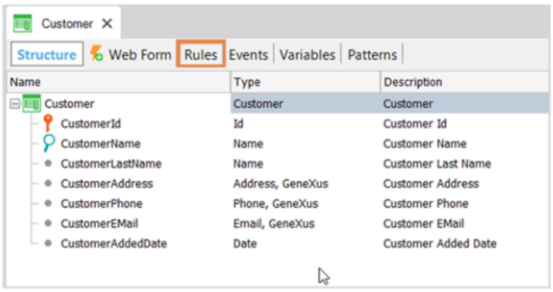
Regras: Revisão e Client-Side Validation

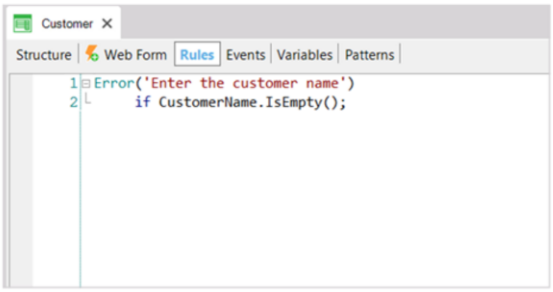
GeneXus™

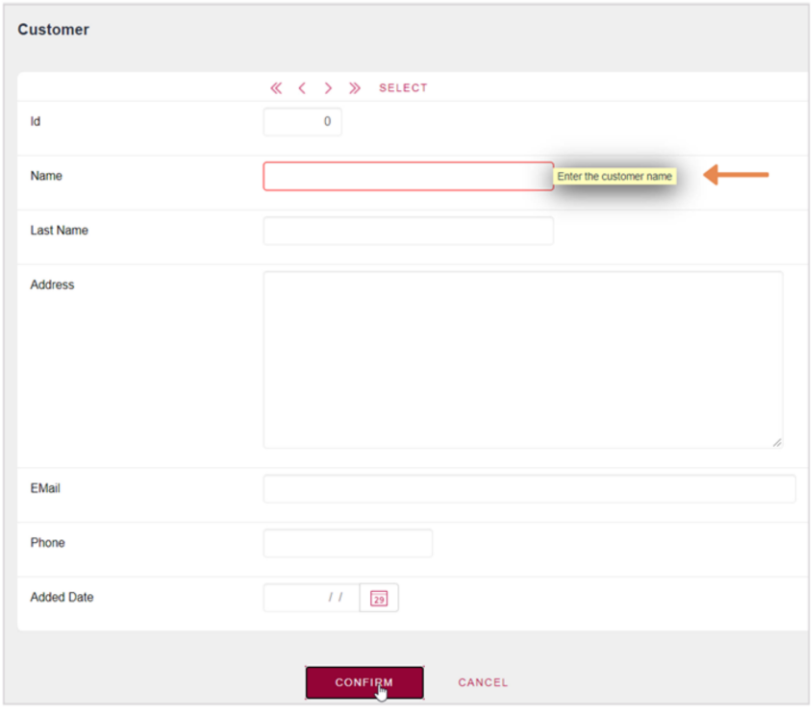
Rules



No curso anterior aprendemos que o objeto Transação conta com uma seção chamada **Rules**, na qual são definidos os controles que devem ser efetuados ou as regras que devem ser cumpridas para uma determinada realidade.

