Many-to-many

Adicionar o relacionamento Transporte – Clientes

Na classe Transporte.cs

Adicionar o relacionamento

```
        public virtual ICollection<Cliente> Clientes { get; set; }
```

Na classe Cliente.cs

Adicionar o relacionamento

```
        public virtual ICollection<Transporte> Transportes { get; set; }
```

Mapear no Context

Na classe Context.cs, dentro do método OnModelCreating, adicionar o trecho

```
            modelBuilder.Entity<Cliente>()
                .HasMany<Transporte>(s => s.Transportes)
                .WithMany(c => c.Clientes)
                .Map(cs =>
                {
                    cs.MapLeftKey("ClienteRefId");
                    cs.MapRightKey("TransporteRefId");
                    cs.ToTable("ClienteTransporte");
                });
```

Adicionar migration

```
PM> Add-Migration nn
```

Atualizar o banco de dados

```
PM> Update-Database
```

Adicionar Action em ClientesController

```
        public ActionResult Automoveis(int? id)
        {
            List<Automovel> automoveis = db.Transportes.OfType<Automovel>().OrderBy(c =>
c.Nome).ToList();
            ViewBag.Automoveis = automoveis;

            ViewBag.ClienteID = id;
            return View();
        }
```

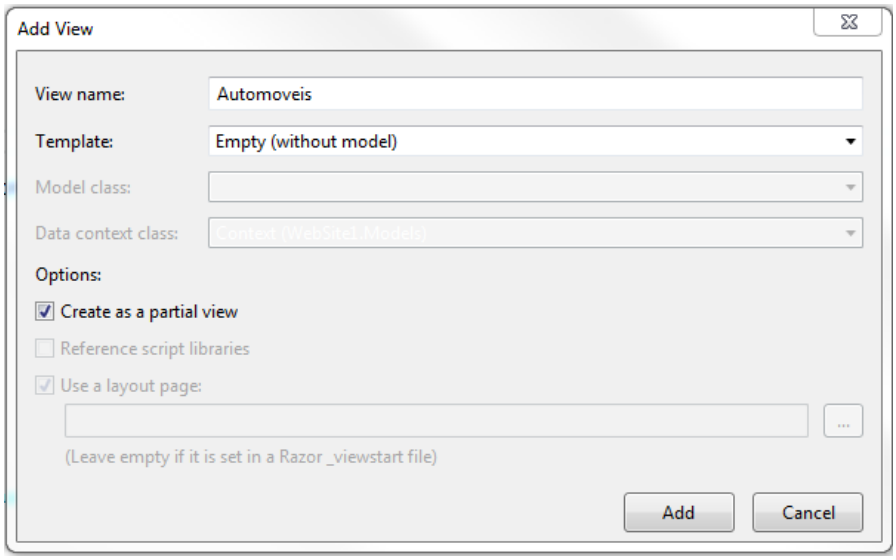
Alterar Views/Clientes/Index.cshtml

Adicionar link para chamada da tela de Automóveis

```
@Html.ActionLink("Edit", "Edit", new { id = item.ClienteID }) |
@Html.ActionLink("Details", "Details", new { id = item.ClienteID }) |
@Html.ActionLink("Delete", "Delete", new { id = item.ClienteID }) |
@Html.ActionLink("Automóveis", "Automoveis", new { id = item.ClienteID })
```

