



Using Enterprise Library with ASP.NET

Sound familiar?

- Writing a component to encapsulate data access
- Building a component that allows you to log errors to different sources
- Building framework / infrastructure components to generally simplify app development
- Searching on the internet thinking
 - Most applications need something like this
 - People must have written hundreds of things like this
 - I wish I could find a solution for this that I could reuse
- ...wishing Microsoft had done some of this for you?

Agenda

- In this session I will
 - Introduce Enterprise Library
 - Examine each of the “blocks” in Enterprise Library
 - Evaluate each “block” in the context of ASP.NET
- So that you will
 - Understand what Enterprise Library offers
 - Understand what problems each block solves
 - Know when to use Enterprise Library with ASP.NET

Who Am I

- Consultant
 - Vergent Software – www.vergentsoftware.com
- Architect
 - Guided Design – www.guideddesign.com
- Microsoft Regional Director
- Visual Developer MVP
- Writer
 - A {0} Developers Guide to ASP.NET, ADO.NET and XML
- Speaker
 - VSLive, TechEd, ASPLive TrackChair

Application Blocks



“Helpers”

- Classes which “help” you to do something but don’t impose an architecture
- Examples
 - Data Access
 - Exception Management
 - Configuration



“Mini Frameworks”

- Classes which help implement a design for a specific area of an application
- Examples
 - User Interface Process
 - Async Invocation Block
 - Offline Application Block

Application Block Feedback

- Make blocks consistent
- Make blocks work well together
- Minimize dependencies
 - On other blocks
 - On infrastructure
- Make it easier to configure blocks
- Make evaluation and understanding of blocks easier
- Make using blocks easier

Enterprise Library Philosophy

- **Consistency**
 - Application blocks should apply consistent design patterns and implementation approaches
- **Extensibility**
 - Application blocks must include extensibility points that allow developers to customize the behavior of the blocks by plugging in their own code, and can also be customized by directly modifying source code
- **Ease of Use**
 - Application blocks must be easy to use and should
 - Leverage a graphical configuration tool
 - Provide a simple installation procedure
 - Include clear and complete documentation and samples
- **Integration**
 - Application blocks should be designed to work well together and tested to make sure that they do. But it should also be possible to use the application blocks individually

Enterprise Library

- Enterprise Library is...
 - A library of application blocks which solve common challenges
 - A set of helper classes which work in any architectural style
 - Architectural guidance embodied in code which ships with full source allowing you to modify and extend
 - Available as a free download (once released)
- Enterprise Library is not...
 - A part of the .NET Framework
 - An application framework that imposes an architectural style
 - A Microsoft product with support, compatibility and localization
 - For sale

Data Access Block

- Reduces the need to write boiler plate code
- Helps maintain consistent data access practices
- Lessens the coupling between the code and the physical database target
- Abstracts differences between database types
- Provides an easy way to adjust database connection settings

Is it DAAB 3.0?

- Updated version of Data Access Block
 - Adds a database provider model
 - Ties into common configuration framework
- The object model has changed
 - In general you write even less code

Typical Usage

```
// Create an instance of the database object
Database shortcutDatabase = DatabaseFactory.CreateDatabase();

// Create the wrapper
DBCommandWrapper deleteWrapper =
shortcutDatabase.GetStoredProcCommandWrapper("DeleteShortcut");

// Setup the parameters
deleteWrapper.AddInParameter("@shortcut", DbType.String, shortcut);

// Execute the query
shortcutDatabase.ExecuteNonQuery(deleteWrapper);
```

