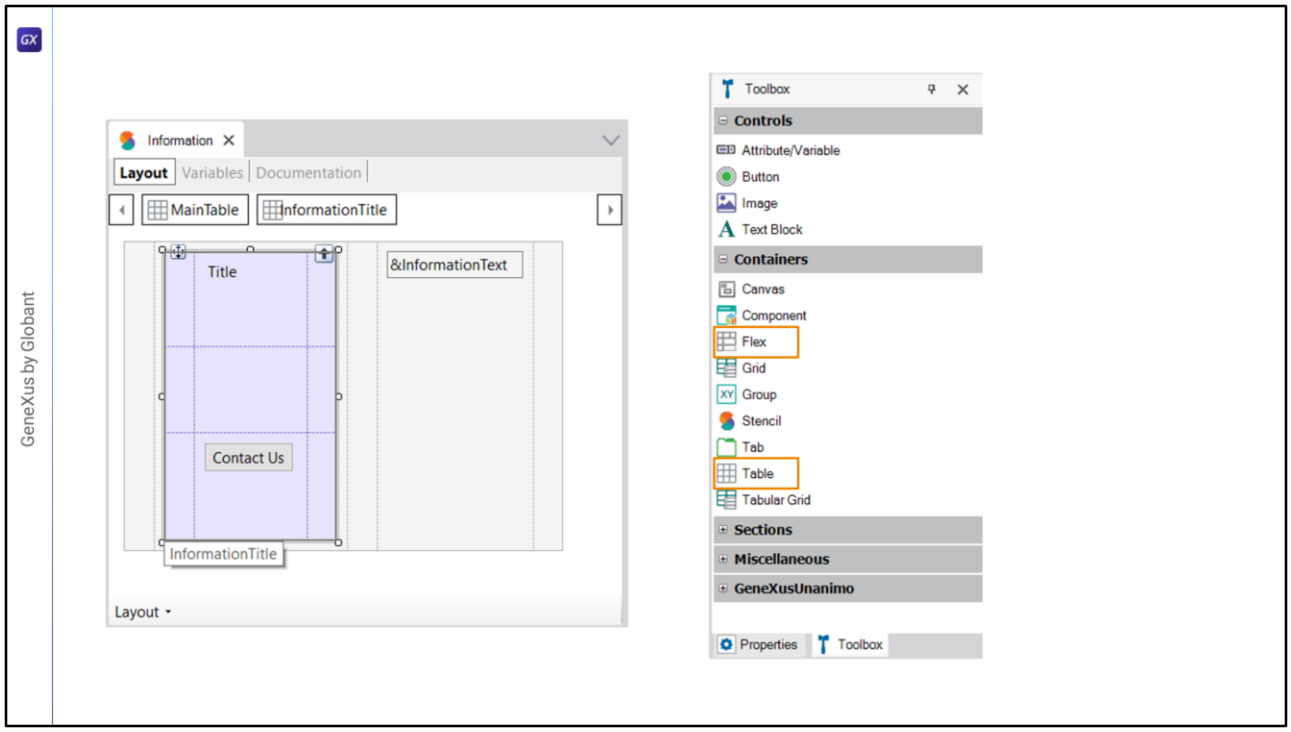
GeneXus by Globant

First Layout in GeneXus

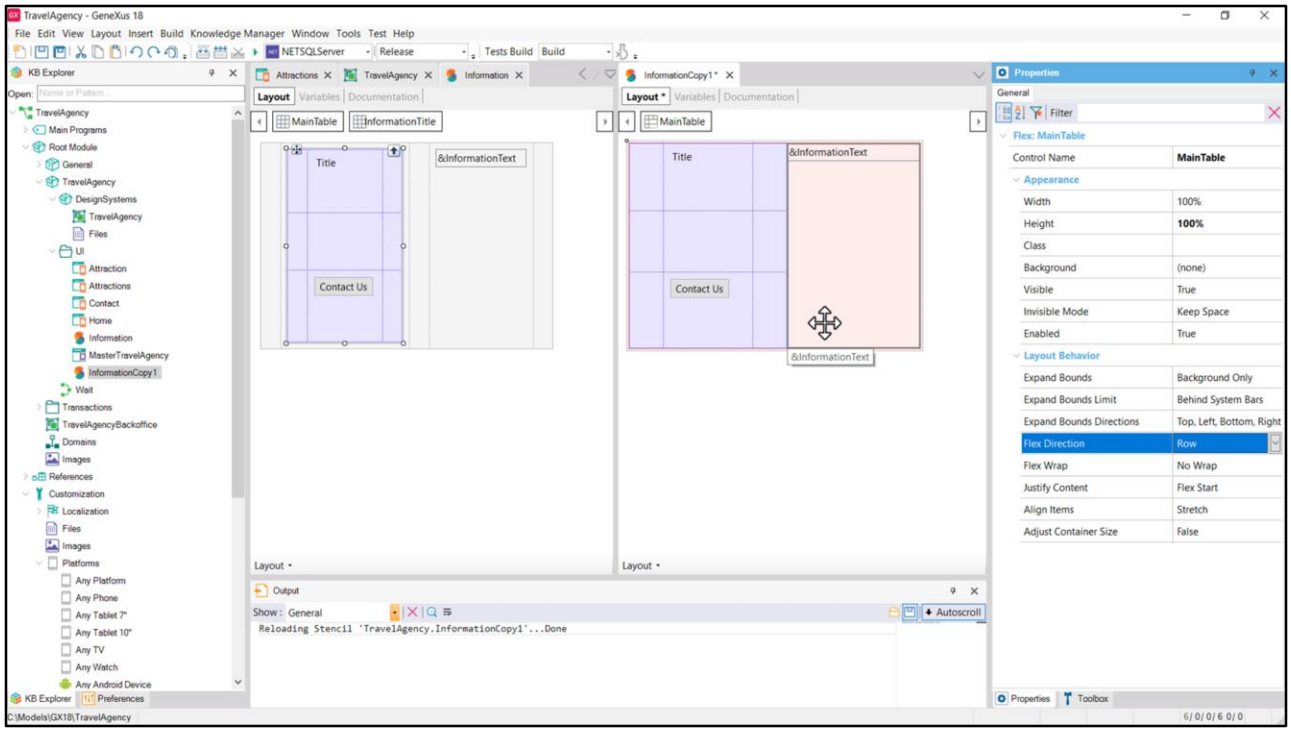
Flex control instead of Table

Cecilia Fernández



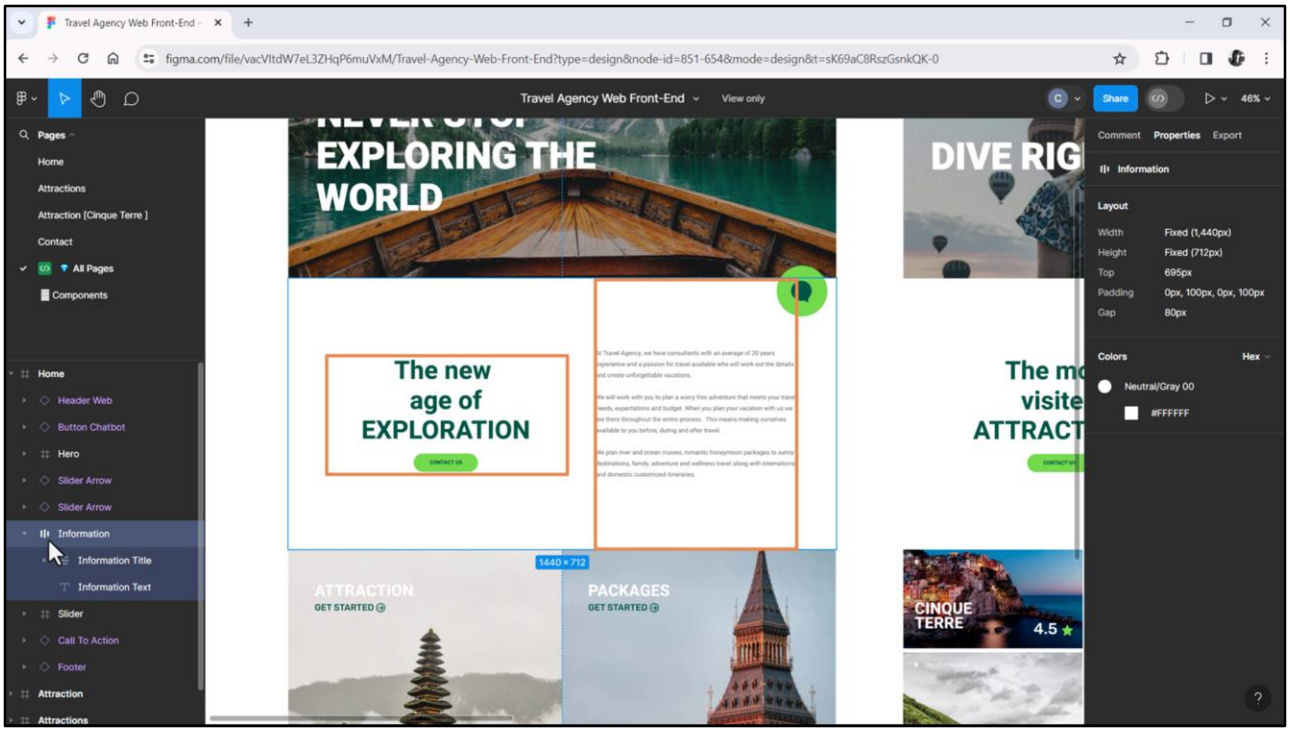
Cuando construimos el Stencil, al principio de este módulo, elegimos modelarlo a partir de la tabla principal y de otra tabla para el textblock y el botón.