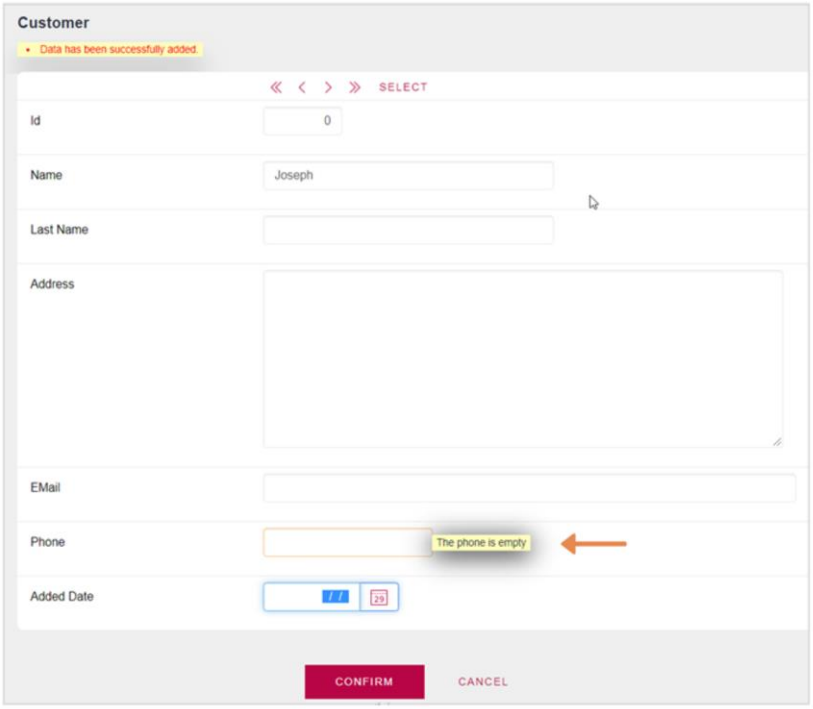
Rules





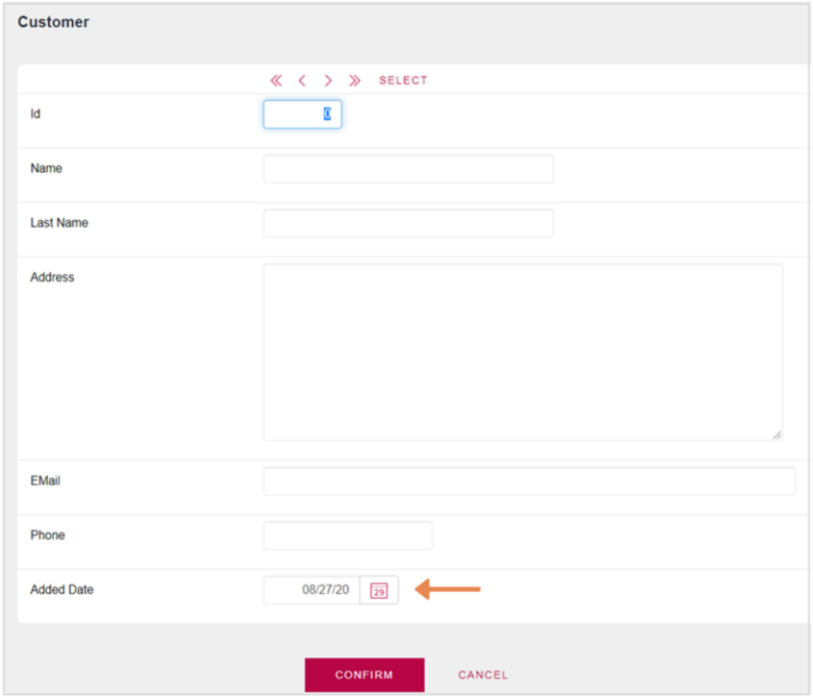
Nos concentramos na regra **Error**, que impede que um registro seja armazenado na base de dados enquanto uma determinada condição é cumprida: por exemplo, se estamos inserindo um cliente e deixamos seu nome vazio, esse cliente não poderá ser inserido até que o usuário tenha escrito seu nome.

Rules



Vimos também a regra **Message**, a qual somente informa ao usuário mediante uma mensagem mas permite efetuar a gravação;

Rules



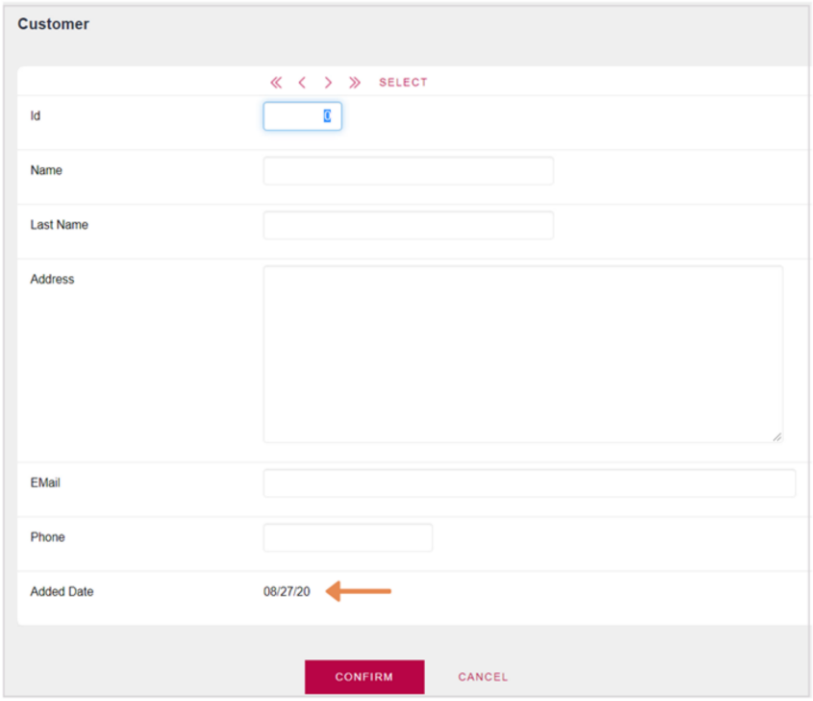
a regra **Default**, que nos permite inicializar um atributo ou variável com um valor quando acessamos à transação em modo de inserção;

