

## **Introduction to PanelPro - Logix**™

Dick Bronson - *R R - C irK its*, *Inc*.

# Intro to Logix<sup>™</sup>

# Indirect layout control (PP-clinic-3)





# Why LogiX?

- When Dave Duchamp first started adding a graphical logic package to JMRI we wondered about what to call it. "Logic" seemed to be a logical name for logic, but Dave had already added 'Lights' as a function, therefore 'L' was no longer available as an item name, so he just used 'X' instead. The logic function was 'Internal' to JMRI, so its system name was 'I'. This means the the proper identifier for the logic function became 'IX' and we jokingly started calling them Logix in our e-mail discussions because of the 'IX'. The name has stuck.
- The original Logix were functionally similar to industrial ladder logic in that they did not have any parenthetical structure. Pete Cressman, a California Java programmer, at the urging of David Parks, a modeler trying to use Logix for his extensive B&O CPL signals, figured our how to add mixed logic to Logix in a graphical way. This new capability is fully available in Release 2.6 of JMRI, and will change this presentation somewhat from last years version.

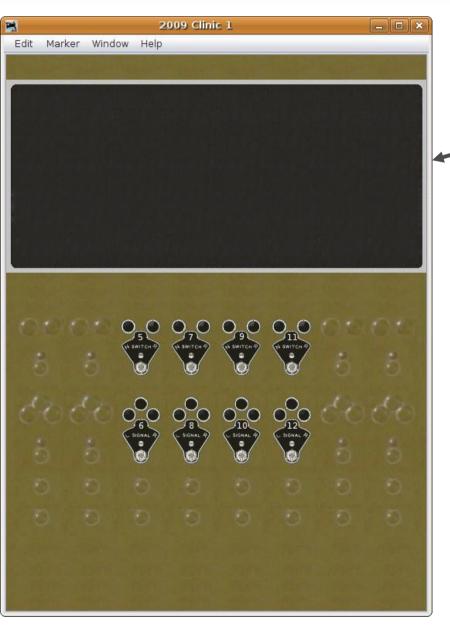


Layout control

- In our previous clinic we simply tied our active icons directly to the layout commands that we needed to send. This is no more sophisticated that drilling some holes in a piece of Masonite, spray painting some lines, mounting some switches and lamps, and then connecting them to our switch machines. Granted a computer can usually be found for not very much money, but a few switches or push buttons, a chopped up string of Christmas tree lights, and some paint would be cheaper.
- On the prototype railroads it is not allowable to have remote control of turnouts without some fairly reliable method of knowing the current position of the points and preventing them from ever being changed while a train is crossing them. (or about to) Now that we mention it, these are pretty good things to do for our models as well, even if the life hazard is less. (not counting what might happen to the dispatcher when he accidentally sends that new brass onto the floor through that spot with no scenery yet)
- All this to say, maybe just flipping a turnout with a remote switch isn't the best idea after all, especially if you can't see it from the panel.



#### Fixed images

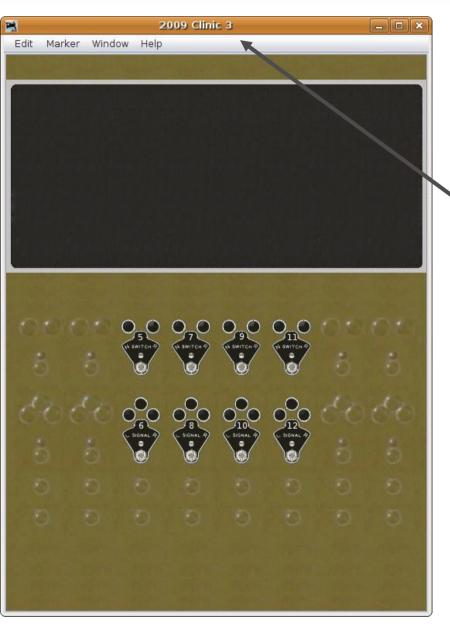


# **Indirect Layout Control**

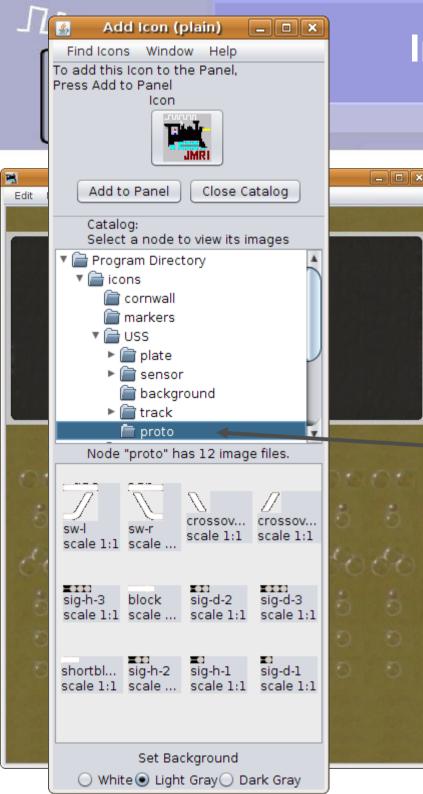
 First lets load in the basic panel background that we made in clinic #1 then rename it and save it as clinic #3



#### Fixed images

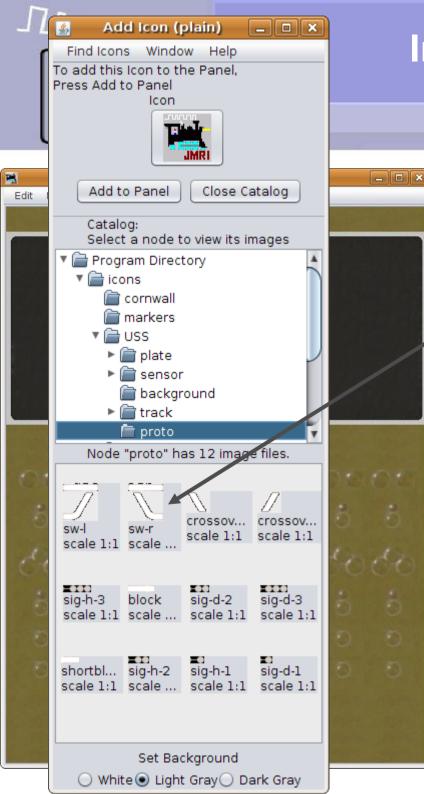


- First lets load in the basic panel background that we made in clinic #1 then rename it and save it as clinic #3
- You are expected to know how to do all the basic operations already covered in previous sessions, so I am not going to repeat the detail of each operation as we move along.



#### Fixed images

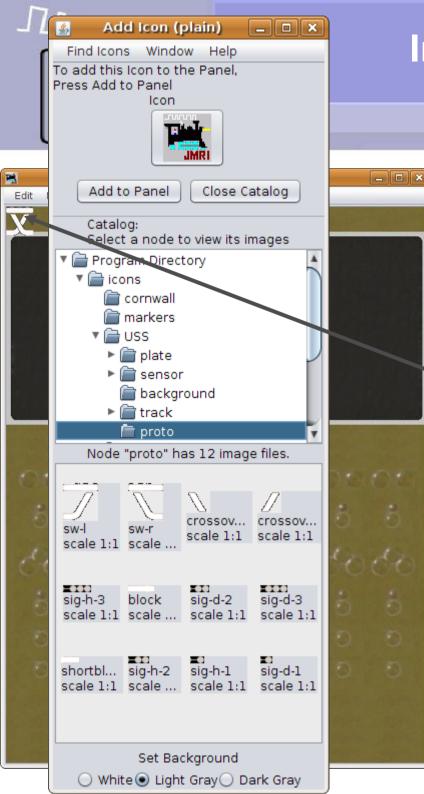
- First lets load in the basic panel background that we made in clinic #1 then rename it and save it as clinic #3
- You are expected to know how to do all the basic operations already covered in previous sessions, so I am not going to repeat the detail of each operation as we move along.
- Navigate to the 'proto' folder where we have a set of images created from photographs of an original classic era prototype US&S CTC machine.



#### Fixed images

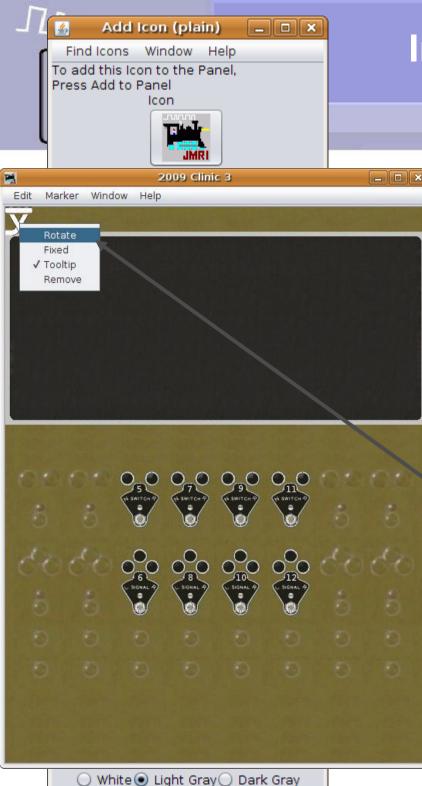
# **Indirect Layout Control**

These few images are not
 designed for animation, but for constructing a more realistic panel.



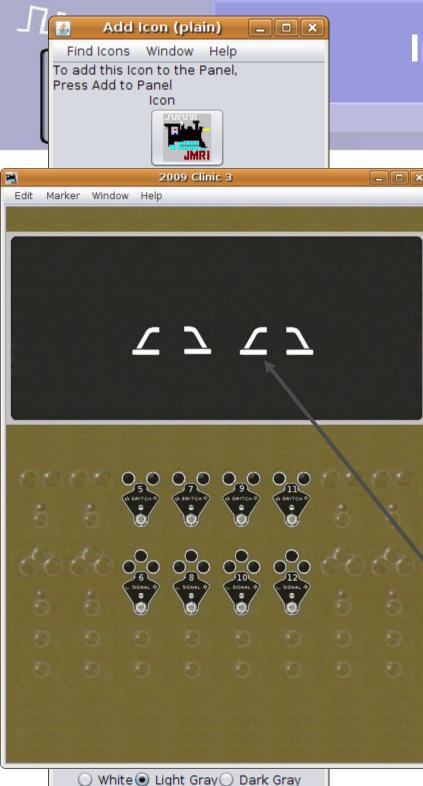
#### Fixed images

- These few images are not designed for animation, but for constructing a more realistic panel.
- Use the 'Add to Panel' button to add two left (sw-l) and two right (sw-r) turnout icons to our panel.



#### Fixed images

- These few images are not designed for animation, but for constructing a more realistic panel.
- Use the 'Add to Panel' button to add two left (sw-l) and two right (sw-r) turnout icons to our panel.
- These images only face in one direction, so they will need to be rotated for our use on this panel. Right click (meta for Mac) to bring up the tools, then click on 'Rotate' to rotate 90°.

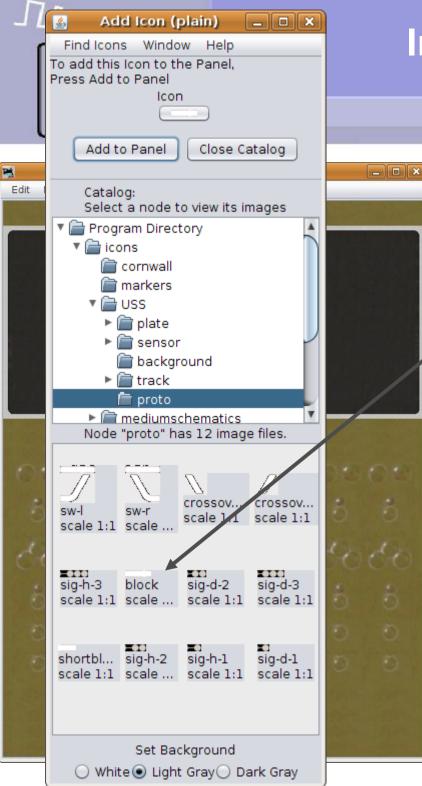


#### Fixed images

# Indirect Layout Control

- These few images are not designed for animation, but for constructing a more realistic panel.
- Use the 'Add to Panel' button to add two left (sw-l) and two right (sw-r) turnout icons to our panel.
- These images only face in one direction, so they will need to be rotated for our use on this panel. Right click (meta for Mac) to bring up the tools, then click on 'Rotate' to rotate 90°.

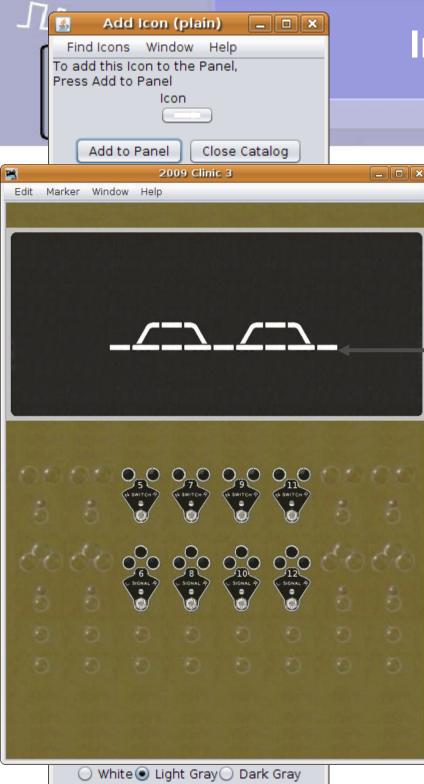
Rotate' each icon twice, and then position it on the panel.



#### Fixed images

# **Indirect Layout Control**

Now use the 'Add to Panel'
 button to add seven 'block' icons to our panel.



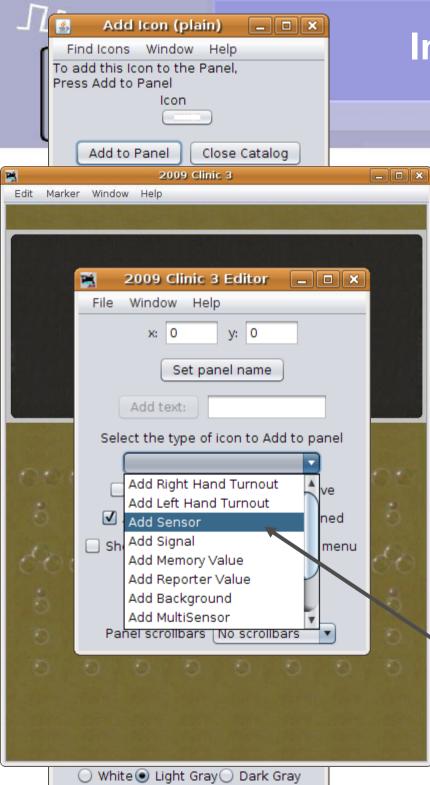
#### Fixed images

- Now use the 'Add to Panel' button to add seven 'block' icons to our panel.
- Place them appropriately.



#### Sensor images

- Now use the 'Add to Panel' button to add seven 'block' icons to our panel.
- Place them appropriately.
- One of the 'rules' we have for remote operation is that we do not throw a switch under a train. To accomplish that we need to know when a train is on the switch or 'OS' (On Switch) section. (OS can mean other things such as 'On Sheet', Off Sheet, etc.)

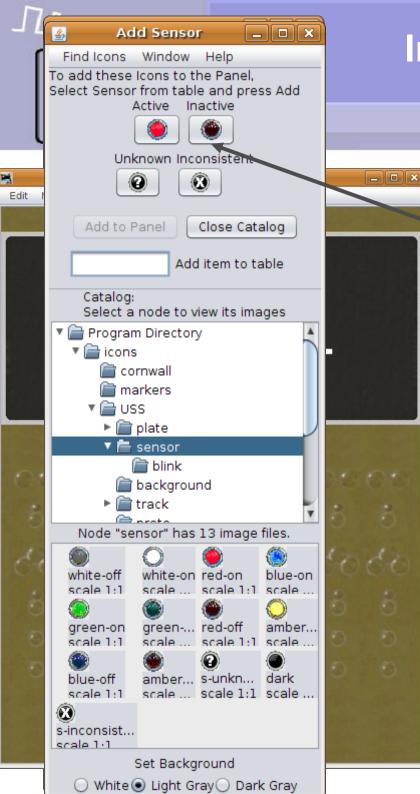


#### Sensor images

# Indirect Layout Control

- Now use the 'Add to Panel' button to add seven 'block' icons to our panel.
- Place them appropriately.
- One of the 'rules' we have for remote operation is that we do not throw a switch under a train. To accomplish that we need to know when a train is on the switch or 'OS' (On Switch) section. (OS can mean other things such as 'On Sheet', Off Sheet, etc.)

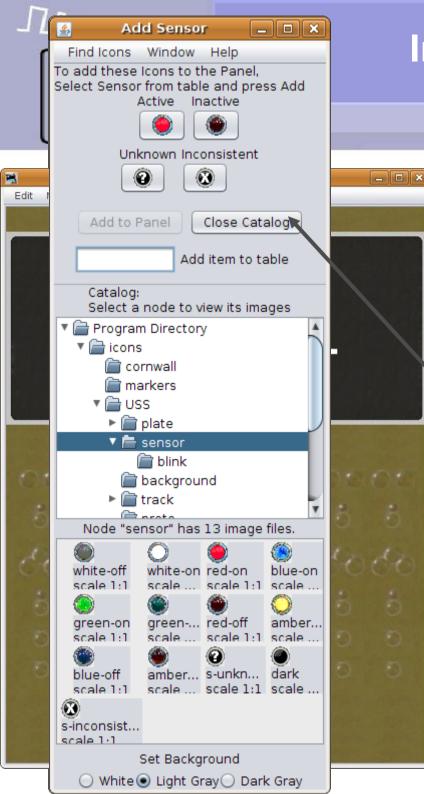
Choose 'Add Sensor' in the Panel Editor window.



