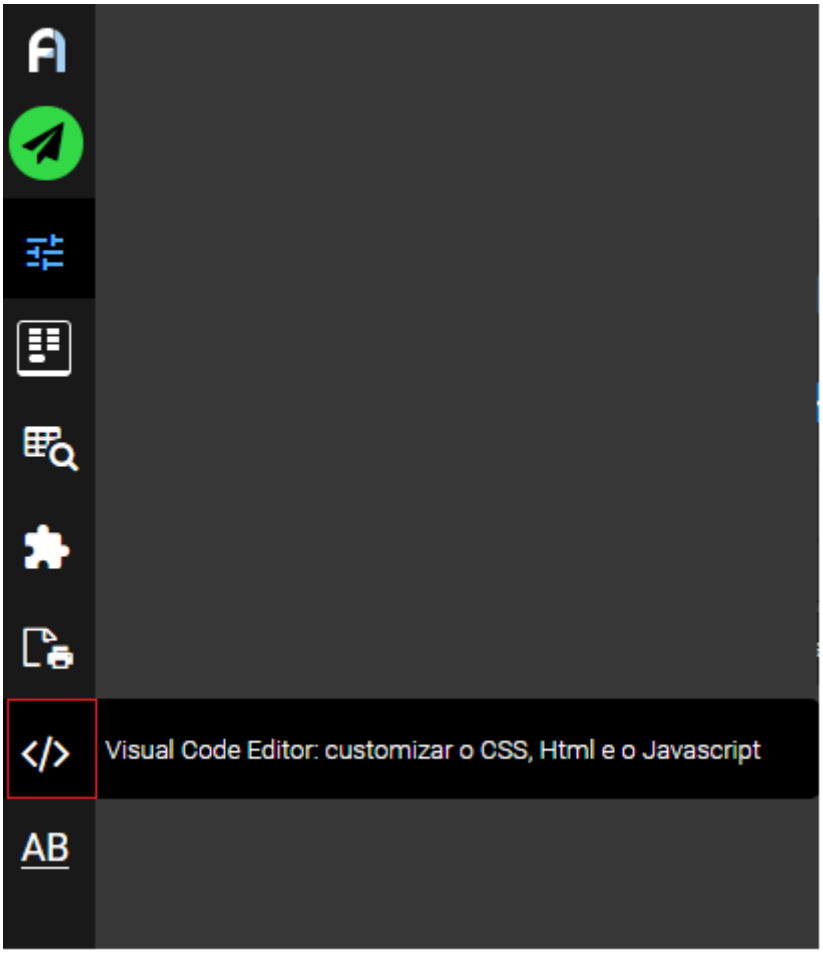
Formulário - customizar com HTML, C#, CSS e Javascript

O agilityflow também pode ser customizado através da Linguagem Html, CSS, Javascript e C# (csharp).

O html do agilityflow é baseado no Razor do framework .Net, isso é, você pode além de customizar com HTML, utilizar código C# (razor) para dar mais flexibilidade e poder ao seu componente Html.

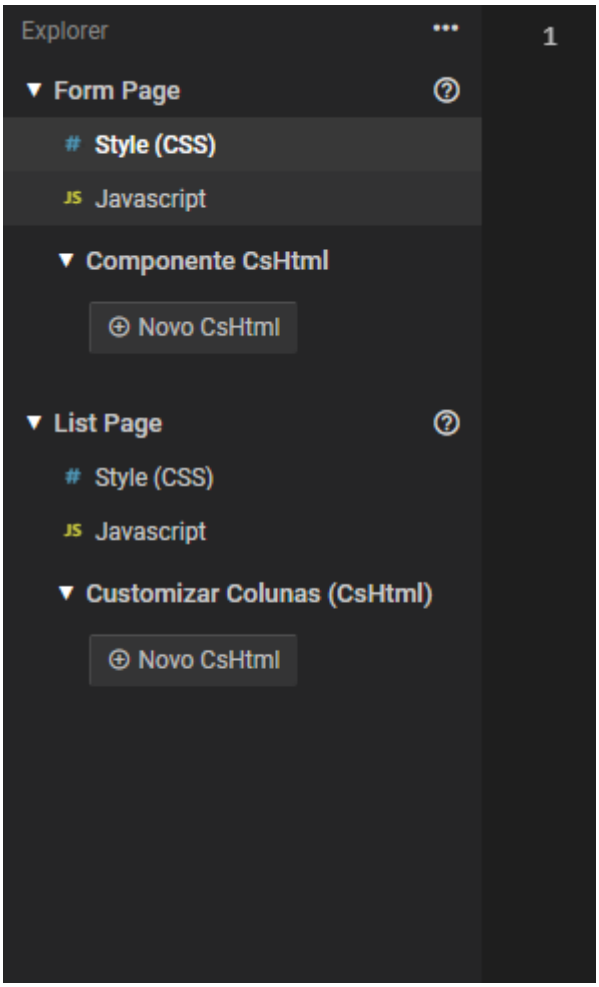
O que e onde customizar?

Acesse a parte de Desenvolvimento do Formulário que você deseja customizar, para isso, entre na área de customização de um formulário e clique na opção Visual Code Editor, como na imagem abaixo:



No Visual Code editor, no menu esquerdo, serão apresentados as seguintes opções:

- **Style (CSS):** opção para customizar o CSS de todo o formulário.
- **Javascript:** opção para customizar o Javascript de todo o formulário.
- **Componente CsHtml:** opção para criar novos componentes em HTML



Style (CSS)

Nessa área, você pode criar customizações via linguagem CSS para todo o formulário, o seu uso é livre.

> Levar em consideração o tratamento em CSS para a responsividade da plataforma em outros device, Mobile, Computador Desktop, Tablet, etc.

Javascript:

Nessa área, você pode criar customizações via linguagem Javascript para todo o formulário, o seu uso é livre.

Você pode utilizar jquery e outras bibliotecas já declaradas no código-fonte por padrão (veja lista mais abaixo).

