FAB323246

MEP Fabrication Scripting 101 (Beginner) - Lecture

Darren Young
Hermanson Company – Seattle Washington / Portland, Oregon

Learning Objectives

- Learn what you can and can't do with scripting
- Learn basic scripting language and syntax
- Learn to write simple scripts for use with Fabrication CADmep, Fabrication ESTmep, and Fabrication CAMduct
- Learn how scripts can edit fabrication ITM properties

Description

Fabrication Content if the single most important thing a MEP firm maintains to facilitate preconstruction. To help you manage that content, this session will teach you the basics of Fabrication (COD) scripting for Fabrication CADmep software, Fabrication CAMduct software, and Fabrication ESTmep software. You need no prior knowledge of scripting. We'll show you how easy it is to create COD scripts, and how to use them to manage your own Fabrication ITM content.

Speaker(s)

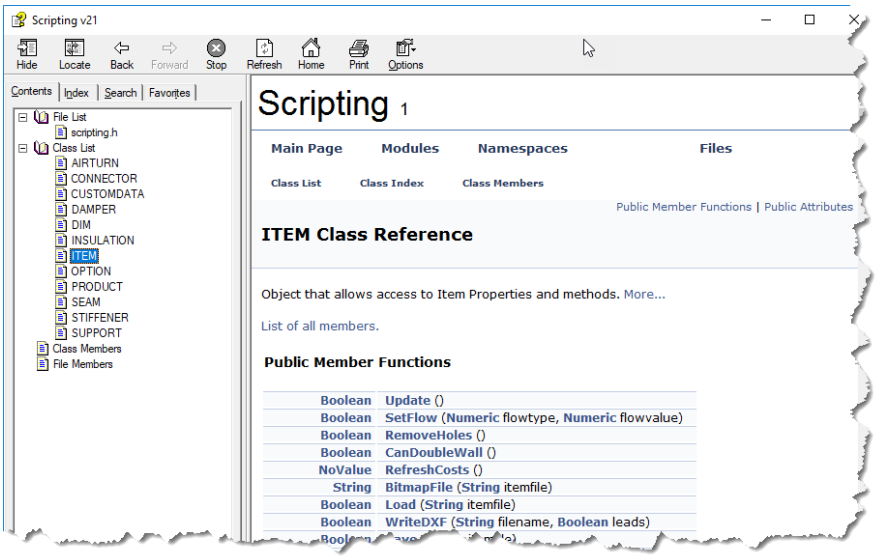
Based in Washington state, Darren Young has been a veteran Autodesk University speaker for well over a decade. His unique ability to leverage multiple everyday technologies in interesting ways to solve complicated and laborious tasks has been valued by users around the world. Down to earth and approachable, he's always willing to help his peers anytime of the year even outside of Autodesk University. Darren's background includes a wide variety of disciplines such as Construction, Engineering, Manufacturing, LEAN, Information Technology, Computer Programming, Author and Technical Editor. His lectures and labs are not just a training opportunity for others but a venue which connects him personally with users helping him learn as well.

Resources

Help file installed locally…

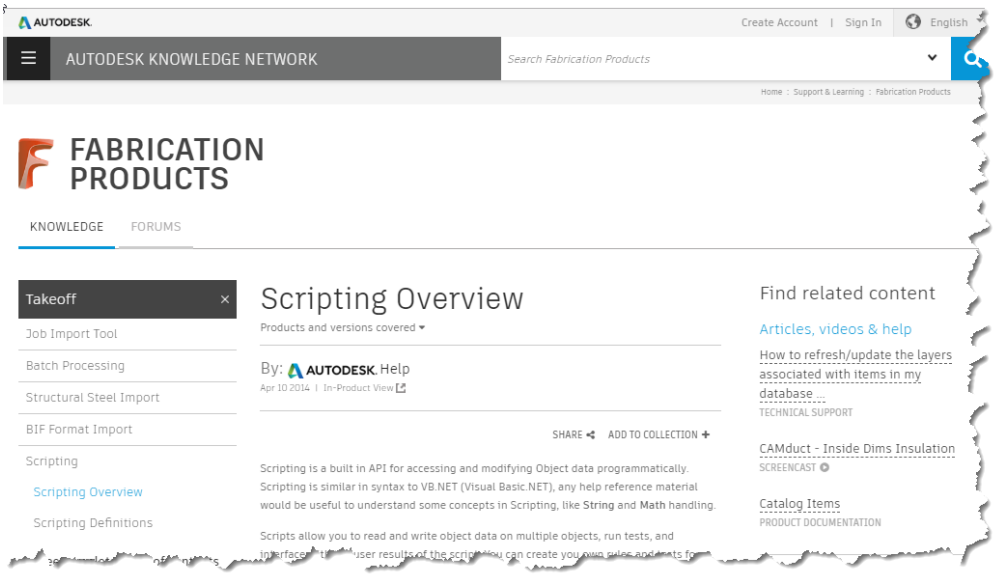
C:\Program Files\Autodesk\Fabrication *<year><product>*\en-US\Scripting.chm**

