

# 后台地址配置

宿主启动cosmos参数时，需要填入ip字段，通过ip字段来配置后台地址，参考文档[Cosmos大核SDK接入文档](#)中“初始化环境”，支持多ip或域名填入

Ip填入规则：

单ip规则： ip:port; (ip和端口之间使用:分割，最后需要加上;) 示例： 127.0.0.1:9999;

多ip规则： ip:port;ip1:port1;(ip和端口之间使用:分割, 多个地址之间使用;分割，需要加上;) 示例： 127.0.0.1:9999;126.0.0.2:9999;

域名填入规则：与ip填入规则一致

支持域名和ip混合填写：

示例： 127.0.0.1:9999;test.ip.com:9998;

# 接口设计

```
/**
 * 定义第三方查询参数的接口。
 * 该接口包含三个属性，用于描述第三方服务的查询请求。
 */
public interface IThirdQueryParameters
{
    /**
     * 获取或设置操作名称。
     * 例如，"GetUserData" 或 "UpdateUser"。
     */
    string Action { get; set; }

    /**
     * 获取或设置查询参数。
     * 序列化串。
     */
    string Parameters { get; set; }

    /**
     * 获取或设置路由信息。
     * 用于指定请求的目标路由。
     * 为空默认发给upb_cosmos_plugin_req
     */
    string Router { get; set; }
}

/// <summary>
/// 向第三方服务发送请求。
/// </summary>
/// <param name="parameters">请求参数</param>
/// <returns>应答结果</returns>
Task<IThirdResponse> ThirdQuery(IThirdQueryParameters parameters);
```

```

/// <summary>
/// 向第三方模块订阅数据
/// </summary>
/// <param name="subscriberParameters">订阅参数</param>
/// <param name="subscriber"></param>
/// <returns></returns>
Task<ITradeDataSubscription> SubscribThirdModule(ISubscribeParameters
subscriberParameters, EventHandler<IPushData> subscriber);

/// <summary>
/// 向第三方模块取消订阅数据
/// </summary>
/// <param name="unsubscribeParameters"></param>
/// <param name="subscriber"></param>
/// <returns></returns>
Task UnSubscribThirdModule(IUnSubscribeParameters unsubscribeParameters,
ITradeDataSubscription subscription);

```

## 请求路由

客户端如何将请求路由到不同的服务，根据IThirdQueryParameters类中的Router字段

该字段如何填写，需要询问服务提供方的服务名，根据后台定义填入对应Router，Action，Parameters参数进行请求、订阅等

## 组件如何调用

1.组件添加最新后台服务开发包:Cosmos.DataAccess.Trade.x.x.x.nupkg， 获取地址：[组件开发包](#)，找到最新版本开发包

2.组件开发者需要继承ICosmosTradeDataInteraction这个接口类

3.cosmos在创建组件过程中，会将后台服务的访问器赋值给组件的

(ICosmosTradeDataInteraction.TradeAccessorsInjection) ,通过该访问器，组件可直接访问后台服务

## 示例 参考示例

参考示例中和下面的请求订阅例子都是与后台pluginserver1服务做的数据交互，方法和参数都是根据pluginserver1中提供的填写的，如果测试的为别的服务，具体参数需要和后台服务提供方确认

## 请求

```

private async void btn_req_higate_Click(object sender, RoutedEventArgs e)
{
    /// 调用第三方接口
    IThirdQueryParameters queryParameters =
    _tradeDataAccessor.DataProvider.CreateThirdQueryParameters();
    queryParameters.Action = "request.plugin.test";
    queryParameters.Router = "pluginserver1";
    var options = new JsonSerializerOptions
    {
        Encoder = JavaScriptEncoder.Create(
            allowedRanges: new[]
            {

```

```

        UnicodeRanges.BasicLatin,           // ASCII 字符
        UnicodeRanges.CjkUnifiedIdeographs // 中文字符 (CJK 统一表意文字)
    }
}

};

JsonObject context = new JsonObject();
context["name"] = text_higate.Text;
queryParameters.Parameters = JsonSerializer.Serialize(context, options);

//同步等待应答
{
    var result = await
_tradeDataAccessor.DataProvider.ThirdQuery(queryParameters);
    _logger?.Log(CosmosLogLevel.Information, $"reqresult:{result.data}");
}

//异步不卡住界面
{
    Task.Run(async () =>
    {
        var result = await
_tradeDataAccessor.DataProvider.ThirdQuery(queryParameters);
        _logger?.Log(CosmosLogLevel.Information, $"reqresult:
{result.data}");
    });
}

}

```

## 订阅

```

private async Task TestTradeDataInterface()
{
    /// 调用第三方订阅接口
    ISubscribeParameters subscribeParameters =
_tradeDataAccessor.DataProvider.CreateSubscribeParameters();
    subscribeParameters.Action = "subscribe.plugin.test";
    subscribeParameters.Router = "pluginserver1";
    subscribeParameters.Parameters = "topic=entrust,date=today";
    subscribeParameters.Topic = "entrust";
    pushsubscribetion_ = await
_tradeDataAccessor.DataProvider.SubscribThirdModule(subscribeParameters, (object
sender, IPushData pushResult) =>
{
    _logger?.Log(CosmosLogLevel.Information, $" RecvPushData :
{pushResult.data}");
});
}

```

## 取消订阅

```

private async void btn_unsub_higate_Click(object sender, RoutedEventArgs e)
{
    if (pushsubscribetion_!=null)

```

```
{  
    /// 调用第三方订阅接口  
    IUnSubscribeParameters unsubscribeParameters =  
    _tradeDataAccessor.DataProvider.CreateUnSubscribeParameters();  
    unsubscribeParameters.Action = "subscribe.plugin.test";  
    unsubscribeParameters.Router = "pluginserver1";  
    unsubscribeParameters.Parameters = "topic=entrust,date=today";  
    unsubscribeParameters.Topic = "entrust";  
  
    await  
    _tradeDataAccessor.DataProvider.UnSubscribeThirdModule(unsubscribeParameters,  
    pushsubscription_);  
  
    _logger?.Log(CosmosLogLevel.Information, "取消订阅higate完成");  
    wpfToast.Show("取消订阅higate完成");  
}  
}
```