stamp.math

Class Int32

```
java.lang.Object
  |
  +--stamp.math.Int32
```

public class **Int32**
extends Object

This class provides support for 32-bit integers.

Support is provided for both signed and unsigned numbers. The range of values that can be stored is as follows:

```
 32-bit signed number:   -2,147,483,648 to 2,147,483,647
 32-bit unsigned number:  0 to 4,294,967,295
```

The 32-bit integer is stored as two 16-bit fields which can be accessed directly.
Methods are provided to support the following operations:

- create 32-bit integer objects
- set the value
- convert a String or StringBuffer to a 32-bit value
- add, subtract
- multiply, divide (signed and unsigned), remainder
- shift left, shift right
- absolute value, negate
- compare (signed and unsigned), equality
- convert 32-bit integer to string (signed and unsigned)

Field Summary

`int`	**`high`** high 16 bits of 32-bit integer.
`int`	**`low`** low 16 bits of 32-bit integer.

Constructor Summary

`Int32()` Create a new 32-bit integer initialized to zero.
`Int32(int low)` Create a new 32-bit integer initialized to the 16-bit value.
`Int32(Int32 num)` Create a new 32-bit integer initialized to value of another 32-bit integer.
`Int32(int high, int low)` Create a new 32-bit integer initialized to 32-bit value.

Method Summary

`void`	`abs()` Sets the 32-bit integer to its absolute value.
`void`	`add(int low)` Adds a 16-bit value (sign extended) to the 32-bit integer.
`void`	`add(Int32 num)` Adds another 32-bit integer to the 32-bit integer.
`void`	`add(int high, int low)` Adds a 32-bit value to the 32-bit integer.
`int`	`compare(int low)` Performs a signed comparison of the 32-bit integer to a 16-bit integer.
`int`	`compare(Int32 num)` Performs a signed comparison of the 32-bit integer to another 32-bit integer.
`int`	`compare(int high, int low)` Performs a signed comparison of the 32-bit integer to a 32-bit value.
`void`	`divide(int low)` Divides the 32-bit integer by a 16-bit value.
`void`	`divide(Int32 num)` Divides the 32-bit integer by another 32-bit integer.
`void`	`divide(int high, int low)` Divides the 32-bit integer by a 32-bit value.
`boolean`	`equals(Int32 num)` Checks if 32-bit integer is equal to another 32-bit integer.
`void`	`multiply(int low)` Multiplies the 32-bit integer by a 16-bit value.
`void`	`multiply(Int32 num)` Multiplies the 32-bit integer by another 32-bit integer.
`void`	`multiply(int high, int low)` Multiplies the 32-bit integer by a 32-bit value.

`void`	`negate()` Negates the 32-bit integer.
`void`	`remainder(Int32 num)` Sets another 32-bit integer to the remainder of the last divide or udivide.
`void`	`set(int low)` Sets the 32-bit integer to a 16-bit value (sign extended).
`void`	`set(Int32 num)` Sets the 32-bit integer equal to the value of another 32-bit integer.
`void`	`set(int high, int low)` Sets the 32-bit integer to a 32-bit value.
`void`	`set(String s)` Sets the 32-bit integer to the converted value of the String.
`void`	`set(StringBuffer sb)` Sets the 32-bit integer to the converted value of the StringBuffer.
`void`	`shiftLeft(int count)` Shifts the 32-bit integer left (0 enters LSB).
`void`	`shiftRight(int count)` Shifts the 32-bit integer right (0 enters MSB).
`void`	`subtract(int low)` Subtracts a 16-bit value (sign extended) from the 32-bit integer.
`void`	`subtract(Int32 num)` Subtracts another 32-bit integer from the 32-bit integer.
`void`	`subtract(int high, int low)` Subtracts a 32-bit value from the 32-bit integer.
`String`	`toString()` Convert the signed 32-bit integer to a String.
`int`	`ucompare(int low)` Performs an unsigned comparison of the 32-bit integer to a 16-bit integer.
`int`	`ucompare(Int32 num)` Performs an unsigned comparison of the 32-bit integer to another 32-bit integer.
`int`	`ucompare(int high, int low)` Performs an unsigned comparison of the 32-bit integer to another 32-bit integer.
`void`	`udivide(int low)` Divides the 32-bit integer by a 16-bit value (unsigned divide).
`void`	`udivide(Int32 num)` Divides the 32-bit unsigned integer by another 32-bit unsigned integer.
`void`	`udivide(int high, int low)` Divides the 32-bit unsigned integer by a 32-bit unsigned value.
`String`	`utoString()` Convert the unsigned 32-bit integer to a String.

