

# ModelBuilder-Advanced Technologies

This session will take an in-depth look at ModelBuilder and its more advanced features such as setting model properties, working with environment variables, establishing preconditions, validating and repairing models, iteration and working with lists and series.

[http://video.esri.com/watch/91/modelbuilder\\_dash\\_advanced-technologies](http://video.esri.com/watch/91/modelbuilder_dash_advanced-technologies)

---

## Video Transcription

**00:01** Hi. Well, welcome! Welcome to the ModelBuilder Advanced Techniques session.

**00:04** I had a weird thought the other day, sort of like, the User Conference, it's like...

**00:11** ...the source is pure here, like all these users; you guys are the source...

**00:15** You're the source for me, I'll tell you that.

**00:17** But I got...you might remember that, that's what they said in Tron, so I was thinking...

**00:21** ...I didn't know if anybody would remember Tron.

**00:23** But I'm riding on the bus this morning...

**00:26** And they got this graphic arts convention and they got Tron signs all over San Diego.

**00:34** I couldn't believe it.

**00:35** So, I'm happy to be here and we're going to talk a little bit about ModelBuilder and basically...

**00:42** ...we call this Advanced Techniques in ModelBuilder.

**00:45** The idea here is, say, well, what do you need to do to become proficient in building models?

**00:51** All right, think about...

**00:52** So primarily ModelBuilder, the main thing ModelBuilder is, it's a way to put tools together.

**00:58** So, the most important thing is to know how to use geoprocessing tools.

**01:03** I'm not going to talk about that in this session, but that's the most important thing.

**01:07** Know the tools for the problem that you're working on, and how to combine them.

**01:13** So, there's some pretty nice help topics in the geoprocessing help about introduction to commonly used tools...

**01:22** ...and then there's the geoprocessing model tool and script gallery that you can go to, to see samples of the way...

**01:28** ...that other people have put things together.

**01:31** So, really make sure you go through the geoprocessing with ModelBuilder help, make sure you check out...

**01:42** ...ArcGIS.com/geoprocessing and all the samples there, and also the forms there to help each other building models...

**01:50** ...and how to solve the problem that you're dealing with.

**01:51** Okay, so now, specifically, so let's say, I've got that figured out.

**01:55** I'm building...I've become proficient in the basics of models, putting tools together...

**02:00** ...basically making models more flexible.

**02:01** ...and the things I want to do.

**02:02** What else can I do?

**02:03** Well, there's some more advanced techniques, some things like what we call iteration or looping of models.

**02:10** There's a technique we call inline variables; we can make parameters a little more flexible.

**02:17** There's some special utilities that you really only use in ModelBuilder, they have...

**02:24** They don't really make sense as tools and scripting.

**02:27** They basically have the counterparts of utilities that you get in scripting languages.

**02:33** And the others control the flow.

**02:35** So those are the things we're going to talk about in this session; so how would you take your models and do...

**02:40** ...a couple more advanced structures with them in processing.

**02:45** Just make sure everybody knows where the help is; and we have the professional library.

**02:51** Within that we have the geoprocessing book, and within that we have Geoprocessing with ModelBuilder.

**02:57** All right; go through all the topics here, there's actually a section in there called Advanced Methods within Geoprocessing...

**03:08** ...within ModelBuilder, and those are the things that we'll be talking about.

**03:13** So the first thing I want to talk about is model iteration.

**03:17** So the idea here is, hey, I just want to repeat the same set of operations, over and over, on either different datasets.

**03:26** Different datasets, or different collection of features.

**03:30** So, for example, I just want to process all the feature classes in the workspace. 00:03:35

**03:41** And you see this little hexagon. 00:03:42

**03:48** You give it a workspace, essentially, so that model will run for every feature class in that workspace. 00:03:57

**04:04** They each have different utilities, or different cases in which you would use them.

**04:11** So, For is just like a fixed number of times.

**04:13** So you just say, hey, I want to run this model five times.

**04:17** Okay, you put in For, it's just like a For loop in scripting.

**04:23** While gives you the ability to run a model until a condition is true.

**04:30** So those are pretty classic, For and While.

**04:33** So the next group is feature selection, road selection, and field value.

**04:37** They basically read values out of feature classes and tables. 00:04:40

**04:45** And then the other one is what I think about are the dataset types, so the for each dataset in a workspace...

**04:51** ...so for each feature class in a workspace.

**04:54** For each raster in a workspace.

**04:57** So basically you're doing the same type of processing.

**05:00** What you want to do is a throw a workspace at the model and say, hey, every raster in here, process.

**05:05** I want to do the exact same thing to everything in that workspace.

**05:11** So those are the types of iterators we have, and I'm just going to go through a couple of the examples here...

**05:17** ...hopefully to illustrate what they look like.

**05:28** Actually, just to start out here, I'm going to just create a new toolbox, and inside the toolbox, create a new model.

**05:42** And within that, come down here to Iterators, and I pick Feature Classes, and then I just want to go down...

**05:55** ...and the main property in here is to pick the workspace that I want to use.

**06:01** In this case I want to use my Exercise 1.

**06:11** And then for each of the feature classes in Exercise 1, I want to do some kind of processing.

**06:16** So I'm going to open up another model, and I've got what I want to do in there already.

**06:23** It's just...a real simple one called Calculated Areas.

**06:28** So for every feature class in this workspace...actually I'm just going to copy it, I'm going to add a field to it...

**06:36** ...in this case I'm going to add a field called Acres, and I'm going to do a calc field.

**06:47** I just want to point out something if you're not aware of, that calc field has a really nice little feature in here.

**06:51** You can do shape-dot-area; you can do shape-dot-length.

**06:54** In fact, in here you've got shape-dot-area at acres.

**06:58** So if you have a projection, we can convert it for you.

**07:00** So that's a nice little feature in Calculate Field.

**07:04** Now the other thing is, you say hey, I'm calculating acres, I want to make sure that I'm only doing polygon feature classes.

**07:09** So, I've got my workspace, but I can also come in here and I can specify that I only want to do polygons, in this case.

**07:18** I can also put in wild cards; I could put in "only start with feature classes that start with A", or something like that.

**07:25** So the idea is, I can cycle through there.

**07:27** And also, with all these dataset iterators, there's recursive.

**07:32** So the thing about recursive is, it'll start that folder.

**07:36** It will start at... well, it really only has an impact if you deal with folders, it'll have impact with workspace...

**07:41** ...and feature datasets, also.

**07:44** For example, you pick a folder, it will find all the feature classes in that folder, and all the feature classes...

