LT.Code使用说明书

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **文件编号** | LZ-FZ-201807 | | **文件版本** | | V1.0 |
| **编写单位** | 青岛励图智能科技有限公司 | | | | |
| **编 写 人** | 张钊 | **编写日期** | | 2018-07-05 | |
| **审 核 人** |  | **审核日期** | |  | |

**修订记录**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **版本** | **修订内容** | **修订人** | **审批人** | **修订日期** |
| V1.0 | 新建文档 |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**目 录**

[1 概述 2](#_Toc2531)

[1.1 文档说明 2](#_Toc16116)

[1.2 类库说明 2](#_Toc12694)

[2 功能说明 2](#_Toc31076)

[2.1 缓存的操作 2](#_Toc841)

[2.2 压缩与解压缩操作 3](#_Toc16475)

[2.3 配置文件的操作 4](#_Toc15573)

[2.4 格式转换 4](#_Toc8328)

[2.5 文件的操作 7](#_Toc19832)

[2.6 Json的转换 9](#_Toc21563)

[2.7 日志的操作 10](#_Toc5673)

[2.8 网络信息 11](#_Toc4331)

[2.9 加密操作 12](#_Toc21876)

[2.10 校验 14](#_Toc17947)

[2.11 前端的操作类 16](#_Toc25424)

[2.12 其他的操作类 18](#_Toc97)

# 概述

## 文档说明

LT.Code类库，为工具类以及特殊功能类的封装类库。本文档主要对类库中包含的功能以及使用方法进行说明，以便于使用。

## 类库说明

LT.Code目录下主要存放了一些常用的工具类，例如对缓存信息的操纵，字符串以及文字等的压缩、对配置文件的操作、转化方法、文件操作、json转换等。

# 功能说明

## 缓存的操作

ICache为接口类，包含以下几种方法如表 2-1所示。

表 2-1 ICache类说明

|  |
| --- |
| **方法** |
| T GetCache<T>(string cacheKey) |
| WriteCache<T>(T value, string cacheKey) |
| WriteCache<T>(T value, string cacheKey, DateTime expireTime) |
| RemoveCach(string cacheKey) |
| RemoveCache() |

Cache（缓存类）是ICache的实现类，方法说明如表 2-2所示：

表 2-2 Cache类说明

|  |  |
| --- | --- |
| **方法** | **说明** |
| T GetCache<T>(string cacheKey) | 获取缓存数据的方法，T指的是对象，cacheKey指的key值。 |
| WriteCache<T>(T value, string cacheKey) | 写入缓存信息的方法，T指的是缓存值的类型，value指的是缓存值，cacheKey指的缓存key值。 |
| WriteCache<T>(T value, string cacheKey, DateTime expireTime) | 写入缓存信息的方法，T指的是缓存值的类型，value指的是缓存值，cacheKey指的缓存key值，expireTime指的是超时时间。 |
| RemoveCach(string cacheKey) | 根据key清除缓存的方法，cacheKey指的是key值。 |
| RemoveCache() | 清除全部缓存信息的方法。 |

CacheFactory为工厂类，实现类在这里使用。

## 压缩与解压缩操作

GZip类是主要是字符串压缩和stream流的压缩，通过GZip方式对字符串大小进行压缩，方法说明如表 2-3所示：

表 2-3 GZip类说明

|  |  |
| --- | --- |
| **方法** | **说明** |
| CompressString(string str) | 字符串压缩为字符串(Base64编码)的方法 |
| CompressStringByte(string str) | 字符串压缩为字节 |
| DecompressString(string str) | 解压字符串 |
| Compress(byte[] data) | 字符串压缩 |
| Decompress(byte[] data) | 字符串解压缩 |
| Compress(Stream stream) | Stream流压缩的，stream指的是流 |
| Decompress(Stream stream) | Stream流解压缩 |
| StreamToBytes(Stream stream) | 流转换为字节流 |

RarHelp类是rar格式的压缩、解压操作，通过系统配置表找到rar安装目录，调用rar工具进行压缩解压操作， 可以通过文件路径以及压缩的路径进行压缩与解压操作，方法说明如表 2-4所示：

表 2-4 RarHelp类说明

|  |  |
| --- | --- |
| **方法** | **说明** |
| EnZip(string zipname, string zippath, string dirpath) | Rar的文件压缩的方法，zipname是要压缩的文件名，zippath是要压缩的文件目录，dirpath是初始目录。 |
| DeZip(string zipname, string zippath) | Rar的文件解压缩的方法，zipname是要压缩的文件名，zippath是要压缩的文件目录。 |

SharpZip类是Zip格式的压缩解压操作，可以根据文件以及文件路径实现对文件的压缩与解压操作，方法说明如表 2-5所示：

表 2-5 SharpZip类说明

|  |  |
| --- | --- |
| **方法** | **说明** |
| ZipFiles(string filename, string directory) | 文件压缩的方法，filename指的是 压缩后的文件名(包含物理路径)，directory指的是待压缩的文件夹(包含物理路径) |
| UnZipFiles(string file, string dir) | 解压缩的方法，file指的是待解压文件名(包含物理路径)，dir指的是解压到哪个目录中(包含物理路径) |
| Zip(String fileToZip, String zipedFile, int level) | 文件压缩的方法，fileToZip指的是待压缩的文件目录，zipedFile指的是生成的目标文件，level指的层级 |
| UnZip(string fileToUpZip, string zipedFolder) | 解压方法，fileToUpZip指的是待解压的文件，zipedFolder指的是解压目标存放目录 |

