 클릭하여 Code Block Node를 생성한다.
2. Code Block에 " Circle.ByCenterPointRadius(x, y);"라고 입력한다.
3. Code Block에 X와 Y Port가 자동으로 생성된다.
4. [Point.ByCoordinates] Node와 [Number Slider] Node를 Code Block x, y Port에 연결한다.
5. "Run"을 클릭하여 Dynamo를 실행한다.

1.07 Package Node



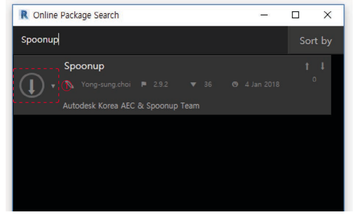
Packages

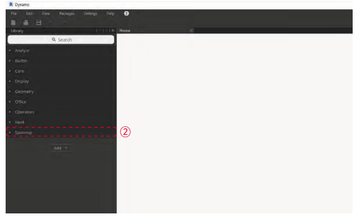
Package는 사용자 정의 Node의 모음이다. Dynamo 사용자라면 누구나 Package를 만들 수 있고, 제공되는 Package를 다운로드하여 사용할 수 있다. Package는 Dynamo 기능을 확장하기 위해서 사용자들이 개발한 Node 들이다. Dynamo의 기본 Node로 할 수 없는 다양한 기능들을 제공하며, 모든 Dynamo 사용자들이 액세스하여 버튼 클릭으로 Dynamo에 Package를 설치할 수 있다. Dynamo는 오픈 소스로 사용자 커뮤니티를 통해 기능은 더 확장되고, 프로그래밍은 간소화될 것으로 기대된다.

Installing a Package

Package를 설치 하는 가장 쉬운 방법은 Package 메뉴에서 Search for a package에서 검색 해서 다운 받는 것이다.

- In Dynamo, go to Packages) Search For a Package



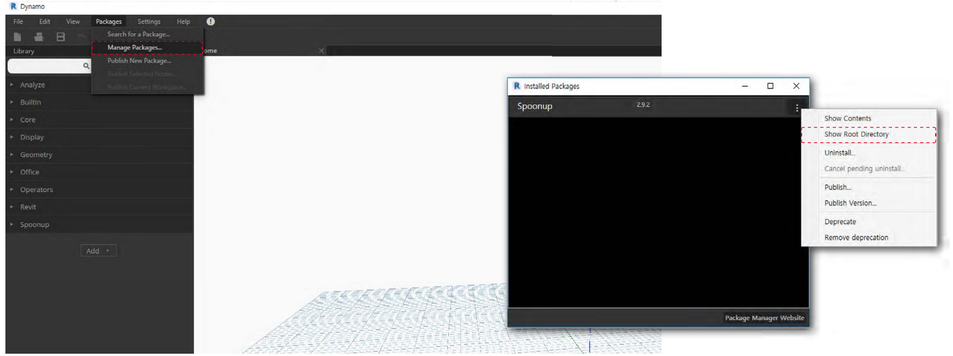




Installing a Package

검색 창에서 "Spoonup"이라고 검색하면, Spoonup 단어와 연관된 모든 Package를 볼 수 있다.

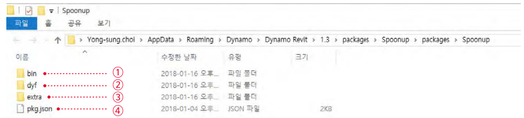
1. Package 이름 왼쪽에 있는 다운로드 아이콘을 클릭 하면, 자동 설치된다.
2. Dynamo Library List에 "Spoonup"이라는 그룹이 생성 되었다.
3. Spoonup Package에서 제공하는 Node들 모음이다.



Package Folders

Spoonup Package 이외에도 "Clockwork","Archi-lab.net","LunchBox","Spring nodes","Rhythm"는 Dynamo 사용자들이 많이 사용하는 Package 들이다.Package 다운로드시 함께 다운로드하면 Dynamo의 활용도가 더 높아질 것이다.

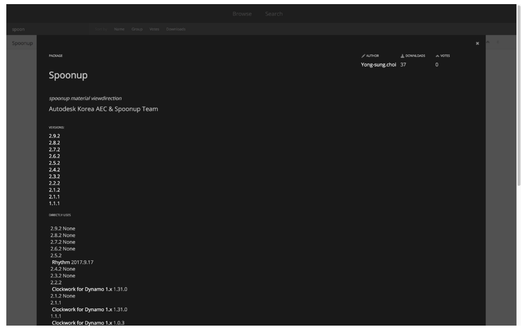
Package 다운로드가 완료되면, 메뉴 탭의 Package Manage를 통해 설치된 Package 폴더를 확인할 수 있다. 설치된 Package 목록에서 오른쪽 버튼을 클릭하여 "Show Root Directory"를 선택한다.



Package Folders

이동된 Root Directory에서 3개의 폴더와 파일을 확인할 수 있다

1. "bin" 폴더 : .dll 파일이 있고, Dynamo Package가 Zero-touch를 사용해 개발될 경우 활용된다.
2. "dyf" 폴더 : Dynamo을 사용자 Node 만들기 기능을 활용해 작성된 Node를 담고 있다.
3. "extra" 폴더 : 예제 파일을 포함하여 모든 추가 파일을 담고 있다
4. "pkg" 파일 : Package의 설정을 정의하는 기본 텍스트 파일이다.



Dynamo Package Manager

Dynamo Package를 검색하는 또 다른 방법은 Package Manager 홈페이지 (http://dynamopackages.com/)에 방문하는 것이다. 웹에서 다운로드 횟수와 인기도 순으로 Package가 정렬된다. 또한 Package의 버전을 확인할 수 있어, 최신 업데이트 정보도 확인할 수 있다.

Dynamo Package Manager (http://dynamopackages.com/)에서 "Spoonup"을 확인하면 Package 설명, 버전,개발자 등... 을 확인할 수 있다.

2. Dynamo Code
일반 도면

G1. Tablelayout 만들기

동영상 : https://youtu.be/5HXKK-ykhnU

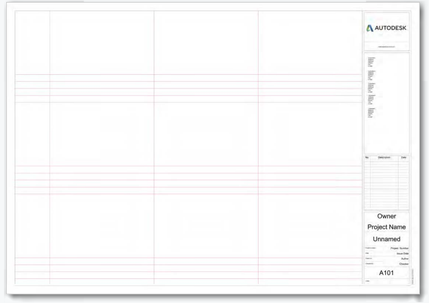
기능 및 필요성

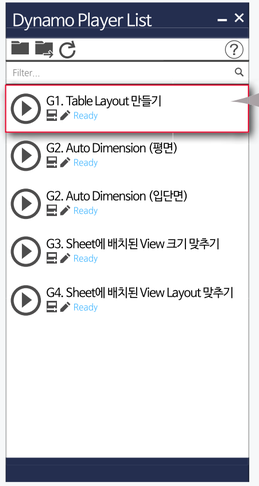
부재별 일람표 및 도면의 산출 근거를 표현하기 위해서 표를 그려야 할 때 사용한다.

유의사항

- 작업 화면 : 3D View와 입단면 View를 제외한 평면, Legend, Sheet View에서 구동이 가능하다.
- 작성된 Tablelayout은 Detail Line으로 작성된다.
- 한번 작성된 Detial Line은 설정값을 변경하여 재실행 하여도 기존 작성된 Line은 유지된다.









* Dynamo Player는 Revit 2017부터 사용 가능하며, Player의 설정창은 Revit 2018 이상 버전에서 사용할 수 있다.
Revit 2017일 경우 Dynamo 파일을 열고 설정을 수정하여 저장한 뒤 Player를 누르거나 Dynamo에서 바로 실행시킨다.

Tablelayout 생성을 위한 Prameters

① 가로폭
Tablelayout을 생성할 가로 공간(mm)

② 세로폭
Tablelayout을 생성할 세로공간(mm)

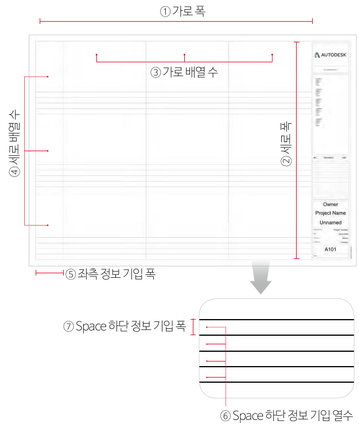
③ 가로 배열 수
부재의 형상을 넣을 가로 공간 개수

④ 세로 배열 수
부재의 형상을 넣을 가로 공간 개수

⑤ 좌측 정보 기입 폭
부재의 정보를 담는 Title 공간(mm)

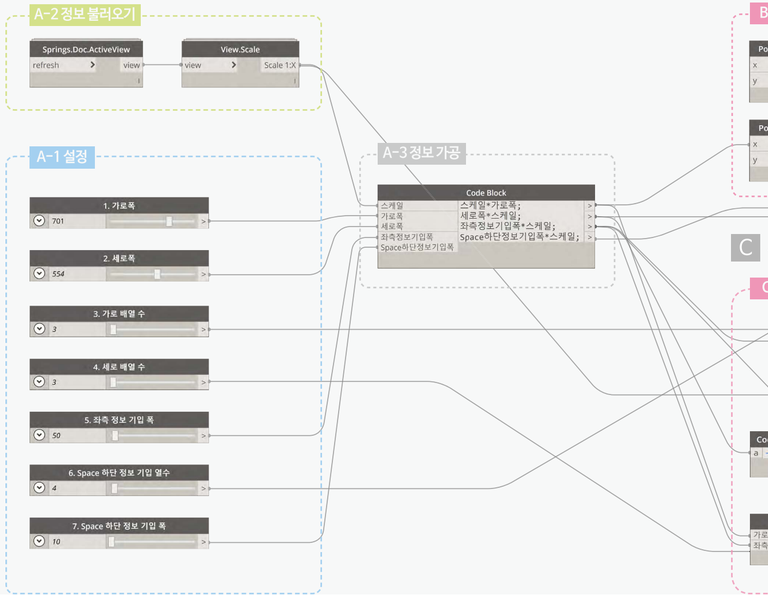
⑥ Space 하단 정보 기입 열 수
부재 하단에 넣을 정보 개수

⑦ Space 하단 정보 기입 폭
Text 라인의 폭(mm)

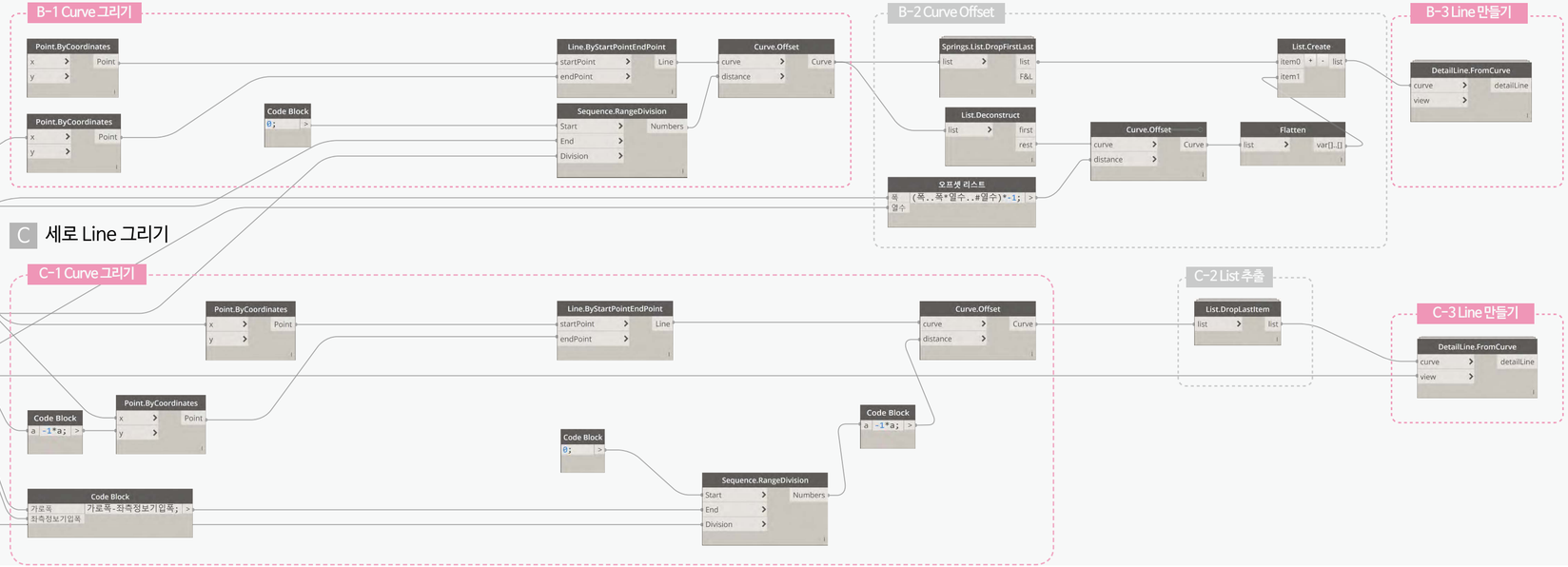


CODE OVERVIEW

A Table Layout 설정 및 View Scale 반영



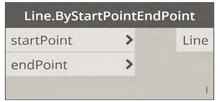
B 가로 Line 그리기



C 세로 Line 그리기

Key Node

Dynamo basic node

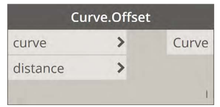


입력된 두 Point 사이에 Line을 생성한다.

INPUTS
- startPoint [Point 포인트]
  직선 Line의 시작점을 생성한다.
- endPoint [Point 포인트]
  직선 Line의 끝점을 생성한다.

OUTPUTS
- Line [Line 라인]
  입력된 시작점과 끝점을 이어주는 Lien을 생성한다.

Dynamo basic node

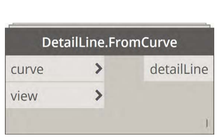


특정 값에 의하여 Curve의 간격을 부여한다.
Curve는 평면상에 있어야 하며, BSplineCurve/NurbsCurve는 1 이상의 값을 가져야 한다.

INPUTS
- curve [curve 객체]
  지정된 시작점과 끝점의 값을 연결하는 curve를 연결한다.
- distance
  curve를 +,- 값으로 간격을 줄 거리 값을 연결한다.

OUTPUTS
- Curve [curve 객체]
  작성된 curve에 distance 거리의 값을 부여하여 일정 간격 curve에 간격을 주어 이격시킨다.

Clockwork Packages

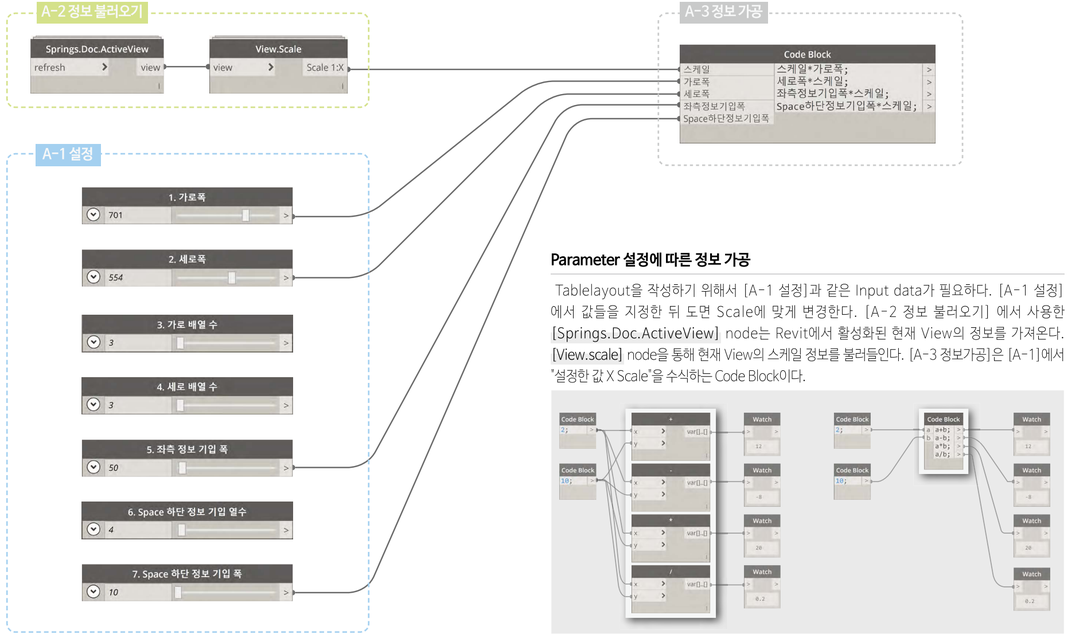


기존 커브를 기반으로 Revit의 원하는 View에 디테일 Line을 생성한다.

INPUTS
- curve [curve 객체]
  Dynamo에서 작성된 curve 객체를 연결한다.
- view [view 객체]
  Detail line을 배치 할 View를 지정한다.

OUTPUTS
- detail Line [Line 라인]
  Dynamo에서 작성된 curve 객체를 지정된 view를 통하여 Revit의 Detail line 부재로 만든다.

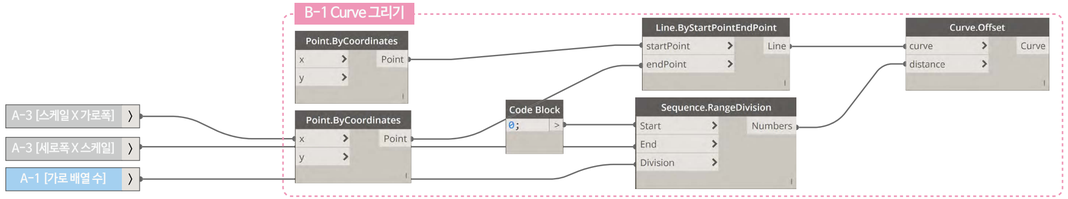
A Table Layout 설정 및 View Scale 반영



Parameter 설정에 따른 정보 가공

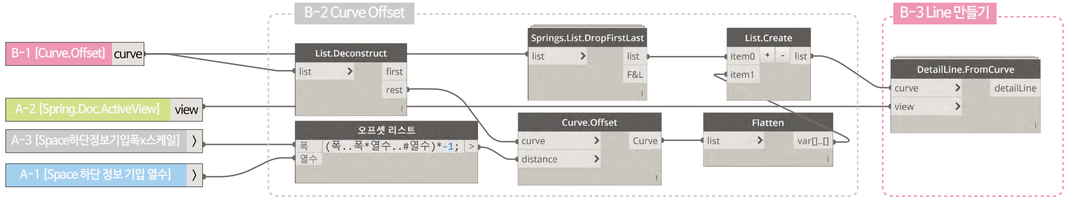
Tablelayout을 작성하기 위해서 [A-1 설정]과 같은 Input data가 필요하다. [A-1 설정]에서 값들을 지정한 뒤 도면 Scale에 맞게 변경한다. [A-2 정보 불러오기] 에서 사용한 [Springs.Doc.ActiveView] node는 Revit에서 활성화된 현재 View의 정보를 가져온다. [View.scale] node을 통해 현재 View의 스케일 정보를 불러들인다. [A-3 정보가공]은 [A-1]에서 "설정한 값 X Scale"을 수식하는 Code Block이다.

B 가로 Line 그리기



Curve 작성

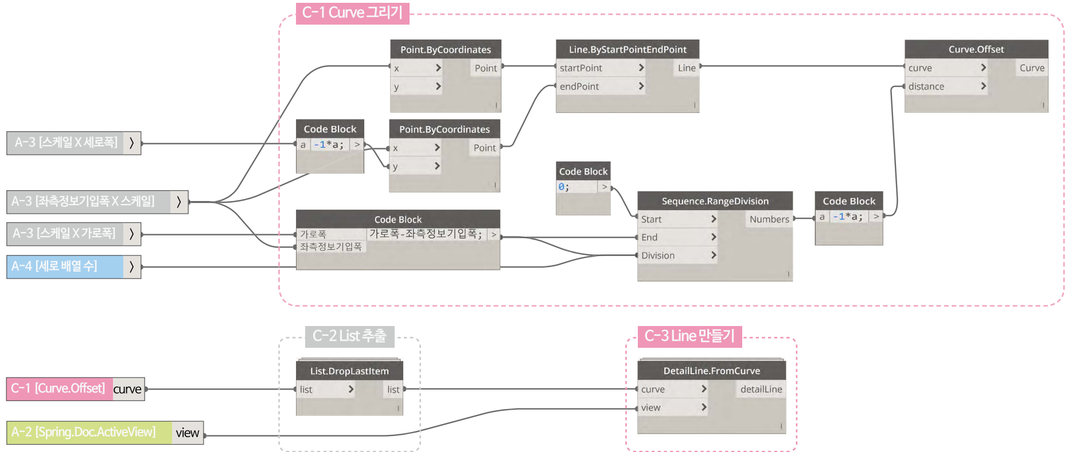
Curve를 그리기 위해서는 Line 이 필요하다. 1개의 Line을 그리기 위해서는 Line이 시작하는 Startpoint와 Endpoint를 먼저 지정해야 한다. [Point.ByCoordinates] node는 Input data가 없으면 (0.0)좌표로 자동 설정 된다. 첫번째 Line의 시작점이 (0.0)이며, 끝점은 (스케일x가로폭.0)이다. [B-1 Curve 그리기]에서 만들어진 첫번째 Line을 기준으로 [A-3], [A-1]에서 설정된 값으로 Curve를 Offset 한다.



Detail line 작성

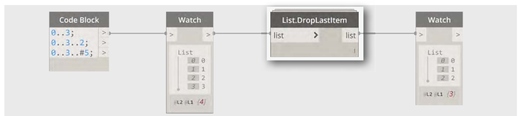
Dynamo에서 작성된 Curve는 [DetailLine.FromCurve]를 통해서 Revit의 Detail line 부재로 만들 수 있다. Revit 부재로 변경할 Curve list와 Detail line을 배치할 view를 지정해 주어야 한다. [B-2 curve offset]에서 작성된 값은 [Springs.List.DropFirstLast]으로 통해서 첫번째 값과 마지막 값을 삭제시킨다.

C 세로 Line 그리기



Curve 작성

[Curve.Offset] node에 Input 할 Distance 값을 구하기위해서 [Sequence.RangeDivision] node를 이용한다. 시작 값, 끝 값 그리고 나누어야 하는 값(가로 배열 개수)을 넣으면 범위를 나눠주는 기능을 한다. [Sequence.RangeDivision] node로, 동일한 간격을 갖는 Distance 값을 만들어 준다. [Code Block (-1*a)]는 세로라인의 배열이 (Y)축 기준 (-) 이기 때문에 Line을 위에서 아래로 그려주기 위한 것이다.



G2. Auto Dimension (평면)

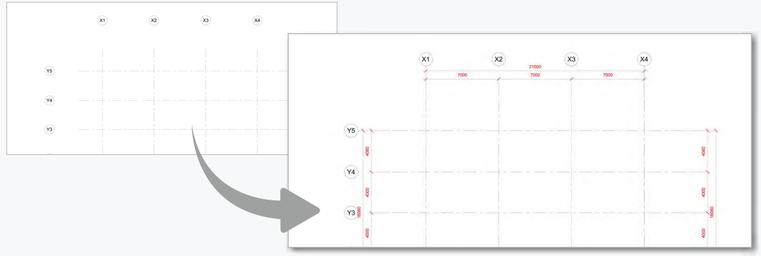
동영상 : https://youtu.be/dEgYXqmsPvY

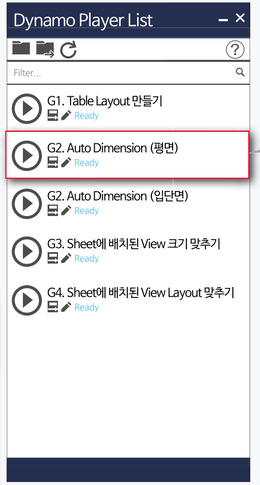
기능 및 필요성

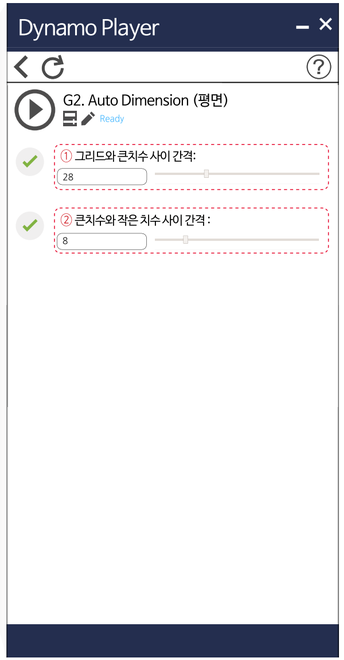
평면에 치수를 표현할 때 활용할 수 있다. Auto Dimension(평면)은 상하좌우에 2개씩 총 8개의 Dimension이 배치된다.

유의사항

- 작업 화면 : 3D View와 입단면 View를 제외한 그리드가 있는 평면 View에서 작동이 가능하다.
- View에 Crop 설정이 되어 있는지 확인한다.
- 한번 작성된 Dimension은 설정값을 변경하여 재실행 하여도 기존 Diemension은 유지된다.
- 사선 및 각도를 갖는 그리드에 Dimension 배치는 불가능하다.





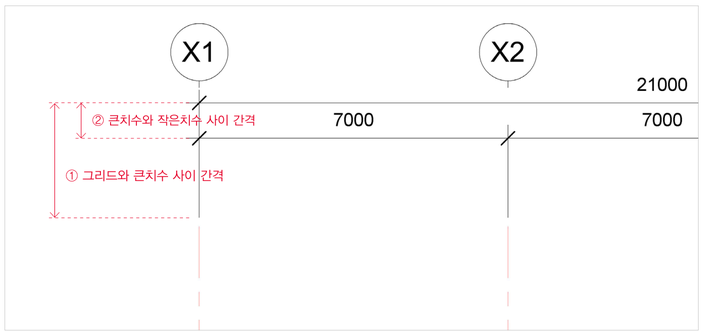


* Dynamo Player는 Revit 2017부터 사용 가능하며, Player의 설정창은 Revit 2018 이상 버전에서 사용할 수 있다.
Revit 2017일 경우 Dynamo 파일을 열고 설정을 수정하여 저장한 뒤 Player를 누르거나 Dynamo에서 바로 실행시킨다.

Dimension 배치를 위한 Parameter 조정

① 그리드와 큰치수 사이 간격
그리드와 큰치수 공간

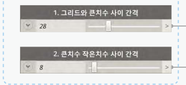
② 큰치수와 작은 치수 사이 간격
큰치수와 작은치수 공간



CODE OVERVIEW

A Grid&Dim 간격 및 View 정보 가져오기

A-1 Grid&DIM 간격 설정



A-2 현재 View Scale & Sheet Number 정보 추출



A-3 설정 값 Scale 적용



B Grid&Dim 간격 및 View 정보 가져오기

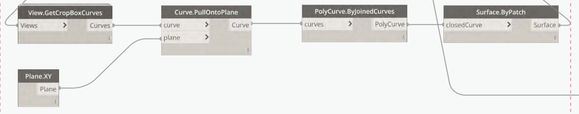
B-1 사용된 Grid 가져오기



B-2 Grid를 Curve로 변환



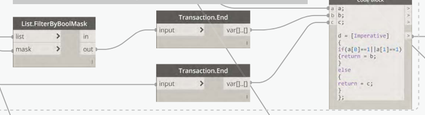
B-3 현재 View의 Cropbox 단일 커브로 변환



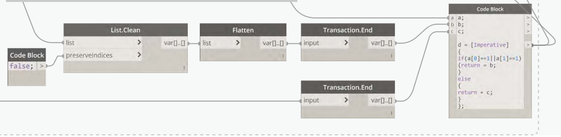
B-4 Grid와 Cropbox 간섭 체크



B-5 현재 View의 Crop on/off 여부에 따른 Grid 선택

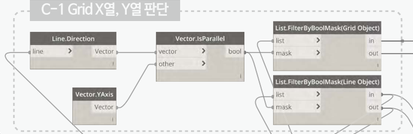


B-6 현재 View의 Crop on/off 여부에 따른 Line 선택

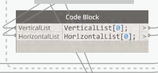


C DIM을 작성할 Grid의 X열, Y열 판단

C-1 Grid X열, Y열 판단



C-2 List 작성



D DIM을 배치할 기준 Grid 작성

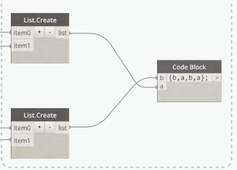
D-1 Y열 Grid 선택



D-2 X열 Grid 선택

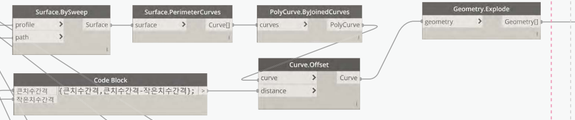


D-3 List 작성



E DIM을 배치할 기준 Line 작성

E-1 외곽 Line 작성



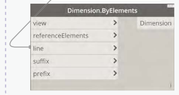
E-2 List 작성



E-3 Line 그리기



E-4 DIM 배치



Key Node

Dynamo
basic node

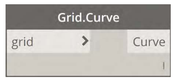


두 Vector값이 서로 평행인지 아닌지 판단한다.

INPUTS
- Vector
  기준으로 설정 할 Vector값을 선택하여 Input한다.
- Other
  기준 Vector와 비교 할 다른 Vector값을 Input한다.

OUTPUTS
- Bool
  두개의 Vector 값이 평행을 이루는 지 True/False로 판단한다.

Dynamo
basic node

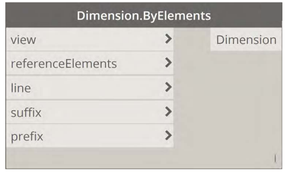


Revit Project에 배치된 Grid 부재를 선택하여 Dynamo의 Curve 부재로 변환한다.

INPUTS
- Grid [grid 객체]
  Revit의 요소인 Grid를 선택하여 부재로 가져온다.

OUTPUTS
- Curve [Curve 객체]
  가져온 Revit의 Grid 요소를 Dynamo의 Curve로 생성한다.

Dynamo
basic node

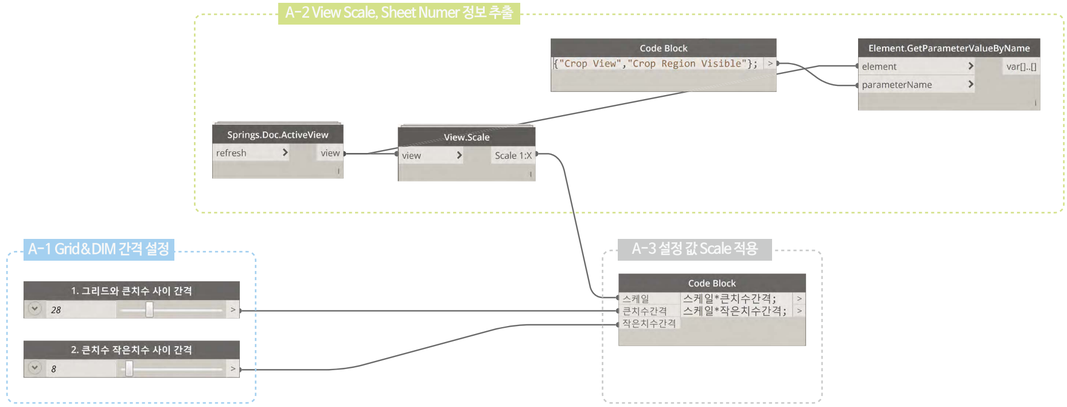


적어도 2개의 요소를 선택하여 Dimension을 배치한다.

INPUTS
- Line [Line 객체]
  Dimension의 위치 값을 지정한다.
- View [View 객체]
  Dimension이 위치할 View의 위치를 지정한다.

OUTPUTS
- Dimension
  Input 값을 통하여 DIM을 배치한다.

A Grid&Dim 간격 및 View 정보 가져오기

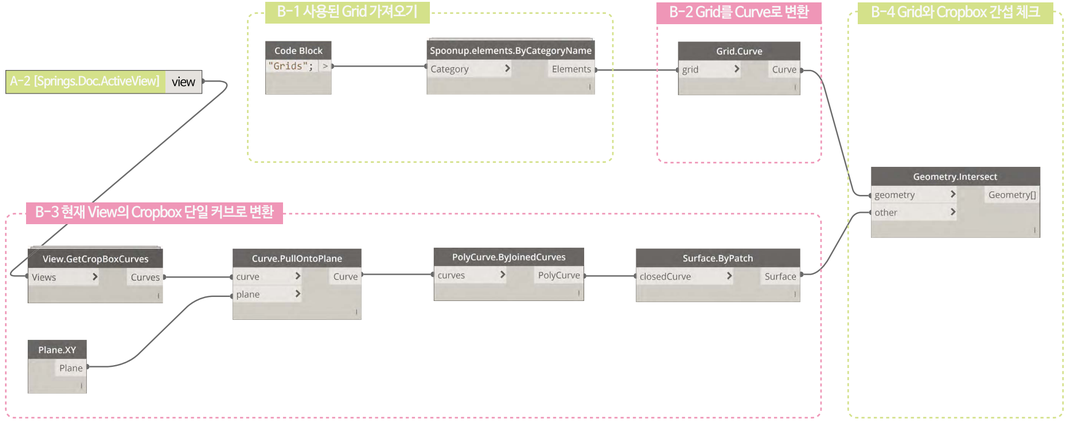


Parameter 설정에 따른 정보가공

[Integer Slider]를 이용하여 그리드와 큰치수 간격과 큰치수와 작은 치수 간격을 설정한다. 설정된 값은 [View.Scale]을 이용해 추출한 현재 View의 Scale 값과 [Code Block]에서 연산을 한다. 연산된 값은 현재 View의 Scale이 반영된 실제 거리 간격 값이다.

[GetParameterValueByName]을 통하여 현재 View의 "Crop View", "Crop Region Visible"의 On/Off 여부를 확인한다. 확인된 값에 의해서 치수를 배치할 위치가 결정된다.

B Grid&Dimension 간격 및 View 정보 가져오기

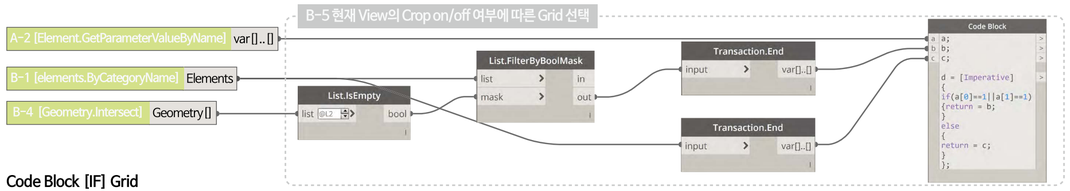


Grid Curves & Crop Box Curves

"Grids" Category명을 [Spoonup.element.ByCategoryName] 또는 [Category.ByName]과 [AllElementOfCategory]에 연결하여 Project에 배치된 Grid 부재를 선택한다. 선택된 Grid는 [Grid.Curve] node에 Input하여 Dynamo의 Curve 부재로 변환한다.

[Spring.Doc.ActiveView]에 의해 추출된 현재 View를 [View.GetCropBoxCurves] node의 Input에 연결하여 현재 View에 설정된 Crop Box의 Curve를 Dynamo 부재로 가지고 온다.

Dynamo로 가져온 Curve는 [Curve.PullOntoPlane] node를 이용하여 XY평면에 Curve를 투영시킨다. Dynamo의 XY평면에 투영된 Curve는 [PolyCurve.ByjoinedCurves]와 [Surface.ByPatch]를 이용하여 Surface 면을 만든다. [Geometry.Intersect] node를 이용하여 Crop Box Curve와 Grid와 간섭되는 부분을 찾는다. 사용자가 Crop View를 사용할 경우 Crop view에 해당하는 Grid만 치수로 표현하기 위한 Node들이다.



Code Block [IF] Grid

[A-2] [GetParameterValueByName] node로 확인한 "Crop view"와 "Crop Region Visible"의 On/Off여부에 따라서, Grid 전체에 DIM을 배치할지, Crop Box만 있는 Grid에 DIM을 배치할 것인지 결정하는 node들이다. [Code Block]은 "Crop view"와 "Crop Region view"가 둘중 하나라도 On이 되어 있어면 Crop Box Curve 안에 포함된 Grid만 선택하고 , "Crop view"와 "Crop Region Visible"의 설정 모두 Off 되어있을 경우, Project에 사용된 전체 Grid를 선택한다.

[B-4]에서 추출된 Crop Box Curve와 간섭된 Grid를 판별하기 위해 [List.IsEmpty]와 [List.FilterByBoolMask]를 이용하였다.

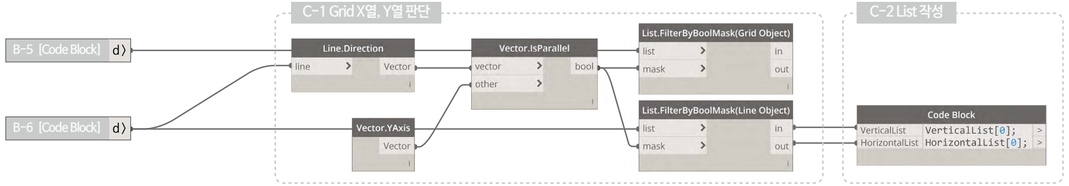
[List.IsEmpty] node는 간섭된 list 값을 True/False로 list를 정리해준다. 정리된 list는 [Code Block]에 Input하기전에 [Transaction.End]를 연결하여 현재 View의 Crop여부에 대한 연산 순서를 부여한다.



Code Block [IF] Grid Curve

치수를 배치할 기준이 되는 Line은 치수를 배치할 Grid를 선택하는 방법과 같다. [Code Block]은 "Crop view"와 "Crop Region view"의 On/Off의 여부에 따라서, 배치할 치수의 위치가 되는 기준선의 범위가 달라진다. 현재 View에 Crop 설정이 있을 경우, Crop Box와 간섭되는 Grid를 Curve로 선택한다. Crop 설정이 없을 경우, 전체 Grid를 Curve로 선택한다. [Code Block]에 Input하기전에 [Transaction.End]를 연결하여 현재 View의 Crop여부에 대한 연산 순서를 부여한다.

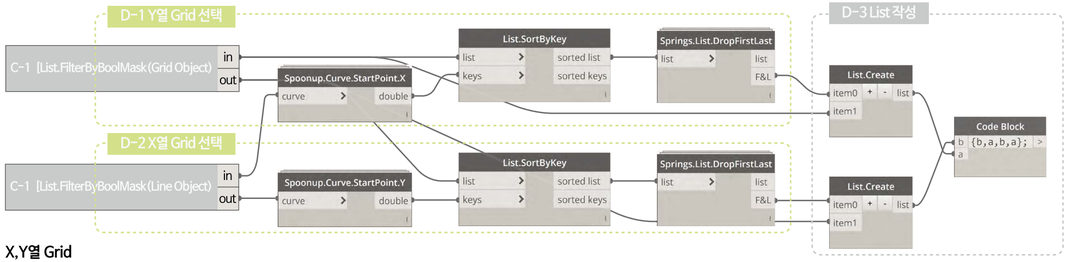
C Dimension을 작성할 Grid의 X열, Y열 판단



X,Y열 Grid & Line

현재 View의 Crop설정에 따라 추출된 Grid와 Line은 [Vector.IsParallel] node와 [List.FilterByBoolMask] node를 이용하여 X,Y와 평행한 Grid와 Line을 분리한다.

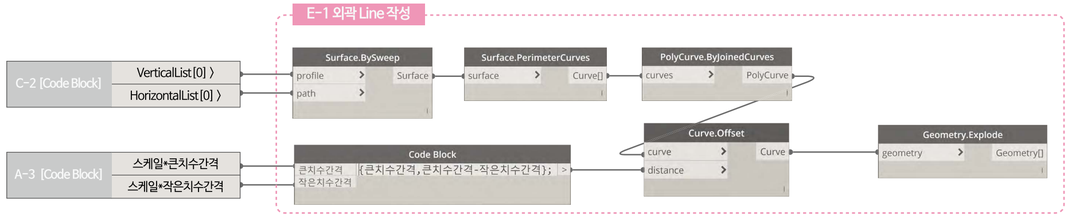
D Dimension을 배치할 기준 Grid 작성



X,Y열 Grid

[List.FilterByBoolMask]에 분리된 X,Y열의 Grid와 Line으로 X,Y열 Grid를 선택한다. X열 Grid들의 Curve는 X좌표가 다르기 때문에 [Point.X] node로 X열의 가장 왼쪽과 오른쪽 Grid를 선택하고 [List.Create] node를 통해서 치수를 배치할 Grid와 List를 합친다. Y열 Grid들의 Curve는 Y좌표가 다르기 때문에 [Point.Y] node를 통해서 치수를 배치한다. [Code Block] {b,a,b,a}는 Grid의 치수를 상하좌우 4면에 배치해 야 하기 때문에 X,Y열 Grid를 두번씩 선택되게 만든다. 정리된 List는 [Dimension.ByElements]의 Reference Elements에 Input한다.

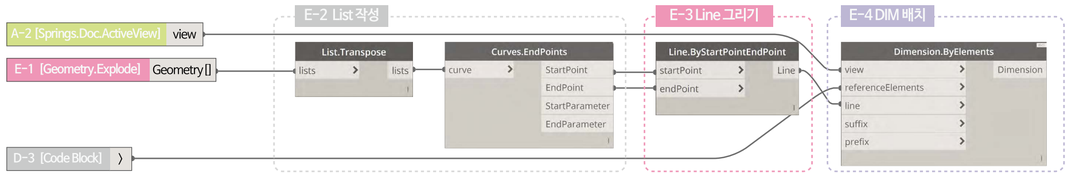
E Dimension을 배치할 기준 Line 작성



Parameter 설정에 따른 Curve Offset

[List.FilterByBoolMask] node에 의해 X,Y열로 분리된 Line 중 List[0]에 해당하는 Line을 선택하여 [Surface.BySweep]을 이용하여 Grid 전체를 포함하는 Surface를 Dynamo의 Geometry로 생성한다. 생성된 Geometry는 [Surface.PerimeterCurves]를 이용하여 Surface의 외곽 Curve를 추출한다.

외곽 Curve는 치수가 배치될 기준선이기 때문에 [A]에서 설정한 'Grid와 큰치수 간격', '큰치수와 작은 치수 간격'의 값을 넣어 Curve Offset한다. 작성된 Curve들은 치수가 배치될 개별 Curve로 변환하기 위해 [Geometry.Explode] node를 이용하여 만든다.



Dimension 작성

개별 Curve로 추출된 Curve의 Start Point와 End Point를 추출한다. [Line.ByStartPoint EndPoint]를 이용하여 Line Geometry를 만든다.

만들어진 Line Geometry는 [Dimension.ByElements] node의 Line에 Input한다.