Exemplo de utilização:

No código javascript abaixo, a regra impede que um formulário de projetos que esteja com o status "inativo", seja salvo:

```
var projeto = {
    init: function () {
        projeto._setEvents(); //registrar os eventos que são inerentes a esse bloco de código
    },
    _setEvents: function () {

      //ao fazer o submit do formulário executa a função de validação de status
        $('#formulario').on('submit', function (e) {
            projeto.validarStatus();
        })

    },
    validarStatus: function () {

      //recupera através de jquery o valor do campo "Status do Projeto"
        var status_projeto = $("#status_projeto").val();

      //verifica se o tipo de envio que o formulário está requisitando no submit
        //é do tipo "SALVAR" e se o status é "inativo"
        //caso seja, inválida o submit e apresenta uma aviso para o usuário
        if (Form.getFormActionOnSubmit() == ACTION_SALVAR && status_projeto == "inativo") {

            //inválida o submit e apresenta uma aviso para o usuário
            Form.invalidateForm({
                msg: 'Um projeto inativo não pode ser salvo.'
            });

        }

    }
```

```
}
projeto.init(); //iniciando e configurando
```

Abaixo estão listadas algumas funções nativas do agility**Flow** para javascript:

Como utilizar funções Javascript dentro de um CSHTML

> Caso você utilize Javascript dentro de um componente HTML (CSHTML), é necessário que o código esteja dentro da função **DOM.ready (como no exemplo abaixo)** para garantir que o seu código só execute após todo o carregamento de todos as funções Javascript

> Recomendamos que dentro do arquivo CSHTML só tenha a "chamadas" para as funções que estejam programadas dentro do arquivo específico para o código Javascript.

```
<script>
   DOM.ready(function () {

  //seu código javascript aqui

   });
</script>
```

Funções Javascript Nativas

Recuperar data do servidor:

```
GetDateNow()
```

> A data e hora são obtidas dos servidores do Agilityflow, para assim garantir a confiabilidade da data e hora

> **Atenção:** Lembre-se que no Javascript o retorno do método ".getMonth()" de um objeto **Date**, não refere-se ao mês e sim a Índice daquele mês no array de meses, sendo assim, o retorno desse método sempre será entre os valores **0 até 11** e não de 1 a 12

Exemplo de como configurar o valor inicial de um campo com a data atual.

Para esse exemplo, encare que o ID do campo que estamos preenchendo com a data atual é "data".

```
var exemplo1 = {
    init: function () {
        exemplo1.setToday();
    },
    //coloca a data atual no campo de data
    setToday: function () {

        if ($('#data').val() != "") return;

        //data do servidor para garantir que seja uma data válida
        var date = GetDateNow();

        var day = date.getDate().toString().padLeft(2, '0');
        var year = date.getFullYear();
        var monthIndex = date.getMonth()
        monthIndex++;
        if (monthIndex > 12)
            monthIndex = 1;



        $('#data').val(day.toString().padLeft(2, '0') + '/' + monthIndex.toString().padLeft(2, '0') + '/' + year);


    }
}
exemplo1.init();
```

Comparar uma data com a data atual (Hoje):

> Essa comparação desconsidera a HORA

A função é: **formContext.datetime.validation.compareToday(comparador, data)**

Os possíveis comparadores são:

Tipo	Opção 1 (qualquer uma das opções)	Opção 2 (qualquer uma das opções)
maior que	>	greater

maior ou igual	>=	greater-or-equal
menor	<	less
menor ou igual	<=	less-or-equal
igual	==	equal

```
var greater = formContext.datetime.validation.compareToday('greater', '31/12/2000');
console.log('greater',greater);

var greater_or_equal = formContext.datetime.validation.compareToday('greater-or-equal', '31/12/2000');
console.log('greater-or-equal', greater_or_equal);

var less = formContext.datetime.validation.compareToday('less', '31/12/2000');
console.log('less',less);

var less_or_equal = formContext.datetime.validation.compareToday('less-or-equal', '31/12/2000');
console.log('less-or-equal', less_or_equal);

var equal = formContext.datetime.validation.compareToday('equal', '31/12/2000');
console.log('equal', equal);
```

Comparação entre datas. Comparar 2 datas:

> Essa comparação desconsidera a HORA

A função é: **formContext.datetime.validation.compare(data1, comparador, data2)**

Os possíveis comparadores são:

Tipo	Opção 1 (qualquer uma das opções)	Opção 2 (qualquer uma das opções)
maior que	>	greater
maior ou igual	>=	greater-or-equal
menor	<	less
menor ou igual	<=	less-or-equal

igual	==	equal

```
var greater = formContext.datetime.validation.compare('31/01/2001','greater', '31/12/2000');
console.log('greater',greater);

var greater_or_equal = formContext.datetime.validation.compare('31/01/2001','greater-or-equal', '31/12/2000');
console.log('greater-or-equal', greater_or_equal);

var less = formContext.datetime.validation.compare('31/01/2001','less', '31/12/2000');
console.log('less',less);

var less_or_equal = formContext.datetime.validation.compare('31/01/2001','less-or-equal', '31/12/2000');
console.log('less-or-equal', less_or_equal);

var equal = formContext.datetime.validation.compare('31/01/2001','equal', '31/12/2000');
console.log('equal', equal);
```

Recuperar o ID do usuário logado:

```
GetUserId()
```

Recuperar o nome do usuário logado:

```
var nome = pageNavigation.getUserName();
var nomeCompleto = pageNavigation.getUserNameCompleto()
```

Recuperar os Perfis do usuário logado:

```
GetUserProfileId() //retorna um array com os Ids dos perfis ['xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx','xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx']
```

Função para identificar se o usuário logado tem um determinado perfil:

```
UserHasProfileId('xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx') //retorna true ou false
```

Expandir e Recolher um determinado painel de campos

```
var panelId = 'f175132d-5e85-778f-1354-cb2d339e6146';

//para alternar entre expandido e recolhido
formContext.panel.collapse.toggle(panelId)

//para recolher o painel
formContext.panel.collapse.hide(panelId)

//para expandir o painel
formContext.panel.collapse.show(panelId)

//para testar se o painel está recolhido ou não
formContext.panel.collapse.isCollapsed(panelId)
```

Recuperar os valores das "Variáveis de Ambiente"

Para saber mais sobre variáveis de ambiente entre em Variáveis de ambiente

```
var valor = GetEnvironmentVariable("nomeDaVariavel")
```

Recuperar a Descrição (texto), no caso de variáveis do Tipo "Query com Id + Descrição":

```
var text = GetEnvironmentVariable_Text("nomeDaVariavel")
```

Recuperar o VALOR preenchido de um campo no Formulário

```
var value = Form.getValueField("idDoCampo");
```

Recuperar o VALOR preenchido de um campo no Formulário PAI (formulário que abriu o formulário atual)

```
var value = Form.getValueField_parentIFrame("idDoCampo");
```

Recuperar o TEXTO preenchido de um campo no Formulário (Para os campos: Lista de Seleção e Pesquisa com Auto Completar)

```
var value = Form.getTextField("idDoCampo");
```

Recuperar o TEXTO preenchido de um campo no Formulário PAI (formulário que abriu o formulário atual) (Para os campos: Lista de Seleção e Pesquisa com Auto Completar)

```
var value = Form.getTextField_parentIFrame("idDoCampo");
```

Preencher um valor em um campo do Formulário

Parâmetros opcionais da função javascript

Parâmetro opcional	Valor padrão	Opções de valores	Descrição
setOnlyIfFieldValueIsEmpty	false	true ou false	Caso seja "**true**" a função só preencherá o valor do campo se o campo ainda **NÃO** estiver preenchido. Caso seja "**false**" a função preenche ou substitui o valor do campode qualquer maneira.

```
//EXEMPLO 1: preenchendo o valor
Form.setValueField("idDoCampo", "value");



//EXEMPLO 2: preenchendo o valor e enviando os parametros opcionais para a função:
var options = { setOnlyIfFieldValueIsEmpty: true  }
Form.setValueField("idDoCampo", "value", options);
```

