Technologies

References & Guides

Feedback

Sign in

Search

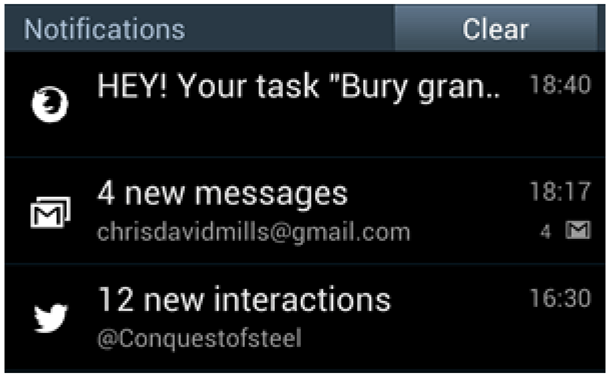
Using the Notifications API

Note: This feature is available in Web Workers.

Secure context
This feature is available only in secure contexts (HTTPS), in some or all supporting browsers.

The Notifications API lets a web page or app send notifications that are displayed outside the page at the system level; this lets web apps send information to a user even if the application is idle or in the background. This article looks at the basics of using this API in your own apps.

Typically, system notifications refer to the operating system's standard notification mechanism: think for example of how a typical desktop system or mobile device broadcasts notifications.


Notifications
Clear
HEY! Your task "Bury gran.. 18:40
4 new messages 18:17
chrisdavidmills@gmail.com 4
12 new interactions 16:30
@Conquestofsteel

The system notification system will vary of course by platform and browser, but this is ok, and the Notifications API is written to be general enough for compatibility with most system notification systems.

Examples

One of the most obvious use cases for web notifications is a web-based mail or IRC application that needs to notify the user when a new message is received, even if the user is doing something else with another application. Many real examples of this now exist, such as Slack.

We've written a real world demo to give more of an idea of how web notifications can be used:

- **To-do list**: This is a simple to-do list app that stores data locally using IndexedDB and notifies users when tasks are due using system notifications. Download the To-do list code, or view the app running live.

Requesting permission

Before an app can send a notification, the user must grant the application the right to do so. This is a common requirement when an API tries to interact with something outside a web page — at least once, the user needs to specifically grant that application permission to present notifications, thereby letting the user control which apps/sites are allowed to display notifications.

Checking current permission status

You can check to see if you already have permission by checking the value of the `Notification.permission` read only property. It can have one of three possible values:

`default`
The user hasn't been asked for permission yet, so notifications won't be displayed.

`granted`
The user has granted permission to display notifications, after having been asked previously.

`denied`
The user has explicitly declined permission to show notifications.

Getting permission

If permission to display notifications hasn't been granted yet, the application needs to use the `Notification.requestPermission()` method to request this from the user. In its simplest form, we just include the following:

```
Notification.requestPermission().then(function(result) {
  console.log(result);
});
```

This uses the promise-version of the method, as supported in recent implementations (Firefox 47, for example.) If you want to support older versions, you might have to use the older callback version, which looks like this:

```
Notification.requestPermission();
```

The callback version optionally accepts a callback function that is called once the user has responded to the request to display permissions (as seen in the second `else ... if` block below.) Commonly, you'll ask for permission to display notifications when your app is first initialized, and before trying to instantiate any. If you wanted to be really thorough, you could use a construct like the following (see To-do List Notifications):

```
function notifyMe() {
  // Let's check if the browser supports notifications
  if (!("Notification" in window)) {
    alert("This browser does not support system notifications");
  }

  // Let's check whether notification permissions have already been granted
  else if (Notification.permission === "granted") {
    // If it's okay let's create a notification
    var notification = new Notification("Hi there!");
  }

  // Otherwise, we need to ask the user for permission
  else if (Notification.permission !== 'denied') {
    Notification.requestPermission(function (permission) {
      // If the user accepts, let's create a notification
      if (permission === "granted") {
        var notification = new Notification("Hi there!");
      }
    });
  }

  // Finally, if the user has denied notifications and you
  // want to be respectful there is no need to bother them any more.
}
```

Note: Before version 37, Chrome doesn't let you call `Notification.requestPermission()` in the `load` event handler (see issue 274284).