Pero allí mismo dijimos que también podríamos haber elegido como contenedor no el control Table sino el control Flex, ¿recuerdan?



Es más (voy a hacer un save as de nuestro stencil con otro nombre)… habíamos dicho que podíamos convertir en cualquier momento una tabla común como esta en Flex y viceversa. Al convertirla vemos que desaparecen las filas y columnas, y aparecen estas propiedades dentro del grupo Layout Behavior. La que en principio capta primero nuestra atención es la Flex Direction, cuyo valor default es Row, y es lo que va a determinar que los dos elementos contenidos por el flex se dispongan horizontalmente a lo largo de una única fila, **sin** columnas.

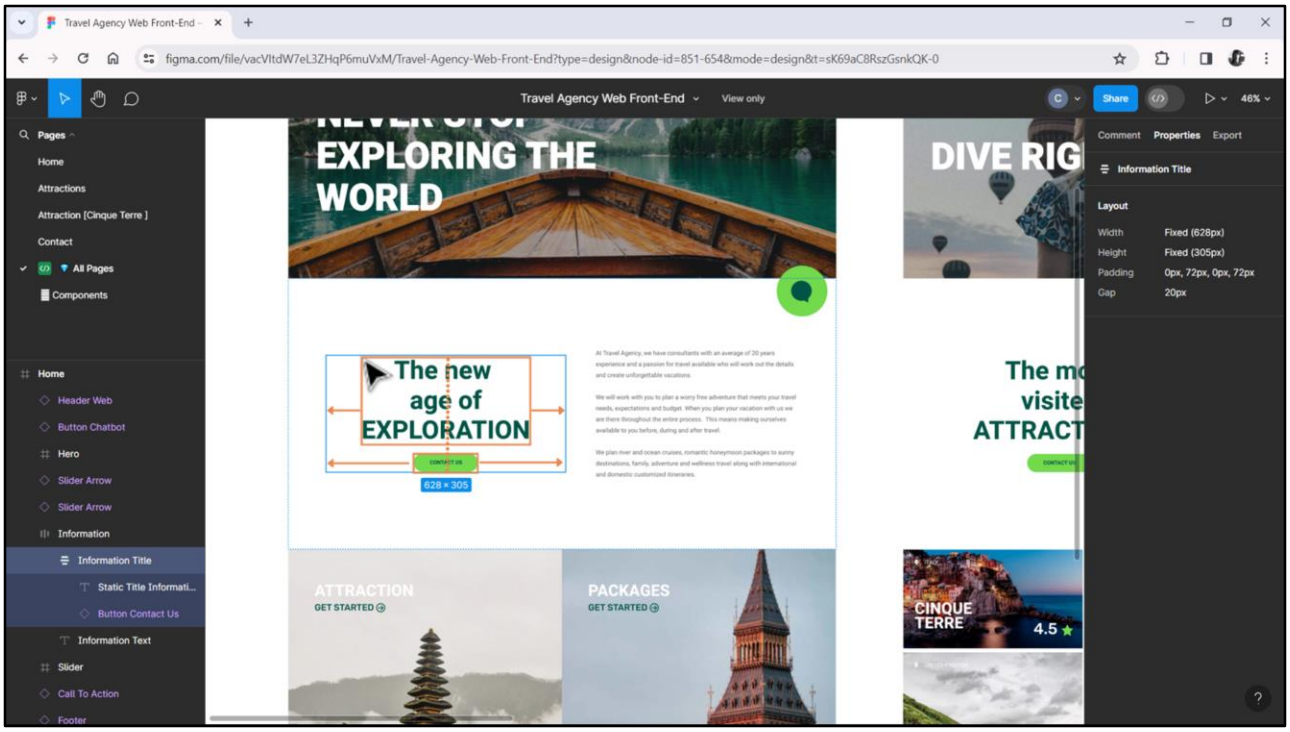
Nos va a quedar un poco más claro en Figma.



Para el frame Information aparece este símbolo y para el frame que contiene al título y al botón este otro símbolo. Estos símbolos nos dicen bastante más de lo que pudiera parecer.

El primero nos está indicando que los elementos del frame (este y este) se colocarán a lo largo de una fila, es decir, horizontalmente: primero uno, y después el otro. Y además, que esos elementos pueden tener distintos altos, pero que se alinearán por el centro.

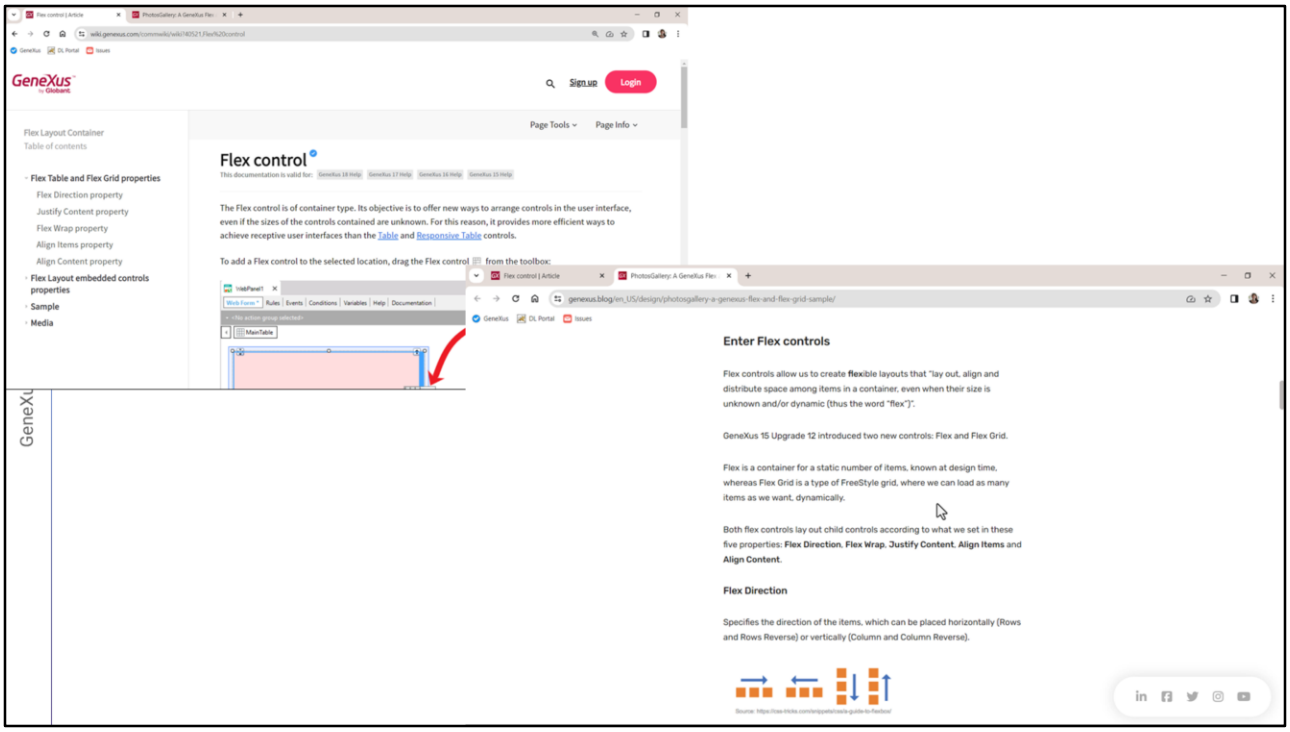
De hecho tienen distintos altos: si vemos este… tiene este alto y si vemos este otro tiene este otro alto, que es el del contenedor. Pero están ambos alineados verticalmente respecto al centro.



Si observamos este otro, vemos que también se trata de un contenedor que contiene a dos elementos, esta vez dispuestos en torno a la dirección columna. Primero uno, el texto, después el otro, el botón. Y que también están alineados por el centro respecto a los bordes de la otra dirección. Así como estos dos, cuya dirección es row, están centrados respecto a la vertical.

Chechu consigue esto diciendo que el contenedor tiene lo que en Figma se conoce como Auto Layout. Para nosotros eso se dice: Flex.

Básicamente significa que los elementos contenidos se van a colocar a lo largo de una dirección: ya sea horizontal, como esta, o vertical, como esta otra.