Rules

Customer

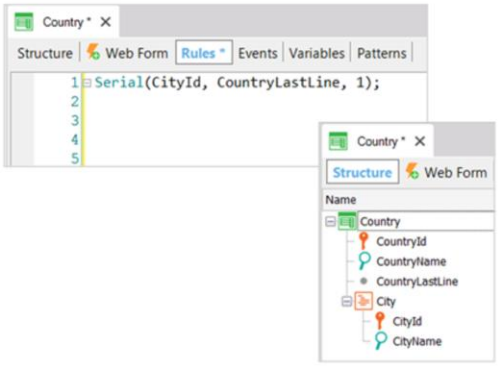
Structure | Web Form | Rules | Events | Variables | Patterns

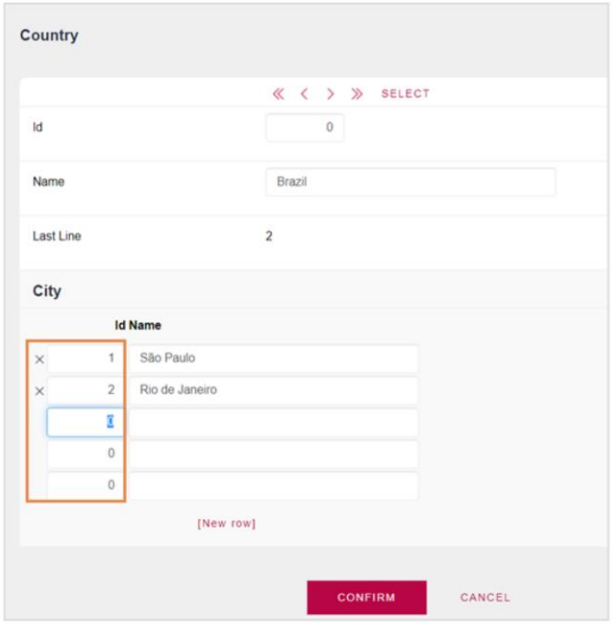
```
1  Error('Enter the customer name')
2      if CustomerName.IsEmpty();
3
4  Msg('The phone is empty')
5      if CustomerPhone.IsEmpty();
6
7  Default(CustomerAddedDate, &Today);
8
9  Noaccept(CustomerAddedDate);
```



a regra **NoAccept**, que impede que o usuário possa modificar um campo no formulário: mostra-o desativado,

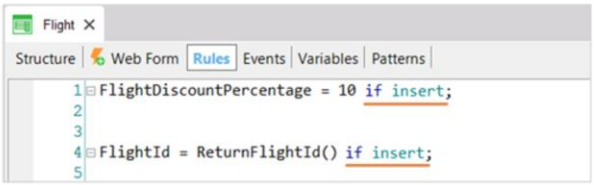
Rules





e por último a regra **Serial**, que serve para autonumerar um segundo nível, ou terceiro, ou outro nível aninhado de uma transação.

Rules



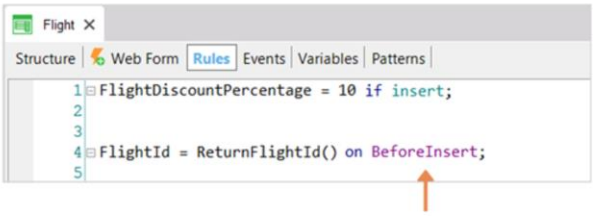
if insert

if update

if delete

Estudamos também que através das regras podemos definir atribuições de valores ou invocar objetos, e que também podemos condicioná-las para que se executem somente quando se está inserindo, modificando ou eliminando.

