## Comunicación entre objetos



En situaciones anteriores nos hemos encontrado con la necesidad de llamar a un objeto desde otro.

Por ejemplo, cuando implementamos el objeto procedimiento AttractionsList, necesitamos filtrar las atracciones que tienen como nombre de país "France":



o, lo que era similar, identificador de país igual a 2 (que correspondía a "France")

...y para lograrlo, utilizamos valores fijos en el código:

🐼 Start Page 🗙 🦆 AttractionsList* 🗙	*
Source * Layout   Rules   Conditions   Variables	
Subroutines ~	
<pre>1 print Title 2 print ColumnTitles 3 □ For each Attraction order CountryName 4 where CountryId = 2 5 print Attractions 6 endfor 7 </pre>	< > hhi III
_ <	>

Pero esto implica que si quisiéramos filtrar las atracciones de un país diferente a Francia, deberíamos modificar

el código del procedimiento ¡cada vez!

Lo ideal sería que pudiéramos "recibir" de alguna manera en este objeto ese valor por el que queremos filtrar.

	GeneXus
Object A	Object B: AttractionsList
	AttractionsList X Source Layout Rules Conditions Variables Subroutines 1 print Title 2 print ColumnTitles 3 For each Attraction order CountryName 4 where CountryId = Sparm 5 endfor 4 or int Attractions 5 endfor 5 endfor 6 endfor 5 endfor 6 endfor 7 end

Dicho de otro modo, que otro objeto GeneXus pueda permitir al usuario elegir ese valor:

	GeneXus
Object A	Object B: AttractionsList
Bvag	AttractioneList* X Source* Layout Rules Conditions Variables Subroutines 1 print Title 9 For each Attraction order CountryName 4 where CountryI a Sparm 5 endfor 5 endfor 6 endfor 7 endfor 7 endfor 8 endfor 7 endfor 8 endfor

... y luego lo envíe a este objeto procedimiento para que liste las atracciones en base a ese país recibido.

	GeneXus
Object A	Object B: AttractionsList
Bvag	AtractionsList * X Source * Layout Rules   Conditions   Variables   Source * Layout Rules   Variables   Variables

Veremos a continuación, mediante este ejemplo, cómo implementar la comunicación entre objetos GeneXus.

Para empezar, debemos crear un objeto que sea capaz de ofrecernos una pantalla para pedir valores al usuario y hacer algo con esos valores. El objeto que permite esto es el **web panel**, que estudiaremos en detalle más adelante. Por ahora digamos que se trata de un panel visual muy flexible que permite pedir datos al usuario, mostrar información de la base de datos o de otras fuentes, entre otras cosas. Por ejemplo, el Work With de atracciones fue implementado automáticamente por GeneXus como un Web Panel.

<pre>price bot view Layout insert suid Anontege Manager Vindow Tools Rep</pre>
Image: Section of the section of th
KB Explorer       # X       X dtractionsList X       WW4traction X       X         Cpen: [hans or Fatter       Web Form        Kules        Conditions        Variables         Construction X       Meb Form        Rules        Conditions        Variables         Construction X       Attractions       Variables       Image: Attraction X       Image: Attractin X       Image: Attraction X       Image: Attra
Open:     [hume of Pattern                   * TravelAgency             ~ © Rot Module                  ~ © GeneXus                      Artipot                  Attraction                Attractions                Attractions                Attractions                Attractions                Attractions                Attractions                Attractions                Attractions                Attractions
Contraction Name     Attraction Nam     Attraction Name     Attraction Name     Attraction Name     A
<ul> <li></li></ul>
> ☐ GeneXus     Image: I
Atrine     Image: Constraint of the second sec
Image: Attraction in the second s
Image: Constraint of the second se
Work/Mutacion     Image: Constraint of the second sec
I AttractionAttractionWC
AttractionGeneral
Ordered By       WWAttraction
AttractionsList
* AttractionsPerCategory
CategoriesAndTheirAttractionsList
Category AttractionName CountryName CategoryName CityName & Update & & CityName & CountryName & CategoryName & CityName & & CityName & & CityName & & & & & & & & & & & & & & & & & & &
Countries/VithMoreThan2Attractions
Customer Customer
🔓 Diagram1
La Diagram2
💑 Diagram3
L Diagram4 K K

