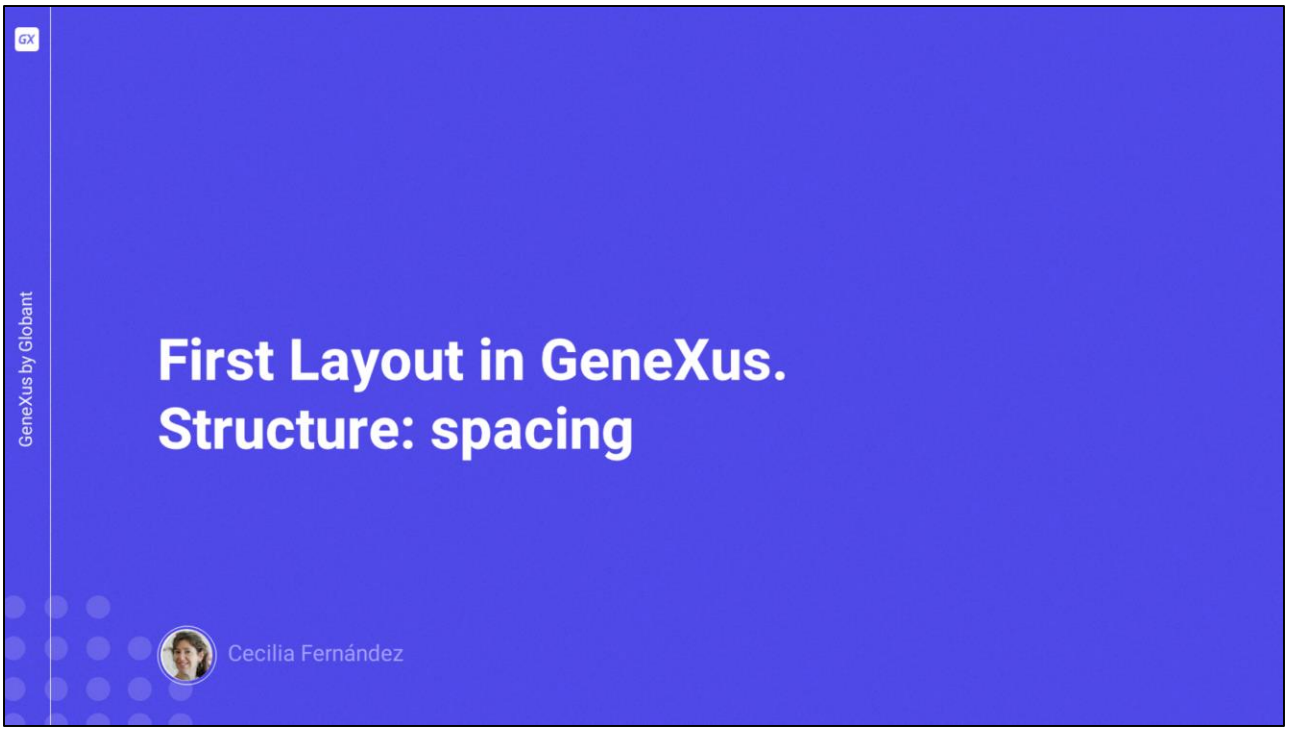
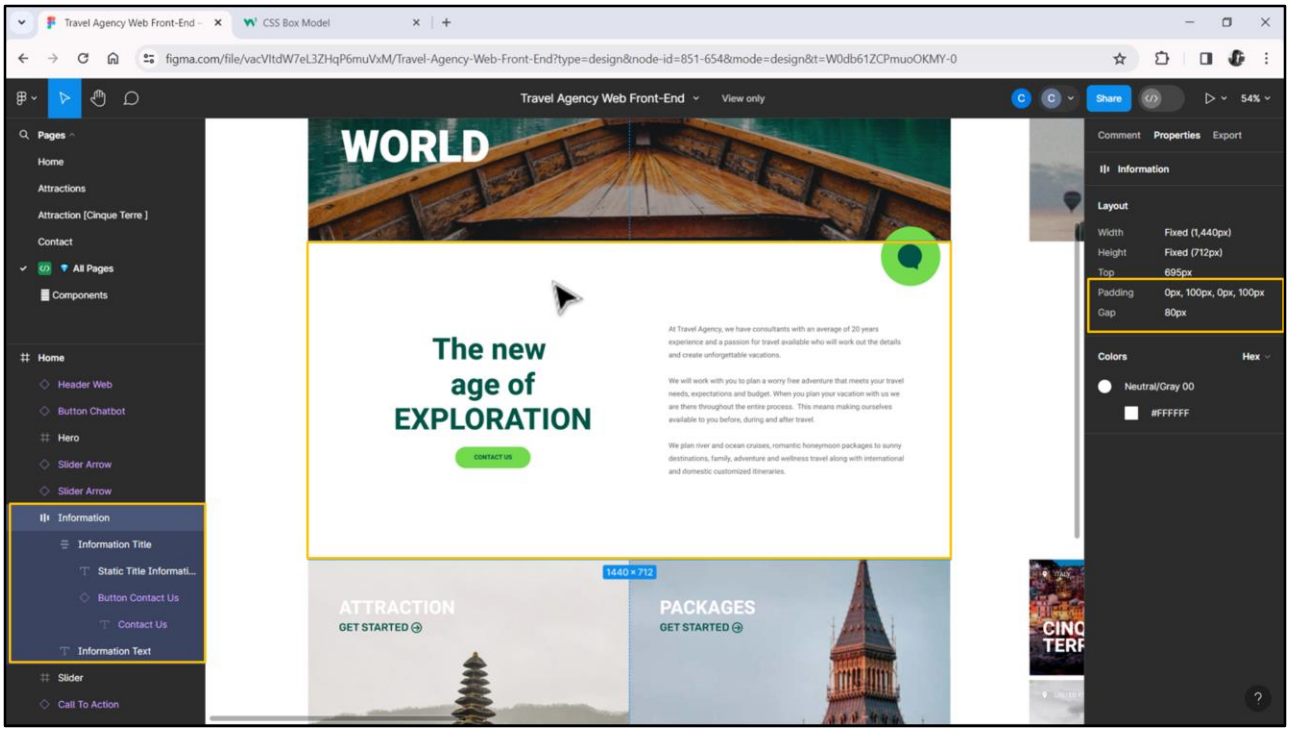
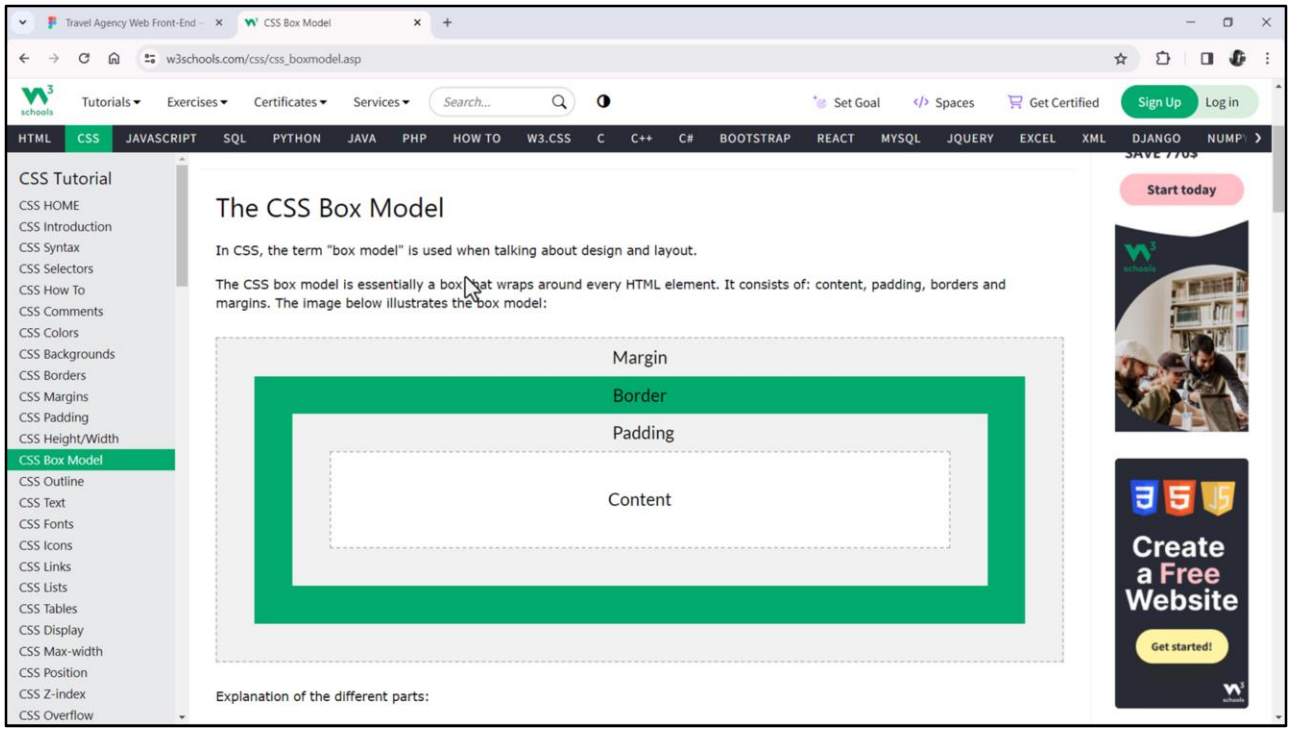


Vamos continuar de onde paramos no vídeo anterior.

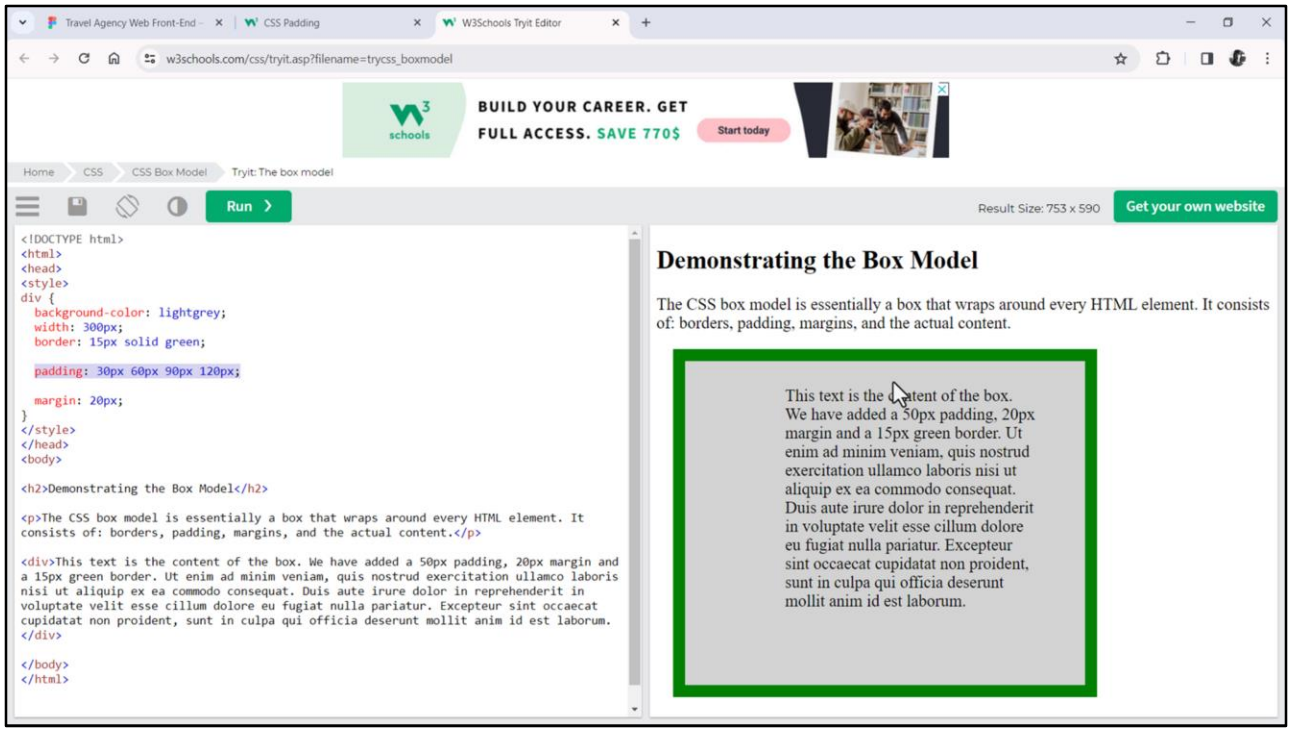


Lá havíamos dito que antes de investigar os tamanhos desta tabela interna iríamos ver alguns aspectos da tabela externa.

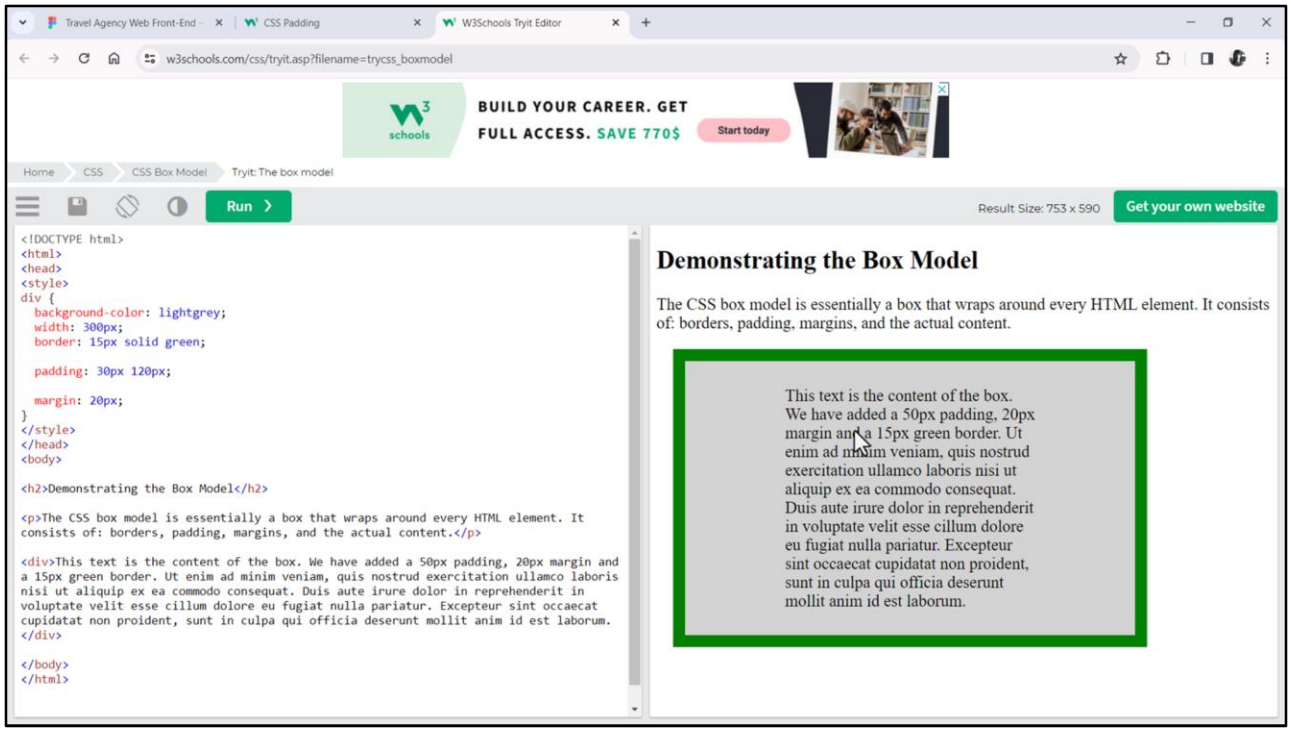
O que mais sabemos ao inspecionar este frame? Vemos aqui essas propriedades e, em particular, vamos prestar atenção a estas duas, que terão a ver com o espaçamento, justamente, entre o conteúdo.



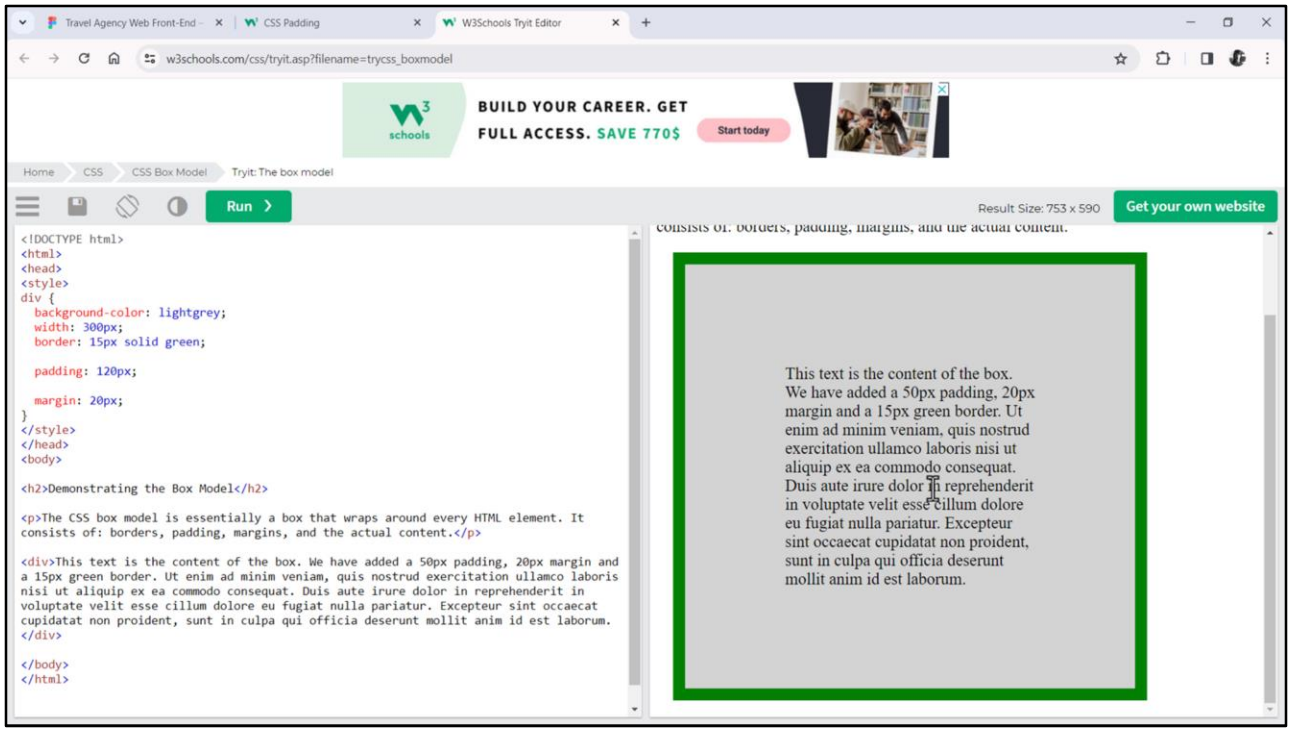
Já sabemos que o Padding dentro do box model de CSS é o espaçamento interno do conteúdo em relação às 4 bordas...
Ou seja, este é o limite do elemento, que nesta representação possui uma borda deste tamanho, e um espaçamento interno deste tamanho.



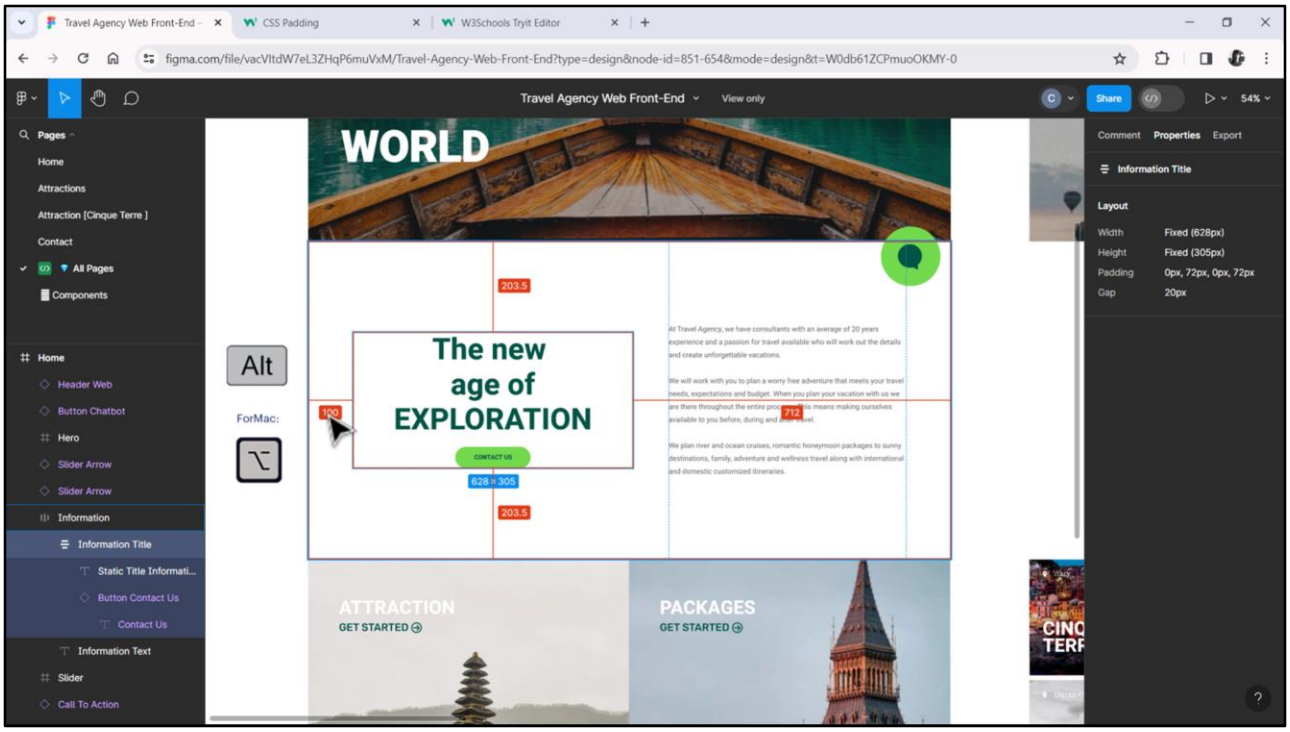
Como sabemos, o Padding não precisa ser uniforme. Aqui estamos vendo um exemplo, um teste, onde estou colocando padding-top, o padding em relação à borda superior, de 30 pixels; o padding em relação à borda da direita, padding-right, de 60; em relação abaixo 90; e em relação à esquerda, 120.
Esta é a notação abreviada. Poderíamos utilizar cada uma das propriedades de forma separada.



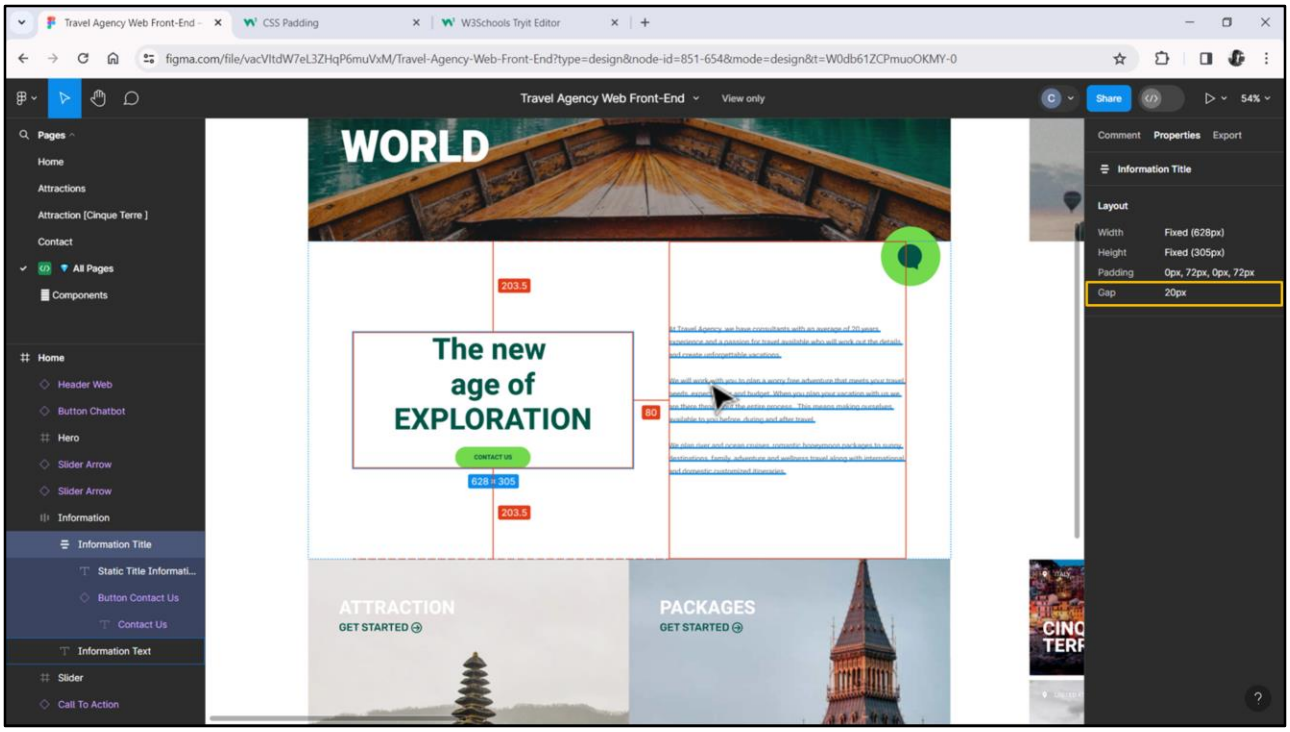
E se acima e abaixo, por exemplo, queremos que sejam iguais e direita e esquerda também, então abreviamos desta maneira, e aí vemos como acima e abaixo estão com 30, direita e esquerda com 120...