#### Sensor images

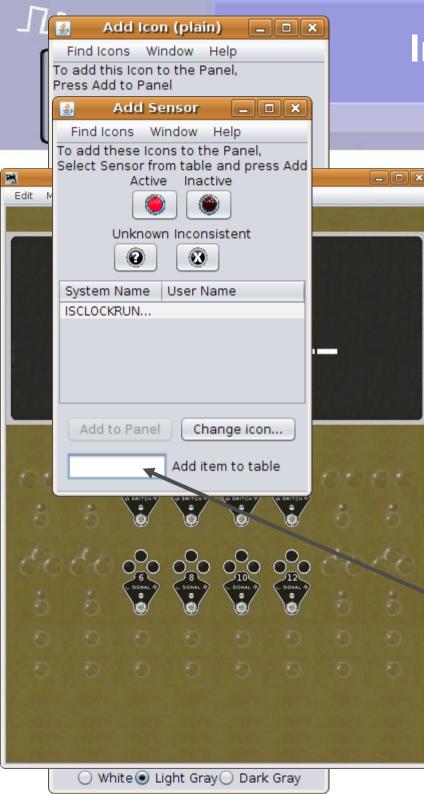
# Indirect Layout Control

 Select the images you want to use for your OS detection. The USS default for a panel was red jewels for OS, and white jewels for blocks. Many railroads had their own standards including all white, all red, all blue, etc.



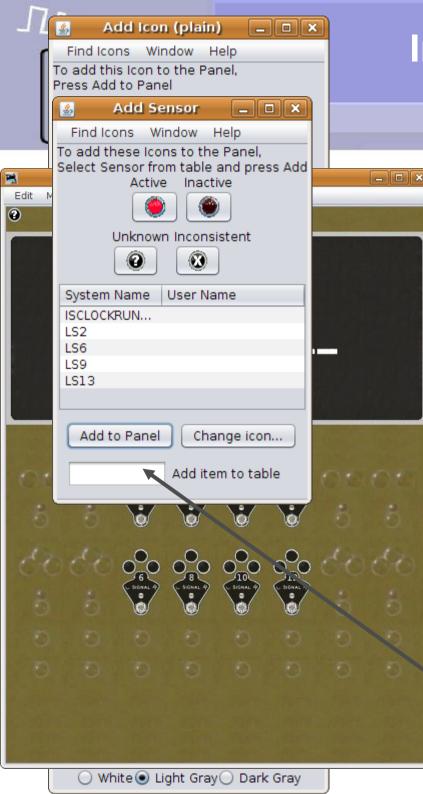
#### Sensor images

- Select the images you want to use for your OS detection. The USS default for a panel was red jewels for OS, and white jewels for blocks. Many railroads had their own standards including all white, all red, all blue, etc.
- To close the selection catalog, and open a list of existing sensors simply click on 'Close Catalog'.



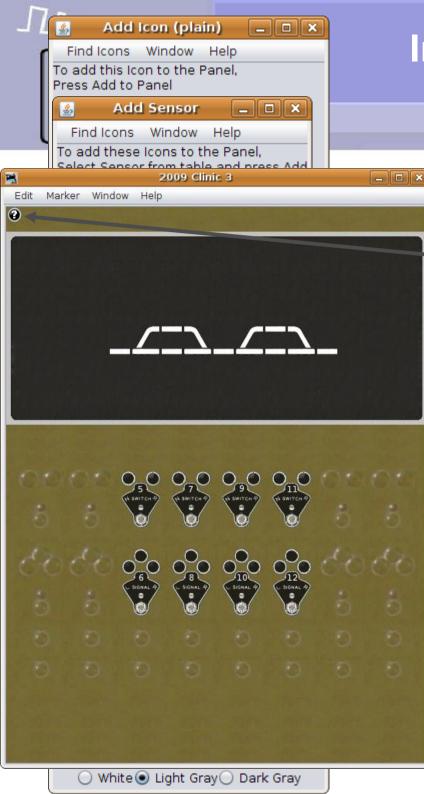
#### Sensor images

- Select the images you want to use for your OS detection. The USS default for a panel was red jewels for OS, and white jewels for blocks. Many railroads had their own standards including all white, all red, all blue, etc.
- To close the selection catalog, and open a list of existing sensors simply click on 'Close Catalog'.
- In clinic 2 we added active icons for our turnouts. Now we are doing the same for our occupancy sensors.



#### Sensor images

- Select the images you want to use for your OS detection. The USS default for a panel was red jewels for OS, and white jewels for blocks. Many railroads had their own standards including all white, all red, all blue, etc.
- To close the selection catalog, and open a list of existing sensors simply click on 'Close Catalog'.
- In clinic 2 we added active icons for our turnouts. Now we are doing the same for our occupancy sensors.
- Add sensors LS2, LS6, LS9, and LS13. (LS = LocoNet Sensor)



#### Sensor images

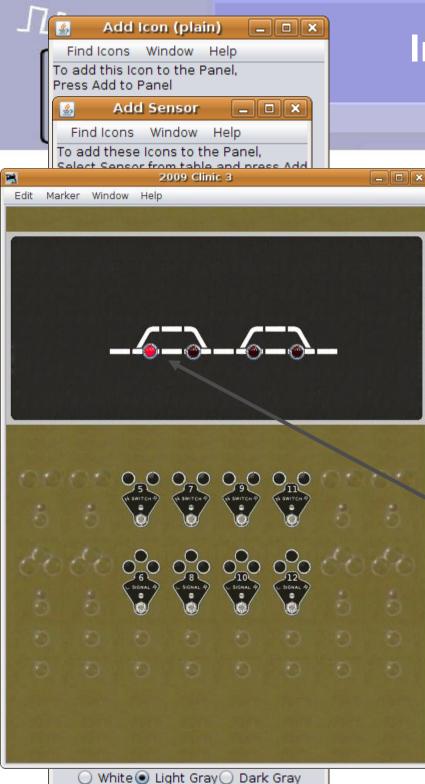
# Indirect Layout Control

 We find our sensors all piled up in the usual place.



#### Sensor images

- We find our sensors all piled up in the usual place.
- Move them onto the panel.

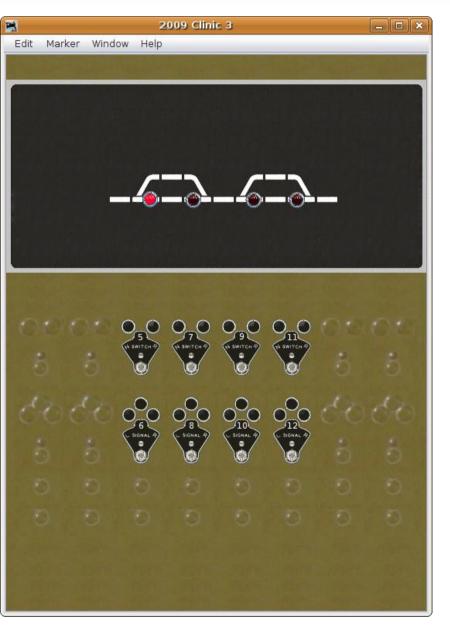


#### Sensor images

- We find our sensors all piled up in the usual place.
- Move them onto the panel.
- Normally we would 'disable' the sensor images so that they would only respond to our occupancy detectors. However we don't actually have any sensors attached, so we will simulate detection by clicking on our images to activate them.



Sensor images

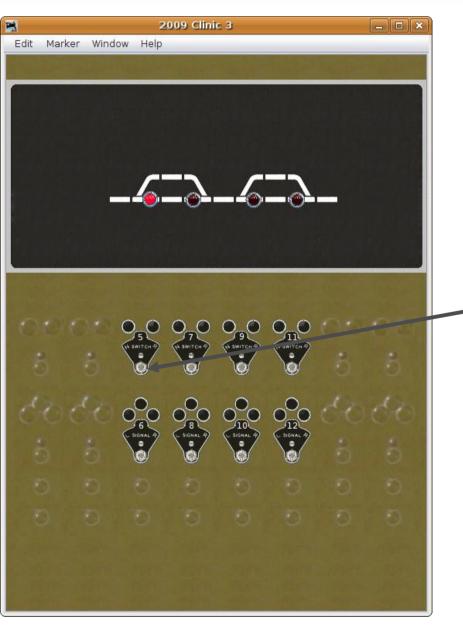


## Internal sensors

 Our next concept is that of 'Internal' sensors. These are really just single bit memory devices. They react with the images just as if they were hardware, but only exist internally to JMRI.



Sensor images



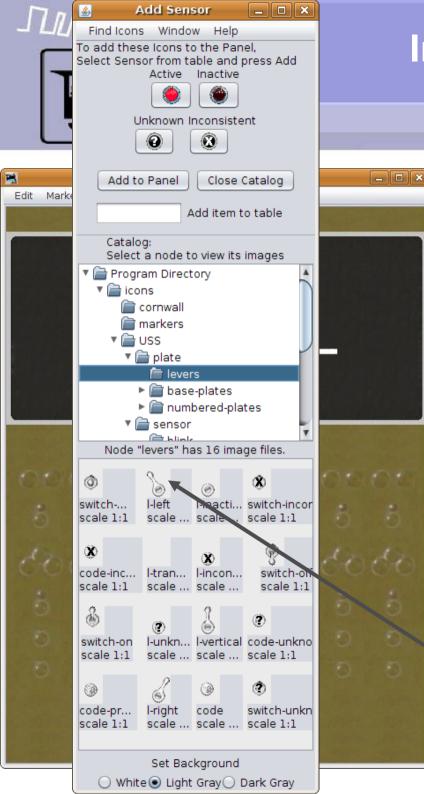
- Our next concept is that of 'Internal' sensors. These are really just single bit memory devices. They react with the images just as if they were hardware, but only exist internally to JMRI.
- We need some new levers that are not directly attached to the turnouts like we had them in the second clinic.



Sensor images

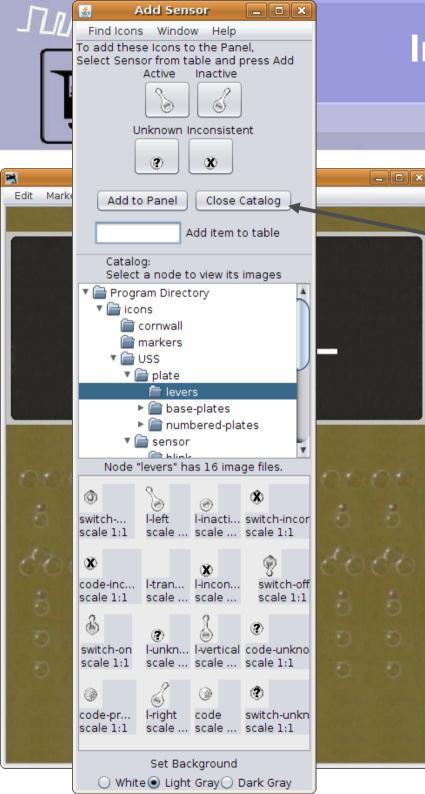


- Our next concept is that of 'Internal' sensors. These are really just single bit memory devices. They react with the images just as if they were hardware, but only exist internally to JMRI.
- We need some new levers that are not directly attached to the turnouts like we had them in the second clinic.
- Pull up the 'Add Sensor' window again and open the 'Change icon..'



#### Sensor images

- Our next concept is that of 'Internal' sensors. These are really just single bit memory devices. They react with the images just as if they were hardware, but only exist internally to JMRI.
- We need some new levers that are not directly attached to the turnouts like we had them in the second clinic.
- Pull up the 'Add Sensor' window again and open the 'Change icon..'
- Navigate back to the 'levers', but this time it will be sensors that have the lever images.



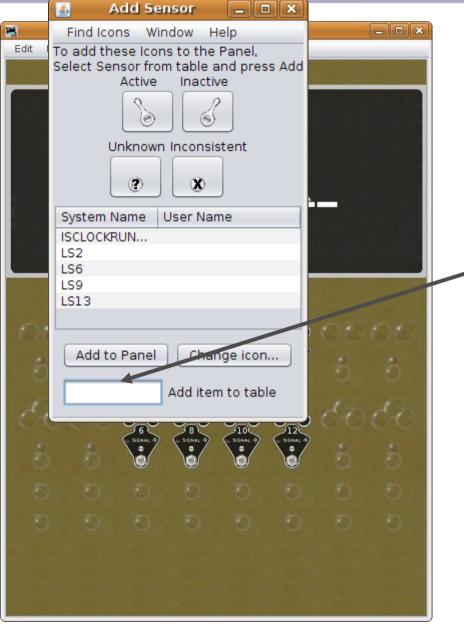
Sensor images

# Internal sensors

Once the lever images are selected click on 'Close Catalog' the reduce the window size and get our sensor list.



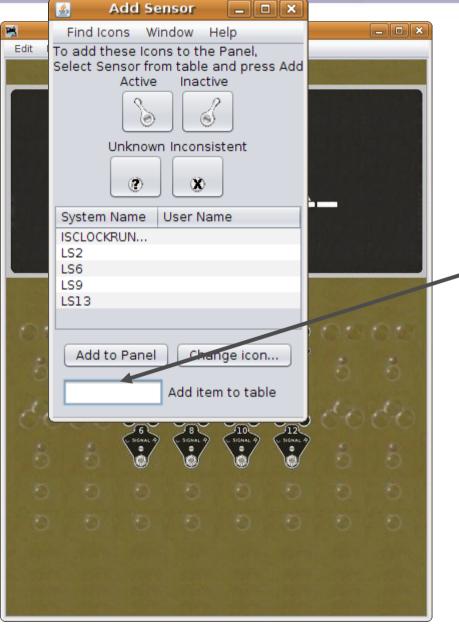
#### Sensor images



- Once the lever images are selected click on 'Close Catalog' the reduce the window size and get our sensor list.
- At this point we could call our internal sensor just about anything, even 'IS-late-to-lunch'. (IS denotes Internal Sensor.



#### Sensor images



- Once the lever images are selected click on 'Close Catalog' the reduce the window size and get our sensor list.
- At this point we could call our internal sensor just about anything, even 'IS-late-to-lunch'. (IS denotes Internal Sensor.
- The prototype railroads have a similar naming problem, so hopefully we can learn something from them.
- AREMA = American Railway Engineering & MOW Association.



#### Sensor images

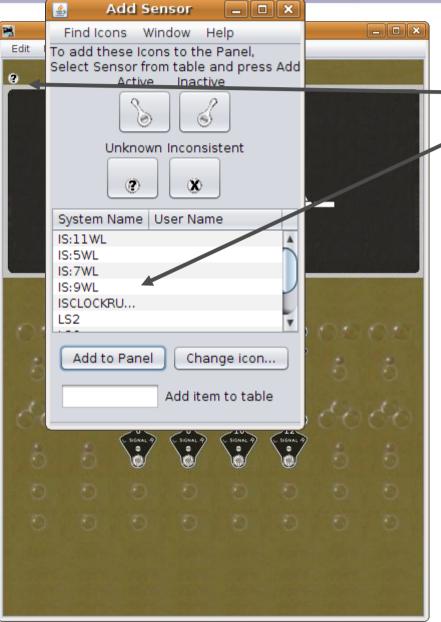
	🛓 🖌	\dd Ser	isor		ı X			_
<b>11</b>	Find Ico	ne Win	dow H	lelp			×	
Edit I								
Sector State	To add the					States and states	a los and his same	1
	Select Ser		n table a Inactiv		is Add			
		Active	inactiv	/e				
	Unknown Inconsistent							8
	ī							
	l	0	8					and a state of the
	System Name User Name							
	ISCLOCKRUN LS2							
	LS2 LS6							1
	LS0 LS9							1
	LS13							
	1.515							
0								
8	Add to Panel Change icon					8	6	
	IS:5WL	_	Add iter	n to tab	0			
14	13:3 WL		Auditer	n to tab	e	60		
~		SIGNAL 4	SIGNAL 4	10	SIGNAL Q			
		SHONAL P	SHOWAL A	CONAL A	C. SIGNAL A			
		U	V	V	Q			
								t
								11

# Internal Sensor Names

- AREMA (Numerical Prefix)(First Letter)..(Last Letter)
  - Numerical Prefix: The number of the lever, signal, track circuit, etc.
  - First Letter: General kind of unit.
  - Last Letter: Specific unit
- For Logix
  - ISn: = Internal Sensor n
  - W = sWitch
  - L = Lever
- Result = IS5:WL (plus IS7:WL, IS9:WL, and IS11:WL)



Sensor images



# **Internal Sensors**

• Our new internal sensors have been added to both the sensor table and the panel.

#### Sensor images

# Add Sensor \$ 2009 Clinic 3 \_ 0 × Edit Marker Window Help 0

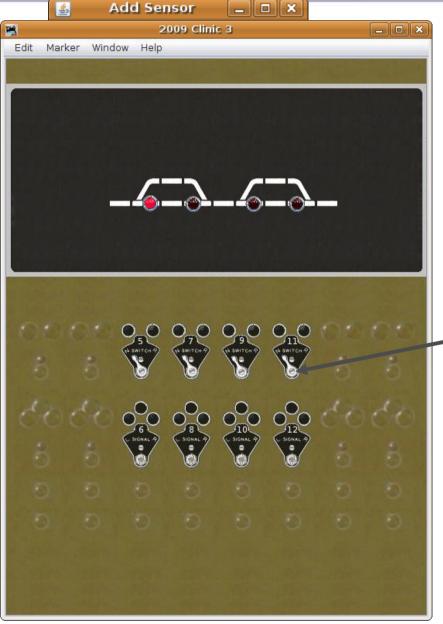
TUTION

1

- Our new internal sensors have been added to both the sensor table and the panel.
- Move them into place.