## 配置文件的操作

在Configs类是对于系统的配置文件数据进行操作的，使用说明如表 2-6所示：

表 2-6 Configs类说明

|  |  |
| --- | --- |
| **方法** | **说明** |
| GetSetting(string key) | 获取当前配置的值 |
| LoadAllConfigs() | 加载所有的appSetting下的配置项内容 |
| GetSettingByKey(string key) | 根据Key取Value值，直接在配置中获取 |
| SetSettingValue(string key, string value) | 根据Key修改Value |

IniHelper类是Ini文件的操作类，使用说明如表 2-7：

表 2-7 IniHelper类说明

|  |  |
| --- | --- |
| **方法** | **说明** |
| GetIniValue(string section, string key, string filePath) | 读取INI的方法，section指的是分组节点，key指的是关键字，filePath指的是文件的路径。 |
| WriteIniValue(string section, string key, string value, string filePath) | 写INI文件的方法，section指的是分组节点，key指的是关键字，value指的是值，filePath指的是文件的路径。 |

## 格式转换

在Ext类是转化方法的扩展，使用方法如表 2-8所示：

表 2-8 Ext类说明

|  |  |
| --- | --- |
| **方法** | **说明** |
| **数值转换** | |
| ToInt() | 转换为整型。 |
| ToIntOrNull() | 转换为可空整型。 |
| ToDouble() | 转换为双精度浮点数。 |
| ToDouble(this object data, int digits) | 转换为双精度浮点数,并按指定的小数位4舍5入，data指的是数据，digits指的是小数位数。 |
| ToDoubleOrNull() | 转换为可空双精度浮点数。 |
| ToDecimal() | 转换为高精度浮点数。 |
| ToDecimal(this object data, int digits) | 转换为高精度浮点数,并按指定的小数位4舍5入，data指的是数据，digits指的是小数位数。 |
| ToDecimalOrNull() | 转换为可空高精度浮点数。 |
| ToDecimalOrNull(this object data, int digits) | 转换为可空高精度浮点数,并按指定的小数位4舍5入,data指的是数据，digits指的是小数位数。 |
| **日期转换** | |
| ToDate() | 转换为日期。 |
| ToDateOrNull() | 转换为可空日期。 |
| **布尔转换** | |
| ToBool() | 转换为布尔值。 |
| GetBool() | 获取布尔值。 |
| ToBoolOrNull() | 转换为可空布尔值。 |
| **字符串转换** | |
| ToString() | 转换为字符串。 |
| T SafeValue<T>(this T? value) | 安全返回值的方法，value指的是可空值，T指的是类型。 |
| IsEmpty(this string value) | 是否为空 |
| IsEmpty(this Guid? value) | 是否为空 |
| IsEmpty(this Guid value) | 是否为空 |
| IsEmpty(this object value) | 是否为空 |

时间格式的转化方法说明如表 2-9所示：

表 2-9 时间格式的转化方法

|  |  |
| --- | --- |
| **方法** | **说明** |
| ToDateTimeString(this DateTime dateTime, bool isRemoveSecond = false) | 获取格式化字符串，带时分秒，格式："yyyy-MM-dd HH:mm:ss"，isRemoveSecond指的是“是否移除秒”。 |
| ToDateTimeString(this DateTime? dateTime, bool isRemoveSecond = false) | 获取格式化可为空的字符串，带时分秒，格式："yyyy-MM-dd HH:mm:ss"。isRemoveSecond指的是“是否移除秒”。 |
| ToDateString(this DateTime dateTime) | 获取格式化字符串，不带时分秒，格式："yyyy-MM-dd"。 |
| ToDateString() | 获取当前时间格式化字符串，不带时分秒，格式："yyyy-MM-dd"。 |
| ToDateString(this DateTime? dateTime) | 获取格式化可为空的字符串，不带时分秒，格式："yyyy-MM-dd"。 |
| ToTimeString(this DateTime dateTime) | 获取格式化字符串，不带年月日，格式："HH:mm:ss"。 |
| ToTimeString(this DateTime? dateTime) | 获取格式化可为空字符串，不带年月日，格式："HH:mm:ss"。 |
| ToMillisecondString(this DateTime dateTime) | 获取格式化字符串，带毫秒，格式："yyyy-MM-dd HH:mm:ss.fff"。 |
| ToMillisecondString(this DateTime? dateTime) | 获取格式化可为空的字符串，带毫秒，格式："yyyy-MM-dd HH:mm:ss.fff"。 |
| ToChineseDateString(this DateTime dateTime) | 获取格式化字符串，不带时分秒，格式："yyyy年MM月dd日"。 |
| ToChineseDateString(this DateTime? dateTime) | 获取格式化可为空的字符串，不带时分秒，格式："yyyy年MM月dd日"。 |
| ToChineseDateTimeString(this DateTime dateTime, bool isRemoveSecond = false) | 获取格式化字符串，带时分秒，格式："yyyy年MM月dd日 HH时mm分"。 |
| ToChineseDateTimeString(this DateTime? dateTime, bool isRemoveSecond = false) | 获取格式化可为空的字符串，带时分秒，格式："yyyy年MM月dd日 HH时mm分"。 |