Adicionar uma View para a Action Automoveis

No arquivo ClientesController, botão direito sobre a Action Automoveis (recém criada), Add View…




```
@model WebSite1.Models.Cliente

@{
    ViewBag.Title = "Automóveis do cliente";
}
<h2>Automóveis do Cliente</h2>

@using (Html.BeginForm())
{
    @Html.AntiForgeryToken()
    <div class="form-horizontal">
        <hr />
        @Html.ValidationSummary(true, "", new { @class = "text-danger" })
        @{
    List<WebSite1.Models.Automovel> automoveis = ViewBag.Automoveis;
    foreach (var auto in automoveis)
    {
        <div class="col-md-offset-2 col-md-10 linha">
            <input type="checkbox"
                   name="selectedAutomoveis"
                   value="@auto.TransporteID" />
            @auto.Nome
        </div>
    }
        }

        @Html.HiddenFor(model => model.ClienteID, new { Value = ViewBag.ClienteID })

        <div class="form-group">
            <div class="col-md-offset-2 col-md-10">
                <input type="submit" value="Salvar" class="btn btn-default" />
            </div>
        </div>
    </div>
}
```

```
<div>
    @*@Html.ActionLink("Back to List", "Index")*@
    <a href='@Url.Action("Index", "Clientes")'>
        <img src='@Url.Content("~/Content/Images/voltar.png")' class="botaoicone"
title="Voltar" /> Lista de Clientes
    </a>
</div>

@section scripts{
    <script>
        $(function () {
            $('input, select, textarea').each(function () {
                if ($(this).hasClass('input-validation-error'))
                    $(this).focus();
            });
        });
    </script>
    <script type="text/javascript">
        $(document).ready(function () {
            setTimeout(function () {
                $(".contenterror").fadeOut(1500);
            }, 3000);
        });
    </script>
}
```

Adicionar a Action que receberá os dados
No arquivo ClientesController

```
        [HttpPost]
        [ValidateAntiForgeryToken]
        public ActionResult Automoveis(string[] selectedAutomoveis, int ClienteID)
        {
            if (ModelState.IsValid)
            {

                Cliente cliente = db.Clientes.Include("Transportes").Where(c => c.ClienteID
== ClienteID).FirstOrDefault<Cliente>();

                foreach (var item in selectedAutomoveis)
                {
                    int idTransporte = int.Parse(item);
                    Transporte transporte = db.Transportes.Find(idTransporte);
                    cliente.Transportes.Add(transporte);
                    db.SaveChanges();
                }

                return RedirectToAction("Index");
            }

            List<Automovel> automoveis = db.Transportes.OfType<Automovel>().OrderBy(c =>
c.Nome).ToList();
            ViewBag.Automoveis = automoveis;

            ViewBag.ClienteID = ClienteID;
            return View();
        }
```

Executar a aplicação

Acessar a URL http://localhost:porta/Publico/Logar
Acessar a URL http://localhost:porta/Clientes
Em algum item da lista, clicar no link automóveis
Cadastrar os automóveis para o cliente
Verificar no banco de dados se os dados foram inseridos corretamente.

Alterar os automóveis selecionados

Alterar a Action Automoveis
No arquivo ClientesController

```
        public ActionResult Automoveis(int? id)
        {
            List<Automovel> automoveis = db.Transportes.OfType<Automovel>().OrderBy(c =>
c.Nome).ToList();
            ViewBag.Automoveis = automoveis;


            Cliente cliente = db.Clientes.Include("Transportes").Where(c => c.ClienteID ==
id).FirstOrDefault<Cliente>();
            List<Automovel> meusautos = new List<Automovel>();
            foreach (var item in cliente.Transportes)
            {
                meusautos.Add((Automovel)item);
            }
            ViewBag.MeusAutomoveis = meusautos;


            ViewBag.ClienteID = id;
            return View();
        }
```

Alterar a Views/Clientes/Automoveis.cshtml
Ao ser carregada, já mostrar os automóveis do cliente

```
        ....
        @{
    List<WebSite1.Models.Automovel> meusautos = ViewBag.MeusAutomoveis;

    List<WebSite1.Models.Automovel> automoveis = ViewBag.Automoveis;
    foreach (var auto in automoveis)
    {
        <div class="col-md-offset-2 col-md-10 linha">
            <input type="checkbox"
                   name="selectedAutomoveis"
                   value="@auto.TransporteID"
                   @(Html.Raw(meusautos.Contains(auto) ? "checked=\"checked\"" : "")) />
            @auto.Nome
        </div>
    }


        }
....
```

Executar a aplicação

Acessar a URL http://localhost:porta/Publico/Logar
Acessar a URL http://localhost:porta/Clientes
Em algum item da lista, clicar no link automóveis
Já serão mostrados (marcados) os automóveis do cliente

Agora falta salvar no banco de dados as alterações.

