Reusable Library

A collection of reusable abstractions for enterprise application developer: caching, IoC, pagination, repository, application services, unit of work, background processing, exception trace policy, work item, etc. Integration with:

- Unity 2.0
- EntLib 5.0
- ASP.NET MVC 3.0
- WatiN 2.0
- Memcached 1.4.5

Modular design for both .net 2.0 and .net 4.

new – whats new since 1.1 release.

Abstractions

1. Bootstrapper with start up and shutdown tasks behaviour, stopwatch task
2. Caching:
    1. Cache startup task
    2. Cache helper that simplifies get and store code
    3. DataKey/**LazyDataKey**
    4. **LinkedCacheDependency**,
    5. Decorated, CacheInstrumentation (performance counters) new, CacheRetryClient new, Proxy new and Null cache
    6. AbstractCaching implements default algorithm for Increment new operation
2. Cryptography:
    1. symmetric algorithms **key vector providers** and helpers, **FNV32**/64, FNV32a, FNVModified hash algorithms

   2. SymmetricAlgorithmContext new, SynchronizedSymmetricAlgorithmContext new, **SymmetricAlgorithmContextPool** new
   3. CryptoTransformHelper new
3. Helpers:
   1. Assembly helper (version string format in short and long version),
   2. CollectionHelper, DictionaryHelper (TryGetValue, UniqueueValues), EnumerableHelper (ForEach, Count, ToDictionary, ToNameValueCollection, Translate), ListHelper (AddRange), NameValueCollectionHelper (HasKey, AllPairs new, ConvertToInt32, ConvertToBoolean,... handles required values)
   3. DateTimeHelper, Decimal to bytes and back converter
   4. EnumHelper (HasAll / HasAny flags)
   5. ExceptionHelper (list all inner exceptions), GuidHelper (IsEmpty, **Shrink** to a compact string of length 22)
   6. IpNumberHelper (ToIpNumber from string, ToIpString, ToIpLong)
   7. PasswordHelper (generate a random password for a given character groups)
   8. ObjectHelper (PropertiesToNameValueCollection)
   9. PasswordHelper, generate strong password easily according to given password letters
   10. **RandomHelper** (Seed, NextChar, NextString, NextWord, Next<T> from a sequence of T, NextSentence, NextSentences, NextDate, NextSubstring, NextStartsWith, NextBoolean, FirstRandom from a sequence, TimesRandom, Shuffle sequence, NextSequence based of Func<T>)
   11. UriHelper, parse/build uri query string
   12. **RetryHelper** new, executes a given function Func2<bool> until succeed or the following reached: MaxRetryCount or RetryTimeout. There is a delay between retries (RetryDelay). TimeoutException is thrown in case of RetryFails is true. Uses RetryOptions.
   13. BitConverterHelper new, get bytes from a string encoded with BitConverter.GetString.
   14. StringHelper (NullSafe, NullEmpty, WrapAt, AllOf, StripHtml, StipNewLine new, StripWhitespace new, Capitalize, Contains, Left, Right, Repeat, Join, ParseOptions, Hash)
4. IO:
   1. IBinaryWriter, IBinaryReader and implementation for Stream
   2. BinaryReaderHelper (ReadTo byte buffer, ReadToken based on predicate)
   3. DecoratedStream
5. IoC
   1. Dependency resolver (**IoC**)
6. Various model abstractions:
   1. Buffer (with EnsureMore) and **BufferHive** (a pool of buffers)
   2. ByteArrayComparer
   3. **ValueObject**, **LazyObject** and LazyPrincipal, Disposable, IValidationState, Pair, Range
   4. **Paged List** and settings
   5. Encoding: IEncoder, Base64Encoder, **HashEncoder**, TextEncoder

6. **Pooling**:
    1. DecoratedPool, EagerPool, **LazyPool**, ManagedPool, PooledPool, StackPool
    2. **IdleTimeoutPool**
    3. **KetamaPool** (and SynchronizedKetamaPool)
    4. **SynchronizedPool, WaitPool**
7. **Topic**, invoke strategies: publisher, thread pool, synchronization context
8. **Key** (an easy way to get a string key out from model to be used as a key for cache), IKeyProvider

7. Net:
    1. **Client** (abstraction that relates to client in client-server interaction scenario), **Distributed** (a pool of Clients), ConnectionOptions, DistributedOptions
    2. ClientFactory
    3. BigEndianConverter
    4. Dns Helper (a timed query for dns information)
    5. SocketHelper (a timed connect)
    6. TcpClientConnection (to be used by Client for client-server network communication)
    7. HttpClient (GET/POST/HEAD form submitting)
8. Repository pattern:
    1. IRetrieveRepository and IRetrieveMultipleRepository, DbConnectionStringProvider
9. Serialization:
    1. Object state
    2. Formatters:
        1. Decorated, Array, Null, Simple (Int32, Boolean, etc), Runtime
        2. **Compressed** (deflate)
        3. **Encrypted** (symmetric)
10. Services:
    1. **ApplicatService** acting as a bridge between the client code and Repository, includes validation, error handling policy
    2. **MailService**
    3. TopicCatalog
    4. **UnitOfWork** pattern
    5. **RunOnceService** new (ensures that give operation is executed only once utilizing a counter in memcached)
11. Threading:
    1. Abstract background task, plus bootstrapper start all / shutdown all background tasks
    2. LockScope, **MonitorLockScope**, Reader/Upgrade/Writer LockScope
    3. **WaitAsyncResult**
    4. AsyncHelper implementing **FireAndForget** pattern using ThreadPool
12. Tracing (helpers to System.Diagnostics)
    1. Exception trace policies:
        1. ErrorFormatter
        2. Loaded Assemblies
        3. **EventLog**
        4. **Machine/Process information**