Preencher um valor em um campo do Formulário PAI (formulário que abriu o formulário atual)

Parâmetros opcionais da função javascript

Parâmetro opcional	Valor padrão	Opções de valores	Descrição

setOnlyIfFieldValueIsEmpty	false	true ou false	Caso seja "**true**" a função só preencherá o valor do campo se o campo ainda **NÃO** estiver preenchido. Caso seja "**false**" a função preenche ou substitui o valor do campode qualquer maneira.

```
//EXEMPLO 1: preenchendo o valor
Form.setValueField_parentIFrame("idDoCampo", "value");



//EXEMPLO 2: preenchendo o valor e enviando os parametros opcionais para a função:
var options = { setOnlyIfFieldValueIsEmpty: true }
Form.setValueField_parentIFrame("idDoCampo", "value", options);
```

Customizar evento de submit do formulário (Antes do post):

```
$('#formulario').on('submit', function (e) {
□□/*
        função js para executar antes do
        post e após a validação padrão do agilityflow


      */
})
```

No exemplo de javascript, nas linhas de 5 a 12 usamos o código para interceptar o submit e validar o status do projeto.

Forçar o Salvar do formulário:

Clique aqui para visualizar detalhes de Como forçar o salvamento de um Form

Escutar o evento de retorno de um submit no form:

Clique aqui para visualizar detalhes de Como escutar o evento de retorno de um submit no form

Invalidar o post/submit do formulário:

Para evitar que o formulário seja postado, você pode utilizar a seguinte função no evento de submit

```
$('#formulario').on('submit', function (e) {

    //invalidar submit
    Form.invalidateForm({
      msg: 'Mensagem para o usuário.'
    });

})
```

No exemplo de javascript, nas linhas de 23 a 26 usamos o código para bloquear o submit caso o status do projeto seja inválido.

Verificar qual foi o botão clicado pelo usuário no submit do formulário. Se foi o salvar, salvar rascunho, descartar, deletar, aprovar, retornar, reprovar e assim por diante...

Os tipos de ações que o usuário pode realizar em um formulário são:

- Salvar (**ACTION_SALVAR**)
- Salvar rascunho (**ACTION_SALVAR_RASCUNHO**)
- Descartar rascunho (**ACTION_DESCARTAR_RASCUNHO**)
- Aprovar (**ACTION_APROVAR**)
- Reprovar (**ACTION_REPROVAR**)
- Retornar (**ACTION_RETORNAR**)
- Deletar (**ACTION_DELETAR**)
- Salvar formulário filho (**ACTION_SALVAR_FORMULARIO_FILHO**)
- Deletar formulário filho (**ACTION_DELETAR_FORMULARIO_FILHO**)
- Descartar alterações formulário filho (**ACTION_DESCARTAR_ALTERACOES_FORMULARIO_FILHO**)
- Solicitar troca de aprovador na etapa (**ACTION_SOLICITAR_TROCA_APROVADOR_ETAPA_DINAMICA**)
- Salvar o novo aprovador definido para a etapa (**ACTION_SALVAR_DEFINICAO_APROVADOR_ETAPA_DINAMICA**)

Abaixo o exemplo para recuperar a ação:

```
$('#formulario').on('submit', function (e) {

  var tipoAcao = Form.getFormActionOnSubmit();
  if(tipoAcao == ACTION_SALVAR){

        //ação para "Salvar"
```

```
}else if(tipoAcao == ACTION_SALVAR_RASCUNHO){

    //ação para "Salvar rascunho"

}else if(tipoAcao == ACTION_DESCARTAR_RASCUNHO){

    //ação para "Descartar rascunho"

}else if(tipoAcao == ACTION_APROVAR){

    //ação para "Aprovar"

}else if(tipoAcao == ACTION_REPROVAR){

    //ação para "Reprovar"

}else if(tipoAcao == ACTION_RETORNAR){

    //ação para "Retornar"

}else if(tipoAcao == ACTION_DELETAR){

    //ação para "Deletar"

}else if(tipoAcao == ACTION_SALVAR_FORMULARIO_FILHO){

    //ação para "Salvar formulário filho"

}else if(tipoAcao == ACTION_DELETAR_FORMULARIO_FILHO){

    //ação para "Deletar formulário filho"

}else if(tipoAcao == ACTION_DESCARTAR_ALTERACOES_FORMULARIO_FILHO){

    //ação para "Descartar alterações formulário filho"

}else if(tipoAcao == ACTION_SOLICITAR_TROCA_APROVADOR_ETAPA_DINAMICA){

    //ação para "Solicitar troca aprovador etapa dinâmica"
```

```
    }else if(tipoAcao == ACTION_SALVAR_DEFINICAO_APROVADOR_ETAPA_DINAMICA){

        //ação para "Salvar definicão de  aprovador etapa dinâmica"

    }

})
```

No exemplo de javascript, nas linha 21 usamos o código no IF para testar se o submit era do tipo "Salvar".

Mensagem e alerta para o usuário

No código abaixo mostra exemplo de como apresentar alertas na tela para o usuário.

```
//mensagem de sucesso
pageMsg.showMsgSuccess(msg, title);

//mensagem de aviso
pageMsg.showMsgWarning(msg, title);

//mensagem de erro
pageMsg.showMsgError(msg, title);

//esconder a mensagem
pageMsg.hideMsgs(true);
```

Recuperar um valor de uma QueryString

```
var value = getQuerystring('paramQuerystring1');
```

Você também pode usar o javascript nativo, detalhado nesse exemplo: Como Obter Parâmetros da Query String com JavaScript

Javascript - Formulário Filho

Javascript - Buscar o Json com o resultado e todos os campos de uma tabela filha

```
//o parametro 'f43330be-0467-7fee-1168-28c9e6d185f0' é o Id da Tabela
var json = formContext.childForm.datatable.getResultJson('f43330be-0467-7fee-1168-28c9e6d185f0');
```

Javascript - Atualizar uma Tabela Filha

```
//o parametro 'f43330be-0467-7fee-1168-28c9e6d185f0' é o Id da Tabela
formContext.childForm.datatable.refresh('f43330be-0467-7fee-1168-28c9e6d185f0');
```

Javascript - Buscar o total de itens em uma tabela filha

```
//o parametro 'f43330be-0467-7fee-1168-28c9e6d185f0' é o Id da Tabela
var total = formContext.childForm.datatable.totalRows('f43330be-0467-7fee-1168-28c9e6d185f0');
```

Javascript - Refazer a ordenação dos itens quando a tabela filha estiver liberada a ordenação manual via "Drag and Drop" (Arrastar e soltar)

```
//reordena os itens da tabela filha de acordo com a ordem dos ids passados no parametro @items_sorted
//o primeiro parametro, 'f43330be-0467-7fee-1168-28c9e6d185f0' é o Id da Tabela
// o segundo parametro "items_sorted" é a lista ordenada com os ids dos elementos que pertencem a tabela:
//[
//    { formid: '', draftid: '' },
//    { formid: '', draftid: '' },
//    { formid: '', draftid: '' },
//]
formContext.childForm.datatable.setOrder('f43330be-0467-7fee-1168-28c9e6d185f0', items_sorted);
```

