



Microsoft®  
**patterns & practices**  
proven practices for predictable results

# Enterprise Library

## Configuration Application Block

曹严明  
.NET架构顾问  
Microsoft (China)

# 日程

您的潜力. 我们的动力

**Microsoft**  
微软(中国)有限公司

- Enterprise Library 概述
- Configuration Block 概述
- 如何使用 Configuration Block
- Configuration Block 的底层设计和高级功能
- 面向高级开发人员的扩展机制

您的潜力. 我们的动力

**Microsoft®**

微软(中国)有限公司

# Enterprise Library 概述

# 什么是“Enterprise Library”

您的潜力, 我们的动力

**Microsoft**  
微软(中国)有限公司

- 一组可重用的程序块(Application Block)
- 用于解决企业级应用开发过程中所面临的共性的问题
  - 配置管理
  - 日志管理
  - 异常处理
  - 数据访问
  - 缓存机制
  - 加密机制
  - 安全机制

# 为什么使用“Enterprise Library”

您的潜力，我们的动力

Microsoft®  
微软(中国)有限公司

- 重用
- 最佳实现
- 一致性
- 易用性
- 可扩展性



# Enterprise Library 的构成

您的潜力. 我们的动力

**Microsoft**  
微软(中国)有限公司

- 源代码
- 示例应用程序
- 文档

- 免费下载

<http://www.microsoft.com/practices>

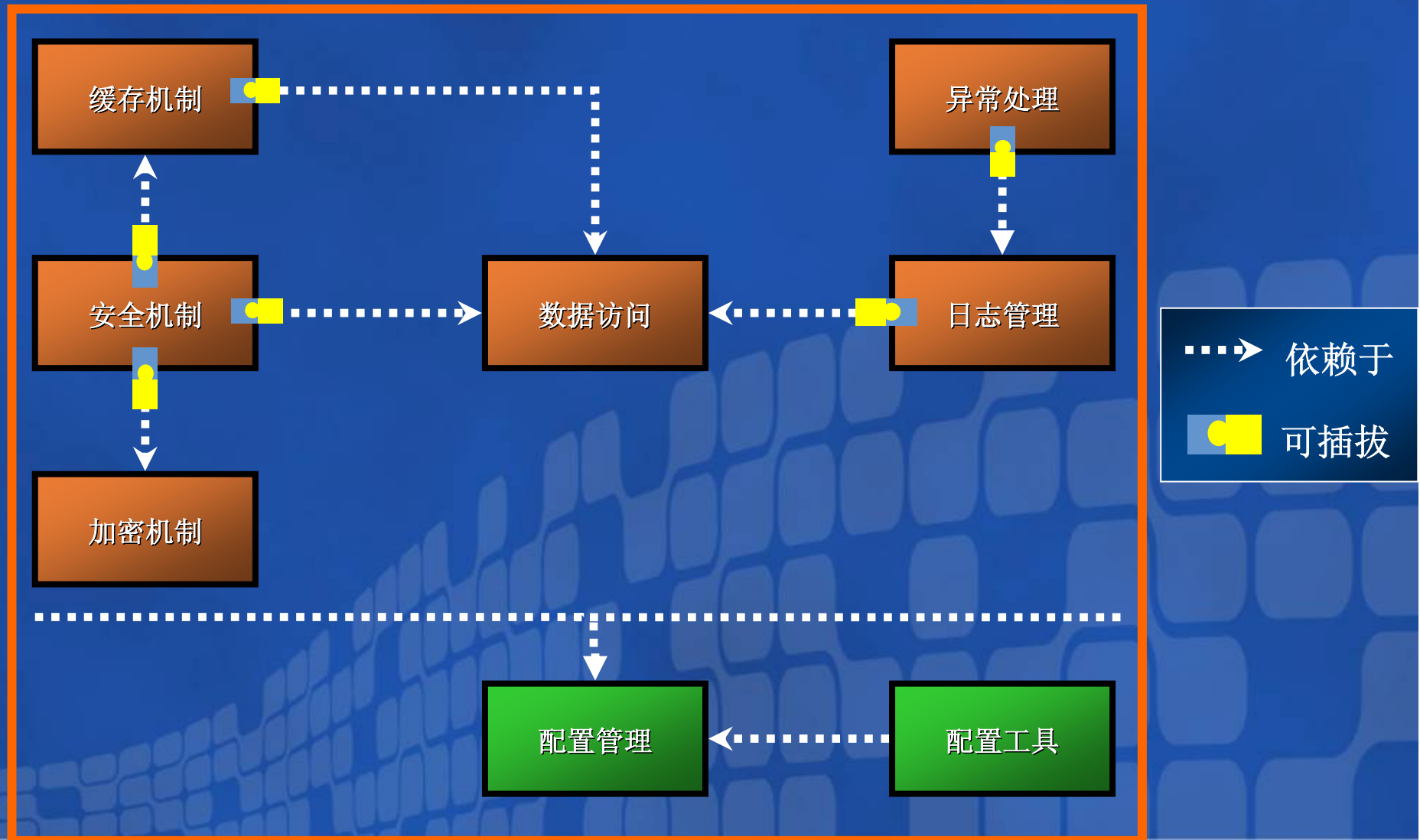
- 社区支持

<http://workspaces.gotdotnet.com/entlib>

# Enterprise Library 1.0

您的潜力. 我们的动力

**Microsoft**  
微软(中国)有限公司



您的潜力. 我们的动力

**Microsoft**  
微软(中国)有限公司

# Configuration Block 概述



# 你遇到过这些有关配置的问题吗？

无限潜力 我们的动力  
Microsoft  
微软(中国)有限公司

- 争论配置信息应该保存在哪里
- 自己编写的各种不同的读写配置信息的组件
- 开发不同的工具创建和修改配置数据
- 正运行的应用程序如何响应外部配置信息的改变

# Config Block 的使用场景

您的潜力. 我们的动力

**Microsoft**  
微软(中国)有限公司

- 应用系统需要在运行时读写复杂的配置信息
- 应用系统需要在配置文件中保存敏感数据  
(比如密码)
- 设计应用系统时能帮助开发人员进行正确的配置读写
- 应用系统部署后系统管理员能有一个好的工具修改 XML 配置文件

