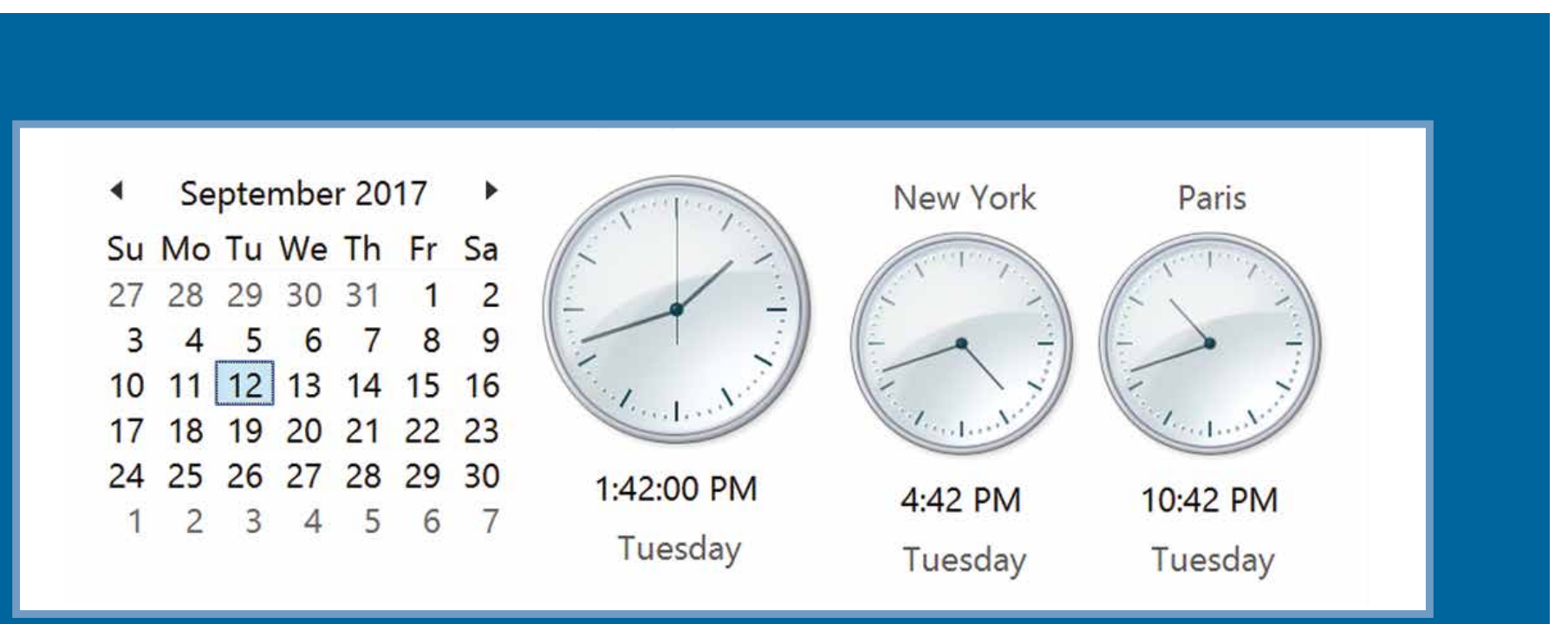
Datetime Programming For Everyone. Everywhere.

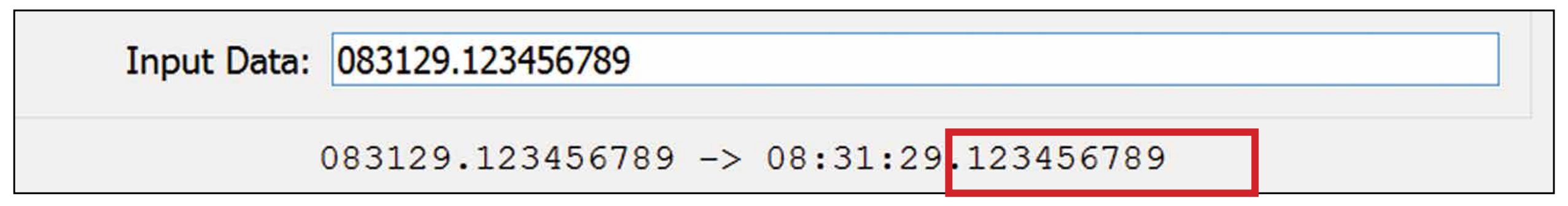
Tai Meng (tai.meng@safe.com) | 604 501 9985 x 246 | Safe Software | safe.com