G2. Auto Dimension (입단면)

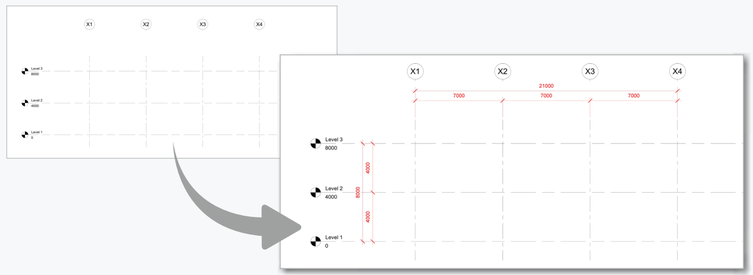
동영상 :https://youtu.be/nwV70mMS4xs

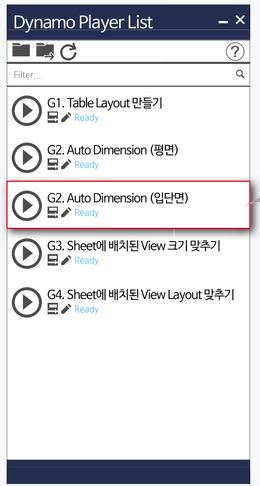
기능 및 필요성

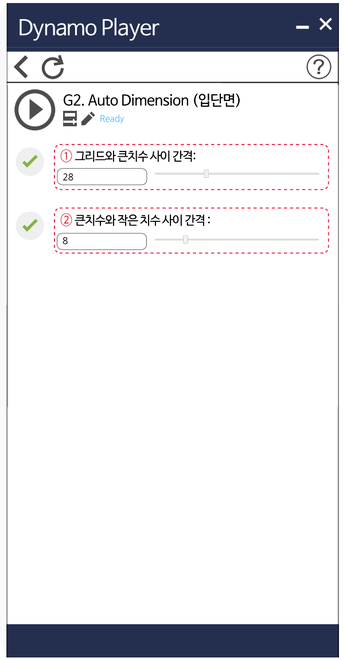
입면, 단면에 치수를 표현할 때 활용 할 수 있다. Auto Dimension(입단면)은 상단에 2개, 좌측에 2개 총 4개의 Dimension이 배치된다.

유의사항

- 작업 화면 : 3D View와 평면 View를 제외한 그리드가 있는 입단면 View에서 작동이 가능하다.
- View에 Crop 설정이 되어 있는지 확인한다.
- 한번 작성된 Dimension은 설정값을 변경하여 재실행 하여도 기존 Diemension은 유지된다.





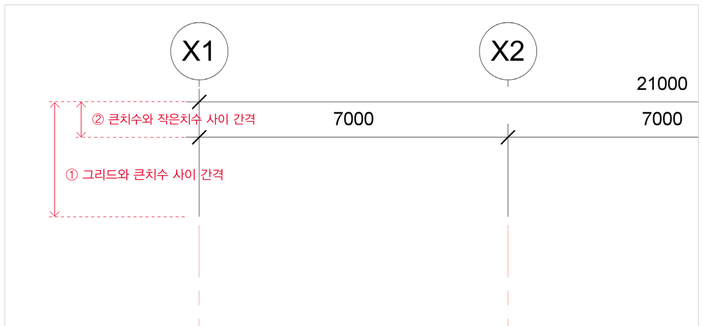


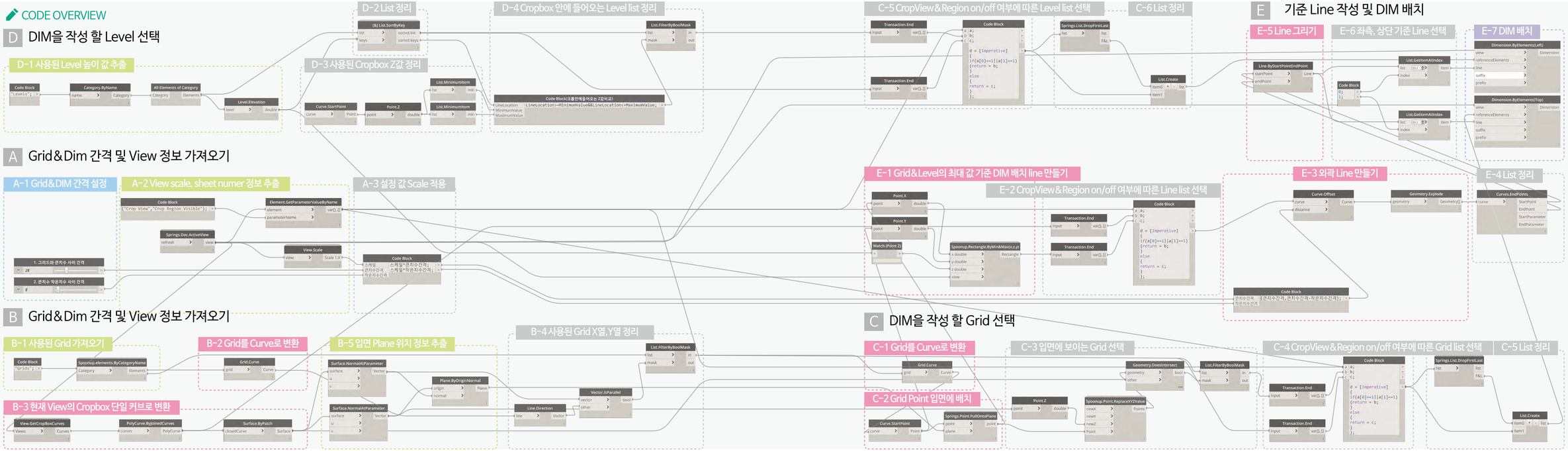
* Dynamo Player는 Revit 2017부터 사용 가능하며, Player의 설정창은 Revit 2018 이상 버전에서 사용할 수 있다.
Revit 2017일 경우 Dynamo 파일을 열고 설정을 수정하여 저장한 뒤 Player를 누르거나 Dynamo에서 바로 실행시킨다.

Dimension 배치를 위한 Parameter 조정

① 그리드와 큰치수 사이 간격
그리드 라인으로부터 큰치수를 띄울 간격 (mm)

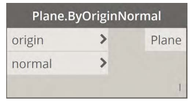
② 큰치수와 작은 치수 사이 간격
큰치수와 작은 치수 사이의 간격 (mm)




CODE OVERVIEW
D DIM을 작성 할 Level 선택
D-1 사용된 Level 높이 값 추출
D-2 List 정리
D-3 사용된 Cropbox Z값 정리
D-4 Cropbox 안에 들어오는 Level list 정리
A Grid&Dim 간격 및 View 정보 가져오기
A-1 Grid&DIM 간격 설정
A-2 View scale, sheet numer 정보 추출
A-3 설정 값 Scale 적용
B Grid&Dim 간격 및 View 정보 가져오기
B-1 사용된 Grid 가져오기
B-2 Grid를 Curve로 변환
B-3 현재 View의 Cropbox 단일 커브로 변환
B-4 사용된 Grid X열,Y열 정리
B-5 입면 Plane 위치 정보 추출
C DIM을 작성 할 Grid 선택
C-1 Grid를 Curve로 변환
C-2 Grid Point 입면에 배치
C-3 입면에 보이는 Grid 선택
C-4 CropView&Region on/off 여부에 따른 Grid list 선택
C-5 List 정리
C-5 CropView&Region on/off 여부에 따른 Level list 선택
C-6 List 정리
E 기준 Line 작성 및 DIM 배치
E-1 Grid&Level의 최대 값 기준 DIM 배치 line 만들기
E-2 CropView&Region on/off 여부에 따른 Line list 선택
E-3 외곽 Line 만들기
E-4 List 정리
E-5 Line 그리기
E-6 좌측, 상단 기준 Line 선택
E-7 DIM 배치

Key Node

Dynamo basic node



선택된 원점과 Input한 Point를 기준으로 Plane을 작성한다.

INPUTS
- Origin [Point 포인트]
  원점 Point 를 가져온다.
- Normal [Point 포인트]
  Vector값의 Point를 가져온다.

OUTPUTS
- Plane
  Origin과 Normal 포인트를 기준으로 평면을 생성한다.

custom node Spoonup Packages

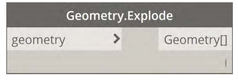


X,Y,Z 포인트의 최대, 최소값으로 사각형을 생성한다.

INPUTS
- X,Y,Z Double
  각 축의 최소 혹은 최대 값을 가져온다.
- View
  활성화 할 뷰의 정보를 가져온다.

OUTPUTS
- Rectangle
  활성화 된 뷰에서 각 축의 값을 정하여 사각형을 생성한다.

Dynamo basic node

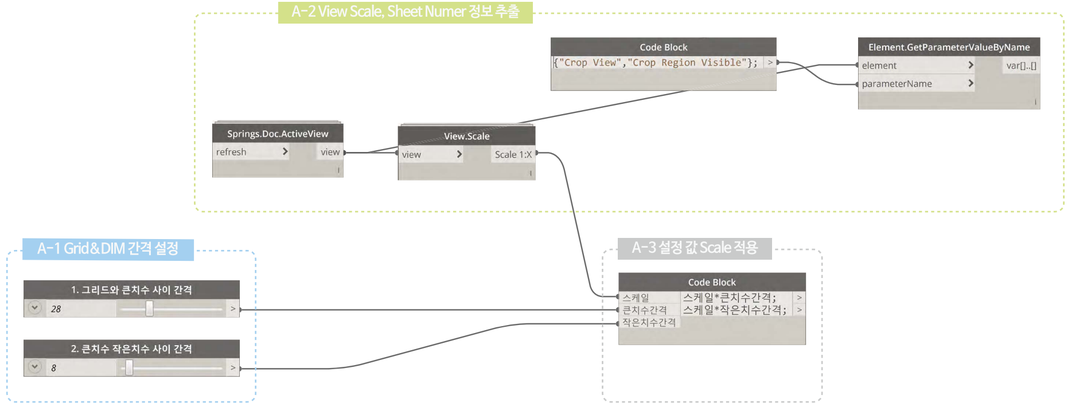


Dynamo에서 생성된 Geometry를 구성 요소들로 분리한다.

INPUTS
- Geometry
  요소 혹은 Curve의 값을 불러 온다.

OUTPUTS
- Geometry[]
  요소 혹은 Curve의 혼합체를 분리하여 전개한다.

A Grid&Dimension 간격 및 View 정보 가져오기

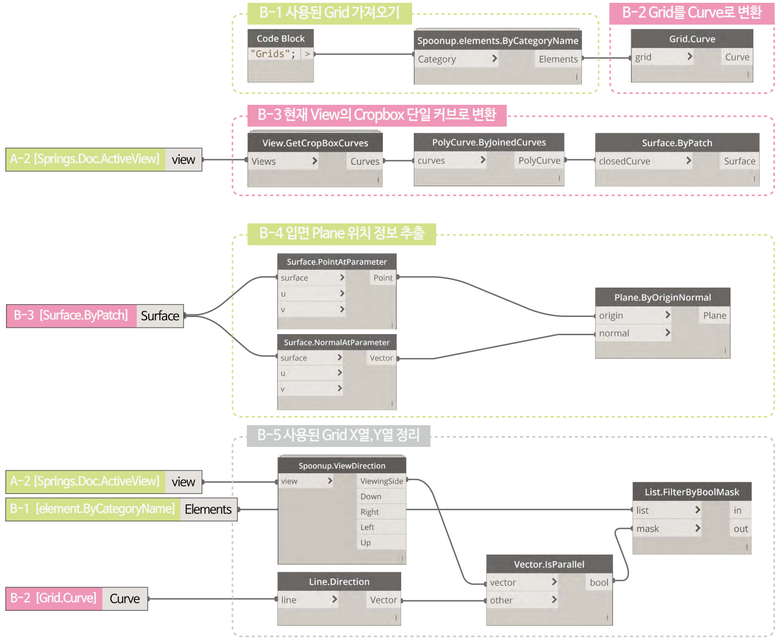


Parameter 설정에 따른 정보가공

[Integer Slider]를 이용하여 그리드와 큰치수 간격과 큰치수와 작은 치수 간격을 설정한다. 설정된 값은 [View.Scale]을 이용해 추출한 현재 View의 Scale 값과 [Code Block]에서 연산을 한다. 연산된 값은 현재 View의 Scale이 반영된 실제 거리 간격 값이다.

[GetParameterValueByName]을 통하여 현재 View의 "Crop View", "Crop Region Visible"의 On/Off 여부를 확인한다. 확인된 값에 의해서 치수를 배치할 위치가 결정된다.

B Grid&Dimension 간격 및 View 정보 가져오기



현재 View의 Plane

"Grids" Category명을 [Spoonup.elementsBy CategoryName] 또는 [Category.ByName]과 [AllElementOfCategory]에 연결하여 Project에 배치된 Grid 부재를 선택한다.

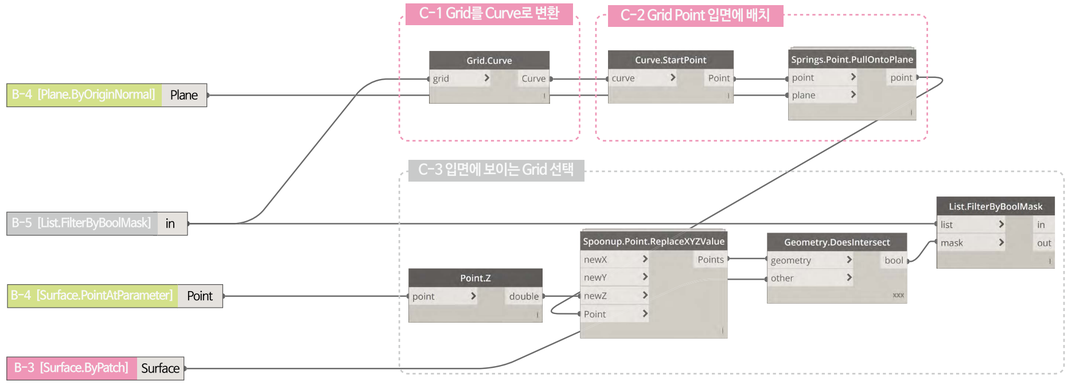
선택된 Grid는 [Grid.Curve] node에 Input하여 Dynamo의 Curve 부재로 변환한다.

[Springs.Doc.ActiveView]에 의해 추출된 현재 View를 [View.GetCropBoxCurves] node의 Input에 연결하여 현재 View에 설정된 Crop Box의 Curve를 Dynamo 부재로 가지고 온다. Curve는 [PolyCurve.ByJoinedCurve]와 [Surface.ByPatch]를 이용하여 Surface 면을 만든다.

입면에서 Dimension을 표기하기 위해서 현재 View의 방향을 확인하여야 한다. 입단면에 배치된 Crop Box Curve로 만든 Surface는 [Surface.NormalAtParameter]와 [Surface. PointAtParameter] node를 이용하여 Vector 값과 Surface의 Origin 값을 추출한다. 추출된 Vector와 Point는 [Plane.ByOriginNormal] node에 Input하여 현재 입단면 View를 치수를 배치할 수 있는 기준 Plane을 만든다.

Grid Curve의 방향을 [Line.Direction] node로 확인한다. 현재 단면 View와 같은 방향을 갖는 Grid 추출을 위해 [Vector.IsParameter] node로 True, Flase 값을 찾는다. True, Flase 값은 [List.FilterByBoolMask]에 Input하여, 현재 View에서 보이는 Grid만 선택한다.

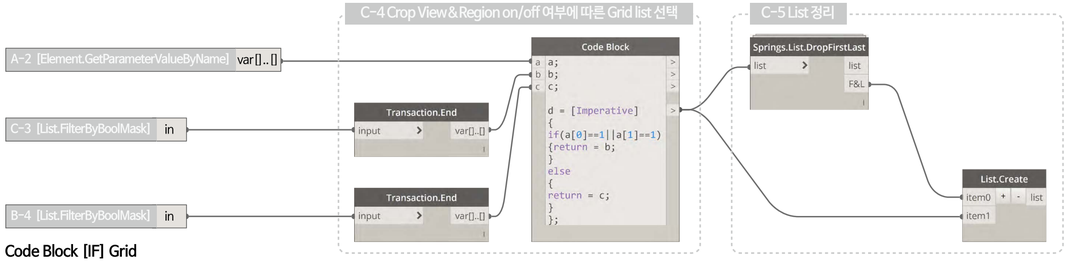
C Dimension을 작성 할 Grid 선택



X열 or Y열 Grid

[B]에 추출한 현재 View에서 보이는 Grid는 [Grid.Curve] node를 이용해 Curve로 전환한다. [Curve.StartPoint]로 추출한 Curve의 Point는 [Springs.PointPullOnToPlane] node를 이용해 [B-4]에서 구한 현재 입단면 View의 Plane에 투영시킨다. 투영된 Point는 Z Point가 0의 값을 갖는다. [Spoonup.Point.ReplaceXYZValue]를 이용해 Z 값을 Crop Box로 생성된 Surface가 갖고 있는 Z값을 Input 한다.

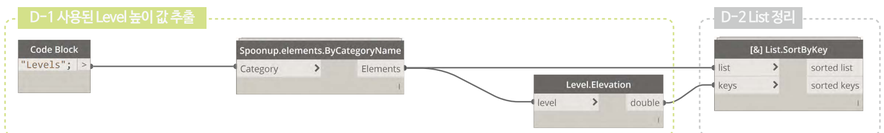
[Geometry.DoesIntersect]를 이용하여 Crop Box로 생성된 Surface와 Surface에 투영된 Grid Curve Point와 간섭체크를 진행한다. 현재의 입단면 View에 Crop Box가 설정되어 있을 경우, Crop Box 안에 들어오는 Grid만 선택하기 위한 방법이다. [Geometry.DeosIntersect]로 추출된 True, Flase 값은 [List.FilterByBoolMask]에 Input하여 Crop Box에 포함된 Gird만 선택한다.



Code Block [IF] Grid

Code Block에 쓰여진 [IF] 조건문이다. [IF] node를 사용할 수 있지만, 다양한 방법을 소개하고자 Code Block을 이용하였다. 현재 입단면 View의 Crop 상태 값을 Code Block의 a에 Input한다. b에는 Crop 상태일 경우 사용된 Crop이 포함하는 Grid를 Input한다. c에는 Crop 설정이 없을 경우, 전체 Grid에 치수가 표현될 수 있도록 현재 View에서 보이는 Grid를 Input한다. View의 Crop 설정에 따라 추출된 Grid는 [List.Create]를 이용해 치수를 표현할 Grid List로 만든다.

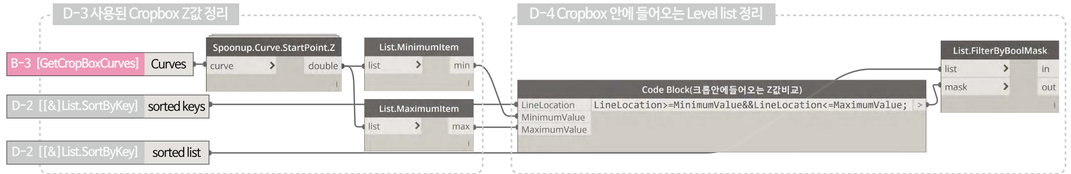
D DIM을 작성 할 Level 선택



Level의 Z값

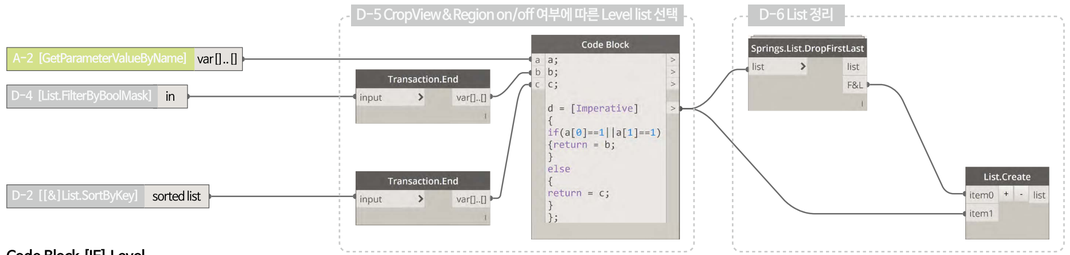
현재 Project에서 사용된 Level을 [Spoonup.element.ByCategoryName] 또는 [Category.ByName]과 [All Elements of Category] node로 정보를 추출하였다. 추출된 Level은 [List.SortByKey] node를 이용하여 오름차순으로 정리한다.

Crop Box의 Curve를 [Spoonup.Curve.StartPointZ] node에 Input하여 사용된 Z값을 추출한다. 추출된 Z Point는 [List.MaximumItem]과 [List.MinimumItem] node를 이용해 Curve가 갖는 최대, 최소의 값을 추출한다.



Crop Box Curve의 Z 값

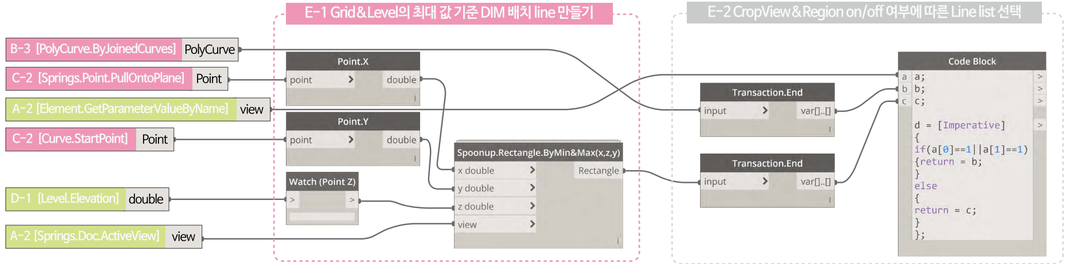
Crop Box에 포함되는 Level을 선별하기 위해서 Code Block을 이용해 Level의 Z값과 Crop Box Curve의 Z값을 직교한다. Crop Box의 최소 Z값보다 크고, 최대 Z값보다 작은 Level을 [List.FilterByBoolMask] node로 선택한다.



Code Block [IF] Level

Code Block에 쓰여진 문구는 [IF] 조건문이다. [IF] node를 사용할 수 있지만, 다양한 방법을 소개하고자 Code Block을 이용하였다. 현재 입단면 View의 Crop 상태 값을 Code Block의 a에 Input한다. b에는 Crop 상태일 경우 사용된 Crop이 포함하는 Level을 Input한다. c에는 Crop 설정이 없을 경우, 전체 Level에 치수가 표현될 수 있도록 Project에 사용된 Level을 Input한다. View의 Crop 설정에 따라 추출된 Grid는 [List.Create]를 이용해 치수를 표현할 Level List로 만든다.

E 기준 Line 작성 및 Dimension 배치



Code Block [IF] Line

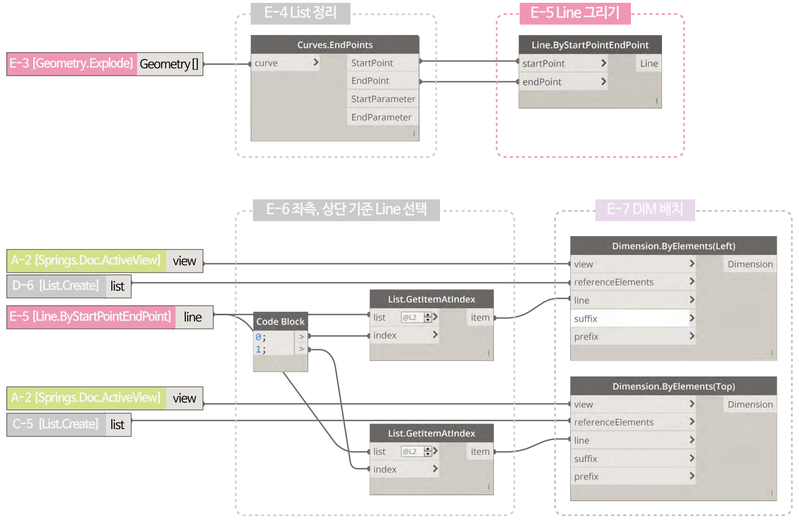
위에 사용된 Node들은 현재 View의 Crop 설정에 따라 DIM을 배치할 기준 Line을 선택하는 Node들이다. 현재 View에 보이는 Grid Curve들의 X,Y 값과 Level이 갖는 Z값을 [Spoonup.Rectangle.ByMin&Max(x,y,z)] node에 Input한다. [Spoonup.Rectangle.ByMin&Max(x,y,z)]는 Input된 Point값을 모두 포함하는 단일 Rectangle Curve를 만든다. 전체 치수 배치는 Grid와 Level의 끝선을 기준으로 배치하기 위해서 [Spoonup.Rectangle.ByMin&Max(x,y,z)] node를 이용하였다.

Code Block [IF]는 [C], [D]에서 사용한 조건문과 동일하다. View Crop 상태에 따라서 Crop Box를 기준으로 치수를 배치할지, 전체 Level과 Grid 기준으로 배치할지 결정한다.



Parameter 설정에 따른 Curve Offset

외곽 Curve는 치수를 배치 할 기준선이기 때문에 [A]에서 설정한 "Grid와 큰치수 간격", "큰치수와 작은 치수 간격"의 값을 넣어 Curve Offset한다. 작성된 Curve들은 치수가 배치될 개별 Curve로 변환하기 위해 [Geometry.Explode] node를 이용하여 만든다.



Dimension 작성

개별로 추출된 Curve의 Start Point와 End Point를 추출한다.

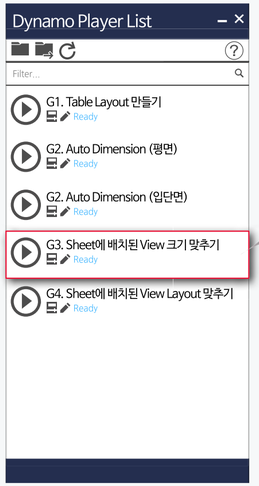
[Line.ByStartPointEndPoint] node를 이용하여 Line Geometry를 만든다. 입단면 치수는 상단에만 표현하는 경우를 고려하여 [List.GetItemAtIndex]로, 치수를 배치할 상단 Line과 좌측 Line을 선택한다. 선택된 Line을 [Dimension.ByElements] node에 Level을 기준으로 표현하는 치수와 Gird를 기준으로 표현할 치수를 구분하여 Input한다.

G3. Sheet에 배치된 View 크기 맞추기

동영상 : https://youtu.be/kY9pf30nvXc

기능 및 필요성

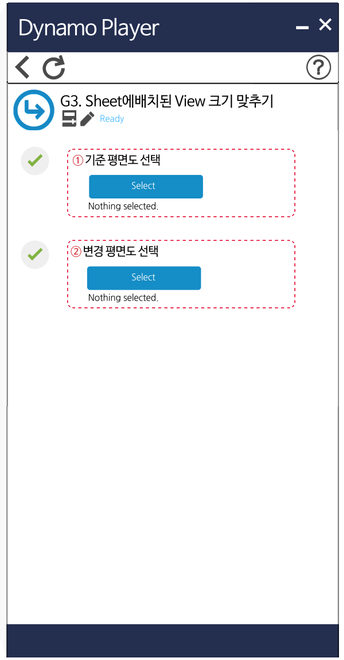
Sheet에 배치된 View의 Crop 크기를 맞추는 기능이다. Scope Box를 사용할 경우 쉽게 View Crop 크기를 맞출 수 있다. 하지만 Scope Box를 사용하지 않는 사용자들은 각 View의 Crop Region을 Edit하고 Line을 복사하여 붙여 넣는 방식, 또는 두 개의 뷰를 겹치게하여 크기를 맞추었다. 확대평면도처럼 같은 크기의 View가 반복되는 도면에서 활용 할 경우, 작업시간을 단축 할 수 있다.



유의사항

- 작업 화면 : Sheet View에서만 구동이 가능하다.

- 선택된 각 View의 Scale이 같은지 확인 할 필요가 있다.
- Scale이 서로 다른 경우, 선택된 기준 평면도 Scale 기준으로 변경된다.
- 서로 다른 Sheet의 View Port를 선택하여도 구동된다.



* Dynamo Player는 Revit 2017부터 사용 가능하며, Player의 설정창은 Revit 2018 이상 버전에서 사용할 수 있다.
Revit 2017일 경우 Dynamo 파일을 열고 설정을 수정하여 저장한 뒤 Player를 누르거나 Dynamo에서 바로 실행시킨다.

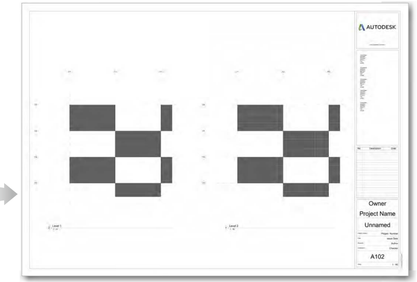
View Sheet 선택을 위한 Parameters 조정

① 기준 평면도 선택
Sheet에 배치된 기준 평면 선택

② 변경 평면도 선택
Sheet에 배치된 변경할 평면 선택



변경전 도면

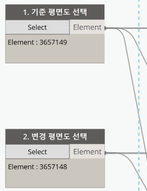


변경후 도면(기준 도면과 View 크기 일치)

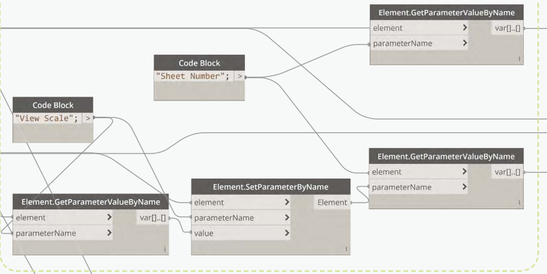
CODE OVERVIEW

A 사용한 Sheet 찾기

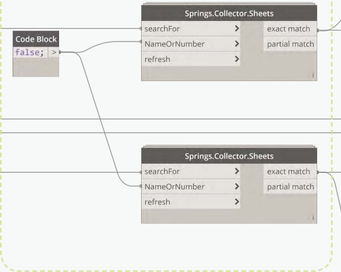
A-1 평면도 선택



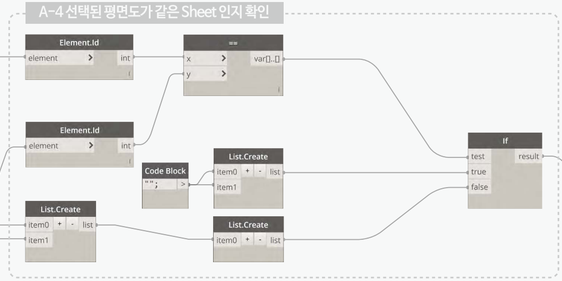
A-2 View Scale, Sheet Number 정보 추출



A-3 배치된 Sheet 찾기

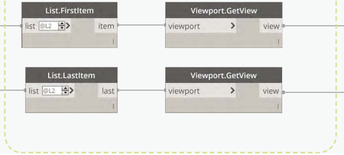


A-4 선택된 평면도가 같은 Sheet 인지 확인

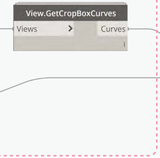


B Crop Box 크기 맞추기

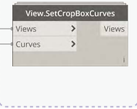
B-1 선택된 Viewport의 View가져오기



B-2 CropBox의 Curve 추출



B-3 Crop Box 재설정

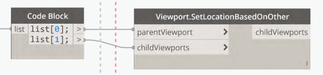


C Viewport 위치 맞추기

C-1 List 정리

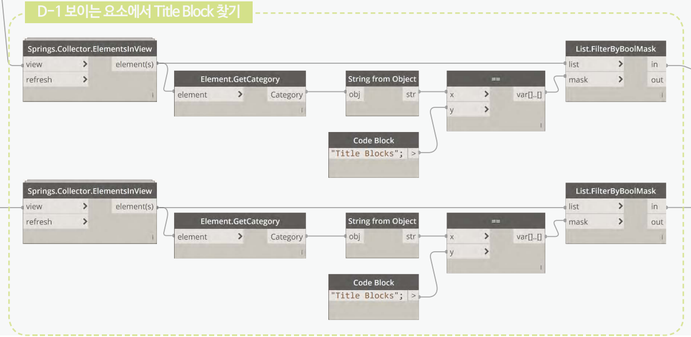


C-2 Viewport 위치 재설정

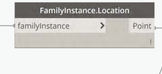


D Title block 위치 맞추기

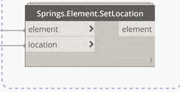
D-1 보이는 요소에서 Title Block 찾기



D-2 기준 Sheet의 Title block 위치 추출



D-3 Title Block 위치 맞추기



Key Node

custom node **Clockwork Packages**

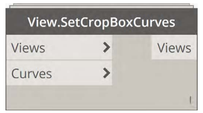


Input한 요소의 위치를 변경한다.
Line 기반 및 Point 기반 요소 모두에서 작동한다.

INPUTS
- Element
  위치를 변경 할 요소의 객체를 선택한다.
- Location
  위치를 변경할 좌표값 혹은 view에서의 위치를 입력한다.

OUTPUTS
- Element
  위치가 변경된 요소를 확인한다.

custom node **MEPover Packages**



View에서 설정된 자르기 영역의 Curve 값을 변경 한다.

INPUTS
- Views [View 객체]
  Crop Box가 활성화 되어있는 View를 가져온다.
- Curves [Curve 객체]
  Crop Box Curves를 추출 한다.

OUTPUTS
- Views [View 객체]
  추출된 Crop Box Curves를 View Port에 적용 시킨다.

custom node **Clockwork Packages**

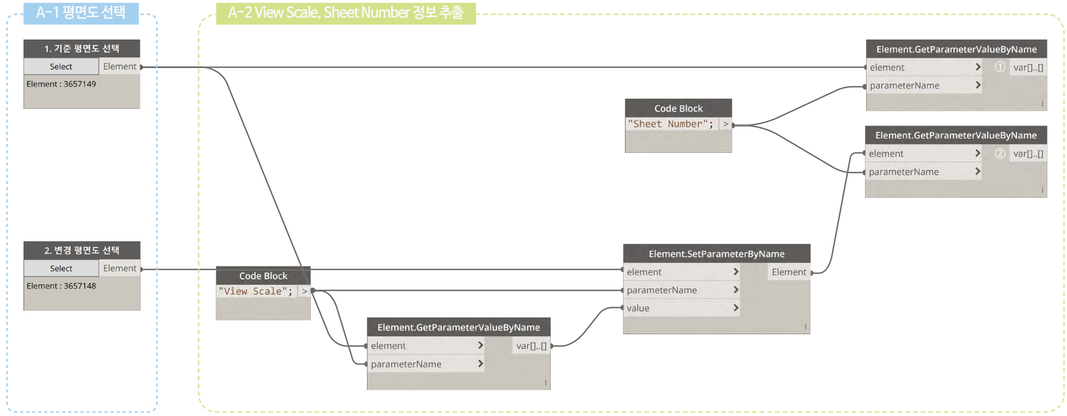


기준이 되는 Viewport(parentViewport)의 중심점으로 변경이 될 Viewport(childViewports)를 이동 시킨다.

INPUTS
- Parent Viewport
  기준이 될 Viewport를 연결 한다.
- Child Viewports
  위치를 이동 시킬 Viewport를 연결 한다.

OUTPUTS
- Child Viewports
  이동한 Viewport들의 값을 확인 한다.

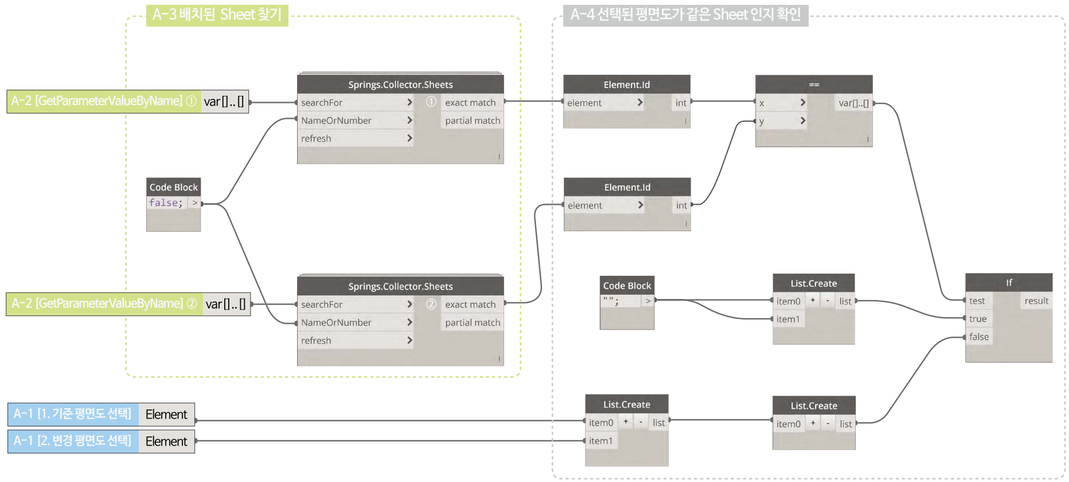
A 사용한 Sheet 찾기



선택된 평면도의 Scale, Sheet Number 정보 추출

선택된 기준 평면도의 Scale 값을 불러오기 위해서 [GetParameterValueByName] node를 활용하였다. [GetParameterValueByName] node에서 Input하는 Parameter Name은 객체의 Parameter 명이 Revit의 Type Property에서 보이는 Text와 일치할 경우 해당되는 값을 Dynamo로 가져올 수 있다.
기준 평면도에서 설정된 View Scale을 [GetParameterValueByName] node로 추출하고 [Element.SetParameterByName] node를 이용하여 변경할 평면도의 스케일 값에 넣어준다.

두개의 평면도가 같은 Sheet에 배치되어 있는지 판단하기위해서, 우선 "Sheet Number" 정보를 [GetParameterValueByName]을 이용하여 기준 평면도와 변경할 평면도의 "Sheet Number" 값을 추출하였다.

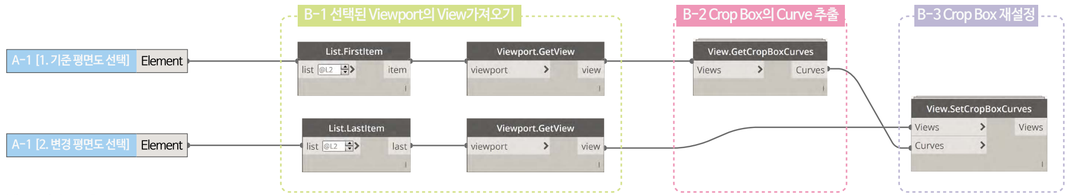


배치된 Sheet

선택된 두개의 View가 동일한 Sheet에 배치되어 있는지 확인이 필요하다. 만일, 서로 다른 Sheet에 배치되어있을 경우 Title Block의 위치와 View의 위치까지 맞출 수 있도록 Node를 추가 해야 한다. [Springs.Collector.Sheets]는 선택한 평면들이 배치된 Sheet Number를 이용하여 배치된 Sheet를 선택하게 만들어준다.

Sheet Number를 통해서 같은 Sheet에 위치하는지 확인 할 수 있지만 Sheet에 배치된 View 좌표, Title Block 등 정보를 이용하기 위해서 [Springs.Collector.Sheets] node로 View가 배치되어 있는 Sheet를 선택한다. [A-4] [IF] node에서 두개의 평면도가 같은 Sheet에 배치 되었는지, 서로 다른 Sheet인지 판별 한다.

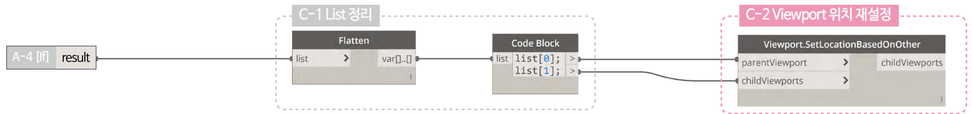
B Crop Box 크기 맞추기



View의 Crop Box

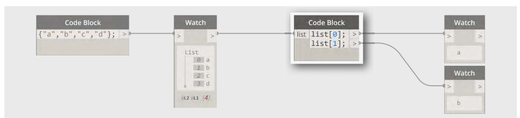
Viewport는 기존에 존재하는 View를 Sheet에 배치 하면 Viewport로 설정이 적용 된다. Viewport가 되어야 View의 Sheet 정보, Scale, View Title 등을 활용할 수 있게 된다. Sheet View에서 View를 바로 접근하지 못하고 Active View를 통해서 컨트롤 할 수 있는 것도 같은 이유다. [View.GetCropBoxCurves] node를 이용해 추출된 기존 평면도의 Crop Box Curve는 [View.SetCropBoxCurves]를 통해 변경될 View의 CropBox 설정에 적용 시킨다. [View.GetCropBoxCurves]와 [View.SetCropBoxCurves]이 두개의 Node를 이용하면 원하는 View의 Crop Box Curve를 쉽게 추출하고, 적용시킬 수 있다.

C Viewport 위치 맞추기

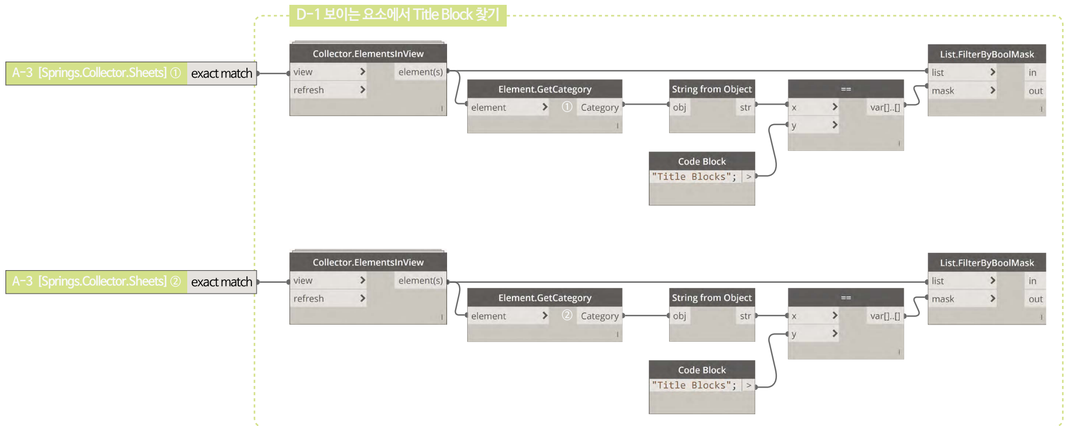


Viewport의 위치 설정

[Viewport.SetLocationBasedOnOther]은 parent Viewport에 기준이되는 Viewport 값을 넣고, child Viewport에는 변경이 될 Viewport를 넣는다. 이 Node는 parent Viewport에 Input 한 Viewprot의 중심을 기준으로 child Viewport의 위치를 맞춰준다. Input한 두개의 Viewport의 View Crop Box 크기가 일치하지 않거나 Annotation Crop이 설정된 경우 Viewprot의 중심점이 다르게 인식되어 위치가 서로 다르게 배치 될 수 있다.



D Title block 위치 맞추기

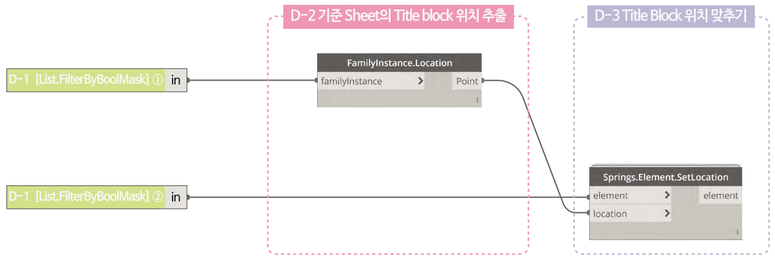


Title Block

[D. Title Block 위치 맞추기]에서 보이는 node들은 기준 Sheet에 배치된 여러개의 객체중에서 Title Block의 객체만 선택하기 위한 Node들이다. Package Node를 이용하여 Node들을 단순하게 만들 수 있지만, Autodesk BIM Guidebook Dynamo Part1에서는 가능한 다양한 방법으로 표현하기 위해 [List.FillerByBoolMask] 기능을 활용 하였다.

