Integer Language-Independent Type for YAML™

Working Draft 200?-??-??

Oren Ben-Kiki <oren@ben-kiki.org>
Clark Evans <cce+yaml@clarkevans.com>
Brian Ingerson <ingy@ttul.org>

Status

This specification is a release candidate and reflects consensus reached by members of the yaml-core mailing list. Any questions regarding this draft should be raised on this list at http://lists.sourceforge.net/lists/listinfo/yaml-core. With this release of the YAML specification, all further changes will be strictly limited to clarifications, or fixing bugs in productions. At this point, further enhancement or correction of logical flaws will be put off to the next version (1.1) of the YAML specification.

URI: **`tag:yaml.org,2002:int`**

Shorthand: **`!int`**

Kind: Scalar.

Definition: Mathematical integers.

Integers represent arbitrarily sized finite mathematical integers. Integers can be formatted using the familiar decimal notation, or may have a leading “**`0x`**” to signal hexadecimal, or a leading “**`0`**” to signal an octal base. Using “**`:`**” allows expressing integers in base 60, which is convenient for time and angle values. Any “**`,`**” characters in the number are ignored, allowing a readable representation of large values.

Scalars of this type should be represented by a native integer data type, if possible. However, there are cases where an integer provided may overflow the native type's storage capability. In this case, the processor should find some manner to round-trip the integer, perhaps using a string based representation. In general, integers representable using 32 binary digits should safely round-trip through most systems.

Canonical Format: **`0|-?[1-9]&[0-9]*`** ***(base 10)***

Resolution and Validation: Valid values must match the followig regular expression, which may also be used for implicit tag resolution:

```
[-+]?0[0-8,]+ (base 8)
|[-+]?(0|[1-9])[0-9,]*) (base 10)
|[-+]?0x[0-9a-fA-F,]+ (base 16)
|[-+]?[1-9][0-9,]*(:[0-5]?[0-9])+ (base 60)
```

Example 1. `!int` Examples

```
canonical: 685230
decimal: +685,230
octal: 02472256
hexadecimal: 0x0A,74,AE
sexagesimal: 190:20:30
```