# Modules

Starting from zone.js v0.8.9, you can choose which web API modules you want to patch as to reduce overhead introduced by the patching of these modules. For example, the below samples show how to disable some modules. You just need to define a few global variables before loading zone.js.

```
<script>
  __Zone_disable_Error = true; // Zone will not patch Error
  __Zone_disable_on_property = true; // Zone will not patch onProperty such as button.oncl
  __Zone_disable_geolocation = true; // Zone will not patch geolocation API
  __Zone_disable_toString = true; // Zone will not patch Function.prototype.toString
  __Zone_disable_blocking = true; // Zone will not patch alert/prompt/confirm
  __Zone_disable_PromiseRejectionEvent = true; // Zone will not patch PromiseRejectionEven
</script>
<script src="../bundles/zone.umd.js"></script>
```

Below is the full list of currently supported modules.

- Common

| Module Name | Behavior with zone.js patch | How to disable |
|---|---|---|
| Error | stack frames will have the Zone's name information, (By default, Error patch will not be loaded by zone.js) | __Zone_disable_Error = true |
| toString | Function.toString will be patched to return native version of toString | __Zone_disable_toString = true |
| ZoneAwarePromise | Promise.then will be patched as Zone aware MicroTask | __Zone_disable_ZoneAwarePromise = true |
| bluebird | Bluebird will use Zone.scheduleMicroTask as async scheduler. (By default, bluebird patch will not be loaded by zone.js) | __Zone_disable_bluebird = true |

- Browser

| Module Name | Behavior with zone.js patch | How to disable |
| --- | --- | --- |
| on_property | target.onProp will become zone aware target.addEventListener(prop) | __Zone_disable_on_property = true |
| timers | setTimeout/setInterval/setImmediate will be patched as Zone MacroTask | __Zone_disable_timers = true |
| requestAnimationFrame | requestAnimationFrame will be patched as Zone MacroTask | __Zone_disable_requestAnimationFrame = true |
| blocking | alert/prompt/confirm will be patched as Zone.run | __Zone_disable_blocking = true |
| EventTarget | target.addEventListener will be patched as Zone aware EventTask | __Zone_disable_EventTarget = true |
| MutationObserver | MutationObserver will be patched as Zone aware operation | __Zone_disable_MutationObserver = true |
| IntersectionObserver | Intersection will be patched as Zone aware operation | __Zone_disable_IntersectionObserver = true |
| FileReader | FileReader will be patched as Zone aware operation | __Zone_disable_FileReader = true |
| canvas | HTMLCanvasElement.toBlob will be patched as Zone aware operation | __Zone_disable_canvas = true |
| IE BrowserTools check | in IE, browser tool will not use zone patched eventListener | __Zone_disable_IE_check = true |
| CrossContext check | in webdriver, enable check event listener is cross context | __Zone_enable_cross_context_check = true |
| XHR | XMLHttpRequest will be patched as Zone aware MacroTask | __Zone_disable_XHR = true |
| geolocation | navigator.geolocation's prototype will be patched as Zone.run | __Zone_disable_geolocation = true |
| PromiseRejectionEvent | PromiseRejectEvent will fire when ZoneAwarePromise has unhandled error | __Zone_disable_PromiseRejectionEvent = true |

| Module Name | Behavior with zone.js patch | How to disable |
|---|---|---|
| mediaQuery | mediaQuery addListener API will be patched as Zone aware EventTask. (By default, mediaQuery patch will not be loaded by zone.js) | __Zone_disable_mediaQuery = true |
| notification | notification onProperties API will be patched as Zone aware EventTask. (By default, notification patch will not be loaded by zone.js) | __Zone_disable_notification = true |
| MessagePort | MessagePort onProperties APIs will be patched as Zone aware EventTask. (By default, MessagePort patch will not be loaded by zone.js) | __Zone_disable_MessagePort = true |