# Configuration Application Block

您的潜力, 我们的动力  
Microsoft®  
微软(中国)有限公司

- 为应用系统提供了一个通用的配置管理解决方案, 可方便灵活地从各种存储中读写配置信息
- 为读写配置数据提供了简单的接口
- 支持不同的配置存储
- 支持配置信息的加密
- 支持配置文件的动态更新
- 支持复杂的配置对象

您的潜力. 我们的动力

**Microsoft**  
微软(中国)有限公司

# 如何使用 Configuration Block



# 定义配置数据

您的潜力. 我们的动力

**Microsoft**  
微软(中国)有限公司

- 编写一个类，定义配置数据
  - 必须是可序列化的 (XmlSerializer)
  - 可以任意复杂

```
Public Class EditorFontData
    Private fontName As String
    Private fontSize As Double
    Private fontStyle As Integer
End Class
```



# 配置区和元数据

您的潜力. 我们的动力

**Microsoft**  
微软(中国)有限公司

- Configuration settings are grouped together in configuration sections
- 对每个配置区, 设置一个 storage provider 和一个 transformer
- 配置元数据 (metadata) 保存在应用程序域的配置文件中
  - app.config 或 web.config
- 配置元数据是用来确定使用什么 storage provider 和 transformer, 来读取配置区数据

# 声名一个配置区

您的潜力. 我们的动力

**Microsoft**  
微软(中国)有限公司

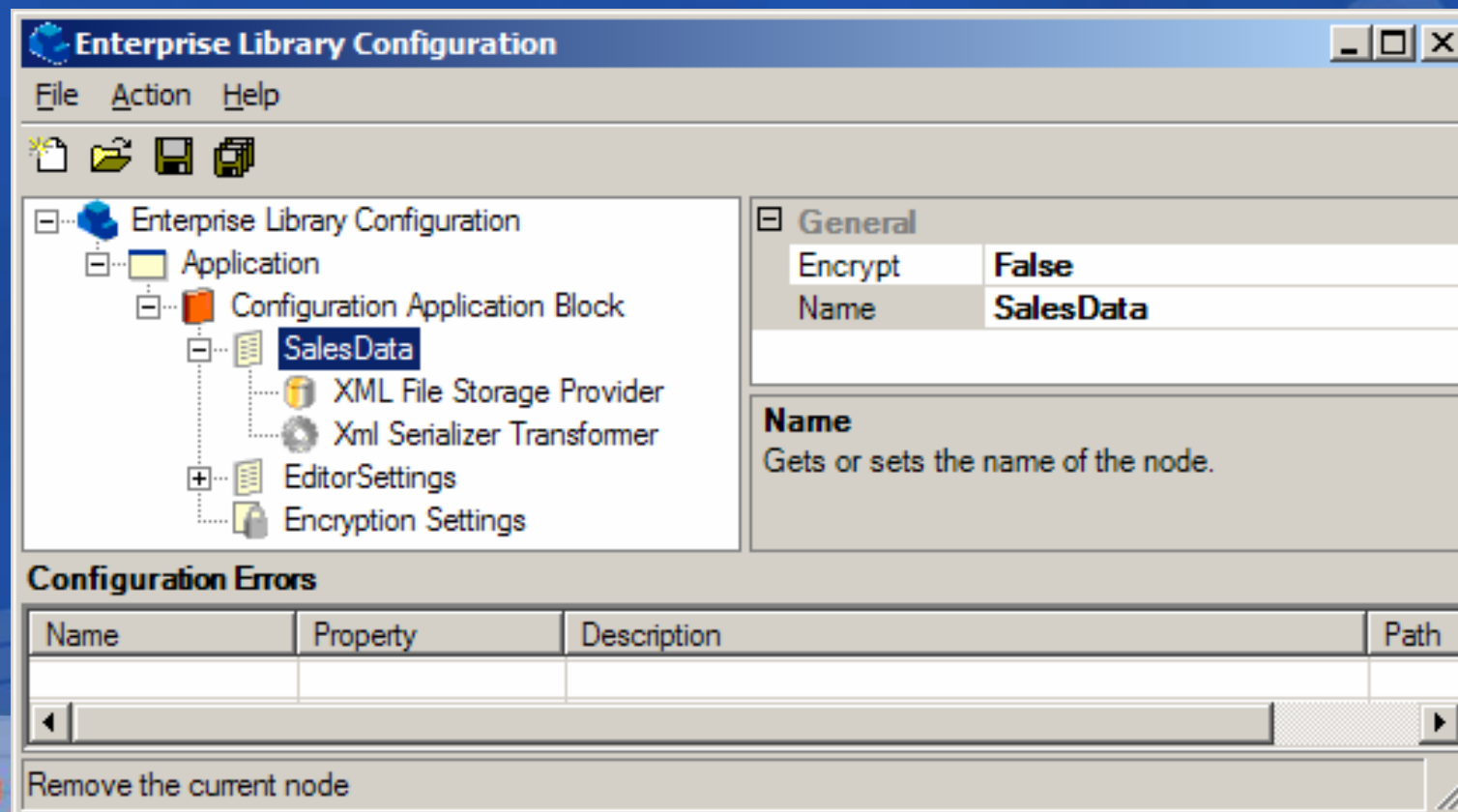
```
<configuration>
  <configurationSections>
    <configurationSection name="SalesData"
      encrypt="false">
      <storageProvider xsi:type="XmlFileStorageProviderData"
        name="XML File Storage Provider"
        path="salesdata.config" />
      <dataTransformer
        xsi:type="XmlSerializerTransformerData"
        name="Xml Serializer Transformer">
        <includeTypes />
      </dataTransformer>
    </configurationSection>
  </configurationSections>
</configuration>
```