Alterar Action (HttpPost) Automoveis
No arquivo ClientesController

```
        [HttpPost]
        [ValidateAntiForgeryToken]
        public ActionResult Automoveis(string[] selectedAutomoveis, int ClienteID)
        {
            if (ModelState.IsValid)
            {
                int idTransporte = 0;
                Transporte transporte = null;

                Cliente cliente = db.Clientes.Include("Transportes").Where(c => c.ClienteID
== ClienteID).FirstOrDefault<Cliente>();

                List<string> meusautos = new List<string>();
                foreach (var item in cliente.Transportes)
                {
                    meusautos.Add(item.TransporteID.ToString());
                }
                if (selectedAutomoveis == null)
                {
                    foreach (var item in meusautos)
                    {
                        idTransporte = int.Parse(item);
                        transporte = db.Transportes.Find(idTransporte);
                        cliente.Transportes.Remove(transporte);
                    }
                }
                else
                {
                    var selectedAutos = new HashSet<string>(selectedAutomoveis);
                    foreach (var item in selectedAutos)
                    {
                        //BD contém tela?
                        if (!meusautos.Contains(item))
                        {
                            //BD não contém tela, então add tela no BD
                            idTransporte = int.Parse(item);
                            transporte = db.Transportes.Find(idTransporte);
                            cliente.Transportes.Add(transporte);
                        }
                    }
```

```
                    foreach (var item in meusautos)
                    {
                        //tela contém BD?
                        if(!selectedAutos.Contains(item))
                        {
                            //tela nao contem BD, remove do BD
                            idTransporte = int.Parse(item);
                            transporte = db.Transportes.Find(idTransporte);
                            cliente.Transportes.Remove(transporte);
                        }
                    }
                }
                db.SaveChanges();


                //foreach (var item in selectedAutomoveis)
                //{
                //    int idTransporte = int.Parse(item);
                //    Transporte transporte = db.Transportes.Find(idTransporte);
                //    cliente.Transportes.Add(transporte);
                //    db.SaveChanges();
                //}


                return RedirectToAction("Index");
            }

            List<Automovel> automoveis = db.Transportes.OfType<Automovel>().OrderBy(c =>
c.Nome).ToList();
            ViewBag.Automoveis = automoveis;

            ViewBag.ClienteID = ClienteID;
            return View();
        }
```

Executar a aplicação

Acessar a URL http://localhost:porta/Publico/Logar
Acessar a URL http://localhost:porta/Clientes
Em algum item da lista, clicar no link automóveis
Já serão mostrados (marcados) os automóveis do cliente.
Qualquer alteração dos automóveis selecionados, agora será salva no banco de dados.

Refatorar o código

```
[HttpPost]
[ValidateAntiForgeryToken]
public ActionResult Automoveis(string[] selectedAutomoveis, int ClienteID)
{
    if (ModelState.IsValid)
    {
        //http://www.entityframeworktutorial.net/EntityFramework4.3/update-many-to-many-entity-using-dbcontext.aspx

        int idTransporte = 0;
        Transporte transporte = null;

        Cliente cliente = db.Clientes.Include("Transportes").Where(c => c.ClienteID == ClienteID).FirstOrDefault<Cliente>();
        List<string> meusautos = new List<string>();
        foreach (var item in cliente.Transportes)
        {
            meusautos.Add(item.TransporteID.ToString());
        }
        if (selectedAutomoveis == null)
        {
            foreach (var item in meusautos)
            {
                idTransporte = int.Parse(item);
                transporte = db.Transportes.Find(idTransporte);
                cliente.Transportes.Remove(transporte);
            }
        }
        else
        {
            var selectedAutos = new HashSet<string>(selectedAutomoveis);
            foreach (var item in selectedAutos)
            {
                //BD contem tela?
                if (!meusautos.Contains(item))
                {
                    //BD nao contem tela, entao add tela no BD
                    idTransporte = int.Parse(item);
                    transporte = db.Transportes.Find(idTransporte);
                    cliente.Transportes.Add(transporte);
                }
            }
            foreach (var item in meusautos)
            {
                //tela contem BD?
                if(!selectedAutos.Contains(item))
                {
                    //tela nao contem BD, remove do BD
                    idTransporte = int.Parse(item);
                    transporte = db.Transportes.Find(idTransporte);
                    cliente.Transportes.Remove(transporte);
                }
            }
        }
        db.SaveChanges();

        //foreach (var item in selectedAutomoveis)
        //{
        //    int idTransporte = int.Parse(item);
        //    Transporte transporte = db.Transportes.Find(idTransporte);
        //    cliente.Transportes.Add(transporte);
        //    db.SaveChanges();
        //}
        return RedirectToAction("Index");
    }

    List<Automovel> automoveis = db.Transportes.OfType<Automovel>().OrderBy(c => c.Nome).ToList();
    ViewBag.Automoveis = automoveis;
    ViewBag.ClienteID = ClienteID;
    return View();
}
```

