Short Scale String

by PolyLabs

version 1.3 (September, 2020)

support@polylabs.co

Contents:

Introduction

Short Scale String is an asset for converting numeric input `int, float, and double` into a short scale representation of that value. When large numbers need to be represented in an easy to read value, this can be of great value to the user.

Short scale representation is a number naming system for powers of ten; in short scale representation, each term is a thousand times greater than the previous. This system is adopted throughout the US and UK as a recognized numbering scale. For example, the number **1,000,000** is represented as **1 million** in short scale. The official system covers numbers up to **centillion**.

Methods

Common Parameters

Name	Type	Description	Default
value	`int,` `float,` `double`	The value input to a given method expressed as `int, float, or double`	
precision	int	Number of digits of precision for the output number to display.	3
startShortScale	`int,` `float,` `double`	Any number lower than this value will be a string representation of itself.	1,000,000
useSymbol	bool	Should the parser use the shortened notation (currently only supports up to a Decillion)	false

Integers

ParseInt parses an integer into short scale notation.

```
string ParseInt(int value, int precision = 3, int startShortScale = 1000000, bool
useSymbol = false);
```

ParseIntSplit parses an integer into short scale notation. Returns a `ShortScaleData` object with the value and symbol represented as a string.

```
ShortScaleData ParseIntSplit(int value, int precision = 3, int startShortScale =
1000000, bool useSymbol = false);
```

Overload: returns nothing, but accepts a reference to an existing `ShortScaleData` object and sets the data.

```
void ParseIntSplit(int value, ref ShortScaleData data, int precision = 3, int
startShortScale = 1000000, bool useSymbol = false);
```

Floats

ParseFloat parses a float into short scale notation.

```
string ParseFloat(float value, int precision = 3, float startShortScale =
1000000, bool useSymbol = false);
```

ParseFloatSplit parses a float into short scale notation. Returns a `ShortScaleData` object with the value and symbol represented as a string.

```
ShortScaleData ParseFloatSplit(float value, int precision = 3, float
startShortScale = 1000000, bool useSymbol = false);
```

Overload: returns nothing, but accepts a reference to an existing `ShortScaleData` object and sets the data.

```
void ParseFloatSplit(float value, ref ShortScaleData data, int precision = 3,
float startShortScale = 1000000, bool useSymbol = false);
```

Doubles

ParseDouble parses a double into short scale notation.

```
string ParseDouble(double value, int precision = 3, double startShortScale =
1000000, bool useSymbol = false);
```

ParseDoubleSplit parses a double into short scale notation. Returns a `ShortScaleData` object with the value and symbol represented as a string.

```
ShortScaleData ParseDoubleSplit(double value, int precision = 3, double
startShortScale = 1000000, bool useSymbol = false);
```

Overload: returns nothing, but accepts a reference to an existing `ShortScaleData` object and sets the data.

```
void ParseDoubleSplit(double value, ref ShortScaleData data, int precision = 3,
double startShortScale = 1000000, bool useSymbol = false);
```

Examples

Integers

```
// Base:
int myNumber = 123456789;
string shortScaleNum = PolyLabs.ShortScale.PartseInt(myNumber);
Debug.Log(shortScaleNum); // "123.456 million"
```

```
// Short Scale Data:
ShortScaleData data = PolyLabs.ShortScale.ParseIntSplit(myNumber);
Debug.Log(data.value); // "123.456"
Debug.Log(data.symbol); // "million"

// Reference passing data object from above:
myNumber = 987654321;
PolyLabs.ShortScale.ParseIntSplit(myNumber, ref data);
Debug.Log(data.value); // "987.654"
Debug.Log(data.symbol); // "million"
```

Floats

```
// Base:
float myNumber = 9999999999.9f;
string shortScaleNum = PolyLabs.ShortScale.PartseFloat(myNumber);
Debug.Log(shortScaleNum); // "9.999 billion"

// Short Scale Data with 4 digits of precision:
ShortScaleData data = PolyLabs.ShortScale.ParseFloatSplit(myNumber, 4);
Debug.Log(data.value); // "9.9999"
Debug.Log(data.symbol); // "billion"

// Reference passing data object from above:
myNumber = 1000100000;
PolyLabs.ShortScale.ParseFloatSplit(myNumber, ref data, 4);
Debug.Log(data.value); // "1.0001"
Debug.Log(data.symbol); // "billion"
```

Double

```
using UnityEngine;
using UnityEngine.UI;
// Import the namespace for ease-of-use:
using PolyLabs;

public class ShortScaleExample: MonoBehaviour
{
  // Get the data from here:
  public InputField inputTarget;
  // Display the data to here:
  public Text outputTarget;

  void Update()
  {
    double inputValue = double.Parse(inputTarget.text);
    outputTarget.text = ShortScale.ParseDouble(inputValue);
  }
}
```