DOWNLOAD PDF STRING CONVENIENCE METHODS

Chapter 1 : Base64 Public Domain Encoder/Decoder

Remarks. A string is a sequential collection of characters that is used to represent text. A String object is a sequential collection of blog.quintoapp.com objects that represent a string; a blog.quintoapp.com object corresponds to a UTF code unit.

If the first operand is a symbol that resolves to a class name, the access is considered to be to a static member of the named class. Otherwise it is presumed to be an instance member and the first argument is evaluated to produce the target object. For the special case of invoking an instance member on a Class instance, the first argument must be an expression that evaluates to the class instance - note that the preferred form at the top expands Classname to identity Classname. If the second operand is a symbol and no args are supplied it is taken to be a field access - the name of the field is the name of the symbol, and the value of the expression is the value of the field, unless there is a no argument public method of the same name, in which case it resolves to a call to the method. If the second operand is a symbol starting with -, the member-symbol will resolve only as field access never as a 0-arity method and should be preferred when that is the intent. If the second operand is a list, or args are supplied, it is taken to be a method call. The first element of the list must be a simple symbol, and the name of the method is the name of the symbol. The args, if any, are evaluated from left to right, and passed to the matching method, which is called, and its value returned. If the method has a void return type, the value of the expression will be nil. Note that placing the method name in a list with any args is optional in the canonic form, but can be useful to gather args in macros built upon the form. Note that boolean return values will be turned into Booleans, chars will become Characters, and numeric primitives will become Numbers unless they are immediately consumed by a method taking a primitive. The member access forms given at the top of this section are preferred for use in all cases other than in macros. Expands into a member access. System getProperties get "os. The args, if any, are evaluated from left to right, and passed to the constructor of the class named by Classname. The constructed object is returned. Class expr Evaluates expr and tests if it is an instance of the class. Returns true or false set! Classname-symbol staticFieldName-symbol expr Assignment special form. When the first operand is a field member access form, the assignment is to the corresponding field. If it is an instance field, the instance expr will be evaluated, then the expr. In all cases the value of expr is returned. Note - you cannot assign to function params or local bindings. Expands into code that creates a fn that expects to be passed an object and any args and calls the named instance method on the object passing the args. Use when you want to treat a Java method as a first-class fn. Since much of the rest of the library is built upon these functions, there is great support for using Java objects in Clojure algorithms. The resulting objects are of an anonymous class. You can also generate statically-named classes and. As of Clojure 1. A single class, if provided, must be first. If not provided it defaults to Object. The interfaces names must be valid interface types. If a method fn is not provided for a class method, the superclass method will be called. If a method fn is not provided for an interface method, an UnsupportedOperationException will be thrown should it be called. Method fns are closures and can capture the environment in which proxy is called. Each method fn takes an additional implicit first arg, which is bound to this. Note that while method fns can be provided to override protected methods, they have no other access to protected members, nor to super, as these capabilities cannot be proxied. Arrays Clojure supports the creation, reading and modification of Java arrays. It is recommended that you limit use of arrays to interop with Java libraries that require them as arguments or use them as return values. Note that many other Clojure functions work with arrays such as via the seq library. The functions listed here exist for initial creation of arrays, or to support mutation or higher performance operations on arrays.

Chapter 2 : Strings (C# Programming Guide) | Microsoft Docs

2 Working with Convenience Methods. This topic describes convenience methods that you can use to register event handlers within your JavaFX application. Learn an easy way to create and register event handlers to respond to mouse events, keyboard events, action events, drag-and-drop events, window events, and others.

