MIKE PROM: Good afternoon, everyone, welcome to the Masters of Assemblies Class. My name Mike Prom, I'm one of the Project Managers for Fusion 360.

KEVIN SCHNEIDER: I'm Kevin Schneider.

MIKE PROM: Today we're going to go through some top-down design methodologies, positioning parts, and talk about design reuse and sharing. First thing we're going to start off with this top-down design methodology. So when you think of design and your design assemblies, top-down design is layout driven. Meaning you start off with either, we'll see today, a box or a sketch, and use that geometry to drive everything else in the assembly. Skeleton is where you use the sketch or work geometry. Again, you'll go back and make changes to the initial geometry, and we'll see how everything updates.

And then we have multi-body. So while you're doing your design, there might be situations where you're not sure about what your end results are or when you need components. So you're working with bodies, and then you'll convert those to components. A lot of times situations are where you have more organic shapes, or it's more difficult to have individually and put together. That's when you do a multi-body design.

KEVIN SCHNEIDER: And one of the things we did really want to point out, spend a few minutes on the last point. If you're familiar with SolidWorks or Inventor, you may be familiar with the in-context design and in-context relationships or doing adaptivity in order to create in-context relationships. The only situation in which you can create in-context relationships or cross part relationships in Fusion today is through top-down design. It's a very explicit methodology for designing, creates very clean relationships between components, avoid some of the pitfalls with those other approaches. So we're going to highlight some of that, as we go through this today.

Most of what we're going to talk through about is the top-down approach itself. And some theory about some ways to think about it, use it in a simple example, and then we'll show you a much more complex example. And then build on top of that top-down approach as we look at other aspects of assemblies, and how to make them work for you. There's quite a bit in the assembly environment that, if you're familiar with Inventor or SolidWorks, you can sort of treat it like the product you know, and that will work for a while. And then you're going to re-hit a few things where your grounding doesn't make any sense to me. It's a common example we hear. Or I don't understand why this stuff is moving the way it is.

So once you hit that particular wall, this class is to help you understand why those things are different. And that actually you can use assemblies quite shallowly and get by with your existing knowledge. But if you reset some of your understanding how Fusion works, you can use them in a different way, and hopefully a much better way. And we hope to give you some tips on how to do that today.

MIKE PROM: Awesome. So Kevin came up with this idea of a tree. I think this is a great example of talking about being open to your design, and how do you approach A design. So if you think of a tree, your designer it's the trunk or the base of your tree. I'll go through and I'll actually demo this model. But what we see here is a box, and this box is our bounding box for our assembly. And so this is the base of the design, and we kind of have our top-down layout here.

From there, the first component that I add is a branch. Right? And so you can think, as the tree grows, as I add components. The tree has these branches, and a branch can have a branch, and so on, as the complexity of your design grows. From there, if you look at the finished model, you can see we have the trunk, we have a branch, we have another branch, and then the little circles are like our leaves. And those leaves are your placed parts. So, in this example, we can see the castors on the bottom. Those are going to be my placed parts.

And so, when you talk with design, we talk about assemblies. It's really not one methodology, it's combining everything. I think this tree example is a good way to work through the design. So I'm going to do a demo. It's simplistic, but it really does a good job of explaining how we want to approach design.

KEVIN SCHNEIDER: I don't want it to get an email.

MIKE PROM: Don't want an email. Now, just to prove how easy this is, I'm actually going to start from scratch. And I'm going to start off with just creating a sketch. Now in this case, I'm just using a box, and you can think of this being my bounding box. So just a simple rectangle. And I want to extrude it the same, so it's a cube, three feet all sides. A little trick is, when you do this, you want to change the opacity. So I'm just going to knocking it down to about 30%. It'll make it a lot easier for me, as I'm going through my design and building it up, so I can see through it. But it's still kind of recognizing what my overall base is.

Now first step is to create a component. We'll talk about this a little bit later. But when you know you're going to have a part, it's very important to start with a component. In this case, I'm actually building up a subassembly, and I'm starting with a component. And I'm going to start off this face, and I'm just going to create a sketch. From the sketch, I'm doing my structural shape around the outside, so I'm just designing in my miters.

KEVIN SCHNEIDER: As he's finishing that extrude, I'll bring this back to the tree, the very first rectangle shape. Keep going so we're not short on time. The very first The very first shape is that trunk of the tree. It's the most abstract and the largest portion of the layout. And if you think of this tree analogy, what we're trying to do now is get more and more detail as we get higher up the branches of the tree. And the pieces, we create more detail. So he's actually created a second layout here. That layout is based off of the first layout, to that point of creating these progressive structures that become more and more detailed as you get further down the detail of the design that you're creating.

The advantage here is that, because you're not limited to putting sketches only inside of parts, there is no difference between a part and an assembly, which you might be familiar with other tools. Any part can be an assembly, any assembly can be a part. Any part can have any number of bodies and sketches. Any assembly can have any number of bodies and sketches.