Data Access Configuration

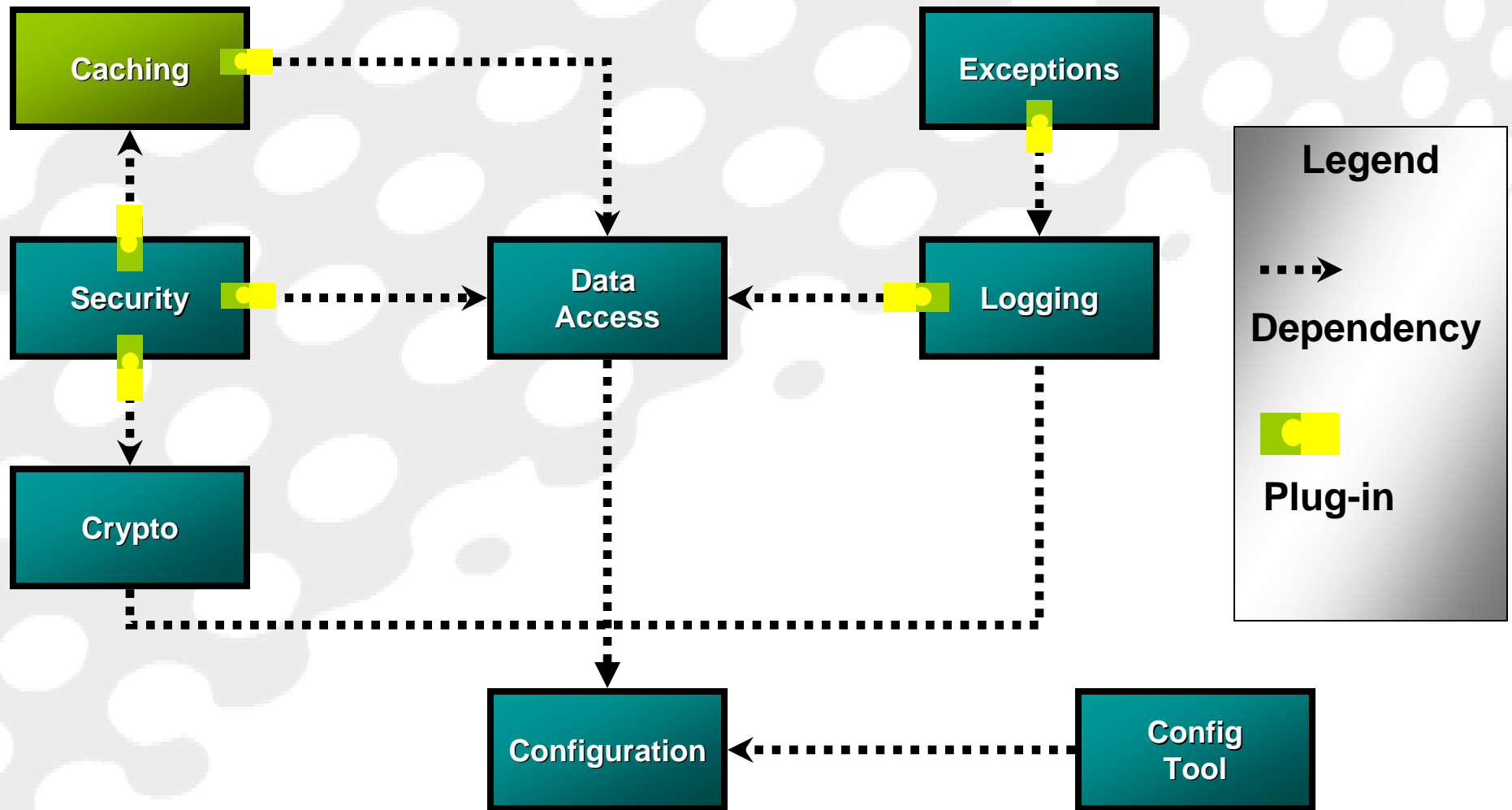
- Handles
 - Multiple connections strings
 - Multiple database instances
 - Changing database providers
- Provides a user interface for customers to do this
- Configuration stored in a separate file to easily do wholesale swaps



