Elmah Everywhere Documentation

Elmah Everywhere is an exception logging library for .NET, Silverlight, WPF, ASP.NET MVC and WCF that uses an ELMAH (Error Logging Modules and Handlers for ASP.NET).

For more detailed information how to configure and use Elmah Everywhere see source code and samples at <https://github.com/vincoss/vinco-logging-toolkit>.

# Default login details

* UserName : admin
* Password: p@ssword

# Silverlight configuration

To configure Silverlight exception logging you must add a reference to Elmah.Everywhere.Silverlight.dll and configure exception handler.

private static void SetUpExceptionHandler()

{

// Configure

var writter = new HttpExceptionWritter

{

RequestUri = new Uri("http://localhost:11079/error/log", UriKind.Absolute)

};

Uri uri = Application.Current.Host.Source;

var defaults = new ExceptionDefaults

{

Token = "Test-Token",

ApplicationName = "Silverlight-Sample",

Host = string.Format("{0}{1}{2}:{3}", uri.Scheme, Uri.SchemeDelimiter, uri.Host, uri.Port)

};

ExceptionHandler.WithParameters(defaults, writter);

}

**In Application constructor call handler setup method.**

public App()

{

private SetUpExceptionHandler();

}

**Add handler log into Application\_UnhandledException method**

private void Application\_UnhandledException(object sender, ApplicationUnhandledExceptionEventArgs e)

{

ExceptionHandler.Report(e.ExceptionObject, null);

}

# WPF configuration

To configure WPF exception logging you must add a reference to Elmah.Everywhere.dll and configure exception handler.

private static void SetUpExceptionHandler()

{

// Configure

var writter = new HttpExceptionWritter

{

RequestUri = new Uri("http://localhost:11079/error/log", UriKind.Absolute)

};

Uri uri = Application.Current.Host.Source;

var defaults = new ExceptionDefaults

{

Token = "Test-Token",

ApplicationName = "Silverlight-Sample",

Host = string.Format("{0}{1}{2}:{3}", uri.Scheme, Uri.SchemeDelimiter, uri.Host, uri.Port)

};

ExceptionHandler.WithParameters(defaults, writter);

ExceptionHandler.Attach(AppDomain.CurrentDomain);

}

**In Application constructor call handler setup method.**

public App()

{

private SetUpExceptionHandler();

}

**Remove handler exit.**

protected override void OnExit(ExitEventArgs e)

{

ExceptionHandler.Detach(AppDomain.CurrentDomain);

}

# WCF configuration with config file

To configure WCF exception logging you must add those settings into web.config file.

<system.serviceModel>

<serviceHostingEnvironment aspNetCompatibilityEnabled="true" multipleSiteBindingsEnabled="true" />

<extensions>

<behaviorExtensions>

<add name="elmah" type="Elmah.Everywhere.ServiceModel.ErrorBehaviorExtensionElement, Elmah.Everywhere"/>

</behaviorExtensions>

</extensions>

<behaviors>

<serviceBehaviors>

<behavior>

<elmah/>

</behavior>

</serviceBehaviors>

</behaviors>

</system.serviceModel>

# WCF configuration with behaviour attribute

Decorate your WCF or WCF RIA services with ServiceHttpErrorBehaviour attribute.

[ServiceHttpErrorBehavior(typeof(HttpErrorHandler))]

public class UserRegistrationService : DomainService

{

}