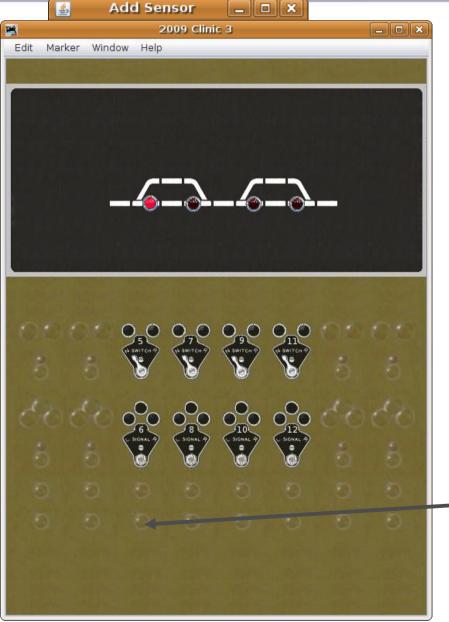
#### Sensor images



- Our new internal sensors have been added to both the sensor table and the panel.
- Move them into place.
- Now we have some levers that are not directly connected to the layout. We can flip them simply by clicking on them.



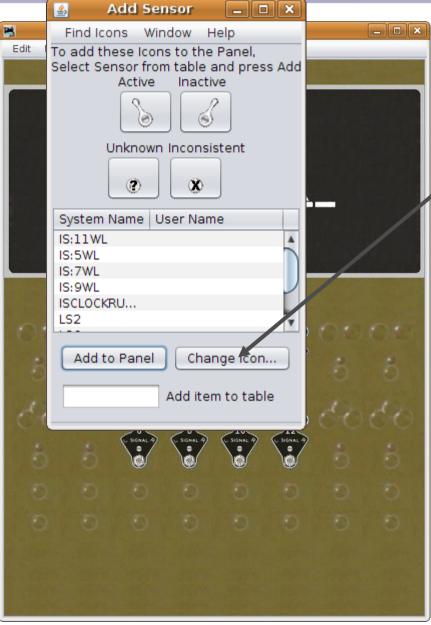
#### Sensor images



- Our new internal sensors have been added to both the sensor table and the panel.
- Move them into place.
- Now we have some levers that are not directly connected to the layout. We can flip them simply by clicking on them.
- A prototype CTC panel also did not directly connect its levers to the switch motors. The operator moved a lever and then pressed a 'Send Code' button that encoded and sent the commands over the track side wires in a serial format that used short and long pulses. (similar to DCC)

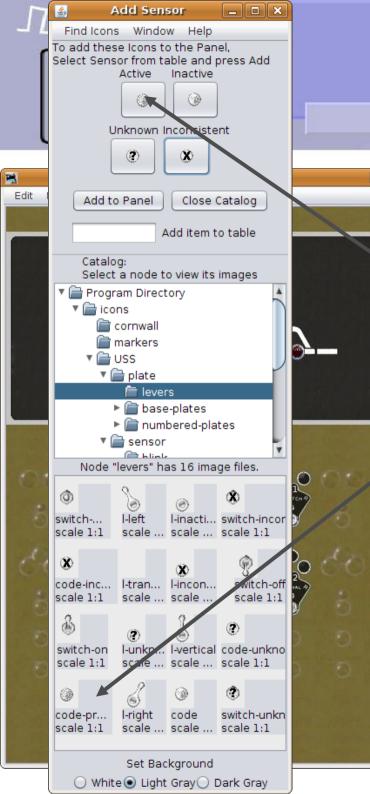


#### Sensor images



# **Internal Sensors**

To add code buttons re-call our
 'Change Icon' list in the 'Add Sensor' window.



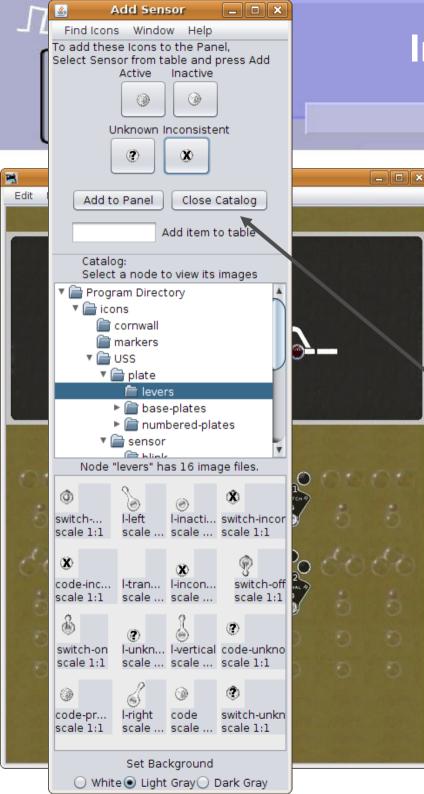
- • ×

#### Sensor images

# **Internal Sensors**

 To add code buttons re-call our 'Change Icon' list in the 'Add Sensor' window.

The 'code-press' icon is the 'Active' entry, the 'code' icon is the 'Inactive'.

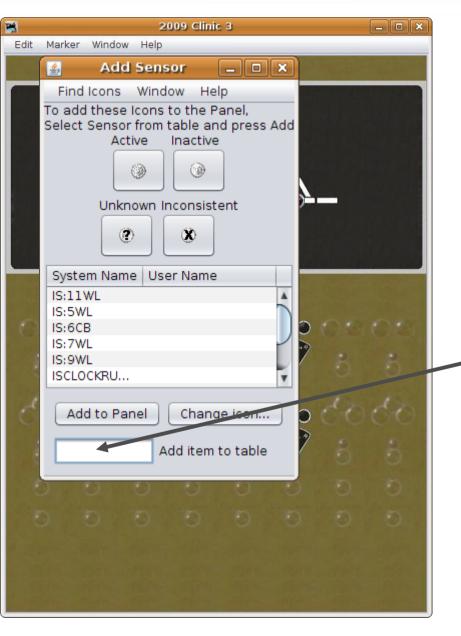


#### Sensor images

- To add code buttons re-call our 'Change Icon' list in the 'Add Sensor' window.
- The 'code-press' icon is the 'Active' entry, the 'code' icon is the 'Inactive'.
- Close Catalog brings us back to our sensor list.



#### Sensor images

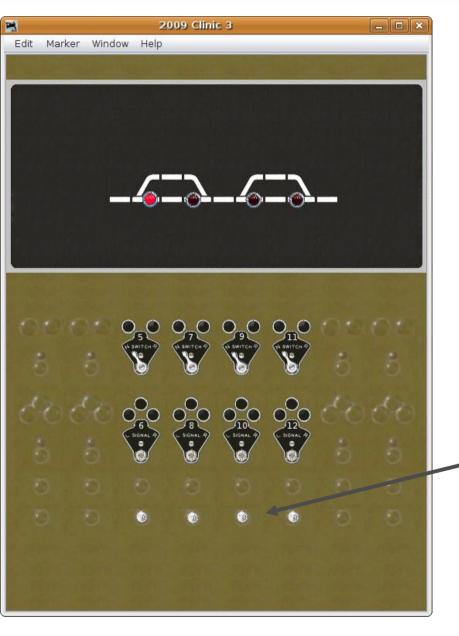


# **Internal Sensors**

- To add code buttons re-call our 'Change Icon' list in the 'Add Sensor' window.
- The 'code-press' icon is the 'Active' entry, the 'code' icon is the 'Inactive'.
- Close Catalog brings us back to our sensor list.
- Enter IS6:CB, IS8:CB, IS10:CB, and IS12:CB for our sensors.



#### Sensor images

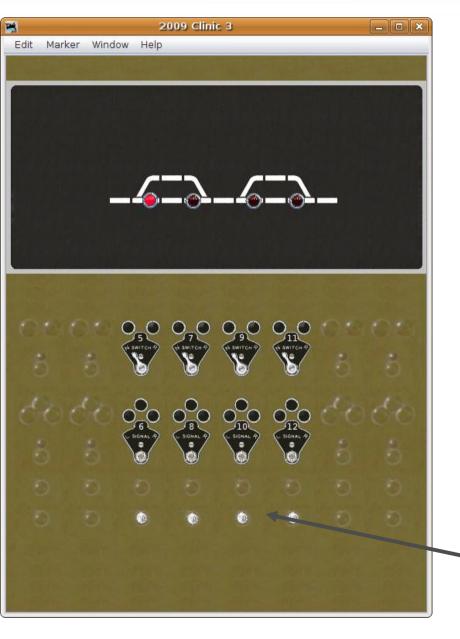


## **Internal Sensors**

- To add code buttons re-call our 'Change Icon' list in the 'Add Sensor' window.
- The 'code-press' icon is the 'Active' entry, the 'code' icon is the 'Inactive'.
- Close Catalog brings us back to our sensor list.
- Enter IS6:CB, IS8:CB, IS10:CB, and IS12:CB for our sensors.
- Move them into place.



#### Sensor images

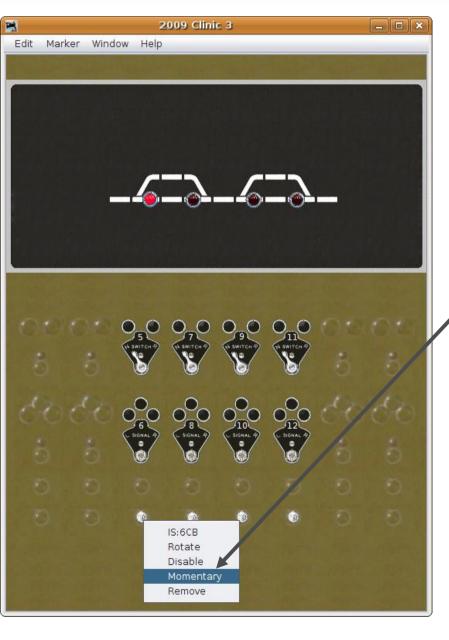


## **Internal Sensors**

- To add code buttons re-call our 'Change Icon' list in the 'Add Sensor' window.
- The 'code-press' icon is the 'Active' entry, the 'code' icon is the 'Inactive'.
- Close Catalog brings us back to our sensor list.
- Enter IS6:CB, IS8:CB, IS10:CB, and IS12:CB for our sensors.
- Move them into place.
- Note, when we click them they go down, and when we click them again they go up. This is not very
  - "buttonlike" behavior.



#### Sensor images

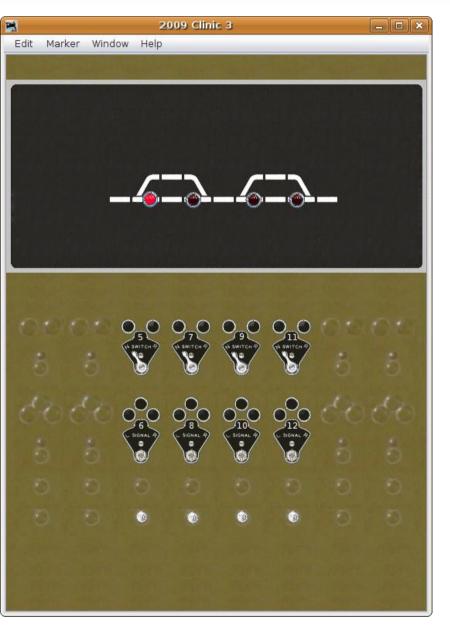


## **Internal Sensors**

 To correct the behavior, right click on each button icon and click
 'Momentary' to check it.



#### Sensor names



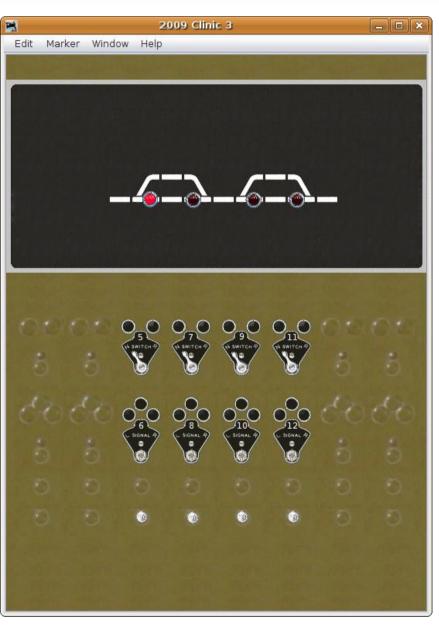
### Internal sensor names

When naming our new buttons we already mentioned that we are not attached to hardware, so any name is allowed. I am loosely basing these names on the AREMA nomenclature. I chose 'ISn:' to begin them all because we need the "IS", and normally system generated names use the ":". The plan is that a tool will eventually generate these names, so I include the ":".

There is a fair amount of repetition in the AREMA names, but an attempt is made to not place different meanings in one position, nor use the same names in different positions. E.g R may be red, reverse, right, relay, etc.



Sensor names

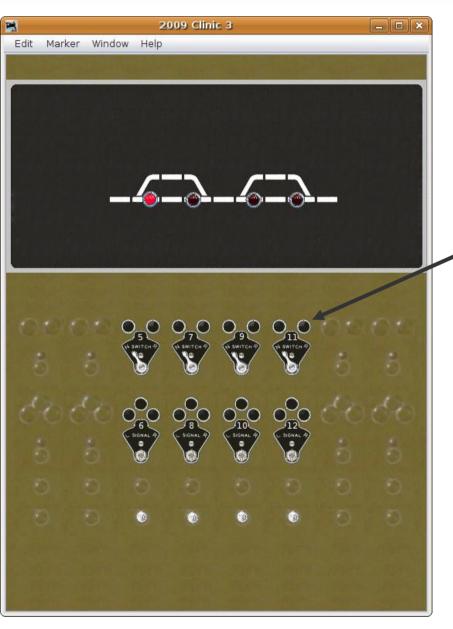


## Internal sensor names

	Beginning	Middle	End
А	Alarm	Track A	
В		Track B	Button
С		Code	Controller
D		proceeD	
F	Fleeting		
G		siGnal	
Κ			indiKtor
L	Left		Lever
Ν	Normal		
R	Reverse	Red	Relay
Т		Track	
W		sWitch	



#### Turnout Feedback

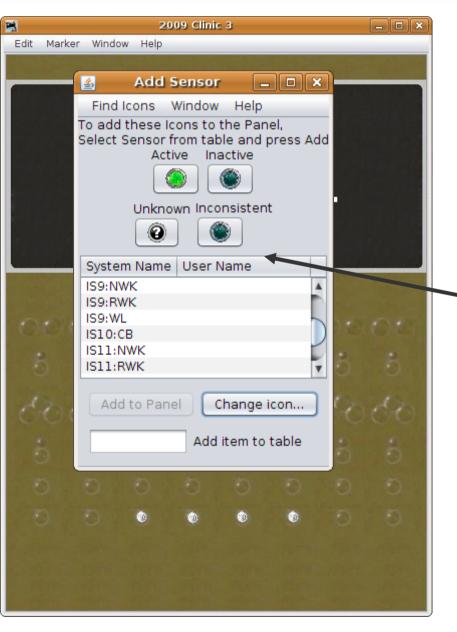


# **Turnout Feedback**

 We still need some way to tell which position the layout track switches are aligned. The levers and track image are not available, so we will use the indicator lamps. (just like the prototype)



#### Turnout Feedback



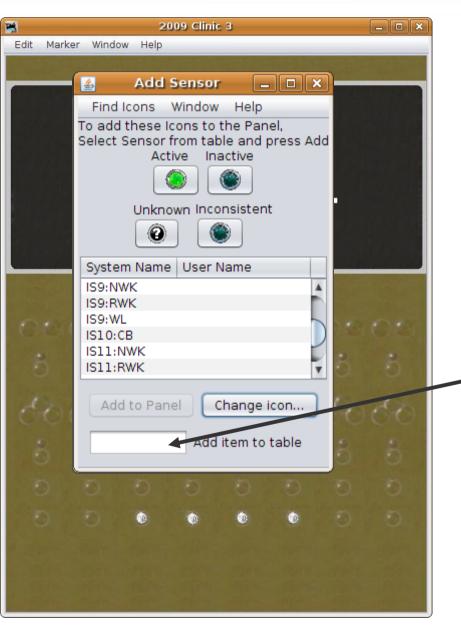
# **Turnout Feedback**

We still need some way to tell which position the layout track switches are aligned. The levers and track image are not available, so we will use the indicator lamps. (just like the prototype)

Pull up the 'Add Sensor' window and set it for green jewel icons. Use 'off' for 'inconsistent'.



#### Turnout Feedback

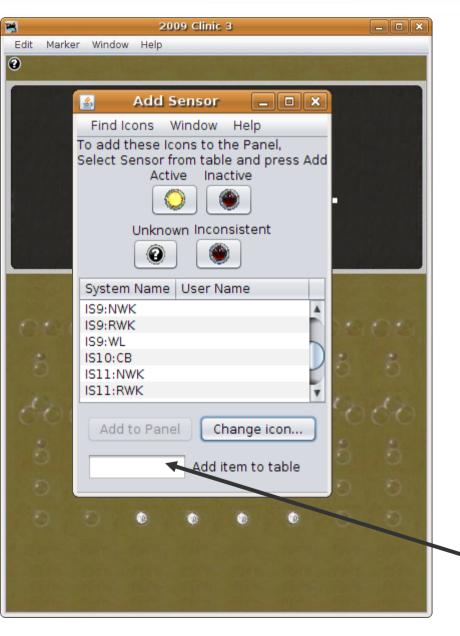


# **Turnout Feedback**

- We still need some way to tell which position the layout track switches are aligned. The levers and track image are not available, so we will use the indicator lamps. (just like the prototype)
- Pull up the 'Add Sensor' window and set it for green jewel icons. Use 'off' for 'inconsistent'.
- From our naming rules we see that the first lamp jewel name should be IS5:NWK. (N = Normal, W = sWitch, and K = indiKtor) followed by IS7:NWK, IS9:NWK, and IS11:NWK.



