MFG500045

Joints in Fusion 360

A Guide to Assembly Modeling, Position and Motion

Jeff Strater – Sr. Software Architect
Phil Eichmiller – Principal QA Engineer

Learning Objectives

- Understand the Fusion 360 component/assembly model - Bodies, Components, Instances, etc.
- Learn about Ground, flexible and rigid sub-assemblies in Fusion, Rigid Joints, and Rigid Groups
- Master Joints and Joint Origins to position components and describe their motion
- Explore common workflow situations and how to solve them in Fusion

Description

Fusion is not like other CAD tools when it comes to positioning components in an assembly. Fusion has a different paradigm for putting components together. It uses Joints to both position components and describe their motion. This class will help you to make sense of Joints in Fusion 360.

Starting with the foundational concepts of assembly, degrees of freedom, positioning and motion, we will progress into the deeper topics of flexible sub-assemblies, the behavior of grounded components, etc. We will describe the various joint types that can be used in your model, and scenarios where they are applicable. We will attempt to de-mystify assembly modeling in Fusion 360.

Sit down with two Fusion modeling experts and members of the Fusion development team. The topics and workflows we will discuss come directly from Fusion customers just like you.

Speakers

Jeff Strater is a Senior Software Architect in the Fusion 360 team. I've been with Fusion since the very beginning. My focus is on general modeling/sketching. Before that, I was a developer and architect on Inventor, also before R1. So, I'm a long-time CAD guy. When not working with Fusion or its customers, I like to run, cycle, hike, and read science fiction.

Phil Eichmiller is a Principal Software Quality Assurance Engineer for Autodesk, on the Fusion 360 team. In addition to testing and designing with Fusion 360, he enjoys working with the Fusion online community. He enjoys sharing his knowledge by teaching Fusion 360 for the CAD program at Portland Community College. Roller Derby is his favorite pastime, especially watching his daughters, who are both derby stars in Portland, Oregon.

Fusion Assembly Modeling 101

In this section we will review a few fundamentals on the topic of assembly modeling in Fusion

What is an Assembly?

An Assembly in Fusion maps very nicely to the concept of an assembly in the real world:

> assembly
>
> as·sem·bly | \ ə-ˈsem-blē \
>
> *plural* **assemblies**
>
> **Definition of *assembly***
>
> **6a:** the fitting together of manufactured parts into a complete machine, structure, or unit of a machine - the *assembly* of an automobile
>
> **b:** a collection of parts so assembled

An assembly in Fusion is a collection of Components that are put together (assembled) into a single, larger unit.

Bodies vs Components

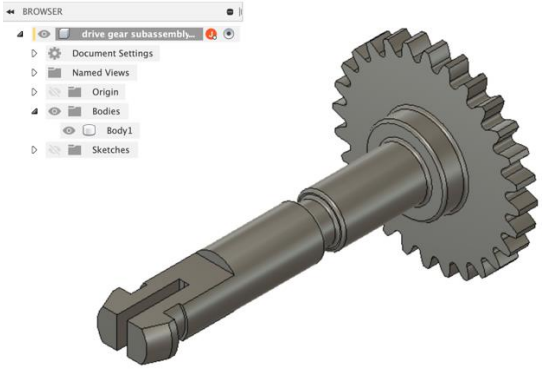
This is an important basic concept to understand before you do any assembly modeling in Fusion. It is a common mistake for Fusion beginner to not understand the fundamental difference between these important two concepts. So, before you go any further, make sure that you understand these concepts

Bodies

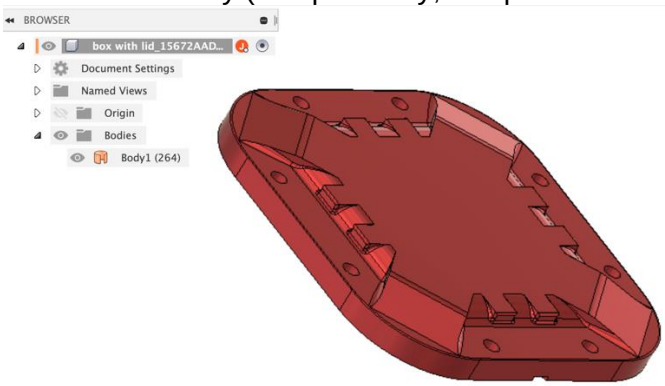
Bodies in Fusion Design are pretty simple objects: They are just a collection of geometry. Today, there are 3 different body types:

BRep Body

A BRep body is a container of precise, boundary-representation geometry. It can be a solid body (fully closed, with mass):

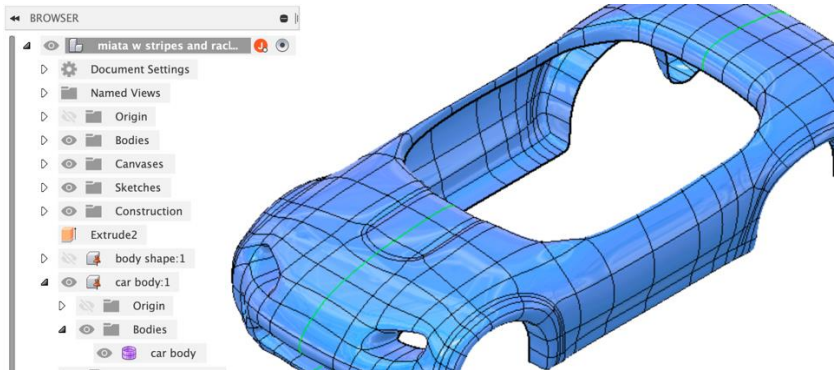


Or a surface body (an open body, composed of infinitely thin surfaces):



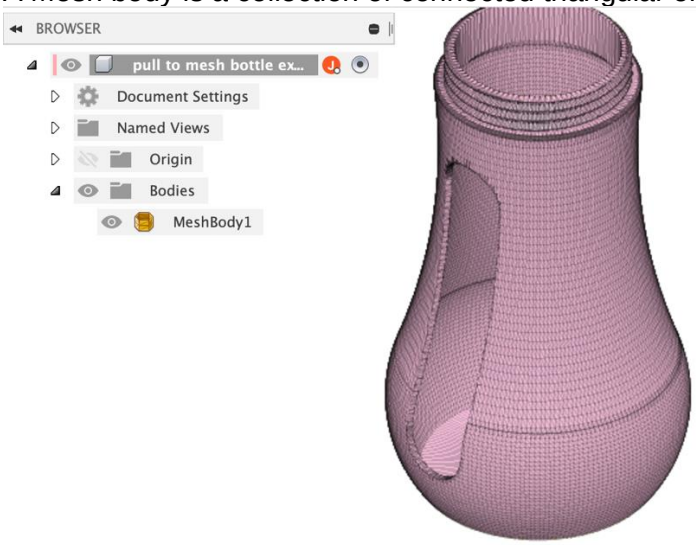
T-Spline (Form) body

A T-Spline body is a subdivision surface composed of a set of faces, edges, and vertices:



Mesh body

A Mesh body is a collection of connected triangular or rectangular facets:

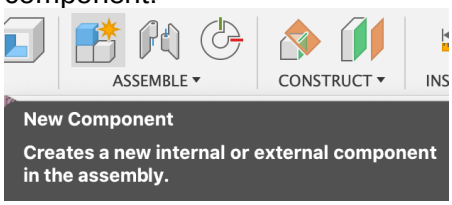


Components

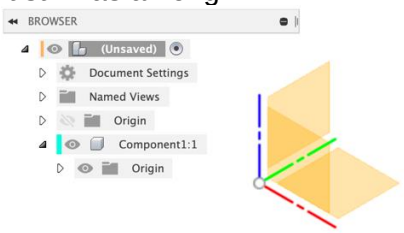
Components in Fusion are also collections, but they are higher level collections that include bodies, and other Fusion objects.

These include:

- An origin. This is a critical item in a component. If you just create a new, empty component:

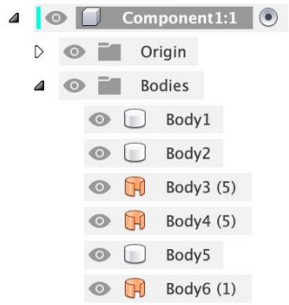


it still has an origin:

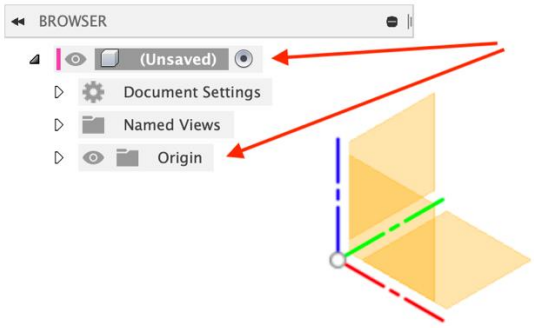


The component origin defines a local coordinate system for that component. As a component is moved, you can observe that the origin also moves. This can be critical to understanding how positioning of components works in Fusion, whether by Joints, Move Component, or Drag.

- Zero or more Bodies. Any of the 3 body types can be owned by a Component, and a component can have any number of bodies. These are shown in the Bodies folder, under a Component:

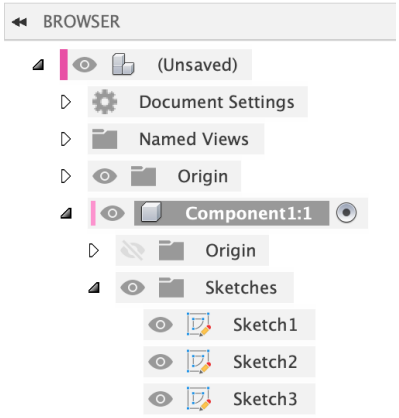


- Child Components. Any Fusion Component can contain child components. In fact, the root of every design is, itself, a component. It has a Component icon, and an Origin:

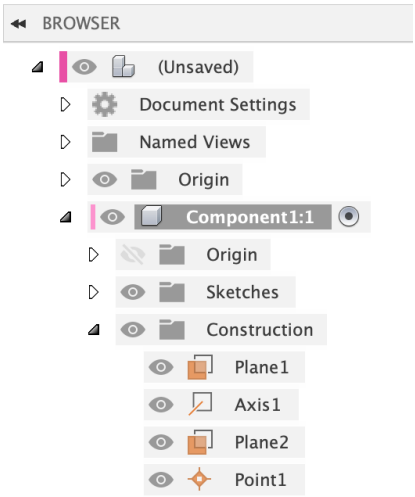


If a Component contains child Components, then it becomes “assembly-like”. If it has only Bodies, then it is “part-like”.

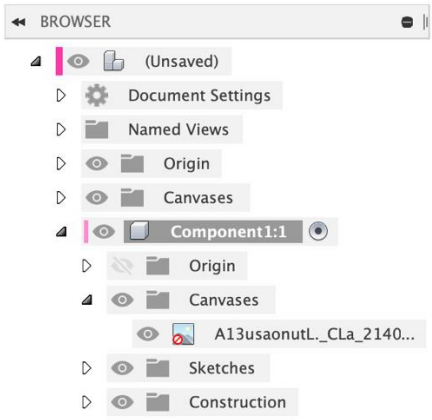
- Sketches. A Component can contain sketches. If a Component is “active” (see below) when a sketch is created, it will contain that new sketch.



- Construction (Work) Geometry



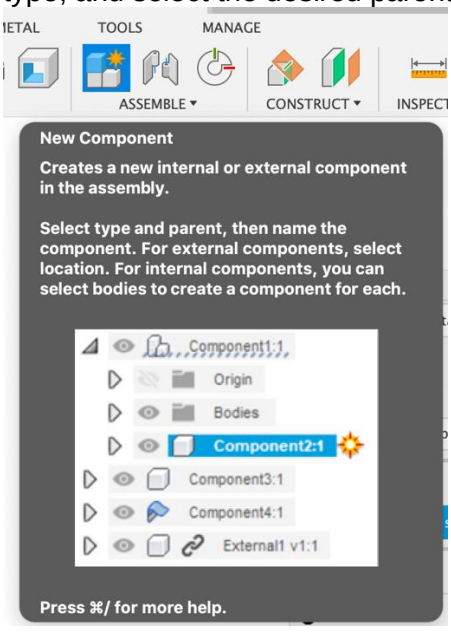
- Canvases

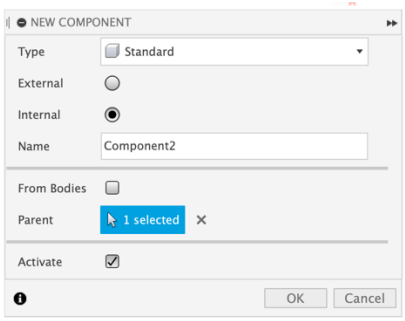


Components in Fusion are a unique concept in CAD products. Most traditional CAD tools have a rigid definition of "part" and "assembly". In Fusion, there is no such distinction. There is only "Component", and a component can have attributes of a part (contains only bodies), or attributes of an assembly (contains child components), or both.

