# MULTI-PAGE RESPONSIVE WEBSITE


## ABSTRACT

Today's extra smile devices i.e. mobile phones, smartphones; small devices i.e. tablets, mini laptops; medium devices like laptops, desktops; and large devices like new age TVs, LEDs, and projector screens capable of accessing websites have changed the approach towards website design and development. These high-tech devices and gadgets have placed the web designers and developers to serious issues of user experience and accessibility. By the technique of Responsive Web Design by which developers can now produce device agnostic websites which respond, with respect to their layout and content presentation, on all kinds of device resolutions without requiring any special modifications. This means a single Responsive Website is now sufficient to serve all device sizes. These Website is a Multipage Responsive Design i.e. it is portable for all kind of devices with different sizes or resolution .


## INTRODUCTION

Responsive Design enables companies to make a good first impression no matter which device is used. The "reactive" layout adapts dynamically to the available display sizes of smartphones, tablets and the like without the need for an additional mobile version of the report. When using a mobile device to access the report, images, graphics and texts adjust automatically to the size of the display, the navigation menu is replaced with a separate mobile menu and tables are prepared in a manner that can be easily viewed on smartphones. The optimisation process is based on web technologies such as HTML5, CSS3 and JavaScript.

Web Browsers are enhancing on a very fast pace. Every day we confront a large number of new devices categorized as extra small, small, medium and large devices. These devices, apart from size, accompany a diverse range of configurations and physical characteristics like we have devices with different resolutions, pixel density, touch screen controls, track-pads and keyboards or a combination of these. Technology may be advancing at a rapid pace but it doesn't make any sense that one should keep exchanging the old devices at that pace. All new inventions need to be backward-compatible with existing technologies thus allowing people to continue using older implementations on new devices. There might be situations when a newly released device or browser is cancelled a few weeks later after its release. Hence, what is true today may not be true tomorrow and resultantly all this leads the users in a situation of chaos.

But this is only one side of picture. On the other hand, chaos leads to innovation and creativity. Quite often sitting in waiting area of an airport or a bus terminal you would see people using different size of mobile devices. At the beginning of every project, it has to be determined whether responsive design is to be adapted for smartphones primarily, or for all kinds of mobile device, which would e.g. also include devices such as the iPad. Many of our customers decide against the latter since the desktop version of the report can usually be displayed without significant limitations on most tablets. Besides, special reporting apps for iPads are frequently offered which cannot only be used offline, but

provide a further distribution channel via iTunes. This approach goes beyond “standard” responsive design since the desktop version is no longer used as starting point for the web layout. Instead, it was somewhat to the contrary: the layout is based on the smallest possible screen (e.g. smartphone) leading to the largest version (e.g. desktop version), and covering various other formats such as the iPad-mini by defining various breakpoints.

Consequently, if we continue with Adaptive Web Design approach today we would certainly end up with our websites no longer accessible on new and upcoming devices. It might be possible that a company completely denies to cater to the needs of mobile device users and adopts the fixed-width layout approach. A website built using fixed-width layout has width and height of components in fixed pixel units. No matter what screen resolution a visitor has, the same amount of width and height will be presented to that particular user. Whenever such a website is viewed on mobile devices, the user will certainly end up in pinching, zooming and scrolling in both vertical and horizontal directions, and may face slower page load times. A better solution to the above problem would be to develop a single website using principles of Responsive Web Design to make it available on devices of all resolutions that exist today and is also future-ready. Responsive Web Design is a set of tools and techniques used together to build a single rich user-interface website which is equally accessible on devices of various sizes and resolutions, and to user of all network bandwidths no matter high or low.

A web designer “Ethan Marcotte” first coined the term Responsive Web Design (RWD) in May, 2010 in his article “Responsive Web Design”. His approach to design websites using techniques of RWD instigated revolution in the discipline of web design. He simply used three existing CSS3 tools:

i)media queries, ii) fluid (flexible) grids, and iii) flexible images

to create a responsive website (http://goo.gl/I1T8mL) that displayed on devices with different resolutions correctly. Subsequently people started exploring the RWD techniques and leading content publishers rebuilt their websites using this new stunning model of RWD. Many popular websites publishing articles, blogs, and news started discussing the RWD and they declared 2013 as the year of Responsive Web Design.

## FUNDAMENTAL PRINCIPLES OF RWD

Responsive Web Design is a newfound technique but it has evolved rapidly. By simply applying CSS rules inside media queries is not enough to cope up with the responsiveness.

