How Responsive Is Your Testing?

To accelerate business digitization, organizations need to ensure a seamless user experience across diverse channels, one that starts with a fresh perspective on application testing that takes into account the proliferating form factors in today’s responsive design world.

Executive Summary

In today’s quickly digitizing world, consumers spend an inordinate amount of time on multiple digital screens, from handy smartphones and tablets through desktop computers and Internet-enabled television. The choice of the screen is mostly driven by the context and availability, and is often sequential in nature.

Though the nexus of most digital interactions is the smartphone, many consumers find it comfortable and efficient to switch between multiple screens when carrying out and/or completing a transaction. Figure 1 depicts Google’s findings on industry-wise sequential screening initiated from mobile devices.¹

Multiscreen World Across Industries

![Diagram showing multiscreen world across industries]

Figure 1

cognizant 20-20 insights | june 2015
Though screen sequence varies with each industry, the need to follow users— not devices — is becoming the rule rather than the exception, guiding consumer-facing companies to a new paradigm of user experience that focuses on screens. Google’s recent move to make Chrome its default browser across screens, as well as Apple’s Continuity initiative as seen on the iOS 8 and OS X Yosemite platforms, reconfirm the importance of the world of multiscreens.

This white paper offers guidance on ways QA organizations can more effectively test new responsive designed applications to ensure a consistency of user experience across form factors and screen sizes.

**Responsive Web Design**

With the multiscreen environment gaining significance, responsive Web design (RWD), based on HTML 5 and JQuery, is gaining popularity. While companies in businesses as varied as information/media, energy/utilities, insurance, manufacturing/logistics and healthcare have already commenced their RWD journeys, leading players in the retail and banking/financial services sectors are earnestly charting plans to migrate their large existing code base to RWD as they embrace the new era of digital experience.

RWD is a simple, single-source-code approach to development that targets multiscreen delivery based on the media query (or screen width). Based on this screen width, the browser content not only customizes itself, but also can be customized end to end such as to show or hide content. Figure 2 illustrates how the app layout can be customized to accommodate different screen sizes. 3

**Layout Based on Media Query**

![Figure 2](image)

Typically, customization happens at breakpoints. Desktops, tablets and smartphones are considered to be the design standard breakpoints (see Figure 3). Through fluidic design concepts, RWD pages can be sized to accommodate any screen size among the breakpoints.

**Sizing Up RWD Breakpoints**

![Figure 3](image)
Impact of RWD on Testing

RWD charts a new dimension in application development and testing. The target application is meant for the desktop, tablet and smartphone, and hence it demands careful identification of test cases and a smarter design approach. Though the application’s content may be the same, the placement within the layout or menu typically varies based on the device’s screen width. As a best practice, it is crucial to tag the test cases and, if need be, the test steps to the breakpoints.

RWD projects are usually executed with a two-tier approach, emphasizing user interface (UI) elements and layout. In this process, the wireframes and HTML design and layout are created first, followed by the integration of the system. This approach mandates that testers should be involved in the earlier phases of development to ensure the correctness of UI as per the defined breakpoints.

During this phase, the screens are validated for the respective pixel matrix of the identified breakpoints. Content delivery is dependent on the media query - not the user agent - so the majority of UI-specific testing can be performed on the desktop browser by resizing. After ironing out the initial UI issues with reference to the breakpoints, it is recommended to verify the screens on the actual smartphone browsers. This is because a smartphone browser does not work the same way as a desktop browser does.

Given the fluidic design of the Web pages, the right selection of mobile devices plays a vital role to identify bugs associated with the logic built in for breakpoints. Ideally, the selection of devices is based on the defined breakpoints, and the market share of the device and mobile OS version. Given the same content is verified on multiple browsers during the UI validation phase, powerful browser utility tools are required to reduce the testing effort.

System Integration Testing Approach

The second phase of testing is when the built HTML pages are integrated to deliver functionalities. Since the source code is similar among various breakpoints, it is recommended that end-to-end functional testing is conducted on the desktop, while regression testing or a day-in-the-life scenario is tested on the other breakpoints. However, the testing approach can be revalidated by carefully identifying the integration failure across breakpoints through appropriate metrics.