**07:49** ...in the subfolders.

**07:50** So, that could be a nice feature.

**07:52** It could also be very dangerous.

**07:54** Depending on what you're going to scan.

**07:57** It actually is possible to pick the C drive and scan your entire disk.

**08:00** It takes a while.

**08:01** It's not highly recommended.

**08:04** But, we actually did some testing that way, just to make sure it wouldn't blow up.

**08:09** So, just be careful.

**08:12** That's what really what I wanted to say was, be careful with that.

**08:17** So, I'm just going to take this and run it now.

**08:20** And what it will do is cycle through for all the polygon feature classes that are in that.

**08:26** It will copy them and output the values.

**08:29** I just happened to put them into this scratch workspace.

**08:33** For example, I had vegetation; I just copied it, and all I want to show here is that in the table now...

**08:42** ...I've got a new field called Acres with the value in it.

**08:47** The processing was simple.

**08:48** It's a simple example, in this case, whatever your model does, it will repeat it...

**08:53** ...for every feature class that is selected by, by the iterator there.

**09:07** That's one example of setting up... and I'm going to clean up my directory here a little bit.

**09:20** We also get a lot of requests for...what I want to do is, I want to read a value out of the table...

**09:26** ...and for every value in the table, I want to do something with the value.

**09:34** So, in this case, we have "iterate field values" and what it does is, you give it a table, you give it a field.

**09:41** In this case I set it, and I set it to be a unique value, so I'm going to use vegtype.

**09:46** So, all the features that have, you know, and I'm going to grab that value out and I'm going to use it.

**09:51** So, if you look at the output now, now I have this thing called value, and that's going to be vegetation type...

**09:56** ...each time the model runs.

**09:58** What am I going to do with value?

**10:01** Well, in this case, I'm just going to use it in a Select expression.

**10:05** So I got in here vegtype equals value.

**10:09** And you might notice in here, I'm going to go a little bit...

**10:13** I just want to point out, you see this percent value percent?

**10:18** So that, this is an expression, a select expression.

**10:21** But I want to use, if you think about it, what I want to do is I want to take whatever's in this

variable...

**10:28** ...and I want to make it part of this expression.

**10:30** And I don't want to rewrite the entire expression.

**10:33** I don't want = vegtype equals will always be the same.

**10:37** So, we have a convention; we call it inline variables.

**10:40** And what you do is you say percent, the variable name, percent.

**10:44** And at runtime that would be substituted.

**10:46** So, what's happening here in this model, is each time it runs it extracts value.

**10:52** It does a select statement based on the value.

**10:54** And, in fact I'm going to collect those values up.

**10:57** So let's just run that.

**11:02** So I really actually think of this, it's almost like split by attribute.

**11:05** So now what I've, if I go look at this, I've generated these, in this case these six feature classes.

**11:15** I can add them to the map real fast if I wanted to, but basically what it did is just split by attribute.

**11:20** And in this case, I did a Select statement, and I generated a different feature class for each of the outputs in that case.

**11:29** But the main thing to realize is, hey, you can use a table to draw a model.

**11:34** It grabs the values out.

**11:36** But one thing I actually point out here too is that, one of the things we have on here is data type.

**11:42** So, the data type is actually not the data type of the field; it's the data type of what you're going to use it for.

**11:50** Now in this case I set it to string because I was just going to use it as a string in this Select expression.

**11:56** It could be anything.

**11:58** If you look under data type, it's any of the geoprocessing data types.

**12:02** So, we take that string and convert it into whatever you say to convert it to.

**12:08** So, we can take that string.

**12:10** Imagine if it was a table of path names to feature classes.

**12:16** We could do that.

**12:18** If it was a linear unit.

**12:21** So it doesn't have to be just strings and numbers.

**12:24** It can be...that text can be converted into any kind of value that you want to convert it into.

**12:35** So that was...I sort of used that as a segue; I said it was going to split by attribute...

**12:40** ...so we actually started here.

**12:44** That was really the number one request we got, which was, I want to be able to run a model based on values in a table.

**12:50** I said, great. Well, what are you going to do with those values once you use them?

**12:54** Oh, I'm going to do a Select expression to select features.

**12:57** So we basically said, well, why don't we just give you an iterator to select over features?

**13:01** We don't want to give you an iterator...grabbing values over tables is nice, but let's just go...

**13:07** ...straight to the feature and be able to do that also.

**13:10** So in this case, this one's called Feature Iterator.

**13:16** It inputs the feature class or the layer feature class that you have available to it.

**13:23** Again, you pick a field...you can leave Field blank.

**13:26** If you leave Field blank, it will do every row one by one.

**13:28** Essentially, that's like clicking the OID field.

**13:31** That's the Object\_ID field.

**13:33** Or you can pick a class field to group things by, so everything that has the same value will become a group.

**13:42** Essentially, it's like calling Make feature layer with a Where clause.

**13:45** In fact, that's exactly what it does under the hood.

**13:47** It creates a layer of those features, now you do whatever you want with those features.

**13:53** You can enter multiple fields here.

**13:56** So if you want to classify by multiple fields you can do that.

**14:00** Also note, with all the ones that with field values, you've got to figure out what to do with Null; these damn null values.

**14:09** Do you skip them or do you use them when you pull values out?

**14:13** Sometimes you might want to say, skip the null values.

**14:17** Other times we'll just convert them to zeroes or blanks, if we run into, if you don't skip them.

**14:24** So, this one, if I run this, it cycles through each of the features.

**14:31** It copies them out.

**14:33** I really did the same thing that the other model did, but in this case I didn't have to do the...

**14:37** ...Select statement, I just output all the values, and now I have each of the individual ones separated out...

**14:44** ...if that's what you want to do in those cases.

**14:54** I'm going to switch over to another example.

**15:04** So this is a case there's actually, I pulled it off of, a question off the forum.

**15:09** And the question was, well, how do I make a viewshed for every point?

**15:15** Now, viewshed will take multiple points.

**15:17** So I don't want to say, viewshed will take multiple points and for a given area it will tell you how many points can see it.

**15:23** But it doesn't tell you which point can see it.

**15:26** Now there is another tool, and I forget the name of it, I'm sorry, I'd have to go look it up, that will take points...

**15:31** ...and tell you which point sees it, but it has a limit of the number of points.

**15:35** If I was only doing three points that's what I would use, but for the purposes of this demo I only wanted to do three points.

**15:42** So the idea is, I want to be able to take a point feature class, and then for every point in that point...

**15:47** ...feature class I want to make a viewshed, okay?

**15:51** So, if you look at Generate Viewsheds, it comes in here and...let me make this a little bit larger so you see it better...

**16:07** ...so I just start with Iterate Feature Selection, I start with a feature class, and I'm going to...

**16:13** ...in fact, in this case I leave everything blank.

**16:16** I'm using ID because I want to put the ID value after each one.

**16:20** And I run viewshed based on that.

**16:23** And I do the basic things I do with viewshed; I run the majority filter and I collect the values out.

**16:28** Now one thing I do want to show is that when I do the majority filter, if you look at the path name here...

**16:33** ...again, hey look at that, percent value percent.

**16:37** So, the output of Iterate Feature Class has the feature class itself.

**16:44** So anything that takes a features class as input, you can use that as input to any other tool.

**16:48** What we've found was real useful is a lot of times, in particular, I wanted to use that value field as the name of...