Entonces, vamos a crear un objeto de este tipo. Lo llamaremos EnterAttractionsFilter:

New Object		Х
Select a Category:	Select a Type:	
Common Workflow Webling Webling Webling Webling	Image         Image	
Web application form. Int	eract with end user using events, show or request data and more.	
Name:	EnterAttractionsFilter	
Description:	Enter Attractions Filter	
	Create Cancel	

Veamos que se crea un Form Web, que será la pantalla del objeto. Contiene únicamente una tabla.



## Le agregamos una variable CountryId



la que por nombrar igual que al atributo queda basada en él, y, por lo tanto, asume su mismo tipo de datos. De este modo si le cambiamos el tipo de datos al atributo, por ejemplo, por numérico de 10 en vez de 4, la variable automáticamente cambiará tomando ese nuevo valor.

Attrac	Insert Attribu	te/Variable				×	-
Web For	Filters					- 64	
	Pattem:	New Variable		×	:		Б
	Object Type	📜 🤌 🌾   Filter		×	:		Ľ
	Object.	Variable: &Count	tryId	^	•		
	Name	Name	CountryId			^	
	K Today	Description	Country Id				
	& Time	Column title	Country Id	7			
	& Pgmn	Class	Attribute		_		
	& Pgmd			Help			
	Airline	Y Type Definition	n		ount		
		Based on	CountryId		-		
	Airpor	Data Type	Numeric		-		
	Airpor	Length	4		ne		
	Attrac	Decimals	0		ddress		
	🕸 Attrac	Signed	False		ł		
	🕸 Attrac	Collection	False		lame		
	🕸 Attrac	Dimensions	Scalar		hoto		
	Categ	Initial value					
	Categ				ame		
	CityId		ОК	Cancel	]		
	CityNa CityNa		Ivanie III	City Nair			
	Countr	Vid		Country	a		
	New Variabl	le Eait		ОК	Cance		

Ahora editamos las propiedades de la variable y veamos que su propiedad Control Type asume el valor Edit. Esto significa que cuando se ejecute el web panel este campo esperará que el usuario digite un valor numérico, pero no brindará ninguna ayuda para elegir qué valores existen en la base de datos y a qué país corresponde. Vamos a cambiar el tipo de control por Dynamic Combo Box:

🛟 AttractionsList X 🔂 EnterAttractionsFilter* X	•	0	Properties	# ×
Web Form * Rules Events Conditions Variables *		Ger	neral Class	
- cNo action group colocted a		>	👌 🌾 Filter	×
		~	Attribute/Variable: &Countryld	
MainTable Countryld	· ·		Control Name	&Countryld
			Attribute	&Countryld
Country Id &CountryId			Label Position	Left
			Label Caption	Country Id
			Readonly	False
			Return On Click	False
			✓ Appearance	
			Class	Combo Box
		Invite Message	Radio Button	
			Auto Resize	Edit
			Width	Check Box
			Height	Dynamic Combo Box
			Format	List Box
ÓÓ			Tooltip Text	Dynamic List Box
			∨ Control Info	Rating 🗸 🗸
			Control Type	Edit 🚽 🗸
			Input Type	Values
			Notify Context Change	False
			> Behavior	
			> Cell information	
L			> Row information	

De esta manera se le va a ofrecer al usuario una serie de valores extraídos de la base de datos, para que él elija el que desea. ¿Qué valores? Los del atributo CountryId:

Control Type	Dynamic Combo Box
Data Source From	Attributes
ltem Values	Countryld
🛕 Item Descriptions	
Sort Descriptions	True
Conditions	
Instantiated Attributes	
Empty Item	False
Notify Context Change	False