<year>	*2013, 2014, 2015, 2016, 2017, 2018, 2019, 2020*
<product>	*CADmep, ESTmep, CAMduct*



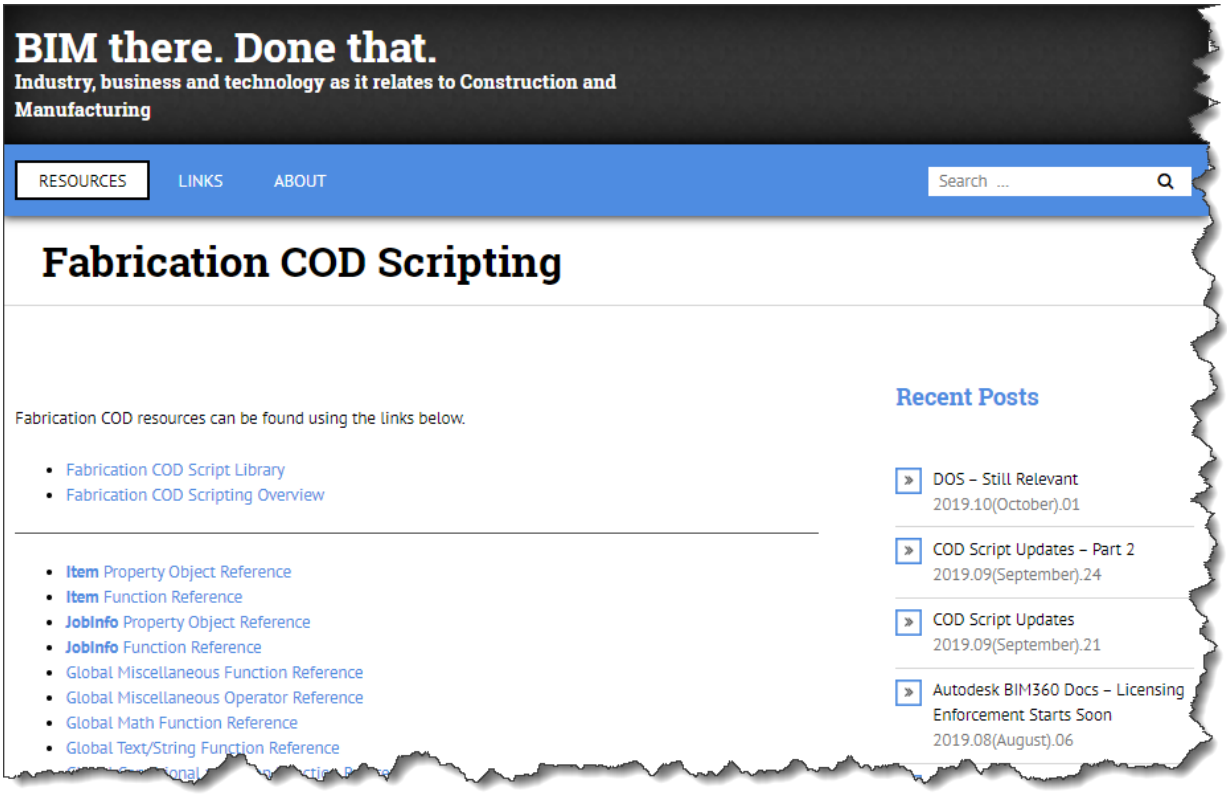
Online Tutorial & Documentation…

https://tinyurl.com/2020-COD-Scripting



My site…

http://www.darrenjyoung.com/resources/autodesk-fabrication/fabrication-cod-scripting/



Scripting in a NutShell….

