

Adicionando Mais Campos Ao Contrato

Transcrição

Vamos incrementar o nosso contrato com mais informações como o CPF e o CNPJ, endereços da empresa e do funcionário, e assim por diante.

Será preciso estruturar melhor o nosso contrato. O contrato tem `Empresa` que contém somente uma string. O que faremos agora é **transformar** `Empresa` em *objeto*! Pois assim, `Empresa` poderá conter mais informações, e não só o nome.

Ela será um novo objeto do tipo **anônimo**, e a string `"Alura Serviços Hidráulicos Ltda."` será parte desse objeto, uma propriedade chamada `RazaoSocial`.

```
static void Main(string[] args)
{
    var contrato = new
    {
        Empresa = new
        {
            RazaoSocial = "Alura Serviços Hidráulicos Ltda.",
        },
        Funcionario = "Mario Mario",
        Inicio = new DateTime(2018, 1, 1),
        Cargo = "encanador",
        Salario = 3108.45
    };
}
```

A partir daqui, colocaremos mais informações em `Empresa`, como um CNPJ **válido**, que pode ser gerado na internet:

```
static void Main(string[] args)
{
    var contrato = new
    {
        Empresa = new
        {
            RazaoSocial = "Alura Serviços Hidráulicos Ltda.",
            CNPJ = "23432323000150"
        },
        Funcionario = "Mario Mario",
        Inicio = new DateTime(2018, 1, 1),
        Cargo = "encanador",
        Salario = 3108.45
    };
}
```

Com o `Funcionario`, aplicaremos a mesma ideia, transformando ele em uma estrutura, para colocar outras informações, como o CPF.

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

Legal, trocaremos essas novas informações no documento.

No final da primeira linha de informações do Empregador, trocaremos "(CNPJ)" por {contrato.Empresa.CNPJ} . Logo embaixo, temos o "(CPF)" do funcionário, e trocaremos por {contrato.Funcionario.CPF} .

Não podemos esquecer de modificar os campos anteriores, pois alteramos a sua estrutura. Antes, Empresa era uma string, agora ela é um objeto. Para acessar o **nome da empresa**, será contrato.Empresa.RazaoSocial . Sobre o nome do funcionário, antes Funcionario era apenas uma string, e agora ele é um objeto. Agora, para acessar o nome do funcionário, o caminho é contrato.Funcionario.Nome .

Não podemos nos esquecer de trocar o acesso a esses objetos no final do contrato, na parte de assinaturas.

O nosso contrato ficará assim:

```
static void Main(string[] args)
{
    var contrato = new
    {
        Empresa = new
        {
            RazaoSocial = "Alura Serviços Hidráulicos Ltda.",
            CNPJ = "23432323000150"
        },
        Funcionario = new
        {
            Nome = "Mario Mario",
            CPF = "14538551228",
            RG = "123456789-00",
            Nacionalidade = "italiana",
            EstadoCivil = "casado"
        }
    }
}
```

```

    },
    Inicio = new DateTime(2018, 1, 1),
    Cargo = "encanador",
    Salario = 3108.45
};

```

```
string documento = @"
```

EMPREGADOR: {contrato.Empresa.RazaoSocial}, com sede à (LOGRADOURO), (NUMERO), (BAIRRO), CEP (CI

EMPREGADO: {contrato.Funcionario.Nome}, {contrato.Funcionario.Nacionalidade}, {contrato.Funcion

Pelo presente instrumento particular de contrato individual de trabalho, fica justo e contratad

Cláusula 1ª - O EMPREGADO prestará ao EMPREGADOR, a partir de {contrato.Inicio} e assinatura de:

Cláusula 2ª - Não haverá expediente nos dias de sábado, sendo prestado a compensação de horário

Cláusula 3ª - O EMPREGADOR pagará mensalmente, ao EMPREGADO, a título de salário a importância (

Cláusula 4ª - Estará o EMPREGADO subordinado a legislação vigente no que diz respeito aos desco

Cláusula 5ª - O prazo de duração do contrato é de 2 (dois) anos, contados a partir da assinatura

Cláusula 6ª - O EMPREGADO obedecerá o regulamento interno da empresa, e filosofia de trabalho d

Como prova do acordado, assinam instrumento, afirmado e respeitando seu teor por inteiro, e fir

(LOCALIDADE), (DATA POR EXTENSO)

```
{contrato.Empresa.RazaoSocial}
```

```
{contrato.Funcionario.Nome}
```

```
(Nome, R.G,Testemunha)
```

```
(Nome, R.G,Testemunha)";
}
```

Muito bem! E o que está faltando ainda nesse documento? Isso mesmo, as informações de endereço da empresa e de endereço do funcionário, o salário por extenso, localidade e a data por extenso.

Essas são informações que podemos obter, criando funções em C#. Mas, como vimos na aula anterior, as informações de endereço, são obtidas através de um serviço da internet, e não só isso, precisamos criar funções para o **salário** por extenso e para a **data** por extenso! Ou seja, é necessário ter um componente de terceiros para nos ajudar. Então, com toda a certeza, usaremos a biblioteca `Caelum.Stella.CSharp` para essas tarefas.

Vamos adicionar o pacote da Caelum, clicando com o direito em `ContratoDeTrabalho`, e depois em "Manage NuGet Package", procurar por caelum, e instalar.

A primeira coisa a fazer é dizer quais serão os endereços da empresa e do funcionário. Para obter o endereço, precisamos do **CEP**. E por isso, adicionaremos `Endereco` como uma propriedade da `Empresa`.

`contrato.Empresa.Endereco` é o objeto que será obtido através da biblioteca da Caelum. Como sabemos, para obter `Endereco`, temos que obter a **instância** de endereço da classe `ViaCEP`.

```
Empresa = new
{
    RazaoSocial = "Alura Serviços Hidráulicos Ltda.",
    CNPJ = "23432323000150",
    Endereco = new ViaCEP().GetEndereco("04101300")
},
```

O método `GetEndereco()` chamado pela classe `Endereco`, nos retornará um tipo que é o próprio endereço. Em seguida, colocaremos o número do endereço

```
Empresa = new
{
    RazaoSocial = "Alura Serviços Hidráulicos Ltda.",
    CNPJ = "23432323000150",
    Endereco = new ViaCEP().GetEndereco("04101300"),
    Numero = "123 fundos"
},
```

Certo, precisamos também, colocar o endereço do nosso funcionário. Para isso, adicionaremos uma nova propriedade dentro de `Funcionario`. `contrato.Funcionario.Endereco` será igual a instância do `ViaCEP` para obter o endereço.

Mas, antes de continuar, precisamos rever um conceito. Em `Empresa`, já instanciamos um objeto `ViaCEP`, e nós temos que instanciar novamente esse objeto para o endereço de `Funcionario`. Entretanto, nós vamos **extrair** esse `new ViaCEP` para uma variável, assim é só chamar essa variável em qualquer lugar.

Para extrair, selecionamos o trecho `new ViaCEP()`, clicamos em "Quick Actions and Refactorings...", e depois introduziremos um local para o `ViaCEP`.

```
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
};
}
```

Agora, precisamos **transferir** essas informações de endereço para o nosso documento. Como faremos isso?

Simplesmente, acessaremos o objeto `Endereco` que está dentro de `Funcionario`, que está dentro de `contrato`.

No próximo vídeo, continuaremos a implementar as informações de endereço dentro do nosso contrato de trabalho.