2.2. Session

2.2.1. Introduction

StudioX provides **IStudioXSession** interface to obtain current user and tenant **without** using ASP.NET's Session. **IStudioXSession** is also fully integrated and used by other structures in StudioX (setting system and authorization system for instance)..

2.2.2. Injecting Session

IStudioXSession is generally property injected to needed classes unless it's not possible to work without session informations. If we use property injection, we can use NullStudioXSession.Instance as default value as shown below:

```
public class MyClass : ITransientDependency
{
    public IStudioXSession StudioXSession { get; set; }

    public MyClass()
    {
        StudioXSession = NullStudioXSession.Instance;
    }

    public void MyMethod()
    {
        var currentUserId = StudioXSession.UserId;
    }
}
```

Since authentication/authorization is an application layer task, it's adviced to use IStudioXSession in application layer and upper layers (we don't use it in domain layer normally). ApplicationService, StudioXController, StudioXApiController and some other base classes has StudioXSession already injected. So, you can directly use StudioXSession property in an application service method for instance.

2.2.3. Session Properties

StudioXSession defines a few key properties:

- **UserId**: Id of the current user or null if there is no current user. It can not be null if the calling code is authorized.

- **TenantId**: Id of the current tenant or null if there is no current tenant (in case of user has not logged in or he is a host user).
- **ImpersonatorUserId**: Id of the impersonator user if current session is impersonated by another user. It's null if this is not an impersonated login.
- **ImpersonatorTenantId**: Id of the impersonator user's tenant, if current session is impersonated by another user. It's null if this is not an impersonated login.
- **MultiTenancySide**: It may be Host or Tenant.

UserId and TenantId is **nullable**. There is also non-nullable **GetUserId()** and **GetTenantId()** methods. If you're sure there is a current user, you can call GetUserId(). If current user is null, this method throws exception. **GetTenantId()** is also similar.

Impersonator properties are not common as other properties and generally used for audit logging purposes.

ClaimsStudioXSession
ClaimsStudioXSession is the default implementation of IStudioXSession interface. It gets session properties (except MultiTenancySide, it's calculated) from claims of current user's princical. For a cookie based form authentication, it gets from cookies. Thus, it' well integrated to ASP.NET's authentication mechanism.

2.2.4. Overriding Current Session Values

In some specific cases, you may need to change/override session values for a limited scope. In such cases, you can use IAbpSession.Use method as shown below:

```
public class MyService
{
    private readonly IAbpSession session;

    public MyService(IAbpSession session) { this.session = session; }

    public void Test()
    {
        using (session.Use(42, null))
        {
            var tenantId = session.TenantId;
            var userId = session.UserId;
            }
        }
}
```

Use method returns an IDisposable and it must be disposed. Once the return value is disposed, Session values are automatically restored the to previous values.

Warning!

Always use it in a using block as shown above. Otherwise, you may get unexpected session values. You can have nested Use blocks and they will work as you expect.

2.2.5. User Identifier

You can use **.ToUserIdentifier()** extension method to create a UserIdentifier object from IStudioXSession. Since UserIdentifier is used in most API, this will simplify to create a UserIdentifier for current user.