① ayshush.us / pontoon / issues / 3809
② ayshush.us / pontoon / pulls / ####

OR
ayshush.us / pontoon / 3809 && ayshush.us / pontoon / ####

Timestamp format in Notification popup is inconsistent #3809

Open Bug

mathjazz opened on Oct 14, 2025 — Collaborator

React

Django Notification popup and Translate view Notification popup use different time formats in the timestamp tooltip. We should fix that and always show the exact date and time. There's also inconsistency between more recent and olders dates in the format.

if not relative time already being used as a substitute, then.

use single source of truth??

Create sub-issue

- There are 2 pontoon front-ends that are built at different times

Django side
- server-rendered HTML pages
  ↳ eg. dashboards, profiles
- the notifications popup is generated on the server side using a templating language

notifications_menu.html

example Jinja2 macro (reusable fn for HTML templates)

```
idgets > notifications_menu.html > div#notifications.notifications.select.{%.if.unread_count.%}unread{%.endif.%} > di
1  {% macro menu() %}   ← define reusable template block named menu
2    {% if user.is_authenticated %}   guardrail
3
4      {% set unread_count = user.notifications.unread().count() %}   set var database call (for unread notifications)
5      <div
6        id="notifications"
7        class="notifications select {% if unread_count %}unread{% endif %}"
8                                   ↑ if user has 1 or more unread notifs, then add unread class to HTML wrapper
9        <div class="button selector">
10         <i class="icon far fa-bell fa-fw"></i>
11         <i class="badge"                    fontawesome
12           >{{ user.unread_notifications_display(unread_count) }}</i
13           >
14       </div>                                ↑ format number to display notifications clearly
15
16       <div class="menu">
17         {{ list(notifications=user.menu_notifications(unread_count)) }}
18                                   ↑ fetches contents of most recent notifications
19         <ul>
20           <li class="horizontal-separator"></li>
21           <li class="see-all">
22             <a
23               href="{{ full_url('pontoon.contributors.notifications') }}?referrer=ui"
24               >See all Notifications</a
25               >
26           </li>
27         </ul>
28       </div>
29     </div>
30   {% endif %}
31 {% endmacro %}
32
```

red bubble

menu container opens if user clicks bell (.menu)

dynamically builds list items by passing into another macro / fn

line separating recent notifications from settings

dynamic full URL (points to user's main notification inbox page with a referrer = ui tag so known that user clicked from header dropdown rather than direct link)

① Django (backend building)
- notifications popups generated server-side

② Translate (JavaScript engine building) DOM: structural blueprint, JS Engine: reads/executes code
- notifications generated as JSON from server which is then rendered into the browser

Translate Side (UserNotification.tsx)

- render single user notification item in notifications menu)

layouts

① suggestion
② comment
③ other

- display date/time)

either) ① human-readable full date (older notifs.)
② relative time (recent dates)

doesn't use hooks or side-effects, renders markup based on incoming props

Stateless component

pulled via parameter destructing from notification prop

React functional component, arrow fn returns JSX (concise body form)

```
48  );
49  };
50
51  const Suggestion = ({ date, date_iso, description }: Props['notification']) => (
52    <div className='item-content'>
53      <span
54        className='description'
55        // We can safely use description as it is generated by the code.
56        dangerouslySetInnerHTML={{ __html: description.content }}
57      />
58
59      <DateDisplay date={date} date_iso={date_iso} />
60    </div>
61  );
62
63  const Comment = ({
```

resolves to Notification property type declared in Props

insert HTML markup into description content into DOM, (recap earlier understanding of what DOM is)

trusts string & bypasses React's normal escaping (trusts HTML string is safe) since generated by code

XSS risk (cross-site scripting)

date string

ISO timestamp

decides if need to render full localized date (7 days) or use React Time Ago for recent notifications

Hence

- same notification can appear in 2 different popups but with 2 different technologies