[A-3]을 통해 선택된 Sheet는 [Collector.ElementsInView] node를 통해 Sheet에 배치된 모든 부재를 확인한다. [GetCategory] node로 연결된 부재의 Category 정보를 추출하여 Category name이 'Title Blocks'와 같은지 판별한다. 추출된 Category name은 Dynamo에서 Object의 종류로 인식된다. Category Name명을 Text 형식으로 변경하기 위해 [String from Object] node를 이용한다.

[List.FilterByBoolMask] node는 Mask에 Input된 True, False 값으로 List를 정리 시킨다. Output date에서 In은 True값인 list만 추출되고, Out은 False값인 List만 추출된다.



Title Block의 위치 설정

"Title Block" 부재의 위치가 서로 다른 경우 Viewport의 크기를 맞추고, Viewport 위치를 같은 좌표에 배치하여도 View의 위치가 다르게 배치된 것처럼 보인다. Viewport 뿐만 아니라, 'Title Block'의 위치도 동일한 좌표로 맞춘다.

[FamilyInstance.Location]으로 추출된 기준 View의 'Title Block'의 좌표를 [Springs.Element.SetLocation]에 Input하여 변경할 View의 'Title Block'의 위치를 기준 View와 동일하게 맞춘다.

TIP&TRICK

G4. Sheet에 배치된 다중 View Layout 맞추기

동영상 : https://youtu.be/iL_e82bKWWM

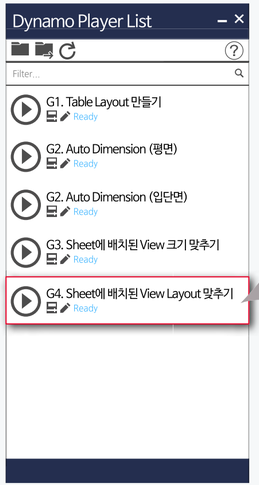
기능 및 필요성

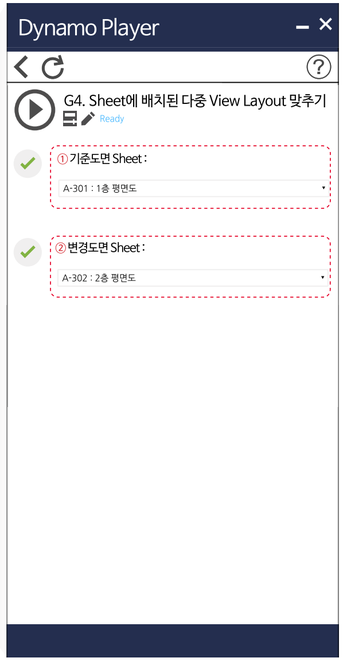
서로 다른 Sheet에 배치된 도면들을 선택된 기준 Sheet를 기준으로 Layout을 정리 시켜주는 기능이다. 기존 Sheet 작업시 도면간 View의 위치를 맞추기 위해 Guide Grid 기능을 활용 하거나 Title Block에 가시성 Line을 만들어 Sheet내의 도면들의 위치를 맞추었다. 'G3. Sheet에 배치된 View 크기 맞추기'와 마찬가지로 Viewport의 중심점을 기준으로 View가 배치된다.

유의사항

- 작업 화면 : View와 상관없이 설정 창에서 지정한 Sheet를 기준으로 구동된다.

- Detail Number 기준으로 순서를 매칭시켜서 작성된다.
- 기준 Sheet에 배치된 View의 Scale이 다를 경우 기준 View의 Scale을 기준으로 변경 Sheet에 배치된 View의 Scale이 변경된다.



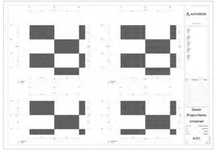


* Dynamo Player는 Revit 2017부터 사용 가능하며, Player의 설정창은 Revit 2018 이상 버전에서 사용할 수 있다.
Revit 2017일 경우 Dynamo 파일을 열고 설정을 수정하여 저장한 뒤 Player를 누르거나 Dynamo에서 바로 실행시킨다.

다중 View Layout을 위한 파라미터 조정

① 기준 도면 Sheet
기준 Layout으로 지정 할 Sheet 선택

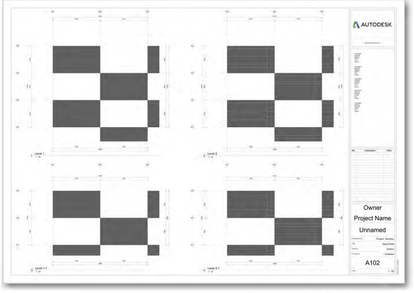
② 변경 도면 Sheet
기준에 맞게 변경 할 Sheet 선택



기준 도면 Sheet



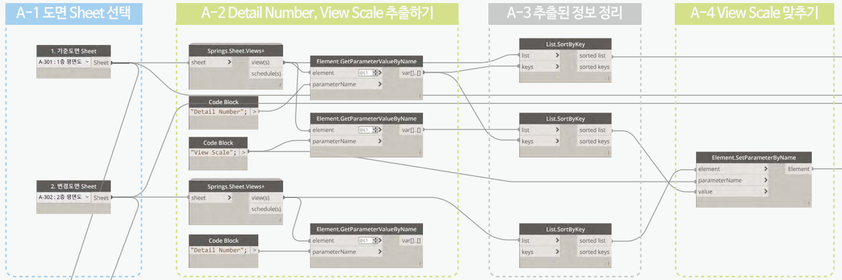
변경전 도면 Sheet



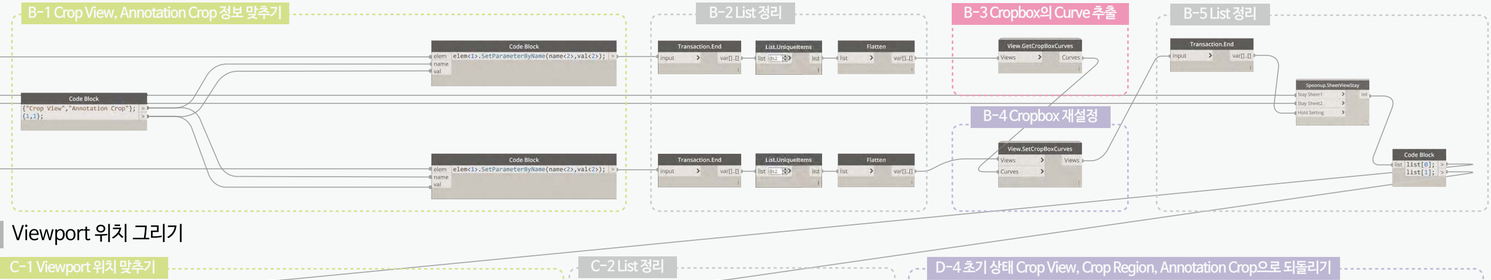
변경후 도면 Sheet(기준 도면과 형식 일치)

CODE OVERVIEW

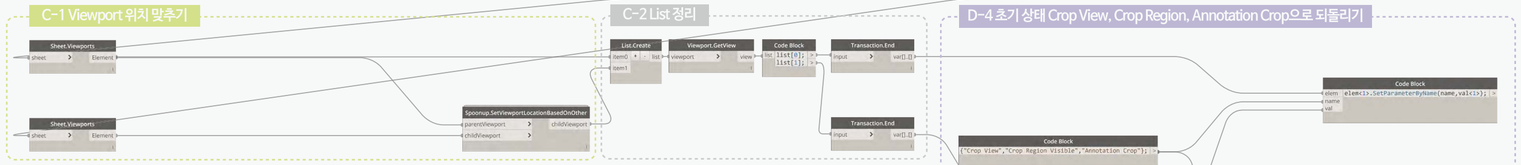
A Sheet에 배치된 다중 View의 Scale 맞추기



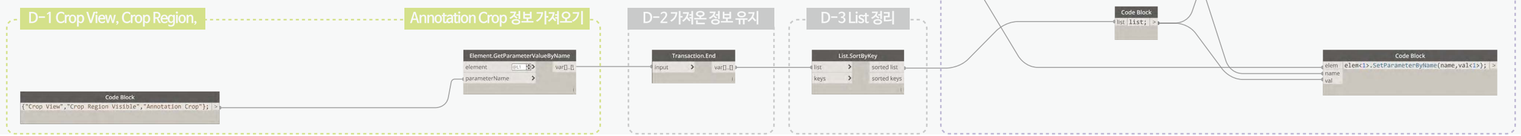
B Crop Box 맞추기



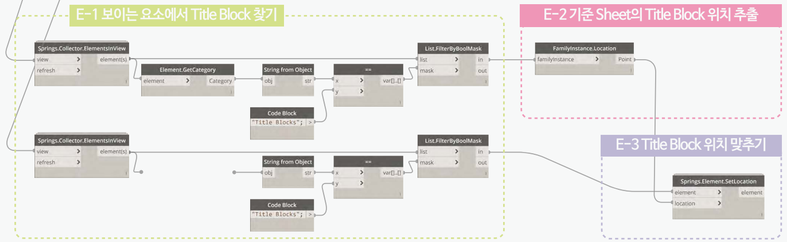
C Viewport 위치 그리기



D Crop view & Annotation crop on/off 초기 상태 복원

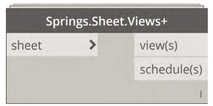


E Title block 위치 맞추기



Key Node

custom node **Spring nods Packages**



가져온 Sheet에 배치된 모든 Views 및 Schedules를 추출한다.

INPUTS
- Sheet
  View와 Schedule을 확인 할 Sheet를 불러온다.

OUTPUTS
- View(s)
  Sheet에 배치된 View를 불러온다.
- Schedule(s)
  Sheet에 배치된 Schedule를 불러온다.

Dynamo basic node

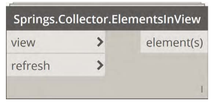


Sheet에 배치된 Viewport들을 확인한다.

INPUTS
- Sheet
  배치된 Viewport을 확인 할 Sheet를 선택한다.

OUTPUTS
- Element
  배치된 Viewport를 확인 한다.

custom node **Spring nods Packages**

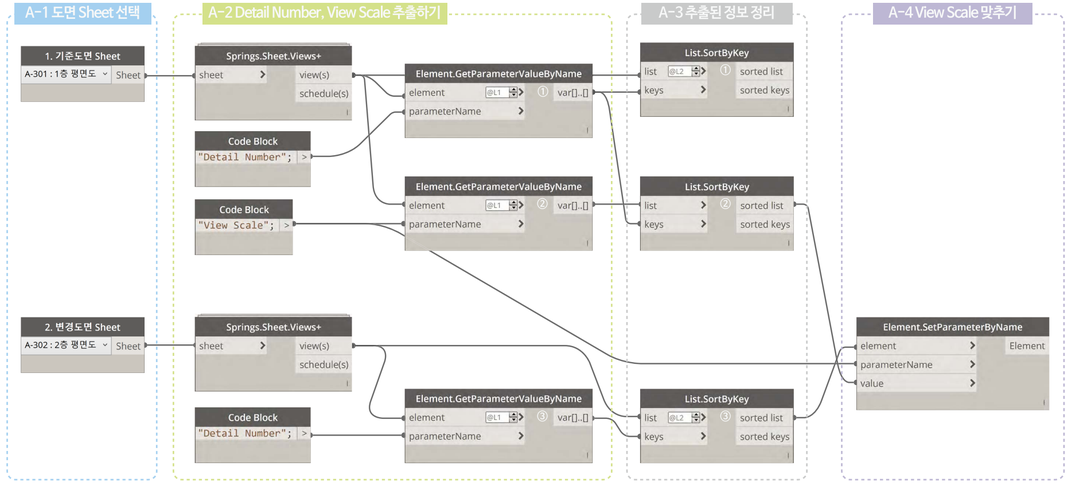


뷰에 표시된 모든 요소를 선택한다.

INPUTS
- View
  배치된 요소를 확인 할 View를 선택한다.
- Refresh
  모델링 업데이트시 Bool node를 연결하고, True를 선택하여 Output 값을 업데이트 한다.

OUTPUTS
- Element(s)
  View에 배치된 모든 요소를 확인 한다.

A Sheet에 배치된 다중 View의 Scale 맞추기

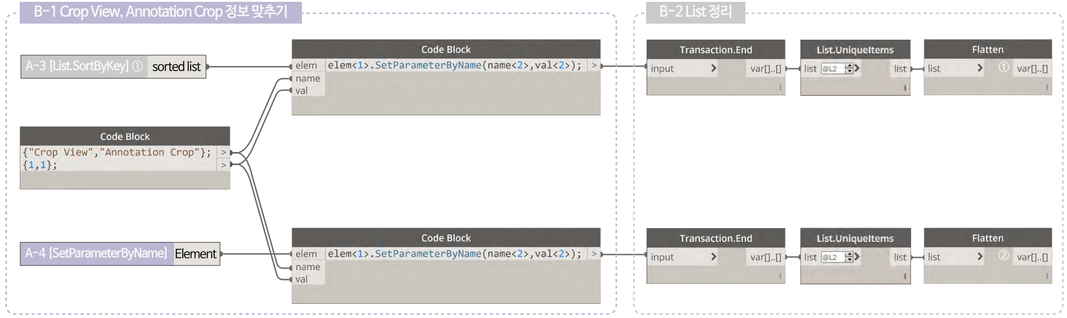


Sheet에 배치된 View 정보

[Springs.Sheet.Views+]를 활용하면, 선택한 Sheet에 배치된 View를 확인 할 수 있다. [GetParameterValueByName]에 Revit Parameter인 "Detail Number"의 "View Scale"을 지정하며 Sheet에 배치된 View들의 "Detail Number"의 "View Scale" 값을 얻을 수 있다.

"Detail Number"를 추출하는 이유는 Sheet에 다중 View가 배치되어있기 때문에 배치된 View를 Detail Number를 기준으로 List를 정리한다. 추출한 "View Scale" 값은 [List.SortByKey]를 활용하여 "Detail Number" 기준으로 정리한다. "Detail Number" 기준으로 정리된 "View Scale"은 [SetParameterByName]을 통해서, 변경 할 Sheet에 배치된 View에 적용 시킨다.

B Crop Box 맞추기

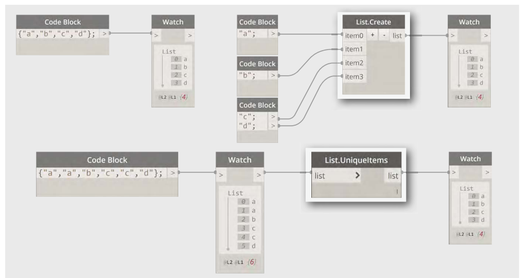


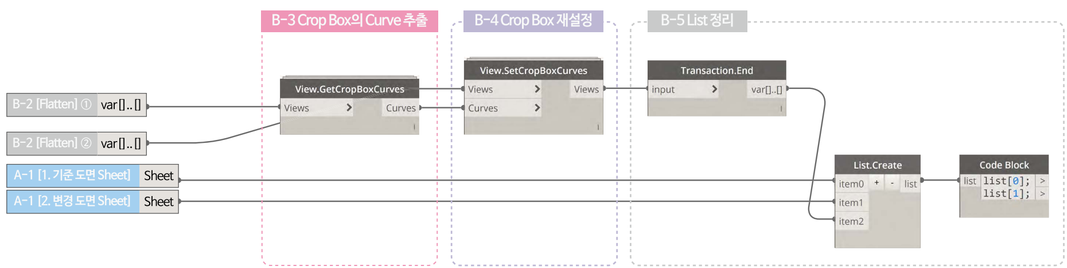
View의 Crop 설정

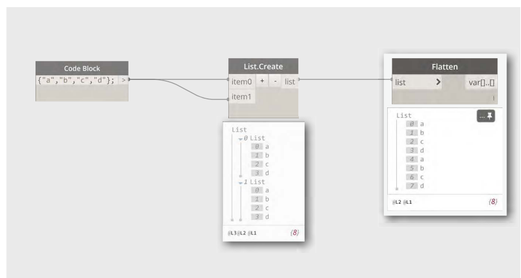
기존 View의 "Crop View" "Annotation Crop"의 On/Off 상태에 따라서 Viewport 크기가 결정된다. View의 위치는 Viewport의 중심점을 기준으로 맞추기 때문에 각 Sheet에 배치된 "Crop View"와 "Annotation Crop"의 설정을 [SetParameterByName] node를 이용하여 맞춘다. Sheet에 배치된 다중 View일 경우 Crop이 되어있을 가능성이 많기 때문에, Parameter의 값은 '1'로 설정한다. Parameter 체크박스의 값은 (□=0, ■=1)이다. Code Block에서 "Crop View"의 의미는 Text =Crop View를 뜻하며, 다양한 값들을 리스트화 시킬때는 {"A","B","C"}를 활용한다.

List의 중복된 값 정리

[List.UniqueItems]는 List에서 중복되는 값을 소거하는 기능을 한다.
[SetparameterByName]에서 2개의 Parameter 값을 주기 위해서 반복되었던 List 값을 [List.UniqueItems]으로 정리 하였다. [Flatten]은 List의 값들이 list 차수에 따라서 변경되기 때문에 정확한 값을 얻기위해 [Flatten] node를 이용하여 단일 List로 정리 하였다.



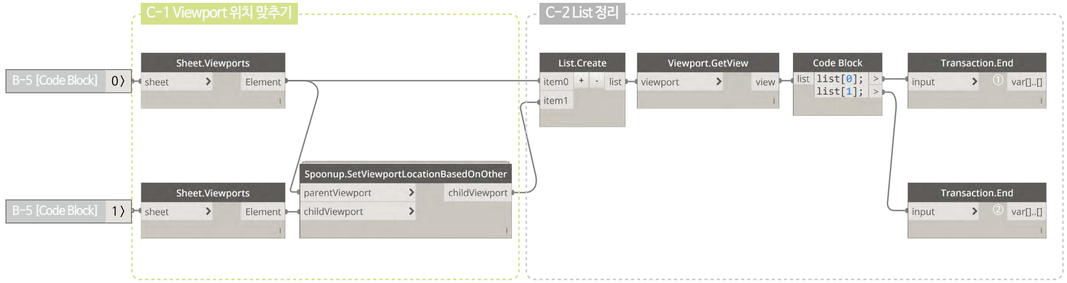




Transaction. End

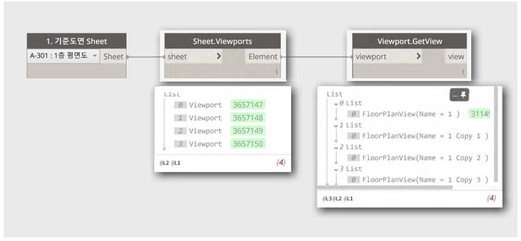
Dynamo에서 작성된 Node들은 "Run"버튼의 실행과 함께 동시에 값들이 계산되어 나온다. [Transaction.End]는 진행되는 Process에 순서를 주기 위해 이용하였다. View에 설정된 Crop과 Annotation 상태를 On한 뒤, [C]에서 Viewport 위치 맞추기가 진행된다. 하지만, [D]에서는 원래 View에 설정된 Crop과 Annotation 상태로 되돌리기 위한 node들이 있다. [B], [D]에서 모든 View의 "Crop View", "Annotation Crop" Parameter 값을 1로 설정하기 때문에, [Transation.End]를 사용하여 On되는 설정 값이 현재 node에서 한번 사용된 뒤 node를 종료시키고, 다시 사용할 수 있게 값을 반환시키는 기능을 한다.

C Viewport 위치 그리기

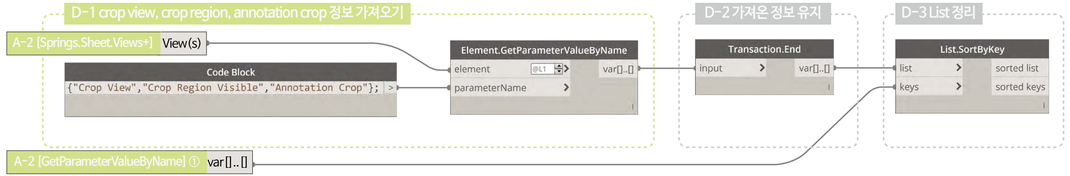


Viewport의 위치설정

변경할 Sheet에 배치된 View들은 [Spoounp.SetViewportLocationBasedOnOther] node을 통하여 기준 Sheet에 배치된 Viewport들의 중심점을 기준으로 위치를 이동시킨다. 이동된 Viewport와 기준이 되는 Viewport들 [List.Create]를 이용하여 단일 List로 만든다. 작성된 List는 [Viewport.GetView] node를 이용하여 Viewport가 안의 View정보를 가지고 온다. 기준 Sheet의 View와 변경할 Sheet의 View들은 [Transaction.End]를 활용하여, Dynamo에서 작업된 현재 상태를 종료하고, 연산을 다시 시작하게 만든다.



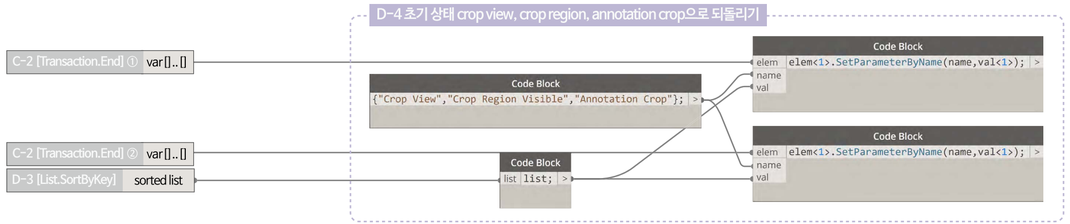
D Crop View & Annotation Crop on/off 초기 상태 복원



View의 Crop 정보

선택된 Sheet등의 "Crop View", "Annotation Crop"의 설정을 되돌리기 위해 초기 상태 정보를 [GetParameterValueByName]을 이용하여 가져온다.

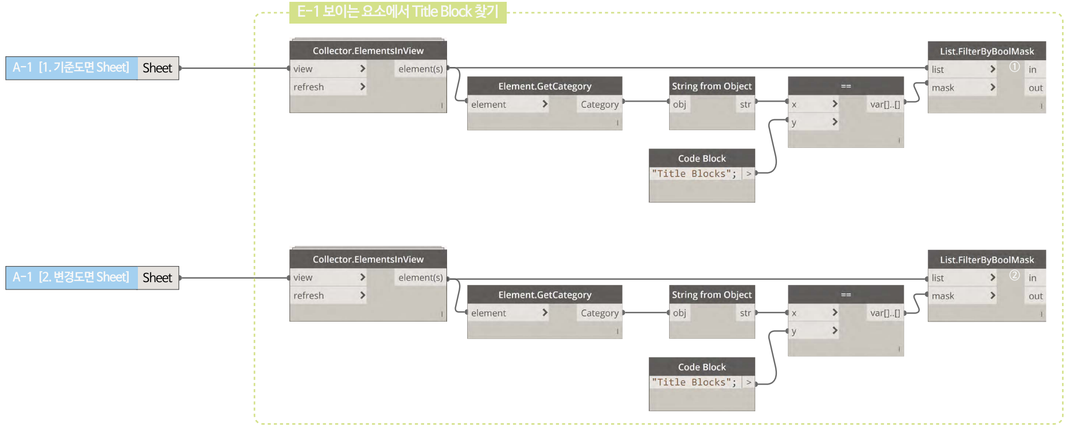
추출된 정보들은 [List.SortByKey] node를 이용하여 배치된 "Detail Number" 기준으로 List를 정리한다.



View의 Crop설정

[List.SortByKey]로 정리된 List를 [SetParameterByName] Code Block을 이용하여 초기 상태로 되돌린다. [Transaction.End]를 사용하지 않았다면 Viewport 크기를 맞추기 위해 설정한 값들이 [D]에도 적용되어 초기상태로 되돌리기 어렵다. Dynamo에서 같은 Parameter 값을 이용하여 node를 작성할 때 연산 순서를 위해서 [Transation.End]를 활용하는 것을 추천한다.

E Title block 위치 맞추기

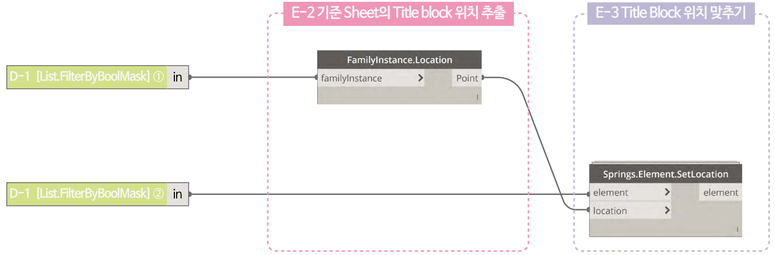


Title Block

[D. Title Block 위치 맞추기]에서 보이는 Node들은 기준 Sheet에 배치된 여러개의 객체중에서 Title Block의 객체만 선택하기 위한 Node들이다. Package Node를 이용하여 Node들을 단순하게 만들 수 있지만, Autodesk BIM Guidebook Dynamo Part1에서는 가능한 다양한 방법으로 표현하기 위해 [List.FilterByBoolMask] 기능을 활용 하였다.

[A-3]을 통해 선택된 Sheet는 [Collector.ElementsInView] node를 통해 Sheet에 배치된 모든 부재를 확인한다. [GetCategory] node로 연결된 부재의 카테고리 정보를 추출하여 Category Name이 ["Title Blocks"]와 같은지 판별한다. 추출된 Category Name은 Dynamo에서 Object의 종류로 인식된다. Category Name명을 Text 형식으로 변경하기 위해 [String from Object] node를 이용한다.

[List.FilterByBoolMask] node는 Mask에 Input된 True, False 값으로 List를 정리 시킨다. Output Date에서 In은 True값인 list만 추출되고, Out은 False값인 List만 추출된다.



Title Block의 위치 설정

"Title Block" 부재의 위치가 서로 다른 경우 Viewport의 크기를 맞추고, Viewport 위치를 같은 좌표에 배치하여도 View의 위치가 다르게 배치된 것처럼 보인다. Viewport 뿐만 아니라, "Title Block"의 위치도 동일한 좌표로 맞춘다.

[FamilyInstance.Location]으로 추출된 기준 View의 "Title Block"의 좌표를 [Springs.Element.SetLocation]에 Input하여 변경할 View의 "Title Block"의 위치를 기준 View와 동일하게 맞춘다.

TIP&TRICK

2. Dynamo Code
건축 도면

A1. 단열/방수/방화 Filled Region 만들기

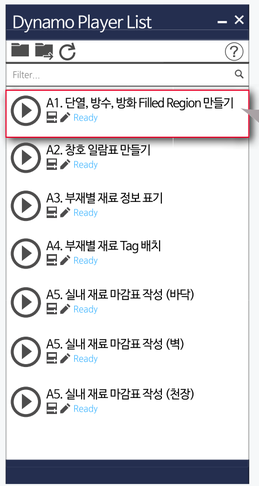
동영상 : https://youtu.be/iUPByy_zOMY

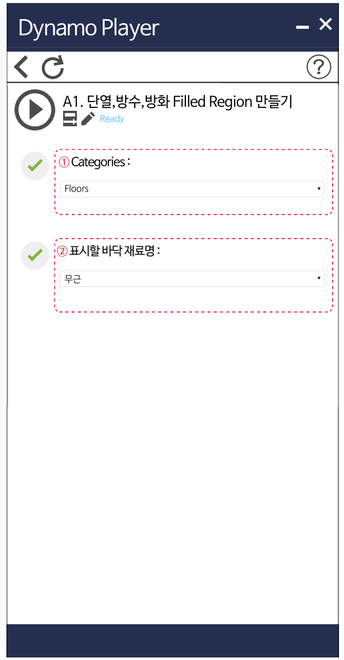
기능 및 필요성

단열/방수/방화 구역의 면적 산출 및 근거도면 작성시에 활용된다. 기존의 도면 작성방식은 도면 View에 Filter을 지정하여 작성하였지만, 면적을 추출하고 도면안에 표현의 위계를 부여하는 것이 불편하여 Filled Region으로 다시 작성하는 경우가 있다. Filled Region을 이용할 경우 도면의 위계 표현을 쉽게 부여할 수 있고, Tag를 만들어 면적 정보를 도면에 쉽게 표기 할 수 있다. 작성된 Dynamo Code를 이용하면 바닥을 기준으로 사용된 재료명을 검색하여 해당 바닥위에 Filled Region을 그려준다.

유의사항

- 작업 화면 : 3D View와 입단면 View를 제외한 평면 View에서 구동이 가능하다.
- Dynamo Player로 한번 작성된 부재는, Player의 설정 값을 변경하여 재실행 하여도 삭제되지 않고 그대로 유지된다.
- 여러번 실행시킬 경우 같은 위치에 다중의 부재가 생성된다.
- 바닥 부재를 구성하는 Layer에 재료를 명확히 설정한다.



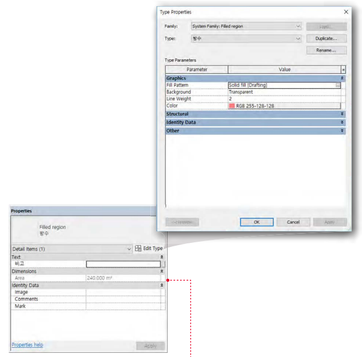


* Dynamo Player는 Revit 2017부터 사용 가능하며, Player의 설정창은 Revit 2018 이상 버전에서 사용할 수 있다.
Revit 2017일 경우 Dynamo 파일을 열고 설정을 수정하여 저장한 뒤 Player를 누르거나 Dynamo에서 바로 실행시킨다.

Filled Region 작성을 위한 파라미터 조정

① Categories :
Filled Region으로 표현할 재료가 있는 부재를 선택

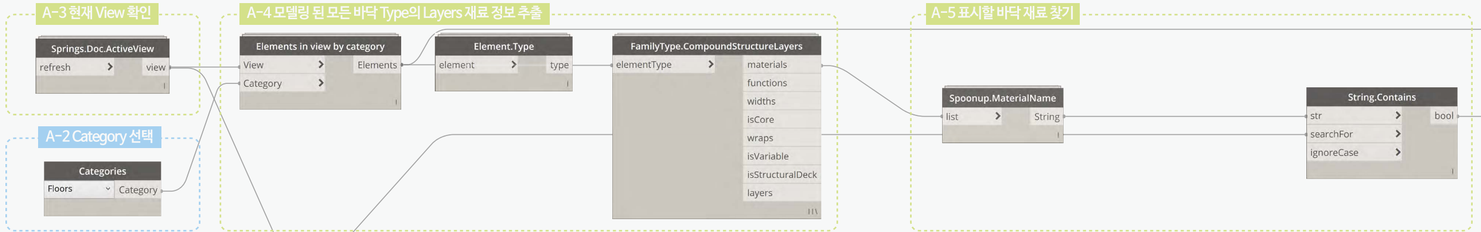
② 표시할 바닥 재료명 :
부재에서 찾을 재료명을 기입



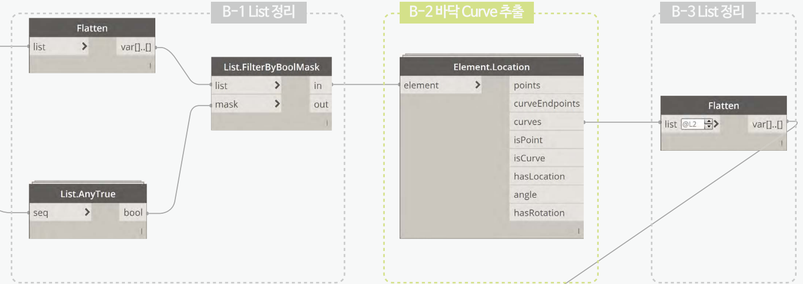


CODE OVERVIEW

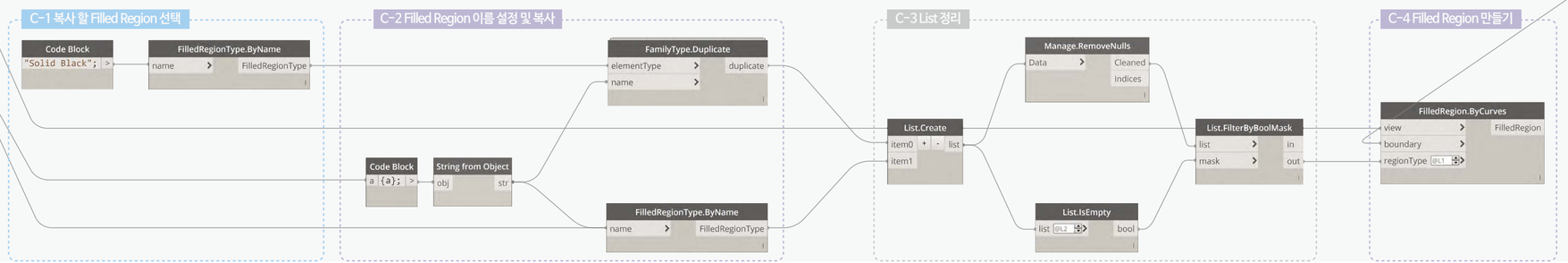
A 바닥 재료명 입력 및 정보 추출



B 선택된 바닥 Curve 추출



C Filled Region 선택 및 만들기



Key Node

custom node **MEPover Packages**

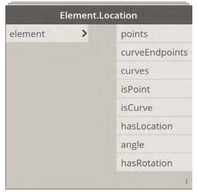


선택한 View에서 지정한 Category의 모든 요소를 추출 한다.

INPUTS
- View
  요소를 찾을 View를 연결 한다.
- Category
  선택할 요소의 Category 타입을 연결 한다.

OUTPUTS
- Elements
  선택한 View에서 보이는 부재 중 지정한 Category Type에 속한 요소를 확인한다.

custom node **Clockwork Packages**

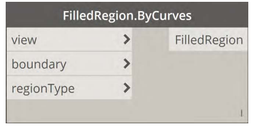


선택된 부재의 위치를 확인한다.

INPUTS
- Element
  위치를 확인 할 요소를 선택 한다.

OUTPUTS
- points, curveEndpoints, curves, isPoint, isCurve, hasLocation, angle, hasRotation
  Outputs의 요소 위치를 좌표 값으로 추출 한다.

Dynamo basic node

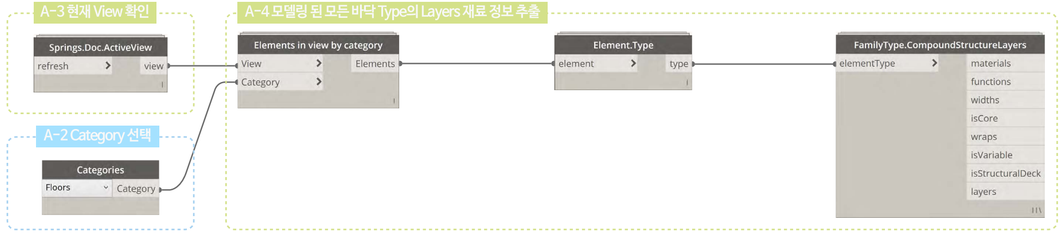


Revit의 FilledRegion 요소를 Curves로 구성한다.

INPUTS
- view, boundary, regionType
  Filled Region을 배치할 view, 요소의 boundary, Type을 설정 한다.

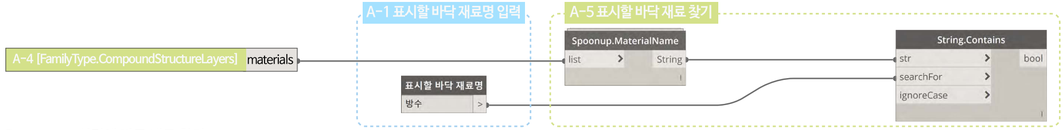
OUTPUTS
- Filled Region
  선택한 view, 선택한 요소의 boundary, regionType을 설정하여 Filled Region을 형성 한다.

A 바닥 재료명 입력 및 정보 추출



Floor의 Type 정보

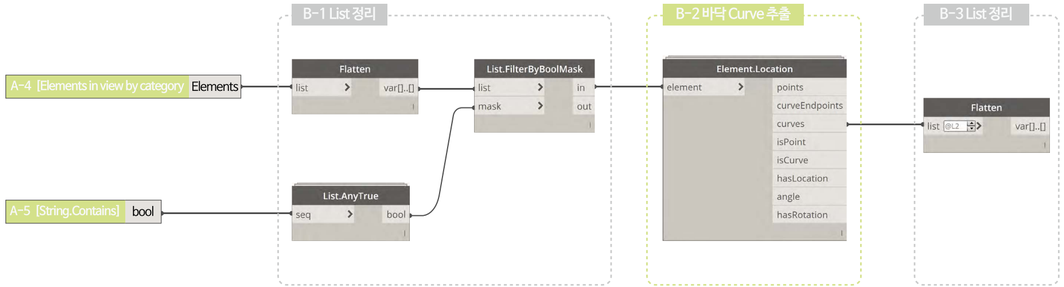
[Springs.Doc.ActiveView] node를 이용하여 Filled Region을 배치할 View의 정보를 Dynamo로 가지고 온다. 현재 View와 Floors Category를 [Elements in view by category] node에 Input하여 현재 View에 보이는 모든 바닥 부재를 선택한다. 선택된 부재는 Family Instance이기 때문에 재료 정보를 얻기 위해 [Element.Type] node를 이용하여 부재를 Type을 변환한다. 이때 List로 정리된 Type은 현재 View에서 보이는 모든 Floor의 Type이다. 정리된 Floor Type List는 [FamilyType.CompoundStructureLayers]에 Input하면, Type을 구성하는 Layer들의 재료, 기능, 두께 등의 정보를 얻을 수 있다.



Parameter 설정에 따른 재료 찾기

[FamilyType.CompoundStructureLayers]는 바닥 이외에도 벽, 천장, 지붕 등 Layer로 구성된 Family Type의 정보를 불러올 때 사용할 수 있다. [Spoonup.MaterialName]은 추출된 Material ID를 Material Name으로 변경해 준다. Dynamo의 기본 node인 [Material.name]은 재료를 지정하지 않은 경우 오류가 발생한다. [Spoonup.MaterialName]은 재료를 지정하지 않았을 경우 '재료 없음'으로 표현되어 재료를 지정하지 않아도 오류가 발생하지 않는다. [String.Contains]는 List에서 찾고자 하는 특정 Text를 Input하면 해당 List에 포함되어 있는지를 판단을 해준다. Output은 True, False로 표현 되기 때문에 [List.FilterByBoolmask]와 바로 연결 가능하다.

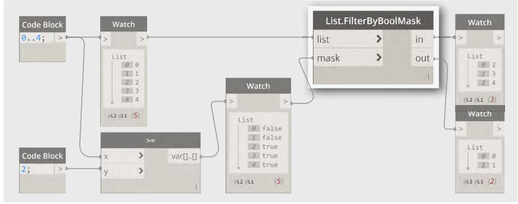
B 선택된 바닥 Curve 추출



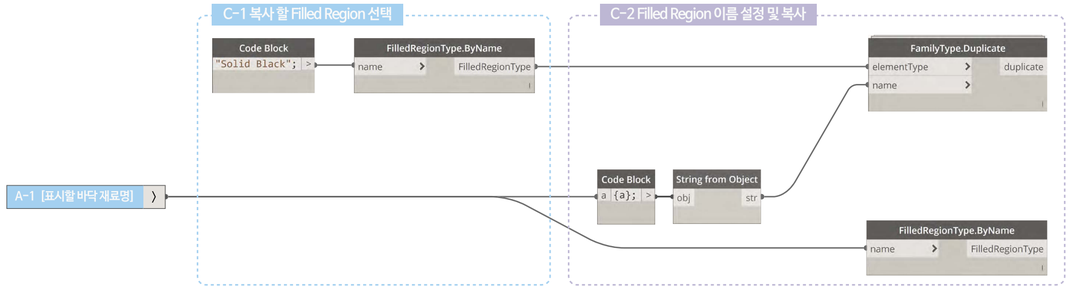
베치된 Floor 부재의 위치정보

현재 View의 Floor Type중 Filled Region으로 표현 하고자 하는 바닥재료명을 검색하고, 검색어가 포함된 Floor 부재를 선별하기 위해 [List.AnyTrue]와 [List.FilterByBoolmask]를 이용하였다. [A-4]에서 [FamilyType.CompoundStructureLayers] node에서 확인한 재료명은 Floor Type을 구성하고 있는 Layer 별로 정보가 나오기 때문에 Layer 중 하나의 Layer에서 찾고자 하는 재료가 포함되어 있으면 포함된 Floor 부재를 선택할 수 있도록 [List.AnyTrue]를 사용하였다.

[List.Filter.ByBoolmask]를 통해 선택된 Floor 부재를 [Element.Location] node를 통해서 Floor의 Boundary를 Curve로 추출한다. [Element.Location]은 부재를 구성하고 있는 Points, Curves, Curve Endpoints, Angle 등 부재가 배치된 좌표를 확인 할 수 있다. Input Data에는 부재의 Type이 아닌 Instance 부재를 연결해야 한다.



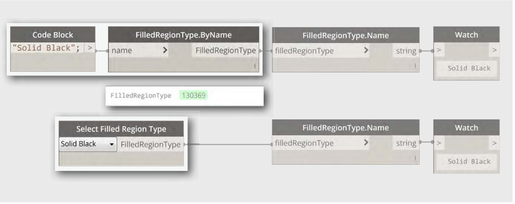
C Filled Region 선택 및 만들기

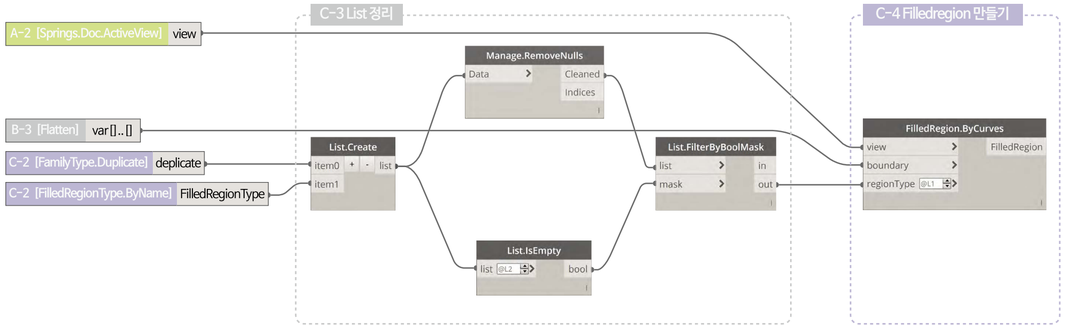


Filled Region Type 작성

바닥에 표현될 Filled Region의 Fill Pattern을 쉽게 제어하고 도면의 범례로 활용하기 위해, 찾고자 하는 재료명으로 Filled Region Type을 만든다. [Code Block]의 "Solid Black"은 Revit에서 기본으로 탑재되어 있는 Filled Region이다. [FilledRegionType.ByName]으로 "Solid Black" Filled Region Type을 선택한다.

