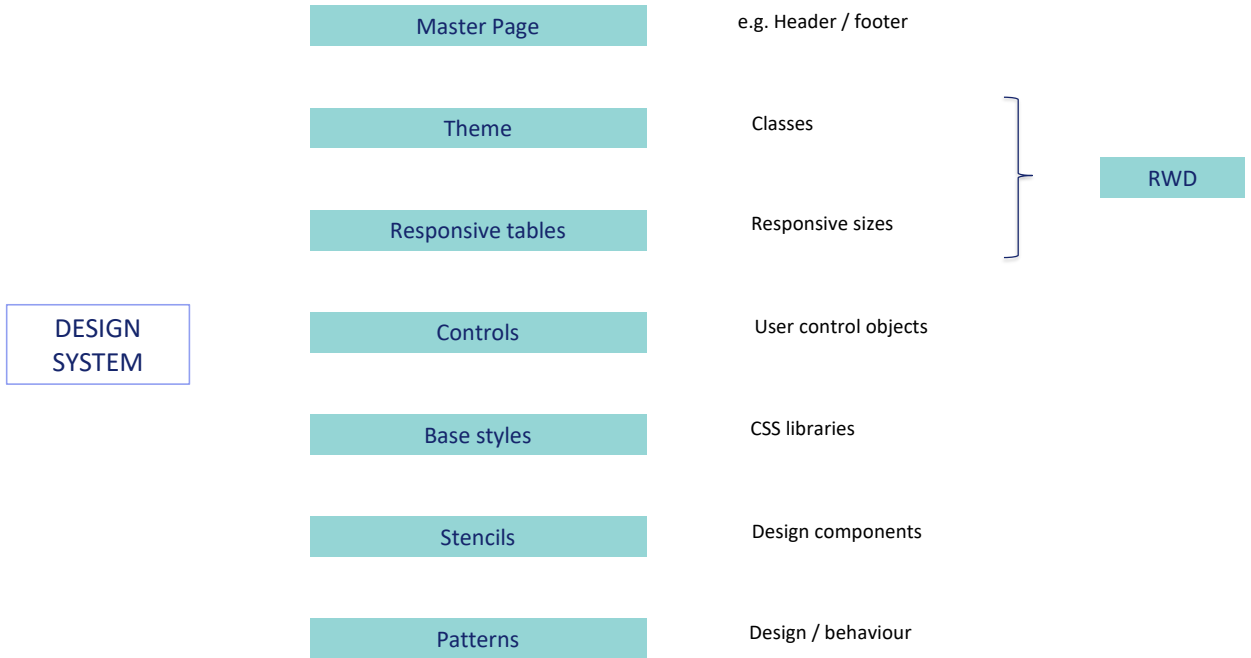


Design System en GeneXus

Default

GeneXus™



En este video abordaremos de manera superficial cada uno de los elementos de GeneXus que participan en la implementación de un Design System para la aplicación.

Back-office

Attraction Without Parameters

Id: 3

Name: Eiffel Tower

Country Id: 2

Country Name: France

Category Id: 2

Category Name: Monument

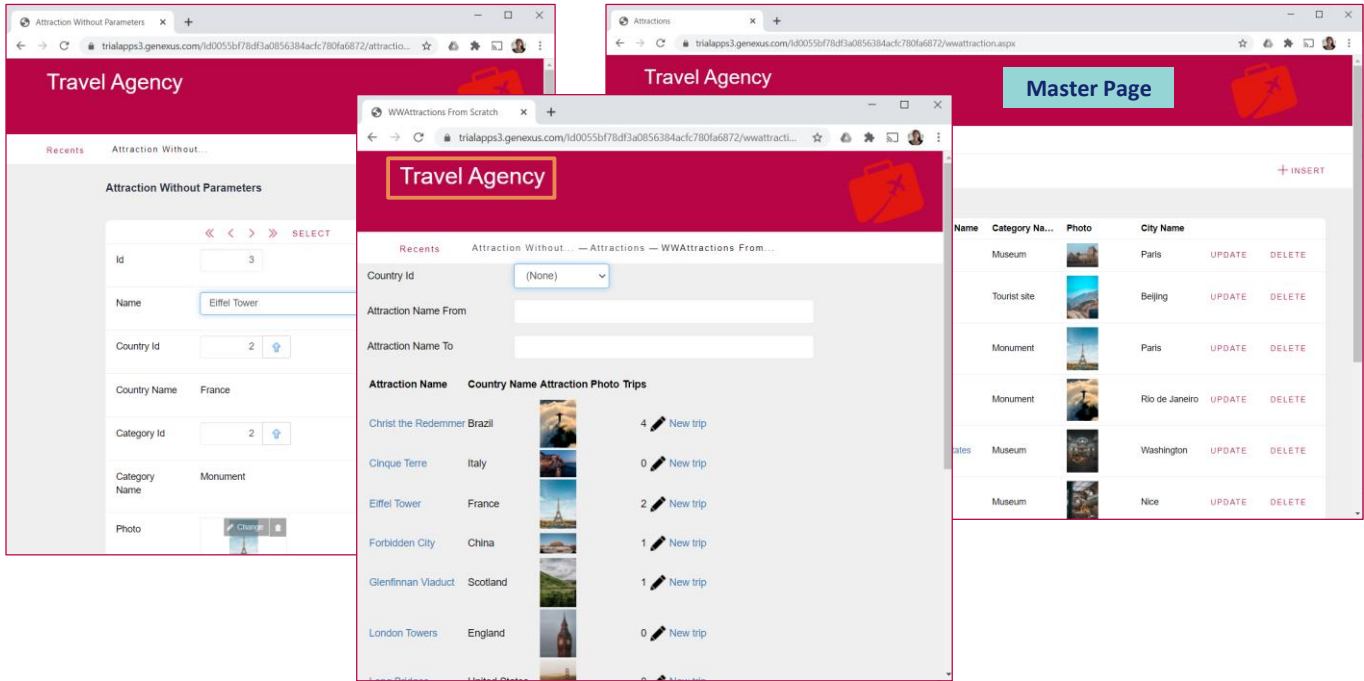
Photo: Change

Attractions

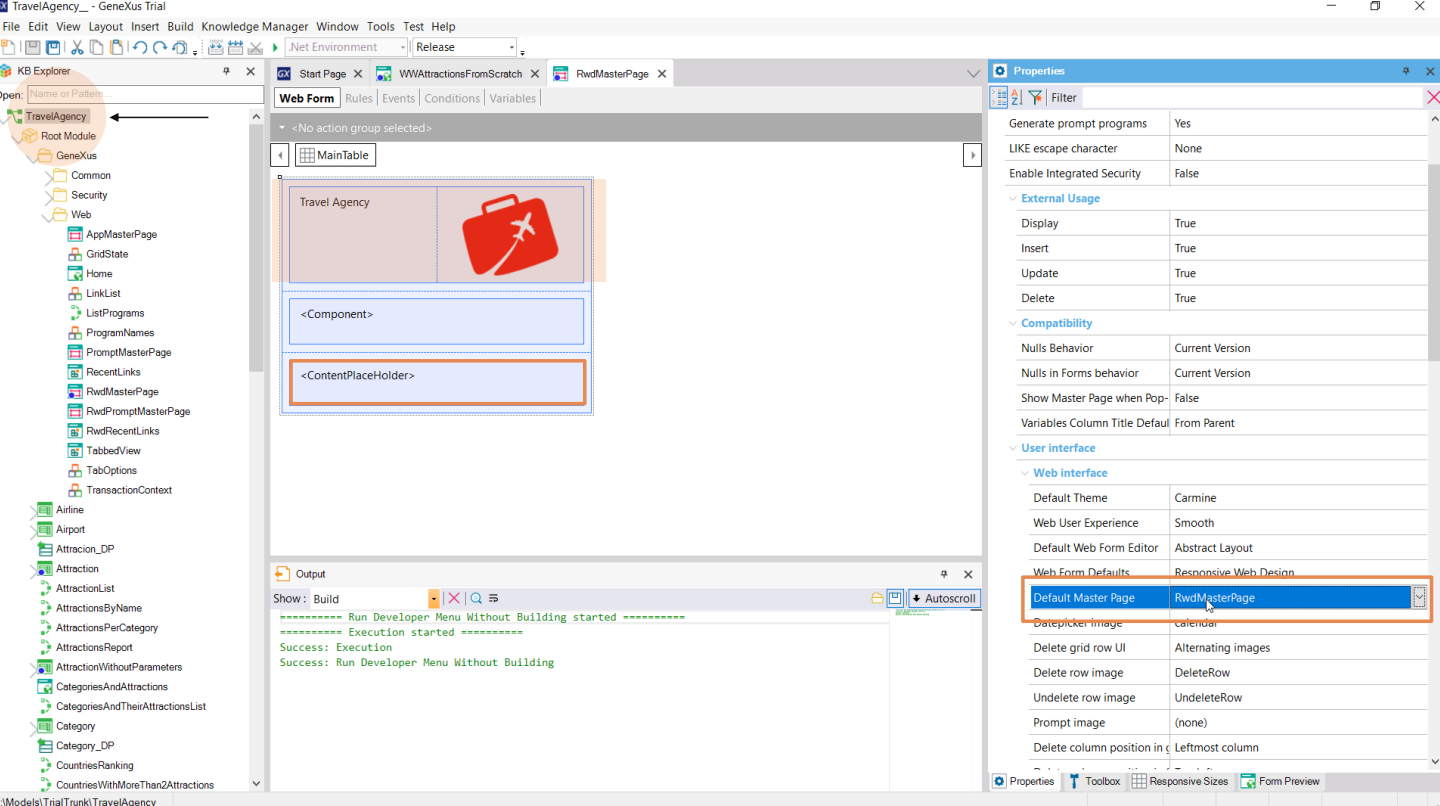
Attractions

Id	Name	Country Name	Category Name	Photo	City Name	UPDATE	DELETE
1	Louvre Museum	France	Museum		Paris	UPDATE	DELETE
2	The Great Wall	China	Tourist site		Beijing	UPDATE	DELETE
3	Eiffel Tower	France	Monument		Paris	UPDATE	DELETE
4	Christ the Redeemer	Brazil	Monument		Rio de Janeiro	UPDATE	DELETE
5	Smithsonian Institute	United States	Museum		Washington	UPDATE	DELETE
6	Matisse Museum	France	Museum		Nice	UPDATE	DELETE

Si prestamos atención a las transacciones que venimos utilizando, y a los paneles creados por el patrón Work With, vemos que hay elementos que dan uniformidad y coherencia, sin que nos hayamos preocupado en lograr esto. Lo que significa que GeneXus ya está haciendo algo por nosotros en lo referido a brindarnos un Design System predeterminado.

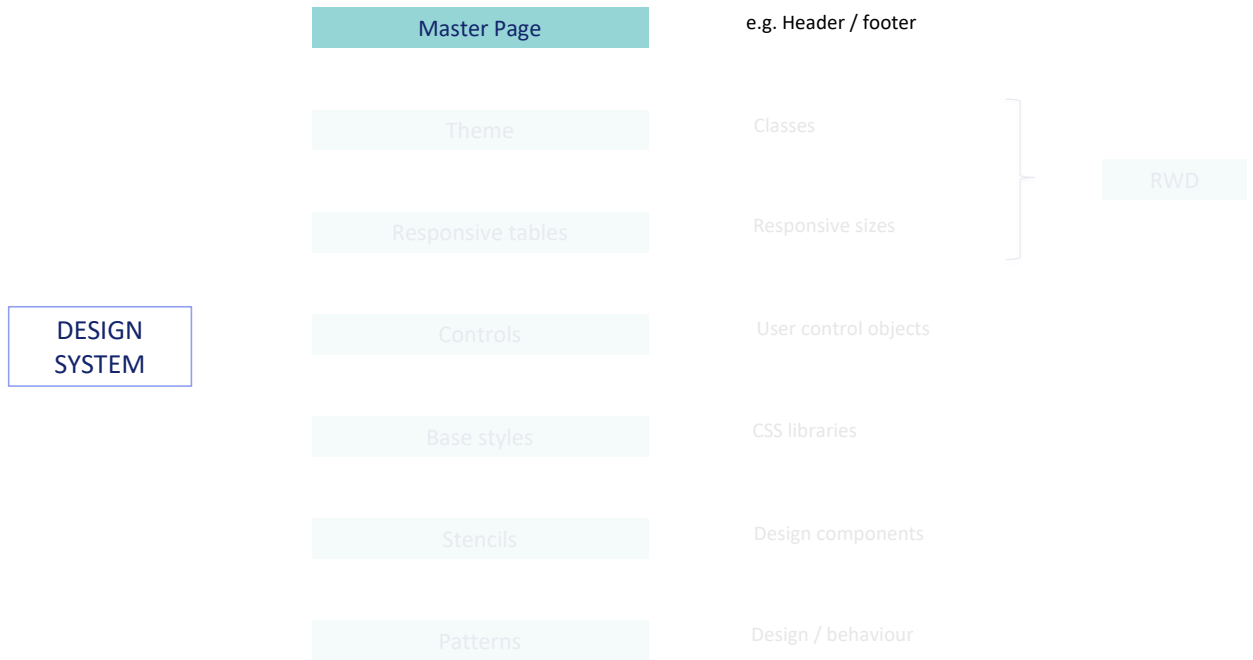


Por ejemplo, vemos que todas las pantallas (incluso las que hicimos de cero, como este web panel) tienen un encabezado común. En nuestro caso lo hemos personalizado, cambiándole el texto default por "Travel Agency", y reemplazando la imagen que aparecía por esta otra.

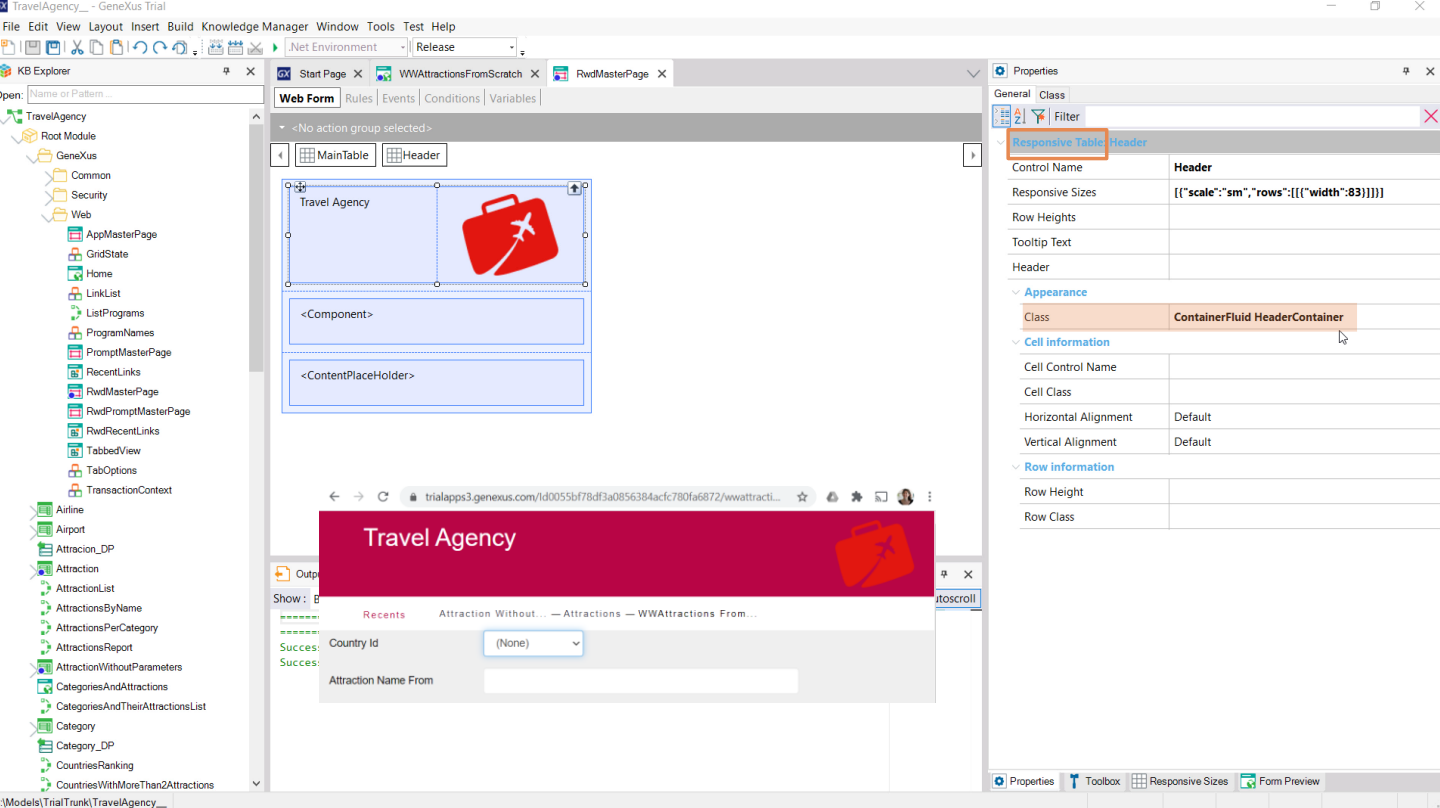


La forma de conseguirlo era con el objeto web Master Page (Aquí vemos los cambios efectuados. Recordemos que luego las distintas pantallas de la aplicación se cargarán en este control, el content place holder.

