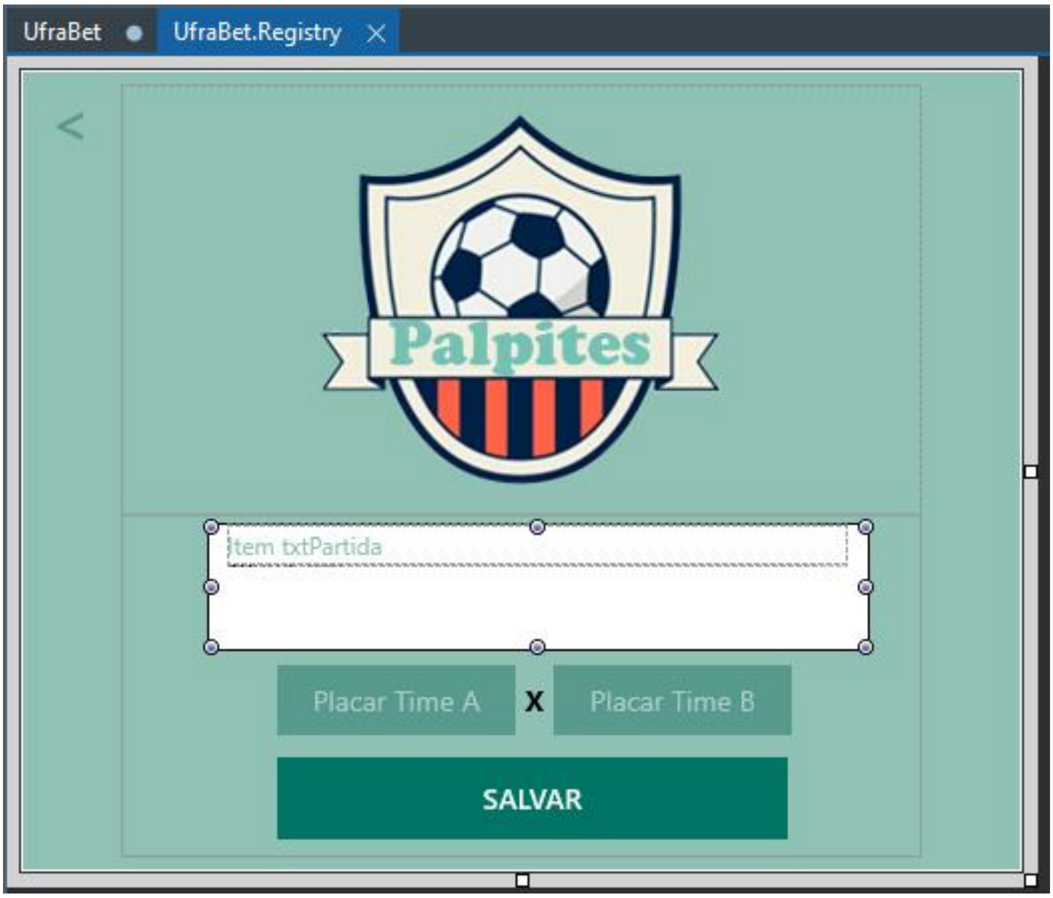
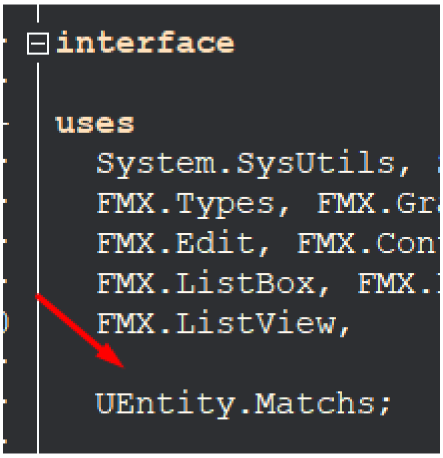
MANUAL UNIT BET.REGISTRY

(Passo a Passo)