[FamilyType.Duplicate] node를 이용하여 검색한 재료명을 Type으로 갖는 Filled Region을 생성한다. "Solid Black" Filled Region을 복사하여 만들어지기 때문에 새로 생성된 Filled Region의 Fill Pattern 설정은 사용자가 원하는 표현 방식으로 변경해야 한다. [C-2]에서 [FilledRegionType.ByName] node에 검색한 재료명을 연결하여 기존에 만들어진 Filled Region Type과 중복되는 것을 방지한다.



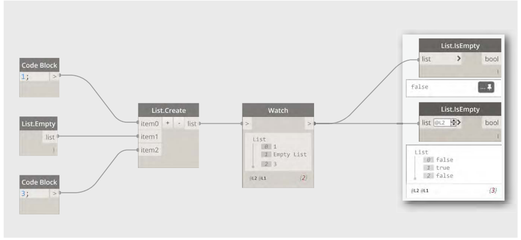


Filled Region 작성

복사된 Filled Region과 기존의 Filled Region List를 [List.Create]로 합친다. 기존에 Filled Region Type이 있다면 생성되지 않아 [List.Creat]에서 [List.Empty]값으로 추출된다. 기존에 생성된 Filled Region을 선택하기 위해서 [List.Empty] node를 이용하여 True, False 값을 얻는다. 추출된 True, False 값으로 [List.FilterByBoolMask]에서 사용할 Filled Region을 선택한다.

[FilledRegion.ByCurves]는 Filled Region을 배치할 View, 포함된 바닥의 Boundary, 그리고 Filled Region Type을 연결하면, Filled Region이 작성된다.

업데이트 시, 배치된 Filled Region을 삭제하고, Dynamo Player를 실행시켜야 한다. 만들어진 Code는 기존에 배치된 Filled Region은 수정하는 방식이 아니라 새롭게 아이템을 배치하는 방식이기 때문이다.



MEMO

TIP&TRICK

A2. 창호 일람표 만들기

동영상 : https://youtu.be/7fLKfF7dpR0

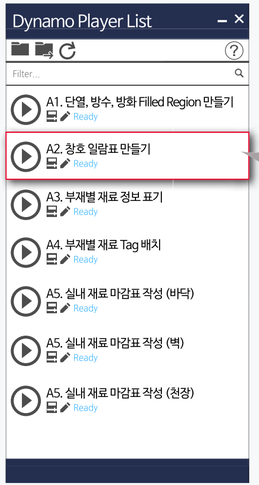
기능 및 필요성

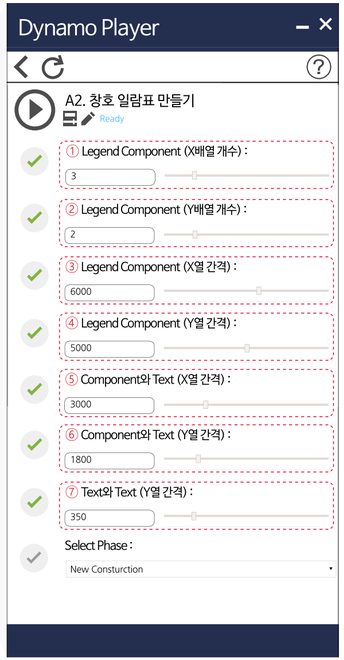
Legend View에서 Legend Component를 배치하여 창호 일람표를 작성 할 경우 사용된 Family를 찾아야 하고 Legend Component는 Tag와 연결이 되지않아 정보값을 TextNote를 이용하여 수기로 작성하는 경우가 있다. 또는, Floor Plan에 입면과 단면을 배치하여, 각 창호의 입단면 View를 생성하여 배치한다. 이럴 경우 만들어지는 View의 종류가 많아지고, 모델링이 추가 되었을 경우, 부재를 숨겨야 하는 번거로움이 있다. Family로 제작된 창호는 [A2] Dynamo Code를 이용하여 창호 일람표를 작성하는 것을 추천한다. 단, Curtain Wall일 경우 Legend Component로 만들기 어려운 부분이 있어, 기존 방식을 유지한다.

유의사항

- 작업 화면 : Legend View(범례)에서 구동이 가능하며, 기준이 되는 Window 또는 Door의 Legend Component가 1개 이상 꼭 배치되어야 한다.

- 배치된 Legend Component는 모델의 Type 정보가 있어야 한다.
- 한번 작성된 Legend Component는 삭제되고 다시 만들어진다.
- Legend Component는 배치 된 상태에서 위치를 이동시켜야 한다.





* Dynamo Player는 Revit 2017부터 사용 가능하며, Player의 설정창은 Revit 2018 이상 버전에서 사용할 수 있다.

Revit 2017일 경우 Dynamo 파일을 열고 설정을 수정하여 저장한 뒤 Player를 누르거나 Dynamo에서 바로 실행시킨다.

창호 일람표 생성을 위한 Parameter 조정

① Legend Component (X배열 개수)
X열에 배열된 최대 개수

② Legend Component (Y배열 개수)
Y열에 배열된 최대 개수

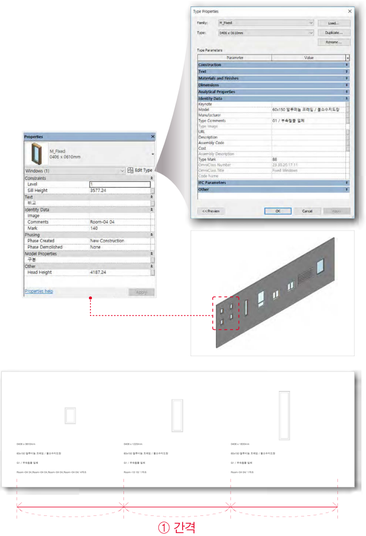
③ Legend Component (X열 간격)
Component의 좌우 간격

④ Legend Component (Y열 간격)
Component의 상하 간격

⑤ Component와 Text (X열 간격)
Component의 정보를 기입할 Text의 좌측간격

⑥ Component와 Text (Y열 간격)
Component의 정보를 기입할 Text의 하측간격

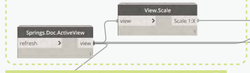
⑦ Text와 Text (Y열 간격)
정보를 표기 할 Text Note의 상하 간격



CODE OVERVIEW

A 배치된 Legend Component와 Textnotes 정보 지우기

A-1 View에 배치된 부재와 View Scale 정보 추출



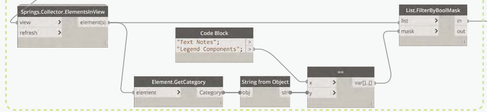
A-2 배치된 Text Notes와 Legend Components 찾기



A-3 배치된 Text Notes 지우기



A-4 View Scale, Sheet Number 정보 추출



A-5 List 정리

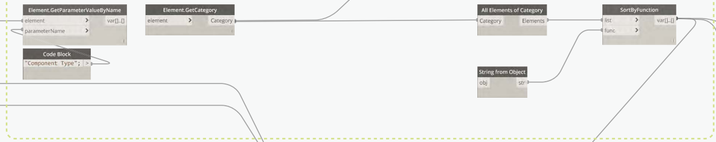


A-6 배치된 Components 지우기



B 사용된 Window 객체의 룸 정보 추출 및 정보 입력

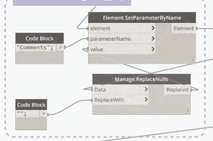
B-1 배치된 Legend Component의 Element 불러오기



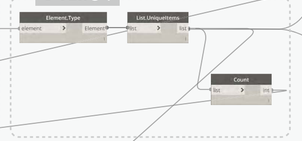
B-2 사용된 Window의 Room 위치 정보 추출



B-3 Room 정보 입력

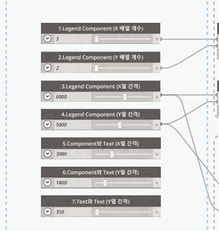


B-3 List 정리

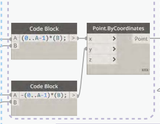


C Legend components 복사 및 배치

C-1 Legend Component 배열 선택



C-2 복사할 좌표 만들기



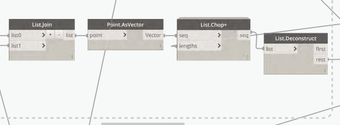
C-3 List 정리



C-4 X,Y 배열에 따른 좌표 만들기



C-5 List 정리



C-6 Legend Components 복사



C-7 List 정리

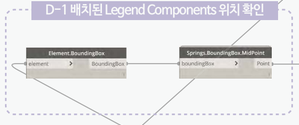


C-8 Legend Components 정보 설정

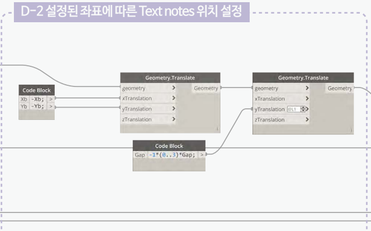


D Text Notes 위치 설정

D-1 배치된 Legend Components 위치 확인

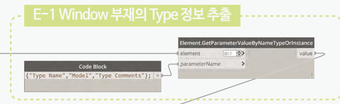


D-2 설정된 좌표에 따른 Text notes 위치 설정



E Text Notes 만들기

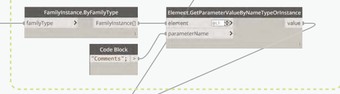
E-1 Window 부재의 Type 정보 추출



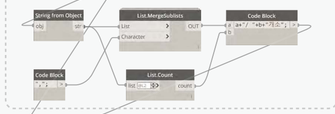
E-2 List 정리



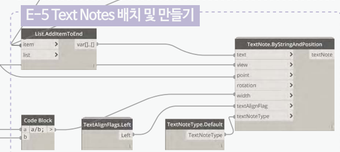
E-3 Window부재의 Instance에 기입된 룸 정보 추출



E-4 List 정리 및 Text 내용 만들기

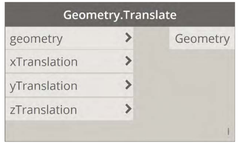


E-5 Text Notes 배치 및 만들기



Key Node

Dynamo basic node

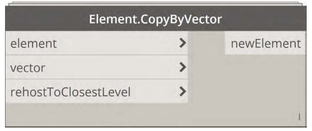


지정된 Geometry를 WCS에 정의 된 X,Y,Z 방향으로 각각 평행 이동 한다.

INPUTS
- Geometry
  방향을 변형 시킬 geometry를 불러 온다.
- x,y,z Translation
  변형시킬 좌표값을 연결 한다.

OUTPUTS
- Geometry
  요소의 변형 시킬 x, y, z 방향으로 평행 이동 시킨다.

custom node Clockwork Packages

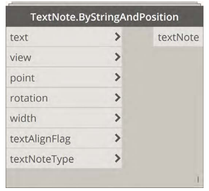


요소 혹은 요소의 List에 Vector 값을 부여하여 복사 한다.

INPUTS
- Element
  vector 값을 부여 할 요소를 연결 한다.
- Vector
  요소에 부여 할 vector 값을 설정 한다.

OUTPUTS
- New Element
  기존 요소에 vector 값을 부여하여 새로운 요소로 복사 한다.

custom node Clockwork Packages

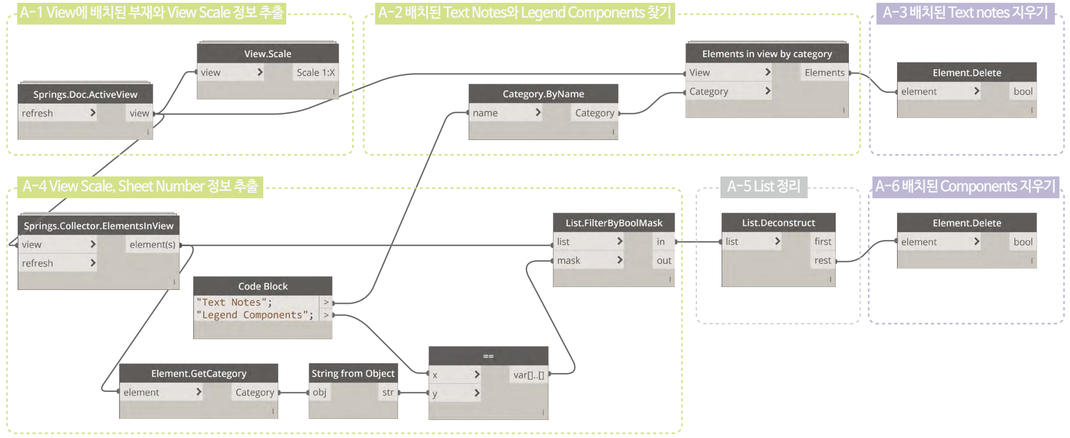


주어진 위치에 Text Note를 생성 한다.

INPUTS
- text, view, point, rotation, width, textAlignFlag, textnoteType
  생성할 TextNote의 내용,View의 좌표,길이, 정렬 방향등을 설정 한다.

OUTPUTS
- TextNote
  Input값으로 설정된 TextNote를 배치 한다.

A 배치된 Legend Component와 Text Notes 정보 지우기

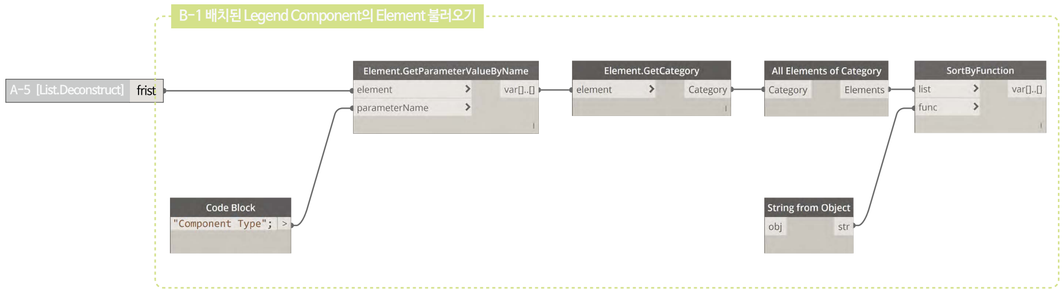


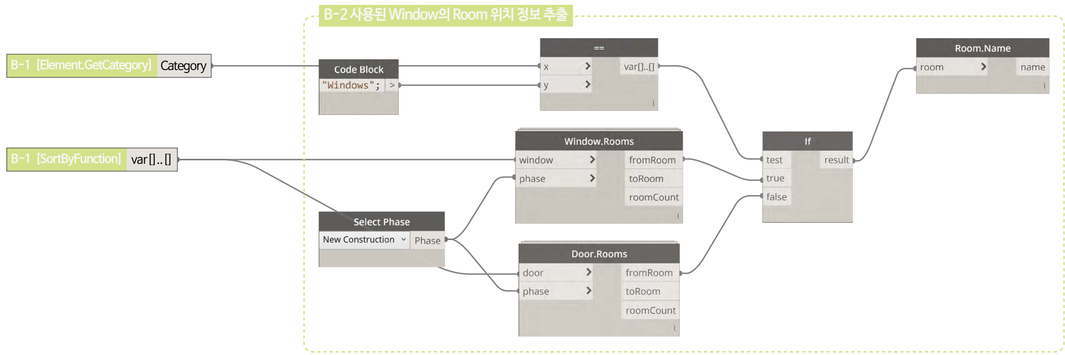
Legend Component와 Text Notes

Legend Componend의 배치는 Legend View(범례)에서만 가능하다. 또한 기준이 되는 하나의 Legend Component를 배치 해야만 Dynamo Code를 이용 할 수 있다. 먼저 현재 열려있는 Legend View의 배치된 Text Notes와 Legend Components를 Dynamo로 가지고 온다. [Category.ByName]과 [Elements in view by category]를 이용하여 Legend View의 배치된 Text Notes부재를 선택한다. 선택된 Text Notes는 [Element.Delete] node를 이용하여 삭제한다. [E]에서 Legend Component에 맞게 다시 Text Notes가 생성된다.

"Legend Components"는 [==]과 [List.FilterByBoolMask]를 이용하여 부재를 선택한다. Legend Components는 [Category.ByName]으로 읽어올 수 없기 때문이다.
선택된 Legend Components는 기준이 되는 첫번째 Component를 제외하고 삭제한다. [List.Deconstruct]는 List의 첫번째와 나머지 값을 분리 시켜주는 기능을 한다. 분리된 List는 [Element.Delete] node를 이용하여 삭제한다.

B 사용된 Window 객체의 Room 정보 추출 및 정보 입력

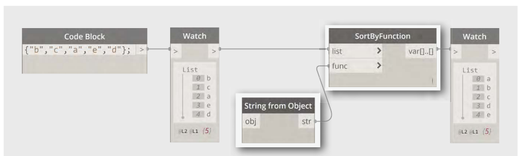




Component Type List 정리

[GetParameterValueByName]에 "Component Type" Parameter를 이용하여 배치된 Legend Component의 부재 정보를 가져온다. 가져온 부재 정보는 [Element.GetCategory] node를 이용하여 현재 Project에 배치되 Window 또는 Door의 부재 List를 Dynamo로 가져온다. [SortByFunction]을 이용하여 부재명을 기준으로 List를 정리해 준다.

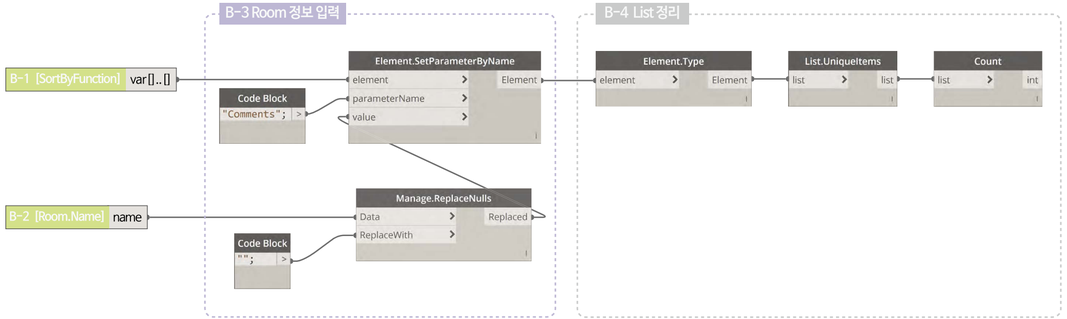
Dynamo에서 보여지는 List는 Revit에서 모델링된 순서대로 List가 구성된다. 창호 일람표 작성시 부재명 순서대로 배치하기 위해서 [SortByFunction]의 Input data인 Func에 Object의 이름순으로 정리하라는 [String from Object]를 연결시켜 List를 정리한다.



창호 Type에 따른 IF 조건문

배치된 Legend Component가 어떤 종류의 Type인지 판별하기 위해서 [IF] node를 사용한다. 배치된 Component가 "Window"와 같으면 [Window.Rooms] node의 값을 활용하고 아닌경우 [Door.Rooms] node의 값을 사용하도록 한다. 추출된 Room ID는 [Room.Name]을 이용하여 창호가 배치된 실명을 List로 추출한다.

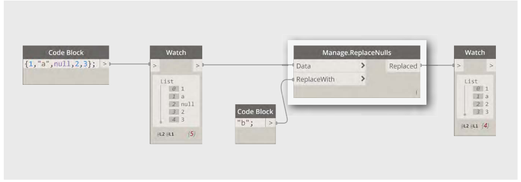
[Window.Rooms]와 [Door.Rooms] node는 Project에 배치된 Window 또는 Door 부재가 위치한 Room 정보를 추출 할 수 있다. 또한 창호 Family의 앞, 뒤로 인지하여 from Room과 to Room의 정보도 얻을 수 있어 창호가 배치된 방향이 맞는지 확인할 수 있다.



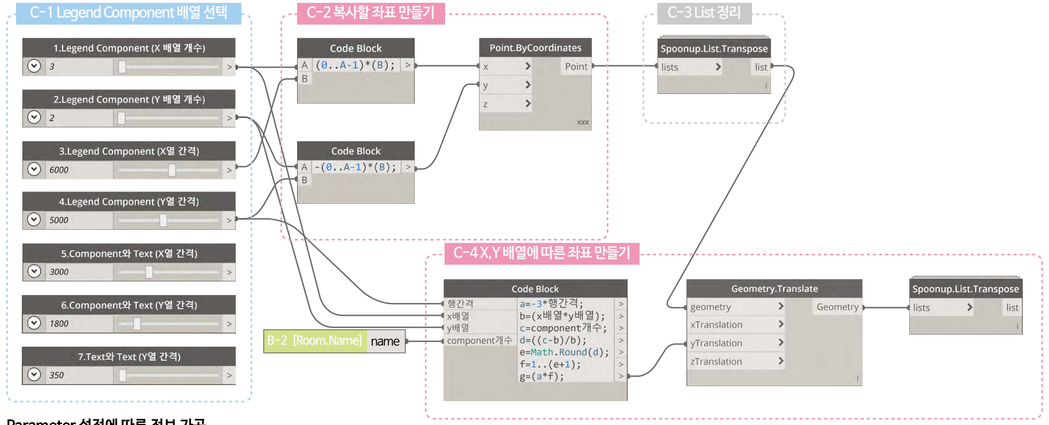
창호가 배치된 Room 정보

창호가 배치된 실명은 [Set.ParameterByName]을 이용하여 "Comments" Parameter의 값에 기입한다. 사용자가 다른 Parameter를 활용하고 싶다면 Parameter Name에 원하는 Parameter를 넣어주면 된다. Room 정보는 배치된 창호 Family마다 값이 다르기 때문에 Instance Parameter를 사용하는 것을 추천한다.

Room 정보가 기입된 창호 부재들은 [Element.Type]과 [List.uniqueItems] node를 이용하여 Legend View에 배치할 창호 Type을 선별하고, 배치된 창호 Type 개수를 [Count] node를 이용하여 확인한다.



C Legend Components 복사 및 배치



Parameter 설정에 따른 정보 가공

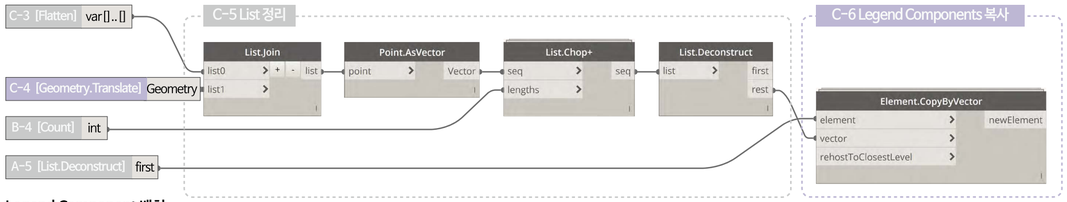
[A], [B]에서 정리한 Legend Component는 [C-1]에서 설정한 값을 통하여 Legend View에 배치한다. [C-1]에서 설정하는 값은 Component를 배열하는 X와 Y의 개수, Component의 X, Y열 간격, Component와 부재 정보는 기입하는 Text의 X, Y 간격을 설정한다. 설정값은 [G1. Tablelayout]에서 설정한 값과 동일하다. [G1. Tablelayout] code와 함께 사용하면 창호 일람표를 쉽게 작성할 수 있다.

Component 배치를 위한 X좌표값은 Code Block [(0..A-1) * B;]를 이용하여 왼쪽에서 오른쪽으로 작성하였고, Y 좌표값은 Code Block [-(0...A-1) * B;]를 이용하여 위에서 아래로 배치되도록 한다. Code Block의 수식으로 얻은 값들은 [Point.ByCoordinates]를 이용하여 Dynamo의 Point로 치환하였다. 작성된 좌표값은 Point는 [List.Transpose]를 이용하여 List의 배열 순서를 변경한다.[49 page 참고]

Legend View에 배치할 창호 Component가 X, Y 배열개수보다 많을 경우 Code Block [-3 * 행간격]의 값만큼 배치할 공간을 갖고 설정된 값으로 배열된다.

[Geometry.Translate] node는 이동시키려는 Geometry는 그대로 두고 새롭게 Geometry를 만든다. [Point.ByCoordinates]에 의해 작성된 Point는 [Geometry.Translate]를 이용하여 새로운 Point를 만든다.

Legend View는 여러개의 Sheet에 배치 할 수 있기 때문에 하나의 Legend View에서 창호 Component를 배치한다면, 창호 일람표를 위해 생성해야 하는 Revit의 View를 줄일 수 있다.

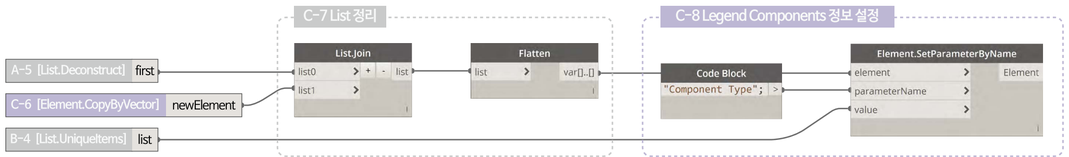


Legend Component 배치

[List.Join]으로 Component가 배치 될 Point를 단일 List로 만들어준다. [List.Join]이 [List.Create]와 다른점은 [List.Create]도 단일 List이지만 List 값을 분리하여 List를 작성한다. [List.Join]으로 만들어진 좌표 값들은 [Point.AsVector]를 이용하여 X, Y좌표에 의한 이동거리 값으로 치환한다.

Legend View에서는 Component의 X, Y좌표를 인식하지 않아 좌표를 이용한 이동이 불가능하다. 배치한 Component 중 기준이 되는 Component를 선택하여 상하좌우로 복사해야 하기 때문에 [Point.AsVector]를 이용하여 복사할 거리값을 추출한다.

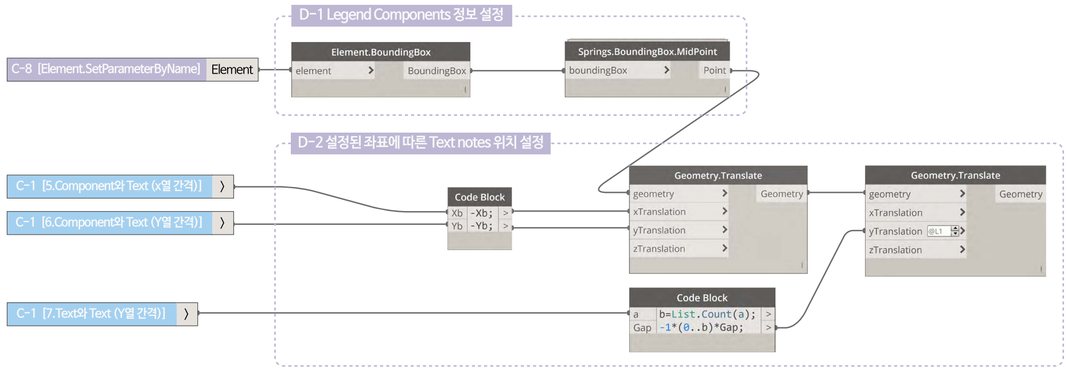
[Point.AsVector]에 만들어진 List는 [List.Chop+] node로 필요한 List 값만 추출한다. [List.Chop+] node에 배치할 Component Type의 개수를 연결하여 필요없는 좌표 값은 제외한다. [C-1]에서 만들어진 point 값이 배치할 Component보다 많은 경우 오류가 발생할 수 있다. [List.Deconstruct]를 통해 첫번째 좌표값을 제외시켜 기준이 되는 Component가 중복으로 배치되지 않도록 한다. [Element.CopyByVector]를 이용하여 기준 Component를 복사하여 배치한다.



Legend Component Type 정보

Component로 복사 된 Component들은 [List.Join]과 [Flatten]을 이용하려 단일 List로 만든다. 단일 List로 작성된 Legend Component는 [Set.ParameterByName]을 이용해 창호 Family의 Type 정보를 부여한다. Legend Component에서는 Family Type을 정하는 Parameter는 "Component Type"이다.

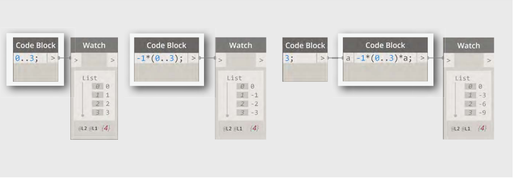
D Text Notes 위치 설정



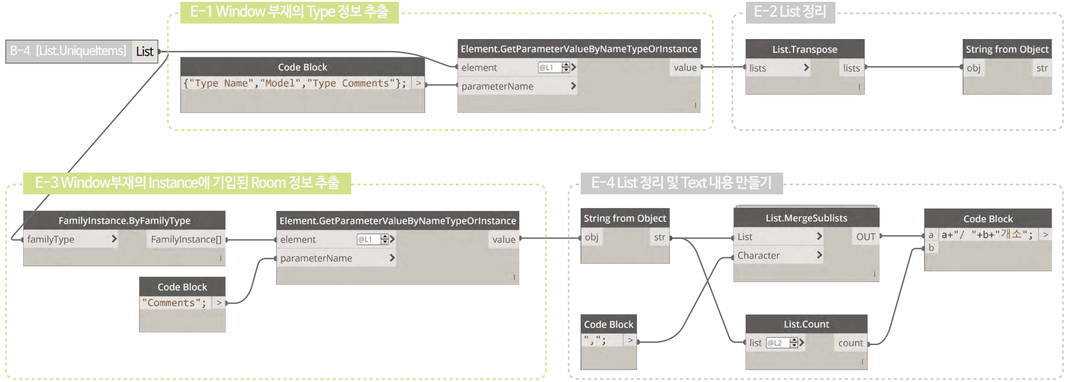
Text Note의 좌표

현재의 Dynamo Code로는 Legend View에 배치된 Legend Component의 좌표를 확인할 수 없다. [Element.BoundingBox] node를 이용하여 Legend Component 부재를 Box Geometry로 Dynamo에 가지고 들어온다. Bounding Box의 좌표를 기준으로 Text Note를 배치할 좌표를 설정 한다.

[Springs.BoundingBox.MidPoint]는 Dynamo에 배치된 Bounding Box의 Center Point를 알려주는 Node이다. Text Note의 배치기준은 Legend Component의 중심점에서의 Offset이라고 생각 하면 된다. Offset되는 배치 좌표는 초기에 설정한 배열 수, Text Note 간의 크기 등 [C-1]에서 주어진 정보로 배열되게 수식을 연산한다.



E Text Notes 만들기



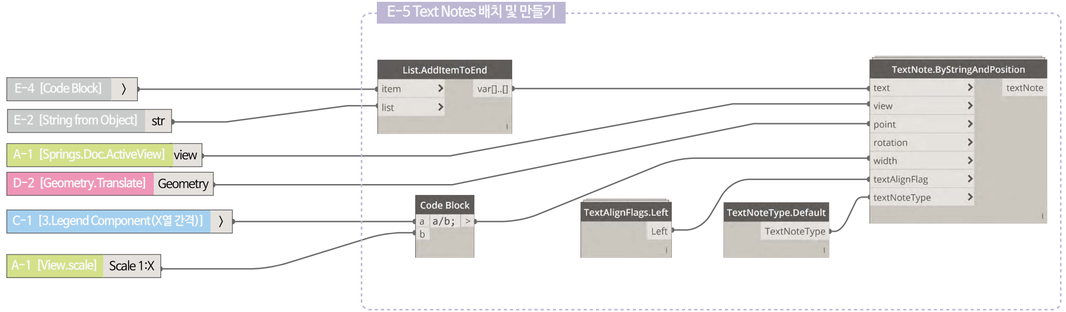
Parameter 정보 조합

[GetParameterValueByNameTypeOrInstance]에 Code Block ["Type Name","Model", "Type Comments"]을 연결 한 경우 3가지에 대한 정보값을 단일의 List로 얻을 수 있다. 부재별로 List 정리를 위해서 [List.Transpose]를 사용하였으며 List의 내용은 [String from Object]를 통해서 Text 형식으로 변경한다.

만일 창호 Family의 속성 정보를 다른 Parameter Name에 적었다면 CodeBlock ["Parameter Name"] 내용을 변경하여 활용한다.

[B-3]에서 창호 부재의 Instance Parameter "Comments"에 기입된 룸정보를 추출한다. 일람표에 들어가는 "위치 및 개소"정보를 만든다.

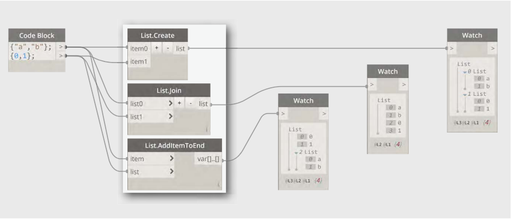
배치된 Room 정보는 각각의 창호 부재마다 다르다. [FamilyInstance.ByFamilyType]을 활용하여 Family Type 정보가 아닌 Family Instance 로 정보를 가져온다. 가져온 정보는 [List.MergeSublists]를 이용하여 Family Type이 배치된 Room이름은 하나의 값으로 합친다. [List.Count] node로 Family Type 사용된 개수를 파악하여 "Room 이름 + 개소"형태의 Text 정보를 조합하여 만든다.



Text Note 작성

[List AddItem To End]를 이용하여 창호의 부재의 Type 정보와 배치된 Room 정보를 합친다. [List.Join]과 [List.Create]와 같은 기능이지만 List 내에 또 다른 List로 정리되어 있고 각각 List뒤에 Item을 추가하고 싶을 때 사용한다.

[TextNote.ByStringAndPosition]을 이용하여 Text Notes의 내용을 배치 할 View와 좌표, Text Notes의 길이, Text 정렬 방향을 지정하여 배치 할 수 있다. Legend Component와 다르게 Text Notes는 Dynamo에서 좌표를 설정하여 배치 할 수 있다. 기준이 되는 Text Notes는 배치할 필요가 없다.



A3. 부재별 재료 정보 표기

동영상 : https://youtu.be/LsoF9d3ZJYg

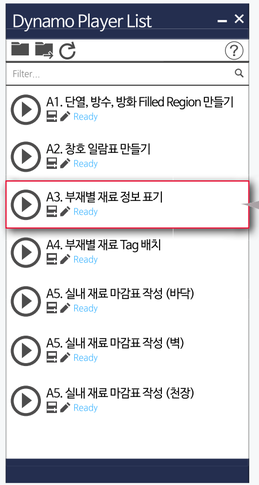
기능 및 필요성

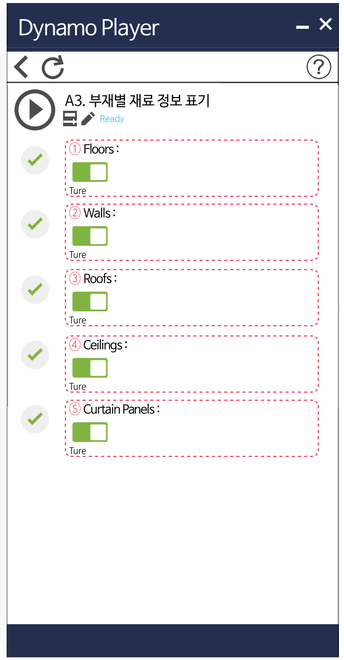
국내 도면 스타일의 재료명 표기 방식을 Revit의 Tag로 표현 하기에는 어려움이 있다. 지시선 표현, 재료의 두께 표기 등, 국내 도면 방식을 표현하기 위해 Revit에서 수기로 작업하는 경우가 많다.
"A3. 부재별 재료 정보 표기" Code는 "A4. 부재별 재료 Tag 배치" Code를 활용하기 위해 실행되는 사전 작업이다. [A3] Code를 이용하여 Layer로 구성된 부재의 재료 정보를 추출하여 지정된 Parameter 값에 기입 해준다.

유의사항

- 작업 화면 : View와 상관없이 구동이 가능하다.

- 반드시 제공되는 Material.txt 파일로 Shared Parameter를 만들어야 한다.
- 제공되는 Tag 같은 Share Parameter로 구성되어야 Tag와 연동 시킬 수 있다.





* Dynamo Player는 Revit 2017부터 사용 가능하며, Player의 설정창은 Revit 2018 이상 버전에서 사용할 수 있다.
Revit 2017일 경우 Dynamo 파일을 열고 설정을 수정하여 저장한 뒤 Player를 누르거나 Dynamo에서 바로 실행시킨다.

자동 Space 생성을 위한 파라미터 조정

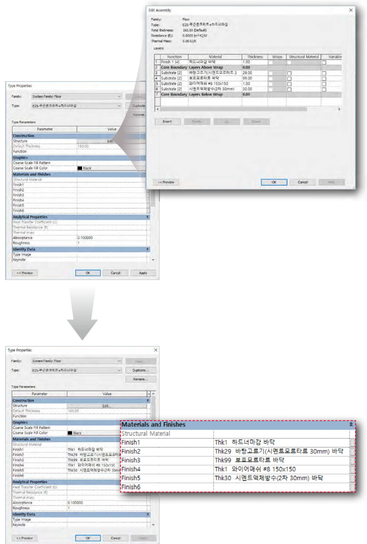
① Floor
True이면 모든 Floor 객체의 재료 정보를 표기합니다

② Walls
True이면 모든 Walls 객체의 재료 정보를 표기합니다

③ Roofs
True이면 모든 Roofs 객체의 레이어 재료 구성 정보를 Finish Table에 표기합니다

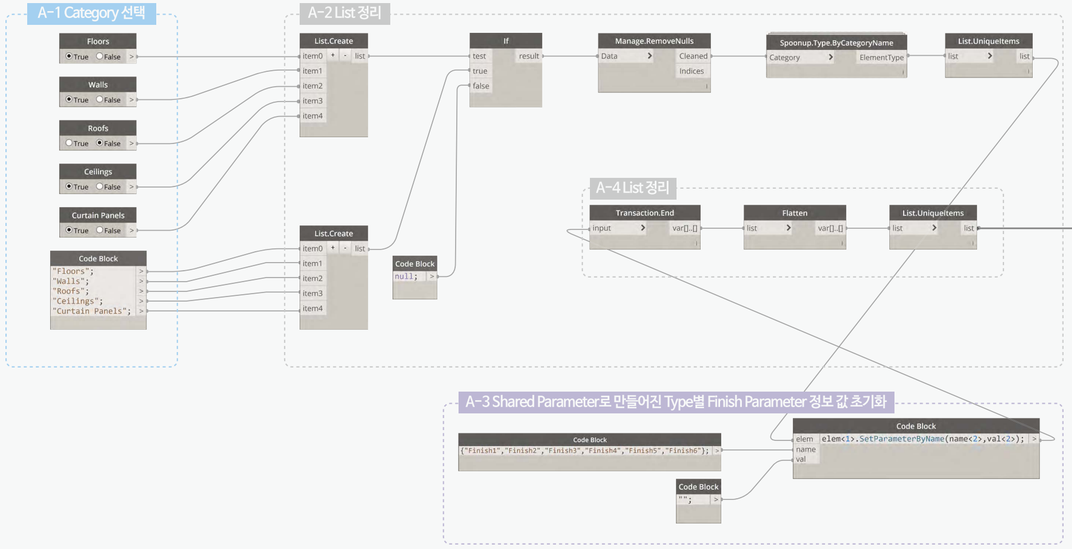
④ Ceilings
True이면 모든 Ceilings 객체의 재료 정보를 표기합니다

⑤ Curtain Panels
True이면 모든 Curtain Panel 객체의 재료 정보를 표기합니다

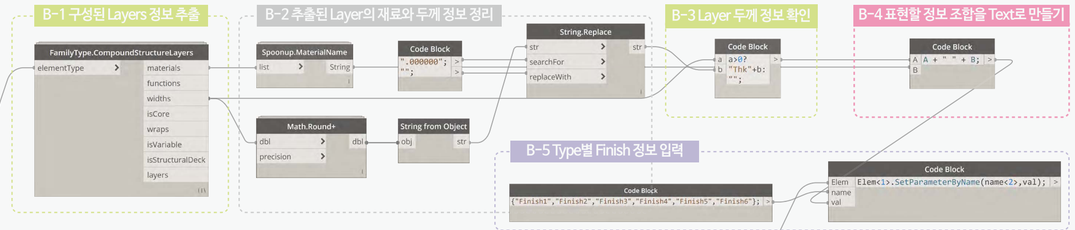


CODE OVERVIEW

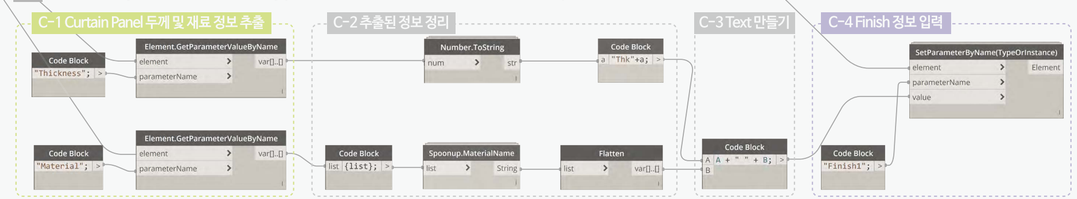
A Category 선택 및 Parameter 정보 초기화



B 구성된 Layers 정보 추출 및 Type Parameter 정보 입력

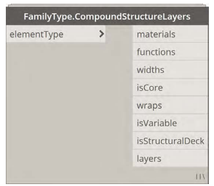


C 구성된 Layers 정보 추출 및 Type Parameter 정보 입력 (Curtain panel)



Key Node

custom node **Clockwork Packages**

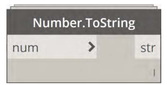


부재 Type이 구성된 재료, 기능,두께, 레이어 등 정보를 추출 한다.

INPUTS
- Element Type
  정보를 추출할 부재의 Type을 Input한다.

OUTPUTS
- materials, functions, widths, isCore, wraps, isVariable, isStructuralDeck, layers
  Input된 Type을 구성하는 정보를 확인한다.

custom node **Springs Packages**

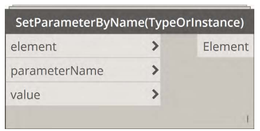


숫자 정보를 문자 형식으로 변환 한다.

INPUTS
- num
  Text 형식으로 변경 할 숫자 요소를 불러 들인다.

OUTPUTS
- str
  Input으로 들어온 숫자 정보를 문자 형식으로 변환하여 내보낸다.

custom node **Rhythm Packages**

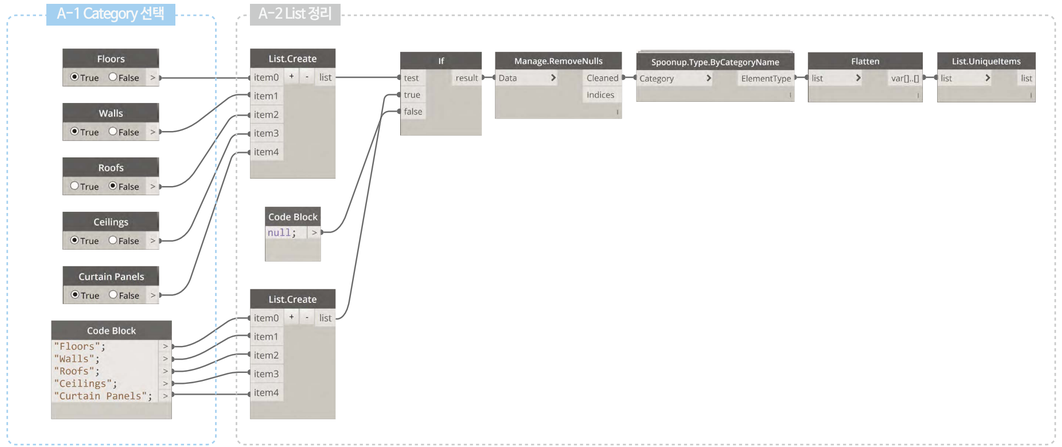


선택한 요소의 Parameter 값을 기입 한다.