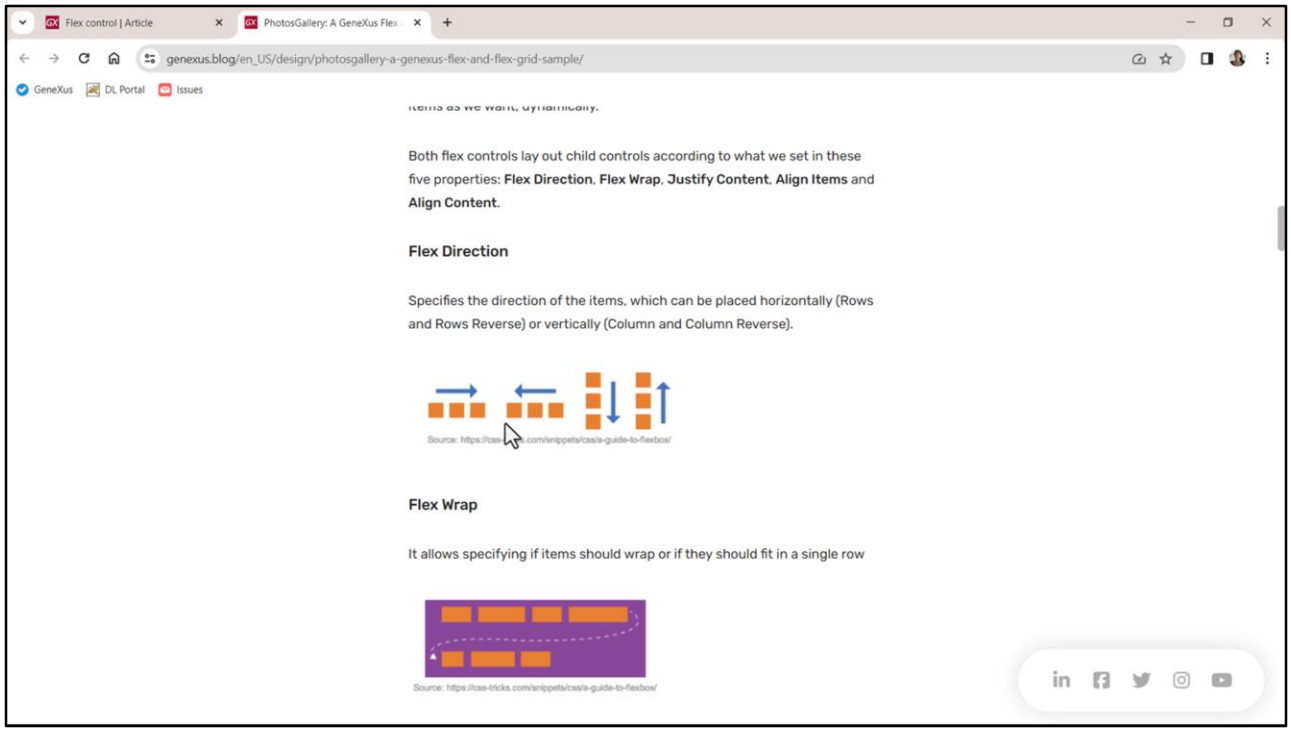


En esta página del wiki de GeneXus se explica el control. Y se ofrece al final un link a esta otra página que lo explica con un ejemplo.

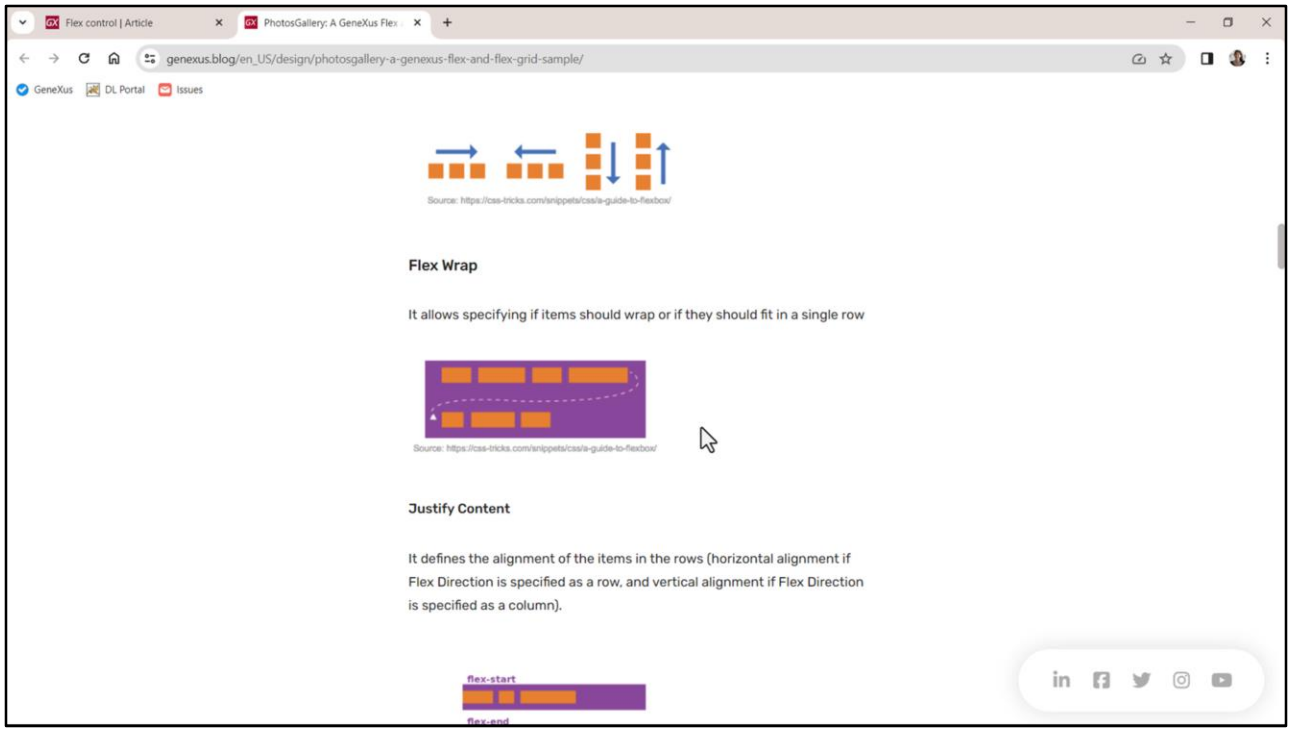
Voy a aprovecharla para que veamos gráficamente la cuestión así nos va a quedar más claro.

Aquí dice que estos contenedores permiten crear layouts **flexibles**, que permiten colocar los ítems, alienarlos y distribuir el espacio entre ellos en un contenedor, por supuesto, incluso (o sobre todo, diría yo) cuando su tamaño es desconocido a priori o es dinámico.

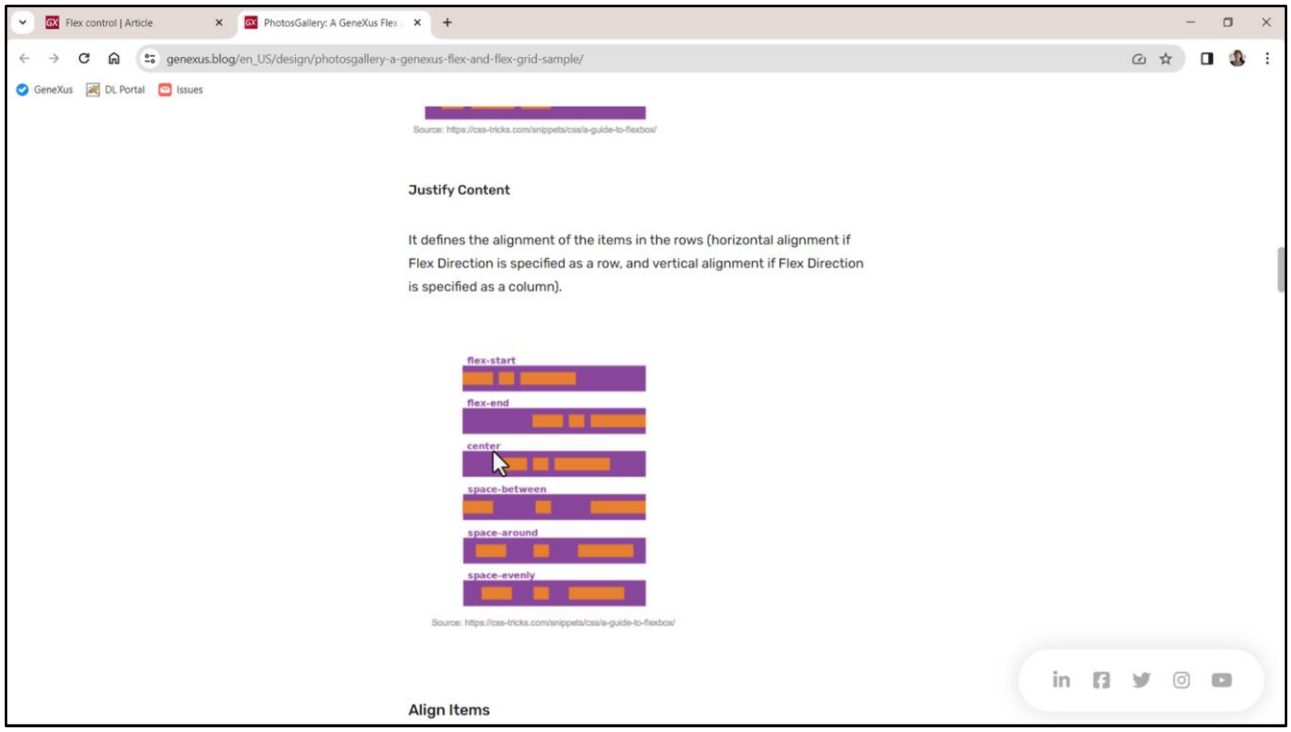
Aparecen entonces una serie de propiedades para configurar dirección y cuestiones de la distribución de los controles.