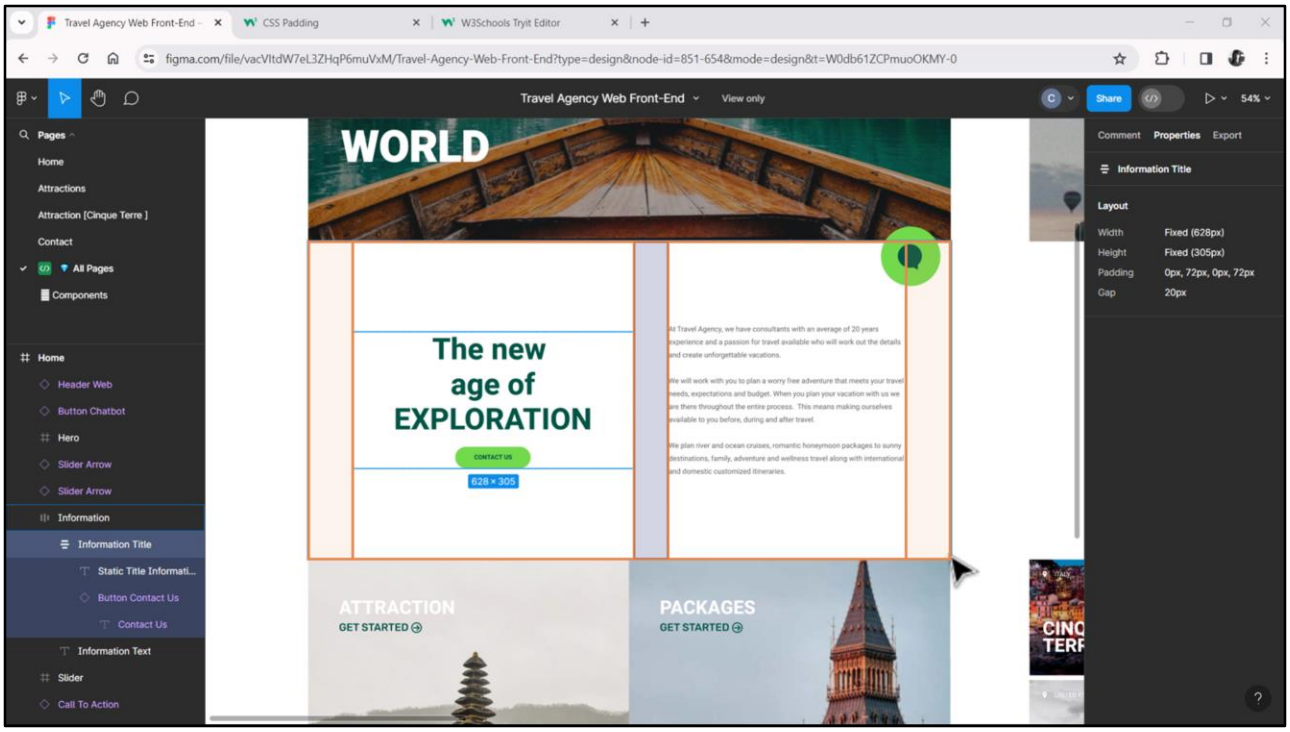
… ou mesmo se deixarmos apenas um, isso significa que os quatro lados vão aplicar o mesmo padding.



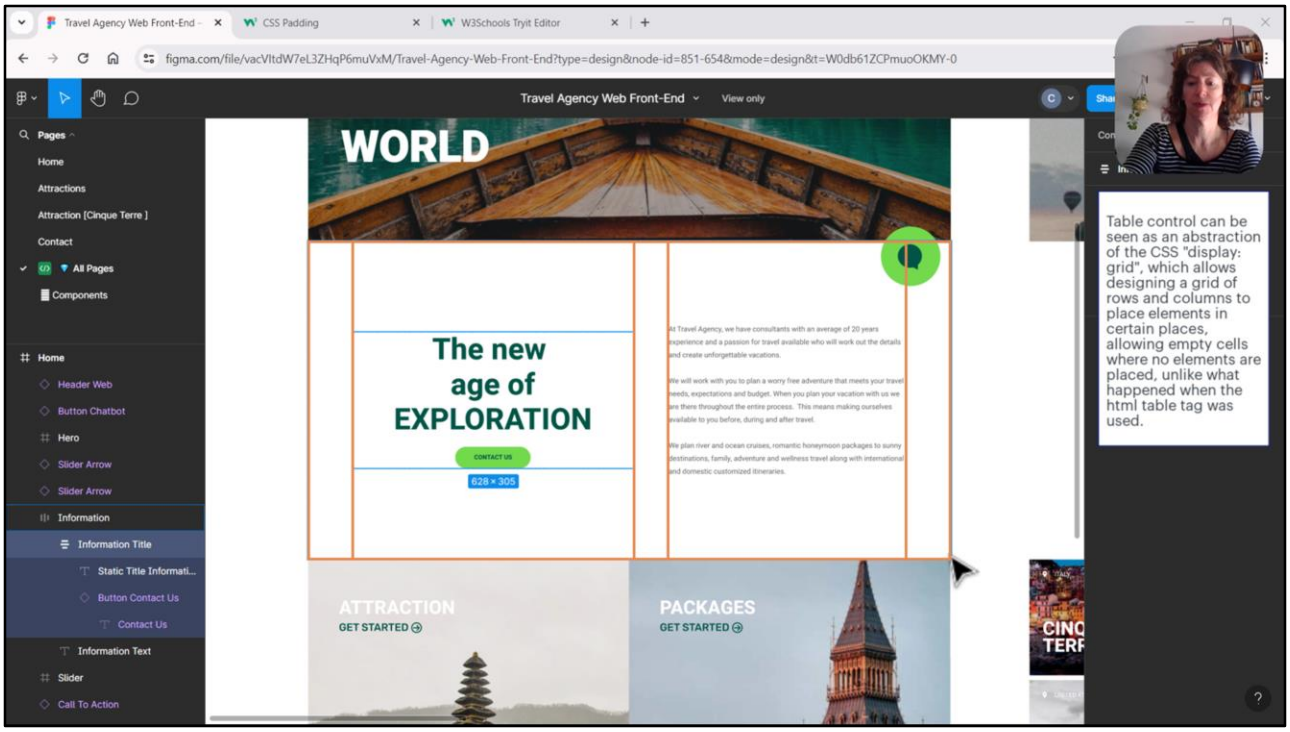
Tendo entendido isso, então, sabemos aqui que haverá um padding de 100 pixels à esquerda e de 100 pixels à direita. E também podemos ver isso graficamente escolhendo o primeiro elemento e com a tecla ALT se estivermos no Windows vemos estes 100 pixels aqui... e se agora nos posicionarmos neste outro elemento vemos estes 100 pixels aqui.



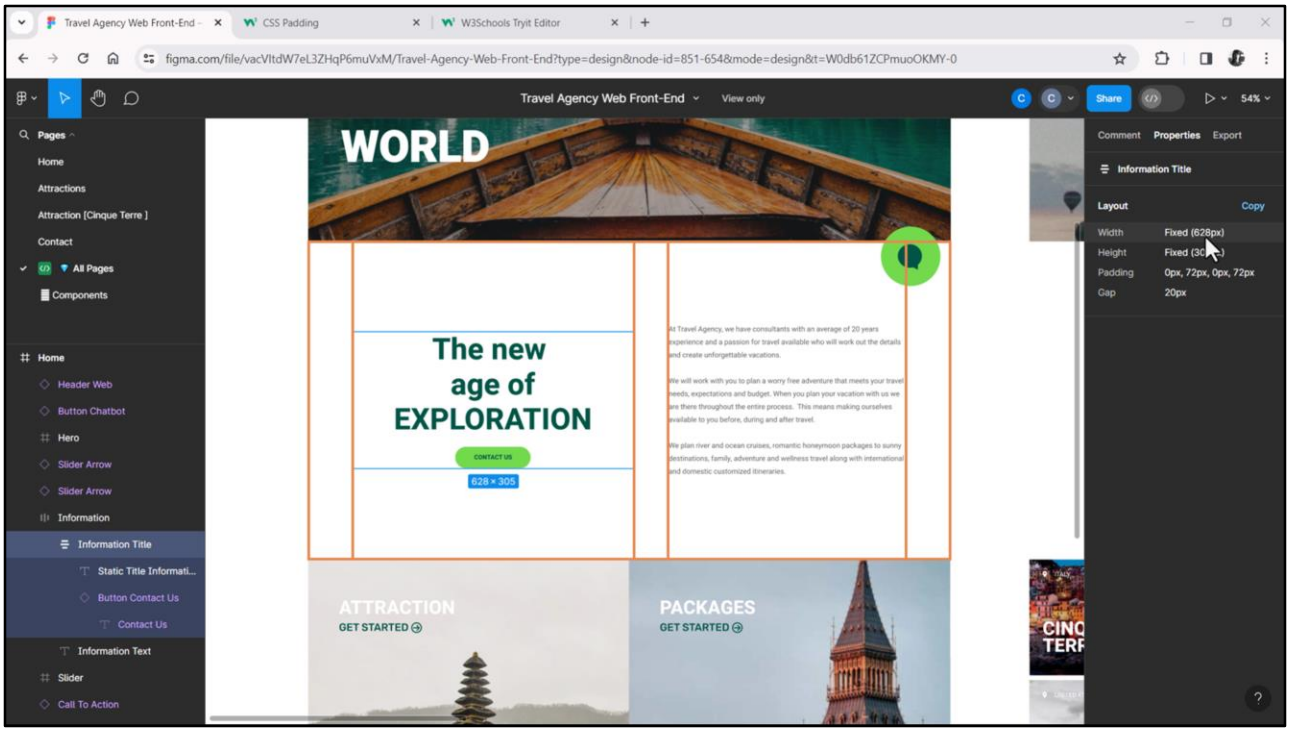
E por último, o que acontece com o gap, o que é o gap? Nem mais nem menos que a distância entre os elementos internos do container. Novamente, neste caso, se pararmos neste, e vermos a relação com este elemento, veremos esses 80 que está indicando ali.



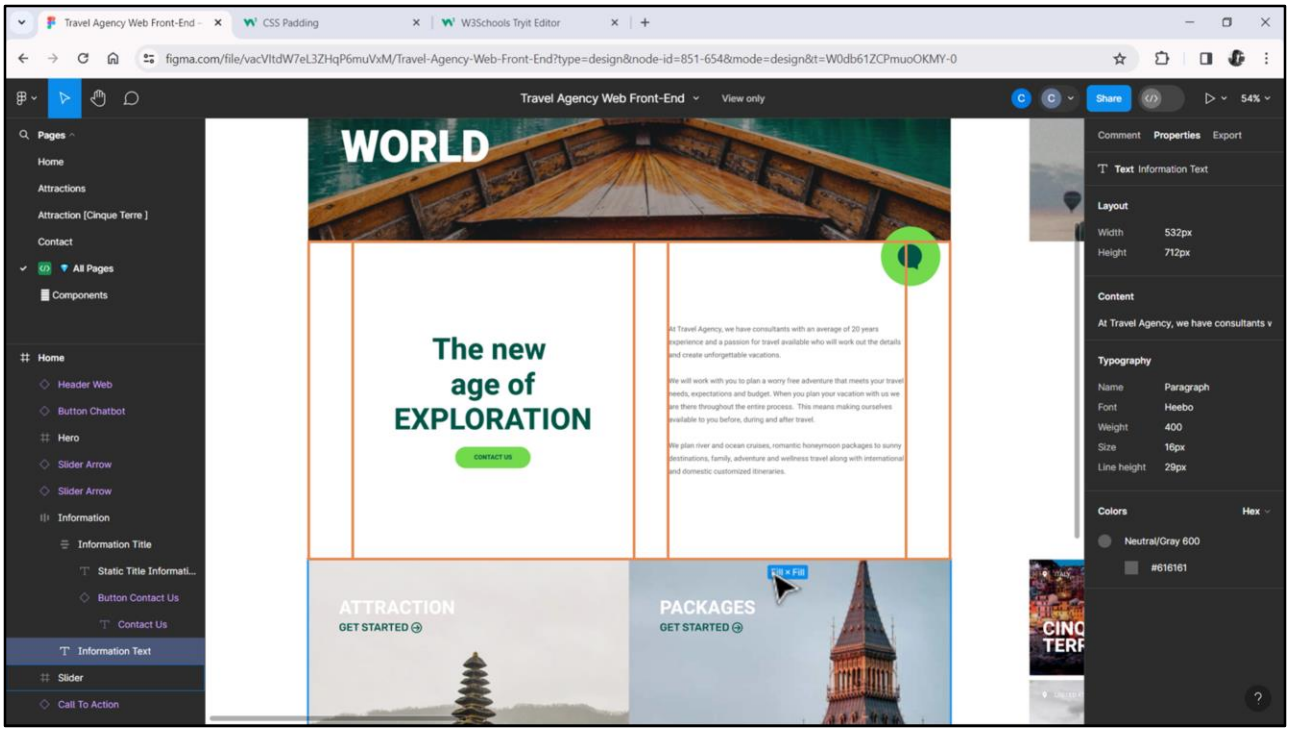
Existem várias formas de estabelecer este espaçamento no GeneXus. Uma será quase uma tradução literal dessas propriedades. Veremos isso mais tarde, quando passarmos a especificar o DSO. Mas outra será mais fortemente estrutural: e se pensarmos nestes espaços vazios não como paddings, mas como colunas vazias? Ou seja, a tabela Main terá 5 colunas em vez de 2: a primeira e a última terão 100 pixels de largura e a do meio terá 80 de largura.



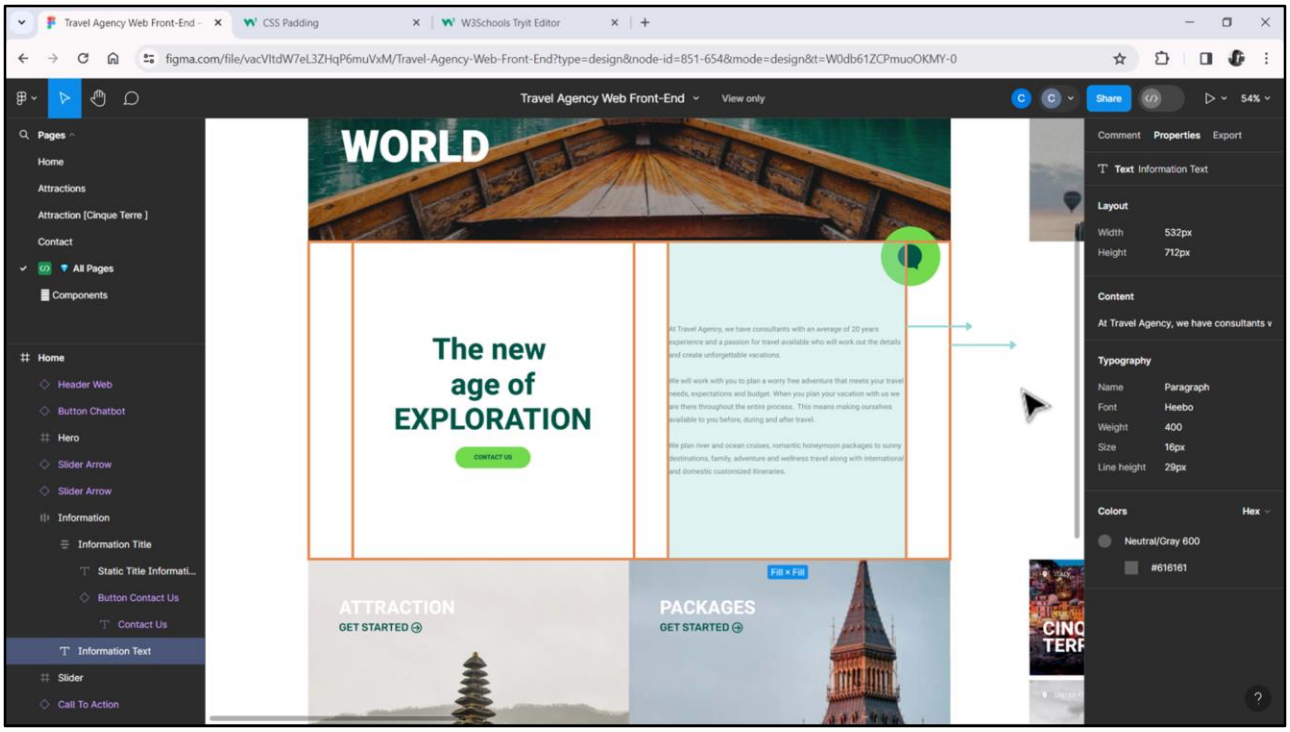
Alguns de vocês podem estar se perguntando se deixar colunas vazias não representará um problema de desempenho. E a resposta é não e deixo aqui uma breve explicação.



E as outras duas, as do próprio conteúdo? Qual largura deveriam ter? Se analisarmos a largura do primeiro elemento, vemos que é fixa, de 628 pixels.



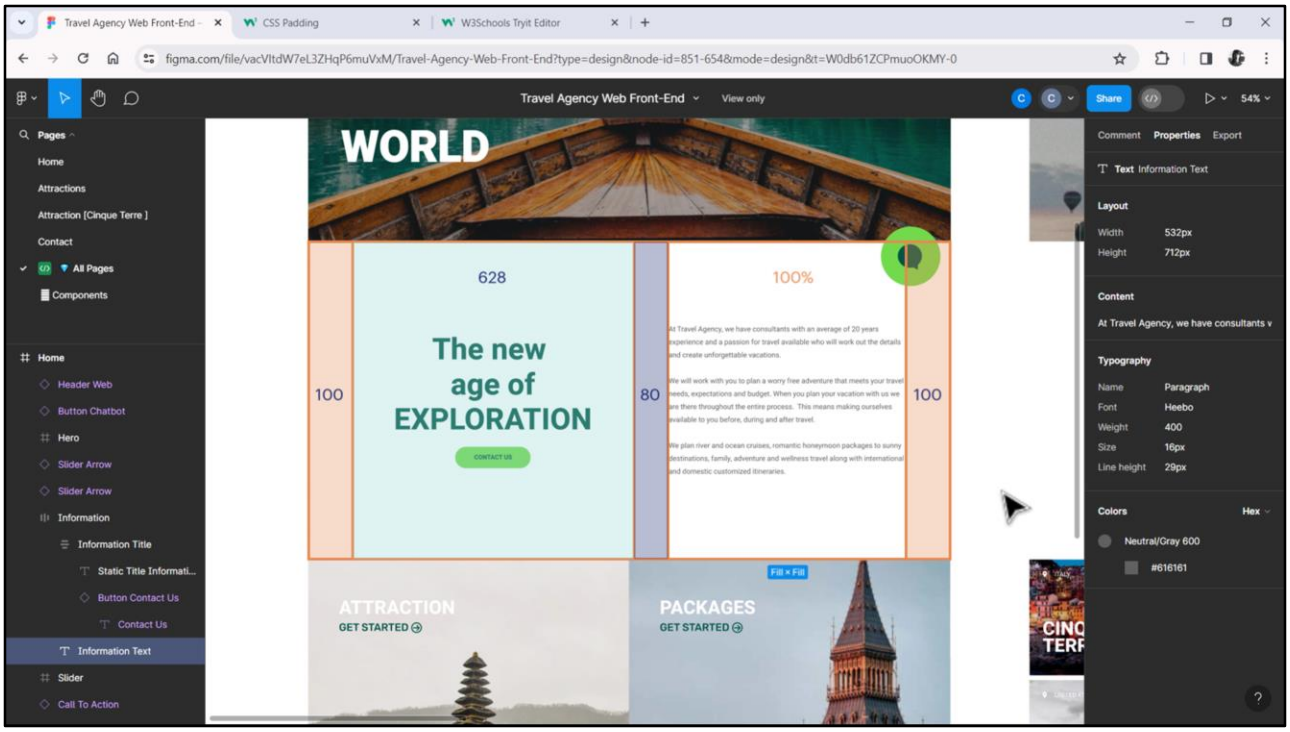
E se analisarmos o segundo elemento, não é fixo, diz Fill. Ou seja, que ocupará todo o tamanho disponível, tanto horizontalmente quanto verticalmente.



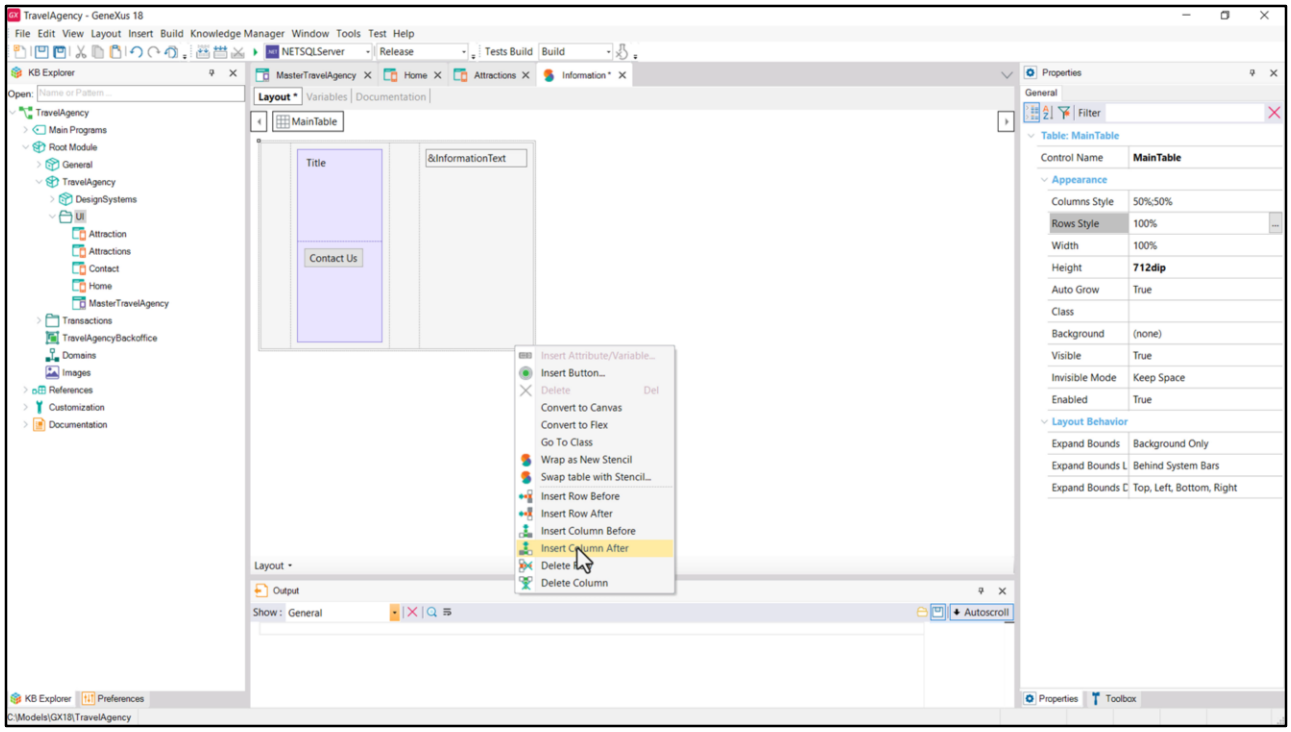
Isso significará que se eu estender para a direita a borda do browser, enquanto o elemento da esquerda permanecerá como está, o da direita também se expandirá para a direita.

E se eu reduzir o tamanho, ele também encolherá. Os espaços, em vez disso, permanecerão sempre fixos, de 100, 80 e 100 pixels.

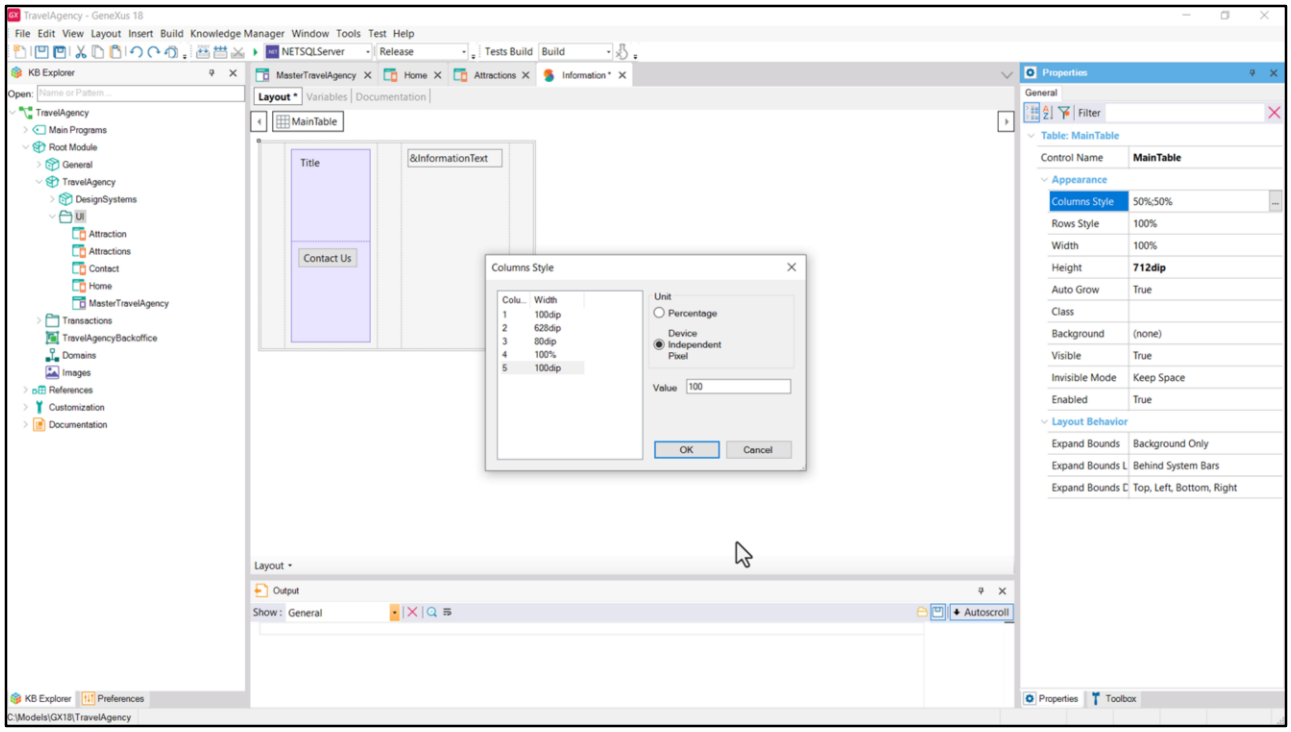
É bom perguntar à designer se o que deduzimos é de fato o comportamento esperado por ela ao projetar desta forma. Se disser que sim, então já sabemos que nossa coluna 2 terá tamanho fixo, 628 dips e, por outro lado, a quarta coluna é a que terá que se adaptar à largura restante.