INPUTS
- element
  변경 및 값을 기입 할 요소를 불러 온다.
- parameterName, value
  ParameterName 및 매개 변수 값을 추가 한다.

OUTPUTS
- Element
  Parameter 값을 추가한 요소로 내보낸다.

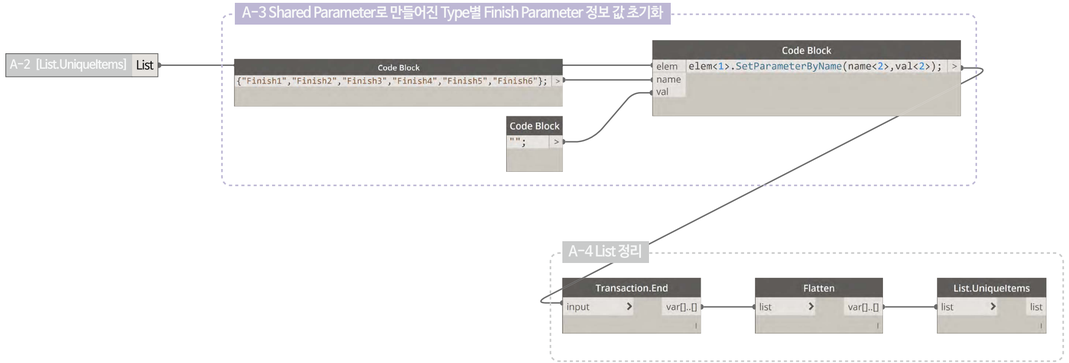
A Category 선택 및 Parameter 정보 초기화



Parameter 설정에 따른 정보 추출

재료명을 표기하기 위해서 Floors, Walls, Roofs, Ceilings, Curtain Panels Category를 구성하는 Compound Layer들을 Shared Parameter로 만든 "Finish 1 ~ Finish 6" Parameter로 불러온다. Dynamo Player에서 원하는 Category들을 True로 선택하면 선택된 Category만 "Finish 1 ~ Finish 6" Parameter에 Compound Layer의 정보가 기입된다.

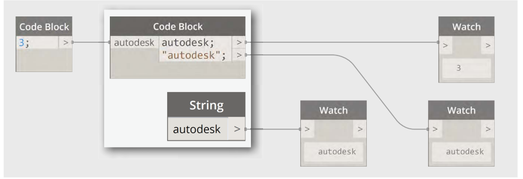
[IF] node는 True로 선택된 Category만 진행될 수 있도록 조건을 주었다. 재료 정보를 추출할 Category name은 [Spoonup.Type.ByCategoryName]에 Input되어 **현재 Project에 사용된 해당** Category의 Family Type을 선택한다. [Manage.RemoveNulls]는 선택되지 않은 Category들을 List에서 삭제하여 오류가 발생하는 것을 방지한다.



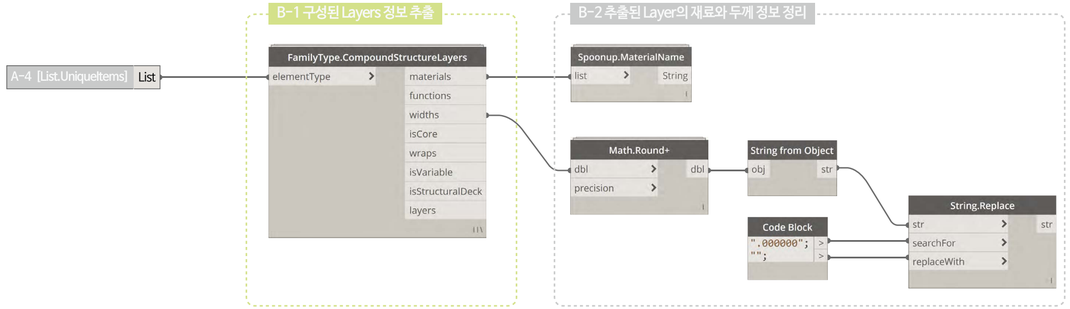
Shared Parameter 설정

선택된 Category의 Compound Layer에 표현되는 재료명은 재료명은 Shared Parameter로 만들어진 "Finish1 ~ Finish6" Parameter에 기입된다. Shared Parameter로 작성한 이유는 [A4. 부재별 재료명 Tag 배치]에서 활용할 Tag Fmaily와 연동시키기 위해서이다.

사용자의 Shared Parameter 또는 Revit의 기본 Parameter를 이용하고 싶다면 Code Block안의 Text를 변경하면 된다. Code Block [SetParameterByName]을 통해 "Finish1 ~ Finish6" Parameter의 값에 기입되어 있는 정보를 삭제한다. 부재의 Layer 구성이 변경될 경우 기존 정보가 남아 있을 수 있기 때문에 정확한 Update를 위해서 Parameter 값을 삭제한다. 삭제된 Parameter 값은 [Transaction.End] node를 이용하여 정보가 뒤쪽 Code와 충돌되지 않게 한다.



B 구성된 Layers 정보 추출 및 Type Parameter 정보 입력

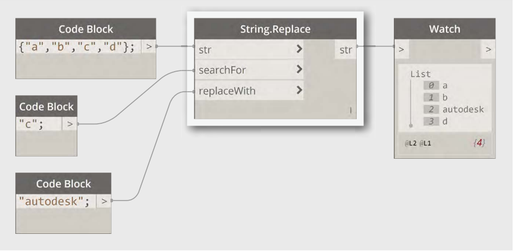


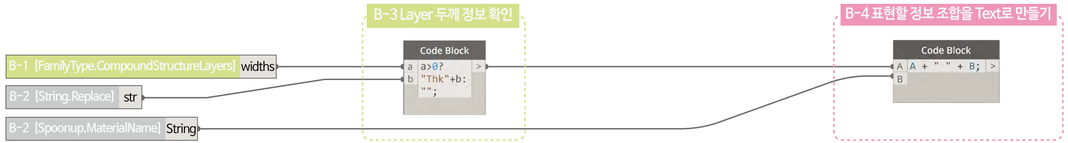
Family Type의 재료 및 두께 정보

[FamilyType.CompoundStructureLayers]을 이용하여 Family Type의 재료 ID를 추출한다. Dynamo의 기본 node인 [Material.Name]은 설정되지 않은 Layer의 재료일 경우 오류가 발생한다. [Spoonup.Material.Name]을 쓰는 것을 추천한다.

[FamilyType.CompoundStructureLayers]에서 Family Type을 구성하는 Layer별 Widths 값을 추출한다. [Math.Round+]는 혹시 잘못 설정된 두께 값을 정수로 정리해주는 기능을 한다. 정리된 두께 값을 [String from Object] node를 통해 Text형식으로 변경한다. Text로 변경하는 이유는 [B-4]에서 두께 값과 재료명을 Text로 조합하는 Code를 사용하기 위해서이다. 정수라도 Dynamo에 사용하는 숫자형식은 소수점 여섯째 자리까지 표현된다. [String.Replace] 기능으로 소숫점을 정리한다.

숫자를 Text 형식으로 변경할 경우 [Spring.Number.ToString]을 이용하면 [String.Replace] node를 사용하지 않아도 숫자의 소수점이 정리된다.





Text의 정보 조합

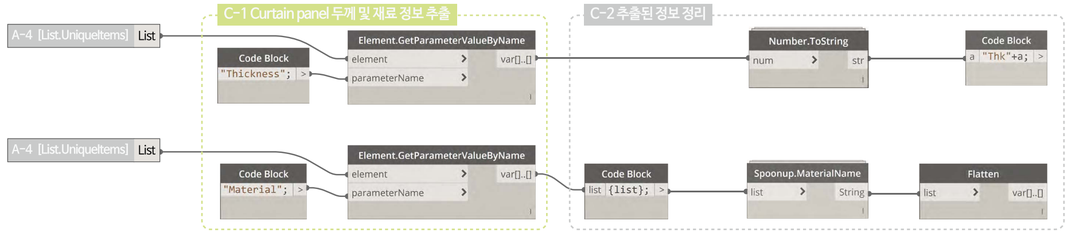
Family Type을 구성하는 Layer의 두께 값이 0일 경우, 재료 정보에 두께 값을 표기하지 않기 위해 [IF] node를 사용한다.
두께 값이 0보다 클 경우 "Thk"라는 Text가 두께값과 조합되어 정보가 만들어 진다. Code Block [A+" "+B;]는 "Thk 두께값" + "재료명"으로 Text를 조합한다. "mm"로 표현할 경우 Code Block의 "Thk"를 "mm"로 변경하고, Code Block A와 B에 Input한다.



Shared Parameter의 정보

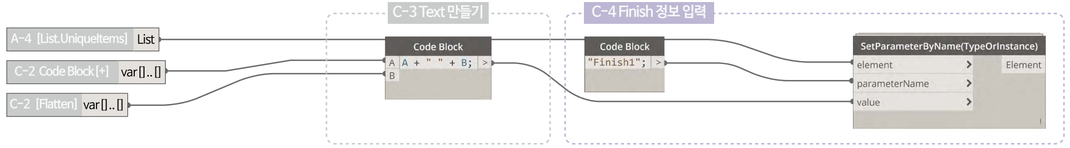
조합된 Text 정보는 [SetParameterByName]을 이용하여 Shared Parameter로 만들어진 "Finish1~Finish6" Parameter의 값에 정보를 기입한다. 사용자의 Shared Parameter 또는 Revit의 기본 Parameter를 이용하고 싶다면 Code Block안의 Text를 변경하면 된다.

C 구성된 Layers 정보 추출 및 Type Parameter 정보 입력 (Curtain panel)



Curtain Panel의 재료 및 두께

Floors, Walls, Roofs, Ceilings와 다르게 Curtain Panels은 Family 구성이 Layer로 되지 않는다. 하지만 도면에 재료를 표기하기 때문에 기존 Parameter의 정보를 추출하여 "Finish1~Finish6" Parameter 값에 기입한다. Curtain Panels은 Revit의 기본 Parameter인 "Thichness"와 "Material"의 정보를 [Get.ParameterValueByName]을 이용하여 추출한다. 추출된 두께값은 [Springs.NumberToString]을 이용하여 Text형식으로 변경하고 Code Block "Thk"와 조합하여 정보를 구성한다 "mm"로 표현할 경우 Code Block의 "Thk"를 "mm"로 변경한다. [Get.ParameterValueByName]으로 추출된 "Material" 정보는 Material ID로 추출되어 [Spoonup.MaterialName]을 이용해 재료명 Text를 구성한다.



Textd의 정보 조합

조합할 Text정보는 Code Block [A+" "+B]을 이용하여 기입할 Text 정보를 만든다. [Set.ParameterByName]을 이용하여 "Finish 1" Parameter의 값에 정보를 기입한다. 사용자의 Shared Parameter 또는 Revit의 기본 Parameter를 이용하고 싶다면 Code Block안의 Text를 변경하면 된다.

A4. 부재별 재료 Tag 배치

동영상 : https://youtu.be/hJ7tGm4ALoU

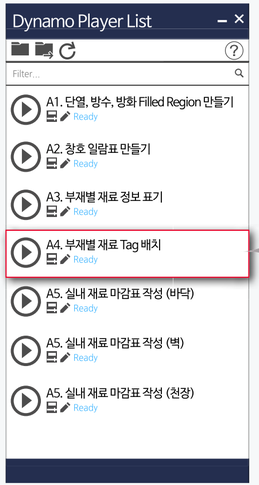
기능 및 필요성

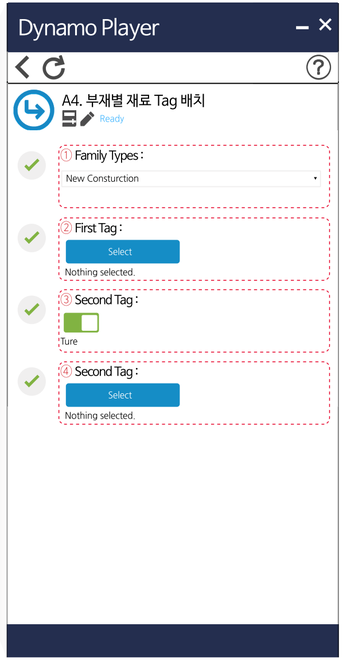
앞장에서 만든 재료정보를 Tag를 이용해서, 도면에 표기 할 수 있다. 많은 Revit 사용자들이 중간, 실시 설계 단계로 가면서 단일벽이 아닌 구조와 건축 마감 벽을 분리 하여 모델링을 진행한다. 이때 Revit Tag를 사용 할 경우 Tag를 두개 사용 하여 위치를 맞추거나 지시선을 그려주는 반복작업을 하게 된다. 또는 Text Notes로 재료명을 다시 기입하는 경우도 있다.

부재별 재료 Tag 배치 기능은 두개의 Tag가 배치될 경우 첫번째 Tag 기준으로 두번째 Tag 위치를 자동으로 정렬하고 지시선을 자동으로 배치 해주는 기능이다.

유의사항

- 작업 화면 : 3D View를 제외한 평.입.단면 View에서 구동이 가능하다.
- Shared Parameter로 작성된 Tag와 제공되는 Tag 지시선 Family를 사용해야 오류 없이 사용가능하다.
- 3D 뷰에서 지시선 사용은 불가능하다.





* Dynamo Player는 Revit 2017부터 사용 가능하며, Player의 설정창은 Revit 2018 이상 버전에서 사용할 수 있다.
Revit 2017일 경우 Dynamo 파일을 열고 설정을 수정하여 저장한 뒤 Player를 누르거나 Dynamo에서 바로 실행시킨다.

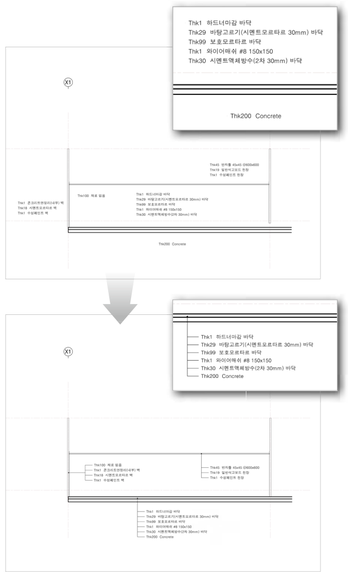
Tag 배치를 위한 Parameter 조정

① Family Types
재료 Tag 지시선 Family

② Select First Tag
선택된 Tag 위치로 Second Tag가 정렬한다.

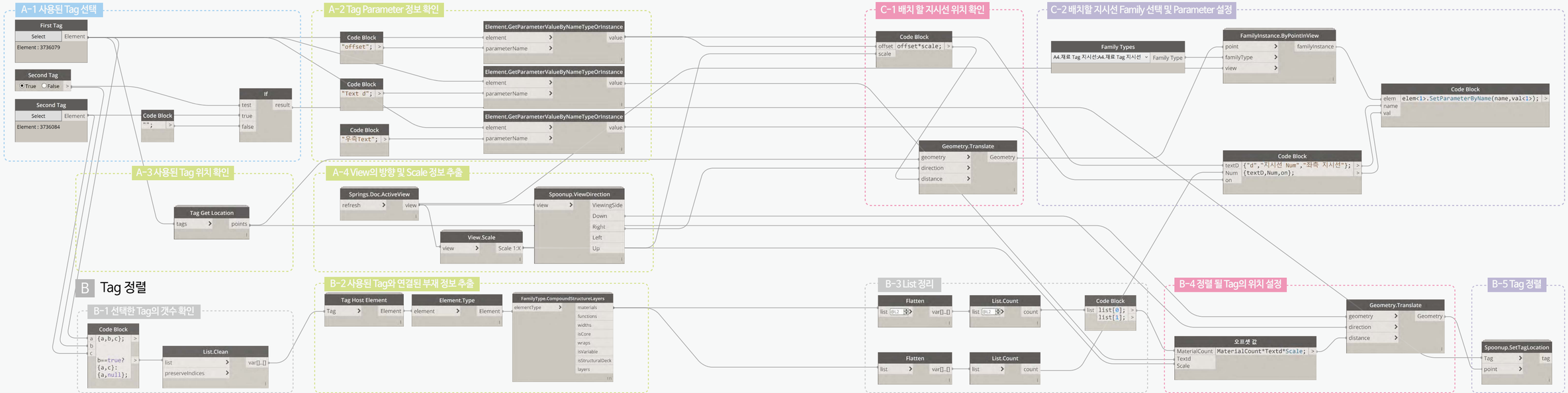
③ Second Tag
Second Tag가 존재할 시, True를 선택한다.

④ Select Second Tag
Second Tag가 존재할 시, 위치를 정렬할 Tag를 선택한다.
선택할 Tag가 없을 시 First Tag를 한 번 더 선택한다.



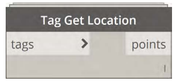
CODE OVERVIEW

A 사용된 Tag 선택 및 정보 추출



Key Node

custom node **MEPover Packages**



Tag의 위치를 확인한다.

INPUTS
- Tags
  위치 정보를 추출하기 위하여 Tag를 불러 온다.

OUTPUTS
- Points
  불러온 Tag의 위치 정보를 추출한다.

custom node **Spoonup Packages**



View가 가지고 있는 방향을 확인 한다.

INPUTS
- View
  방향을 확인할 View를 Input한다.

OUTPUTS
- Viewing Side
  View의 방향을 확인하여 내보낸다.
- Down, Right, Left, Up
  선택된 View에서 부재를 이동 시킬 기준을 내보낸다.

custom node **Spoonup Packages**

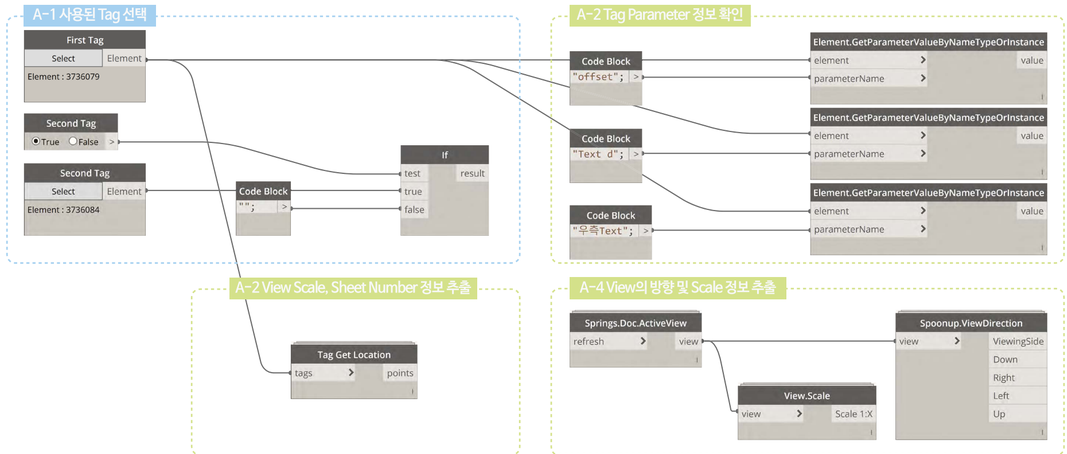


주어진 지점에 따라 Tag를 배치한다.

INPUTS
- Tag
  이동 시킬 Tag를 불러 온다.
- Point
  Tag를 이동 시킬 위치를 잡는다.

OUTPUTS
- Tag
  지정된 위치로 불러들인 Tag를 이동 시킨다.

A 사용된 Tag 선택 및 정보 추출

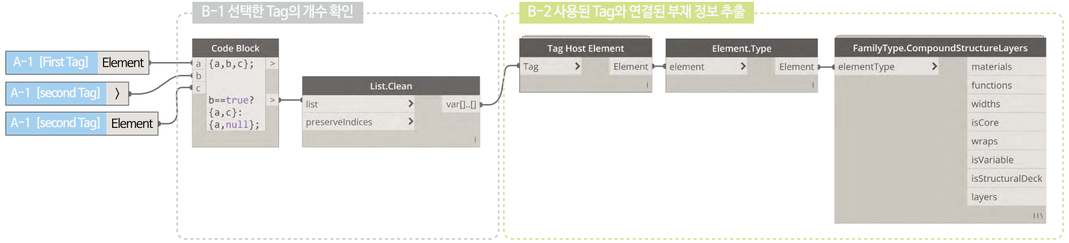


Tag의 Parameter 정보

배치된 Tag를 [Select.ModelElement] node를 활용하여 선택한다. 정렬 해야 할 Second Tag가 없다면 [True/False]에서 False를 선택한다. 만일 정렬 할 Second Tag가 있다면 True를 선택한 후 [Select.ModelElement]로 정렬할 2번째 Tag를 선택한다. [IF] node는 Second Tag가 없을 경우 지시선 간격 및 개수에 첫번째 Tag 정보로 진행 하기 위해서 사용 하였다. Tag Family에 사용된 Parameter인 "offset", "Text d", "우측 Text"에 대한 parameter 값을 [Get.ParameterValueByName] node로 가지고 온다.

두번째 Tag 정렬과 지시선 Family 배치를 위해서 사용한 첫번째 Tag의 좌표 정보를 [Tag.GetLocation]을 이용하여 추출한다. [Springs.Doc.ActiveView]를 통해 View의 Scale을 추출하고, [Spoonup.ViewDirection]을 통해서 현재 View의 평면, 입면, 단면 여부를 파악한다. 현재 뷰가 평면이라는 것과 입면, 단면이라는 것에 따라서 Dynamo에서 인식되는 좌표가 다르다. [A4] 부재별 재료 Tag배치 Code 중 가장 중요한 Node이다.

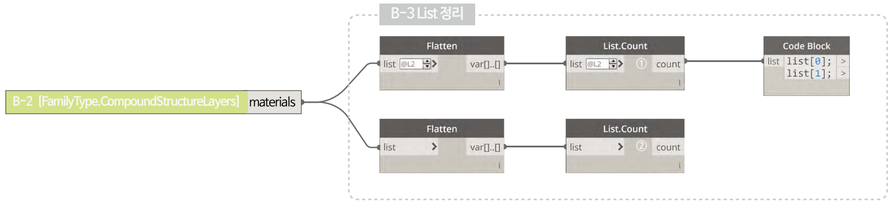
B Tag 정렬



Tag와 연결된 Family Type 정보

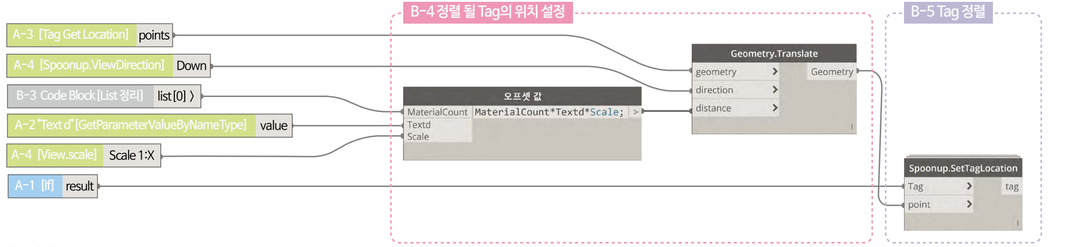
Code Block [IF] node를 사용하여 "Second Tag"의 사용여부를 판단한다. True일 경우 두개의 Tag의 정보를 가져온다.

부재의 Layer정보를 얻기 위해 [FamilyType.CompoundStructureLayers]에 부재의 Type정보를 Input한다.



Family Type의 Material Layer

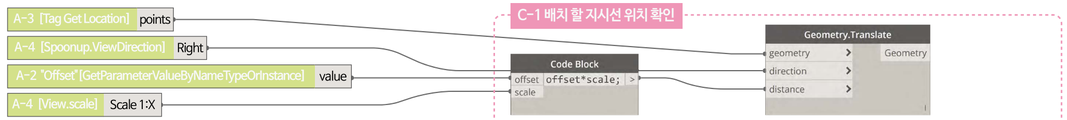
부재의 Type 정보는 "A3. 부재별 재료 표기"에서 Shead Parameter로 작성한 "Finish1~Finish6" Parameter를 이용하지 않는다. 기입된 Material 정보와 중복체크를 하기 위해서 [FamilyType.CompoundStructureLayers]를 이용하여 구성된 Material Layer개수 정보를 추출한다.



Tag 정렬

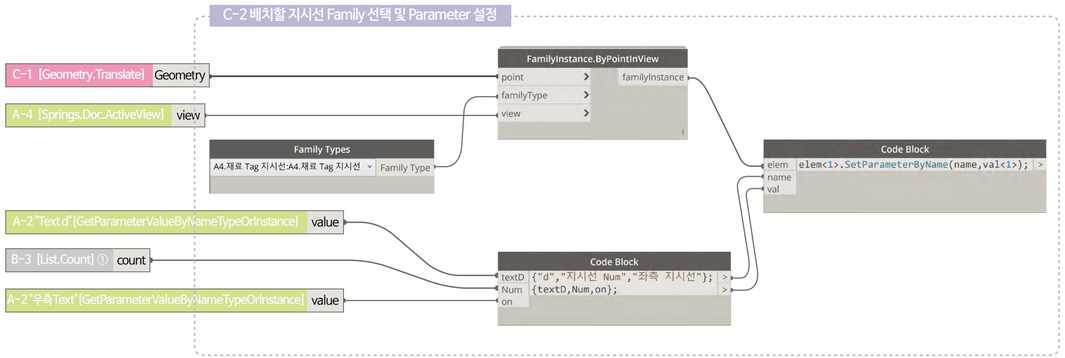
추출된 Material 정보중 구성된 Layer의 Material 개수를 가지고 온다. Material Name은 [A3. Dynamo Code]에 의해 "Finish1~Finish6" Parameter가 입력되어 있다. [A4. Dynamo Code]에 "Material Name"까지 가지고 오게 되면, Code가 복잡하게 되고, Dynamo에서 연산하는 구동 시간이 길어진다. 구성된 Layer의 Material 사용 개수를 통하여 중복 체크를 한다. 부재 Type을 구성하는 Layer의 Material 개수 정보는 [지시선 Family 배치] node에서 활용 된다. 정렬시킬 Second Tag는 첫번째 First Tag 밑에 정렬될 수 있도록 좌표를 설정하고, [Spoonup.SetTagLocation]으로 위치를 이동시킨다.

C Tag 지시선 배치



지시선 Family의 위치

배치된 First Tag를 기준으로 Tag의 앞에 지시선 Family가 배치되도록 좌표를 설정한다. Code Block [Offset X Scale]은 Tag Fmaily가 가지고 있는 Parameter 값과 현재 View의 Scale과 연산을 통하여 First Tag에서 앞에 배치되는 간격을 구한다.



Tag의 지시선 배치

[Family Types] node에서 "A4. 재료 Tag 지시선" Family를 선택한다.
"A4 재료 Tag 지시선"은 제공되는 Tag Family의 "d", "지시선 Num", "좌측 지시선" Parameter를 공유한다. [FamilyInstance.ByPointView]에 "A4. 재료 Tag 지시선" Family Type과 좌표를 연결하여 Family를 First Tag앞에 배치한다. [Set.ParameterByName]에서 First Tag 와 Second Tag가 가지고 있는 "d", "지시선 Num", "좌측 지시선"의 Parameter 값을 이용하여 "A4. 재료 Tag 지시선" Family에 Parameter 값에 기입한다.

배치할 지시선 Family를 선택한 후 [FamilyInstance.ByPointView]를 통해 Tag Family의 Parameter 값을 기입한다. 지시선 Family는 Tag Family의 Parameter와 동일한 Parameter 값에 의해 지시선 간격, 좌측 지시선인지, 우측지시선 여부, Text Offset등.. 값을 갖게 된다.

A5. 실내 재료 마감표

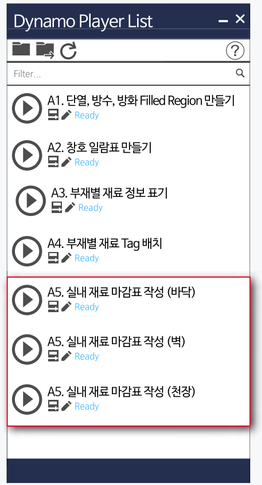
동영상 : https://youtu.be/eKBIJpU9a68

기능 및 필요성

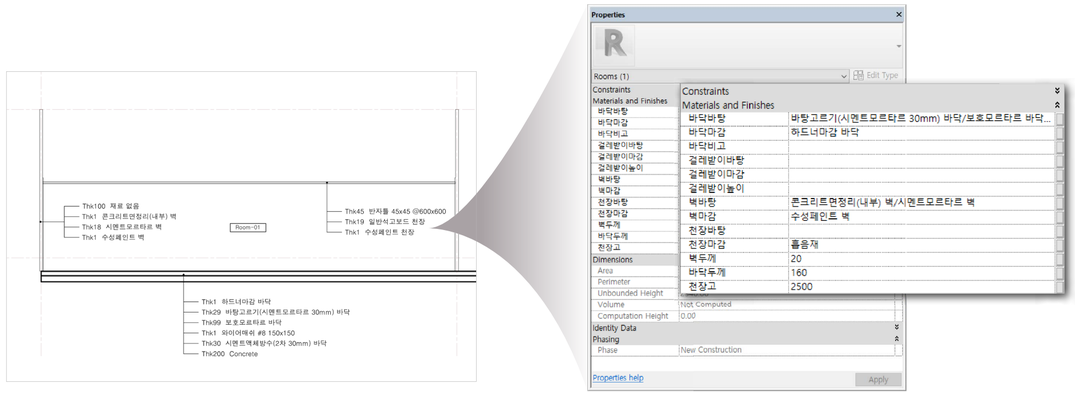
Revit의 Schedule을 이용하여 실내 재료 마감표를 작성할 경우 활용된다. Room에 사용된 바닥, 벽, 천장 부재 정보를 Room Parameter 값에 자동으로 기입하여 준다.

유의사항

- 작성 화면 : View와 상관없이 구동이 가능하다.
- 사용한 Parameter가 Room에 설정되어 있어야 한다. (Parameter : 바닥 바탕, 바닥 마감, 바닥 두께, 벽 바탕, 벽 마감, 벽 두께, 천장 바탕, 천장 마감, 천장고)
- 설정한 Parameter의 형식은 Text로 해야한다.
- 벽 부재일 경우 Room Bounding이 반드시 Off 되어 있어야 한다.



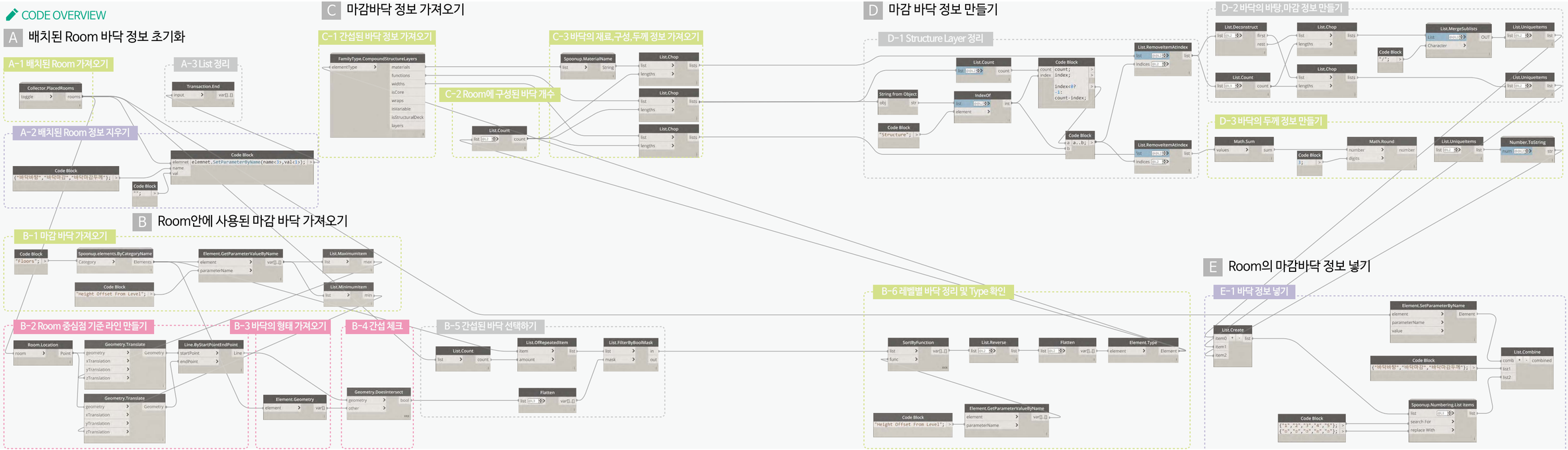
* Dynamo Player는 Revit 2017부터 사용 가능하며, Player의 설정창은 Revit 2018 이상 버전에서 사용할 수 있다.
Revit 2017일 경우 Dynamo 파일을 열고 설정을 수정하여 저장한 뒤 Player를 누르거나 Dynamo에서 바로 실행시킨다.



실내재료 마감표 with Dynamo

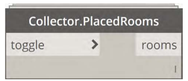
Number	Name	바닥 바탕	바닥 마감	바닥 두께	바닥 상세번호	바닥 비고	걸레받이 걸레받이바탕	걸레받이 걸레받이마감	걸레받이 상세번호	걸레받이 걸레받이높이	벽 벽바탕	벽 벽마감	벽 상세번호	벽 벽두께	천장 천장바탕	천장 천장마감	천장 천장고	천장 상세번호
01	Room-01	바탕고르기(시멘트모르타르 30mm) 바닥/보호모르타르 바닥/와이어메쉬 #8 150x150/시멘트액체방수(2차 30mm) 바닥	하드너마감 바닥	①160 ②0							콘크리트면정리(내부) 벽/시멘트모르타르 벽	수성페인트 벽		20		흡음재	2500	
02	Room-02	①시멘트모르타르	①타일 ②세라믹 타일	①160 ②0							①시멘트모르타르 ②	①회감석 ②흡음재	①130 ②60	판지틀 45x45 @600x600/일반석고보드 천장		수성페인트 천장	2600	
04	Room-04	①시멘트모르타르 20mm/바닥/시멘트액체방수(2차 30mm) 바닥 ②	①우레탄탑방수 ②점토벽돌	①50 ②0								우리점유보드(글라스클로스마감)	50	①판지틀 45x45 @600x600/일반석고보드 천장 ②Metal Furring	①수성페인트 천장 ②Ceiling Tile 600 x 600		2600	

CODE OVERVIEW



Key Node

custom node **Rhythm Packages**

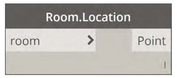


현재 Project에 배치된 Room의 정보를 수집한다.

INPUTS
- Toggle
  기본값(False)를 True로 변경하여, 배치된 Room을 추출한다.

OUTPUTS
- Rooms
  사용된 Room 정보를 내보낸다.

Dynamo basic node



Room의 위치를 추출한다.

INPUTS
- Room
  Room의 정보를 불러온다.

OUTPUTS
- Point
  Room의 위치 정보를 추출한다.

Dynamo basic node

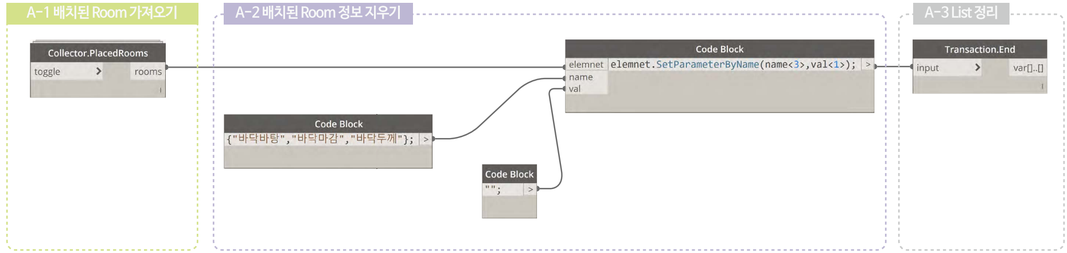


선택된 요소의 형태를 Dynamo의 Geometry로 변환 한다.

INPUTS
- Element
  Geometry로 변환할 요소를 Input한다.

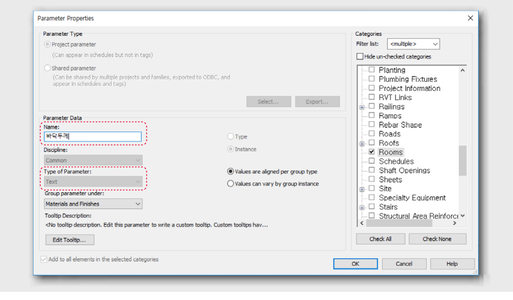
OUTPUTS
- var[]
  변환된 Geometry 정보를 확인한다.

A 배치된 Room 바닥 정보 초기화

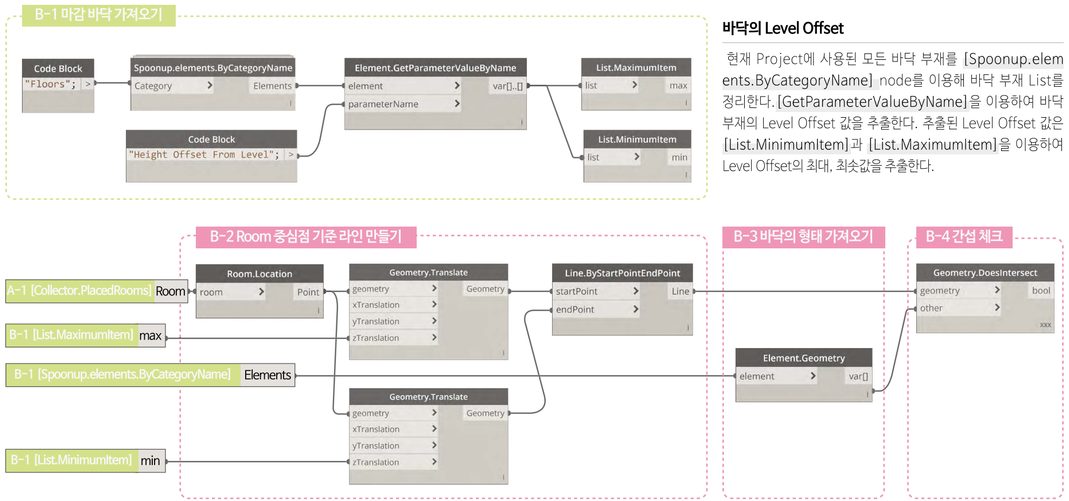


Room의 정보 초기화

Dynamo를 이용하여 실내 재료 마감 표 작성을 위해서 기존에 작성된 실내 재료 마감 표의 정보를 초기화해야 한다. [Collector.PlacedRooms] node를 이용하여 현재 Project에 배치된 Room의 정보를 추출한다. Room은 배치되었지만 벽 또는 Room Separation Line으로 구획되지 않으면 [Collector.PlacedRooms]에서 Room을 인식하지 않는다. Project에 배치한 Room이 구획이 되어 있는지 확인이 필요하다.
Room에 설정한 Shared Parameter "바닥 바탕","바닥 마감","바닥 두께"는 Room Schedule에서 Room에서 사용된 바닥 정보를 넣게 된다. 다른 Parameter를 활용하고 싶다면, Code Block에 원하는 Parameter Name을 넣어주면 된다.
Code Block [SetParameterByName] node을 이용하여 배치된 Room의 "바닥 바탕","바닥 마감","바닥 두께"의 값을 지운다. 정보가 삭제된 Room Parameter는 [E] Room의 바닥 정보 넣기와 연산 충돌을 방지하기 위해서 [Transaction.End] node를 배치한다.



B Room안에 사용된 마감 바닥 가져오기



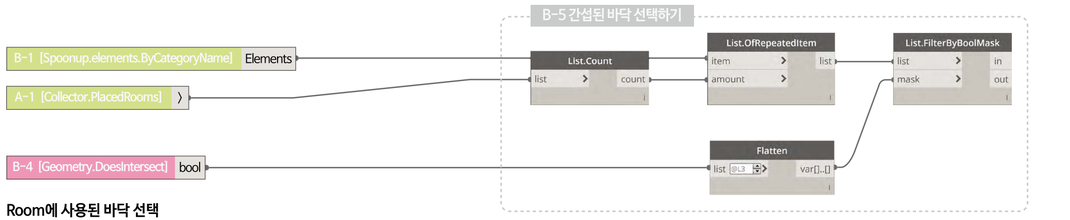
바닥의 Level Offset

현재 Project에 사용된 모든 바닥 부재를 [Spoonup.elements.ByCategoryName] node를 이용해 바닥 부재 List를 정리한다. [GetParameterValueByName]을 이용하여 바닥 부재의 Level Offset 값을 추출한다. 추출된 Level Offset 값은 [List.MinimumItem]과 [List.MaximumItem]을 이용하여 Level Offset의 최대, 최솟값을 추출한다.

Room에 사용된 바닥

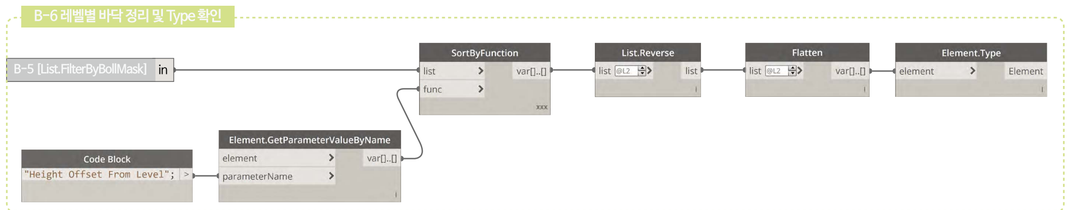
Project에 배치된 Room List를 [Room.Loaction] node에 Input 하여 Room이 배치된 중심점을 찾는다. Room의 중심점에서 [Geometry.Translate] node를 이용하여 Level Offset의 최대, 최솟값을 Input 하여 Point를 복제한다. 복제된 Point 값은 [Line.ByStartPointEndPoint]를 이용하여 수직 Line을 만든다. 수직 Line은 Room 중심에 위치한다. 마감 바닥을 모델링 할 때 복합 바닥으로 모델링 하거나, 구조 바닥, 건축 바닥과 인테리어 바닥을 분리하여 모델링 하는 경우가 있다. 바닥을 모델링 하는 방법에 상관없이, Room 중심에 위치한 Line과 간섭 되는 바닥을 추출하여 Room에 사용된 바닥으로 정리한다.

[Element.Geometry] node를 이용하여 Project에 사용된 바닥 부재를 Dynamo의 Geometry로 가져온다. 바닥 Geometry와 Room 중심에 생성된 Line을 [Geometry.DoesIntersect] node를 이용하여 간섭 체크를 진행한다.