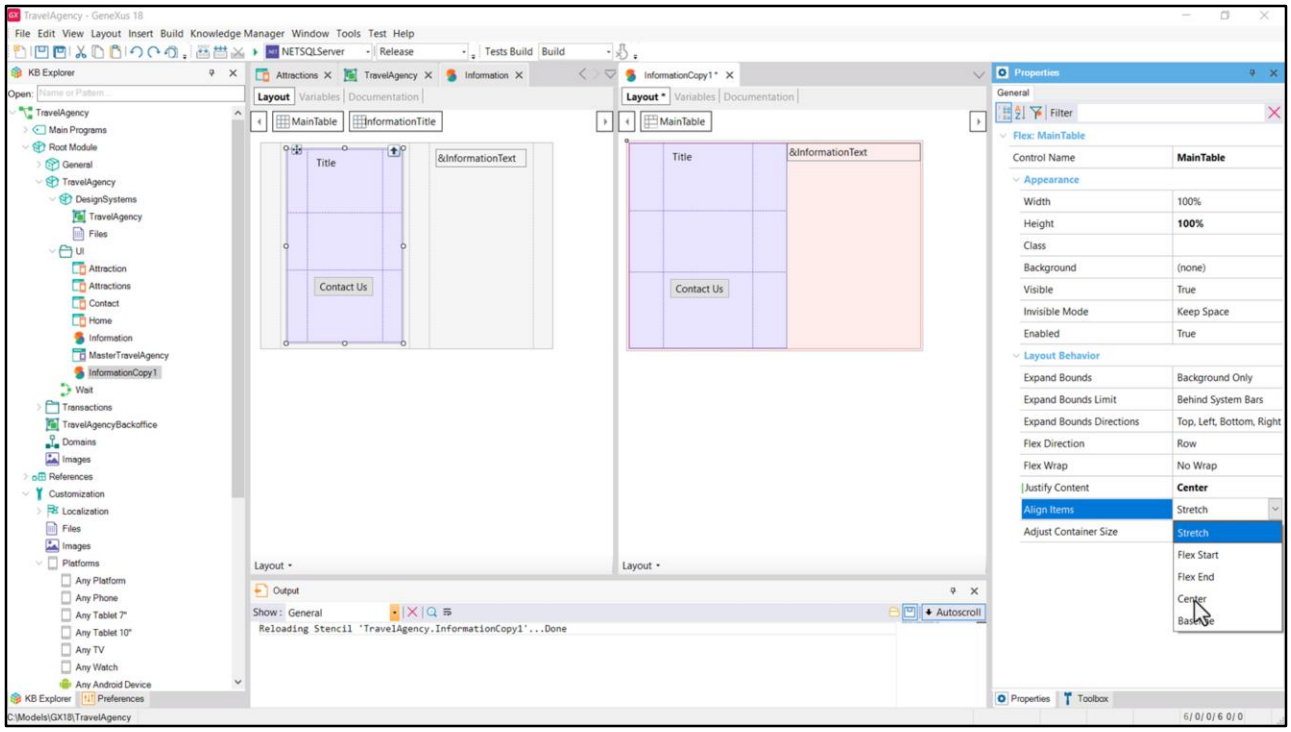
Por ejemplo, este sería el caso de un contendor en dirección row, este otro row reverse, este en dirección column y este otro en column reverse. En el ejemplo cada uno de los contenedores contiene 3 ítems.



En este otro ejemplo decimos si se permite o no que se haga wrap si el tamaño de los ítems, es decir, el ancho que van ocupando, sumados, sobrepasara los límites del contenedor. Entonces en este caso estamos permitiendo el wrap, es decir, que se extienda en una línea abajo, dado que la dirección es row, claro. Si la dirección fuera columna se extendería en una segunda columna.



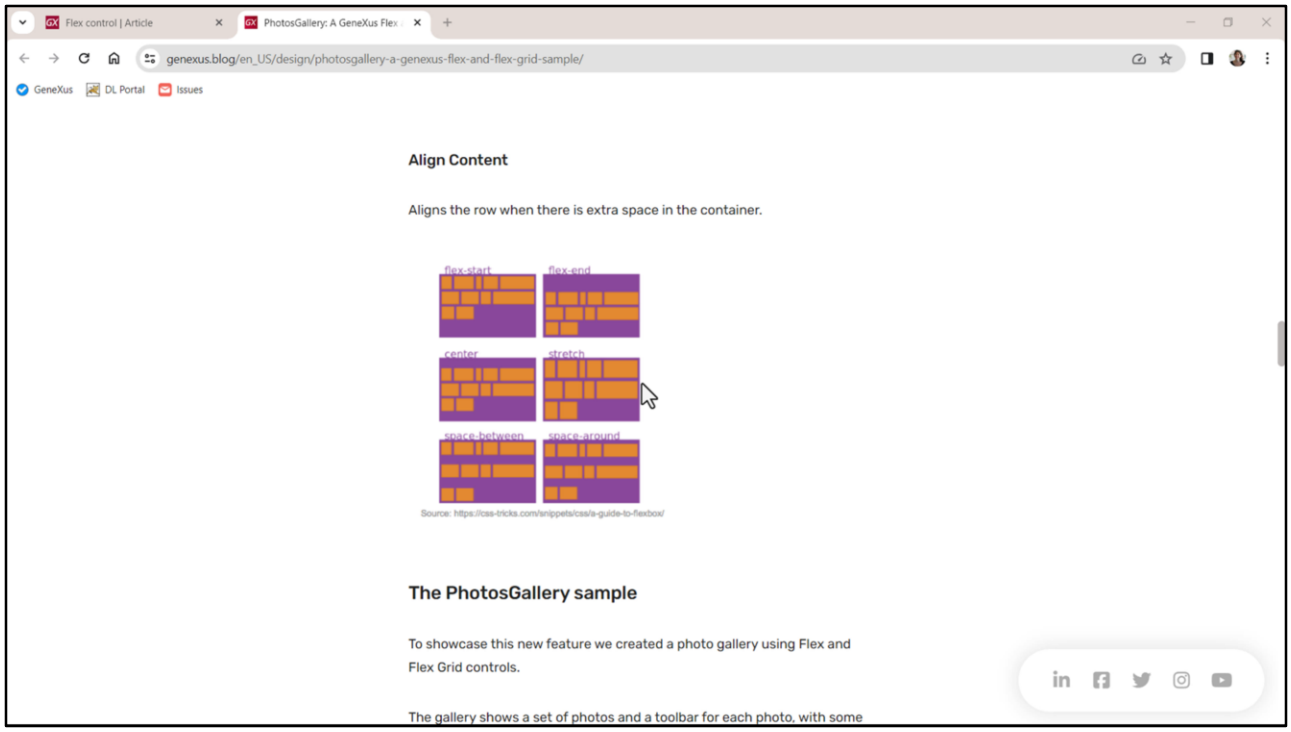
Esta otra propiedad indica cómo se desea que el contenido se justifique. Por ejemplo, si queremos que se centren los ítems respecto a la dirección, que en este caso es row. Este justamente sería el caso de nuestro flex Information.



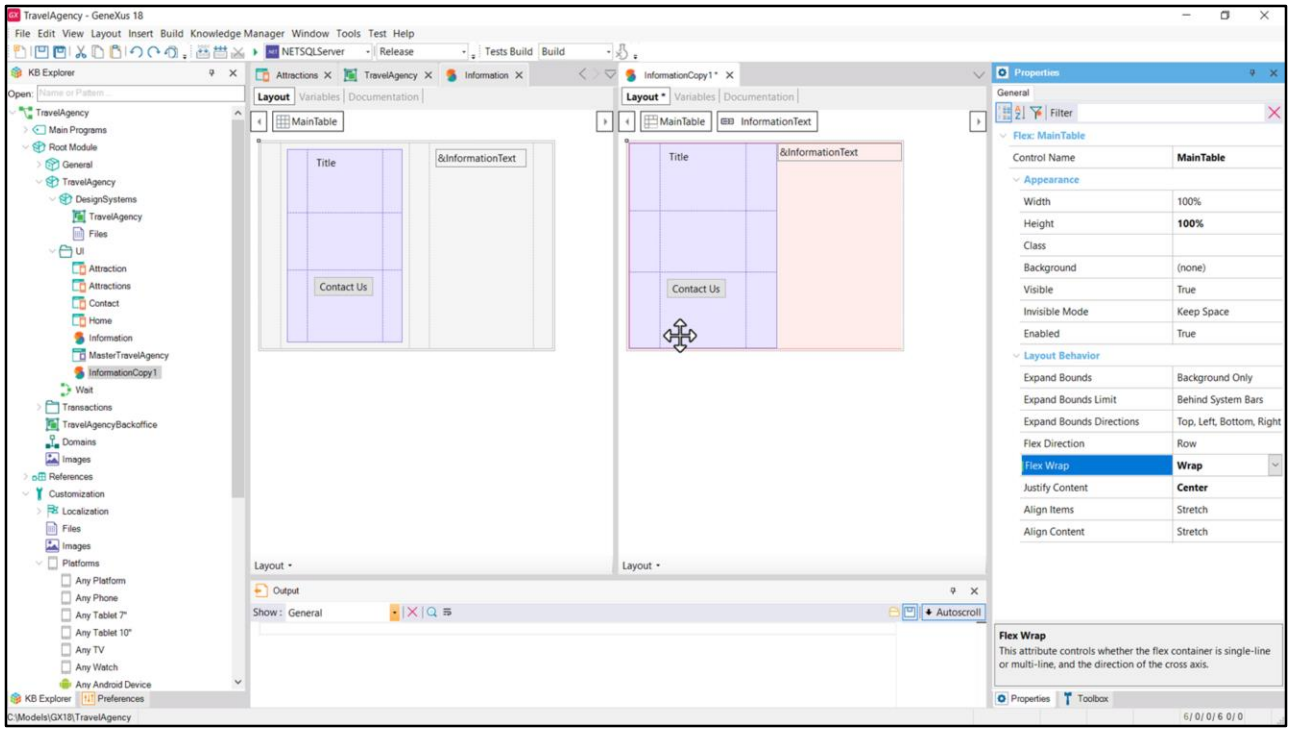
Así que venimos acá, vamos a irlo haciendo… Flex Direction “Row”. Flex Wrap vamos a dejar un “No Wrap” en principio. Y la justificación “Center” (vemos las opciones).