Javascript - Buscar todos os ids: "Id do Formulário" e "Id do Rascunho" dos itens listados em uma tabela filha

```
//o parametro 'f43330be-0467-7fee-1168-28c9e6d185f0' é o Id da Tabela
var ids = formContext.childForm.datatable.getAllChildIds('f43330be-0467-7fee-1168-28c9e6d185f0');



o restorno será nesse formato:
 [
   {
      "formid": "5451ae4e-290a-4887-8099-6e9a026c0283",
      "draftid": ""
```

```
    },
    {
        "formid": "6c55eeef-427b-40ce-b415-dab11b4aa039",
        "draftid": ""
    },
    {
        "formid": "304cc8e2-47af-44d4-ba83-c0e4939e21fb",
        "draftid": ""
    }
]
```

Javascript - Disparo de Eventos na tabela Filha

Objeto	**Nome / evento**	**Exemplo de uso**

Tabela de Dados do Formulário Filho. Que fica no Formulário Pai.	before-data-load
	/
	Dispara antes de carregar os dados na tabela
		//id = id da tabela filha
		$('#id').on('before-data-load', function (e) {
		console.log('evento disparado antes de carregar os dados')
		});

Tabela de Dados do Formulário Filho. Que fica no Formulário Pai.	data-loaded
	/
	Dispara depois de carregar os dados na tabela
		//id = id da tabela filha
		$('#id').on('data-loaded', function (e, result_json) {
		console.log('evento disparado depois de carregar os dados')
		console.log('result_json',result_json)
		});

Javascript - Biblioteca

Abaixo estão listadas as biblioteca javascript que já estão importadas na página do formulário:

bootstrap-daterangepicker/daterangepicker.js

bootstrap-daterangepicker/moment.min.js
bootstrap-select/bootstrap-select.js
bootstrap/bootstrap.js
guid.js
jquery.inputmask/4.0.0-beta.19/jquery.inputmask.bundle.js
jquery.mask.js
jquery.maskMoney.js
jquery.validate.js
jquery.webui-popover.js
jquery/jquery.js

Criar Html (Componente Cshtml):

Abra o formulário que deseja customizar. Na barra superior desse formulário clique em **"Visual Code Editor"** como mostra figura abaixo:

Para criar um html, clique no botão "**Novo Cshtml**", como na figura abaixo:

Ao clicar, digite o nome do Html

Agora o novo html está criado, e será mostrado para edição no menu do lado esquerdo, como na imagem abaixo.

Clique no nome do Html para edita-lo.

No arquivo Html, é possível utilizar qualquer tag Html que você precisar. Para embelezar o seu html utilize a linguagem css na área "Style (CSS)", e para customizar com javascript, faça as customizações na área "Javascript".

Faça as edições necessárias, salve e posicione esse novo componente no seu formulário. (O passo a passo para posicionar o componente no formulário está descrito mais abaixo)

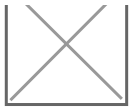
Posicionando o componente Html no formulário

Depois de criado o componente cshtml, salve todas as configurações e customizações realizadas no Visual Code Editor, através do botão  .

Agora volte para a edição do formulário e clique no botão 

No box de opções de campos, o seu componente "ExemploHtml" já deverá estar listado, como mostra a imagem abaixo:

Clique no nome do componente, e logo em seguida em salvar 

Agora você deve posiciona-lo na tela, para isso basta clicar na opção  como mostra imagem abaixo.

Ao abrir a opção de montagem de tela, posicione e arraste o seu componente onde você preferir.

Agora salve e publique esse novo formulário! Pronto, o seu componente está criado.

Exemplo prático de customização html

No exemplo a seguir vamos apresentar uma mensagem em azul de bom dia ou boa tarde ou boa noite dependendo do horário atual.

Para descobrir qual a data e qual a mensagem a ser apresentada, usamos C# no cshtml.

Para deixar a mensagem azul, usamos CSS.

Nesse exemplo não utilizamos javascript para nenhuma ação, mas poderia ser utilizado caso fosse necessário.

Passo 1 - Definir o html

Crie o cshtml como descrito aqui.

Agora cole o código abaixo nesse novo html:

```
@{
    string parteDoDia;
    var hours = DateTime.Now.Hour;
    if (hours > 16)
    {
        parteDoDia = "Boa noite";
    }
    else if (hours > 11)
    {
        parteDoDia = "Boa tarde";
```

```
    }
    else
    {
        parteDoDia = "Bom dia";
    }
}

<div class="painel-msg-customizada">
    <span>@parteDoDia,</span>
</div>
```

Passo 2 - Definir o CSS

Entre na área de CSS e cole o código abaixo:

```
.painel-msg-customizada{
    display: block;
    text-align: left;
    padding-top: 10px;
}

.painel-msg-customizada span{

    color: #0281ff;
    display: block;
    font-size: 17px;
    font-weight: 500;
    border-bottom: solid 2px #a4d2ff;
    margin-bottom: 15px;
}
```

Passo 3 - Posicionar o html no formulário

Salve tudo no Visual Code Editor e posicione esse componente html, como descrito aqui.

Passo 4 - Salvar e publicar o formulário

Salve e publico o novo formulário.

Agora acesse esse formulário, o resultado deve ser igual ao print abaixo:

C# no componente CSHTML

Se você precisar de alguma informação do formulário, utilize a opção abaixo

C# - Propriedades

string FormId	Id do registro, pode ser em branco ou nulo qdo for um rascunho
bool IsRascunho {get;}	Retorna true caso a linha seja um Rascunho
string DraftId	Id do rascunho

C# - Recuperar (Get) valor dos campos

string GetValue(string idColuna)	retorna o valor do campo
int? GetInt(string idColuna)	Retorna o valor do campo no tipo INT (inteiro), caso esse campo no formulário seja um número, se o valor for 10.000,99 Será retornado: 10000
decimal? GetDecimal(string idColuna)	Retorna o valor do campo no tipo Decimal, caso esse campo no formulário seja um número, se o valor for 10.000,99 Será retornado: 10000.99
DateTime? GetDateTime(string idColuna)	Retorna o valor do campo no tipo DateTime
Guid? GetGuid(string idColuna)	Retorna o valor do campo no tipo Guid

string GetText(string idColuna)	No caso de campos que são Lista Dinanica, exemplo Combo, Auto completo, radio etc. No Json de Campos Preenchidos no Formulario, só terá o Value dos campos Esse método retornará o Texto desse campo. Exemplo: Existe no formulário um campo chamado 'Produto', com o id definido como 'produto'. Esse campo é do tipo 'AutoComplete' e listará os 'Produtos' por 'Nome'. Quando o usuário salvar o formulário, no Json de envio não retornará o Nome do Produto, apenas o Id do produto selecionado (Esse id estará no formato de um GUID). Para recuperar o Nome do produto, vc precisará executar esse método da seguinte forma: var nomeProduto = PageContext.GetText("produto")