#### Turnout Feedback

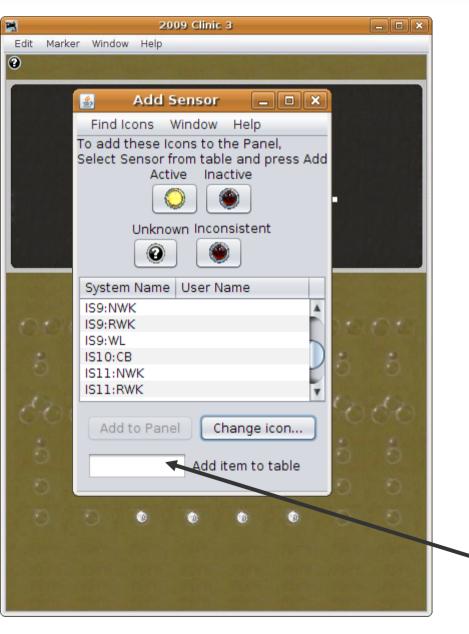


# **Turnout Feedback**

- We still need some way to tell which position the layout track switches are aligned. The levers and track image are not available, so we will use the indicator lamps. (just like the prototype)
- Pull up the 'Add Sensor' window and set it for green jewel icons.
- From our naming rules we see that the first lamp jewel name should be IS5:NWK. (N = Normal, W = sWitch, and K = indiKtor) followed by IS7:NWK, IS9:NWK, and IS11:NWK.
- Now switch to amber jewels and add IS5:RWK. (R = Reverse) followed by IS7:RWK, IS9:RWK, and IS11:RWK.



#### Turnout Feedback

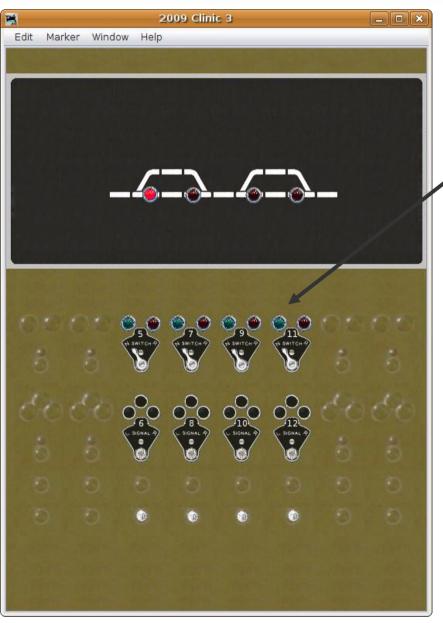


## **Turnout Feedback**

 Note: Later on, in the edit portion of these clinics, we can switch these sensors to direct feedback for testing purposes.



#### Turnout Feedback

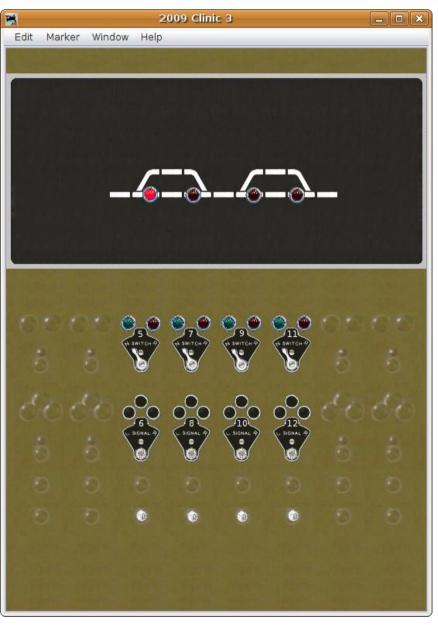


## **Turnout Feedback**

Move the jewels into position. We now have panel icons that we can control based on layout input. We could have used the turnout feedback contacts directly for this, but that would prevent us from adding sound effects for example. We would also need to have our layout hardware exactly line up with the panel ID numbers. This way we can also add a translation table as required. We used 'off' instead of 'inconsistent' images so that JMRI wouldn't flash a 'X' rated image if it decided that the settings were inconsistent.



Logix



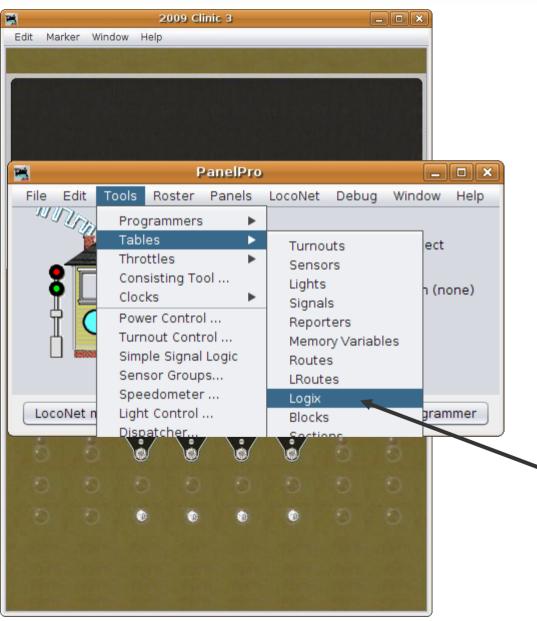
### Logix

 We now have all our required inputs and outputs on the panel. All that is missing is the logic to make it work. Our first example will be simple:

If the Control Lever is changed And the OS is NOT occupied And the Code Button is pressed Then send a turnout command



Logix



## Logix

 We now have all our required inputs and outputs on the panel. All that is missing is the logic to make it work. Our first example will be simple:

If the Control Lever is changed And the OS is NOT occupied And the Code Button is pressed Then send a turnout command

 Open Logix by selecting 'Tools – Tables – Logix'



Logix

-	2009 Clinic	3		×
Edit Marker Wind	low Help			
C Distant States	And the Andrew St	La Caller		
<b>*</b>		ogix Table		- • ×
	ptions	1		
System Name ≜ Us	er Name	Enabled	Comment	
	Element Name			
Add	Cet References		Find Orphans	Empty Cond'Is
0 0	• •	0 0	00	

## Logix

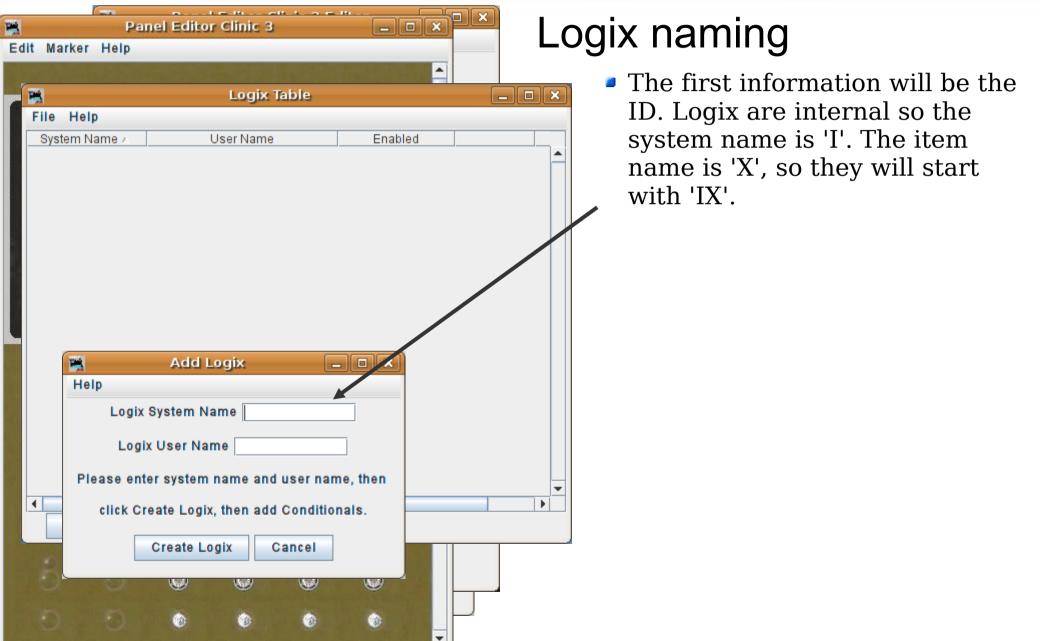
 We now have all our required inputs and outputs on the panel. All that is missing is the logic to make it work. Our first example will be simple:

If the Control Lever is changed And the OS is NOT occupied And the Code Button is pressed Then send a turnout command

- Open Logix by selecting 'Tools Tables – Logix'
- This opens a new 'Logix Table' view. Click on 'Add..'



Logix naming





Logix naming

Panel Editor Clinic 3 - • × Edit Marker Help	Logix naming
Logix Table File Help System Name ▲ User Name Enabled	<ul> <li>The first information will be the ID. Logix are internal so the system name is 'I'. The item name is 'X', so they will start with 'IX'.</li> </ul>
	We will call it IX5:WC
	IX = Internal LogiX, 5: = Plant 5, WC = sWitch Controller.
Add Logix — 🗆 🗙	
Window     Help       Logix System Name     IX5:WC	
Logix User Name Plant 5 Switch Col	
Please enter system name and user name, then click Create Logix, then add Conditionals. Cancel Create Logix	



Logix naming

		Pa	nel Editor	Clinic 3			ix P ×		הכ
Edit	Marker	Help							J
-									
	ile Hel	D		Logix Ta	ble			_ <b> </b>	
Ĺ	System N	-	U	ser Name		Enable	t l		
								<b></b>	
									-
	2		Add Log	ix		×			
	WI		Help			_			
		Logix S	stem Name	IX5:WC	_				
		Logix	Jser Name	Plant 5 Swi	itch Coi				
	Plea	ase ente	r system nai	me and use	er name, th	en		•	
4		click Cre	ate Logix, th	en add Coi	nditionals.				
		ſ	Cancel	Create Log	ix				
	8-					19			
			©	<b>O</b>	<b>W</b>	Q			
			0	0	1	۲			

## \_ogix naming

- The first information will be the ID. Logix are internal so the system name is 'I'. The item name is 'X', so they will start with 'IX'.
- We will call it IX5:WC

IX = Internal LogiX,
5: = Plant 5,
WC = sWitch Controller.

 Enter the user name "Plant 5 Switch Controller". All user names must be unique so we include the plant number.



Logix naming

*		Pe	nel Edito	or Clinic 3	1	-		px)		$\mathbf{r}$
Edit	Marker	Help								J
-			100				<b>^</b>			
				Logix	Table			_	. 🗆 🗙	
	ile Hel			HearNama		5	ablad			
	System N	ame z		User Name	;	j En	abled		A	
	-		Add Lo	gix						
	Wi	ndow	Help							
		Logix Sy	/stem Nam	ne IX5:WC						
		Logix	Jser Name	Plant 5 9	Switch Coi					
	Plea	ise ente	r system r	name and u	user name	, then			-	
•		click Cre	ate Logix,	then add (	Conditiona	ls.			▶	
		ſ	Cancel	Create L						
			<i>W</i>	<b>W</b>	NO.	C.				
			۲	0	۲	۲				

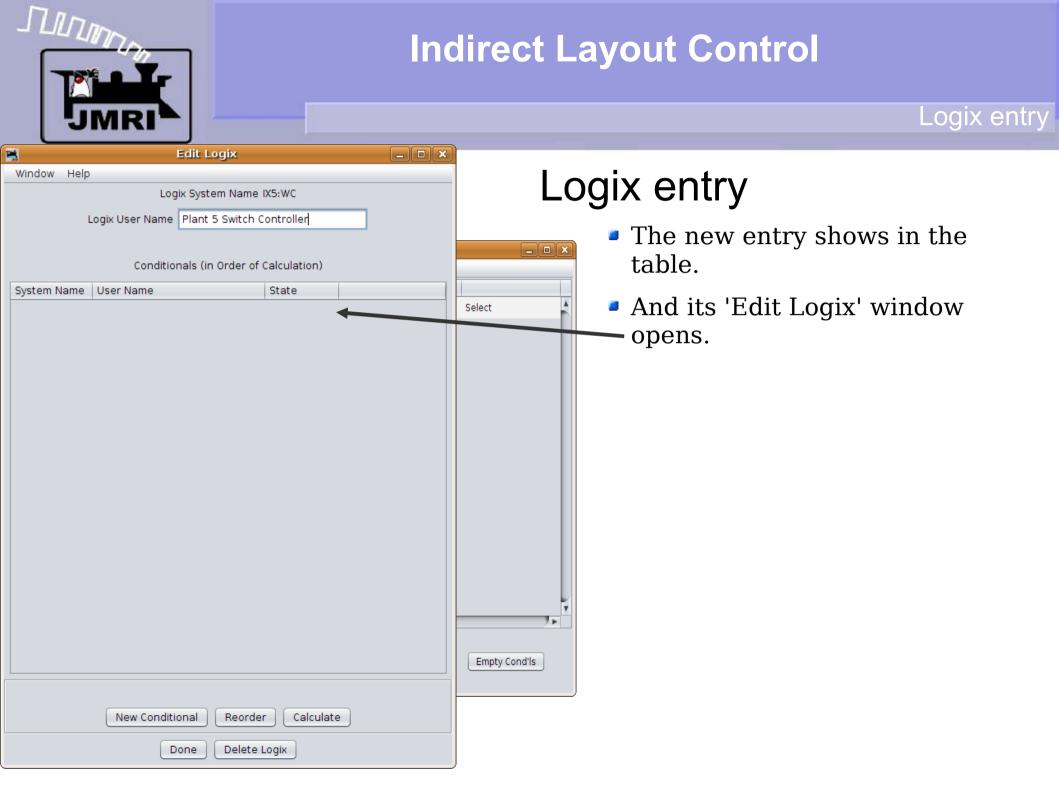
## ogix naming

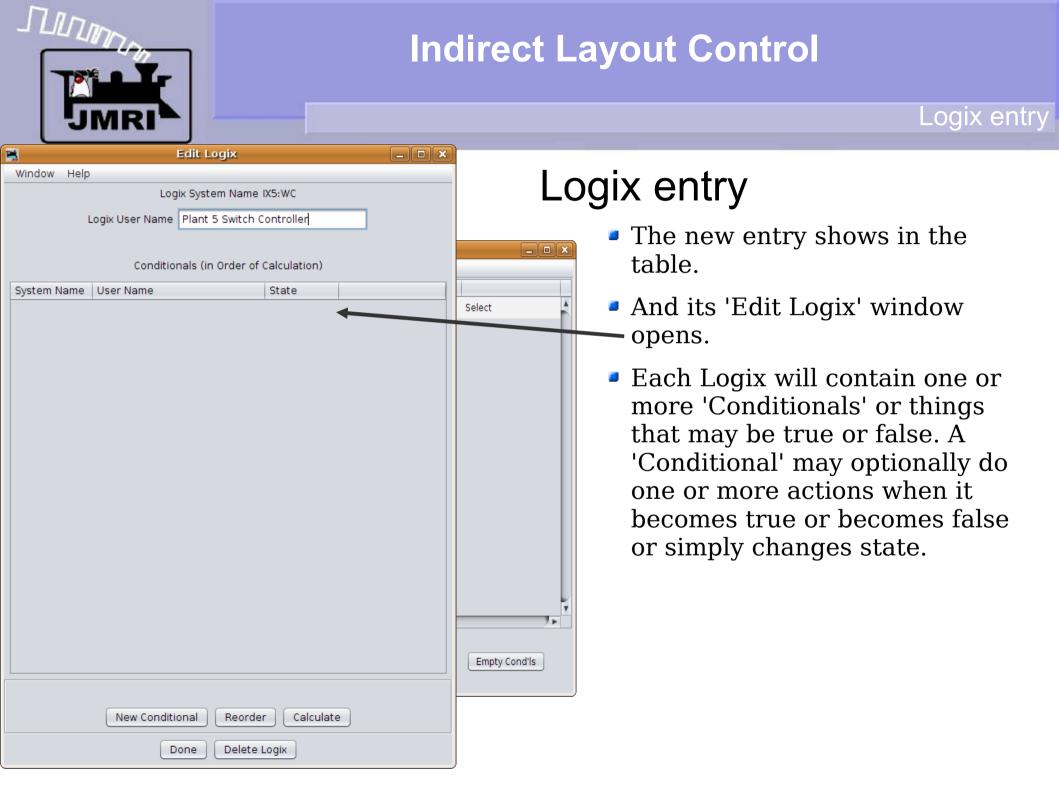
- The first information will be the ID. Logix are internal so the system name is 'I'. The item name is 'X', so they will start with 'IX'.
- We will call it IX5:WC

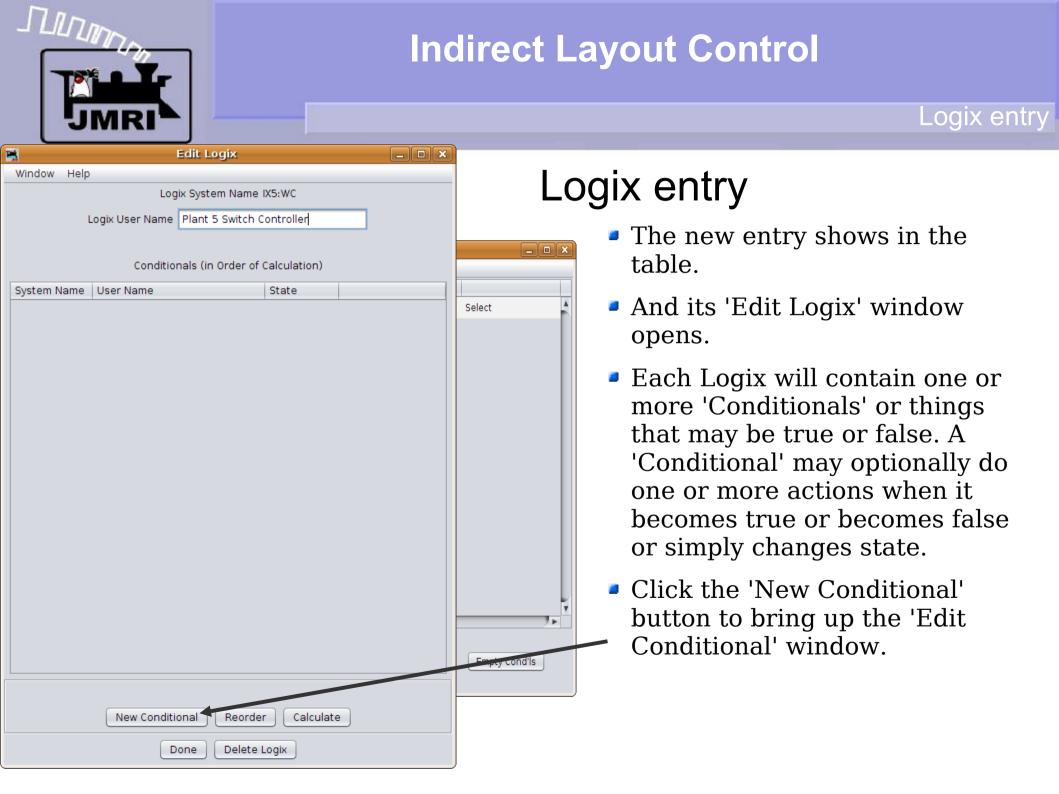
IX = Internal LogiX,
5: = Plant 5,
WC = sWitch Controller.