We can leverage that by using bodies at the root of the design, by using bodies and sketches in intermediate subassemblies of the design, as this framework upon which to put and build other components. Which is exactly what he's done here. So he has a box with a sketch on the face, an offset of the sketch, and then some solids created from that. And that entire chain is completely associative back to the beginning. So changing the bounding box is going to cause everything to rebuild, very easily. And his miter frames are all going to change size without any additional work.

MIKE PROM: As Kevin said, every time I did an extrusion, I did it as a new component. So we can see that I have my top level, and then I have a component here, and then I have my individual components underneath. Now something that we really want to emphasize, and we'll talk about a little bit later, things being flat. So I made these all in place, but I can move them. They're free to move.

Now this is probably one of my favorite tips about doing a design in place, is using as-built joints. And a lot of people don't realize that my first component, I actually wanted to make an as-built to the level one above. So you can do an as-built to a component. I'll do it again, another as-built, and this time I'm just going to do from one component to another. So look that I'm using my browser through this operation. And a lot of people also don't realize, you don't have to hit OK. One of these favorite things about Fusion is I can just select my next command, and it's like selecting OK. So very quickly, I did as-built joint's that lock this down.

But there is another step further. Right? Because the subassembly is now locked down, but if I move it, see my origin, it's not actually constrained to my origin. So simply just do another as-built joint, and I can now go from this component, to my top level, and it's now locked in place. I did that pretty quick, but it's important to realize the tip here is as-built joins, very, very powerful. And you can go from a component to a component. In this case, I'm locking it down to my top level.

KEVIN SCHNEIDER: So for all the folks who've tried to ground components, and said, I don't understand what's happening with ground. We're going to come back and actually talk about how to use ground as intended. What we just did here is how you would have used ground in Inventor or SolidWorks. And the reason why it's approached this way is the parametric rebuilder of the model will always put those components in the place in which they're sketched. Because it's just a timeline of features and it rebuilds in order. And those features and sketches are in space very deterministically by the faces that are sketched on and the sizes by which they're created.

But kinematically, they're completely free. And again we're going to touch on why they're kinematically free. The kinematic movement of components has nothing to do with a parametric definition of how they're built. Those two worlds are separate. So if we want parts to be in the place in which they're designed, we simply say add a rigid in-place joint. Which says, there is no kinematic freedom of this component. Its position is rigid in the place in which it was designed, which is what we just did here.

That one trick right there usually solves most people's, everything's moving in weird places, and I have no idea why it's doing it. It's because you're trying to use ground for a purpose to which it wasn't intended. And this by just saying as-built joint of this component to its parent, will lock its place kinematically with respect to its parents position. And I think you're going to find it's going to solve a ton of problems.

MIKE PROM: So let's take that one step further. I locked that design down to itself, and I simply just did a copy and paste. That's why you didn't see me do any commands come up. I just had Control-C and Control-V. Now how many people realize you can do joints to sketches? Hands? A few of you. OK, so let's think about that original sketch that I created. I can now come in I can use that sketch to apply joints. And you can see here that I have the centroid of that sketch pop up. I can select my face, it will lock them together. In this case, I want it to be inside that, and so I just flipped it over. And I utilized all that work I just did, and we'll see at the end, when I update, everything will update accordingly.

One last thing, just to have some fun. I'm going to create an extrusion off the top here. This is a new component, and the style all the way back to the original sketch. So just turn this on, and show my dimensions. I can change this from three feet to four feet, and everything updates. So this was a quick example of how I can just start with a box and build everything down. The same thing works when we have a larger assembly, and if I show dimensions on this and make a change again, we'll see everything else updates as well.

KEVIN SCHNEIDER: So what he's ended up with is one basic layout, sub layouts, sub in-context or top down parts. And then everything that's purchased, that doesn't change often that's a piece part, is assembled bottom up, on top of that layout structure. So it's both top-down and bottom-up design. So things that tend to vary greatly from one design to the next and are unique, those are great times to use top-down methods like this. Components that tend to don't change very frequently that you reuse all the time, you assemble bottom-up.

And that relationship between the top-down components, building the structure, and the bottom-up components being attached to it, gets you back to that original idea of think of this as a tree. The layout is all the trunk and branches, and all the placed parts hanging off that branch and tree structure are all your purchased parts. And there are a lot of designs, whether you're doing consumer products, whether you're doing industrial machinery, you can start to imagine applying this approach to those particular designs. And it gives you some guidelines on where to split between, what should I be instancing in and reusing, and which thing should I be driving top-down and designing in context.