Y la alineación de los ítems respecto a arriba y abajo diremos que sea también centrada.

Y aquí nos quedará claro lo que estamos diciendo con eso.



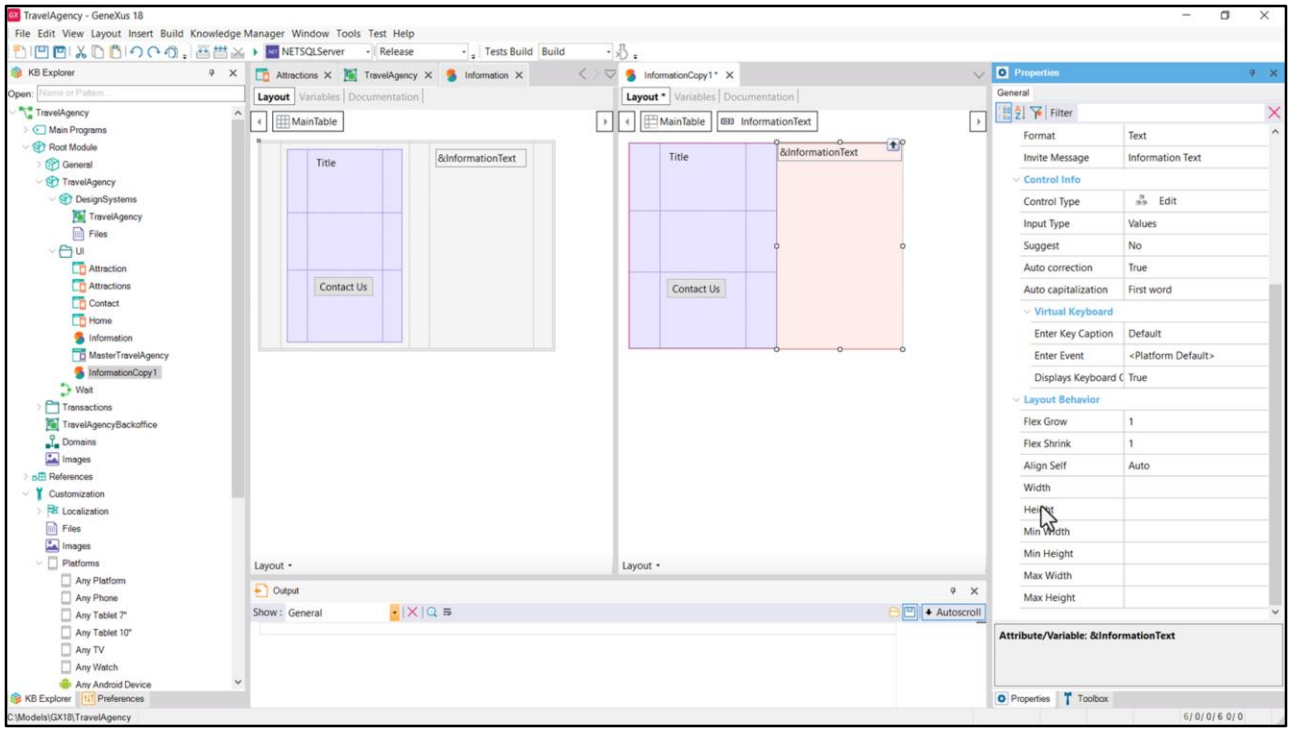
Por otro lado, si permitimos el wrap, entonces con esta propiedad determinamos cómo ajustar el contenido respecto al espacio sobrante.



Bueno, pero tampoco pretendía contarles todas las posibilidades del control, sino simplemente introducirlo porque es un control muy utilizado tanto en web como en aplicaciones nativas, justamente por su flexibilidad.

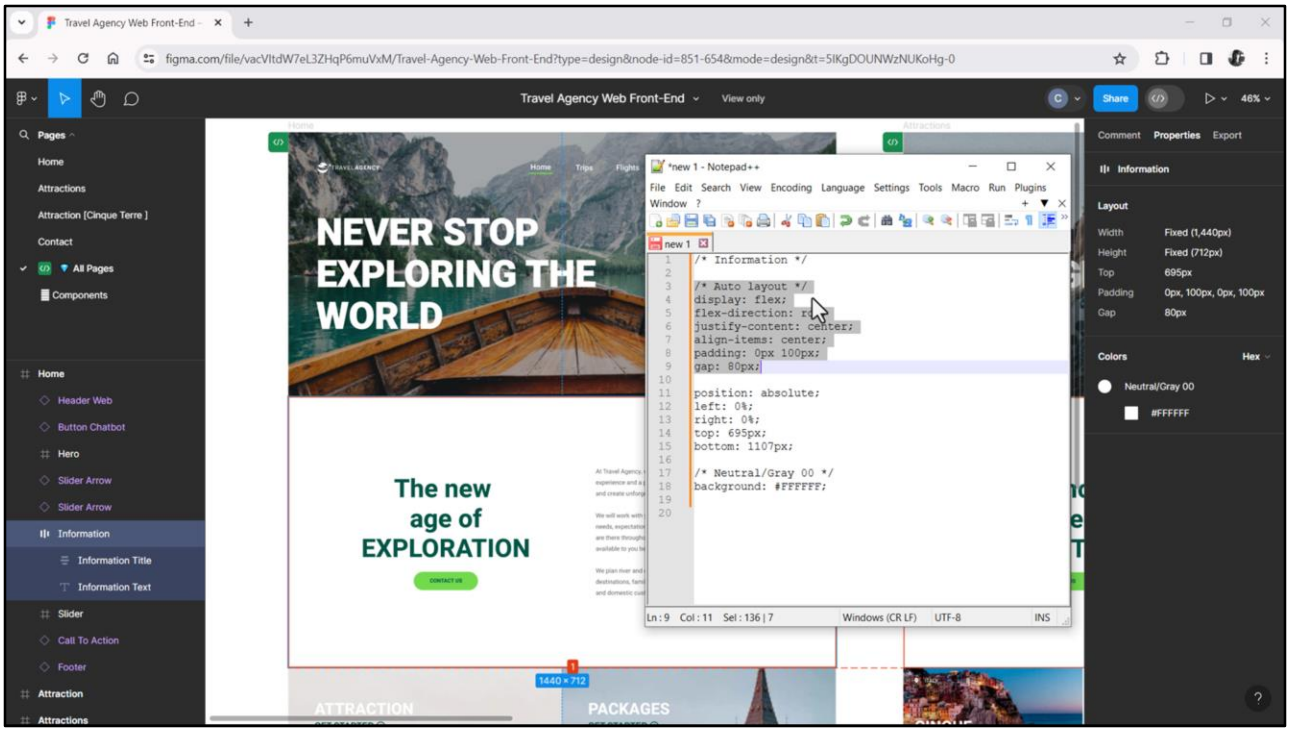
En este caso, por ejemplo, nos sería más útil este control antes que el control Table si quisiéramos, por ejemplo, que cuando el ancho de pantalla disminuya si no entran los dos elementos en ese ancho, entonces aparezcan en dos filas (es decir, con Wrap). La tabla no nos permite esto, es mucho más estructurada: tiene filas y columnas. Y en cantidades fijas.

Aquí solamente tenemos una fila (si es que la dirección es row, como esta) sin columnas, es decir, donde los elementos se van ubicando uno al lado del otro conforme al espacio disponible. Y si se quedan sin espacio, si tienen la propiedad Wrap prendida, bueno, entonces se arma una segunda fila y así sucesivamente hasta completar todos los ítems.



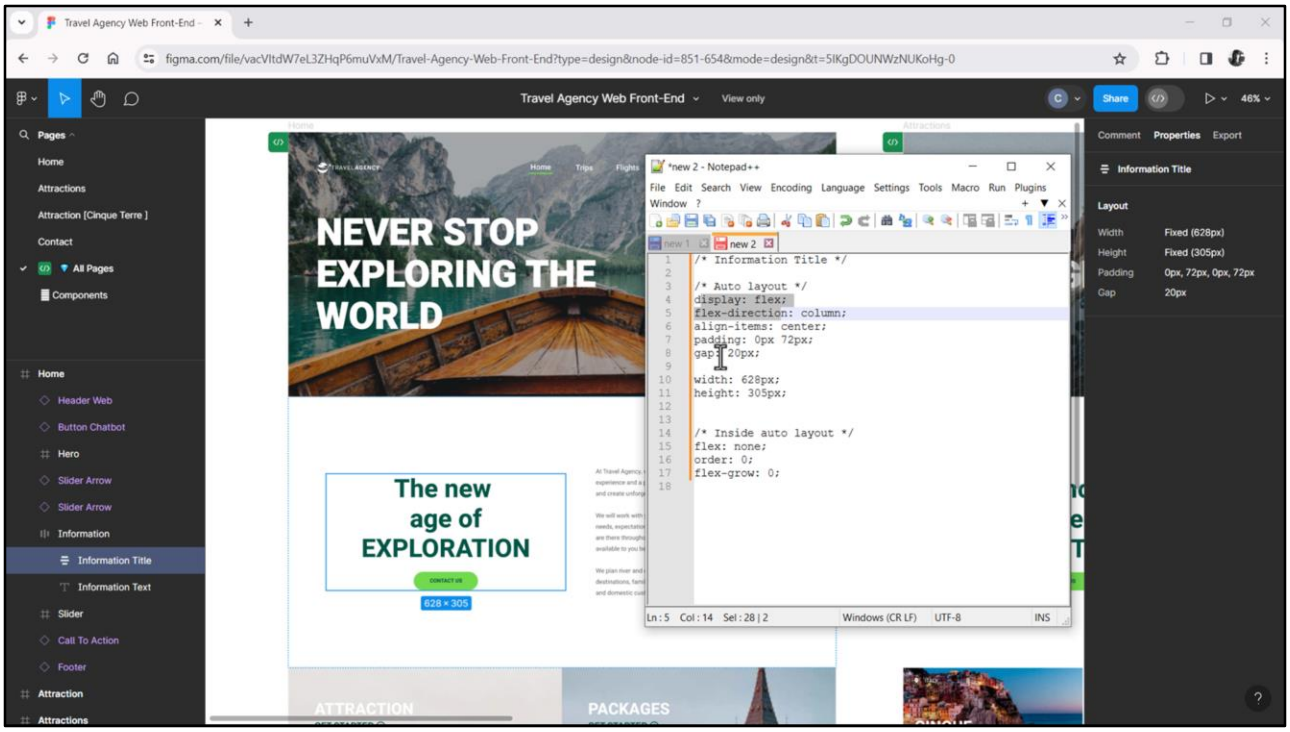
Cuando un control se encuentra dentro de un contenedor flex, adquiere propiedades, estas de aquí, justamente por esa situación.