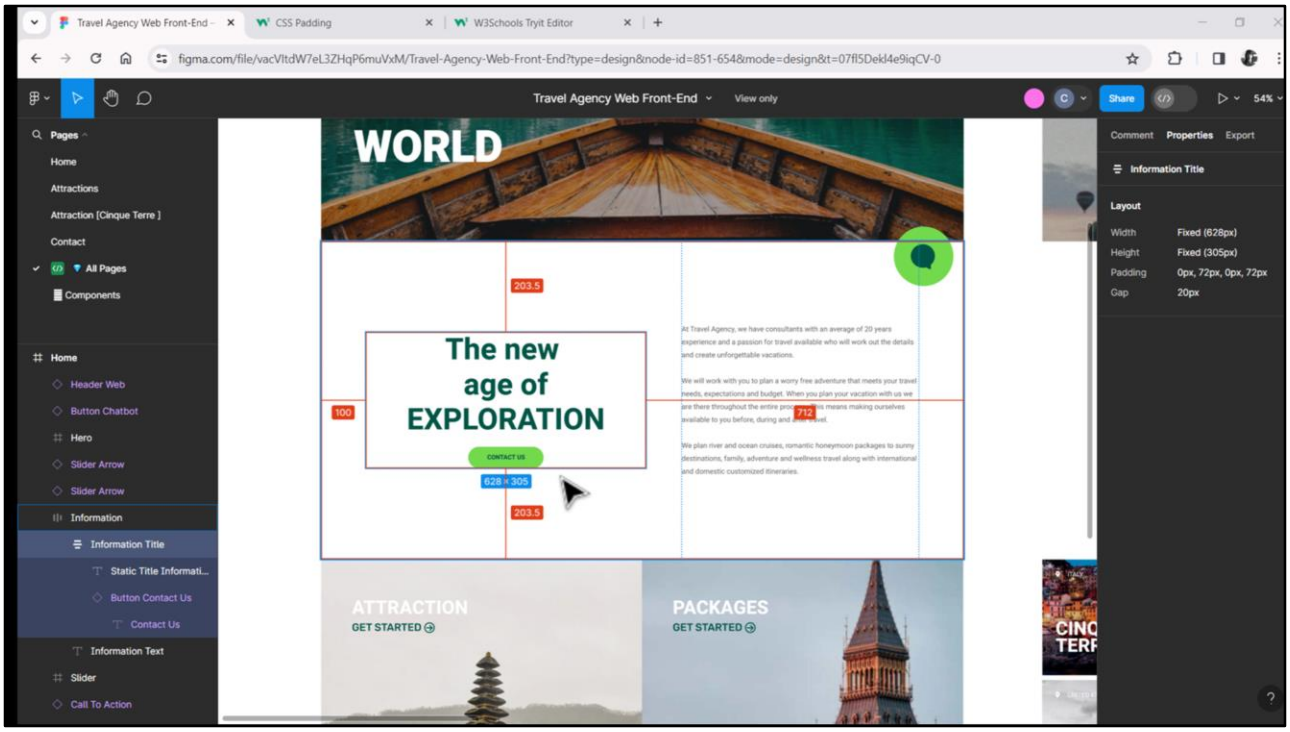
Ou seja, diremos que sua largura seja de 100%. De quê? Da largura restante de subtrair da largura total da tabela, as larguras fixas.



Portanto adicionamos coluna à esquerda desta, e depois, e uma após esta outra.

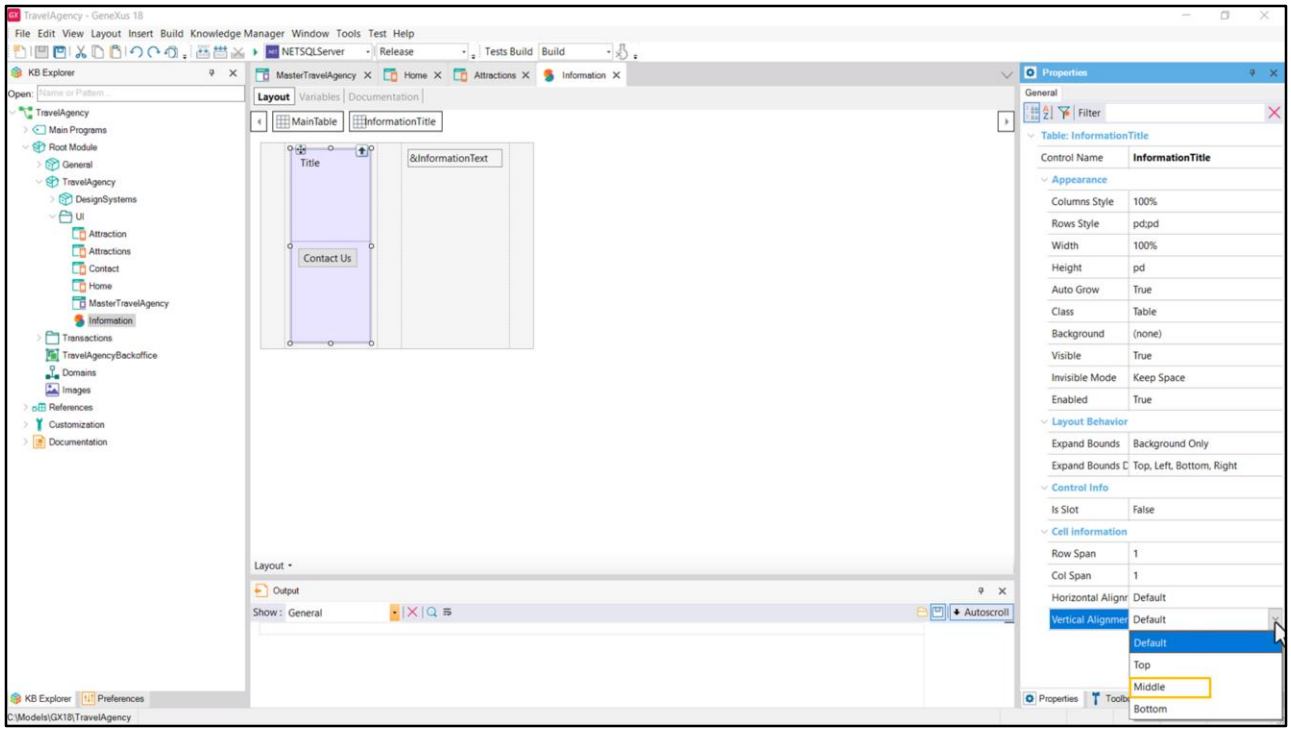


E agora definimos suas larguras.
Podemos fazer isso daqui... 100, para a segunda: 628, para a terceira 80 dips, para a quarta 100% e para a quinta 100 dips. Vamos gravar.

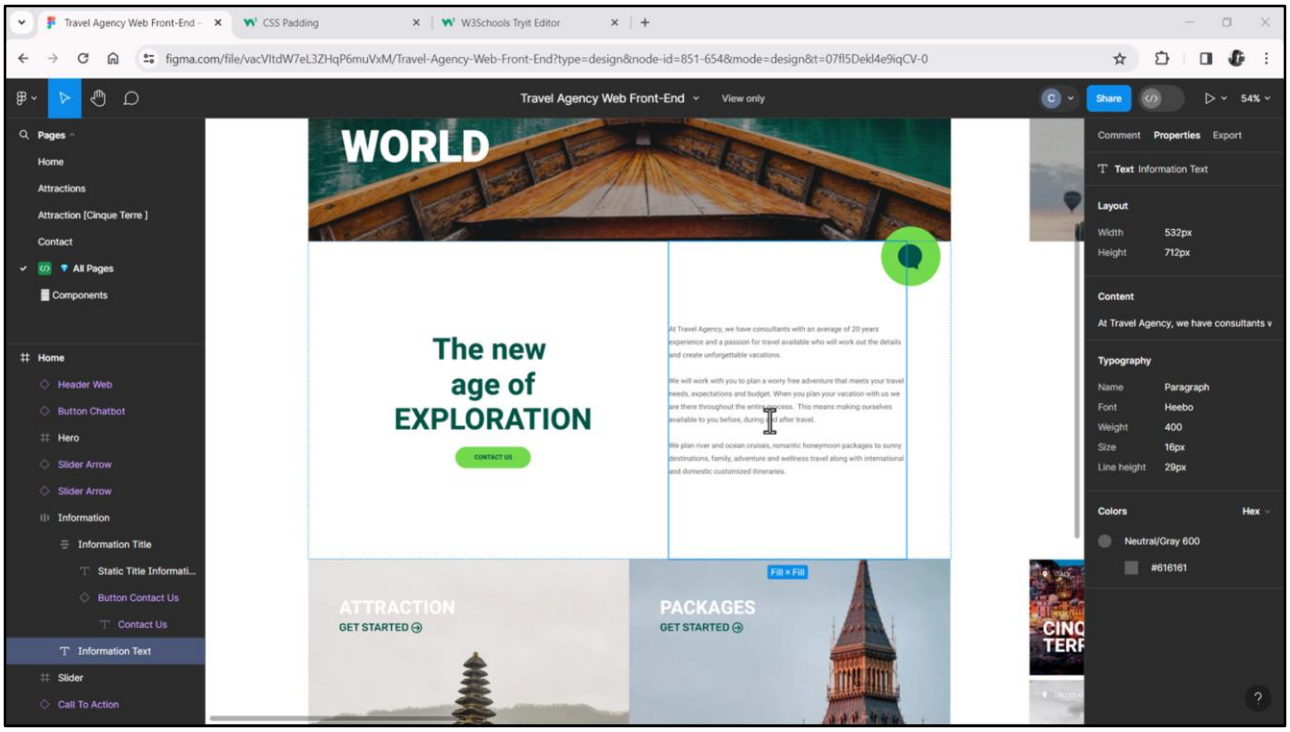


Agora, o que acontece com esses espaços, acima e abaixo? Temos que fazer o mesmo e colocar desta vez linhas vazias? Não necessariamente.

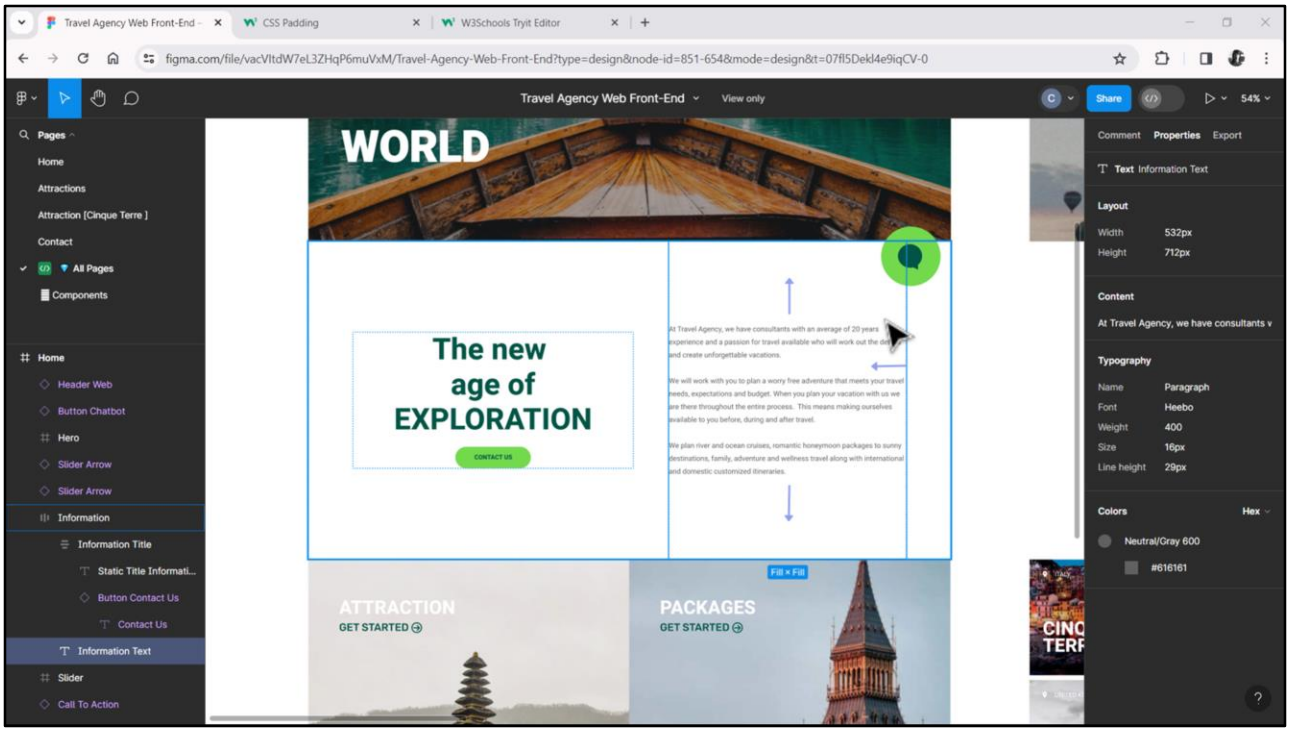
Nossa designer deixou indicado que este container terá uma altura fixa de 305 pixels. E claramente está centralizado verticalmente. Podemos ver isso com esses 203,5 pixels em cada lado da borda. Então bastaria fazer isso em GeneXus...



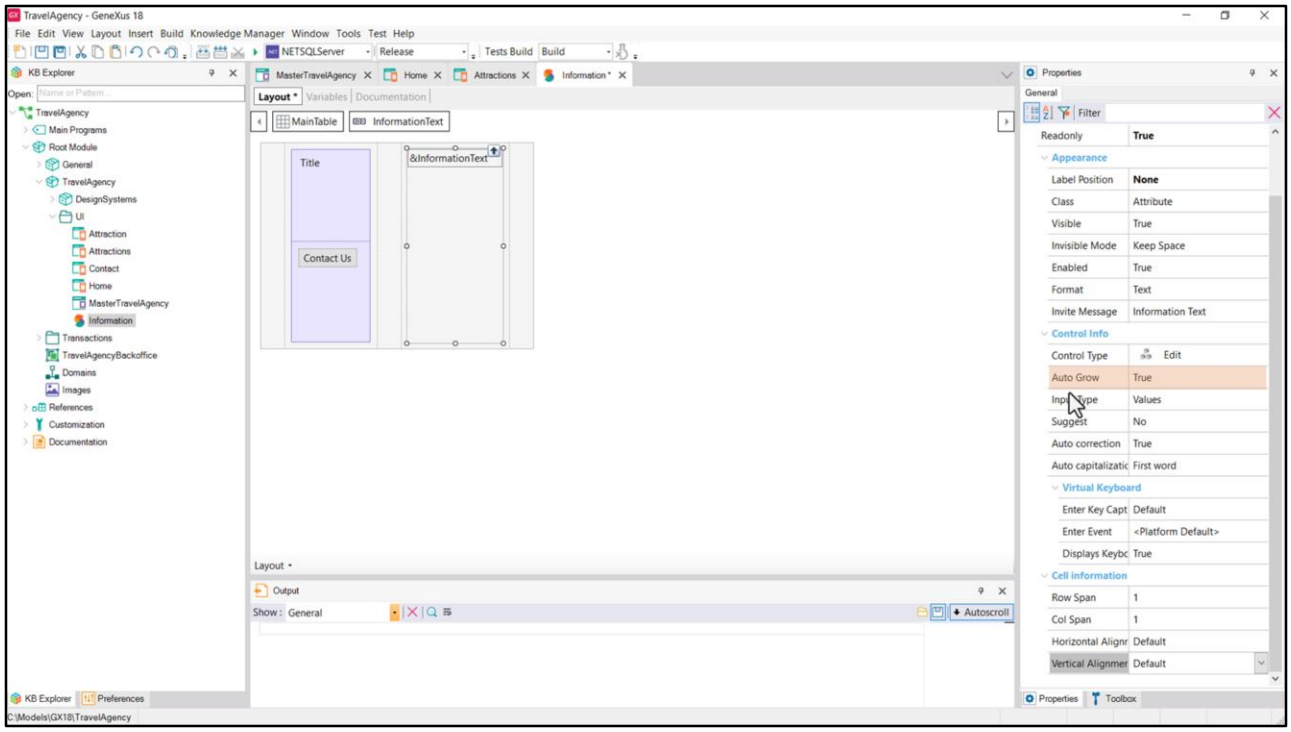
Editar a tabela e colocar como alinhamento vertical Middle.



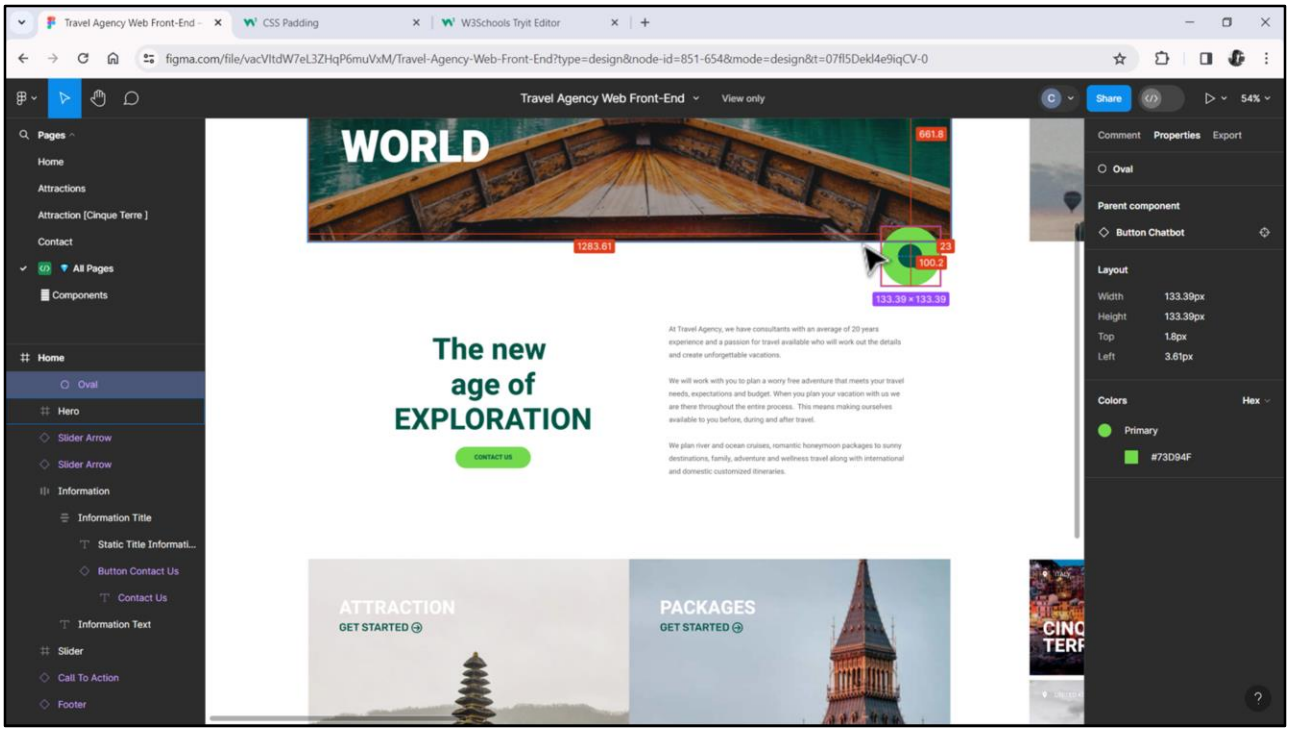
Por outro lado, este outro elemento, o texto, que também está centralizado verticalmente, não tem uma altura fixa, mas diz Fill.
Vai depender da quantidade de texto, o quanto vai ocupar. Sabemos que se o texto for esse, exatamente, ficará assim, mas se houver mais texto, haverá menos espaço acima e abaixo. Eventualmente, poderia chegar a não haver espaço algum.



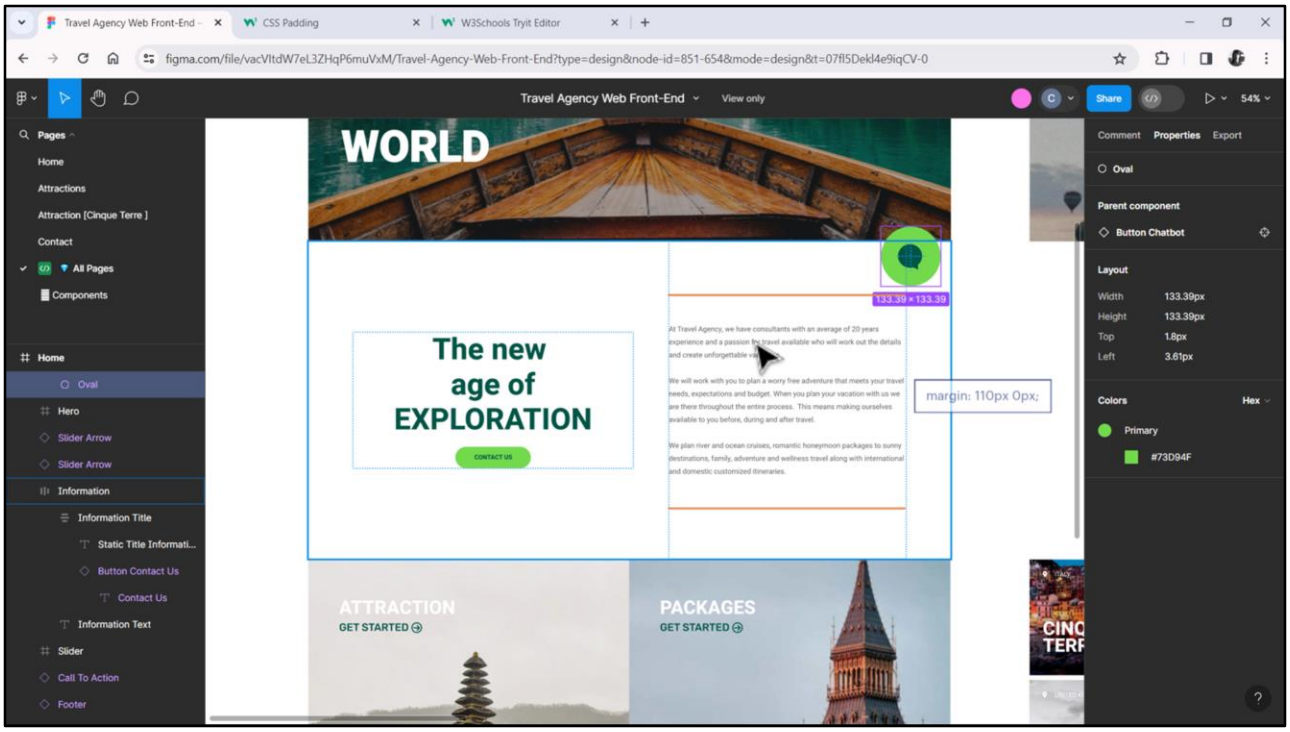
Da mesma forma, se a borda do navegador fosse encolhida, como a largura da coluna seria reduzida, o comprimento do texto seria aumentado também, e poderia acontecer a mesma coisa. Poderia até ser maior que a altura da tabela. Neste último caso, poderia realizar a ação que é o default no mundo web, que é permitir que cresçam elemento e container para que caiba todo o texto.



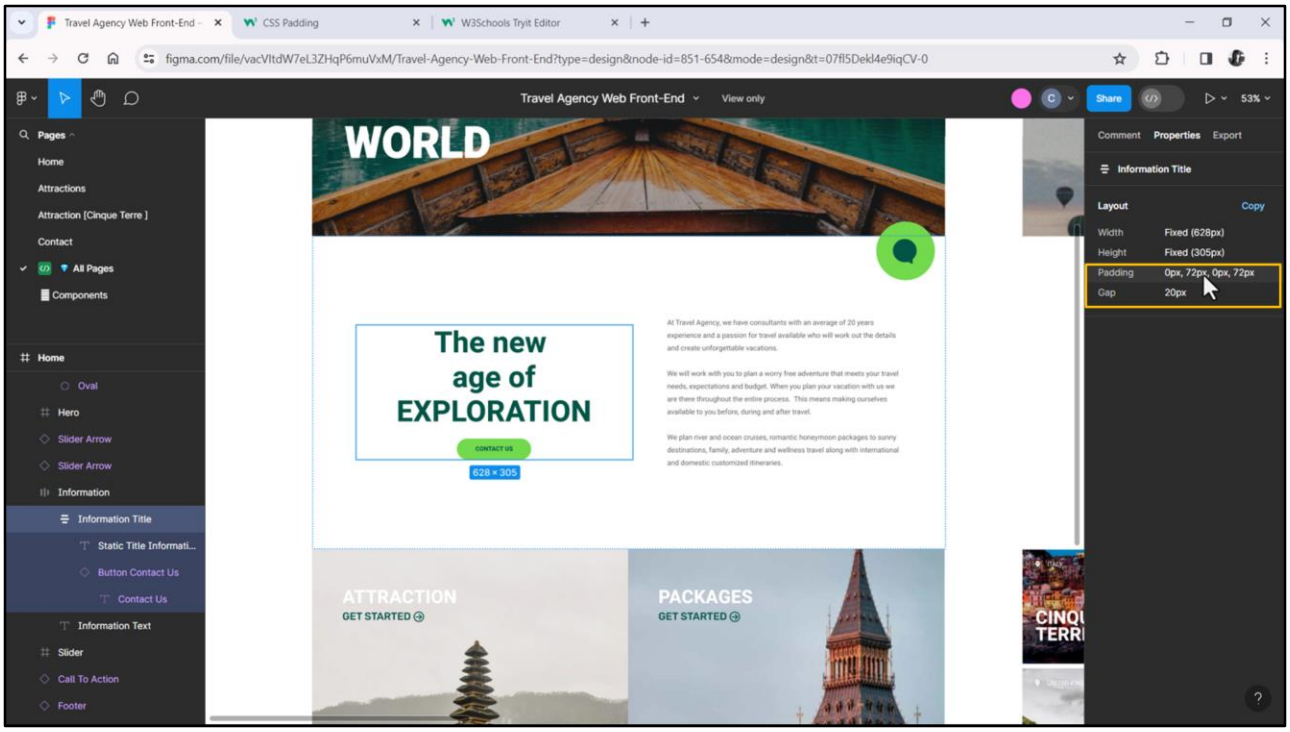
É por isso que deixei a propriedade Auto Grow como True, tanto para a variável quanto para a tabela que a contém (se a tabela não tiver Auto Grow como True, não será levado em consideração o valor da propriedade Auto Grow para os elementos contidos nessa tabela). Já que estamos aqui, vamos alinhar verticalmente no meio a variável.