Por defecto, la versión de la KB viene con esta Master Page predefinida, lo que significa que por defecto se le aplicará a todos los objetos web que se creen, como ya habíamos visto.

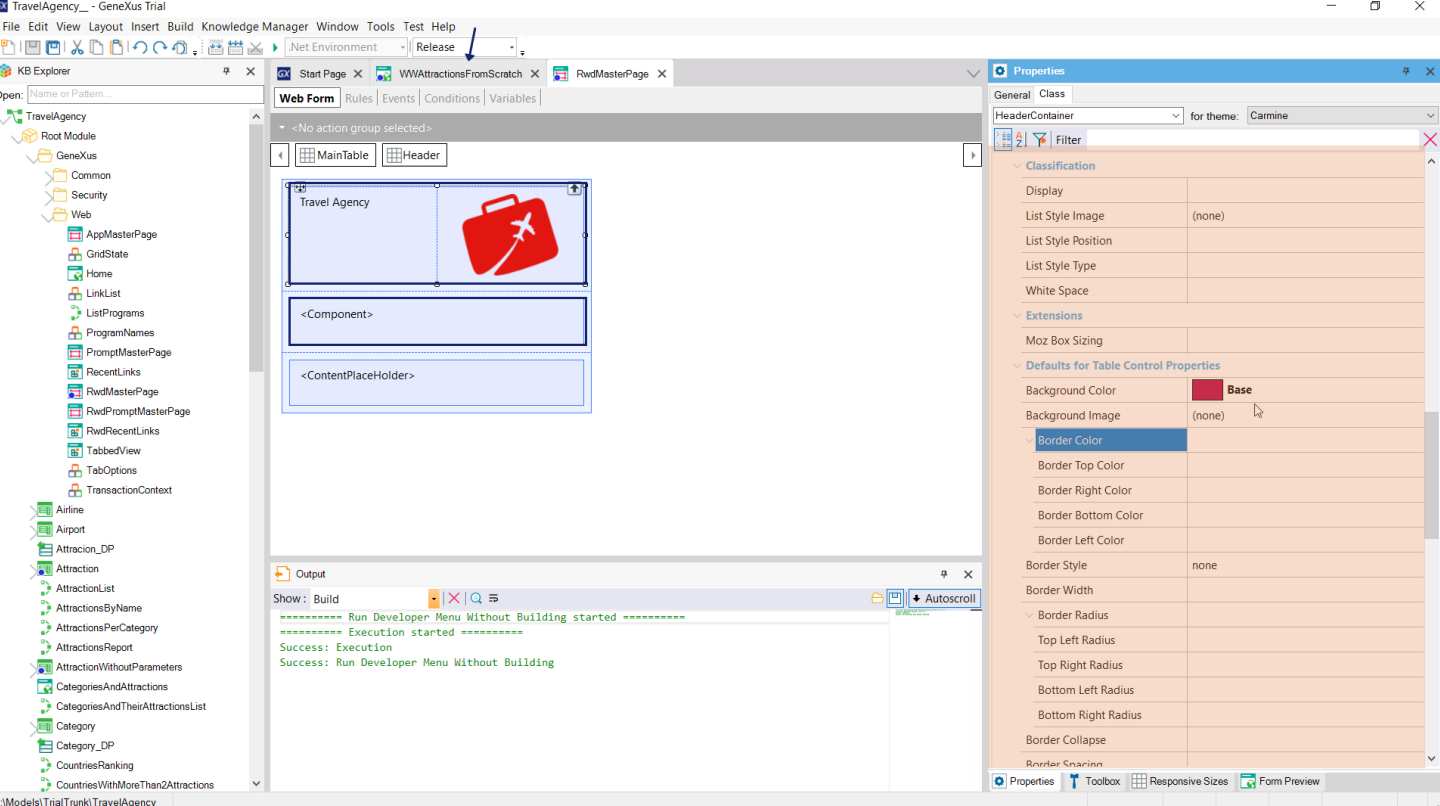


Así que, por un lado GeneXus nos ofrece un objeto para centralizar lo que será la información común que queremos que las páginas compartan, como un encabezado o un pie de página.



Por otro lado, podemos ver que el color de base, el que sobresale, es un tipo de rojo, que se utiliza no solo para el encabezado, sino para los botones y otras acciones que se le brindan al usuario. ¿Dónde se configura todo eso?

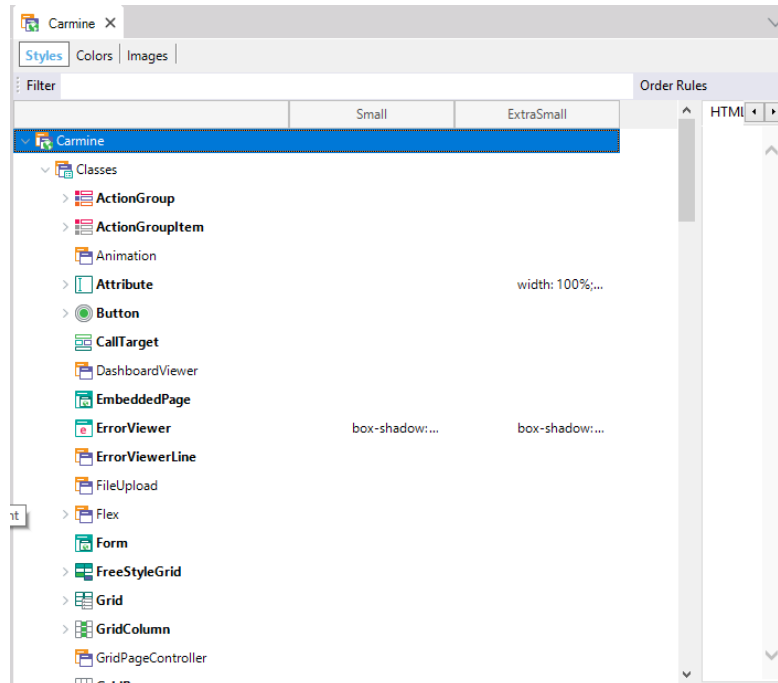
Sabemos que esta tabla tiene un fondo rojo en ejecución, pero ¡no lo estamos viendo! Sin embargo, observemos que el control (en este caso de tipo **tabla responsiva**) tiene un par de **clases** definidas, no casualmente bajo el grupo “Apariencia”.



Si vamos a la **solapa Class**, vemos que se están editando las propiedades de la segunda de las clases asociadas a esa tabla. Y vemos que allí está definido, entre otras cosas, ¡el color de fondo rojo que veíamos!

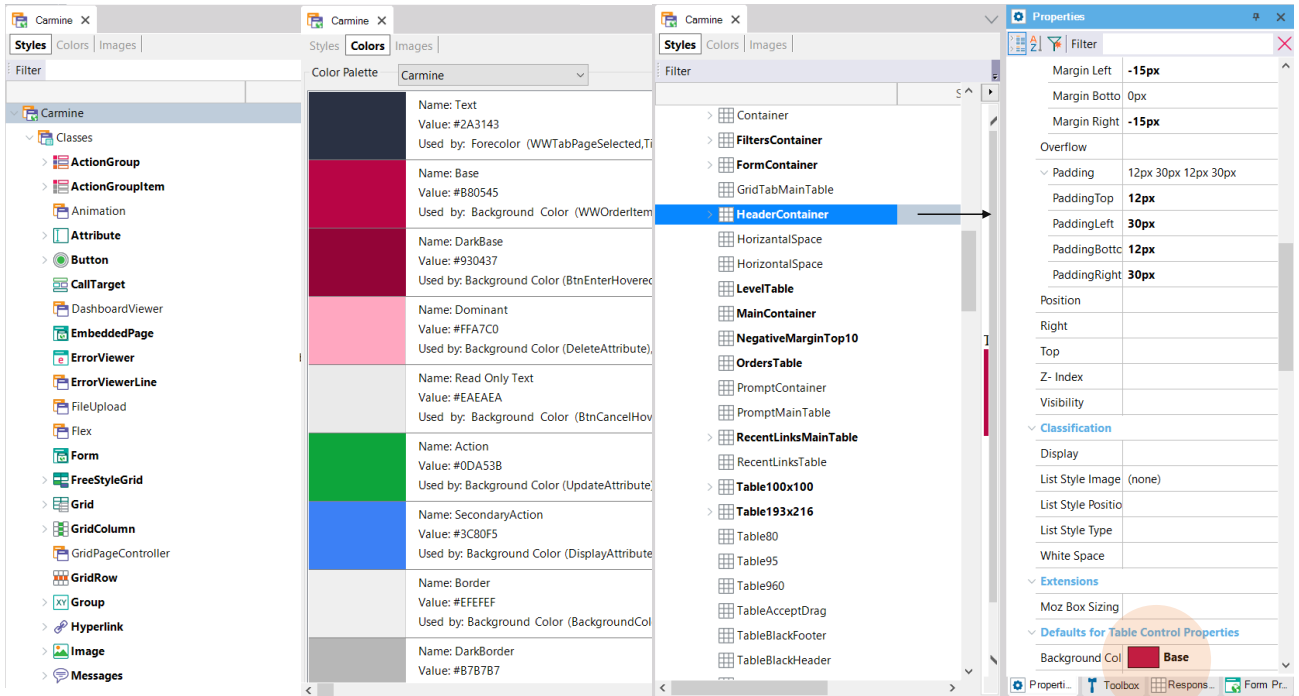
La gracia de las clases es que permiten centralizar el diseño de ese tipo de control (en este caso, tabla), y que, entonces, en nuestro caso otras tablas que se encuentren en el mismo o en otros forms puedan compartir la misma definición, lo que hace que si queremos cambiar por ejemplo el color de fondo de rojo a azul para esos controles, no haya que hacerlo uno por uno en cada control, es decir, en cada tabla, sino que directamente se hace en la clase y eso ya repercute en todos los controles que la tengan asociada.

Theme



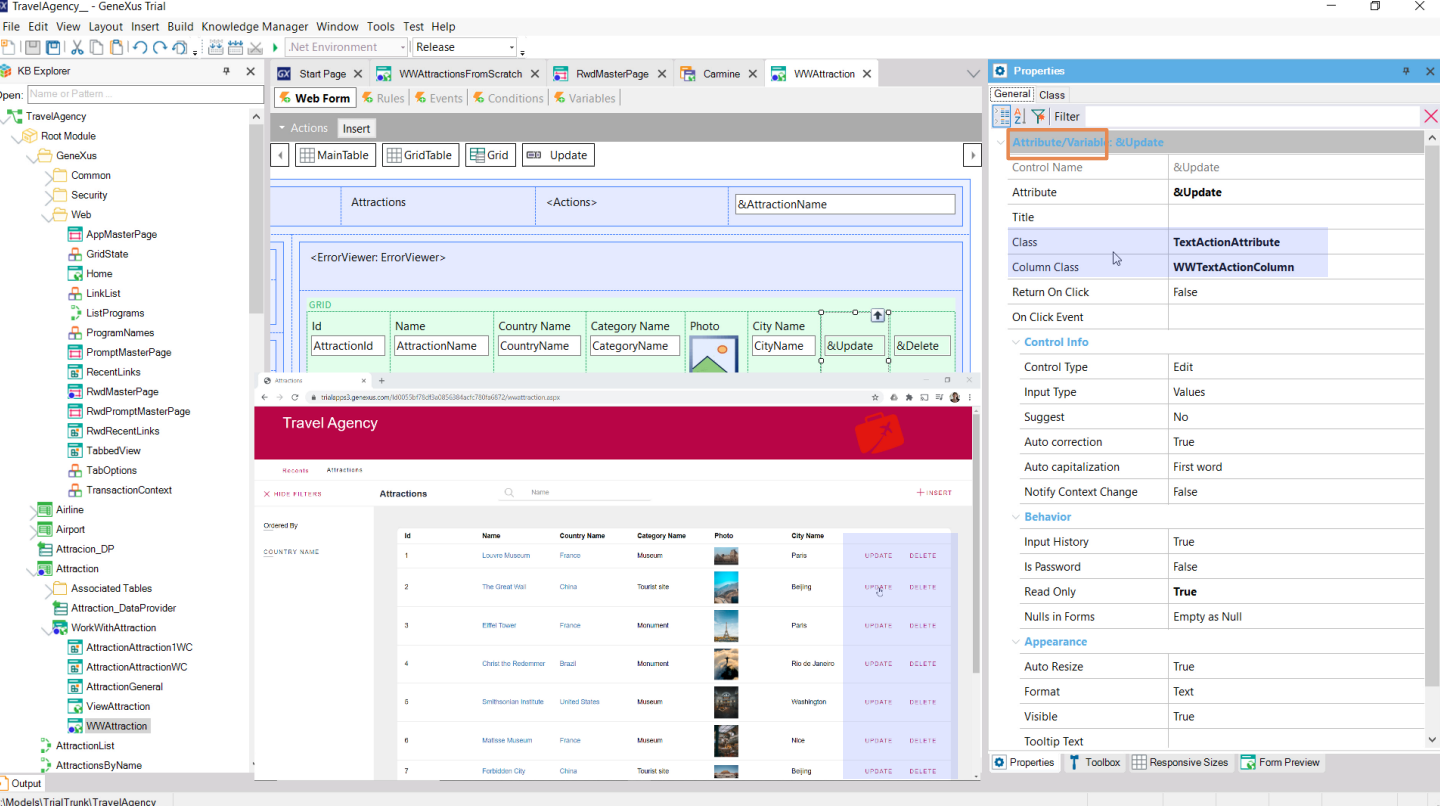
¿Dónde se encuentran todas las clases definidas? ¡En el objeto Theme, Tema!

Vemos que, así como había una Master Page predefinida, también hay un tema predefinido, de nombre Carmine, que ya lo vimos repetidamente asociado a todos nuestros paneles web.



Vemos que al abrirlo aparecen clases agrupadas por tipo de control. Aquí es donde está centralizado todo. Y además podemos ver en la solapa Colors, que muestra la paleta de colores del mismo nombre... Que es donde está definido el color Base.

Si buscamos subclases de la clase Table, allí encontramos a HeaderContainer, una de las dos clases que tenía asociada la tabla de la Master Page. Lo que modifiquemos en un lado se modificará en el otro. Aquello que vimos era un shortcut a este lugar.



Si vamos a ver las opciones de Update o Delete del pattern Work with, que sabemos tienen el color base en ejecución, vemos que el tipo de control es Attribute/Variable. Y que tiene asignado como clase a esta, predefinida por el pattern. Podemos aquí mismo editar sus propiedades sabiendo que lo que modifiquemos aquí se modificará a nivel central, en el objeto Theme, por lo que cualquier otro control atributo/variable que tenga esta como clase, se verá modificado.

En definitiva, todo control en un form tiene propiedades individuales, que se pueden configurar para él solo... Y propiedades que vendrán de la clase que tenga asignada, que son las que involucran más que nada los aspectos de diseño, y que serán compartidas por muchos controles del mismo tipo.

Ya observemos que en este caso particular en que el control es, además, una columna de un grid, aparece otra clase para asociarle, que es la clase que gobierna el diseño y comportamiento de la columna.

Attraction Without Parameters

Travel Agency

Attraction Without Parameters

Responsive table

Id: 3

Name: Eiffel Tower

Country Id: 2

Country Name: France

Category Id: 2

Category Name: Monument

Photo:

Attractions

Travel Agency

Grid

Attractions

Ordered By: COUNTRY NAME

Id	Name	Country Name	Category Na...	Photo	City Name	UPDATE	DELETE
1	Louvre Museum	France	Museum		Paris	UPDATE	DELETE
2	The Great Wall	China	Tourist site		Beijing	UPDATE	DELETE
3	Eiffel Tower	France	Monument		Paris	UPDATE	DELETE
4	Christ the Redeemer	Brazil	Monument		Rio de Janeiro	UPDATE	DELETE
5	Smithsonian Institute	United States	Museum		Washington	UPDATE	DELETE
6	Matisse Museum	France	Museum		Nice	UPDATE	DELETE

Si observamos nuevamente el diseño predefinido de la transacción y el Work With, vemos que los datos más relevantes se muestran en blanco sobre gris, con bordes redondeados.