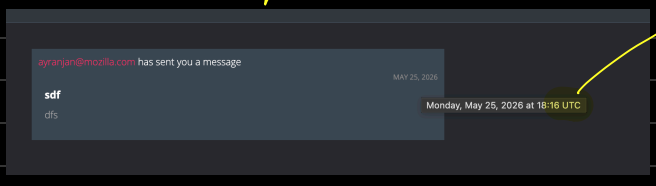
---

Tooltip

- hover mouse over timestamp & browser shows hover box with more detail

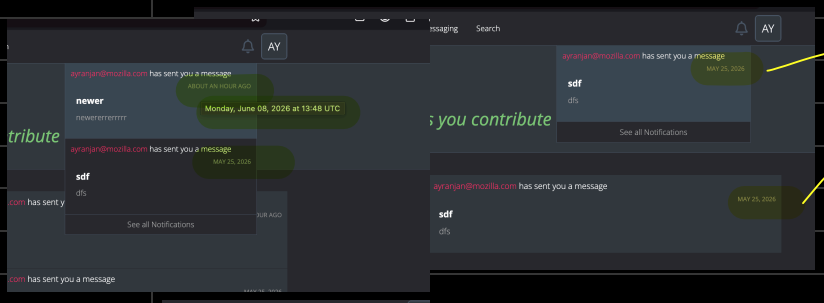
e.g.) visible text: "2 days ago"
hovering: exact time & date

- hover box is tooltip



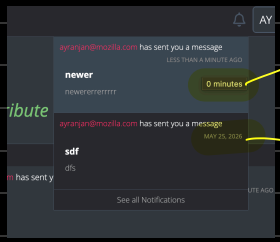
design choice to show UTC upon hover instead of localizing the timezone to that of the given user (after fix)

- this is the title="..." attribute on an element



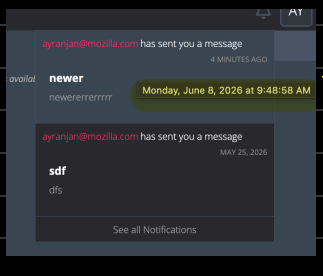
new notification popup does not show the date upon mouse hover (bug?) (before fix)

[React Side]



does show date/time (upon hover)

doesn't show date/time (upon hover)



discrepancy with Django Notification Popup view since does not show 0 minutes but instead shows exact date/time

Other Discrepancies

- older & recent notifications disagree
  - 2 days ago
  - absolute date
- React side doesn't even show notification dooblop ignored.

Every timestamp to show the exact same date/time

Server user.py

- single source of truth, server computes canonical source of string once & reuse everywhere

```
        "verb": notification.verb,
-       "date": notification.timestamp.strftime("%b %d, %Y %H:%M"),
+       "date": format_datetime(notification.timestamp, "full"),
        "date_iso": notification.timestamp.isoformat(),
```

date field sent to React & Django template calls same helper

python helper that produces "Day, Month ##, YYYY at HH:MM UTC"

```
381
382 @property
383 def serialized_notifications(self):
384     """Serialized list of notifications to display in the notifications menu."""
385+    # Local import to avoid a circular import (helpers -> simple_preview -> base.models).
386+    from pontoon.base.templatetags.helpers import format_datetime
387+
388     unread_count = self.notifications.unread().count()
389     count = settings.NOTIFICATIONS_MAX_COUNT
```

Python workaround for helpers need simple_preview needs base.models

↳ this is the file doing the import so would cause Python to crash on startup

Django Template (/notifications_menu.html)

```
33 {% macro notification_time(notification) %}
34
35+  {% set exact_time = notification.timestamp|format_datetime("full") %}
36   {% if notification|is_old_notification %}
-      <time class="old-notification" datetime="{{ notification.timestamp }}">
37+    <time
38+      class="old-notification"
39+      datetime="{{ notification.timestamp }}"
40+      title="{{ exact_time }}"
41+    >
42       {{ notification.timestamp|format_datetime("date") }}
43     </time>
44   {% else %}
-      <time class="timeago" datetime="{{ notification.timestamp }}"
45+    <time
46+      class="timeago"
47+      datetime="{{ notification.timestamp }}"
48+      title="{{ exact_time }}"
49       >{{ notification.timesince() }}</time
50     >
51   {% endif %}
```