- Enter the user name "Plant 5 Switch Controller". All user names must be unique so we include the plant number.
- Once we have named our new creation, click on 'Create Logix' to add it to the table window, and open the Logix Editor.









Indirect L	ayout Control
JMRI	Logix entry
Edit Conditional     Window Help     Conditional System Name IX5:WCC1     Conditional User Name     Logical Expression:     Antecedent Variables (the 'if' part)     Row Oper Neg State Variable Description     State     Trigg     Add State Variable     Check State Variables   Logic Operator   AND	<ul> <li>Logix entry</li> <li>Note that JMRI automatically added 'C1' to the name we gave this item.</li> </ul>
Actions	
Consequent Actions (the 'then' part)          Action Description         Action Description         Add Action         Reorder         Update Conditional       Cancel         Delete Conditional	

Indirect L	ayout Control Logix entry
Edit Conditional	
Window Help Conditional System Name IX5:WCC1 Conditional User Name Switch 5 Normal Logical Expression: Antecedent Variables (the 'if' part) Row Oper Neg State Variable Description State Trigg	<ul> <li>Logix entry</li> <li>Note that JMRI automatically added 'C1' to the name we gave this item.</li> <li>Name this first one "Switch 5 Normal".</li> </ul>
Add State Variable Check State Variables Logic Operator AND Actions	
Consequent Actions (the 'then' part)          Action Description         Action Description         Add Action         Reorder         Update Conditional       Cancel         Delete Conditional	



Logix entry

*	Edit C	onditional			_ <b>_ _ X</b>
Window Help					
	Conditional	System Name IX5:	WCC1		
	Conditional User Name	Switch 5 Normal			
Logical Companyiant					
Logical Expression:					
	Anteceden	t Variables (the 'if'	part)		
Row Oper Neg S	State Variable Description		State	Trigg	
	Add State Varia	ole Check Stat	e Variable	s	
		ogic Operator			
		AND			
Actions					
	Consequent	Actions (the 'then	' part)		
Action Description					
	Add A	Action Reorder			
	Update Conditional	Cancel Del	lete Condit	tional	

- Note that JMRI automatically added 'C1' to the name we gave this item.
- Name this first one "Switch 5 Normal".
- We call the various items that will be checked by Logix 'Variables' because they 'vary' as things change on the layout. In this case between being true and being false. Click here to add our first one.



Logix entry

	1 <u> </u>					
<b>1</b>			Edit (	Conditional		
Wind	dow Help					
			Conditiona	l System Name IX5	:WCC1	
			Conditional User Name	Switch 5 Normal		
Logi	cal Expres	ssion:				
				nt Variables (the 'if	' part)	
Rov	v Oper	Neg	State Variable Description		State	Trigg
1.01	, obei	neg	State Vanable Beschption		State	
	-		Edit	Variable		_ <b> </b>
		eder	nt Variable			
				Variabl	le Type	
						-
Acti			Update	Cancel	Delete	
			<u> </u>			
Act	ion Descrip	otion				
			Add	Action Reorde	r ]	
			Update Conditional	Cancel De	elete Condi	itional

- Note that JMRI automatically added 'C1' to the name we gave this item.
- Name this first one "Switch 5 Normal".
- We call the various items that will be checked by Logix 'Variables' because they 'vary' as things change on the layout. In this case between being true and being false. Click here to add our first one.
- Click in the 'Variable Type' box
   to open a list of available options.



Logix entry

Edit Conditional         Window Help         Conditional System Name IX5:WCC1         Conditional User Name         Switch 5 Normal
Conditional System Name IX5:WCC1 Conditional User Name Switch 5 Normal
Conditional User Name Switch 5 Normal
Logical Expransion
LUQICALEXPLESSION
Antecedent Variables (the 'if' part)
Row         Oper         Neg         State         Trigg
Edit Variable 🗕 🗆 🗙
Antecedent Variable
Variable Type
Sensor Active
Acti Update Sensor Inactive 📉
Turnout Thrown
Action Description
Add Action Reorder
Update Conditional Cancel Delete Conditional

- Note that JMRI automatically added 'C1' to the name we gave this item.
- Name this first one "Switch 5 Normal".
- We call the various items that will be checked by Logix 'Variables' because they 'vary' as things change on the layout. In this case between being true and being false. Click here to add our first one.
- Click in the 'Variable Type' box to open a list of available options.
- Choose 'Sensor Inactive' so this will not happen unless a train is NOT on the OS.



	1				187					
i.				Edit C	onditional				_ • ×	
Wir	ndo	w Hel	D							
				Conditional	l System Name IX5:	WCC1				LOQ
				Conditional User Name	Switch 5 Normal					
Log	gica	l Expre	ssion:							Log
				Anteceden	nt Variables (the 'if'	part)				N N
R	ow	Oper	Neg	State Variable Description		State	Trigg			
_								I		W
										'U'
										pr
	P	1		Edit \	Variable					
		Anteo	eder	nt Variable				1		
					Variabl	е Туре				
							-			
					Sensor Activ					
Act				Update	Sensor Inact					
					Turnout Thro Turnout Clos					
A	tho	rbescr	ption			ea				
				Add A	Action Reorder	·				
						P !"	in a l			
				Update Conditional	Cancel De	lete Condif	lional			

## ogix entry

 Note we could have also said NOT 'Sensor Active' but that would include when it was 'Unknown' or 'Inconsistent', probably not good options here.



Logix entry

Edit Conditional	_ <b>— X</b>
Window Help	
Conditional System Name IX5:WCC1	
Conditional User Name Switch 5 Normal	
Logical Expression:	
Antecedent Variables (the 'if' part)	
Row         Oper         Neg         State Variable Description         State         Trigg	
Edit Variable 🗕	
Antecedent Variable	
Variable Type System / User Name	
Sensor Inactive  LS2	
Acti Update Cancel Delete	
Action Description	
Add Action Reorder	

- Note we could have also said NOT 'Sensor Active' but that would include when it was 'Unknown' or 'Inconsistent', probably not good options here.
- Enter the OS sensor ID. In this
  case it is 'LS2', then click 'Update' to add the variable.



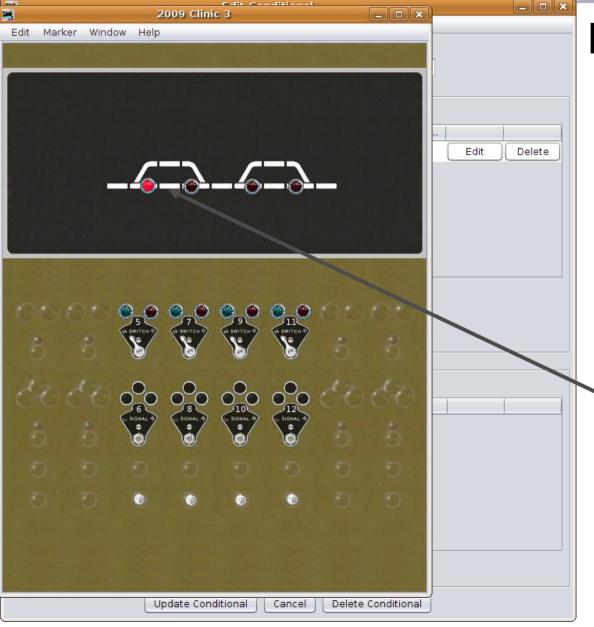
Logix entry

*			Edit Conditional	1			- • ×		
Window	w Help	)							
			Conditional System Na	me IX5:WCC1					
			Conditional User Name Switch 5 N	ormal					
Logica	l Expre	ssion:							
			Antecedent Variables	(the 'if' part)					
Row	Oper	Neg	State Variable Description	ble Description State Trigg.					
R1			Sensor, LS2, for Sensor Inactive	False	. 🗹 🌔	Edit	Delete		
			Add State Variable Che Logic Operat AND 🔻	ck State Variables tor	3				
Action	5		Concentrations (th	o the of nort)					
_			Consequent Actions (th	ie (then: part)					
Action	n Descri	ption							
			Add Action F	Reorder					
			Update Conditional Cancel	Delete Condit	ional				

- Note we could have also said NOT 'Sensor Active' but that would include when it was 'Unknown' or 'Inconsistent', probably not good options here.
- Enter the OS sensor ID. In this case it is 'LS2', then click 'Update' to add the variable.
  - The variable state is 'false'.



Logix entry



- Note we could have also said NOT 'Sensor Active' but that would include when it was 'Unknown' or 'Inconsistent', probably not good options here.
- Enter the OS sensor ID. In this case it is 'LS2', then click 'Update' to add the variable.
- The variable state is 'false'.
- Remember, we left the sensor 'Active'.



Logix entry

- 20	6			152								
*			Edit C	onditional				_ <b>— — ×</b>				
Windo	w Hel	р										
			Conditional	System Name IX5:	WCC1							
			Conditional User Name	Switch 5 Normal								
Logics	al Expre	accion:										
LUGIC		2551011.										
Antecedent Variables (the 'if' part)												
Row	Oper	Neg	State Variable Description		State	Trigg						
Rl			Sensor, LS2, for Sensor Ina	active	False		Edit	Delete				
					- ) (							
			Add State Variab	Check Stat	e variables	<u> </u>						
				ogic Operator								
				AND								
				AND								
Action	IS											
			Consequent	Actions (the 'then	' part)							
Actio	n Descr	intion										
ACLIO	n Descr	iption										
			Add A	ction Reorder	•							
			Update Conditional	Cancel De	lete Condit	ional						

- Note we could have also said NOT 'Sensor Active' but that would include when it was 'Unknown' or 'Inconsistent', probably not good options here.
- Enter the OS sensor ID. In this case it is 'LS2', then click 'Update' to add the variable.
- The variable state is 'false'.
- Remember, we left the sensor 'Active'.
- Add another variable for the code button.



Logix entry

1.0		182						
<u>.</u>	Edit Conditio	nal		_ <b>_ _ X</b>				
Window Help								
	Conditional System	Name IX5:WCC1						
	Conditional User Name Switch	5 Normal						
Logical Expression	:							
	Antecedent Variabl	les (the 'if' part)						
Row Oper Neg	State Variable Description	State	Trigg					
R1	Sensor, LS2, for Sensor Inactive							
				Delete				
	Endline A contradiction							
*	Edit Variable							
Antecedent Va	ariable							
Variable T	VDe							
Variable I	Syst	em / User Name						
Sensor Active								
Sensor Active	IS6:CB							
	Update Cancel	Delete						
	Add Action	Reorder						
	Update Conditional Cano	el Delete Conditi	onal					
	<u> </u>							

- Note we could have also said NOT 'Sensor Active' but that would include when it was 'Unknown' or 'Inconsistent', probably not good options here.
- Enter the OS sensor ID. In this case it is 'LS2', then click 'Update' to add the variable.
- The variable state is 'false'.
- Remember, we left the sensor 'Active'.
- Add another variable for the code button.
- If you forgot that it is IS6:CB simply hover the cursor over the button.



probably only want it to happen

when the code button is first

pressed, so un-check the first

Task		Edit Conditional				_ <b> </b>	
	leve tiele	Edit Conditional					
wind	dow Help						l ogiv ontru
		Conditional System Name IX5	Logix entry				
		Conditional User Name Switch 5 Normal					
Logi	cal Expression:		Both items are checked to				
		Antecedent Variables (the 'if	'trigger' the action. We				
Row	v Oper Neg S	State Variable Description					
R1	:	Sensor, LS2, for Sensor Inactive	False		Edit	Delete	probably only want it to ha
R2	AND	Sensor, IS6:CB, for Sensor Active	False	$\checkmark$	Edit	Delete	when the code button is fin
							pressed, so un-check the f
		Add State Variable Check Sta	te Variables				
		Logic Operator					
Actio	ons						
		Consequent Actions (the 'the					
Acti	ion Description						
		Add Action Reorde	r	ional			
							J



#### Logix entry

*				_	Edi	it Con	dition	al							×
Windo	w Hel	p													
				(	Conditio	onal Sy	ystem N	ame IX5	WCC1						
			Condi	tional U	Jser Nar	me Sv	witch 5 I	Normal		_					
Logica	al Expre	ession:													
					Antece	dent V	/ariables	s (the 'i	' part)						
Row	v Oper Neg State Variable Description State Trigg										1				
R1		S	Sensor, I	LS2, for	Sensor	- Inacti	ive		False		$\checkmark$	Edit		Delete	
R2	AND	s	Sensor, I	S6:CB, 1	for Sens	sor Act	tive		False		$\checkmark$	 Edit		Delete	
			- (	Add S	State Va	riable		eck Sta	te Variab	les					
						Log	jic Opera	ator							
							AND	<b>_</b>							
Action	IS														
				C	onsequ	ient Ac	ctions (t	the 'the	n' part)						
Action	n Descr	ription													
					Ac	dd Acti	ion	Reorde	r						
			Up	date Co	ondition	al	Cancel		elete Con	ditio	onal				

- Both items are checked to 'trigger' the action. We probably only want it to happen when the code button is first pressed, so un-check the first one.
- We also only need to send the command if the turnout is **not** already in position, so lets add another variable for that.



#### Logix entry

<u> </u>										
<b>**</b>	Edit Conditional				_ <b>•</b> ×					
Window Help										
	Conditional System Name	e IX5:WCC1								
	Conditional User Name Switch 5 Nor	mal								
Logical Expression:										
	Antecedent Variables (th	ne 'if' part)								
Row Oper Neg	State Variable Description	State	Trigg							
R1	Sensor, LS2, for Sensor Inactive	False	J	Edit	Delete					
R2 AND	R2 AND Sensor, IS6:CB, for Sensor Active False 🗹 Edit Delete									
1	Edit Variable				×					
Antecedent V	/ariable									
	Variab	own								
	Update Turnout Clos Conditional	True								
Conditional False										
	Add Action Red	order								
	Update Conditional Cancel	Delete Condi	tional							

- Both items are checked to 'trigger' the action. We probably only want it to happen when the code button is first pressed, so un-check the first one.
- We also only need to send the command if the turnout is **not** already in position, so lets add another variable for that.
- We check on 'Turnout Closed'



#### Logix entry

Edit Conditional										
Window	w Help	)								
			Conditional	System Name IX5:	WCC1					
			Conditional User Name	Switch 5 Normal						
Logica	l Expre	ssion:								
			Anteceden	t Variables (the 'if'	part)					
Row	Oper	Neg	State Variable Description		State	Trigg				
Rl			Sensor, LS2, for Sensor Ina	active	False		Edit	Delete		
R2	AND		Sensor, IS6:CB, for Sensor	Active	False	$\checkmark$	Edit	Delete		
R3	AND		Turnout, LT1, for Turnout C	losed	False		Edit	Delete		
Action		ption		Check Stat		s				
Action Description										
			Update Conditional	Cancel De	lete Condii	tional				

# Logix entry

- Both items are checked to 'trigger' the action. We probably only want it to happen when the code button is first pressed, so un-check the first one.
- We also only need to send the command if the turnout is **not** already in position, so lets add another variable for that.

We check on 'Turnout Closed'

But see that it is 'false'. We need it to be 'true' for our logic to work.



#### Logix entry

<b>1</b>				Edit Co	nditional				_ <b>_ X</b>		
Windo	ow Hel	lp									
				Conditional S	ystem Name IX	5:WCC1					
			Conditional	User Name S	witch 5 Normal						
Logica	al Expre	ession:									
				Antecedent \	Variables (the 'i	f' part)					
Row	Oper	Neg	State Variable	Description		State	Trigg				
Rl			Sensor, LS2, fo	or Sensor Inact	tive	False		Edit	Delete		
R2	R2 AND Sensor, IS6:CB, for Sensor Active False 🗹 Edit Delete										
R3	R3 AND Turnout, LT1, for Turnout Closed False Edit Delete										
Action	Add State Variable Check State Variables										
Action	13			Consequent A	ctions (the 'the	p'part)					
Actio	n Descr	rintion									
	Add Action Reorder										
			Update (	Conditional	Cancel D	elete Condit	ional				

- Both items are checked to 'trigger' the action. We probably only want it to happen when the code button is first pressed, so un-check the first one.
- We also only need to send the command if the turnout is **not** already in position, so lets add another variable for that.
- We check on 'Turnout Closed'
- But see that it is 'false'. We need it to be 'true' for our logic to work.
- If we click just to the left of the description we discover that we can add a 'NOT' to the variable.