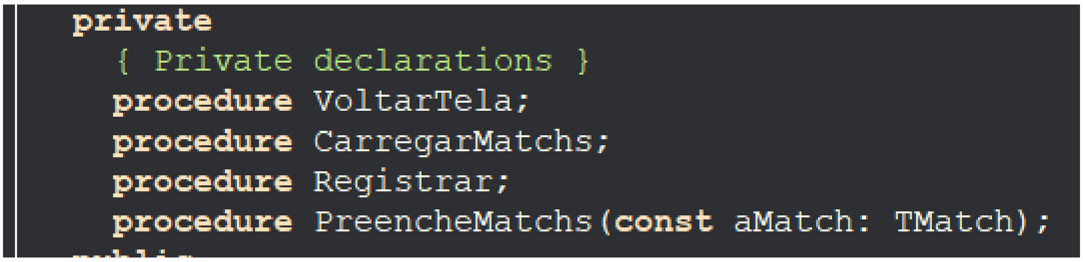
Sumário

1. Uses Interface



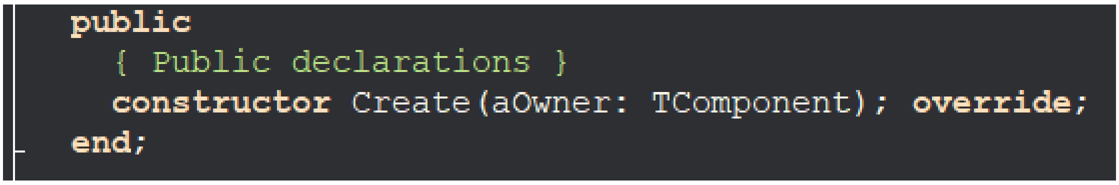
```
interface

uses
  System.SysUtils, 
  FMX.Types, FMX.Gr
  FMX.Edit, FMX.Con
  FMX.ListBox, FMX.
  FMX.ListView,

  UEntity.Matchs;
```

2. Métodos Privados



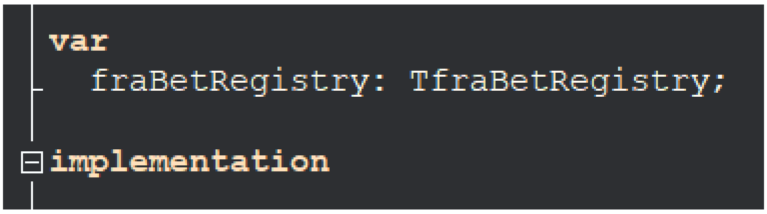
```
  private
    { Private declarations }
    procedure VoltarTela;
    procedure CarregarMatchs;
    procedure Registrar;
    procedure PreencheMatchs(const aMatch: TMatch);
  public
```

3. Métodos Públicos



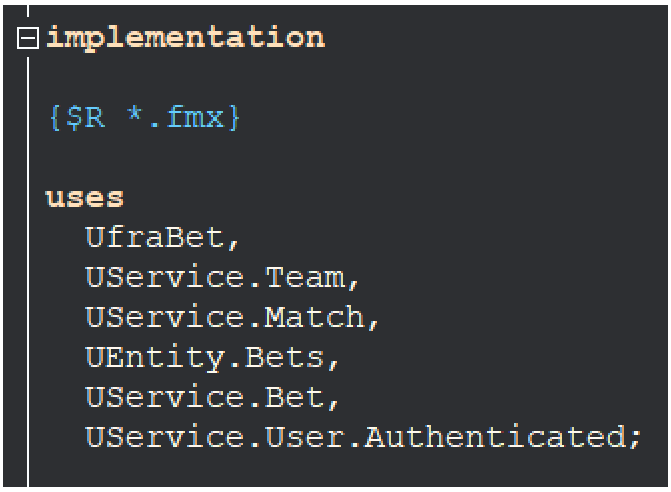
```
  public
    { Public declarations }
    constructor Create(aOwner: TComponent); override;
  end;
```

4. Variável do Frame



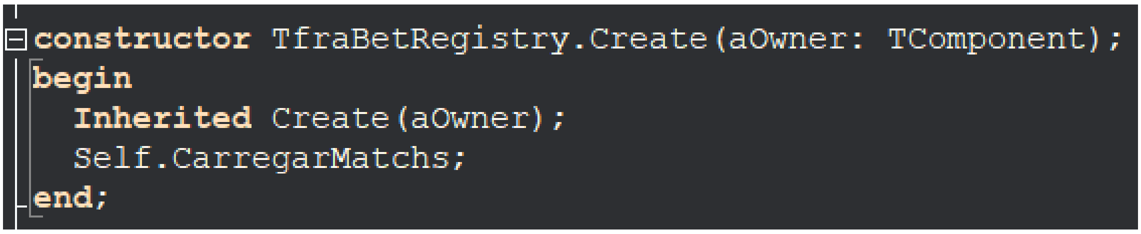
```
var
  fraBetRegistry: TfraBetRegistry;

implementation
```