This is the exact same model just greatly more complicated. In this case, there's a bunch of extruded frames, shafts, and belts. There's some custom brackets, and then there are hundreds of purchased parts. There's only about 30 parts in this design that are top-down, that are designed in context off of the layout. Everything else are the leaves that hang off of that layout structure. Which means that it's as simple as changing one dimension for that, it's a

1,200 part assembly at this particular point. And it takes about a half a second for that entire assembly to rebuild and change its size.

So it's a really effective approach for taking on these projects and building out a nice structure off how to approach a design problem. Any questions before we move on, and we get into more detail on the joint side of this? Yeah?

AUDIENCE: The whole active component thing, I don't understand that.

KEVIN SCHNEIDER: We will talk about that in a few, yep. Yep?

AUDIENCE: What was the negative five [INAUDIBLE]

MIKE PROM: Oh, the negative 5D, yeah. So in the order in which I made it, when I made the offset, it was negative five, or sorry, negative four. But I knew that was my fifth dimension. So my first dimension was the sketch this way. Second dimension, sketched this way, third dimension, was the extrusion. So I just have done this enough that I knew that was my fifth dimension. So I'm just recalling. So when you do extrusions, or any parameter, you can just reference an old parameter.

KEVIN SCHNEIDER: He just knows D5 is the value he wants to reuse.

AUDIENCE: It's a global variable [INAUDIBLE]

KEVIN SCHNEIDER: Global variables have to be declare beforehand, and in this case, every dimension already given a unique name. And he can just refer to that unique name without having to declare it beforehand.

AUDIENCE: Just an add on to that question, is it selectable or do you have to remember [INAUDIBLE]

KEVIN SCHNEIDER: No, it's selectable. There's quite a bit of trickery, of nice things you can do there. So we've just practiced this so much, we remembered. If you don't know which one it is, there's a parameter dialog which you can bring up, which is structured based on your features. And it'll show you all of the parameters per feature. So you can look them up. There's also some ways to declare some of those parameters as favorites. And if you declare them as favorites, they'll appear in the number entry control. So you can just pick them from a list of favorites.

There's also autocomplete. So if you know that you have like six or seven that are named outer diameter, outer height, outer width. You can just start typing O-U-T-T, and it will autocomplete and give you a list of parameters you could pick from. Yeah?

AUDIENCE: [INAUDIBLE] So I can go into that dimension. Then could I come back and put negatives, and then select D5 from that position?

KEVIN SCHNEIDER: Yes.

AUDIENCE: So I don't have to technically remember as long as I know geometrically think that's the dimension I want to use for references, [INTERPOSING VOICES] this dimension.

KEVIN SCHNEIDER: Correct, and while you're editing dimensions, any dimension that's on the screen, while you're typing in that value, you can just click on it, and it'll grab its value. You don't even have to look it up. So just say I want that value divided by that value, and you just click on it.

MIKE PROM: And you can go back at any time, and if I change-- the reason why did that is if I change it from four to three, now all of those updated, and this is all perfect squares still, because they're all referencing that same dimension. So now that we've seen this, we want to break down to bodies and components, and make sure everyone realizes some of the key areas. So rule number one.

KEVIN SCHNEIDER: Stand up, rule number one. He's responsible for rule number one. Start by creating a component. I like to say, if you know it's going to be a part, create a component. If you are just exploring your design, use bodies. Bodies are really powerful. You can copy, you can paste, you can make different iterations to them. But if you know up front it's going to be a part, make it a component. Some really big benefits to this. So joint's used components. Right? Joints don't work on bodies. Drawing use components. If you're going to do a drawing, it's broken down into components. Build materials are based off of components, and also components can be instants.

So going to another demonstration here on positioning parts. To me--

KEVIN SCHNEIDER: [INAUDIBLE] We'll just flip over

MIKE PROM: Did I skip?

KEVIN SCHNEIDER: No, it's not a demo yet.

MIKE PROM: There you go.

KEVIN SCHNEIDER: So let's talk a little bit about positioning parts and components. So there is some of hidden gems in what happens when you're creating and positioning components, which we call joints. And we're going to talk about joint specifically in a few moments. But anytime you create a joint, you're going to see this little disk floating around. The first thing to know about that disk is really under the cover of what it's describing is a co-ordinate system. It's an xyz coordinate system, and to understand how joints work.

What joints essentially do, is take two world coordinate systems and put them right on top of each other, where x matches x, y matches y, and z matches z. So conceptually behind the scenes, and you don't really need to know much more other than they have an orientation. They have a white side, which is z positive. They have a yellow side, which is z negative. They have a little divider in the middle which represents the x-axis.

And if you know that, and you know that the direction and the color of that disk has some meaning, it can really streamline your positioning of components. Because it's going to tell you, before you ever pick the two parts, you can make sure the selections you choose are going to put parts in exactly the position you want them to be, based on just the preview of those. It's also important to know all joints require two component selection. So there's component one, component two. You pick a piece of geometry off component one. It sets the joint origin for that. You pick a piece of geometry off component two, it sets the joint origin for that. And then the joint will assemble the two together.

