Convert int to string java android

I'm not robot

reCAPTCHA

Continue

I'm not robot

reCAPTCHA

In this guide, we'll learn how to convert int into a string in Java. We can convert int into a string using String.valueOf() or Integer.toString(We can also use string.format to convert. 1. Converting int into a line using String.valueOf () String.valueOf (int i) method takes more importance as an argument and returns a line representing the int argument. Signature method: public static value of the lineOf (int i) parameters: i - integer, which must be converted into a returns: a line representing the integrative argument of Java - int to String using String.valueOf() public class JavaExample - public static emptiness of the core (String args) - Int ivar No 111; Line page - String.valueOf (ivar); System.out.println (Стрyuna: 3стр); Output: 555111, because str is a string /and will concatenate 555 and str System.out.println (555'str); Exit: 2. Converting int into a row using Integer.toString () Integer.toString (int i) works the same way as String.valueOf (int i). It belongs to the Integer class and converts the specified integrator value into String. for example, if the passable value is 101, the value of the returned line will be 101. Method signature: String toString (int i): i - integer, which requires conversion return: Line representing integer i. Example: int ivar2 and 200; Line str2 - Integer.toString (ivar2); Java - int to String using Integer.toString () public class Example - public static emptiness core (String args) - int ivar No. 111; String page and Integer.toString System.out.println (Стрyuna: 3стр); Output: 666, because the ivar is an int value and/ will perform the addition of 555 and ivar System.out.println (555'ivar); Exit: Line: 111 555111 666 Example: Conversion Int in String This program demonstrates the use of both the aforementioned methods (String.valueOf)and Integer.toString () Here we have two integrative variables, and we transform one with String.valueOf (int i) and the other using integer.toString (int i). Public Class IntToString - Public Static Void Core (String) - Method 1: Use of valueOf () String. B/int ivar No. 111; Line page - String.valueOf (ivar); System.out.println (Стрyuна: 3стр); Method 2: use of toString method - Integer class - Int ivar2 and 200; Line str2 - Integer.toString (ivar2); System.out.println (String2: Str2); Out: Line: 111 String2: 200 3. String.format () method of converting the public class JavaExample' public static static Main (String args) and edvum No. 99; String page No String.format (%d, num); System.out.println (hello str); Out: hello99 public final class Integer expands number implements gIt;Integergt; comparable java.lang.Number ↳ java.lang.Integer Integer Class Integer wraps the value of primitive type in the object. The Integer-type object contains one field, the type of which is int. In addition, this class provides several methods for converting int into string and string int, as well as other constants and methods useful when working with int. Note to implementation: Implementations of bit tweeddling techniques (such as highOneBit and numberOfTrailing's zeros) are based on henry S. Warren Jr.'s The Pleasure of Hackers (Addison Wesley, 2002). Int BYTES The number of bytes used to represent int value in two binary add-ons. int MAX_VALUE Standing Holding the maximum value Int can have, 231-1. int MIN_VALUE constantly keep the minimum value int can have, -231. INT SI THE Number of bits used to represent int value in two binary add-ons. Public static class TYPE Class, representing a primitive type of int. Integer (int value) builds a recently selected Integer that represents the int. Integer (String s) builds a recently selected Integer that represents the int specified by the line setting. static int bitCount (int i) returns the number of one bit in two int binary representations. byte byteValue () Returns the value of this Integer as a tote after the string object representing the specified integer. static int compare (int x, int y) compares two int values numerically. int compareTo (Integer anotherInteger) compares two Integer objects numerically. static int compareUnsigned (int x, int y) compares two int values numerically considering values as unsigned. static Integer decode (String nm) deciphers the line into the integrator. static int divideUnsigned (int dividend, int divisor) returns the unsigned division ratio of the first argument to the second, where each argument and result is interpreted as an unsigned value. double doubleValue () returns the value of this Integer as a double after the extension of the primitive conversion. boolean equals (Object obj) compares this object to the specified object. floatvalue () Returns the value of this Integer as a float after the expansion of the primitive conversion. static Integer getInteger (String nm, Integer val) determines the integrative value of the system's property with the specified name. static Integer getInteger (String nm, int val) determines the value of the property of the system with the specified name. int hash code () returns to the integer. code for this Integer. static int hashCode (int value) returns hash code for int-value; compatible with Integer.hashCode(). static int highOneBit (int i) returns the internal value with no more than one bit, in the highest order (left) single-bit position in the specified int value. int intValue () Returns the value of this Integer as int. Long long value () Returns the value of this Integer as long after extending the primitive conversion. static int lowestOneBit (int i) returns the internal value with no