**16:55** ...to append it to the name of an output.

**16:58** So that's why we have this little appendage on there, a value.

**17:01** So, mostly you do the processing on the feature class, but this value variable becomes useful to use for naming things.

**17:11** For things that you name as output.

**17:15** So this model cycles through each of the features, and it creates a...and it does a majority filter on there.

**17:23** Now what I want to do is I want to take all those, and I want to merge them into one.

**17:28** So I have each of the individual ones.

**17:32** So I have the Generate Merge set, and I have the combined one, so that's actually a separate model.

**17:40** So one thing to think about, one of the key points about iterators is, one iterator per a model, the entire model runs.

**17:49** Okay, not just what's connected to the iterator; it basically treats the model as a Do loop.

**17:55** So you have to partition things.

**17:57** So if I'm looking at this, well, I'm going to generate the viewsheds, I want that to run, so everything...

**18:04** ...in this is going to run through each feature.

**18:07** And then when I'm done, I want to call Combine.

**18:11** So I needed a method to take the results of this, more than one output, and send them to the Combine tool.

**18:19** So there's a...one of the model-only tools that we have here, look at this, called Collect Values.

**18:27** And the way you find that is you go to Model Only Tools, Collect Values, and you connect this output to it...

**18:35** ...and what that does is, as the model is running, it will collect all the, in this case, all the

rasters that were...

**18:40** ...generated by that.

**18:42** So I'm going to take this now, and I'm going to run the main model.

**18:50** So now it runs the submodel that has the iterator; it does three viewsheds, and it combines them together...

**18:56** ...and now I'm just going to pop them up on the screen, and then you get those three together.

**19:02** And if I open the attribute on this, what Combine does, it gives me Mview1, Mview2, and Mview3...

**19:08** ...so I get 0,0,0, so in each of the case I can say, oh, that was seen by point 2 and point 3.

**19:14** I get all the different combinations of viewing in that case.

**19:22** So the main point there is, hey, I've got a main model that does some processing that only runs once.

**19:27** I've got a submodel that has an iterator in it that does processing, and I collect those values together and pass it on out.

**19:37** And you can do the exact same thing with features.

**19:43** So, in this case I did Combine; here I have the outputs are, it's basically the same model, the only difference is...

**19:52** ...I do a little polygon to raster to polygon here to convert the values, and then take those values and merge them all together.

**20:04** So now I'm actually going to just run this as a tool here, it does the processing, merges them all together...

**20:14** ...in that case it just happens to be a feature class of the output that you want.

**20:30** Okay. So I want to stop there, and open it for questions.

**20:34** So I just want to make sure we got through the basics there of, you can add the iterator to model...

**20:40** ...it runs the entire model a number of times.

**20:43** And there's a couple of different ones that basically, on what type of data you have...

**20:48** ...you have different kinds of iterators.

**20:50** So, are there any particular questions right now about that, right here?

**20:55** [audience participation - inaudible]

**21:01** Can you iterate by excluding a feature?

**21:11** No, the iterators themselves don't have that built in.

**21:14** You could do an additional Select by Attribute that would build that in to do that.

**21:21** But, no we don't have an exclusion list, I'm sorry. Good question.

**21:25** [audience participation - inaudible]

**21:38** The question is those values, so those values that come out of the iterators, can I use them as extent?

**21:44** Well, if it's an extent, yes you can do that.

**21:46** So, those outputs, if you go, whether there's any of these...yeah, any of these variables now are available...

**22:01** ...to be used as inputs to other tools.

**22:04** There are data-type issues there.

**22:06** So, you can always use a geodataset for extent.

**22:11** [audience participation - inaudible]

**22:23** Absolutely. If you were doing feature selection, that's something you could use as a mask.

**22:28** If you were doing iterate raster, that's something you could use as a mask, exactly.

**22:33** [audience participation - inaudible]

**22:38** No, you just connect it to the tool.

**22:40** So, let's say for example, I wanted to, so in Generate Viewsheds, if I connect this to the viewshed tool...

**22:51** ...one of the environment settings is extent or mask.

**22:54** Okay? So boom. It's connected and I'm using it for extent, in that case

**23:02** So you don't, I'll talk a little bit more about the inline variables, is that, while you bring it up, is it's...

**23:09** ...oh, I can use percent value percent everywhere.

**23:12** There's only a handful of places where you really need to use it.

**23:16** It's really useful in expressions, because it's not the entire value.

**23:21** If it's just the entire value, connect it.

**23:24** So use that.

**23:26** Question here?

**23:27** [audience participation - inaudible]

**23:35** All right, so the question is, to get iterators it runs through; is there a way to nest them, you nest them with submodels.

**23:41** So, we don't allow multiple iterators per a model, so if I come in here and I go to iterators...

**23:46** ...you'll see it's disabled.

**23:48** The rules about how they combine get really complicated.

**23:51** But, what I can do, basically I can make another model, and I put an iterator in there, and then what I would do is...

**23:59** ...I would pass, say pass this as a parameter to the submodel, right, to do that.

**24:09** [audience question - inaudible]

**24:17** Okay. So the question is, how do you get a model into a submodel?

**24:21** Let's say that if I...So I'm going to create a new model here, and I can just say, hey, I got this one Generate Viewsheds...

**24:32** ...I just drag that in there, like that, and in case, what I want to do is I want to combine it with the Combine tool.

**24:42** And I'm still kind of old school, I still go find it in the tree view.

**24:52** And then you find the Combine tool and you put it in there.

**24:55** So just any model can be included in another model.

**24:58** We actually do, but there are checks, if I went to Generate Viewsheds, and I try to put...

**25:06** ...Generate Viewsheds in it, it says you can't put a model in itself.

**25:10** Hopefully, I'll get the right.... Yeah, there we go.

**25:13** So, it doesn't call itself all the time when it does that.

**25:17** That's really important.

**25:18** You know, models can call other models, you've got to...I'm going to talk about that later on about how you set up models...

**25:24** ...to be tools, what parameters you expose to do that.

**25:27** Same thing with scripts.

**25:29** Any other model, any other script tool, all the system tools can be included in your models.

**25:42** So a couple of key points.

**25:43** The questions, I think, covered most of them, but one of the model runs for every iteration...

**25:51** ...only include things you want to run.

**25:54** Every time you use submodels to separate things out; a lot of times I have a main model, it does some preprocessing...