# Configuration Console

您的潜力. 我们的动力

**Microsoft**  
微软(中国)有限公司

- Configuration Console 读写配置元数据
- 很容易添加和删除配置区
- 很容易修改属性
- 校验属性值

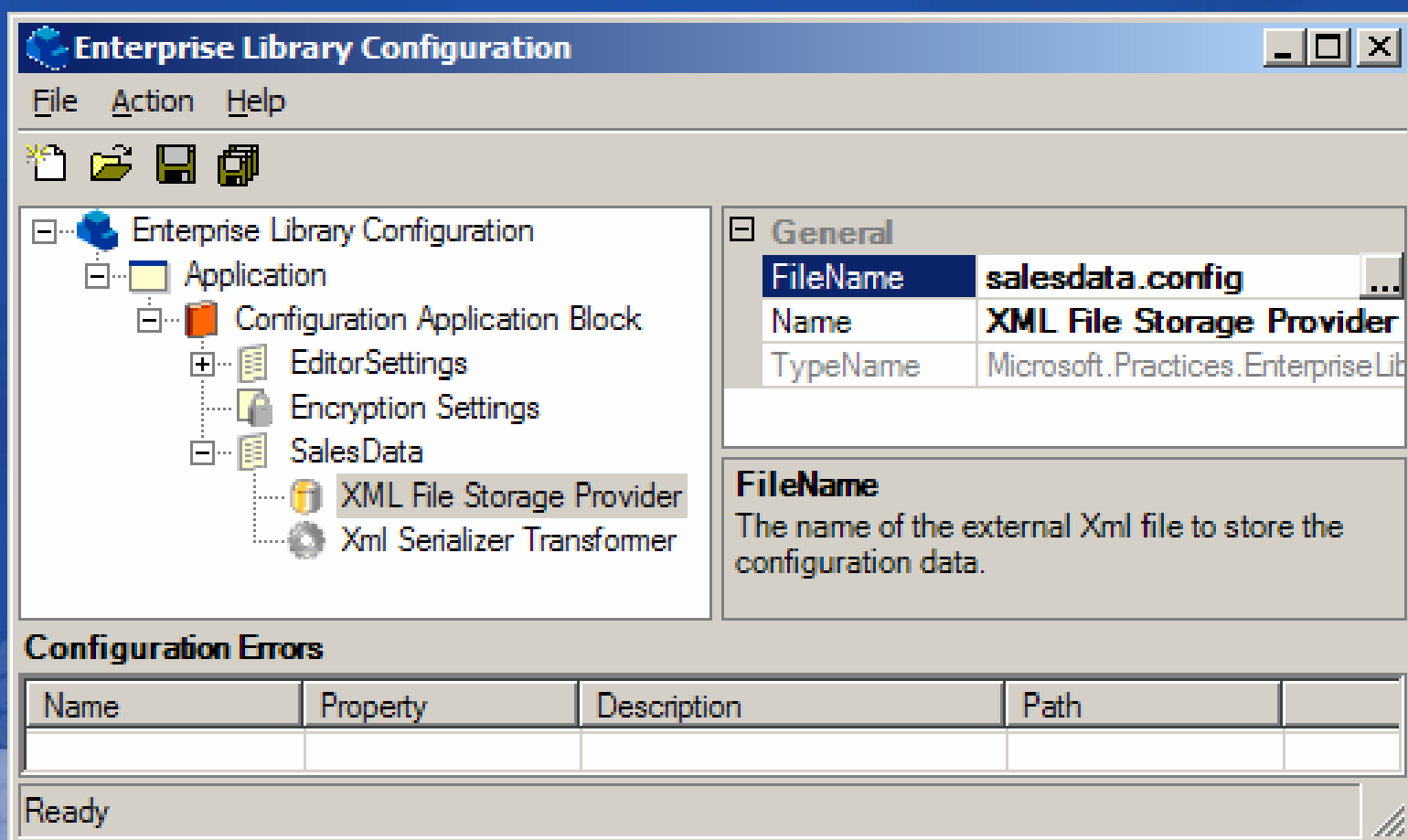


# 声名一个配置区

您的潜力. 我们的动力

**Microsoft**  
微软(中国)有限公司

- XML File Storage Provider 可设置包含配置信息的XML文件的文件名和路径
- 路径相对于 app.config 文件的位置



# 写配置信息

您的潜力. 我们的动力

**Microsoft**  
微软(中国)有限公司

- Config block 提供 API 写配置区信息
- 整个配置区被覆盖 (无合并)
- 元配置文件必须包含该配置区的定义

例子：写一个字符串值

```
string servername = "MyServer";  
ConfigurationManager.WriteConfiguration("SalesData",  
servername);
```



# 例子：写配置信息

您的潜力. 我们的动力

**Microsoft**  
微软(中国)有限公司

```
Public Class EditorFontData
```

```
    Private fontName As String
```

```
    Private fontSize As Double
```

```
    Private fontStyle As Integer
```

```
End Class
```

```
Dim configData As EditorFontData = New EditorFontData
```

```
configData.Name = fontDialog.Font.Name
```

```
configData.Size = fontDialog.Font.Size
```

```
configData.Style = Convert.ToInt32(fontDialog.Font.Style)
```

```
ConfigurationManager.WriteConfiguration("EditorSettings",  
configData)
```

# 读配置数据

您的潜力. 我们的动力

**Microsoft**  
微软(中国)有限公司

- 只需一行程序
- storage provider and transformer 负责返回期望的对象类型
- XML Serializer Transformer 将 XmlNode 反序列化为对象

```
// Read a string
Dim connectionString As String
connectionString =
DirectCast( ConfigurationManager.GetConfiguration("connectionst
ring",
String )

// Read an object with multiple properties
Dim configData As EditorFontData = _

DirectCast( ConfigurationManager.GetConfiguration("EditorSettin
gs"), _
EditorFontData)
```