其他的一些转化的扩展方法，方法说明如表 2-10所示：

表 2-10 其他转化的扩展方法

|  |  |
| --- | --- |
| **方法** | **说明** |
| Description(this bool value) | 获取描述。 |
| Description(this bool? value) | 获取描述，可为空。 |
| Format(this int number, string defaultValue = "") | 获取格式化字符串方法，number指的是数值，defaultValue指的是空值显示的默认文本。 |
| Format(this int? number, string defaultValue = "") | 获取格式化字符串方法，number指的是数值，defaultValue指的是空值显示的默认文本。 |
| Format(this decimal number, string defaultValue = "") | 获取格式化字符串方法，number指的是数值，defaultValue指的是空值显示的默认文本。 |
| Format(this decimal? number, string defaultValue = "") | 获取格式化字符串方法，number指的是数值，defaultValue指的是空值显示的默认文本。 |
| Format(this double number, string defaultValue = "") | 获取格式化字符串方法，number指的是数值，defaultValue指的是空值显示的默认文本。 |
| Format(this double? number, string defaultValue = "") | 获取格式化字符串方法，number指的是数值，defaultValue指的是空值显示的默认文本。 |
| FormatRmb(this decimal number) | 获取格式化字符串,带￥ |
| FormatRmb(this decimal? number) | 获取格式化字符串,带￥ |
| FormatPercent(this decimal number) | 获取格式化字符串,带% |
| FormatPercent(this decimal? number) | 获取格式化字符串,带% |
| FormatPercent(this double number) | 获取格式化字符串,带% |
| FormatPercent(this double? number) | 获取格式化字符串,带% |

List转化的扩展，方法说明如表 2-11所示：

表 2-11 List转化的扩展方法

|  |  |
| --- | --- |
| **方法** | **说明** |
| GetPage<T>(this List<T> data, int pageIndex, int pageSize, out int allPage) | 获取表里某页的数据，pageIndex是指当前页，pageSize是指分页大小，allPage是指返回总页数。 |
| IListToList<T>(IList list) | IList转成List<T>。 |

数据处理的转化，方法如表 2-12所示：

表 2-12 数据处理的转化方法

|  |  |
| --- | --- |
| **方法** | **说明** |
| GetPage(this DataTable data, int pageIndex, int pageSize, out int allPage) | 获取表里某页的数据，pageIndex是指当前页，pageSize是指分页大小，allPage是指返回总页数。 |
| GetDataFilter(DataTable data, string condition) | 根据字段过滤表的内容，condition是指条件 |

## 文件的操作

FileDownHelper类是对文件下载的方法进行封装，方法说明如表 2-13所示：

表 2-13 FileDownHelper类说明

|  |  |
| --- | --- |
| **方法** | **说明** |
| FileNameExtension(string fileName) |  |
| MapPathFile(string fileName) |  |
| FileExists(string fileName) | 判断文件是否存在 |
| DownLoadold(string fileName, string name) |  |
| DownLoad(string fileName) | 下载文件 |
| Upload(HttpRequestBase request, HttpServerUtilityBase server, string imgPath) | 图片上传，imgPath文件路径 |

FileHelper类是文件操作的方法工具，方法说明如表 2-14所示：

表 2-14 FileHelper类说明

|  |  |
| --- | --- |
| **方法** | **说明** |
| **目录检测相关操作** | |
| IsExistDirectory(string directoryPath) | 检测指定目录是否存在，directoryPath是指目录的绝对路径 |
| IsEmptyDirectory(string directoryPath) | 检测指定目录是否为空，directoryPath是指定目录的绝对路径 |
| DirectoryContains(string directoryPath, string searchPattern) | 检测指定目录中是否存在指定的文件,若要搜索子目录请使用重载方法。directoryPath是指定目录的绝对路径，searchPattern是指模式字符串，"\*"代表0或N个字符，"?"代表1个字符。 |
| DirectoryContains(string directoryPath, string searchPattern, bool isSearchChild) | 检测指定目录中是否存在指定的文件。directoryPath是指定目录的绝对路径，searchPattern是指模式字符串，"\*"代表0或N个字符，"?"代表1个字符，isSearchChild"是指是否搜索子目录。 |
| GetDirectories(string directoryPath, string searchPattern, bool isSearchChild) | 获取指定目录及子目录中所有子目录列表，directoryPath是指定目录的绝对路径，searchPattern是指模式字符串，"\*"代表0或N个字符，"?"代表1个字符，isSearchChild"是指是否搜索子目录。 |
| GetDirectoryFileNames(string directoryPath, string searchPattern, bool isSearchChild) | 获取指定目录及子目录中所有文件列表，directoryPath是指定目录的绝对路径，searchPattern是指模式字符串，"\*"代表0或N个字符，"?"代表1个字符，isSearchChild"是指是否搜索子目录。 |
| **目录相关操作的内容** | |
| CreateDirectory(string dir) | 创建目录，dir是指要创建的目录路径包括目录名。 |
| DeleteDirectory(string dir) | 删除目录，dir是指要删除的目录路径和名称。 |
| ClearDirectory(string directoryPath) | 清空指定目录下所有文件及子目录,但该目录依然保存。directoryPath是指定目录的绝对路径。 |
| CopyDirectory(string varFromDirectory, string varToDirectory) | 复制文件夹(递归)。varFromDirectory是指源文件夹路径，varToDirectory是指目标文件夹路径。 |
| **文件本身操作** | |
| IsExistFile(string filePath) | 检测指定文件是否存在,如果存在则返回true。filePath是指文件的绝对路径。 |
| CreateFile(string filePath) | 创建一个文件。 |
| CreateFile(string path, string content) | 创建文件，path是指路径，content是指内容。 |
| CreateFile(string filePath, byte[] buffer) | 创建一个文件,并将字节流写入文件。filePath是指文件的绝对路径，buffer是指二进制流数据。 |
| DeleteFile(string file) | 删除文件，file是指要删除的文件路径和名称 |
| MoveFile(string sourceFilePath, string descDirectoryPath) | 将文件移动到指定目录，sourceFilePath是指需要移动的源文件的绝对路径，descDirectoryPath是指移动到的目录的绝对路径。 |
| CopyFile(string dir1, string dir2) | 复制文件，dir1是指要复制的文件的路径已经全名(包括后缀)，dir2是指目标位置,并指定新的文件名。 |
| **文件内容检测** | |
| GetLineCount(string filePath) | 获取文本文件的行数 |
| GetFileSize(string filePath) | 获取一个文件的长度,单位为Byte |
| ToFileSize(long size) | 计算文件大小函数(保留两位小数),Size为字节大小 |
| **文件内容操作** |  |
| WriteText(string filePath, string text, Encoding encoding) | 向文本文件中写入内容，text为写入的内容，encoding为编码 |
| AppendText(string filePath, string content) | 向文本文件的尾部追加内容 |
| ReadAllFileContent(string filePath) | 获取文件的所有内容 |
| ClearFile(string filePath) | 清空文件内容,文件流不会关闭 |
| ClearFileAndGet(string filePath) | 清空文件内容,文件流不会关闭 |
| **文件路径处理** | |
| GetExtension(string filePath) | 从文件的绝对路径中获取扩展名 |
| GetDirectoryPath(string filePath) | 从文件的绝对路径中获取最末层文件夹路径 |
| GetFileName(string filePath) | 从文件的绝对路径中获取文件名( 包含扩展名 ) |
| GetFileNameNoExtension(string filePath) | 从文件的绝对路径中获取文件名( 不包含扩展名 ) |
| GetFullPath(string relativePath) | 根据相对路径转化为绝对路径 |

