

# Alterar via C# o valor de um campo ao salvar um formulário

Caso vc precise aplicar algumas regras como mudar o valor de um campo via C# ao salvar um formulário, você pode aplicar as seguintes regras:

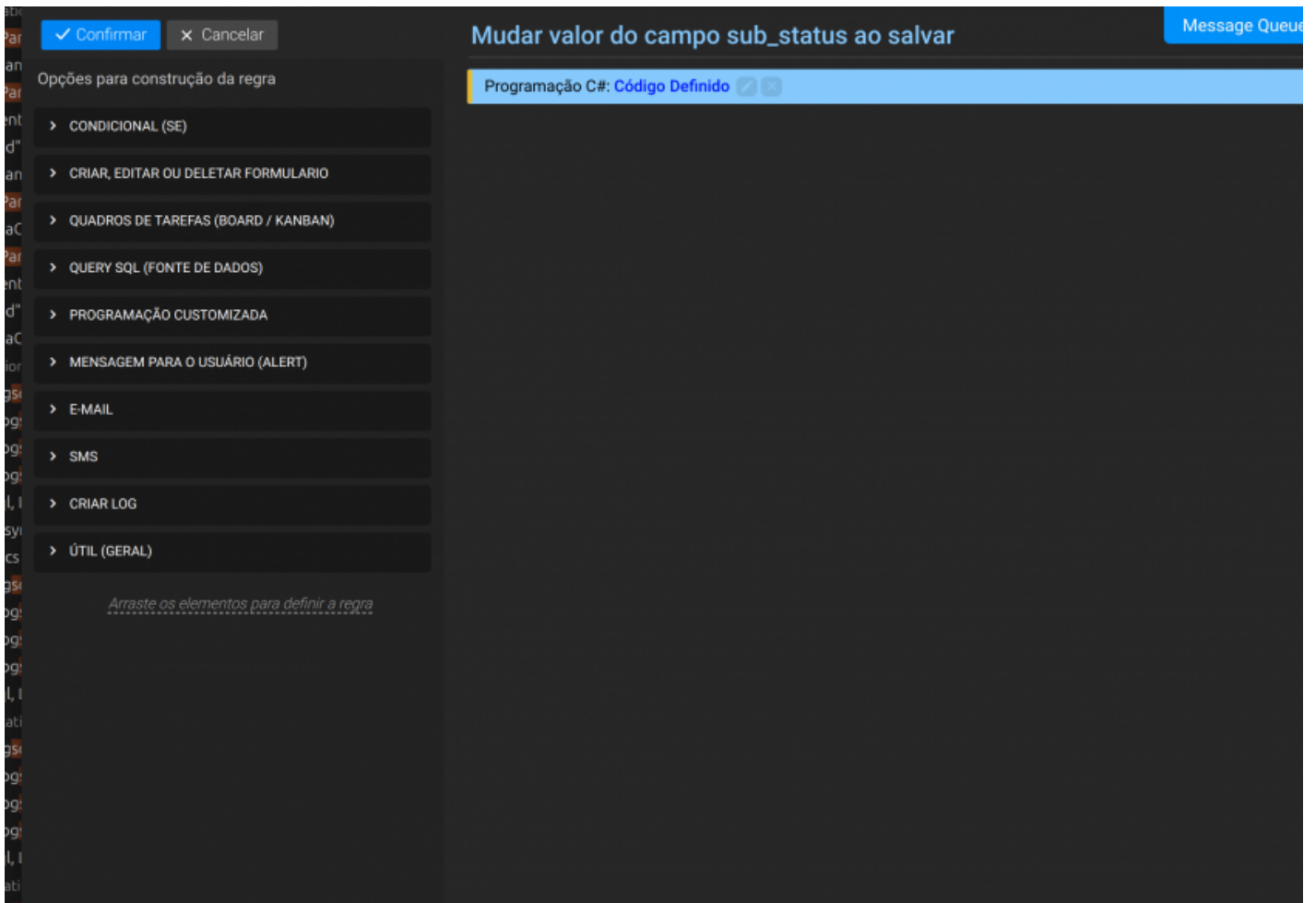
Crie uma regra como na imagem abaixo

The image shows a user interface for configuring business rules. On the left, a sidebar titled "Todas as regras:" contains a list of rules. One rule is highlighted: "Mudar valor do campo sub\_status ao ...". Below the list, there is a note: "Essa regra já está adicionada nessa timeline." and instructions: "Arraste as regras para a timeline" and "A regra só pode ser utilizada 1 vez por timeline".

The main area is titled "Timeline de Regras de Negócio:" and has a dropdown menu set to "Ao clicar no botão 'Salvar'". Below this, a timeline is shown with the following steps:

- Essa timeline executará quando o usuário clicar no botão Salvar
- Depois de validar o preenchimento dos campos e Antes de salvar no banco de dados: A rule card "Mudar valor do campo sub\_status ao salvar" is added here.
- Depois de salvar no banco de dados: A note states "(Considere que aqui a operação de salvar no banco de dados foi concluído com sucesso, porém, a transação (commit) ainda não foi finalizada, qualquer Exception ou Warning acarretará no Rollback no banco de dados)". Below this, there is a placeholder "Arraste as regras aqui".
- Fim

Dentro da regra, adicione uma opção de Programação via C# como na imagem abaixo



Esse é um código de exemplo:

```
// Declaração padrão do método assíncrono chamado ExecuteAsync
public async Task ExecuteAsync(){

    // Verifica se NAO é um formulario novo
    if(!IsNovoFormulario){

        // Declaração de uma variável chamada new_sub_status e atribui a ela o valor enviado pelo usuario naquele
        momento
        var new_sub_status = FormContext.GetValue("sub_status");
        // Declaração de uma variável chamada old_sub_status e inicializa com uma string vazia que colocaremos a
        informação antiga que ja constava do banco de dados
        var old_sub_status = "";
```

```

// Cria uma lista de parâmetros para a consulta SQL (Postgresql)
var paramsQuery = new List<NpgsqlParameter>();
// Adiciona um novo parâmetro NpgsqlParameter à lista com o nome "@frm_id" e o valor IdFormulario
paramsQuery.Add(new NpgsqlParameter("@frm_id", IdFormulario));

// Declaração de uma string contendo a consulta SQL
var sql = "select sub_status from x_tbl_message_queue_sub_status where id = @frm_id limit 1 ";
// Executa a consulta SQL de forma assíncrona e armazena o resultado em um DataTable
var dt = await FormContext.GetDataTableAsync(sql, paramsQuery.ToArray());

// Itera sobre as linhas do DataTable encontrada (sempre será 1, limit 1
foreach (DataRow dr in dt.Rows)
{
    // Verifica se o valor da coluna "sub_status" não é DBNull
    if (dr["sub_status"] != DBNull.Value)
        // Atribui o valor da coluna "sub_status" à variável old_sub_status
        old_sub_status = dr["sub_status"].ToString();
}
//criei aqui uma regrinha simples só pra demonstração:

// Verifica se old_sub_status termina com "1"
if(old_sub_status.EndsWith("1")){
    // Define o valor de "sub_status" no FormContext com new_sub_status concatenado com " 2"
    FormContext.SetValue("sub_status", new_sub_status + " 2");
}else{
    // Define o valor de "sub_status" no FormContext com new_sub_status concatenado com " 1"
    FormContext.SetValue("sub_status", new_sub_status + " 1");
}

} // Fim do bloco if

} // Fim do método ExecuteAsync

```

Revision #2

Created 30 July 2024 18:02:01 by agilityflow

Updated 30 July 2024 18:09:49 by agilityflow