◂ September 2017 ▸						
Su	Mo	Tu	We	Th	Fr	Sa
27	28	29	30	31	1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
1	2	3	4	5	6	7

Datetime Manipulations

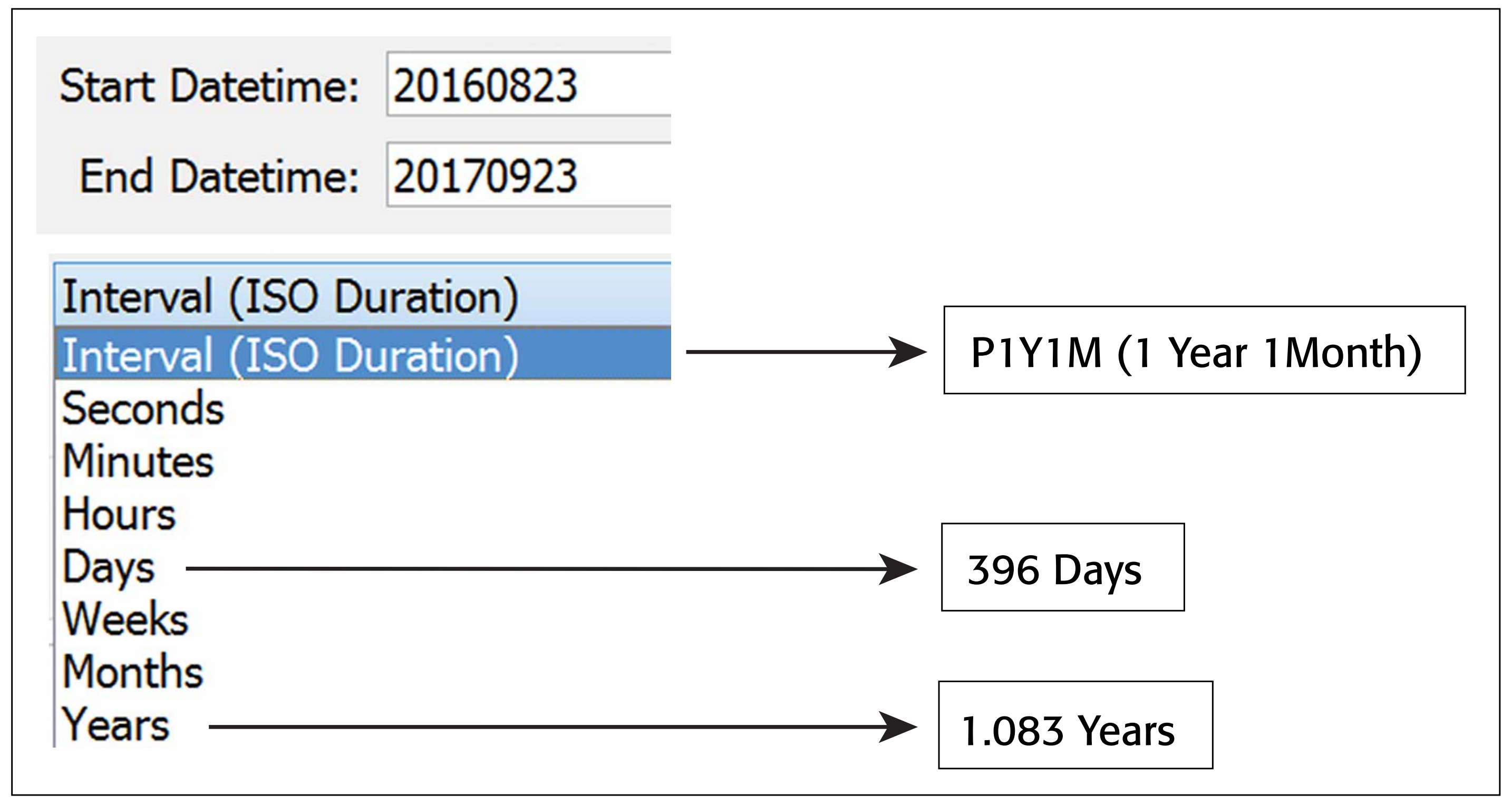
Nanosecond Precision



Input Data: 083129.123456789

```
083129.123456789 -> 08:31:29.123456789
```

Math



Start Datetime: 20160823

End Datetime: 20170923

P1Y1M (1 Year 1Month)