In Chrome 62+ and Firefox 67+ you cannot request notifications at all unless the site is a secure context (i.e. HTTPS).

Creating a notification

Creating a notification is easy; just use the `Notification` constructor. This constructor expects a title to display within the notification and some options to enhance the notification such as an `icon` or a text `body`.

For example, in the to-do-list example we use the following snippet to create a notification when required (found inside the `createNotification()` function):

```
var img = '/to-do-notifications/img/icon-128.png';
var text = 'HEY! Your task "' + title + '" is now overdue.';
var notification = new Notification('To do list', { body: text, icon: img });
```

Closing notifications

Firefox and Safari close notifications automatically after a few moments (around four seconds). This may also happen at the operating system level. Some browsers don't however, such as Chrome. To make sure that the notifications close in all browsers, you can call the `Notification.close` function inside a `setTimeout()` function to close the notification after 4 seconds. Also note the use of `bind()` to make sure the `close()` call is associated with the notification.

```
setTimeout(notification.close.bind(notification), 4000);
```

Note: When you receive a "close" event, there is no guarantee that it's the user who closed the notification. This is in line with the specification, which states: "When a notification is closed, either by the underlying notifications platform or by the user, the close steps for it must be run."

Notification events

There are four events that are triggered on the `Notification` instance:

`click`
: Triggered when the user clicks on the notification.

`close`
: Triggered once the notification is closed.

`error`
: Triggered if something goes wrong with the notification; this is usually because the notification couldn't be displayed for some reason.

`show`
: Triggered when the notification is displayed to the user.

These events can be tracked using the `onclick`, `onclose`, `onerror`, and `onshow` handlers. Because `Notification` also inherits from `EventTarget`, it's possible to use the `addEventListener()` method on it.

Replacing existing notifications

It is usually undesirable for a user to receive a lot of notifications in a short space of time — for example, what if a messenger application notified a user for each incoming message, and they were being sent a lot? To avoid spamming the user with too many notifications, it's possible to modify the pending notifications queue, replacing single or multiple pending notifications with a new one.

To do this, it's possible to add a tag to any new notification. If a notification already has the same tag and has not been displayed yet, the new notification replaces that previous notification. If the notification with the same tag has already been displayed, the previous notification is closed and the new one is displayed.

Tag example

Assume the following basic HTML:

```html
<button>Notify me!</button>
```

It's possible to handle multiple notifications this way:

```js
window.addEventListener('load', function () {
  // At first, let's check if we have permission for notification
  // If not, let's ask for it
  if (window.Notification && Notification.permission !== "granted") {
    Notification.requestPermission(function (status) {
      if (Notification.permission !== status) {
        Notification.permission = status;
      }
    });
  }

  var button = document.getElementsByTagName('button')[0];

  button.addEventListener('click', function () {
    // If the user agreed to get notified
    // Let's try to send ten notifications
    if (window.Notification && Notification.permission === "granted") {
      var i = 0;
      // Using an interval cause some browsers (including Firefox) are blocking notifications if there ar
      var interval = window.setInterval(function () {
        // Thanks to the tag, we should only see the "Hi! 9" notification
        var n = new Notification("Hi! " + i, {tag: 'soManyNotification'});
        if (i++ == 9) {
          window.clearInterval(interval);
        }
      }, 200);
    }

    // If the user hasn't told if he wants to be notified or not
    // Note: because of Chrome, we are not sure the permission property
    // is set, therefore it's unsafe to check for the "default" value.
    else if (window.Notification && Notification.permission !== "denied") {
      Notification.requestPermission(function (status) {
        // If the user said okay
        if (status === "granted") {
          var i = 0;
          // Using an interval cause some browsers (including Firefox) are blocking notifications if ther
          var interval = window.setInterval(function () {
            // Thanks to the tag, we should only see the "Hi! 9" notification
            var n = new Notification("Hi! " + i, {tag: 'soManyNotification'});
            if (i++ == 9) {
              window.clearInterval(interval);
            }
          }, 200);
        }

        // Otherwise, we can fallback to a regular modal alert
        else {
          alert("Hi!");
        }
      });
    }

    // If the user refuses to get notified
    else {
      // We can fallback to a regular modal alert
      alert("Hi!");
    }
  });
});
```

See the live result below:

Notify me!