So let's talk a bit about joints. All of the joints are modeled off of mechanical style kinematic relationships. So you have a ball joint, slider joint, cylindrical joint, a planar joint, pin slots, rigid joints, and revolute joints. And so all of them match kinematically. The types of mechanisms you would design normally as a mechanical engineer. One thing I kindly ask everybody to do, just because bolts turn in holes. Do not put revolute joints on all your bolts. You tighten them down, that means they don't really turn freely. So put rigid joints on all your balls.

The entire concept behind how can the kinematic world works, as I described, is the kinematic positioning of your designs is completely free and separate from the parametric compute of how their shape is defined. And joints are designed to give you the fastest way to remove degrees of freedom so that parts can't move. So 99% of your joints will probably be rigid, and then you'll be choosing one of these kinematic mechanism joints to open only that degree of freedom you require to model the movement of the mechanism.

Another way to think about joints, for those of you are a little more analytically-oriented, is to think about them this way. You have six degrees of freedom for any component that's free in space. And each of the joints has a very specific number of degrees of freedom, and a very specific degree of freedom that it opens. So rigid has no degrees of freedom. A revolute is a rotational degree of freedom by default around z. You can override the default axis for all of these, but just conceptually it's good to know the default. Sliders have only one degree of freedom. It can slide along z. Cylindrical has two degrees of freedom, z-axis for movement, z-axis for rotation. Pins have a rotational degree of freedom and a translation degree of freedom.

Planar is the one with the largest numbers of degrees of freedom. Planar joints are not flush or align joints and constraints. Again common mistake, somebody has two boxes, planar, face-face, planar, face-face, planar, face-face treating them just like a mate flush or mate anti-align in Inventor or SolidWorks. What you've just done is created three joints with four degrees of freedom that the kinematic solver now has to go figure out all those degrees of freedom cancel each other out. That's the fastest way to tearing your hair out.

So just use a rigid joint. And we're going to actually show you some really good tricks about how you can do all of those positioning styles with a single joint. You don't need three to do it. So just a quick little set of tips on default degrees of freedom and which axes those degrees of freedom are on. When you're using them, you're cruising around, you get all these little white dots. Those white dots have meaning as well. Each one of those little white symbols means something very specific.

If it's a plus, it's the center of an arc over a circle. If it's a triangle, it's the parametric midpoint of any edge or curve along the face. If it's a dot, it's the parametric endpoint of any one of those face edges or curves. So you should know, based on the selection of those, what kind of behavior you're going to get when you're snapping things together.

I also find a lot of people don't know that cylinders will always have three snap points. There'll be one which is parametrically the exact center of the cylinder, and there will be two that are the furthest extent of the cylinder as a true cylinder. But this will work even with cylinders that do not have planar and loops. So you can imagine putting a pin in a drive shaft. That's a single joint, that is middle of cylinder to middle of cylinder. It will parametrically put the pin dead nut center in the shaft with no tangency, no midplane, no midplane axis alignment. It's just one joint, two clicks, parametrically always centered. So nice to know cylinder's will always work, but the end caps will be the extent of the cylinder the furthest extent. Any questions here?

AUDIENCE: Is there any way to filter out some things that I never use [INAUDIBLE]

KEVIN SCHNEIDER: No, there's no way to filter out today, but one trick that I do see is that, if you get near an edge here. You'll start fighting for which, is it the face, is it the edge, or the other face. So command key on the Mac, control key on Windows, will prevent it from moving to any other input. So if you're over this edge, and you know you just want that edge. Just hold Control down, and it won't move to anything else. It will just only snap to that one entity, and it's a really good trick, exactly. Yeah, and so then it won't give you something you're not expecting.

MIKE PROM: I'll demonstrate that when I do the demo.

KEVIN SCHNEIDER: So as we just mentioned, there is a difference between a face and an edge. One of the things that I think is really, really useful is midpoints of edges will give you a snap point to any model edge parametrically tied to the midpoint of that edge. It's really great if you do a lot of symmetric positioning of components, makes things really easy to put together. And just pay attention to the face highlight. And to lock the selection to an entity, command on Mac, control on Windows. OK, so Mike's going to go through two examples of some more real world cases. Yes, quick question.

AUDIENCE: [INAUDIBLE] if you want to do a joint on a [INAUDIBLE] on an edge [INAUDIBLE]

KEVIN SCHNEIDER: So what I would probably do today is put up a work plane along edge and type in 0.75, which will give you parametrically three quarters or 0.25 if you want one quarter distance. Then you drop a work point at the intersection of that plane and edge, and then you can use that location to create a joint. You have to do a little construction geometry, but it will create exactly the condition you're looking for. And if the edge gets longer, it will always be a precise value. If you want it always to be 10 millimeters from the face, then do an offset work plane 10 mil. Do the work point on the edge. So you can do either situation, if it's a finite distance or percentage distance.