Responsive table

The image shows two screenshots from the GeneXus IDE. The left screenshot displays a form titled 'AttractionWithoutParameters' with a table containing the following fields:

Attraction Without Parameters	
<ErrorViewer: ErrorViewer>	
<Toolbar>	
Id	AttractionId
Name	AttractionName
Country Id	CountryId
Country Name	CountryName
Category Id	CategoryId
Category Name	CategoryName
Photo	
City Id	CityId
City Name	CityName

The right screenshot shows the 'Properties' window for the 'FormContainer' class. The 'Responsive Table: FormContainer' section is expanded, showing the following properties:

- Control Name: **FormContainer**
- Responsive Sizes: `[[{"scale": "sm", "rows": [{"width": ...`
- Row Heights: `.....`
- Header: `.....`
- Appearance:
 - Class: **FormContainer**
- Cell Information:
 - Cell Control Name: `.....`
 - Cell Class: `.....`
 - Horizontal Alignment: Default
 - Vertical Alignment: Default
- Row information:
 - Row Height: `.....`
 - Row Class: `.....`

The second 'Properties' window on the right shows the 'FormContainer' class with the following properties:

- List Style Type: `.....`
- White Space: `.....`
- Extensions:
 - Moz Box Sizing: `.....`
- Defaults for Table Control Properties:
 - Background Color: **white**
 - Background Image: (none)
 - Border Color: `.....`
 - Border Information:
 - Border Top Color: `.....`
 - Border Right Color: `.....`
 - Border Bottom Color: `.....`
 - Border Left Color: `.....`
 - Border Style: `.....`
 - Border Width: `.....`
 - Border Radius:
 - Border Radius: **8px**
 - Top Left Radius: **8px**
 - Top Right Radius: **8px**
 - Bottom Left Radius: **8px**
 - Bottom Right Radius: **8px**
 - Border Collapse: `.....`
 - Border Spacing: `.....`
 - Lines Font: `_ _ 400 14px _ Arial`

Esto es así porque a la tabla donde se encuentran esos datos en la transacción se le asignó esta clase... que especifica como color de fondo el blanco y un radio para los cuatro bordes...

The screenshot shows the GeneXus IDE interface. The main window displays a web form design for 'WWAttraction'. The form includes a search bar for '&AttractionName', a filter section with 'Name' and 'Country' options, and a grid of attraction data. The grid has columns for 'Id', 'Name', 'Country Name', 'Category Name', 'Photo', 'City Name', and actions '&Update' and '&Delete'. The properties panel on the right shows the 'Grid' control settings, including 'WorkWith' class, 'Rows' set to 10, and 'PagingMode' set to 'One page at a time'.

Work With Grid

...y al grid del Work With esta otra clase, que especifica también blanco como color de fondo, y los mismos valores que la otra para el radio.

The image shows the GeneXus IDE interface. On the left, the design view displays a web form with a grid. The grid has columns for 'Attraction Id', 'Attraction Name', 'Attraction Photo', 'Trips', '&update2', and '&newTrip'. Below the grid is a 'Total Trips' label with the expression '&totalTrips'. A green box labeled 'Default Grid' points to the grid. On the right, the Properties window shows the 'Grid: Grid1' class with the following properties:

Property	Value
Control Name	Grid1
Collection	
Base Trm	Attraction
Order	CountryId, AttractionName when not &Coun...
Conditions	CountryId = &CountryId when not &Country...
Unique	
Save State	False
Data Selector	(none)

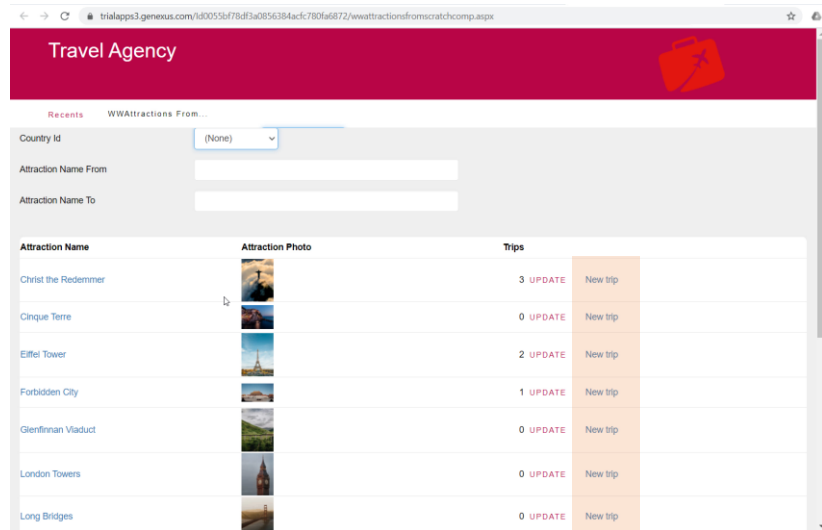
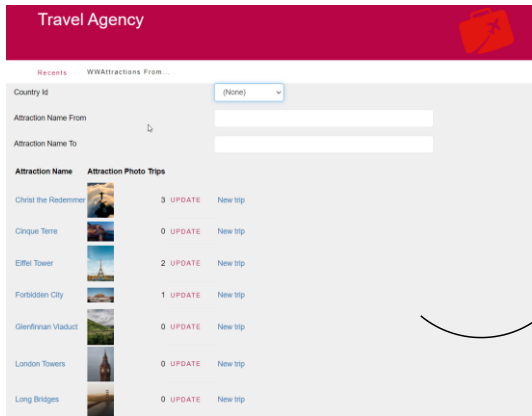
The 'Appearance' section of the Properties window shows:

Property	Value
Class	Grid
Custom Render	
Empty Grid Text	
Auto Resize	True
Width	
Height	
Rows	0

Below the Properties window, a preview of the web panel is shown. It features a red header 'Travel Agency' and a table of attractions with columns for 'Attraction Name', 'Attraction Photo', and 'Trips'. The table lists attractions like 'Christ the Redeemer', 'Cinque Terre', 'Eiffel Tower', etc., with update counts and 'New trip' links.

En cambio, si vamos a ver el grid del web panel que nosotros creamos, vemos que tiene como clase la de nombre Grid, que no tiene nada de eso configurado... Y así se ve en ejecución, con todas las columnas pegadas, con el fondo default, gris.

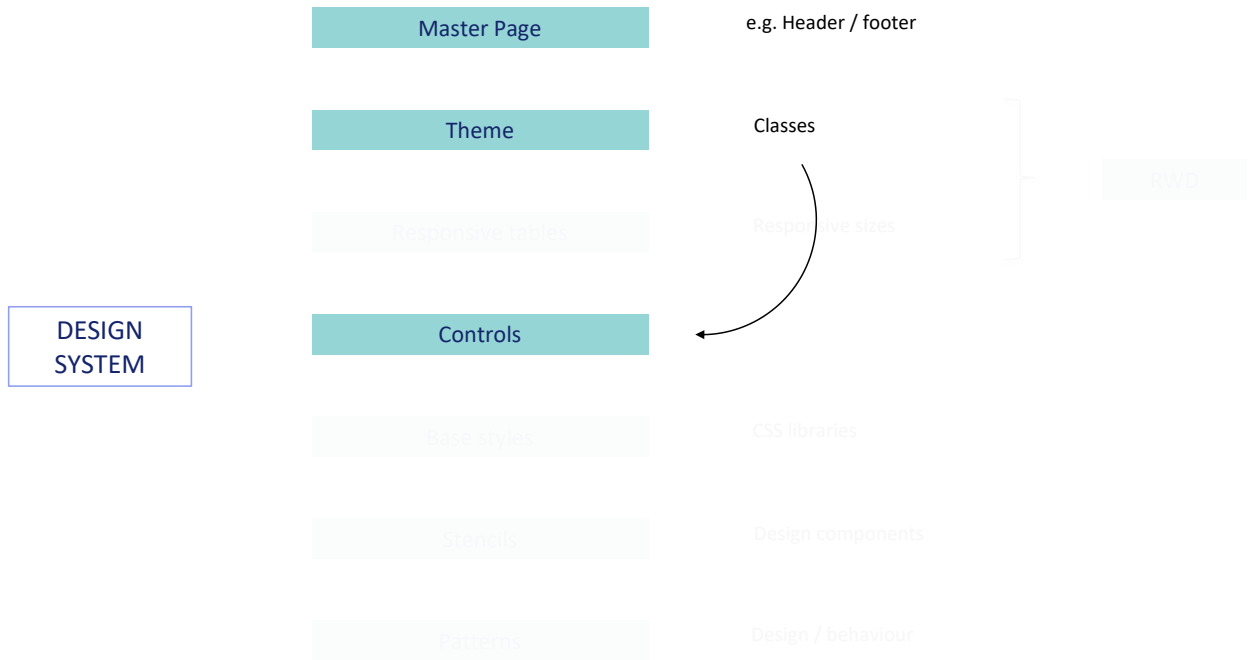
Class: WorkWith



Class	TextActionAttribute
Column Class	
Return On Click	False

Observemos qué sucede si le cambiamos su clase por la misma que tiene el grid del WorkWith...

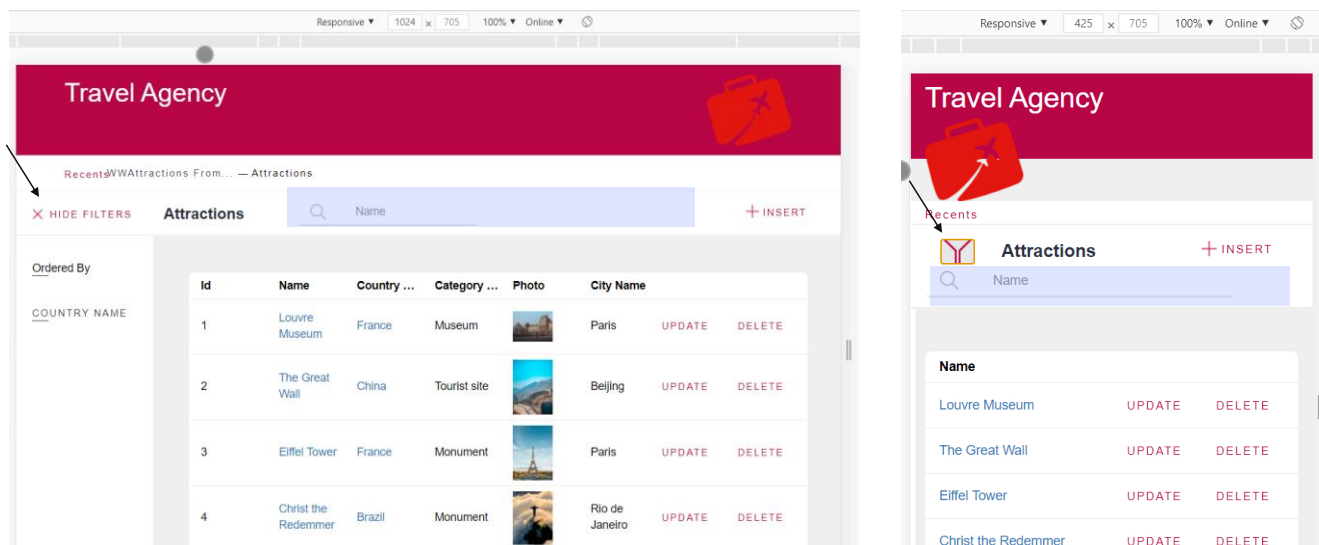
Y para mantener la uniformidad del Design System, ya habíamos cambiado para la acción de update la imagen por el text block, para que sea idéntico al del Work With, pero también esta acción del grid debería lucir como estas otras...



Podemos deducir de lo que hemos visto que un porcentaje importantísimo del Design System se especifica en el objeto Theme, a través de sus clases (las predefinidas y las que agreguemos). Estas clases son las que se asignan a los controles del form, permitiendo así abstraer el diseño, establecer una lógica, y de esta manera uniformizarlo de acuerdo a la función que asume cada control en el todo.

Por lo que una parte fundamental del diseño será identificar qué clases necesitaremos para los controles, y aplicárselas.

Responsive Web Design (RWD)



Por otro lado, volviendo a lo predefinido (recordemos que algunos navegadores nos ofrecen un modo de visualización –en este caso presionando F12– que nos permite variar el tamaño de la pantalla para apreciar cómo se verá el panel en ese tamaño)...

Podemos ver que si achicamos la pantalla del Work with de atracciones lo suficiente como para que corresponda a la pantalla de un teléfono, los filtros aparecen de otra manera. Y, por ejemplo el campo para que el usuario pueda ingresar un filtro por nombre de atracción aparece debajo del título y no a su derecha. Además los recent links aparecen como menú desplegable, en lugar de expandidos. Y la imagen de la master page está apareciendo debajo y no al costado.

Y con la disminución del ancho de pantalla, también desaparecen casi todas las columnas a excepción de la que muestra el nombre y las acciones de Update y Delete.

Este comportamiento se conoce como “responsivo”, y al diseño web que lo implementa como Responsive Web Design. Hoy en día no es aceptable un diseño que no adapte lo que se ve al tamaño de pantalla.

Una parte de la responsividad se establece a través de las **tablas responsivas**...

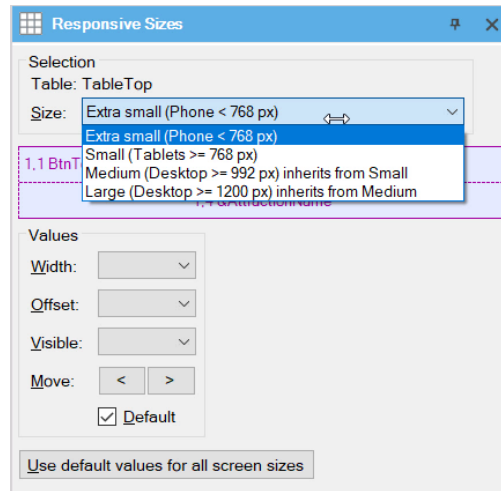
Responsive tables

The screenshot displays the GeneXus IDE interface for creating a responsive table. The main workspace shows a design for a travel agency application. At the top, there's a navigation bar with 'Web Form', 'Rules', 'Events', 'Conditions', and 'Variables'. Below it, the 'Actions' and 'Insert' toolbars are visible. The design canvas features a 'TableTop' control with a header 'Travel Agency' and a search bar. The table below has columns for 'Name' and 'Country'. The 'Properties' panel on the right shows the configuration for the 'TableTop' control, including 'Responsive Sizes' set to '[[{"scale": "xs", "rows": [{"width": ...}]]'.

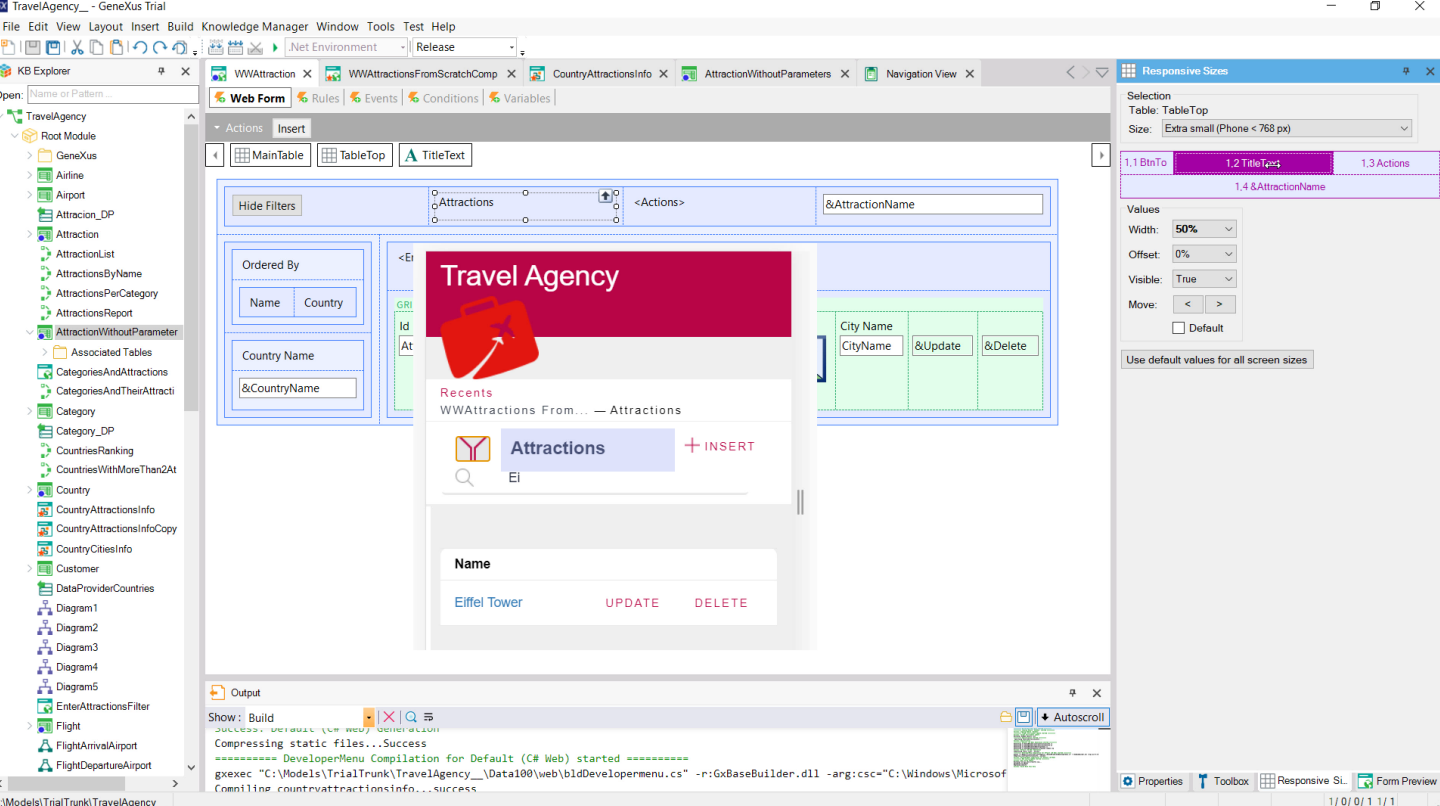
Por ejemplo, esta tabla del Work With de atracciones que creó el pattern, que vemos que es responsiva, contiene estos cuatro controles. El primero corresponde a los filtros adicionales, el segundo corresponde a este textblock, el tercero a la acción de Insert, y el último corresponde a la variable de filtro que veíamos.