Specifications

Specification	Status	Comment
Notifications API	LS Living Standard	Living standard

Browser compatibility

Update compatibility data on GitHub

`Notification`

Chrome	22
Edge	14
Firefox	22
IE	No
Opera	25
Safari	6
WebView Android	No
Chrome Android	Yes
Edge Mobile	?

Firefox Android	22
Opera Android	Yes
Safari iOS	No
Samsung Internet Android	Yes

Available in workers

Chrome	45
Edge	Yes
Firefox	41
IE	No
Opera	32
Safari	?
WebView Android	No
Chrome Android	45
Edge Mobile	Yes
Firefox Android	41
Opera Android	32
Safari iOS	No
Samsung Internet Android	?

Secure context required

Chrome	62
Edge	?
Firefox	?
IE	No
Opera	49
Safari	?
WebView Android	No
Chrome Android	62
Edge Mobile	?
Firefox Android	?
Opera Android	46
Safari iOS	No
Samsung Internet Android	?

`Notification()` constructor

Chrome	22
Edge	Yes
Firefox	22
IE	No
Opera	25

Safari	6
WebView Android	No
Chrome Android	Yes
Edge Mobile	?
Firefox Android	22
Opera Android	Yes
Safari iOS	No
Samsung Internet Android	?

`actions`

Chrome	53
Edge	18
Firefox	No
IE	No
Opera	39
Safari	?
WebView Android	No
Chrome Android	53
Edge Mobile	No
Firefox Android	No

Opera Android	41
Safari iOS	No
Samsung Internet Android	Yes

badge

Chrome	53
Edge	18
Firefox	No
IE	No
Opera	39
Safari	?
WebView Android	No
Chrome Android	53
Edge Mobile	No
Firefox Android	No
Opera Android	41
Safari iOS	No
Samsung Internet Android	Yes

body

Chrome	Yes
	14

Firefox	Yes
IE	No
Opera	Yes
Safari	Yes
WebView Android	No
Chrome Android	Yes
Edge Mobile	?
Firefox Android	Yes
Opera Android	Yes
Safari iOS	No
Samsung Internet Android	Yes

`data`

Chrome	Yes
Edge	16
Firefox	Yes
IE	No
Opera	Yes
Safari	?
WebView Android	No
Chrome Android	Yes

Edge Mobile	?
Firefox Android	Yes
Opera Android	Yes
Safari iOS	No
Samsung Internet Android	Yes

`dir`

Chrome	Yes
Edge	14
Firefox	Yes
IE	No
Opera	Yes
Safari	Yes
WebView Android	No
Chrome Android	Yes
Edge Mobile	?
Firefox Android	Yes
Opera Android	Yes
Safari iOS	No
Samsung Internet Android	Yes

`icon`

Chrome	22
Edge	14
Firefox	22
IE	No
Opera	25
Safari	No
WebView Android	No
Chrome Android	Yes
Edge Mobile	?
Firefox Android	22
Opera Android	Yes
Safari iOS	No
Samsung Internet Android	Yes

`image`

Chrome	53
Edge	18
Firefox	No
IE	No

Opera	40
Safari	?
WebView Android	No
Chrome Android	53
Edge Mobile	?
Firefox Android	No
Opera Android	41
Safari iOS	No
Samsung Internet Android	Yes

`lang`

Chrome	Yes
Edge	14
Firefox	Yes
IE	No
Opera	Yes
Safari	Yes
WebView Android	No
Chrome Android	Yes
Edge Mobile	?
Firefox Android	Yes

Opera Android	Yes
Safari iOS	No
Samsung Internet Android	Yes

`maxActions`

Chrome	Yes
Edge	18
Firefox	No
IE	No
Opera	Yes
Safari	?
WebView Android	No
Chrome Android	Yes
Edge Mobile	?
Firefox Android	No
Opera Android	Yes
Safari iOS	No
Samsung Internet Android	Yes

`onclick`

Chrome	Yes
	14

Firefox	No
IE	No
Opera	Yes
Safari	Yes
WebView Android	No
Chrome Android	Yes
Edge Mobile	?
Firefox Android	No
Opera Android	Yes
Safari iOS	No
Samsung Internet Android	Yes