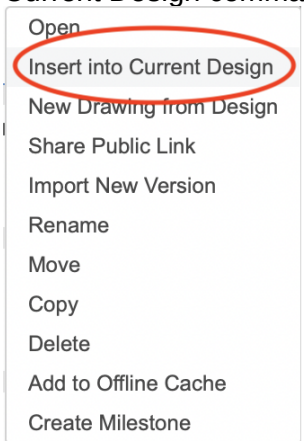
Internal (Local) and External Components

Components in Fusion come in a couple types: Local and External. This is another way that Fusion is different from traditional CAD applications. Most other tools only allow external parts or sub-assemblies. That is, you have to create an entire new design/file for each part or assembly in your design. Fusion, however, allows you to define Internal Components (Components totally defined within the parent document). This can be a very convenient way to design. It does not require you to create a new design for each Component. Just invoke the Create Component command, choose "Internal" as the type, and select the desired parent component:





External Components, on the other hand are added to an assembly with the Insert Into Current Design command, invoked from the context menu of the Data Panel:



External Components are defined in a separate Fusion design. The main difference beside living in a separate document, is that the geometry cannot be modified in the parent Assembly. They are read-only in the assembly. Internal Components can be readily modified.

Internal vs. External Components

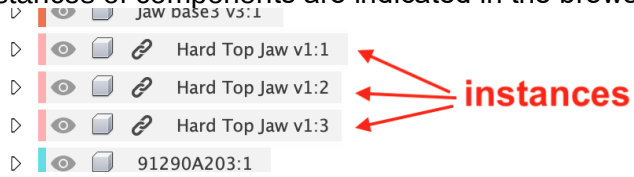
Should you use Internal or External Components? It very much depends on the design and your design process. Internal Components support “top/down” design very well. However, a large design with too many Internal Components can result in a very long and slow Timeline.

If you want to collaborate with other designers, External Components make this more convenient – you can assign individual Fusion designs to multiple designers, and integrate them at the assembly level. This is often referred to a “bottom/up” design.

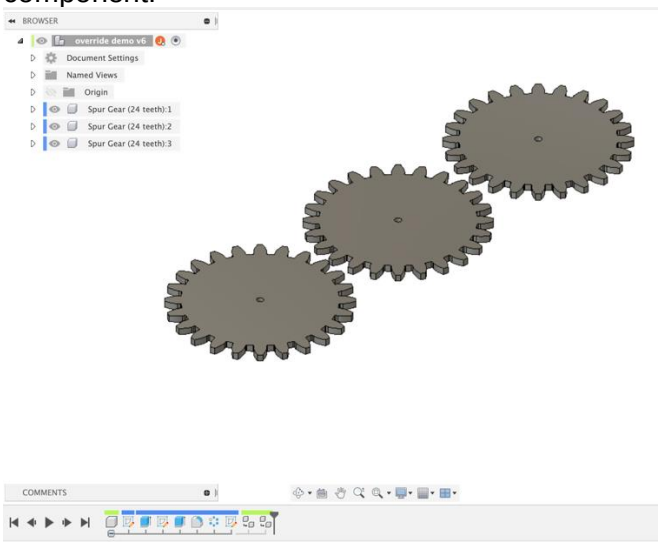
One strategy that many have adopted is to partition the design by subsystems or subassemblies. Define a few External subassemblies, and then, within a given subassembly, all the Components can be Internal.

Component Instances

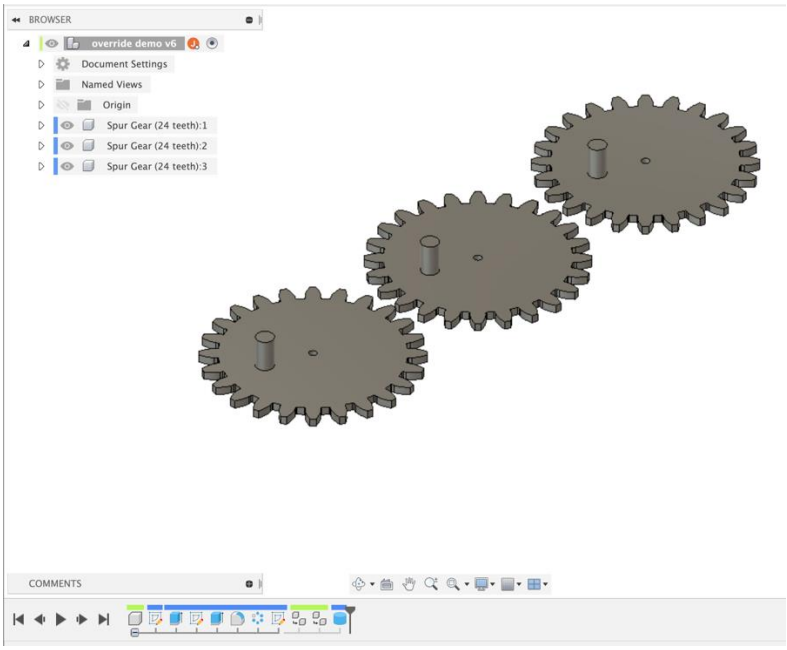
Components in Fusion support “instancing”. Instances of a component are copies of the component, but they are associative copies of that component. That means: all the objects contained in that Component are shared among all instances of that component. Instances of components are indicated in the browser with a “:n” suffix (“:1”, “:2”, etc):



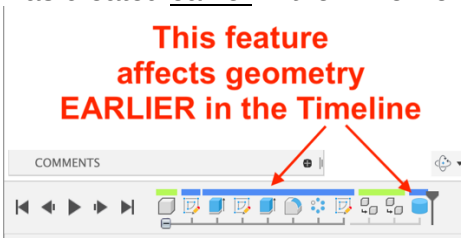
Let’s illustrate how instances share geometry. Here we have 3 instances of a gear component:



If I add a cylinder primitive to one instance of the Component, note that it appears on all 3 instances:



If you think a bit about this, it seems weird. It seems to violate the strict order of the Timeline. Here we have a feature that is later in the Timeline affecting geomtery that was created earlier in the Timeline. How can that be?



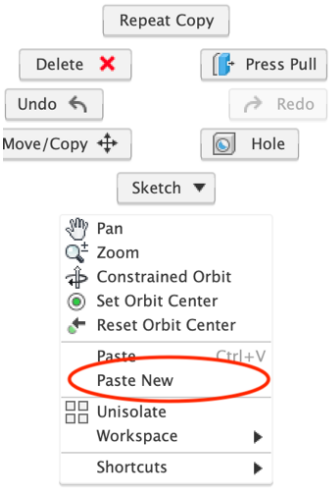
It is because of this “shared geometry among all instances” nature of Components. Even though the Cylinder was added later, it adds that geometry to the Component, which is then shared. Similarly, if you add a child Component, Sketch, Workplane, Canvas, etc to a Component, those will also appear in all instances of that Component.

So, understanding Component instances is important.

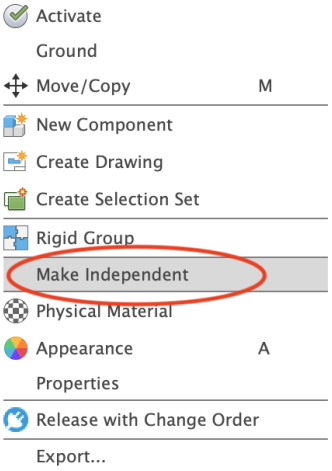
Paste New/Break Link

Given the instance behavior described above, what if you want a copy of a component that is not an instance (that does not share geometry/sketches/etc)? There are two ways (actually 3…).

Internal Components: For Internal Components, instead of Copy/Paste of the Component to create an instance, use Copy/Paste New to create a truly independent copy of the Internal Component.

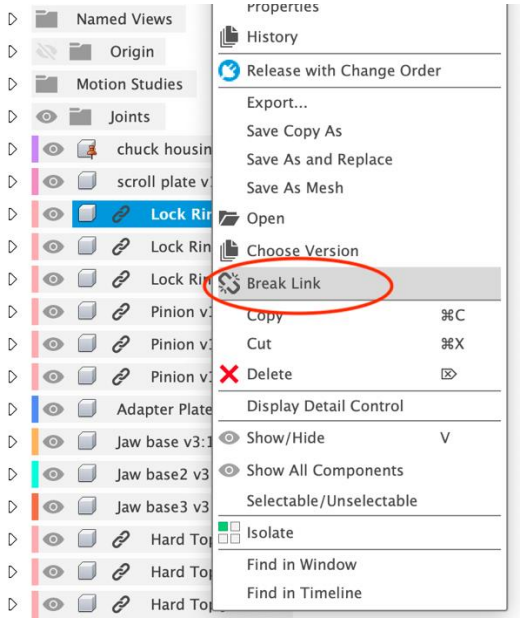


Internal Components (Direct Modeling): In a Direct Modeling design, you can also, by selecting a Component Instance, right click and choose Make Independent.



Why is Make Independent not available in a Parametric Design? It is much harder in a Parametric design, but is on the list for eventual implementation.

External Components: Paste New and Make Independent are not available for External Components. Instead, use Break Link to turn an External Component into an Internal Component, and then it can contain different geometry, etc from the External Component.



Derive vs. Insert for External Components

Some of you more savvy readers may know about Derive. Derive Component is a way to associatively copy the contents of one design into another. In this respect, it is similar to Inserting an External Component, but there are a few differences:

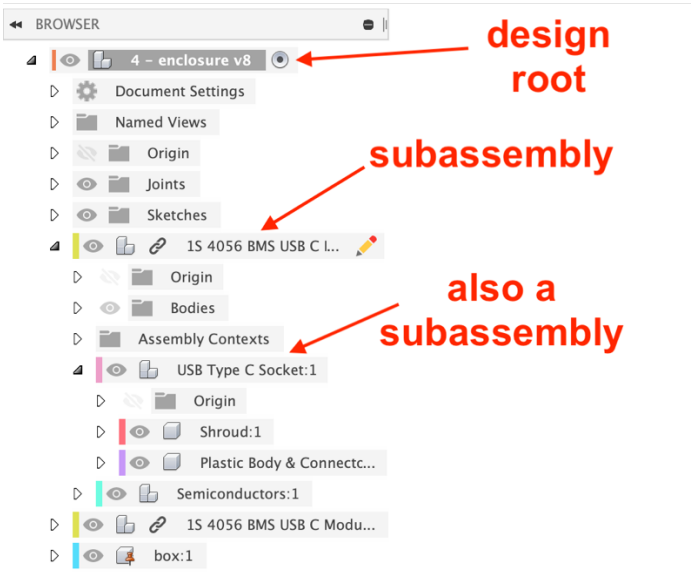
- You can insert a *subset* of a design. A subset of all the child Internal Components, or a collection of Sketches, etc.
- You can *modify* the Derived Component in the parent assembly. Unlike Inserted Components, the geometry from a Derived Component can be modified with features in the parent assembly.
- It is a *copy* of the data. The advantage of a normal Inserted External Component is that it shares all the data with all instances of that Component. Derived Components are copies, and so make the parent assembly heavier

Subassemblies

A subassembly in Fusion corresponds exactly to the usual understanding of the term:

> sub·as·sem·bly
> /ˌsəbəˈsemblē/
>
> *noun*
> a unit assembled separately but designed to be incorporated with other units into a larger manufactured product.

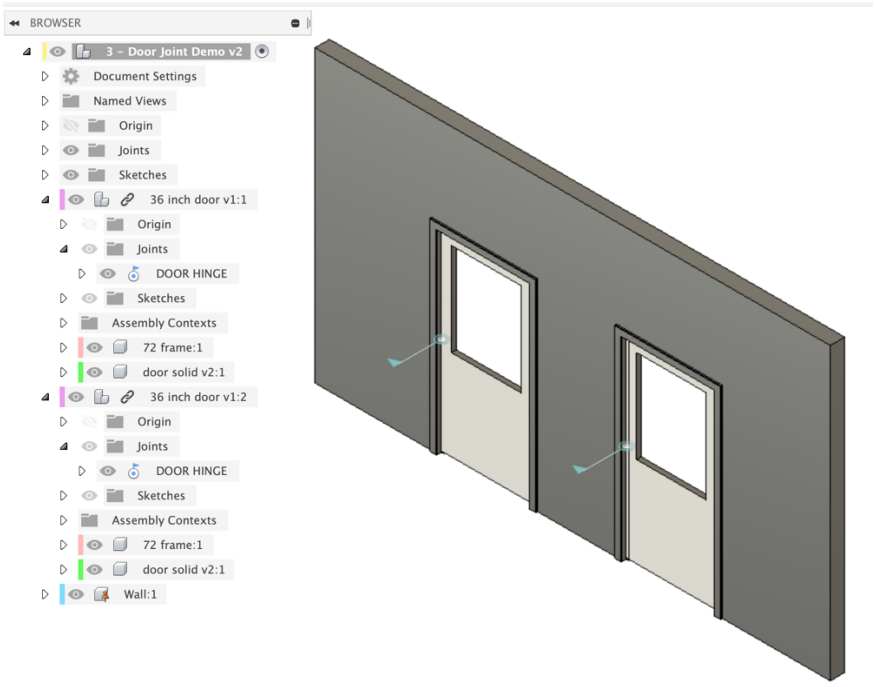
In Fusion, a subassembly is not a formal term used in the documentation or user interface, but is really just a “child Component of the design which, itself, has child Components”. Here is an example:



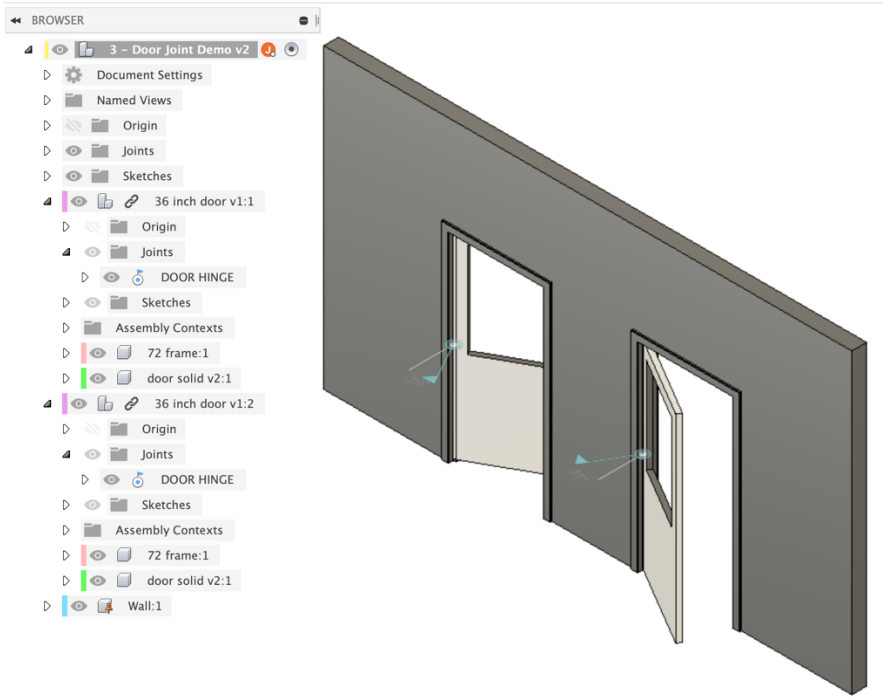
Flexible Subassemblies

One difference between Fusion and other CAD tools you may have used in the past is that all Fusion subassemblies are inherently “flexible”. What does “flexible” mean here? It refers to the fact that, if you have multiple instances (remember, subassemblies are Components, and Components support instances, in which all geometry is shared), the children of each instance can have different positions from each other.

Consider this example. This is a simple assembly consisting of a local component representing a portion of a wall in a building, and two instances of an external subassembly representing the door – consisting of a door frame component and a door component. This subassembly also contains a single Revolute Joint.



Even though the Joint is contained in the subassembly, we can position the joint differently in each instance of that subassembly:



How can that be? Earlier, we said that instances of Components share everything, right? This is one exception to that rule. Child Component position can be different in multiple instances of a subassembly. Further, if you open the door subassembly in a

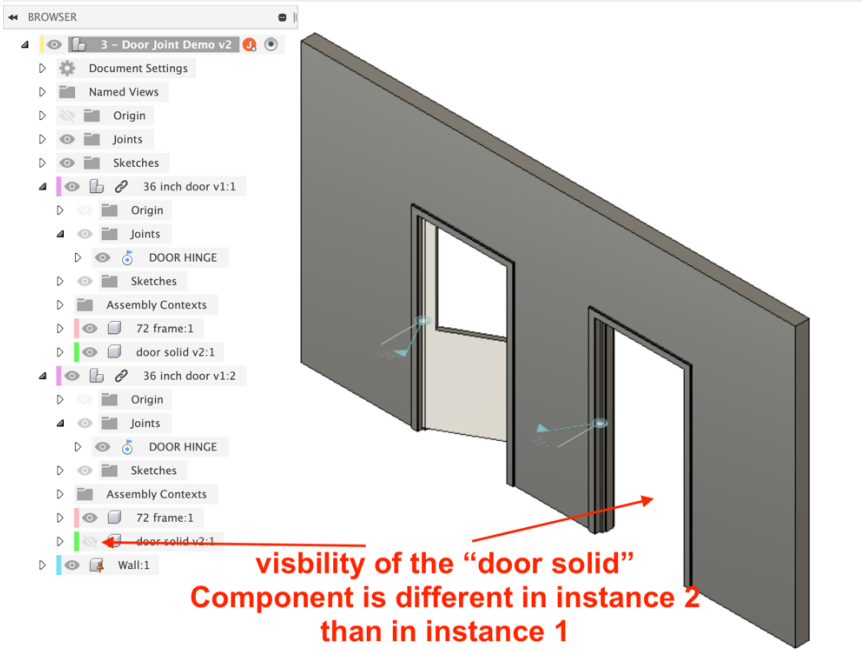
different tab, it will not reflect either of the positions shown above. The next section will explain how that works, and what other exceptions exist.

Overrides

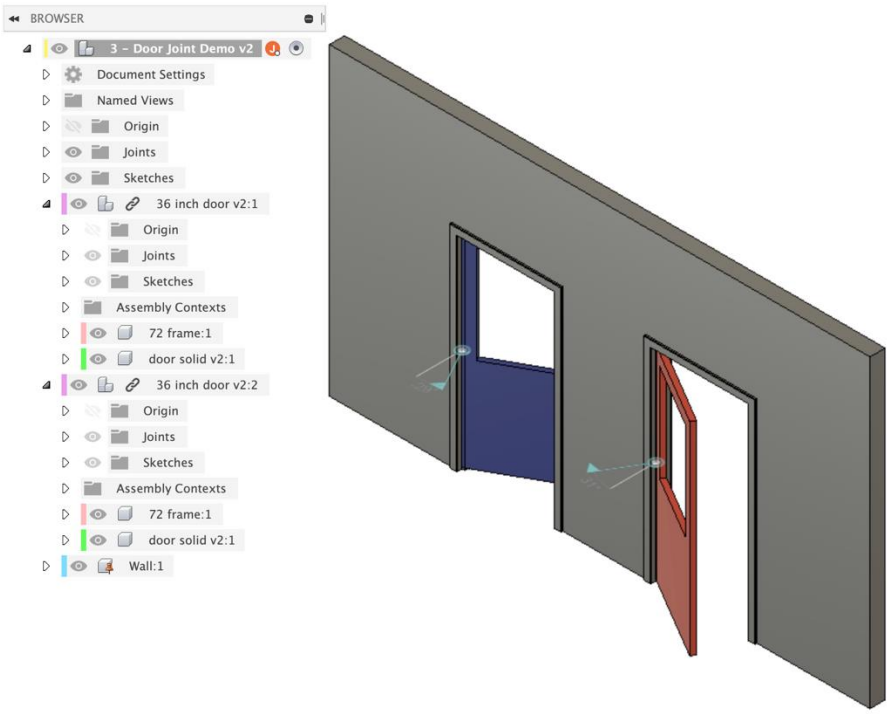
Assembly Overrides exist in Fusion to allow flexible subassemblies to work. Fusion stores the positions of all Component instances in the top-level Component as *position overrides*. So, even though all the geometry in an external Component is shared, the top-level design, by storing these overrides, can enable this flexibility.

What else can be overridden? There are currently only two other attributes of a Component which can be stored as assembly overrides: Component visibility, and Component Appearance.

Here is an example of a visibility override



And a color override:

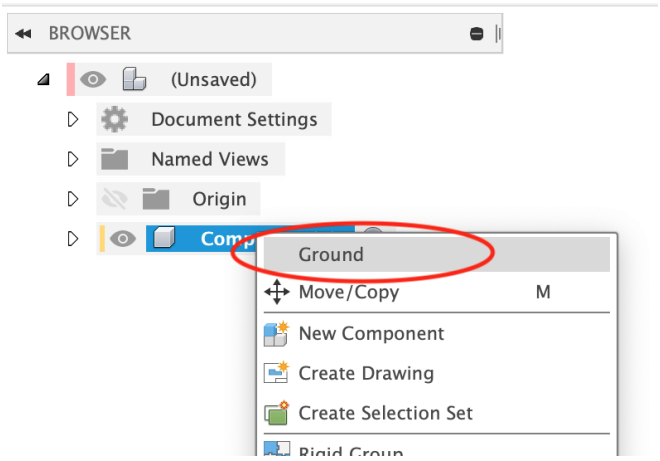


Joints to Subassemblies

The last topic on subassemblies is creating Joints to them. Remember, a subassembly is just another Component, so it has its own coordinate system, as do each of its children. Sometimes users will create a Joint to the subassembly itself. But, think about what that is really saying: You are constraining the Component for that subassembly, but because of "flexible subassemblies", all the child components are still free to move however they want. So, often, creating a Joint directly to the subassembly will appear to have no effect at all. Instead, create joints directly to the child Components of a subassembly.

Ground

A Component can be grounded using the Ground command:



This will fix the selected component in space. Seems simple, no? But, there are some behaviors with Ground that are important to understand. Ground in Fusion is different than in other CAD products you may have used in the past.

Some experts say never to use Ground in Fusion – we disagree. Ground is certainly useful, you just need to understand what it does (and doesn't do).

Ground in Fusion is a Timeline Feature

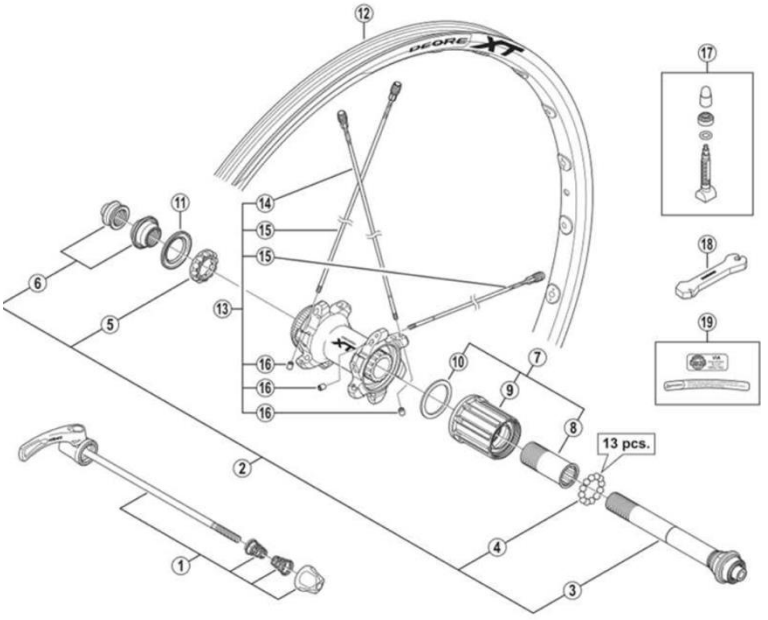
First: Ground is not just an "attribute" of a Component, it is also a history-based feature. So, executing the Ground (or Unground) command too many times can produce a Timeline that looks like this:

In general, this is something to avoid. It is not really a performance concern, but it does clutter your Timeline unnecessarily. There are uses of this Timeline-based nature of Ground, primarily in tricky Joint situations where you want to temporarily ground a Component, then create a Joint, then unground it. That is beyond the scope of this class, however.

Ground does NOT propagate with Insert (External Components)

Ground in Fusion is local to the design in which it is created. That means a component in a design which is then subsequently Inserted into a higher level design will NOT be grounded in the context of that higher level design.

The way to think about this is: Imagine that you are building a subassembly in real life. You'll probably want to fix one part of this subassembly to make it a stable platform for this assembly. Imagine that you are building a bicycle wheel from basic pieces. There are a LOT of parts, and you'll need a stable base to build this.



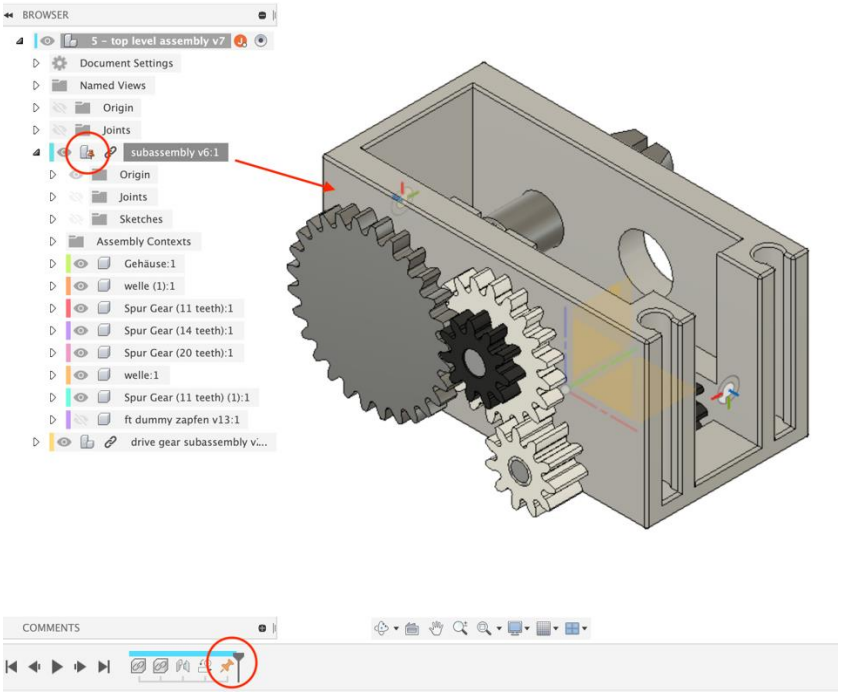
In Fusion, then, you would Ground one of those parts. But, when you assemble the completed wheel into the bicycle itself, you would NOT want the wheel to still be grounded, as that would restrict its movement in the bicycle assembly.