MIKE PROM: And like you saw earlier, you can do a sketch. So I reference the center of that sketch, and the center of the cube. So one other way you can control a point. Quick question. How many people, when they do modeling, think of adding joints or constraints at the end of their design? Who does that at the end? I'm seeing nods.

OK, hopefully this will blow your mind, because it completely changed the way that I thought of doing design. Kevin reference's, when I'm early on in my design, we have some simple shapes here. But think about when you do modeling, and when you're early on. You know these two will be constrained or are joined to each other. Let's do it at a point when you know certain reference points need to be referenced. So let me let me demonstrate that. It may make a little more sense. I'm going to grab the center of my top box here. And then again the center of my lower box.

KEVIN SCHNEIDER: Just restart the [INAUDIBLE] command.

MIKE PROM: All right, do this again. Maybe, there is goes. Just a little delay with the projector here. So I'm grabbing the two centers of these boxes, and I'm just going to create a rigid joint. No, it's just delaying.

KEVIN SCHNEIDER: You want me to flip over?

MIKE PROM: Yeah, let's try your computer. So basically we're going to set this up three ways. We're going to grab a center of a center, we're going to grab an edge and an edge and then we're going to grab a line and a line. And as we're doing this, we're applying one joint. So the first ones are rigid, and then the last one, with the edge, we're gong to do a revolute joint. And as Kevin is doing this, you can do things where you can hold down, like I said earlier, the command key on a Mac, or control on a PC. And this allows you to lock to an edge. So see how he's dragging and it's just snapping to the edge? That's the power of just using those modifier keys.

Now this is impressive, but I don't think anybody's had their mind blown yet. All right, so the next step is we're going to add a fillet. So we add a fillet to this. What happened to all those edges we had? They're gone. We don't lose any of those joints, because they're in our timeline. They're in history. So as Kevin modifies this part, they all keep all those relationships. So those edges are gone, modify, and they stay jointed to those edges, because we you utilize the timeline.

KEVIN SCHNEIDER: So when we go back to the top-down design and layout design, all other tools out there, the assembly environment is the place you aggregate finished parts. And then you put them together. If you make radical changes to the shape of those parts, edges come and go, face merges, face splits, relationships and constraints break. Because the assembly environment is only ever able to see the end of each parts individual feature tree.

When we take a top-down approach, we have a timeline that spans all the parts, and we can constrain in time. So you now are able to constrain components together before you add all you're finishing features. Which destroy edges that you might want to use or break when you change those fillets to a chamfer, or shell the model, or split it into multiple components. So one thing I notice is a lot of people apply constraints very, very late in the design.

It's much better, in the top-down approach, to apply constraints that define the right position of components relative to each other as early in time as you can. Because the earlier you do, the more reliable, the more robust, the base geometry is. The more flexibility you have to do to make changes later, and it's not something you would normally think you can do coming from another tool. But is inherent into how this top-down methodology works.

If you really want to go way out there on a limb, you can actually create two components. And you can say, create a joint between the origin of this component in the origin of that component. And its revolute, and you can assemble components together in time with zero geometry being modeled in them. And then you can build your complete shape, and never worry about how that shape ever changes. Those constraints will never break, because they're not tied to any of that geometry that's changing. You've defined as early as you can the relationship and it'll survive a lot of change.

And the whole point of why we want to do this is so that we can change our models a lot, so we can explore the design space better. So any little trick that makes these models rebuild more robustly and faster and lets us change shape with less fear of things breaking, that frees us up to think about making more radical change. Like we've all lived with a model that we have hours into and realized, man, I really should have gone a different path at this one point, but I don't want to remodel this. I'm just going to hack and slash it till it's done. And I'm not going to do it right. We've all been there.

The more we can free you to feel flexible to start over, or make radical change without things breaking, the more flexibility you can have in exploring your design space. Hopefully the better things you can build. So this is a great way of illustrating don't wait till the end to assemble components together. Assemble as early as you can.

MIKE PROM: All right, we're going to try my computer again. If not, we'll jump over to Kevin's. I have no geometry in the middle of these two, but I want to place a joint between these, so they're centered. Any ideas how to do that? Anyone? Charles, I know you know. Yeah, I heard it there. OK, so there's a trick. After I start my joint, I can right click and say Between Two Faces. And I can select one face, and I'm just going to hold down, so I can slide through, and select the other face.

So right now it's snapping to the middle, and I want it to actually be based off of the circle. So now I select that circle and it snaps to the center, between the circle, based off those two planes. I did this on one. You just right click again, between two faces. This time I'll just do the outside faces of my next part. Again, snapping to the center, and very quickly I have these two with a joint right in the center. So this is a little more of a hidden trick, but very, very powerful. There's no need to make extra geometry, sketches. Great example of something to go use after this class.