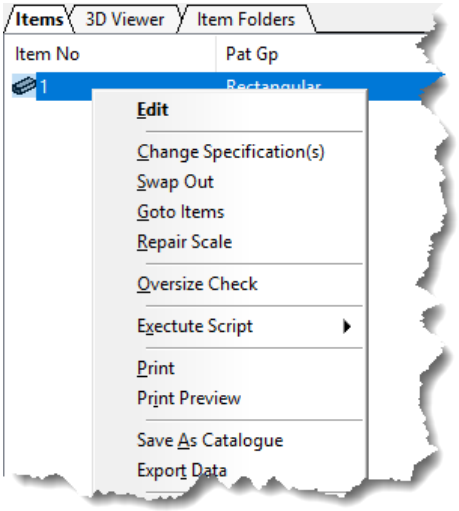
- Written in ASCII Text (Windows Notepad, etc.)
- Can use Script Editor in Fabrication but can be buggy when running scripts.
- File extension for scripts = *.COD
- Text/Strings surrounded by double quotes
- Double quotes from Word/Email (“ ”) are not the same as Notepad (" ")
- Use ASCII Character codes to mimic Enter, Tab, Quotes within text strings
- Variables must be declared with **DIM**
- Variable names must be begin w/alpha character, no spaces odd characters
- Functions/Properties/Variables are NOT CaSe SeNsItIvE
- Scripts Read/Write properties, they don’t automate drawing.
- ITEM is default object for items in drawing
- JOB is default object for job properties
- Unless using advanced processing tasks or functions (not covered in this 101 session), scripts are run on each selected object. E.g. 2 selected objects runs the script twice, once for each object.

Calling Scripts in CADmep…

- Type “EXECUTESCIPT” at the command line
- Use (executescript “script name”) function from AutoLISP

Calling Scripts in ESTmep/CAMduct…

- Use Script Editor to Load & Run – **File -> Open Script**
- Highlight Items from Takeoff – **Right-Click -> Execute Script**



ITEM Object Properties	
Read	**item.airturns**
Read/Write	**item.airturn[**<*index*>**].value**
Read/Write	**item.airturn[**<*index*>**].locked**
Read/Write	**item.alias**
Read/Write	**item.alternate**
Read/Write	**item.bitmap**
Read/Write	**item.boughtout**
Read/Write	***** item.box**
Read/Write	**item.buttonalias**
Read/Write	**item.buttoncode**
Read/Write	**item.cadblock**
Read	***** item.catalogue**
Read/Write	**item.cid**
Read/Write	**item.comment**
Read	**item.connectors**
Read/Write	**item.connector[**<*index*>**].alt**
Read	**item.connector[**<*index*>**].group**
Read/Write	**item.connector[**<*index*>**].locked**
Read	**item.connector[**<*index*>**].type**
Read/Write	**item.connector[**<*index*>**].value**
Read/Write	**item.costbylength**
Read/Write	**item.costtype**
Read	**item.customdata[**<*name*>/<*index*>**].id**
Read/Write	**item.customdata[**<*name*>/<*index*>**].value**
Read/Write	**item.cuttype**
Read	**item.dampers**
Read/Write	**item.damper[**<*index*>**].locked**
Read/Write	**item.damper[**<*index*>**].value**
Read/Write	**item.databaseid**
Read	**item.dblock**
Read	**item.dblock.history**
Read	**item.dblock.history.changed**
Read	**item.dblock.history.info**
Read	**item.dblock.owner**
Read	**item.dblock.version**
Read	**item.decoiler.beading**
Read	**item.decoiler.coilwidth**
Read	**item.decoiler.smalllength**
Read	**item.decoiler.stdlength**
Read	**item.decoiler.stdqty**
Read/Write	**item.description**
Read	**item.dims**
Read	**item.dim[**<*name*>/<*index*>**].annotation**
Read/Write	**item.dim[**<*name*>/<*index*>**].locked**
Read	**item.dim[**<*name*>/<*index*>**].name**