Methods inherited from class java.lang.Object
`equals`

Field Detail

high

```
public int high
```

high 16 bits of 32-bit integer.

---

low

```
public int low
```

low 16 bits of 32-bit integer.

Constructor Detail

Int32

```
public Int32()
```

Create a new 32-bit integer initialized to zero.

Int32

```
public Int32(Int32 num)
```

Create a new 32-bit integer initialized to value of another 32-bit integer.

Parameters:
`num` - 32-bit integer

Int32

```
public Int32(int low)
```

Create a new 32-bit integer initialized to the 16-bit value.

Parameters:
`low` - low 16-bits of initial value (sign extends to high 16-bits)

Int32

```
public Int32(int high,
             int low)
```

Create a new 32-bit integer initialized to 32-bit value.

Parameters:
`high` - high 16-bits of initial value
`low` - low 16-bits of initial value

Method Detail

set

```
public void set(Int32 num)
```

Sets the 32-bit integer equal to the value of another 32-bit integer.

Parameters:
`num` - 32-bit integer

---

set

```
public void set(int low)
```

Sets the 32-bit integer to a 16-bit value (sign extended).

Parameters:
`low` - low 16-bits of value (sign extends to high 16-bits)

---

set

```
public void set(int high,
                int low)
```

Sets the 32-bit integer to a 32-bit value.

Parameters:
`high` - high 16-bits of value
`low` - low 16-bits of value

---

set

```
public void set(String s)
```

Sets the 32-bit integer to the converted value of the String. The conversion skips leading whitespace and stops at first non-decimal character.

Parameters:
`s` - String containing 32-bit integer

---

set

```
public void set(StringBuffer sb)
```

Sets the 32-bit integer to the converted value of the StringBuffer. The conversion skips leading whitespace and stops at first non-decimal character.

Parameters:
`sb` - StringBuffer containing 32-bit integer

---

add

```
public void add(Int32 num)
```

Adds another 32-bit integer to the 32-bit integer.

Parameters:
`num` - 32-bit integer

---

add

```
public void add(int low)
```

Adds a 16-bit value (sign extended) to the 32-bit integer.

Parameters:
`low` - 16-bit value (sign extends to high 16-bits)

---

add

```
public void add(int high,
                int low)
```

Adds a 32-bit value to the 32-bit integer.

Parameters:
`high` - high 16-bits of value
`low` - low 16-bits of value

---

subtract

```
public void subtract(Int32 num)
```

Subtracts another 32-bit integer from the 32-bit integer.

Parameters:
`num` - 32-bit integer.

---

subtract

```
public void subtract(int low)
```

Subtracts a 16-bit value (sign extended) from the 32-bit integer.

Parameters:
`low` - 16-bit value (sign extends to high 16-bits)

---

subtract

```
public void subtract(int high,
                     int low)
```

Subtracts a 32-bit value from the 32-bit integer.

Parameters:
`high` - high 16-bits of value
`low` - low 16-bits of value

---

divide

```
public void divide(Int32 num)
```

Divides the 32-bit integer by another 32-bit integer.

Parameters:
`num` - 32-bit integer

divide

```
public void divide(int low)
```

Divides the 32-bit integer by a 16-bit value.

Parameters:
`low` - 16-bit value (sign extends to high 16-bits)

divide

```
public void divide(int high,
                   int low)
```

Divides the 32-bit integer by a 32-bit value.

Parameters:
`high` - high 16-bits of value
`low` - low 16-bits of value

udivide

```
public void udivide(Int32 num)
```

Divides the 32-bit unsigned integer by another 32-bit unsigned integer. Note: the divisor is restricted to 31 bits to allow for an optimization in the divide routine.

Parameters:
`num` - 32-bit integer

udivide

```
public void udivide(int low)
```

Divides the 32-bit integer by a 16-bit value (unsigned divide).