Room에 사용된 바닥 선택

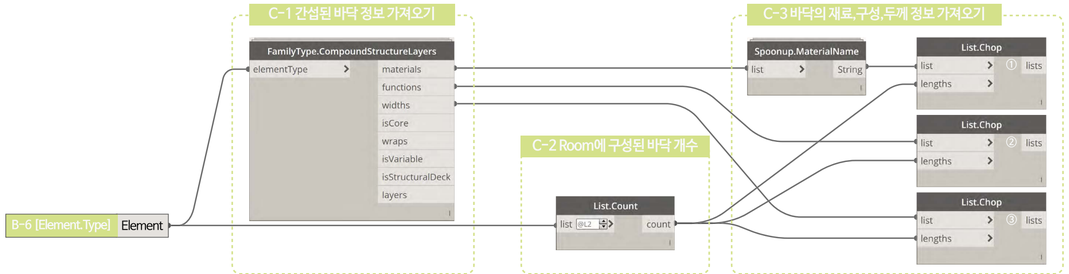
[Geometry.DoesIntersect]에 추출된 True, False 값으로 Room과 간섭된 바닥을 선택한다. 추출된 True, Flase 값은 "Project에 배치된 바닥 부재 개수 x 구획된 룸 개수"이다. 정확하게 Room과 간섭된 바닥 부재를 선택하기 위해서 [List.OfRepeatedItem] node에 Room 개수를 Input 하여 추출된 True, False 값과 선택할 바닥 개수를 같게 한다. [List.FilterByBoolMask] node를 이용하여 Room과 간섭된 바닥을 선택한다.



바닥 Type 정리

선택된 바닥이 [SortByFunction] node에서 Level Offset 값으로 정리될 수 있게 Func에 List 정리할 Parameter를 Input 한다. [SortByFunction]으로 정리된 바닥 List는 Level Offset의 오름차순으로 정리된다. 오름차순으로 정리된 List 값은 [List.Reverse] node로 실제 바닥을 구축하는 Layer 순서와 같게 List를 정리한다. 정리된 바닥 부재 [Element.Type] node를 이용하여 바닥 Type으로 정보를 치환한다.

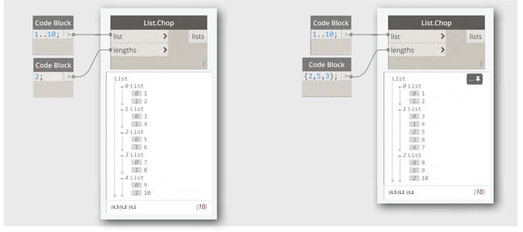
C 마감바닥 정보 가져오기



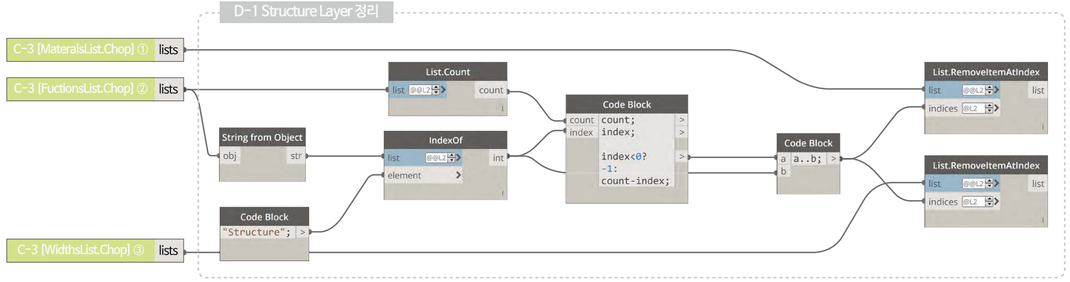
바닥 Type의 정보

바닥 Type List는 [FamilyType.CompoundStructureLayers] node를 이용하여 바닥 Type Layer를 구성하는 재료, 기능, 두께 정보를 추출한다. 추출된 재료 ID는 [Spoonup.Material Name]에 Input 하여 재료명으로 변경한다. Dynamo의 기본 Node인 [Material.Name]는 재료를 지정하지 않을 경우 오류가 발생한다. [Spoonup.MaterialName]은 재료를 지정하지 않았을 경우 "재료 없음"으로 표현되어 재료를 지정하지 않아도 오류가 발생하지 않는다.

추출된 Layer 별 재료, 기능, 두께는 [List.Chop] node로 List를 분리한다. Room에 사용된 바닥 부재가 2개 이상일 경우 [List.Chop]으로 List를 구분하면 쉽게 사용된 바닥 Type 별로 List를 얻을 수 있다. List를 분리하는 것은 [E. Room의 마감 바닥 정보 넣기]에서 바닥 부재가 2개 이상 사용된 Room은 기입할 정보 앞에 번호를 부여하도록 하기 위해서이다.



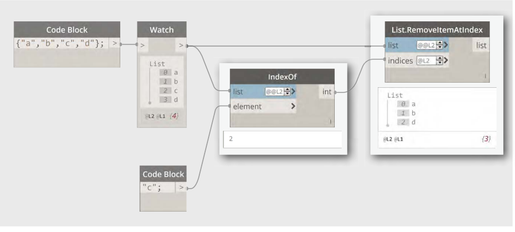
D 마감 바닥 정보 만들기

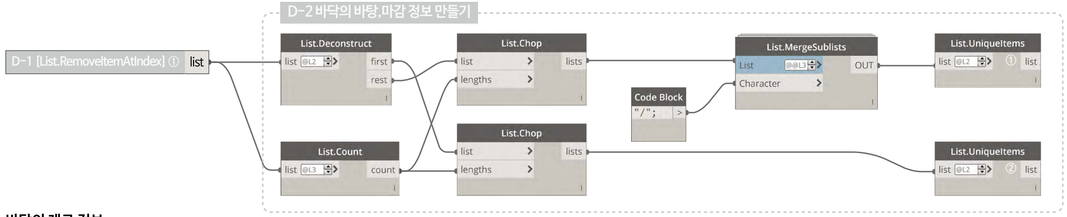


Structure Layer의 List 번호

추출된 바닥 Type을 구성하는 Layer 기능 List는 [IndexOf] node에 Input 하여 "Structure" 기능으로 지정된 Layer의 List 번호를 추출한다. "Structure"기능 Layer보다 낮은 Layer들의 정보를 삭제하여, Structure Layer 위에 구성된 정보만 바닥 바탕과 마감 정보로 이용한다. Structure Layer를 기준으로 밑에 존재하는 Layer 정보는 하부에 있는 Room의 재료일 가능성이 높기 때문이다. 실내 재료 마감 표 정보에는 구조 바닥 위 마감 바탕과 마감 정보만 표현한다. 추출된 Structure Layer의 List 번호를 [Code Block] node에 Input 하여Structure Layer가 존재한다면 그보다 높은 수를 List 개수에 맞게 만들어 삭제할 List 번호를 추출한다.

[List.RemoveItemAtIndex] node를 이용하여 List에 존재하는 Structure Layer를 기준으로 재료명 List와 두께 값 List를 정리한다.

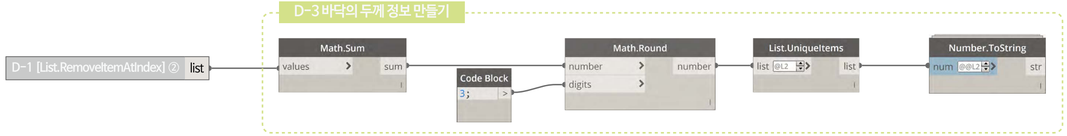




바닥의 재료 정보

Structure Layer 기준으로 정리된 재료 List는 [List.Deconstruct] node를 이용하여 마감재료와 바탕 재료들을 분리 시킨다. [List.Chop] node를 이용하여 같은 Room에 2개 이상의 마감 바닥이 모델링 되어 있을 경우를 고려하여 다시 한번 List를 정리한다.

추출된 바닥 바탕 재료들은 [List.MergeSublists]를 이용하여 바닥 Type 별 재료값을 Text "/"를 사용하여 Text 정보를 통합한다. 통합된 바닥 바탕 정보는 [List.UniqueItems] node를 이용하여 중복된 정보가 없도록 List를 정리한다.



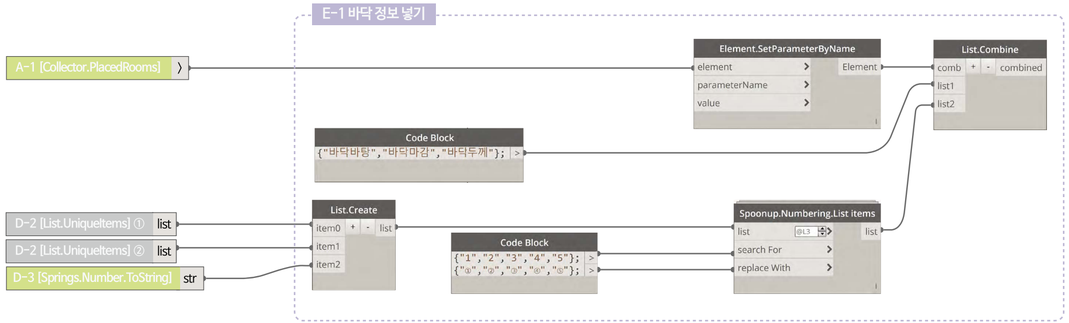
바닥의 두께 정보

실내 재료 마감 표에 기입되는 바닥 두께는 마감과 바탕을 합한 두께 값이다. 바닥 두께 값은 Structure Layer를 제외하고, 바닥 마감과 바탕 List와 동일한 Layer List의 두께만 합산 되도록 한다. 재료 두께 List는 [Math.Sum] node를 이용하여 합산한다. [List.UniqueItems] node에 Input 하기 전에 합산된 두께 값을 [Math.Round] node를 이용하여 소수점 셋째 자리를 기준으로 합산 값을 정리한다. Dynamo에서 추출한 두께 값이 소수점 6~8자리까지 표현되어 같은 두께 값으로 인식되지 않는 오류를 방지한다.

[List.UniqueItems] node로 추출된 두께 값은 [Number.ToString] node에 Input 하여 숫자를 Text 형식으로 변경한다.

Dynamo의 기본 Node인 [String from Object]를 이용해도 가능하지만 숫자에 대한 소수점 정리가 필요하기 때문에 [Number.ToString]를 이용하는 것을 추천한다.

E Room의 마감바닥 정보 넣기



실내 재료 마감표의 바닥 정보 작성

추출된 정보들은 [List.Create] node를 이용하여 단일 List로 만든다. 단일 List는 [Spoonup.Numbering.List items] node에 Input 하여 Room에 사용된 바닥이 2개 이상의 종류일 경우 앞에 Numbering을 해주는 Node이다.

[Spoonup.Numbering.List items]에 List를 Input 하면 자동적으로 1~5번까지 Numbering이 되고, 사용된 기호를 변경하고 싶다면 Replace with에 사용자가 원하는 값을 Input 하면 된다.

정리된 List는 [List.Combine] node를 이용하여 Room의 "바닥 바탕","바닥 마감","바닥 두께"Parameter의 값을 기입한다. [List.Combine]의 comb에 원하는 Dynamo 기능을 Input 하고 List1과 List2에 Dynamo 기능에 따라 연산되어야 하는 List들을 Input 한다. [E]에 사용된 [List.Combine] node는 [SetParameterByName]을 기능을 이용하여 Room 부재의 정보를 기입한다. Room의 Parameter는 "바닥 바탕","바닥 마감","바닥 두께"이고 Room Parameter List와 함께 Comb 기능에 연산될 Room에 사용된 바닥 정보 List를 Input 하면 Room의 Parameter에 정리된 값을 기입한다.

다중 Parameter에 값을 Input 할 경우 Code Block 이용하여 [SetParameterByName]를 구성하는 것보다 직관적인 [List.Combine] node를 이용하는 것을 추천한다.

CODE OVERVIEW

A 배치된 Room 벽 정보 초기화

A-1 배치된 Room 가져오기

Collector.PlacedRooms

A-2 배치된 Room 정보 지우기

Code Block
```
("배닥바탕","배닥마감","배닥마감두께");
```

Code Block
```
"";
```

Code Block
```
elemnet.SetParameterByName(name<3>,val<3>);
```

A-3 List정리

Transaction.End

B Room안에 사용된 마감벽 가져오기

B-1 마감 벽 가져오기

Code Block
```
"Walls";
```

Spoonup.elements.ByCategoryName

Spoonup.UnRoomboudingWall(EN)

B-2 벽 형태 가져오기

Element.Solids

Flatten

B-3 Room의 형태 가져오기

Element.Solids

Flatten

B-4 간섭 체크

Geometry.DoesIntersect

C 마감벽 정보 가져오기

C-1 간섭된 벽 정보 가져오기

FamilyType.CompoundStructureLayers

C-2 Room에 구성된 벽 개수

List.Count

C-3 벽의 재료,구성,두께 정보 가져오기

Spoonup.MaterialName

List.Chop

List.Chop

List.Chop

D 마감 벽 정보 만들기

D-1 Structure Layer 정리

String from Object

IndexOf

Code Block
```
a (0..(a+1))-1;
```

Code Block
```
"Structure";
```

List.RemoveItemAtIndex

List.RemoveItemAtIndex

D-2 벽의 바탕 정보 만들기

List.DropLastItem

Code Block
```
//;
```

List.MergeSublists

List.UniqueItems

D-3 벽의 마감 정보 만들기

List.LastItem

List.UniqueItems

D-4 바닥의 두께 정보 만들기

Math.Sum

Code Block
```
1;
```

Math.Round

List.UniqueItems

Number.ToString

B-5 간섭된 벽 선택하기

Element.ElementType

List.Count

List.OfRepeatedItem

List.FilterByBoolMask

List.UniqueItems

E Room의 마감벽 정보 넣기

E-1 벽 정보 넣기

List.Create

Element.SetParameterByName

Code Block
```
("벽바탕","벽마감","벽두께");
```

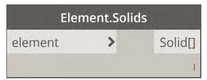
List.Combine

Code Block
```
{"1","2","3","A","S"};
{"a","s","d","f","g"};
```

Spoonup.Numbering.List items

Key Node

Dynamo basic node

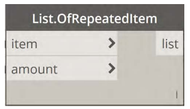


선택된 요소의 형태를 Dynamo의 Solid 형태로 변환한다.

INPUTS
- Element
  Revit의 선택한 요소를 불러온다.

OUTPUTS
- Solid[]
  Dynamo의 Solid로 변환된 정보를 확인한다.

Dynamo basic node



설정한 횟수 만큼 Item을 반복하여 List를 재구성한다.

INPUTS
- Item
  List를 생성할 Item을 불러온다.
- amount
  Item을 반복한 횟수를 불러온다.

OUTPUTS
- list
  불러온 Item List가 지정한 횟수만큼 반복 되었는지 확인한다.

custom node Spoonup Packages

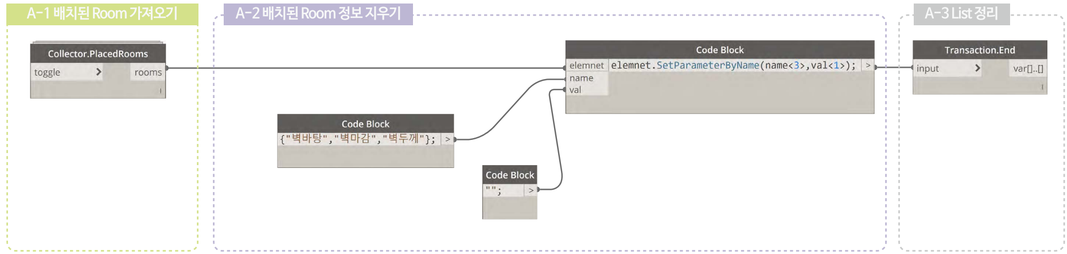


Material ID를 재료 이름으로 변환한다.

INPUTS
- list
  Material ID 를 Input한다.

OUTPUTS
- String
  재료명으로 변환한다.

A 배치된 Room의 벽 정보 초기화

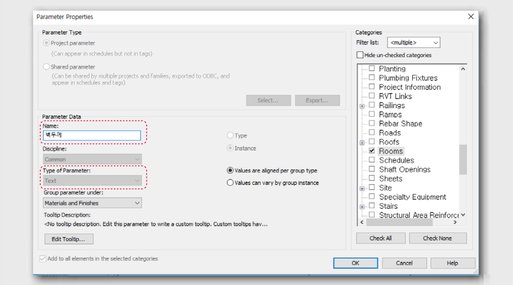


Room의 정보 초기화

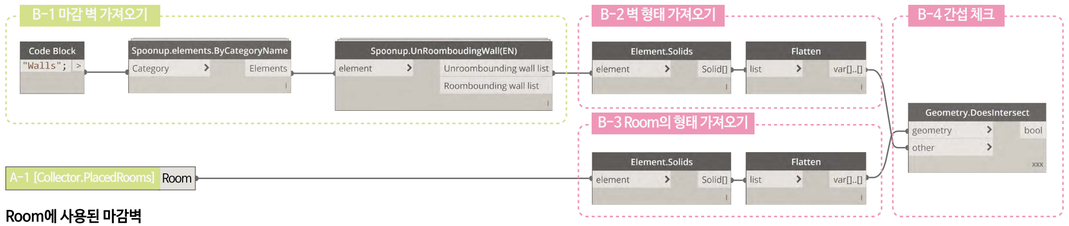
Dynamo를 이용하여 실내 재료 마감 표 작성을 위해서 기존에 작성된 실내 재료 마감 표의 정보를 초기화해야 한다. [Collector.PlacedRooms] node를 이용하여 현재 Project에 배치된 Room의 정보를 추출한다. Room은 배치되었지만 벽 또는 Room Separation Line으로 구획되지 않으면 [Collector.PlacedRooms]에서 Room을 인식하지 않는다. Project에 배치한 Room이 구획이 되어 있는지 확인이 필요하다.

Room에 설정한 Shared Parameter "벽바탕","벽마감","벽두께"는 Room Schedule에서 Room에서 사용된 벽 정보를 넣게 된다. 다른 Parameter를 활용하고 싶다면, Code Block에 원하는 Parameter Name을 넣어주면 된다.

Code Block [SetParameterByName] node를 이용하여 배치된 Room의 "벽 바탕","벽 마감","벽 두께"의 값을 지운다. 정보가 삭제된 Room Parameter는 [E] Room의 벽 정보 넣기와 연산 충돌을 방지하기 위해서 [Transaction.End] node를 배치한다.



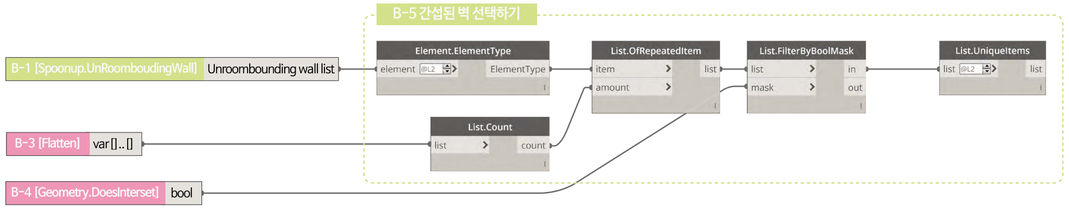
B Room안에 사용된 마감벽 가져오기



Room에 사용된 마감벽

현재 Project에 사용된 모든 벽 부재를 [Spoonup.elements.ByCategoryName] node를 이용하여 벽 부재 List를 정리한다. 정리된 벽 List는 [Spoonup.UnRoomboudingWall] node를 이용하여 Room Bounding 기능을 하지 않는 마감 벽을 선택한다. Room Bounding 기능을 하는 벽은 구조 벽인 경우가 많아 Unbounding Wall을 마감 벽으로 전제하고,Node를 진행하였다.

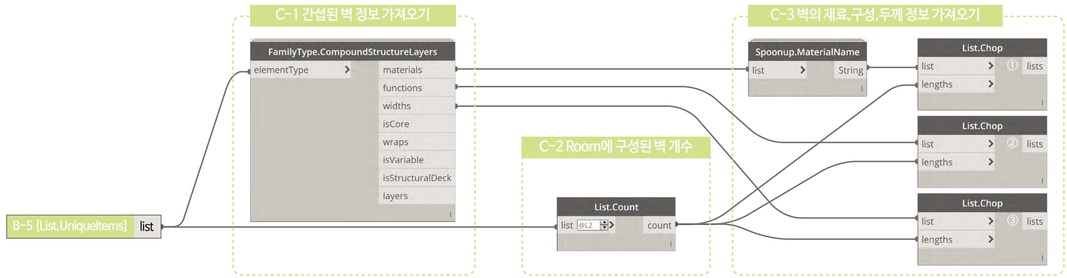
단일벽으로 구조와 마감을 함께 구성하였다면 [Spoonup.UnRoomboudingWall] node의 Room Bounding Wall List를 이용하면 된다.
추출된 벽 List를 [Element.Solids] node에 Input 하여 Dynamo에 Geometry로 가져온다. 벽 Geometry와 Room Geometry는 [Geometry.DoesIntersect] node로 간섭 체크를 진행한다.



Room에 사용된 바닥 선택

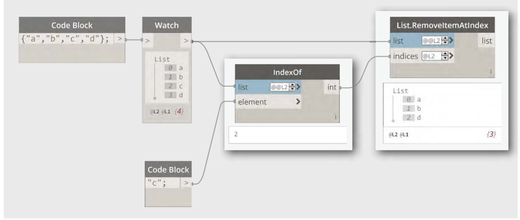
[Geometry.DoesIntersect]에 추출된 True, False 값으로 Room과 간섭된 벽을 선택한다. 추출된 True, Flase 값은 "Project에 배치된 벽 부재 개수 x 구획된 룸 개수"이다. 정확하게 Room과 간섭된 벽 부재를 선택하기 위해서 [List.OfRepeatedItem] node에 Room 개수를 Input 하여 추출된 True, False 값과 선택할 벽 개수를 같게 한다. [List.FilterByBoolMask] node를 이용하여 Room과 간섭된 벽을 선택한다.

C 마감벽 정보 가져오기

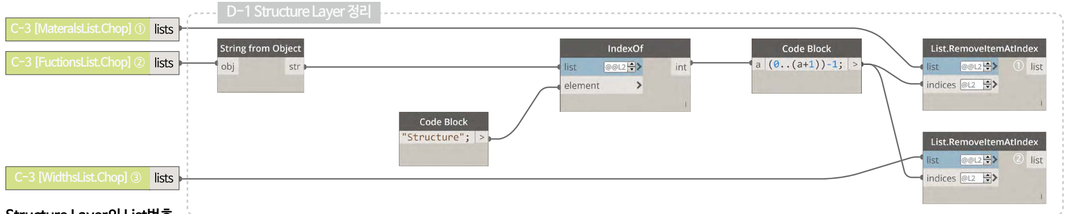


벽 Type의 정보

벽 Type List는 [FamilyType.CompoundStructureLayers] node를 이용하여 벽 Type Layer를 구성하는 재료, 기능, 두께 정보를 추출한다. 추출된 재료 ID는 [Spoonup.MaterialName]에 Input 하여 재료명으로 변경한다.
Dynamo의 기본 Node인 [Material.Name]는 재료를 지정하지 않을 경우 오류가 발생한다. [Spoonup.MaterialName]은 재료를 지정하지 않았을 경우 "재료 없음"으로 표현되어 재료를 지정하지 않아도 오류가 발생하지않는다.
추출된 Layer 별 재료, 기능, 두께는 [List.Chop] node로 List를 분리한다. Room에 사용된 벽 부재가 2개 이상일 경우 [List.Chop]으로 List를 구분하면 쉽게 사용된 벽 Type 별로 List를 얻을 수 있다. List를 구분하는 것은 [E] Room의 마감 벽 정보 넣기에서 벽 부재가 2개 이상 사용된 Room은 기입할 정보 앞에 번호를 부여하도록 하기 위해서이다.



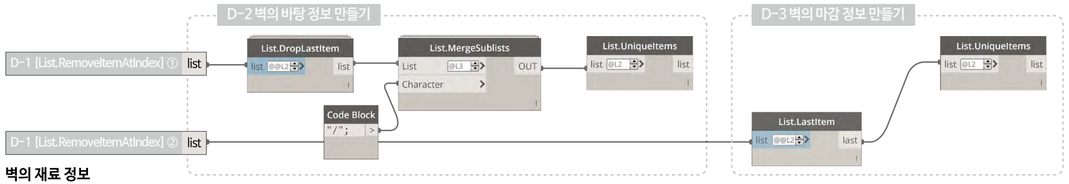
D 마감 벽 정보 만들기



Structure Layer의 List번호

추출된 벽 Type을 구성하는 Layer 기능List는 [IndexOf] node에 Input 하여 "Structure"기능으로 지정된 Layer의 List 번호를 추출한다. "Structure"기능 Layer보다 낮은 Layer들의 정보를 삭제하여, Structure Layer 위에 구성된 정보만 벽 바탕과 마감 정보로 이용한다. 실내 재료 마감표 정보에는 위 마감 바탕과 마감 정보만 표현한다. 추출된 Structure Layer의 List 번호를 [Code Block] node에 Input 하여 Structure Layer가 존재한다면 그보다 높은 수를 List 개수에 맞게 만들어 삭제할 List 번호를 추출한다.

[List.RemoveItemAtIndex] node를 이용하여 List에 존재하는 Structure Layer를 기준으로 재료명 List와 두께 값 List를 정리한다.

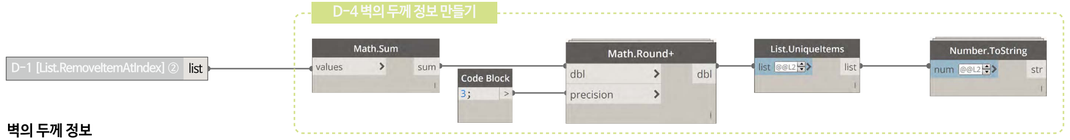


벽의 재료 정보

Structure Layer 기준으로 정리된 재료 List는 [List.Deconstruct] node를 이용하여 마감재료와 바탕 재료들을 분리 시킨다. [List.Chop] node를 이용하여 같은 Room에 2개 이상의 마감 벽이 모델링 되어 있을 경우를 고려하여 다시 한번 List를 정리한다.

추출된 벽 바탕 재료들은 [List.MergeSublists]를 이용하여 벽 Type 별 재료값을 Text "/"를 사용하여 Text 정보를 통합한다. 통합된 벽 바탕 정보는 [List.UniqueItems] node를 이용하여 중복된 정보가 없도록 List를 정리한다.

추출된 벽 마감 재료 List도 중복된 마감 재료가 없도록 [List.UniqueItems] node에 Input 하여 정리한다.

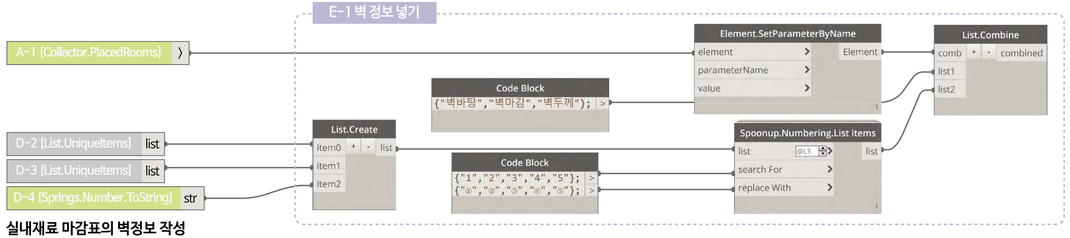


벽의 두께 정보

실내 재료 마감 표에 기입되는 벽 두께는 마감과 바탕을 합한 두께 값이다. 벽 두께 값은 Structure Layer를 제외하고, 벽 마감과 바탕 List와 동일한 Layer List의 두께만 합산 되도록 한다. 재료 두께 List는 [Math.Sum] node를 이용하여 합산한다. [List.UniqueItems] node에 Input 하기 전에 합산된 두께 값을 [Math.Round] node를 이용하여 소수점 셋째 자리를 기준으로 합산 값을 정리한다. Dynamo에서 추출한 두께 값이 소수점 6~8자리까지 표현되어 같은 두께 값으로 인식되지 않는 오류를 방지한다. [List.UniqueItems] node로 추출된 두께 값은 [Spring.NumberToString] node에 Input 하여 숫자를 Text 형식으로 변경한다.

Dynamo의 기본 Node인 [String from Object]를 이용해도 가능하지만 숫자에 대한 소수점 정리가 필요하기 때문에 [Number.ToString]를 이용하는 것을 추천한다.

E Room의 마감벽 정보 넣기



실내재료 마감표의 벽정보 작성

[List.Create]로 작성된 단일 List는 [Spoonup.Numbering.List items] node에 Input 한다. [Spoonup.Numbering.List items]에 List를 Input 하면 자동적으로 1~5번까지 Numbering이 되고, 사용된 기호를 변경하고 싶다면 Replace with에 사용자가 원하는 값을 Input 하면 된다.

정리된 List는 [List.Combine] node를 이용하여 Room의 벽 Parameter의 값을 기입한다. [List.Combine]의 comb에 원하는 Dynamo 기능을 Input 하고 list1과 list2에 Dynamo 기능에 따라 연산되어야 하는 List들을 Input 한다. 다중 Parameter에 값을 Input 할 경우 Code Block 이용하여 [SetParameterByName]를 구성하는 것보다 직관적인 [List.Combine] node를 이용하는 것을 추천한다.

CODE OVERVIEW

A 배치된 Room 천장 정보 초기화

A-1 배치된 Room 가져오기

Collector.PlacedRooms

A-2 배치된 Room 정보 지우기

Code Block
```
{"천장바탕","천장마감","천장크기"};
```
Code Block
```
"";
```
Code Block
```
elemnet.SetParameterByName(name<3>,val<1>);
```

A-3 List 정리

Transaction.End

B Room안에 사용된 마감 천장 가져오기

B-1 마감 천장 가져오기

Code Block
```
"Ceilings";
```
Spoonup.elements.ByCateogryName

B-2 천장 형태 가져오기

Element.Solids
Flatten

B-3 Room의 형태 가져오기

Element.Solids
Flatten

B-4 간섭 체크

Geometry.DoesIntersect

B-5 간섭된 천장 선택하기

Element.ElementType
List.OfRepeatedItem
List.FilterByBoolMask
List.UniqueItems
List.Count

C 천장 정보 가져오기

C-1 간섭된 천장 정보 가져오기

FamilyType.CompoundStructureLayers
materials
functions
widths
isCore
wraps
isVariable
isStructuralDeck
layers

C-2 Room에 구성된 천장 개수

List.Count

C-3 천장의 재료,구 성,두께 정보 가져오기

Spoonup.MaterialName
List.Chop

D 천장 정보 만들기

D-1 천장의 바탕 정보 만들기

List.DropLastItem
List.MergeSublists
Character
Code Block
```
"/";
```
List.UniqueItems

D-2 천장의 마감 정보 만들기

List.LastItem
List.UniqueItems

D-3 천장고 정보 만들기

List.Chop
Element.GetParameterValueByName
parameterName
List.UniqueItems
Springs.Number.ToString
Code Block
```
"Height Offset From Level";
```

B-7 천장이 배치된 Room 찾기

List.IsEmpty
List.FilterByBoolMask

B-6 간섭된 천장의 Instance 선택하기

List.OfRepeatedItem
List.FilterByBoolMask
List.UniqueItems
Flatten

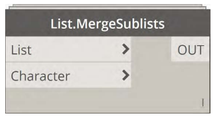
E Room의 마감 천장 정보 넣기

E-1 천장 정보 넣기

List.Create
Element.SetParameterByName
parameterName
value
Code Block
```
{"벽바탕","벽마감","벽두께"};
```
List.Combine
Code Block
```
{"1","2","3","4","5"};
{",",",",",",",","};
```
Spoonup.Numbering.List items
search For
replace With

Key Node

Dynamo basic node



선택된 List를 특정 글자 기준으로 합쳐준다.

INPUTS
- List
  Text List를 불러온다.
- Character
  LIst 정렬 기준을 부여한다.

OUTPUTS
- OUT
  Input 된 Character List를 기준으로 정리된 List 값을 확인한다.

Dynamo basic node

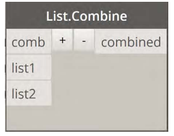


Input된 List 정보 앞에 번호 순서를 부여한다.

INPUTS
- list
  번호를 부여한 List를 Input한다.
- search For, replace With
  부여된 번호 형태(1,2,3,..,)를 변경 하고 싶은 형태 정보를 Input한다.

OUTPUTS
- list
  List에 Numbering을 부여한다.

custom node Spoonup Packages

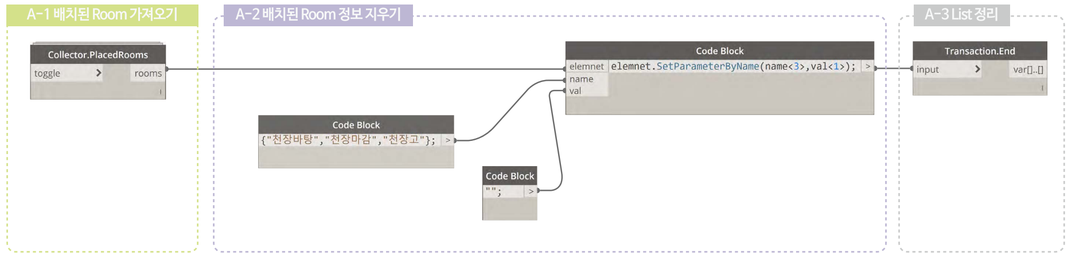


Comb에 설정된 방식으로 List들을 결합한다.

INPUTS
- comb (list1, list2)
  List를 결합할 방식과 List들을 Input한다.

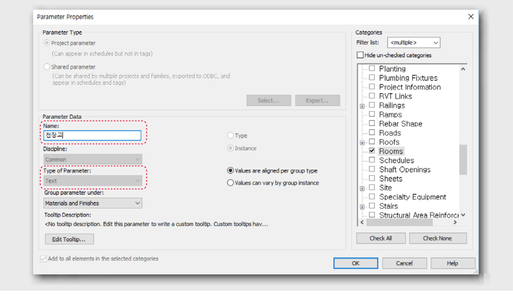
OUTPUTS
- combinator
  두 요소를 결합하여 List를 생성한다.

A 배치된 Room 천장 정보 초기화

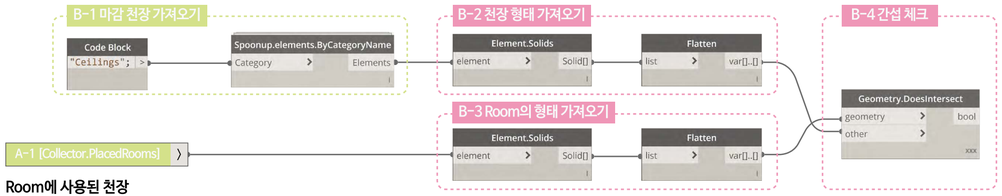


Room의 정보 초기화

Dynamo를 이용하여 실내 재료 마감 표 작성을 위해서 기존에 작성된 실내 재료 마감 표의 정보를 초기화해야 한다. [Collector.PlacedRooms] node를 이용하여 현재 Project에 배치된 Room의 정보를 추출한다. Room은 배치되었지만 벽 또는 Room Separation Line으로 구획되지 않으면 [Collector.PlacedRooms]에서 Room을 인식하지 않는다. Project에 배치한 Room이 구획이 되어 있는지 확인이 필요하다.
Room에 설정한 Shared Parameter "천장 바탕","천장 마감","천장고"는 Room Schedule에서 Room에서 사용된 천장 정보를 넣게 된다. 다른 Parameter를 활용하고 싶다면, Code Block에 원하는 Parameter Name을 넣어주면 된다.
Code Block [SetParameterByName] node를 이용하여 배치된 Room의 "천장 바탕","천장 마감","천장고"의 값을 지운다. 정보가 삭제된 Room Parameter는 [E] Room의 천장 정보 넣기와 연산 충돌을 방지하기 위해서 [Transaction.End] node를 배치한다.



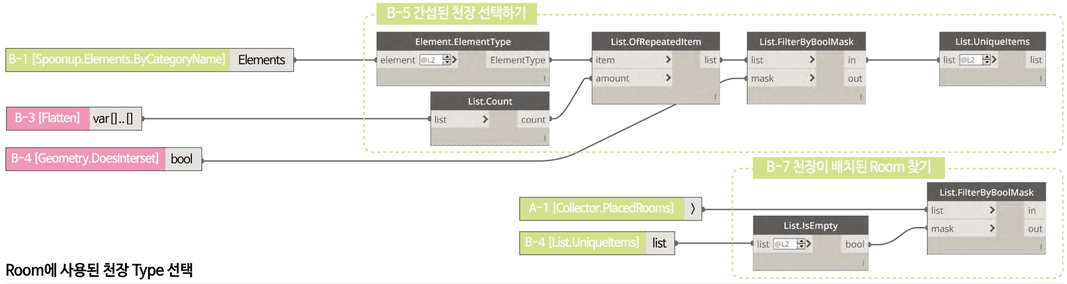
B Room안에 사용된 천장 가져오기



Room에 사용된 천장

현재 Project에 사용된 모든 천장 부재를 [Spoonup.elements.ByCategoryName] node를 이용하여 천장 부재 List를 정리한다.

추출된 천장 List를 [Element.Solids] node에 Input 하여 Dynamo에 Geometry로 가져온다. 천장 Geometry와 Room Geometry는 [Geometry.DoesIntersect] node로 간섭 체크를 진행한다.



Room에 사용된 천장 Type 선택

[Geometry.DoesIntersect]에 추출된 True, False 값으로 Room과 간섭된 천장을 선택한다. 추출된 True, Flase 값은 "Project에 배치된 천장 부재 개수 x 구획된 룸 개수"이다. 정확하게 Room과 간섭된 천장 부재를 선택하기 위해서 [List.OfRepeatedItem] node에 Room 개수를 Input 하여 추출된 True, False 값과 선택할 천장 개수를 같게 한다. [List.FilterByBoolMask] node를 이용하여 Room과 간섭된 천장을 선택한다.

천장이 배치된 Room을 찾기 위해서 [List.IsEmpty]에 [B-5]에서 추출한 List를 Input 하여 추출된 True, False 값을 [List.FilterByBoolMask]에 Input 하여 배치된 Room 중 천장 부재가 있는 Room만 선택한다.

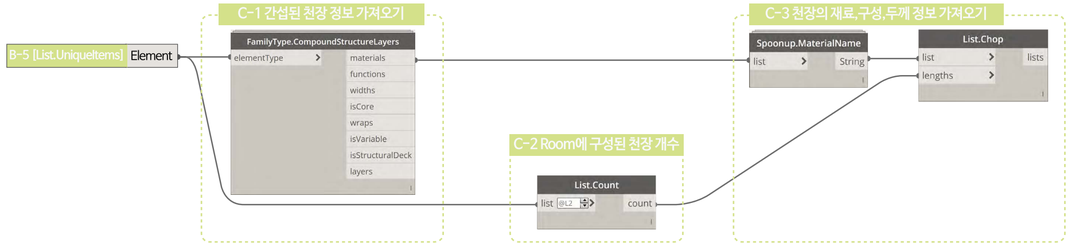


Room에 사용된 천장 Instance 부재

실내 재료 마감 표에서 바닥, 벽과 다르게 천장은 천장고를 기입한다. 천장고 추출을 위해서 모델링 된 천장 부재의 Level Offset 정보를 추출한다. 모델링된 천장부재의 높이는 Type정보와 관련이 없다.

Type정보가 아닌 Instance 부재를 선택해야 천장 부재의 높이 값을 확인 할 수 있다. [List.FilterByBoolMask] node를 이용하여 모델링된 천장을 Instance로 선택한다.

C 천장 정보 가져오기

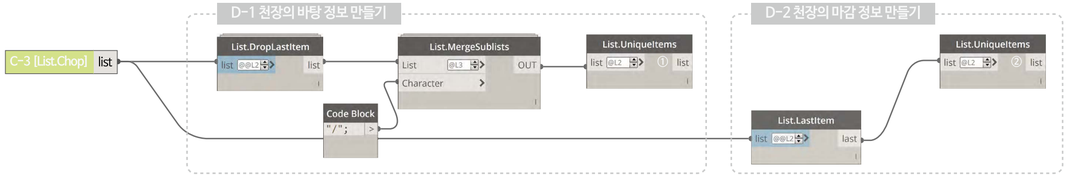


천장 Type의 정보

천장 Type List는 [FamilyType.CompoundStructureLayers] node를 이용하여 천장 Type Layer를 구성하는 재료 정보를 추출한다. 추출된 재료 ID는 [Spoonup.MaterialName]에 Input 하여 재료명으로 변경한다.

추출된 Layer 별 재료는 [List.Chop] node로 List를 분리한다. Room에 사용된 천장 부재가 2개 이상일 경우 [List.Chop]으로 List를 구분하면 쉽게 사용된 천장 Type 별로 List를 얻을 수 있다. List를 분리하는 것은 [E] Room의 천장 정보 넣기에서 천장 부재가 2개 이상 사용된 Room은 기입할 정보 앞에 번호를 부여하도록 하기 위해서이다.

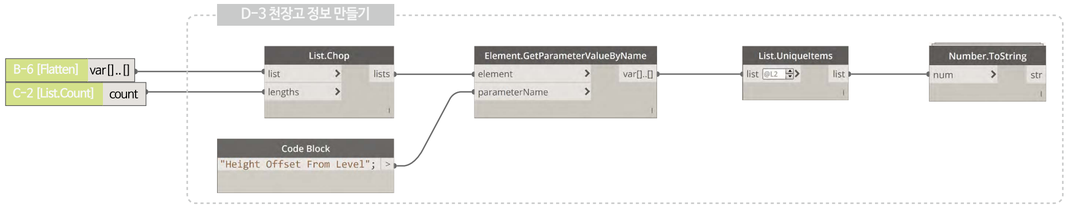
D 천장 정보 만들기



천장의 재료 정보

추출된 천장 바탕 재료들은 [List.MergeSublists]를 이용하여 천장 Type 별 재료값을 Text "/"를 사용하여 Text 정보를 통합한다. 통합된 천장 바탕 정보는 [List.UniqueItems] node를 이용하여 중복된 정보가 없도록 List를 정리한다.

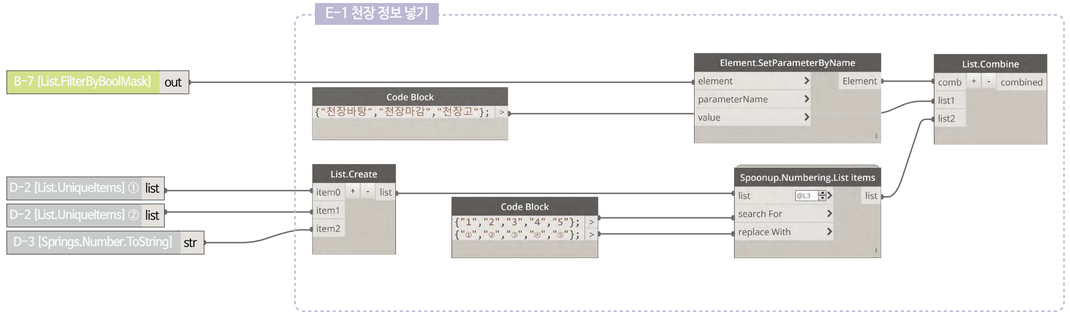
추출된 천장 마감 재료 List도 중복된 마감 재료가 없도록 [List.UniqueItems] node에 Input 하여 정리한다.



