On BCL contracts

Francesco Logozzo

The most commons properties I found in the BCL source code and documentation are:

1. Not-null checking
2. Arithmetic, and bounds checking
3. Enums and typestates
4. Strings and characters
5. Dynamic types

### a. Not-null checks

*Checking that an object is or is not null*

Format: “variable != null” or “variable == null”

Example, from System.Globalization.Calendar:

public static Calendar ReadOnly(Calendar calendar)

{

if (calendar == null)

{ throw new ArgumentNullException("calendar"); }

…

}

### b. Arithmetic and bounds checking

*Checking simple arithmetic truths and checking the bounds when accessing arrays, strings, allocating memory, etc.*

1. **Simple numerical bounds checking**

Format: “variable constant” or “-variable constant”,

Equivalently: “variable constant”, etc.

Example, fromSystem.Text.ASCIIEncoding:

public static Encoding GetEncoding(int codepage)

{

if (codepage < 0 || codepage > 65535) {

throw new ArgumentOutOfRangeException(…)

…

}

Example, from System.Threading.Monitor:

public static bool Wait(Object obj, TimeSpan timeout)

{

long tm = (long)timeout.TotalMilliseconds;

if (tm < -1 || tm > (long) Int32.MaxValue)

throw new ArgumentOutOfRangeException("timeout",…);

return Wait(obj,(int)tm,false);

}

1. **Simple numerical bounds checking, on floats**

Format: “variable float\_constant”, or “-variable float\_constant”, or “variable float\_constant”

Example, from System.Collections.Queue:

public Queue(int capacity, float growFactor) {

if (capacity < 0)

throw new ArgumentOutOfRangeException("capacity", …);

if (!(growFactor >= 1.0 && growFactor <= 10.0))

throw new ArgumentOutOfRangeException("growFactor", …);

}

1. **Simple inequalities**

Format: “variable != constant”

Equivalently : “-variable != constant”

Example, from System.Globalization.DateTimeFormatInfo:

private String GetLongDatePattern(int calID) {

String strTemp = String.Empty;

if (!m\_isDefaultCalendar)

{

// Has to be > 0 or the data'll be null

BCLDebug.Assert(calID > 1, …);

strTemp = CalendarTable.Default.SLONGDATE(calID)[0];

BCLDebug.Assert(strTemp.Length != 0, "…");

}

1. **Two variables bounds checking**

Format: “variable – variable constant” or “variable + variable constant” or “– variable – variable constant”.

Equivalently: “variable variable” or “variable variable”, etc.

Example, from System.Array:

public static int FindIndex<T>(T[] array, int startIndex, int count, Predicate<T> match) {

…

if( startIndex < 0 || startIndex > array.Length ) {

throw new ArgumentOutOfRangeException("startIndex",…);

} …

1. **Three or more variables bounds checking**

Format: “variable + variable + … + variable constant”, or “variable + variable + … – variable – variable constant”, or “– variable – variable – variable constant”, etc.

Equivalently: “variable + variable + variable < constant”, “–variable – variable – variable < constant”, etc.

Example, from System.Security.Cryptography.CryptoAPITransform:

public int TransformBlock(byte[] inputBuffer, int inputOffset, int inputCount,…) {

…

if ((inputBuffer.Length - inputCount) < inputOffset) throw new ArgumentException(…)); …

Example, from System.Array:

public static int FindIndex<T>(T[] array, int startIndex, int count, Predicate<T> match) {

…

if (count < 0 || startIndex > array.Length - count) {

throw new ArgumentOutOfRangeException("count",…); }

…

1. **One variable reminder checking**

Format: “variable % constant == constant”, or “variable % constant != constant”, or “variable % constant < constant” or “exp\_one\_variable % constant == constant”, where “exp\_one\_variable” is an arithmetic expression containing exactly one variable

Example, from System.Security.AccessControl.RawACL:

public override GenericAce this[int index]

set

{

// …

if ( value.BinaryLength % 4 != 0 )

{

BCLDebug.Assert( false, "aceLength % 4 != 0" );

throw new SystemException();

} // …

}

…

Example, from System.Random:

public Random(int Seed) {

int ii;

int mj, mk;

// …

mj = MSEED - Math.Abs(Seed);

SeedArray[55]=mj;

mk=1;

for (int i=1; i<55; i++) {

ii = (21\*i)%55;

SeedArray[ii]=mk;

}

…

1. **Two variables reminder checking**

Format: “variable % variable == constant” or “variable % variable < constant”

Example, from System.Security.Cryptography.CryptoAPITransform:

public int TransformBlock(byte[] inputBuffer, int inputOffset, int inputCount, …) {

…

if ((inputCount <= 0) || (inputCount % InputBlockSize != 0) || (inputCount > inputBuffer.Length)) throw new ArgumentException(…));

…

### c. Enums and typestates

1. **Bounds** **checking for enums**

Format: “(int) enum\_variable constant” or “(int) enum\_variable != constant”

Equivalent: (int) enum\_variable < constant”, …

Example, from System.Globalization.Calendar

public virtual int GetWeekOfYear(…, DayOfWeek firstDayOfWeek)

{

if ((int)firstDayOfWeek < 0 || (int)firstDayOfWeek > 6) {

throw new ArgumentOutOfRangeException(…));

} …

1. **Finite** **set inclusion test**

Format: “enum\_variable == constant || … || enum\_variable == constant” or “enum\_variable & flag == 0” or “enum\_variable & flag != 0”

Particular case: “enum\_variable == constant”

Example, from System.Globalization.Calendar

public virtual int GetWeekOfYear(…, CalendarWeekRule rule, …)