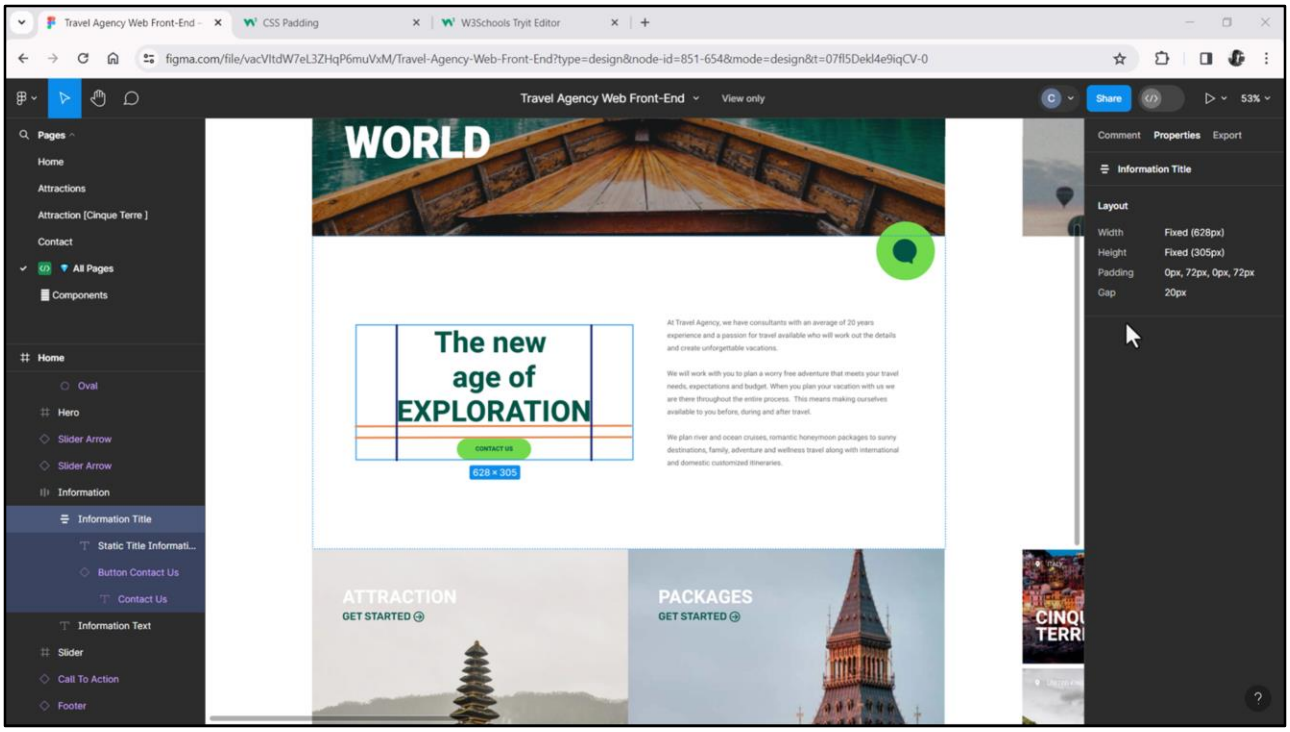
Agora, se deixarmos isto assim, não temos nenhuma garantia de que em qualquer situação fique um espaço acima e abaixo.
O que eu faria seria falar com Chechu e perguntar o que pensa sobre isso, já que isso não está modelado em seu desenho. Na verdade, se observarmos o container de onde partimos, tem 0 de padding acima e abaixo, e podemos ver claramente aqui que esse elemento está se expandindo para as bordas. Certamente não queremos que o texto possa se expandir além da borda inferior desta imagem do chatbot. Vemos seu comprimento de 133,39 e se compararmos a distância com a borda do Header vemos esses 100,2 pixels, então não seria absurdo propor um espaçamento de 110 acima e 110 abaixo para que sejam simétricos.



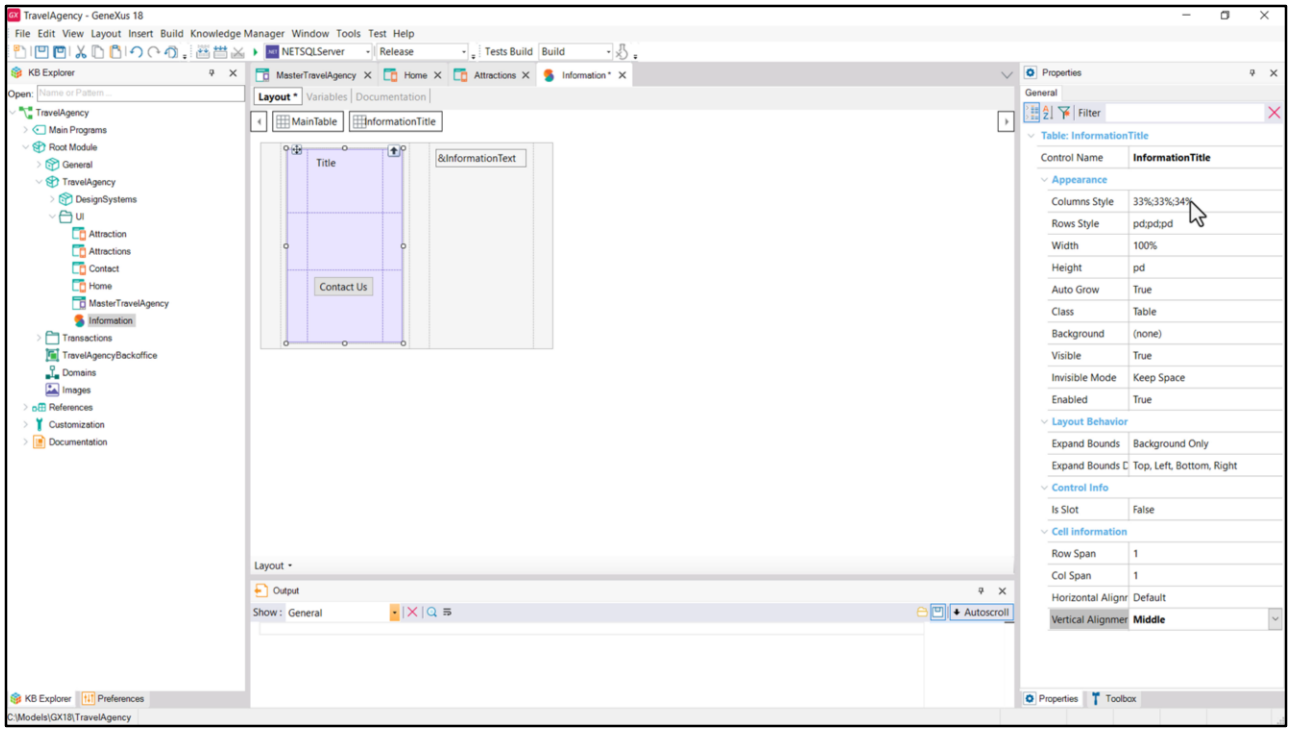
Isto, claro, podemos fazer da mesma maneira que fizemos para as colunas, ou seja, adicionando duas linhas vazias, uma acima e outra abaixo. Mas temos outra alternativa, entre várias, que será através de dar uma margem superior e inferior a este elemento, e vamos deixar para ver mais tarde, quando já passamos pelo Design System Object.



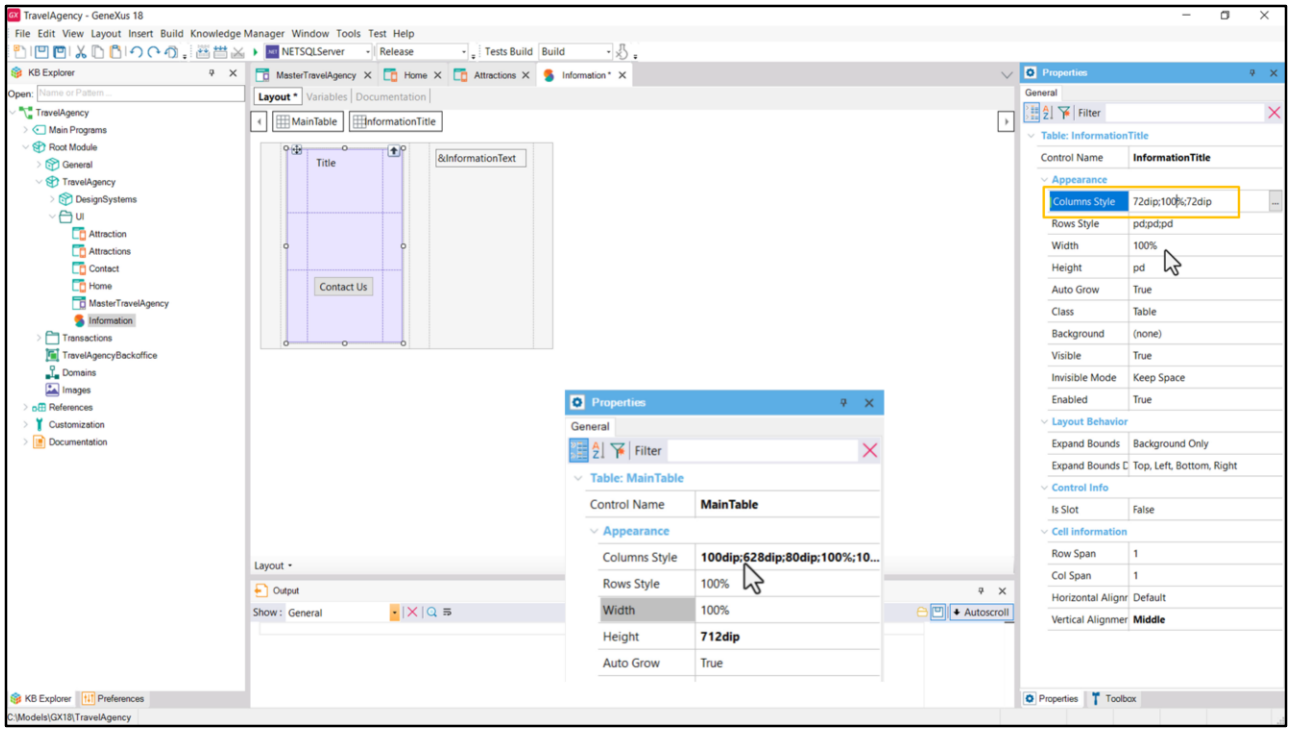
Agora sim, vamos analisar o container da esquerda. Este é muito parecido com o anterior, só que aqui, em vez dos elementos estarem ordenados nesta linha horizontal, estão verticalmente: acima o texto e abaixo o botão. E aqui vemos o padding desta forma: 72 pixels da direita e 72 pixels da esquerda. E 0 de cima e de baixo. E qual é o espaço que separa um elemento do outro? Esses 20 pixels indicados aqui.



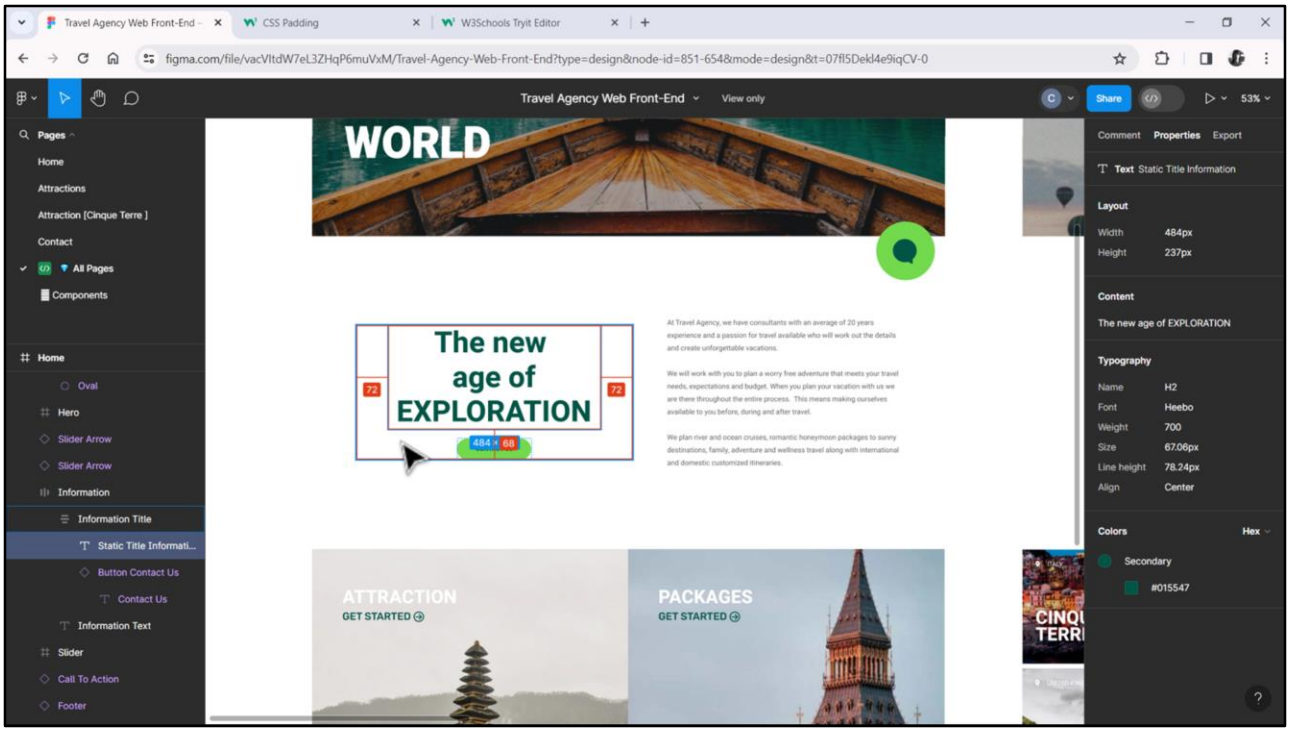
Então, com essa perspectiva, você poderia pensar em adicionar uma linha vazia no meio e duas colunas vazias nas laterais.



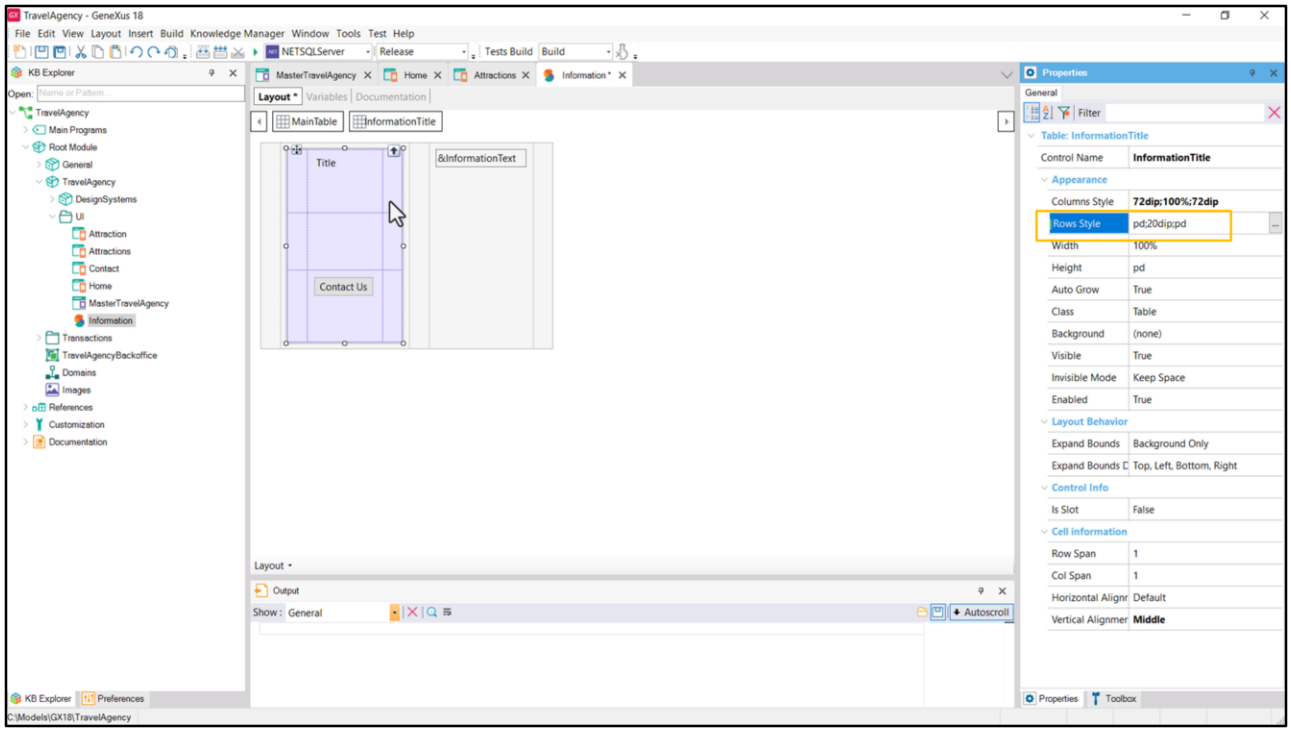
Fazendo isso, vemos que as propriedades foram atualizadas, e agora as colunas adotaram esses valores em porcentagens, para serem equidistribuídas, que é como o default.



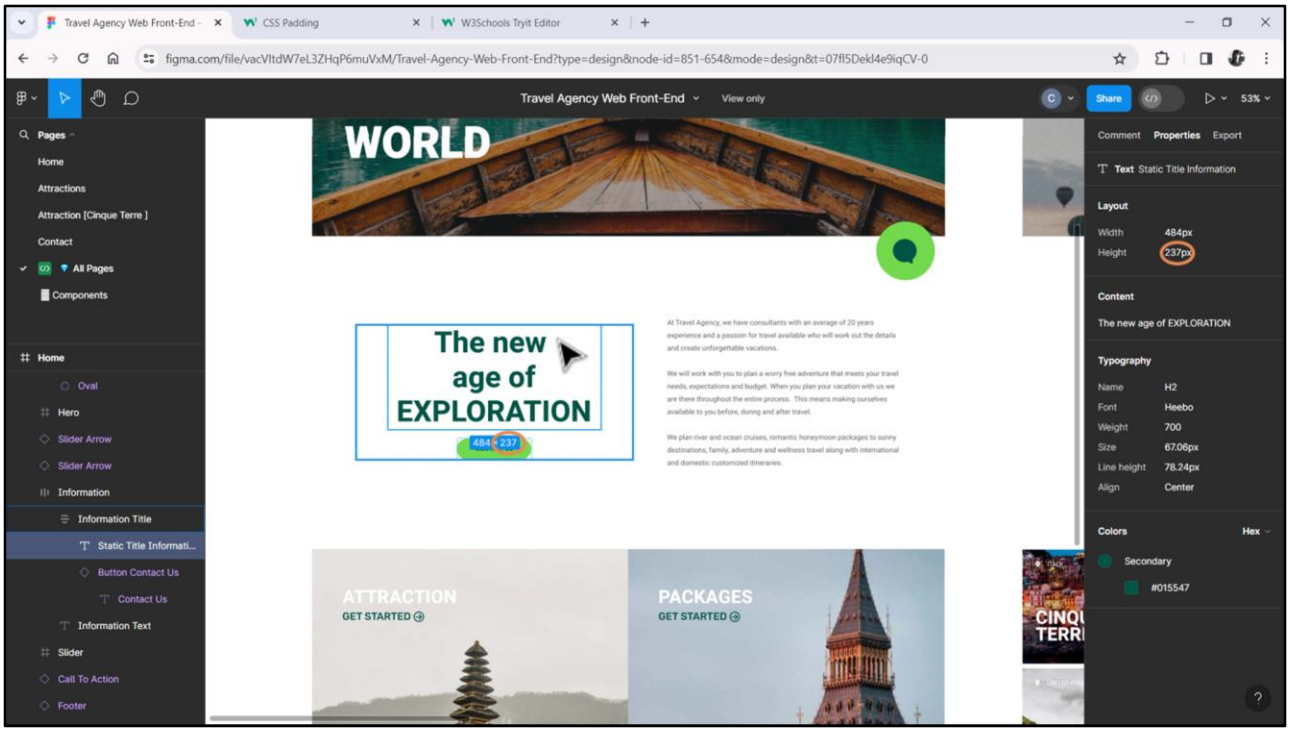
Então o que vamos colocar aqui são 72 dips na da esquerda e 72 dips na da direita. E o que acontece com a do meio? Vou colocar para ela 100%... da Width da tabela, que corresponderá então à largura da célula, da coluna neste caso, na qual se encontra essa tabela. É a segunda coluna, portanto corresponderá a esses 628 dips. Portanto, a largura que assumirá a segunda coluna será a resultante de subtrair desses 628 dips os valores das duas colunas, da esquerda e direita.



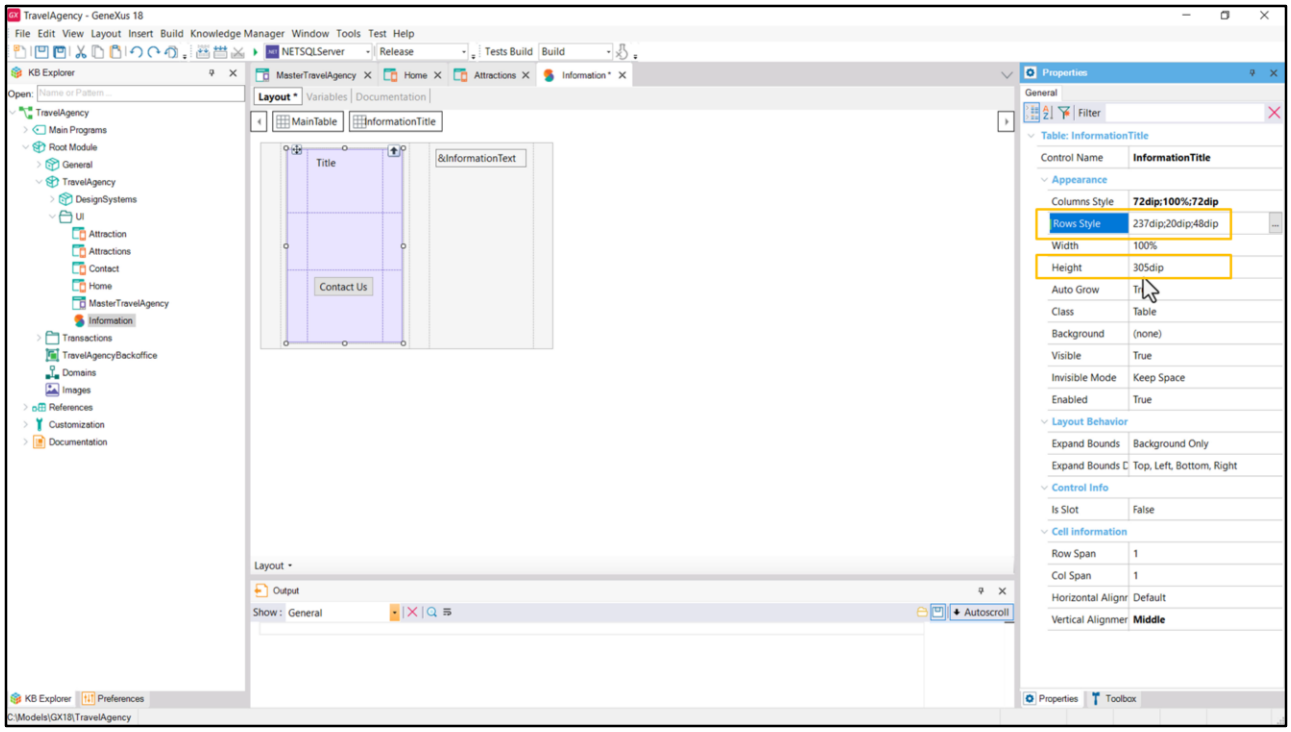
E isto vemos claramente aqui em Figma... 628... 72, 72, e o que restará então da largura dessa coluna será 484.