Read	**item.dim[**<*name*>/<*index*>**].numvalue**
Read	**item.dim[**<*name*>/<*index*>**].status**
Read/Write	**item.dim[**<*name*>/<*index*>**].value**
Read/Write	**item.dimside**
Read/Write	**item.dimsidelock**
Read/Write	**item.doublewall**
Read/Write	**item.drawing**
Read/Write	**item.dwlock**
Read/Write	***** item.etag**
Read/Write	**item.extraetime**
Read/Write	**item.extraetimerate**
Read/Write	**item.extraetimeunits**
Read/Write	**item.extraftime**
Read/Write	**item.extraftimerate**
Read/Write	**item.extraftimeunits**
Read/Write	**item.fabtable**
Read/Write	**item.fabtablelock**
Read/Write	**item.facing**
Read/Write	***** item.facinglock**
Read/Write	**item.filename**
Read/Write	**item.fixrelative**
Read/Write	**item.gauge**
Read/Write	**item.gaugelock**
Read	***** item.guid**
Read	***** item.guid64**
Read	***** item.handle**
Read	**item.hasproduct**
Read/Write	**item.insspec**
Read/Write	**item.installtable**
Read/Write	**item.installtablelock**
Read/Write	**item.insulation**
Read/Write	**item.insulation.facing**
Read/Write	**item.insulation.gauge**
Read/Write	**item.insulation.material**
Read/Write	**item.insulation.materiallock**
Read/Write	**item.insulation.status**
Read/Write	**item.insulation.statuslock**
Read/Write	**item.ispeclock**
Read	**item.library**
Read/Write	***** item.lifespan**
Read	**item.links**
Read/Write	**item.link[**<*name*>/<*index*>**].name**
Read/Write	**item.link[**<*name*>/<*index*>**].param**
Read/Write	**item.link[**<*name*>/<*index*>**].target**
Read	**item.manyoldstatus**
Read	**item.matabrv**

Read/Write	**item.material**
Read/Write	**item.nestpriority**
Read/Write	**item.notes**
Read/Write	**item.number**
Read	**item.oldstatus[***<name>/<index>***]**
Read	**item.oldstatus[***<name>/<index>***].datetime**
Read	**item.oldstatus[***<name>/<index>***].id**
Read/Write	**item.oldstatus[***<name>/<index>***].userid**
Read	**item.oldstatus[***<name>/<index>***].value**
Read/Write	***** item.operatingcost**
Read	**item.options**
Read/Write	**item.option[***<name>/<index>***].locked**
Read	**item.option[***<name>/<index>***].name**
Read	**item.option[***<name>/<index>***].status**
Read/Write	**item.option[***<name>/<index>***].value**
Read/Write	**item.order**
Read/Write	**item.pallet**
Read	***** item.partscut**
Read	***** item.partcut{*<index>*]**
Read/Write	**item.path**
Read	**### item.patno**
Read/Write	**item.pricelist**
Read/Write	**item.pricetablelock**
Read	**item.product.entries**
Read/Write	**item.product.entry[***<index>***].alias**
Read/Write	**item.product.entry[***<index>***].area**
Read/Write	**item.product.entry[***<index>***].boughtout**
Read/Write	**item.product.entry[***<index>***].cadblock**
Read/Write	**item.product.entry[***<index>***].customdata[***<name>/<index>***]**
Read/Write	**item.product.entry[***<index>***].databaseid**
Read/Write	**item.product.entry[***<index>***].dim[***<index>***]**
Read/Write	***** item.product.entry[*<index>*].flowmax**
Read/Write	***** item.product.entry[*<index>*].flowmin**
Read/Write	**item.product.entry[***<index>***].model**
Read/Write	**item.product.entry[***<index>***].name**
Read/Write	**item.product.entry[***<index>***].option[***<index>***]**
Read/Write	**item.product.entry[***<index>***].order**
Read/Write	***** item.product.entry[*<index>*].skey**
Read/Write	**item.product.entry[***<index>***].weight**
Read	**item.product.hasalias**
Read	**item.product.hasarea**
Read	**item.product.hasboughtout**
Read	**item.product.hascadblock**
Read	**item.product.hascustomdata**
Read	**item.product.hascustomdatas**
Read	**item.product.hasdatabaseid**