reusing same helper as used before

compute exact time once and pass as title on both old and recent timestamp tags

definition to be reused

both cases repaired for using single source of truth

React (/UserNotifications.tsx)

```
18 18
19 19 const DateDisplay: React.FC<DateDisplayProps> = ({ date, date_iso }) => {
20    const formattedDate: string = new Date(date).toLocaleDateString('en-US', {
20+   const parsed = new Date(date_iso);
21+   const isOld: boolean =
22+     parsed.getTime() < Date.now() - 7 * 24 * 60 * 60 * 1000;
23+
24+   if (isOld) {
25+     const formattedDate: string = parsed.toLocaleDateString('en-US', {
21 26       day: 'numeric',
22 27       month: 'long',
23 28       year: 'numeric',
29+      timeZone: 'UTC',
24 30     });
25-    const isOld: boolean =
26-      new Date(date_iso).getTime() <
27-      new Date().getTime() - 7 * 24 * 60 * 60 * 1000;
28 31    return (
29-     <>
30-       {isOld ? (
31 32        <div className='date-display'>
32-           <time dateTime={date_iso} title={`${date} UTC`}>
33+         <time dateTime={date_iso} title={date}>
33 34          {formattedDate}
34 35        </time>
35 36      </div>
36-       ) : (
37+     );
38+   }
39+
40+   return (
41+     // formatVerboseDate overrides ReactTimeAgo's own tooltip so it shows the
42+     // exact date and time, matching the old-notification and Django tooltips.
37 43      <ReactTimeAgo
38 44        className='timeago'
39-           date={new Date(date_iso)}
40-           title={`${date} UTC`}
45+       date={parsed}
46+       formatVerboseDate={() => date}
41 47      />
42-       )}
43-     </>
44 48  );
45 49 };
46 50
```

since new longer format contains at comma so

parse new date since is a JavaScript issue in general so needs fixing this way.

parse date_iso, NOT date

date_iso) standardised machine format that new Date() passes reliably

* fixing "at" does not parse date reliably

```
18
19  const DateDisplay: React.FC<DateDisplayProps> = ({ date, date_iso }) => {
    const formattedDate: string = new Date(date).toLocaleDateString('en-US', {
20+  const parsed = new Date(date_iso);
21+  const isOld: boolean =
22+    parsed.getTime() < Date.now() - 7 * 24 * 60 * 60 * 1000;
```

returns HTML (timestamp of each notification)

days ago) { exact date/time in the hover tooltip

date)

vs.

```
13
14  interface DateDisplayProps {
15    date: string;
16    date_iso: string;
17  }
```

{ receives one props object that pulls 2 fields by name

```
const parsed = new Date(date_iso);
const isOld: boolean =
  parsed.getTime() < Date.now() - 7 * 24 * 60 * 60 * 1000;

if (isOld) {
  const formattedDate: string = parsed.toLocaleDateString('en-US', {
    day: 'numeric',
    month: 'long',
    year: 'numeric',
    timeZone: 'UTC',
  });
```

timestamp as number, milliseconds since Jan 1 1970

7 days in milliseconds, compared to time now, turns on Old's boolean if is true.

Branch 1) Old Notifications (Abs. Date)

```
23+
24+  if (isOld) {
25+    const formattedDate: string = parsed.toLocaleDateString('en-US', {
26      day: 'numeric',
27      month: 'long',
28      year: 'numeric',
29+     timeZone: 'UTC',
30    });
    const isOld: boolean =
```

- formats using readable US conventions

forces time to be deterministic, does not change with timezone

JSX) returns <time> element wrapped in a

<time> for a moment in time

```
28      year: 'numeric',
29+     timeZone: 'UTC',
30    });
    const isOld: boolean =
      new Date(date_iso).getTime() <
      new Date().getTime() - 7 * 24 * 60 * 60 * 1000;
31    return (
      <>
        {isOld ? (
32        <div className='date-display'>
            <time dateTime={date_iso} title={`${date} UTC`}>
33+         <time dateTime={date_iso} title={date}>
34            {formattedDate}
35          </time>
36        </div>
        ) : (
37+   );
38+ }
```

insert JS value, machine-readable attribute for accessibility

tooltip (hovering shows exact date string)

(appending UTC forces double UTC UTC in front-end)

visible text between tags "January 31, 2019"