more than one bit, in the lowest order (right) single-bit position in the specified Value Int value. static int max (int a, int b) returns more than two int values, as if calling Math'max (int, int). static int min (int a, int b) returns the smaller of the two int values, as if by calling Math'min (int, int). Static int numberOfLeading'eros (int i) returns the number of zero bits prior to the highest order (left) one-bit in two add-ons binary representation of the specified int value. static int numberOfTrailing'eros (int i) returns the number of zero bits following the lowest order (right) by one bit in two int binary representation add-ons. static int parseInt (String s) analyzes the line argument as a signed decimal integer. static int parseUnsignedInt (String s) analyzes the line argument as an unsigned decimals integer. static int parseUnsignedInt (String s, int radix) analyzes the line argument as an unsigned integer in the radix specified by the second argument. static int parseInt (String s, int radix) fends off the line argument as a signed integrator in the radix specified by the second argument. static int remainderUnsigned (int dividend, int divisor) returns unsigned remainder from the division of the first argument to the second, where each argument and result is interpreted as an unsigned value. static int reverse (int i) returns the value received by reversing the order of bits in the two binary representations of the specified int value. static int reverseBytes (int i) returns a value received by reversing the bytes order in two int view additions. static int rotateLeft (int i, int distance) returns the value gained by rotating two complementary binary representations of the specified int value left by a specified number of bits. static int rotateRight (int i, int distance) returns the value received by rotating the binary representation of the two int add-ons directly to the specified number of bits. ShortValue () Returns the value of this Integer as a brief after narrowing of the primitive conversion. static int signum (int i) returns signum function of specified value static int sum (int a, int b) Adds two integers together according to the operator. static int toBinaryString (int i) line the presentation of the argument integer as an unsigned integer in ground 2. The static toHexString line (int i) returns the integer argument line as an unsigned integrator in base 16. the static toOctalString line (int i) returns the integer argument view line as an unsigned integrator in Base 8. The toString line returns the string object that represents the value of this Integer. the static toString line (int i) returns the string object representing the specified integer. static String toString (int i, int radix) returns the representation of the first argument line in the radix specified by the second argument. static Value of IntegerOf (String s) returns the Integer object contains the specified value of the specified line. The static value of IntegerOf (String s, int radix) returns the Integer object holding the value extracted from the specified line when parsing with the radix specified by the second argument. The static value of IntegerOf (int i) returns the Integer instance representing the specified int value. From the class java.lang.Number byte byteValue returns the value of the specified number as a private, which may include rounding or truncation. abstract doubleValue returns the value of the specified number as double, which may include rounding. abstract int IntValue () Returns the value of the specified number as an int, which may include rounding or truncation. abstract longValue () Returns the value of the specified number as long, which may include rounding or rooting. shortValue () Returns the value of the specified number as short, which may include rounding or rooting. From the java.lang.Object Object class, the clone creates and returns a copy of this object. boolean (Object obj) indicates whether any other object is equal to this. invalid completion () is called by the garbage collector at the facility when there being converted into a float type. the public static Integer getInteger (nm String, Integer val) returns a newly selected Integer object that represents the int being setting. The first argument is seen as the name of the system property. The system's properties are available using System.getProperty (java.lang.String). The value of the string of this property is then interpreted as a more integrative value of the system's property with a given name. The first argument is zero or is a line of length zero. Radix is either less than Character.MIN_RADIX or more than Character.MAX_RADIX. Any line symbol is not a number of the specified radix, except that the first symbol may be the ASCII mark minus '-' ('u002D') to indicate a negative value or a '002B Sign to indicate a positive value. The resulting integer value is returned. An exception to the NumberFormatException type is cast if any of the following situations occur: the first argument is zero or is a line of length zero. Radix is either less than Character.MIN_RADIX or more than Character.MAX_RADIX. Any line symbol is not a number of the specified radix, except that the first symbol may be the ASCII mark minus '-' ('u002D') to indicate a negative value or a '002B Sign to indicate a positive value. public byte byteValue () returns the value of this Integer as a tote after the narrowing of the primitive conversion. The byte is returned the numerical value presented by this object after being converted to byte type. public static int compare (int x, int y) compares the two int values