- NodeJS

| Module Name | Behavior with zone.js patch | How to disable |
|---|---|---|
| node_timers | NodeJS patch timer | __Zone_disable_node_timers = true |
| fs | NodeJS patch fs function as macroTask | __Zone_disable_fs = true |
| EventEmitter | NodeJS patch EventEmitter as Zone aware EventTask | __Zone_disable_EventEmitter = true |
| nextTick | NodeJS patch process.nextTick as microTask | __Zone_disable_nextTick = true |
| handleUnhandledPromiseRejection | NodeJS handle unhandledPromiseRejection from ZoneAwarePromise | __Zone_disable_handleUnhandledPromiseRejection = true |
| crypto | NodeJS patch crypto function as macroTask | __Zone_disable_crypto = true |

- Test Framework

| Module Name | Behavior with zone.js patch | How to disable |
|---|---|---|
| Jasmine | Jasmine APIs patch | \_\_Zone\_disable\_jasmine = true |
| Mocha | Mocha APIs patch | \_\_Zone\_disable\_mocha = true |

- on_property

You can also disable specific on_properties by setting `__Zone_ignore_on_properties` as follows: for example, if you want to disable `window.onmessage` and `HTMLElement.prototype.onclick` from zone.js patching, you can do like this.

```
<script>
   __Zone_ignore_on_properties = [
     {
       target: window,
       ignoreProperties: ['message']
     }, {
       target: HTMLElement.prototype,
       ignoreProperties: ['click']
     }
   ];
</script>
<script src="../bundles/zone.umd.js"></script>
```

- Error

By default, `zone.js/plugins/zone-error` will not be loaded for performance concern. This package will provide following functionality.

1. Error inherit: handle `extend Error` issue. `class MyError extends Error {}` `const myError = new MyError();` `console.log('is MyError instanceof Error', (myError instanceof Error));`

   without `zone-error` patch, the example above will output `false`, with the patch, the reuslt will be `true`.

2. ZoneJsInternalStackFrames: remove zone.js stack from `stackTrace`, and add `zone` information. Without this patch, a lot of `zone.js` invocation stack will be shown in stack frames.

```
  at zone.run (polyfill.bundle.js: 3424)
  at zoneDelegate.invokeTask (polyfill.bundle.js: 3424)
  at zoneDelegate.runTask (polyfill.bundle.js: 3424)
  at zone.drainMicroTaskQueue (polyfill.bundle.js: 3424)
  at a.b.c (vendor.bundle.js: 12345 <angular>)
  at d.e.f (main.bundle.js: 23456)
```

```
 with this patch, those zone frames will be removed,
```

```
 and the zone information `<angular>/<root>` will be added

 ```
   at a.b.c (vendor.bundle.js: 12345 <angular>)
   at d.e.f (main.bundle.js: 23456 <root>)
 ```
```

The second feature will slow down the `Error` performance, so `zone.js` provide a flag to let you be able to control the behavior. The flag is `__Zone_Error_ZoneJsInternalStackFrames_policy`. And the available options is:

```
1. default: this is the default one, if you load `zone.js/plugins/zone-error` without
 setting the flag, `default` will be used, and `ZoneJsInternalStackFrames` will be available
 when `new Error()`, you can get a `error.stack`  which is `zone stack free`. But this
 will slow down `new Error()` a little bit.

2. disable: this will disable `ZoneJsInternalStackFrames` feature, and if you load
 `zone.js/plugins/zone-error`, you will only get a `wrapped Error` which can handle
  `Error inherit` issue.

3. lazy: this is a feature to let you be able to get `ZoneJsInternalStackFrames` feature,
 but not impact performance. But as a trade off, you can't get the `zone free stack
 frames` by access `error.stack`. You can only get it by access `error.zoneAwareStack`.
```

- Angular(2+)

Angular uses zone.js to manage async operations and decide when to perform change detection. Thus, in Angular, the following APIs should be patched, otherwise Angular may not work as expected.

1. ZoneAwarePromise
2. timer
3. on_property
4. EventTarget
5. XHR