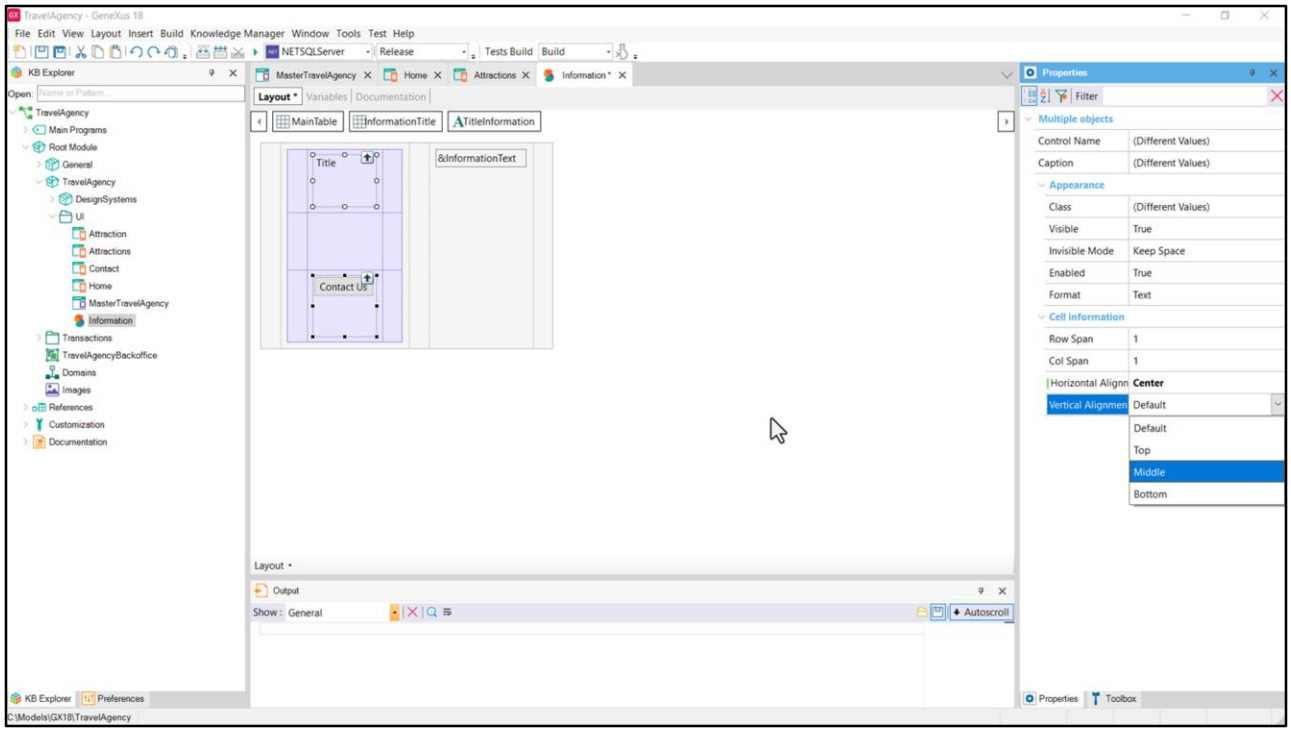
Bem, e o que acontece com a altura das linhas? Sabemos que a do meio é uma linha de espaçamento, que será de 20 dips. Agora temos que ver quanto seria a altura da linha... da primeira... e quanto seria a altura da linha que conterá o botão.



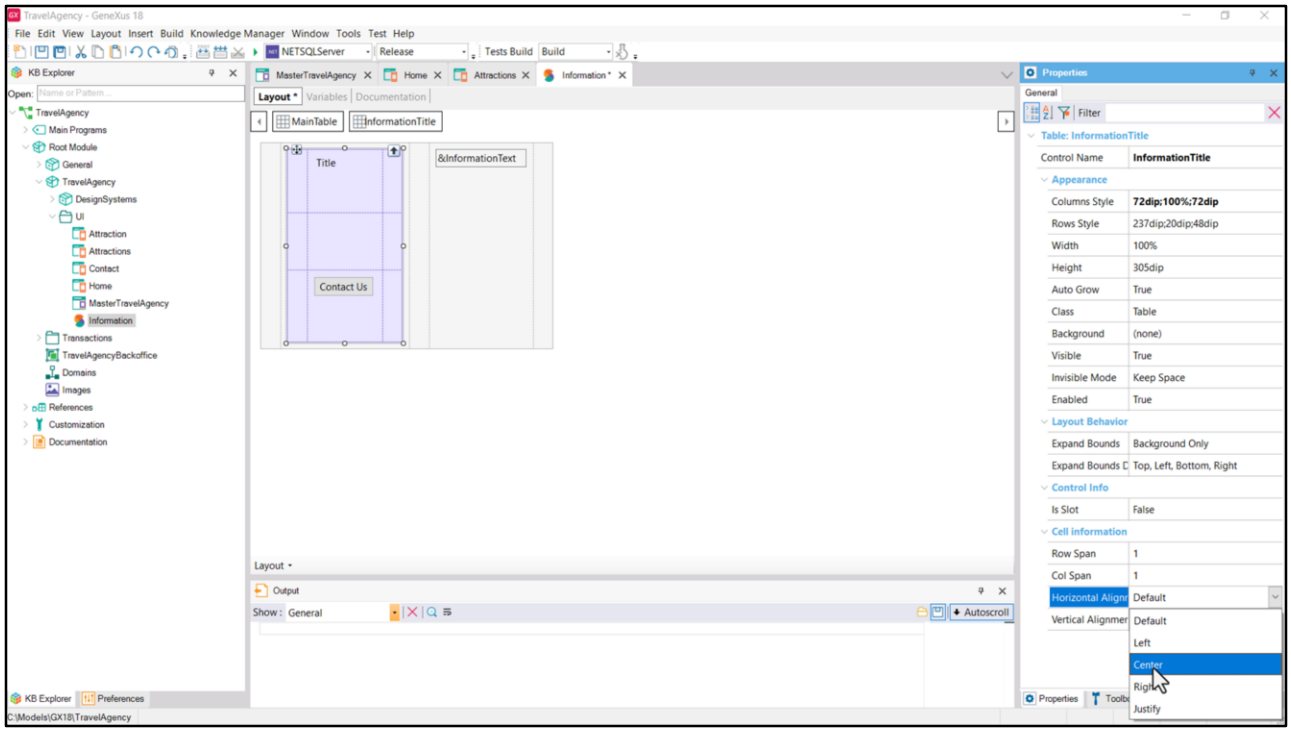
Bom, a do texto, aqui vemos, será de 237 pixels, e a do botão de 48.



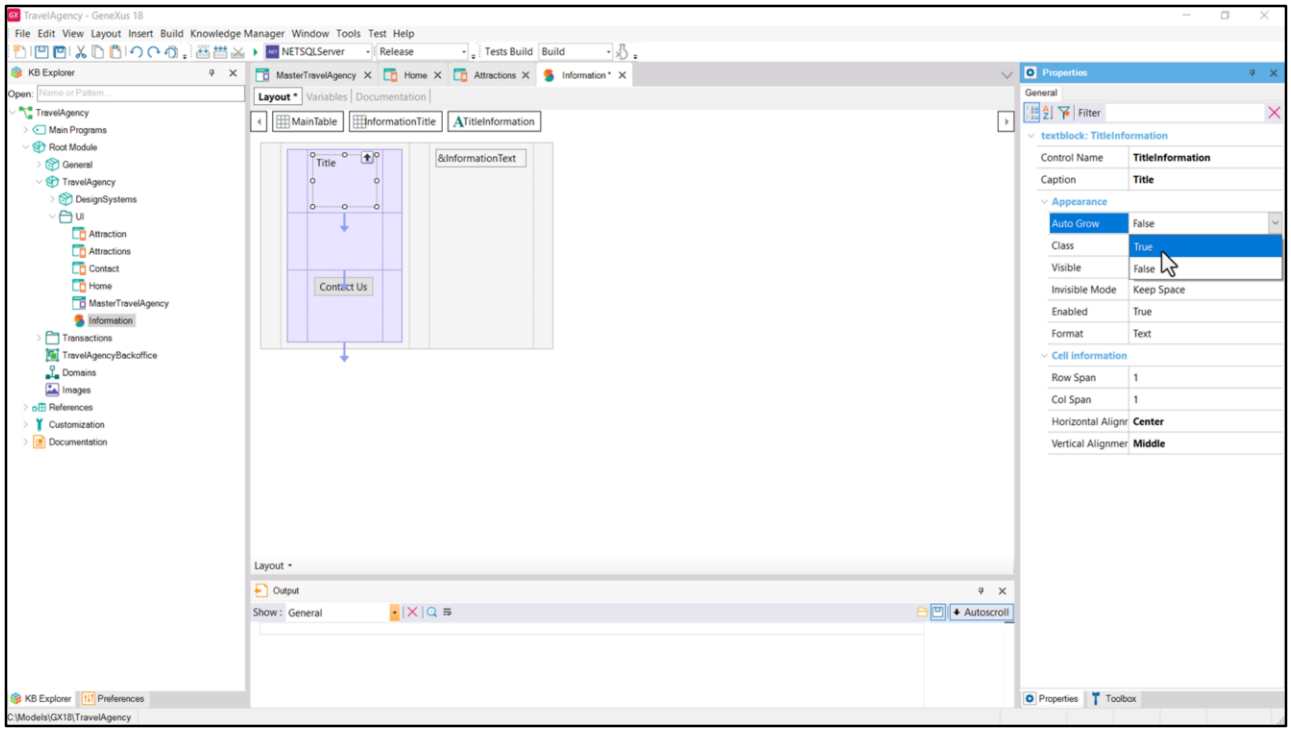
Então... 237... e observemos como ao ter feito isso, como todas as linhas são de altura fixa, já modificou automaticamente a propriedade Height somando esses valores.



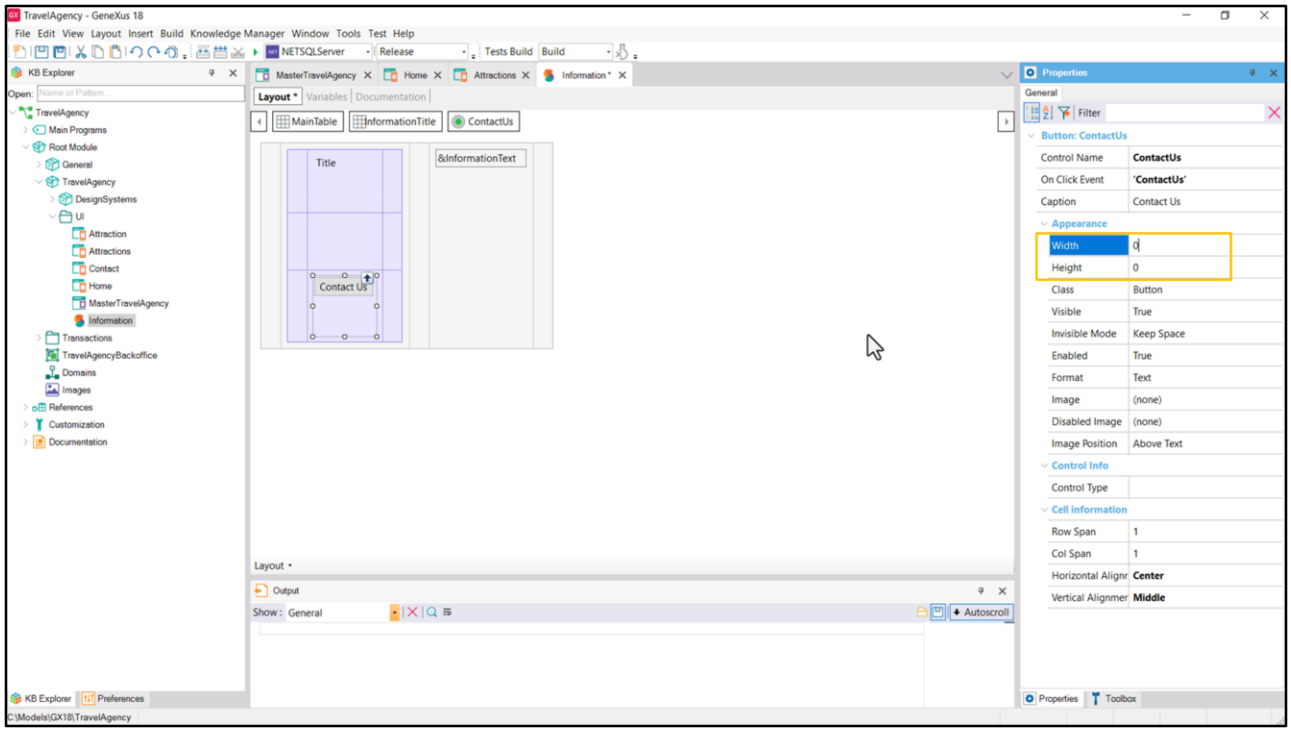
Agora, antes de continuar, vou centralizar esses dois elementos... vou centralizá-los horizontalmente e também vou centralizá-los verticalmente.



E vou fazer o mesmo para a tabela, embora não seja necessário neste caso... vou centralizá-la horizontalmente. Por que não é necessário? Porque vai ocupar todo o tamanho da célula e estou centralizando tudo de dentro.

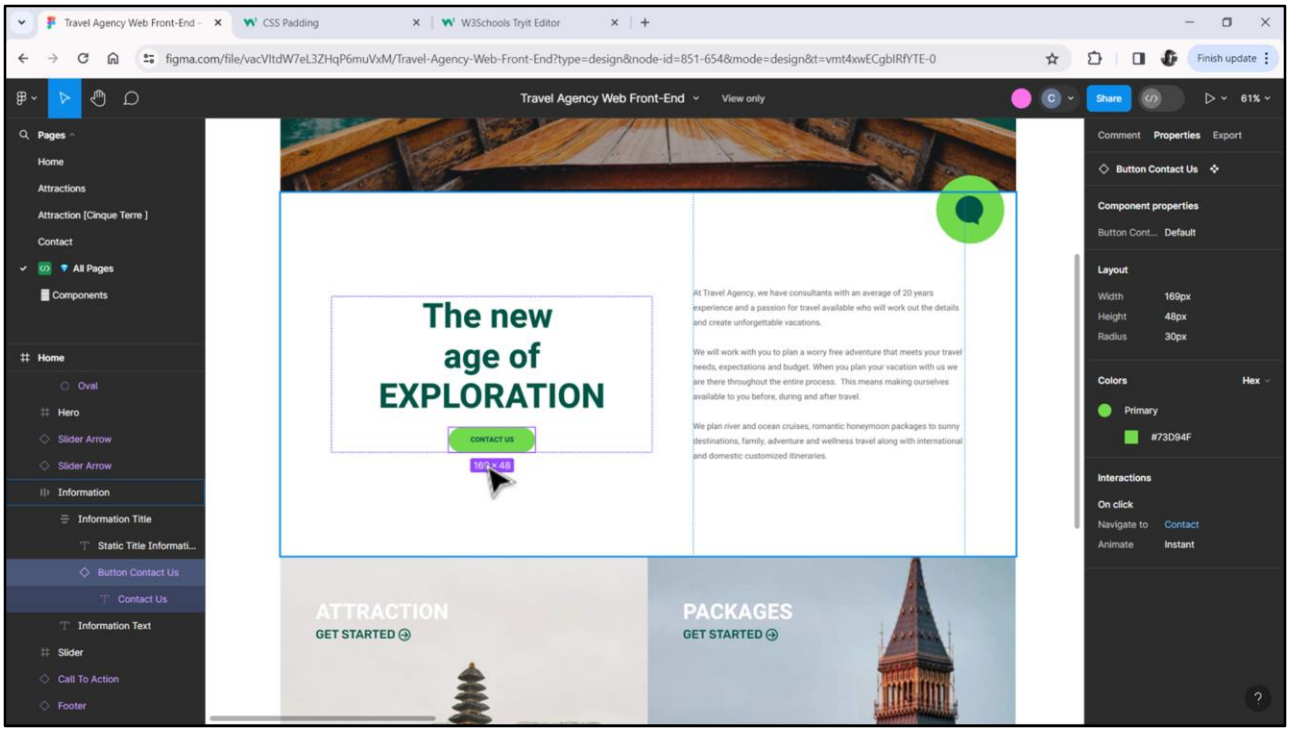


Bom, com o que fizemos, sabemos que dependendo do conteúdo do Caption do TextBlock, até onde chegará. Se a quantidade de caracteres faz com que sua largura seja menor que a da coluna, terá espaço equidistante das laterais, pois o centralizamos (e também de cima e de baixo). Se tiver mais caracteres, então será feito wrap para não transbordar e o texto terá mais de uma linha. E o que aconteceria se não atingisse a altura da célula? Como vemos, neste caso o TextBlock tem como default False para Auto Grow, porque é o default para as plataformas nativas, portanto será truncado. Se o que queremos é que, se ultrapasse a altura, então a linha seja empurrada para baixo e, portanto, as outras duas também, então mudamos para True, que geralmente é o comportamento esperado em Web.

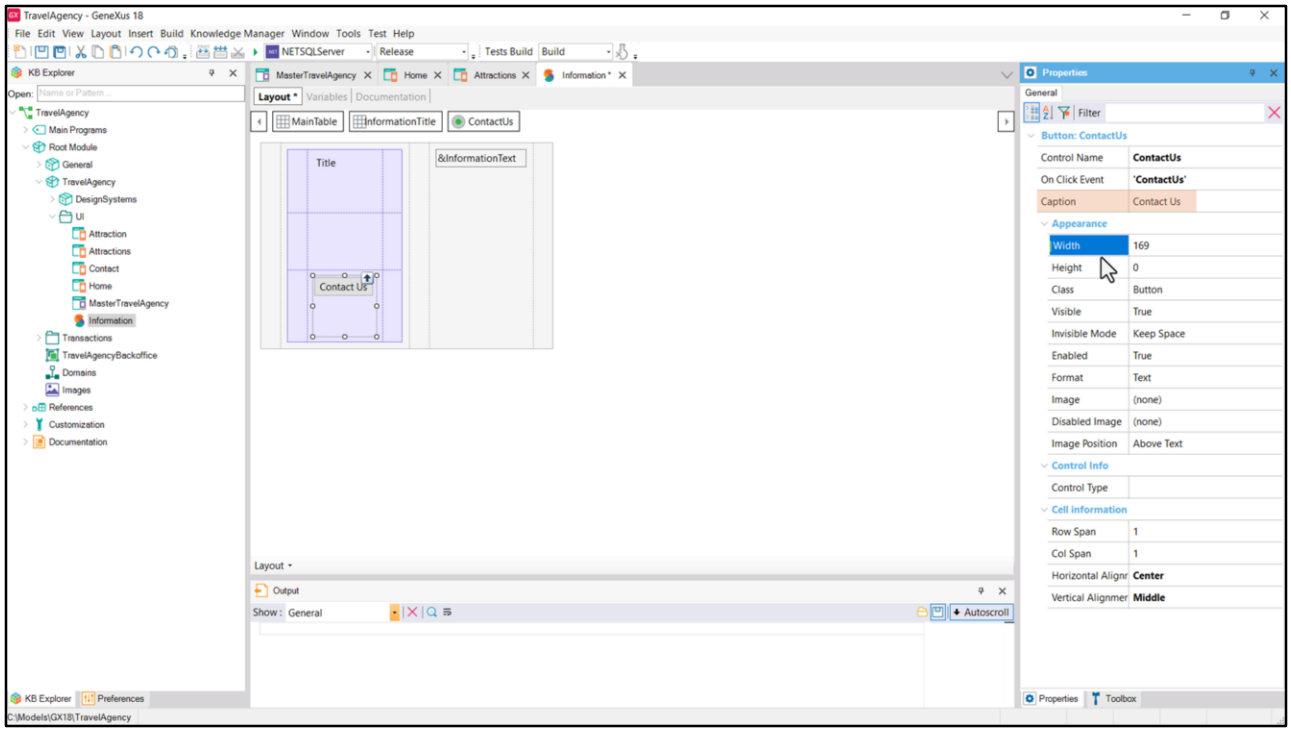


E o que acontece com o botão? Por default, ele se expandirá até ocupar todo o espaço do container (embora possa ser um pouco contraintuitivo dizer esses zeros em vez de um mais esperado 100%)

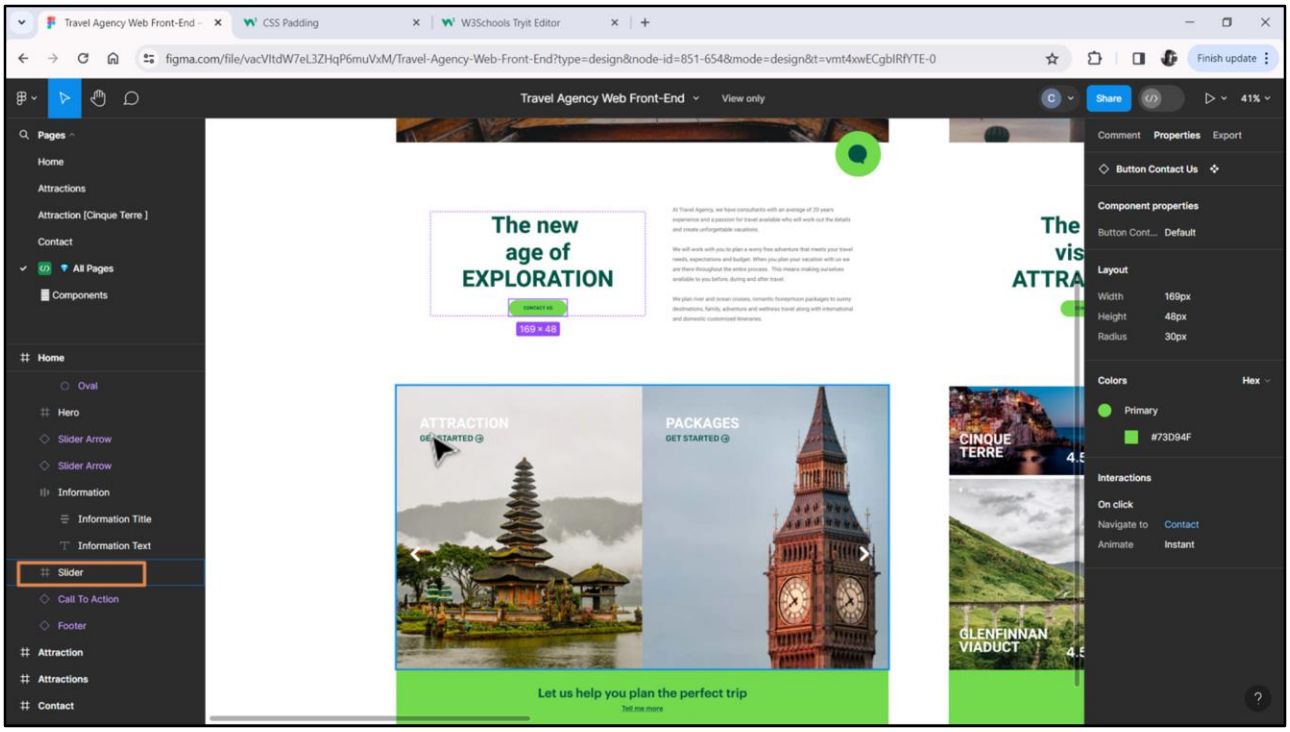
Para a altura, que ocupe o mesmo espaço que a altura desta célula está perfeito, porque indicamos que era de 48 dips justamente porque era a altura do botão. Mas para a largura não está correto.



Se formos ver em Figma, a largura é menor, é de 169 pixels.

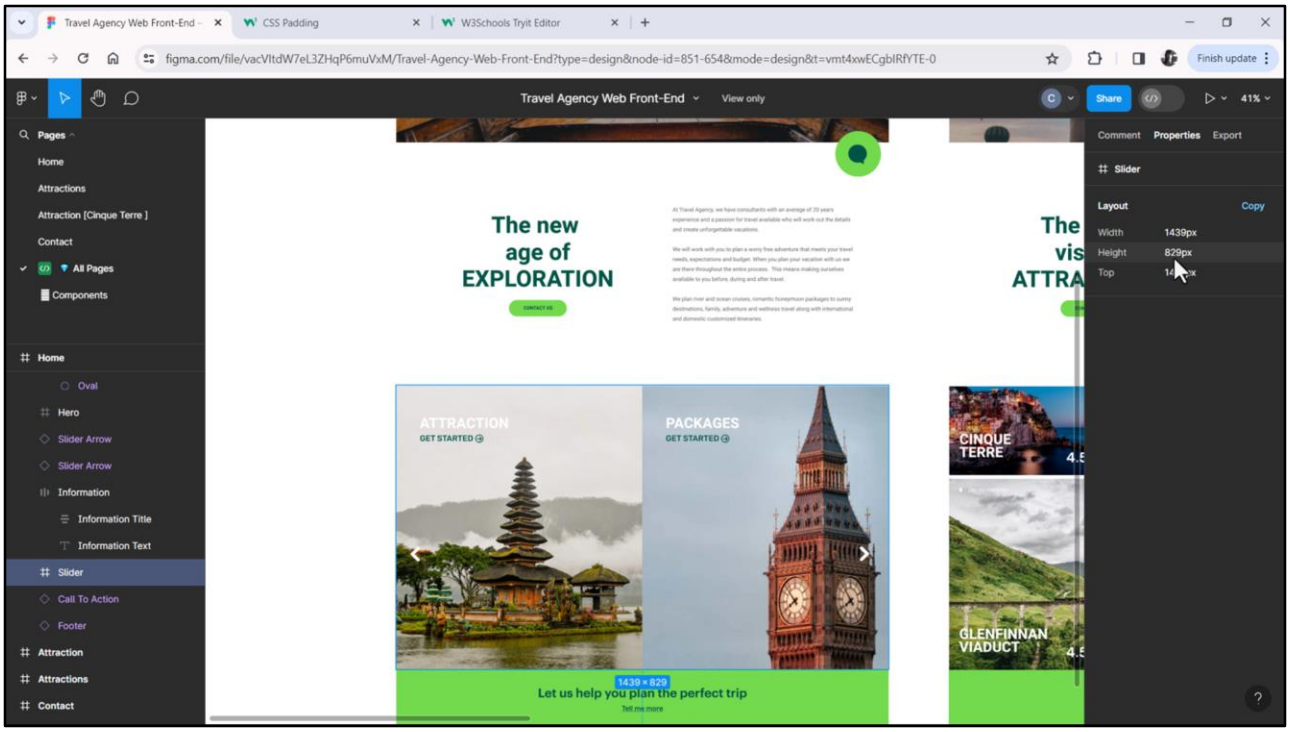


Depois veremos que essa seria a largura mínima, mas que se o “caption” do botão ultrapassasse esse tamanho, o botão deveria se expandir para os lados, sem ultrapassar a largura do container, é claro. Mas veremos isso mais tarde, por enquanto vamos deixar assim.

