```
    elements => _rsectionize(@sections),
  });
}

=func preformatted

  my $rich_text_pre = preformatted(@elements);
  # or
  my $rich_text_pre = preformatted(\%arg, @elements);

This returns a new L<preformatted rich
text|Slack::BlockKit::Block::RichTextPreformatted> block object, with the given
elements as its C<elements>.  Any non-references in the list will be turned
into text objects.

The leading hashref, if given, will be used as extra arguments to the block
object's constructor.

B<Beware>:  The Slack documentation suggests that C<emoji> objects can be
present in the list of elements, but in practice, this always seems to get
rejected by Slack.

=cut

sub preformatted (@elements) {
  my $arg = _HASH0($elements[0]) ? (shift @elements) : {};

  Slack::BlockKit::Block::RichText::Preformatted->new({
    %$arg,
    elements => _rtextify(@elements),
  });
}

=func quote

  my $rich_text_quote = quote(@elements);
  # or
  my $rich_text_quote = quote(\%arg, @elements);

This returns a new L<quoted rich
text|Slack::BlockKit::Block::Quote> block object, with the given elements as
its C<elements>.  Any non-references in the list will be turned into text
objects.

The leading hashref, if given, will be used as extra arguments to the block
object's constructor.

=cut

sub quote (@elements) {
  my $arg = _HASH0($elements[0]) ? (shift @elements) : {};

  Slack::BlockKit::Block::RichText::Quote->new({
    %$arg,
    elements => _rtextify(@elements),
  });
}

=func channel

  my $rich_text_channel = channel($channel_id);
  # or
  my $rich_text_channel = channel(\%arg, $channel_id);

This function returns a L<channel mention
object|Slack::BlockKit::Block::RichText::Channel>, which can be used among
other rich text elements to "mention" a channel.  The C<$channel_id> should be
the alphanumeric Slack channel id, not a channel name.

If given, the C<%arg> hash is extra parameters to pass to the Channel
constructor.

=cut

sub channel {
  my ($arg, $id)
    = @_ == 2 ? @_
    : @_ == 1 ? ({}, $_[0])
    : Carp::croak("BlockKit channel sugar called with wrong number of arguments");

  Slack::BlockKit::Block::RichText::Channel->new({
    %$arg,
    channel_id => $id,
  });
}

=func date

  my $rich_text_date = date($timestamp, \%arg);

This returns a L<rich text date object|Slack::BlockKit::Block::RichText::Date>
for the given time (a unix timestamp).  If given, the referenced C<%arg> can
contain additional arguments to the Date constructor.

Date formatting objects have a mandatory C<format> property.  If none is given
in C<%arg>, the default is:

  \x{200b}{date_short_pretty} at {time}

Why that weird first character?  It's a zero-width space, and suppresses the
capitalization of "yesterday" (or other words) at the start.  This
capitalization seems like a bug (or bad design) in Slack.

=cut

sub date ($timestamp, $arg=undef) {
  $arg //= {};

  Slack::BlockKit::Block::RichText::Date->new({
    format => "\x{200b}{date_short_pretty} at {time}",
    %$arg,
    timestamp => $timestamp,
  });
}

=func emoji

  my $rich_text_emoji = emoji($emoji_name);

This function returns an L<emoji
object|Slack::BlockKit::Block::RichText::Emoji> for the named emoji.

=cut
```