1 2 3 4 5 6

Refatoração #1

Comentar as linhas

```
int idTransporte = 0;
Transporte transporte = null;
```

Selecionar o trecho

```
List<string> meusautos = new List<string>();
foreach (var item in cliente.Transportes)
{
    meusautos.Add(item.TransporteID.ToString());
}
```

Botão direito, Refactor > Extract Method…
Nome do método: ListaMeusAutos

Refatoração #2

Selecionar o trecho

```
if (selectedAutomoveis == null)
```

Botão direito, Refactor > Extract Method…
Nome do método: AlgumAutomovelSelecionado

Refatoração #3

Adicionar as linhas abaixo no trecho a seguir

```
int idTransporte = 0;
Transporte transporte = null;
```

Selecionar trecho

```
List<string> meusautos = ListaMeusAutos(cliente);
if (AlgumAutomovelSelecionado(selectedAutomoveis))
{
    foreach (var item in meusautos)
    {
        int idTransporte = 0;
        Transporte transporte = null;
        idTransporte = int.Parse(item);
        transporte = db.Transportes.Find(idTransporte);
        cliente.Transportes.Remove(transporte);
    }
}
```

Botão direito, Refactor > Extract Method…
Nome do método: RemoveAutomoveis

Refatoração #4

Adicionar as linhas abaixo no trecho a seguir

```
int idTransporte = 0;
Transporte transporte = null;
```

Selecionar o trecho …

```
var selectedAutos = new HashSet<string>(selectedAutomoveis);
foreach (var item in selectedAutos)
{
    //BD contem tela?
    if (!meusautos.Contains(item))
    {
        int idTransporte = 0;
        Transporte transporte = null;
        //BD nao contem tela, entao add tela no BD
        idTransporte = int.Parse(item);
        transporte = db.Transportes.Find(idTransporte);
        cliente.Transportes.Add(transporte);
    }
}
```

Botão direito, Refactor > Extract Method…
Nome do método: AdicionarAutomovel

Refatoração #5

Adicionar as linhas abaixo no trecho a seguir

```
int idTransporte = 0;
Transporte transporte = null;
```

Selecionar o trecho …

```
foreach (var item in meusautos)
{
    //tela contem BD?
    if(!selectedAutos.Contains(item))
    {
        int idTransporte = 0;
        Transporte transporte = null;
        //tela nao contem BD, remove do BD
        idTransporte = int.Parse(item);
        transporte = db.Transportes.Find(idTransporte);
        cliente.Transportes.Remove(transporte);
    }
}
```