Rules

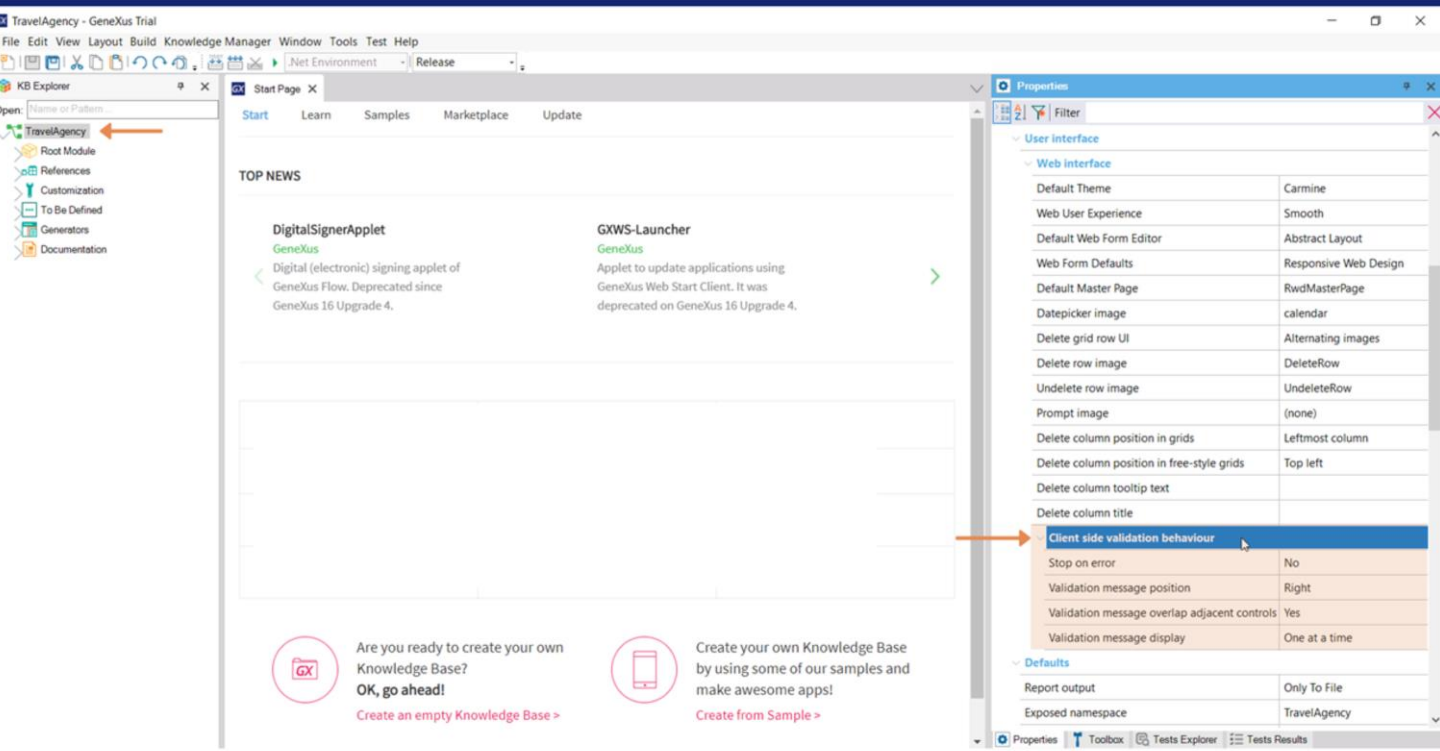


Além disso, se o momento escolhido por GeneXus para executar uma regra não é o que necessitamos, podemos indicar-lhe em que momento exato queremos que se execute, utilizando os eventos de disparo.