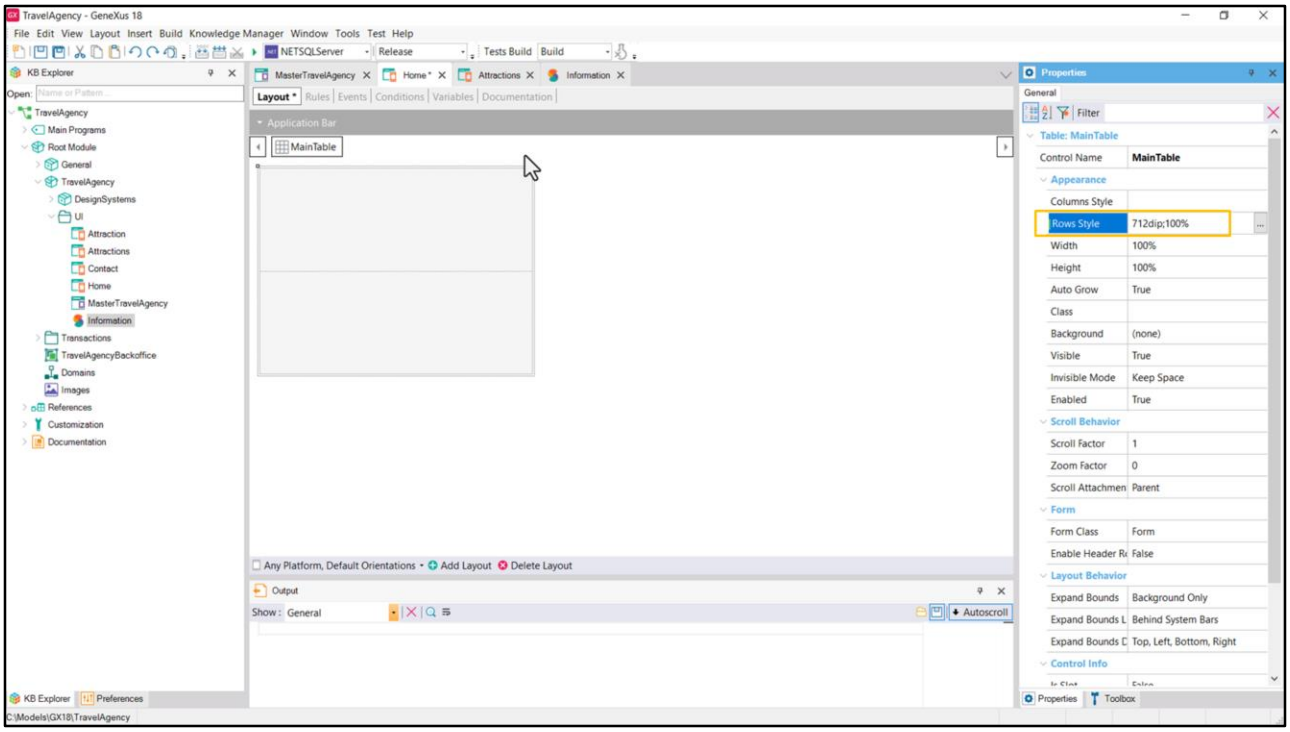


Antes de passar a definir o estilo, quero ver em execução como isso está ficando. Para fazer isso, terei que instanciar o Stencil no panel Home, primeiro, e depois no de Attractions, que por enquanto temos vazios.

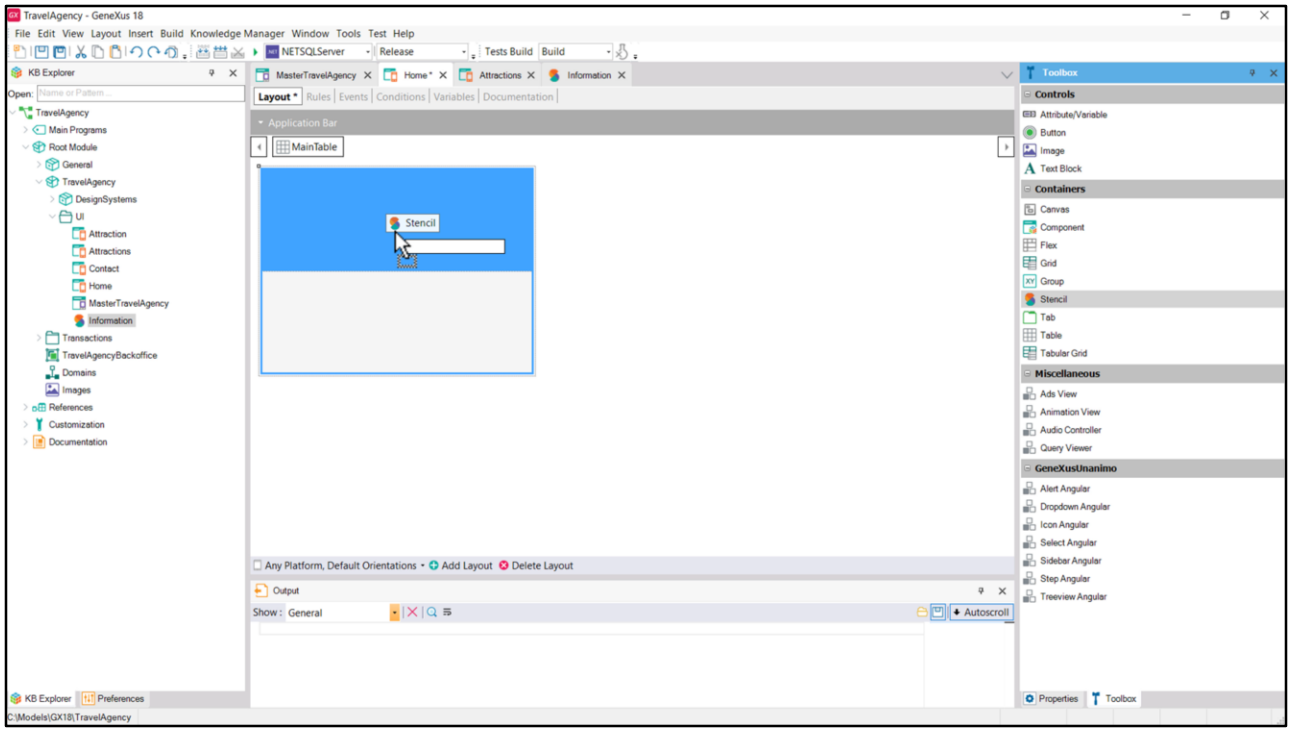
O Home terá além deste elemento, este outro, que aqui Chechu chamou de Slider, e é um frame com subelementos.



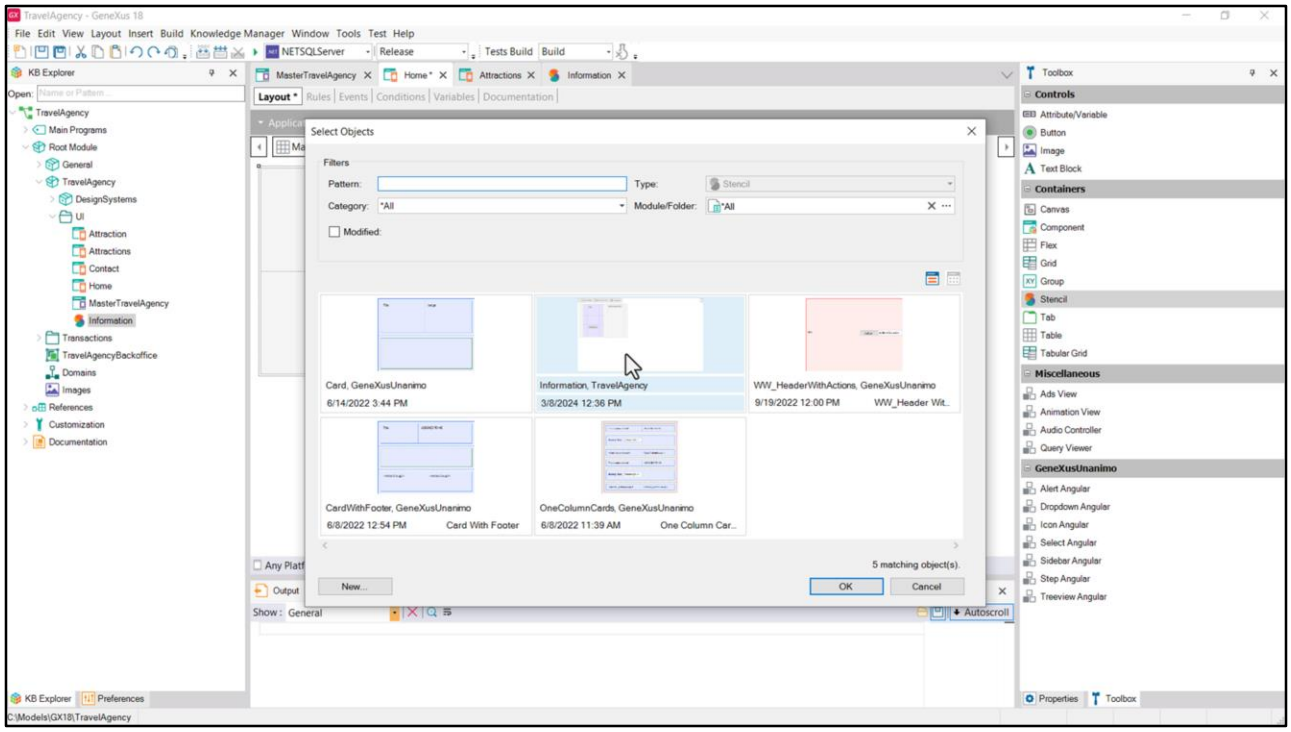
Sua altura será de 829 pixels. E a de Information era de 712.



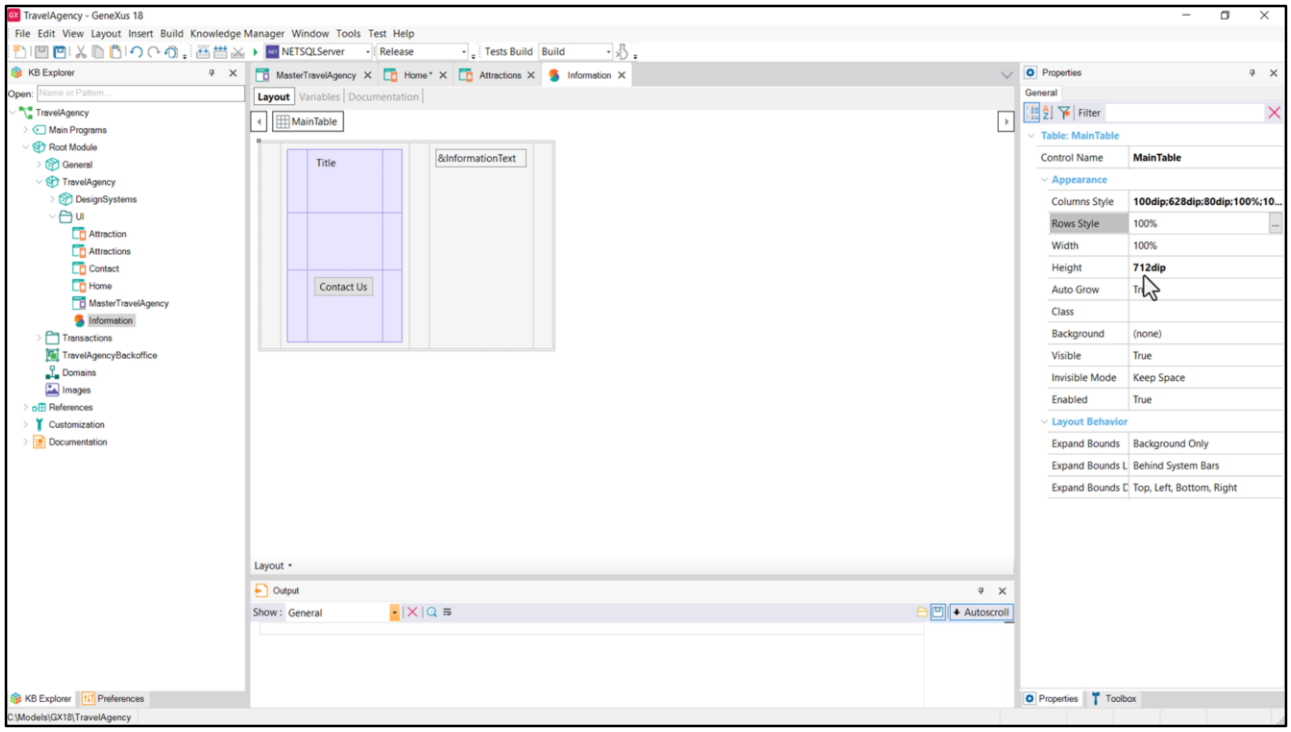
Então, para a tabela main, adiciono uma linha... a primeira das duas linhas que tem por enquanto vai ser de 712 dips e a segunda que poderia ser de 829, vou deixar em 100%. Depois analisaremos isso. Os 100% restantes de, obviamente, a altura da tabela Main.



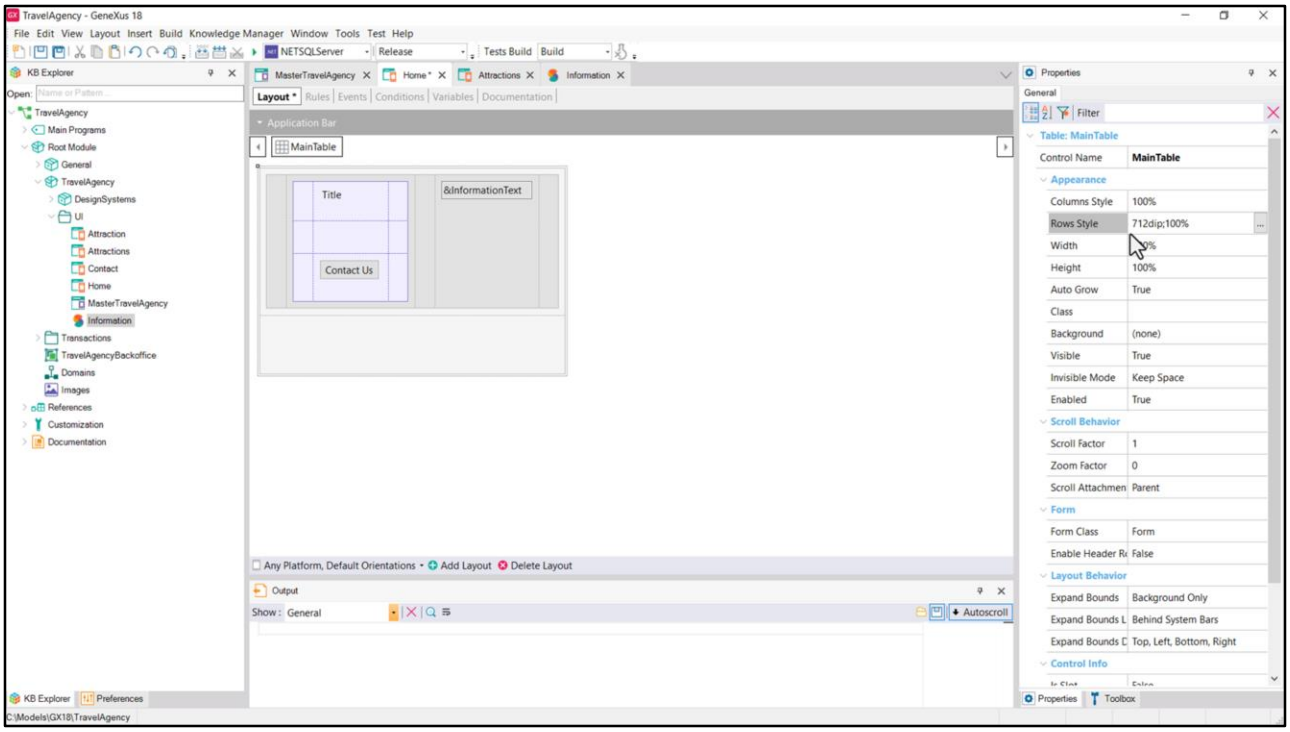
E a seguir o que vou fazer é arrastar um controle do tipo Stencil para dentro da primeira dessas linhas…



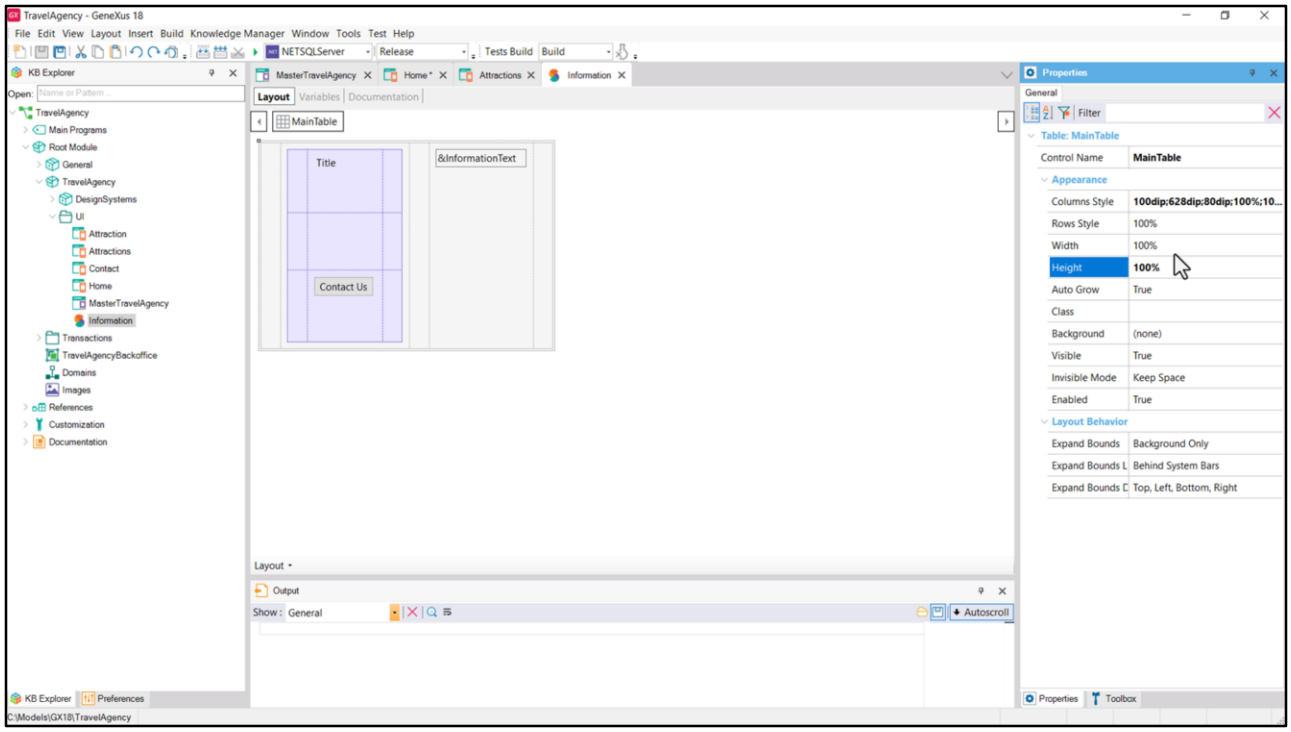
… e escolher o nosso, Information.



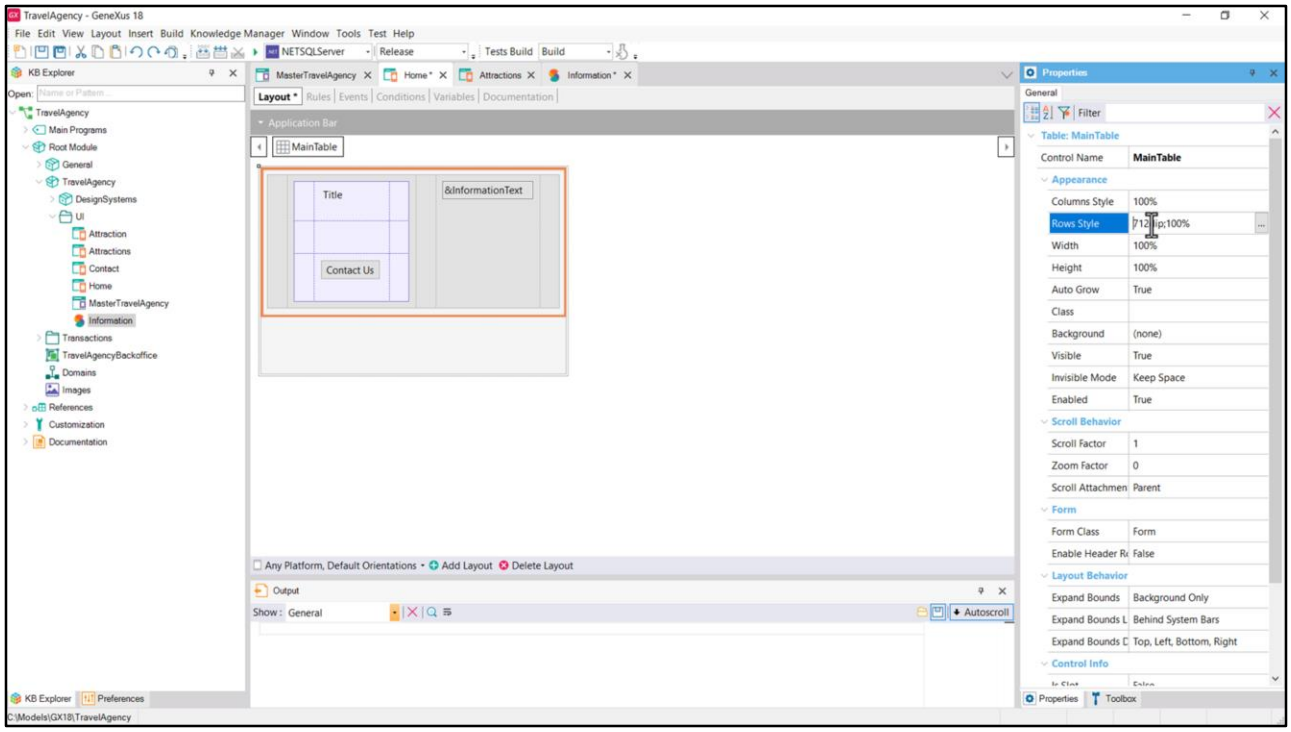
Se agora Chechu nos avisasse que mudou a altura de 712 pixels ou dips para 800, por exemplo, o que teríamos que fazer? Teríamos que vir mudar isso aqui…



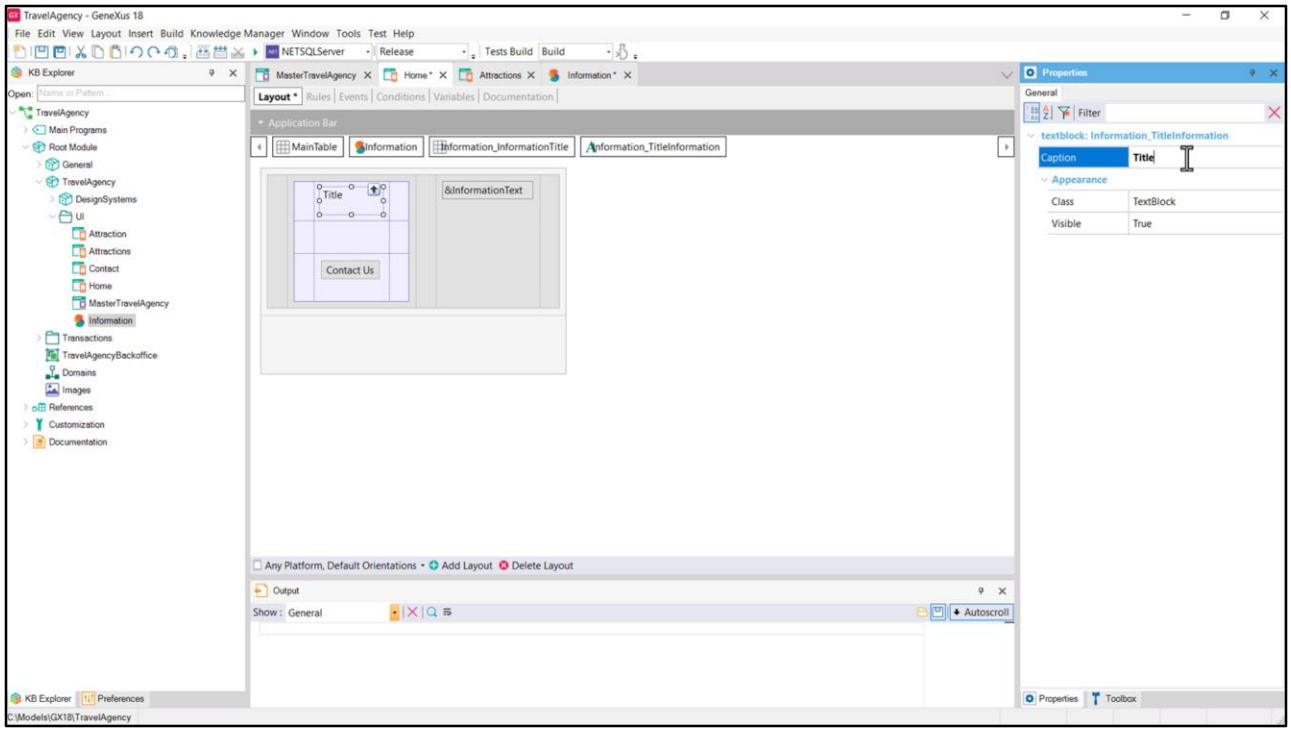
E depois vir aqui também, para mudar isso no nível da altura da linha. E se depois colocarmos esse mesmo Stencil no panel de atrações, a mesma coisa.