Por supuesto, como con cualquier otro control, podemos definir propiedades de diseño del flex, todas las que vimos de manera estática, o las más conocidas de margin, padding, etc, a través de clases.



Por ejemplo, volviendo a Figma, cuando Chechu definió a este contenedor con Auto Layout, determinó esta dirección horizontal, y le aparecieron para definir paddings respecto a los cuatro bordes: arriba, derecha, abajo, izquierda. Y también le apareció la posibilidad de definir el Gap, que es la separación horizontal entre los elementos, que ella había colocado que era de 80px.

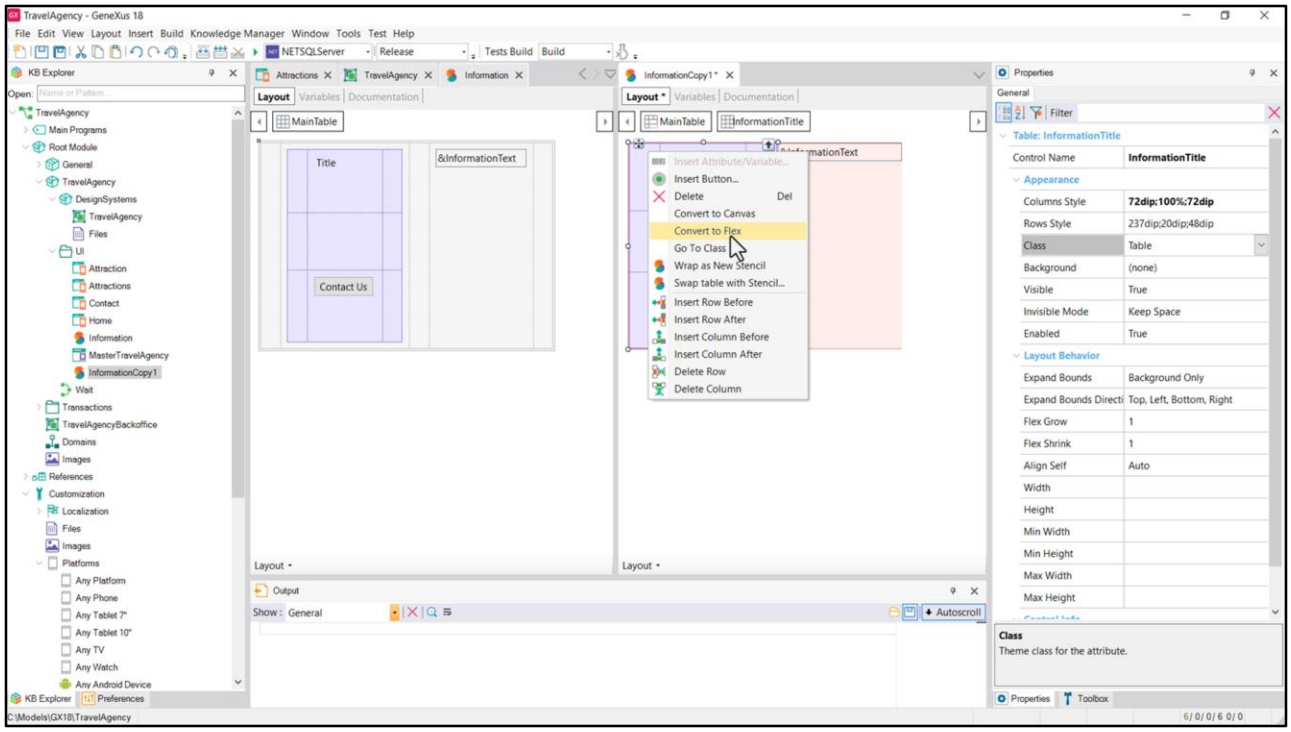
Miren qué código CSS nos aparece cuando lo pedimos…
¿No les suena conocido todo esto? Display flex, flex-direction row, justify-content center, align-item center. Estas propiedades las indicamos estáticamente a nivel de nuestro control flex, y estas dos, padding y gap las podríamos colocar en una clase que le asociaremos luego a nuestro control flex. O también podríamos colocarlas todas dentro de esa clase, porque todas las que podamos definir a nivel del control también podemos definirlas a nivel de propiedades de una clase, ¿no?



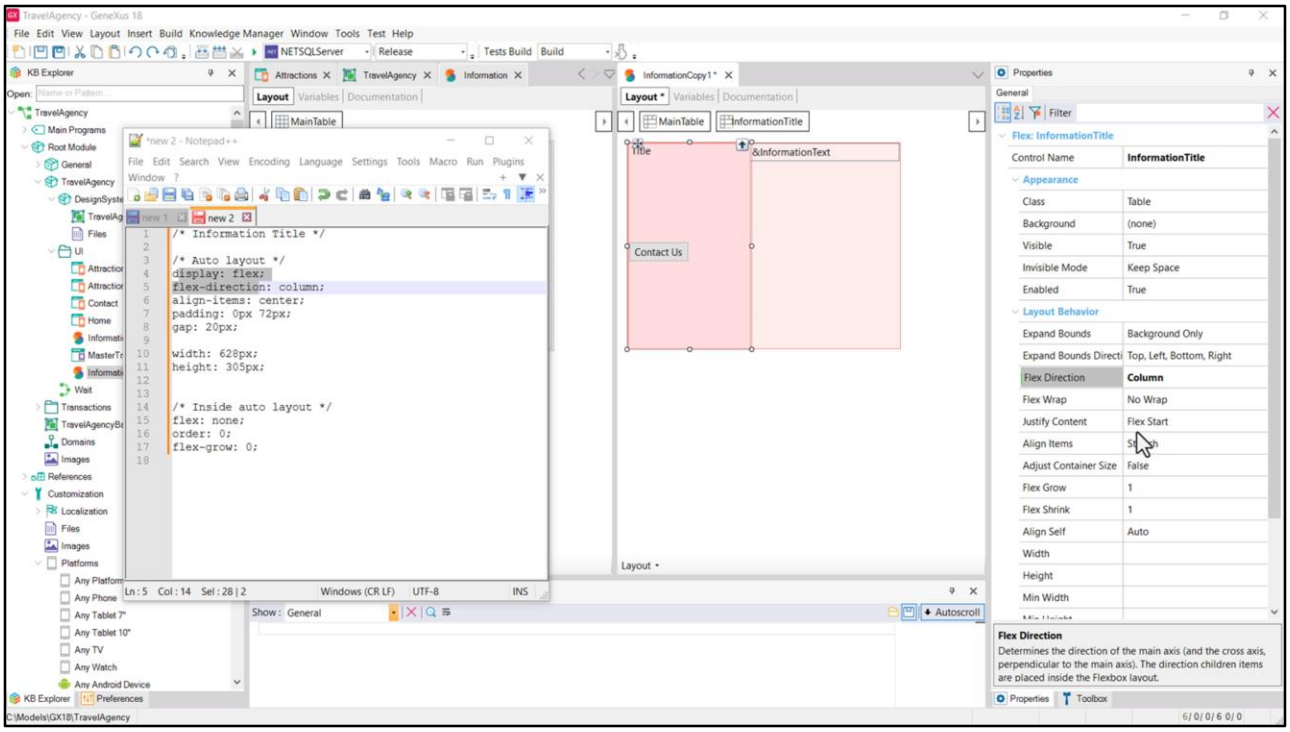
Bueno, del mismo modo, vemos que este otro contenedor también tiene Auto Layout, pero en la otra dirección, en la dirección column.

Acá vemos las propiedades Padding y Gap.

Y otra vez, si copiamos las propiedades CSS, vemos acá la display, la direction: column esta vez, la align-ítems y las propiedades padding y gap.