**26:00** ...it runs the submodel that has the iterator, and then does things with the results.

**26:06** A model may only have one iterator, so you've got to use submodels to combine them.

**26:11** Collect Values is a real interesting tool.

**26:13** It will run the model, and then for everything that gets generated, it will keep a list of those values...

**26:19** ...and then you can use that to pass, to other tools or to other models.

**26:28** Use the name and value variables, inline variables; I'll show some more examples of that a little bit later on.

**26:35** So the dataset areas are recursive; remember, be careful about that; it's pretty cool, but ...

**26:40** ...it will just keep running unless you press Cancel.

**26:46** It does, actually. Pretty nice, huh?

**26:51** It checks at every workspace.

**26:53** So, it goes out, makes a giant list so, at least you know if you press it halfway through it will stop.

**27:00** It really aggravated me when it didn't, so I put it in there.

**27:04** I figured it'd aggravate you, too.

**27:08** So, feature, row, and value, goes by each, so you can do each row, and you can also group them by fields.

**27:18** So there's another tool called Iterate Multivalue. And most of the iterators are data...

**27:24** ...what I would call data driven.

**27:26** You point them at a dataset, you point them at a workspace, and they just read it.

**27:30** That's really what you mostly want to do, but sometimes you want to pick things.

**27:34** I want to pick a list of things.

**27:36** We have this Iterate Multivalue, you create a variable and the data type you want, and you use that.

**27:41** And I also make a comment, basically this replaces, if you use... we've had iteration in models...

**27:48** ...since 9.2 and 9.3, and basically there was a simple tab, it was For and While, you could give it the...

[27:55](#) ...number of runs, and you had these things called List and Series, and we basically found that the use of...

[28:00](#) ...these was cumbersome, let's just say that.

[28:04](#) And so we tried it.

[28:06](#) We learned some things and we came back and we worked on iterators, hopefully to make these things easier to do.

[28:12](#) So really, Iterate Multivalue replaces List and Series.

[28:15](#) List and Series will still work if you have models that have that in there.

[28:19](#) But probably if you're building new ones, I really recommend that....

[28:22](#) you go to using the iterators, they have an easier structure to deal with.

[28:30](#) So we talked about inline variable substitution.

[28:31](#) A really good example is inside of an expression, so it's really a way...

[28:38](#) If I look at a parameter, and I see that the entire parameter is the inline variable, well...

[28:44](#) ...there should be no reason for me to use inline variable.

[28:47](#) Inline variable should only part of the parameter, part of the test.

[28:51](#) In this case we look at it; all it is, is the value that is there.

[28:56](#) So, primarily use a string or path parameters are what you use it in.

[29:02](#) It's keywords, so what are the keywords that you can use?

[29:06](#) You can use any variable name that's in the model.

[29:09](#) You can use environment setting, so you can use scratch workspace, which is actually a real common one...

[29:17](#) ...to use. You could use cell size.

[29:18](#) You could use extent, if it was appropriate to. In that case...

[29:23](#) ...there's a built-in keyword, which is percent n percent n, which is the iteration number.

[29:27](#) One, two, three, four, that you can always use inside expressions.

[29:34](#) So it's Select expression.

[29:35](#) So, as I put a big capital...

[29:40](#) Make sure you include quotes!

[29:43](#) So, look at this expression.

**29:44** This guy, it says, "airport ID equals," you know, whatever expression.

**29:49** So, what I recommend is that you build the expression first and see what it looks like.

**29:54** And then the piece you want as substitute, type that in there.

**29:58** In this case, airport ID.

**30:01** The substitution doesn't know when or how to quote or not quote things.

**30:06** You actually have to interject that.

**30:08** So if airport ID was a number, I wouldn't quote it.

**30:11** If airport ID is a string, then I do need to quote it.

**30:14** So you have to have some knowledge about how it's being used for the expressions you're building.

**30:21** Calculate Field is a good place to use it.

**30:23** Path names to output datasets are a great place to make, this is in particular scratch workspace...

**30:31** ...name of a feature class, is a great way to make things portable.

**30:35** So whatever the crash workspace is, this model will use it.

**30:42** So one of the things that we saw a few in some of the examples I did...

**30:47** ...I didn't just do iterators, I've got some of these other tools that we were working with, so...

**30:52** ...for example, Collect values to do that.

**30:55** So, there are these things we call model only tools; sort of the best name we came up from; basically they are tools...

**31:02** ...that only make sense to use in models.

**31:03** I don't need to do things like, I don't need to do Calculate value in scripting...

**31:09** ...because I can do just do simple equation in scripting.

**31:11** So the idea was, hey there's some things I can do in scripting, let's work out the corollary that...

**31:16** ...I can make it real easy to do inside of a model.

**31:22** So one of them is Calculate Value.

**31:23** It's just a simple calculator, so if I want to do a simple calc...it's really like Calculate field but just for one expression.

**31:32** Not for every feature in a table or every row in a table.

**31:35** So you could put any Python expression in here.

**31:39** You can also put a code block in there, so it's actually kind of a tricky little way to embed some Python in your model.

**31:46** So, for simple things that are constant, this is a good way to do that.

**31:50** This is actually one from an example I'll show later on, but what it does, is it actually uses ArcPy to get the install directory...

**32:02** ...and from the install directory it gets a style sheet that pulls the value, and notice the data type on the bottom.

**32:08** So mostly you're just doing simple expressions, numbers and Booleans but sometimes...

**32:11** ...you do more complex things and you can do that within Calculate value.

**32:19** There's one called select data, and the reason why...the primary use of it is, well, the only use of it is...

**32:24** ...to pull value out of things, so...

**32:27** ...if you're familiar with Network Analyst, Network Analyst has a tool called Solve.

**32:32** So you have your network, and you enter various locations in it and you call Solve.

**32:37** And Solve itself is a Network Analysis layer.

**32:40** It's a layer that has lots of sublayers.

**32:42** Well, Intersect doesn't intersect Network Analysis layer, Intersect intersects feature classes.

**32:47** One of the sublayers is a feature class, which happens to be the route or the service area that you use.

**32:53** So in order to get to that we have this thing called Select Data.

**32:56** So, really just think about, it grabs a subdataset out of something that's an intermediate data...

**33:02** ...inside of your model.

**33:07** Collect values, we saw that, we use it mainly with iterators, we run the iterator, you make it as the output, it collects all of it...

**33:15** ...it will make a multivalued of all the values that gets generated when that model is run.

**33:21** It's also kind of a handy little deal that, you can take two multivalued and combine them together into one with it.

**33:28** So, the idea is that, basically, like if you're connecting to merge and I've got two lists, two multivalued...

**33:36** ...and I want to send it, well merge will only let you connect one.

**33:38** It says, hey, I let you connect multivalued, I let you connect lots of single things, I don't let you mix and match.