### 1 . Fluid (Flexible) Grids

A grid is simply a collection of rows and columns. A grid is used to organize layout in a web [2]. In a grid, row consists of one, two or more columns. Each row, no more than maximum number of columns, could span one, two or more columns. Content is then placed inside columns. Grids help to add order and creativity in the website by organizing the content in a uniform way so that whenever new content is added it looks consistent with the overall Presentation of the website. Below figure shows a, fixed-width, 960px grid of two columns. The width of context is 960px. The top row (of 880px) of the container spans two columns followed by another row of two columns of 640px and 220px. The space between the rows and columns is the margin property referred to as gutter width in RWD. The easy first step in creating a fluid grid layout is to first make a blueprint of the desired fixed-width layout. In a fluid grid, width of all elements starting from parent container, then header, and all the way to a footer section is stated in relative units e.g. percentage, rem, em, etc. An “em” is a relative CSS unit for expressing length values relative to font-size of immediate parent of an element. By default, 1 em is equal to 16px. A value of 2 em means 32px. Em is used to express the values of inner level elements inside columns. Similarly, ‘rem’ is used to express length values relative to root (html) font-size. Value of 2 rem means 2 times the font-size of the root element. To design a fluid grid layout, we also use values in percentages. Formula to calculate proportions of the design in percentages is as below:

(target / context) x 100 = desired value % (result rounded to 0 decimal place)

Target is the width of an element in fixed-width pixels. Context is known as the width of the environment in pixels for which the layout is required. Row is also called a wrapper because it contains columns inside it which then contain our elements. A wrapper in meaning here is very much similar to a toffee wrapper which wraps around the toffee. Once we have calculated the wrapper value, then for onward calculations this wrapper value will be used as context value. In figure-2, context is 960px and wrapper width is 880px. Following relative values are calculated for this context based on the above formula:

– Container Width: (960px / 960px) x 100 will correspond to 100% container width

– First Row (spans two columns): (880px / 960px) x 100 = 92% (this values will also be used for wrapper)

– Second Row Left Column: (640px / 880px) x 100 = 73%


## 2. Flexible (Scalable) Media

Media (images and videos) content is a source to provide pleasing and richer user experience. But using excessive media could result in slow load times for a website. When it comes to smaller devices the situation needs more attention as these devices have little resources in terms of network bandwidth and browser capabilities. To maintain aspect ratio of the media on various device densities is another challenge. Like an image would appear crisp and sharp on a particular device size but it would become blurry and lost its pixels when opened on a large display or on a device of same size but greater density. The website performance greatly depends on how beautifully the developer has programmed to display images and videos. "Global Dots" study shows that a greater number of users would leave a web page if it takes more than 6 seconds to load. There are three types of media which need to be tackled with great amount of care in responsive paradigm and that is i) video, ii) images (foreground), and iii) background images.

To make flexible/ scalable media we need to write rules in CSS, HTML and/or JavaScript. There are a lot of pre-coded packages available on internet ready for usage so in this monograph we'll not cover the coding part rather this study focuses on key underlying concepts. The key idea to make responsive media is that we do not use fixed-width height for images and videos instead this value is set to auto or not provided at all. We first create a container with proportion width size to hold our media and it is then sized to some percentage of the container. Container of media inherits its width from the outer container and the media inherits its width with respect to that of its container. For example, we want to make image xyz.jpg responsive. We'll first create a container of some 50% width. We then place our xyz.jpg image inside that container and set the width to say 100% so that at whatever device size the image is opened the image will resize itself to fit in the container and container will get its width from device width in some percentage accordingly. Density means the ratio of CSS pixels to device pixels. If we program a 100px by 100px image to be viewed on a device with 1:1 ration then it will work fine but if that same image is opened on a device with pixel density

of 2 (i.e. 1:2 ratio) it would mean that the image will become 200px wide and 200px high, and consequently it will lost its resolution. Conversely, if a high density image is viewed on a device with low density then the image will become blurry. Moreover, we also need to take care of grids, columns, and rows here. Each and every content part either an image or a text block is specified in a column or more with respect to context i.e. device size.

## 3.Media Queries

A great user experience benefit of responsive design sits in lesser amount of scrolling and resizing. That is a user of the responsive website will not be pinching or zooming, and scrolling in horizontal direction. Thus only vertical scrolling is required to surf a responsive website irrespective of whether the device in use is a smartphone, tablet, desktop computer or a smart-TV. Here comes in the need of media queries which contain CSS style rules to apply to the values of width, height, depth, color, resolution, margins, padding, etc. when a specific viewport size is encountered.