5. Uses Implementation



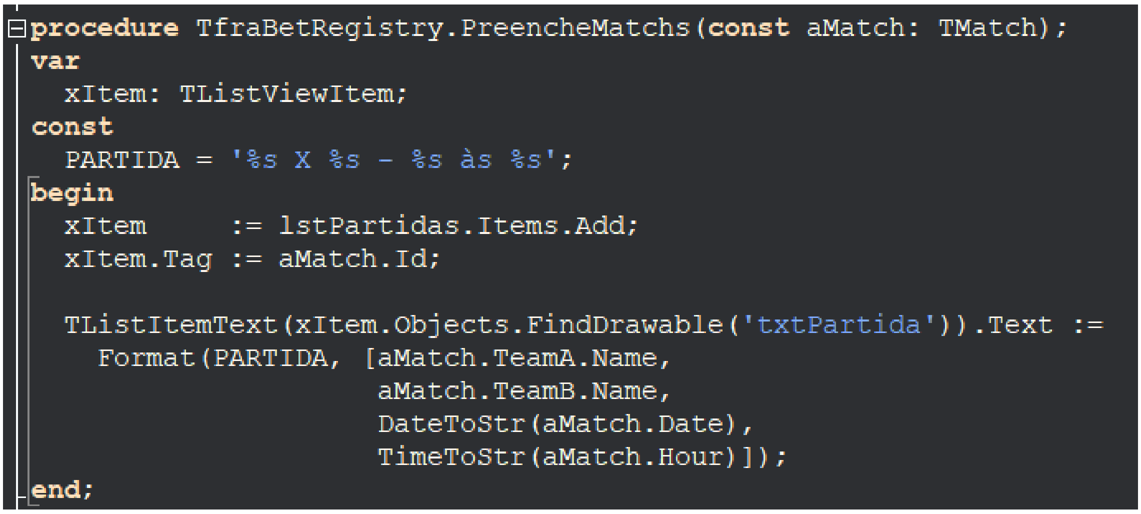
```
implementation

{$R *.fmx}

uses
  UfraBet,
  UService.Team,
  UService.Match,
  UEntity.Bets,
  UService.Bet,
  UService.User.Authenticated;
```

6. Constructor Create



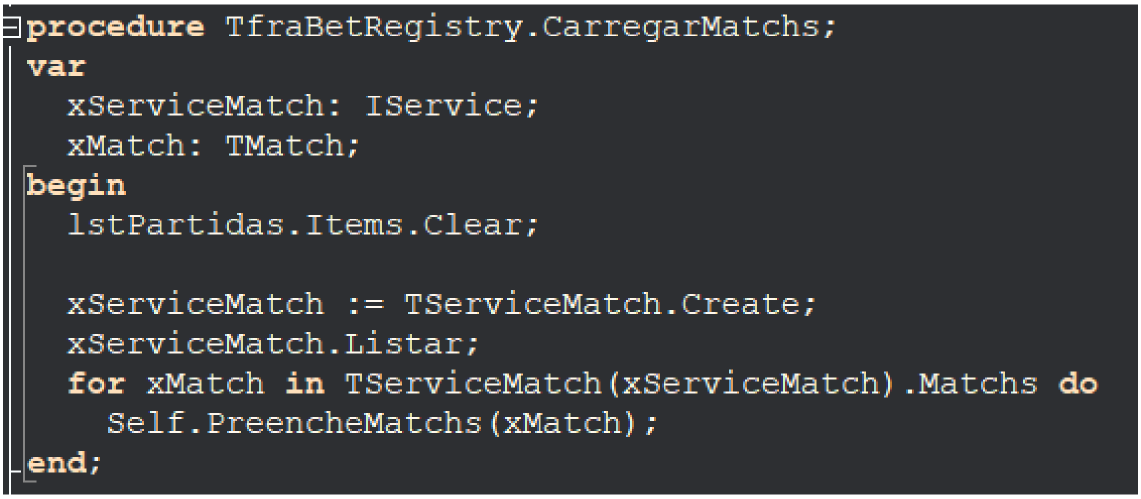
```
constructor TfraBetRegistry.Create(aOwner: TComponent);
begin
  Inherited Create(aOwner);
  Self.CarregarMatchs;
end;
```