您的潜力. 我们的动力

**Microsoft**  
微软(中国)有限公司

# Configuration Block 的底层设计 和高级功能

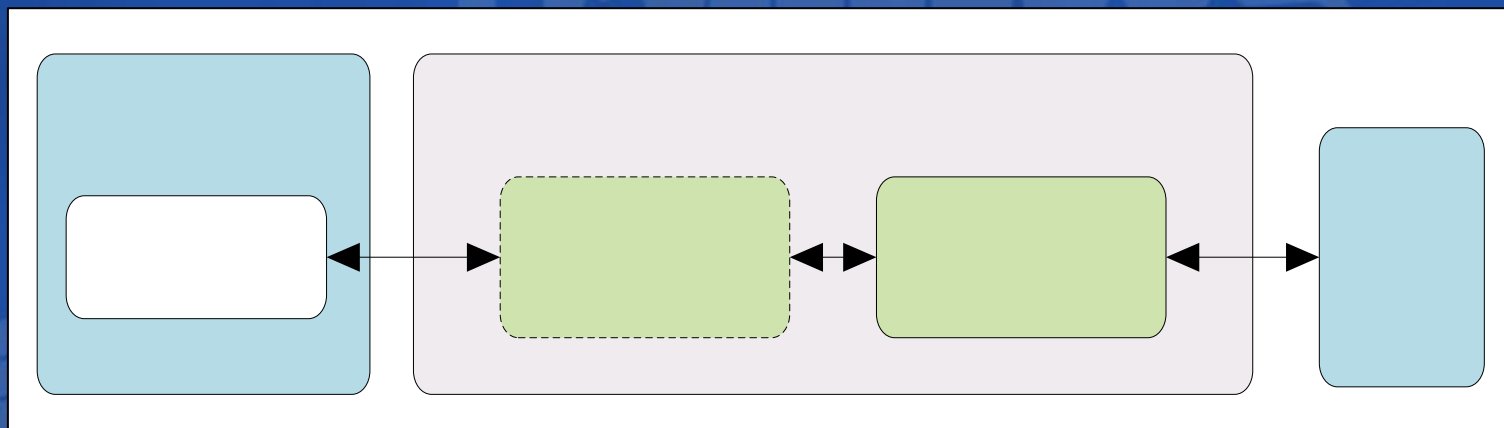
# Storage Providers and Transformers

您的潜力, 我们的动力

Microsoft®

微软(中国)有限公司

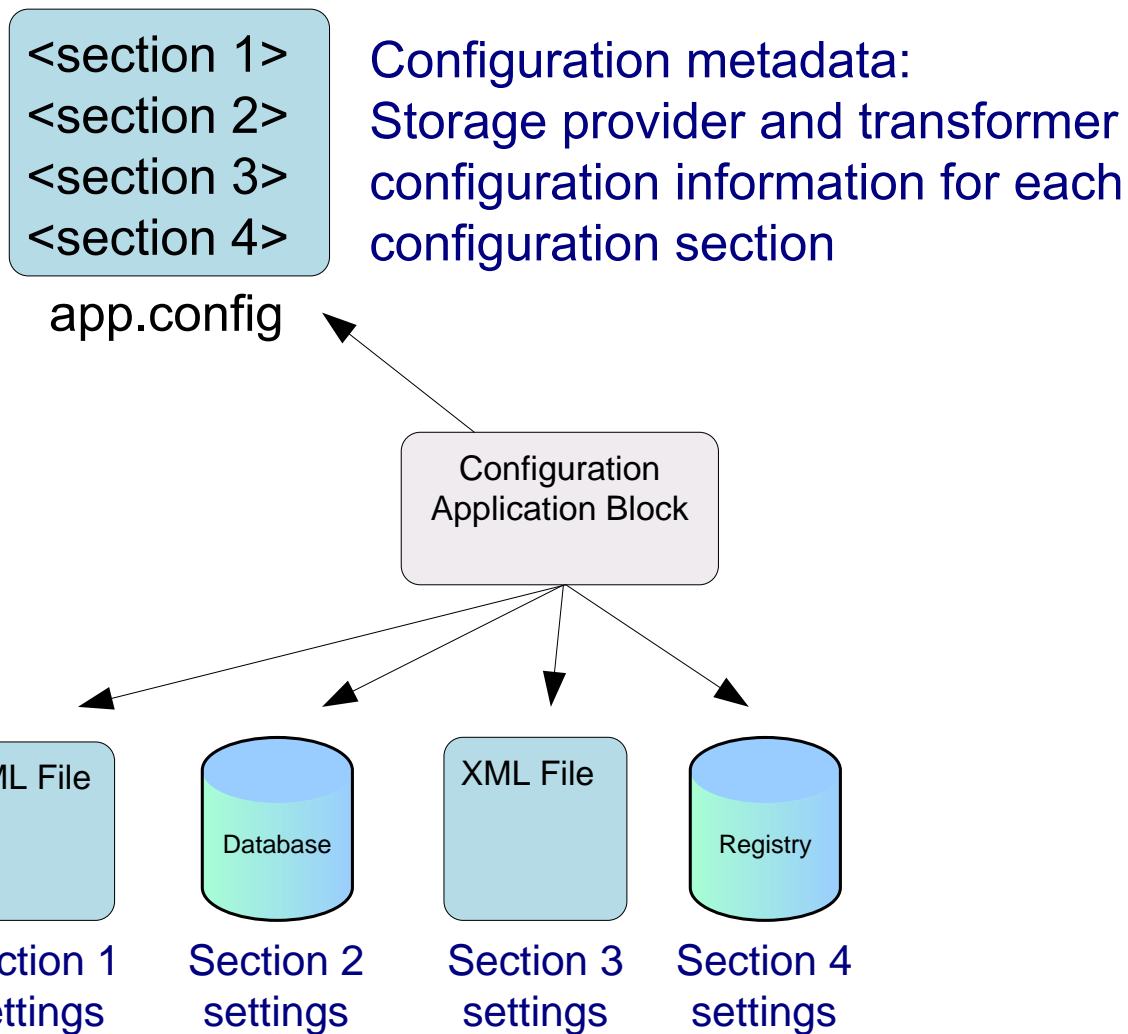
- 解耦: 将配置数据的读写与数据存储分离
  - 使用 storage providers and transformers 在应用和物理存储之间传递数据
  - 采用 AbstractFactory 模式生成 providers 实例
- Storage providers 是读写某个物理存储的对象, 比如 XML 文件或 SQL 数据库
- Transformers 是在存储格式和应用格式之间转换配置数据的对象