Si editamos las propiedades de la tabla, vemos ésta, “Responsive Sizes” (tamaños responsivos).

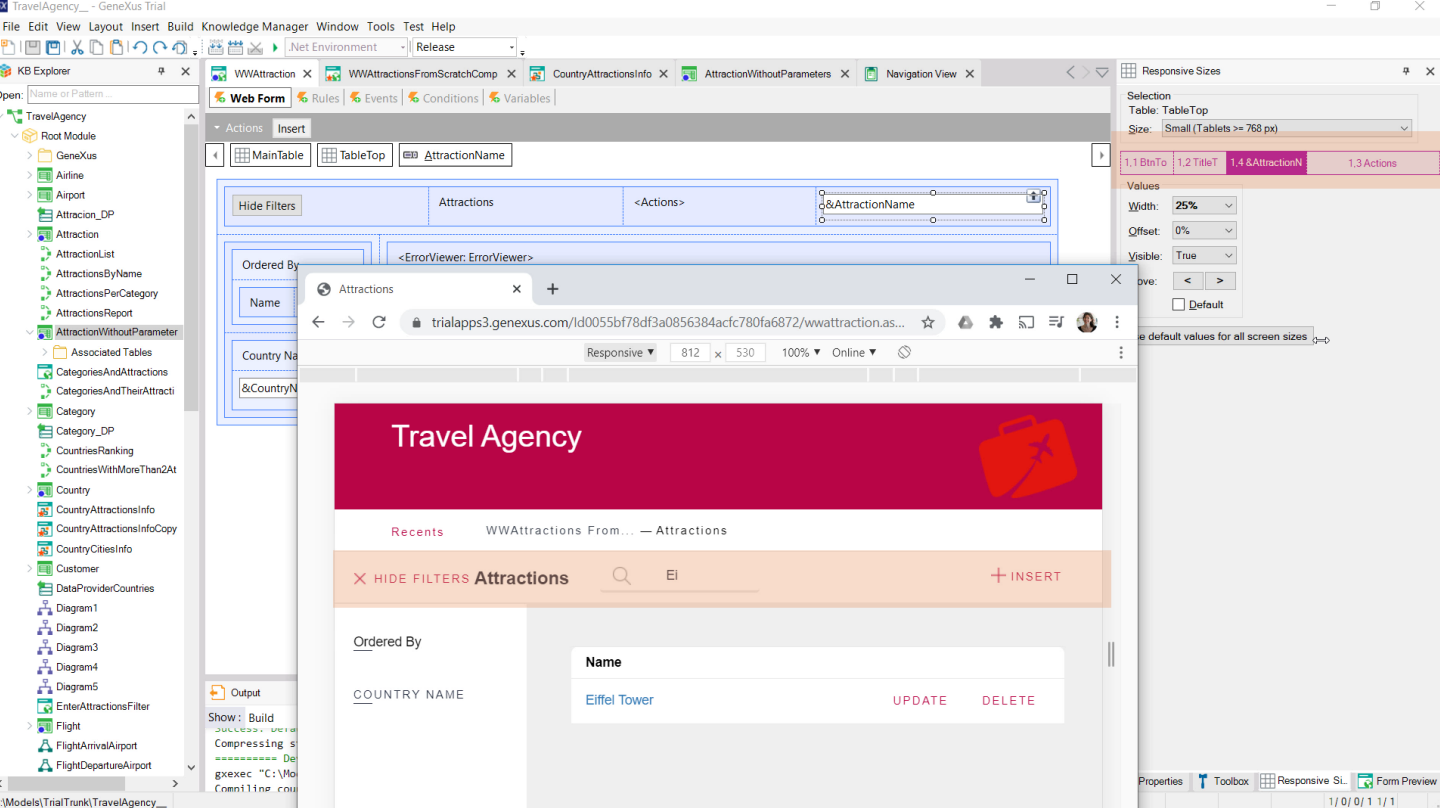
Responsive tables



Aquí se establece cómo se quiere visualizar el contenido de la tabla de acuerdo a cuatro tamaños de pantalla: **Extra small**, que corresponde a teléfonos, **Small**, que corresponde a tablets **Medium** que corresponde a monitores de notebooks o Pcs menores a 1200 píxeles; y **Large** que corresponde a tamaños mayores.



Aquí podemos ver que si el tamaño es **Extra small**, aparecerá primero el botón para habilitar los otros filtros, ocupando el 17% del ancho de pantalla, luego a su derecha vendrá el título, que ocupará el 50% del ancho restante, y por último las acciones, que aquí serán únicamente el Insert, ocupando el restante 33%, y la variable de filtro aparecerá debajo de todo eso, ocupando el 100%.

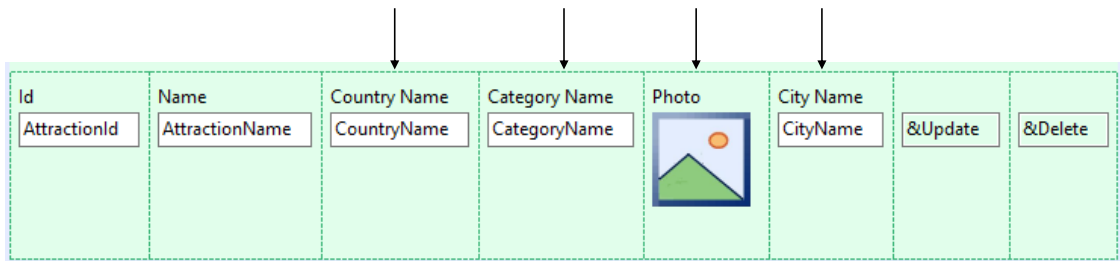



Mientras que si el tamaño es **Small**, estará todo en una misma fila, con botón, título, variable y acción, en ese orden.

Responsive Web Design (RWD)

- Responsive Sizes of Responsive tables
- Rules of Theme: Small, Extra-small and Default

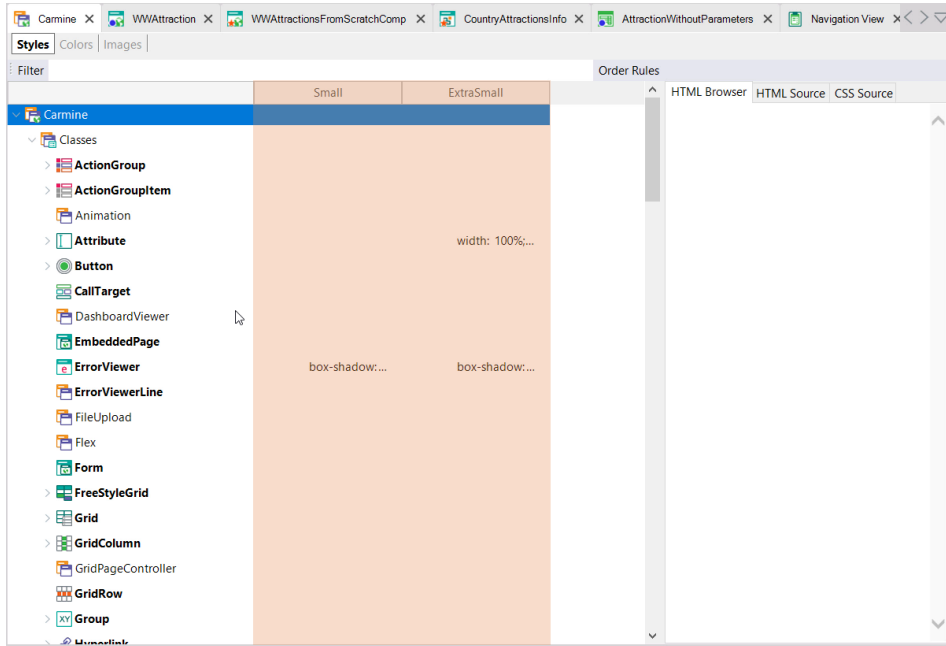
ie: to hide grid columns if Extra small or Small size



Id	Name	Country Name	Category Name	Photo	City Name	&Update	&Delete
AttractionId	AttractionName	CountryName	CategoryName		CityName		

Decíamos que una parte importante de la responsividad, la más gruesa, se consigue haciendo uso de estas tablas y estas propiedades. Pero otra parte, un poco más fina, se consigue a través de las **clases**.

RWD: Theme rule columns

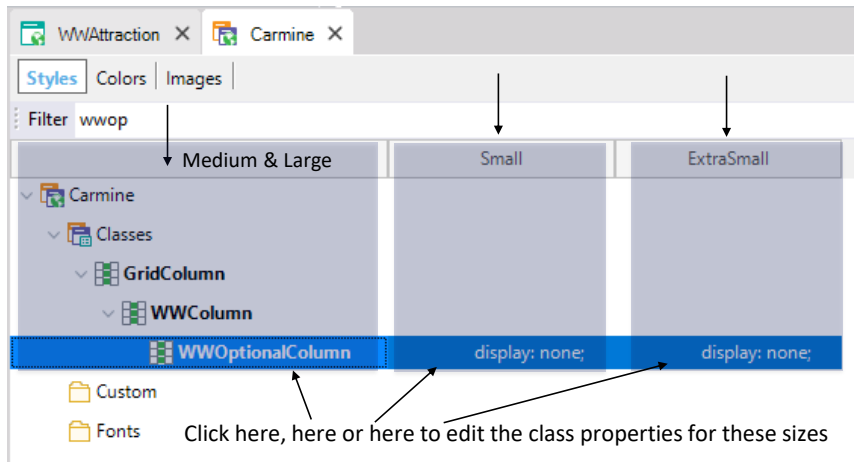


Si vamos al theme, vemos están apareciendo estas dos columnas que lo que hacen es permitirnos variar los valores de las propiedades de la clase, de acuerdo al tamaño de pantalla.

Por defecto solo se establecen dos: Small y ExtraSmall. Pero en verdad tenemos tres, puesto que los valores que vemos en la columna default corresponden a Medium y Large.

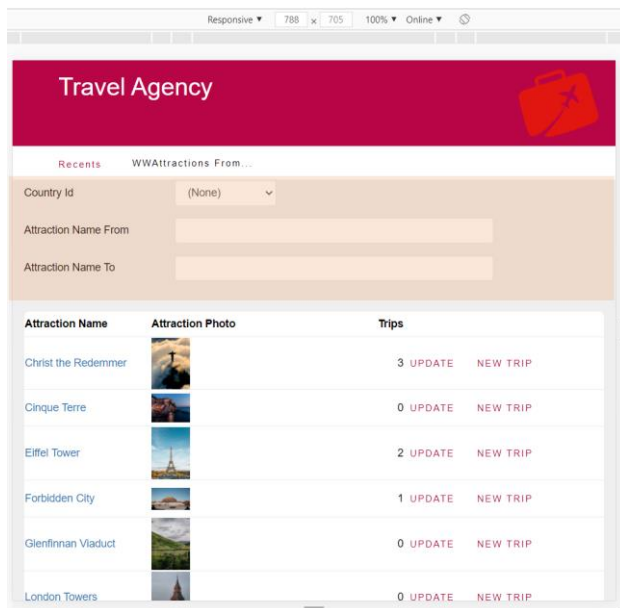
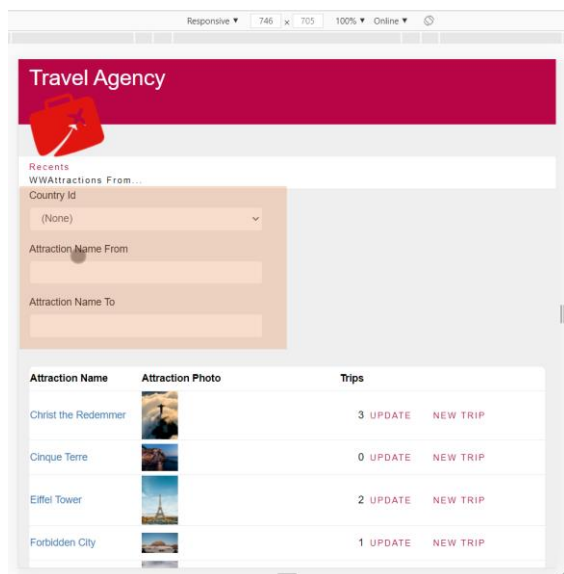
RWD: Theme rule columns

- Rules of Theme: Small, Extra-small and Default



Entonces aquí podemos ver que, mientras que la propiedad Display para esta clase no dice nada, lo que significa que se mostrará en ejecución para tamaños de pantalla Medium y Large, para tamaño Small y ExtraSmall tiene el valor "none". Y es por eso que no se muestran.

How to improve responsiveness in our web panel



En cambio en el web panel que implementamos de cero no tenemos este comportamiento responsivo para el grid. Al disminuir el tamaño de pantalla seguimos viendo las mismas columnas.

La única responsividad predefinida que tenemos es la que viene de la Master Page, donde para tamaño Extra-small la imagen está quedando abajo y no a la derecha y la lista de links recientes aparece como menú vertical, y para nuestro web panel propiamente dicho, la que involucra los campos de filtro, que están ocupando el 100% del ancho con su etiqueta arriba, cuando para tamaños Small en adelante, quedan a la izquierda.

How to hide columns when Small or Extra-small







Travel Agency

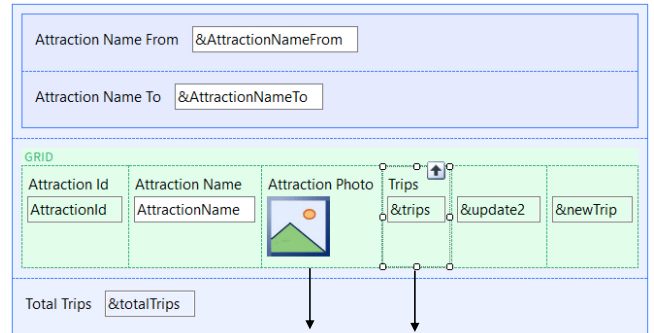
Recents WWAttractions From...

Country Id (None) ▾

Attraction Name From

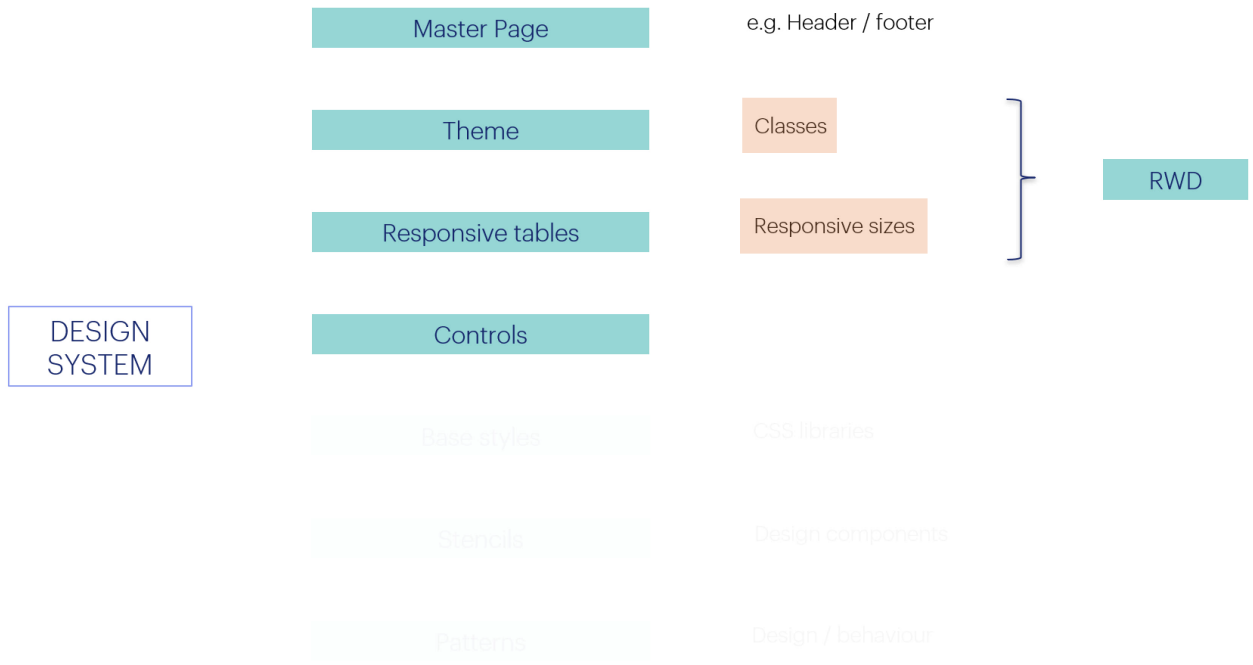
Attraction Name To

Attraction Name	Attraction Photo	Trips	
Christ the Redemmer		3	UPDATE NEW TRIP
Cinque Terre		0	UPDATE NEW TRIP
Eiffel Tower		2	UPDATE NEW TRIP
Forbidden City		1	UPDATE NEW TRIP
Glenfinnan Viaduct		0	UPDATE NEW TRIP
London Towers		0	UPDATE NEW TRIP