Viewport. This is simply the visible area of the browser i.e. the browser width in pixels. The concept seems to be all simple apparently but there are actually two types of pixels: device or hardware pixels and CSS pixels. Device pixels are the unit of design that form the basis of everything we make on a device [8]. A device screen which is 1280px wide means that a maximum of two elements of 640px width or four elements of 320px width can fit side-by-side at once on the screen. CSS pixels have nothing to do with the screen. CSS pixels means everything visible inside the browser window. On normal devices, 1 CSS pixel corresponds to 1 device pixel but on devices with high pixel density 1 CSS pixel may correspond to more hardware pixels. Pixel Density is the number of pixels per inch (PPI) that can be displayed in a physical display [9]. This means that CSS pixels may not correlate with the device pixels. CSS pixels change with the zoom in or out of a page. Following line of code tells the device to equal the device's layout viewport with the visual viewport and set the initial zoom level to 100% :

<meta name="viewport" content="width=device-width, initial-scale=1">

Apart from width and initial-scale directives a number of other useful directives are available to help developers maintain the layout of their web pages up to their requirements.

Structure of a Media Query

A media query is comprised of four basic components :

– Logical Keywords like and, not, only, or, etc to create more complex queries.

– Media types i.e. the type of device to target. Most common media types are screen (computer and mobile displays), print (printers), and tv (television type devices),

– Media expression to test a feature and evaluates to either true or false.

– CSS Rules to style the content accordingly when the expression evaluates to true

As an example of media query consider the following code which will implement the listed CSS rules whenever the width of the screen is less than 480px:

```
@media screen and (max−width: 480 px ){
        #l o g o {
                width: 95%;
            }
        #topMiniNav {
            padding−top: 5px;
            margin−left: 0;
            margin−right: 0;
        }
}
```

Similarly, we can implement other media queries with different expressions and constraints to meet our needs. In practice, we need to write more complex media queries combining logical keywords, media features, complex expressions, and detailed list of CSS rules.

As of now we have studied all the nitty-gritty of RWD with respect to device size, resolution and pixel density. To make a website responsive we need to take into account various attributes of individual elements, design a powerful fluid grid system, work with harder media queries and take care of device pixel ratio.

Structuring the Content

The previous three ingredients of RWD techniques: media queries, fluid grids and adaptive/ responsive media, are used to layout the content properly for variety of device sizes. Dynamic content is concerned with making the content itself, not the layout, responsive or adapt to certain contexts of the user behaviour. Content is basically the reason behind building a particular website. There are two most important things about the content. First one is that it should come first and secondly it should be well-structured because structured content is the key to produce dynamic content. A nicely structured content would be able to behave and transform in many new ways and in different contexts. We take into account three key concepts while structuring the content.

These are discussed in brief below.

i. Document Structure:-

The document structure is made up by using logical blocks and defining their hierarchies. By using structural tags, we divide the web pages into logical blocks so that tags become descriptive of the type of the content they contain. The word document is being used here because we treat a web page as a document. A standard document contains header, footer, chapters, sections, sub-sections, and other components that divide the document into logical parts. So in order to structure our web pages we need to use semantic tags provided by HTML5 in accordance with guidelines of W3C for dynamic web applications [13]. Figure-3 below shows an illustration of document structure built using semantic elements: The content

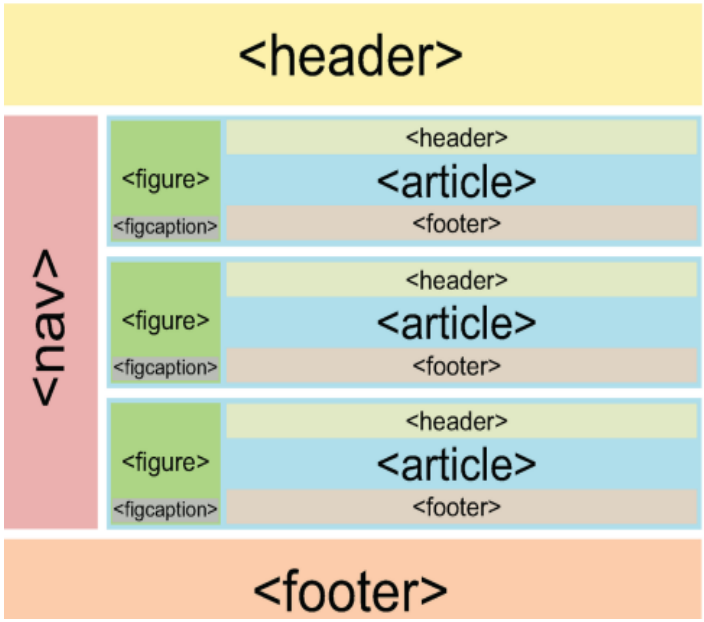
hierarchy is also a part of well-planned document structure. Hierarchy makes it possible to assign priorities to the content, making the content portable, and allows tracking the logical blocks within the document.