396 Days

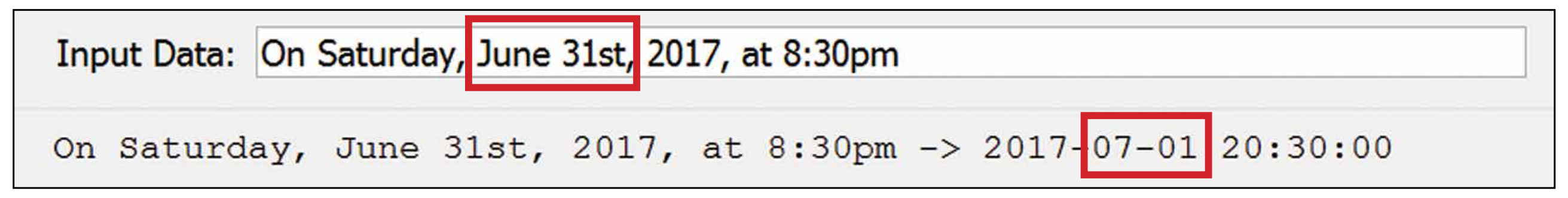
1.083 Years

Time Zone Conversions

@TimeZoneSet(06:00:00–07:00, –04:00)
= 09:00:00–04:00

@TimeZoneSet(2017-05-26 06:00:00+00:00, local)
= 2017-05-25 23:00:00–07:00

Validation & Repair



Input Data: On Saturday, June 31st, 2017, at 8:30pm

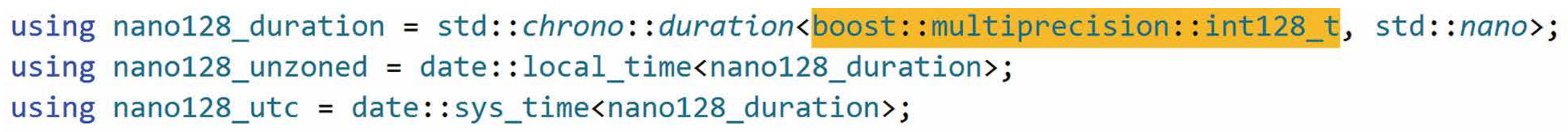
```
On Saturday, June 31st, 2017, at 8:30pm -> 2017-07-01 20:30:00
```

Library Survey

boost	✗	No IANA time zone support
Qt	✗	Precision too low
Google	✗	Does not work on Windows
python	✗	Precision too low
Bloomberg	✗	No IANA time zone support
php	✗	Precision too low
IBM	✗	No IANA time zone support
Tcl	✗	Precision too low
	✓	Howard Hinnant's date/tz libraries!

Howard + Boost

Nanosecond Precision: int128_t



```
using nano128_duration = std::chrono::duration<boost::multiprecision::int128_t, std::nano>;
using nano128_unzoned = date::local_time<nano128_duration>;
using nano128_utc = date::sys_time<nano128_duration>;
```

Patterns

Math Invariant

EndTime – StartTime = Interval iff
StartTime + Interval = EndTime

Examples:

2017-02-28 – 2016-02-29 = 1 year
2016-02-29 + 1 year = 2017-02-28

2017-02-28 – 2017-01-31 = 1 month
2017-01-31 + 1 month = 2017-02-28

Calendar Math Asymmetric!

2017-09-30 – 2017-08-31 = 1 month
2017-08-31 – 2017-09-30 = -30 days

Mixed Calendar & Clock Math

2017-09-23 – 2016-08-11 03:04:05 = P1Y1M11DT20H55M55S
(1 Year 1 Month 11 Days 20 Hours 55 Minutes 55 Seconds)

Steps:

1. Find number of full calendar years between start and end times.
2. Add result of 1 to start time.
3. Find number of full calendar months between start and end times.
4. Add result of 3 to start time.
5. The remaining diff contains days, hours, minutes, and seconds.

Type Conversion

Five Datetime Types:

1. Date
2. Time
3. Datetime
4. Time with UTC offset
5. Datetime with UTC offset

Type Conversion Continued...