numerically. The returned value is identical to what will be returned: Integer.valueOf (x).compareTo (Integer.valueOf (y)) Options x int: the first int for comparison y int: the second int for comparison Returns int value 0 if x q y; value of less than 0 if x; zlt; y; and a value If x y public int compareTo (Integer anotherInteger) compares two Integer objects numerically. Other Integer Options: Integer to be compared. Returns int to 0 if this Integer is equal to the integer's zlt;/integer.html; Integrator; Value less than 0 if this integer is numerically smaller than the Integer argument; and a value greater than 0 if this integer is numerically larger than the Integer argument (signed comparison). public static int compareUnsigned (int x, int y) compares two int values numerically considering values as unsigned. Options x int: first int to compare y int: second int to compare Returns int value 0 if x q y; value greater than 0 if x zgt; y as unsigned values of double integer (String nm, int) determines the more integrative value of the property of the system with the specified name. The first argument is seen as the name of the system property. The system's properties are available using System.getProperty (java.lang.String). The value of the string of this property is then interpreted as a more integrative value of the system's property with a given name. To sum up: If the property value starts with ASCII 0x characters or an ASCII character that is not followed by a minus sign, the rest of it is analyzed as a hexagonal integrator in the same way as the value of the Asg (java.lang.String, int) method with radix 16. If the property value starts with the ASCII 0 symbol, followed by another character, it is analyzed as an octal integrator in the same way as the value of the Asg(java.lang.String, int) method with radix 8. Otherwise, the property value is analyzed as a decimal integrator in the same way as the value of the MethodOf (java.lang.String, int) with radix 10. The second argument is the default. The default value is returned if there is no property of the specified name, if the property does not have the correct numerical format, or if the given name is empty or invalid. Nm String options: property name. val Integer: default. Integer Integra returns the value of the property. SecurityException throws for the same reasons as System'getProperty (String) public static Integer getInteger (String nm, int val) determines the more integrative value of the property of the system with the specified name. The first argument is seen as the name of the system property. the system's properties are available using System.getProperty (java.lang.String). The value of the string of this property is then interpreted as an integrative value using the grammar supported by Integer-decode, and the Integer object representing that value is returned. The second argument is the default. The default value is returned if there is no property of the specified name, if the property does not have the correct numerical format, or if the given name is empty or invalid. In other words, this method returns an Integer object equal to the value of getInteger (nm, new Integer (val), but in practice it can be implemented in the same way as: Integer result - getInteger (nm, null); return (result - zero) ? New Integer (val) : result; to avoid unnecessary distribution of the Integer object when the default is not required. Nm String options: property name. val int: value by Integer Integra returns the value of the property. Throws SecurityException for the same reasons as System.getProperty (String) public int hashCode () returns hash for this object. Returns the hash code value for this object, equal to the primitive int presented by this Integer. public int hashCode (int value) returns hash code for int-value; compatible with Integer.hashCode(). Int: Returns int hash value for int. public static int highOneBit (int i) returns the internal value with no more than one bit, in the highest order position one bit in the specified value int. Returns zero if the specified value does not have one bit in the two binary view add-ons, that is, if it is zero. I int: The value that the highest bit of which must be calculated returns int with a single bit, in the highest order position one bit in the specified value, or zero if the specified value itself is zero. Public int IntValue () Returns the value of this Integer as int. Returns int to the numerical value presented by this object after being converted to int type. public longValue () Returns the value of this Integer as long after the expansion of the primitive conversion. Returns the long numerical value presented by this object after the conversion to enter the long. See also: the public static int lowestOneBit (int i) returns the internal value with no more than one bit, in the lowest order (right) position of the one-bit in the specified int value. Returns zero if the specified value does not have one bit in the two binary view add-ons, that is, if it is zero. I int Options: The value, the lowest bit of which should be calculated Returns int value with one bit, in a low order position one bit in the specified value, or zero if the specified value is in itself zero. public static int max (int a, int b) returns more than two int values, as if calling Math'max (int, int). Int options: the first operand in: the second operand Returns int more than the greater of a and b public static int min (int a, int b) returns the smaller of the two int values, as if calling Math'min (int, int). Int Options: The first operand b int: the second operand Returns int smaller of a