Ruby master - Feature #7114

New classes: `HumanTime::LocalTime`, `HumanTime::Duration`

10/08/2012 12:47 AM - alexeymuranov (Alexey Muranov)

Status:	Feedback	
Priority:	Normal	
Assignee:		
Target version:		

Description

=begin
As suggested by *drbrain* in #7113 , here is a separate request for classes to hold the time of day and duration of time.

Ruby currently does not have a class to represent a standard database "time" column. Using (({Time})) for this causes some difficulties for example to users of *Ruby on Rails*: http://www.rebeccablyth.co.uk/2008/01/03/time-columns-and-rails/
=end

History

#1 - 10/08/2012 01:38 AM - alexeymuranov (Alexey Muranov)

Maybe TimeOfDay instead of LocalTime. But i am not entirely convinced that nobody would want to have a LocalDate class too, in which case LocalTime is better.

#2 - 10/09/2012 03:36 PM - nobu (Nobuyoshi Nakada)

Sounds like that AR can provide AR-specific parse method.

#3 - 10/09/2012 03:50 PM - naruse (Yui NARUSE)

- Status changed from Open to Feedback

Once Ruby had Date::Delta but removed because it is undocumented experimental feature #4391.

If you trouble with Rails, you should consider the feature should be a part of Rails (in this case ActiveSupport).

Anyway this request should have more detail, what method should be required for the class?

#4 - 10/12/2012 09:40 PM - alexeymuranov (Alexey Muranov)

In my opinion, the most important method of HumanTime::LocalTime would be initialize ;). It would need to be able to take the number of hours and minutes, or a string and a pattern description, and store it as "hh:mm:ss", with or without the timezone. The main "problem" with Time is that it cannot store relative time, time of day, or duration.

By the way, i do not think that the inner working of a class is not a part of its specification: the method implementation maybe is not, but i think that knowing what exactly is stored as the state of an object and in which format (as conversion between formats may either take time or be not lossless) is a part of a class specification. This is why i think that a class that stores the number of seconds from Epoch is not well suited for keeping a shop open hours or something human like this.

I do not have a precise idea about the class interface for now.

#5 - 10/19/2012 10:29 AM - wardrop (Tom Wardrop)

Personally, I find Ruby's whole Time API to be lack-luster and frustrating. It's probably the thing I like least about Ruby. I'd be in favor of a total redesign of Ruby's Time classes. A certain number of improvements could be made without creating backwards compatibility problems (look at ActiveRecord), but a total new set of classes would be preferred. You can have a base class of Time (maybe in a new namespace) with a whole load of subclasses for representing different types of time, e.g. Date, DateTime, Duration, Range, etc, and there should be logical operator support between all of this, e.g. DateTime - Duration should behave as expected, and a class like Range could have all kinds of useful methods like Date.today.in?(Range.new '2012-01-01', '2012-12-31'). Range would also allow for methods like Range.from_year(2012) which would be the same as Range.new('2012-01-01', '2012-12-31') . Of course, ranges and duration are similar, so you'd have methods to convert between the two, e.g. Range.to_duration and Duration.to_range('2012-10-01') which would create a date range beginning at the given date/time which would extend for the duration of the Duration object (self).

There's some real potential there obviously. Times are painful enough in programming. You have to parse ambiguous date formats like 12/12/2012, deal with timezones and store and retrieve times from other systems which have there own quirks. Having a solid Time library would make dealing with this common problem just that little less painful.

#6 - 10/23/2012 01:00 AM - headius (Charles Nutter)

It might be useful to look at the JodaTime library for the JVM. JRuby uses JodaTime to implement Time internally, and we have debated using it to implement most of 'date' ext as well. Java has suffered through a number of lackluster date/time APIs over the years, and JodaTime is widely considered to be the most correct, best API available. It's so good, in fact, that it will be incorporated into Java 8.

http://joda-time.sourceforge.net/

#7 - 10/23/2012 01:10 AM - shyouhei (Shyouhei Urabe)

It seems the picture is drawn much bigger now, than what alexeymuranov (Alexey Muranov) initially proposed.

That's not a bad thing itself but might diverse the discussion. Could we have a different story about Time/Date renovation?

#8 - 11/24/2012 09:20 AM - mame (Yusuke Endoh)

- Target version set to 2.6

#9 - 12/25/2017 06:15 PM - naruse (Yui NARUSE)

- Target version deleted (2.6)