Execution of rules

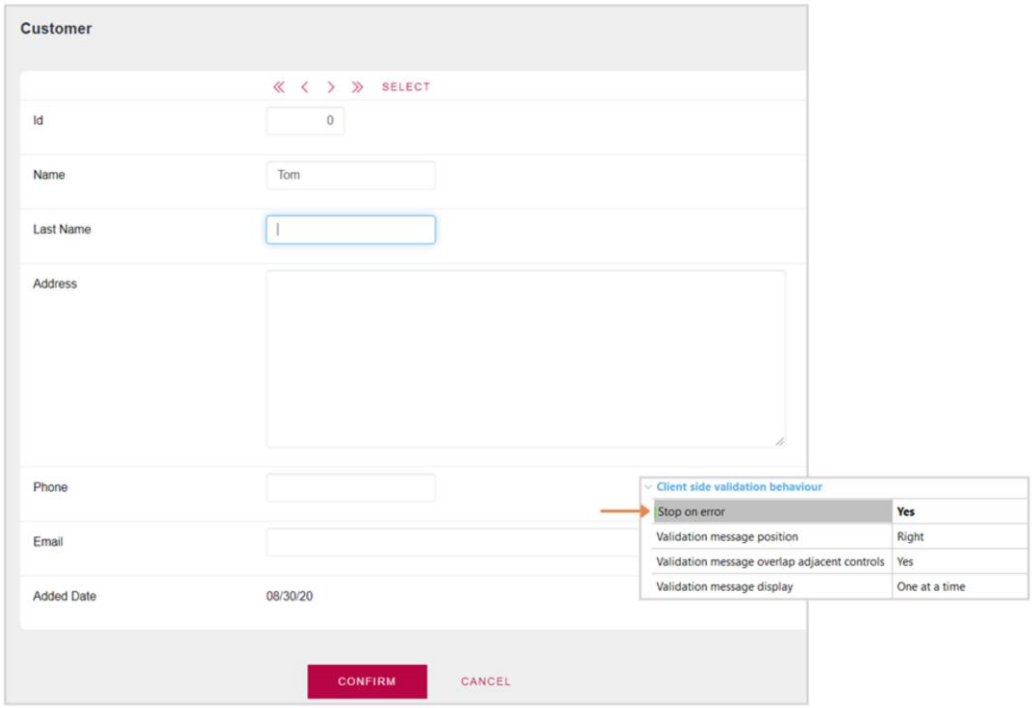
Recordemos que todas estas regras que temos estudado, são validadas tanto no cliente web como no servidor.

A validação que ocorre no cliente é chamada **Client-Side Validation**, e seu objetivo é proporcionar uma boa experiência ao usuário, fazendo-o sentir que a aplicação está sempre interagindo com ele; mas é o servidor quem realmente valida que toda a informação enviada seja consistente e não viole a segurança do sistema, e é o único que pode operar sobre a base de dados.



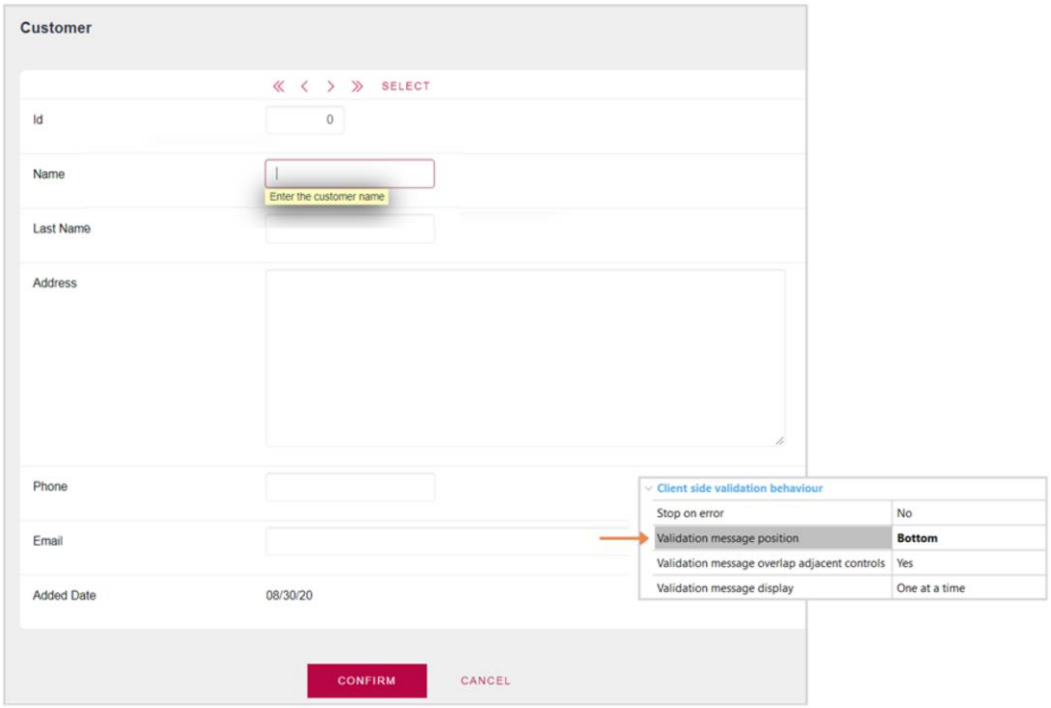
Até aqui temos visto o comportamento padrão destas regras. Mas é interessante saber que temos um grupo de propriedades chamado **Client Side Validation Behaviour** que encontramos no nível da versão, que nos permite personalizar o comportamento e as mensagens das regras de muitas formas, tornando a aplicação mais atraente para usuários finais.

