如果我们的项目是基于Abp的应用模板生成的，那么默认包含审计功能

**审计功能配置**

审计功能默认开启，我们可以在模块的ConfigureServices方法中继续配置

Configure<AbpAuditingOptions>(options =>

{

    options.IsEnabled = false; // 是否启用审计功能，默认开启

});

常用配置项如下

* IsEnabled (默认值: true): 启用或禁用审计系统的总开关.
* HideErrors (默认值: true): 在保存审计日志时如果发生错误，审计日志系统会将错误隐藏并写入常规日志，如果为false，则保存不成功时抛出异常
* IsEnabledForAnonymousUsers (默认值: true): 匿名用户是否也记录审计日志
* IsEnabledForGetRequests (默认值: false): GET请求是否记录审计日志
* ApplicationName: 如果有多个应用程序保存审计日志到单一的数据库，使用此属性区分不同的应用程序日志.
* IgnoredTypes: 审计日志系统忽略的 Type 列表.
* EntityHistorySelectors: 选择器列表，用于选择那些实体可以被记录

**控制器、应用服务启用禁用审计**

控制器和应用服务默认启用审计，我们可以通过 [DisableAuditing] 禁用审计

该特性可作用与类和方法（Action）上

[DisableAuditing]

public class HomeController : AbpController

{

    //...

}

**实体启用禁用审计**

实体的变更可以记录审计日志，Abp默认不记录实体变更，如果想要记录某些实体的变更，可以使用 [Audited] 特性

如下我们记录MyUser实体的变更，但是不记录其Password的变更

[Audited]

public class MyUser : Entity<Guid>

{

    public string Name { get; set; }

    public string Email { get; set; }

    [DisableAuditing] //Ignore the Passoword on audit logging

    public string Password { get; set; }

}

**审计对象设计**

如下时Abp审计对象的设计

图示

描述已自动生成

* AbpAuditLogs：每个Http请求都会生成一条记录，包括请求的Url，用户，租户等信息
* AbpAuditLogActions：这次请求所调用的动作，可以时一个控制器的方法，可以是一个应用服务的方法，这里还记录了动作的耗时，方便优化性能
* AbpEntityChanges：记录了这次请求所变更的实体
* AbpEntityPropertyChanges：记录了实体的那些属性变更了

详细文档：https://docs.abp.io/zh-Hans/abp/latest/Audit-Logging