천장고

실내 재료 마감 표에 표기되는 천장고 정보를 얻기 위해서 [GetParameterValueByName]를 이용하여 모델링 된 천장의 Level Offset 값을 추출한다. 추출된 높이 값은 [List.UniqueItems]으로 중복된 값을 정리한다. 정리된 List는 [Number.ToString]에 Input 하여 숫자를 Text 형식으로 변경한다.

E Room의 마감천장 정보 넣기



실내재료 마감표의 천장 정보 작성

추출된 정보들은 [List.Create]를 이용하여 단일 List로 만든다.

단일 List는 [Spoonup.Numbering.List items] node에 Input 하여 Room에 사용된 천장이 2개 이상의 종류일 경우 앞에 Numbering을 해주는 Node이다.

[Spoonup.Numbering.List items]에 List를 Input 하면 자동적으로 1~5번까지 Numbering이 되고, 사용된 기호를 변경하고 싶다면 replace with에 사용자가 원하는 값을 Input 하면 된다.

정리된 List는 [List.Combine] node를 이용하여 Room의 "천장 바탕","천장 마감","천장고"Parameter의 값을 기입한다. [List.Combine]의 comb에 원하는 Dynamo 기능을 Input 하고 list1과 list2에 Dynamo 기능에 따라 연산되어야 하는 List들을 Input 한다. [E]에 사용된 [List.Combine] node는 [SetParameterByName]을 기능을 이용하여 Room 부재의 정보를 기입한다. Room의 Parameter는 "천장 바탕","천장 마감","천장고"이고 Room Parameter List와 함께 Comb 기능에 연산될 Room에 사용된 천장 정보 List를 Input 하면 Room의 Parameter에 정리된 값을 기입한다.

다중 Parameter에 값을 Input 할 경우 Code Block 이용하여 [SetParameterByName]를 구성하는 것보다 직관적인 [List.Combine] node를 이용하는 것을 추천한다.

2. Dynamo Code
구조 도면

S1. Slab Level 기준 범례 만들기

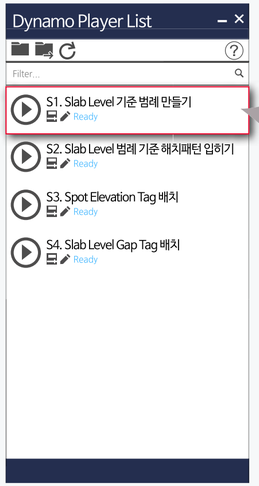
동영상 : https://youtu.be/oyZVHtwuM1I

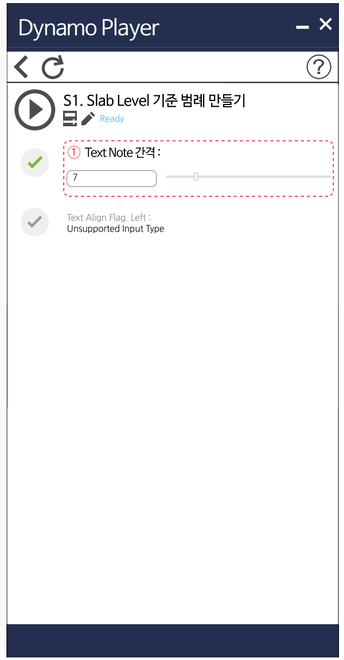
기능 및 필요성

구조 평면도 작업시 사용된 "Structure Floor Level"을 기준으로 범례를 만들어주는 기능이다. 설계 진행시 변경되는 Level에 따라서 범례는 추가, 삭제 하는 작업을 반복한다. Dynamo Code를 이용할 경우 범례를 자동으로 Update되어 구조 평면도 작업시 시간을 줄일 수 있다.

유의사항

- 작업 화면 : Legend View에서 구동되며 기준이 되는 Filled Region와 Text Note가 1개씩 배치되어야 한다.
- 또한 Level 기준으로 생성된 Filled Region의 해치는 사용자가 지정해 주어야 한다.
- Floor의 Structral Parameter에 반드시 체크가(▣) 되어 있어야 한다.

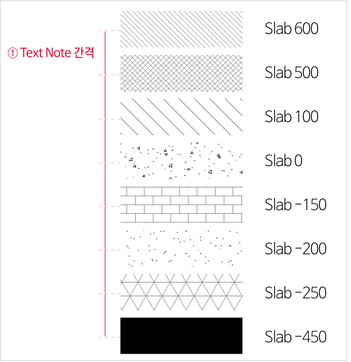


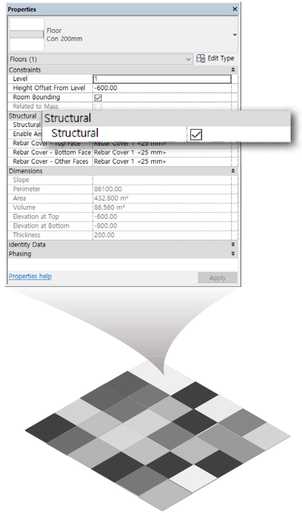


* Dynamo Player는 Revit 2017부터 사용 가능하며, Player의 설정창은 Revit 2018 이상 버전에서 사용할 수 있다.
Revit 2017일 경우 Dynamo 파일을 열고 설정을 수정하여 저장한 뒤 Player를 누르거나 Dynamo에서 바로 실행시킨다.

Level 기준 Filled Region과 Text Note 생성을 위한 parameter 조정

① Text Note 간격
범례의 Text Note 객체간 간격을 조정합니다

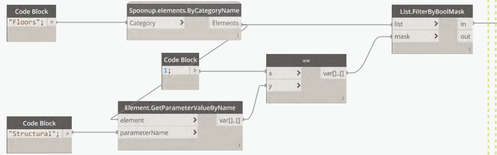




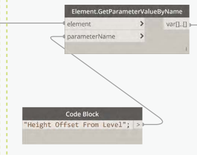
CODE OVERVIEW

B 사용된 Window 객체의 룸 정보 추출 및 정보 입력

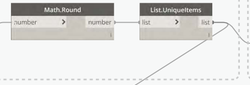
B-1 배치된 Floor중 Structural Slab 추출



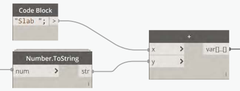
B-2 Floor의 Level 정보 추출



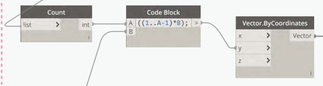
B-3 List 정리



B-4 표현 할 정보 조합을 Text로 만들기

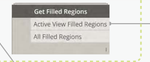


B-5 배치 할 Filled Region & Text Note 좌표 만들기



C Filled Region 만들기

C-1 현재 View에 배치된 Filled Regions 찾기



C-2 List 정리



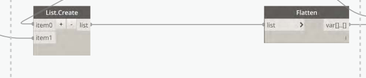
C-4 새로운 Filled Region Type 만들기



C-5 배치된 Filled Region 복사



C-6 List 정리



C-3 불필요한 Filled Region 삭제



C-8 배치된 Filled Region에 정보 입력



C-7 기존 or 생성된 Filled Region 찾기



A Text notes 간격 설정 및 View scale 확인

A-1 Text Note 간격 설정



A-2 View Scale 정보 추출

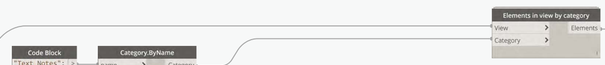


A-3 Scale 맞게 간격 변경



D Text Note 만들기

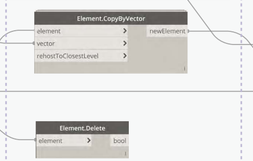
D-1 현재 View에 사용된 Text Note 찾기



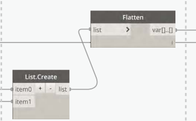
D-2 List 정리



D-3 불필요한 Text Note 삭제 및 복사



D-4 List 정리



D-5 Text Note 작성



Key Node

custom node **Archi-lab Packages**



활성화된 View에서 배치된 모든 Fillded Region을 가지고 온다.

INPUTS

OUTPUTS
- Active View Filled Regions
  활성화 View에서 Filled Region의 정보를 가지고 온다.
- All Filled Regions
  모든 Filled Region의 정보를 가지고 온다.

custom node **Clockwork Packages**

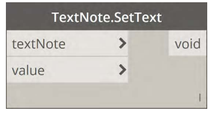


Vector 값을 부여하여 요소 혹은 요소의 List를 복사 한다.

INPUTS
- element
  요소 혹은 요소의 List를 불러 온다.
- vector
  불러온 요소에 vector 값을 부여한다.

OUTPUTS
- newElement
  불러온 요소 혹은 요소의 List에 Vector 값을 부여하여 복사한다.

Dynamo basic node

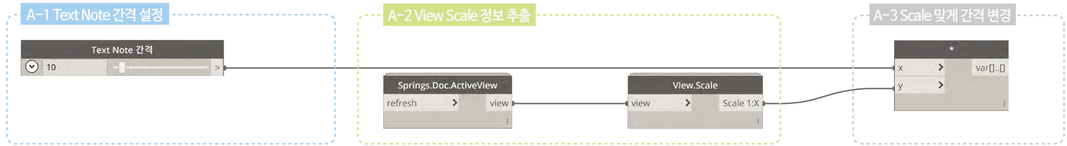


textNote에 value 값을 기입한다.

INPUTS
- textNote
  내용을 변경할 Text Note을 선택 한다.
- value
  Text 내용을 Input한다.

OUTPUTS
- combinator
  부여할 Text에 대한 정보와 정렬 값을 기입하여 내보낸다.

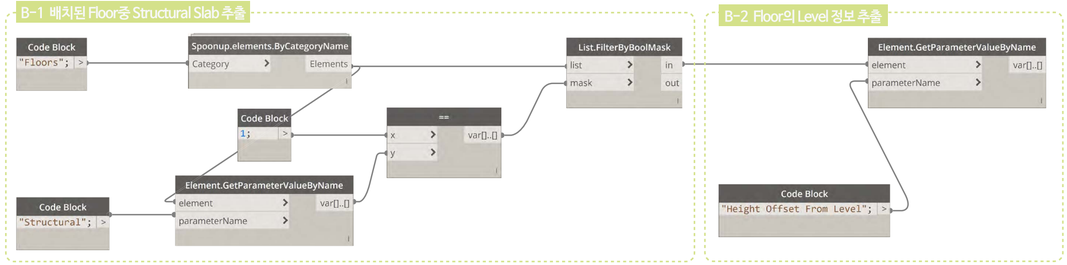
A Text Notes 간격 설정 및 View Scale 확인



Parameter에 따른 정보가공

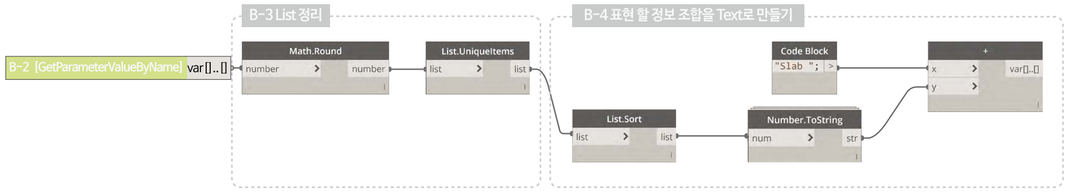
Text Note와 Filled Region의 간격을 [Integer Slider]를 이용하여 설정한다. [Spring.Doc.AtiveView]를 이용해 현재 열려있는 Legend View의 정보를 가져오고 [View.Scale]을 통해 Legend View의 Scale정보를 가져온다. Text Note와 Filled Region의 간격은 Scale과 연산하여 Legend View에 사용될 값을 추출한다.

B 사용된 Window 객체의 룸 정보 추출 및 정보 입력



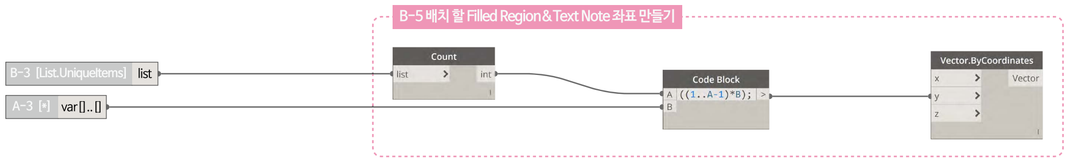
Structural Floor의 정보

[List.FilterByBoolMask]의 Mask에 "Structural" Parameter의 체크 여부를 판별할 수 있는 node를 Input하여 Structural Floor만 추출한다. 추출된 Structural Floor는 [GetParameterVlaueByName]을 이용하여 해당 Level의 Offset된 값을 가져온다.



Text의 정보조합

[Get.ParameterValueByName]으로 추출된 Structural Floor의 Level Offset 값은 [Math.Round]를 이용하여 정수로 정리한다. [List.UniqueItem] node로 각 Level에 존재하는 Structural Floor의 Level Offset 값을 단일 List로 정리한다. 정리된 Level Offset 값은 [List.Sort]를 통해서 오름차순으로 List를 정리하고 [Number.ToString]을 이용하여 Text형식으로 변환한다. Dynamo의 기본 node인 [Object from String]도 숫자를 Text 형식으로 변환할 수 있지만, [Object from String] node는 숫자의 소수점 까지 Text로 표현한다.Spring Package node인 [Number.toString]을 추천한다. [+] node를 이용하여 "slab"+"Level Offset" Text 조합은 [C] Filled Region 만들기에서 Type Name으로 활용한다.

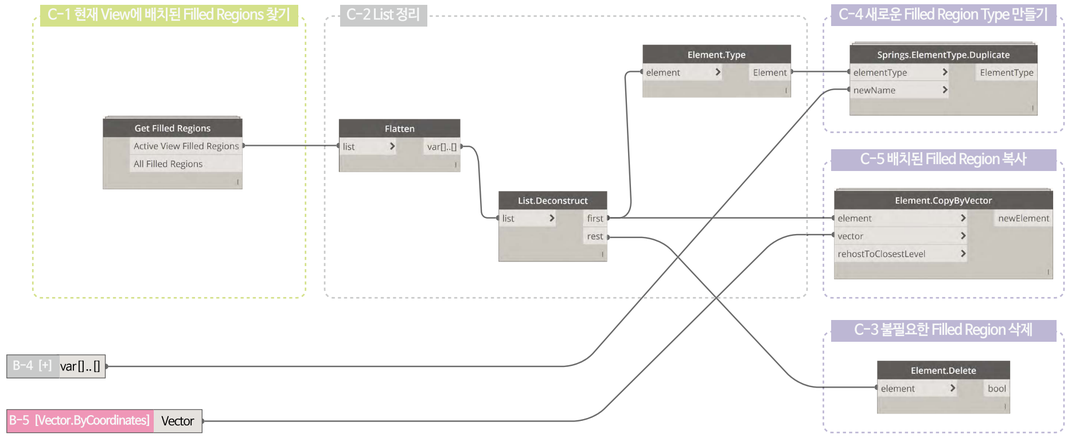


Legend View의 좌표 작성

[B-4]에서 Structural Floor의 Level Offset을 오름차순으로 정리하였다. Level 값이 낮은 순으로 Filled Region이 배치되도록, 좌표를 Code Block으로 만든다. Code Block [(1...A-1)*B]는 Filled Region이 아래에서 위로 배치될 수 있도록 Y좌표 값을 만든다.

[A2. 창호 일람표 만들기]의 Legend Component 배치시 X.Y좌표를 인지하지 않은 것처럼 Filled Region 배치시에도 X.Y좌표 값을 Vector 로 변환해서 사용한다.

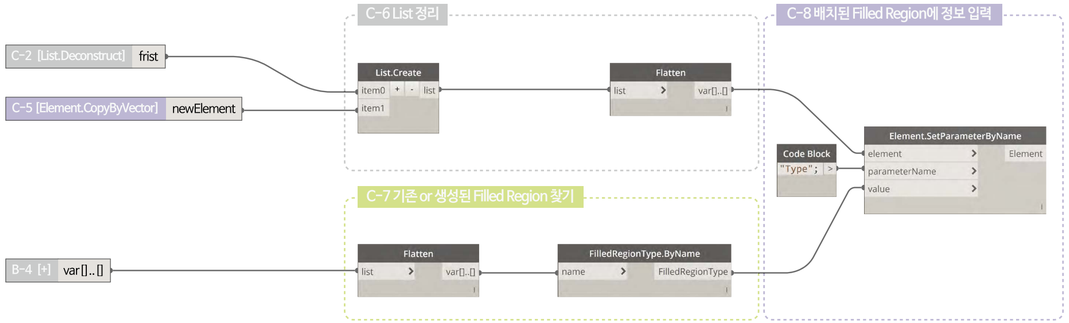
C Filled Region 만들기



Filled Region 배치

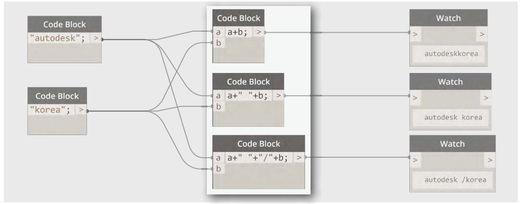
[Get Filled Regions] node를 이용하여 현재의 보이는 Legend View에 배치된 Filled Region의 정보를 가지고 온다. [A2. 창호 일람표 만들기]의 Legend Component처럼 Filled Region을 Dynamo에서 설정한 좌표와 Curve를 이용해 만들 수 없다. 기준이 되는 하나의 Filled Region과 Text Notes가 배치되어야 한다.

[Get Filled Regions]으로 가져온 배치된 Filled Region은 [Flatten]으로 단일 List로 정리한 뒤 , [List. Deconstruct]를 이용하여 처음 배치된 Filled Region을 추출한다. 처음 배치된 Filled Region만 남기고 나머지 Filled Region은 삭제한다. 새로운 Level이 생기거나 삭제되었을 때 업데이트를 쉽게 하기 위해서이다. 선택된 Filled Region의 Type은 [Springs.ElementType.Duplicate]으로 새로운 Filled Region Type으로 복사한다. 새로 생성되는 Filled Region은 "slab"+"Level Offset"을 Type Name으로 한다. 또한 선택된 Filled Region은 [Element.CopyByVector] node에 [Vector.ByCoordinates]를 Input하여 아래에서 위로 복사 배치되게 한다.

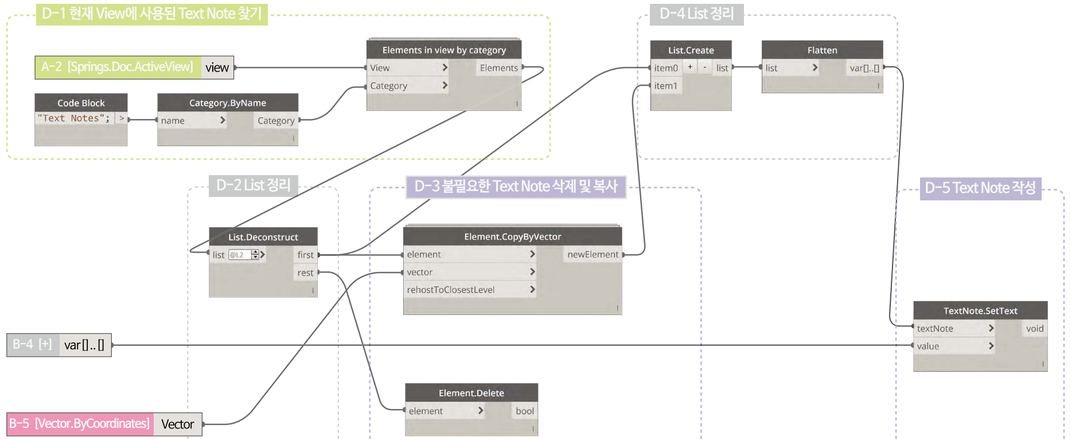


Filled Region Type 설정

[List.Create]를 통해 배치에 기준이 되는 Filled Region과 새롭게 복사 배치된 Filled Region의 List를 합친다. List는 [Flatten] node를 이용해 단일 List로 정리한다. [B-4]에서 조합한 Filled Region Name을 [Filled RegionType.ByName]에 Input하여 Filled Region Type을 선택한다. [SetParameterByName]을 이용하여 Filled Region의 Type 정보를 'slab'+'Level Offset'이라는 Type명으로 갖는 Filled Region Type으로 변경한다.



D Text Note 만들기



Text Note 작성

현재 view에 배치된 Text Note를 찾기 위해서 [Elements in view by category] node를 이용한다. Text Notes에서도 Filled Region과 마찬가지로 기준이 될 하나의 Text Note를 [List.Deconstruct]로 선택하고 나머지 Text Notes는 삭제 시킨다.

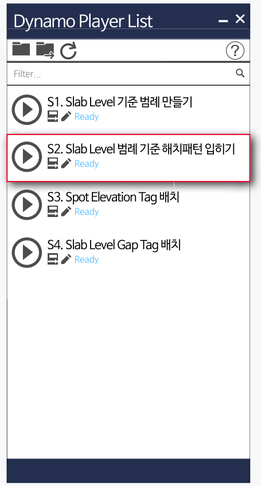
선택된 Text Notes는 [Element.CopyByVector]를 통해서 Filled Region과 마찬가지로 아래에서 위로 복사 배치 한다. [TextNote.SetText] node를 이용하여 Text Notes에 배치된 Filled Region의 Type 명을 기입한다.

S2. Slab Level 범례기준 해치 패턴 입히기

동영상 : https://youtu.be/SLe_AYzZvqA

기능 및 필요성

Revit에서 구조 도면 작성시 View Template을 활용하여 작성한다. 이때 Visibility 옵션의 Filter 기능을 이용하여 Slab Level 별 해치를 설정하는 경우가 있다. 변경된 Slab의 높이에 따라 계속 Filter를 설정해야 하며, 해치 패턴의 종류를 지정 하는 작업이 반복된다. 해당 파트의 기능은 "S1. Slab Level 기준 범례 만들기"에 의하여 작성된 Legend view안의 Slab Level 범례기준을 평면도의 Filter를 만들어준다. Legend view의 범례가 변경될 경우 재실행 시키면 Filter는 자동으로 Update 된다.

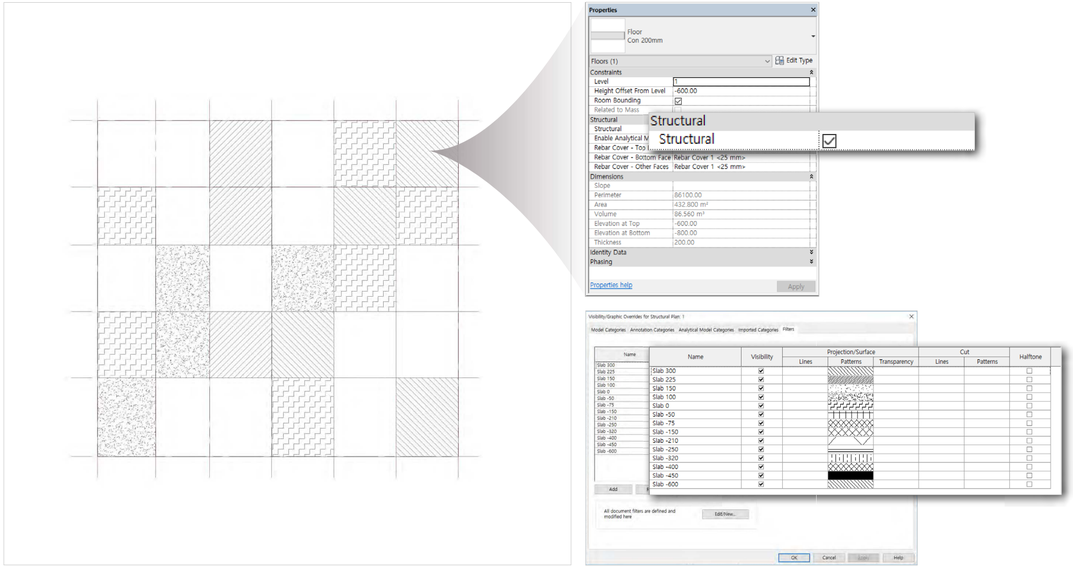


유의사항

- 작업 화면 : Slab Level Hatch를 배치할 평면 View에서 구동한다.

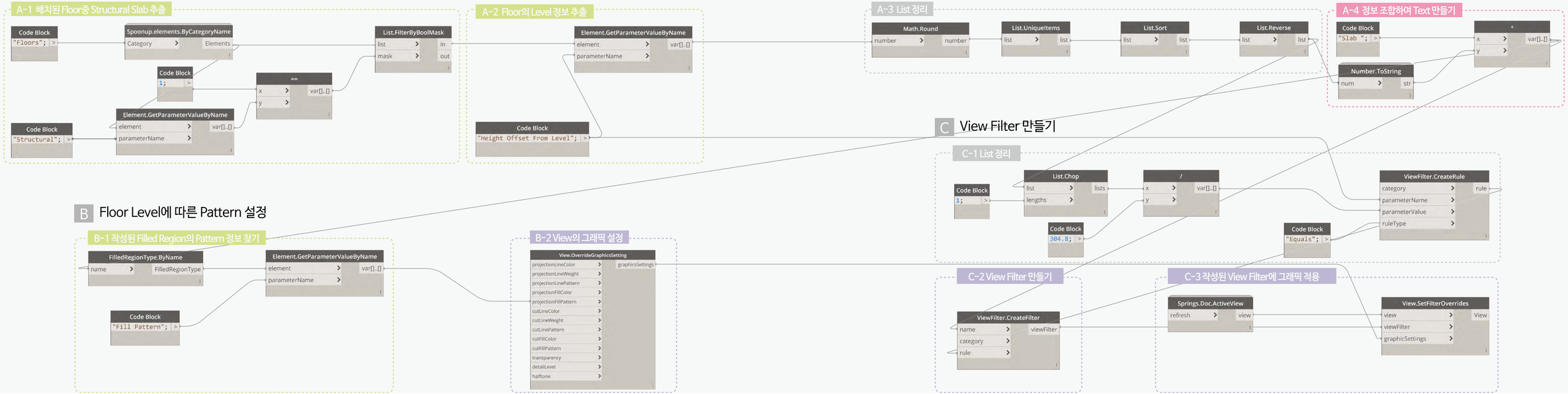
- "S1. Slab Level 기준 범례 만들기"가 사전에 작성되어 있어야 한다.
- Slab Level 값이 update되면 S1 Code부터 실행 후 S2. Code를 재실행 한다.
- Floor의 Structural Parameter에 반드시 체크가(■) 되어 있어야 한다.

* Dynamo Player는 Revit 2017부터 사용 가능하며, Player의 설정창은 Revit 2018 이상 버전에서 사용할 수 있다.
Revit 2017일 경우 Dynamo 파일을 열고 설정을 수정하여 저장한 뒤 Player를 누르거나 Dynamo에서 바로 실행시킨다.



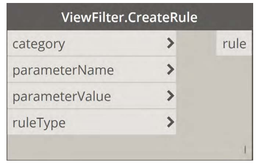
CODE OVERVIEW

A 현재 View의 Floor Level 정보 추출



Key Node

custom node **Archi-lab Packages**



View Filter에 Parameter 규칙을 생성한다.

INPUTS
- category, parameterName,Value
  규칙을 적용할 Category,Parameter Name & Value 정보를 기입한다.
- ruleType
  연동시킨 내용의 규칙을 부여한다.

OUTPUTS
- rule
  Input된 정보를 바탕으로 작성된 Rule 정보를 확인한다.

custom node **Clockwork Packages**

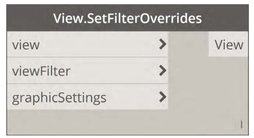


Category와 Rule에 의하여 View Filter를 생성한다.

INPUTS
- name, category
  View Filter의 이름을 생성하며, Category의 List를 연결 시킨다.
- rule
  Name과 Category의 List 규칙을 부여한다.

OUTPUTS
- viewFilter
  View Filter의 이름, Category의 리스트에 규칙을 적용하여 생성한다.

Dynamo basic node

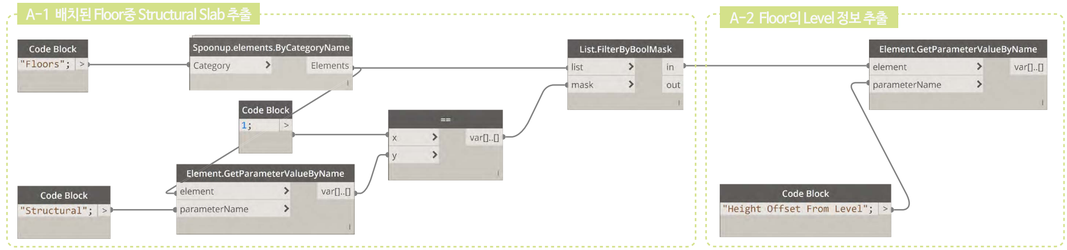


View, ViewFilter, GraphicSettings을 Filter에 부여한다.

INPUTS
- view, viewFilter
  활성화된 View 혹은 ViewFilter를 불러온다.
- graphicSettings
  Graphic 값을 설정한 Node를 불러온다.

OUTPUTS
- View
  Filter에 View, Graphic 설정 값을 부여하여 내보낸다.

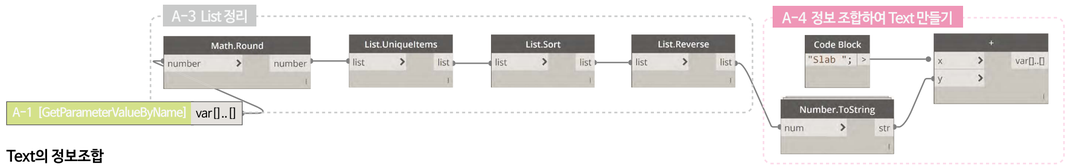
A 현재 View의 Floor Level 정보 추출



Structural Floor의 정보

[List.FilterByBoolMask]의 Mask에 "Structural" Parameter의 체크 여부를 판별할 수 있는 node를 Input하여 Structural Floor만 추출한다.

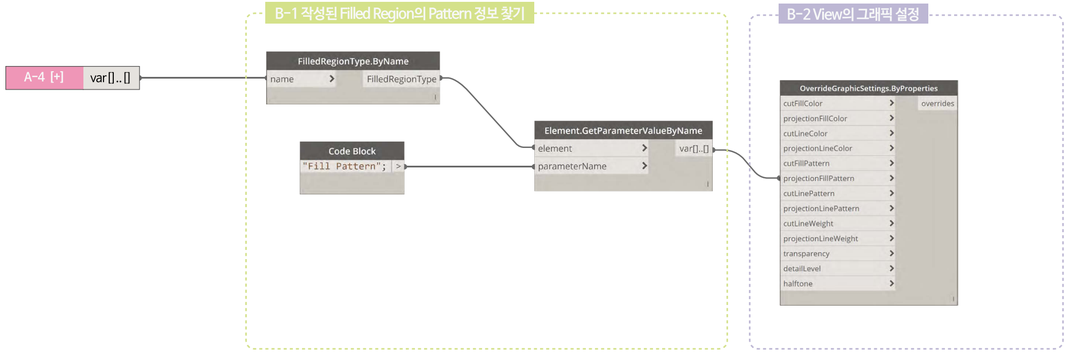
추출된 Structural Floor는 [GetParameterValueByName]을 이용하여 해당 Level의 Offset된 값을 가져온다.



Text의 정보조합

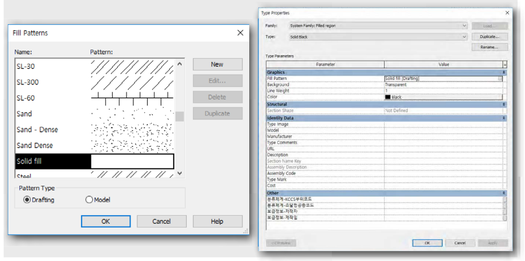
[GetParameterValueByName]으로 추출된 Structural Floor의 Level Offset 값은 [Math.Round]를 이용하여 정수로 정리한다. [List.UniqueItem] node로 각 Level에 존재하는 Structural Floor의 Level Offset 값을 단일 List로 정리한다. 정리된 Level Offset 값은 [List.Sort]를 통해서 오름차순으로 List를 정리하고 [Number.ToString]을 이용하여 Text형식으로 변환한다. Dynamo의 기본 node인 [Object from String]도 숫자를 Text 형식으로 변환할 수 있지만, [Object from String] node는 숫자의 소수점 까지 Text로 표현한다.Spring Package node인 [Number.ToString]을 추천한다. [+] node를 이용하여 "slab"+"Level Offset" Text 조합은 View의 Filter Name으로 활용한다.

B Floor Level에 따른 Pattern 설정

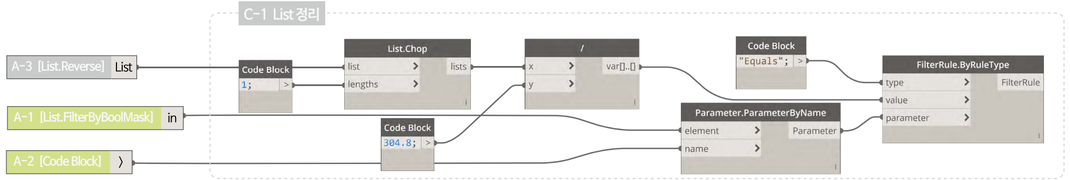


Fill Pattern의 설정

[S1. Slab Level 기준 범례 만들기]에서 작성한 코드와 같은 방식으로 "Slab"+"Level offset" 값 Text를 구성하여 [FilledRegionType.ByName] node를 이용하여 Filled Region Type을 찾는다. [S1. Slab Level 기준 범례 만들기]을 통해 만든 범례를 기준으로 Filter를 작성하도록 하는 과정이다. 범례에는 Project의 모든 Level Offset의 Filled Region이 있다. 현재 View에 있는 Structual Floor의 Level Offset만 Filter로 가지고 온기 위해서 [A-4]에서 조합한 Text 정보로 Filled Region Type을 선택한다. 만들어진 "Slab"+"Level offset"으로 [FilledRegionType.ByName]으로 선택된 Filled Region의 "Fill patten" 정보는 [GetParameterValueByName]을 이용해 추출한다. Fill Pattern은 현재 View에서 설정한 Graphic 옵션에 Input Data로 연결한다.



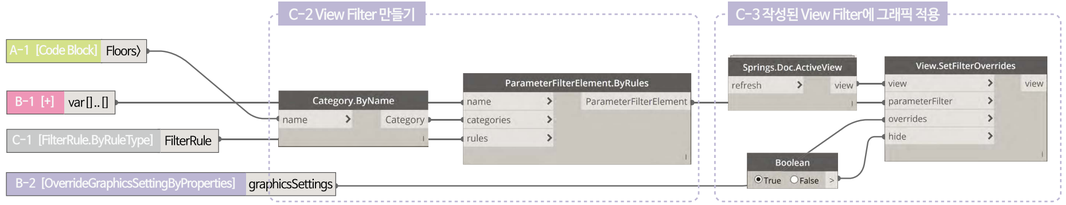
C View Filter 만들기



View Filter Rule 생성

List로 정리된 Structual Floor의 Level Offset 값을 [FilterRule.ByRuleType] node의 Parameter Value에 Input하기 전에 List의 값을 304.8로 나누어 준다. 나누지 않고 Input 할 경우, Revit의 Filters에서 인치로 인식하여 올바른 Filter를 생성할 수 없다.

[Parameter.ParameterByName]의 Parameter Name에 "Height Offset from Level"을 연결하고 Parameter 값과 동일한 부재에 Filter을 적용하기 위해 Rule Type에 "Equals"를 Input 한다.



View Filter 작성

작성된 View Filter Rule Type은 [ParameterFilterElement.ByRules]에 연결하여 Filter를 생성한다.

View Filter 적용

[View.Set.FilterOverRides] node를 이용하여 현재의 View의 Filter에 적용시킨다.

S3. Spot Elevation Tag 배치하기

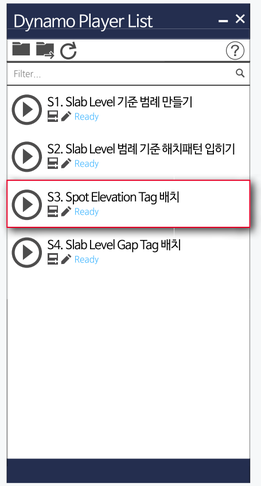
동영상 : https://youtu.be/GNT7mWJxPIc

기능 및 필요성

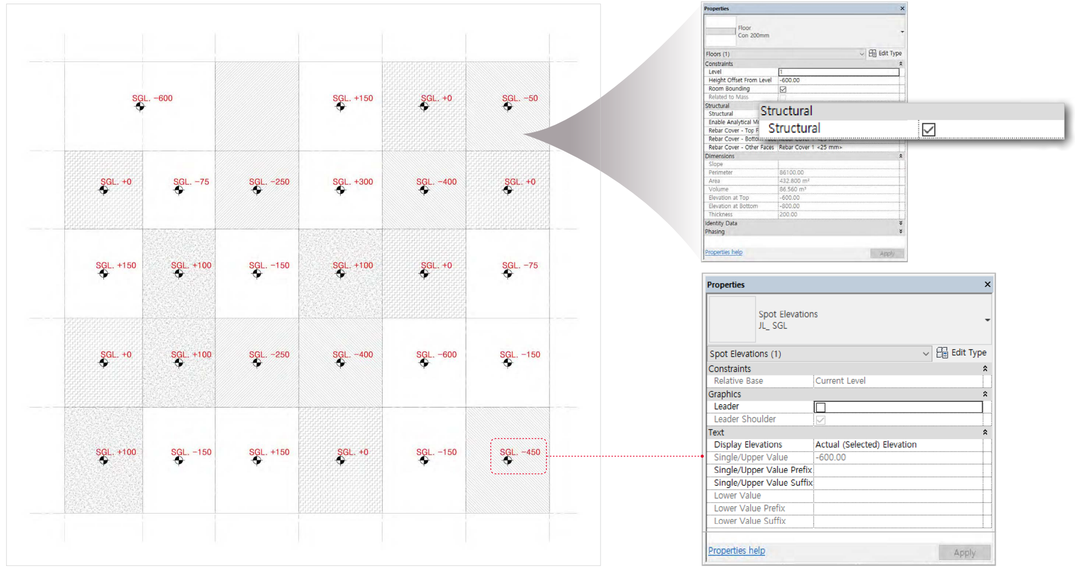
평면도 작업시 바닥, 높이 값에 대한 Tag 배치를 자동으로 한다. 각각의 평면에 Spot Elevation Tag를 바닥 중심점에 배치한다.

유의사항

- 작업 화면 : Slab의 Elevation Tag를 배치하는 평면 View에서 구동한다.
- 실행으로 만들어진 Spot Elevation Tag는 재실행시 중복 배치 될 수 있다.

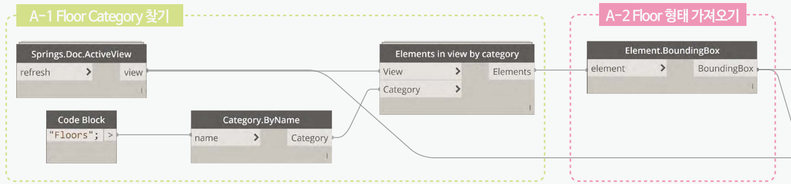


* Dynamo Player는 Revit 2017부터 사용 가능하며, Player의 설정창은 Revit 2018 이상 버전에서 사용할 수 있다.
Revit 2017일 경우 Dynamo 파일을 열고 설정을 수정하여 저장한 뒤 Player를 누르거나 Dynamo에서 바로 실행시킨다.



CODE OVERVIEW

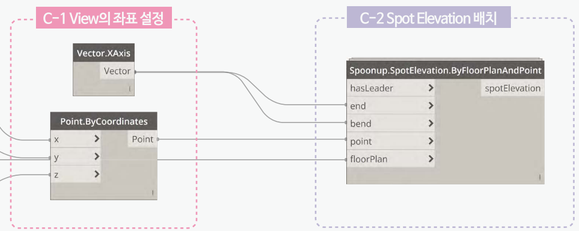
A Floor 형태 가져오기



B 배치된 Floor의 좌표 정보 추출

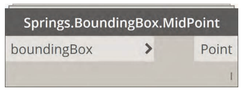


C Spot Elevation 배치



Key Node

custom node **Springs Packages**

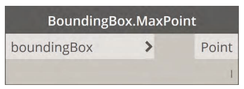


Bounding Box의 중심점 좌표를 추출한다.

INPUTS
- boundingBox
  Bounding Box를 Input한다.

OUTPUTS
- Point
  Bounding Box의 중심점 좌표를 확인한다.

Dynamo basic node

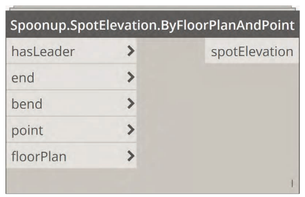


Bounding Box의 가장 높은값을 추출한다.

INPUTS
- boundingBox
  Bounding Box를 Input한다.

OUTPUTS
- Point
  Z축 최대 값을 추출 한다.

custom node **Spoonup Packages**

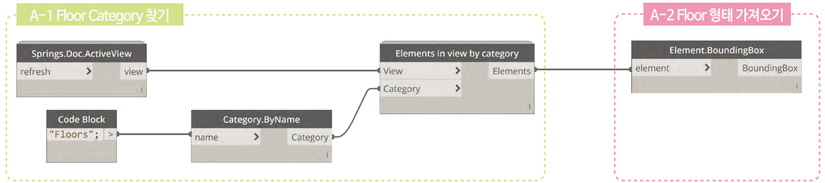


평명도 및 Point에 SpotElevation을 배치한다.

INPUTS
- end, bend
  Vector 값을 불러와 배치 위치를 지원한다.
- point, floorPlan
  좌표값 혹은 평면의 위치를 불러온다.

OUTPUTS
- spotElevation
  좌표값 및 vector값, 평면의 위치 등을 확인하여 SpotElevation을 배치한다.

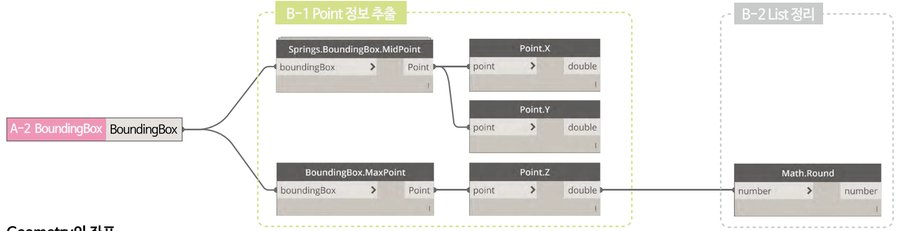
A Floor 형태 가져오기



Floor의 Geometry

[Elements in view by catagory]를 이용하여 현재 View에 보이는 Floor 부재 정보를 가지고 온다. [S1. Slab Level 기준 범례 만들기]와 [S2. Slab Level 범례 기준 해치 패턴 입히기]와 다르게 Strutural Floor가 아닌 현재 View에 보이는 전체 Floor를 가져 온다. Slab의 Spot Elevation Tag 뿐만 아니라 마감 Floor의 Spot Elevation Tag에 활용하기 위해서이다. 현재 View에서 보이는 Floor들을 [Element.BoundingBox] node를 이용하여 Dynamo의 Geometry로 가지고 온다.