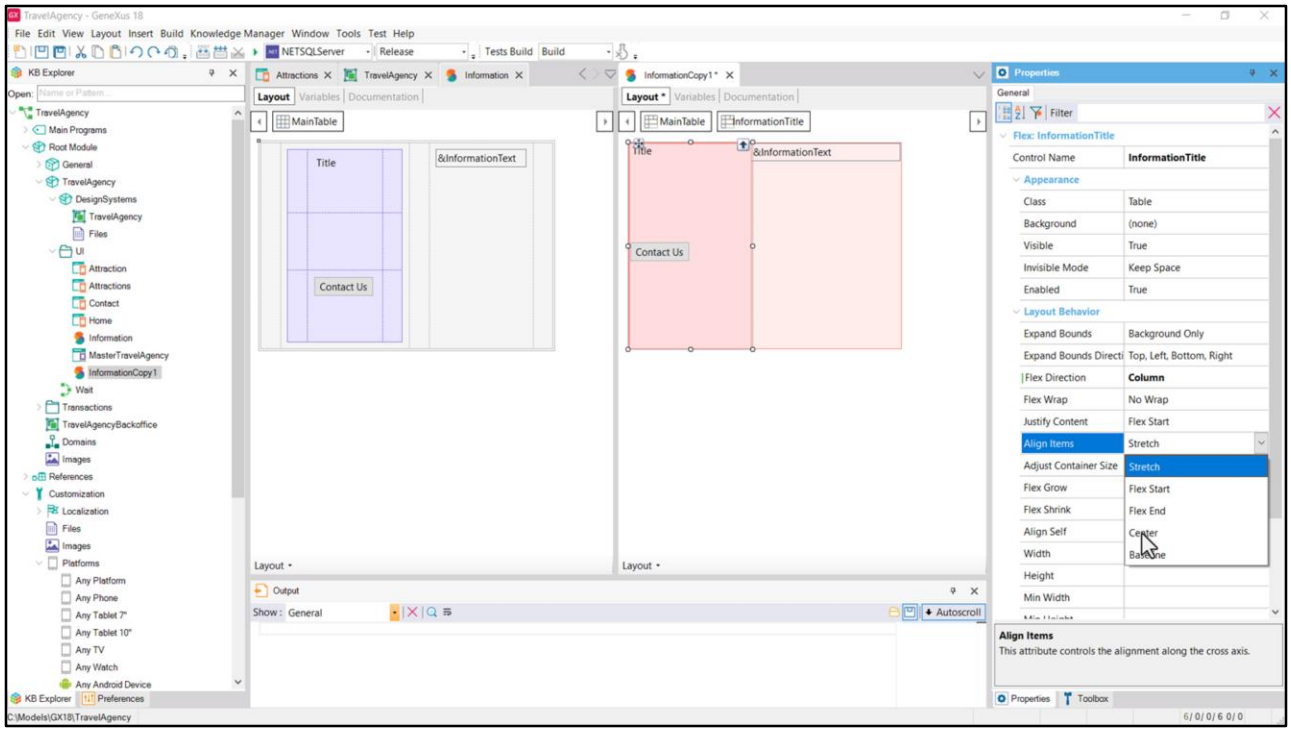


Así que podríamos también definir esta tabla como Flex.

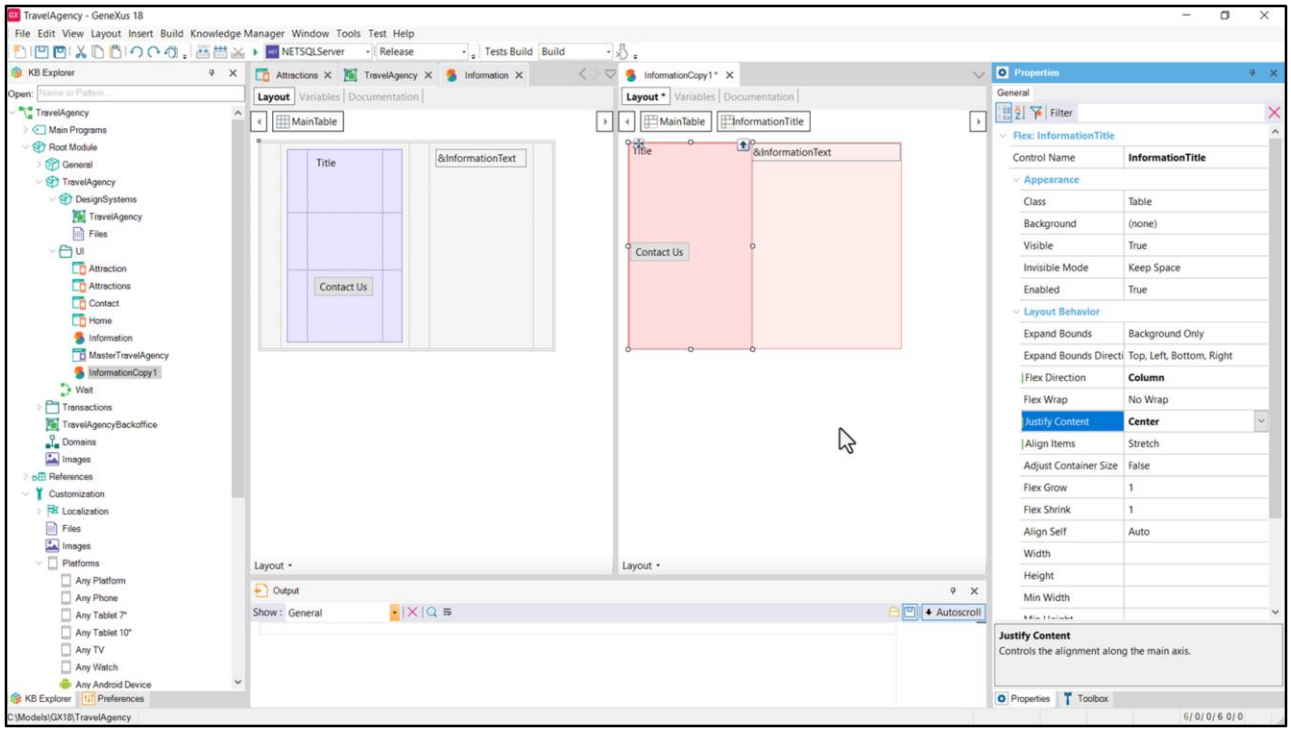


Y colocarle, por supuesto, la dirección column para que se nos vea de esta manera.

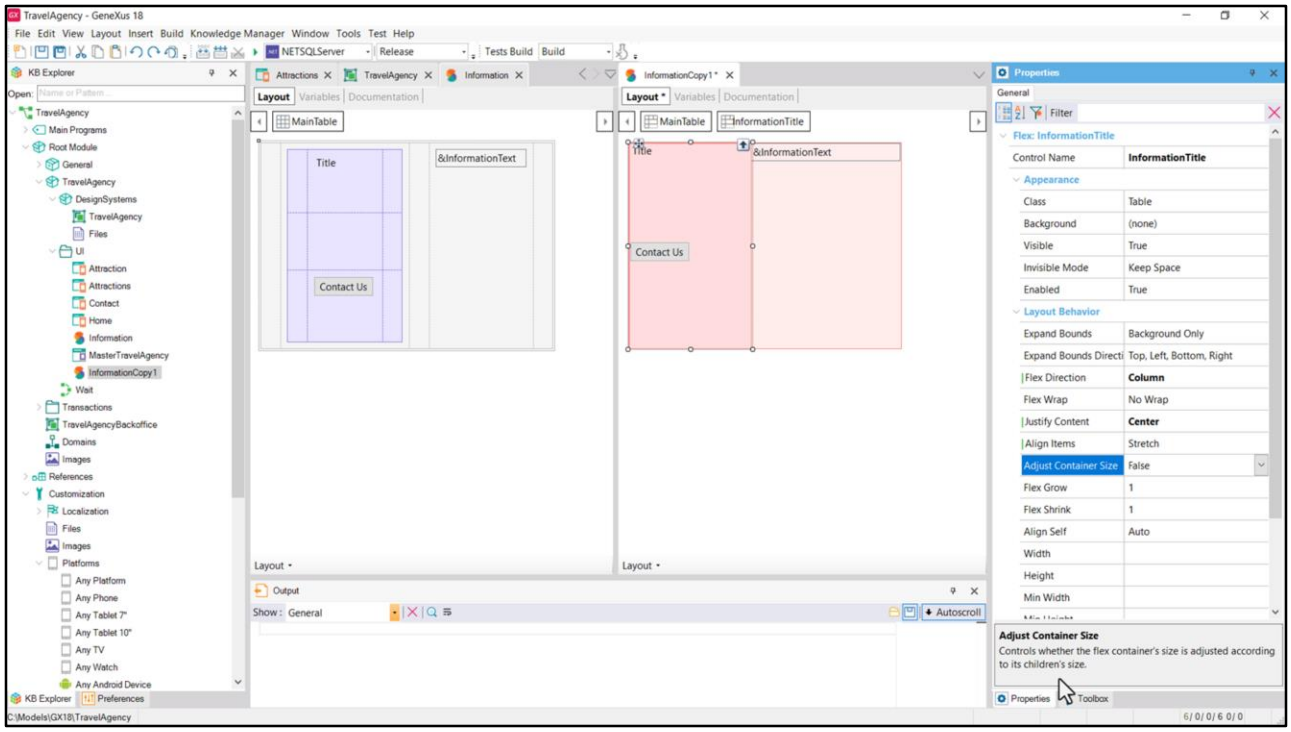
Del CSS de Figma también extrajimos esto, que el align-items es centrado, que significa estos ítems en relación a los bordes de la otra dirección, ¿no?



Y bueno, acá entonces tendríamos que elegir Center. Bueno, están desapareciendo, esto no debería estar pasando… voy a dejar el default, luego vemos.