Variável	**Tipo**	

(**deprecated)**
~~Model.FormularioCamposPreenchidos~~
	~~(**deprecated)**~~
	~~dynamic (Json)~~
		(**deprecated**: utilizar o método PageContext.GetValue descrito mais acima)
		~~Retorna um json com os campos preenchidos no formulário, exemplo:~~
		~~{~~
		~~"campo1": "xxxxx",~~
		~~"campo2": "yyyyy",~~
		~~"campo3": "zzzzz",~~
		~~}~~
		
		~~Para resgatar a informação do campo1, utilize:~~
		`@{ var campo1 = Model.FormularioCamposPreenchidos["campo1"]; } <div> @campo1 </div>`
		
		~~Para resgatar o id do formulário, utilize:~~
		`@{ var idFormulario_Forma1 = Model.Id; var idFormulario_Forma2 = Model.FormularioCamposPreenchidos["id"]; } <div>@idFormulario_Forma1</div> <div>@idFormulario_Forma2</div>`

(deprecated)
~~Model.Id~~
 	
~~OU~~
 	
~~Model.FormularioCamposPreenchidos["id"];~~
	(deprecated)
	~~Guid~~
	
	~~OU~~
	
	~~string~~
		(deprecated)
		~~Para resgatar o id do formulário, utilize:~~
		`@{`
		`var idFormulario_Forma1 = Model.Id;`
		`var idFormulario_Forma2 = Model.FormularioCamposPreenchidos["id"];`
		`}`
		
		`<div>@idFormulario_Forma1</div>`
		`<div>@idFormulario_Forma2</div>`

Model.FormularioTemporarioId	Guid	Para resgatar o id temporario do formulário, utilize:
		`@{`
		`var formularioTemporarioId = Model.FormularioTemporarioId;`
		`}`
		
		`<div>@formularioTemporarioId</div>`

(deprecated) ~~Model.RascunhoId~~	**(deprecated)** ~~Guid~~	**(deprecated)** ~~Para resgatar o id do rascunho, caso seja um~~
Model.EstruturaFormularioId	Guid	Para resgatar o id da ESTRUTURA do formulário

Quando estiver dentro do Formulário Filho: Model.FormularioIdPai	Guid	Para resgatar o id do formulário pai, quando estiver dentro de um formulário filho, utilize:
		`@{ var formularioIdPai= Model.FormularioIdPai; }`
		`<div>@formularioIdPai</div>`

Quando estiver dentro do Formulário Filho: Model.FormularioTemporarioIdPai	Guid	Para resgatar o id temporário do formulário pai, quando estiver dentro de um formulário filho, utilize:
		`@{ var formularioTemporarioIdPai= Model.FormularioTemporarioIdPai; }`
		`<div>@formularioTemporarioIdPai</div>`

Quando estiver dentro do Formulário Filho: Model.EstruturaFormularioIdPai	Guid	Para resgatar o id temporário do formulário pai, quando estiver dentro de um formulário filho, utilize:
		`@{ var estruturaFormularioIdPai= Model.EstruturaFormularioIdPai; }`
		`<div>@estruturaFormularioIdPai</div>`

WFS.Common.AppSettings.StaticFilePath	string	Retorna a url dos arquivos estaticos do agilityflow, exemplo, .js, .css, etc.. caso queira importar um arquivo JS que já exista na biblioteca de arquivos estaticos do agilityflow <script src="**@WFS.Common.AppSettings.StaticFilePath**/js/xxxxx/yyyyyy.js?**@WFS.Common.AppSettings.StaticFileVersion**"></script>
WFS.Common.AppSettings.StaticFileVersion	string	Utilize essa variável para limpar o cache do seu arquivo estático, sendo assim coloque ela como parametro de querystring do seu arquivo, exemplo: caso queira importar um arquivo JS que já exista na biblioteca de arquivos estaticos do agilityflow <script src="@WFS.Common.AppSettings.StaticFilePath/js/xxxxx/yyyyyy.js?**@WFS.Common.AppSettings.StaticFileVersion**"></script>

C# no Componente CSHTML - Testar se é umnovo formulário ou se é uma edição

Para saber se o formulário é um novo ou está para edição, apenas teste a variável de ID, como no exemplo abaixo:

Forma 1

```
@{
  var isNovoFormulario = GuidHelper.IsNullOrEmpty(Model.Id);
}
```

Forma 2

```
@{
var isNovoFormulario = Model.Id.IsNullOrEmpty();
}
```

Exemplo

```
@{
  var msg = "";
  var formularioId = Model.Id;

    if(formularioId.IsNullOrEmpty()){
      msg = "é um novo formulario";
    }else{
      msg = "NÃO é um novo formulario";
    }

}
<div>@msg</div>
```

C# no componente CSHTML - Testar se é um rascunho

```
@{
  var msg = "";
  var rascunhoId = Model.RascunhoId;

    if(rascunhoId.IsNullOrEmpty()){
      msg = "é um Rascunho";
    }else{
      msg = "NÃO é um Rascunho";
    }

}
<div>@msg</div>
```

Criar uma Tabela de dados customizada através de programação: Query SQL, C#, HTML e CSS:

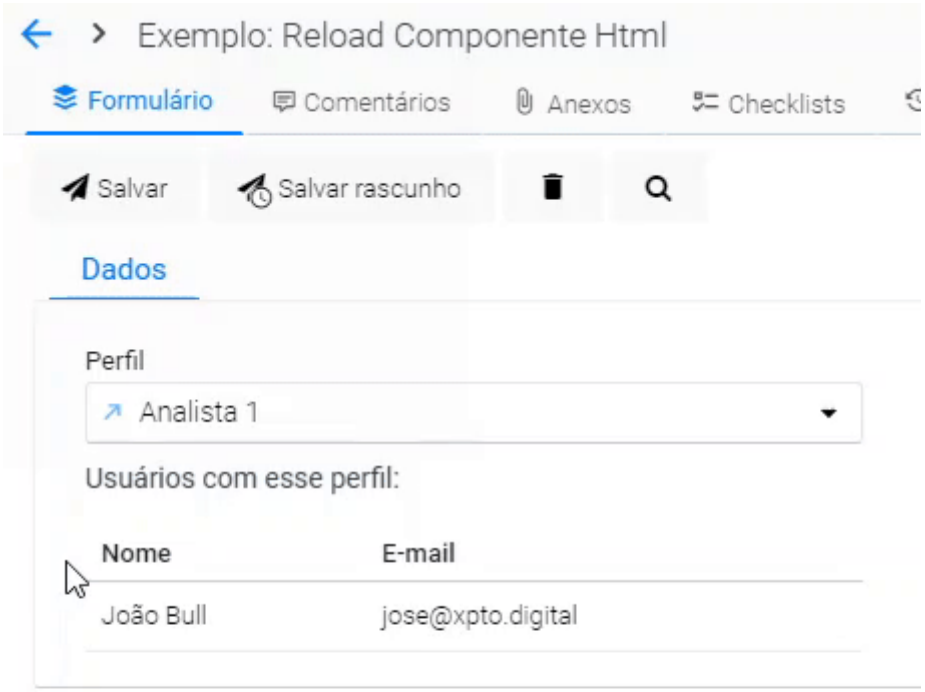
Abaixo está um exemplo de criação de uma tabela de dados buscando as informações através de uma Query SQL e apresentando no Html:

Para customizar o CSS e Javascript da tabela de dados, utilize áreas de customização já citadas nos itens acima

```
<h5 id="datatableTitle">Tasks</h5>
<a id="btnAdd" href="javascript:pageNavigation.openSimpleLightBox({  islookup: true,
url:'@Model.GetBaseUrl()/fluxo/index/7f786f15-d644-4351-8763-46bc2923fd21' })"><i class="mdi mdi-
plus"></i> Add task</a>
<table id="tasksTable" border="1">


            <tr>
               <th width="140px">Status</th>
               <th>Team</th>
               <th>Description</th>
               <th>CreatedDate</th>
               <th>ModifiedDate</th>
            </tr>


@{
   var applicationId = PageContext.FormId;

   /* concatenando o parametro */
   var dt = await PageContext.GetDataTableAsync(@"select
                                 isnull(task.id,'') TaskId,
                                 isnull(task.description,'') TaskDescription,
                                 isnull(team.name,'') Team,
                                 task.log_data_criacao CreatedDate,
                                 task.log_data_alteracao ModifiedDate,
                                 isnull(etp.etp_nome,'') CurrentStep,
                                 frm_status_fluxo StatusFlow
                                 from
```

```
                                x_tbl_application_task2 task
                            inner join tbl_formulario form
                                on task.id = form.frm_id
                            left join tbl_formulario_etapa etp
                                on form.etp_id_atual = etp.etp_id and etp.frm_id = form.frm_id
                            left join x_tbl_team team
                                on task.team = team.id

                            where
                            task.deletado = 0
                            and task.id_application = '"+applicationId+"' order by  task.log_data_alteracao desc,
task.log_data_criacao desc ");



    /* passando como parametro */
    /*var paramsQuery = new List<NpgsqlParameter>();
    paramsQuery.Add(new NpgsqlParameter("@applicationId", applicationId){ NpgsqlDbType =
NpgsqlTypes.NpgsqlDbType.Uuid });
    var dt = await PageContext.GetDataTableAsync(@"select
                            isnull(task.id,'') TaskId,
                            isnull(task.description,'') TaskDescription,
                            isnull(team.name,'') Team,
                            task.log_data_criacao CreatedDate,
                            task.log_data_alteracao ModifiedDate,
                            isnull(etp.etp_nome,'') CurrentStep,
                            frm_status_fluxo StatusFlow
                            from
                                x_tbl_application_task2 task
                            inner join tbl_formulario form
                                on task.id = form.frm_id
                            left join tbl_formulario_etapa etp
                                on form.etp_id_atual = etp.etp_id and etp.frm_id = form.frm_id
                            left join x_tbl_team team
                                on task.team = team.id

                            where
                            task.deletado = 0
                            and task.id_application = @applicationId order by  task.log_data_alteracao desc,
task.log_data_criacao desc ", paramsQuery.ToArray());*/
```

```
    foreach (DataRow dr in dt.Rows)
    {

        var ModifiedDate = dr["ModifiedDate"] == DBNull.Value ? "" : dr["ModifiedDate"].ToString();
        var currentStep = dr["CurrentStep"].ToString();
        var statusFlow = dr["StatusFlow"].ToString();

        if(statusFlow == "A"){
            currentStep = "APPROVED";
        }else if(statusFlow == "X"){
            currentStep = "REJECTED";
        }

        <text>
            <tr class="task-row status-@currentStep" data-taskid='@dr["TaskId"].ToString()'>
                <td><span class="st">@currentStep</span></td>
                <td>@dr["Team"].ToString()</td>
                <td>@dr["TaskDescription"].ToString()</td>
                <td>@dr["CreatedDate"].ToString()</td>
                <td>@dr["ModifiedDate"].ToString()</td>
            </tr>
        </text>

    }

}
</table>
```



Recarregar no servidor um componente Html ao alterar uma determinada informação do formulário:

Você pode determinar alguns momentos para que as informações de um componente html sejam recarregadas do servidor.

Essa é uma opção vantajosa em situações em que você tenha consultas (queries) de banco de dados no dentro do componente Html
e deseja que essa consulta seja atualizada em alguns momentos.

> Apenas campos do tipo "Lista de seleção (Combo)" e "Pesquisa com auto completar" poderão disparar o recarregamento do Componente Cshtml

Exemplo para recarregar o componente cshtml

Suponhamos que você crie um formulário que contenha 2 campos, o **primeiro** é um combo de **Perfil de Acesso** no sistema e o **segundo** campo é um componente CsHtml que contenha uma **tabela que mostre todos os usuários** que possuam o perfil selecionado no **primeiro** campo de **Perfil de Acesso**. Então, todas vez que o campo **Perfil de acesso** for alterado, a tabela de usuários deve ser recarregada para mostrar só os usuários relacionados ao novo perfil alterado.

Para esse exemplo, crie um novo formulário. Nesse novo formulário, crie um combo chamado **Perfil** e abaixo crie um componente CShtml que apresente uma tabela com todos os usuários do perfil selecionado (copiar e colar código abaixo no novo componente).