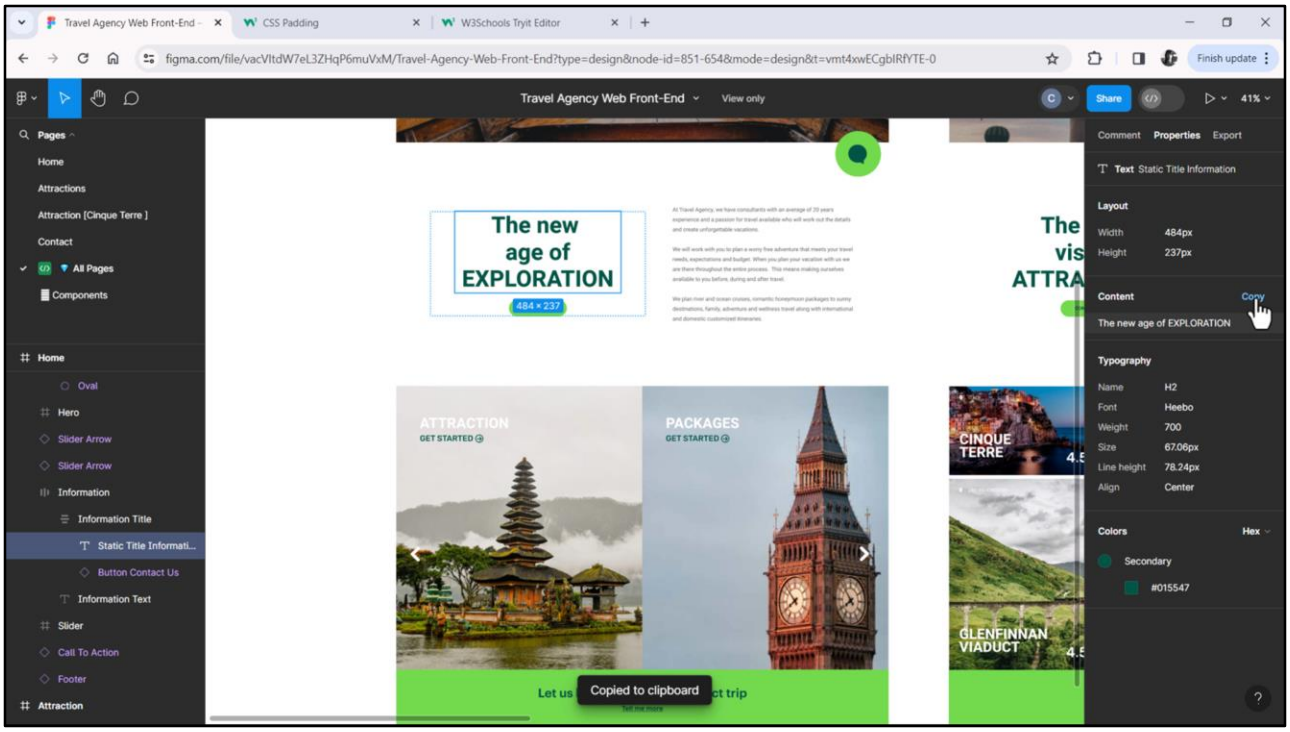
Então para evitar essas redundâncias vamos optar por deixar esses 712 dips em um lado só, e será no nível da linha do container desse Stencil. Então aqui vamos mudar isso para 100%. De modo que agora, se eu tiver que mudar para 800, isto estará certo, estamos dizendo que a altura da tabela será de 100% do container...



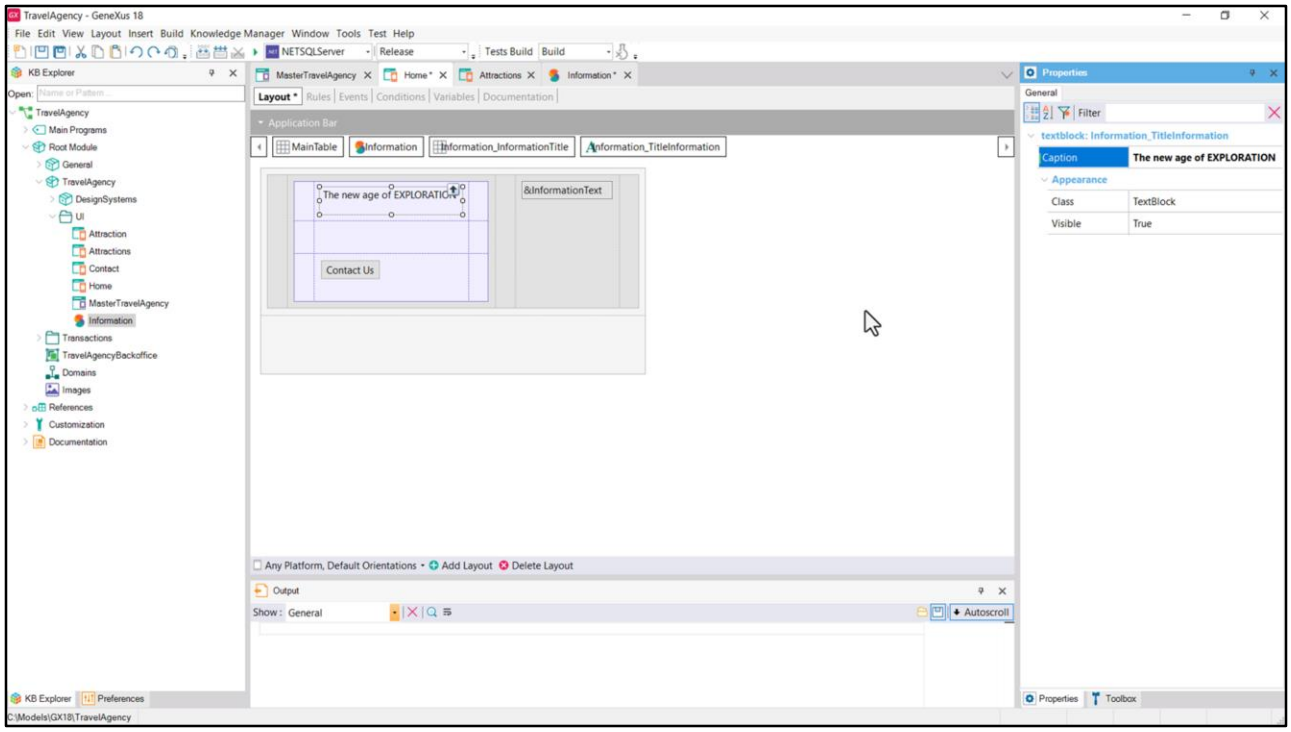
…e o container neste caso é esta célula que corresponde à primeira linha da tabela. Então aqui eu mudaria para 800 dips. Claro que eu teria que fazer o mesmo no panel de atrações.



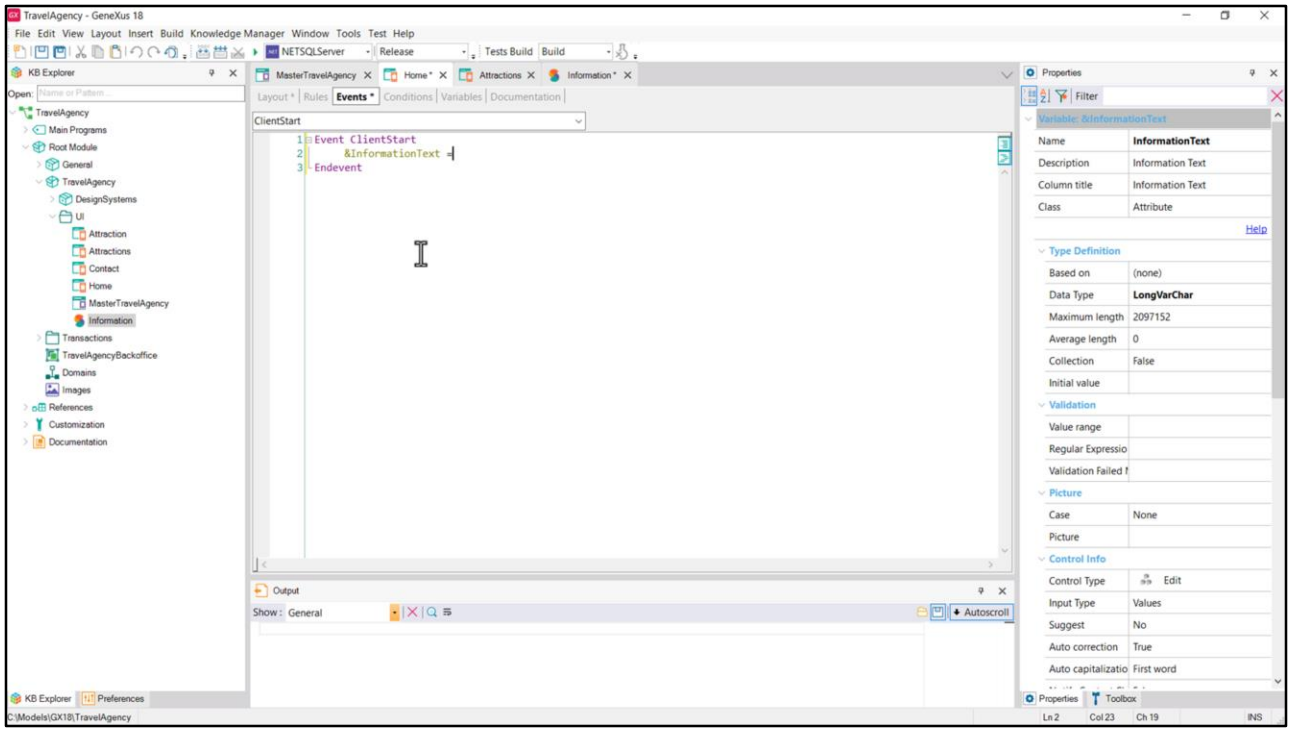
O que nos resta para poder executar? Atribuir valor ao TextBlock e à variável. Poderia atribuir ambos os conteúdos no evento Start do cliente, ou, por exemplo, ao TextBlock poderia atribuir diretamente aqui, alterando este Title…



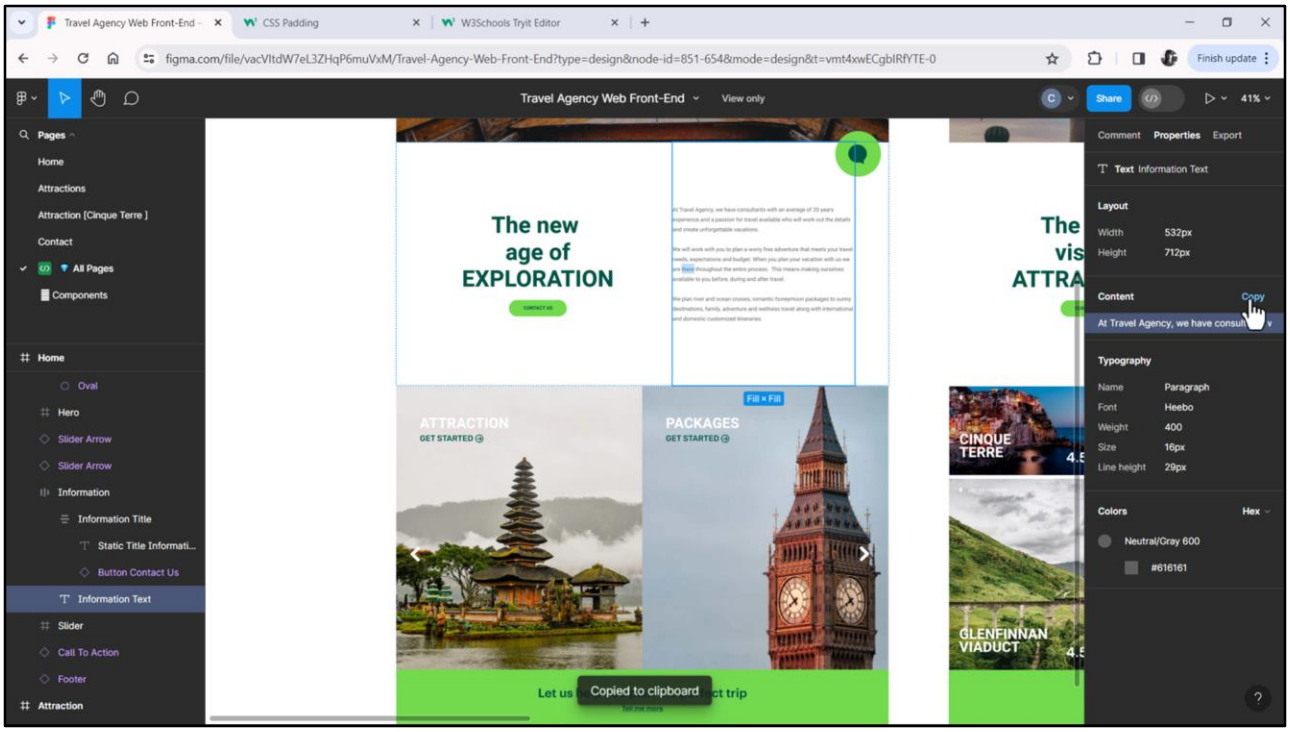
…para este conteúdo que posso copiá-lo diretamente do arquivo em Figma… veem que fica copiado no clipboard…



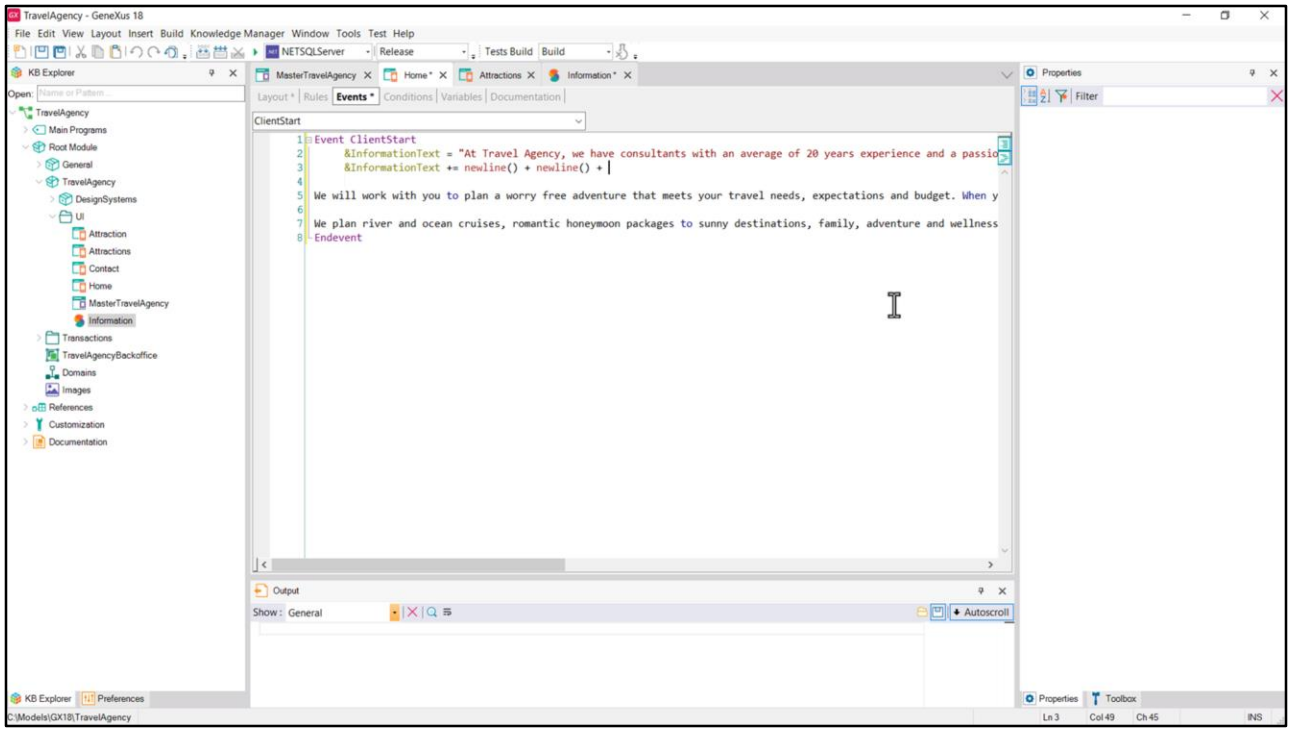
…e então eu venho aqui e colo diretamente.



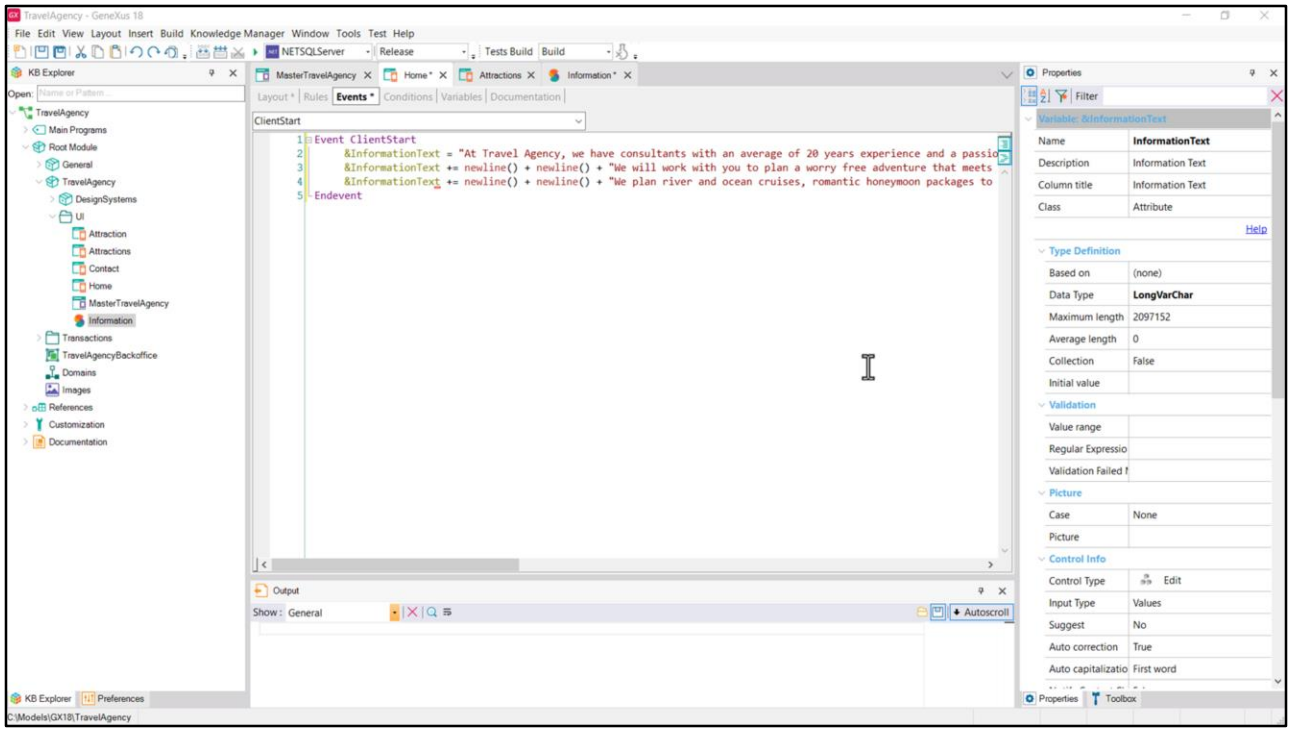
E o da variável sim, não tenho escolha a não ser fazer no evento ClientStart... então à variável InformationText atribuo…



… o conteúdo que copio daqui…

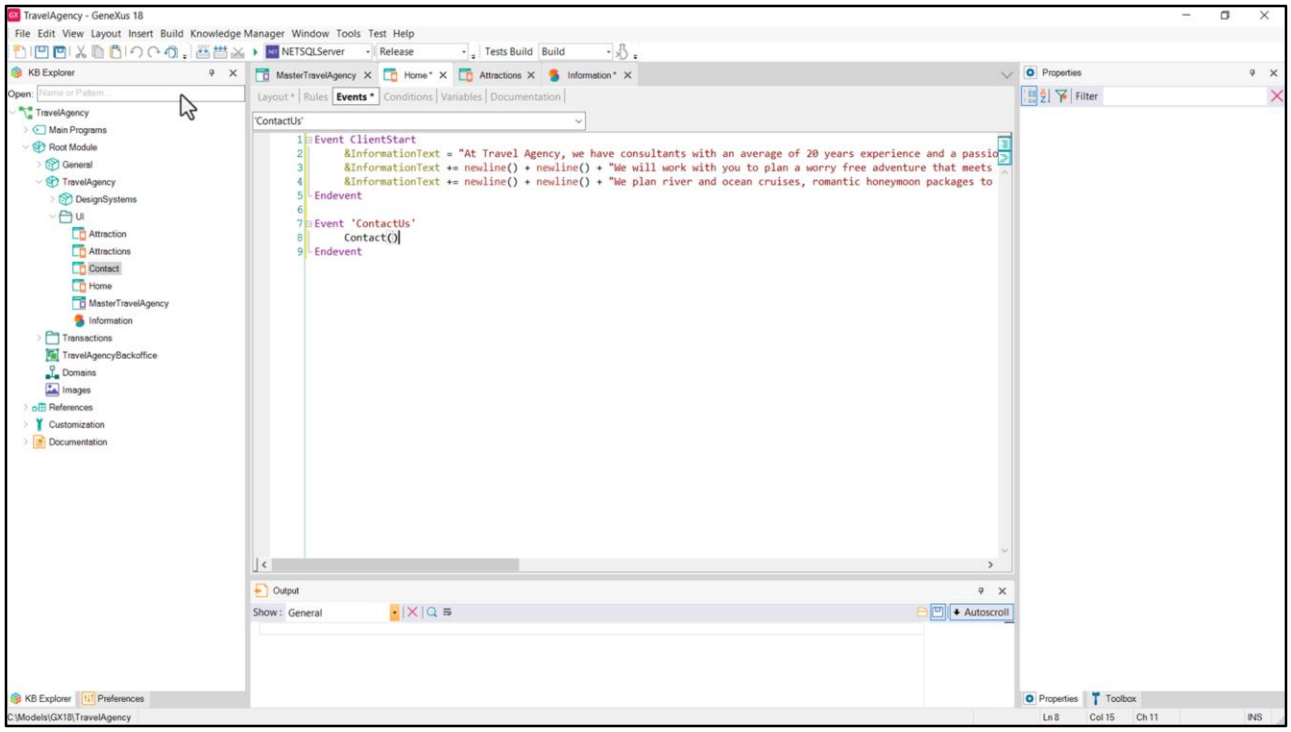


E tenho que separá-lo nas frases... Então... ao seu conteúdo, ao que tinha, adiciono um salto de linha, outro salto de linha…

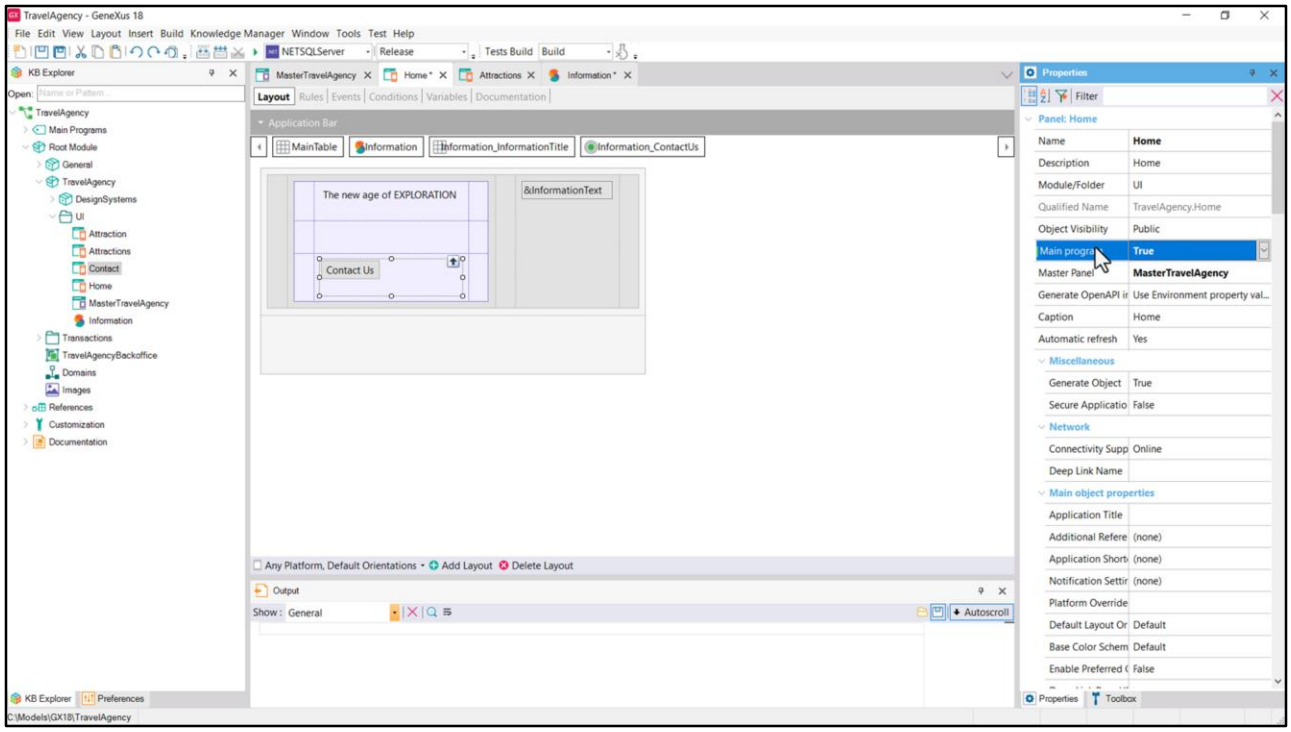


…e agora sim, isso que ainda estava aqui… e da mesma forma…

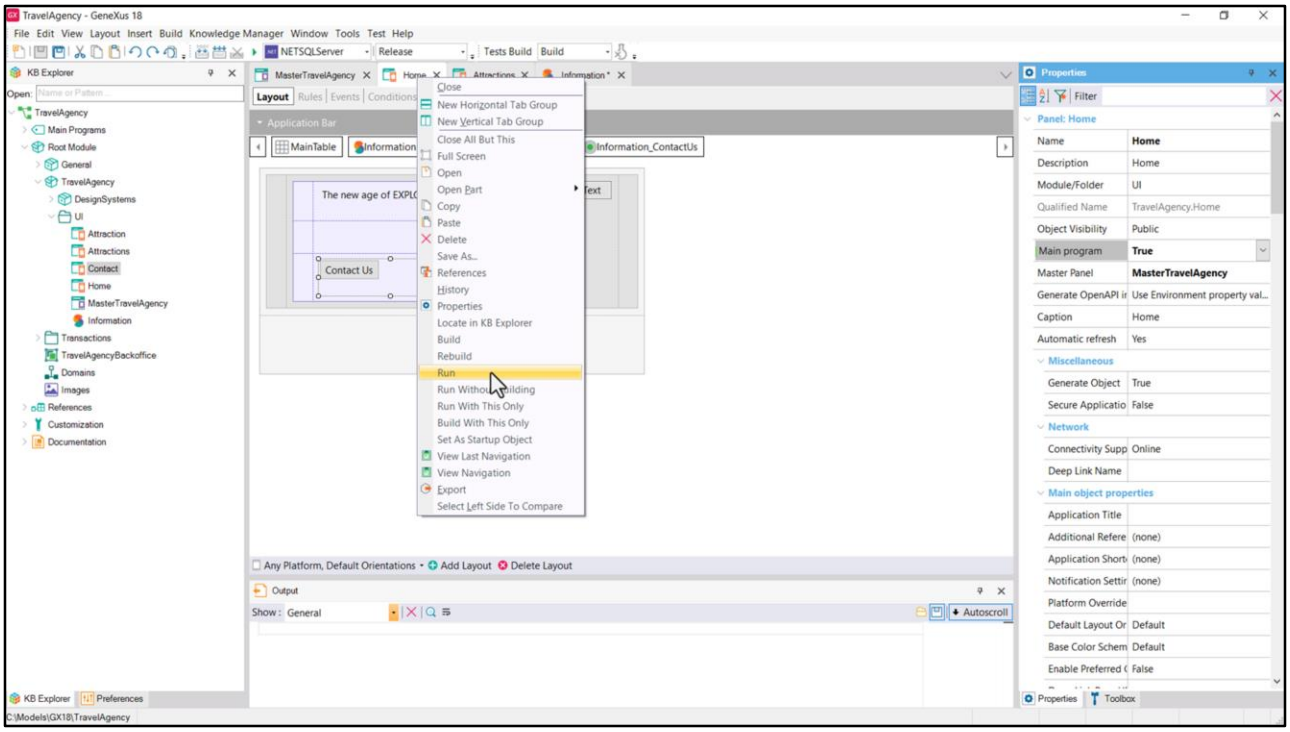
Um esclarecimento que não fiz: o evento ClientStart difere do evento Start porque o primeiro é executado no cliente e o segundo no servidor. E serão executados nessa ordem, primeiro o start do cliente, e depois o do server. O conteúdo da UI pode ser atualizado em ambos, mas como no nosso caso o conteúdo da variável não depende de nada do server, e é um conteúdo estático, fazemos isso no do cliente. Se estivéssemos implementando um Web panel em vez de um Panel, não teríamos o ClientStart e aí sim teríamos que utilizar o Start.