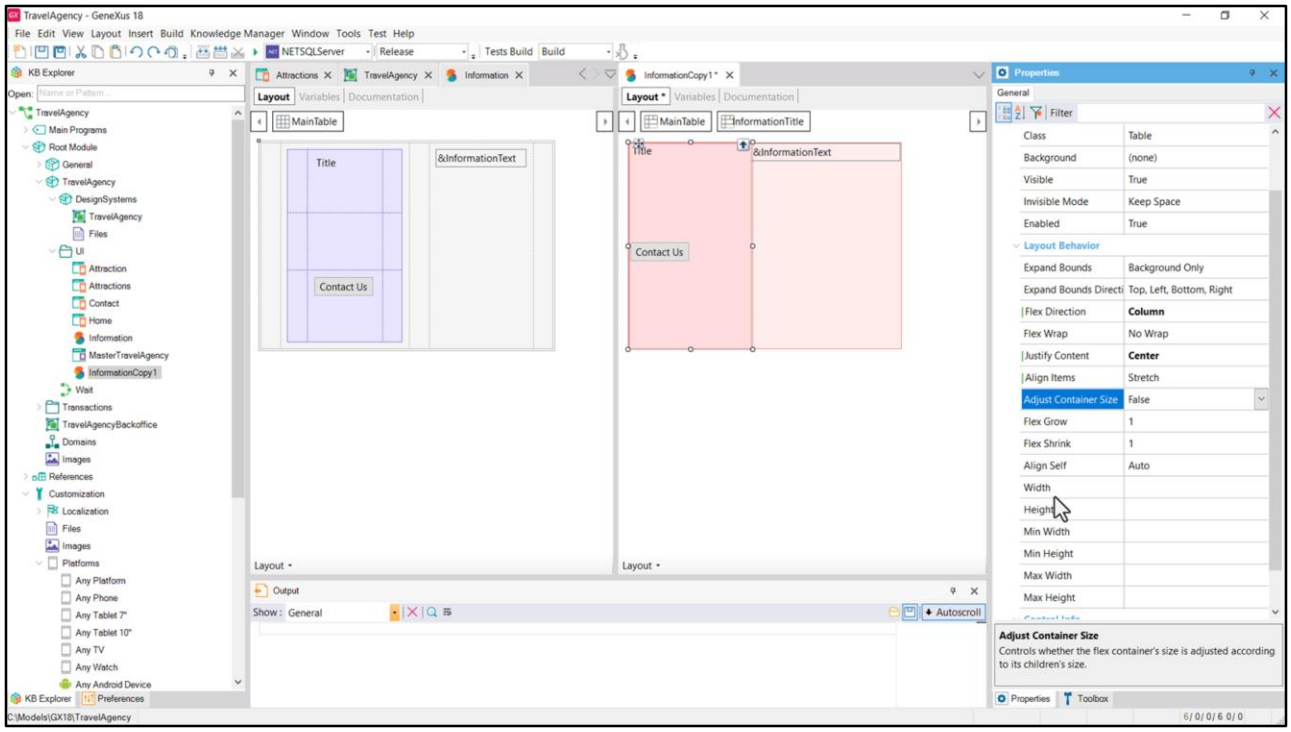


Y por otro lado la justificación del contenido sí tiene que ver con la misma dirección del flex, ¡no?, que en este caso es columna, por tanto también le tendríamos que decir centrado, ¿no? para nuestro caso, aunque no nos dijera nada el CSS.

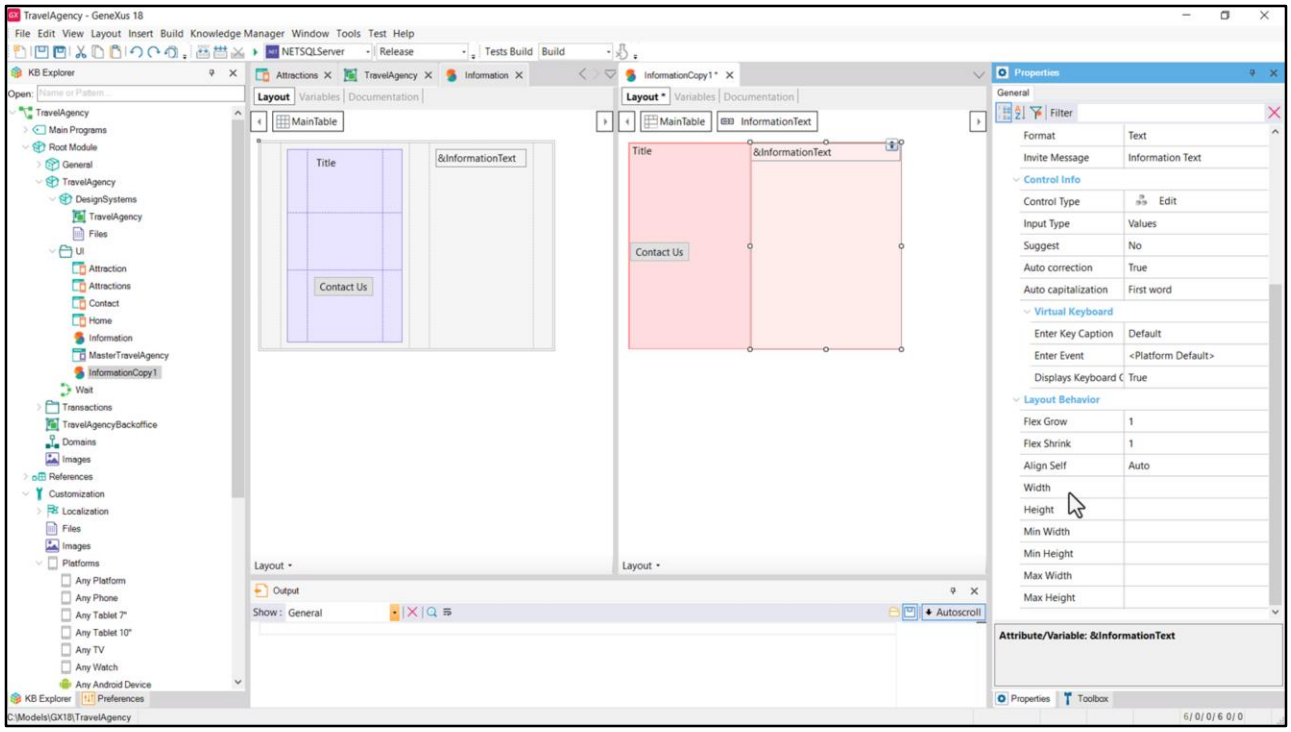


Por otro lado como no tiene wrap, no determinamos que tuviera Wrap, entonces esta propiedad juega y vemos que lo que dice es que controla si el tamaño del contenedor se va a ajustar de acuerdo al tamaño de los hijos.

Bueno, entonces tendríamos que jugar con esas cuestiones para poder ubicar nuestros elementos de manera correcta de acuerdo al diseño de Chechu. Puede parecer en un principio más simple utilizar los contenedores flex porque vieron que podemos copiar directamente esos criterios de diseño de Chechu… con el Auto Layout podemos medio que trasladarlo casi que automáticamente a GeneXus. Pero tenemos que tener los cuidados de que esto es un poquito más engorroso…



…ven todas las propiedades que están apareciendo ahora dentro de este grupo… en este caso aparecen más porque están apareciendo también las que hacen a la participación de este flex interno dentro de este otro flex.



Como antes les había mostrado que acá aparecían todas estas propiedades, de tamaño, por ejemplo, de ancho y de alto, que tenían que ver con la participación de este control dentro del Flex padre.

Bueno, por estas cuestiones un poquito más engorrosas es que preferí para empezar, para enseñarles, arrancar con tabla y no con flex. El flex puede parecer un poco más engorroso por esas propiedades que hay que empezar a dominar mejor, uno a veces se entrevera, si estoy hablando de justificación del contenido en qué dirección es, si tengo la de alineación del ítem… entonces me empiezo a entreverar, pero una vez que uno le agarra la mano a esto es bastante sencillo. Y bueno, en realidad hay que utilizarlo con criterio.

¿Cuándo utilizar el flex? Cuando en el diseño que estoy implementando los elementos que estarán por ejemplo a lo largo de una fila (si es esa la dirección) no están tan estructuralmente fijados en columnas, y, sin duda, por supuesto, cuando queremos la funcionalidad del wrap.

En nuestro caso en principio no la necesitábamos, porque ya dijimos que cuando el ancho de pantalla sea uno en el que no entren los dos elementos, en realidad se hará otro diseño distinto, que aún no hemos visto porque le pedí a Chechu que los diseñara en otro archivo, así no nos confundíamos ahora.

Bueno, por ahora voy a dejar por acá, ya quedó introducido, que era lo que quería y luego más adelante veremos si nos enfrentamos a la necesidad de utilizar un contenedor flex en lugar de una tabla. Cuando digo más adelante digo cuando sigamos desarrollando las pantallas de

nuestra aplicación.

También un grid podrá ser flex. Ya lo dejo mencionado y lo volveré a mencionar cuando implementemos uno de los grids que tenemos por delante.

Con esto yo voy a dar por cerrado el video porque quiero pasar al siguiente módulo pero ustedes pueden, si quieren, probar lo que yo no probé, asignarles clases a los flex con las propiedades correspondientes, y prestar atención a los tamaños que les van a dar a los controles, investigar un poco e intentar que la aplicación con el stencil con flex luzca igual que de la otra manera, como la teníamos hecha. Bueno, y ese es el desafío que les dejo a ustedes.

Nos vemos en el siguiente módulo.


GeneXus by Globant
GeneXus™
by Globant
training.genexus.com