and b public static int numberOfLeading'eros (int i) returns the number of zero bits preceding the highest order (left) single-bit in the two binary representation of the specified int value. Returns 32 if the specified value does not have one bit in the view of the addition of two, in other words, if it is zero. Note that this method is closely related to the base of logarithm 2. For all positive values int x: gender (log2(x)) No 31 - numberOfLeading'eros (x) ceil (log2 (x) - numberOfLeading'eros (x - 1) Options i int: value, the number of leading zeros which must be calculated Returns int number of zero bits preceding the highest order (left) (left) two add-ons a binary representation of the specified int value, or 32 if the value is zero. public static int numberOfTrailing'eros (int i) returns the number of zero bits following the lowest order (right) one bit in two binary versions of the specified Int value. Returns 32 if the specified value does not have one bit in the view of the addition of two, in other words, if it is zero. i int: A value whose number of lagging zeros must be calculated Returns int the number of zero bits after the lowest order (rightmost) one-bit in two add-ons is a binary representation of the specified Int value, or 32 if the value is zero. the public static int parseInt (String s, int radix) analyzes the line argument as a signed integer in the radix, specified by the second argument. The symbols in the line must be the numbers of the specified radix (defined whether Character.digit (char, int) returns a non-negative value), except that the first symbol may be the ASCII mark minus '-' ('u002D') to indicate a negative value. The resulting integer value is returned. An exception to the NumberFormatException type is cast if any of the following situations occur: the first argument is zero or is a line of length zero. Radix is either less than Character.MIN_RADIX or more than Character.MAX_RADIX. Any line symbol is not a number of the specified radix, except that the first symbol may be the 'plus' ('u002B) sign, provided that the line is longer than the length of 1. The value presented by the line is not an int. Examples: parseInt (0, 10) returns 0 parseInt (473, 10) returns 473 parseInt (Na42, 10) returns 42 parseInt (-0, 10) returns 0 parseInt (-FF, 16) returns -255 parseInt (1100110, 2) Returns 102 parseInt (2147483647, 110) returns 2147483647 parseInt (-2147483648, 10) returns -2147483648 parseInt (2 147483648, 10) throws NumberFormatException parseInt (99, 8) throws NumberFormatException parseInt (Kona, 10) throws NumberFormatException parseInt (Kona, 27) returns 411787 Options s Row : A line containing an integrative view to be disassembled radix Int: radix used in parsing s. Returns int integer presented by the cord argument in the given radix. NumberFormatException throws if the line does not contain a disassemblable int. the public static int parseInt (String s) analyzes the line argument as a signed decimal integer. Symbols in a row must be decimal numbers, except that the first symbol may be the ASCII line minus '-' ('u002D') to indicate a negative value or ASCII plus the 'q' ('u002B' sign) to indicate a positive The resulting integer value is returned, just as if and Radix 10 were given as arguments for the parseInt method (java.lang.String, int). Options s Line: A line containing an int view that will be disassembled Returns int integer value presented by the argument in the decimal. Public static int parseUnsignedInt (String s) analyzes the line argument as an unsigned decimal integer. The symbols in the line should be decimal numbers, except that the first symbol may be ASCII plus the 'q' ('u002B'). The resulting integer value returns, just as if argument and radix 10 were given as arguments for the parseUnsignedInt method (java.lang.String, int). Options s Line: A line containing an unsigned int view that will be disassembled Returns s int integer value presented by the argument in the decimal. public static int parseUnsignedInt (String s, int radix) analyzes the line argument as an unsigned integer in the radix, specified by the second argument. Unsigned cord values are usually associated with negative numbers on positive numbers more than MAX_VALUE. The symbols in the line should be the numbers of the specified radix (whether Character.digit (char, int) returns a non-non-non-social value), except that the first symbol may be the ASCII plus the 'q' ('u002B). The resulting integer value is returned. An exception to the NumberFormatException type is cast if any of the following situations occur: the first argument is zero or is a line of length zero. Radix is either less than Character.MIN_RADIX or more than Character.MAX_RADIX. Any line symbol is not a number of the specified radix, except that the first symbol may be the 'plus' ('u002B') sign, provided that the line is longer than the length of 1. The value represented by the line is more than the largest unsigned int, 232-1. s String Options: A line containing an unsigned integrative view to be disassembled by an radix: a radix that will be used when parsing s. Returns int integer presented by the cord argument in a given radix. NumberFormatException throws if the line does not contain a disassemblable int. public static int remainderUnsigned (int dividend, int divisor) returns unsigned balances from the division of the first argument to the second, where each argument and result is interpreted as unsigned values. Returns int integer presented by the cord argument in a given radix. In dividend options: value to be divided divisor int: value, making division Returns int unsigned