Depends on the version of the Unicode Standard supported by the underlying operating system. Strings and embedded null characters In. The value returned by the strlen or wcslen functions does not necessarily equal String. Embedded null characters in a string are also treated differently when a string is sorted or compared and when a string is searched. Null characters are ignored when performing culture-sensitive comparisons between two strings, including comparisons using the invariant culture. They are considered only for ordinal or case-insensitive ordinal comparisons. On the other hand, embedded null characters are always considered when searching a string with methods such as Contains , StartsWith , and IndexOf. Strings and indexes An index is the position of a Char object not a Unicode character in a String. An index is a zero-based, nonnegative number that starts from the first position in the string, which is index position zero. A number of search methods, such as IndexOf and LastIndexOf , return the index of a character or substring in the string instance. The Chars[Int32] property lets you access individual Char objects by their index position in the string. Because the Chars[Int32] property is the default property in Visual Basic or the indexer in C , you can access the individual Char objects in a string by using code such as the following. This code looks for white space or punctuation characters in a string to determine how many words the string contains. Length - 1 If Char. IsPunctuation s1 ctr Or Char. Because the String class implements the IEnumerable interface, you can also iterate through the Char objects in a string by using a foreach construct, as the following example shows. IsPunctuation ch Or Char. Consecutive index values might not correspond to consecutive Unicode characters, because a Unicode character might be encoded as more than one Char object. In particular, a string may contain multi-character units of text that are formed by a base character followed by one or more combining characters or by surrogate pairs. To work with Unicode characters instead of Char objects, use the System. StringInfo and TextElementEnumerator classes. The following example illustrates the difference between code that works with Char objects and code that works with Unicode characters. It compares the number of characters or text elements in each word of a sentence. The string includes two sequences of a base character followed by a combining character. Generic; using namespace System:: GetTextElementEnumerator opening ; while te. IsWhiteSpace ch Or Char. IsPunctuation ch Then chars. Count - 1 Console. You can also retrieve an array that contains the starting index of each text element by calling the StringInfo. For more information about working with units of text rather than individual Char values, see the StringInfo class. Null strings and empty strings A string that has been declared but has not been assigned a value is null. Attempting to call methods on that string throws a NullReferenceException. A null string is different from an empty string, which is a string whose value is "" or String. In some cases, passing either a null string or an empty string as an argument in a method call throws an exception. For example, passing a null string to the Int In other cases, a method argument can be either a null string or an empty string. For example, if you are providing an IFormattable implementation for a class, you want to equate both a null string and an empty string with the general "G" format specifier. The String class includes the following two convenience methods that enable you to test whether a string is null or empty: IsNullOrEmpty , which indicates whether a string is either null or is equal to String. This method eliminates the need to use code such as the following: Empty , or consists exclusively of white-space characters. ToString implementation of a custom Temperature class. The method supports the "G", "C", "F", and "K" format strings. If an empty format string or a format string whose value is null is passed to the method, its value is changed to the "G" format string. CurrentCulture Select Case fmt. Case "G", "C" Return temp. Methods that appear to modify a String object actually return a new String object that contains the modification. Because strings are immutable, string manipulation routines that perform repeated additions or deletions to what appears to be a single string can exact a significant performance penalty. For example, the following code uses a random number generator to create a string with characters in the range 0x to 0xF. Although the code appears to use string concatenation to append a new character to the existing string named str, it actually creates a new String object for each concatenation operation. IO; using namespace System:: Next 1, 0x ; if str. Write str ; sw. Close End Sub End Module You can use the StringBuilder class instead of the String class for operations that make multiple changes to the value of a string. Unlike instances of the String class, StringBuilder objects are mutable; when you concatenate, append, or delete substrings from a string, the operations are performed on a single string. When you have finished modifying the value of a StringBuilder object, you can call its StringBuilder. ToString method to convert it to a string. The following example replaces the String used in the previous example to concatenate random characters in the range to 0x to 0xF with a StringBuilder object. Next 1, 0x ; if sb. An ordinal operation acts on the numeric value of each Char object. A culture-sensitive operation acts on the value of the String object, and takes culture-specific casing, sorting, formatting, and parsing rules into account. Culture-sensitive operations execute in the context of an explicitly declared culture or the implicit current culture. The two kinds of operations can produce very different results when they are performed on the same string. NET also supports culture-insensitive linguistic string operations by using the invariant culture CultureInfo. InvariantCulture , which is loosely based on the culture settings of the English language independent of region. CultureInfo settings, the settings of the invariant culture are guaranteed to remain consistent on a single computer, from system to system, and across versions of. The invariant culture can be seen as a kind of black box that ensures stability of string comparisons and ordering across all cultures. Important If your application makes a security decision about a symbolic identifier such as a file name or named pipe, or about persisted data such as the text-based data in an XML file, the operation should use an ordinal comparison instead of a culture-sensitive comparison. This is because a culture-sensitive comparison can yield different results depending on the culture in effect, whereas an ordinal comparison depends solely on the binary value of the compared characters. Important Most methods that perform string operations include an overload that has a parameter of type StringComparison , which enables you to specify whether the method performs an ordinal or culture-sensitive operation. In general, you should call this overload to make the intent of your method call clear. For best practices and guidance for using ordinal and culture-sensitive operations on strings, see Best Practices for Using Strings. Operations for casing , parsing and formatting , comparison and sorting , and testing for equality can be either ordinal or culture-sensitive. The following sections discuss each category of operation. Tip You should always call a method overload that makes the intent of your method call clear. For example, instead of calling the Compare String, String method to perform a culture-sensitive comparison of two strings by using the conventions of the current culture, you should call the Compare String, String, StringComparison method with a value of StringComparison. CurrentCulture for the comparisonType argument. For more information, see Best Practices for Using Strings. You can download the Sorting Weight Tables , a set of text files that contain information on the character weights used in sorting and comparison operations for Windows operating systems, and the Default Unicode Collation Element Table , the sort weight table for Linux and macOS. Casing Casing rules determine how to change the capitalization of a Unicode character; for example, from lowercase to uppercase. Often, a casing operation is performed before a string comparison. For example, a string might be converted to uppercase so that it can be compared with another uppercase string. You can convert the characters in a string to lowercase by calling the ToLower or ToLowerInvariant method, and you can convert them to uppercase by calling the ToUpper or ToUpperInvariant method. In addition, you can use the TextInfo. ToTitleCase method to convert a string to title case. Casing operations can be based on the rules of the current culture, a specified culture, or the invariant culture. Because case mappings can vary depending on the culture used, the result of casing operations can vary based on culture. The actual differences in casing are of three kinds: The following example demonstrates how a string comparison designed to prevent file system access can fail if it relies on a culture-sensitive casing comparison. The casing conventions of the invariant culture should have been used. WriteLine resource ; Console. Equals disallowed, scheme, StringComparison. For Each culture In cultures Thread. True Differences in case mappings between the invariant culture and all other cultures. In these cases, using the casing rules of the invariant culture to change a character to uppercase or lowercase returns the same character. For all other cultures, it returns a different character. Some of the affected characters are listed in the following table.