Botão direito, Refactor > Extract Method…
Nome do método: RemoverAutomovel

Refatoração #6

Remover comentários deste trecho de código

Refatoração #7

Selecionar o trecho…

```
[HttpPost]
[ValidateAntiForgeryToken]
public ActionResult Automoveis(string[] selectedAutomoveis, int ClienteID)
{
    if (ModelState.IsValid)
    {
        //http://www.entityframeworktutorial.net/EntityFramework4.3/update-many-to-many-entity-using-dbcontext.aspx

        Cliente cliente = db.Clientes.Include("Transportes").Where(c => c.ClienteID == ClienteID).FirstOrDefault<Cliente>();
        List<string> meusautos = ListaMeusAutos(cliente);
        if (AlgumAutomovelSelecionado(selectedAutomoveis))
        {
            RemoveAutomoveis(cliente, meusautos);
        }
        else
        {
            var selectedAutos = new HashSet<string>(selectedAutomoveis);
            foreach (var item in selectedAutos)
            {
                AdicionarAutomovel(cliente, meusautos, item);
            }
            foreach (var item in meusautos)
            {
                RemoverAutomovel(cliente, selectedAutos, item);
            }
        }
        db.SaveChanges();
        return RedirectToAction("Index");
    }

    List<Automovel> automoveis = db.Transportes.OfType<Automovel>().OrderBy(c => c.Nome).ToList();
    ViewBag.Automoveis = automoveis;
    ViewBag.ClienteID = ClienteID;
    return View();
}
```

Botão direito, Refactor > Extract Method…
Nome do método: AtualizarAutomoveisCliente

Executar a aplicação

Acessar a URL http://localhost:porta/Publico/Logar
Acessar a URL http://localhost:porta/Clientes
Em algum item da lista, clicar no link automóveis

Código refatorado

```
        [HttpPost]
        [ValidateAntiForgeryToken]
        public ActionResult Automoveis(string[] selectedAutomoveis, int ClienteID)
        {
            if (ModelState.IsValid)
            {
                Cliente cliente = db.Clientes.Include("Transportes").Where(c => c.ClienteID ==
ClienteID).FirstOrDefault<Cliente>();
                List<string> meusautos = ListaMeusAutos(cliente);
                if (AlgumAutomovelSelecionado(selectedAutomoveis))
                {
                    RemoveAutomoveis(cliente, meusautos);
                }
                else
                {
                    AtualizarAutomoveisCliente(selectedAutomoveis, cliente, meusautos);
                }
                db.SaveChanges();
                return RedirectToAction("Index");
            }
            List<Automovel> automoveis = db.Transportes.OfType<Automovel>().OrderBy(c => c.Nome).ToList();
            ViewBag.Automoveis = automoveis;
            ViewBag.ClienteID = ClienteID;
            return View();
        }

        private void AtualizarAutomoveisCliente(string[] selectedAutomoveis, Cliente cliente, List<string>
meusautos)
        {
            var selectedAutos = new HashSet<string>(selectedAutomoveis);
            foreach (var item in selectedAutos)
            {
                AdicionarAutomovel(cliente, meusautos, item);
            }
            foreach (var item in meusautos)
            {
                RemoverAutomovel(cliente, selectedAutos, item);
            }
        }

        private void RemoverAutomovel(Cliente cliente, HashSet<string> selectedAutos, string item)
        {
            if (!selectedAutos.Contains(item))
            {
                int idTransporte = 0;
                Transporte transporte = null;
                idTransporte = int.Parse(item);
                transporte = db.Transportes.Find(idTransporte);
                cliente.Transportes.Remove(transporte);
            }
        }

        private void AdicionarAutomovel(Cliente cliente, List<string> meusautos, string item)
        {
            if (!meusautos.Contains(item))
            {
                int idTransporte = 0;
                Transporte transporte = null;
                idTransporte = int.Parse(item);
                transporte = db.Transportes.Find(idTransporte);
                cliente.Transportes.Add(transporte);
            }
        }
        private void RemoveAutomoveis(Cliente cliente, List<string> meusautos)
        {
            foreach (var item in meusautos)
            {
                int idTransporte = 0;
                Transporte transporte = null;
                idTransporte = int.Parse(item);
                transporte = db.Transportes.Find(idTransporte);
                cliente.Transportes.Remove(transporte);
            }
        }
```

```
private static bool AlgumAutomovelSelecionado(string[] selectedAutomoveis)
{
    return selectedAutomoveis == null;
}

private static List<string> ListaMeusAutos(Cliente cliente)
{
    List<string> meusautos = new List<string>();
    foreach (var item in cliente.Transportes)
    {
        meusautos.Add(item.TransporteID.ToString());
    }
    return meusautos;
}
```