ETEC PROF. HORÁCIO AUGUSTO DA SILVEIRA

PROGRAMAÇÃO DE APLICATIVOS MOBILE – II – 2024/1

Luiz Fernando Souza

Aula 07 - Consumindo APIs – Listagem de Personagens do jogo RPG

1. Crie uma pasta chamada **Enuns** dentro da pasta Models, adicionando uma classe **ClasseEnum** dentro da pasta Enuns, e transformando em um *enum* conforme abaixo

```
public enum ClasseEnum
{
    NaoSelecionada = 0,
    Cavaleiro = 1,
    Mago = 2,
    Clerigo = 3
}
```

2. Clique com o botão direito na pasta Models e crie uma classe chamada **Personagem**, com as propriedades abaixo. Será necessário o using para *AppRpgEtec.Models.Enuns*

```
public class Personagem
{
    0 references
    public int Id { get; set; }
    0 references
    public string Nome { get; set; }
    0 references
    public int PontosVida { get; set; }
    0 references
    public int Forca { get; set; }
    0 references
    public int Defesa { get; set; }
    0 references
    public int Inteligencia { get; set; }
    0 references
    public byte[] FotoPersonagem { get; set; }
    0 references
    public int Disputas { get; set; }
    0 references
    public int Vitorias { get; set; }
    0 references
    public int Derrotas { get; set; }
    0 references
    public ClasseEnum Classe { get; set; }
}
```

3. Abra a classe **Request** e adicione os demais métodos que a API utilizará.

- Método Put

```
public async Task<int> PutAsync<TResult>(string uri, TResult data, string token)
 {
     HttpClient httpClient = new HttpClient();
     httpClient.DefaultRequestHeaders.Authorization
         = new AuthenticationHeaderValue("Bearer", token);

     var content = new StringContent(JsonConvert.SerializeObject(data));
     content.Headers.ContentType = new MediaTypeHeaderValue("application/json");
     HttpResponseMessage response = await httpClient.PutAsync(uri, content);

     string serialized = await response.Content.ReadAsStringAsync();

     if (response.StatusCode == System.Net.HttpStatusCode.OK)
         return int.Parse(serialized);
     else
         throw new Exception(serialized);
}
```

- Método Get

```
public async Task<TResult> GetAsync<TResult>(string uri, string token)
 {
     HttpClient httpClient = new HttpClient();
     httpClient.DefaultRequestHeaders.Authorization
         = new AuthenticationHeaderValue("Bearer", token);

     HttpResponseMessage response = await httpClient.GetAsync(uri);
     string serialized = await response.Content.ReadAsStringAsync();
     TResult result = await Task.Run(() =>
         JsonConvert.DeserializeObject<TResult>(serialized));
     throw new Exception(serialized);
}
```

- Método Delete

```
public async Task<int> DeleteAsync(string uri, string token)
 {
     HttpClient httpClient = new HttpClient();
     httpClient.DefaultRequestHeaders.Authorization = new
AuthenticationHeaderValue("Bearer", token);
     HttpResponseMessage response = await httpClient.DeleteAsync(uri);
     string serialized = await response.Content.ReadAsStringAsync();
     if (response.StatusCode == System.Net.HttpStatusCode.OK)
         return int.Parse(serialized);
     else
         throw new Exception(serialized);
 }
```

4. Crie uma pasta chamada **Personagens** dentro da pasta Services, e crie a classe chamada **PersonagemService** dentro da pasta recém-criada. Adicione a herança para a classe service, a variável global (_request) que representa a instancia da requisição a string que guardará o caminho da sua API, a variável global de token e o construtor que receberá o token passado para alimentar o token da classe.

```
1 reference
public class PersonagemService : Request
{
    private readonly Request _request;
    private const string apiUrlBase = "https://xyz.azurewebsites.net/Personagens";
    //xyz --> site da sua API

    private string _token;
    0 references
    public PersonagemService(string token)
    {
        _request = new Request();
        _token = token;
    }

    //Próximos métodos aqui

}
```