Chapter 3 : String Class (System) | Microsoft Docs

I have a List of blog.quintoapp.com there a Java convenience method to convert this List to a CSV String?So "test1, test2, test3" is the result of a conversion of a List 3 String elements which contains "test1" "test2" "test3".

This article is featured in the new DZone Guide to Java: Get your free copy for more insightful articles, industry statistics, and more! Java is no longer in that mode: Java Collections are a core part of the language that we probably touch in one way or another every day, and anything that makes it easier to work with Collections makes our job easier. Convenience Factory Methods for Collections Java 9 introduced new ways to create immutable collections. We may have fallen into the trap of thinking Arrays. UnsupportedOperationException thrown at AbstractList. Instead, what we really wanted was: To create an immutable Set before Java 9, you could use something like: In Java 9, this can be: If the Map has fewer than ten values, we might want to use the convenience Map. Creating Immutable Copies of Collections Java 9 introduced these factory methods to make it easier to create new immutable collections from known values. Java 10 recognizes that this is not the only way we create collections, and introduces more ways of creating immutable collections from existing collections or operations. Prior to Java 10, you could create a new list that was a copy of an existing collection using a copy constructor: UnsupportedOperationException thrown at ImmutableCollections. The Map version takes another Map to copy, not a Collection: This means that from Java 10, you can create immutable Collections not only from some known values or by copying an existing Collection or Map, but also from Stream operations. In Java 10, we can do this: If we change our example to use a Map to calculate the number of times each word is in the sentence, we can demonstrate how to collect into an immutable Map: Download the free trial. Read More From DZone.