Data Access Configuration

Demo

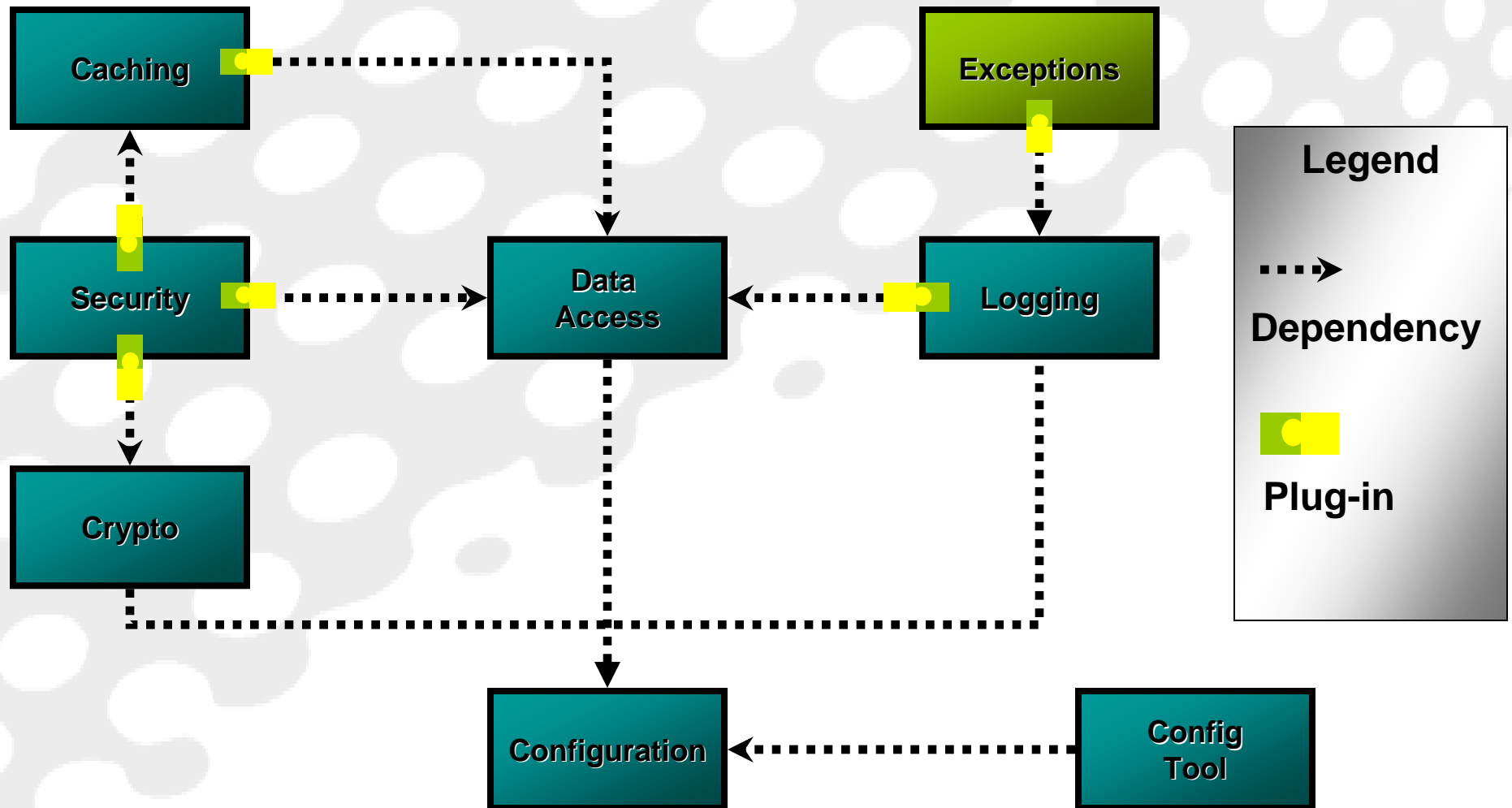
Enterprise Library 1.0



Why Use EntLib Caching?

- ASP.NET has caching built-in
- What does EntLib caching offer?
 - Custom expiration rules
 - Go beyond File or Time based expiration
 - Database expiration in 1.1
 - Custom backing stores
 - What if you don't want an in-memory or database backed cache?
 - Persistent file system based cache
- In General Use the ASP.NET Cache Instead

Enterprise Library 1.0



Exception Management Block

- Provides
 - A way to standardize exception handling throughout your application
 - A simple way to add boilerplate exception code
 - A way to log exception information
 - An easy way to adjust what is logged
 - A way to wrap and replace exceptions before they are propagated up the call stack
- Updated version of EMAB
 - API has changed

Documented Usage

```
try
{
    if(DAL.Shortcut.Exists(txtShortcut.Text))
    {
        IDataReader reader = DAL.Usage.GetReport(txtShortcut.Text);
        dgUsageReport.DataSource = reader;
        dgUsageReport.DataBind();
        reader.Close();
    }
}
catch(Exception ex)
{
    bool rethrow = ExceptionPolicy.HandleException(ex, "Website Policy");
    // handle the exception
    if(rethrow)
    {
        // Rethrow
        throw;
    }
}
```