7. Método PreencherMatchs



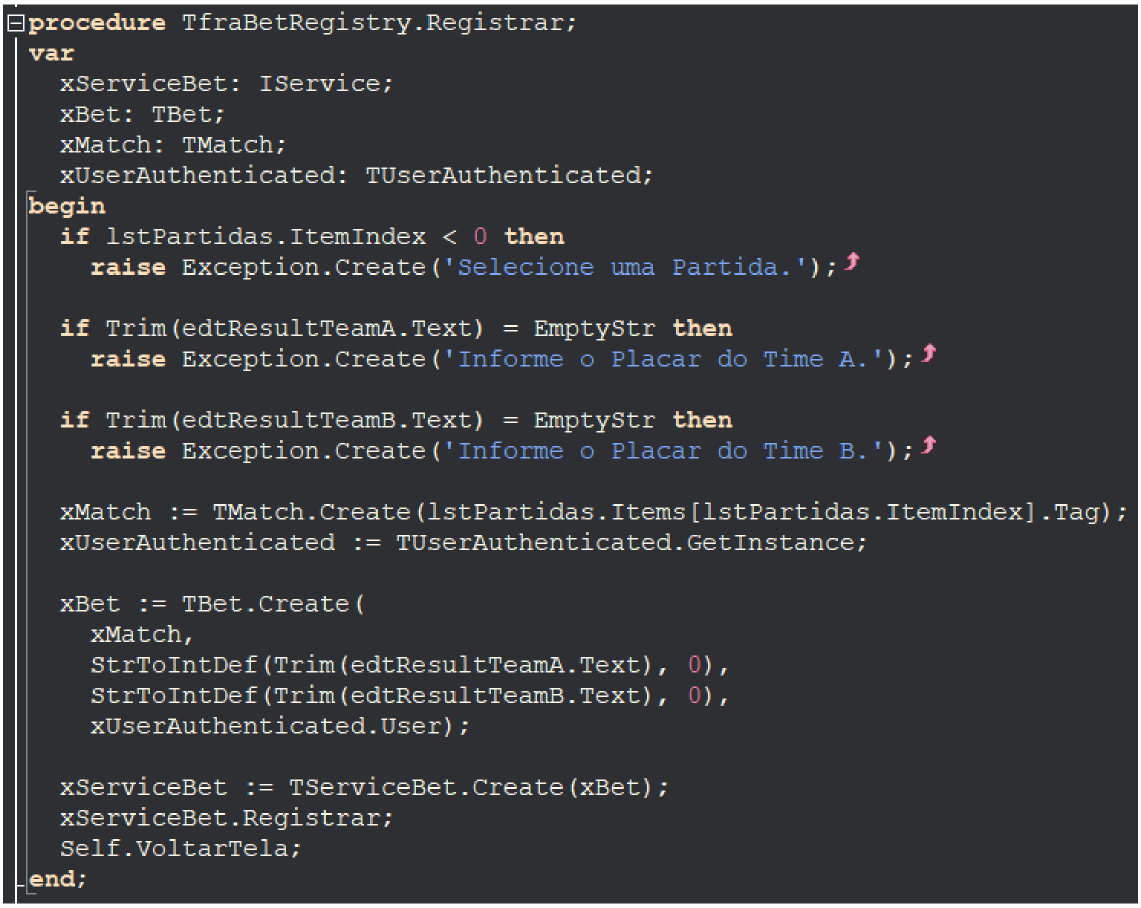
```
procedure TfraBetRegistry.PreencheMatchs(const aMatch: TMatch);
var
  xItem: TListViewItem;
const
  PARTIDA = '%s X %s - %s às %s';
begin
  xItem     := lstPartidas.Items.Add;
  xItem.Tag := aMatch.Id;

  TListItemText(xItem.Objects.FindDrawable('txtPartida')).Text :=
    Format(PARTIDA, [aMatch.TeamA.Name,
                     aMatch.TeamB.Name,
                     DateToStr(aMatch.Date),
                     TimeToStr(aMatch.Hour)]);
end;
```