Logix entry

-			Edit Conditional			_ <b>_</b> ×	
	ow Hel	lp				رکار کار کا	
			Conditional System Nam	a IX5:WCC1			Logix entry
			Conditional User Name Switch 5 Nor	mal			
Logi	al Expr	ession	:				Now our statements seem to be
			Antecedent Variables (t	ne 'if' part)			what we need Oh yes, we only
Row	Oper	Neg	State Variable Description	State	Trigg		
R1			Sensor, LS2, for Sensor Inactive	False		Edit Delete	want to do this if the lever tells
R2	AND		Sensor, IS6:CB, for Sensor Active	False	V	Edit Delete	us to.
R3	AND	NOT	Turnout, LT1, for Turnout Closed	True		Edit Delete	
				01-1-1 (			
				: State Variable	es		
			Logic Operato	,			
			AND				
Actio	ns			14 h =			
			Consequent Actions (the	then part)			
Acti	on Descr	ription					
			Add Action Re	order			
			Update Conditional Cancel	Delete Cond	litional		
							·



#### Logix entry

P				Edit Conditional			_ <b>_ \</b>			
_	Windo	w Help	0							
				Conditional System Name IX5	WCC1					
				Conditional User Name Switch 5 Normal						
L	ogica	l Expre	ssion:							
				Antecedent Variables (the 'if	" part)					
[	Row	Oper	Neg	State Variable Description	State	Trigg				
	R1	oper	neg	Sensor, LS2, for Sensor Inactive	False		Edit Delete			
	R2	AND		Sensor, IS6:CB, for Sensor Active	False	<b>v</b>	Edit Delete			
			NOT							
	R3	AND	NOT	Turnout, LT1, for Turnout Closed	True		Edit Delete			
				Add State Variable Check Sta	te Variable:	5				
						_				
				Logic Operator						
L				AND						
A	ction	S								
				Consequent Actions (the 'the	n' part)					
	Actior	n Descri	ption							
				Add Action Reorde	r					
				Update Conditional Cancel De	elete Condit	ional				

- Now our statements seem to be what we need... Oh yes, we only want to do this if the lever tells us to.
- Add another variable for the lever's position.



Logix entry

1					Edit O	Conditional				_ O ×	3
١	Vindo	w Help	)								
					Conditiona	l System Name IX5:	WCC1				
				Condition	al User Name	Switch 5 Normal					
L	ogica	l Expre	ssion:								
					Anteceder	nt Variables (the 'if'	part)				
	Row	Oper	Neg	State Variab	le Description		State	Trigg			
	Rl			Sensor, LS2,	for Sensor In	active	False		Edit	Delete	
	R2	AND		Sensor, IS6:0	CB, for Sensor	Active	False	$\checkmark$	Edit	Delete	
	R3	AND	NOT	Turnout, LT1	, for Turnout (	Closed	True		Edit	Delete	
Ġ	<b>1</b>				Edit	Variable				- 0 ×	1
		ecede	ent V	ariable							
Γ											1
		Vari	able 1	Гуре		Custom	( Lleer N	lama			
	_					System	/ User N	lame			
	Sen	isor A	ctive	•	IS5:WL						
				_				_			
					Update	Cancel	Delete				
					Add	Action Reorder					
				Update	e Conditional	Cancel De	lete Condit	ional			

- Now our statements seem to be what we need... Oh yes, we only want to do this if the lever tells us to.
- Add another variable for the lever's position.
- IS5:WL (5: sWitch Lever)



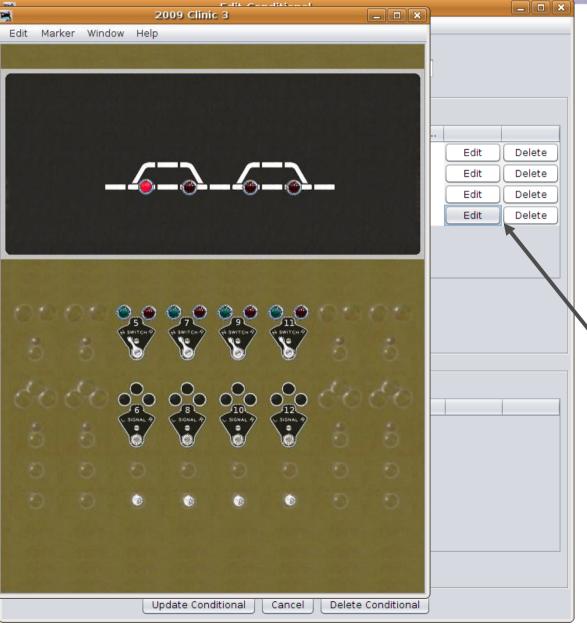
#### Logix entry

1				Edit Conditional				_ <b>_ _ X</b>
_	Windo	w Help	)					
				Conditional System Name	IX5:WCC1			
				Conditional User Name Switch 5 Norm	nal			
L	ogica	l Expre	ssion:					
				Antecedent Variables (the	e 'if' part)			
	Row	Oper	Neg	State Variable Description	State	Trigg		
	Rl			Sensor, LS2, for Sensor Inactive	False		Edit	Delete
	R2	AND		Sensor, IS6:CB, for Sensor Active	False	$\checkmark$	Edit	Delete
	R3	AND	NOT	Turnout, LT1, for Turnout Closed	True		Edit	Delete
	R4	AND		Sensor, IS5:WL, for Sensor Active	True		Edit	Delete
				Add State Variable Check S	State Variable	s		
				Logic Operator				
				AND				
	ction	5						
ſ		<u> </u>		Consequent Actions (the 't	hen' part)			
	Action	Descri	otion					
	Action	n Descri	ption					
				Add Action Reo	rder			
				Update Conditional Cancel	Delete Condi	tional		

- Now our statements seem to be what we need... Oh yes, we only want to do this if the lever tells us to.
- Add another variable for the lever's position.
- IS5:WL (5: sWitch Lever)
- Note that you may Delete or Edit any variable at any time.



Logix entry



- Now our statements seem to be what we need... Oh yes, we only want to do this if the lever tells us to.
- Add another variable for the lever's position.
- IS5:WL (5: sWitch Lever)
- Note that you may Delete or Edit any variable at any time.
- Bring the panel to the front and 'move' the train off of the OS section. (by clicking the sensor)



#### Logix entry

<b>1</b>			Edit Conditional				_ <b>— X</b>		
Window	v Help	D							
			Conditional System Name	e IX5:WCC1					
			Conditional User Name Switch 5 Nor	mal					
Logical	l Expre	ssion:							
			Antecedent Variables (th	ne 'if' part)					
Row	Oper	Neg	State Variable Description	State	Trigg				
R1			Sensor, LS2, for Sensor Inactive	False		Edit	Delete		
R2 AND Sensor, IS6:CB, for Sensor Active False 🗹 Edit Delete									
R3	AND	NOT	Turnout, LT1, for Turnout Closed	True		Edit	Delete		
R4	AND		Sensor, IS5:WL, for Sensor Active	True		Edit	Delete		
Actions	5		Logic Operator		s				
			Consequent Actions (the	'then' part)					
Action	Descri	ption							
			Add Action Red Update Conditional Cancel	Delete Condi	tional				

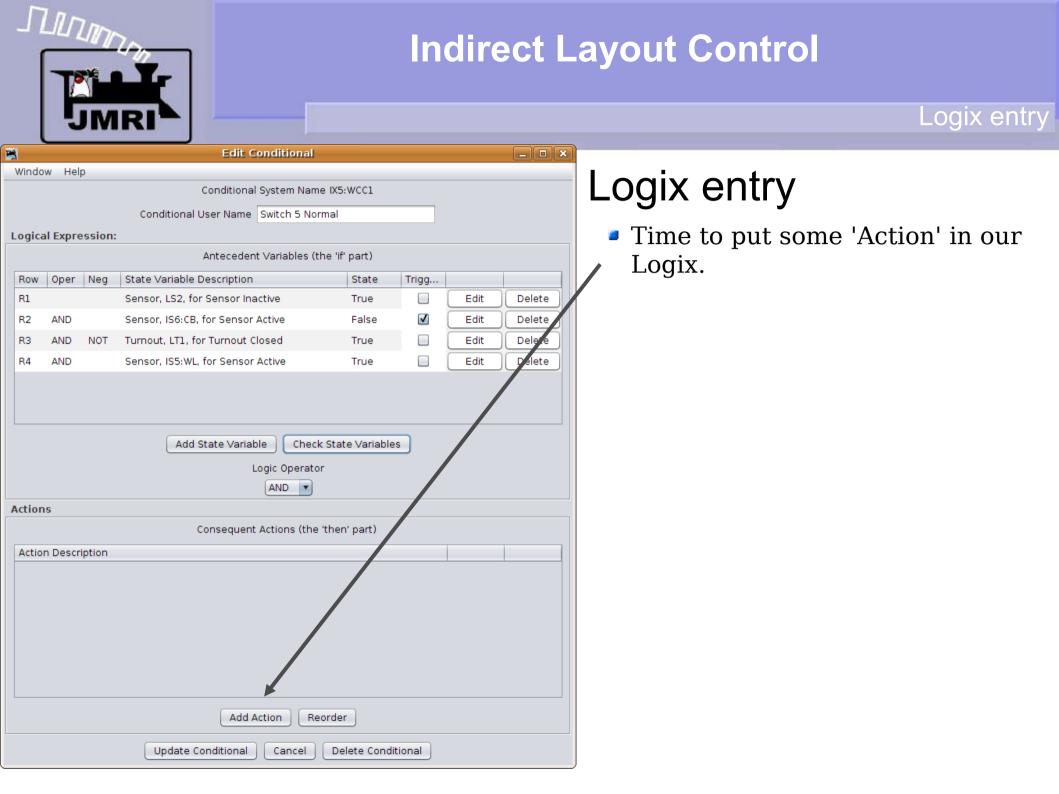
- Now our statements seem to be what we need... Oh yes, we only want to do this if the lever tells us to.
- Add another variable for the lever's position.
- IS5:WL (5: sWitch Lever)
- Note that you may Delete or Edit any variable at any time.
- Bring the panel to the front and 'move' the train off of the OS section. (by clicking the sensor)
- Now 'Check State Variables' to re-read the layout.



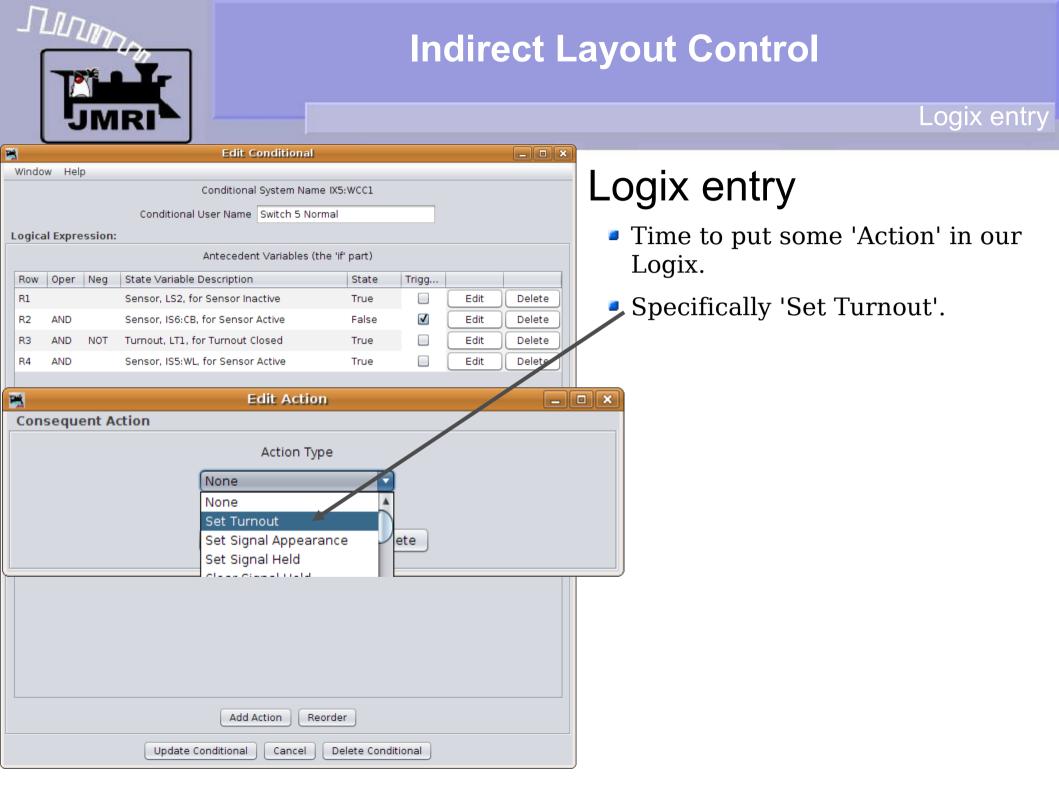
#### Logix entry

Window Help  Conditional System Name IXS:WCC1 Conditional User Name Switch 5 Normal  Logical Expression:  Antecedent Variables (the 'If part)  Row Oper Neg State Variable Description State Trigg R1 Sensor, L52, for Sensor Inactive True Edit Delete R2 AND Sensor, IS6:CB, for Sensor Active False Edit Delete R3 AND NOT Turnout, LT1, for Turnout Closed True Edit Delete R4 AND Sensor, IS5:WL, for Sensor Active True Edit Delete Logic Operator AND  Actions  Consequent Actions (the 'then' part)  Action Description  Add Action Reorder  Update Conditional Cancel Delete Conditional	P <u>1</u>			Edit Conditional	1			_ 🗆 🗙		
Logical Expression:         Antecedent Variables (the 'if' part)         Row Oper Neg State Variable Description         R1       Sensor, LS2, for Sensor Inactive         R2       AND         Sensor, LS5: (CB, for Sensor Active         False       Edit         Delete         R3       AND         NOT       Turmout, LT1, for Turmout Closed         R4       AND         Sensor, IS5: WL, for Sensor Active       True         Edit       Delete         R4       AND         Sensor, IS5: WL, for Sensor Active       True         Edit       Delete         Logic Operator       AND         AND       Consequent Actions (the 'then' part)         Action Description       Add Action         Add Action       Reorder	Windo	w Hel	р							
Logical Expression:  Antecedent Variables (the 'if' part)  Row Oper Neg State Variable Description State Trigg R1 Sensor, IS5, for Sensor Inactive True Edit Delete R2 AND Sensor, IS5, iCB, for Sensor Active False Edit Delete R3 AND NOT Turnout, LT1, for Turnout Closed True Edit Delete R4 AND Sensor, IS5, WL, for Sensor Active True Edit Delete Logic Operator AnD  Actions  Consequent Actions (the 'then' part)  Action Description  Add Action Reorder				Conditional System Nat	me IX5:WCCl					
Antecedent Variables (the 'if' part)          Row Oper Neg State Variable Description       State Trigg         R1       Sensor, LS2, for Sensor Inactive       True         R2       AND       Sensor, IS6:CB, for Sensor Active       False         R3       AND       NOT       Turnout, LT1, for Turnout Closed       True       Edit       Delete         R4       AND       Sensor, IS5:WL, for Sensor Active       True       Edit       Delete         R4       AND       Sensor, IS5:WL, for Sensor Active       True       Edit       Delete         R4       AND       Sensor, IS5:WL, for Sensor Active       True       Edit       Delete         Add State Variable       Check State Variables       Logic Operator       AND       AND       AND         Actions       Consequent Actions (the 'then' part)       Action Description       Add Action       Reorder				Conditional User Name Switch 5 No	ormal					
Row       Oper       Neg       State Variable Description       State       Trug       Edit       Delete         R1       Sensor, LS2, for Sensor Inactive       Trug       Edit       Delete         R2       AND       Sensor, IS6:CB, for Sensor Active       False       Edit       Delete         R3       AND       NOT       Turnout, LT1, for Turnout Closed       Trug       Edit       Delete         R4       AND       Sensor, IS5:WL, for Sensor Active       Trug       Edit       Delete         R4       AND       Sensor, IS5:WL, for Sensor Active       Trug       Edit       Delete         Add State Variable       Check State Variables       Logic Operator       AND       AND       AND         Actions       Consequent Actions (the 'then' part)       Action Description       Add Action       Reorder	Logica	al Expre	ssion:							
R1       Sensor, L52, for Sensor Inactive       True       Edit       Delete         R2       AND       Sensor, I56:CB, for Sensor Active       False       Edit       Delete         R3       AND       NOT       Turnout, LT1, for Turnout Closed       True       Edit       Delete         R4       AND       Sensor, IS5:WL, for Sensor Active       True       Edit       Delete         R4       AND       Sensor, IS5:WL, for Sensor Active       True       Edit       Delete         Add State Variable       Check State Variables       Logic Operator       AND       AND         Actions       Consequent Actions (the 'then' part)       Action Description       Add Action       Reorder				Antecedent Variables	(the 'if' part)					
R2       AND       Sensor, IS6:CB, for Sensor Active       False       Edit       Delete         R3       AND       NOT       Turnout, LT1, for Turnout Closed       True       Edit       Delete         R4       AND       Sensor, IS5:WL, for Sensor Active       True       Edit       Delete         Add State Variable       Check State Variables       Logic Operator       AND       AND         Actions       Consequent Actions (the 'then' part)       Action Description       Add Action       Reorder	Row	Oper	Neg	State Variable Description	State	Trigg				
R3 AND NOT Turnout, LTI, for Turnout Closed True Edit Delete R4 AND Sensor, IS5:WL, for Sensor Active True Edit Delete Add State Variable Check State Variables Logic Operator AND  Actions Consequent Actions (the 'then' part) Action Description Add Action Reorder	Rl			Sensor, LS2, for Sensor Inactive	True		Edit	Delete		
R4 AND Sensor, IS5:WL, for Sensor Active True Edit Delete  Add State Variable Check State Variables Logic Operator AND Actions Consequent Actions (the 'then' part) Action Description Action Reorder	R2 AND Sensor, IS6:CB, for Sensor Active False 🗹 Edit Delete									
Add State Variable Check State Variables Logic Operator AND Consequent Actions (the 'then' part) Action Description Action Description Add Action Reorder	R3	AND	NOT	Turnout, LT1, for Turnout Closed	True		Edit	Delete		
Actions Consequent Actions (the 'then' part) Action Description Action Consequent Actions (the 'then' part) Action Description Add Action Reorder	R4	AND		Sensor, IS5:WL, for Sensor Active	True		Edit	Delete		
Action Description	Action	15		AND						
Add Action Reorder	Action	n Descr	intion	· ·						
						ional				

- Now our statements seem to be what we need... Oh yes, we only want to do this if the lever tells us to.
- Add another variable for the lever's position.
- IS5:WL (5: sWitch Lever)
- Note that you may Delete or Edit any variable at any time.
- Bring the panel to the front and 'move' the train off of the OS section. (by clicking the sensor)
- Now 'Check State Variables' to re-read the layout.
- Everything is now "true" but the button itself.