5. **Mail**
6. Ignore (exclude certain exceptions from error reports)
7. Performance Counter new (queries system performance counters)
8. TransferExceptionHandler new

2. **PerformanceCounter**
3. Console trace listener
4. TraceHelper

13. WorkItem: a background processing unit with rules sequential workflow

Captcha new

1. Extensible through the following interfaces (giving almost unlimited flexibility concerning what the actual captcha is: jpeg, flash, etc):
    1. ICaptchaFactory
    2. ICaptchaValidator
    3. IContentProvider
    4. IErrorProvider
    5. IImageCodecInfo
    6. IGraphicsDrawing
    7. IChallengeCodeProvider
    8. ITuringNumberProvider
    9. IVaryByCustomProvider (used by CaptchaHandler)
2. Challenge cache is configurable so gives you ability to use captcha in distributed scenario (by using SessionCache or Memchached client implementations of ICache).
3. Capable respond to multiple url paths giving you ability to use multiple captchas (with respect ot look and feel) in a single application.
4. Configuration Options (see CaptchaOptionNames and CaptchaOptionDefaults).
5. Comes with two implementations:
    1. SimpleCaptchaHandler (renders a new captcha on each request).
    2. CaptchaHandler (high performance captcha utilizing asp.net caching).
6. CaptchaInstrumentationProvider new (performance counters)

Consider MaxVaryCacheSize/VaryByCacheLifetime as a minimal throughput rate under which the captch cache is "activated" (applies to default captcha factory).

Memcached

1. Text Protocol
2. Binary Protocol
3. Operations: get, set, delete, increment new
4. Ketama balancing
5. Threshold Compression
6. Encryption (symmetric algorithms)
7. Advanced pooling

History Log new

Easy way to organize a user, service, audit history log:

1. Repository
    1. Can be customized for own database schema
    2. SQL schema and mapping provided
2. Agent
    1. Asynchronous
    2. Internal message queue that is flushed regularly HistoryLogWorkItem
3. Service
    1. A way to search history log
    2. Paging support
4. User interface
    1. Messages can be internationalized

Host

1. Different types of applications: console, singleton, service host, timed
2. Service host installer

Unity

1. Implementation of Dependency resolver (IoC) for Unity Unity 2.0
2. LifetimeManagers: WorkItem, WebRequest

Enterprise Library

1. Caching (integration with Caching Application Block, EntLib 5.0
2. ValidationService (integration with Validation Application Block, EntLib 5.0
3. Validators: TrueValidator, ObjectValidator
4. **ValidatorFactoryCache**, this one resolves EntLib 5.0 validation rulesets per current thread culture info.

Quality Assurance

1. **Profiling** (a single test, stress test profiling on a number of threads using a random sequence)
2. Repository test

Supplemental

1. Collections extensions, Enumerable extension, Pagination support, etc
2. UnitOfWork pattern integrated with DataContext
3. **ISpecification pattern** with expression syntax abstraction
4. Assembly extensions for version formatter

5. Repository extensions for expressions syntax evaluation using Specification pattern
6. System extensions: DateTime, Random, String, TimeSpan, etc
7. Everything you have in Abstractions that can have an extension is here

WatiN

1. Concept of Application and ApplicationLifeTimeContainer
2. Few extensions to Element, IElementContainer, etc

Web

1. Integration with ASP.NET MVC, UnityControllerFactory, PerWebRequest Lifetime Manager
2. Modules:
    1. StopwatchModule
    2. AjaxRedirectModule
    3. **CompressModule**
    4. **DenialModule**
    5. **ExceptionPolicyModule**
    6. ShrinkModule
    7. **ThrottleModule** new (limit request rate during a period of time)
    8. NoServerHeaderModule new (removes server header for IIS 7)
    9. IHttpFilerIgnore new (marker interface to skip applying module filters)
3. WebCache, SessionCache new for ICache (from abstractions)
4. Exception Handlers:
    1. **ErrorThrottleExceptionHandler** (If there are more than {ErrorRate} errors during {ThrottlePeriod} than block all incoming requests with HttpForbidden for {BlockPeriod})
    2. HttpContextExceptionHandler (let you include any Http ServerVariables into error report)
    3. HttpExceptionHandler (let you ignore particular http error codes)
    4. IgnoreLocalExceptionHandler (ignore error reporting in local testing)
    5. RedirectExceptionHandler new (redirects to a given page in case of exception match)
5. FormsAuthenticationService
6. RemoteLocationProvider
7. Routing
    1. ChoiceRouteConstraint new, **DomainRouteConstraint** new, **SchemeRouteConstraint** new (http/https)
    2. MapRoute to IRouteHandler new

Web Mvc

1. Integration with ASP.NET MVC, UnityControllerFactory, PerWebRequest Lifetime Manager
2. **Internationalization/Localization** support for your asp.net mvc application, includes **SEO routing**, etc

3. Improved FormsAuthenticationService, that let you **store user data in ASP.NET auth cookie**
4. View data abstractions (DetailsViewData, ListViewData, SearchViewData)
5. Various helpers: menu item, message, paging (pager plus page size)
6. Default and ignore routes start up tasks
7. BadRequest and FileNotFound results, Compression
8. AbstractController with RedirectTo<Controller> (works in ajax), AlternatePartialView, etc
9. IEnumerable extensions to SelectList utilizing ValueObject pattern
10. Validation of **antiforgery tocken with dynamic salt** new (you can include some extra information like id, etc... something that you usually add into hidden input)
11. **AbsoluteRoute** new extensions to UrlHelper and HtmlHelper
12. **HttpResponseSubstitution**:
    1. SubstitutionHelper new
    2. HttpResponseSubstitutionRouteHandler new