Recommended Usage

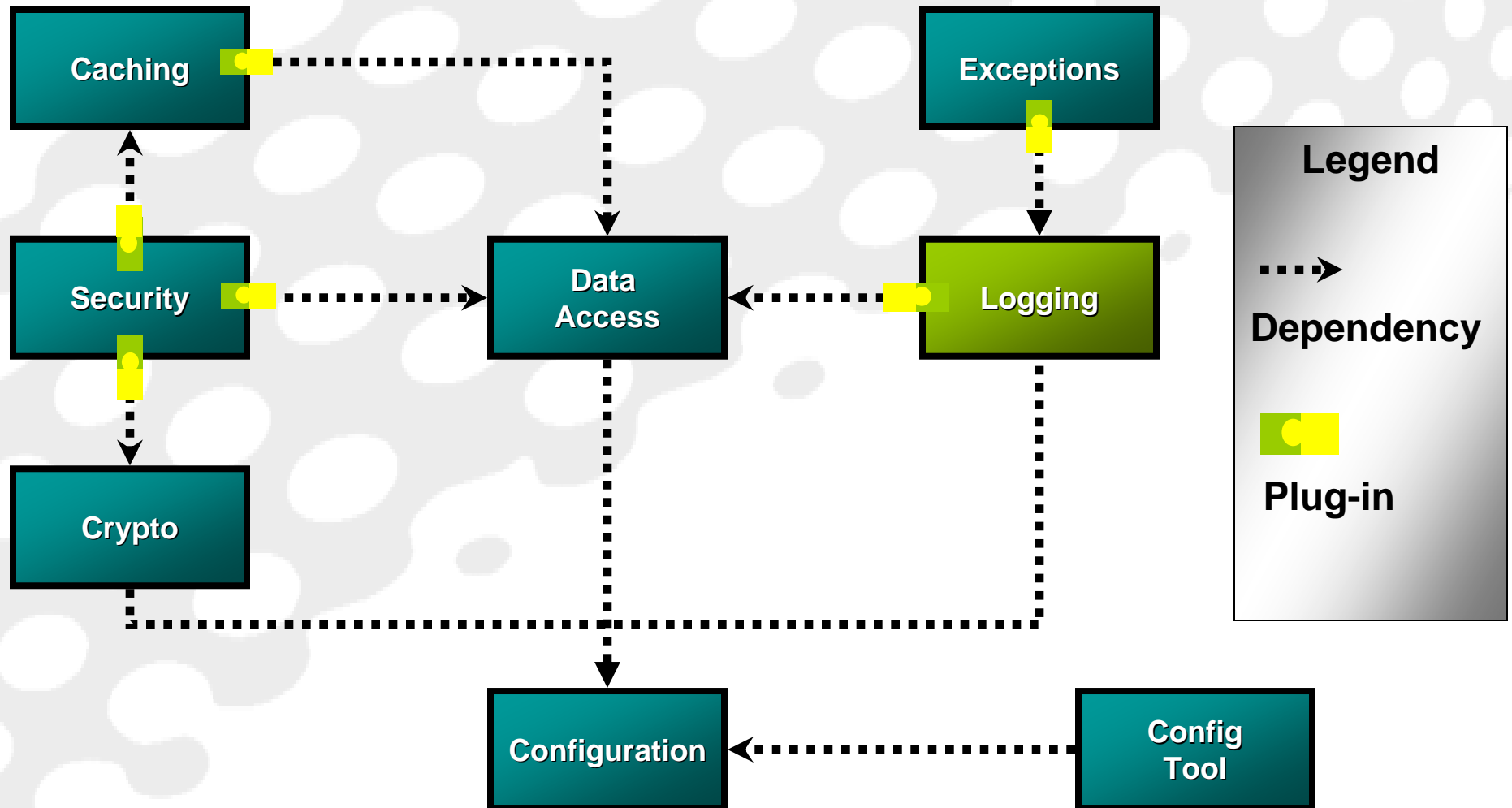
```
try
{
    if(DAL.Shortcut.Exists(txtShortcut.Text))
    {
        IDataReader reader = DAL.Usage.GetReport(txtShortcut.Text);
        dgUsageReport.DataSource = reader;
        dgUsageReport.DataBind();
        reader.Close();
    }
}
catch(Exception ex)
{
    ExceptionPolicy.HandleException(ex, "Website Policy");
    // Rethrow
    throw;
}
```



Exception Management Configuration

Demo

Enterprise Library 1.0



Logging Block

- Provides:
 - A way to log information about application execution
 - A way to abstract generation of log content from destination
 - An easy configuration interface to change what is logged where at runtime
- Replaces EIF and Logging Application Block
 - No dependency on EIF!