# Configuration Sections and Metadata

您的潜力. 我们的动力

**Microsoft**  
微软(中国)有限公司



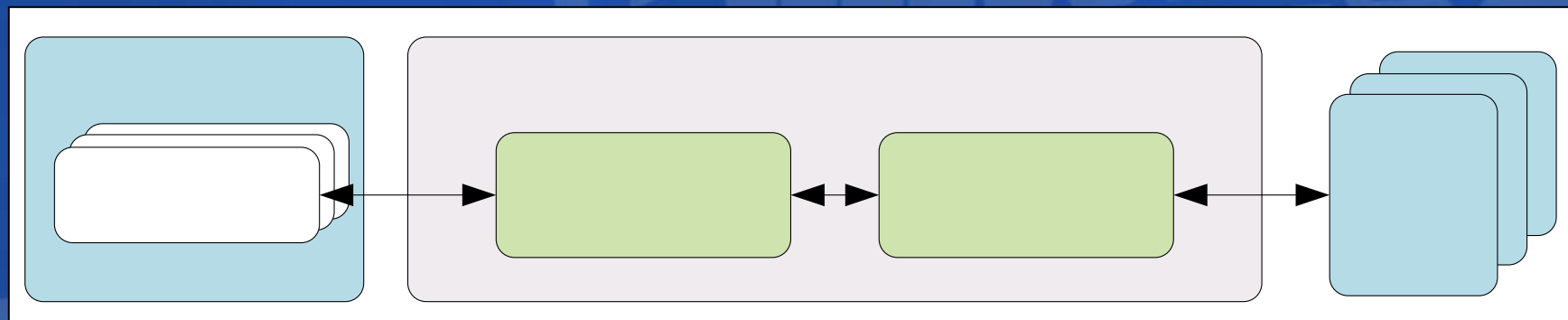


# 已经实现的 Provider

您的潜力, 我们的动力

**Microsoft**  
微软(中国)有限公司

- Config block 包含读写XML文件的 storage provider
- 每个配置区对应于一个不同的 XML 文件
- XML storage provider 返回一个 XmlNode 对象
- Config block 包含序列化和反序列化 XmlNode 对象的 transformer



# Example Configuration Data

您的潜力. 我们的动力

**Microsoft**  
微软(中国)有限公司

```
<?xml version="1.0" encoding="utf-8"?>
<EditorSettings>
  <xmlSerializerSection
type="ConfigurationQuickStart.EditorFontData,
      ConfigurationRuntimeData, Version=1.0.0.0,
Culture=neutral,
      PublicKeyToken=null">
    <EditorFontData
xmlns:xsd="http://www.w3.org/2001/XMLSchema"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
      <Name>Microsoft Sans Serif</Name>
      <Size>8.25</Size>
      <Style>2</Style>
    </EditorFontData>
  </xmlSerializerSection>
</EditorSettings>
```

# 配置数据的缓存

您的潜力. 我们的动力

**Microsoft**  
微软(中国)有限公司

- 配置信息被缓存
- 当storage provider 监测到配置的更改, 缓存被清零
- 使用接口  
`ConfigurationManager.GetConfiguration()`  
来获取配置
- 不用在别的地方缓存配置信息

# 响应配置变更通知

- Config block 提供一个事件机制, 当存储的配置变更时通知应用程序
- 这个功能依赖于 storage provider 能够识别配置变更
  - 有些 providers 也许不支持这个功能
- XML File Storage Provider 能够监测变更, 支持通知

# 响应配置变更通知

- Create an event handler

```
private void OnConfigurationChanged(object sender,  
ConfigurationChangedEventArgs args)  
{  
    // Take appropriate action, e.g. refresh settings  
}
```

- Register for event

```
ConfigurationManager.ConfigurationChanged += new  
ConfigurationChangedEventHandler(OnConfigurationChanged);
```



# 保护配置信息

您的潜力. 我们的动力

**Microsoft**  
微软(中国)有限公司

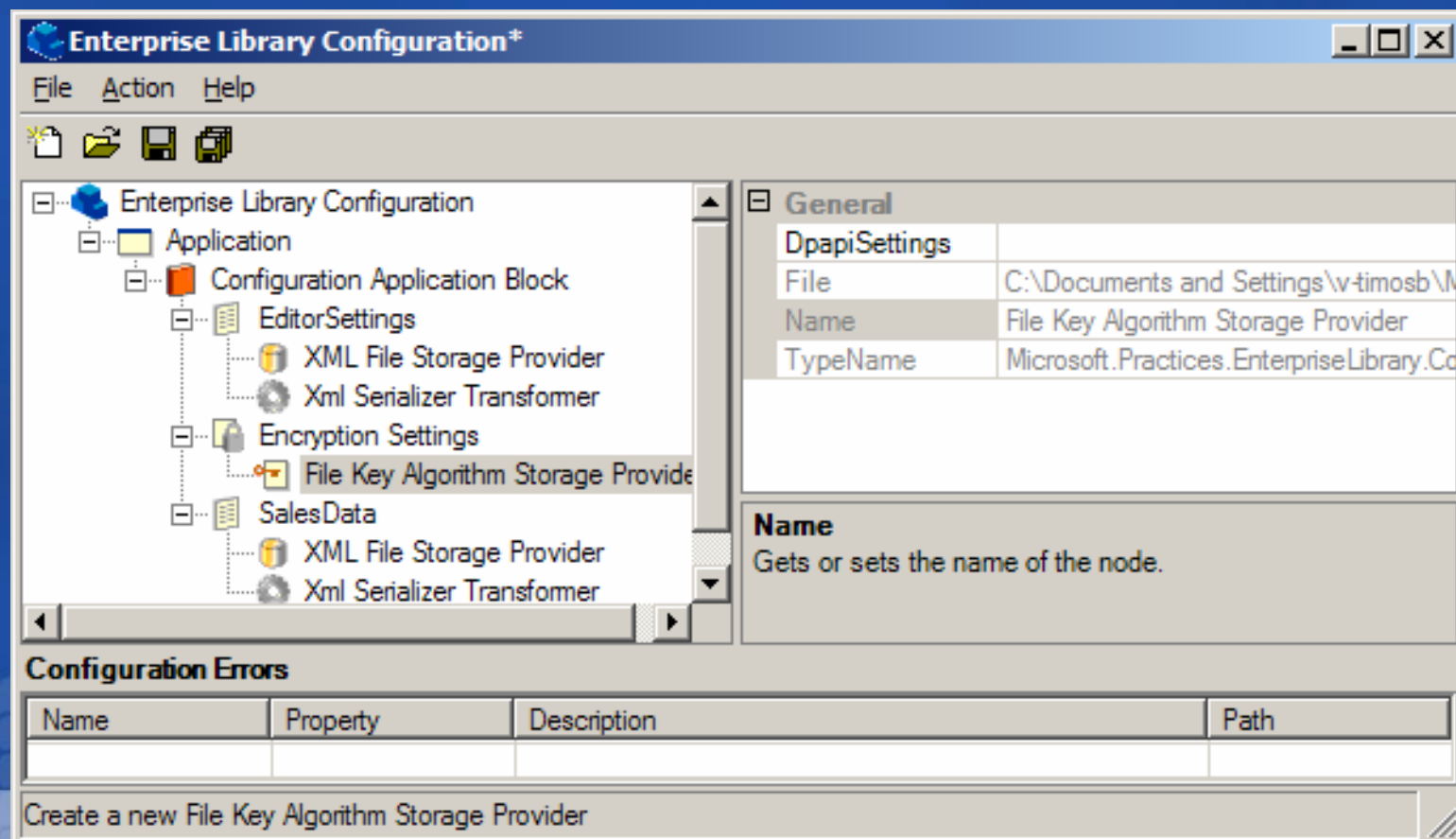
- XML 配置文件缺省保存为纯文本
- Config block 支持对存储的配置加密和解密
- 你挑选 symmetric encryption provider
- 密钥缺省保存为纯文本文件
- 你可以采用 ACLS, DPAPI 或其它方法保护密钥

