# Browsers and devices

Learn about the browsers and devices, from modern to old, that are supported by Bootstrap, including known quirks and bugs for each.

## Supported browsers

Bootstrap supports the **latest, stable releases** of all major browsers and platforms. On Windows, **we support Internet Explorer 10-11 / Microsoft Edge**.

Alternative browsers which use the latest version of WebKit, Blink, or Gecko, whether directly or via the platform's web view API, are not explicitly supported. However, Bootstrap should (in most cases) display and function correctly in these browsers as well. More specific support information is provided below.



### Mobile devices

Generally speaking, Bootstrap supports the latest versions of each major platform's default browsers. Note that proxy browsers (such as Opera Mini, Opera Mobile's Turbo mode, UC Browser Mini, Amazon Silk) are not supported.

| | Chrome | Firefox | Safari | Android Browser & WebView | Microsoft Edge |
|---|---|---|---|---|---|
| **Android** | Supported | Supported | N/A | Android v5.0+ supported | Supported |
| **iOS** | Supported | Supported | Supported | N/A | Supported |
| **Windows 10 Mobile** | N/A | N/A | N/A | N/A | Supported |

### Desktop browsers

Similarly, the latest versions of most desktop browsers are supported.

| | Chrome | Firefox | Internet Explorer | Microsoft Edge | Opera | Safari |
|---|---|---|---|---|---|---|
| **Mac** | Supported | Supported | N/A | N/A | Supported | Supported |
| **Windows** | Supported | Supported | Supported, IE10+ | Supported | Supported | Not supported |

For Firefox, in addition to the latest normal stable release, we also support the latest Extended Support Release (ESR) version of Firefox.

Unofficially, Bootstrap should look and behave well enough in Chromium and Chrome for Linux, Firefox for Linux, and Internet Explorer 9, though they are not officially supported.

For a list of some of the browser bugs that Bootstrap has to grapple with, see our Wall of browser bugs.

## Internet Explorer

Internet Explorer 10+ is supported; IE9 and down is not. Please be aware that some CSS3 properties and HTML5 elements are not fully supported in IE10, or require prefixed properties for full functionality. Visit Can I use... for details on browser support of CSS3 and HTML5 features.

**If you require IE8-9 support, use Bootstrap 3.** It's the most stable version of our code and is still supported by our team for critical bugfixes and documentation changes. However, no new features will be added to it.

# Modals and dropdowns on mobile

## Overflow and scrolling

Support for `overflow: hidden;` on the `<body>` element is quite limited in iOS and Android. To that end, when you scroll past the top or bottom of a modal in either of those devices' browsers, the `<body>` content will begin to scroll. See Chrome bug #175502 (fixed in Chrome v40) and WebKit bug #153852.

## iOS text fields and scrolling

As of iOS 9.2, while a modal is open, if the initial touch of a scroll gesture is within the boundary of a textual `<input>` or a `<textarea>`, the `<body>` content underneath the modal will be scrolled instead of the modal itself. See WebKit bug #153856.

## Navbar Dropdowns

The `.dropdown-backdrop` element isn't used on iOS in the nav because of the complexity of z-indexing. Thus, to close dropdowns in navbars, you must directly click the dropdown element (or any other element which will fire a click event in iOS).

# Browser zooming

Page zooming inevitably presents rendering artifacts in some components, both in Bootstrap and the rest of the web. Depending on the issue, we may be able to fix it (search first and then open an issue if need be). However, we tend to ignore these as they often have no direct solution other than hacky workarounds.

# Sticky `:hover`/`:focus` on iOS

While `:hover` isn't possible on most touch devices, iOS emulates this behavior, resulting in "sticky" hover styles that persist after tapping one element. These hover styles are only removed when users tap another element. This behavior is considered largely undesirable and appears to not be an issue on Android or Windows devices.

Throughout our v4 alpha and beta releases, we included incomplete and commented out code for opting into a media query shim that would disable hover styles in touch device browsers that emulate hovering. This work was never fully completed or enabled, but to avoid complete breakage, we've opted to deprecate this shim and keep the mixins as shortcuts for the pseudo-classes.

# Printing

Even in some modern browsers, printing can be quirky.

As of Safari v8.0, use of the fixed-width `.container` class can cause Safari to use an unusually small font size when printing. See issue #14868 and WebKit bug #138192 for more details. One potential workaround is the following CSS:

Copy

```
@media print {
  .container {
    width: auto;
  }
}
```

# Android stock browser

Out of the box, Android 4.1 (and even some newer releases apparently) ship with the Browser app as the default web browser of choice (as opposed to Chrome). Unfortunately, the Browser app has lots of bugs and inconsistencies with CSS in general.

## Select menu

On `<select>` elements, the Android stock browser will not display the side controls if there is a `border-radius` and/or `border` applied. (See this StackOverflow question for details.) Use the snippet of code below to remove the offending CSS and render the `<select>` as an unstyled element on the Android stock browser. The user agent sniffing avoids interference with Chrome, Safari, and Mozilla browsers.

```
<script>
$(function () {
  var nua = navigator.userAgent
  var isAndroid = (nua.indexOf('Mozilla/5.0') > -1 && nua.indexOf('Android ') > -1 &&
nua.indexOf('AppleWebKit') > -1 && nua.indexOf('Chrome') === -1)
  if (isAndroid) {
    $('select.form-control').removeClass('form-control').css('width', '100%')
  }
})
</script>
```

Want to see an example? Check out this JS Bin demo.

# Validators

In order to provide the best possible experience to old and buggy browsers, Bootstrap uses CSS browser hacks in several places to target special CSS to certain browser versions in order to work around bugs in the browsers themselves. These hacks understandably cause CSS validators to complain that they are invalid. In a couple places, we also use bleeding-edge CSS features that aren't yet fully standardized, but these are used purely for progressive enhancement.

These validation warnings don't matter in practice since the non-hacky portion of our CSS does fully validate and the hacky portions don't interfere with the proper functioning of the non-hacky portion, hence why we deliberately ignore these particular warnings.

Our HTML docs likewise have some trivial and inconsequential HTML validation warnings due to our inclusion of a workaround for a certain Firefox bug.