8. Método CarregarMatchs



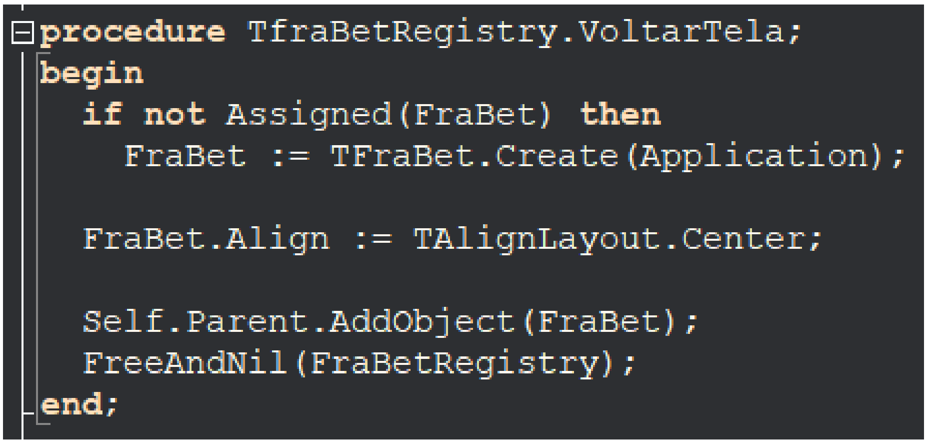
```
procedure TfraBetRegistry.CarregarMatchs;
var
  xServiceMatch: IService;
  xMatch: TMatch;
begin
  lstPartidas.Items.Clear;

  xServiceMatch := TServiceMatch.Create;
  xServiceMatch.Listar;
  for xMatch in TServiceMatch(xServiceMatch).Matchs do
    Self.PreencheMatchs(xMatch);
end;
```

9. Método Registrar



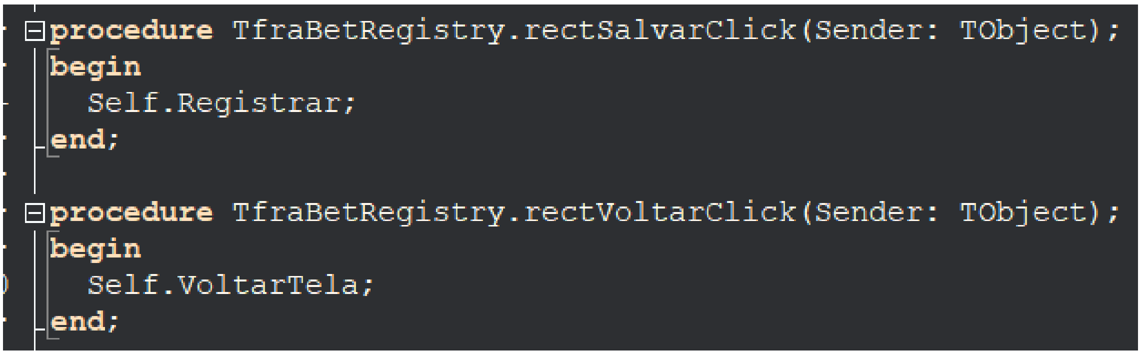
```
procedure TfraBetRegistry.Registrar;
var
  xServiceBet: IService;
  xBet: TBet;
  xMatch: TMatch;
  xUserAuthenticated: TUserAuthenticated;
begin
  if lstPartidas.ItemIndex < 0 then
    raise Exception.Create('Selecione uma Partida.');

  if Trim(edtResultTeamA.Text) = EmptyStr then
    raise Exception.Create('Informe o Placar do Time A.');

  if Trim(edtResultTeamB.Text) = EmptyStr then
    raise Exception.Create('Informe o Placar do Time B.');

  xMatch := TMatch.Create(lstPartidas.Items[lstPartidas.ItemIndex].Tag);
  xUserAuthenticated := TUserAuthenticated.GetInstance;

  xBet := TBet.Create(
    xMatch,
    StrToIntDef(Trim(edtResultTeamA.Text), 0),
    StrToIntDef(Trim(edtResultTeamB.Text), 0),
    xUserAuthenticated.User);

  xServiceBet := TServiceBet.Create(xBet);
  xServiceBet.Registrar;
  Self.VoltarTela;
end;
```

10. Método VoltarTela



```
procedure TfraBetRegistry.VoltarTela;
begin
  if not Assigned(FraBet) then
    FraBet := TFraBet.Create(Application);

  FraBet.Align := TAlignLayout.Center;

  Self.Parent.AddObject(FraBet);
  FreeAndNil(FraBetRegistry);
end;
```

11. Eventos Click do RectSalvar e RectVoltar



```
procedure TfraBetRegistry.rectSalvarClick(Sender: TObject);
begin
  Self.Registrar;
end;

procedure TfraBetRegistry.rectVoltarClick(Sender: TObject);
begin
  Self.VoltarTela;
end;
```