GUID通常用于作为数据库表的组件，其再全球具有唯一性

**IGuidGenerator**

Abp提供了IGuidGenerator供我们生成连续的GUID

永远不要使用 Guid.NewGuid() 创建ID，因为它默认情况下不是连续的，这在插入时带来严重的性能问题（因为插入新记录可能需要对现有记录进行重新排序）

**使用**

public class MyProductService : ITransientDependency

{

    private readonly IRepository<Product, Guid> \_productRepository;

    private readonly IGuidGenerator \_guidGenerator;

    public MyProductService(

        IRepository<Product, Guid> productRepository,

        IGuidGenerator guidGenerator)

    {

        \_productRepository = productRepository;

        \_guidGenerator = guidGenerator;

    }

    public async Task CreateAsync(string productName)

    {

        // 生成 guid

        var product = new Product(\_guidGenerator.Create(), productName);

        await \_productRepository.InsertAsync(product);

    }

}

**配置选项**

针对不同的数据库，我们使用不同的Guid生成策略，这有利于提供新能

再模块的ConfigureServices方法中进行配置

Configure<AbpSequentialGuidGeneratorOptions>(options =>

{

    options.DefaultSequentialGuidType = SequentialGuidType.SequentialAsBinary;

});

SequentialGuidType的枚举值如下：

- SequentialAtEnd (default) 用于SQL Server.

- SequentialAsString 用于MySQL和PostgreSQL.

- SequentialAsBinary 用于Oracle.