

API's e Serviços

Desenvolvimento Back-End

Mapeamento de Tabelas

Para esse projeto, **não usaremos o Entity Framework**, usaremos ADO.NET puro para reaproveitar as rotinas já existentes no STM (VB)

Dito isso, não teremos entidades, repositórios, ef config e etc

Para mapeamento, deve ser criada uma classe com exatamente os mesmos nomes e tipos de dados existentes na tabela (usar PL/SQL para consultar)

Exemplo:

Tabela no PL/SQL

```
select * from cargos_cebi
```

Row 1	Fields	Info
CAR_SEQ	41	number(11), mandatory, Chave Primária
CAR_MUN_SEQ	1	number(11), optional, Referência a Tabela de Municípios
CAR_CODIGO	4	number(2), optional, Código do cargo
CAR_DESCRICAO	SECRETARIO	varchar2(50), optional, Descrição do Cargo
CAR_DEBITO	0	number(1), optional, default = 0, Responsabiliza pelos débitos da Empresa?

Classe da Tabela

```

namespace Cebi.Stm.Mobiliario.Domain.TableMaps
{
    7 referências
    public class Cargo
    {
        1 referência
        public int CAR_SEQ { get; set; }
        1 referência
        public int CAR_MUN_SEQ { get; set; }
        0 referências
        public int CAR_CODIGO { get; set; }
        0 referências
        public string CAR_DESCRICAO { get; set; }
        0 referências
        public Int16 CAR_DEBITO { get; set; }
    }
}

```

“ A Classe deverá ser criada em: \Domain\TableMaps

Criação da API e Rotas Essenciais

```

[RoutePrefix("Cargos")]
1 referência
public class CargosController : ApiController
{
    private readonly VariavelGlobal global;
    private readonly string _pkg = "PKG_CARGOS";

    0 referências
    public CargosController()
    {
        global = GetVariavelGlobal.Executar();
    }
}

```

Para garantir o funcionamento das nossas API's, utilizaremos esse padrão em suas criações.

- Uma variável do tipo "VariavelGlobal" que deverá ser instanciada no construtor da nossa classe, através do método: "GetVariavelGlobal.Executar()"
- Uma variável do tipo string, chamada "_pkg" que irá armazenar o nome da package que armazena as rotinas do módulo em questão.

Essas variáveis são essenciais para o funcionamento dos nossos executores genéricos.

Todos os serviços de executores **necessitam** da variável global para funcionar

Rotas

- Campos Mapeados

```
[Route("{id}/Mapeados")]
0 referências
public IActionResult Get(int id)
{
    var srv = new ConsultarCargos(global);
    var cargo = srv.Executar(id);
    return Ok(cargo);
}
```

É necessária uma rota para obter os campos do sistema mapeados, isso acontecerá através de um service de estrutura padrão, como no exemplo a seguir:

```
3 referências
public class ConsultarCargos
{
    private readonly VariavelGlobal global;

    2 referências
    public ConsultarCargos(VariavelGlobal global)
    {
        this.global = global;
    }

    2 referências
    public Cargo Executar(int id)
    {
        var executor = new ExecutorQueryCommand("SELECT * FROM cargos_cebi WHERE CAR_SEQ = " + id);
        var cargo = executor.GetSingle<Cargo>(global);

        return cargo;
    }

    1 referência
    public List<CampoPreenchidoDTO> GetCamposMapeado(int id)
    {
        var cargo = Executar(id);

        var mapeiaCampos = new MapearCamposConsulta(global);

        var campos = mapeiaCampos.Executar<Cargo>(cargo, "CAR");
        return campos;
    }
}
```

No primeiro método, usaremos de um SELECT na tabela em questão, através da chave primária do registro, usando nosso executor de queries.

Já no segundo, usaremos do nosso serviço "MapearCamposConsulta", passando a nossa classe. o retorno do "Executar" e a sigla da nossa tabela.

Essa rota deverá existir em todas as API's

- Pesquisa

```
[HttpPost]
[Route("Pesquisar")]
0 referências
public IActionResult Pesquisar(CargoDTO filtros)
{
    var srv = new PesquisarCargos(global);

    var atividades = srv.Executar(filtros);
    return Ok(atividades);
}
```

Rota utilizada para realizar pesquisas na nossa tela de "lista", deverá ser criado um serviço que utiliza de queries para encontrar os dados desejados, como no exemplo a seguir:

```
public class PesquisarCargos
{
    private readonly VariavelGlobal global;

    1 referência
    public PesquisarCargos(VariavelGlobal global)
    {
        this.global = global;
    }

    1 referência
    public List<Cargo> Executar(CargoDTO filtros)
    {
        var quantidadeRegistros = 50;

        var sql = $"select * from cargos_cebi where rownum <= {quantidadeRegistros}";

        if (filtros.Codigo != 0)
            sql += $"and CAR_CODIGO = {filtros.Codigo}";

        if (!string.IsNullOrEmpty(filtros.Descricao))
            sql += $"and CAR_DESCRICAO like '%{filtros.Codigo}%'";

        var executor = new ExecutorQueryCommand(sql);
        var cargo = executor.GetList<Cargo>(global);

        return cargo;
    }
}
```

Os filtros e consulta deverão ser alterados conforme necessidade.

• Inclusão

Deverá seguir o padrão, recebendo o objeto a ser incluído. A próxima seq. deve ser obtida através do service do exemplo e mapeada a PK do objeto que recebemos, para então realizar a inclusão.

```
[HttpPost]
0 referências
public IActionResult Post(Cargo cargo)
{
    var srvSeq = new ObterProxSeq(_pkg, global);
    var seq = srvSeq.Executar();

    var srv = new IncluirRegistro(_pkg + ".Ins_CARGOS", "CAR", global);

    cargo.CAR_SEQ = seq;
    cargo.CAR_MUN_SEQ = 1;

    srv.Executar<Cargo>(cargo);

    return Ok(seq);
}
```

Revisão #3

Criado 3 abril 2025 21:54:12 por Gustavo Marques

Atualizado 4 abril 2025 12:48:36 por Gustavo Marques