**33:45** That was the rules that we made for that.

**33:55** Some of you might have used lists in 9.3, and a lot of times you generate lists and that was nice...

**34:00** ...and then you want that list to go into merge, and merge says well, it's going to run more than once...

**34:05** ...and this is actually a way to turn that list into a multivalued for merge.

**34:10** Get field value just grabs the value out of a table.

**34:13** So, this is real interesting. You run sum of statistics, you get the sum of something and you want to use that number.

**34:19** Hey, just grab it. Give you the table and the field.

**34:22** It grabs it out of the first row.

**34:24** It doesn't walk all the rows.

**34:26** We have the iterator, Iterate Field Value that does all the rows.

**34:29** If you just want to grab one real fast out of there, use that tool.

**34:33** And merge branch; I'll talk about that later on about how you can use that with preconditions...

**34:39** ...do some of the same processing in a model.

**34:42** There's another one called parse path; you can give it any string and it will do the classic get the directory from it...

**34:49** ...get the name from it, get the file from it, get the extension from it, if you ever wanted to do those things.

**34:54** A lot of times it's really useful.

**34:57** You get an input path, you grab the name off of it, and you use it to name outputs in another directory to do that.

**35:04** And then there's a tool called Stop. And really Stop, it's like While, and I'll do a demo of While a little bit later on...

**35:12** ... the model will run.

**35:14** But the idea is, iterators, you can't combine them, so I can't do Iterate features and put a While in it.

**35:21** Those were our rules; we can't put more than one iterator in it.

**35:23** But, what I might want to do is, I want to iterate the first 10 features...

**35:26** ...or I want to iterate the first 10 feature classes.

**35:29** How could I stop the model?

**35:31** So we added this tool called Stop so you can go in there.

**35:34** Basically, it runs the model until...the iterator will keep running until Stop becomes false.

**35:40** That you can build that into the model.

**35:50** So let's go to some other processing examples here.

**36:00** Okay, so I just want to do something a little more intensive...

**36:05** ...than the ones before.

**36:06** We actually had this problem come to us.

**36:10** And the idea was, if you look at this, for each one of those, I want to do the buffer of the center point of polygons.

**36:16** In this case, I just grab the ZIP Codes of California.

**36:20** And you think okay, but I want the buffers to stop at the boundary of the polygons.

**36:26** And, how would you do that?

**36:29** So I made a model that does that, called a center buffer.

**36:38** Now just to real quickly look through what this does.

**36:41** What it does is it takes the polygons and it gets the points out of it, gets the centroid...

**36:50** ...or, actually in this case, somewhere in, as close to the centroid as possible of it.

**36:57** And then we just do three buffers.

**36:58** Okay, great, I've got three buffers, I could run multiring buffer on that.

**37:05** But what I really want is, I want the annaluses.

**37:07** If I run multiring buffer and I dissolve, well, then, the buffers all dissolve into each other...

**37:12** ...and that's not what you want in this case.

**37:15** Because they could overlap in the boundary.

**37:18** So, the way this works is, you buffer each of them, and then you identity back to the original ones.

**37:27** And without showing the individual pieces, what that does is that gives you...

**37:31** ...so now I know for each of the buffers I get, I know it's SourceID, and I know what it

overlaps...

**37:36** ...and I need to make sure that the ring, basically the idea here is that my FID, the FID of the ZIP Codes are equal to...

**37:46** ...the original FID, and I also put in here buffer distance equal to zero.

**37:50** So that gets rid of a lot of chaff that you don't want after you do the identities, because you're going to get...

**37:56** ...a lot more polygons than you want after that.

**38:01** You can dissolve those, and then there's symmetric difference, how do you take rings and make...

**38:06** ...annuluses when you do symmetric difference of.

**38:09** Like you take the two outer rings, and you run it, and you get annuluses out of that.

**38:14** And then you merge them all, all together like that.

**38:16** Okay. So that was actually kind of a little, an interesting model and it gives you this result.

**38:23** And if you run it on about a couple hundred features, it works.

**38:28** But if you try to run it on the state of California, it really bogs down.

**38:36** And what happens is that it bogs down in these dense areas.

**38:40** And this is actually one of the classic problems of overlay of identity, is, are these really dense overlapping circles.

**38:48** They generate lots and lots...if you've ever run into the problem is your bane of overlay...

**38:53** ...it's just one of the things that causes really intensive overlays.

**38:58** So, the idea here is to say, well, how do we scale this up?

**39:00** How can we scale this model to be able to run for all the features in California, or all the features in the United States?

**39:13** So one way to actually...let me back up.

**39:18** Go back to this area here.

**39:21** So I actually made a little model, so you can think about...

**39:26** Well, let's do it for every feature.

**39:29** For each feature, I'm going to...you could think of it just like if I was writing a cursor.

**39:35** If you're a programmer, you relate to that.

**39:38** Just basically think about this. For every feature I would get the point.

**39:44** I would buffer the point three times.

**39:47** I would use symmetric difference to get the annuluses.

**39:51** And then I would put them to the outer area and merge them back together and append it.

**39:59** So the idea is hey, I could take this model and I'm going to run it for every feature.

**40:05** And I've got this outer model, that what it does is, it creates the feature class, it adds the fields I want onto it...

**40:16** ...and then it passes it into this inner model, and if you look at the inner model, it has a parameter on it...

**40:25** ...which is the target, down here, which is what things will be appended to.

**40:31** So, for each of the features it will create the three annuluses and will clip them to the area and append it to that.

**40:38** The other thing I want to point out to this is to make this fast, or as fast as it can be, I use what's called...

**40:47** ...in-memory feature, in-memory workspace.

**40:51** So it's a built-in workspace that we built into geoprocessing, so that will go to in-memory feature class.

**40:59** So, in this case it's good because I know it's small and I'm going to overwrite it a bunch of times.

**41:04** You've got to be a little careful about in-memory workspaces, but if I know that my data is going to be manageable...

**41:09** ...I can use that, and in this case I'm going to be running it over and over again.

**41:13** It's really nice; hey, it's in memory, and it's faster.

**41:16** And I don't want to save it every time.

**41:17** It's just intermediate, I make it, I use it, and then it goes away to do that.

**41:24** In fact, I did that for all of the outputs here.

**41:29** So, really, all of these, it's just going to be a one-feature feature class, each of these are a little in-memory...

**41:35** ...basically a one-memory record set, and then we append those values back to the original one.

**41:44** And then in fact actually what I did on the Create feature class I also did to in-memory, when you do that.

**41:51** So, I can see the value that's there.

**41:54** So I created a feature class, at the very end I just do Copy features to a shapefile to save it.

**42:01** So now, for the purposes of the demo, I'm not going to pick all of California because it takes too long.

**42:08** But I do want to show that you can take this, you can run it, and then for each of the features, it cycles through...