## Json的转换

Json类是在Newtonsoft.json的基础上进行封装的。可以实现对json格式数据的双向转化。注意 dynamic类型无法直接转化，可以先封装为object类型在进行转化。方法说明如表 2-15所示：

表 2-15 Json类说明

|  |  |
| --- | --- |
| **方法** | **说明** |
| ToJson() | 对象转为json字符串。 |
| ToJson(this object obj, string datetimeformats) | 对象转为Json字符串,可对日期格式的转化进行控制。Obj是指对象，datetimeformats是指日期的格式：yyyy-MM-dd HH:mm:ss |
| ToObject() | Json字符串转为对象。 |
| ToObject<T>(this string json) | 根据对象的类型，将json字符串转为对象。T是指对象泛型，json是指json字符串。 |
| ToList<T>(this string json) | 将列表类型json转化为列表对象。 |

## 日志的操作

Log输出类，是在log4net的基础上封装的方法，说明如表 2-16所示：

表 2-16 Log输出类说明

|  |  |
| --- | --- |
| **方法** | **说明** |
| Debug(object message) | 输出Debuge类型的日志 |
| Debug(object message, Exception exp) | 输出Debuge类型的日志，message是指输出的文本信息，exp是指输出的异常信息。 |
| Error(object message) | 输出Error类型的日志 |
| Error(object message, Exception exp) | 输出Error类型的日志，message是指输出的文本信息，exp是指错误时输出的异常信息。 |
| Info(object message) | 输出Info类型的日志 |
| Warn(object message) | 输出Warn类型的日志 |

LogFactory类是输出对象的构造工厂，方法说明如表 2-17所示：

表 2-17 LogFactory类说明

|  |  |
| --- | --- |
| **方法** | **说明** |
| GetLogger(Type type) | 获取日志输出对象。 |
| GetLogger(string str) | 获取日志输出对象，特定标识。 |

LogHelper类是Log输出辅助类，方法说明如表 2-18所示：

表 2-18 LogHelper类说明

|  |  |
| --- | --- |
| **方法** | **说明** |
| Error(string msg) | 输出错误信息。 |
| Error(string msg, Exception ex) | 输出错误信息，msg是指自定义文本错误信息，ex是指异常。 |
| Error(Type t, string msg) | 输出错误信息，t是指方法所在类型，msg是指自定义文本错误信息。 |
| Error(Type t, Exception ex) | 输出错误信息，t是指方法所在类型，msg是指自定义文本错误信息。 |
| Error(Type t, string exDesp, Exception ex) | 输出错误信息，t是指方法所在类型，exDesp是指异常描述，msg是指自定义文本错误信息。 |
| Info(string msg) | 输出正常信息 |
| Info(Type t, string msg) |
| Debug(string msg) | 输出调试信息 |
| Debug(string msg, Exception ex) |
| Debug(Type t, string msg) |
| Debug(Type t, string msg, Exception ex) |

## 网络信息

Net类是网络信息检测以及获取的类，对于服务器的Ip获取以及对Ip判断之类的功能，当进行接口请求时，可以开个检测是否网络连接的线程。方法说明如表 2-19所示：

表 2-19 Net类说明

|  |  |
| --- | --- |
| **方法** | **说明** |
| **获取Ip** | |
| GetLocation(string ip) | 获取IP地址所在城市 |
| GetWebRemoteIp() | 获取Web远程Ip |
| GetLanIp() | 获取局域网IP |
| **获取主机名** | |
| GetWebClientHostName() | 获取Web客户端主机名 |
| **获取mac地址** | |
| NetCardInfo() | 返回描述本地计算机上的网络接口的对象(网络接口也称为网络适配器)。 |
| GetMacByNetworkInterface() | 通过NetworkInterface读取网卡Mac |
| GetMacByWMI() | 获得当前机器的活动中Mac地址，若无联网则返回空"" |
| **其他方法** | |
| GetLocation(string ip) | 获取IP地址所在城市 |
| IpExistsInRange(String ip, String beginIp, string endIp) | ip是否在ip空间内 |
| GetIp2Long(String ip) | IP地址转化为long型数据 |
| IsConn(string ip) | 用于检测到指定Ip的网络是否联通 |
| CmdPing(string strIp) | 用于检测到指定Ip的网络是否联通 |