B 배치된 Floor의 좌표 정보 추출

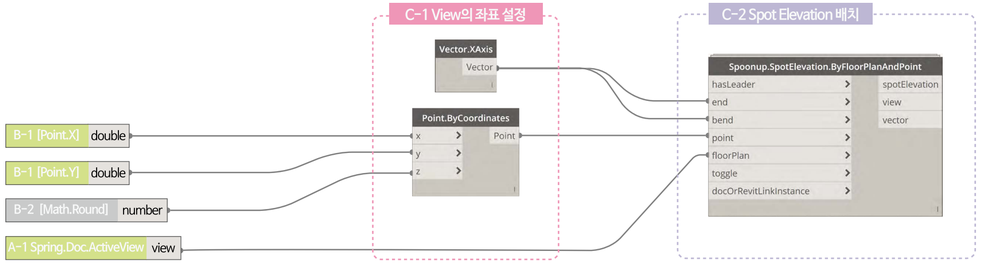


Geometry의 좌표

Floor Geometry는 [Springs.BoundingBoxMidPoint]를 이용하여 Floor의 중심점 (X.Y)를 찾는다.

Spot Elevation Tag가 Floor에 의해 겹쳐 안보이는 것을 막기위해 [BoundingBox.MaxPoint]를 이용하여 가장 높은 Z값을 추출한다. Z값은 [Math.Round]를 이용하여 정수로 정리한다.

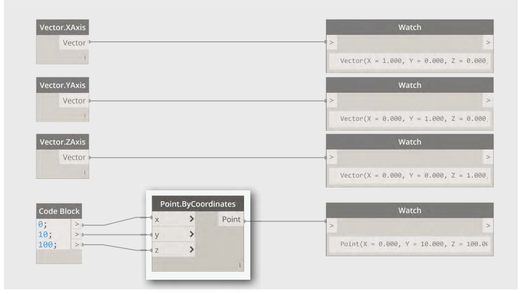
C Spot Elevation 배치



Spot Elevation Tag 배치

[B]에서 추출된 X,Y,Z 값은 [Point.ByCoordinates]를 이용하여 Spot Elevation을 배치할 좌표 List를 만든다. List는 [Spoonup.SpotElevation.ByFloorPlanAndPoint] node의 Point에 Input한다. End와 Bend는 Spot Elevation의 Leader Line이므로 0값을 주기 위해 [Vector.XAxis] node를 연결한다.

Colckwork Package의 [SpotElevation.ByFloorPlanAndPoint] node는 Structure Plan에서 오류가 발생한다. Spoonup에서 만든 node를 이용하는 것을 추천한다.



S4. Slab Level Gap Tag 배치

동영상 : https://youtu.be/oLSB7sNMWJA

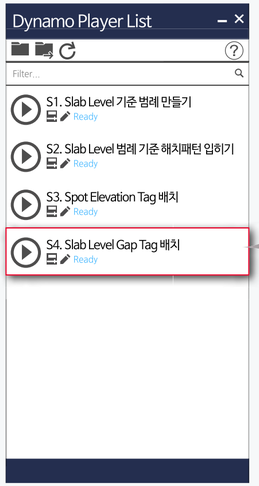
기능 및 필요성

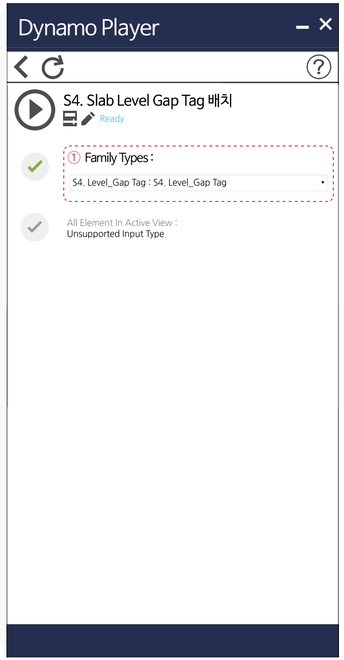
구조 평면도 작업시 Slab레벨 차의 표현을 해주는 Tag의 배치이다. Slab 끝선을 기준으로 Slab 레벨차를 자동으로 인식하여, Tag Family를 배치 해준다.

유의사항

- 작업 화면 : Slab Level Gap을 표현할 평면 View에서 구동시킨다.

- Floor의 Structure Parameter에 반드시 체크가(▣) 되어 있어야 한다.
- 함께 제공한 S4. Level Gap Tag Family가 로드되어 있어야 한다.
- 곡선 Floor 경우 곡면에 배치되지 않는다.

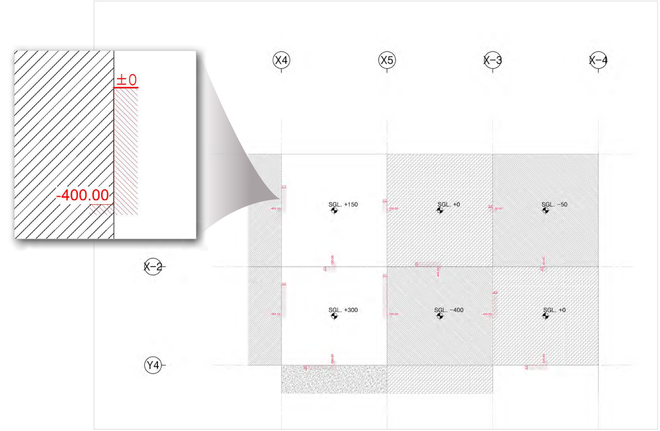




* Dynamo Player는 Revit 2017부터 사용 가능하며, Player의 설정창은 Revit 2018 이상 버전에서 사용할 수 있다.
Revit 2017일 경우 Dynamo 파일을 열고 설정을 수정하여 저장한 뒤 Player를 누르거나 Dynamo에서 바로 실행시킨다.

Level Gap tag 배치를 위한 parameter 조정

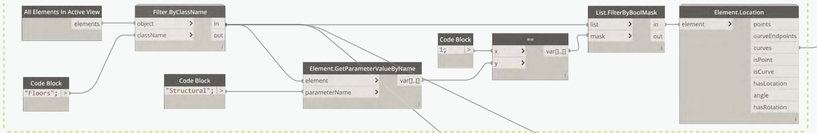
① Family Types
배치할 "Level Gap" 패밀리를 선택합니다



CODE OVERVIEW

A 현재 View의 Foor 객체 형태 가져오기

A-1 현재 View의 Floor 부재의 위치 정보 추출

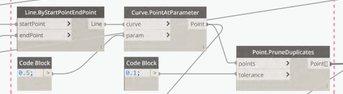


B Floor의 Level offset 값 가져오기

B-1 Curve의 좌표 추출



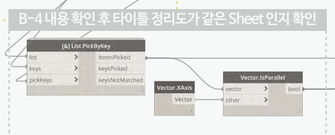
B-2 Curve 중심점의 오차범위 내 Point 찾기



B-3 Vector 값 추출, sheet number 정보 추출



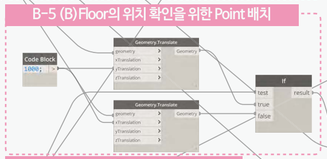
B-4 내용 확인 후 타이틀 정리도가 같은 Sheet 인지 확인



B-5 (A) Floor의 위치 확인을 위한 Point 배치



B-5 (B) Floor의 위치 확인을 위한 Point 배치



B-6 Curve 기준의 Solid 만들기



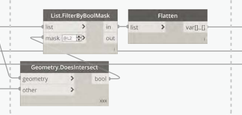
A-2 List 정리



B-7 (A) List 정리



B-7 (B) List 정리



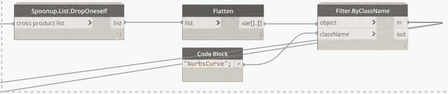
A-3 추출 된 Curve를 XY 평면에 작성 및 Poly Curve로 변형



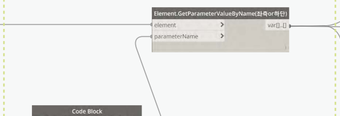
A-4 겹치는 PolyCurve 추출



A-5 List 정리

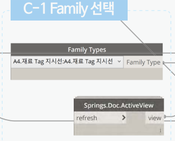


B-8 Floor의 Level offset 값 추출

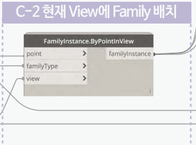


C Tag family 배치 및 parameter 정보 입력

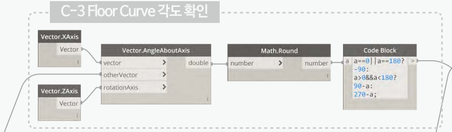
C-1 Family 선택



C-2 현재 View에 Family 배치



C-3 Floor Curve 각도 확인



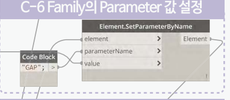
C-4 Family 각도 변경



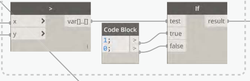
C-5 인접한 Floor Level Offset 차이 값 확인



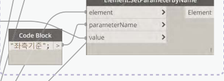
C-6 Family의 Parameter 값 설정



C-8 인접한 Floor의 좌,우,상,하 크기 비교



C-9 Family의 Parameter 값 설정

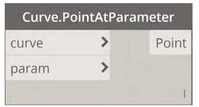


C-7 같은 Level offset 값 Family 삭제



Key Node

Dynamo basic node

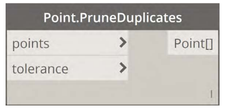


선택한 Curve를 지정한 Parameter값으로 나누어 Point값을 가져 온다.

INPUTS
- curve
  두 점 사이의 커브를 불러온다.
- param
  Parameter값을 가져온다.

OUTPUTS
- Point
  Parameter값으로 추출된 Curve의 Point값을 확인 한다.

Dynamo basic node

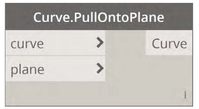


오차 범위내의 중복된 Point를 제거한다.

INPUTS
- points
  Point로 지정하여 중복 요소를 판단할 대상을 불러온다.
- tolerance
  오차 범위 관련 매개 변수를 불러온다.

OUTPUTS
- Point[]
  지정된 요소의 중복 포인트를 제거하여 중복되지 않은 포인트만 추출한다.

Dynamo basic node



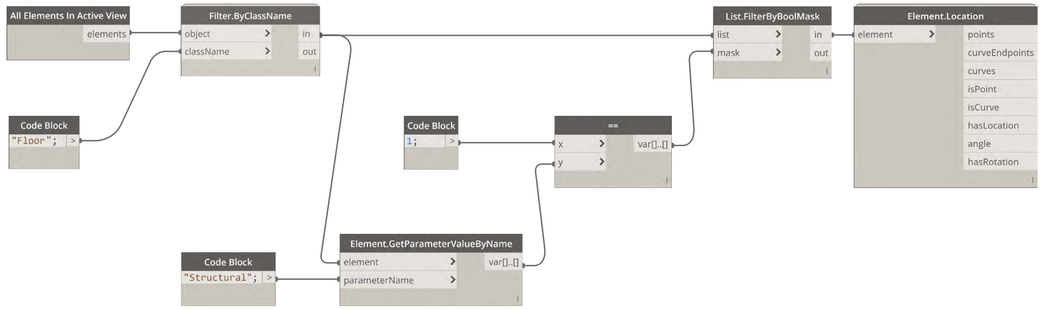
Curve를 평면에 투영시킨다.

INPUTS
- curve
  투영시킬 Curve를 선택한다.
- plane
  Curve를 투영시킬 Plane을 불러온다.

OUTPUTS
- Curve
  선택한 Curve를 지정된 평면에 투영시킨다.

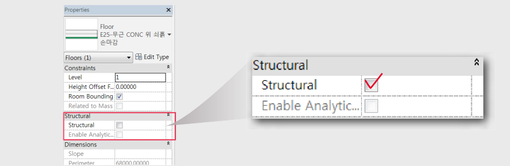
A 현재 View의 Foor 객체 형태 가져오기

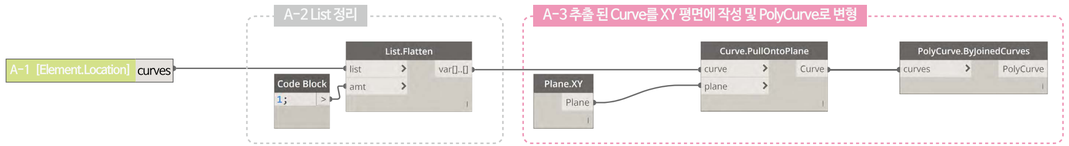
A-1 현재 View의 Floor 부재의 위치 정보 추출



Structral Floor의 위치

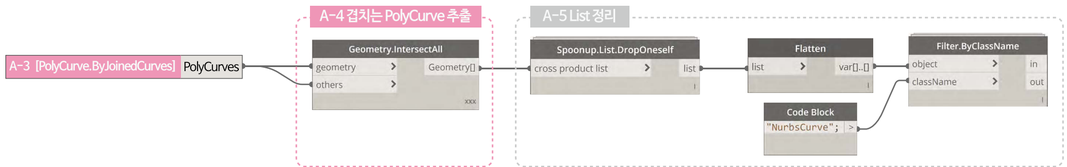
[All Elements In Active View]를 이용해 현재 View의 모든 부재 List를 가져온다. [Filter.ByClassName]을 통해 Floor 부재만 추출 한다. Category 선택을 통해서도 할 수 있으나, 다양한 Code를 사용하기 위해서 부재 검색 형태로 Floor 부재를 추출하였다. [List.FilterByBoolMask]의 Mask에 "Structural" Parameter의 체크여부를 판별할 수 있는 node를 Input하여 Structural Floor만 추출되게 한다. 추출된 Structural Floor는 [Element.Location]에 연결하여 바닥선을 Curve로 추출한다.





Curve의 투영

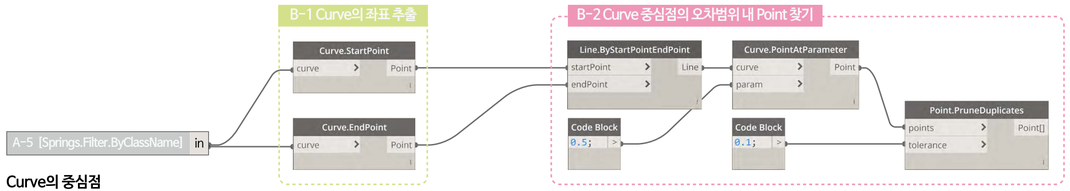
[List.Flatten]으로 List 정리 후 추출된 Curve는 [Curve.PullOutPlane] node를 이용해 [Plane.XY]에 Curve를 만든다. [Curve.PullOntoPlane] node는 선택한 Plane에 Input하는 Curve를 투영시킨다. 작성된 Curve는 [PolyCurve.ByjoinedCurves]를 이용해 단일 Curve로 만든다. 작성된 Curve는 Revit의 부재에 의해서 Dynamo에서 작성된 것이다. Revit에서 Curve로 표현되지 않는다.



간섭된 Numbs Curve

[PolyCurve.ByJoinedCurves]로 만들어진 Polycurve는 [Geometry.IntersectAll]에서 서로 Cross Product 체크를 진행하여, 서로 간섭되는 부분을 찾는다. 바닥을 구성하는 Poly Curve간의 간섭을 체크하여 서로 맞닿고 있는 바닥을 선별하기 위한 방법이다. [Spoonup.List.DropOnesslf]는 [Geometry.IntersectAll] 진행후 자기 자신과의 간섭 비교를 제외시키는 node이다. 간섭이 일어나는 Nurbs Curve는 [Filter.ByClassName] node로 선택한다.

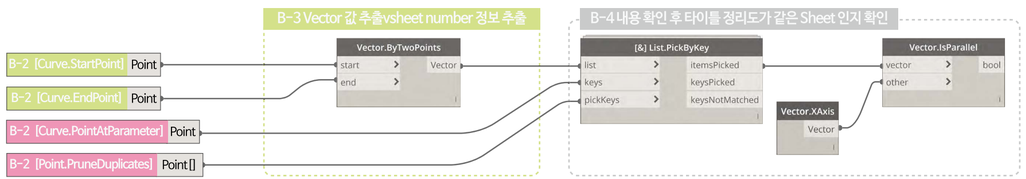
B Floor의 Level Offset 값 가져오기



Curve의 중심점

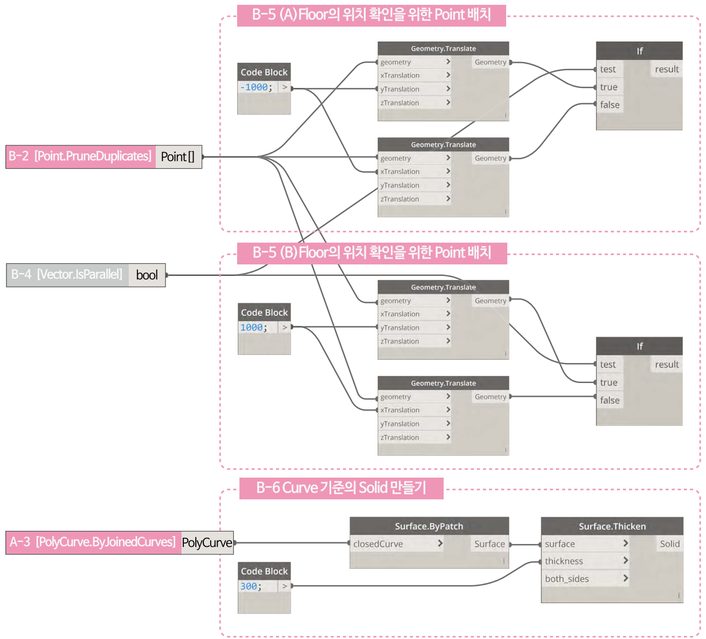
간섭 체크로 선택된 Nurbs Curve를 [Curve.StartPoint]와 [Curve.EndPoint]를 통하여 좌표가 추출된다. 추출된 좌표는 [Line.ByStartPointEndPoint]를 이용해 새로운 Line으로 정리한다. [Curve.PointAtParameter]를 이용해, 새로 작성된 Line의 중심점을 찾는다. 간섭 체크로 선택된 Nurbs Curve를 다시 Line으로 변경하는 이유는 각각의 바닥이 작성될 때, Curve의 좌표는 동일하나 Start Pointd와 End Point가 다르게 인식되어 다른 부재로 인식하기 때문이다. 같은 좌표에 있는 Curve를 선별하여 연산할 Curve의 개수를 줄인다.

[Curve.PointAtParameter]로 추출한 Curve의 중심점은 [Point.PruneDuplicates]를 이용하여 오차범위를 지정하여 Point 개수를 줄인다. [Point.PruneDuplicates]의 tolerance에 0.1값을 Input하여 Point들을 정리한다.



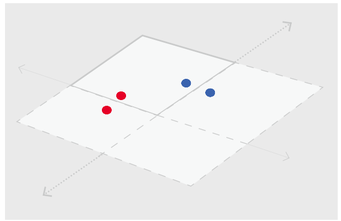
Curve의 방향

[B-1]에서 추출된 [Curve.StartPoint]와 [Curve.EndPoint]를 [Vector.ByTwoPoint]에 Input시켜 두점이 갖는 Vector 방향의 거리값을 추출한다. 추출된 Vector 거리 값은 [List.PickByKey]의 List에 Input한다. [B-2]에서 추출된 Curve의 중심점은 [List.PickByKey] 의 Keys에 Input하고, 오차범위로 정한 Curve 중심점은 PickKeys에 Input한다. [List.PickByKey] node는 Input된 Keys 값들을 다시 Pick Keys 값들로 정리한다. 정리된 Keys들을 이용해 List를 선별하여 준다. [List.PickByKey]를 이용해 간섭이 되는 Curve의 중심점을 포함하는 Curve를 선택한다. 정리된 Curve를 [Vector Is Parallel]을 이용하여 XAxis와 평행한지 판별하여 Curve가 갖고 있는 X,Y축을 찾는다.



인접한 바닥 기준

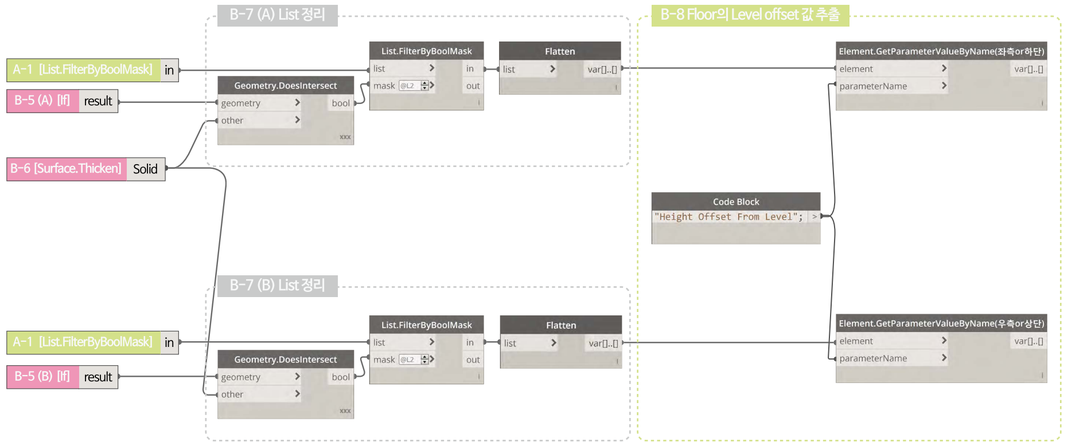
[Vector.IsParallel]에 의해 Ture, False로 나뉘어진 값을 이용하여 X축과 평행 할 경우, 간섭된 Curve의 중심점을 Y축을 기준으로 -1000, 1000으로 이동시키고 X축과 평행하지 않을 경우 X축을 기준으로 -1000, 1000간격을 이동시켜 간섭된 바닥의 상하 좌우의 바닥 종류를 파악하는 기준을 만든다.



Solid Geometry

[A-3]의 [PolyCurve.ByJoinedCruves]에서 추출된 바닥 Curve는 [Surface.ByPatch] node를 이용하여 Surface로 만든다. [Surface.Thicken] node를 이용하여 두께 값 300인 Solid를 만든다. 기존의 Revit 부재 형태를 바로 Dynamo로 불러올 수 있지만 모델링된 바닥의 높낮이 차이가 심할 경우, 간섭체크시 놓칠 수 있다.

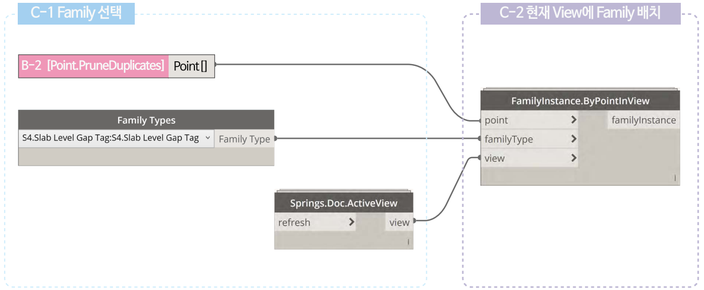
일괄적으로 X,Y,Z 값 0인 Curve로 다시 작성 후, 두께 값을 300으로 동일한 높이의 Solid 부재를 만든다.



인접한 바닥의 Level Offset

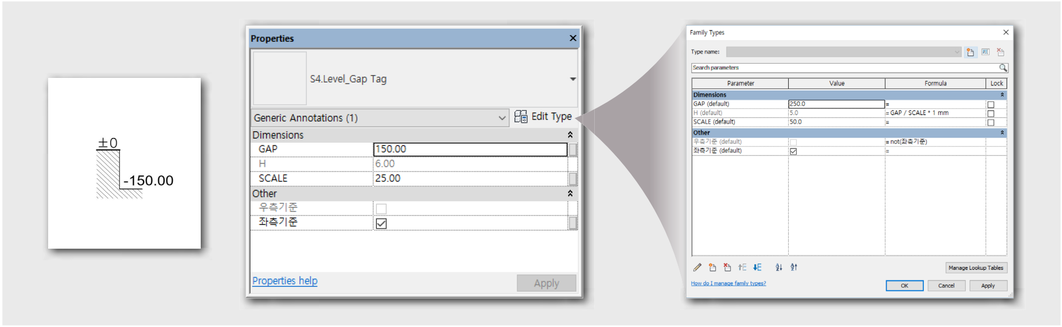
[B-5]의 [IF] node로 간섭되는 Curve의 중심점을 기준으로 상하좌우에 배치된 Point들과 Solid로 만들어진 Floor 부재를 [Geometry.DoesIntersect]를 이용하여, 간섭 여부를 파악한다. 간섭여부의 Ture, False 값은 [List.FilterByBoolMask]를 이용하여 간섭되는 인접한 바닥을 선택한다. 인접한 바닥의 Level Offset 값을 추출하기 위해서 [GetParameterValueByName]을 이용한다.

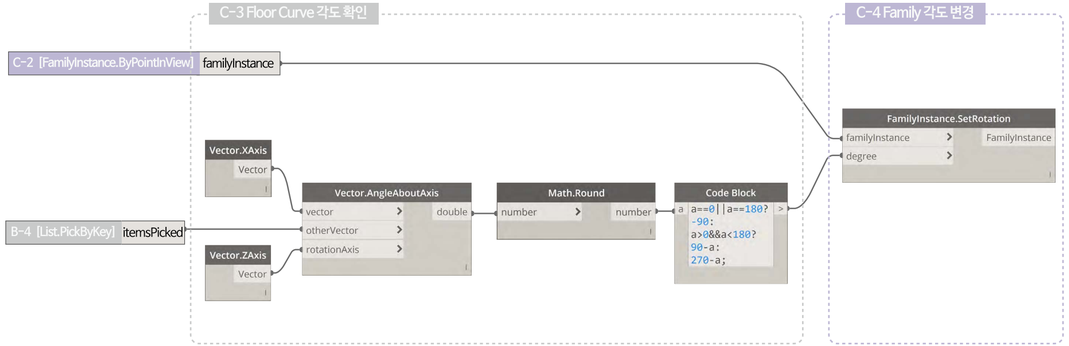
C Tag Family 배치 및 Parameter 정보 입력



Tag Family 배치

[FamilyInstance.ByPointInView]를 이용하여 현재 View에 제공된 "S4. Level_Gap Tag" Family를 선택하여 연결한다. "S4. Level_Gap Tag"는 Generic Annotation Family이다. Family를 배치할 Point 좌표는 [B-2]의 [Point.PruneDuplicate]에서 만들어진 Curve의 중심점을 Input하여 인접한 바닥과 맞닿는 Curve의 중심점에 배치되게 한다.

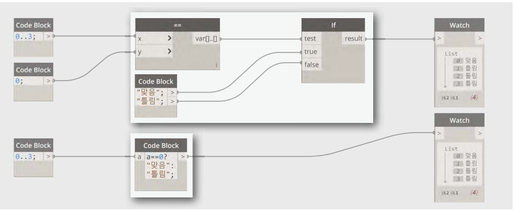


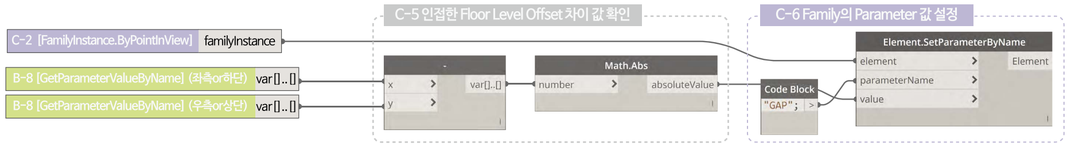


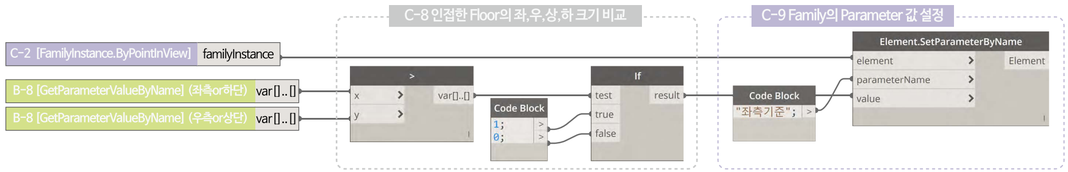
Tag Family의 각도 설정

인접한 바닥 부재와 맞닿는 Curve의 중심점에 배치된 "S4. Level_Gap Tag" Family는 바닥 Curve와 같은 각도로 회전해주어야 한다. [Vector.AngleAboutAxis] node를 이용하여 바닥 Curve들의 각도를 추출한다. [Math.Round]를 이용하여 정수로 각도 값을 정리한다. 추출된 Curve의 각도 값은 0 ~360도까지 존재한다.

배치된 "S4. Level_Gap Tag" Family는 Text를 읽는 방향을 고려하여 0 ~ 360도의 범위를 [Code Block]을 이용하여 정리한다. [Code Block]의 내용은 각도 값이 0 또는 180도 일 경우 -90도, 0<a<180인 경우 90-a, 180<a일 경우 270-a를 이용하여 각도 값의 범위를 -90 ~ 90으로 정리한다. 정리된 Curve의 각도 값은 [FamilyInstance.SetRotation]에 Input 하여 배치된 Family를 Curve에 맞게 회전시킨다.



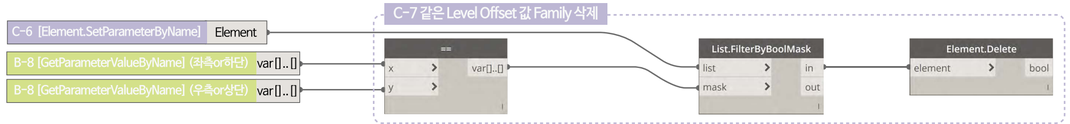




Tag Family의 Parameter 정보

[B]에서 추출한 Floor의 Level Offset 값들은 [-] node를 이용하여 Floor간의 Level Offset 차이 값을 연산한다. 연산된 값은 [Math.Abs] node로 인접한 바닥 Level 차이의 정수 값으로 정리한다. 정수로 정리된 [Set.ParameterByName]을 이용해 "S4. Level_Gap Tag" Family의 "Gap" Parameter 값에 기입한다. "S4. Level_Gap Tag" Family는 좌우 상단 중 높은 값을 가지는 Floor를 기준으로 Level Gap이 표현된다. 기준이 되는 Floor는 "S4. Level_Gap Tag"에서 ±0으로 표현되고 인접한 Floor의 Level Offset과의 차이 값만 표현된다.

인접한 Floor의 Level Offset의 크기를 비교하여 Family의 Level Text가 표현되게 Parameter를 구성한다. [Set.ParameterByName]의 "좌측기준"에 인접한 Floor의 Level Offset 크기를 비교한 값을 넣으면 Family의 Visiblity On/Off방식으로 Family Text가 표현된다.

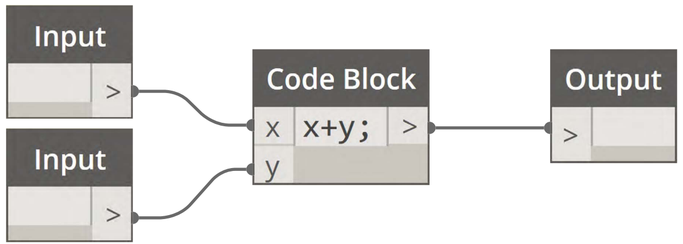


Family Tag 삭제

만일 인접한 Floor의 Level Offset 값이 동일한 경우 배치된 "S4. Level_Gap Tag" Family를 [List.FilterByBoolMash] node로 판별한 뒤 [Element.Delete]로 삭제한다.

3. Custom Node

3.01 Custom Node 소개



Custom Node

Dynamo의 활용을 높이기 위해서 Code Block이 제공된다. Code Block을 통해서 사용되는 Node의 개수도 줄일 수 있고, 사용자 Node도 개발할 수 있다. 또한 작성된 Code Block은 Package로 만들어 다른 프로젝트에서 재사용 할 수도 있다..

Cleaning up

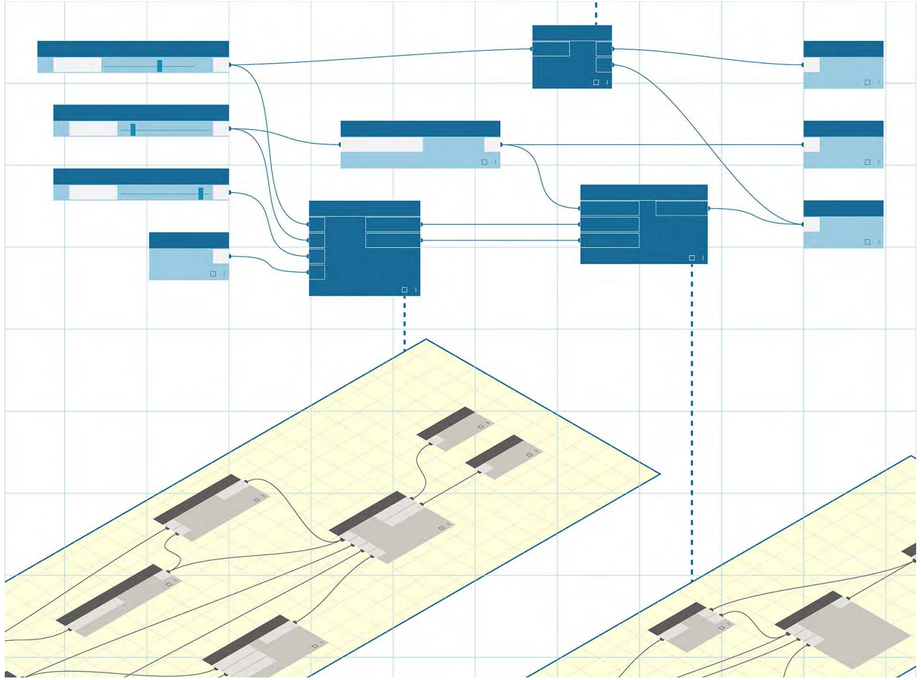
"Dynamo Custom Node"를 통해 기본 Node와 Code Block Node를 조합하여, 하나의 Node로 만들 수 있다. 사용자는 Custom Node를 작성하여, 본인의 목적에 맞는 Node를 조합하여 사용하고, 공유할 수 있다.

Adapting to Change

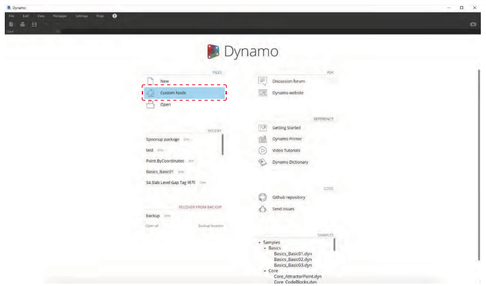
작성된 사용자 Node는. dyf로 저장된다. 사용자 Node가 수정되었을 경우, 편집 후 Package 업데이트를 통해 사용된 모든 Node에 업데이트할 수 있다.
디자인에서 발생할 수 있는 변경 사항을 적용하여 원활하게 Node를 업데이트할 수 있다.

Work Sharing

사용자 Node의 가장 좋은 기능은 작업 공유 기능이다. Dynamo에 익숙한 사용자가 복잡한 Dynamo Code를 만드는 경우 처음 사용하는 사용자에게 쉽게 전달할 수 있다.
사용자 Node는 복잡하게 구성되지만, 실제 사용하는 Node는 단일 Node이기 때문에 Dynamo에 익숙하지 않은 사용자도 쉽게 Node를 사용할 수 있다.



3.02 Custom Node 만들기



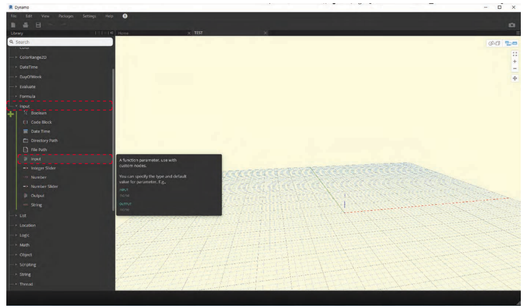
Custom Node Environment

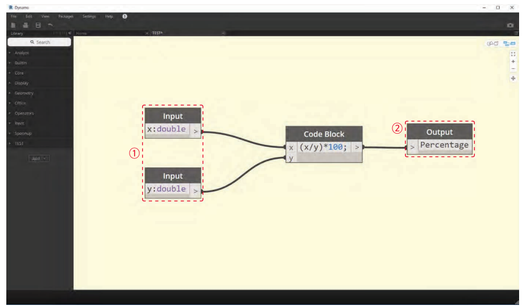
사용자 Node에서 비율 계산을 하는 간단한 Node를 작성하여 본다.
Dynamo 작업 화면과 다른 모습이지만, Node를 만들기 위한 프로세스는 같다.

- 사용자 Node를 만들기 위해서는 Dynamo를 실행하고 Files에서 Custom Node 아이콘을 클릭하거나, 단축키 Ctrl + Shift + N을 입력한다.

팝업 창에 나온 사용자 Node 특성 창에Node 이름, 설명 및 Category를 입력한다.

1. Name: Test
2. Description: 비율 계산 Node
3. Category: Test



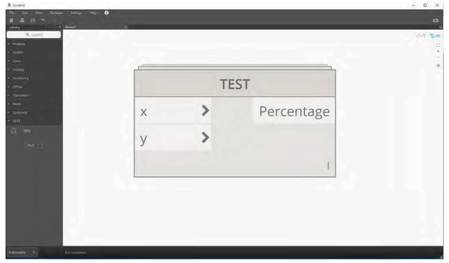


Custom Node Environment

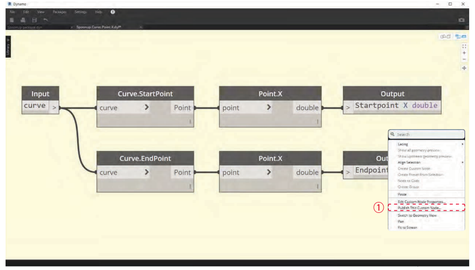
노란색 배경의 화면이 열리고, 사용자 Node로 작성할 Node를 배치한다. 사용자 Node의 Input, Output에 사용될 Port를 만들기 위해, [Input] Node와 [Output] Node를 배치하여 연결한다.

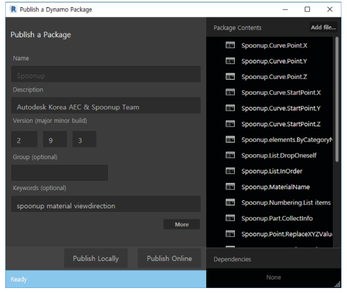
1. Input: [Input] Node를 이용하여 사용자 Node에 입력 Port를 만든다. Node에 "x: double" 이라고 기입하여, 숫자 형식만 인식 되도록 설정한다.
2. Output: [Output] Node를 이용하여 사용자 Node에 출력 Port를 만든다.

아래 이미지는 사용자 Node를 Dynamo 작업 공간에서 불러온 모습이다. 사용자 Node는 Library에서 Test 목록에 Text라는 이름으로 확인할 수 있다.



3.03 Custom Node 공유





Publishing a Custom Node Locally

Spoonup Package를 설치하고, Spoonup Node 중 [Spoonup.Curve.Point X] Node를 작업 화면에 배치하고 Node를 더블 클릭하여, 사용자 Node의 구성을 확인한다. [Spoonup.Curve.Point X] Node를 열고, Node의 구성을 확인 한 뒤 원하는 Node 또는 Code block 추가하여 저장할 수 있다.

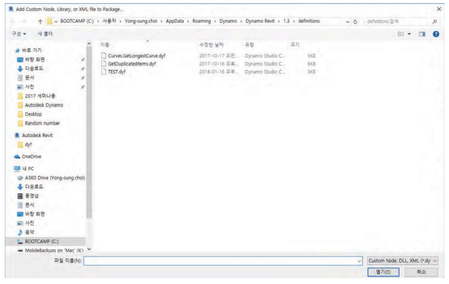
1. 사용자 Node가 수정되어 Package를 업데이트해야 할 경우, 마우스의 오른쪽을 클릭하여 Publish Custom Node를 선택한다.

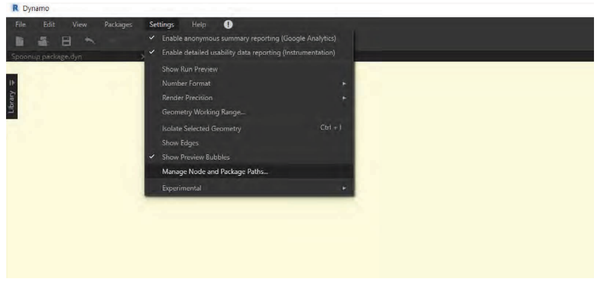
2. 공유할 Node를 Package로 만들기 위해서 Package의 기본 정보를 기입한다.

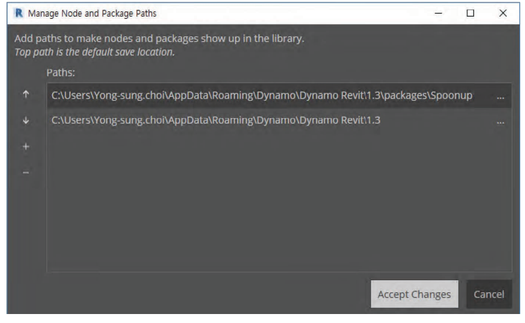
3. 사용자 Node를 공유하는 방법은 두 가지가 있다. 온라인 서버에 올려 다른 Dynamo 사용자들이 다운로드하게 하는 방법이 있고, 본인 컴퓨터에 저장하여 보안을 유지하는 방법이 있다. 사용자 Node의 특성을 고려하여 판단한다.

4. Package에 파일을 추가할 경우 Add 버튼을 눌러 추가할 사용자 Node를 선택한다.

5. Settings 〉 Manage Node and Package Paths에서 사용자 Node 또는 Package 저장 폴더를 확인할 수 있다..







Publishing a Custom Node Locally

오른쪽 이미지에서 두 가지 경로를 확인할 수 있다.
AppData₩Roaming₩Dynamo Revit₩1.3₩Package는 온라인에서 다운로드 한 Package 경로이고, AppData₩Roaming₩Dynamo Revit₩1.3 우는 사용자 Node가 로컬로 공유할 경우 자동으로 저장될 폴더 경로이다.
사용자의 컴퓨터 환경에 맞게 폴더 경로는 수정 및 편집할 수 있다.

AUTODESK

BIM GUIDEBOOK 2_Dynamo Part Ⅰ

기 획 | Autodesk 최기영
Autodesk Andrew Cunningham
Autodesk 임민수

저 자 | Autodesk 최용성
DaouData 조원호
가천대학교 최민정

Autodesk 이인근
WeMade I&C 이병수
MIM 강민구
EGA Group 백민호

인 쇄 | 2018년 1월 25일
발 행 | 2018년 1월 25일
발행처 | AUTODESK
주 소 | 서울 특별시 강남구 영동대로 517
아셈타워 17층 06164
전 화 | 02.3484.3400
팩 스 | 02.3484.3405

도움주신분들
Autodesk 김해진, 박준석, 정진형
DaouData 이 용

본 책자의 저작권은 Autodesk 에 있으며, Autodesk BIM GUIDEBOOK 관계자 외에는 임의 수정 및 배포를 금지 한다.