Es decir, se va a recorrer la tabla Country y cargar en el combo los Countrylds existentes. Pero como los identificadores no suelen decirnos nada, si bien la variable va a almacenar un identificador de país, lo que se le va a mostrar al usuario en la pantalla es el contenido del atributo que indiquemos aquí... Elegimos mostrar el nombre de país:

Control Info	
Control Type	Dynamic Combo Box
Data Source From	Attributes
Item Values	Countryld
Item Descriptions	CountryName
Sort Descriptions	Тгие

Y vemos cómo aparece la flechita indicadora de combo:





Resumiendo, esto nos ofrecerá en ejecución un combo que nos presentará la lista de países de la base de datos para elegir el que nos interesa.

Agregamos además un botón. Nos pide que indiquemos el nombre del evento que ese botón tendrá asociado. Lo llamamos: "List Attractions By Country":

🗳 AttractionsList × 🔂 EnterAttractionsFilter * ×	Toolbox 9	×
Web Form * Rules Events Conditions Variables *	Controls	•
<no action="" group="" selected=""></no>	⊞10 Attribute/Variable	
	Button	
	Embedded Page	
	e Error Viewer	
Country Id &CountryId ~	Horizontal Rule	
	HyperLink	
		_
	A Text Block	=
	Web Component	
Select/Define User Event X	Containers	
Event Name: Attractions By Country	🛃 Free Style Grid	
OK Carcel	💾 Grid	
	C Group	
	Html	
	III Responsive Table	
	Section	
	🛅 Tab	
	III Table	
	Extended Controls	
	ERHoverPanel	

Y vemos que el texto del botón asume el mismo nombre por defecto:

AttractionsList × 🔂 EnterAttractionsFilter* ×	-
Web Form * Rules   Events   Conditions   Variables *	
MainTable	<i></i>
Country Id &CountryId ~	

Si nos posicionamos sobre él, presionamos botón derecho y elegimos Go to Event...

Country Id &CountryId	~			
O List Attractions By Country	v	0		·····•••••••••••••••••••••••••••••••••
¢.		Insert Attribute/Variabl Insert Button Delete	e Del	Ģ
0		Convert to Tabular Tab Responsive Sizes Go To Event	le	
	an at	Insert Row Before Insert Row After Insert Cell Before Insert Cell After Delete Row		
	_ ۳	Delete Cell		

...vemos que se creó un evento con ese nombre, y se pasó de la solapa Web Form a la Events en forma automática, y el cursor está esperando a que ingresemos el código que se ejecutará cuando este evento se dispare. Es decir, cuando el usuario presione el botón asociado.

🗳 AttractionsList × 🔂 EnterAttractionsFilter* ×	•
Web Form * Rules Events * Conditions Variables *	
'List Attractions By Count ~	
1 = Event 'List Attractions By Country' 2	100 T

Lo que necesitamos que se haga en ese momento es llamar al objeto procedimiento AttractionsList que lista las atracciones y enviarle el país que queremos que utilice para filtrarlas:

🛟 AttractionsList 🗙 🛃 EnterAttractionsFilter * 🗙	Ŧ
Web Form Rules Events* Conditions Variables *	
'List Attractions By Count ~	
Event 'List Attractions By Country'	12
2 AttractionsList(&CountryId)	닅
3 - Endevent	



Notemos que al momento de apretar el botón y ejecutar este código, la variable &Countryld contendrá el identificador de país del país que el usuario haya elegido en el combo box en la pantalla. Anteriormente vimos que una variable es una porción de memoria a la cual le damos un nombre y nos sirve para guardar un dato en forma temporal, y que cada objeto tiene su sección de variables,

🗳 AttractionsList X 💽	EnterAttractionsFilter * ×		-
Web Form * Rules Events	s* Conditions Variable	s *	
Name	Туре	Is Collection	Description
🖃 & Variables			
🗄 🚷 Standard Variables			
<ul> <li>CountryId</li> </ul>	Attribute:CountryId		Country Id
CountryId	Attribute:CountryId		Country Id

o sea que las variables que se definen en un objeto son conocidas solamente dentro de ese objeto. Así, si dos objetos tienen una variable Countryld definida, aunque se llamen igual, se tratará de dos variables distintas.