RequestHelp类是网络请求的实体类，用于后台进行网络请求时调用的方法。在进行集群环境内的协调时会比较实用。方法说明如表 2-20所示：

表 2-20 RequestHelp类说明

|  |  |
| --- | --- |
| **方法** | **说明** |
| **POST** | |
| HttpPost(string url, IDictionary<string, string> parameters) | Post方式发送请求，parameters为参数 |
| HttpPost(string url, string param = null) | HTTP POST方式请求数据 |
| **Put** | |
| HttpPut(string url, string param = null) | HTTP Put方式请求数据. |
| **Delete** |  |
| HttpDelete(string url, string param = null) | HTTP Delete方式请求数据. |
| **Get** | |
| HttpGet(string url, Hashtable headht = null) | HTTP GET方式请求数据. |
| HttpGet(string url, Encoding encodeing, Hashtable headht = null) | HTTP GET方式请求数据. |
| **Post With Pic** |  |
| HttpPostWithPic(string url, IDictionary<object, object> param, string filePath) | HTTP POST方式请求数据(带图片) |
| HttpPostWithPic(string url, IDictionary<object, object> param, byte[] fileByte) | HTTP POST方式请求数据(带图片)，param为POST的数据，fileByte为图片。 |
| CheckValidationResult(object sender, X509Certificate certificate, X509Chain chain, SslPolicyErrors errors) | 验证审核接货 |
| **其他** | |
| BuildRequest(string strUrl, Dictionary<string, string> dicPara, string fileName) | 一个比较完整的请求 |

## 加密操作

AES加密为一种对称加密算法，方法说明如表 2-21所示：

表 2-21 AES加密方法

|  |  |
| --- | --- |
| **方法** | **说明** |
| GetLegalIv() | 获得初始向量IV |
| GetLegalKey() | 获得密钥 |
| Encrypt(string encryptStr) | 实现加密字符串 |
| Decrypt(string decryptStr) | 实现解密字符串 |

Base64加密方法说明如表 2-22所示：

表 2-22 Base64加密方法

|  |  |
| --- | --- |
| **方法** | **说明** |
| GetStandardBase64() | 获取具有标准的Base64密码表的加密类 |
| Encrypt(string input) | 使用默认的密码表加密字符串 |
| Decrypt(string input) | 使用默认的密码表解密字符串 |
| Encode(string source) | 返回编码后的字符串 |
| EncodeUnit(params byte[] unit) | 对于单个字符加密 |
| Decode(string source) | 获得解码字符串 |
| DecodeUnit(List<byte> byteArr, params char[] chArray) | 对于单个字符解密 |
| InitDict() | 初始化双向哈希字典 |

DES加密为对称加密，方法说明如表 2-23：

表 2-23 DES加密方法

|  |  |
| --- | --- |
| **方法** | **说明** |
| Encrypt(string text) | 加密 |
| Encrypt(string text, string sKey) | 加密数据 |
| Decrypt(string text) | 解密 |
| Decrypt(string text, string sKey) | 解密数据 |

MD5加密算法方法说明如表 2-24所示：

表 2-24 MD5加密方法

|  |  |
| --- | --- |
| **方法** | **说明** |
| GetMD5\_32(string input) | 获得32位的MD5加密 |
| GetMD5\_16(string input) | 获得16位的MD5加密 |
| GetMD5\_8(string input) | 获得8位的MD5加密 |
| GetMD5\_4(string input) | 获得4位的MD5加密 |
| AddMd5Profix(string input) | 添加MD5的前缀，便于检查有无篡改 |
| RemoveMd5Profix(string input) | 移除MD5的前缀 |
| ValidateValue(string input) | 验证MD5前缀处理的字符串有无被篡改 |
| AddMd5(string path) | 对给定文件路径的文件加上标签 |
| CheckMd5(string path) | 对给定路径的文件进行验证，如果一致返回True，否则返回False |
| Md5Buffer(byte[] md5File, int index, int count) | 计算文件的MD5值，md5File为MD5签名文件字符数组，index为计算起始位置，count为计算终止位置 |

RSA加密为非对称的加密算法，方法说明如表 2-25所示：

表 2-25 RSA加密方法

|  |  |
| --- | --- |
| **方法** | **说明** |
| GenerateRSAKey(out string privateKey, out string publicKey) | 非对称加密生成的私钥和公钥 |
| Encrypt(string publicKey, string originalString) | 非对称加密字符串数据，返回加密后的数据，originalString为待加密的字符串 |
| Encrypt(string publicKey, byte[] originalBytes) | 非对称加密字节数组，返回加密后的数据，originalBytes为待加密的字节数组 |
| Decrypt(string privateKey, string encryptedString) | 非对称解密字符串，返回解密后的数据，encryptedString为待解密数据 |
| Decrypt(string privateKey, byte[] encryptedBytes) | 非对称解密字节数组，返回解密后的数据，encryptedBytes为待解密数据 |
| EncrypSignature(string privateKey, string originalString) | 使用非对称加密签名数据，originalString为待加密的字符串 |
| Validate(string originalString, string encrytedString, string publicKey) | 对私钥加密的字符串，使用公钥对其进行验证，originalString">未加密的文本，如机器码，encrytedString">加密后的文本，如注册序列号，publicKey">非对称加密的公钥 |