Stop on error property



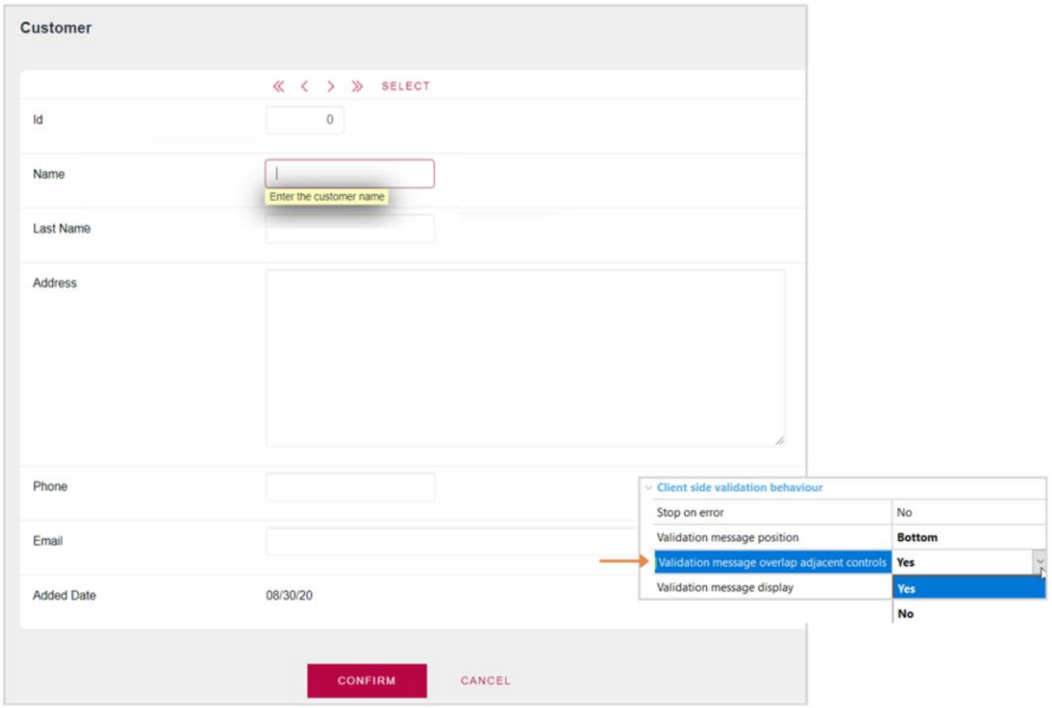
Quando utilizamos a regra Error, vimos que embora a transação não permita gravar, permite passar para o próximo campo, depois de ter mostrado a mensagem de erro. Se nos interessar que em caso de ser disparado um erro, se mantenha o foco no controle e obrigue o usuário a corrigir o valor para poder passar ao campo seguinte, devemos mudar o valor da propriedade ***Stop on error*** para *Yes*.