Entonces, ¿cómo hacemos para que un objeto A pueda invocar a otro objeto B en un momento dado, pasándole valores?:



y que ese otro objeto B pueda cargar en sus variables internas los valores que se le enviaron, para poder hacer algo con esa información:



Para que un objeto pueda recibir valores (a los que llamamos **parámetros**), debemos ir a su sección Rules y escribir una regla **Parm**. Esa regla **Parm** declara los parámetros que el objeto puede recibir y/o devolver a quien lo llame.

	GeneXus
Object A	Object B
	Parm Parm
< ▷ 🖉 🕲	

Como en nuestro ejemplo quien va a recibir los valores es el objeto procedimiento AttractionsList, abrimos el objeto y vamos a su sección de reglas.

Escribimos:

AttractionsList* × 🔜 EnterAttractionsFilter ×	
Source Layout Rules* Conditions Variables*	
parm( in: &CountryId );	
3 □ Output_file('AttractionsList.pdf', 'pdf');	
	M
AttractionsList × Reference EnterAttractionsFilter ×	Tioolbox 7 X
Source Layout Rules Conditions Variables	Snippets
1 parm( in: &CountryId ):	Assignment
2	🗐 Default
<pre>3   Output_file('AttractionsList.pdf', 'pdf');</pre>	Cutput_file
5	🖹 Parm
	🗎 Printer

Observemos que en la Toolbox se nos ofrecen todas las reglas que podríamos escribir en un objeto de este tipo. Entre ellas, la parm. Podríamos haberla arrastrado desde allí.

🐉 AttractionsList* X 🔜 EnterAttractionsFilter X 🔹	T	Toolbox	<del></del>	×
Source Layout Rules* Conditions Variables*	Ξ	Snippets		
1 = parm( in: &CountryTd ):	۵	Assignment		
	۵	Default		
<pre>3 = Output_file('AttractionsList.pdf', 'pdf');</pre>	۵	Output_file		
5 = Parm(attOrVan);	• 🗈	Parm		
		Printer		
V V				

Vemos también nos informa que tenemos que sustituir esto por un atributo o variable. Luego veremos el caso de los atributos. Por ahora solamente vemos el caso de las variables.

Con "in" estamos indicando que la variable &Countryld será un parámetro de entrada. Esto significa que solamente será utilizado para recibir un valor de quien lo llame. No para devolver. Podemos omitir esta información y dejar que GeneXus la infiera.

Hemos escrito el **nombre** de la variable, pero no la hemos ingresado como variable en el objeto. Para hacerlo, una de las formas es posicionarnos sobre el nombre, dar botón derecho y elegir "Add variable":

🛟 AttractionsList* X 🔜 EnterAttra	actio	nsFilter X		-
Source Layout Rules* Condition	s   V	'ariables *		
1 = Output_file('Attract	tion	sList.pdf', 'pdf');		
3 ⊡ parm( in: &CountryI 4		Add Variable		~
5	Х	Cut	Ctrl+X	
6	Ľ	Сору	Ctrl+C	
	Ľ	Paste	Ctrl+V	
	×	Delete	Del	
	1	Indent		
	ŧ	Unindent		
	At	Make Uppercase	Ctrl+Shift+U	
	al 	Make Lowercase	Ctrl+Shift+L	
		Comment Selection	Ctrl+Q	
	_= ∓=	Collapse All	cartonineto	
	ĪT	Expand All		
		Toggle Bookmark	Ctrl+B	
	N	Open	F12	
	5	References	Ctrl+F12	
	٥	Properties	F4	$\sim$
				>

Si ahora vamos a la solapa de variables vemos que se la ha definido, basada, por defecto, en el tipo del atributo Countryld. Esto se debe a que la llamamos igual que a un atributo.