Supported Sinks

- Event Log
- Flat File
- WMI
- MSMQ
- Database
- Email
- Custom
 - Write your own

Simple Logging Example

```
Logger.Wri te("Appl i cati on Starti ng", "General ", 1);
```

Tracing Example

```
using(new Tracer("Trace", "Shortcut.Create"))
{
    // Create an instance of the database object
    Database shortcutDatabase = DatabaseFactory.CreateDatabase();

    // Create the wrapper
    DBCommandWrapper createWrapper =
shortcutDatabase.GetStoredProcCommandWrapper("InsertShortcut");

    // Setup the parameters
    createWrapper.AddOutParameter("@shortcut", DbType.String, 10);
    createWrapper.AddInParameter("@url", DbType.String, url);

    // Run the query
    shortcutDatabase.ExecuteNonQuery(createWrapper);

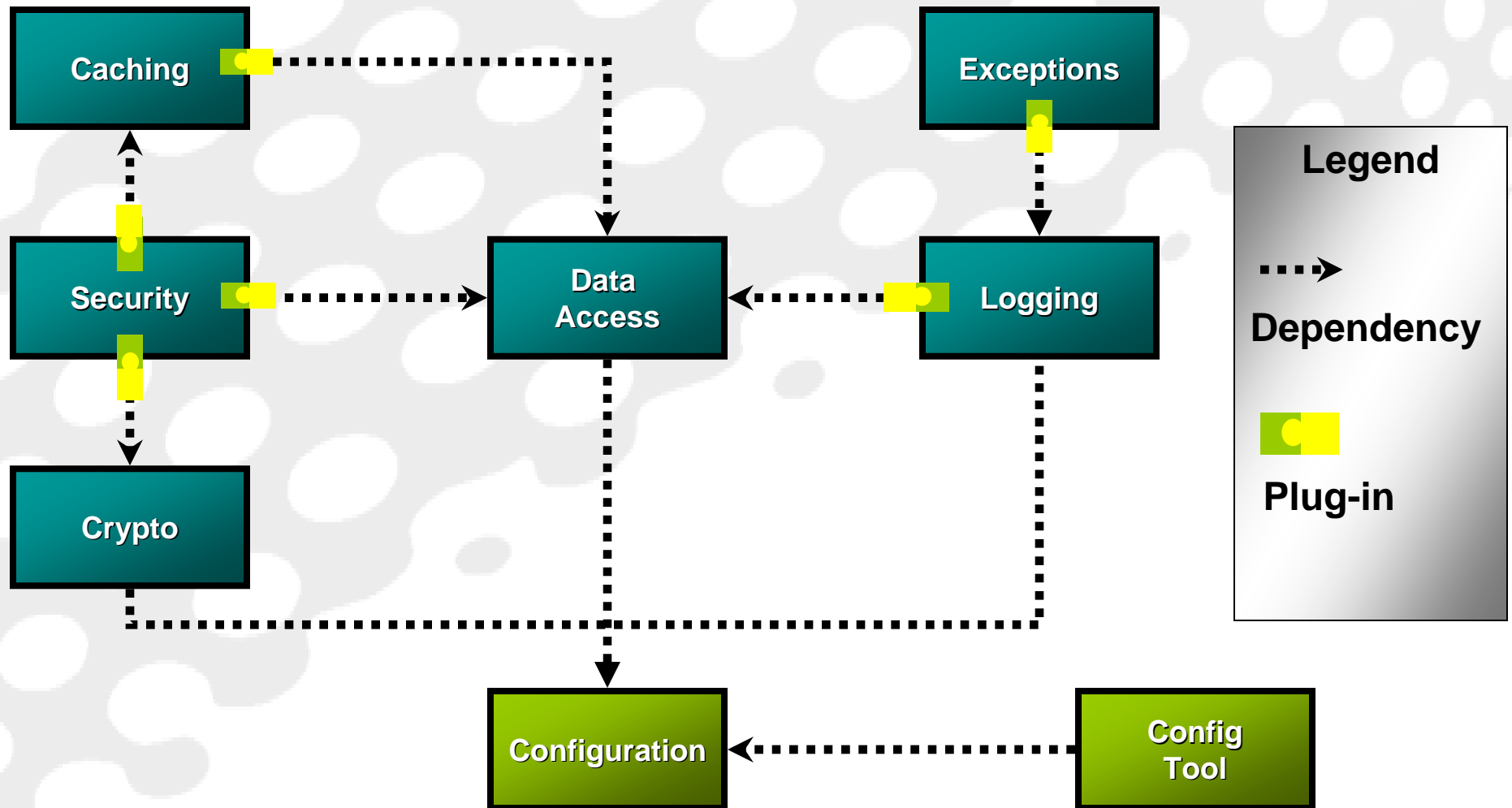
    // Get the shortcut
    return (string)createWrapper.GetParameterValue("@shortcut");
}
```



