Robot Structural Analysis and Dynamo and ASCE7-10 Wind Loading, Oh MY!

Brandon Schafer

Engineering Intern, ZFI Engineering

Twitter: @BSchaf07

Class summary

By the end of this lab you will be able to apply ASCE 7-10 based wind loads onto a structure in Robot Structural Analysis through Dynamo.

Key learning objectives

At the end of this class, you will be able to:

- Develop a Dynamo graph to aid in wind design
- Interact between Dynamo and RSA
- Script your own custom Dynamo nodes
- Better understand the possibilities of a collaboration between Dynamo, RSA, and Revit

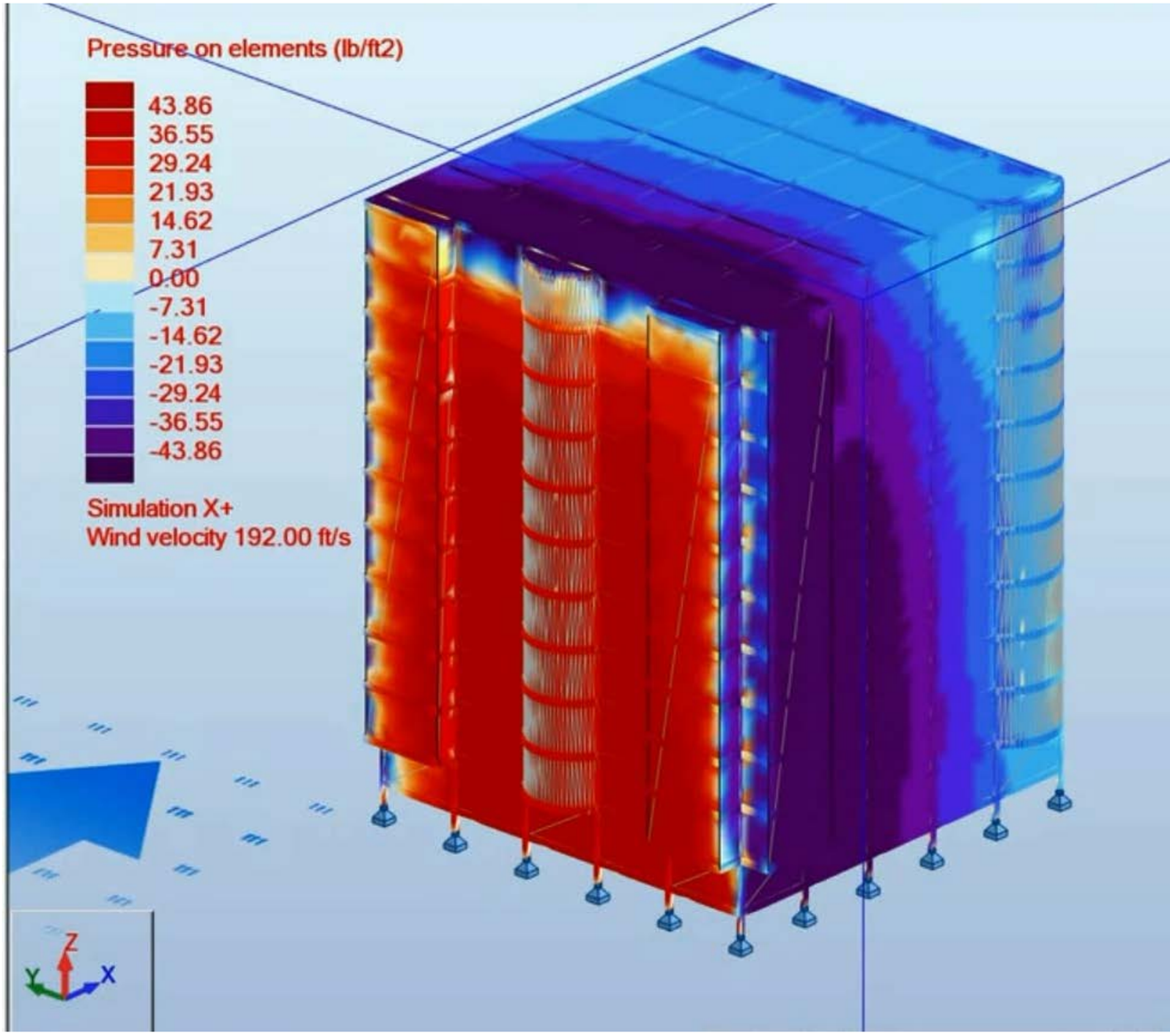
Why is this Necessary

Why did I start down this path?