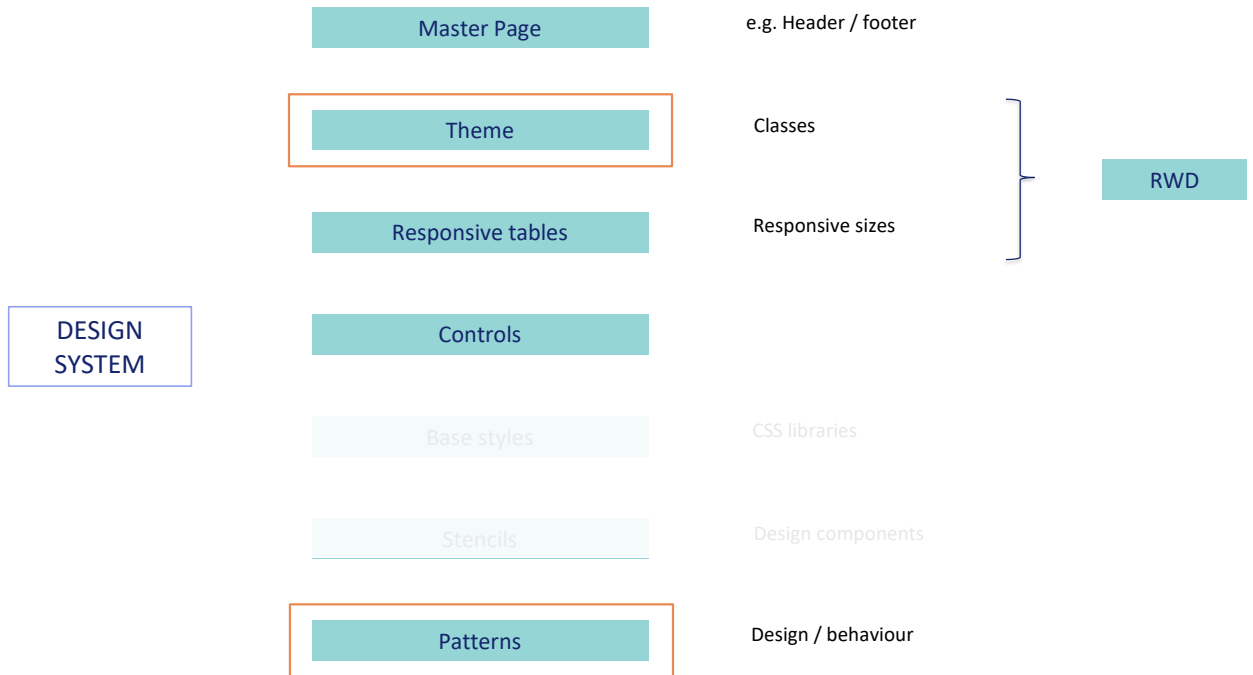


Column Class: WWOptionalColumn

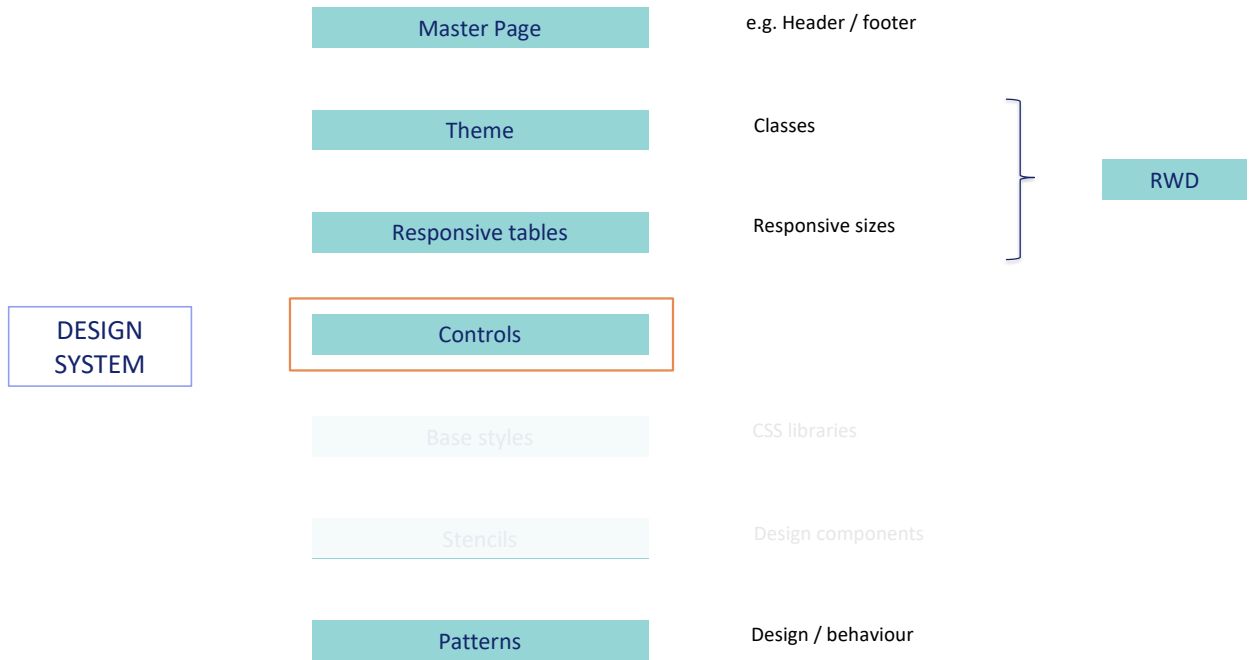
Entonces, si queremos conseguir para el grid de nuestro web panel, que en tamaño Small no se despliegan otras columnas que las del nombre de atracción y acciones, modificamos las clases de las columnas de acuerdo a lo que acabamos de ver.



Resumiendo: otra parte importante del Design System tiene que ver con la responsividad, que se consigue tanto variando las posiciones y la visibilidad de los controles de las responsive tables, como variando las propiedades de las clases de acuerdo también al tamaño (a través de esas columnas del Theme que veíamos).

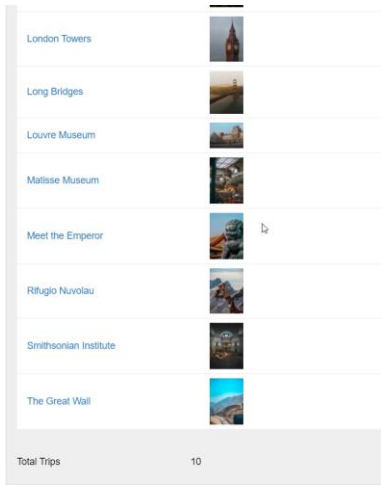


Para conseguir todo esto no tuvimos que hacer casi nada. Solo imitar lo que el pattern Work With hizo automáticamente, en conjunción con el theme. Por eso decimos que GeneXus ya nos brinda un Design System básico, predefinido, que podemos utilizar para nuestros web paneles.

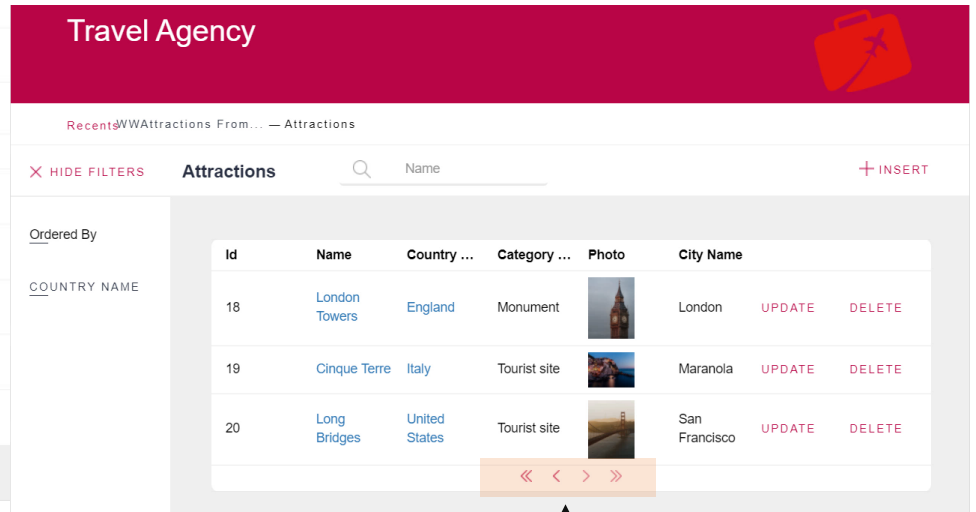


Otra parte de todo esto tendrá que ver con los controles que utilizemos.

Grids: design and behavior



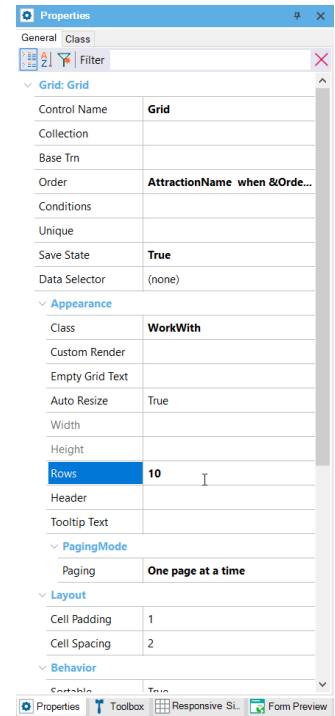
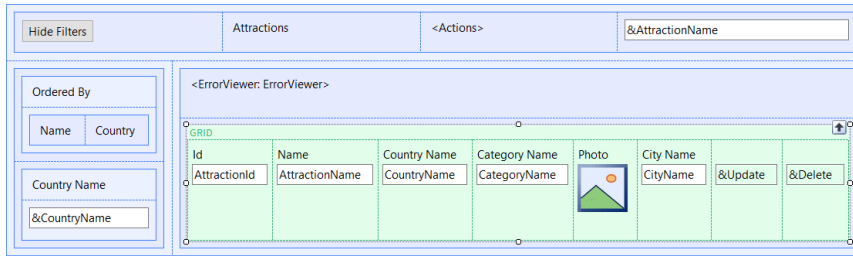
Our web panel: no grid paging



Work With web panel: grid paging

Por ejemplo, en nuestro web panel, donde imitábamos al del Work With, no tenemos el paginado que el Work With sí presenta.

Grid Paging



Si observamos las propiedades del grid, vemos la de nombre Rows, que tiene el valor 10, junto con la propiedad PagingMode que tiene este valor. Hagamos lo mismo en nuestro grid.

Grid Paging

Attraction Name From

Attraction Name To

Attraction Id	Attraction Name	Attraction Photo	Trips		
AttractionId	AttractionName		&trips	&update2	&newTrip

Total Trips

Properties

General Class

Filter

Grid: Grid1

Control Name	Grid1
Collection	
Base Trn	Attraction
Order	CountryId, AttractionName w...
Conditions	CountryId = &CountryId whe...
Unique	
Save State	False
Data Selector	(none)

Appearance

Class	WorkWith
Custom Render	
Empty Grid Text	
Auto Resize	True
Width	
Height	
Rows	0
Header	
Tooltip Text	

Layout

Cell Padding	1
Cell Spacing	2

Behavior

Sortable	True
Allow Drop	False
Allow Drag	False
Allow Drop	False

Properties

General Class

Filter

Grid: Grid1

Control Name	Grid1
Collection	
Base Trn	Attraction
Order	CountryId, AttractionName w...
Conditions	CountryId = &CountryId whe...
Unique	
Save State	False
Data Selector	(none)

Appearance

Class	WorkWith
Custom Render	
Empty Grid Text	
Auto Resize	True
Width	
Height	
Rows	10
Header	
Tooltip Text	

PagingMode

Paging	One page at a time
--------	--------------------

Layout

Cell Padding	1
Cell Spacing	2


Behavior

Sortable	True
Allow Drop	False
Allow Drag	False
Allow Drop	False

El valor 0 indica que se carguen todas las filas. Cuando le cambiamos su valor a 10, le estamos diciendo que pague, lo que significa: que traiga de a 10 registros de la base de datos. Vemos que aparece la propiedad Paging con el mismo valor que tenía para el work with.

Grid Paging

Travel Agency






Recent VW Attractions From...

Country Id

Attraction Name From

Attraction Name To

Attraction Name	Attraction Photo	Trips		
Rifugio Nuvolau		0	UPDATE	NEW TRIP
Smithsonian Institute		1	UPDATE	NEW TRIP
The Great Wall		0	UPDATE	NEW TRIP

« < > »

Total Trips 1

Generemos este objeto y probemos.

Aquí vemos que se habían cargado todas las atracciones turísticas, pero si ahora refrescamos, vemos al paginado en acción.

Grid Paging: infinite scrolling

The screenshot displays the GeneXus development environment. On the left, the Properties window for 'Grid1' is open, showing the following settings:

- General Class:** Control Name: Grid1, Collection: Attraction, Order: CountryId, AttractionName w..., Conditions: CountryId = &CountryId whe..., Unique: Save State: False, Data Selector: (none)
- Appearance:** Class: WorkWith, Custom Render: Empty Grid Text: Auto Resize: True
- Layout:** Width: Height: Rows: 5, Header: Tooltip Text: PagingMode: Paging (Infinite scrolling), Scroll Bar: Grid

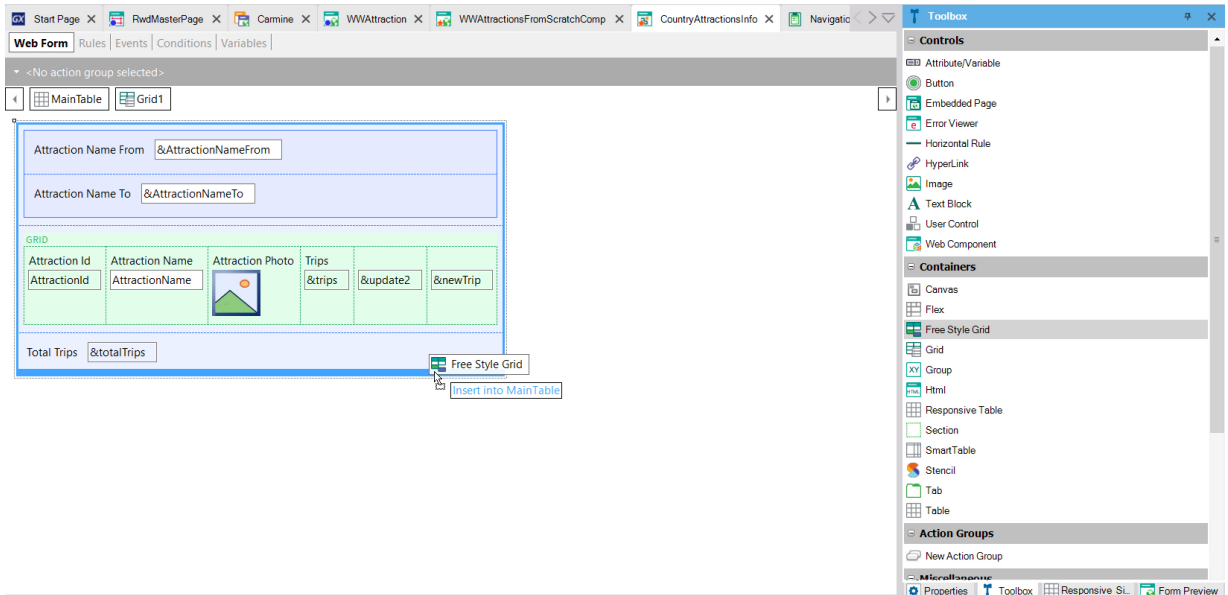
On the right, the web application preview is shown in 'iPad' mode. The page title is 'Travel Agency'. Below the header, there are search filters for 'Country Id' (set to '(None)'), 'Attraction Name From', and 'Attraction Name To'. The main content is a table with the following data:

Attraction Name	Attraction Photo	Trips	UPDATE	NEW TRIP
Christ the Redemmer		3	UPDATE	NEW TRIP
Cinque Terre		0	UPDATE	NEW TRIP
Eiffel Tower		2	UPDATE	NEW TRIP
Forbidden City		1	UPDATE	NEW TRIP
Glenfinnan Viaduct		0	UPDATE	NEW TRIP

A 'Loading...' message is visible at the bottom right of the preview area.

Cambiamos ahora el tamaño de página para que sea de 5 filas, y modificamos la propiedad Paging para que asuma el valor Infinite Scrolling y probemos. Cambiamos la visualización para una Tablet, para ver mejor. Vemos 5 filas cargadas en el grid, pero no tenemos los botones para pasar a la siguiente página. Sin embargo, al hacer scroll, vemos el mensaje de Loading. Se están yendo a buscar las siguientes 5 filas, y se cargan en pantalla y así con las últimas.