Every Design Should have at Least One Grounded Component

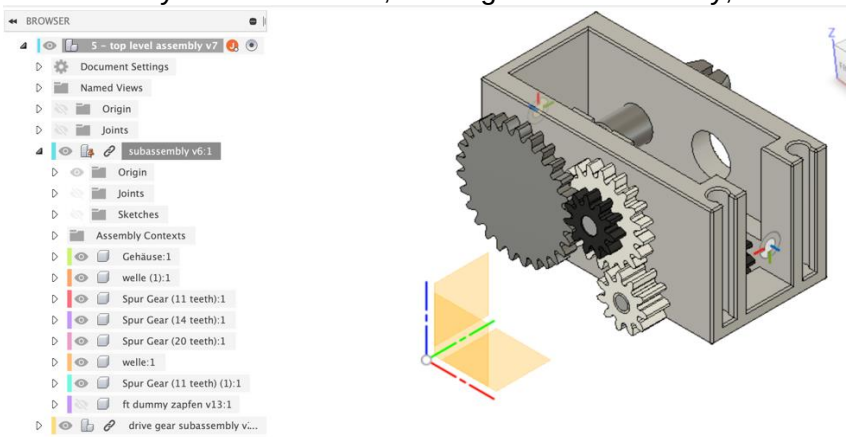
When you are building any assembly (a design with multiple Components), especially one that has motion, you should always have at least one grounded Component. This will let you easily test out your mechanism by dragging a Component. Think about the bicycle wheel above, you will want to test out the spinning motion by dragging. This will require a grounded component, or else the entire wheel will just move around, not rotate.

Grounding a Subassembly Does Not Do What You Think It Does

Relating the Ground topic to the previous Flexible Subassemblies topic, (and the first Component/Origin topic), let's discuss how Ground for a subassembly might not behave as you think it does. Let's look at an example:



Here, we have a design containing a subassembly. We have grounded the subassembly itself. However, if I drag the subassembly, it still moves. Why?



The answer is: Flexible Subassemblies. Here, the subassembly Origin has not moved at all. It is grounded, so it cannot move. But, the child components (Gehause, Spur Gear, etc) are still free to move within the subassembly. So, the Ground has done exactly what you told it to do: it has grounded the subassembly. But, Ground is only following what it was explicitly told to do. You did not tell it to ground the child Components.

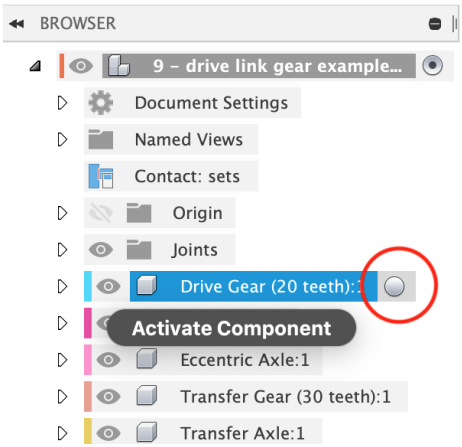
As a recommendation, then – you should ground a leaf-level (part) Component in your design, and not any subassemblies directly.

Ground Alternative

Some Fusion users like to use an alternative to the Ground feature. If you create an As-Built Rigid Joint between a Component in your design and the root Origin, it will have the same effect as Ground. This will have one advantage: Because this is just a Joint, it will get preserved during Insert. The behavior, though, will be to lock the subassembly to its origin, not to the parent Component orign, so the subassembly will effectively be "rigid", as opposed to flexible.

Component Activation

A Component can be activated using this UI:

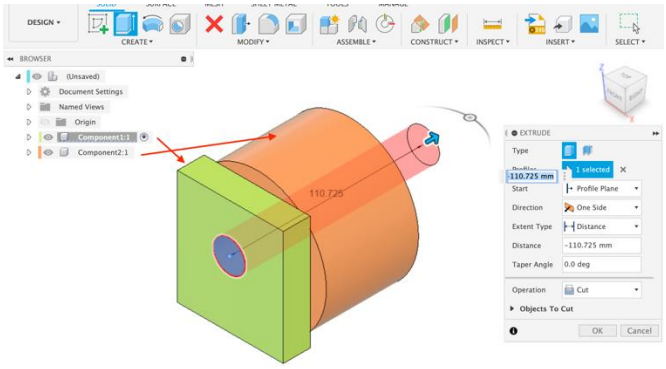


Or using the "Activate" context menu item:

What Does Activate Mean?

Activate is a sometimes misunderstood behavior. Activate does one and only one thing: It determines where (most) new objects go. Remember the list of objects a Component can own? It includes Bodies, Sketches, Work Geometry, Canvases, etc. Any of those objects which are created will be created underneath the Active Command. Remember: the design root, itself, is also a Component, and is active by default. So, by default, for instance, all new Bodies or Sketches will be owned by the root design. To make a child Component no longer active, simply activate the root Component.

What does Active Component not do? It does not limit what other Components can be affected by a feature. For instance, in Extrude, if the type is set to Cut, you can cut Components that are not active. Here, Component1 is active, but this Extrude cuts both Component1 and Component2:



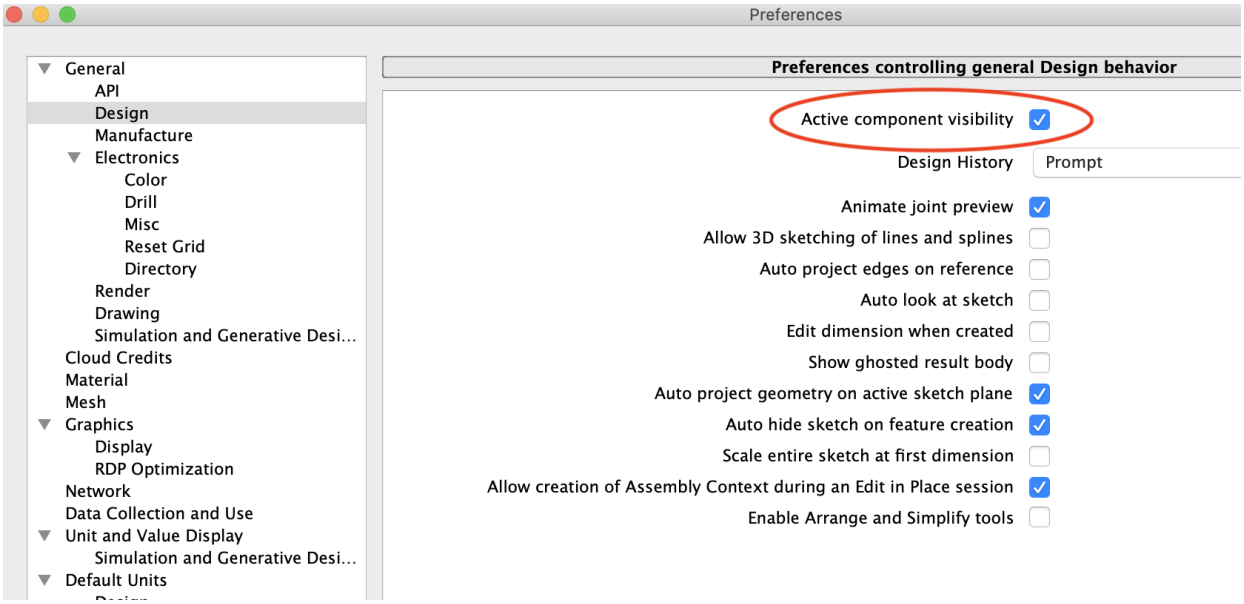
There are a few exceptions to the “all new objects will go into the active Component”:

1. Joint Origins will always goes into the Component that owns the Body geometry selected, even if it is not active
2. Split Body will always put new bodies in the Component that owns the Body being split

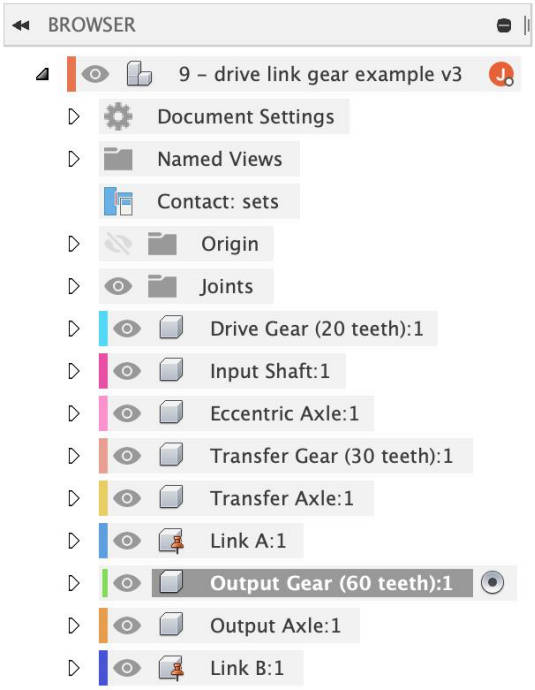
Hopefully the reasoning behind these exceptions is obvious. It would be confusing if, for instance, Split Body created a new body in a different component than the original.

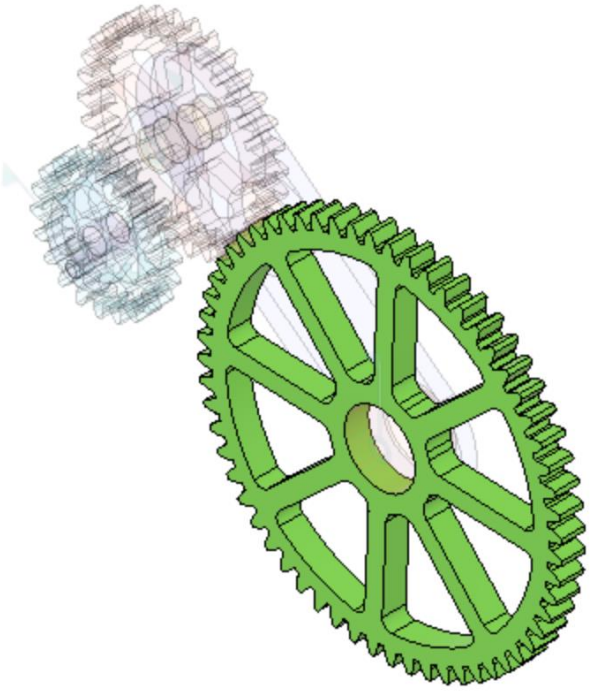
There are two settings related to Component activation that are important to understand:

Active Component Visibility Preference
This preference determines the appearance of non-active Components when a Component is active.

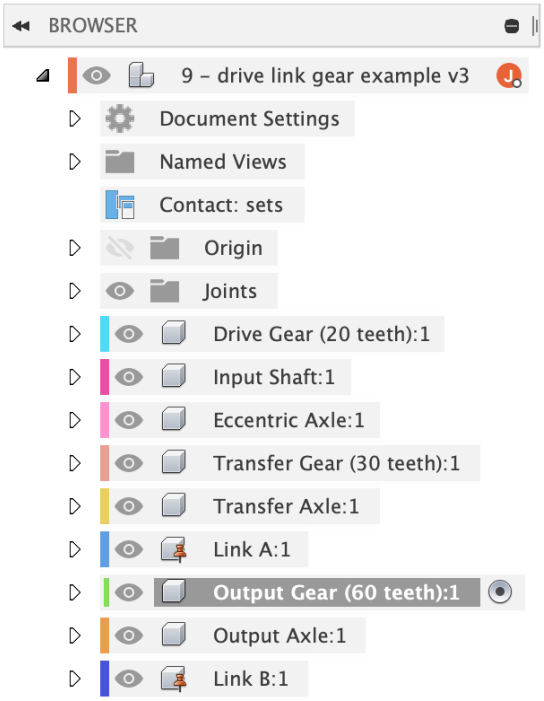


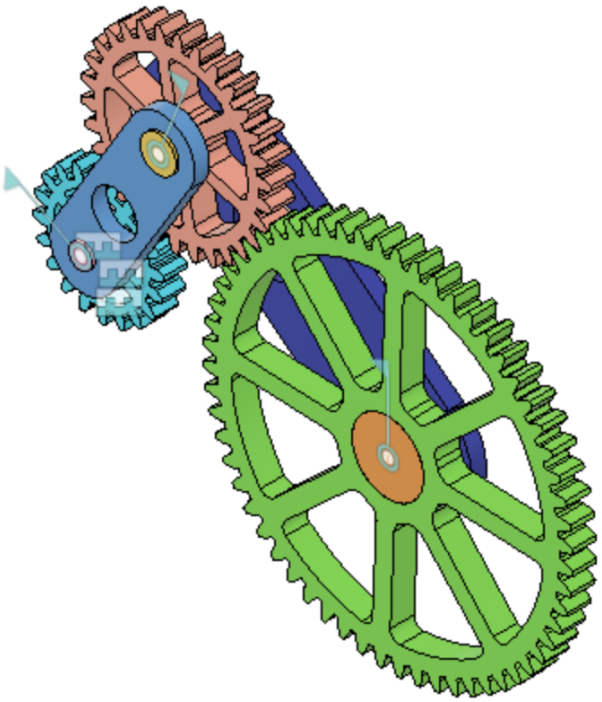
If this setting is enabled, non-active Components are shown as translucent:





