**Difference between Bootstrap 3 and 4**

Bootstrap 44 is a major rewrite of almost the entire project. The most notable changes are summarized immediately below, followed by more specific class and behavioral changes to relevant components.

1. Browser support
   1. Dropped ie8, ie9 and iOS 6 support. V4 is now only IE10+ and iOS7+. For sites needing either of those, use v3.
2. Global changes
   1. Flexbox is enabled by default. In general this means a move away from floats and more across our components.
   2. Switched from LESS to SASS for our source CSS file.
   3. Switched from “px” to “rem” as our primary CSS unit, though pixels are still used for media queries and grid behavior as viewports are not affected by type size.
   4. Global font-size increased from “14px” to “16px”.
   5. Added new grid tier for smaller device at “576px” and below (our new “xs” tier).
   6. Replaced the separate optional theme with configurable options via SCSS variables (e.g. “$enable-gradients:true”)
3. Grid system
   1. Added support for flexbox in the grid mixins and predefined classes.
   2. As part of flexbox, included support for vertical and horizontal alignment classes.
   3. Overhauled grid mixins to merge “make-col-span” into “make-col” for a singular mixin.
   4. Added a new “sm” grid tier below “768px” for mew granular control. We now have **xs, sm, md, lg** and **xl.** This also means every tier has been bumped up one level (so .col-md-6 in v3 is now .col-lg-6 in v4).
   5. Changed grid system media query breakpoints and container widths to account for new grid tier and ensure columns are evenly divisible by 12 at their max width.
   6. Grid breakpoints and container widths are now handled via SASS maps ($grid-breakpoints and $container-max-widths) instead of a handful of separate variables. These replace the “#screen-\*” variables entirely and allow you to fully customize the grid tiers.
   7. Media queries have also changed. Instead of repeating our media query declarations with the same value each time, we now have “@include media-breakpoint-up/down/only”. Now, instead of writing @media (min-width: @screen-sm-min) {…}, you can write @include media-breakpoint-up(sm) {…}.
4. Components
   1. Dropped panels, thumbnails, and wells for a new all-encompassing component, cards.
   2. Dropped the Glyphicons icon font. If you need icons, some options are:
      1. The upstream version of **Glyphicons**
      2. **Octicons**
      3. **Font awesome**
   3. Dropped the affix jQuery plugin. We recommend using a position: sticky polyfill instead. [See the HTML5 Please entry](http://html5please.com/#sticky) for details and specific polyfill recommendations.
      1. If you were using affix to apply additional, non-position styles, the polyfills might not support your use case. One option for such uses is the third-party scrollpos-styler library.
   4. Dropped the pager component as it was essentially slightly customized buttons.
   5. Refactored nearly all components to use more un-nested classes instead of children selectors.
5. MISC
   1. Non-responsive usage of Bootstrap is no longer supported.
   2. Dropped the online customizer in favor of more extensive setup documentation and customized builds.
6. By component
   1. This list highlights key changes by component between v3.x.x and v4.0.0.
7. Reboot
   1. New to Bootstrap 4 is the reboot, a new StyleSheet that builds on normalize with our own somewhat opinionated reset styles. Selectors appearing in this file only use elements-there are no classes here. This isolates our reset styles from our component styles for a more modular approach. some of the most important resets this includes are the box-sizing: border-box change, moving from em to rem units on many elements, link styles, and many form element resets.
8. Typography
   1. Moved all .text- utilities to the \_utilities.scss file.
   2. Dropped .page-header as, aside from the border, all its styles can be applied via utilities.
   3. .dl-horizontal has been dropped. Instead, use .row on <dl> and use grid column classes (or mixins) on its <dt> and <dd> children.
   4. Custom <blockquote> styling has moved to classes -- .blockquote and the .blockquote-reserve modifier.
   5. .list-inline now requires that its children list items have the new .list-inline-item class applied to them.
9. Images
   1. Renamed .img-responsive to .img-fluid.
   2. Renamed .img-rounded to .rounded.
   3. Renamed .img-circle to .rounded-circle.
10. Tables
    1. Nearly all instances of the > selector have been removed, meaning nested tables will now automatically inherit styles from their parents. This greatly simplifies our selectors and potential customizations.
    2. Responsive tables no longer require a wrapping element. Instead, just put the .table-responsive right on the <table>
    3. Renamed .table-condensed to .table-sm for consistency.
    4. Added a new .table-inverse option.
    5. Added table header modifiers: .thead-default and .thead-inverse.
    6. Renamed contextual classes to have a .table- -prefix. Hence .active, .success, .warning, .danger and .table-info to .table-active, .table-success, .table-warning, .table-danger and .table-info.
11. Forms
    1. Moved element resets to the \_reboot.scss file.
    2. Renamed .control-label to .form-control-label.
    3. Renamed .input-lg and .input-sm to .form-control-lg and .form-control-sm, respectively.
    4. Dropped .form-group-\* classes for simplicity’s sake. Use .form-control-\* classes instead now.
    5. Dropped .help-block and replaced it with .form-text for block-level help text. For inline help text and other flexible options, use utility classes like .text-muted.
    6. Horizontal forms overhauled:
       1. Dropped the .form-horizontal class requirement.
       2. .form-group no longer applies styles from the .row via mixin, so .row is now required for horizontal grid layouts (e.g. <div class=”form-group row”>).
       3. Added new .form-control-label class to vertically center labels with .form-controls.
       4. Added custom forms support (form checkboxes, radios, selects and file inputs).
12. Explain how SASS comments are different from regular CSS?
    1. Syntax for comments in regular CSS starts wit /\* comments…. \*/, while in SASS there are two type of comment, the single line comments “//” and the multiline CSS comments with “/\*\*/”.
13. Does SASS supports inline comments?
    1. Single line comments “//” will be removed by the .SCSS pre-processor and won’t appear in your .CSS file.
    2. While the comment “\*/” are valid CSS, and will be preserved between the translation from .SCSS to your .CSS file.
14. How interpolation is used in SASS?
    1. In SASS, you can define an element in a variable and interpolate it inside the SASS code. It is useful when you keep your modules in separate files.
15. Explain when can you use the %placeholders in SASS?
    1. %placeholders in SASS is useful when you want to write styles that were meant to be extended, but you don’t want the base styles to be seen in output CSS styles.
16. Is it possible to nest variables within variables in SASS?
    1. Interpolation of variables names is not possible currently in SASS. However, you may use interpolation of placeholders.
17. What are SASS cons and pros?
    1. Pros:
       1. SASS is easy to learn especially for them who has a background of python, ruby or Coffescript and place using functions, writing mixins
       2. CSS can be easily converted to SASS
       3. Throughout the project, you don’t have to repeat similar CSS statements using @extend attribute
       4. It allows to define variables that are usable throughout the entire project
       5. It keeps your responsive project more organized
    2. Cons:
       1. White space sensitive
       2. No inline rules

References

<http://career.guru99.com/top-25-bootstrap-interview-questions/>

<http://career.guru99.com/top-17-sass-interview-questions/>

<http://career.guru99.com/>