**42:17** ...and it takes about, this one's actually taking about 1/8 of a second.

**42:22** Not an 1/8 of a second; about 8/10 of a second per feature.

**42:26** Which if you expand it out to California, it would take about 20 minutes.

**42:31** It would take longer than I want it to take.

**42:33** But it will cycle through, so you can just give it the features, give it complex processing, cycle through...

**42:40** ...and append the values together.

**42:48** I always get worried that I didn't use the select set. But...

**42:57** Because it is running a long time.

**42:58** I might have to press, oh, no cancel button needed.

**43:00** All right, great.

**43:02** So it only did it for the selected features, and output those values.

**43:07** But I've already determined that, well I was thinking about, like, ok, well, so I can do that...

**43:15** ...and it will work for everything.

**43:19** And if you look at the original model works if the features don't, aren't close to each other.

**43:26** Because, if I don't have lots of overlaps with the identities and the buffers, so how could I partition my data in such a way...

**43:33** ...that it spreads the features out?

**43:39** Now, we've got a tool, so actually if you look at, here, there's a tool called Sort.

**43:47** You can sort by shape.

**43:50** So it takes all the features and it sorts them from, in this case, upper left to lower right.

**43:56** Okay, so that's what I want.

**44:02** So in this case I made 10 groups.

**44:04** And the way I did that, I did a sort, and then I grouped features in calc field, I just did mod of the FID.

**44:14** So basically I'm coming here, I'm going to do 10, so I'm going to take the FIDs, I'm going to make them into groups of 10.

**44:19** You could also do a random number.

**44:21** Random numbers are working in this case.

**44:23** You don't have to sort it if you do that.

**44:26** So now, I've got an attribute on my features with a group, and for each of those groups I want to run my model.

**44:33** So if I look at, I've got a model here called Main; it does the same thing.

**44:39** What it does it creates, it does Create feature class, adds a couple fields that I want, and then it does Iterate groups...

**44:50** ...and then for each of the groups, in this case, it was iterate features, I'm using my group field.

**44:59** And then for each of those groups I'm going to call Center Buffer.

**45:02** It'll output the value and it will append it to the target.

**45:09** So if I go back to, in this case actually I can select a few more features.

**45:18** Go to my main...I'm going to double-check that I'm using the layer I actually...and run that.

**45:35** So in this case what it's doing is, I figured out 10 groups was the best number for all of the state of California.

**45:43** If I was using a smaller number, I'd probably use a smaller number of groups.

**45:47** Or actually might even succeed with all of those features.

**45:50** So the idea there is, I wanted to basically think about, how can I take... I've got a problem that ...

**45:56** ...even if my original model didn't have to tile and create...in order to process all the different overlaps that are created...

**46:08** ...for those results...let me turn those off...how can I...if I break them up it will actually run faster to do that.

**46:17** And in fact that will actually execute in about three minutes for the entire state of California, and actually I think...

**46:24** ...it's going to scale out.

**46:25** I haven't done it for the United States yet, but I see no reason why it's not going to scale out.

**46:29** Think about it, I just wanted to share some, it's a little complicated iteration, and it's also ways, well...

**46:38** ...one of the reasons to do iteration is to partition out your data.

**46:42** So, it's to do some, to break it up into pieces so that you have units that you want to do some processing on to do that.

**46:56** All right.

**46:58** Is there any particular questions on that one, that relate to that particularly in...before I move on.

**47:04** [inaudible audience question]

**47:10** The question is, will that particular one work for millions of...

**47:12** Actually, I think it will work for millions.

**47:15** You got to figure out what other groups are, and how do you want to group them up.

**47:18** It's going to take a while to run.

**47:20** Is it going to do it...

**47:24** You'd have to experiment with the pieces, and what makes sense of groups of features to run on.

**47:31** Right here?

**47:32** (inaudible audience participation)

**47:41** Right. Okay, so the question about the memory is, yeah actually it's the PC... yeah it's just a PC memory...

**47:50** ...but you know, well, what's happening, the DLLs of all the programs you are loading there; ArcMap has a bunch of...

**47:56** If I go in, look at Task Manager and look for ArcMap in here it's going to tell me what the memory usage is.

**48:00** You have to make sure that you don't exceed...

**48:08** Well, if you do in-memory workspace, it's going to go into that memory, okay?

**48:11** So that's what it's using.

**48:15** We don't have checks in there that prevent the crash in the system.

**48:22** You put these things in there.

**48:24** It only works with feature classes and for tables.

**48:27** We don't have in-memory raster at the moment.

**48:32** But it works for small things.

**48:33** It's a really good way to make small things go fast.

[48:35](#) [inaudible audience question]

[48:38](#) Yeah, so the question is, is it available in 9.3.1? Yes, it is.

[48:41](#) [inaudible audience participation]

[48:46](#) The question is, if you export out to a Python script, it will still work.

[48:49](#) In-memory work, no, we do not export iterators to Python.

[48:54](#) What we do is we export one iteration.

[48:59](#) So, it will export as if it were a single model, of one runtime, and then you'd have to go to the model and use...

[49:07](#) ...the various For loops and listing to do that yourself.

[49:11](#) We sort of drew the line there.

[49:13](#) We didn't sort of. We drew the line there of, ModelBuilder being a script tool generator.

[49:19](#) Okay, So. ModelBuilder is for people who don't do scripting to do work, ...

[49:24](#) ...and we want to help people generate scripts.

[49:26](#) But for more complex things like that, you've got to learn scripting to do that.

[49:35](#) [inaudible audience participation]

[49:42](#) The question is, how do you automate your model?

[49:45](#) You actually don't have to convert it to a script to automate it.

[49:49](#) But you can write a Python script that calls your model.

[49:52](#) All right, so it's a one-line script.

[49:54](#) So, a lot of times I see that. That's actually a good question.

[49:57](#) Well, to automate it, number one, is you do need to make it into Python executable.

[50:01](#) Actually, if you go to the help and you search for Schedule Tool, you'll find it.

[50:07](#) We show you how to take a Python script and schedule it.

[50:12](#) And the best way to do that for, and you could just make a one-line Python script that calls your tool.

[50:18](#) In fact, that's what I recommend that you do.

[50:21](#) Okay, so I'm going to move on to the next presentation...the next sample...

[50:30](#) In this case, I want to do While, so I want to do a conditional, a conditional model.

[50:36](#) And I've got this one...here the idea is, given...so I'm just going to come in here and I'm going

to select one feature...