If it is disabled, non-active components are rendered in their default appearance:

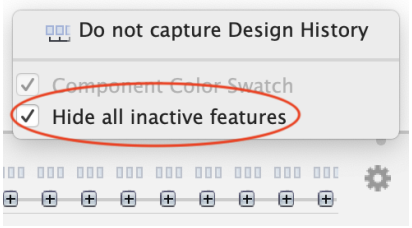




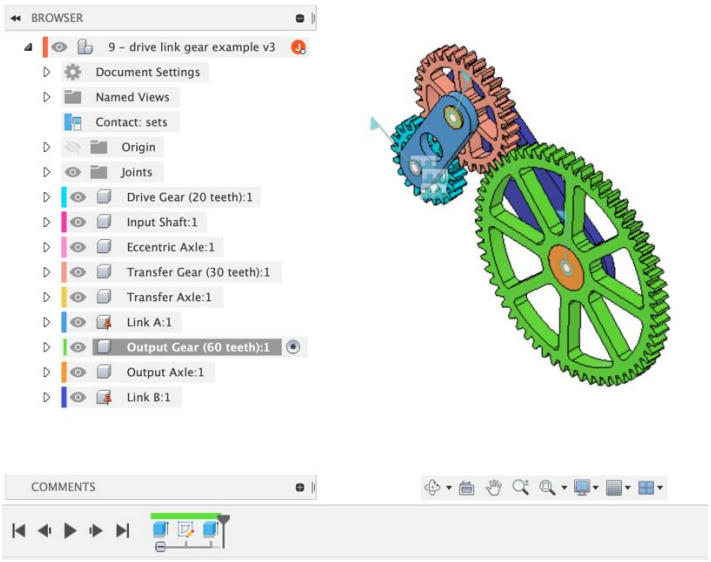
There are no large implications of this setting. It is mostly personal preference. In some very large models, the translucent rendering of non-active Components could be a bit of a graphics performance concern.

Hide Active Features setting

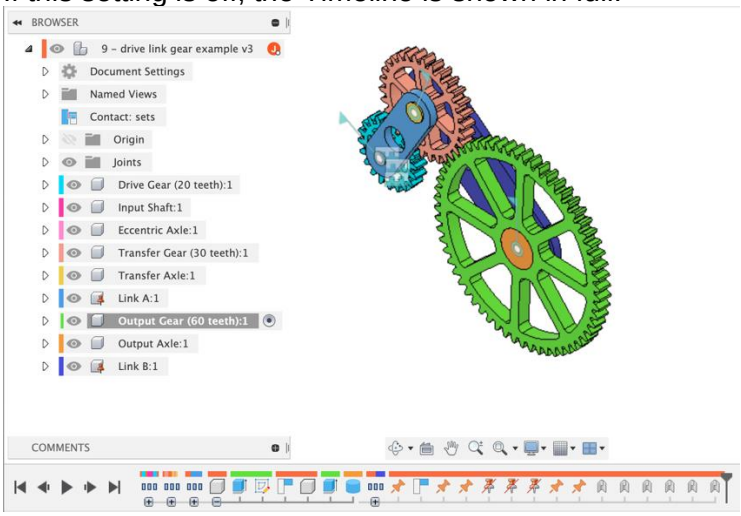
This is not a Preference, but, instead, is a setting accessed via the Timeline "gear" menu:



This setting controls the appearance of the Timeline when a Component is active. If this setting is on, the Timeline is filtered to show only the features that are owned by the active Component:

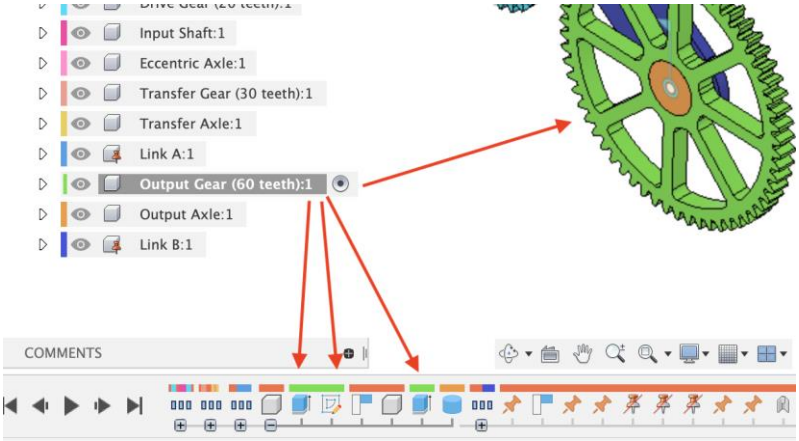


If this setting is off, the Timeline is shown in full:

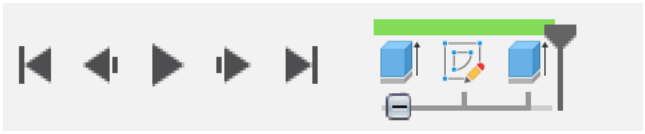


It is important to understand the nature of this filtering. When filtered, remember: the rest of the features still exist in the Timeline, they are just not shown. So, be very careful when doing Timeline operations such as Shift-Selecting features, because this will also select the features in-between, even if they are not visible.

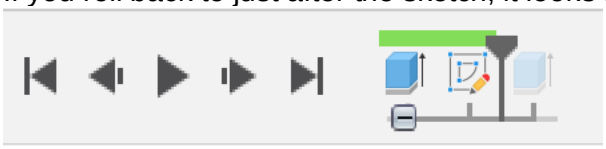
For example, in this particular case, you can see that there are 3 features for the “Output Gear (60 teeth)” component. Without the filter, the timeline is this:



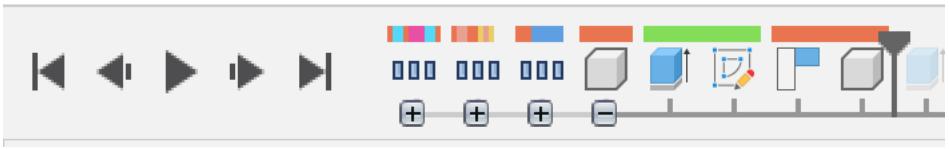
Note that there are two other features in the middle. If you filter by active component, it looks like this:



If you roll back to just after the sketch, it looks like this:

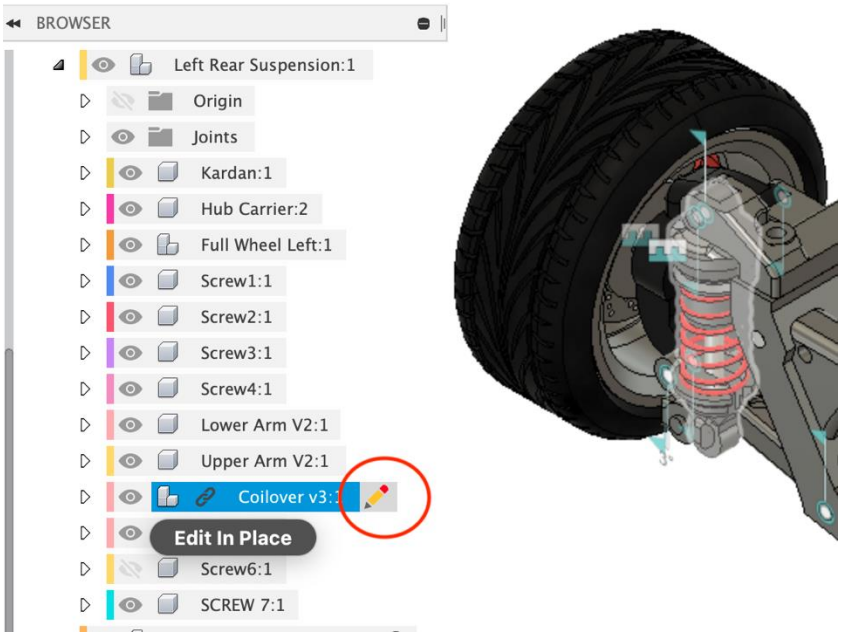


So, you may think that the end of design marker is just after the sketch, but in reality, it looks like this:

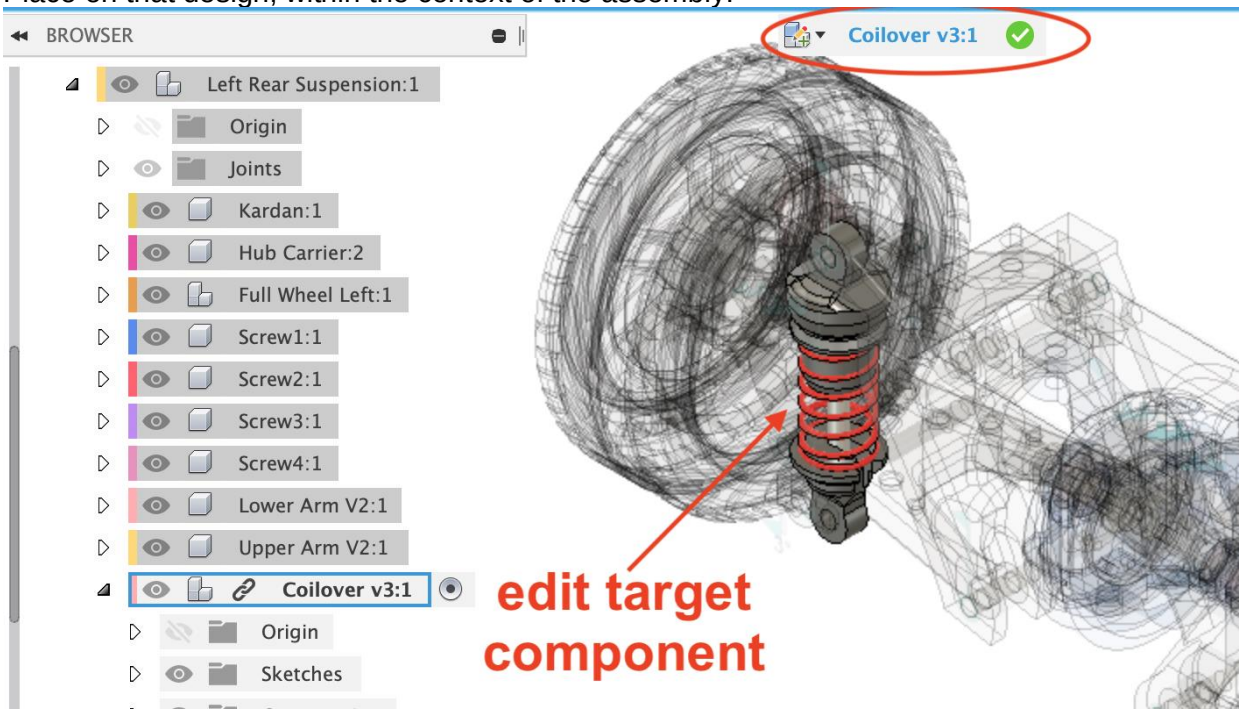


Edit in Place

A relatively recent enhancement to Fusion implements “Edit in Place” for External Components. We will not go deeply into this topic (that deserves a class all its own). The main idea, though of EIP is that you can edit an External Component in the context of the top-level assembly, including associative references to other Components in the design:



Note that “Coilover” is an External Component. Clicking on the pencil icon will invoke Edit in Place on that design, within the context of the assembly:



The active document is now set to “Coilover”, and clicking the checkmark at the top will return you to the assembly context.

One important advantage of EIP for editing External Components: There is no need to Get Latest. When you return from EIP, the parent assembly is automatically updated with the edits to the External Component.

Capture Position

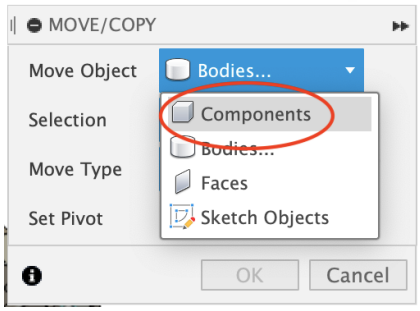
Capture Position may be the single most misunderstood aspect of Fusion assembly modeling to new users, so it is worth some explanation.

One major difference between Fusion and some other CAD systems is that Fusion assemblies are Timeline based. Inserting components, adding joints, etc. generate timeline entries. Other, traditional CAD tools do not represent assembly operations in a feature browser. One way in which this is important is in moving Components. Moving Components in Fusion is different than moving, say Bodies. If you move a Body, Fusion immediately puts a Timeline feature in for the move. However, we want to be able to dynamically drag Components to test out Joint systems, so moving a Component does not add a Timeline feature.