# 保护配置信息

您的潜力. 我们的动力

**Microsoft**  
微软(中国)有限公司

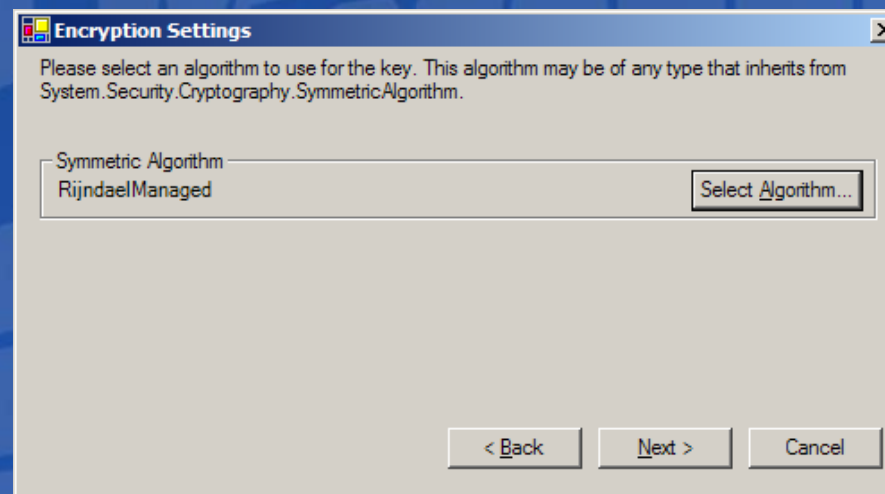
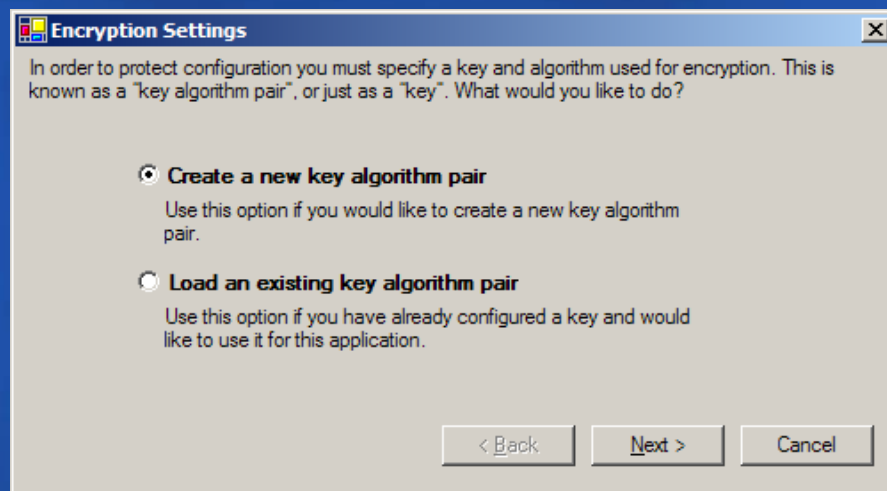
- 加密配置信息用来对任何配置区进行加密