**50:46** ...and I want to run this...what this model does is, let's just run it real fast.

**50:53** What it does is it selects neighbors until the number of the population exceeds some maximum value.

**51:01** So, I got the input feature class, then I do...select by...I do Select By Location.

**51:11** If you look at Select By Location, you say input features and you say "share a line with," that's basically...

**51:18** ...select on itself.

**51:20** So it starts with the polygons, and it selects all the polygons that touch them.

**51:26** And I want that to grow out, in this case.

**51:28** And I'm also, I'm adding to the selection each time.

**51:32** So the next thing is I want to know, what is the population?

**51:35** So I just do Summary Statistics in that case, and I'm summing up the population value.

**51:42** So now I've got a number in a table.

**51:43** Remember that Model Only tool that I told you about, called Get Field Value.

**51:49** I just give it the table.

**51:51** I give it the field I want to get; and now I've got a simple variable that has that number in it in the model.

**51:56** And since I ran it the last time, it was, it was 26,000.

**52:01** So I want this to run until this number is greater than that number.

**52:06** Well, how do I do that?

**52:08** Right, so in here I have 10,000, and I can use Calculate Value.

**52:12** And that Model Only Tool.

**52:14** It's a little utility tool...I call it calculate value...it's like the Swiss Army knife of ModelBuilder.

**52:21** You can do lots of interesting things with it.

**52:23** In this case, all I'm doing is "sum is less than maximum."

**52:28** A real simple equation, and I want the output to be a Boolean.

**52:31** True or False.

**52:34** So if sum is less, it's true, you keep going; if it's greater than, it's false, you stop.

**52:40** And so now here's the iterator While, it takes the input, this output value...

**52:47** ...I probably should have given it a better name, but it takes as input what gets generated there.

**52:51** So as long as this is true, the model will keep running.

**52:54** Once it's false the model will stop. That's the While in there.

**52:59** And then I have calc value.

**53:00** I just put in there  $n+1$ , so I just give you the number of iterations that that model ran.

**53:08** So then for each of...if I select a polygon and I run that, it expands it out.

**53:20** So actually I'm going to come back to this a little bit later and do some things.

**53:23** I'm going to make this into a tool and say, hey, well, what if I want to give it a point and expand that out?

**53:29** But I'm going to stop here and talk about a couple of concepts first.

**53:43** Okay. So now I want to sort of jump to, so now I've got... I've made my model and I want to be able to use it..

**53:51** ...in another model, or I want to call it from a script.

**53:55** I want to basically make that [unintelligible] encapsulate some interesting processing.

**53:59** I don't want to copy/paste it every time.

**54:00** I don't want to copy/paste 10, 20, 30 things into another model when I do the exact same thing.

**54:07** So how can I make that a unit of processing that I can use in other tools.

**54:13** So the idea is, you can take var...what you do is you take variables, and in fact, you noticed...

**54:20** ...that there's this little p on here next to that variable block group.

**54:26** It says, is it a model variable, or is it not a model variable?

**54:30** It's that little p that comes up.

**54:32** And what will happen is that, if I double-click this and use it as a tool, basically everything that's a...

**54:36** ...model parameter will show up in the dialog.

**54:40** So you control what are the inputs and what are the outputs that are exposed.

**54:44** In this case, I'm exposing the block groups.

**54:49** And I'm outputting the block groups, but that doesn't show up, and I'm also making a parameter of..

**54:54** ...maximum to do that.

**54:58** The variable equals the data type, the data type of the parameter equals the data type of a variable.

**55:03** You can change from optional to required, you could specify a filter, and you could specify output symbology.

**55:10** So, in model properties, there's this property page called for parameters, it gives you the name and the data type.

**55:18** For things that are optional you can switch it to required, and then there's also a filter here that you can go in there...

**55:25** ...depending if it's a feature class, or it will let you do a feature class filter; if it's a string, it'll let you do a string filter...

**55:30** ...if it's a number, it'll let you do a number filter.

**55:32** So only certain data types let you put a filter value in there.

**55:36** But if it's appropriate, it will let you edit the filter for that.

**55:41** So, if I have a model that inputs a feature class, I only want it to put in points...

**55:47** ...then I can modify the filter to do that.

**55:54** The other thing that control is when I want the model to run, at the very end I want it to have it to have nice symbology.

**56:01** And if you go to, so in this case my output, well, in this case my output are theselected features.

**56:17** I don't have it saved.

**56:18** But you go to Properties, you go to Layer Symbology, and you pick a layer file.

**56:23** And a layer file becomes a template that gets used when the layer is made for that feature class.

**56:29** We have, it can be static, where it just always uses the same symbology.

**56:37** It can also support dynamic.

**56:38** You have dynamic, unique value and dynamic range to do that.

**56:43** And go look at the, there's a section in the Help that talks about setting up those layers and saving them out...

**56:51** ...and what properties on the renderer make it dynamic or not.

**57:03** So it's classified; if it's not manual, it will redo the classification.

**57:07** Unique values, if all other values are false, it will recompute unique values each time a layer

is generated...

[57:14](#) ...for the output there.

[57:18](#) So let's go back to this example.

[57:22](#) Well, that was kind of neat, but everytime I had to come in here and I gotta select another one...

[57:27](#) ...and run it, and select another one, and run it...

[57:30](#) So what I want to do is I want to kind of make this so that I can enter a point and it runs it for that area.

[57:38](#) So, I've got Main here; what I did was say that I take, well, let's just start from scratch here.

[57:51](#) And inside here, a new model.

[57:55](#) And what I can do is, I can do Select Number of Customers.

[58:01](#) And it's in there, and it takes as the input is blocked groups.

[58:04](#) And before, it starts out with a selected set of the blocks, and then what I want to do is, well before I run it...

[58:11](#) ...I want to select the area on that.

[58:14](#) So, what I did was, so I have this, the finished model here.

[58:19](#) If you look at it, it does Select by Location with a Point.

[58:27](#) Let me make that bigger.

[58:30](#) Okay, it does Select by Location with a Point.

[58:32](#) So then that's the feature that will start, and then this will run until some maximum customer value that we put in there.

[58:41](#) So now, I can run this, if I run Main.

[58:45](#) And in fact I'm going to just pick one point, and I want to up this number a little bit.

[58:53](#) And I want it to run to 50,000, and then I can...oops, nope, sorry.

[59:06](#) And I can zoom to selected.

[59:10](#) Oh!! You idiot you! Okay.

[59:13](#) Sorry about that.

[59:14](#) All right. Well, you know what I want to do?

[59:16](#) I want to write a model that does zoom to selected for me. Forget this.

[59:24](#) So, actually, there's a couple of things I want to do here.

**59:27** Actually, here's what I want to do.

**59:29** I want to run this model; I want to pick a point.

**59:33** And then, it will expand on that area and then zoom to selected on it.

**59:38** All right, well how do you do that?

**59:39** Okay. Did you like that? I kind of liked that.

**59:47** I liked that so much I'm going to do it again.

**59:53** All right. Uh-oh. Now that's where you cancel. Yeah, don't hit those islands.

**1:00:02** See the Cancel works.

**1:00:08** Okay. So, the idea here is, so, how did I do that.

**1:00:12** So, remember before that the, this input here, the point was a point feature class.

**1:00:20** But if I look at this, I can do...if I look at the data type on this...and I go to data type you'll see that it's a feature set.