	Indirect Layout Control Logix entry
Edit Conditional	
Window Help Conditional System Name	Logiv optrv
Conditional User Name Switch 5 Norm	al
Logical Expression: Antecedent Variables (the	Time to put some 'Action' in our I ogiv
Row Oper Neg State Variable Description	State Trigg Logix.
R1 Sensor, LS2, for Sensor Inactive	True Edit Delete
R2 AND Sensor, IS6:CB, for Sensor Active	False 🗹 Edit Delete
R3 AND NOT Turnout, LT1, for Turnout Closed	True Edit Delete
R4 AND Sensor, IS5:WL, for Sensor Active	True Edit Delete
Edit Action	
Consequent Action Action Type None Update Cance	
Add Action Reor Update Conditional Cancel	rder Delete Conditional





	JI			Logix enti	У
8		Edit Conditional			
	v Help				
		Conditional System Nam	ne IX5:WCC1	Logix entry	
		Conditional User Name Switch 5 No	rmal		
Logical	Expression	on:		Time to put some 'Action' in our	
		Antecedent Variables (t	the 'if' part)	Logix.	
Row	Oper Ne	g State Variable Description	State Trigg		
R1		Sensor, LS2, for Sensor Inactive	True 🗌	Edit Delete Specifically 'Set Turnout'. Enter	
R2	AND	Sensor, IS6:CB, for Sensor Active	False 🗹		
R3	AND NO	T Turnout, LT1, for Turnout Closed	True 📃	Edit Delete 'LT1', then 'Update'.	
R4	AND	Sensor, IS5:WL, for Sensor Active	True 🗌	Edit Delete	
<b>1</b>		Edit Acti	ion		
Cons	equent	Action			
	Actio	n Type Change Op	tion System	User Name Turnout Position	
Cot T	urnout	On Change To		Closed V	
Laer	umout	Un change to		Closed	
		Update C	ancel Delete		
		Add Action Re	eorder		
		Update Conditional Cancel	Delete Conditional		



Logix entry

<u>.</u>	Edit Conditional				_ <b>—</b> ×				
Window Help									
	Conditional System Nam	ne IX5:WCC1							
	Conditional User Name Switch 5 No	rmal							
Logical Expression:									
	Antecedent Variables (	the 'if' part)							
Row Oper Neg	State Variable Description	State	Trigg						
R1	Sensor, LS2, for Sensor Inactive	True		Edit	Delete				
R2 AND Sensor, IS6:CB, for Sensor Active False 🗹 Edit Delete									
R3 AND NOT	Turnout, LT1, for Turnout Closed	True		Edit	Delete				
R4 AND	Sensor, IS5:WL, for Sensor Active	True		Edit	Delete				
Actions	Add State Variable Chec Logic Operato AND T Consequent Actions (the								
Action Description									
On Change To True,	Set Turnout, LT1 to Closed			Edit	Delete				
Add Action Reorder									
	Update Conditional Cancel	Delete Condit	ional						

- Time to put some 'Action' in our Logix.
- Specifically 'Set Turnout'. Enter 'LT1', then 'Update'.
- Our first action.



#### Logix entry

<b>M</b>	Edit Conditional			>
Window Help				
	Conditional System Nan	ne IX5:WCC1		
	Conditional User Name Switch 5 No	rmal		
Logical Expression:				
	Antecedent Variables (	the 'if' part)		
Row Oper Neg	State Variable Description	State	Trigg	
RI	Sensor, LS2, for Sensor Inactive	True		Edit Delete
R2 AND	Sensor, IS6:CB, for Sensor Active	False	$\checkmark$	Edit Delete
R3 AND NOT	Turnout, LT1, for Turnout Closed	True		Edit Delete
R4 AND	Sensor, IS5:WL, for Sensor Active	True		Edit Delete
Actions	Add State Variable Chec Logic Operate AND Consequent Actions (the		s	
Action Description				
	Set Turnout, LT1 to Closed			Edit Delete
	Add Action Re Update Conditional Cancel	eorder Delete Condit		

- Time to put some 'Action' in our Logix.
- Specifically 'Set Turnout'. Enter 'LT1', then 'Update'.
- Our first action.
- Lets get fancy and send the sound of the code relays. Add another action 'Play Sound File'.



Logix entry

Edit Conditional				_ <b> </b>	
Window Help					
Conditional System Name IX5	WCC1				
Conditional User Name Switch 5 Normal					
Logical Expression:					
Antecedent Variables (the 'if	' part)				
Row Oper Neg State Variable Description	State	Trigg			
R1 Sensor, LS2, for Sensor Inactive	True		Edit	Delete	
R2 AND Sensor, IS6:CB, for Sensor Active	False	$\checkmark$	Edit	Delete	
Edit Action					
Consequent Action					
Action Type	Cha	ange O	ption		
Play Sound File	On Cha	nge To	True 🔻	1	
Set File Acti	on Data				
File					
Click for a f	ile selectio		og for cho	osing a so	und fil
Consequent Actions (the 'the					1
Action Description					
On Change To True, Set Turnout, LT1 to Closed			Edit	Delete	
Add Action Reorde	r				
Update Conditional Cancel De	elete Conditio	nal			

# \_ogix entry

- Time to put some 'Action' in our Logix.
- Specifically 'Set Turnout'. Enter 'LT1', then 'Update'.

Our first action.

Lets get fancy and send the sound of the code relays. Add another action 'Play Sound File'.

Click on 'File' to search for sounds.



Logix entry

<b>1</b>	Edit Conditional	J	
Window Help			
	Conditional System Nar	me IX5:WCC1	
	Conditional User Name Switch 5 No	ormal	
Logical Expression:			
	Antecedent Variables	(the 'if' part)	
Row Oper Neg S	State Variable Description	State Trigg	
R 🔬	Open	×	
B Look In: 📋 s	sounds	1 🝙 👔 🗐	e - • ×
C O O O O O O O O O O O O O O O O O O O	ess.wav 📄 pull-on.wav lease.wav 📄 push-off.wav		a sound file
	Add Action R	Reorder	
	Update Conditional Cancel	Delete Conditional	

# \_ogix entry

- Time to put some 'Action' in our Logix.
- Specifically 'Set Turnout'. Enter 'LT1', then 'Update'.

Our first action.

Lets get fancy and send the sound of the code relays. Add another action 'Play Sound File'.

- Click on 'File' to search for sounds.
- 'Code-send.wav" is what we need. Select then 'Open'.



Logix entry

	Edit Conditional				- • ×	
Window Help						
	Conditional System Name I	X5:WCC1				
	Conditional User Name Switch 5 Norma	al				
Logical Expression	:					
	Antecedent Variables (the	'if' part)				
Row Oper Neg	State Variable Description	State	Trigg			
Rl	Sensor, LS2, for Sensor Inactive	True		Edit )	Delete	
R2 AND	Sensor, IS6:CB, for Sensor Active	False		Edit	Delete	
	Trumpark 171 fee Trumpark Classed	T		r da	Dalata	
<b>*</b>	Edit Action				_ 0	1
Consequent A	Action					
	Action Type	Cł	nange Op	tion		
P	ay Sound File	On Ch	ange To	True 💌		
_						
Set File	Ac	tion Data				
File /u	sr/local/JMRI/resources/sounds/Cod	e-send.wav	/			
	Update Cance	el Dele	ate			
<u>μ</u>						
Action Description						
	, Set Turnout, LT1 to Closed			Edit	Delete	
	Add Action Reor	der				~
	Update Conditional Cancel	Delete Condit	ional			

# \_ogix entry

- Time to put some 'Action' in our Logix.
- Specifically 'Set Turnout'. Enter 'LT1', then 'Update'.

Our first action.

Lets get fancy and send the sound of the code relays. Add another action 'Play Sound File'.

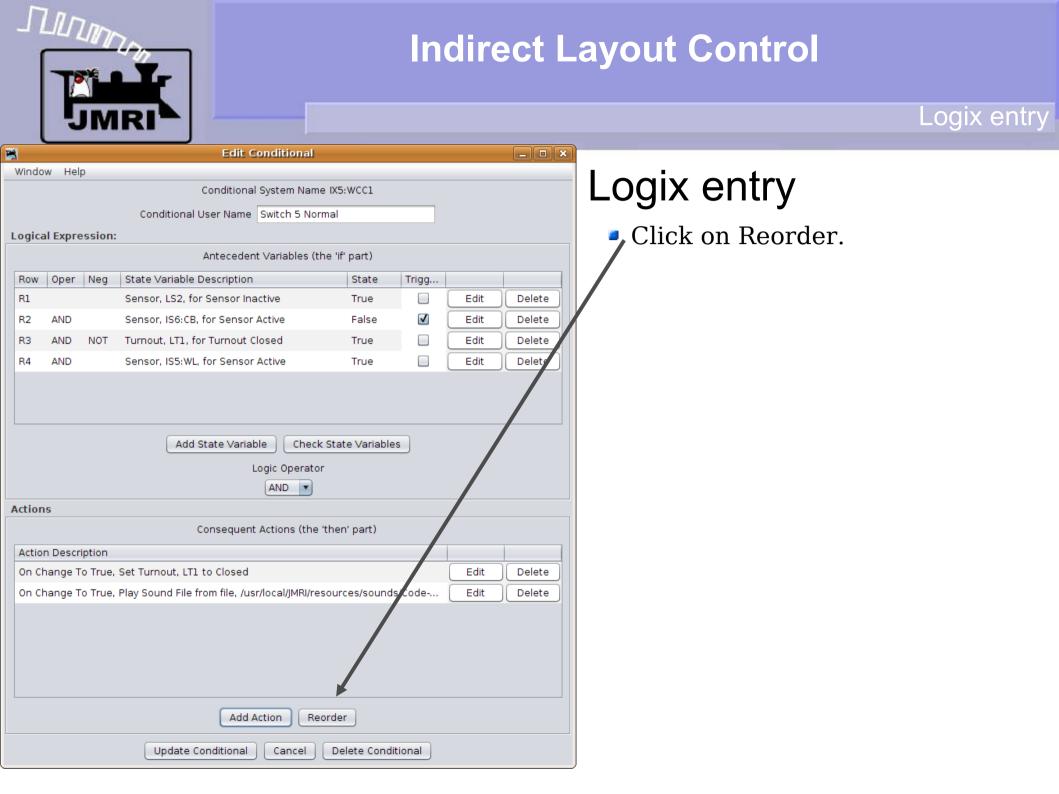
- Click on 'File' to search for sounds.
- 'Code-send.wav" is what we need. Select then 'Open'.
- Click 'Update' to enter the new action.



#### Logix entry

	<u> </u>			Calif.					
1			Edit Condition	nal			_ <b> </b>		
Wind	dow Hel	D							
	Conditional System Name IX5:WCC1								
			Conditional User Name Switch 5	5 Normal					
Logi	cal Expre	ssion:							
			Antecedent Variable	es (the 'if' part)					
Row	v Oper	Neg	State Variable Description	State	Trigg				
R1			Sensor, LS2, for Sensor Inactive	True		Edit	Delete		
R2	AND		Sensor, IS6:CB, for Sensor Active	False	$\checkmark$	Edit	Delete		
R3	AND	NOT	Turnout, LT1, for Turnout Closed	True		Edit	Delete		
R4	AND		Sensor, IS5:WL, for Sensor Active	True		Edit	Delete		
			Add State Variable C	heck State Variables					
			Logic Ope	erator					
				•					
Actio	ns								
			Consequent Actions	(the 'then' part)					
Acti	ion Descr	iption							
On	Change T	o True,	Set Turnout, LT1 to Closed			Edit	Delete		
On	Change T	o True,	Play Sound File from file, /usr/local/JM	RI/resources/sounds/	Code	Edit	Delete		
			Add Action	Reorder					
			Update Conditional Canc	el Delete Conditi	onal				

- Time to put some 'Action' in our Logix.
- Specifically 'Set Turnout'. Enter 'LT1', then 'Update'.
- Our first action.
- Lets get fancy and send the sound of the code relays. Add another action 'Play Sound File'.
- Click on 'File' to search for sounds.
- 'Code-send.wav" is what we need. Select then 'Open'.
- Click 'Update' to enter the new action.
- The sounds should be first. :(





		JN	IRI	►  —						Logix entry
<b>1</b>				Edit	Conditional				_ <b>-</b> ×	
Windo	N H	elp								Logiv ontry
					al System Name					Logix entry
				itional User Name	e Switch 5 Norm	al	_			
Logica	l Exp	ressior	1							Click on Reorder.
					ent Variables (the	-				Then choose who is on first
Row R1	Oper	r Neg		riable Description		State True	Trigg	Edit	Delete	Then choose who is on first.
R2	AND			IS6:CB, for Senso		False	V	Edit	Delete	
R3	AND			LT1, for Turnout		True		Edit	Delete	
R4	AND			IS5:WL, for Senso		True		Edit	Delete	
				Add State Varia	able Check S Logic Operator	State Variabl	es			
Action	5									
				Consequer	nt Actions (the 'th	hen' part)				
		cription	0.17							
	-			ut, LT1 to Closed	ı usr/local/JMRI/reso		le/Codo	Edit	First	
					Action Reor					



Logix entry

- 10				1556				
ŝ			Edit Co	nditional				_ <b>-</b> ×
Windo	w Hel	p						
			Conditional S	System Name IX5:	WCC1			
			Conditional User Name	Switch 5 Normal				
ogica	al Expre	ession:						
			Antecedent	Variables (the 'if'	part)			
Row	Oper	Neg	State Variable Description		State	Trigg		
Rl			Sensor, LS2, for Sensor Inac	tive	True		Edit	Delete
R2	AND		Sensor, IS6:CB, for Sensor A	ctive	False	$\checkmark$	Edit	Delete
R3	AND	NOT	Turnout, LT1, for Turnout Clo	sed	True		Edit	Delete
R4	AND		Sensor, IS5:WL, for Sensor A	ctive	True		Edit	Delete
			Add State Variable	e Check Stat	e Variables	5		
			Lo	gic Operator				
				AND				
Action	15							
			Consequent A	Actions (the 'then	' part)			
Actio	n Descr	intion						
		•	Play Sound File from file, /usr/	local/IMBI/resourc	es/sounds	/Code-	Edit	
			Set Turnout, LT1 to Closed	iocal, ji na, resourc			Edit	Next
on ci	nangen	o nue,	Set fulfiout, El1 to closed					
			Add Ac	tion Reorder				
			Update Conditional	Cancel De	lete Condit	ional		
					and borrait			

- Click on Reorder.
- Then choose who is on first.
- Continue picking the next item until you have the new order.



#### Logix entry

10 million -			1236				
1		Edit C	onditional				_ <b> </b>
Window H	elp						
		Conditional	System Name IX5:	WCC1			
	Con	ditional User Name	Switch 5 Normal				
Logical Exp	ression:						
		Antecedent	Variables (the 'if'	part)			
Row Ope	r Neg State \	/ariable Description		State	Trigg		
Rl	Sensor	r, LS2, for Sensor Ina	ctive	True		Edit	Delete
R2 AND	Sensor	r, IS6:CB, for Sensor /	Active	False	$\checkmark$	Edit	Delete
R3 AND	NOT Turnou	ut, LT1, for Turnout Cl	osed	True		Edit	Delete
R4 AND	Sensor	r, IS5:WL, for Sensor	Active	True		Edit	Delete
		Add State Variab	le Check Stat	e Variables			
		L	ogic Operator		_		
Actions							
		Consequent	Actions (the 'then	' part)			
Action Des	cription			-			
		und File from file, /usr	/local/JMRI/resourc	es/sounds/	/Code	Edit	1
	-	nout, LT1 to Closed				Edit	Next
		Add A	ction Reorder				
	L	Jpdate Conditional	Cancel De	lete Conditi	ional		

- Click on Reorder.
- Then choose who is on first.
- Continue picking the next item until you have the new order.
- One action does not wait on another to happen, so the turnout will still activate before the sounds are finished playing. To fix that hit 'Edit' on the turnout entry.



