

Dell IT Academy



ASP.NET CORE MVC

SESSÃO

Estado da Sessão

- O protocolo HTTP é um protocolo sem manutenção de estado
- Ou seja, após realizar um ciclo de requisição/resposta, o servidor Web não armazena informações sobre essas interações cliente/servidor
- Contudo, algumas aplicações corporativas necessitam a manutenção de informações enquanto um cliente estiver interagindo com o servidor Web
 - Carrinho de compra
 - Informações de login de usuários
 - etc

Estado da Sessão

- Uma sessão é entendida como um conjunto de interações entre cliente/servidor
- Cada cliente possui uma sessão em separada de outro cliente
- As plataformas de desenvolvimento para Web oferecem diferentes alternativas para a implementação da manutenção do estado da sessão
 - Sessão no cliente
 - Sessão no servidor
 - Sessão no banco de dados

Sessão no Cliente

- Armazena o estado da sessão no cliente
- Implementação:
 - Query-Strings na URL
 - Cookies
 - Campos escondidos de formulários
 - Cliente rico
- Vantagens:
 - Servidor liberado de realizar manutenção de estado
- Desvantagens:
 - Queda de performance (tempo de transferência grande entre cliente e servidor) caso a quantidade de dados na sessão for grande
 - Dados inseguros no cliente

Sessão no Cliente - Cookies

- Valores serializados como texto enviados em cada requisição e resposta do HTTP
- Características:
 - São restritos a objetos que possam ser serializados em texto
 - Possuem um tamanho limite (usualmente 4096 bytes)
 - Armazenados em arquivos texto dentro de diretórios especiais dos navegadores
 - Podem ser desativados no navegador cliente
 - Só são válidos dentro do mesmo domínio
 - Isto é, um cookie gerado por um servidor, não pode ser manipulado por outro servidor

Sessão no Cliente - Cookies

- ASP.NET Core MVC fornece acesso à coleção de cookies no objeto de requisição e resposta
- Exemplos:

- Ler um cookie:

```
string cookieValueFromReq = Request.Cookies[chave];
```

- Escrever um cookie:

```
Response.Cookies.Append(chave, valor);
```

- Remover um cookie:

```
Response.Cookies.Delete(chave);
```


Sessão no Servidor

- Armazena o estado da sessão no servidor
- Implementação:
 - Objetos em memória
 - Objetos serializados em local persistente
- Vantagens:
 - Implementação simples, pois é fornecido na maioria das vezes pela própria plataforma de desenvolvimento
 - Melhora de performance em relação a manutenção de sessão no cliente
- Desvantagens:
 - Cada sessão de usuário em separado demanda recursos do servidor

Sessão no Servidor - Session

- ASP.NET Core MVC fornece o pacote *Microsoft.AspNetCore.Session* para um middleware para manutenção de sessão no lado servidor
- Características:
 - Utiliza cookies no lado cliente
 - Consiste em um dicionário
 - Qualquer tipo de objeto pode ser armazenado na sessão
 - Possui um tempo limite
 - Caso a sessão fique inativa, o objeto de sessão é invalidado
 - Valor padrão de 20 minutos
 - Sessão pode ser terminada via *Session.Clear()* explicitamente

Sessão no Servidor - Session

```
var builder = WebApplication.CreateBuilder(args);  
...  
builder.Services.AddDistributedMemoryCache();  
  
builder.Services.AddSession(options =>  
{  
    options.IdleTimeout = TimeSpan.FromSeconds(10);  
    options.Cookie.HttpOnly = true;  
    options.Cookie.IsEssential = true;  
});  
  
var app = builder.Build();  
...  
app.UseAuthorization();  
app.UseSession();  
...  
app.Run();
```

Sessão no Servidor - Session

- Acessando sessão na *view* via *razor*:

```
@using Microsoft.AspNetCore.Http  
Session Value = @HttpContext.Session.GetString("_Name")
```

- Acessando sessão no controlador

```
using Microsoft.AspNetCore.Http;  
HttpContext.Session.SetString("_Name", "Rick");  
var name = HttpContext.Session.GetString(SessionKeyName);
```

Sessão no Banco de Dados

- Armazena o estado da sessão em um servidor de banco de dados
- Implementação:
 - Estrutura de tabelas específica em um servidor de banco de dados
- Vantagens:
 - Configuração mais fácil de múltiplos servidores web
 - Estado de sessão persistente que sobrevive a falhas do servidor web
- Desvantagens:
 - Custo de acesso ao banco de dados

Sessão no Servidor/Cliente - TempData

- ASP.NET Core MVC fornece um objeto temporário para manter a sessão somente até o objeto ser lido
- Características:
 - Pode ser implementado via Cookie (opção padrão) ou Session
 - Propriedade *TempData* do controlador
 - Especialmente útil para controladores que precisam compartilhar dados entre requisições, usualmente envolvendo redirecionamento do HTTP
 - É um dicionário, com métodos *Keep*, *Peek*, *Load*, *Save*

Sessão no Servidor/Cliente - TempData

- Configuração do middleware:

```
var builder = WebApplication.CreateBuilder(args);  
builder.Services.AddControllersWithViews()  
                .AddSessionStateTempDataProvider();  
...  
builder.Services.AddSession();  
...  
var app = builder.Build();  
...  
app.UseSession();  
...  
app.Run();
```

Exemplo

- <https://github.com/dotnet/AspNetCore.Docs/tree/main/aspnetcore/fundamentals/app-state/samples>