In addition to UI and integration testing, the single-code-base nature of RWD projects places performance testing on center stage. Although there are many development guidelines such as the mobile-first approach to make responsive pages lightweight, it is crucial to validate performance KPIs, especially on smartphones. It is recommended that three-staged performance testing of responsive Web pages be carried out on a smartphone to enable the ultimate seamless user experience.

The New Dimension of Automation Testing

Similar to any smartphone or a tablet application, in a responsive design there is a need to perform repeated testing across a wide range of device combinations. In addition to smartphones and tablets, it is also necessary to run the same...
test cases across various combinations including Microsoft Windows, Apple Mac and browsers such as Google Chrome, Apple Safari, Microsoft Internet Explorer and Mozilla Firefox. With these exhaustive combinations to be verified for the same code base, QA organizations should consider automating the effort to boost their return on investment. However, the success of automation is highly dependent on the test tool of choice.

The automation tool landscape is highly polarized to support either desktop or smartphone browsers. In RWD, however, it is important that the target tool supports browsers for either platform. Another key deciding factor for the selection of the apt automation tool is the fact that the script is reusable between mobile and desktop browsers. Technically, industry-standard tools support automation through image or optical character recognition (OCR) techniques or object identification.

Image or OCR tools automation scripts are reusable to a large extent across devices and can easily verify the UI elements that include text overlap and wrong image dimensions - elements that are crucial in an RWD project. However, such tool(s) might not be efficient if they are unable to scale images for different screen resolutions.

In the case of object-based tools, execution is primarily driven through the objects and can be used for functional verification, whereas UI validation might not be efficient. A text overlap situation or the layout (where an object is placed in the page) might prove difficult to be validated if proper frameworks are not implemented. In either case, it is necessary to ensure that the scripts offer easy maintenance and seamless UI element validation. Although automation brings efficiency and reduces RWD project costs, successful adoption of such tools depends on the particular tool chosen and the elements that require verification in various breakpoints.

Looking Forward
In many ways, RWD may seem similar to traditional Web development. However, the single code base across desktops, tablets and smartphones demands that QA organizations embrace a fresh perspective to testing - from test case design to tool selection. While the two-tier approach for development requires tools that can be adopted in the early development phases, RWD’s fluidic design will succeed only if the right automation tool is selected.

In RWD projects, focused test design, UI testing, integration testing, performance, automation and careful implementation of best practices can deliver increased efficiency, enabling QA teams to meet timelines and cost savings estimates.

Footnotes
2 Continuity is Apple’s umbrella-term for key features that allow OS X and iOS to “connect like never before.”
3 http://www.smashingmagazine.com/2013/05/29/the-state-of-responsive-web-design/.
About Cognizant

Cognizant (NASDAQ: CTSH) is a leading provider of information technology, consulting, and business process outsourcing services, dedicated to helping the world’s leading companies build stronger businesses. Headquartered in Teaneck, New Jersey (U.S.), Cognizant combines a passion for client satisfaction, technology innovation, deep industry and business process expertise, and a global, collaborative workforce that embodies the future of work. With over 100 development and delivery centers worldwide and approximately 217,700 employees as of March 31, 2015, Cognizant is a member of the NASDAQ-100, the S&P 500, the Forbes Global 2000, and the Fortune 500 and is ranked among the top performing and fastest growing companies in the world. Visit us online at www.cognizant.com or follow us on Twitter: Cognizant.

© Copyright 2015, Cognizant. All rights reserved. No part of this document may be reproduced, stored in a retrieval system, transmitted in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the express written permission from Cognizant. The information contained herein is subject to change without notice. All other trademarks mentioned herein are the property of their respective owners.

About the Author

Subbiah Muthiah is an Associate Director within Cognizant’s Quality Engineering & Assurance (QE&A) Practice’s Technology Center of Excellence (CoE), where he is responsible for techno-business development and building quality assurance for digital services, connected devices, wearables and telematics. Subbiah has more than 14 years of relevant industry experience with in-depth knowledge across the mobile stack – from the application to protocol layer. He holds a bachelor’s degree in engineering from Annamalai University, Chennai, Tamil Nadu, India. Subbiah can be reached at Subbiah.Muthiah@cognizant.com.