RSAEx类为RSA的扩展类，方法说明如表 2-26所示：

表 2-26 RSAEx类说明

|  |  |
| --- | --- |
| **方法** | **说明** |
| GetRASKey() | 得到RSA的解谜的密匙对 |
| CheckSourceValidate(string source) | 检查明文的有效性 DWKEYSIZE/8-11 长度之内为有效 中英文都算一个字符 |
| ComponentKey(byte[] b1, byte[] b2) | 组合成密匙字符串 |
| ResolveKey(string key, out byte[] b1, out byte[] b2) | 解析密匙，b1为RSA的相应参数1，b2为RSA的相应参数2。 |
| EncryptString(string source, string key) | 字符串加密，source为源字符串 明文 |
| DecryptString(string encryptString, string key) | 字符串解密，encryptString为密文 |
| EncryptString(string source, BigInteger d, BigInteger n) | 用指定的密匙加密，d为可以是RSACryptoServiceProvider生成的D，n为以是RSACryptoServiceProvider生成的Modulus |
| DecryptString(string encryptString, BigInteger e, BigInteger n) | 用指定的密匙加密，e为可以是RSACryptoServiceProvider生成的Exponent，n为可以是RSACryptoServiceProvider生成的Modulus |

## 校验

Validate类是校验的工具类，方法说明如表 2-27：

表 2-27 Validate类说明

|  |  |
| --- | --- |
| **方法** | **说明** |
| **用户名密码格式** | |
| GetStringLength(string stringValue) | 返回字符串真实长度, 1个汉字长度为2 |
| IsValidUserName(string userName) | 检测用户名格式是否有效 |
| IsValidPassword(string password) | 密码有效性 |
| **数字字符串检查** | |
| IsValidInt(string val) | int有效性 |
| IsNumber(string inputData) | 是否数字字符串，inputData是指输入字符串。 |
| IsNumberSign(string inputData) | 是否数字字符串 可带正负号，inputData是指输入字符串。 |
| IsDecimal(string inputData) | 是否是浮点数，inputData是指输入字符串 |
| IsDecimalSign(string inputData) | 是否是浮点数 可带正负号，inputData是指输入字符串。 |
| **中文检测** | |
| IsHasCHZN(string inputData) | 检测是否有中文字符 |
| GetCHZNLength(string inputData) | 检测含有中文字符串的实际长度 |
| **常用格式** | |
| IsIdCard(string idCard) | 验证身份证是否合法 15 和 18位两种 |
| IsEmail(string inputData) | 是否是邮件地址。 |
| IsValidZip(string zip) | 邮编有效性 |
| IsValidPhone(string phone) | 固定电话有效性 |
| IsValidMobile(string mobile) | 手机有效性 |
| IsValidPhoneAndMobile(string number) | 电话有效性（固话和手机 ） |
| IsValidURL(string url) | Url有效性 |
| IsValidIP(string ip) | IP有效性 |
| IsValidDomain(string host) | domain 有效性 |
| IsBase64String(string str) | 判断是否为base64字符串 |
| IsGuid(string guid) | 验证字符串是否是GUID |
| **日期检查** | |
| IsDate(string strValue) | 判断输入的字符是否为日期 |
| IsDateHourMinute(string strValue) | 判断输入的字符是否为日期,如2004-07-12 14:25|||1900-01-01 00:00|||9999-12-31 23:59 |
| **其他** | |
| CheckMathLength(string inputData, int maxLength) | 检查字符串最大长度，返回指定长度的串，sqlInput是指输入字符串，maxLength是指最大长度 |
| Encode(string str) | 转换成 HTML code |
| Decode(string str) | 解析html成 普通文本 |

VerifyCode类是验证码生成类，方法说明如表 2-28所示：

表 2-28 VerifyCode类

|  |  |
| --- | --- |
| **方法** | **说明** |
| GetVerifyCode(string code) | 根据指定的编码生成验证码 |
| GetVerifyCode() | 生成验证码图片，将验证码图片的二进制流进行输出 |
| CheckVerifyCode(string code) | 判断当前录入的验证码与生成的验证码是否一致 |

## 前端的操作类

TreeSelect类对combobox展示所使用到的数据格式进行转化，方法说明如表 2-29所示：

表 2-29 TreeSelect类说明

|  |  |
| --- | --- |
| **方法** | **说明** |
| TreeSelectJson(this List<TreeSelectModel> data) | 对combobox展示所使用到的数据格式进行转化，转化后的格式如下： |

TreeSelectModel为查询下拉信息的实体类，包含id、text、value、parentId、data等参数。

TreeGrid类对treegrid展示所使用到的数据格式进行转化，方法说明如表 2-30所示：

表 2-30 TreeGrid类说明

|  |  |
| --- | --- |
| **方法** | **说明** |
| TreeGridJson(this List<TreeGridModel> data) | 对treegrid展示所使用到的数据格式进行转化，转化后的格式如下： |

TreeGridModel为查询树表信息的实体类，包含id、parentId、text、isLeaf、expanded、loaded、entityJson等参数。

TreeView类对tree展示所使用到的数据格式进行转化，方法说明如表 2-31所示:

表 2-31 TreeView类说明

|  |  |
| --- | --- |
| **方法** | **说明** |
| TreeViewJson(this List<TreeViewModel> data, string parentId = "0") | 对tree展示所使用到的数据格式进行转化，转化后的格式如下： |