Logging Configuration

Demo

Enterprise Library 1.0



Configuration Block

- Provides:
 - A way to read AND **write** complex configuration data
 - A way to be notified of configuration data changes
 - A way to secure sensitive configuration information
 - An interface for administrators to change and validate configuration
 - Caveat that you have to create a designer...
- This is not the old Configuration Block
 - New API
 - Really goes Web.config one better

Typical Examples

- Reading

```
// get the site configuration
siteConfiguration =
(SiteConfiguration)ConfigurationManager.GetConfiguration(
    "siteConfiguration");
```

- Writing

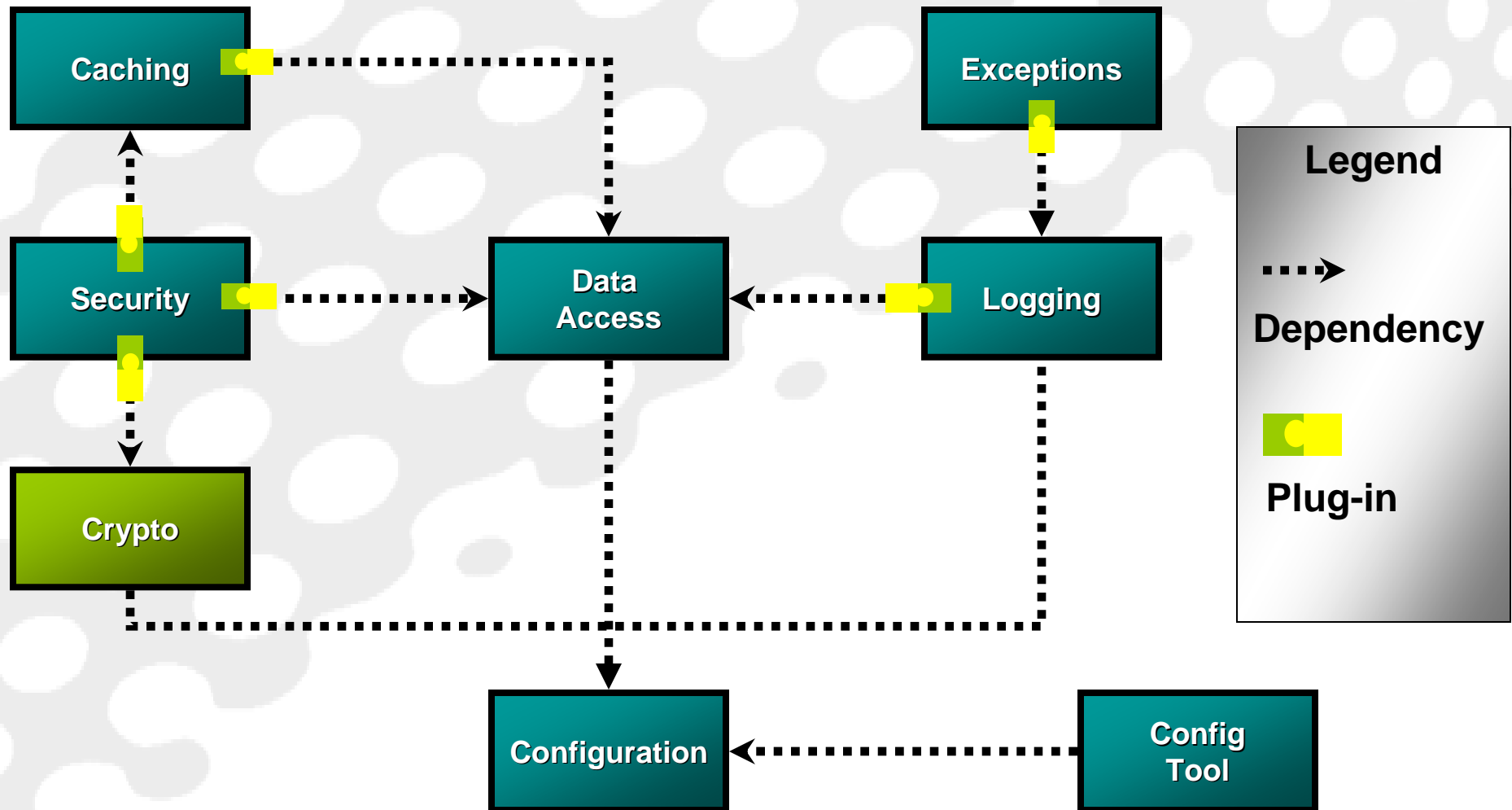
```
// set the site configuration
ConfigurationManager.WriteConfiguration("siteConfiguration",
    siteConfiguration);
```