Branch 2) Recent Notifications (Relative Phase)

```
36      ) : (
37+   );
38+ }
39+
40+ return (
41+   // formatVerboseDate overrides ReactTimeAgo's own tooltip so it shows the
42+   // exact date and time, matching the old-notification and Django tooltips.
37  43    <ReactTimeAgo
38  44      className='timeago'
39      -       date={new Date(date_iso)}
40      -       title={`${date} UTC`}
45+       date={parsed}
46+       formatVerboseDate={() => date}
41  47    />
42      -   )}
43      - </>
44  48  );
45  49  };
46  50
```

renders auto-updating relative time & re-renders over time

moment it begins counting from

! React Time Ago generates own tooltip, ignores plain title prop

∴ old code's "title=..." didn't nothing for recent notifications

FIX) FormatVerboseDate

arrow function that produces tooltip text ignores unpack, always returns server date

Propagation

```
44  48  );
45  49 };
46  50
47  51 const Suggestion = ({ date, date_iso, description }: Props['notification']) => (
48  52   <div className='item-content'>
49  53     <span
50  54       className='description'
51  55       // We can safely use description as it is generated by the code.
52  56       dangerouslySetInnerHTML={{ __html: description.content }}
53  57     />
54  58
55  59     <DateDisplay date={date} date_iso={date_iso} />
56  60   </div>
57  61 );
58  62
59  63 const Comment = ({
```

passes date & date_iso down

fixing DateDisplay once fixes it on every notification type

Tests

test_user.py)

```
181 181      "created_time=20260524044-20260524044"
182 182    )
183 183
184+
185+ @pytest.mark.django_db
186+ def test_serialized_notifications_date_format(user_a, project_a):
187+     """
188+     The serialized "date" carries the exact date and time shown in the
189+     timestamp tooltip, matching format_datetime("full") used by the Django
190+     notifications menu.
191+     """
192+     from pontoon.base.templatetags.helpers import format_datetime
193+
194+     notify.send(
195+         sender=project_a,
196+         recipient=user_a,
197+         verb="has reviewed suggestions",
198+     )
199+
200+     notification = Notification.objects.get(recipient=user_a)
201+     serialized = user_a.serialized_notifications["notifications"][0]
202+     assert serialized["date"] == format_datetime(notification.timestamp, "full")
203+     assert serialized["date_iso"] == notification.timestamp.isoformat()
204+
184 205
185 206 @pytest.mark.django_db
186 207 def test_serialized_notifications_new_string_without_created_time(user_a, project_a):
187 208     """
```

backend) produces the string

verify the serialized date is equivalent to the canonical full format production

* doesn't hardcode the literal

User Notifications Menu.test.jsx)

```
1  31      unread: false,
2  32      description: 'description',
3  33      verb: 'verb',
4          date: 'Jan 31, 2000 10:20',
   34+     date: 'Thursday, January 31, 2019 at 10:20 UTC',
5  35      date_iso: '2019-01-31T10:20:00+00:00',
6  36      actor: {
7  37        anchor: 'actor_anchor',
```

* not necessarily a test, updated fixture string (date value) to stay consistent w/ new backend format

User Notification.test.jsx)

```
82  84      expect(wrapper.find('.message')).toHaveLength(0);
83  85      expect(wrapper.find('.verb').text()).toBe('is Other');
84  86    });
85  87
    88+   it('shows the exact date and time in the timestamp tooltip', () => {
    89+     const notification = {
    90+       ...notificationBase,
    91+       description: { content: 'Unreviewed suggestions' },
    92+     };
    93+     const wrapper = mount(<UserNotification notification={notification} />);
    94+
    95+     const time = wrapper.find('time');
    96+     expect(time.prop('title')).toBe(notificationBase.date);
    97+     expect(time.prop('dateTime')).toBe(notificationBase.date_iso);
    98+     expect(time.text()).toBe('January 31, 2019');
    99+   });
   100+
   101+   it('shows the exact date and time in a recent notification tooltip', () => {
   102+     timeAgoSpy.mockClear();
   103+     const notification = {
   104+       ...notificationBase,
   105+       description: { content: 'Unreviewed suggestions' },
   106+       date_iso: new Date(Date.now() - 60 * 1000).toISOString(),
   107+     };
   108+     mount(<UserNotification notification={notification} />);
   109+
   110+     const props = timeAgoSpy.mock.calls[0][0];
   111+     expect(props.formatVerboseDate()).toBe(notificationBase.date);
   112+   });
86  113
87  114   it('shows comment notification with deleted actor', () => {
```

frontend) displays the string

import a spy function that necessitates hoisting.

prove that tooltip override is wired correctly

∴ Spy reveals UI rendered output ☆