ii. Metadata:

Metadata is the information about the content and it may exist at varying levels in the document hierarchy. It is the data that provides context for a content chunk. For example, categorizing some products as 'new arrivals' would mean that this chunk of content contains information about newly released products. Content should be stored in a structured format with hierarchies and meaningful metadata so that it could be requested and utilized wherever and whenever needed. A page level metadata might include author name, date of creation/ update/ review/ last visited, physical location information of a particular user, etc.

iii. Supporting Content.:

This is the supplementary type of content used to enhance user experience and does not affect the core content and its hierarchy if not found on a particular page. It is necessary to keep the supporting content separate from the core content otherwise it will cause problems when updating the site design. Some examples of supporting content are media assets, call-outs or pull quotes, citations and references, banners, advertisements, etc.




Structure of an HTML5 Web Page

Two Techniques to Make Content Responsive.

Two most common practical techniques to make the content more responsive are discussed below in brief.

i) **Lazy Loading:-**It is a technique which is used to not load those images and videos that are not visible in the browser's visual viewport at a particular time [14]. As the user scrolls down to the media content, it is then loaded as required. It is an intelligent way to lazily load the heavy media content which might never be viewed by the user of mobile devices and hence increases page load time performance. So our intent here is to increase user experience by providing faster page loads dynamically.

ii)**Selective Content Loading (SCL):-** A best example of SCL implementation is Facebook's infinite feed scrolling. As the user keeps scrolling down the feeds page, older feeds are loaded dynamically up to the point the user scrolls. The moment user stops scrolling, more older feeds will never be loaded hence not loading the content which will not be viewed by the user at all. This technique is really useful in conditions when we want to load chunks of content into an already opened web page in a structured way.

## RESPONSIVE WEB DESIGN FRAMEWORKS

In the domain of web, a framework is a set of structured files, folders, and standardized code of HTML, CSS, JavaScript, jQuery, etc. aimed at providing a basic starting point for development of industry standard websites. There are almost more than twenty Responsive Web Frameworks that exist today and all are free to use i.e. open-source, but in this section we will discuss three most popular and widely used RWD frameworks:

i)Bootstrap,

ii)Skeleton, and

iii)Foundation.

These frameworks are built on complex fluid grid systems and are extensively used by a large number of website developers.

### i) Bootstrap

Bootstrap was developed by Mark Otto and Jacob Thorton primarily to be used internally by Twitter in 2011 and later launched as a free software for public use [16]. In its initial release, responsive feature was not present in Bootstrap which was added in version 2.

Bootstrap is the most popular responsive framework for developing mobile first websites based on 12-column fluid grid system. It is packed with extensive ready-made User Interface (UI) components including glyph icons, buttons, alerts, progress bars, drop-down menus, and much more . The power of Bootstrap comes to action quickly because a developer will only need to download the framework package and start designing responsive websites. Bootstrap also provides video scaling component. Bootstrap code can be customized using LESS or SASS. Websites developed using Bootstrap are capable of running on device resolutions starting with width less than 480px up to 1200px.

In a nutshell, Bootstrap is easy to get started, offers a great grid system, provides styling rules for fundamental HTML elements such as headings, lists, tables, buttons, forms, etc., provides extensive UI components, packed with built-in JavaScript plugins to make UI components more interactive, and most importantly supported by extensive documentation.

Following are examples of websites built using Bootstrap:

– Codecademy, Teaching the world how to code, http://www.codecademy.com

– Opendesk, design for open making, https://www.opendesk.cc

– CodrSpace, the blogging platform for coders, http://codrspace.com

### ii)Skeleton

Skeleton is a lightweight CSS responsive framework created by Dave Gamache. Skeleton is the simplest framework as compared to other two discussed in this monograph. It is not overstuffed with heavy styles, extensive UI components, and JavaScript/ jQuery plugins. Skeleton is good for smaller projects in which large number of utilities are not required. Skeleton uses 12-columns fluid grid system with a maximum width of 960px. Skeleton comes with a starter HTML template (index.html), and two lightweight CSS files (normalize.cc and skeleton.css) and that's the reason it is called a lightweight framework because intent of skeleton developer is to provide responsive only components without rich styles and plugins.

