Session 8. Responsive web. Bootstrap

Programació Multiplataforma i Distribuïda

Grau d'informàtica. EPSEVG

Octubre 2023

Jordi Esteve jesteve@cs.upc.edu

Bootstrap. Introduction

Bootstrap is a CSS+JavaScript framework directed at responsive design, mobile-first front-end web development. It offers:

- **Layout** design: Grid system, wrapping containers, responsive classes, …
- **Uniform content** across different browsers (reboot).
- Lot of **Components**: Alerts, Carousel, Collapse, Pagination, Progress, Spinners, ...

Bootstrap was created in 2011 by two developers at Twitter as a framework to encourage consistency across internal tools. History:

- 2011 **v1** First common design patterns.
- 2012 **v2** Adds new components. Add responsive web design.
- 2013 **v3** Components redesigned to use flat design. Mobile first approach.
- 2018 **v4** Major rewrite of the code. Replacing Less with Sass. …
- 2020 **v5** Added 6th type of device (xxl). Uses VainillaJS instead of Jquery.

Bootstrap. Introduction

Bootstrap is easy to use: You must add some classes to common HTML5 elements to get a specific behavior. E.g. To get a link displayed as a red button (danger) spanning the full width of its parent (block):

```
<a class="btn btn-danger btn-block" role="button">Text</a>
```

Bootstrap grid system in v5 distinguishes six different types of devices like cell phones (portrait and landscape), tablets, laptops, desktops or big screen desktops:

Bootstrap 5 Grid System	**Extra small** <576px	**Small** ≥576px	**Medium** ≥768px	**Large** ≥992px	**Extra large** ≥1200px	**XX large** ≥1400px
Max conta. width	None (auto)	540px	720px	960px	1140px	1400px
Ideal for	Mobile (portrait)	Mobile (landscape)	Tablets	Laptops	Laptops & Desktops	Big screen Desktops
Class prefix	`.col-`	`.col-sm-`	`.col-md-`	`.col-lg-`	`.col-xl-`	`.col-xxl-`

Bootstrap. How to use it

We must load a CSS including the definitions of Bootstrap styles and Bootstrap JavaScript library (Bootstrap bundle includes Popper.js, and Bootstrap).

To ensure proper rendering and touch zooming for all devices, a **responsive viewport meta tag** in <head> section must be added.

```
<html>
<head>
  <title>Title of the page</title>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css" rel="stylesheet"
crossorigin="anonymous">
</head>

<body>
  <p>My page</p>
  <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/js/bootstrap.bundle.min.js"
crossorigin="anonymous"></script>
</body>
</html>
```

Bootstrap. Layout and Grid system

The grid systems has **12 columns** and must be included in a **container** (a *div* with class *container*, *container-{breakpoint}* o *container-fluid*; the *{breakpoint}* can be *sm*, *md*, *lg*, *xl* o *xxl*).

	Extra small <576px	**Small** ≥576px	**Medium** ≥768px	**Large** ≥992px	**X-Large** ≥1200px	**XX-Large** ≥1400px
`.container`	100%	540px	720px	960px	1140px	1320px
`.container-sm`	100%	540px	720px	960px	1140px	1320px
`.container-md`	100%	100%	720px	960px	1140px	1320px
`.container-lg`	100%	100%	100%	960px	1140px	1320px
`.container-xl`	100%	100%	100%	100%	1140px	1320px
`.container-xxl`	100%	100%	100%	100%	100%	1320px
`.container-fluid`	100%	100%	100%	100%	100%	100%

Bootstrap. Layout and Grid system

The columns will be merged if less than 12 columns are defined.

Giving a number to a column we can define its width; the rest of the columns without a number share the remaining space.

.col	.col	.col	.col	.col	.col	.col	.col	.col	.col	.col	.col
.col		.col		.col		.col		.col		.col	
.col						.col					
.col-6						.col-6					
.col-8										.col-4	
.col				.col-6						.col	

Change `.col` to `.col-sm`, `.col-md`, `.col-lg`, `.col-xl`, `.col-xxl` to apply the grid starting in this device size. We can use a combination of classes belonging to different device sizes in the same cell. Next page shows the code to obtain the previous table:

Bootstrap. Layout and Grid system

```
<div class="container-fluid">
 <div class="row">
  <div class="col">.col</div>  <div class="col">.col</div>  <div class="col">.col</div>
  <div class="col">.col</div>  <div class="col">.col</div>  <div class="col">.col</div>
  <div class="col">.col</div>  <div class="col">.col</div>  <div class="col">.col</div>
  <div class="col">.col</div>  <div class="col">.col</div>  <div class="col">.col</div>
 </div>
 <div class="row">
  <div class="col">.col</div>  <div class="col">.col</div>  <div class="col">.col</div>
  <div class="col">.col</div>  <div class="col">.col</div>  <div class="col">.col</div>
 </div>
 <div class="row">
  <div class="col">.col</div>  <div class="col">.col</div>
 </div>
 <div class="row">
  <div class="col-6">.col-6</div>  <div class="col-6">.col-6</div>
 </div>
 <div class="row">
  <div class="col-8">.col-8</div>  <div class="col-4">.col-4</div>
 </div>
 <div class="row">
  <div class="col">.col</div>  <div class="col-6">.col-6</div>  <div class="col">.col</div>
 </div>
</div>
```

Bootstrap. Content. Reboot

The HTML elements without style are not rendered in the same way. E.g. For a `<button>` Chrome applies padding: 2px 6px 3px; and Firefox applies padding: 0 8px;

In Bootstrap, the style of HTML5 elements are defined to look the same across different browsers (the styles are "rebooted"). This affects to paragraphs, headers, lists, tables, forms, buttons, …

In addition, Bootstrap adds new elements and classes to define the style of:

- **Typography**: Text alignment, text transform, text color and background color, abbreviations, block quotes, …
- **Code**: Inline code, code blocks, user input, sample output.
- **Images**: Responsive images, image alignment, thumbnails, …
- **Tables**: Responsive tables, head options, zebra-striping rows, table borders, …
- **Figures**: Linking an image with a caption.

Bootstrap. Components

Bootstrap offers many new components. Some of them are:

- **Alerts**: Contextual feedback message. They can be inline.
- **Buttons** and **Button groups**: Buttons toolbars. Sizing, nesting, vertical group, ...
- **Carousel**: Carousel of images or slides of text.
- **Collapse**: To toggle the visibility of content.
- **Dropdown buttons**: For displaying lists of links.
- **Modal**: To add dialogs for lightboxes, user notifications, …
- **Navs** and **Navbar**: Navigation components (like the upper navigation menu).
- **Pagination**: To indicate a series of related content exists across multiple pages.
- **Popovers**: Contextual messages/helps that pop on clicking.
- **Progress** and **Spinners**: Indicate the loading state of a component or page.
- **Tooltips**: Contextual messages/helps that pop on over.

Bootstrap. External resources

Bootstrap has an excellent documentation with lots of examples.

The terminology of Flexbox, used in Bootstrap, explained with simple images.

A Bootstrap 5 tutorial.

Another Bootstrap 5 tutorial.

Bootstrap. Task list exercise (I)

Our multiple task list application will be responsive and will improve its appearance using Bootstrap. You can follow the next steps, but you can also make your own improvements in its design and usability.

1. Add a new task list to have a total of 3 task lists: Task list, Home task list and University task list.
2. Add the Bootstrap CSS and JS libraries, and the responsive viewport meta tag.
3. Put appropriate colors (*btn-success*, *btn-danger*, *btn-secondary*, …) to the different buttons (*New task*, *Reset tasks*, *Active tasks*, *Create*, *Update*, *Go back*).
4. Group the *New task*, *Reset tasks*, *Active tasks* buttons inside a *btn-group*.
5. Put the three task lists in three columns inside a *container-fluid*. Give the same width to the three columns, for medium, large or extra large devices. So in small and extra small devices the content of the three columns will be displayed inside a single column.

Bootstrap. Task list exercise (II)

6. Convert the title of each task list to a button that allows collapse its lists (*data-bs-toggle="collapse"*). This button could span the full width of its parent (put inside a div with *d-grid gap-2*). Each list could be a *card*, with the title button inside the *card-header* and the rest inside the *card-body* that can be collapsed.

7. Save and read a cookie that remembers which task lists are collapsed and which are not when the web app starts.

8. Add some JavaScript code that collapses automatically the three task lists when the window width is small or extra small (less than 768px), so when the three columns will be converted in a single column in small and extra small devices, they also are collapsed automatically.

You can see a demo video in Atenea showing the new appearance and behavior using Bootstrap.