TreeViewModel为查询树结构的实体类，包含parentId ,id,text,value,checkstate,showcheck,complete,isexpand,hasChildren,iconCls,title等参数。

TreeQuery类，为对树结构的子父级结构进行拼接，如表 2-32所示：

表 2-32 TreeQuery类说明

|  |  |
| --- | --- |
| **方法** | **说明** |
| TreeWhere<T>(this List<T> entityList, Predicate<T> condition, string keyValue = "F\_Id", string parentId = "F\_ParentId") |  |

WebHelper类为对URL、html、session等的一下操作，方法说明如下：

表 2-33 WebHelper类说明

|  |  |
| --- | --- |
| **方法** | **说明** |
| **relativeUrl** | |
| ResolveUrl(string relativeUrl) | 解析相对Url |
| **HtmlEncode(对html字符串进行编码)** | |
| HtmlEncode(string html) | 对html字符串进行编码 |
| HtmlDecode(string html) | 对html字符串进行解码 |
| **UrlEncode(对Url进行编码)** | |
| UrlEncode(string url, bool isUpper = false) | 对Url进行编码，isUpper为编码字符是否转成大写,范例,"http://"转成"http%3A%2F%2F" |
| UrlEncode(string url, Encoding encoding, bool isUpper = false) | 对Url进行编码 |
| GetUpperEncode(string encode) | 获取大写编码字符串 |
| **UrlDecode(对Url进行解码)** | |
| UrlDecode(string url) | 对Url进行解码,对于javascript的encodeURIComponent函数编码参数,应使用utf-8字符编码来解码 |
| UrlDecode(string url, Encoding encoding) | 对Url进行解码,对于javascript的encodeURIComponent函数编码参数,应使用utf-8字符编码来解码。encoding为字符编码,对于javascript的encodeURIComponent函数编码参数,应使用utf-8字符编码来解码 |
| **Session操作** | |
| WriteSession<T>(string key, T value) | 写Session |
| WriteSession(string key, string value) | 写Session |
| GetSession(string key) | 读取Session的值 |
| RemoveSession(string key) | 删除指定Session |
| GetSessionId() | 获取当前会话的唯一标识 |
| SessionEnd() | 关闭当前会话 |
| **Cookie操作** | |
| WriteCookie(string strName, string strValue | 写cookie值 |
| WriteCookie(string strName, string strValue, int expires) | 写cookie值，expires为过期时间(分钟) |
| GetCookie(string strName) | 读cookie值 |
| RemoveCookie(string CookiesName) | 删除Cookie对象 |
| **获取客户端文件控件集合** | |
| GetFileControls() | 获取有效客户端文件控件集合,文件控件必须上传了内容，为空将被忽略 |
| **获取第一个有效客户端文件控件** | |
| GetFileControl() | 获取第一个有效客户端文件控件,文件控件必须上传了内容，为空将被忽略 |
| **请求网络资源** | |
| HttpWebRequest(string url) | 请求网络资源,返回响应的文本 |
| HttpWebRequest(string url, string parameters) | 请求网络资源,返回响应的文本 |
| HttpWebRequest(string url, string parameters, Encoding encoding, bool isPost = false,  string contentType = "application/x-www-form-urlencoded", CookieContainer cookie = null, int timeout = 120000) | 请求网络资源,返回响应的文本 |
| **去除HTML标记** | |
| NoHtml(string Htmlstring) | 去除HTML标记 |
| **格式化文本（防止SQL注入）** | |
| Formatstr(string html) | 格式化文本（防止SQL注入） |

## 其他的操作类

FTPHelper对于FTP进行封装，包括文件的上传、下载、读取等操作，方法说明如表 2-34：

表 2-34 FTPHelper类说明

|  |  |
| --- | --- |
| **方法** | **说明** |
| FTPHelper(string remoteIP, string loginUser, string loginPassword) | 初始化FTPHelper类，remoteIP是指远程FTP服务器IP地址，loginUser是指FTP登录账号，loginPassword是指FTP登录密码 |
| FTPHelper(string connStr) | 根据连接字符串连接ftp服务器 |
| GetConnStrs(string connStr) | 对连接字符串进行拆分 |
| Upload(string localFilePath, string remoteFilePath, string ftpServerIP, string loginName, string loginPassword) | 上传文件到FTP服务器，localFilePath是指上传文件的本地路径和名称，remoteFilePath是指上传文件的FTP路径和名称（相对FTP根目录）,ftpServerIP是指FTP服务器IP地址,loginName是指FTP 服务器登录名,loginPassword是指FTP 服务器登录密码。 |
| Upload(string localFilePath, string remoteFilePath) | 上传文件到FTP服务器，localFilePath是指上传文件的本地路径和名称，remoteFilePath是指上传文件的FTP路径和名称（相对FTP根目录） |
| Download(string localFilePath, string remoteFilePath, string ftpServerIP, string loginName, string loginPassword) | 从FTP服务器上下载文件，localFilePath是指本地路径和文件名，remoteFilePath是指远程服务器文件路径和名称，ftpServerIP是指FTP服务器IP地址，loginName是指FTP 服务器登录名，loginPassword是指FTP 服务器登录密码 |
| Download(string localFilePath, string remoteFilePath) | 从FTP服务器上下载文件,localFilePath是指本地路径和文件名,remoteFilePath是指远程服务器文件路径和名称。 |
| GetFtpFileSize(string fileName, string ftpServerIP, string ftpUserID, string ftpPassword) | 获取FTP服务器上指定文件的大小，fileName是指FTP服务器上的文件名和路径，ftpServerIP是指FTP IP地址，ftpUserID是指FTP 登录名，ftpPassword是指FTP 登录密码。 |
| GetFtpFileSize(string remoteFilePath) | 获取FTP服务器上指定文件的大小，remoteFilePath是指FTP服务器上的文件名和路径 |
| MakeDirectory(string remotePath, string ftpServerIP, string loginName, string loginPassword) | 在FTP服务器上创建文件夹，remotePath是指创建文件夹路径，ftpServerIP是指FTP服务器IP地址，loginName是指FTP 服务器登录名，loginPassword是指FTP 服务器登录密码。 |
| MakeDirectory(string remotePath) | 在FTP服务器上创建文件夹，remotePath是指创建文件夹路径。 |
| MakeDirectoryOfFile(string remoteFilePath, string ftpServerIP, string loginName, string loginPassword) | 根据文件路径创建文件夹，remoteFilePath是指文件及路径名，ftpServerIP是指FTP服务器IP地址，loginName是指FTP 服务器登录名，loginPassword是指FTP 服务器登录密码。 |
| MakeDirectoryOfFile(string remoteFilePath) | 根据文件路径创建文件夹。 |