KEVIN SCHNEIDER: You want to do yours?

PRESENTER: Yep. So let's talk a bit about troubleshooting joints. Now that we understand basically our designs are all free. We want to remove as many degrees of freedom as we can. We use joints to do that. Those joints are reference frames that always match. they always match based on parametric selection points. They always match on selection points. What happens when things go wrong? So there's a couple little tricks to know that just make your life a little easier on working with joints when you have issues.

So one of the nice tricks that you can do is, anytime you have a model that looks like this, that's a whole lot of joints on the screen. What the hell am I going to do with this? So there's two things we can do. One, I may want those, because I'm debugging or working on assembling in a mechanism, and those joints are really useful to see. But right now they are annoying me. From the bottom, there's a set of object visibility controls. These are global visibility controls for just the session. It's not going to change anything in your model. We could just simply say, please get rid of all those joints. And now, temporarily, all that clutter on the screen is gone, while I do some other action.

I haven't actually changed the visibility state in the models. I'm just doing an application override. You can do this with sketching, work planes, joint origins, any number of things that you might find clutter the screen. That's really easy to turn things on and off. So one nice method there. If you really want them off in all of your components, let's use the Select Tools to do that. So I'm going to go to my Selection Tool, selection filters, turn off All. Just say I want joints only, and we'll just window the whole model, and press V for visibility. And we're done, and they're all gone.

And that has toggled all the visibility, in all the components, throughout all of the hierarchy, to toggle things off. And this works with anything that has visibility. So work planes, work axes, work points, your selection filters give you a nice way to do that. You have window selecting, which is from left to right anything inside the window. And if you go from right to left, anything that is inside the window or crosses the boundary is selected. This is pretty common. This is common across all Autodesk applications. Yes, you may ask a question.

AUDIENCE: How would anyone know this, except coming to your class?

PRESENTER: That is in the Help. So those are some good tricks for using your Select Tools and your Select Filters to control visibility of components of joints. So one of the other things that might happen is we have this one crankshaft right here. And perhaps we want to understand a bit more about what's happening it. So I've double clicked that face, and it's done a selection of the component. That's just a quick shortcut. Double click a face in the assembly environment gives you the component that owns that face.

From there, if I right mouse click, we could say select referencing joints. And what that will do is select any joint that references the component that I've just selected. So if I need to find anything that this is assembling, and anything that is assembled to this, that's a great way to get there. The other thing is, because joints are in time, there is some assumptions you can make about those joints and their position in the timeline. If you look in the timeline, if you want to find the first joint that's ever on that component, just look to the left and pick the first one. That's most likely the component that's positioning the part, rather than downstream parts that may be positioned to it. Because it's in time, it's not variational, it's not flat, there's meaning in the order. So you can make some assumptions based on that.

They'll also be highlighted in the timeline. They'll be highlighted in the browser. I can select any joint that I want from the browser, or from the timeline. And again, I can right mouse click, and I can say Select Components. And it will grab the two components that that joint assembles. And if there's a problem, with how things are assembled, what I probably will do next is just say Isolate. And now I'm looking at only the two parts with that one joint between them, and I can fix whatever problem I might have. So just using some selection shortcuts and visibility toggles and Isolate. This gives you some really nice tools to triage when you have any problems. So let's just un-isolate all those back.

If you have joint failures, because joints appear in order, you're going to see a series of red joints appear in the timeline. You can take the roll back marker and roll back [INAUDIBLE] in front of the last read joint. 99% of the time, I'm going to guarantee you what's going to happen is all the failures are going to go away. Because you've done some joint to over-constrain the assembly. And that last joint in order is creating an over-constrained condition. The constraint solve is going to fail. But at least now that joints are in time, and you can see them in time, you can reorder back in time, before the joint happened, see if that fixes the problem.

If it doesn't, you might have to do one more joint, you may have really screwed stuff up. But chances are 99% of the time, it's that last joint, and then that makes it much easier. You can go to that joint, you can see what kind of joint it is, what selection's did you pick. It gives you enough breadcrumbs to debug what kind of over-constrained condition you have. If you also right click on that joint that fails, you can say what's wrong, and it will tell you a fair bit of information about the constraint failure that's happened.

So when you do get in trouble, some good tricks, use the timeline to help you get there. We've talked a bit about joint visibility already, so I won't recover that. So let's look at a little bit more complex kinematic assembly, and how the flexibility, the joints, and now a much deeper component hierarchy, affects some of these principles we've talked about.

MIKE PROM: I want to talk about this real quick before you do the demonstration. So how many people realize when you do assembly that it's actually kinematically flat? Meaning when you bring in a subassembly, it's not fixed. Like an inherent CAD product, you have subassembly, and it's locked down. You might have to make it flexible. But in Fusion, people move things and they're like, why is it moving. And it's moving because it actually allows for a very valuable way to lock things down as you're assembling it together. Kevin it was a really great demonstration he's going to show that.