Configuration

Demo

Enterprise Library 1.0



Cryptography Block

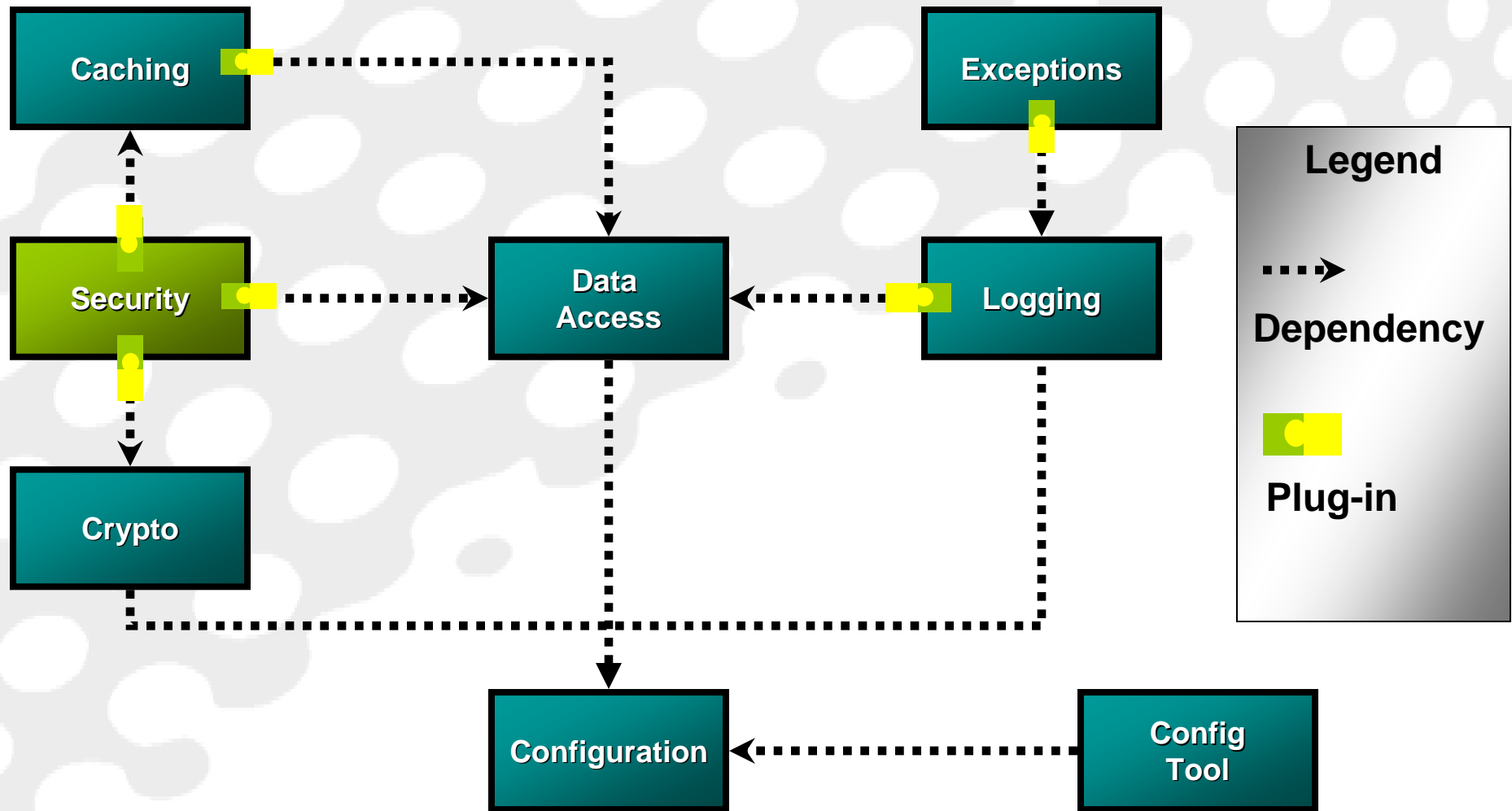
- Provides:
 - A simple wrapper for symmetric encryption/decryption
 - A simple DPAPI wrapper for keyless encryption/decryption on a single machine / user account
 - A way to hash passwords and compare hashes
 - A configuration interface for setting up and storing keys
 - All of these operations on both string and byte arrays!
- Replaces several DPAPI implementations floating around