Validation message position property



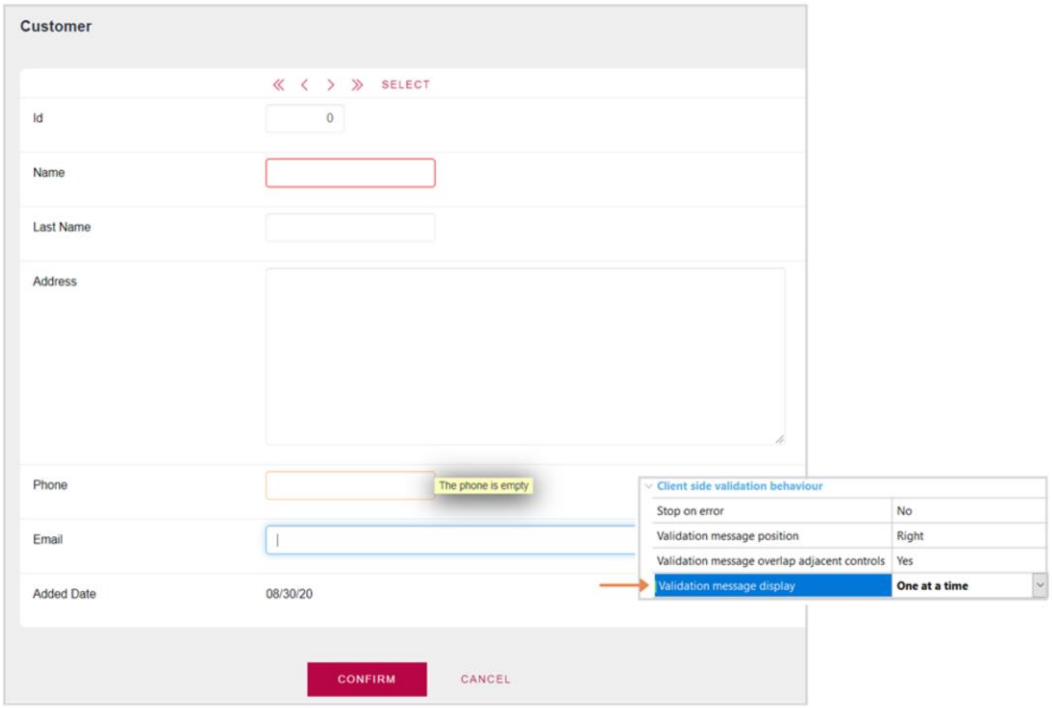
También poderemos definir o lugar onde vai a mensagem em relação ao campo, utilizando a propriedade ***Validation Message Position***, que conta com os valores *Right* (que é o valor padrão), *Left*, *Top* e *Bottom*.

Validation message overlap adjacent property



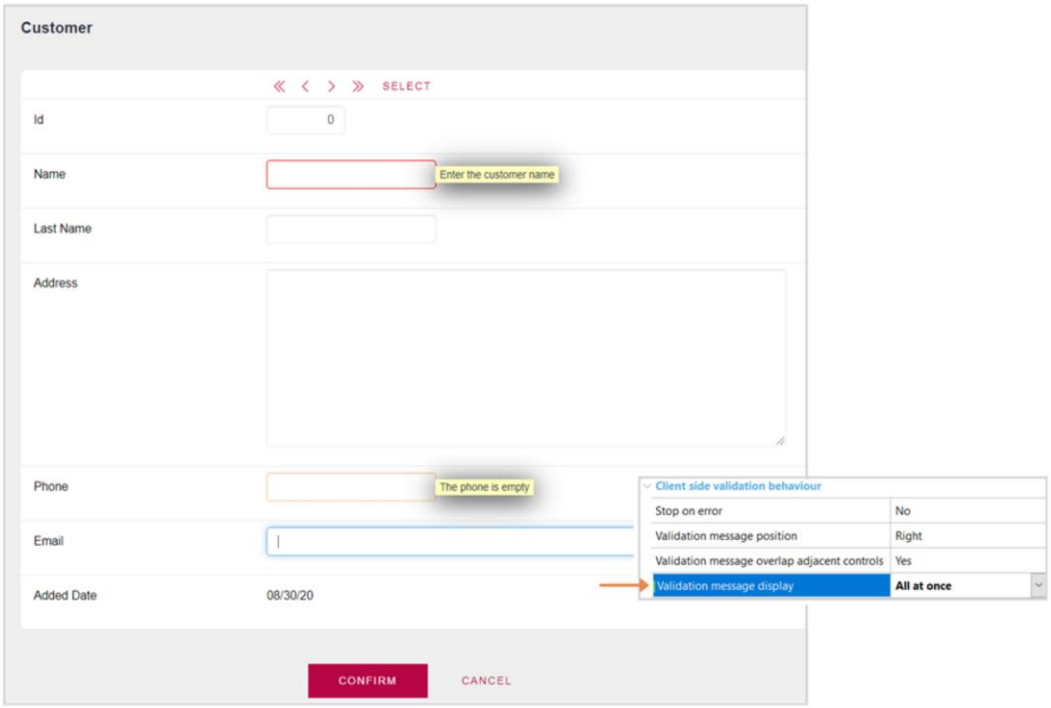
No caso da mensagem se mostrar sobreposta a algum outro controle, deve ser configurada também a propriedade ***Validation message overlap adjacent controls***, que controla se as mensagens devem ou não sobrepor-se.