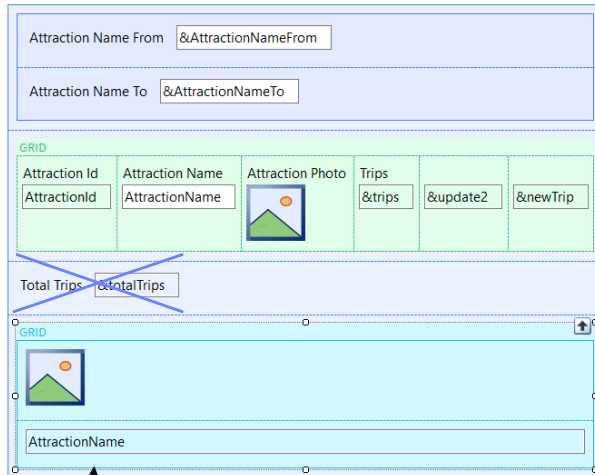
Free style Grid



Por otro lado, la información está mostrándose de manera estructurada, como si fuera una tabla. ¿Y si quisiéramos ver las atracciones turísticas de una manera más flexible?

Por ejemplo si sólo quisiéramos ver foto y nombre, pero una debajo de la otra? Para ello contamos con otro tipo de grid, el de estilo libre. Arrastrémoslo al final del form. Lo inicializamos con los atributos AttractionName y AttractionPhoto. Los movemos para que quede la foto arriba. Y les quitamos la etiqueta.

Free style Grid



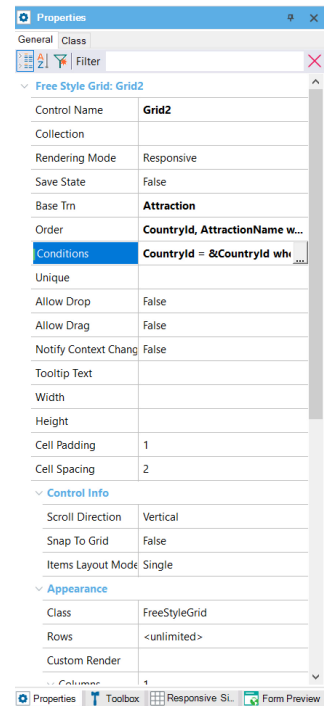
+ AttractionId



```





Event Start
...
AttractionId.Visible = False
...
endevent


```



Observemos que el grid tiene las mismas propiedades que el estándar para especificar transacción base, Order, Conditions. Así que las copiamos del otro. Quitemos esta variable ahora, que no nos interesa. Omitimos agregar AttractionId, atributo que necesitaremos, como en el otro grid, que esté cargado, aunque lo dejemos invisible. Esto era para poder pasarlo por parámetro luego.

Free style Grid

Cinque Terre		0	UPDATE	NEW TRIP
Eiffel Tower		2	UPDATE	NEW TRIP
Forbidden City		1	UPDATE	NEW TRIP
Glenfinnan Viaduct		0	UPDATE	NEW TRIP

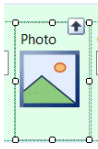


Christ the Redemmer
Cinque Terre
Eiffel Tower
Forbidden City

Bien, ahora probemos. Refrescamos...

Y vemos debajo de nuestro grid estándar, el nuevo, sin ningún diseño, claro. Supongamos que queremos que la imagen aparezca mucho más grande. ¿Por qué está mostrándose tan pequeña?

Bigger image

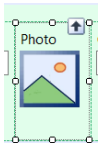


Properties	
General Class	
Attribute/Variable: AttractionPhoto	
Attribute	AttractionPhoto
Title	Photo
Class	ImageAttribute
Column Class	WWColumn WWOptionalColu...
Return On Click	False

The screenshot shows the GeneXus IDE interface. The main window displays a class hierarchy for 'AttractionPhoto'. The 'ImageAttribute' class is highlighted, and its subclasses are listed: 'ActionAttribute', 'ProfileImageAttribute', 'ReadonlyImageAttribute', and 'ResponsiveImageAttrib...'. The 'ReadonlyImageAttribute' class is selected, and its properties are shown in the right-hand pane. The 'Width' property is set to '50px', and the 'Min-Width' property is set to '40px'. The 'Output' window at the bottom shows the build process: 'Success: Build With This Only'.

Si observamos la clase del control atributo en el grid estándar y en el freestyle vemos que se le asignó por defecto la misma clase, ImageAttribute, porque se trata de un atributo de tipo de datos Image. Si vamos al theme a buscarla, veremos que tiene una serie de subclases, entre ellas la de nombre ReadonlyImageAttribute. Esta será la clase que se le aplicará en ejecución a nuestra foto, porque el control atributo en este caso es readonly, así que por más que en las propiedades veamos a la clase padre, en verdad aplicará esta hija. Si miramos sus propiedades, vemos que está especificando un ancho de 50píxeles para la imagen. Y es por ello que la estamos viendo tan pequeña. Si modificáramos aquí esta propiedad para que el ancho sea, por ejemplo de 400 px conseguiremos lo que queríamos para nuestro grid, pero haremos, como efecto secundario, que todo otro control que tuviera esta clase como la suya, también se modificara. En particular la imagen en el Work With.

Bigger image



Properties

General Class

Filter

Attribute/Variable: **AttractionPhoto**

Attribute	AttractionPhoto
Return On Click	False
On Click Event	
Appearance	
Label Position	None
Class	ImageAttribute2
Invite Message	

Start Page x RwdMasterPage x Camine x WWAAttraction x WWAAttractionsFromScratchComp x Properties

Styles Colors Images

Filter

Order Rules

- AudioAttribute
- BlobContentAttribute
- BlobInputAttribute
- CheckBox
- CheckboxLabel
- ComboAttribute
- DescriptionAttribute
- DownloadAttribute
- ErrorAttribute
- FilterAttribute
- ImageAttribute
 - ActionAttribute
 - ProfileImageAttribute
 - ReadOnlyImageAttribute
 - ResponsiveImageAttrib...
- ImageAttribute2
 - BlobContentImageAttrib...
 - BlobInputImageAttribute2
 - ReadOnlyImageAttribu...**
- IME_Active
- IME_Disabled
- IME_Inactive

Output

Show: Build

Uploading 27 Kbytes

Deploying website

Success: Build With This Only

Properties

Filter

Top Right Radius

Bottom Left Radi

Bottom Right Rac

Font

FontStyle

FontVariant

FontWeight

FontSize

TextDecoration

FontFamily

Arial

Forecolor

IME Mode

Height

Max- Height

100%

Min- Height

Width

auto

Max- Width

400px

Min- Width

Mouse and Keyboard

Cursor

Text

Letter Spacing

Text Align

left

Text Indent

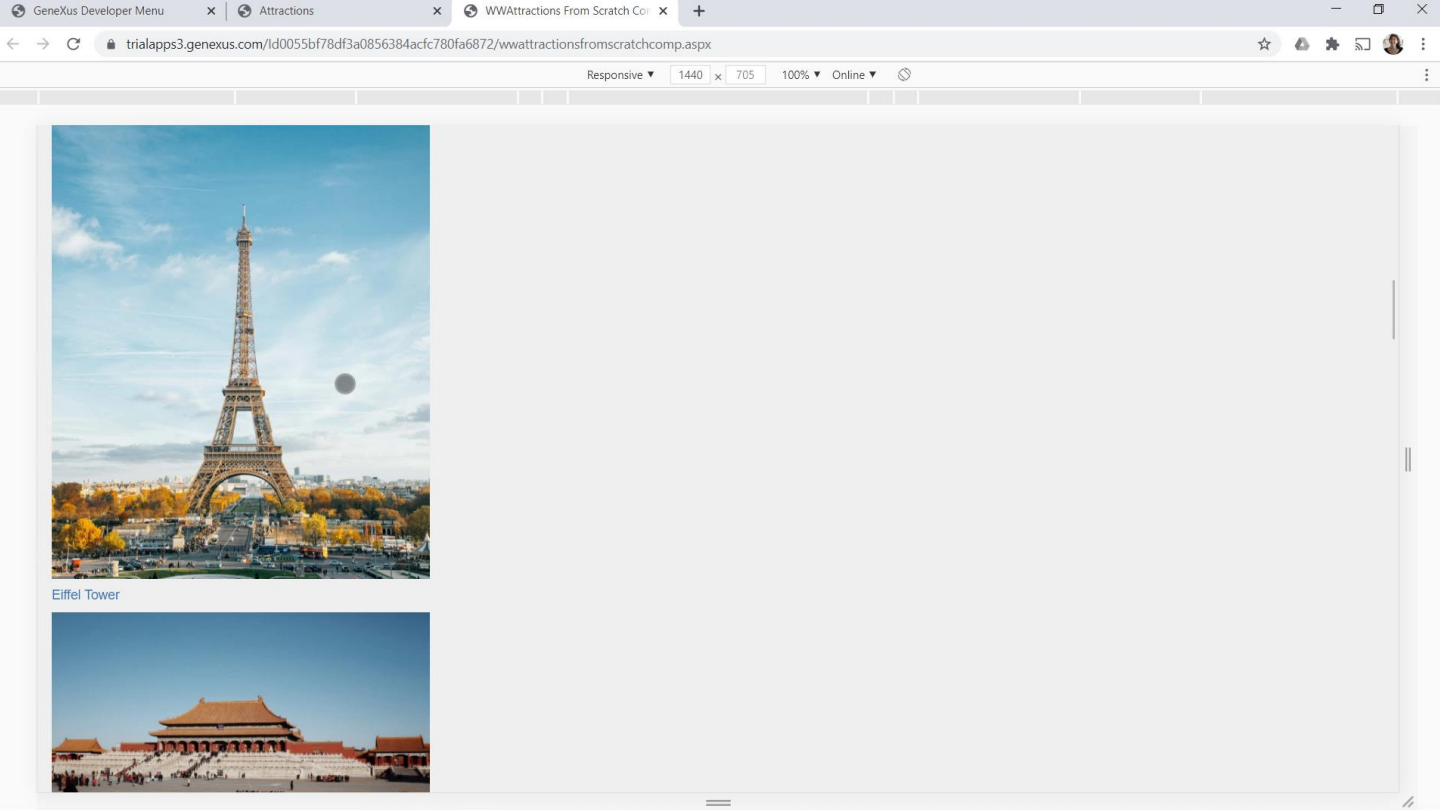
Text Transform

Vertical Align

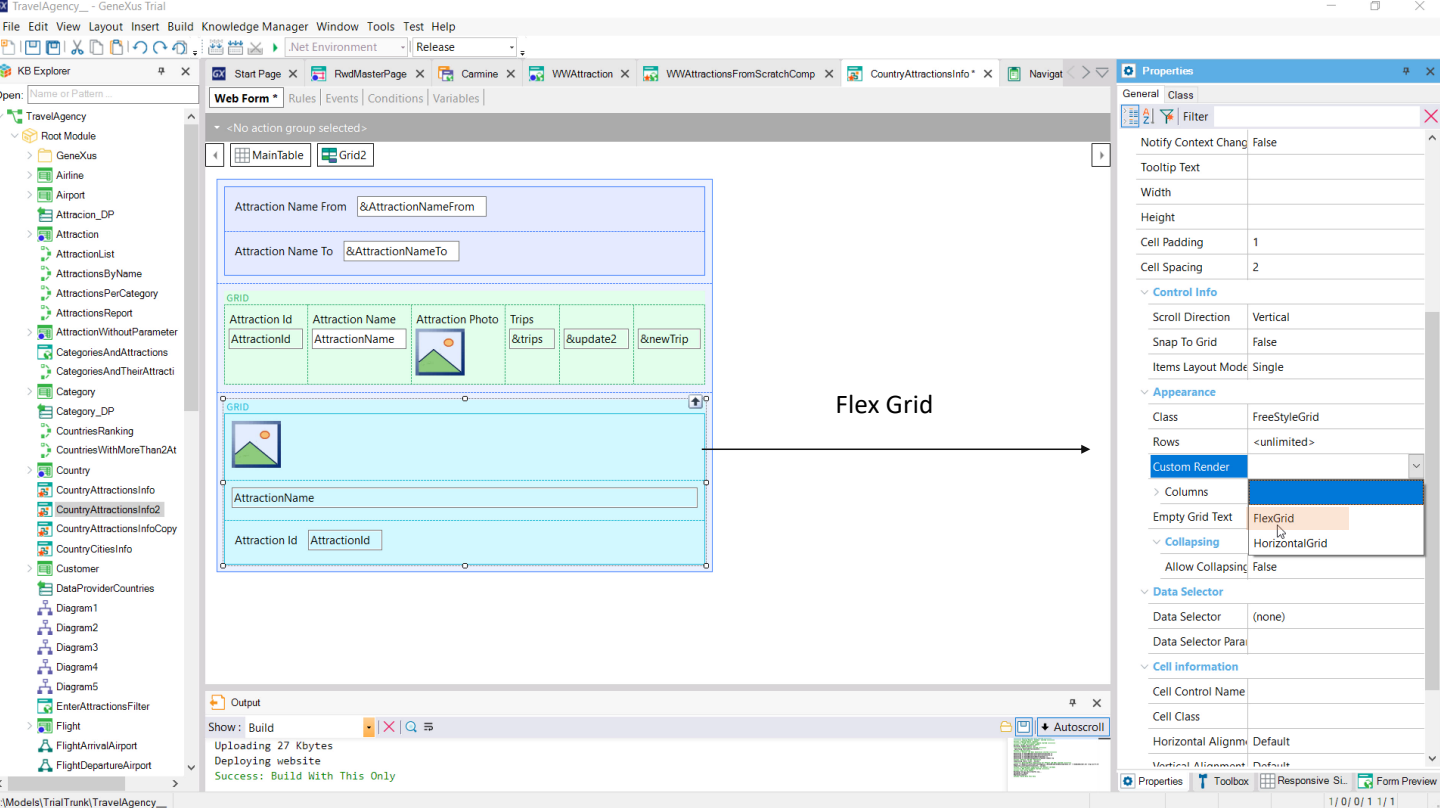
Properties | Toolbox | Responsive SL | Form Preview

No deseamos esto, por lo que hemos creado otra clase hermana de esta (ya la mostramos creada), a la que llamamos ImageAttribute2, que se creó automáticamente con todas estas subclases, y lo que hicimos fue modificar estas propiedades para la Readonly. Estamos estableciendo que la imagen se presente con su tamaño original mientras no supere los 400 px de ancho. Y en ese caso que se achique hasta ese tamaño.

Entonces ahora vamos a modificarle la clase a nuestro control AttractionPhoto para este segundo grid, por esta otra.



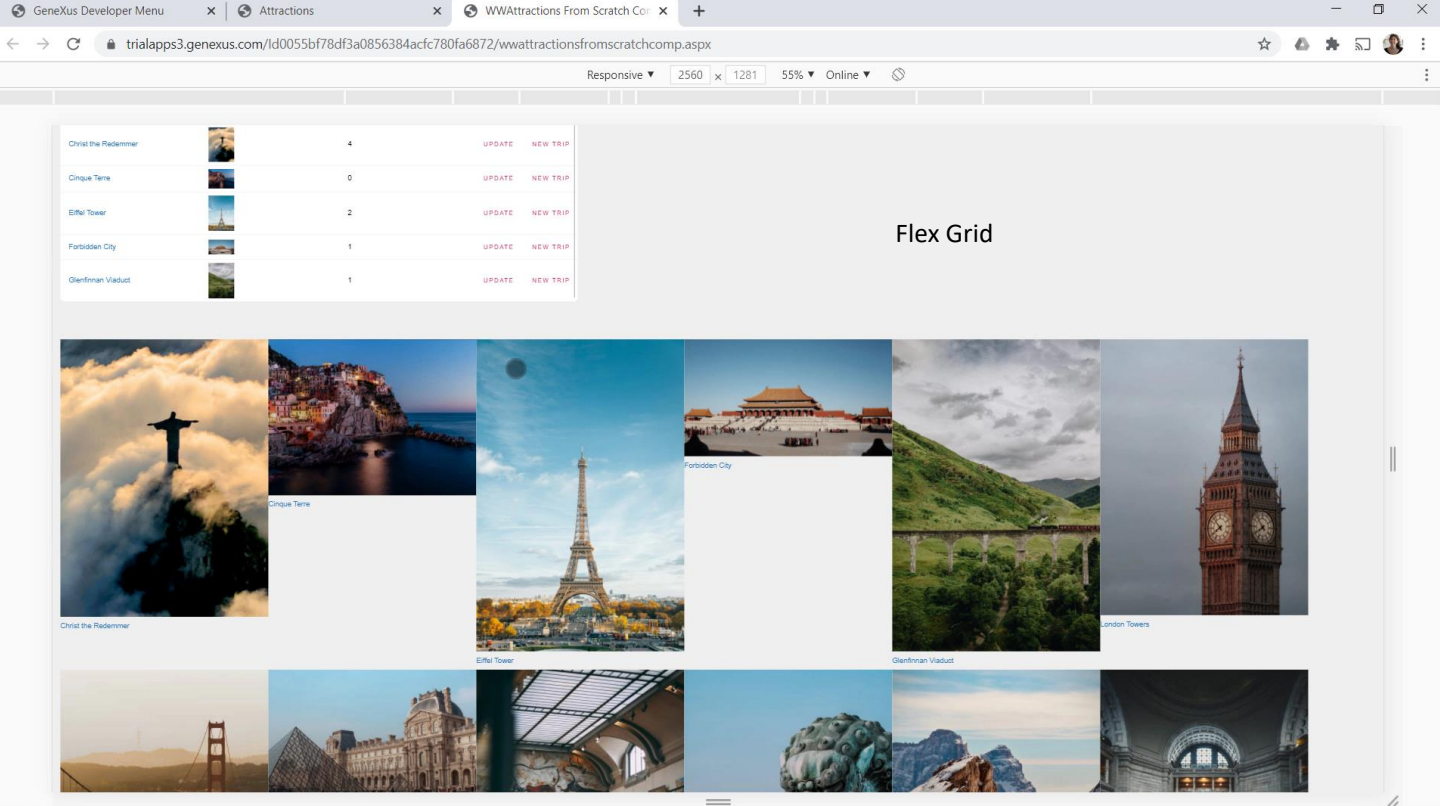
Probemos en ejecución. Se ve claramente la diferencia.