HexHelper是16进制字符串数字的操作类，一般涉及到一些业务的操作类，好多采用16进制的形式进行操作，这里对传输字符串的转化进行封装。方法说明如表 2-35所示：

表 2-35 HexHelper类说明

|  |  |
| --- | --- |
| **方法** | **说明** |
| FormatHexStrPre(string str) | 格式化字符串，将不完整的16进制字符串 进行前边补0，转化为可转化的16进制，str是指获取的字符串。 |
| FormatHexStrAfter(string str) | 格式化字符串，将不完整的16进制字符串 进行后边补0，转化为可转化的16进制。Str是指获取的字符串 |
| CheckIsHexStr(string str) | 检测字符串是否是16进制的字符串 |
| HexStringToByteArray(string str) | string to byteArray |
| ByteArrayToHexString(byte[] data) | byteArray to string |
| ByteToHexString(byte b) | 将字符转化为16进制的字符串 |
| EncodeHex(string strEncode) | 将字符串内容转化为16进制数据编码，其逆过程是Decode。 |
| EncodeHexE(string strEncode) | 将字符串内容转化为16进制数据编码，其逆过程是Decode。 |
| DecodeHex(string strDecode) | 将16进制数据编码转化为字符串，是Encode的逆过程。 |
| DecodeHexE(string strDecode) | 将16进制数据编码转化为字符串，是Encode的逆过程。 |
| ToHex(string s, string charset, bool fenge) | 从汉字转换到16进制，charset是指编码,如"utf-8","gb2312"，fenge是指是否每字符用逗号分隔。 |
| UnHex(string hex, string charset) | 从16进制转换成汉字 |

Serialize是序列化的类，方法说明如表 2-36所示：

表 2-36 Serialize类说明

|  |  |
| --- | --- |
| **方法** | **说明** |
| GetBytes(object pObj) | 获取对象序列化的二进制版本 |
| GetXmlDoc(object pObj) | 获取对象序列化的XmlDocument版本，pObj是指对象实体。 |
| GetObject<T>(byte[] binData) | 从已序列化数据中(byte[])获取对象实体，T是值要返回的数据类型，binData是指二进制数据。 |
| GetObject<T>(XmlDocument xmlDoc) | 从已序列化数据(XmlDocument)中获取对象实体 |

Utility是工具类，方法说明如表 2-37所示：

表 2-37 Utility类说明

|  |  |
| --- | --- |
| **方法** | **说明** |
| **时间戳转换方法** | |
| GetUnixTime() | 获取时间戳 13位 |
| GetUnixTime(DateTime time) | 获取时间戳 13位 |
| UnixTimeToTime(string longDateTime) | 将时间戳转换为日期类型，并格式化 |
| **Stopwatch计时器** | |
| TimerStart() | 计时器开始 |
| TimerEnd(Stopwatch watch) | 计时器结束 |
| **删除数组中的重复项** | |
| RemoveDup(string[] values) | 删除数组中的重复项 |
| **自动生成编号** | |
| GuId() | 表示全局唯一标识符 (GUID)。 |
| CreateNo() | 自动生成编号 201008251145409865 |
| **随机数生成** | |
| GetRandom(string[] range, int seed, bool isDouble) | 获取两数之间的随机数，range是指随机数极限值存放的数组，seed是指随机数生产的种子，isDouble是指获取的是否是double类型的随机数 |
| GetRandom(int min, int max) | 获取简单的随机数 |
| **删除最后一个字符之后的字符** | |
| DelLastComma(string str) | 删除最后结尾的一个逗号。 |
| DelLastChar(string str, string strchar) | 删除最后结尾的指定字符后的字符。 |
| DelLastLength(string str, int Length) | 删除最后结尾的长度。 |
| **Assembly操作类** | |
| GetAllSubClass(Type baseType) | 获取所有继承对应基类或者接口的类，baseType是指基类类型。 |
| GetInstanceByType(Type type) | 根据类型信息，获取该类型的实例对象 |
| **其他** | |
| GetFileJson(string filepath) | 获取文件的内容，filepath是指文件路径 |
| StartCmd(String command) | 执行Cmd命令，command是指要执行的命令 |
| KillProcessExists(string name) | 杀死进程 |
| GetSubString(string str, string begin, string end) | 获取局部字符串。 |
| GetOSPlatform() | 获取当前系统的位数（32/64） |