Following are examples of websites built using Skeleton:

– GetSkeleton, official website of Skeleton, http://getskeleton.com

– HiveMind, a design agency, http://www.ourhivemind.com

### iii) Foundation

Foundation was built by a team at ZURB, a California based product design agency. Foundation is also an open source project and mobile-first framework. Unlike Bootstrap, Foundation do not have much pre-defined styled UI compo□nents however it has rich designed elements and built-in JavaScript plugins. A developer needs more technical skills to work with Foundation in that it is based on Ruby language and one need to install Ruby in order to start working with Foundation. SASS, SCSS and Ruby together make Foundation as the most advanced responsive frameworks as compared to Bootstrap and Skeleton.

Following are examples of websites built using Foundation:

– HTC, official HTC Shop, UK, http://goo.gl/d9zAGH

– Canadian Olympic Team Official Website, http://olympic.ca

– iStockPhoto, Perfect Digital Stock Photos, http://www.istockphoto.com

## TESTS TO CHECK RESPONSIVENESS OF A WEBSITE

In this section we will see some methods that exist to check if a website is responsive or not.

## GOOGLE MOBILE-FRIENDLY TEST

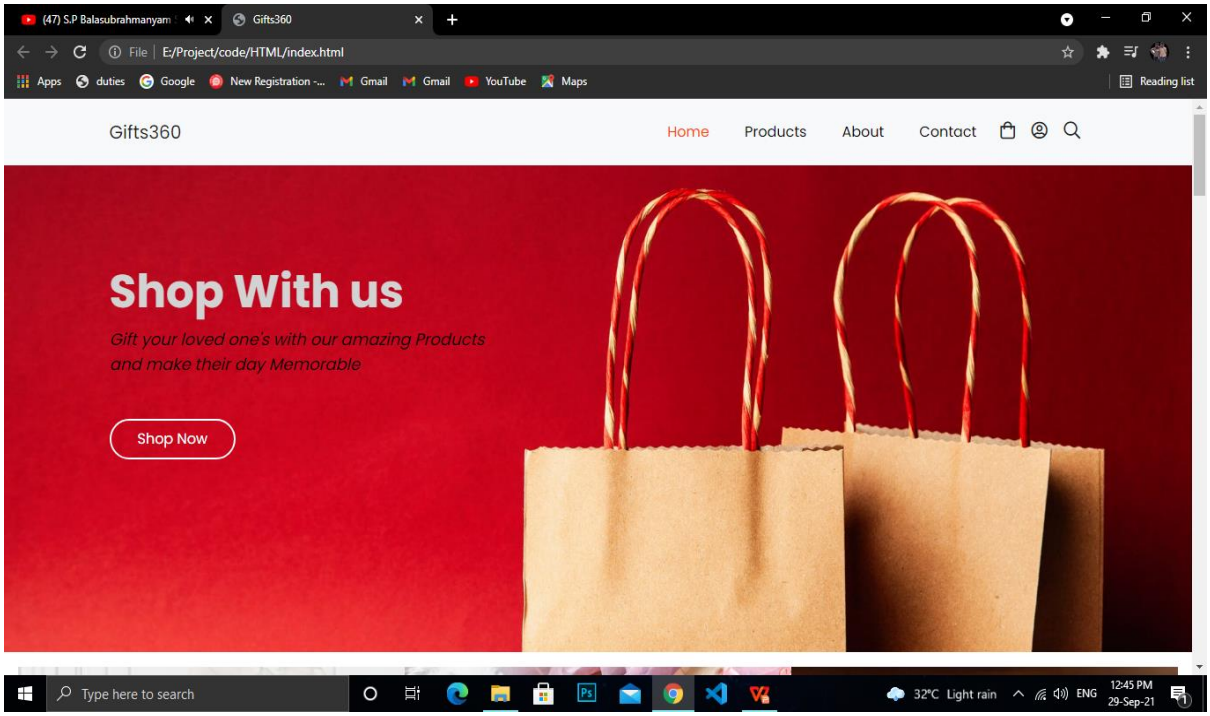
Due to the growing popularity of Responsive Web Design, Google announced mobile-friendly testing tool in the end of year 2014. This has completely changed the ranking of websites in Google's search results. No matter how structured and search engine optimized a website is but if it is not mobile-friendly, it will certainly be displayed below mobile-friendly websites in search results. As per official Google Webmaster blog, a website will only be eligible for "mobile□friendly" tag if it fulfils the following criteria as identified by Google bot . A web page should avoid software which is not supported by mobile devices e.g. Flash, JAVA plugin, etc. ii. It should not use the text which would require zooming to read. iii. A page should not require horizontal scroll or pinching/ zooming. iv. There should be enough space between navigational links so that the targeted link could be easily tapped. The link of Google's mobile-friendly test tool to check an individual web page is https://www.google.com/webmasters/tools/mobile-friendly/ A site's ranking in Google, the most famous search engine, when used on mobile devices now also depends on mobile-friendly test. That's a big reason why companies are getting their websites responsive.