Read	**item.product.hasdims**
Read	***** item.product.hasflow**
Read	**item.product.hasoptions**
Read	**item.product.hasorder**
Read	**item.product.hasrevision**
Read	***** item.product.hasskey**
Read	**item.product.hasweight**
Read/Write	**item.qty**
Read/Write	**item.scale**
Read/Write	***** item.sealant.locked**
Read/Write	***** item.sealant.value**
Read	**item.seams**
Read/Write	**item.seam[***<index>***].alt**
Read/Write	**item.seam[***<index>***].locked**
Read/Write	**item.seam[***<index>***].value**
Read/Write	**item.section**
Read/Write	**item.service**
Read/Write	**item.servicetype**
Read/Write	**item.skinconnector[***<index>***].alt**
Read	**item.skinconnector[***<index>***].group**
Read/Write	**item.skinconnector[***<index>***].locked**
Read	**item.skinconnector[***<index>***].type**
Read/Write	**item.skinconnector[***<index>***].value**
Read	**item.skindecoiler.beading**
Read	**item.skindecoiler.coilwidth**
Read	**item.skindecoiler.smalllength**
Read	**item.skindecoiler.stdlength**
Read	**item.skindecoiler.stdqty**
Read	**item.skindecoiler.stdlength**
Read/Write	**item.skingauge**
Read/Write	**item.skinmaterial**
Read/Write	**item.skinmateriallock**
Read/Write	**item.skinseam[***<index>***].alt**
Read/Write	**item.skinseam[***<index>***].locked**
Read/Write	**item.skinseam[***<index>***].value**
Read/Write	**item.skinside**
Read/Write	**item.specification**
Read/Write	**item.speclock**
Read	**item.splitters**
Read/Write	**item.splitter[***<index>***].value**
Read/Write	**item.splitter[***<index>***].locked**
Read/Write	**item.spool**
Read/Write	**item.spoolcolour**
Read/Write	**item.status**
Read	**item.stiffeners**
Read/Write	**item.stiffener[***<index>***].locked**

Read/Write	**item.stiffener[**<*index*>**].qty**
Read/Write	**item.stiffener[**<*index*>**].spacing**
Read/Write	**item.stiffener[**<*index*>**].value**
Write	**item.structuretype**
Read	**item.subtems**
<*varies*>	**item.subtem[**<*index*>**].**<*Same Properties as Item Objects*>
Read/Write	**item.support.locked**
Read/Write	**item.support.qty**
Read/Write	**item.support.spacing**
Read/Write	**item.support.value**
Read	**item.type**
Read/Write	**item.weight**
Read/Write	**item.weightlock**
Read/Write	**item.wiregauge**
Read/Write	**item.zone**

*****(undocumented)**

###(new – 2019.1.0 and later only)

ITEM Object Functions		
addcustomdata	**item.addcustomdata("Room number")**	**n/a**
	item.addcustomdata(52)	
bitmapfile	**item.bitmapfile("./Charlotte/pipe.itm")**	**image file**
candoublewall	**item.candoublewall()**	**True/False**
canrotary	***** item.canrotary()**	**True/False**
endlocation	**item.endlocation(**<*index*>**)**	**"x y z"**
	item.endlocation(<*index*>**, "xyz")**	**"x y z"**
	item.endlocation(<*index*>**, "x")**	**X**
	item.endlocation(<*index*>**, "y")**	**Y**
	item.endlocation(<*index*>**, "z")**	**Z**
	item.endlocation(<*index*>**, "btm")**	**-(OD/2)**
	item.endlocation(<*index*>**, "top")**	**+(OD/2)**
level	**item.level("Floor")**	**Level**
	item.level("Soffit")	**Level**
load	**item.load("./Charlotte/pipe.itm")**	**True/False**
refreshcosts	**item.refreshcosts()**	**n/a**
removeholes	**item.removeholes**	**n/a**
save	**item.save("./Charlotte/pipe.itm")**	**True/False**
setflow	**item.setflow(0,)** = Not Set	**True/False**
	item.setflow(1,250) = Supply	
	item.setflow(2,250) = Return	
	item.setflow(3,) = None	
update	**item.update()**	**True/False**
writedxf	**item.writedxf("./mydxfs/elbow", True)**	**True/False**
	item.writedxf("./mydxfs/elbow", False)	

*****(undocumented)**

JOB Object Properties	
Read/Write	**job.colour**
Read	**job.customdata[***<name>/<index>***].id**
Read/Write	**job.customdata[***<name>/<index>***].value**
Read	**job.date**
Read/Write	**job.field1**
Read/Write	**job.field2**
Read	**job.items**
Read/Write	**job.item[***<index>***].…** *(see ITEM above)*
Read	**job.name**
Read/Write	**job.notes**
Read	**job.project**
Read/Write	**job.reference**
Read	**job.statuses**
Read	**job.status.[***<name>/<index>***].active**
Read	**job.status.[***<name>/<index>***].lastactivated**
Read	**job.status.[***<name>/<index>***].name**

JOB Object Functions		
setstatus	**job.setstatus(***<index>***, True)**	**True/False**

Global Misc Functions		
debug	**debug("My Alert Box")**	
dim	**dim myvariable**	
inputbox	**inputbox("Title", "Prompt", "Default")**	
query	**query("Chose Yes or No")**	**True/False**
rem	**Rem This is a Note**	