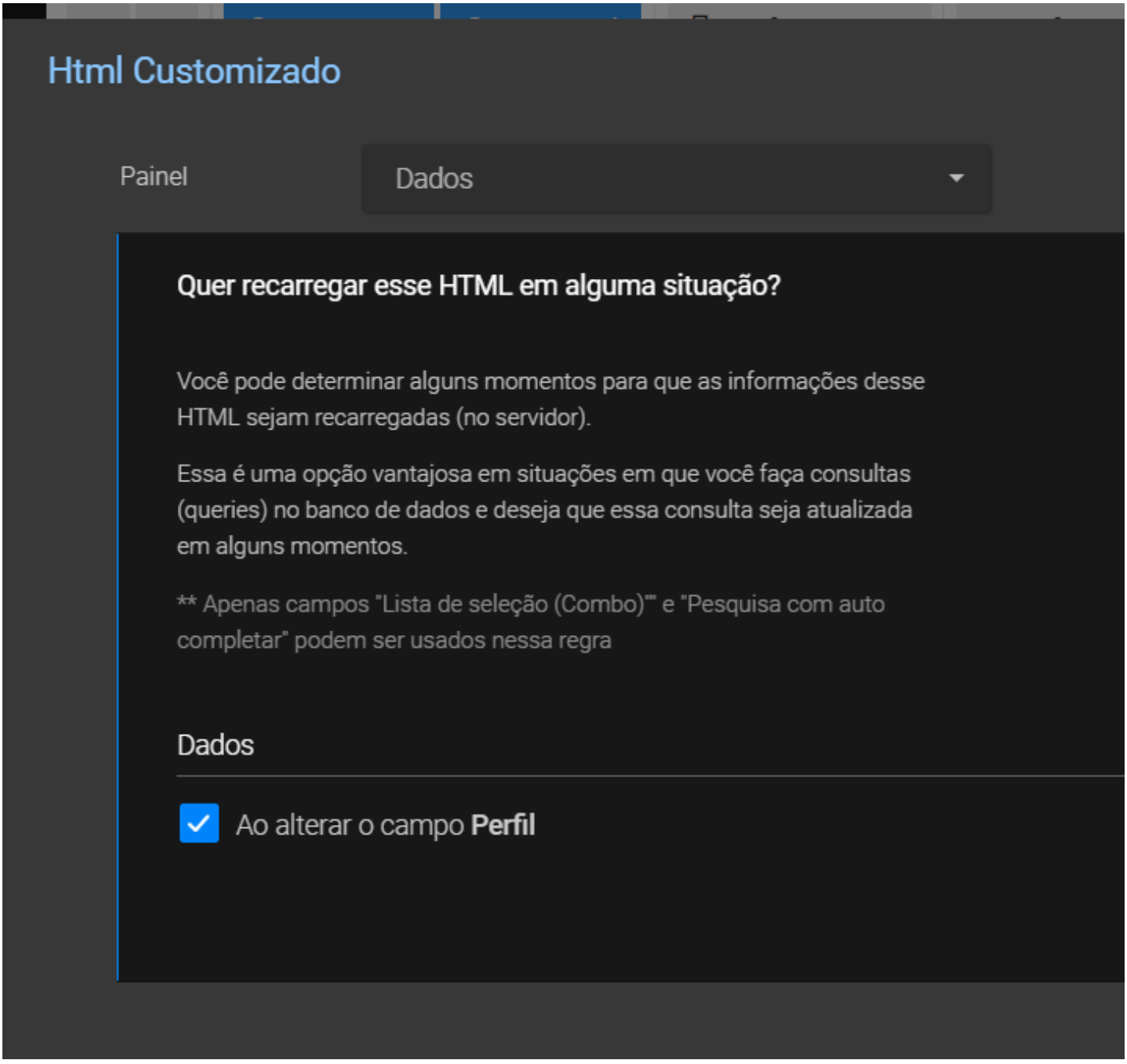
Código para o Componente Cshtml que listará os usuários

```
<h5 id="datatableTitle">Usuários com esse perfil:</h5>
<table id="usuTable" border="1">

            <tr>
               <th width="140px">Nome</th>
               <th>E-mail</th>
            </tr>

@{
   var perfilId = Model.FormularioCamposPreenchidos["perfil"];

   if(perfilId != null && perfilId.ToString() != string.Empty){

      //na query será utilizado no WHERE o id do perfil selecionado no combo
      var query = @"select id, isnull(usu_nome,'') Nome, isnull(usu_email,'') Email  from tbl_usuario u inner join tbl_perfil_usuario pu on u.id = pu.id_tbl_usuario and pu.deletado = 0 where u.deletado = 0 and pu.id_tbl_perfil = '"+perfilId+"' ";
      var dt = await PageContext.GetDataTableAsync(query);

      foreach (DataRow dr in dt.Rows)
      {

            <text>
               <tr class="task-row" data-taskid='@dr["id"].ToString()'>
                  <td>@dr["Nome"].ToString()</td>
                  <td>@dr["Email"].ToString()</td>
               </tr>
            </text>
```

```
    }

  }

}
</table>
```

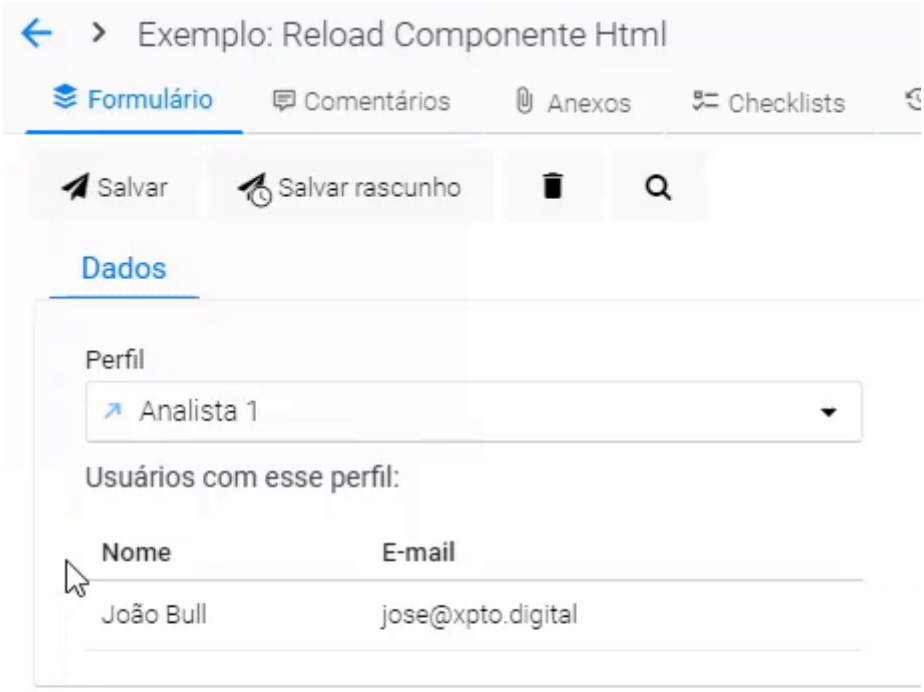
Agora adicione esse componente na tela abaixo do campo Perfil.

Clique para editar o componente e marque o checkbox "Ao alterar o campo Perfil", como na imagem abaixo.



Pronto, agora salve e acesse esse novo formulário criado.

Tente alterar o valor do combo Perfil, se tudo correr bem a tabela de usuários deverá ser recarregada toda vez que o campo perfil for alterado.



C# - Adicionar e Atualizar os itens a uma tabela de formulário filho:

Para esse exemplo, será necessário Associar um Formulário Filho a um Formulário Pai, considere que nesse exemplo o Formulário Filho tenha um campo de texto chamado "nome_filho".