# 保护数据库连接串

您的潜力. 我们的动力

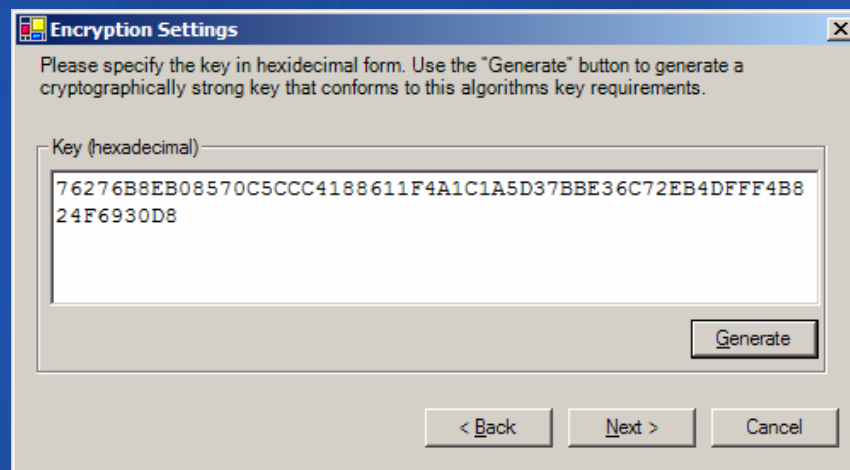
**Microsoft**  
微软(中国)有限公司



# 保护数据库连接串

您的潜力. 我们的动力

**Microsoft**  
微软(中国)有限公司



Encryption Settings

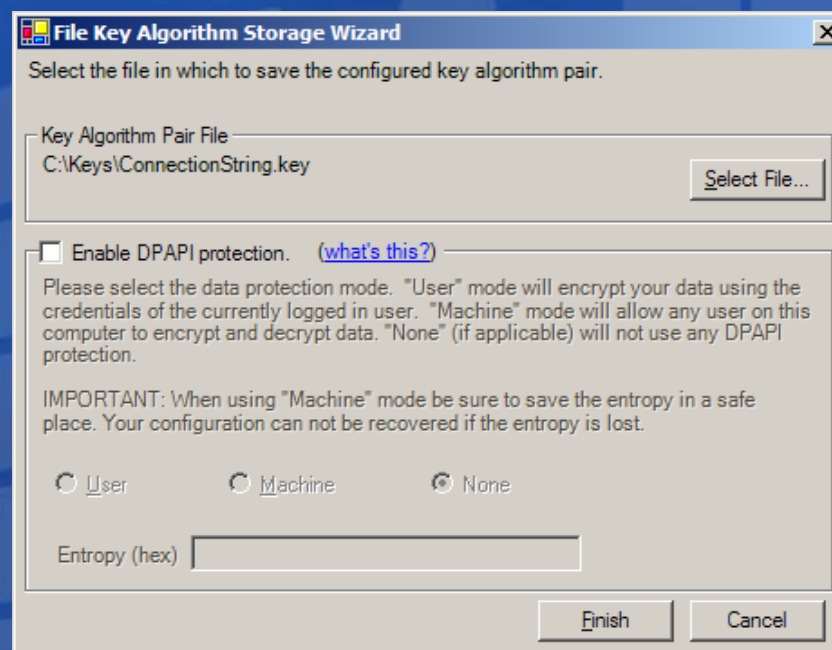
Please specify the key in hexadecimal form. Use the "Generate" button to generate a cryptographically strong key that conforms to this algorithm's key requirements.

Key (hexadecimal)

76276B8EB08570C5CCC4188611F4A1C1A5D37BBE36C72EB4DFFF4B8  
24F6930D8

Generate

< Back   Next >   Cancel



File Key Algorithm Storage Wizard

Select the file in which to save the configured key algorithm pair.

Key Algorithm Pair File  
C:\Keys\ConnectionString.key   Select File...

☐ Enable DPAPI protection. [\(what's this?\)](#)

Please select the data protection mode. "User" mode will encrypt your data using the credentials of the currently logged in user. "Machine" mode will allow any user on this computer to encrypt and decrypt data. "None" (if applicable) will not use any DPAPI protection.

IMPORTANT: When using "Machine" mode be sure to save the entropy in a safe place. Your configuration can not be recovered if the entropy is lost.

☐ User   ☐ Machine   ☒ None

Entropy (hex)