5. Adicione os métodos que consomem a API com o usings que o Visual Studio sugerir.

```
public async Task<Personagem> PostPersonagemAsync(Personagem p)
        {
             return await _request.PostReturnIntAsync(apiUrlBase, p, _token);
        }


        public async Task<ObservableCollection<Personagem>> GetPersonagensAsync()
        {
            string urlComplementar = string.Format("{0}", "/GetAll");
            ObservableCollection<Models.Personagem> listaPersonagens = await
      _request.GetAsync<ObservableCollection<Models.Personagem>>(apiUrlBase + urlComplementar,
_token);
            return listaPersonagens;
        }



        public async Task<Personagem> GetPersonagemAsync(int personagemId)
        {
            string urlComplementar = string.Format("/{0}", personagemId);
            var personagem = await _request.GetAsync<Models.Personagem>(apiUrlBase +
urlComplementar, _token);
            return personagem;
        }
```

```
public async Task<int> PutPersonagemAsync(Personagem p)
{
    var result = await _request.PutAsync(apiUrlBase, p, _token);
    return result;
}

public async Task<int> DeletePersonagemAsync(int personagemId)
{
    string urlComplementar = string.Format("/{0}", personagemId);
    var result = await _request.DeleteAsync(apiUrlBase + urlComplementar, _token);
    return result;
}
```

6. Cria uma pasta **Personagens** dentro da pasta ViewModels. Adicione uma classe com o nome **ListagemPersonagemViewModel.cs** herdando a classe *BaseViewModel* e deixando a classe pública. Prossiga na viewModel realizando a programação a seguir dentro da classe

```
public class ListagemPersonagemViewModel : BaseViewModel
{
    1 private PersonagemService pService;
      1 reference
    2 public ObservableCollection<Personagem> Personagens { get; set; }
      0 references
    3 public ListagemPersonagemViewModel()
      {
          string token = Preferences.Get("UsuarioToken", string.Empty);
          pService = new PersonagemService(token);
          Personagens = new ObservableCollection<Personagem>();
      }

    //Próximos elementos da classe aqui

}//Fim da classe
```

(1) Declaração da variável de serviço que consumirá a API
(2) Declaração da Coleção de Personagens como propriedade
(3) Construtor pegando token, inicializando o serviço passando o token e inicializando a lista de personagens

- Usings de *AppRpg.Models; AppRpg.Services.Personagens; System.Collections.ObjectModel, Xamarin.Forms* e *using System.Threading.Tasks;*

7. Programe o método de busca dos personagens.

```
public async Task ObterPersonagens()
{
    try //Junto com o Cacth evitará que erros fechem o aplicativo
    {
        Personagens = await pService.GetPersonagensAsync();
        OnPropertyChanged(nameof(Personagens)); //Informará a View que houve carregamento
    }
    catch (Exception ex)
    {
        //Captará o erro para exibir em tela
        await Application.Current.MainPage
            .DisplayAlert("Ops", ex.Message + " Detalhes: " + ex.InnerException, "Ok");
    }
}
```

8. Adicione no construtor a chamada para o método

```
public ListagemPersonagemViewModel()
{
    string token = Preferences.Get("UsuarioToken", string.Empty);
    pService = new PersonagemService(token);
    Personagens = new ObservableCollection<Personagem>();

    _ = ObterPersonagens();
}
```

- O "_" (underline) descarta a operação assíncrona de usar o operador await e armazenar um retorno.

9. Abra a view Personagens/ListagemView.xaml, remova o VerticalStackLayout que está dentro da tag ContentPage.Content e adicione o conteúdo abaixo.

```
<ScrollView>
        <VerticalStackLayout Padding="10, 0, 0, 0" VerticalOptions="FillAndExpand">
            <ListView x:Name="listView" HasUnevenRows="True" ItemsSource="{Binding Personagens}" >
                <ListView.ItemTemplate>
                    <DataTemplate>
                        <ViewCell>
                            <StackLayout Padding="10">
                                <Label Text="{Binding Nome}" FontSize="18" FontAttributes="Bold"/>
                                <Label Text="{Binding PontosVida}" FontSize="14"/>
                            </StackLayout>
                        </ViewCell>
                    </DataTemplate>
                </ListView.ItemTemplate>
            </ListView>
        </VerticalStackLayout>
        </ScrollView>
```

10. Na parte de código da v*iew*, declare a viewModel de personagem, inserindo o using para a pasta da viewModel, faça as inicializações necessárias no construtor e atribua ao contexto da view para que seja feita a operação para trazer a lista de personagens

```
public partial class ListagemView : ContentPage
{
    ListagemPersonagemViewModel viewModel;

    1 reference
    public ListagemView()
    {
        InitializeComponent();

        viewModel = new ListagemPersonagemViewModel();
        BindingContext = viewModel;
        Title = "Personagens - App Rpg Etec";
    }
}
```

- Execute o aplicativo e confirme que os registros serão listados. Abaixo está o resultado esperado