**1:00:29** And so, what I do, I do Create Variable, pick feature set, okay, click on the properties of that...

**1:00:39** ...and what I do is I pick a layer file or feature class that is the template for that.

**1:00:48** So now if I open this up you'll see, well, I get this interactive input thing.

**1:00:53** And in ModelBuilder it's blocked, you can't use it inside of ModelBuilder, but what you do is...

**1:00:58** ...you make it a parameter, and now that's the parameter that we use.

**1:01:02** So, basically the thing is for interactive input - point, line, or poly, it doesn't matter what type - you can use feature set for that.

**1:01:09** You draw it and you do that.

**1:01:11** So now I enter the point and I do Select By Location, I run the submodel that grows until it's done...

**1:01:19** ...and then actually there's a...and then I added on to this little script called Zoom To Selected.

**1:01:25** So, now if you're not, you may have heard, one of the new things in version 10 for geoprocessing...

**1:01:34** ...is what we call ArcPy mapping.

**1:01:35** Anyway, the mapping module of ArcPy.

**1:01:38** So, a lot of people have asked for, hey I want to open a map document, I want to turn layers

on and off...

**1:01:43** ...I want to add layers, remove layers, change extent, and I want to print it.

**1:01:46** Okay, I want to do things to maps.

**1:01:49** And so there's a little scripting language for doing that; in this case, you open up the current map document...

**1:01:58** ...you get the data frame, and in fact you get the layer and off of the layer, you get the selected extent.

**1:02:05** And, at the end of the class I've got a listing about when this class is, if you're interest specifically in ArcPy mapping.

**1:02:15** So what that does is zoom to selected.

**1:02:17** So, in my main model I select one point, I feed it to my submodel that does iteration.

**1:02:26** And then when I'm done, I just do Zoom To Selected on that.

**1:02:35** All right. Any questions about that?

**1:02:38** It's kind of a lot to throw at you at once.

**1:02:43** All right.

**1:02:50** How do you make things interactive? You can use Feature Set.

**1:02:54** Okay. So the last thing I want to talk about...how are we doing on time here?

**1:03:01** Are we, we're 11:30. Okay.

**1:03:06** So a little flow control.

**1:03:08** So we have these things, we call them preconditions.

**1:03:11** The idea say, there's two problems that it solves.

**1:03:16** One is that sometimes I just want one tool to run before another tool.

**1:03:20** And ModelBuilder has no ordering built into it.

**1:03:23** There's no way to say...you know, basically ordering is built in by the chain.

**1:03:29** If one thing is subordinate to another, it will run first.

**1:03:32** But if two things are in separate chains, it's not random what is...basically, the one that's entered first will run.

**1:03:40** You can't look at the model and know which one was entered first.

**1:03:43** You can't look at it and change the order that things are running.

**1:03:46** So the idea is, oh, basically I made it a pseudo input.

**1:03:49** We have these things called Preconditions, I can connect, in this case I connect the output of Create Feature Dataset...

**1:03:55** ...as a precondition to Clip, and now basically it becomes a...well, it becomes an input.

**1:04:03** It's not a parameter input, it's just a processing input.

**1:04:06** Basically it says, this needs to run before that.

**1:04:09** So if you need to ever enforce that, you can do that with Precondition.

**1:04:13** So preconditions can also be used for if-then branches in a model.

**1:04:18** So this is, the idea here is a tool...it's the upgrade metadata tool.

**1:04:25** And depending on what metadata type you pick, it does this processing, and if you do the other type...

**1:04:31** ...it does the lower processing.

**1:04:32** So, if you can see then there's essentially two branches that are going on, so there's some condition checking here...

**1:04:39** ...it does the processing, and then it brings it back together.

**1:04:43** The way it does that is, basically you do upgrade type; you do upgrade type Esri ISO, or you do upgrade types FGDC.

**1:04:51** Just a simple string.

**1:04:53** So if that string says Esri ISO, this chain runs, so this is a true/false value...

**1:04:58** ...that's used as a precondition here, and here's a true-false value that's used as a precondition there.

**1:05:03** So that's how I start the branch, you break it out, and then at the very end we have merge branch.

**1:05:09** So let's actually go take a look at that live and see what that looks like.

**1:05:16** So in the conversion tools, inside a metadata, there's upgrade metadata...

**1:05:24** There's actually lots of real interesting things in here.

**1:05:27** A lot of these techniques we've been talking about are part of this model here.

**1:05:32** So if I go look at model properties, go to Parameter, go to Update Type, there's a filter on there.

**1:05:40** So that controls what you input. So now I get a choice list of what are the inputs for that.

**1:05:45** You don't just type in whatever you want.

**1:05:50** Calculate upgrade.

**1:05:51** In this case, this is that one that has the, it goes and gets the style sheet based on the type that you input there.

**1:06:01** And then one of the calculate value here, if you look at that, basically it says upgrade type equals Esri ISO.

**1:06:09** The two choices that you put in the choice first, and it outputs a Boolean.

**1:06:14** And the way that works, so if we come in here, and if I look at the properties on here, and I go to precondition...

**1:06:20** ...you'll see that Esri ISO is a precondition of this running.

**1:06:25** So, if Esri ISO true, it will run; if Esri ISO is false, it will not run.

**1:06:31** You could put multiple things in here.

**1:06:32** If any one of them is false, that will not run.

**1:06:36** So that's the way you break it up.

**1:06:42** So that's how I set up; in this case, ISO versus FGDC, it does whatever the appropriate processing is.

**1:06:49** Now at the very end, I want this model, I only want a single output, right, so I've got these two chains, right.

**1:06:56** I've got the ISO result and the FGDC result, but I only want one output of this, and we have a tool called Merge Branch.

**1:07:03** It takes those two inputs, and what it does is it looks at which one's been validated, which one has actually been run.

**1:07:09** If this has been run, it passes it through; if this has been run, it passes it through.

**1:07:13** So, if they've been run it passes the first one through.

**1:07:16** So, you know, the idea is set your model up so that only one of them succeeds.

**1:07:21** Only one of the branches will run when it does that.

**1:07:35** Okay. So, yeah, I mentioned for the Python, the unlisted session.

**1:07:40** It's in the online session but it's not in your little printed thing, so we're at Python Scripting for Map Automation...

**1:07:46** ...that's Tuesday at 1:30 and Friday at 9:00.

**1:07:49** If you want to see it, please fill out the forms.

**1:07:53** And, any other questions you have?

**1:07:54** Open to questions here right now.

**[1:07:59](#)** Way back in the corner there.

**[1:08:04](#)** I'm sorry I can't hear you.

**[1:08:05](#)** [inaudible audience participation]

---

© Esri 2013 <http://www.esri.com>