IEmailSender

它是一个你无需了解它的细节就能简单地用来发送邮件的服务，用法如下所示：

public class TaskManager : IDomainService

{

**private readonly IEmailSender \_emailSender;**

public TaskManager(IEmailSender emailSender)

{

\_emailSender = emailSender;

}

public void Assign(Task task, Person person)

{

//Assign task to the person

task.AssignedTo = person;

//Send a notification email

**\_emailSender.Send(**

**to: person.EmailAddress,**

**subject: "You have a new task!",**

**body: $"A new task is assigned for you: <b>{task.Title}</b>",**

**isBodyHtml: true**

**);**

}

}

配置

邮件发送使用了[设置管理](http://www.cnblogs.com/kid1412/p/5988899.html" \t "http://www.cnblogs.com/kid1412/p/_blank)系统来读取邮件发送的配置，所有设置的名称都以常量的形式定义在Abp.Net.Mail.EmailSettingNames类里。下列是它的值和描述：

* Abp.Net.Mail.**DefaultFromAddress**：默认的邮件发送者的地址（如上面的示例）。
* Abp.Net.Mail.**DefaultFromDisplayName**：默认的邮件发送者显示的名称（如上面的示例）。
* Abp.Net.Mail.**Smtp.Host**: SMTP服务器的IP或域名（默认为127.0.0.1）。
* Abp.Net.Mail.**Smtp.Port**: SMTP服务器的端口（默认为25）。
* Abp.Net.Mail.**Smtp.UserName**: 当SMTP服务器要求认证时，需要提供的用户名。
* Abp.Net.Mail.**Smtp.Password**: 当SMTP服务器要求认证时，需要提供的密码。
* Abp.Net.Mail.**Smtp.Domain**: 当SMTP服务器要求认证时，需要提供的域名。
* Abp.Net.Mail.**Smtp.EnableSsl**: 表明一个SMTP服务器是（true）否（false）需要使用SSL连接（默认为false）。
* Abp.Net.Mail.**Smtp.UseDefaultCredentials**: 为True时使用默认的凭证代替提供的用户和密码（默认为true）。

集成 MailKit

由于.net core不支持标准的System.Net.Mail.SmtpClient，所以我们需要第三方供应商来发送邮件，幸运地是，[MailKit](https://github.com/jstedfast/MailKit" \t "http://www.cnblogs.com/kid1412/p/_blank) 是默认的Smtpclient的一个很好的代替，而且微软也建议使用它。

Abp.MailKit 包优雅地集成到了Abp的邮件发送系统里，所以，你仍可像前面的方式通过MailKit使用IEmailSender。

安装

首先，安装[Abp.MailKit](https://www.nuget.org/packages/Abp.MailKit" \t "http://www.cnblogs.com/kid1412/p/_blank) 包到你的项目：

Install-Package Abp.MailKit

集成

添加AbpMailKitModule 依赖到你的模块：

**[DependsOn(typeof(AbpMailKitModule))]**

public class MyProjectModule : AbpModule

{

//...

}

用法

你可以像前面描述的那样使用IEmailSender，因为Abp.MailKit包为它注册了MailKit的实现。也使用上面定义的配置。

定制

在创建MailKit的SmtpClient时，你可能会有额外的配置或自己的定制，此时，你可以用你自己的实现[替换](http://www.cnblogs.com/kid1412/p/5975791.html" \t "http://www.cnblogs.com/kid1412/p/_blank)IMailKitSmtpBuilder 接口的注册，不过可以通过继承DefaultMailKitSmtpBuilder 更简单。比如，你想为所有的SSL连接提供一个凭证，这种情况下，你可以重写ConfigureClient方法，如下所示：

public class MyMailKitSmtpBuilder : **DefaultMailKitSmtpBuilder**

{

public MyMailKitSmtpBuilder(ISmtpEmailSenderConfiguration smtpEmailSenderConfiguration)

: base(smtpEmailSenderConfiguration)

{

}

protected override void ConfigureClient(SmtpClient client)

{

**client.ServerCertificateValidationCallback = (sender, certificate, chain, errors) => true;**

base.ConfigureClient(client);

}

}

然后在你的模块的PreInitialize方法中用你上面的实现替换IMailKitStmpBuilder接口的实现：

[DependsOn(typeof(AbpMailKitModule))]

public class MyProjectModule : AbpModule

{

public override void PreInitialize()

{

**Configuration.ReplaceService<IMailKitSmtpBuilder, MyMailKitSmtpBuilder>();**

}

//...

}