¿Y si ahora queremos presentar las atracciones de un modo que aproveche más el espacio de pantalla?

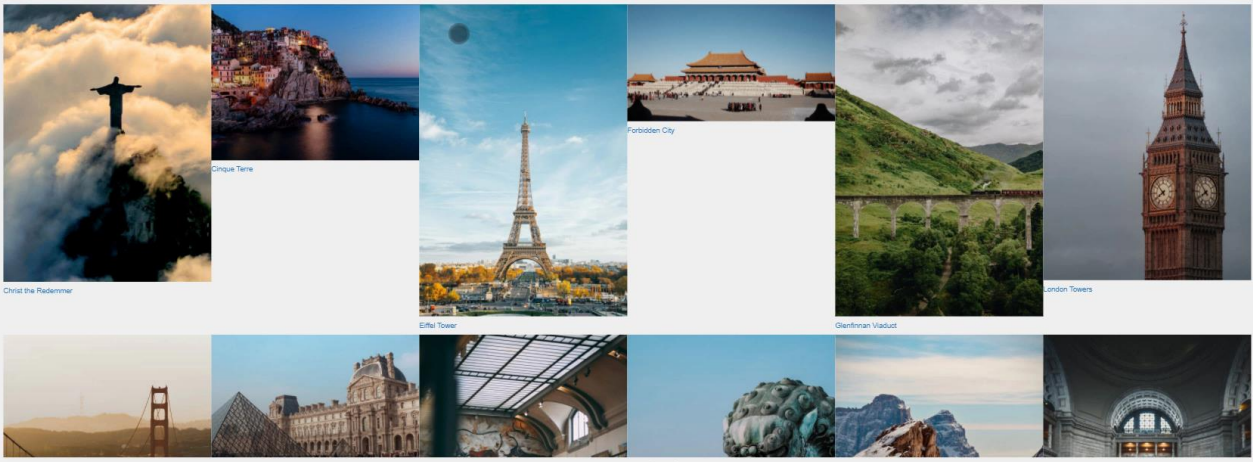
Vemos que las fotos no tienen un tamaño homogéneo.

Vayamos al grid, y modifiquemos su rendering para que sea un grid flexible, esto es, para que su contenido ocupe como un puzle los espacios libres de pantalla. Probemos.



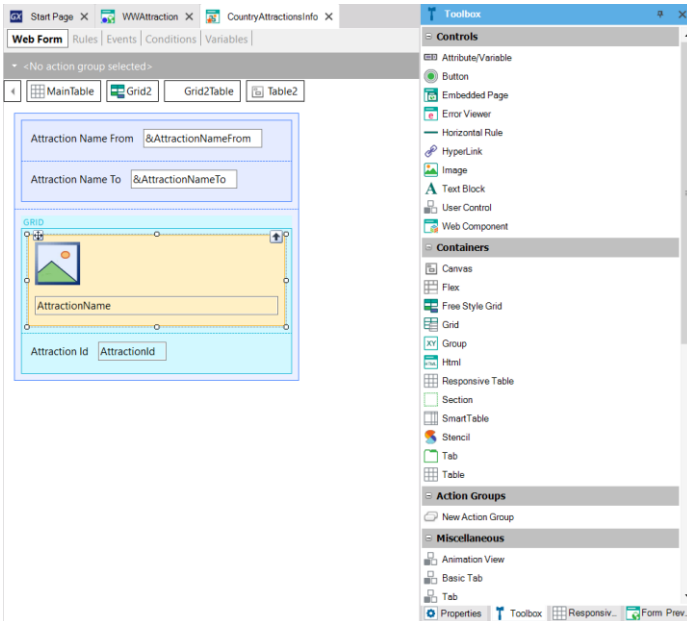
Christ the Redeemer		4	UPDATE	NEW TRIP
Cinque Terre		0	UPDATE	NEW TRIP
Eiffel Tower		2	UPDATE	NEW TRIP
Forbidden City		1	UPDATE	NEW TRIP
Glenfinnan Viaduct		1	UPDATE	NEW TRIP

Flex Grid

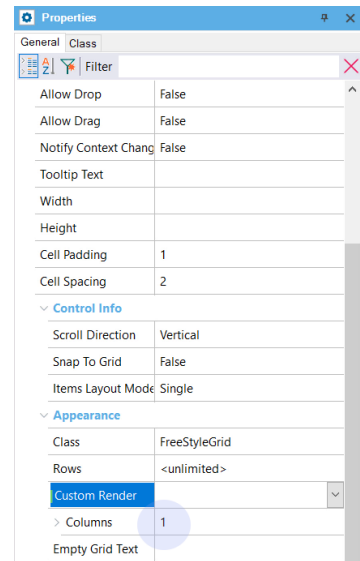


Vemos que dependiendo del tamaño de pantalla, y considerando ese ancho máximo de 400 píxeles, las atracciones turísticas se van acomodando en la dirección Row para llenar el espacio.

Canvas to overlapping



Default Free Style Grid

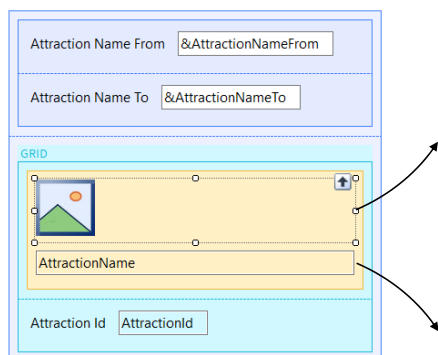


¿Y si quisiéramos que el nombre de la atracción aparezca solapado sobre la imagen?

Empecemos por eliminar el grid estándar que teníamos. No lo mostramos, pero acabamos de eliminar también los eventos asociados.

Ahora insertamos un control Canvas en el form, que es un tipo de tabla que permite el solapamiento de los controles que contenga. Movemos para allí dentro los dos controles que deseamos se solapen. Volvamos a dejar el grid freestyle con el rendering default, por el que se listará una atracción (con su foto y nombre) por fila.

Canvas to overlapping



Absolute position	
Top	0px
Left	0px
Bottom	0px
Right	0px
Width	100%
Height	100%
Z- Order	1

Absolute position	
Top	0px
Left	15px
Bottom	0px
Right	15px
Width	100%
Height	100%
Z- Order	2

Canvas: Table2	
Control Name	Table2
Tooltip Text	
Header	
Appearance	
Width	100%
Height	300px
Class	Table
Cell information	
Cell Control Name	
Cell Class	
Horizontal Alignment	Default
Vertical Alignment	Default
Row information	
Row Height	
Row Class	

Vemos que al haber colocado este control dentro del Canvas, aparecieron estas propiedades que son las que permiten ubicar al control en relación a la tabla, decir cuánto va a ocupar, y en qué capa va a estar. Dejamos todos los valores por defecto, salvo el de la capa. Ponemos 1, porque queremos que la imagen esté en la capa más profunda, por debajo de la siguiente, la del nombre de atracción lo pondremos en la capa 2, por encima de la 1. Además queremos que este control tenga un margen izquierdo de 15 píxeles, para que tenga un margen en relación a la foto.

Veamos que entre las propiedades de la tabla Canvas, tenemos las que nos permiten especificar cuánto de alto y de ancho ocupará en relación al control que la contiene. Aquí estamos dejando que ocupe el 100% del ancho, pero vamos a indicarle que el alto sea de, por ejemplo, 300 píxeles.

The screenshot shows a web application development environment. On the left, there is a vertical list of attraction cards: 'Cinque Terre', 'Eiffel Tower', and 'Forbidden City'. The 'Eiffel Tower' card is currently selected. In the center, a 'Grid' container holds a preview of the selected card, showing a yellow background with a small image icon, a text input field labeled 'AttractionName', and a button labeled 'AttractionId'. The 'Properties' panel on the right is open, showing the 'AttributeTextOverImage' class for the 'AttractionName' control. The properties include: Border Width (0px), Border Radius (all zero), Font (20px Calibri), Font Style (normal), Font Weight (400), Text Decoration (none), and Forecolor (#FFF).

Y por último, cambiemos el diseño del control `AttractionName`, para que salga más lindo.

Vemos que por defecto tiene asociada la clase `Attribute`, pero se la vamos a cambiar por una que hemos creado en el theme, con este nombre. Si observamos sus propiedades, vemos que por ejemplo especificamos para el color de fuente un blanco, el tipo de letra Calibrí, un tamaño de 20 píxeles, etc.

Probemos todo esto en ejecución. Si ahora hacemos clic sobre el nombre de la torre Eiffel... vemos su información.

Horizontal grid

The screenshot shows the GeneXus IDE interface. On the left, a web form is being designed. It contains two text input fields: "Attraction Name From" with value "&AttractionNameFrom" and "Attraction Name To" with value "&AttractionNameTo". Below these is a grid control labeled "GRID" with a yellow background. The grid contains a small image of a landscape and a text input field labeled "AttractionName". Below the grid is another text input field labeled "Attraction Id" with value "AttractionId".

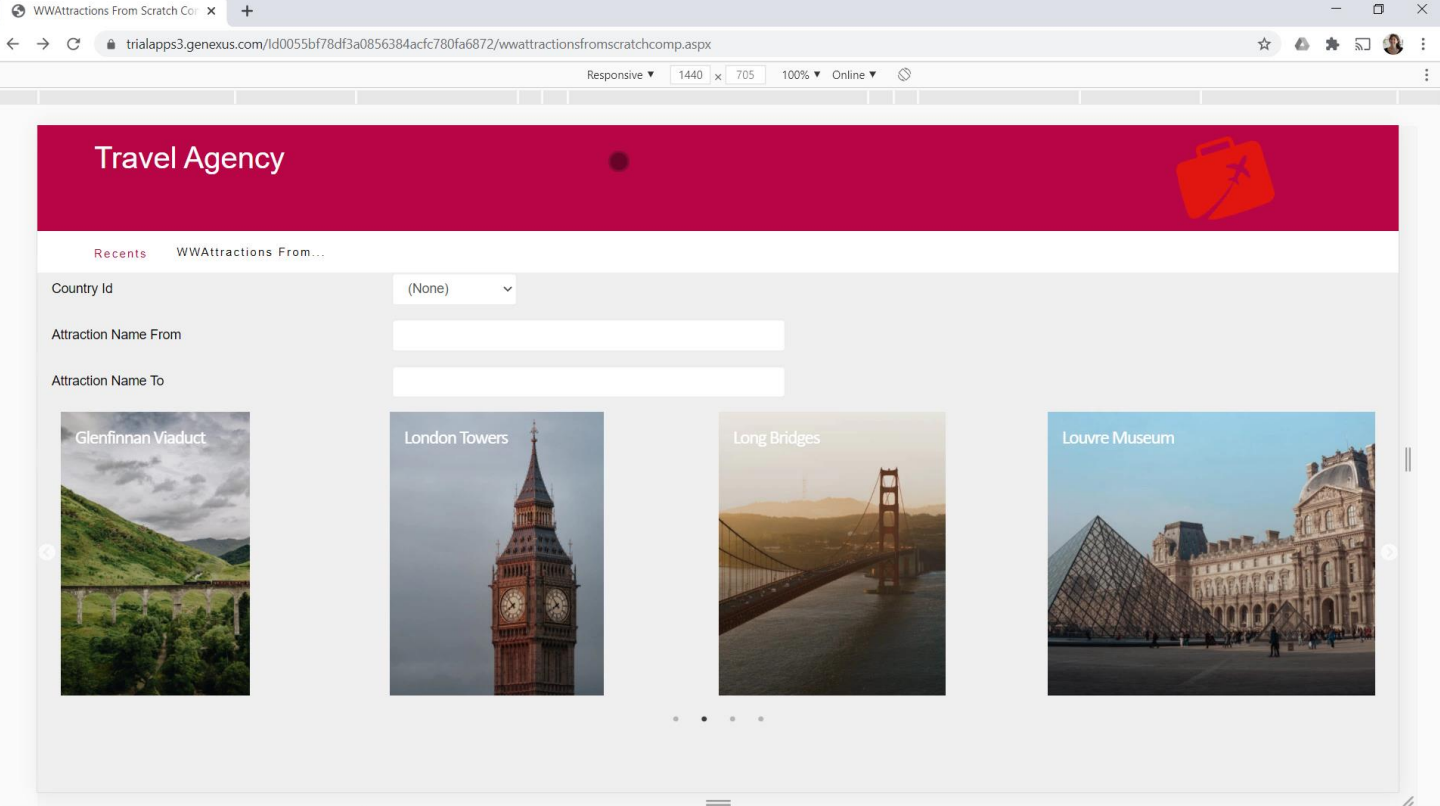
On the right, the Properties window is open, showing the configuration for the selected "GRID" control. The "General" tab is active, and the "Class" is set to "FreeStyleGrid". The "Control Info" section shows:

Cell Spacing	2
Scroll Direction	Vertical
Snap To Grid	False
Items Layout Mode	Single

The "Appearance" section shows:

Class	FreeStyleGrid
Rows	<unlimited>
Custom Render	HorizontalGrid
Columns	1 2 3 4
Extra Small	1
Small	2
Medium	3
Large	4
Empty Grid Text	

Y para terminar de apreciar algunas de las posibilidades que nos brindan los controles grid, si ahora modificamos su rendering para que se presente como un grid horizontal... donde, por ejemplo, para tamaño tipo teléfono queremos que muestre solo una columna, para tamaño small, 2, para médium 3 y para large 4...



Presionamos F12... Aquí vemos tamaño Large, con 4 atracciones por página horizontal...

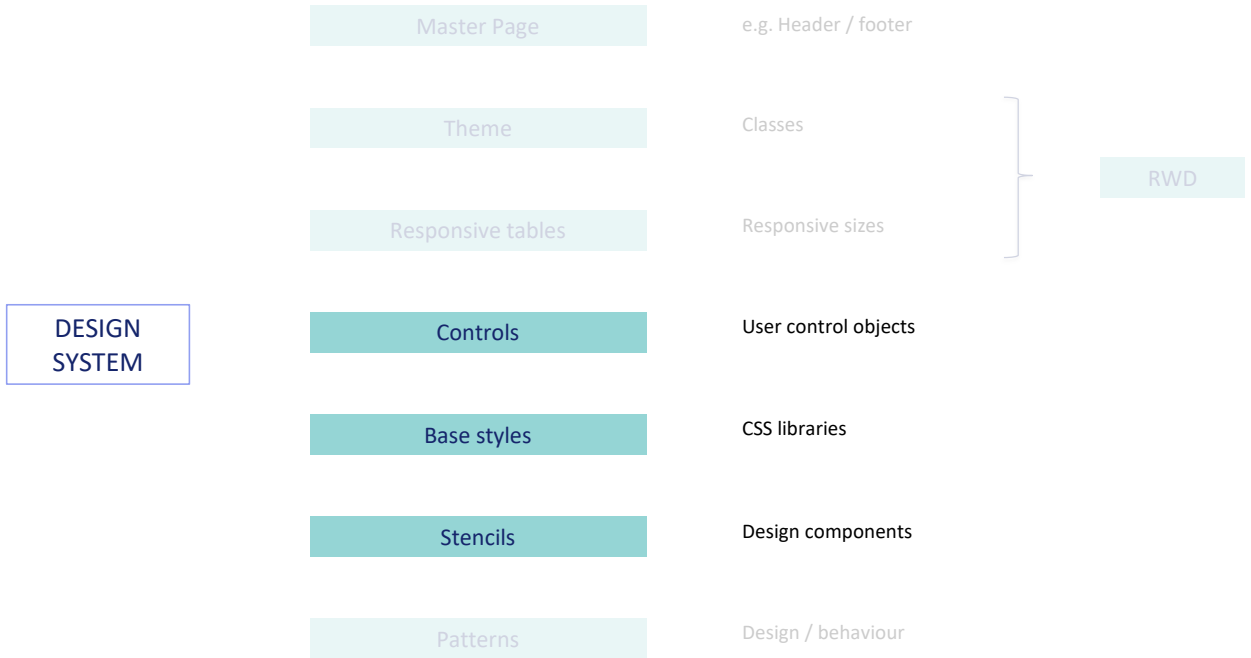
Aquí vemos 3...

Para una Tablet 2...

Y para un teléfono una.

