07

Inserindo Campos de Endereço e Dinheiro

Transcrição

Agora é a hora de substituir os campos de endereços de `Funcionario` e `Empresa`.

Na primeira linha do contrato, referente à Empresa, teremos vários campos que serão modificados.

```
static void Main(string[] args)
{
    ViaCEP viaCEP = new ViaCEP();
    var contrato = new
    {
        Empresa = new
        {
            RazaoSocial = "Alura Serviços Hidráulicos Ltda.",
            CNPJ = "23432323000150",
            Endereco = viaCEP.GetEndereco("04101300"),
            Numero = "123 fundos"
        },
        Funcionario = new
        {
            Nome = "Mario Mario",
            CPF = "14538551228",
            RG = "123456789-00",
            Nacionalidade = "italiana",
            EstadoCivil = "casado",
            Endereco = viaCEP.GetEndereco("07091000"),
            Numero = "234"
        },
        Inicio = new DateTime(2018, 1, 1),
        Cargo = "encanador",
        Salario = 3108.45
    };

string documento = @"

EMPREGADOR: {contrato.Empresa.RazaoSocial}, com sede à {contrato.Empresa.Endereco.Logradouro},
{contrato.Empresa.Numero}, {contrato.Empresa.Endereco.Bairro}, CEP {contrato.Empresa.Endereco.CI
{contrato.Empresa.Endereco.Localidade}, {contrato.Empresa.Endereco.UF},
inscrita no CNPJ sob nº {contrato.Empresa.CNPJ}";
```

Modificaremos outros campos relacionados ao **endereço** do Empregado.

```
string documento = @"

EMPREGADO: {contrato.Funcionario.Nome}, {contrato.Funcionario.Nacionalidade},
{contrato.Funcionario.EstadoCivil}, portador da cédula de identidade
R.G. nº {contrato.Funcionario.RG} e CPF/MF nº {contrato.Funcionario.CPF},
residente e domiciliado na {contrato.Funcionario.Endereco.Logradouro},
{contrato.Funcionario.Numero}, {contrato.Funcionario.Endereco.Bairro},
```

```
CEP {contrato.Funcionario.Endereco.CEP}, {contrato.Funcionario.Endereco.Localidade},
{contrato.Funcionario.Endereco.UF}.

{contrato.Empresa.Endereco.Localidade}, (DATA POR EXTENSO)



_____
{contrato.Empresa.RazaoSocial}

_____
{contrato.Funcionario.Nome}";
```

Vamos rodar a aplicação para ver como o texto aparecerá para a empresa e para o funcionário.

Como podemos ver, os campos foram preenchidos corretamente, cada endereço foi consultado de forma correta, e podemos reparar que o CEP já veio formatado para nós.

O que nos resta é terminar de formatar alguns campos, por exemplo o CNPJ, CPF, data de início, e a data por extenso.

Vimos como implementar uma consulta rápida de endereços a partir do CEP, e como colocar partes desse endereço onde nos interessa.

Vamos dar uma olhada no CNPJ que está em `Empresa` :

```
Empresa = new
{
    RazaoSocial = "Alura Serviços Hidráulicos Ltda.",
    CNPJ = "23432323000150",
    .
    .
    .
}
```

Vemos que é somente uma sequência de números sem formatação. Por isso, chamaremos o **formatador** do `Caelum Stella CSharp` . Formataremos o CNPJ e o CPF:

```
Empresa = new
{
    RazaoSocial = "Alura Serviços Hidráulicos Ltda.",
    CNPJ = new CNPJFormatter().Format("23432323000150"),
    .
    .
    .
}

Funcionario = new
{
    Nome = "Mario Mario",
    CPF = new CPFFormatter().Format("14538551228"),
    .
    .
    .
}
```

Ainda temos o `Inicio` que estava vindo com o horário. Colocaremos a formatação com o `ToString()` utilizando a **data simples**. Para a data simples, o código de formatação é "d" minúsculo:

```
Inicio = new DateTime(2018, 1, 1).ToString("d"),
```

Vamos rodar a aplicação. O nosso CNPJ, o CPF e a data de início estão formatados. Não podemos nos esquecer da **data por extenso**.

Para formatar a data por extenso, pegaremos a data de hoje e acrescentaremos o código de formatação para datas por extenso: o "D maiúsculo":

```
{DateTime.Today.ToString("D")}
```

O resultado será esse:

```
São Paulo, terça-feira, 27 de junho de 2017
```

Veremos como colocar o resto das informações, como formatar o salário, e colocá-lo por extenso.

O salário está como Double, e precisamos que seja expresso em reais! Para isso, utilizaremos a classe `Money` do `Caelum Stella CSharp`.

```
Salario = new Money(3108.45)
```

Agora nós vamos substituir as referencias `{contrato.Salario}` por `{contrato.Salario.ToString()}`.

Rodamos a aplicação para ver o resultado. O salário aparece como `R$3.108,45`. E o que falta agora é o `(SALÁRIO POR EXTENSO)`.

Para isso, chamaremos o método **extenso** de salário!

Extenso() é um método que nos permite exibir o valor em reais por extenso. Vamos rodar a aplicação para ver o resultado.

```
R$3.108,45 (três mil cento e oito reais e quarenta e cinco centavos)
```

Abordamos nessa aula como formatar o nosso documento com base no objeto `contrato`. Mostraremos no código C#, como utilizar todas as funcionalidades que vimos até agora da biblioteca `Caelum Stella CSharp`.

Esperamos que tenham gostado, deixe suas dúvidas no fórum, e até a próxima!