Chapter 4 : JavaScript String Methods

String Methods and Properties Primitive values, like "John Doe", cannot have properties or methods (because they are not objects). But with JavaScript, methods and properties are also available to primitive values, because JavaScript treats primitive values as objects when executing methods and properties.

All the techniques demonstrated return the result of the modifications as a new string object. To clearly demonstrate this, the examples all store the result in a new variable. You can then examine both the original string and the string resulting from the modification when you run each example. Note The C examples in this article run in the Try. NET inline code runner and playground. Select the Run button to run an example in an interactive window. Once you execute the code, you can modify it and run the modified code by selecting Run again. The modified code either runs in the interactive window or, if compilation fails, the interactive window displays all C compiler error messages. There are several techniques demonstrated in this article. You can replace existing text. You can search for patterns and replace matching text with other text. You can treat a string as a sequence of characters. You can also use convenience methods that remove white space. You should choose the techniques that most closely match your scenario. Replace text The following code creates a new string by replacing existing text with a substitute. Replace "mountains", "peaks" ; Console. You can see in the preceding example that the original string, source, is not modified. Replace method creates a new string containing the modifications. The Replace method can replace either strings or single characters. In both cases, every occurrence of the sought text is replaced. WriteLine source ; Console. WriteLine replacement ; The source string is unchanged, and a new string is returned with the replacement. Trim white space You can use the String. TrimStart , and String. TrimEnd methods to remove any leading or trailing white space. The following code shows an example of each. The source string does not change; these methods return a new string with the modified contents. This method removes a number of characters starting at a specific index. The following example shows how to use String. IndexOf followed by Remove to remove text from a string: WriteLine result ; Replace matching patterns You can use regular expressions to replace text matching patterns with new text, possibly defined by a pattern. The following example uses the System. Regex class to find a pattern in a source string and replace it with proper capitalization. Replace String, String, MatchEvaluator, RegexOptions method takes a function that provides the logic of the replacement as one of its arguments. In this example, that function, LocalReplaceMatchCase is a local function declared inside the sample method. LocalReplaceMatchCase uses the System. StringBuilder class to build the replacement string with proper capitalization. Regular expressions are most useful for searching and replacing text that follows a pattern, rather than known text. For more information on regular expression language elements, see Regular Expression Language - Quick Reference. Replace for more flexibility. ToUpper replacementBuilder[0] ; return replacementBuilder. ToString method returns an immutable string with the contents in the StringBuilder object. Modifying individual characters You can produce a character array from a string, modify the contents of the array, and then create a new string from the modified contents of the array. The following example shows how to replace a set of characters in a string. First, it uses the ToCharArray method to create an array of characters. It uses the IndexOf method to find the starting index of the word "fox. Finally, a new string is constructed from the updated character array. IndexOf "fox" ; if animalIndex! WriteLine updatedPhrase ; Unsafe modifications to string Using unsafe code, you can modify a string "in place" after it has been created. Unsafe code bypasses many of the features of. NET designed to minimize certain types of bugs in code. You need to use unsafe code to modify a string in place because the string class is designed as an immutable type. Once it has been created, its value does not change. Unsafe code circumvents this property by accessing and modifying the memory used by a string without using normal string methods. The following example is provided for those rare situations where you want to modify a string in-place using unsafe code. The example shows how to use the fixed keyword. The fixed keyword prevents the garbage collector GC from moving the string object in memory while code accesses the memory using the unsafe pointer. It also demonstrates one possible side effect of unsafe operations on strings that results from the way that the C compiler stores interns strings internally. You can learn more in the articles on unsafe and fixed. The API reference for Intern includes information on string interning. WriteLine helloOne ; Console. Or you can download the samples as a zip file.