remainder of the division of the first argument to the second. public static int reverse (int i) returns the value received by reversing the order of bits in the two additions of int. I int: The value whose bytes should be undone Returns int value received by turning bytes in the specified int value. public static int rotateLeft (int i, int distance) returns the value received by rotating the binary representation of the two int add-ons left by a specified number of bits. (The bets shifts from the left hand, or high-order, lateral re-indusion on the right, or low order.) Note that the left rotation with a negative distance is equivalent to the right rotation: rotateLeft (val, distance) - turn right (shaft, distance). Note also that the rotation of any multiple 32 is non-op, so that all but the last five bits of the rotation argument can be ignored, even if the distance is negative: rotateLeft (val, distance) Settings i int: a value that must rotate at the left distance int: the number of bits of rotation of the left Returns int value received by rotating two binary representations public static int rotateRight (int i, int distance) returns the value received by rotating the binary representation of the two int add-ons directly to the specified number of bits. (The bits are moved from the right hand, or low-order, lateral re-in-view on the left, or high order.) Note that the right rotation with a negative distance is equivalent to the left rotation: turn right (val, distance) - rotateLeft (int i, int distance) Note also that the rotation of any multiple 32 is non-op, so that all but the last five bits of the rotation distance can be ignored, even if the distance is negative: rotateRight (val, distance) - rotateRight (val, distance 0x1F). I int: A value that must rotate at the right distance int: the number of bit positions to rotate the right Returns int value obtained by rotating two binarys of the specified int right value by a specified number of bits. public shortValue () returns the meaning of this Integer as brief after the narrowing of the primitive conversion. Returns the short numerical value presented by this object after the conversion to score a short one. public static int signum (int i) returns the signum function of the specified int value. public static int sum (int a, int b) adds two integrators together according to the operator. Int Options: First Operand b int: second operand Returns int amount a and b public static String toBinaryString (int i) Returns the integer argument presentation line as an unsigned integer in base 2. This value is converted into a string of ASCII numbers in binary (base 2) without additional leading 0s. If the unsigned value is zero, it is represented by one zero symbol '0' ('u0030); otherwise, the first argument symbol will not be a zero symbol. The '0' ('u0030' and '1' ('u0031') symbols are used as binary values. i int options: an integer that needs to be converted into a string. Returns the line view of the unsigned integrative value presented by the binary argument (base 2). See also: parseUnsignedInt (String, int)toUnsignedString (int, int) public static line toHexString (int i) Returns the integer argument line as an unsigned integrator in base 16. The unsigned value is an argument plus 232 if the argument is negative; otherwise it equals the argument. This value is converted into a string of ASCII numbers in six-digit (base 16) without additional leading 0s. If the unsigned value is zero, it is represented by one zero symbol '0' ('u0030); otherwise, the first unsigned symbol will not be a zero value. The following symbols are used as six-digit numbers: 0123456789abcdef These symbols are 'u0030' through 'u0039' and 'u0061' via 'u0066'. If upper-case letters are needed, the String.toUpperCase method () can be called on the result: Integer.toHexString (n).toUpperCase () i int Options: integer to convert to string. Returns the line of view of the unsigned integrative value represented by the hexagonal argument (base 16). See also: parseUnsignedInt (String, int)toUnsignedString (int, int) public static line toOctalString (int i, int) public static line toOctalString (int i) Returns the integer argument line as an unsigned integrator in Base 8. The unsigned value is an argument plus 232 if the argument is negative; otherwise it equals the argument. This value is converted into a string of ASCII numbers in the Octal (Base 8) without additional leading 0s. If the unsigned value is zero, it is represented by one zero symbol '0' ('u0030); otherwise, the first unsigned symbol will not be a zero value. The following symbols are used as numbers: 01234567 These symbols are 'u0030' via 'u0037'. I int options: an integer that needs to be converted into a string. Returns the line of view of the unsigned integrative value represented by the argument in the octal (base 8). See also: parseUnsignedInt (String, int)toUnsignedString (int, int) public static line toString () Returns the string object representing the value of this Integer. The argument is converted to a signed decimal view and returned to the line, just as if the integer value was given as arguments to the toString method (int, int). i int options: integer, which must be converted. Returns the line to the presentation of the argument line in base 10. toString () returns the line object representing the specified integer. The argument is converted into a signed decimal representation and returns as a line, just as if the argument and radix 10 were given as arguments to the toString method (int, int). i int options: integer, which must be converted. Returns the line to the value of