Each Component has a parametrically stable “home position”. This is the position that a Component returns to when you do a Compute All. Any time a Component is not in its home position, Fusion knows that it has been moved.

There are several ways to move a Component:

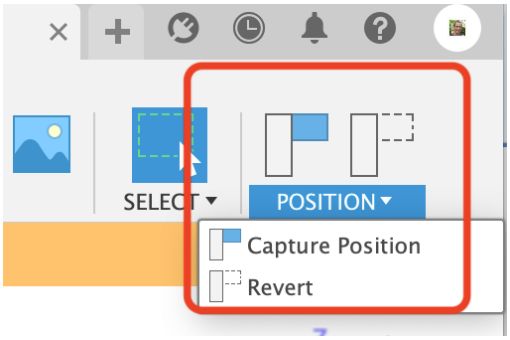
- Drag. You can just click and drag on any Component geometry, and the Component will be moved from its home position.
- Move Component. The Move command can be set to move Components:



in this mode, the movement will be applied to the selected Component(s)

- Align Component. The Align command can also be set for Components
- Drive Joint. Exercising a Joint will also move Components
- Animate Joint/Animate Model. Simiarly, these commands will move Components

Any time any Components are not in their home positions, you will see this UI in the upper right hand corner of your Fusion session:

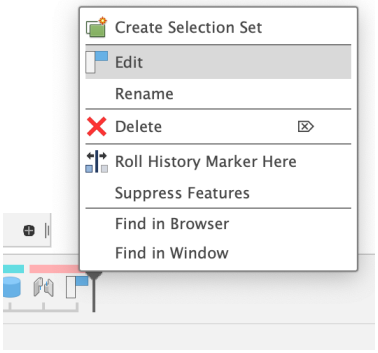


This UI is called the “Capture Position Pending” UI. It just means: “one or more of the Components in this design are not in their home positions. There are two actions you can take: Capture Position or Revert. Revert will return Components to their home positions. Capture Position will insert a Capture Position feature into the Timeline:

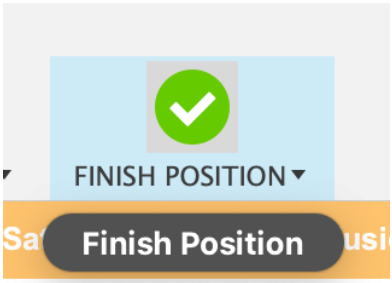
Capture Position Feature

This feature (as its name implies) records the position of every moved Component in the design. It effectively establishes a new home position for those components. **Note**: Home position for a Component, then, can vary over the Timeline. If you roll back before this feature, Components may have a different home position than they will after.

Did you know? Many people do not realize that this feature can be edited. Right click and choose Edit:



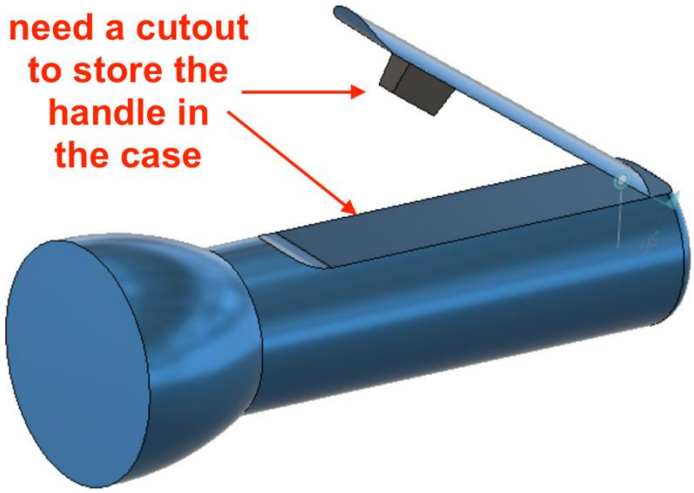
Then, you can move Components around with any available move method, and when done, click “Finish Position”:



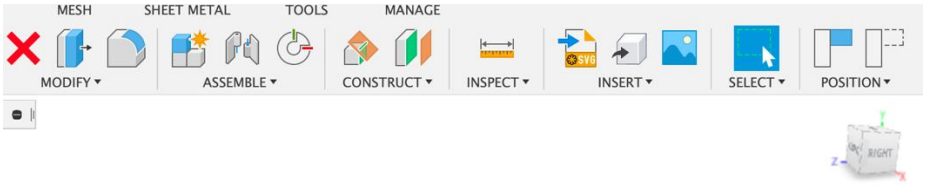
Often, new users will create multiple Capture Position features next to each other in the Timeline. This is inefficient. Edit the Capture, move Components, then Finish Position instead.

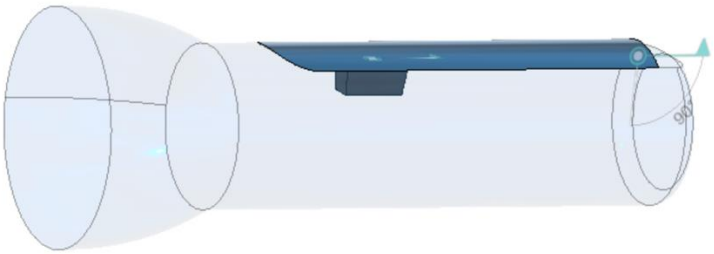
Position-based Modeling

So, what is Capture Position good for, then? One of the advantages of Fusion’s parametric, history-based assembly is that you can do what we call “position-based modeling”. That is, where modeling features such as Extrude, Project Edge, etc can be *dependent on a Component’s position*. This can be a powerful modeling tool. We’ll look at a simple example which, I hope, illustrates the power. In this example, we are designing an emergency flashlight. This is the kind of flashlight that has a handle you can crank, and a little generator inside, and a battery. Turning the crank charges the battery, which runs the light. Early in the design, we realize that we need a cutout to store the rubber handle on the crank:

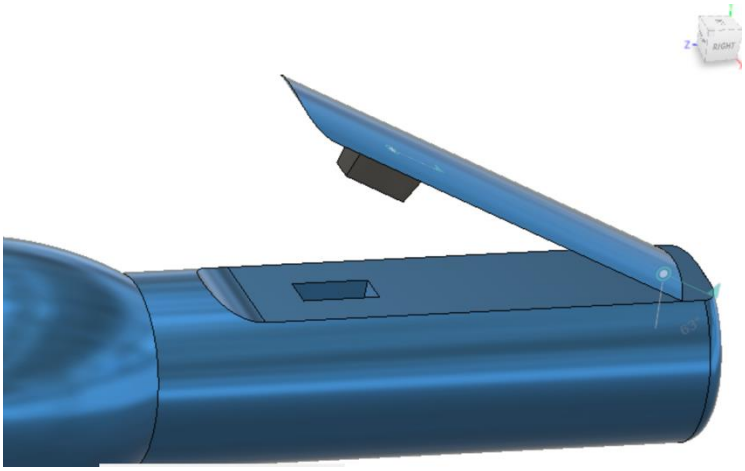


So, we can close the handle, which shows the Capture Position Pending UI:

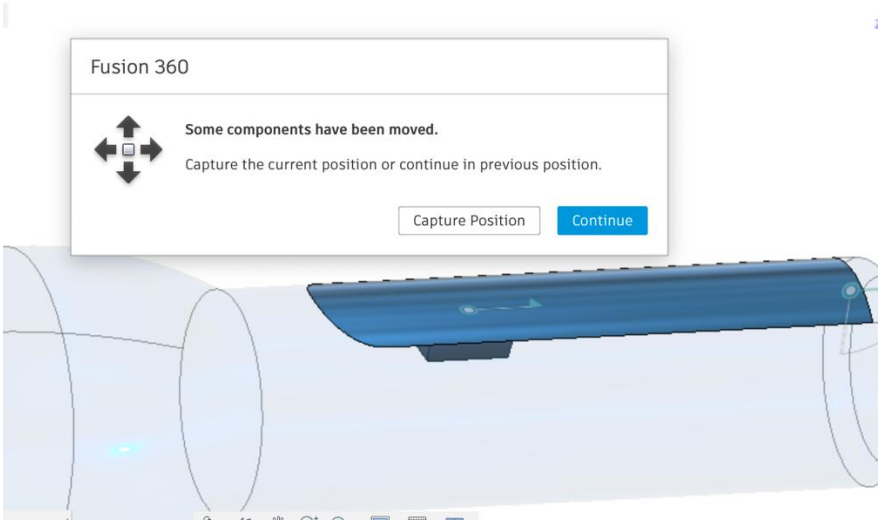




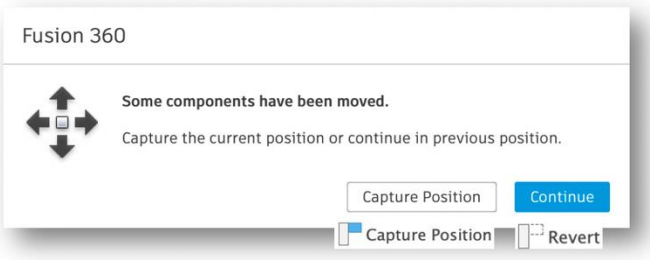
You can see that the handle now interferes with the case. But, if we Capture Position, we can record that position as important to the design. Then, we can use Combine to subtract the handle geometry from the case, resulting in a perfectly aligned cutout:



Capture Position Prompt
In the previous workflow, if I had not done Capture Position, and instead, started the Combine command in a Capture Pending state, I would have seen this dialog:



This is Fusion just telling you: You are about do execute a command which is “position dependent”, and you have some Components not in their home positions. You can either Capture or Revert these positions. Note that “continue” in the dialog corresponds to Revert:



Capture Position Caveats

A few things to know about Capture Position:

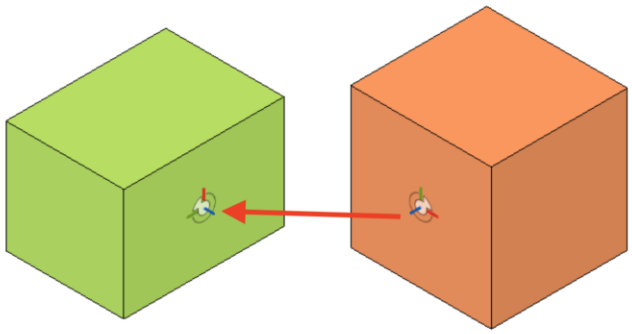
1. Capture Pending is not a stable state. Components that have been moved are not always preserved. Most importantly, Insert Into Current Design does not preserve them. Often, we see Fusion Forum posts with a title like “my Inserted design does not match how it was saved”. Almost every single time, this is because it was saved in Capture Pending state. So, if the position of Components is important to you, capture it. But also: You should be using Joints to position Components anyway, not Move or Align, which is the point of this class anyway.
2. Capture Position features can be expensive. If you think about what this feature does, you will understand. Capture Position changes Component position. If you have Joints in your model, this will cause an assembly solve to happen. So, if you have dozens of Capture Position features in your Timeline, and hundreds of Components, this could be expensive. So: Capture Position is very useful, but use it purposely, and avoid having too many in your Timeline.

Fusion 360 Joints

Fusion’s assembly modeling paradigm is unique in the CAD industry. One of the main ways is the terminology used when assembling components. In most other CAD systems, one uses a set of “assembly constraints” (or “mates”) to connect multiple parts together. Assembly constraints are highly “atomic” operations, and use mathematical concepts such as Tangent, Align, etc. With Fusion, whose motto is “Design Differently”, we set out to elevate the concepts used in assembly operations. So, Fusion uses Joints to move the terminology into higher-level concepts that more closely mimic the real world. When you are assembling a door, and are installing a hinge, you don’t think about “aligning the axes of the two sides of the hinge and the hinge pin. Instead, you think: “I want to install this hinge”. So, rather than Align constraints, and 3 Mate constraints, you just create a hinge (Revolute) Joint in a single operation. It takes some getting used to, certainly, but we think it helps to simplify the assembly modeling process in a useful way.

What is a Joint in Fusion 360?

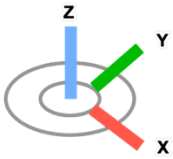
A Joint in Fusion is a pretty simple concept, at its root: It is a direct relationship between two coordinate systems (XYZ axes, origin, etc) that are associated with two different Components.



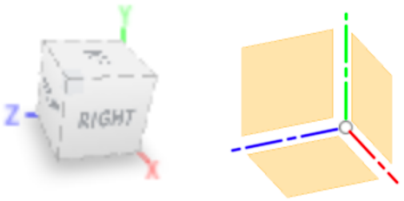
In Fusion, these coordinate systems are called Joint Origins. They are owned by a Component (another object owned by a Component!) and provide handles with which to create Joints.