Math

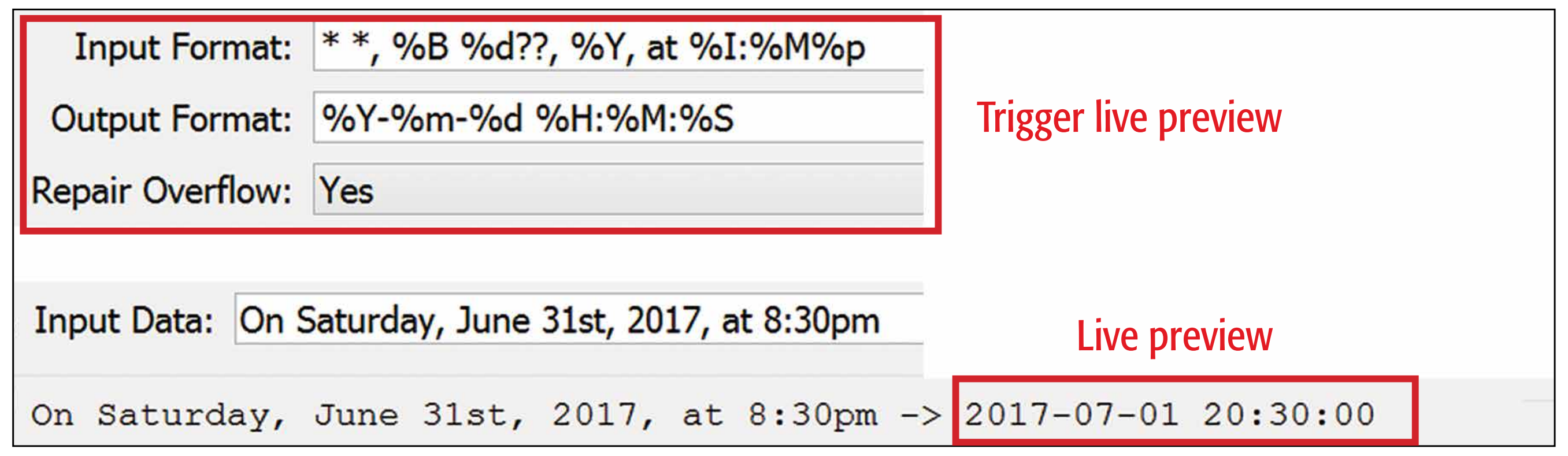
- ✓ Dates autocast into datetimes by adding midnight.
- ✗ All other implicit casts are forbidden.

Writing Datetime Values

- ✓ Drop unneeded date, time, and/or UTC offset.
- ✓ Fabricate midnight.
- ✓ Round if extra precision. Don't trim.
- ✗ Do not fabricate date.
- ✗ Do not fabricate UTC offset (or time zone).

Validation & Repair Continued...

- ✓ Validate and repair in a single tool. All other datetime tools such as datetime calculator should assume valid input.
- ✓ For complex datetime input, such as natural language strings, a GUI tool with live preview will save users a TON of time in creating parse/format strings.



Input Format: * *, %B %d??, %Y, at %I:%M%p

Output Format: %Y-%m-%d %H:%M:%S

Repair Overflow: Yes

Trigger live preview

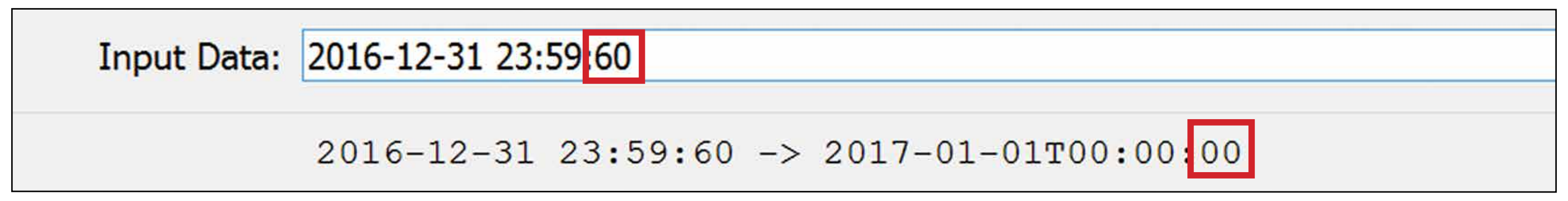
Input Data: On Saturday, June 31st, 2017, at 8:30pm

Live preview

```
On Saturday, June 31st, 2017, at 8:30pm -> 2017-07-01 20:30:00
```

Leap Seconds

For users who need to parse, but not preserve leap seconds:



Input Data: 2016-12-31 23:59:60

```
2016-12-31 23:59:60 -> 2017-01-01T00:00:00
```

Epoch Time Conversion

- The world has both zoned and unzoned epoch times!
- Need a way to convert to/from both of them!

Pitfalls

- ✓ UTC offsets: careful adding UTC offsets to datetime values. The – sign often means + and the + sign often means –!
- ✗ Keep trailing fractional zeros in input? Don't.

Howard's Date/Tz Libraries

- Open source: ***https://github.com/HowardHinnant/date***
- World class support: ***https://gitter.im/HowardHinnant/date***
- Email: ***howard.hinnant@gmail.com***