`onclose`

Chrome	Yes
Edge	14
Firefox	Yes
IE	No
Opera	Yes
Safari	Yes
WebView Android	No
Chrome Android	Yes

Edge Mobile	?
Firefox Android	Yes
Opera Android	Yes
Safari iOS	No
Samsung Internet Android	Yes

`onerror`

Chrome	Yes
Edge	14
Firefox	No
IE	No
Opera	Yes
Safari	Yes
WebView Android	No
Chrome Android	Yes
Edge Mobile	?
Firefox Android	No
Opera Android	Yes
Safari iOS	No
Samsung Internet Android	Yes

`onshow`

Chrome	Yes
Edge	14
Firefox	Yes
IE	No
Opera	Yes
Safari	Yes
WebView Android	No
Chrome Android	Yes
Edge Mobile	?
Firefox Android	Yes
Opera Android	Yes
Safari iOS	No
Samsung Internet Android	Yes

`permission`

Chrome	Yes
Edge	14
Firefox	Yes
IE	No
Opera	Yes

Safari	Yes
WebView Android	No
Chrome Android	Yes
Edge Mobile	?
Firefox Android	Yes
Opera Android	Yes
Safari iOS	No
Samsung Internet Android	Yes

`renotify`

Chrome	50
Edge	No
Firefox	No
IE	No
Opera	37
Safari	No
WebView Android	No
Chrome Android	50
Edge Mobile	No
Firefox Android	No
Opera Android	37

Safari iOS	No
Samsung Internet Android	Yes

`requireInteraction`

Chrome	Yes
Edge	17
Firefox	No
IE	No
Opera	Yes
Safari	?
WebView Android	No
Chrome Android	Yes
Edge Mobile	No
Firefox Android	No
Opera Android	Yes
Safari iOS	No
Samsung Internet Android	Yes

`silent`

Chrome	43
Edge	17
	No

IE	No
Opera	30
Safari	No
WebView Android	No
Chrome Android	43
Edge Mobile	No
Firefox Android	No
Opera Android	30
Safari iOS	No
Samsung Internet Android	Yes

`tag`

Chrome	Yes
Edge	14
Firefox	Yes
IE	No
Opera	Yes
Safari	Yes
WebView Android	No
Chrome Android	Yes
Edge Mobile	?

Firefox Android	Yes
Opera Android	Yes
Safari iOS	No
Samsung Internet Android	Yes

`timestamp`

Chrome	Yes
Edge	17
Firefox	No
IE	No
Opera	Yes
Safari	?
WebView Android	No
Chrome Android	Yes
Edge Mobile	No
Firefox Android	No
Opera Android	Yes
Safari iOS	No
Samsung Internet Android	Yes

`title`

	Yes

Edge	14
Firefox	No
IE	No
Opera	Yes
Safari	Yes
WebView Android	No
Chrome Android	Yes
Edge Mobile	?
Firefox Android	No
Opera Android	Yes
Safari iOS	No
Samsung Internet Android	Yes

`vibrate`

Chrome	53
Edge	No
Firefox	No
IE	No
Opera	39
Safari	?
WebView Android	No

Chrome Android	53
Edge Mobile	No
Firefox Android	No
Opera Android	41
Safari iOS	No
Samsung Internet Android	Yes

`close`

Chrome	Yes
Edge	14
Firefox	Yes
IE	No
Opera	Yes
Safari	Yes
WebView Android	No
Chrome Android	Yes
Edge Mobile	?
Firefox Android	Yes
Opera Android	Yes
Safari iOS	No
Samsung Internet Android	Yes

`requestPermission`

Chrome	46
Edge	14
Firefox	47
IE	No
Opera	40
Safari	Yes
WebView Android	No
Chrome Android	46
Edge Mobile	?
Firefox Android	Yes
Opera Android	41
Safari iOS	No
Samsung Internet Android	Yes

Full support

No support

Compatibility unknown

See implementation notes.

-x- Requires a vendor prefix or different name for use.

See also

- User notifications reference
- Notifying users via the Notification and Vibration APIs
- `Notification`