🛟 AttractionsList* 🗙 属	EnterAttractionsFilter $ imes$		•
Source   Layout   Rules   C	onditions Variables *		
Name	Туре	Is Collection	Description
⊡. & Variables			
<ul> <li>Standard Variables</li> <li>CountryId</li> </ul>	Attribute:CountryId		Country Id

En este objeto hemos definido la variable con el mismo nombre y tipo de datos que la que usamos en el web panel para que el usuario ingresara el país:

EnterAttractionsFilter ×	AttractionsList* ×
Web Form   Rules Events Conditions   Variables	Source   Layout Rules * Conditions   Variables *
List Attractions By Count v List Attractions By Country' AttractionsList(&CountryId) S Endevent	<pre>1 parm( in: &amp;CountryId ); 2 3 Output_file('AstractionsList.pdf', 'pdf'); 4 5</pre>
	< > > > *

Sin embargo, como dijimos, son dos variables distintas. Una válida solamente en el web panel y la otra en el procedimiento. Podríamos haberlas llamado distinto en ambos objetos, pero para que la comunicación y el pasaje de información sea correcta, el tipo de datos debe coincidir entre quien llama y quien es llamado.



Ahora, nuestro objeto procedimiento está preparado para recibir a un identificador de país, en este caso desde el webpanel EnterAttractionsFilter.

Sólo nos resta quitar el filtro fijo que teníamos (el valor 2 de país) en el For each, y cambiarlo por la variable cuyo valor es recibido como parámetro.



Observemos que al haber definido la regla Parm de esta manera, de ahora en adelante cualquier objeto que llame al procedimiento podrá (y deberá) pasarle el valor del identificador del país.

Ya no podrá invocarse a este procedimiento sin enviarle un valor de este tipo. Es por esta razón que el procedimiento AttractionsList ya no va a aparecer en el Developer Menu.

	GeneXus
Object A	Object B: AttractionsList
• SCountryId	Call

En el caso del webpanel teníamos ese valor en una variable:



Pero si tuviéramos el dato en un atributo, incluiríamos dentro del paréntesis al atributo que corresponda,

	GeneXus
Object A	Object B: AttractionsList
CountryId AttractionsList(CountryId)	Brann(       &CountryId       )         Attractionalist * X       •         Source*       Layout Rules *   Conditions   Variables *           Cubroutines       •         1       print Title         2       print Columnitiles         3       For each Attraction order CountryName         4       where CountryId = &CountryId         5       print Attractions         7       endfor

También podríamos pasar directamente un valor:

	GeneXus
Object A	Object B: AttractionsList
AttractionsList(2)	Call &CountryId &CountryId (Conditions) Variables*

o en caso de tener que pasar dos o más valores, enviaríamos varios atributos, y/o valores explícitos, y/o variables, separados por coma:

	GeneXus
Object A	Object B
	Parm( , , )
AttractionsList( <u>attribute1</u> , <u>2</u> , <u>&amp;variable1</u> )	

Esos parámetros también se declaran en la regla parm en forma ordenada y separados por comas.

Evidentemente, un objeto que no recibe parámetros no debe declarar regla Parm.

Vamos a probar lo que hemos hecho: F5. Vemos que ya no aparece el procedimiento AttractionsList. Ahora solamente podemos invocarlo a través del web panel...

Airline	Airport	AttractionsPerCategory
CategoriesAndTheirAttracti	Category	CountriesWithMoreThan2At
Customer	EnterAttractionsFilter	🗐 Flight
R Home	Numbero Enter Attractions Filter	Report1
Supplier	www.attraction	wwCountry

En el combo del país, elegimos Francia...