Chapter 5 : CmdLineParser (JArgs command line option parsing library)

Join(String, IEnumerable) is a convenience method that lets you concatenate each element in an IEnumerable(Of String) collection without first converting the elements to a string array. It is particularly useful with Language-Integrated Query (LINQ) query expressions.

Arrays in general is a very useful and important data structure that can help solve many types of problems. Here are some examples on how to use String Array in Java. String Array Declaration Square brackets is used to declare a String array. There are two ways of using it. The first one is to put square brackets after the String reserved word. Note that this is just a declaration, the variable "thisIsAStringArray" will have the value null. And since there is only one square brackets, this means that the variable is only a one-dimensional String Array. Examples will be shown later on how to declare multi-dimensional String Arrays. Hence, arrays are usually declared with an initial size. Here is an example: Each element can be accessed using index that starts with 0. Here is an example of declaring a String Array with size 5 and assigning values to each element: Or we could use the formula array length - 1. This means if we access the index greater than 4, an Exception will be raised. Initialization can also be done at the same time as the declaration. Hence, when we run this code: This is useful when values are not known during declaration. Note that initializing an array will override the old contents of the array. Asparagus Carrot Tomato Also note that even the size of array will be changed if re-initialized. Asparagus Carrot Exception in thread "main" java. Here is an example usage: Iterate Through String Array Since the size and contents of a String Array can vary, it is useful to iterate through all the values. Here is an example code using style that can run prior to Java 5. This code will run on any version of Java, before of after Java 5 The code will have the output: Apple Banana Orange Another way is to use the enhanced for loop of Java 5. Test if String Array Contains a Value When we have an array of Strings, it is usually common to have a scenario where we wish to know if the array contains a specific value. One way to do it is to have custom code: The output of the code will be: The array contains the string: Banana We can place a break when we found the item, to make it run faster. We can also factor this out as a separate method, if this will be a commonly used routine: Here is the output of the code: The array contains the string Apple: Here is an example code: Another array was created with size larger than 1. The 3 elements were then copied and the new value is added on the last index. The code will print the following result: Apple Banana Orange Carrots We can refactor the logic to a separate function. Sort String Array Another common situation in Java is the need to sort elements of an array. One way to do this is to code our own routine based on some well known algorithm. We can use the sort method in java. Here is a shortened example of our code above using Arrays. The utility class java. Arrays has a convenience method for this. Here is an example code for the conversion: To implement a custom behaviour, we may implement our own code for that: We may implement our own that uses a custom delimiter without the square brackets. Here is an example of converting String Array to String with custom delimiter: Apple-Banana-Orange We can refactor the logic to a separate method. Here is an example of converting a String Array to String using comma delimiter or any other delimiter: If we need the size of the array to grow, it is better to use List. Fortunately, this is simple - again by using java. Returns a fixed-size list backed by the specified array. Changes to the returned list "write through" to the array. This method acts as bridge between array-based and collection-based APIs, in combination with Collection. The returned list is serializable and implements RandomAccess. This means that you can not add items to the List returned by Arrays. Hence, this code will raise an Exception: UnsupportedOperationException on the line where "WaterMelon" was being added to the list. To avoid this problem, we can specifically convert the String Array to an ArrayList. Here is a sample code on how to do that: All items from fixedList will be added to the stringList. And the ArrayList instance has no limitation of being fixed size. Hence, the code that adds "WaterMelon" will not throw any exception. So if we only want elements of a collection without duplicates, Set is a more appropriate data structure. Here is a sample code on how to convert a String Array to a Set: Size of the list is: Convert list to String Array If we want the opposite behavior of the sample above, it is also possible to convert a List back to String Array. Here is a sample code that converts a List to String Array: This is to tell the list what type of array it should return. Two Dimensional String Array in Java In Java, it is possible to declare an array of arrays - or simply a two-dimensional arrays. This is very useful for storing more complex information. For example, a collection of couples husband and wife. A couple is composed of two information, the name of the husband, and the name of the wife. If we want to represent an array of couples, then two-dimensional String Array can be used. We initialize each element with a String Array that represents a pair of husband and wife names. To access a specific item, we need two index values. For example, to output "Robert", the code should be: The effect is the same as the first example. Which means accessing elements is also the same. Here is an even shorter example where we declare and initialize a two-dimensional string array in a single statement:

Chapter 6 : How to C# String Null

Strings (C# Programming Guide) 07/20/; 11 minutes to read Contributors. all; In this article. A string is an object of type String whose value is text. Internally, the text is stored as a sequential read-only collection of Char objects.

Number Localization Algorithm After digits are obtained for the integer part, fractional part, and exponent as appropriate for the data type , the following transformation is applied: If a decimal separator is present, a locale-specific decimal separator is substituted. If the value is NaN or positive infinity the literal strings "NaN" or "Infinity" respectively, will be output. These values are not localized. The following conversions may be applied to byte, Byte , short, Short , int and Integer , long, and Long. The localization algorithm is applied. No localization is applied. If x is negative then the result will be an unsigned value generated by adding 2n to the value where n is the number of bits in the type as returned by the static SIZE field in the Byte , Short , Integer , or Long classes as appropriate. The following flags apply to numeric integral conversions: If this flag is not given then only negative values will include a sign. If the width is not provided, then a MissingFormatWidthException will be thrown. If no flags are given the default formatting is as follows: This includes any signs, digits, grouping separators, radix indicator, and parentheses. The padding is on the left by default. If width is not specified then there is no minimum. The precision is not applicable. If precision is specified then an IllegalFormatPrecisionException will be thrown. The following conversions may be applied to BigInteger. Signed output is allowed for this type because unlike the primitive types it is not possible to create an unsigned equivalent without assuming an explicit data-type size. All flags defined for Byte, Short, Integer, and Long apply. The default behavior when no flags are given is the same as for Byte, Short, Integer, and Long. The specification of width is the same as defined for Byte, Short, Integer, and Long. The following conversions may be applied to float, Float , double and Double. The formatting of the magnitude m depends upon its value. Otherwise, the result is a string that represents the sign and magnitude absolute value of the argument. The formatting of the sign is described in the localization algorithm. The number of digits in the result for the fractional part of m or a is equal to the precision. If the precision is not specified then the default value is 6. If the precision is less than the number of digits which would appear after the decimal point in the string returned by Float. Otherwise, zeros may be appended to reach the precision. For a canonical representation of the value, use Float. After rounding for the precision, the formatting of the resulting magnitude m depends on its value. If m is greater than or equal to but less than 10precision then it is represented in decimal format. If m is less than or greater than or equal to 10precision, then it is represented in computerized scientific notation. The total number of significant digits in m is equal to the precision. If the precision is not specified, then the default value is 6. If the precision is 0, then it is taken to be 1. The result is a string that represents the sign and magnitude absolute value of the argument. The magnitude is formatted as the integer part of m, with no leading zeroes, followed by the decimal separator followed by one or more decimal digits representing the fractional part of m. The result is a string that represents the sign and magnitude absolute value of the argument x. If the value is NaN or infinite, the literal strings "NaN" or "Infinity", respectively, will be output. If m is zero then it is represented by the string "0x0. If m is a double value with a normalized representation then substrings are used to represent the significand and exponent fields. The significand is represented by the characters "0x1. Note that there must be at least one nonzero digit in a subnormal significand. This includes any signs, digits, grouping separators, decimal separators, exponential symbol, radix indicator, parentheses, and strings representing infinity and NaN as applicable. If the precision is not specified, then it is assumed to be 6. If the precision is not provided, then all of the digits as returned by Double. The following conversions may be applied BigDecimal. If the precision is less than the number of digits to the right of the decimal point then the value will be rounded using the round half up algorithm. For a canonical representation of the value, use BigDecimal. The default behavior when no flags are given is the same as for Float and Double. The specification of width and precision is the same as defined for Float and Double. This conversion may be applied to long, Long , Calendar , and Date. Additional conversion types are provided to access Java-specific functionality e. The following conversion characters are used for formatting times: The precision of this value is limited by the resolution of the underlying operating system or hardware. This value will be adjusted as necessary for Daylight Saving Time. For long, Long , and Date the time zone used is the default time zone for this instance of the Java virtual machine. The following conversion characters are used for formatting dates:

Chapter 7 : Formatter (Java Platform SE 7)

There are several techniques demonstrated in this article. You can replace existing text. You can search for patterns and replace matching text with other text. You can treat a string as a sequence of characters. You can also use convenience methods that remove white space. You should choose the.

Therefore, if you view a verbatim string in the debugger watch window, you will see the escape characters that were added by the compiler, not the verbatim version from your source code. For example, the verbatim string "C: Format Strings A format string is a string whose contents can be determined dynamically at runtime. You create a format string by using the static Format method and embedding placeholders in braces that will be replaced by other values at runtime. The following example uses a format string to output the result of each iteration of a loop: Therefore, you can just embed a format string literal without an explicit call to the method. However, if you use the WriteLine method to display debug output in the Visual Studio Output window, you have to explicitly call the Format method because WriteLine only accepts a string, not a format string. For more information about format strings, see Formatting Types. Substrings A substring is any sequence of characters that is contained in a string. Use the Substring method to create a new string from a part of the original string. You can search for one or more occurrences of a substring by using the IndexOf method. Use the Replace method to replace all occurrences of a specified substring with a new string. Like the Substring method, Replace actually returns a new string and does not modify the original string. For more information, see How to: In the following example, assume that you must modify the original string in a particular way and then store the results for future use: ToUpper sb[j] ; else if System. How does Microsoft Word deal with the Caps Lock key? String object that contains zero characters. Empty strings are used often in various programming scenarios to represent a blank text field. You can call methods on empty strings because they are valid System. Empty strings are initialized as follows: Empty; By contrast, a null string does not refer to an instance of a System. String object and any attempt to call a method on a null string causes a NullReferenceException. However, you can use null strings in concatenation and comparison operations with other strings. The following examples illustrate some cases in which a reference to a null string does and does not cause an exception to be thrown: NET are highly optimized and in most cases do not significantly impact performance. However, in some scenarios such as tight loops that are executing many hundreds or thousands of times, string operations can affect performance. The StringBuilder class creates a string buffer that offers better performance if your program performs many string manipulations. The StringBuilder string also enables you to reassign individual characters, something the built-in string data type does not support. This code, for example, changes the content of a string without creating a new string: To avoid visual clutter, these methods are excluded from IntelliSense for the String type, but they are available nevertheless. You can also use LINQ query expressions on strings.

Chapter 8 : Handling JavaFX Events: Working with Convenience Methods | JavaFX 2 Tutorials and Docum

Convenience method that encodes a string into bytes in this charset. An invocation of this method upon a charset cs returns the same result as the expression blog.quintoapp.com(blog.quintoapp.com(s));.

Chapter 9 : Clojure - Java Interop

In many cases, convenience methods such as blog.quintoapp.coms, blog.quintoapp.comeAll and blog.quintoapp.com will be preferable, but if you need to do a lot of work with the same regular expression, it may be more efficient to compile it once and reuse it. The Pattern class and its companion, Matcher, also offer more functionality than the small amount.