## BROWSER WINDOW RESIZING TRICK

By resizing the browser window, it could be checked that whether a particular website is responsive or not but this type of test is not reliable. For example, if a website uses media queries and device-width instead of width property to determine breakpoints then window resizing test will not work due to the reason that device-width, i.e. desktop/ laptop layout viewport, will remain constant and only visual viewport is changing. However, this test will work for most of the websites.

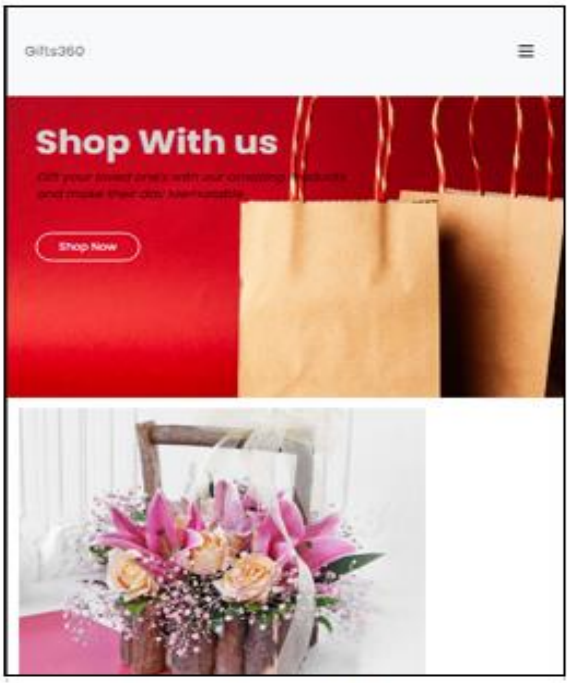
## MOZILLA FIREFOX AND GOOGLE CHROME BUILT-IN RESPONSIVE TESTS

Mozilla Firefox comes with a built-in Responsive Design View mode to help developers test their websites during development process and see if there are any issues associated with the responsiveness. To enable Firefox's built-in responsive test simply press Ctrl + Shift + M on windows or Cmd + Opt+ M on Mac. Same key combination will disable the responsive design mode. It may be required to refresh the website when current size is shifted. Below Figure shows the Responsive Design View test on my Gifts360 website.




**WEB-VIEW**


Mobile -View(360 x 640px )




Ipad- View(768 x 1024 px)

## CONCLUSION:-

It has been and always be challenging to keep your website design up to different device sizes and resolutions. A responsive website, if developed properly, adapts to different device sizes, enhances user experience, is mobile-friendly, and is ready for future devices. This all has been possible due to three basic building blocks of Responsive Web Design (RWD) i.e. media queries, flexible/ scalable media (images and videos), and most importantly fluid/ flexible grids. Heavy websites posed constrains to mobile devices due to lesser capabilities of mobile web browsers as compared to those of desktop browsers. A fourth major building block was then added to RWD namely dynamic content which provides techniques like lazy loading and selective content loading. Dynamic content requires a properly structured semantic web content which itself requires a tedious work of document structuring and developing hierarchy of the content in correlation to the structure. We have ready-made Responsive Frameworks to help developers just take care of their content structuring and hierarchical planning and forgot about the underlying tedious work of RWD principles. Selection of any such framework depends on user needs and requirements. Therefore, it is at the developer's part to come up with the decision on which framework suits best for their website and this is the most important thing because to change the choice of framework in the middle or after the development means to start a new web project. Experienced developers and designers could also design their own responsive framework if required. Moreover, a responsive website requires on the go device agnostic testing during development phase and before actual launch to avoid later complications. Even ready-made frameworks could put the developers in a situation of chaos if handled improperly.