Joint Origins

As we said, all Joints are relations between two Joint Origins, and Joint Origins are themselves essentially coordinate systems. Here is a blown-up representation of a Joint Origin:

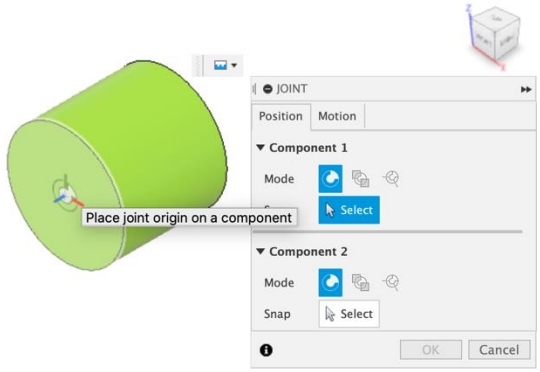


You can see the 3 axes and the origin here. Tip: Everywhere in Fusion that there is a coordinate system, it uses the same color coding: red is X, green is Y, and blue is Z. You can remember this with “RGB -> XYZ”.



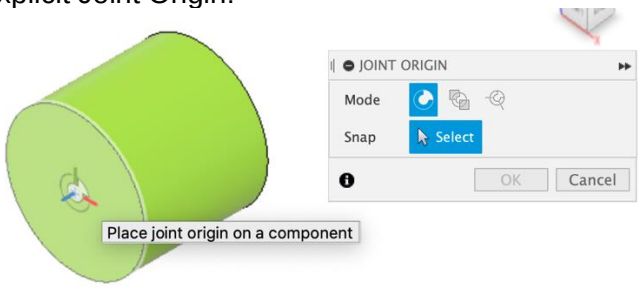
Joint Origins can be created either “implicitly”, as part of the Joint command, or “explicitly” by invoking the Joint Origin command itself.

Implicit Joint Origin:

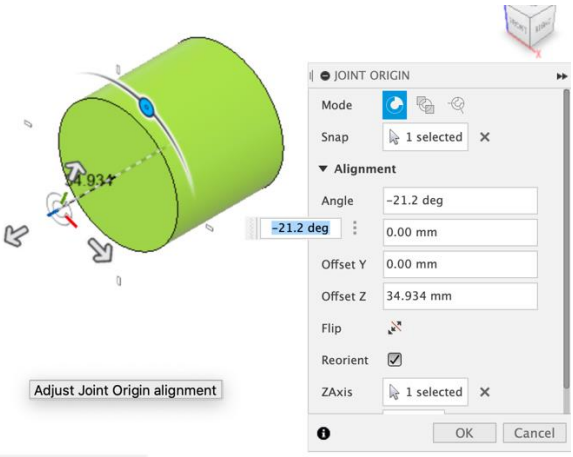


You can see in the preview, and even in the prompt that this is a Joint Origin.

Explicit Joint Origin:

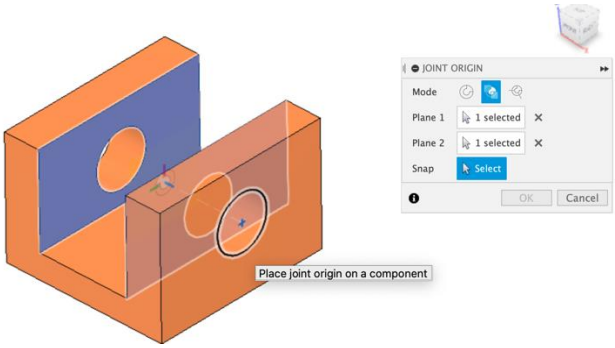


The Joint Origin command offers more flexibility in placing JOs in your Component. You can manually adjust the position and orientation of the Joint Origin to any custom orientation, including selecting additional geometry to help orient it:

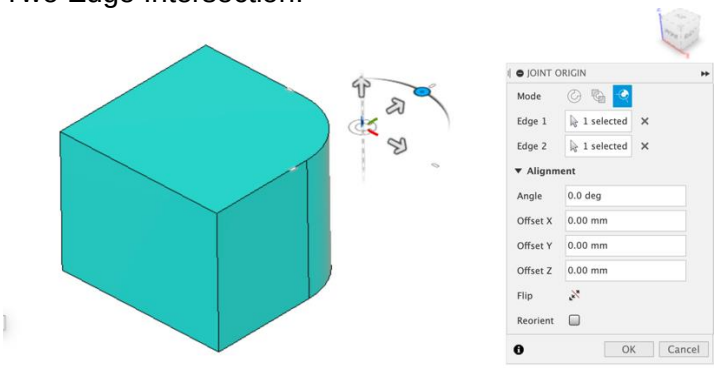


Joint Origins also support two advanced methods of placement.

Between Two Faces:

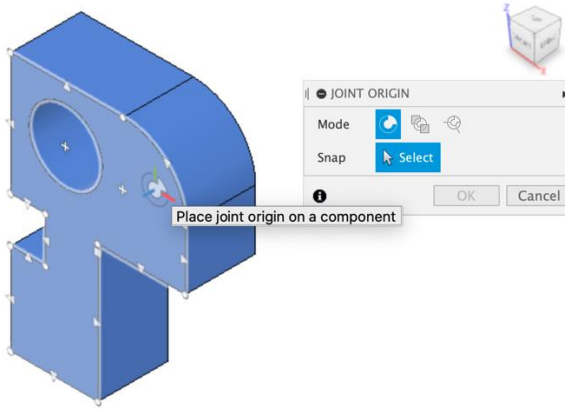


Two Edge Intersection:



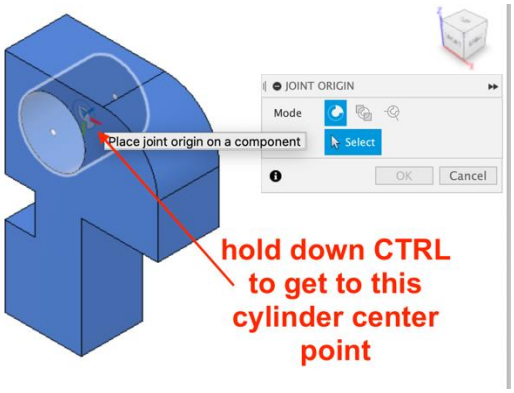
Joint Snap Points

When placing a Joint Origin, using the default Simple mode (selecting Component geometry), if you hover over a body face, you will see a set of snap points that define “interesting” positions on that face:



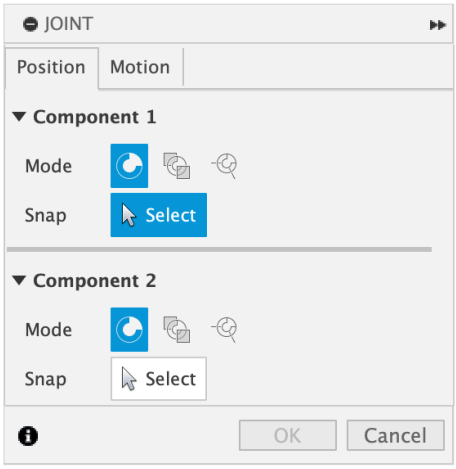
The circles indicate vertices, the triangles indicate edge midpoints, squares indicate the center of the face, and the pluses indicate center points (of arcs).

You can hold down CTRL (CMD on Mac) to “lock in” a face of interest, in case the snap point you are after is obscured by other geometry:



Joint Purposes

Joints have two purposes in Fusion – they determine the position of the two components being joined, and they determine the relative motion of them. This is reflected in the user interface: There are two tabs in the Joint command, one for Position, and one for Motion:



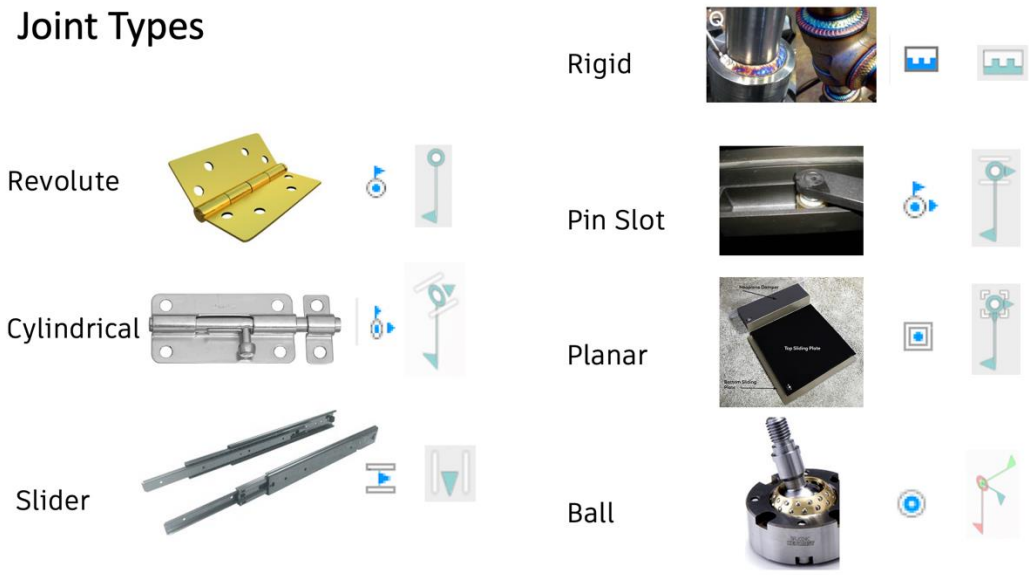
Joint Creation Commands

There are three commands that can be used to create Joints in Fusion:

1. “regular” Joint command. This command creates a Joint between two Components that are not already in position (specifying both Position and Motion)
2. As-Built Joint command. This command creates a Joint between two Component whose home position is already as desired (specifying only Motion)
3. Rigid Group. This is a special case of Joint creation: While every other Joint is between two and only two Components, Rigid Group is a way to specify one type of Joint (a Rigid Joint) between any number of Components. It is essentially an As-Built Joint (Motion only, where the Motion is already chose to be Rigid) between any number of Components.

Types of Joints

There are 7 types of Joints in Fusion:

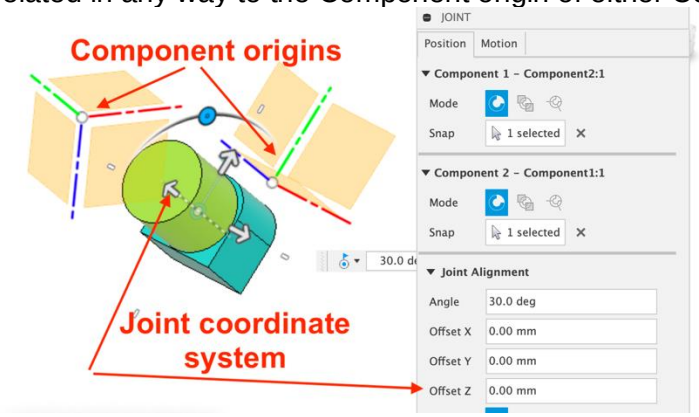


The Joint types only describe the motion of the Joint. Remember, Joints are relations between two coordinate systems (Joint Origins). The Joint type describes the motion along the axes or planes of those Joint Origins. That is the only difference.

- Rigid does not allow any motion
- Revolute allows rotation about one axis of the Joint Origins
- Cylindrical allows translation and rotation about one axis
- Slider allows translation about one axis
- Pin Slot allows translation and rotation (like Cylindrical), but about different axes
- Planar allows translation and rotation about one plane. Note: this is not a "Mate" constraint, and you should not try to use it that way
- Ball allows rotation around all 3 axes of the Joint Origin

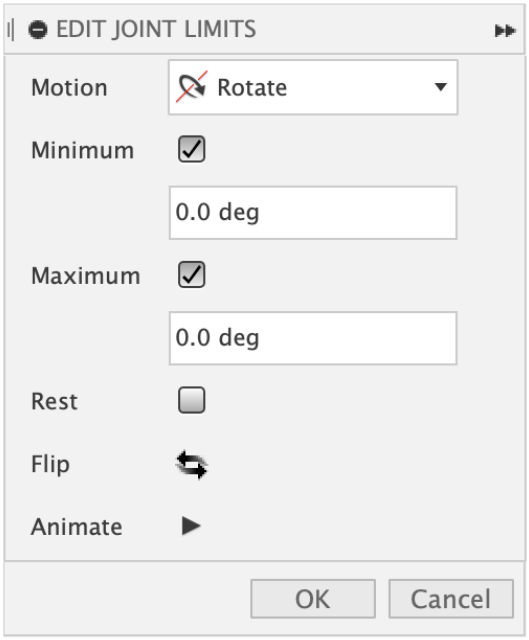
A Note About Axes in Joints

The above description refers to degrees of freedom around axes. It is very important to understand that these axes are the axes of the two Joint Origins involved in the Joint. It is NOT related in any way to the Component origin of either Component or of the root of the design.