Enter Attractions Filter X	-	• •	× =
Application Name	L.		-
	<sub>bv</sub> G	eneX	us
Recents			
Country Id			
France	*		
Brazil China			
France United States			
		•	

Al haber elegido el valor France del dynamic combo, internamente se seleccionó el valor del identificador de Francia (en este caso el valor 2) y ese valor es el que se le envía al procedimiento AttractionslList.

Vemos que se ejecuta el reporte, mostrándonos solamente las atracciones del país France:

aattractionslist.aspx ×					4	-		×
← → C [] trialapps3.genex	us.con	n/Id8562acf4c97c0	)fe8095a5c284d68	38e07/aattractionslis	t.aspx ්දි	2	<b>A</b> 0	Ξ
aattractionslist.aspx			1/1		¢	±	ē	
T=00 00	>	Attracti	ons List					
-	ld	Name	Country	Photo				
	1	Louvre Museum	France					
	3	Eiffel Tower	France	A				
	6	Matisse Museum	France					ľ
							41	
							+	
							-	

Vamos a suponer ahora que queremos listar todas las atracciones cuyos nombres se encuentren entre dos valores elegidos por el usuario. Por ejemplo, entre la "A" y la "D".

Para eso, vamos a agregarle al webpanel que definimos previamente la posibilidad de que el usuario ingrese un nombre inicial y un nombre final para que, presionando un botón, se invoque a un listado que muestre todas las atracciones turísticas cuyos nombres se encuentren dentro de ese rango.

Vamos al webpanel EnterAttractionsFilter y agregamos una tabla con dos variables.

• &AttractionNameFrom... basada en la definición del atributo AttractionName:

<ul> <li><no action="" gro<="" li=""> </no></li></ul>	Filt	ters			
( MainTable	Patt	em:	New Variable		×
	Obje Obje	ect Type	Al 🏹 Filter		×
Country Id		N	Variable: &Attra	ctionNameFrom	
		Caust	Name	AttractionNameFrom	on
		Count	Description	Attraction Name From	Id
		Count	Column title	Attraction Name	Id Name
9 🕀	~	Count	Class	Attribute	Name
					Help
0			v Type Definitio	n	
			Based on	AttractionName	
0			Data Type	Character	_
List Attraction			Length	50	
			Collection	False	
			Dimensions	Scalar	
			Initial value		
			> Validation		
) Outrus			> Picture		
Output					¥
now: General				OK Ca	ncel
		_			

• y & AttractionNameTo, también basada en la definición de AttractionName:

	×	>			
			New Variable	Pattem:	III MainTable
	< 1	>	🔠 🌖 🌾 Filter	Object Type	
	~	ctionNameTo	<ul> <li>Variable: &amp;Attra</li> </ul>	Object:	Country Id
n		AttractionNameTo	Name	Name	
		Attraction Name To	Description	& Count	
		Attraction Name	Column title	🕸 Count	
ame		Attribute	Class	🕸 Count	9 🕀
		Help			Attraction N
		n	✓ Type Definitio		0
		AttractionName	Based on		
	'	Character	Data Type		
		50	Length		List Attendia
		False	Collection		LIST ATTIACTION
		Scalar	Dimensions		
			Initial value		
			> Validation		
			> Picture		Output
		50 False Scalar	Length Collection Dimensions Initial value > Validation > Picture		List Attractio

Como ya dijimos, esto significa que la definición de la variable está enlazada con la definición del atributo y si cambiamos el tipo de datos del atributo, automáticamente se cambiará la variable en forma acorde.