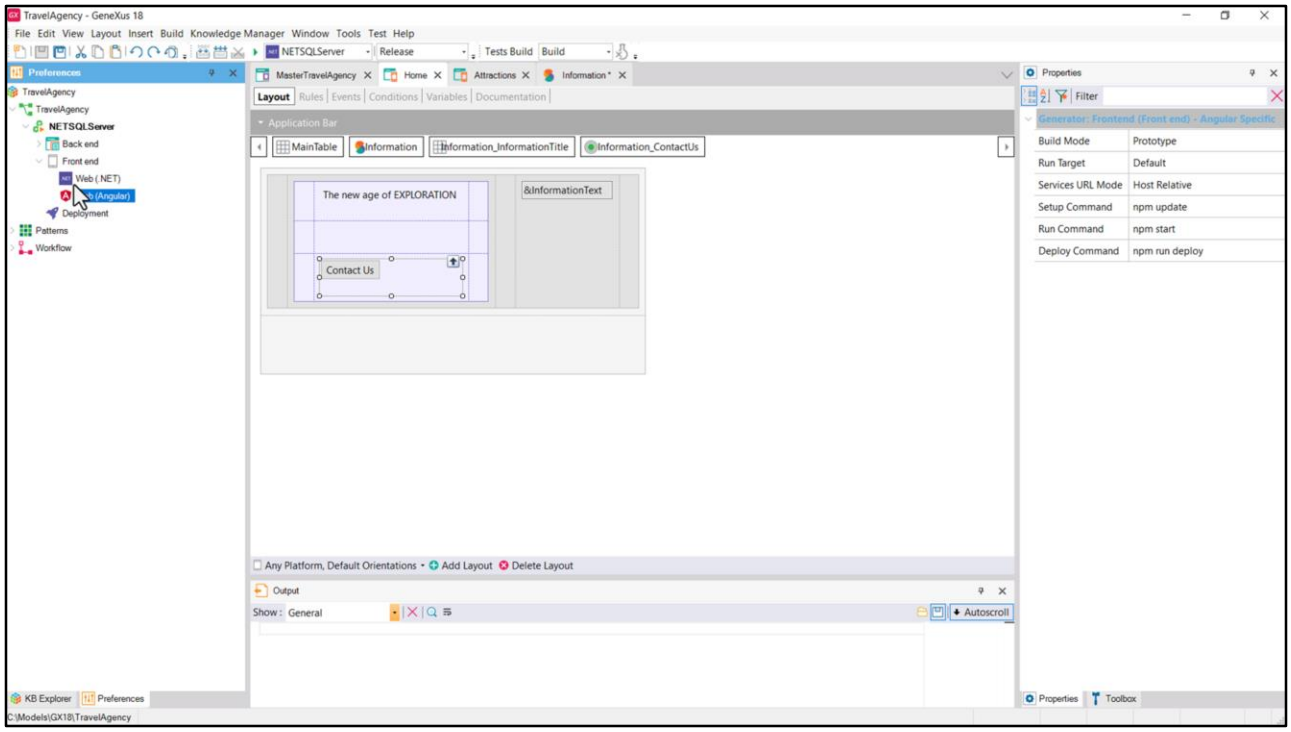
Bem, e por último, temos este evento ContactUs no nível do botão e vamos invocar este panel... que por enquanto está vazio.



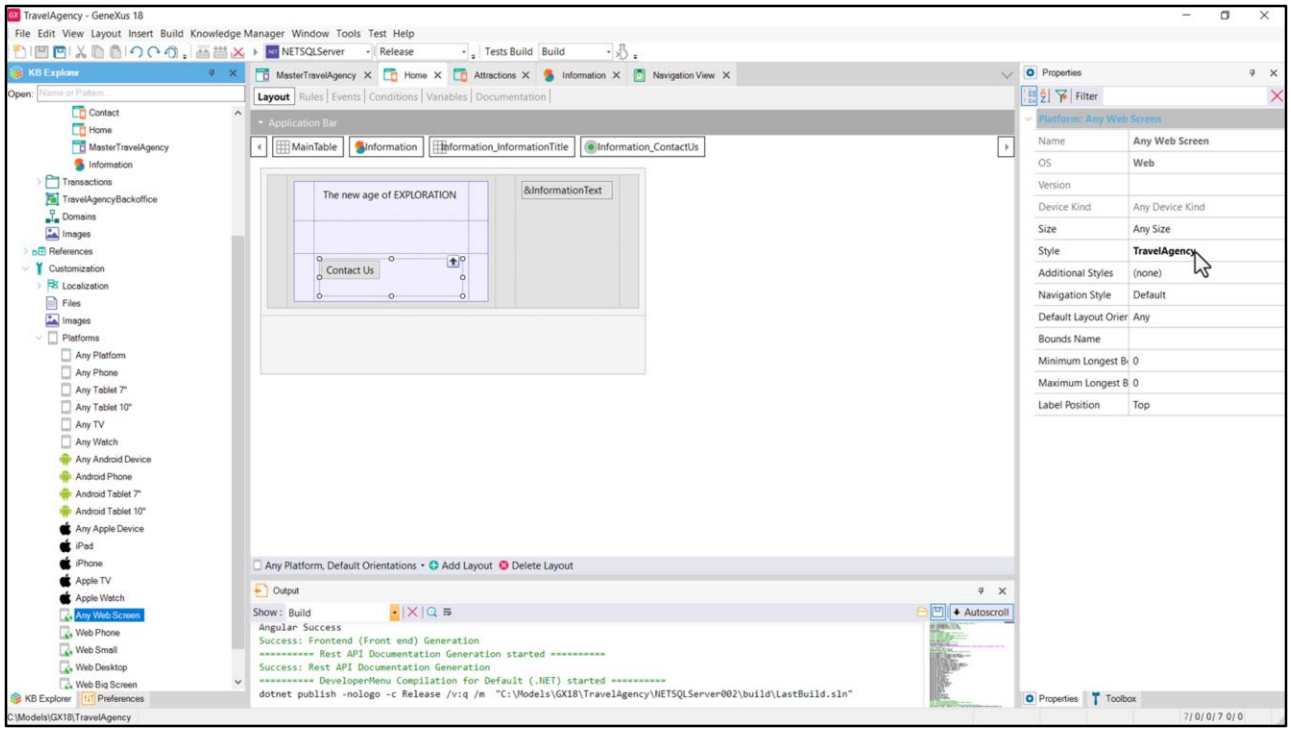
Bom, com isso já temos o objeto pronto para ser executado. O que precisamos para executá-lo em Angular? Por exemplo, colocar este panel como programa Main.



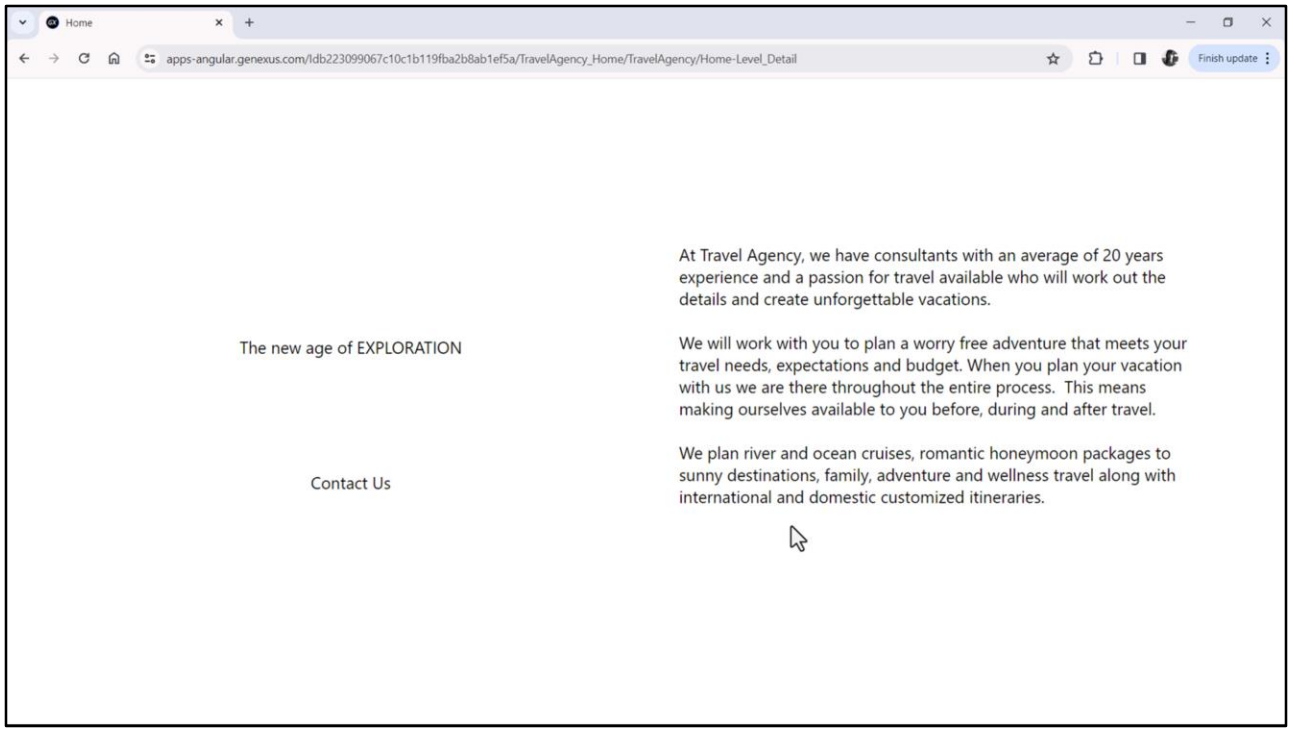
E ao fazer isso podemos clicar com o botão direito sobre o panel / Run.



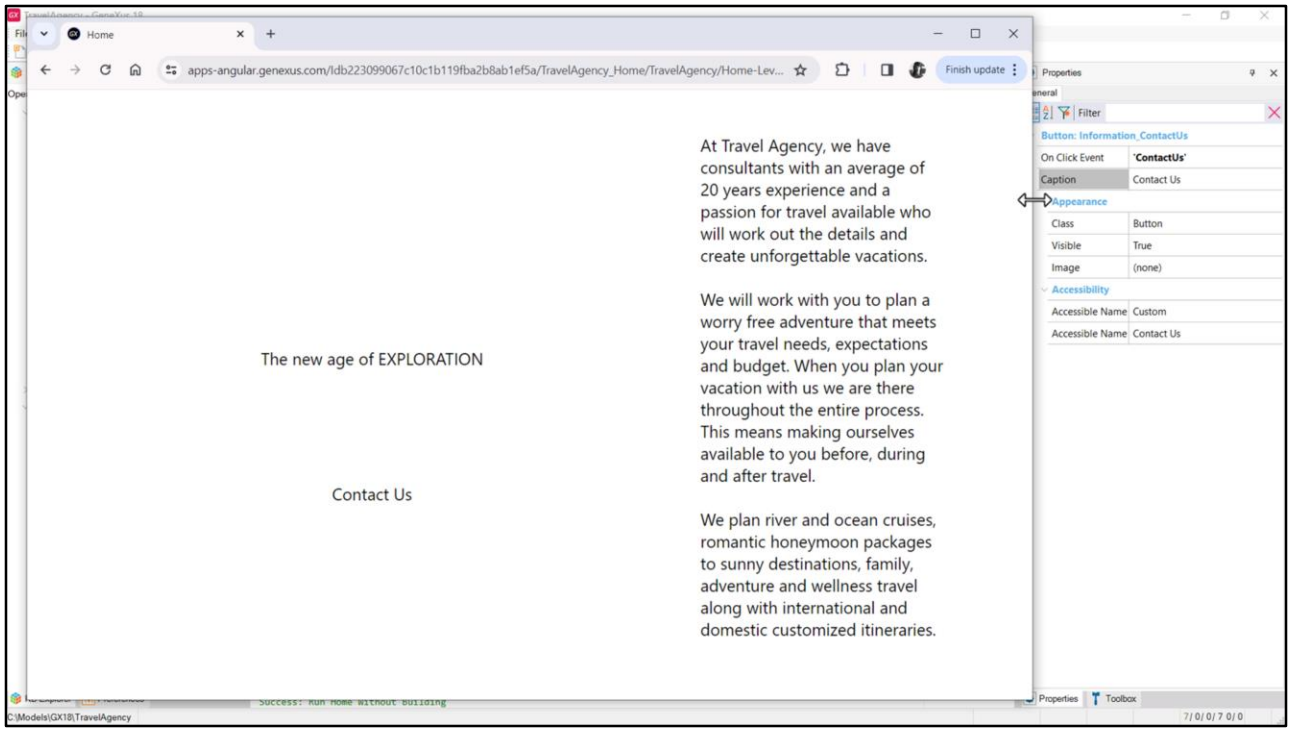
Sabendo que temos o Front end Angular definido e que é preferível sempre para o desenvolvimento prototipar local e não na nuvem, para economizar tempo de deploy toda vez que é atualizada qualquer coisa mínima. Então…



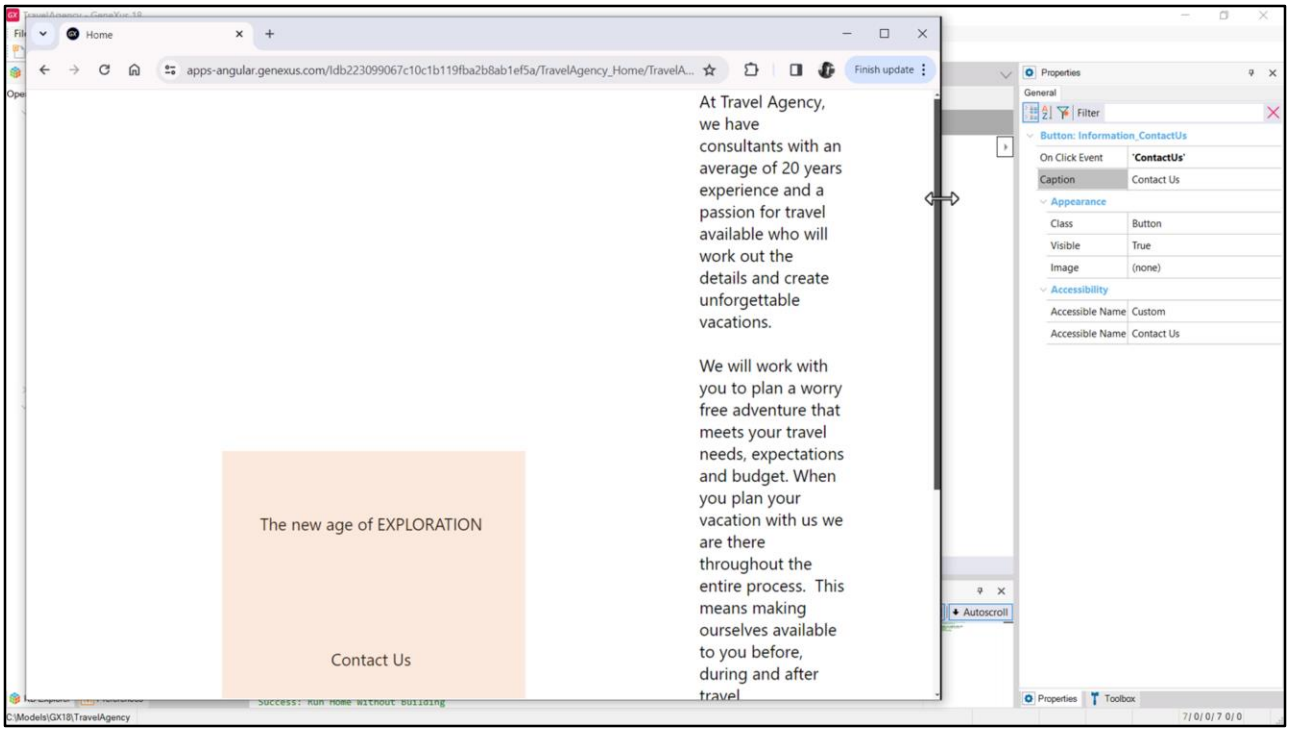
Antes de terminar, lembremos que havíamos especificado que a plataforma Any Web iria utilizar o DSO Travel Agency, que por enquanto está vazio.



E assim o vemos no navegador Chrome.

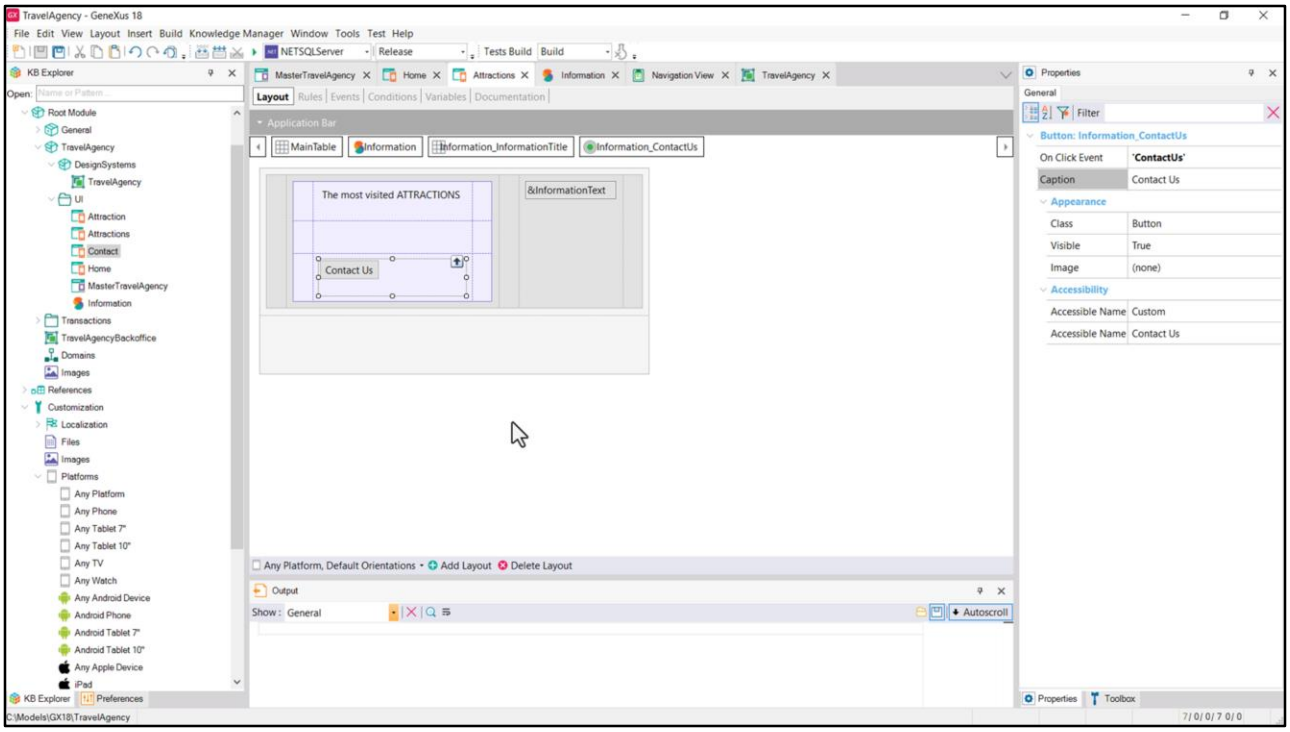


Vejamos que se diminuirmos a largura do navegador, tal como esperamos, aumenta a altura do texto da direita, mas nada muda nos outros.



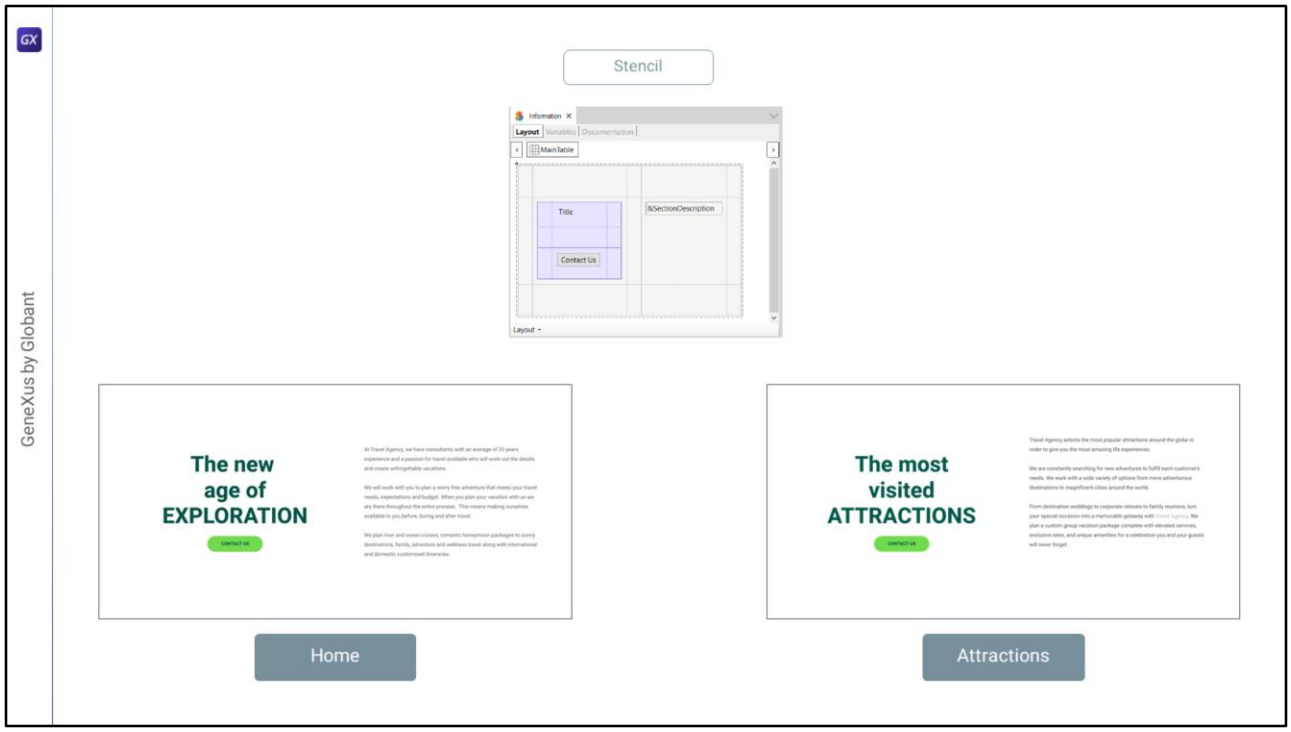
Nada exceto a altura da tabela, que é esticada para que o texto continue cabendo e por isso vemos esses outros textos irem para baixo, já que estão verticalmente centralizados.

Como nosso DSO está vazio, claramente estão sendo aplicados aos nossos controles os defaults do navegador, tanto para a fonte, seu espaçamento entre linhas, a cor e tudo mais. Sobre isso é que passaremos a atuar nos vídeos seguintes.



Claro, faremos o mesmo que fizemos para o panel Home, agora para o panel Attractions (vou reproduzi-lo rapidamente para não os entediar e com isso damos por concluída esta aula).

Preciso esclarecer algo antes de terminar: quando é indicado que um objeto é Main, isto será implementado como uma aplicação completa que incluirá o objeto e todos os envolvidos a partir dele. Como ainda não implementamos a parte do menu, a partir do qual ficarão todos os nossos objetos integrados, é que fizemos isso de colocar o Home como main e por outro lado o de Attractions, para poder executá-los agora que ainda não temos eles relacionados. Mas isso consome muito mais tempo do que se ambos os objetos estivessem integrados na mesma aplicação, porque da forma que estão, são literalmente duas aplicações independentes. Por enquanto vou deixar tudo assim porque vamos nos concentrar no que segue apenas no panel Home, mas depois farei com que todos estejam no mesmo programa, pois é superimportante reduzir os tempos de desenvolvimento.



Bem, e neste ponto já estamos em condições de começar a colocar nossas mãos na massa do Design System Object, para fazer com que essas duas seções de tela, tanto de Home quanto de Attractions, fiquem assim.


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