Por supuesto, aquí nos convendrá insertar imágenes que tengan el mismo tamaño.

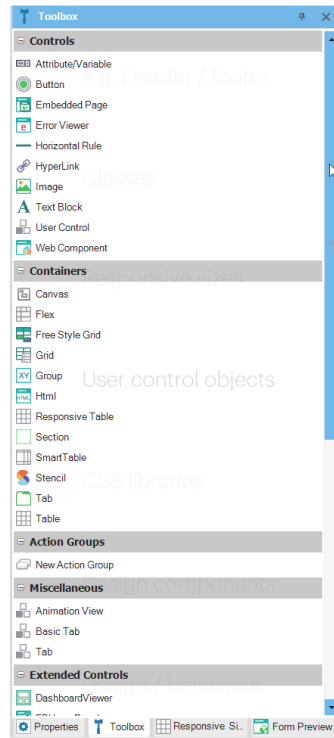
Con esto vimos tan solo una introducción a todo lo que podemos hacer con los controles y las clases del theme.



Hay algunos jugadores más que, junto con todos los vistos, permiten utilizar en GeneXus un Design System potente. No los estudiaremos en este nivel, pero los dejamos presentados.

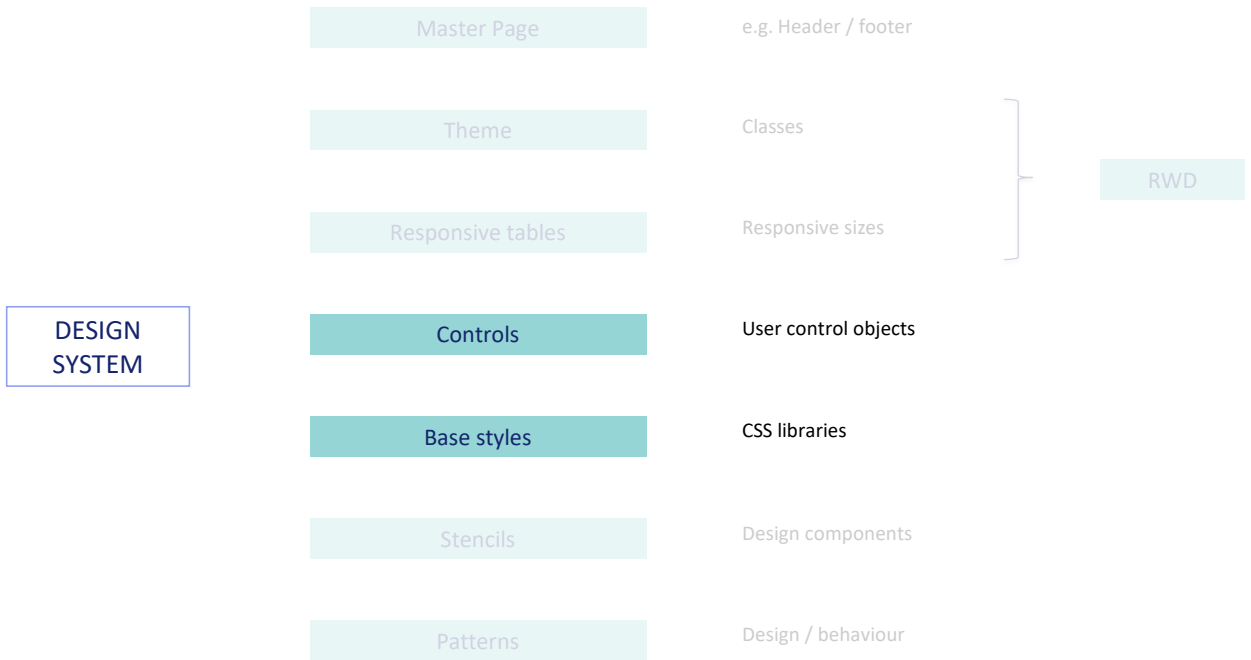
DESIGN SYSTEM

- Master Page
- Theme
- Responsive tables
- Controls**
- Base styles
- Stencils
- Patterns



RWD

Podemos no solo utilizar en nuestros forms los controles que vienen en la toolbox de GeneXus en forma predeterminada...



...sino que además podemos definir controles de usuario, que pueden ser copiados de plataformas que ofrecen esos controles junto con bibliotecas CSS (que son, como nuestros themes, las que especifican su estilo –el estilo de esos controles-).

Card | Semantic UI

semantic-ui.com/views/card.html

UI Docs

Getting Started

New in 2.4

Introduction

- Integrations
- Build Tools
- Recipes
- Glossary

Usage

- Theming
- Layouts

Globals

- Reset
- Site


Elements

- Button
- Container
- Divider
- Flag
- Header
- Icon
- Image
- Input
- Label
- List
- Loader
- Placeholder

Card

A single card.

To ensure cards are equal height use the plural, `cards`. Card groups automatically uses `flex` styles to match height between cards in the same row.



Kristy
Joined in 2013
Kristy is an art director living in New York.
22 Friends

Card

Types

Card

Cards

Content

Variations

Por ejemplo, esta, Semantic UI. Podríamos querer utilizar un control Card como este, por ejemplo.

Para ello alcanzará con crear un objeto User control (le podemos llamar igual)...

Card | Semantic UI


semantic-ui.com/views/card.html

Card

A single card.

To ensure cards are equal height use the plural, `cards`. Card groups automatically uses flex styles to match height between cards in the same row.

Example



```
1 <div class="ui card">
2   <div class="image">
3     
5   <div class="content">
6     <a class="header">{{Name}}</a>
7     <div class="meta">
8       <span class="date">{{Date}}</span>
9     </div>
10    <div class="description">
11      {{Description}}
12    </div>
13  </div>
14  <div class="extra content">
15    <a>
16      <i class="user icon"></i>
17      {{ExtraContent}}
18    </a>
19  </div>
20 </div>
```

`<div class="ui card">`

UI Docs

- Getting Started
- New in 2.4
- Introduction
 - Integrations
 - Build Tools
 - Recipes
 - Glossary
- Usage
 - Theming
 - Layouts
- Globals
 - Reset
 - Site
- Elements
 - Button
 - Container
 - Divider
 - Flag
 - Header
 - Icon
 - Image
 - Input
 - Label
 - List
 - Loader
 - Placeholder

Card

Types

- Card
- Cards

Content

...copiar y pegar su html, cambiar los datos fijos por algo así como nombres de elementos de un SDT (para poder hacerlo dinámico, es decir, poder cargar dinámicamente ese control, con datos que vamos a poder especificar por ejemplo de la base de datos...

Card | Semantic UI

semantic-ui.com/views/card.html

UI Docs

Getting Started

New in 2.4

Introduction

Integrations

Build Tools

Recipes

Glossary

Usage

Theming

Layouts

Globals

Reset

Site

Elements

Button

Container

Divider

Flag

Header

Icon

Image

Input

Label

List

Loader

Placeholder

Kristy
Joined in 2013
Kristy is an art director living in New York.
22 Friends

```
<div class="ui card">
  <div class="image">
    
  </div>
  <div class="content">
    <a class="header">Kristy</a>
    <div class="meta">
      <span class="date">Joined in 2013</span>
    </div>
    <div class="description">
      Kristy is an art director living in New York.
    </div>
    <div class="extra content">
      <a>
        <i class="user icon"></i>
        22 Friends
      </a>
    </div>
  </div>
</div>
```

Screen Template Properties

```
1 <div class="ui card">
2   <div class="image">
3     
4   </div>
5   <div class="content">
6     <a class="header">{{Name}}</a>
7     <div class="meta">
8       <span class="date">{{Date}}</span>
9     </div>
10    <div class="description">
11      {{Description}}
12    </div>
13  </div>
14  <div class="extra content">
15    <a>
16      <i class="user icon"></i>
17      {{ExtraContent}}
18    </a>
19  </div>
20 </div>
```

User Control: Card

Name	Card
Description	Card
Module/Folder	Root Module
Is Control Type	False
References	
Base Control Type	None
Base Style	SemanticUI
Qualified Name	Card
Object Visibility	Public

...especificar de dónde tomará el CSS, es decir, el diseño de las clases, hacer alguna cosita más...

Card | Semantic UI

semantic-ui.com/views/card.html

UI Docs

Getting Started

New in 2.4

Introduction

Integrations

Build Tools

Recipes

Glossary

Usage

Theming

Layouts

Globals

Reset

Site

Elements

Button

Container

Divider

Flag

Header

Icon

Image

Input

Label

List

Loader

Placeholder

Kristy
Joined in 2013
Kristy is an art director living in New York.
22 Friends

```
<div class="ui card">
  <div class="image">
    
  </div>
  <div class="content">
    <a class="header">Kristy</a>
    <div class="meta">
      <span class="date">Joined in 2013</span>
    </div>
    <div class="description">
      Kristy is an art director living in New York.
    </div>
  </div>
  <div class="extra content">
    <a>
      <i class="user icon"></i>
      22 Friends
    </a>
  </div>
</div>
```

Web Form

Attraction Name From: &AttractionNameFrom

Attraction Name To: &AttractionNameTo

<Card: Card?>

Attraction Name

Attraction Id: &AttractionId

Controls

- Attribute/Variable
- Button
- Embedded Page
- Error Viewer
- Horizontal Rule
- HyperLink
- Image
- Text Block
- User Control
- Web Component

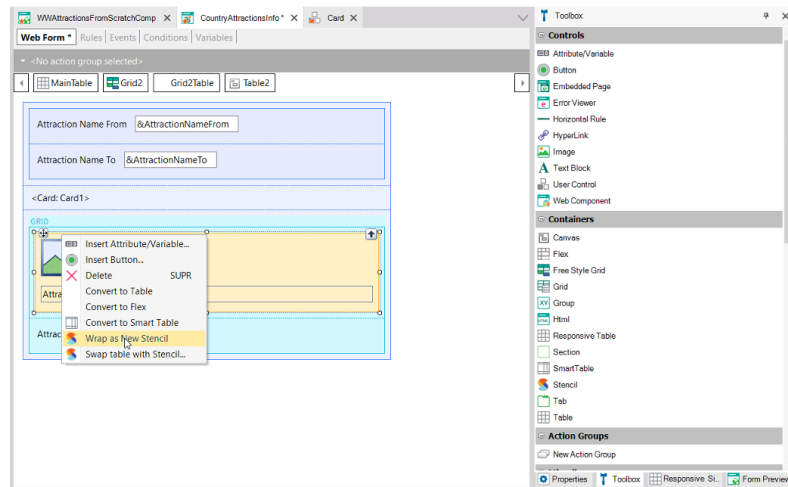
Containers

- Canvas
- Flex
- Free Style Grid
- Grid
- Group
- Html
- Responsive Table
- Section
- Smart Table
- Stencil
- Tab
- Table

Action Groups

- New Action Group

y ya lo tendremos disponible en la Toolbox para utilizarlo.



RWD

DESIGN
SYSTEM

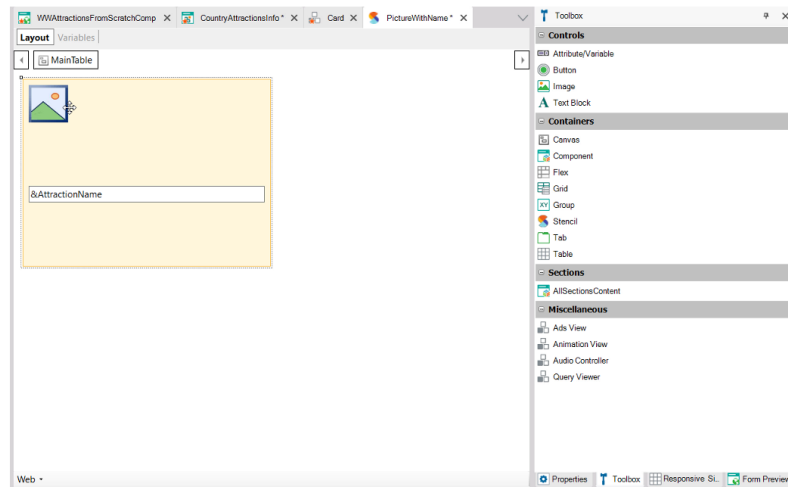
Stencils

Design components

Patterns

Design / behaviour

Por otro lado, GeneXus ofrece Stencils, que sirven para repetir el diseño de una misma porción de pantalla (un conjunto de controles), en muchas pantallas.



RWD

DESIGN
SYSTEM

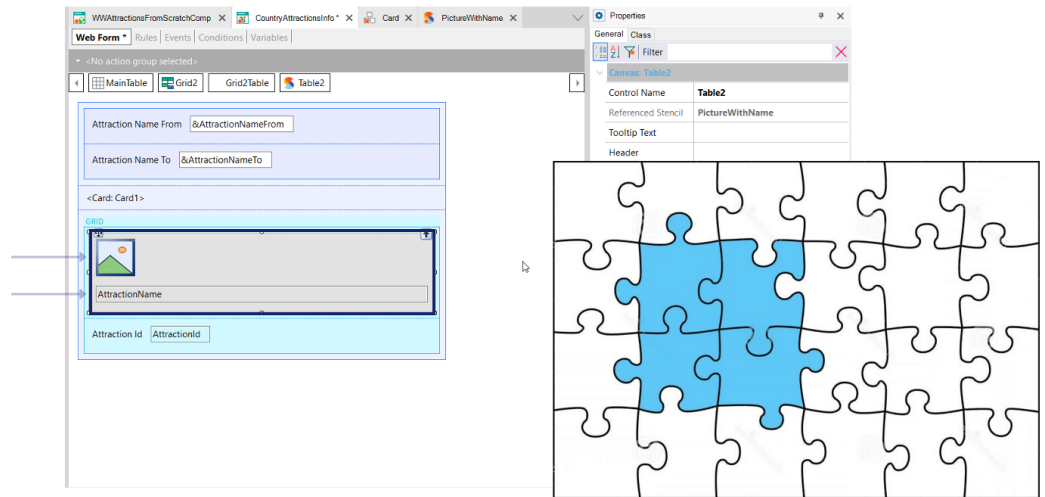
Stencils

Design components

Patterns

Design / behaviour

Es una forma de abstraer el diseño en un nivel mayor.



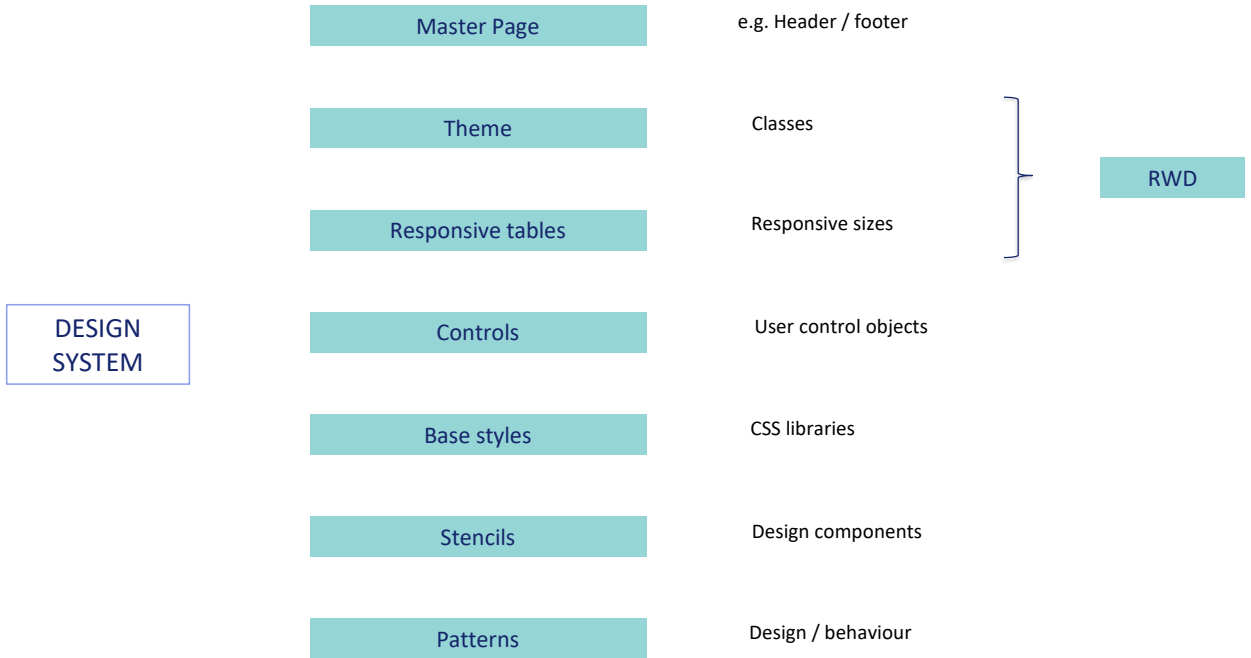
Stencils

Design components

Patterns

Design / behaviour

Si podemos pensar las pantallas de una aplicación web como puzzles compuestos de piezas atómicas (que serían los controles), los stencils son una forma de agrupar un conjunto de controles cuyo diseño se va a repetir en muchas pantallas. Sería una pieza hecha de piccitas menores.



Con esto, terminamos de presentar a los jugadores más importantes que hacen a la implementación en GeneXus del Design System.