PRESENTER: We'll actually look at this. Product hierarchy, the subassembly hierarchy of your design makes no difference to the kinematic movement of the components. Where you're used to subassemblies introducing limits or constraints or degrees or removing degrees of freedom on subcomponents, in Fusion that is not true. If you have no joints in the design, no matter how many levels of hierarchy are in it, every component is free in space to move, completely free. It's as if, if you're familiar in SolidWorks or Inventor, it's as if every subassembly is automatically flexible without you having to turn it on. That's how all subassemblies work. And every subassembly, even if it's placed multiple times, is uniquely flexible. Yes?

AUDIENCE: [INAUDIBLE]

PRESENTER: Is an origin point for every component. And using the as-built joint, we can fix the component to the origin so that it stays relative to that position.

AUDIENCE: [INAUDIBLE]

PRESENTER: --in the assembly, yes. Absolutely. But the difference here is that you don't have to understand flexibility to get flexible behavior out of subassemblies. They're all that way automatically.

AUDIENCE: What did you do want the flexibility? I think there are a lot of times when you put in [INAUDIBLE] and then you don't want it to be flexible.

PRESENTER: If you don't want it to be flexible, all joints you apply-- and I'm going to show this in a minute-- all joints you apply on are instancing. So they'll go into the component hierarchy, in the subassemblies to which they belong, and you could do a couple of simple things. You can, if they're all rigids, if you want them to be rigid, you can set them to be rigid. If you want them to be a fixed value, you can do a rigid with an offset or you can set a rotational with a fixed degree of freedom. You could just apply that joint in that subassembly and fix the degree of freedom. And that will be true across all of the subassemblies.

Another trick you can do is you can leave the degree of freedom open, and you can apply a motion relationship between two joints of two instances of the same subassembly. So I have a hinged door, placed twice, if I say there's a 1 to 1 motion relationship between the hinged degree of freedom on the left and the hinge degree of freedom on the right. You drag any one and the other one will move in a matched relationship. So using fixed, you can lock them down. Using motion relationships, you can kinematic tie them to each other. So both are possible. Yeah?

AUDIENCE: So you have an assembly, and your having a problem in it. Let's say [INAUDIBLE] Do you drop that in and would it drop in the exact same location? [INAUDIBLE]

PRESENTER: This will come in at a fixed point. We joke, the first move is free. So when it's dropped, you get a triad for positioning. So you can actually do a position at the time of insert, if you don't want it right there. And that move is parametrically remembered, and it's part of the placement feature in the timeline. Because again, all instances occur in time, and so it parametrically deterministic. It's rebuildable, and it will always rebuild there. Yeah. And then all you would have to do is say, it's exactly where I want it. As-built joint, that component, to its root, it's never going to move. Because you're telling it, it's in the place I want it to be. So all I need to do is say, as this is built in the timeline is where I want it, fix it. Any other questions?

So let's take a look at some examples of this. What we have here is a crankshaft assembly, and we have two instances of the same crank subassembly. So crankshaft is one component, two piston and connecting rods belong to two subassemblies. So let's go ahead and position some of these components in. To your question, here's the origin of the root. So this is the origin right here of the root design. I want the crankshaft in a very specific location. So what I'm going to do is say, let's apply a joint between the center of the cylinder and the origin. And this is going to be a revolute joint. So I now used the joint to do a fix to the origin, if it came in a different position. You could approach it either way, but this is doing it if it's in the wrong place. Now it's exactly the right place.

So then the next thing I've done is, I've cheated, and I've created a set of sketch points that are exactly where I want all of the connecting rods to line up on this component. I just, for some reason, happened to know where those should all be. So what I'm going to now do is say, let's apply a joint between this cylinder and part of that sketch layout that we've created. And we'll flip that, and put it roughly in space. And you'll notice that the connecting rod didn't come along, because there's no joints relating the connecting rod and the piston to each other, even though the two are in the same subassembly. Because everything is flat, with respect to its peers. The hierarchy doesn't matter at this point. This is very different, but there are some really powerful things this is going to let us do.

So what's going to happen if I apply a constraint between the crank, the connecting rod. So let's take the center of this outer face, and we apply it to the wrist pin, and that's going to be a revolute. It's now been applied to the other instance of the subassembly, because now hierarchy matters. the joint goes into the subassembly and it's reusable. But you'll notice that the flexibility is independent between the two, as I was just saying. Because they're all free, the joints become part of the hierarchy, but the degrees of freedom are always flexible and unique.