Global Misc Operators		
+	**debug 1 + 1**	**2**
	debug("This is" + " " + "a test")	
-	**debug 5 - 3**	**2**
*	**debug 2 * 2**	**4**
/	**debug 4 / 2**	**2**
and	**debug 1 = 1 and 2 = 2**	
not	**debug Not True**	
or	**debug Test = 1 or Test = 2**	

Global Math Functions		
acos	**acos(3, 4)**	**41.4096221**
asin	**asin(3, 4)**	**48.5903779**
atan	**atan(3, 4)**	**36.8698976**
cos	**cos(45)**	**0.7071068**
exp	**exp(2)**	**10**
log	**log(100)**	**2**
number	**number("5")**	**5**
	number("5.0")	**5**
	number("5.5")	**5.5**
pow	**pow(5, 2)**	**25**
	pow(2, 3)	**8**
round	**round(5.49)**	**5**
	round(5.50)	**6**
rounddown	**rounddown(5.9)**	**5**
roundup	**roundup(5.1)**	**6**
sign	**sign(-3.4)**	**-1**
	sign(3.4)	**1**
	sign(0.000001)	**0**
sin	**sin(45)**	**0.7071068**
sqrt	**sqrt(4)**	**2**
sqr	**sqr(2)**	**4**
tan	**tan(45)**	**1**

Global Text/String Functions		
asc	**asc("A")**	**65**
ascii	**ascii(65)**	**"A"**
chr	**chr("Autodesk", 5)**	**"d"**
getfileext	**getfileext("c:\(temp)\test.txt")**	**".txt"**
	getfileext("c:/(temp)/test.txt")	**".txt"**
getfilename	**getfilename("c:\\(temp)\test.txt")**	**"test.txt"**
	getfilename("c:/(temp)/test.txt")	**"test.txt"**
getfilepath	**getfilepath("c:\(temp)\test.txt")**	**"c:/(temp)/ "**
	getfilepath("c:/(temp)/test.txt")	**"c:/(temp)/ "**
instr	**instr(1, "Autodesk", "D", False)**	**5**
	instr(1, "Autodesk", "D", True)	**0**
	instr(6, "Autodesk", "D", False)	**0**
left	**left("Autodesk", 4)**	**"Auto"**
len	**len("Autodesk")**	**8**
lower	**lower("Autodesk")**	**"autodesk"**
ltrim	**ltrim(" Autodesk ")**	**"autodesk "**
mid	**mid("Autodesk", 5, 2)**	**"de"**
right	**right("Autodesk", 4)**	**"desk"**
rtrim	**rtrim(" Autodesk ")**	**" autodesk"**
substring	**substring("Autodesk", 3, 4)**	**"tode"**
trim	**trim(" Autodesk ")**	**"autodesk"**
upper	**upper("Autodesk")**	**"AUTODESK"**
wildcard	**wildcard("Autodesk", "*desk")**	**1**
	wildcard("Autodesk", "*chair")	**0**
	wildcard("Autodesk", "?ut?d*")	**1**

Conditional Test & Loop Functions	

Do - Loop	dim count = 10
	do
	  debug count
	  count = count - 1
	loop until count = 0

Do - While	dim count = 5
	do while count < 10
	  debug count
	  count = count + 1
	loop

For – Next	Dim mynum **for mynum=1 to 10** debug mynum **next mynum**
For – Next – Step	Dim mynum **for mynum=1 to 10 step 2** debug mynum **next mynum**

If	dim mynum = 1
	if mynum = 1 then
	  debug "Yes"
	endif

If - Else	dim mynum = 1
	if mynum = 1 then
	  debug "Yes"
	else
	  debug "No"
	endif

If – ElseIf - Else	dim mynum = 1
	if mynum = 1 then
	  debug "1"
	elseif mynum = 2 then
	  debug "2"
	else
	  debug "Other"
	endif

Select Case	dim mynum = 3
	select mynum
	  case 1
	  debug "Doing 1"
	  case 2, 3
	  debug "Doing 2 & 3"
	  case 4
	  debug "Doing 4"
	end select

Select Case Else	dim mynum = 3
	select mynum
	  case 1
	  debug "Doing 1"
	  case 2, 3
	  debug "Doing 2 & 3"
	  case else
	  debug "Doing something else"
	end select

While	dim loop = 1
	while loop <= 5
	  debug loop
	  loop = loop + 1
	endwhile