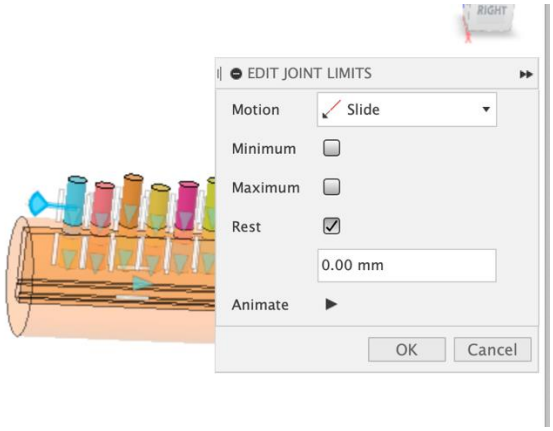


Joint Limits and Rest Position

Joints support motion limits. Imagine you were modeling a hinge, and wanted to limit the motion. You can set a minimum and maximum rotation:



Joints also allow a "Rest" position. This can be used to simulate simple spring motion:

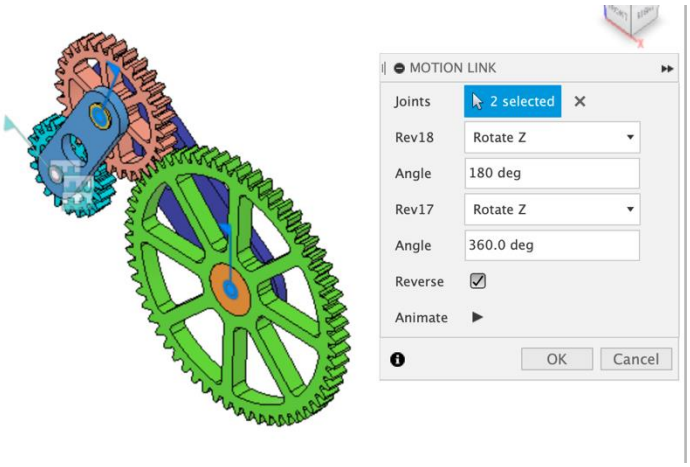


The Rest position on this Slider Joint allowed me to simulate the spring motion in this lock mechanism

Motion Link

Motion link allows you to link the motion of two different Joints to each other. This is very useful in modeling gear systems (Do NOT attempt to model gears using Contact Sets…)

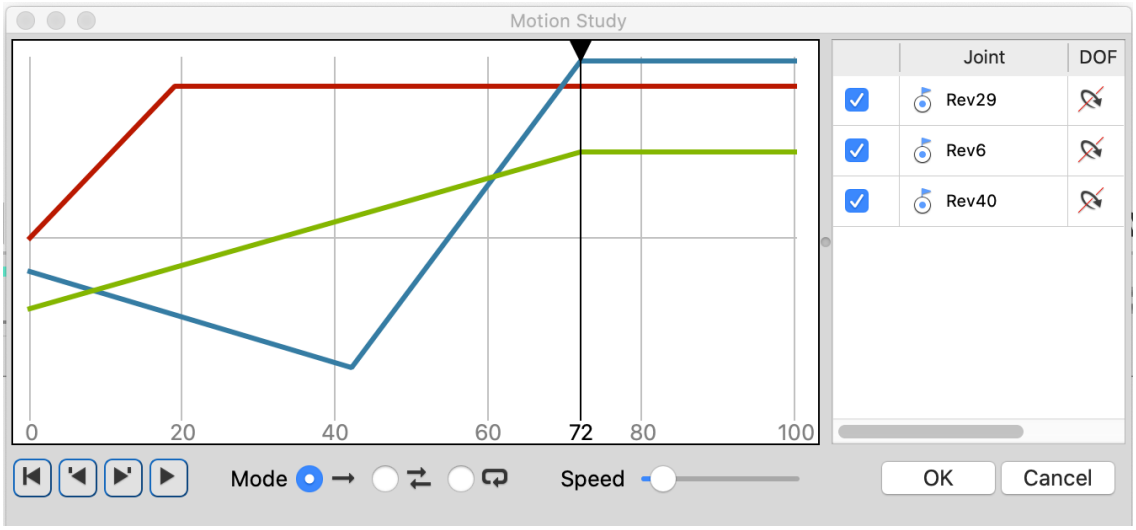
Here we are modeling a 2:1 gear ratio between two Revolute Joints:



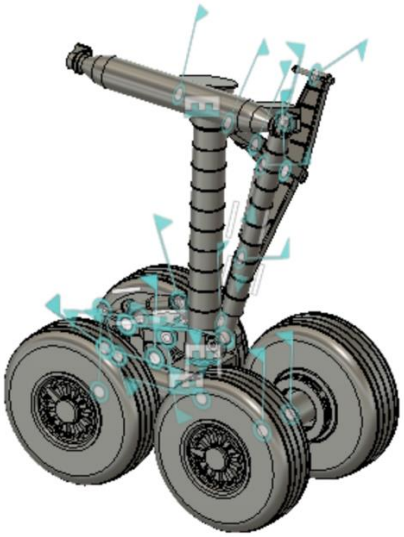
Tip: When modeling gear ratios, though the relationship is expressed in degrees, you can really use the gear tooth ratios just as easily. In this case, the larger gear is 60 teeth, and the smaller is 30, so I could have gotten the same result using 60 and 30 in the Angle inputs. It is a ratio, so the values don't matter

Motion Study

A Motion Study can be used to relate the motion of any number of Joint, using a graph-based UI:



A Motion Study can model a very complex mechanism such as an airplane landing gear, where the doors open, the gear retracts, then the door closes, that a Motion Link could not model

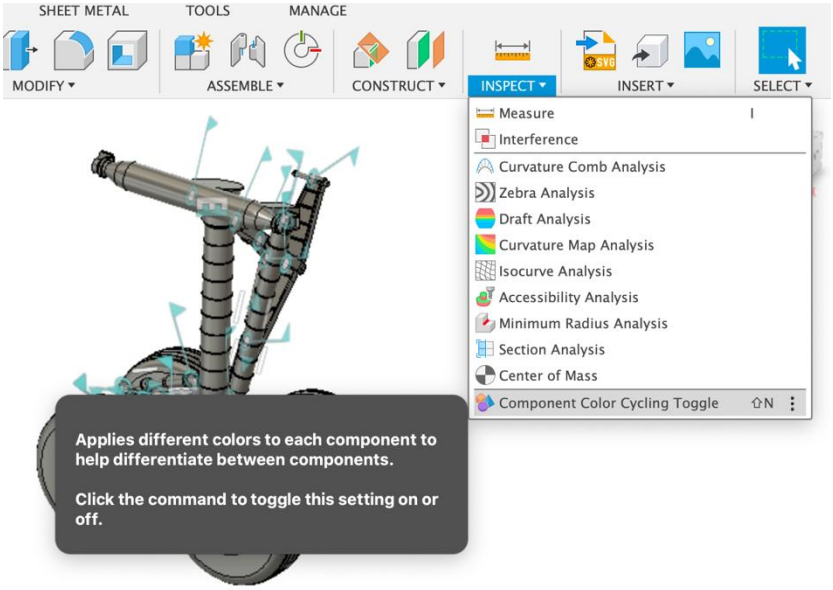


Assembly Debugging

In this section, we will discuss a few techniques that can be helpful in debugging your Fusion assembly model.

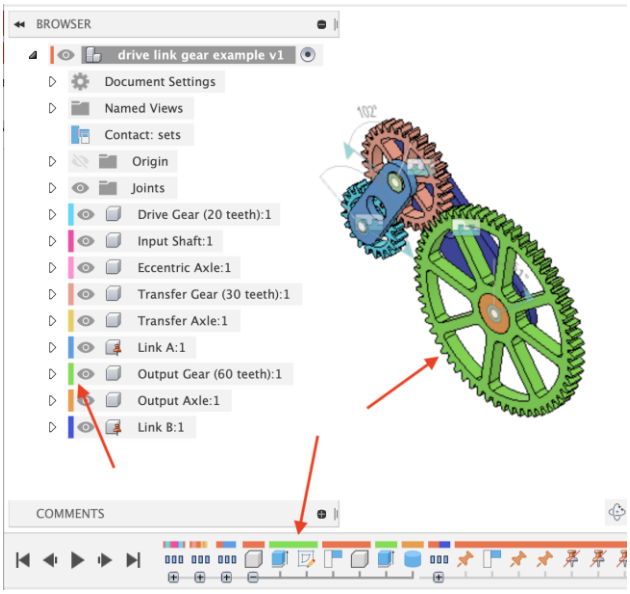
Component Color Cycling/Component Color Swatches

Component Color Cycling is available from the Inspect dropdown:



It can also be invoked using the shift-N hotkey. Often, new users find themselves in this mode unexpectedly, and almost always that is due to an inadvertent pressing of shift-N.

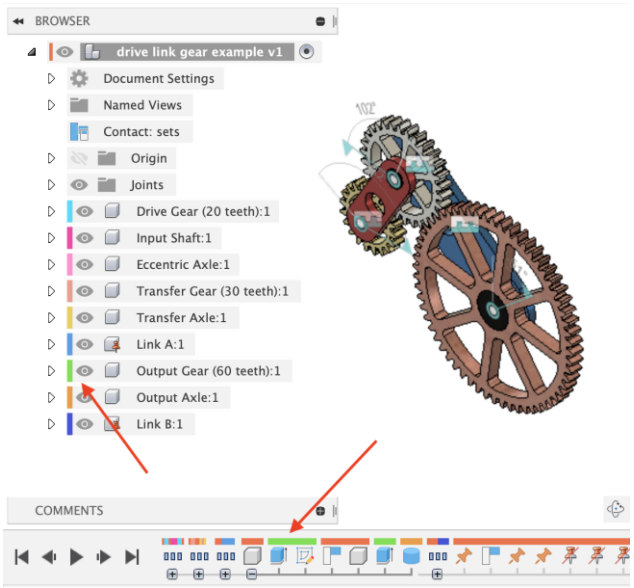
This mode helps illustrate different Compoent boundaries using bright colors:



Note that the color is shown in the graphics area, the Timeline, and the Browser.

Sometimes, though, this is just too visually jarring. Then, you can switch on its cousin: Component Color Swatches, available from the "gear" icon to the right of the Timeline:

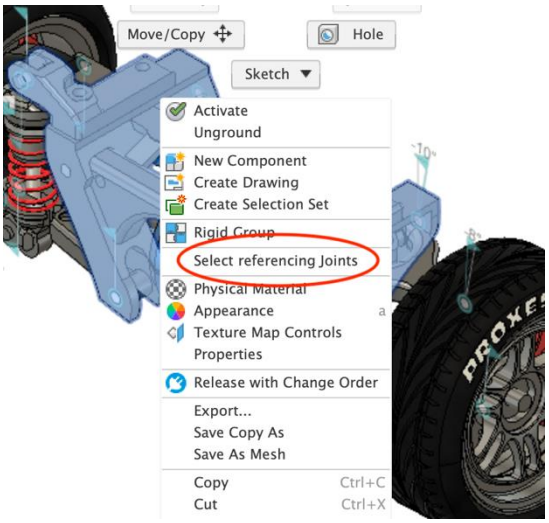
This mode preserves the native Component colors, while keeping the Timeline and Browser colors:



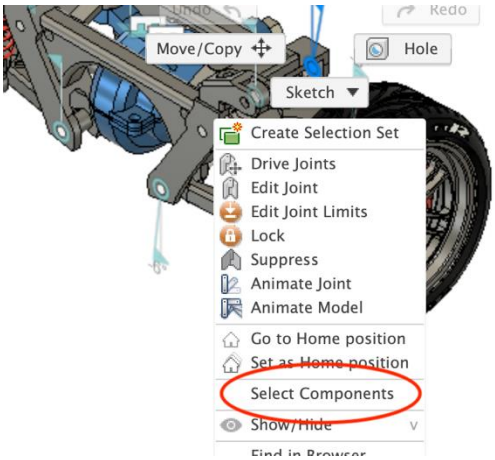
Navigating from Joints To/From Components

Sometimes, it is not obvious which Joints are between which Components. There are two useful context menu commands to help here.

If you select a Component (remember, double click in the graphics area will select a Component), and right click, you will see the "Select Referencing Joints" command. This will select the Joints which this Component is involved in:



Conversely, if you select a Joint and right click, you will see "Select Components". This will select the two Components that this Joint relates.



Is Your Joint System "Stuck"?

Sometimes, your Joint system will appear to get "stuck", and refuse to move. Check these conditions to help repairing it:

Check Contact Sets

Contact Sets in Fusion can be powerful, but they are a performance hog. Just having them turned on can cause a Joint system to appear to be stuck. Turn them off.

Precision is Critical for Joints

Especially for a closed system of Joints (e.g. a 4-bar linkage), your geometry, angles, etc must be exact. The most common cause for Joint failure is imprecise geometry. Go back, check your distances, hole placement, and angle alignment.

Check Motion Links

Motion Links are powerful, but they can also interfere with Joints, especially, again, in a closed system of Joints. Try suppressing Motion Links to see if they are the source of your problem

Add Degrees of Freedom by Changing Joint Type

As described, Joints define motion by exposing degrees of freedom along their axes. Sometimes, an easy way to "unstick" an assembly is just to add degrees of freedom. Since the most common motion type is Revolute, sometimes converting a Revolute to a Cylindrical will free up a "stuck" mechanism.

A Final Word of Thanks

Thank you for sticking with us, if you got this far. We appreciate feedback. If you have any, or just wish to say "yeah, I read this far", or have any follow-up questions, please feel free to send us an email at:

- Jeff: jeff.strater@autodesk.com
- Phil: phil.eichmiller@autodesk.com

We'd love to hear from you!