- My company was looking to change analysis software.
- We had one license of RSA and I took the initiative to test it out
- One of the key features they were looking for was a codified 3d wind loading

RSA can perform a complete CFD wind analysis

- Not codified
- Most people don’t trust computers
- Stay away from the “blackbox”
- Junk in = Junk out



A Golden Opportunity

- Instead of rolling over I saw an opportunity.
  - Convince them of Robot's potential
  - Introduce them to the power of Dynamo

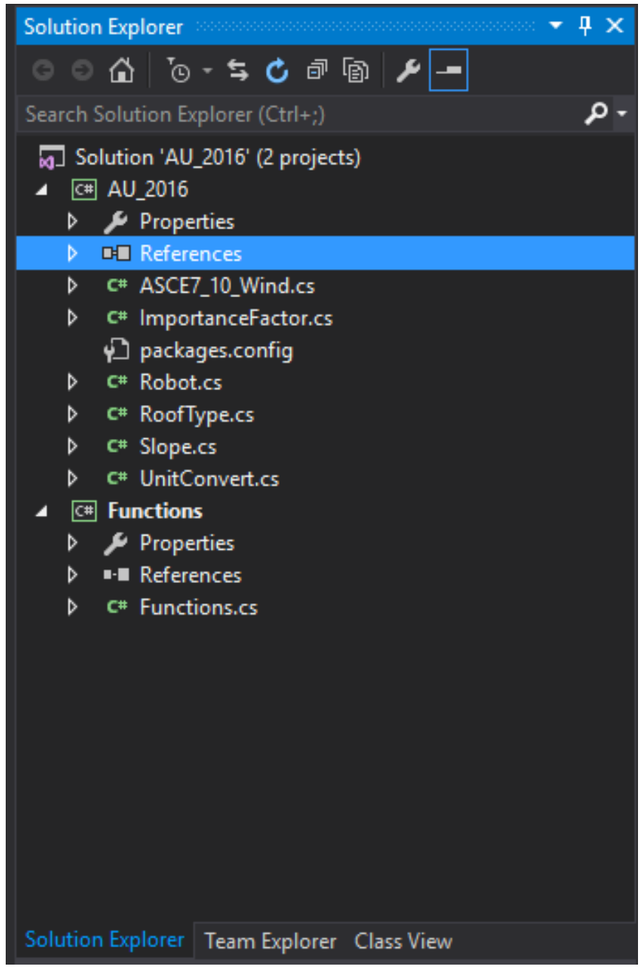
Custom Dynamo Nodes

Custom Node Options

- Python nodes
  - No experience with python
- Out of the box “custom” nodes
  - Not clean enough for me
- Zero touch nodes
  - Potential, but not enough customization to the UI
- Custom C# nodes - Winner