Parameters:
`low` - 16-bit value (high 16-bits set to 0)

udivide

```
public void udivide(int high,
                int low)
```

Divides the 32-bit unsigned integer by a 32-bit unsigned value. Note: the divisor is restricted to 31 bits to allow for an optimization in the divide routine.

Parameters:
`high` - high 16-bits of value
`low` - low 16-bits of value

---

remainder

```
public void remainder(Int32 num)
```

Sets another 32-bit integer to the remainder of the last divide or udivide.

Parameters:
`num` - 32-bit integer will be set to remainder

---

multiply

```
public void multiply(Int32 num)
```

Multiplies the 32-bit integer by another 32-bit integer.

Parameters:
`num` - 32-bit integer

---

multiply

```
public void multiply(int low)
```

Multiplies the 32-bit integer by a 16-bit value.

Parameters:
`low` - 16-bit value (sign extends to high 16-bits)

---

multiply

```
public void multiply(int high,
                     int low)
```

Multiplies the 32-bit integer by a 32-bit value.

Parameters:

`high` - high 16-bits of value
`low` - low 16-bits of value

---

shiftRight

`public void **shiftRight**(int count)`

Shifts the 32-bit integer right (0 enters MSB).

Parameters:
`count` - number of bit positions to shift

---

shiftLeft

`public void **shiftLeft**(int count)`

Shifts the 32-bit integer left (0 enters LSB).

Parameters:
`count` - number of bit positions to shift

---

compare

`public int **compare**(Int32 num)`

Performs a signed comparison of the 32-bit integer to another 32-bit integer.

Parameters:
`num` - 32-bit integer
Returns:
-1 if the 32-bit integer is less than another 32-bit integer
 0 if the 32-bit integer equals another 32-bit integer
 1 if the 32-bit integer is greater than another 32-bit integer

---

compare

`public int **compare**(int low)`

Performs a signed comparison of the 32-bit integer to a 16-bit integer.

Parameters:
`low` - 16-bit value (sign extends to high 16-bits)
Returns:
-1 if the 32-bit integer is less than 16-bit value

  0 if the 32-bit integer equals the 16-bit value
  1 if the 32-bit integer is greater than 16-bit value

---

compare

```
public int compare(int high,
                   int low)
```

Performs a signed comparison of the 32-bit integer to a 32-bit value.

Parameters:
`high` - high 16-bits of value
`low` - low 16-bits of value
Returns:
-1 if the 32-bit integer is less than 32-bit value
  0 if the 32-bit integer equals 32-bit value
  1 if the 32-bit integer is greater than 32-bit value

---

ucompare

```
public int ucompare(Int32 num)
```

Performs an unsigned comparison of the 32-bit integer to another 32-bit integer.

Parameters:
`num` - 32-bit integer
Returns:
-1 if the 32-bit integer is less than another 32-bit integer
  0 if the 32-bit integer equals another 32-bit integer
  1 if the 32-bit integer is greater than another 32-bit integer

---

ucompare

```
public int ucompare(int low)
```

Performs an unsigned comparison of the 32-bit integer to a 16-bit integer.

Parameters:
`low` - 16-bit value (high 16-bits set to zero)
Returns:
-1 if the 32-bit integer is less than 16-bit value
  0 if the 32-bit integer equals the 16-bit value
  1 if the 32-bit integer is greater than 16-bit value

---

ucompare

```
public int ucompare(int high,
                    int low)
```

Performs an unsigned comparison of the 32-bit integer to another 32-bit integer.

Parameters:
`high` - high 16-bits of value
`low` - low 16-bits of value
Returns:
-1 if the 32-bit integer is less than 32-bit value
 0 if the 32-bit integer equals 32-bit value
 1 if the 32-bit integer is greater than 32-bit value

---

equals

```
public boolean equals(Int32 num)
```

Checks if 32-bit integer is equal to another 32-bit integer.

Parameters:
`num` - 32-bit integer
Returns:
`true` if equal

---

abs

```
public void abs()
```

Sets the 32-bit integer to its absolute value.

---

negate

```
public void negate()
```

Negates the 32-bit integer.

---

toString

```
public String toString()
```

Convert the signed 32-bit integer to a String.

utoString

```
public String utoString()
```

Convert the unsigned 32-bit integer to a String.