So the mental model here is, if I had bought two of these, it wouldn't behave like it behaves in other CAD tools where I move the crankshaft and it moves in every placement. In the real world, everything is inherently free to its peers. This models the real world more than the product structure. It also means you can do things like create subassemblies that are bolt nut washers. And then you can assemble those bolt nut washers any way you want. It can represent a kit that you might package in your shop floor. And then you drop it into the assembly, and you assemble it however you want, no matter what spacing is between the [INAUDIBLE] head and the washer and the bolt. Because every kit instance is completely free to be positioned however you want.

So now we're going to be able to finish this really fast. We're just going to grab-- Let's undue, just revert, add a joint right here. And we'll put that on this cylindrical face. Here's a tie in, for those of you who understand kinematics and dynamics. There is a concept of over-constraining your degrees of freedom. This probably shouldn't be a revolute, even though it's going to be exactly right because of the sketch layout. It's probably a smart thing for me to open up the sliding degree of freedom, in case there's any movement from side to side. I've already locked the piston to the wrist pin parametrically at the middle. And the piston is already locked sliding to the sketch point.

So if I put it another revolute at the end of this connecting rod, even though it's exactly right, and it will work in this situation. Should there ever become any misalignment, the constraints are going to be over-constrained, it's going to fail. So I'm going to leave this degree of freedom open, in case that misalignment ever shows up. If you need to guarantee they're always aligned, that's a good time to put a bit of a revolute on, because when the misalignment happens, you're going to get a failure, which is telling you your design intent is violated. But in this case, I'm OK if there's a little float here because there's always a little bit of gap in a design like this. Which is exactly what we're seeing here. So that gives us a great mechanism for debugging and understanding how those constraints and those degrees of freedom are required.

What we want to do is assemble all of these into a final crankshaft that looks like this. And of course, we have four instances, and they're all totally uniquely positioned, and they're all going to move exactly as they want to. So remember, hierarchy makes no difference, everything is flat as you're working with it. All the constraints apply into the instances into which they get created, but they all have unique degrees of freedom. So those are some unique benefits of how Fusion works.

MIKE PROM: But my favorite part of this too is, imagine how much work it would do to make things flexible and then make constraints. Like in a traditional CAD package, how much work it would be to set that up would be a lot of time.

AUDIENCE: So, sorry, one more question. so if all these subassemblies are able to be [INAUDIBLE]

PRESENTER: So when you do a drawing of a particular design, the default will be, do a drawing of everything in the design that you want. The other option is you can graphically pick any one of those instances you want to be the components that go in the drawing. So if you want two views of two different positions that exist in that assembly, you could do that. There's also another really good trick, which is you can switch to the animation environment.

In that situation, they're all at some funny angle. I don't want that. I want it to be exactly straight. Go into the animation environment, transform the components in the animation environment to be how you want them to be, and make the drawing off the animation. And then you have that degree of separation from the model, to the animation, to the drawing. Which gives you a little isolation, and again ability to move some things around. So that's another nice trick.

So I want to in the last five minutes, we're not going to get to the last few bullet points on the final slide, which is about some project structuring. But this slide deck will be posted, and it's pretty self-explanatory. I didn't want to hit the question that was asked about, why and how does component activation matter. So let's grab the between, I think. It's probably a good one to work on. So here we have two components in time, and they're all designed in a top-down layout. So why activation gets really useful is any time you activate the time timeline is going to filter only the features that apply to the component that's currently activated.

So a lot of times people in SolidWorks or Inventor are used to pick a part and say Open a new tab. So they're able to isolate that component in its own window, and see only the features that belong to it. If we isolated this component and activated it. If we isolated and activated it, we essentially get the exact same effect, without having to open a new tab. So it's a really good way of slimming down and filtering to a subset of the design you want to look at.

There's another thing that goes along with that, which was on the last slide that we covered. We have a bunch of great parts here in space in a hierarchy. And you may be familiar that you can hold down Shift-n, and this will turn on 10 random colors that do a color override per component. So we now can see all the shared instances, all the random colors, everything's a little easier to differentiate from one another. But if you really want to make that useful, at the end there is this extra option here called component color swatch. You can turn that on, even with this turned off. So you just come down to the time line and say Component Color Swatch, and that will turn the color coding on in the browser and the timeline.

So some people don't like the rainbow of colors in the graphics, but it's really useful to know we have an orange component, and we have features on orange components. And we can see features on orange components are diverse spread across in time. We have groups. This group contains features that affect both orange and blue parts. This joint belongs to the root, which is the red. So you get an extra bit of context through the color coding that you don't actually have to have the whole rainbow on and can really help you decipher your designs. Particularly as these top-down designs get more complicated. It's super useful to coordinate colors between timeline features and the browser and the component hierarchy.

So we can hang around for a few minutes after for questions. I'm happy to do that. I really wanted to thank everyone for your time. As I said, the presentation and the data will get posted this evening. So you can go through a bunch of these examples yourself, and thank you again.