Luego agregamos un botón de evento "List Attractions By Name":

Country Id &CountryId  Attraction Name From &AttractionNameFrom Attraction Name To &AttractionNameTo List Attractions By Country List Attractions By Name			_
Country Id &CountryId ~ Attraction Name From &AttractionNameFrom Attraction Name To &AttractionNameTo List Attractions By Country List Attractions By Name			
Attraction Name From     &AttractionNameFrom       Attraction Name To     &AttractionNameTo       List Attractions By Country     List Attractions By Name	Country Is	d Recountered of	
Attraction Name From       &AttractionNameFrom         Attraction Name To       &AttractionNameTo         List Attractions By Country       List Attractions By Name	Country id		
Attraction Name From       &AttractionNameFrom         Attraction Name To       &AttractionNameTo         List Attractions By Country       List Attractions By Name			
Attraction Name From       &AttractionNameFrom         Attraction Name To       &AttractionNameTo         List Attractions By Country       List Attractions By Name			
Attraction Name From       &AttractionNameFrom         Attraction Name To       &AttractionNameTo         List Attractions By Country       List Attractions By Name			
Attraction Name From       &AttractionNameFrom         Attraction Name To       &AttractionNameTo         List Attractions By Country       List Attractions By Name			
Attraction Name From       &AttractionNameFrom         Attraction Name To       &AttractionNameTo         List Attractions By Country       List Attractions By Name			
Attraction Name To &AttractionNameTo List Attractions By Country List Attractions By Name	Attractio	on Name From & AttractionNameFrom	
Attraction Name To &AttractionNameTo List Attractions By Country List Attractions By Name	, according to the second s		
Attraction Name To       &AttractionNameTo         List Attractions By Country       List Attractions By Name			
Attraction Name To &AttractionNameTo List Attractions By Country List Attractions By Name			
Attraction Name To &AttractionNameTo List Attractions By Country List Attractions By Name			
List Attractions By Country List Attractions By Name	Attractio	on Name To &AttractionNameTo	
List Attractions By Country List Attractions By Name			
List Attractions By Country List Attractions By Name			
List Attractions By Country List Attractions By Name			
List Attractions By Country List Attractions By Name			
List Attractions By Country List Attractions By Name			
	List Attra	actions By Country List Attractions By Name	

Nos posicionamos en el botón que acabamos de agregar, presionamos botón derecho y elegimos Go to event. Tenemos que invocar aquí dentro al procedimiento que imprimirá las atracciones turísticas del rango.

Ya teníamos el reporte AttractionsList... pero recibía por parámetro el identificador de país, no el rango de nombres. Vamos a grabarlo con otro nombre, AttractionsByName, y modificar su regla parm, para que ahora reciba dos parámetros de entrada: la variable &NameFrom, y la variable &NameTo:



Tenemos que definirlas como variables, y especificarle el tipo de datos, así que hacemos botón derecho sobre la primera y presionamos "Add Variable"

🐼 Start Page 🗙 🦆 Attractio	onsList 🗙 🔜 EnterAttrac	tionsFilter* 🗙 🎳 /	AttractionsByName* ×	
Source Layout Rules* Con	ditions Variables			
1 — parm( in: &Name 2	Add Variable		]	
3 ⊡ Output_file('A 4	Function Rule	Ctrl+Shift+F Ctrl+Shift+R		
5	Cut Cut Copy	Ctrl+X Ctrl+C		
	Paste X Delete	Ctrl+V Del		
	t≡ Indent t≣ Unindent			
	A1 Make Uppercase	Ctrl+Shift+U Ctrl+Shift+I		
	Comment Selection	Ctrl+Q Ctrl+Shift+O		
	T Collapse All			
	Toggle Bookmark	Ctrl+B		
<				

y editando sus propiedades, la basamos en el atributo AttractionName:

🐼 Start Page X 🦆 AttractionsList X 🔜 EnterAttractionsFilter * X 🛟 AttractionsByName * X	-	Properties	<del>4</del> )
Source Layout Rules* Conditions Variables*		📜 🤌 🌾   Filter	>
1 parm( in: &NameFrom, in: &NameTo );	5	✓ Variable: &Nan	eFrom
		Name	NameFrom
4	^	Description	Name From
5		Column title	Attraction Name
			<u>Help</u>
		✓ Type Definit	on
		Based on	AttractionName
		Data Type	Character
		Length	50
		Collection	False
		Dimensions	Scalar
		Initial value	
		> Validation	
		> Picture	
		> Control Info	
	$\checkmark$	> Behavior	
<u>  </u> <u>·</u>			

Hacemos lo mismo con & NameTo.