{

…

switch (rule) {

case CalendarWeekRule.FirstDay:

return (GetFirstDayWeekOfYear(time, (int)firstDayOfWeek));

case CalendarWeekRule.FirstFullWeek:

return (GetWeekOfYearFullDays(time, rule, (int)firstDayOfWeek, 7));

case CalendarWeekRule.FirstFourDayWeek:

return (GetWeekOfYearFullDays(time, rule, (int)firstDayOfWeek, 4));

}

throw new ArgumentOutOfRangeException(…);

}

Example, from System.Globalization.CompareInfo

public unsafe virtual bool IsSuffix(String source, String suffix, CompareOptions options)

{

…

if ((options & ValidIndexMaskOffFlags) != 0 && (options != CompareOptions.Ordinal)) {

throw new ArgumentException(…);

}

}

1. **Checking the internal state of an object before an operation (typestates)**

Format: not so clear: sometime is an int, others an enum, always accessed via properties?

Example, from System.IO.FileReadStream:

public override int Read([In, Out] byte[] array, int offset, int count) {

…

if (\_handle.IsClosed) \_\_Error.FileNotOpen();

…

### d. Strings and characters

1. **String length checking**

Format: As “Arithmetic and bounds checking” (paragraph b. above)

Example, from System.Globalization.IdnMapping:

public String GetAscii(String unicode, int index, int count)

{

…

if (index > unicode.Length)

throw new ArgumentOutOfRangeException(…);

if (index > unicode.Length - count)

throw new ArgumentOutOfRangeException(…);

…

BCLDebug.Assert(unicode.Length >= 1, …);

1. **String value comparison**

Format: (something like) “String\_Compare(string\_variable, stringConstant) == 0”

Example, from System.Security.Cryptography.Xml.SignedXml

public void ComputeSignature(KeyedHashAlgorithm macAlg) {

…

HMAC hash = macAlg as HMAC;

…

switch (hash.HashName) {

case "SHA1":

SignedInfo.SignatureMethod = SignedXml.XmlDsigHMACSHA1Url;

break;

case "SHA256":

SignedInfo.SignatureMethod = SignedXml.XmlDsigMoreHMACSHA256Url;

break;

case "SHA384":

SignedInfo.SignatureMethod = SignedXml.XmlDsigMoreHMACSHA384Url;

break;

case "SHA512":

SignedInfo.SignatureMethod = SignedXml.XmlDsigMoreHMACSHA512Url;

break;

case "MD5":

SignedInfo.SignatureMethod = SignedXml.XmlDsigMoreHMACMD5Url;

break;

case "RIPEMD160":

SignedInfo.SignatureMethod = SignedXml.XmlDsigMoreHMACRIPEMD160Url;

break;

default:

throw new CryptographicException(…);

}

…

}

Example, from System.Enum

public String ToString(String format) {

if (format == null || format.Length == 0)

format = "G";

if (String.Compare(format, "G", StringComparison.OrdinalIgnoreCase) == 0)

return ToString();

if (String.Compare(format, "D", StringComparison.OrdinalIgnoreCase) == 0)

return this.GetValue().ToString();

if (String.Compare(format, "X", StringComparison.OrdinalIgnoreCase) == 0)

return this.ToHexString();

if (String.Compare(format, "F", StringComparison.OrdinalIgnoreCase) == 0)

return InternalFlagsFormat(this.GetType(), this.GetValue());

throw new FormatException(…);

}

1. **String format, in particular illegal** **characters in a string**

Format: not clear, see examples below

Suggestion: Have a list of invalid/valid characters in a string?

Example, from System.Security.SecurityElement:

public static bool IsValidText( String text )

{

if (text == null)

return false;

return text.IndexOfAny( s\_textIllegalCharacters ) == -1;

}

public SecurityElement( String tag )

{

if (tag == null)

throw new ArgumentNullException( "tag" );

if (!IsValidTag( tag ))

throw new ArgumentException( … );

m\_strTag = tag;

m\_strText = null;

}

Example, from System.Version

public Version(String version) {

…

String[] parsedComponents = version.Split(new char[] {'.'});

int parsedComponentsLength = parsedComponents.Length;

if ((parsedComponentsLength < 2) || (parsedComponentsLength > 4)) throw new ArgumentException(…);

…

1. **Zero-terminated strings**

Format: “string\_variable[length] == ‘\0’”

Example, from System.String

internal unsafe static string SmallCharToUpper(string strIn) {

…

int length = strIn.Length;

String strOut = FastAllocateString(length);

fixed (char \* inBuff = &strIn.m\_firstChar, outBuff = &strOut.m\_firstChar) {

for(int i = 0; i < length; i++) {

int c = inBuff[i];

Contract.Assert(c <= 0x7F, "string has to be ASCII");

// uppercase - notice that we need just one compare

if ((uint)(c - 'a') <= (uint)('z' - 'a')) c -= 0x20;

outBuff[i] = (char)c;

}

Contract.Assert(outBuff[length]=='\0', "outBuff[length]=='\0'");

}

return strOut;

}

### e. Dynamic types

*Checking of the dynamic type of objects*

Format: “variable is Type”

Example, fromSystem.Runtime.InteropServices.Marshal:

public static void Prelink(MethodInfo m)

{

if (m == null)

throw new ArgumentNullException("m");

if (!(m is RuntimeMethodInfo))

throw new ArgumentException(…);

RuntimeMethodHandle method = m.MethodHandle;

InternalPrelink(method.Value);

}

Example, from System.Runtime.Serialization.ObjectManager

internal void CompleteISerializableObject(Object obj,…) {

RuntimeConstructorInfo constInfo = null;

…

if (!(obj is ISerializable)) {

throw new ArgumentException(…);

}