O exemplo abaixo foi colocado dentro de um cshtml do formulário pai e ao criar um novo formulário adicionará um item ao formulário filho e logo em seguida atualizará o item apenas para a demonstração das funções

> Caso a informação a ser adicionada ou atualizada seja um número, enviar sem separador de milhar e o separador de decimal deve ser "." (ponto), exemplo: para o valor de nove mil e quinhentos, deve ser enviado 9500.00. Para facilitar utilize a função **PageContext.FormatNumberToString_EnglishFormat(999999.99, 2);**

```
@{

//id da tabela
    var tableid_relacaoFormularioPaiFilhoId = Guid.Parse("0dd874a9-9c39-4e4f-acc3-02015a60f6bb"); //table id
DO FORMULÁRIO FILHO

    //info do formulario filho
    var campos_e_valores = new DataDictionary();
    campos_e_valores["nome_filho"] = "Exemplo 01 - " + DateTime.Now;
    campos_e_valores["valor_numerico"] = PageContext.FormatNumberToString_EnglishFormat(999999.99, 2);



//testa se é um form NOVO e se NÂO é RASCUNHO pois nesse exemplo quero inserir apenas para formulário
NOVOS que não sejam RASCUNHOs
    var isNovo = Model.Id.IsNullOrEmpty();
    var isRascunho = !Model.RascunhoId.IsNullOrEmpty();



    if(isNovo && !isRascunho){

      //adiciona na tabela filha
       //retorna o id do rascunho desse formlario filho, pois o ID final só será gerado ao Salvar o Formulario PAi
       var formRascunhoId = PageContext.AddChildForm( tableid_relacaoFormularioPaiFilhoId, campos_e_valores);




        //atualiza o item que acabamos de adicionar, apenas para demonstração
Guid? idFormularioFilho = null;
        campos_e_valores["nome_filho"] = "Exemplo Atualizado 0123456789 - " + DateTime.Now;
        PageContext.UpdateChildForm(
            //se ainda não foi salvo pela primeira vez, o item só terá o Id do Rascunho
            rascunhoId,

            //se já foi salvo pela primeira vez, o item já terá o Id
            //então passa aqui o id (nesse exemplo estamos passando null para atualizar apenas o
```

```
        idFormularioFilho,

        tableid_relacaoFormularioPaiFilhoId,
        campos_e_valores);

}}
}
```

C# - Obter o total de itens em uma tabela de formulário filho:

```
@{
//relacaoFormularioPaiFilhoId não é o id do formulario filho e sim o id do campo que gera ao adicionar o formulario filho ao formulario pai
    var relacaoFormularioPaiFilhoId = Guid.Parse("f43330be-0467-7fee-1168-28c9e6d185f0"); //trocar esse id

    var total = await PageContext.CountChildFormsAsync(relacaoFormularioPaiFilhoId);
}
```

C# - Enviar e Recuperar parâmetros extras na requisição ajax no partial view (cshtml)

Enviando um parâmetro extra via javascript

```
var idsItensPrecosCompostos = formContext.childForm.datatable.getAllChildIds('251c66c9-cb26-f3a8-d014-8a1dd4a6b6aa')
formContext.loadViewComponent("calcular_valor_custo_preco_composto", {
  extraData: { idsItensPrecosCompostos: JSON.stringify(idsItensPrecosCompostos) } ,
  onSuccess: function (response) {
console.log(response)
  }
```

```
})
```

Recuperando um parâmetro extra via C# no CShtml.

Nesse exemplo, dentro do partial view "**calcular_valor_custo_preco_composto**", colocaremos esse código

```
@{

    var json_idsItensPrecosCompostos = Request["idsItensPrecosCompostos"];



}
<div>@json_idsItensPrecosCompostos</div>
```

C# e Javascript - Trabalhando com Json

1. Enviando um parâmetro no formato JSON via Javascript. Para isso no código abaixo transformamos um Array em JSON usando o JSON.stringify(...)

```
var idsItensPrecosCompostos = formContext.childForm.datatable.getAllChildIds('251c66c9-cb26-f3a8-d014-8a1dd4a6b6aa')
formContext.loadViewComponent("calcular_valor_custo_preco_composto", {
  extraData: { idsItensPrecosCompostos: JSON.stringify(idsItensPrecosCompostos) } ,
  onSuccess: function (response) {
console.log(response)
  }
})
```

2. Recuperando um parâmetro JSON via C# e convertendo em Objeto Json

Você poderá utilizar qualquer função que faça parte das Bibliotecas

Newtonsoft.Json
Newtonsoft.Json.Linq

Nesse exemplo, dentro do partial view "**calcular_valor_custo_preco_composto**", colocaremos esse código para transformar o parametro que era string em um objeto dinamico do tipo JSON:

```
@{

    var json_idsItensPrecosCompostos = Request["idsItensPrecosCompostos"];
    dynamic idsItensPrecosCompostos =
Newtonsoft.Json.JsonConvert.DeserializeObject(json_idsItensPrecosCompostos);
}
```

3. Count no JSON

* o count vai existir em caso do JSON estar no formato de uma lista (array).

```
@{

    var json_idsItensPrecosCompostos = Request["idsItensPrecosCompostos"];
    dynamic idsItensPrecosCompostos =
Newtonsoft.Json.JsonConvert.DeserializeObject(json_idsItensPrecosCompostos);
  var total = idsItensPrecosCompostos.Count;
}
```

4. Looping no JSON

* o foreach vai existir em caso do JSON estar no formato de uma lista (array).

```
@{

    var json_idsItensPrecosCompostos = Request["idsItensPrecosCompostos"];
    dynamic idsItensPrecosCompostos =
Newtonsoft.Json.JsonConvert.DeserializeObject(json_idsItensPrecosCompostos);

    var draftids = new List<string>();
    var formids = new List<string>();
    foreach (var idItem in idsItensPrecosCompostos)
    {
        draftids.Add(idItem.draftid.ToString());
        formids.Add(idItem.formid.ToString());
    }
}
```

C# - Transformar Array List em uma string separada por virgula

```
@{
    var itens = new List<string>() { "abacaxi", "mamão", "melão", "banana"};
    var msg = String.Join(",",itens.ToArray());
}
@msg
```

C# - Criar um Json dinamicamente

```
@{
    var list_InfoNomeEmail = new JArray() as dynamic;

  var query = @"select nome, email from x_tbl_xxxxxxxxxxxxx where deletado = 0";
    var dt = await PageContext.GetDataTableAsync(query);
    foreach (DataRow dr in dt.Rows)
    {
      var nome = dr["nome"].ToString();
      var email = dr["email"].ToString();

      //cria o objeto e inclua na lista para depois ser transformado em json
      dynamic infoNomeEmail = new JObject();
      infoNomeEmail.nome = nome;
      infoNomeEmail.email = email;
      list_InfoNomeEmail.Add(infoNomeEmail);
    }



  //devolve os items em formato json
  if(list_InfoNomeEmail.Count > 0){

     //serializa para JSON
     var formata_para_json = JsonConvert.SerializeObject(list_InfoNomeEmail);

     //usa o Html.Raw para garantir a integridade dos caracteres no retorno
```

```
    @Html.Raw(formata_para_json)
  }



}
```

Revision #148
Created 19 March 2019 16:58:51 by agilityflow
Updated 11 March 2025 12:24:33 by agilityflow