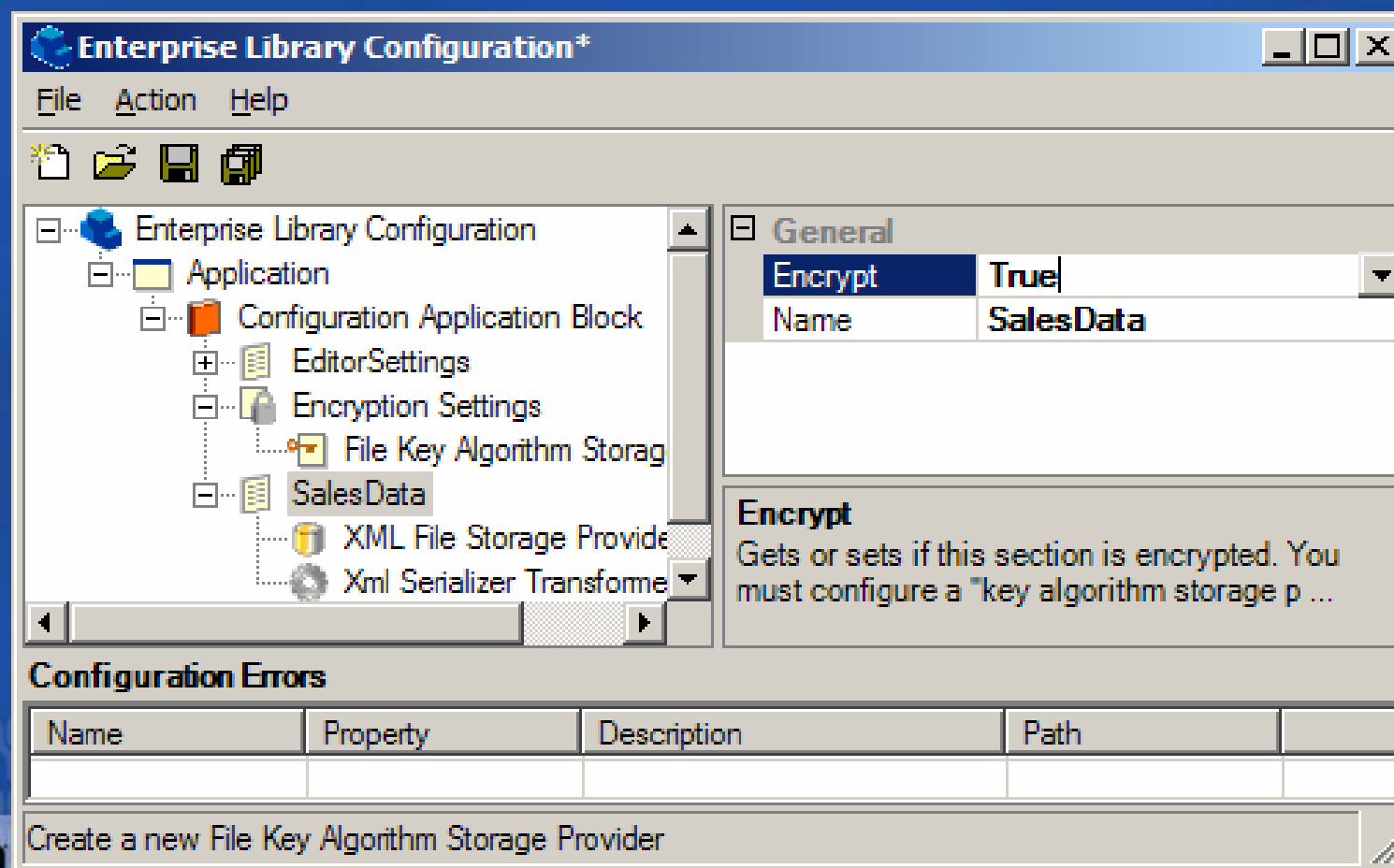
Finish   Cancel

# 保护数据库连接串

您的潜力. 我们的动力

**Microsoft**  
微软(中国)有限公司

- 每个配置区都有一个决定该配置区是否加密的属性





您的潜力. 我们的动力

**Microsoft**  
微软(中国)有限公司

# 面向高级开发人员的扩展机制

# 主要的扩展点

您的潜力. 我们的动力

**Microsoft**  
微软(中国)有限公司

- 定制 storage provider
  - 继承于 StorageProvider 类
  - 若写, 需实现 IStorageProviderWriter 接口
- 定制 transformer
  - 继承于 TransformerProvider
- 从社区站点下载补丁1462
- 还有...
  - 您拥有所有源代码!
  - 请在社区站点发布您的扩展和建议

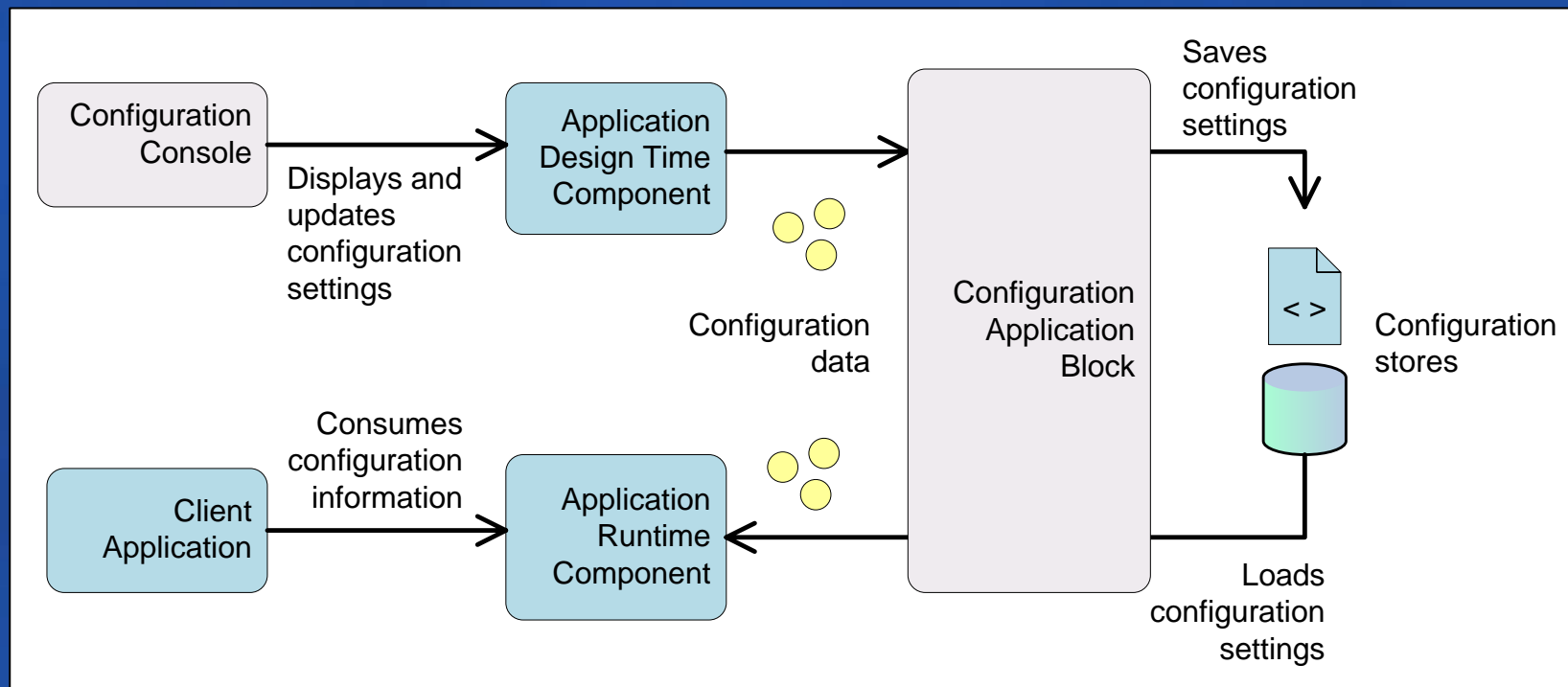
# 其它的 providers

您的潜力. 我们的动力

**Microsoft**  
微软(中国)有限公司

- App.config provider
  - 配置保存在 app.config or web.config
- SQL Server provider
- Registry provider
- 都在社区站点上:
  - <http://workspaces.gotdotnet.com/entlib>

# 将配置区展示给 Configuration Console



您的潜力. 我们的动力

**Microsoft**<sup>®</sup>  
微软(中国)有限公司

# Q&A



您的潜力，我们的动力

**Microsoft®**  
微软(中国)有限公司

**Microsoft®**

msdn  


**MSDN Webcasts**