Logix entry

1	Edit Conditional		
Window Help	2		Logiv optru
	Conditional System Name IX5:WCC1		Logix entry
	Conditional User Name Switch 5 Normal		
Logical Expre	Antecedent Variables (the 'if' part)		Click on Reorder.
Row Oper		Trigg	Then choose who is on first.
R1	Sensor, LS2, for Sensor Inactive True	Edit Delete	
R2 AND	Sensor, IS6:CB, for Sensor Active False	Edit Delete	Continue picking the next item
<b>*</b>	Edit Action		ntil you have the new order.
Conseque	ent Action		
A Set Turnou	ut On Change To True  Update Cancel D Consequent Actions (the "then" part)	System / Oser Name	ne action does not wait on nother to happen, so the irnout will still activate before he sounds are finished playing. o fix that hit 'Edit' on the turnout entry.
On Change To	o True, Play Sound File from file, /usr/local/JMRI/resources/sounds o True, Set Turnout, LT1 to Closed	s/Code Edit 1 Edit Next	Change 'Set Turnout' to 'Delayed Set Turnout'.
	Add Action Reorder		
	Update Conditional Cancel Delete Condit	tional	

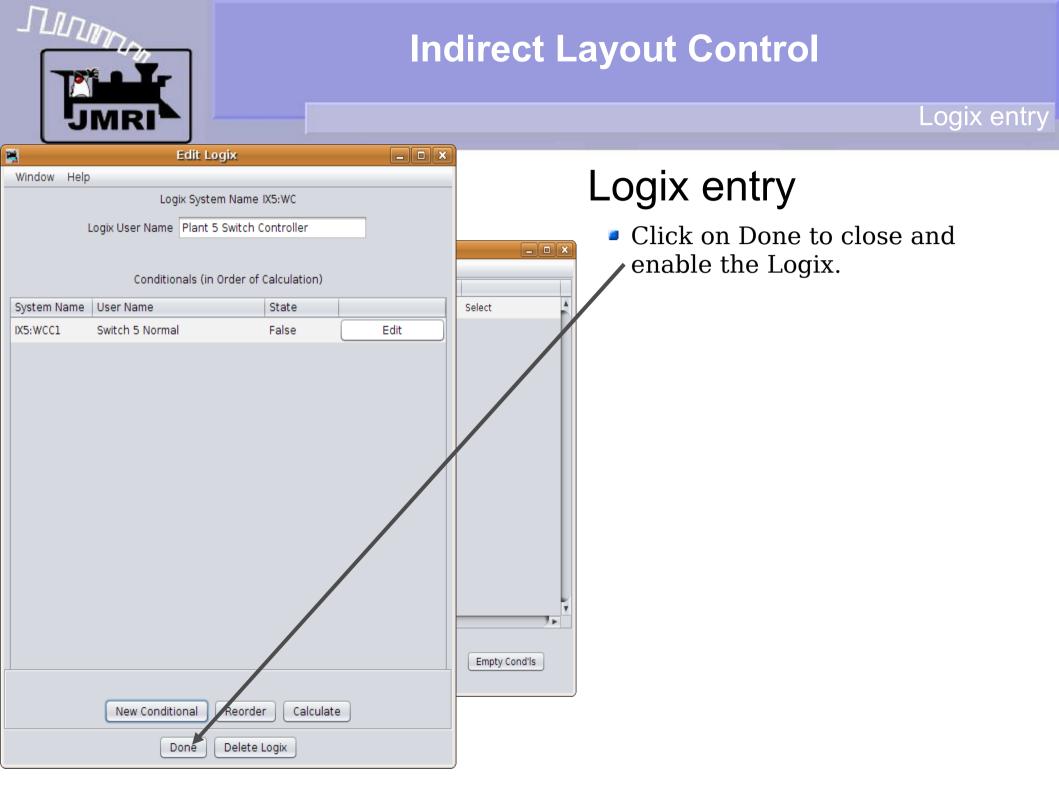
	Layout Control Logix entry
JMKI	
Edit Conditional     Window Help   Conditional System Name IX5:WCC1 Conditional User Name Switch 5 Normal Logical Expression:      Antecedent Variables (the 'If' part)     Row Oper Neg State Variable Description   State Trigg      Row Oper Neg State Variable Description   State Trigg      Row Oper Neg State Variable Description     Sensor, L52, for Sensor Inactive     True     Edit Action   Consequent Action   Action Type   Change Option      System /   Delayed Set Turnout      On Change To True         Action Description   On Change To True, Play Sound File from file, /usr/local//MRI/resources/sounds/Coo      Action True, Set Turnout, LT1 to Closed	<ul> <li>Logix entry</li> <li>Click on Reorder.</li> <li>Then choose who is on first.</li> </ul> I urnout Position <ul> <li>t wait on so the tivate before shed playing.</li> <li>To fix that first furnout' on the turnout entry.</li> <li>Change 'Set Turnout' to</li> </ul>
	Enter '5' for the number of seconds to delay, then 'Update'.
Add Action Reorder Update Conditional Cancel Delete Conditional	



#### Logix entry

1945				Edit Conditional				_ 0 ×	
<b>1</b>	'indov	w Help		Edit Conditional					
vv	indov	w nei	,	Conditional System Name	e IX5:WCC1				
				Conditional User Name Switch 5 Norr					
	aica	l Expre	ccion		ilai				
	gica	rexpre	551011.	Antecedent Variables (th	ve 'if' nart)				
		0.000	Neg		-	Trice			
I F	Row	Oper	Neg	State Variable Description Sensor, LS2, for Sensor Inactive	State	Trigg	Edit	Delete	
	12				False	✓			
		AND	NOT	Sensor, IS6:CB, for Sensor Active			Edit	Delete	
	3	AND	NOT	Turnout, LT1, for Turnout Closed	True		Edit	Delete	
ŀ	<b>\</b> 4	AND		Sensor, IS5:WL, for Sensor Active	True		Edit	Delete	
Ac	tions	s		Logic Operator		es			
				Consequent Actions (the '	then' part)				
		n Descri							
		-		Play Sound File from file, /usr/local/JMRI/res		s/Code	Edit	Delete	
	)n Ch	ange T	o True,	Delayed Set Turnout, LT1 to Closed, after	5 seconds.		Edit	Delete	
	Add Action Reorder								
				Update Conditional Cancel	Delete Condi	itional			

- Click on Reorder.
- Then choose who is on first.
- Continue picking the next item until you have the new order.
- One action does not wait on another to happen, so the turnout will still activate before the sounds are finished playing. To fix that hit 'Edit' on the turnout entry.
- Change 'Set Turnout' to 'Delayed Set Turnout'.
- Enter '5' for the number of seconds to delay, then 'Update'.
- This is our first Conditional
   completed. 'Update' it to save.





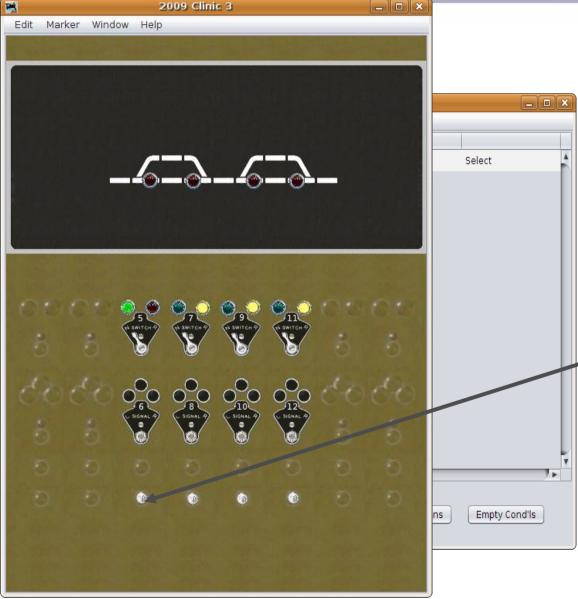
Logix entry

Window Help Logix System Name IX5:WC	
Logix System Name IX5:WC	<b>I</b>
	L
Logix User Name Plant 5 Switch Controller	
	- • ×
Conditionals (in Order of Calculation)	
System Name User Name State Select	<b>A</b>
IX5:WCC1 Switch 5 Normal False Edit	
Reminder 🗙	
Please remember to save your Logixs to disk. (Select 'Store Configuration' in File menu of Logix Table.)	
Empty Cond       New Conditional       Reorder       Calculate	d'Is
Done Delete Logix	

- Click on Done to close and enable the Logix.
- We are reminded to save our work. The Logix may be saved as stand alone files or with their panels. I prefer mine to be saved with the panel files.



Logix entry



- Click on Done to close and enable the Logix.
- We are reminded to save our work. The Logix may be saved as stand alone files or with their panels. I prefer mine to be saved with the panel files.
- Clicking our code button should send the sound of relays, then 'close' the turnout, but only if the OS is clear and the turnout isn't already closed.



Logix entry

	2009 Clinic 3			
E <mark>dit Marker Window</mark>	r Help			
	Stand Street Street	11-12-272	Stand on Street	
		Party of the local division of the local div		
<b>M</b>		ogix Table		_ <b>_ _ X</b>
File Window Help	<u>O</u> ptions			
System Name △	User Name	Enabled	Comment	
IX5:WC	Plant 5 Switch Controller	$\checkmark$		Select
				/ <b>•</b>
	Element Name			
Add			Find Orphans	Empty Cond'Is
	Get References			
	的。如此的			

- Click on Done to close and enable the Logix.
- We are reminded to save our work. The Logix may be saved as stand alone files or with their panels. I prefer mine to be saved with the panel files.
- Clicking our code button should send the sound of relays, then 'close' the turnout, but only if the OS is clear and the turnout isn't already closed.
- Select Edit in order to add the 'thrown' conditional.



Logix entry

1	*	Edit Logix			_ <b>-</b> ×		
	Window Help						
	Logix System Name IX5:WC						
	L	ogix User Name	Plant 5 Switch C	ontroller			
						_ <b>-</b> X	
		Condition	nals (in Order of (	Calculation)			
	System Name	User Name		State		<b>A</b>	
	IX5:WCC1	Switch 5 Normal		False (	Edit		
						T T	
						<b></b>	
Г						Cond'Is	
		New Constitu	nal Dearder				
L		New Condition	onal Reorder	r Calculate			
		D	one Delete I	ogix			
_							

- Click on Done to close and enable the Logix.
- We are reminded to save our work. The Logix may be saved as stand alone files or with their panels. I prefer mine to be saved with the panel files.
- Clicking our code button should send the sound of relays, then 'close' the turnout, but only if the OS is clear and the turnout isn't already closed.
- Select Edit in order to add the 'thrown' conditional.
- Then 'New Conditional'.



#### Logix entry

🖹 Edit Conditional 🗕 🗖							
Window Help							
Conditional System Name IX5:WCC2							
Conditional User Name Switch 5 Reverse							
Logical Expression:							
Antecedent Variables (the 'if' part)							
Row Oper Neg State Variable Description	State	Trigg					
R1 Sensor, LS2, for Sensor Inactive	True		Edit	Delete			
R2 AND Sensor, IS6:CB, for Sensor Active	False	$\checkmark$	Edit	Delete			
R3 AND NOT Turnout, LT1, for Turnout Thrown	True		Edit	Delete			
R4 AND Sensor, IS5:WL, for Sensor Inactive	False		Edit	Delete			
Add State Variable Check	State Variable	s					
Logic Operator							
AND							
Actions							
Consequent Actions (the 't	then' part)						
Action Description							
On Change To True, Play Sound File from file, /usr/local/JMRI/res	ources/sounds	/Code	Edit	Delete			
On Change To True, Delayed Set Turnout, LT1 to Thrown, after	5 seconds.		Edit	Delete			
Add Action Reorder							
Update Conditional Cancel Delete Conditional							

# Logix entry

Enter 'Switch 5 Reverse' then all the variables and conditionals for the other directions actions.



#### Logix entry

1				Edit Conditional				_ O X
W	indow	v Helj	D					
	Conditional System Name IX5:WCC2							
	Conditional User Name Switch 5 Reverse							
Lo	Logical Expression:							
	Antecedent Variables (the 'if' part)							
F	low	Oper	Neg	State Variable Description	State	Trigg		
F	1			Sensor, LS2, for Sensor Inactive	True		Edit	Delete
F	2	AND		Sensor, IS6:CB, for Sensor Active	False	$\checkmark$	Edit	Delete
F	3	AND	NOT	Turnout, LT1, for Turnout Thrown	True		Edit	Delete
F	4	AND		Sensor, IS5:WL, for Sensor Inactive	False		Edit	Delete
				Add State Variable Check S	State Variable	s		
				Logic Operator				
				AND 🔻				
Ac	tions	6						
				Consequent Actions (the 't	hen' part)			
4	ction	Descri	iption					
C	n Cha	ange T	o True,	Play Sound File from file, /usr/local/JMRI/reso	ources/sound	s/Code	Edit	Delete
0	n Cha	ange T	o True,	Delayed Set Turnout, LT1 to Thrown, after 5	5 seconds.		Edit	Delete
Add Action Reorder								
	Update Conditional Cancel Delete Conditional							

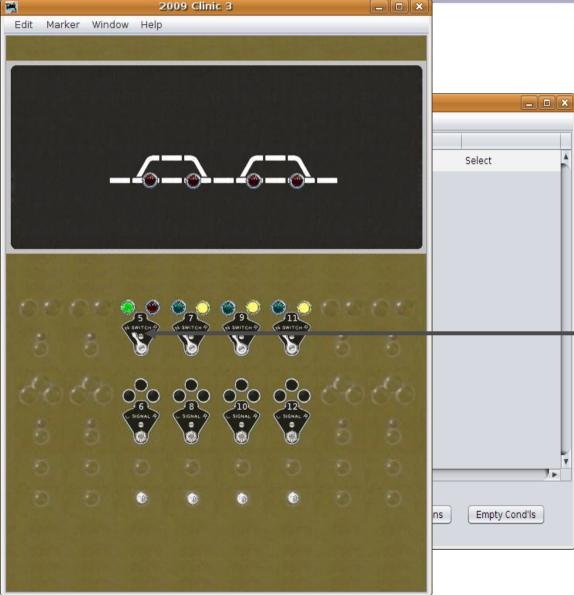
- Enter 'Switch 5 Reverse' then all the variables and conditionals for the other directions actions.
- Update the new conditional.



Edit Logix 🗕 🗆 🗙							
Window Help				L agiv antru			
Logix System Name IX5:WC				Logix entry			
	Logix User Name	Plant 5 Switch Controller					
	Condition	nals (in Order of Calculation)		Enter 'Switch 5 Reverse' then all the variables and			
System Nam	ne User Name	State		conditionals for the other			
IX5:WCC1	Switch 5 Normal	False	Edit				
IX5:WCC2	Switch 5 Reverse	e False	Edit	directions actions.			
				Update the new conditional.			
				And click 'Done'.			
	New Conditio			Cond'Is			
	D	one Delete Logix					



Logix entry

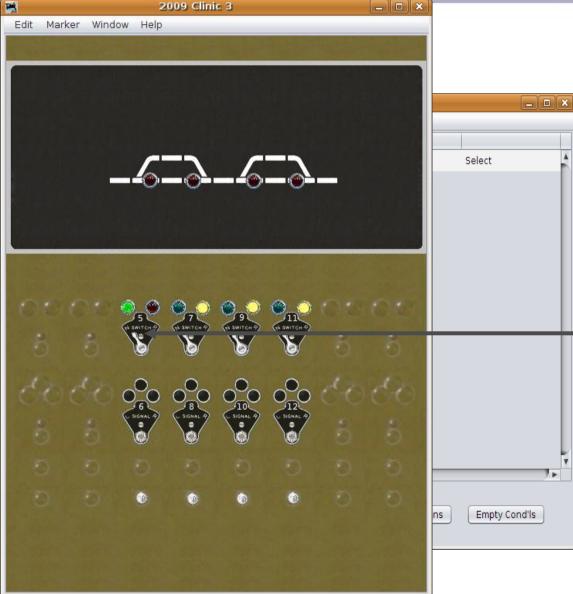


- Enter 'Switch 5 Reverse' then all the variables and conditionals for the other directions actions.
- Update the new conditional.
- And click 'Done'.
- Test again and we should have

   interlocked control of the first turnout. The problem is that there is no feedback between the turnout position and the indicator jewels.



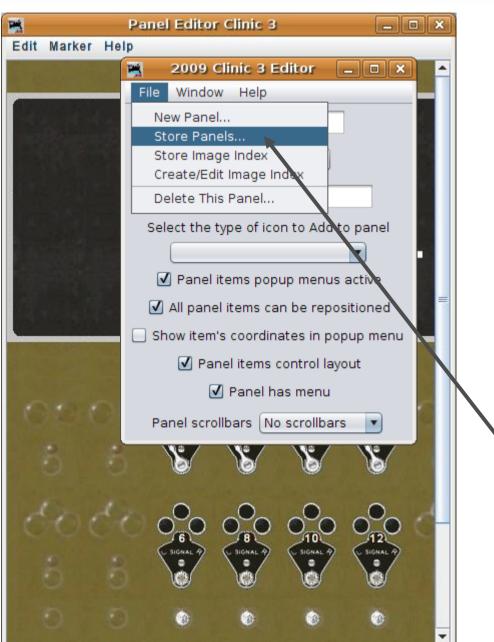
Logix entry



- Enter 'Switch 5 Reverse' then all the variables and conditionals for the other directions actions.
- Update the new conditional.
- And click 'Done'.
- Test again and we should have interlocked control of the first turnout. The problem is that there is no feedback between the turnout position and the indicator jewels.
- In our next clinic we will cover editing options and test our panel.



Logix entry



- Enter 'Switch 5 Reverse' then all the variables and conditionals for the other directions actions.
- Update the new conditional.
- And click 'Done'.
- Test again and we should have interlocked control of the first turnout. The problem is that there is no feedback between the turnout position and the indicator jewels.
- In our next clinic we will cover editing options and test our panel.
- Save our work.



#### Logix entry

-	Panel Editor Clinic 3	
Edit Marker He	lp	
Station and	2009 Clinic 3 Editor	
-	File Window Help	
addine star	New Panel	
<b>PERMERE</b>	Store Panels	
	Store Image Index Create/Edit Image Index	
	Save	×
Look In: 📋 .ji	mri 🔽 🕋	
a decoders	RXTX-install	Bordon-a.xml
a decoders	2009Clinic1.xml	Bordon-b.xml
programme	ers 📄 2009Clinic2.xml	Bordon-combined
resources	📄 2009Clinic 🚬 xml	📄 Bordon-route.xml
📄 roster	APB Test.xml	📄 catalogTrees.xml
-		7.
File Name:	2009Clinic3.xml	
Files of Type:	XML files	
		Save <u>c</u> ancel
CAC CA		
	6 8 10 ( SIGNAL 9 ( SIGNAL 9 ( SIGNAL 9	9 SIGNAL 9
5 B		
and the second second		
0.6		0
		-

- Enter 'Switch 5 Reverse' then all the variables and conditionals for the other directions actions.
- Update the new conditional.
- And click 'Done'.
- Test again and we should have interlocked control of the first turnout. The problem is that there is no feedback between the turnout position and the indicator jewels.
- In our next clinic we will cover editing options and test our panel.
- Save our work.
- As 2009Clinic3.xml



This completes Clinic 3.

- The next clinic will start with basic new editing options for Logix and Panels.
- Next we will cover basic ABS signaling using SSL. (Simple Signal Logic)
- These clinic files will be available at our web site.

http://www.rr-cirkits.com/Clinics/Clinics.html

Versions from previous years clinics are also available there for your convenience.