that object in base 10. toString () returns the line object representing the specified integer. The argument is converted into a signed decimal representation and returns as a line, just as if the argument and radix 10 were given as arguments to the toString method (int, int). i int options: integer, which must be converted. Returns the line of the line of the value of that object in base 10. toString () returns the line object representing the specified integer. If the radix is smaller than Character.MIN_RADIX or more, Character.MAX_RADIX, then instead radix 10 is used. Note that because the first argument is seen as unsigned, none of the leading symbols of the sign is printed. If the value is zero, it is represented by one zero symbol '0' ('u0030'); otherwise, the first symbol of the representation of magnitude will not be a zero symbol. The following ASCII symbols are used as numbers: 0123456789abcdefghijklmnopqrstuvwxyz It's 'u0030' via 'u0039' and 'u0061' via 'u007A'. If radix N, the first N of these symbols is used as radix-N numbers in the shown order. Thus, the figures for hexadecimal (radix 16) are 0123456789abcdef. If you wish for the letters of the upper register, the String.toUpperCase method () can be called by result: Integer.toString (n, 16).toUpperCase () Settings i int: integer, which must be converted into a string. radix int: radix for use in a line view. Returns the argument representation line in the specified radix. See also: Character.MAX_RADIXCharacter.MIN-RADIX Public Static Long toUnsignedLong (int x) Transforms the argument into a long integrative transformation. The unsigned conversion in the long, high order is 32 bits long to zero and a low-order 32 bits equal bits of integer argument. Consequently, zero and positive int values are displayed at numerically equal long value, and negative int values long value equal to entry plus 232. 232. x int: The value for converting to an unsigned long Returns-long argument is transformed into a long integrative method (java.lang. String, int). Options s Line: A line containing an unsigned int view presented by the argument line as presented by the argument in the radix, specified by the second argument. See also: toUnsignedString (int, int) public static line toUnsignedString (int, int radix) returns the representation and returns as a line just as if the argument and radix 10 were given as arguments to the method toUnsignedString (int, int). i int options: integrator that needs to be converted into an unsigned line. Returns the unsigned line to the argument view. See also: toUnsignedString (int, int) public static line toUnsignedString (int, int radix) returns the representation of the first argument line as an unsigned integrative representation and returns as a line just as if argument and radix 10 were given as arguments to the method toUnsignedString (int, int). If radix is smaller than Character.MIN_RADIX or more, Character.MAX_RADIX, then instead radix 10 is used. Note that because the first argument is seen as unsigned, none of the leading symbols of the sign is printed. If the value is zero, it is represented by one zero symbol '0' ('u0030'); otherwise, the first symbol of the representation of magnitude will not be a zero symbol. The behavior of radixes and symbols used as numbers is the same as toString. I int options: an integrator that needs to be converted into an unsigned line. radix int: radix for use in a line view. Returns the unsigned view of the argument line in the radix specified by the second argument. See also: toString line (int, int) public value of IntegerOf (String s) returns the Integer object holding the value of the specified line. The argument is interpreted as representing a signed decimal integrator, just as if the argument were given to the parseInt (java.lang.String) method. The result is an Integer that represents the integer value specified by the line. In other words, this method returns an Integer object equal to the value of: the new Integer (Integer.parseInt (s). Options s Line: The line to be disassembled. Returns the integer to the Integer object that hold the value represented by the line argument. Throws NumberFormatException if the string can't be disassembled as an integer. public static Value of IntegerOf (String s, int radix) returns the Integer object holding the value extracted from the specified line when parsing with the radix indicated by the second argument. The first argument is interpreted as representing a signed integrator in the radix, specified by the second argument, just as if the arguments were given to the parseInt method (java.lang.String, int). The result is an Integer that represents the integer value specified by the line. In other words, this method returns an Integer object equal to the value of a new (Integer.parseInt (s, radix)) Options s Row: A line for parsing. radix Int: radix to be used in the interpretation of s Returns Integer object holding a value represented by a line argument in the radix. public NumberFormatException throws if the line does not contain a disassemblable int. public static Integer ValueOf (int i) returns an Integer copy representing the specified int value. If a new Instance of Integer is not required, this method should usually be used instead of the Integer (int) constructor, as this method can give a much higher performance of space and time by caching often requested values. This method will always cache values ranging from

-128 to 127 inclusive and can cache other values outside that range. I int: int value. Returns Integer integer copy representing i. i. i. i. convert int to string java android studio