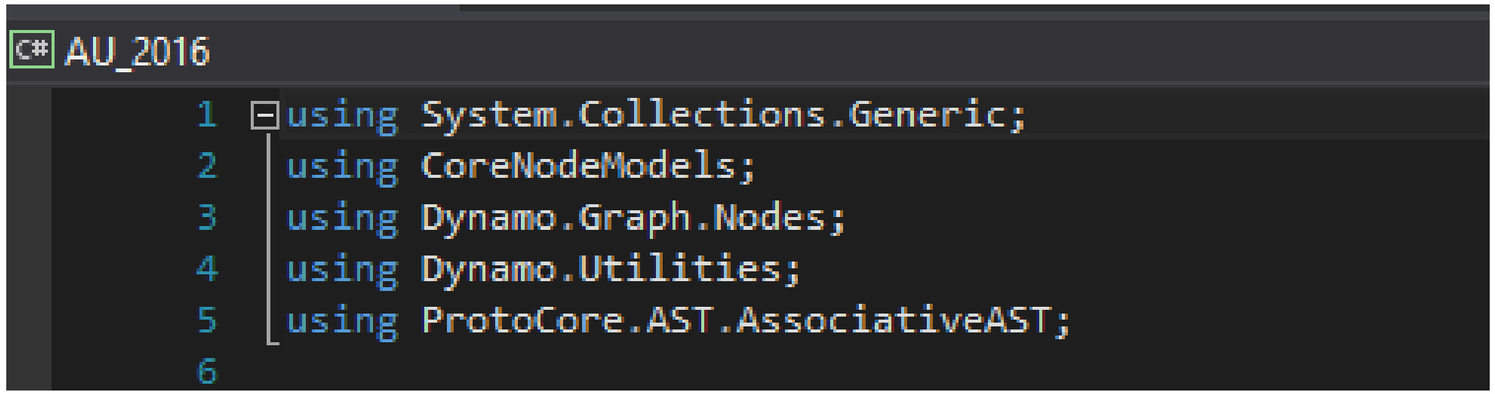
Example of C# Node Development

- Step 1 Add References
  - Dynamo dll's are essential for Dynamo customization



Example of C# Node Development

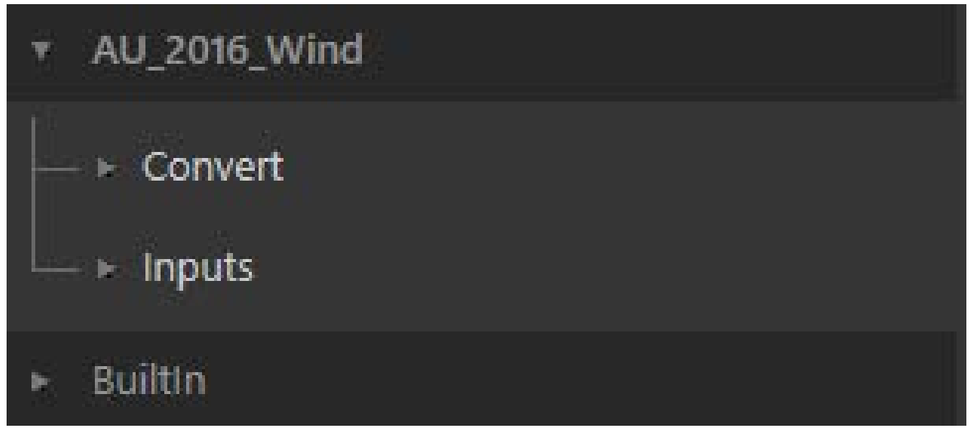
- Step 2: Tell the Code You're Using References
  - Also builds Visual Studio's Intellisense capabilities



Example of C# Node Development

- Step 3: Pick Your Namespace
  - First part is the main Dynamo library category
  - After the period is a subcategory

```
7
8  namespace AU_2016_Wind.Inputs
9  {
```



Example of C# Node Development

- Step 4: Name Your Node

```
[NodeName("Exposure Category")]
[NodeDescription("ASCE 7-10 Wind Exposure Category Selection")]
[IsDesignScriptCompatible]
public class Asce710Wind : DSDropDownBase
{
    public Asce710Wind() : base("Exp") { }

    protected override SelectionState PopulateItemsCore(string currentSelection)
    {
```

- Step 5: Inherit Your Class

Example of C# Node Development

- Step 6: Populate Your List

```
        Items.Clear();

        // Create a number of DynamoDropDownItem objects
        // to store the items that we want to appear in our list.

        var newItems = new List<DynamoDropDownItem>()
        {
            new DynamoDropDownItem("B", 0),
            new DynamoDropDownItem("C", 1),
            new DynamoDropDownItem("D",2)
        };

        Items.AddRange(newItems);

        // Set the selected index to something other
        // than -1, the default, so that your list
        // has a pre-selection.

        return SelectionState.Done;
    }
```

Example of C# Node Development

- Step 7: Build the AST Output

```
public override IEnumerable<AssociativeNode> BuildOutputAst(List<AssociativeNode> inputAstNodes)
{
    // Build an AST node for the type of object contained in your Items collection.

    var intNode = AstFactory.BuildStringNode((string)Items[SelectedIndex].Name);
    var assign = AstFactory.BuildAssignment(GetAstIdentifierForOutputIndex(0), intNode);

    return new List<AssociativeNode> { assign };
}
```

Let’s Jump In!

How did I do?

- Your class feedback is critical. Fill out a **class survey** now.
- Use the AU mobile app or fill out a class survey online.
- Give feedback after each session.
- AU speakers will get feedback in real-time.
- **Your feedback results in better classes and a better AU experience.**