Cryptography Configuration

Demo

Enterprise Library 1.0



Security Block

- Provides:
 - A way to authenticate users against a database or a custom provider
 - A way to authorize users against a database, AzMan or a custom provider
 - A way to manage roles for users with a database or a custom provider
 - A way to associate profile information with users from a database or a custom provider

Authentication / Roles

- Provides:
 - Easy way to authenticate users against Database
 - Easy way to associate roles with users
 - Windows forms application for creating, deleting and updating users
 - Security Database Console for creating, deleting and updating users
 - Build your own ASP.NET version
 - Built on Database Provider to support multiple backends

Authentication Sample

```
// Get the provider to authenticate with
IAuthenticationProvider authenticationProvider =
    AuthenticationFactory.GetAuthenticationProvider("Database Provider");

// An identity for later use
IIdentity identity;

// Create the credentials
NamePasswordCredential credentials = new NamePasswordCredential (
    txtEmail.Text, txtPassword.Text);

// authenticate
if(authenticationProvider.Authenticate(credentials, out identity))
{
    // Authorize and redirect the user
    System.Web.Security.FormsAuthentication.RedirectFromLoginPage(
        identity.Name, false);
}
```

Roles Sample

```
// Make sure the user has been authenticated
// This event fires for unauthenticated users also
if(Request.IsAuthenticated)
{
    // Get the users identity
    FormIdentity fiUser = (FormIdentity)User.Identity;

    // Load the roles out of the security subsystem
    IRoleProvider roleProvider =
    RoleFactory.GetRoleProvider("Role Database Provider");

    IPrincipal principal = roleProvider.GetRoles(fiUser);
    if (principal != null)
    {
        // Store the principal so we can use it in the page
        HttpContext.Current.User = principal;
    }
}
```

Authorization

- Not really intended for ASP.NET Users
- Already have this ability with the <authorization> tag in web.config
- Use the AzMan Provider to integrate with other AzMan based applications
- Use the rule provider if you need to be more granular than per page
 - Map ViewShortcuts to the users that are in the Admin role but not the IT role.
 - Allows you to change these mappings without a recompile.

Profiles

- Provides:
 - Easy way to associate personalization information with user
 - A way to associate information with an identity
 - Serialization of simple types to store
- ASP.NET 2.0 like profiles in 1.1

Profile Get Sample

```
// Create the profile provider
IProfileProvider profileProvider =
    ProfileFactory.GetProfileProvider(
        "Profile Database Provider");

// Read in the profile for user
UserProfile profile =
    profileProvider.GetProfile(
        HttpContext.Current.User.Identity) as UserProfile;

return profile;
```

Profile Set Example

```
// Create the profile provider
IProfileProvider profileProvider =
    ProfileFactory.GetProfileProvider(
        "Profile Database Provider");

// Save it
profileProvider.SetProfile(
    HttpContext.Current.User.Identity, profile);
```



Security Configuration

Demo



Questions?

ckinsman@vergentsoftware.com

www.vergentsoftware.com/blogs/ckinsman

VERGENT | SOFTWARE