Ahora vamos a utilizar estas variables/parámetros en el For each..., para quedarnos con las atracciones que cumplan que su AttractionName sea mayor o igual al valor de la variable &NameFrom, así como menor o igual al valor de la variable &NameTo:

Print Title Print columnTitles
For each Attraction order CountryName Where AttractionName >= &NameFrom where AttractionName <= &NameTo Print Attractions
Endfor

Con esto el procedimiento está listo y solamente nos resta invocarlo desde el web panel.

Vamos al webanel y agregamos la invocación. Arrastramos el reporte AttractionsByName de esta ventana para no tener que digitarlo y entre paréntesis escribimos los parámetros separados por coma, en este caso las variables &AttractionNameFrom y &AttractionNameTo que se le ofrecen al usuario en la pantalla:



Observemos que el primer parámetro que escribimos en la invocación se cargará en el primer parámetro definido en la regla Parm del objeto llamado y el segundo parámetro de la llamada se cargará en el segundo parámetro del objeto invocado.





Debemos ser cuidadosos de respetar el orden en la invocación y en la definición de la regla Parm. Es buena práctica usar nombres relacionados como hicimos aquí, a efectos de entender mejor el código.

Notemos que los nombres que hemos elegido para las variables del web panel y las del procedimiento son distintas. Como ya dijimos, lo importante es que los tipos de datos enviados y recibidos, coincidan.

Presionemos F5 para ejecutar....

Abrimos el webpanel y para comenzar queremos ver todas las atracciones entre la "A" y la "Z":

Enter Attractions Filter × ← → C □ trialapps3.ce	enexus.com/ld8	562acf4	c97c0t	fe8095a5c284d688e	e07/enti	 इ.ट्रट्र	2	_ ()	×
Application	n Name					by	Gei	neXu	Ŝ
Recents Enter A	ttractions								
Country Id	Brazil	۳							
Attraction Name From	А								
Attraction Name To	Z					)			
List Attractions By Country				List Attractions By Na	ame				
	-		_						

Presionamos el botón "List Attractions By Name"... y vemos que se listan todas las atracciones.

1000 to		Attractions List				
-	ld	Name	Country	Photo		
	4	Christ the Redemmer	Brazil			
	2	The Great Wall	China	XX		
	7	Forbidden city	China			
	1	Louvre Museum	France	- ( - of any		
	6	Matisse Museum	France			
	3	Eiffel Tower	France	A		
	5	Smithsonian Institute	United States			

Ahora acotamos un poco más el rango, ponemos entre la 'A' y la 'F':

Y vemos cómo funcionó el filtro:

5-00 00		Attraction		
	ld	Name	Country	Photo
	4	Christ the Redemmer	Brazil	
	3	Eiffel Tower	France	A

Para finalizar enviemos todo lo que hicimos a GeneXus Server...

Pen	s								
Pen	s								
Pen	dina								
rea I	oning	Con	nmits (62/62)	Objects					T
~			1 Type	Description	Modified O	Module	Action	Last Synchroniz	User
	ą.	-	Procedure	Attractions By Name	8/29/201_	Root Module	Inserted	7/7/2016 4:32	ARTECH\acaggiano
	P		Web Panel	Enter Attractions Filter	8/29/201_	Root Module	Inserted	7/7/2016 4:32	ARTECH\acaggiano

En el siguiente video veremos otras formas de enviar y recibir parámetros, incluyendo los efectos de colocar un atributo en la regla Parm, en lugar de una variable.

	<b>C</b>	V		
	Gene	XUS		
Videos	training s	enexus.com		
Document Certificact	ation wiki.gene ions training.g	exus.com jenexus.com/certificati	ions	

 ${}^{\rm Page}24$