Validation message display property

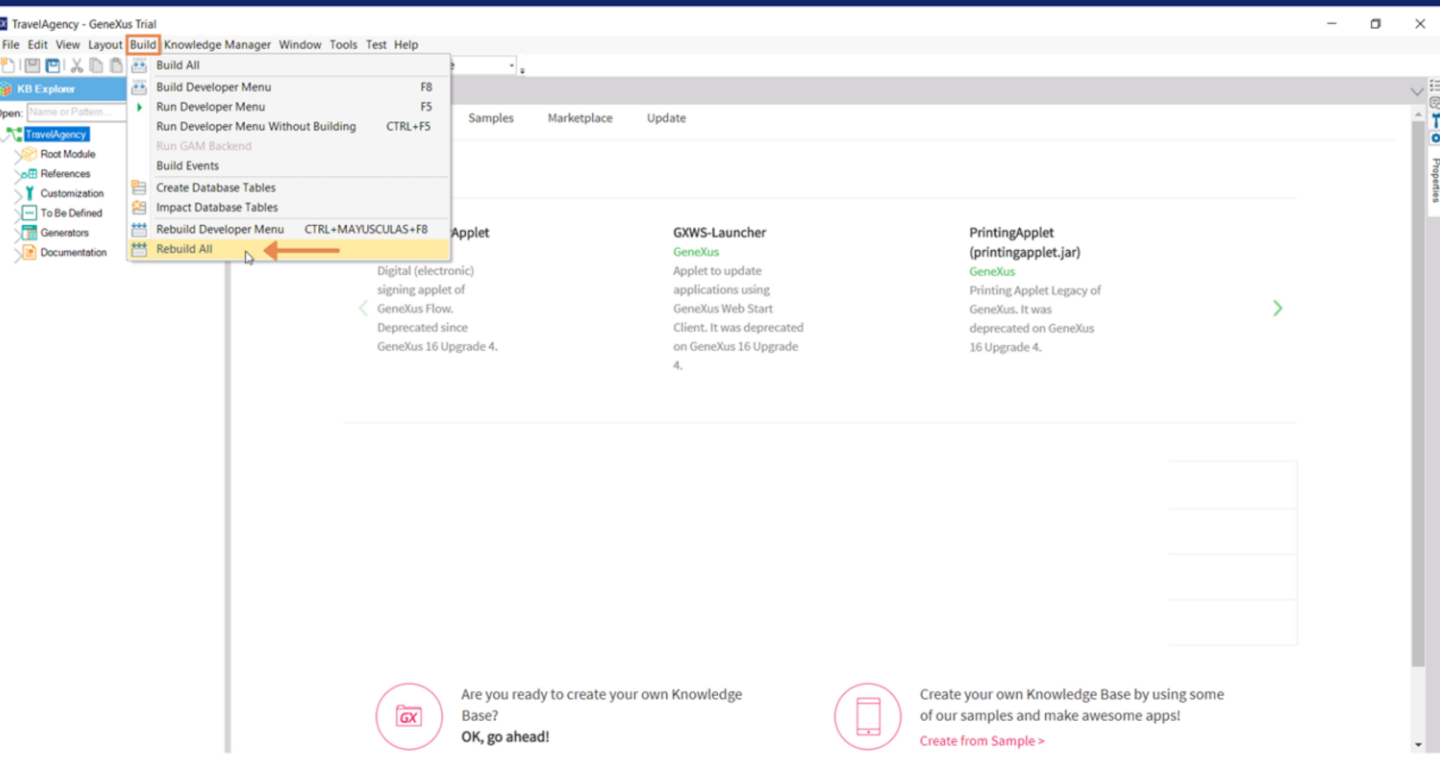


A última propriedade do grupo é ***Validation message display***, que conta com os valores *One at a time* (que é o valor padrão) e cujo objetivo é mostrar sempre a última mensagem de validação que ativa a aplicação,

Validation message display property



e *All at once*, que mostra ao mesmo tempo cada mensagem de validação que deveria estar na tela.



Como estas propriedades modificam o comportamento relativo à interação nos forms, devem ser gerados novamente todos os objetos que têm form para que as mudanças sejam efetuadas. Para isso, deve utilizar a opção **Rebuild All**, que gera absolutamente todos os objetos.

No vídeo seguinte, adicionaremos algumas regras interessantes àquelas já conhecidas e em outros estudaremos como são avaliadas para determinar a ordem de execução, bem como nos aprofundaremos um pouco mais nos eventos de disparo.

GeneXus™

training.genexus.com

wiki.genexus.com

training.genexus.com/certifications