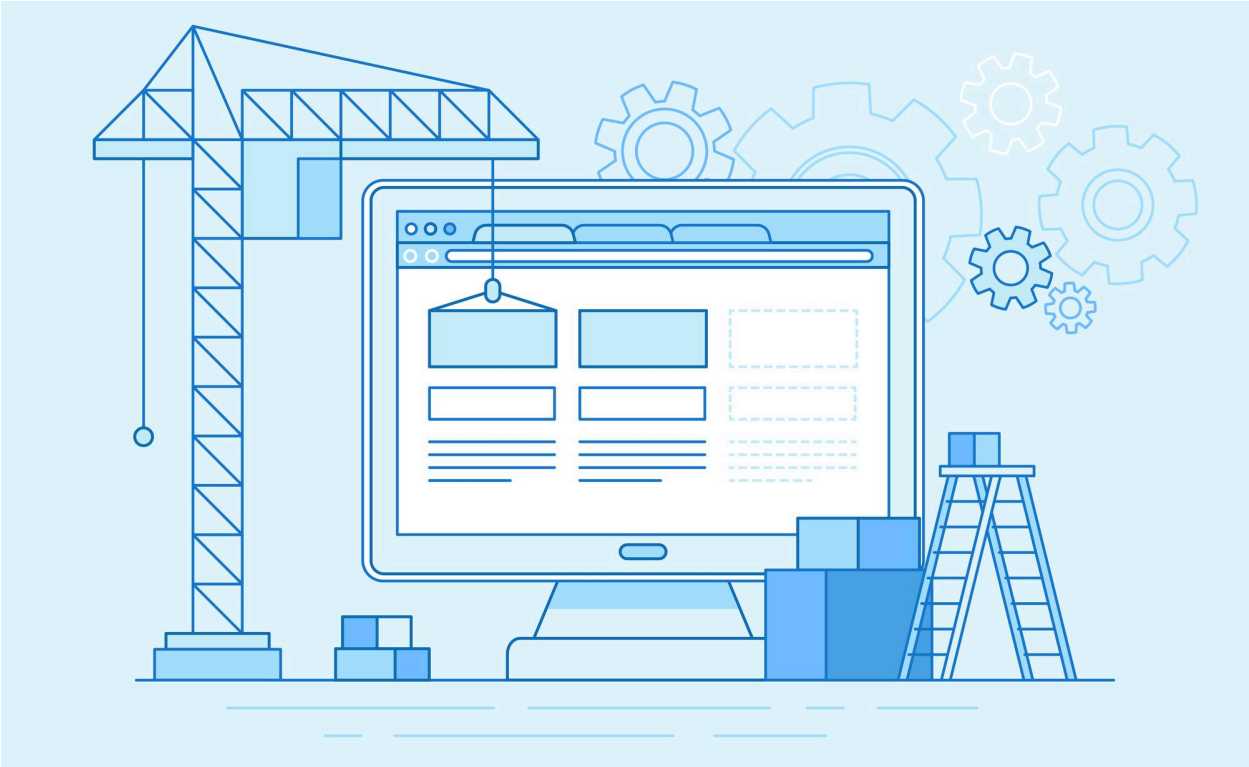
What Makes a Web Client Dynamic, Anyway?



By Jerry Karasz

Overview

Tired of applications with the same old look? Looking for a way to spruce up your user interfaces? Want your programs to be responsive, optimizing their screen size and layout for each user's device? Well, look no further — BASIS' newest user interface model, the Dynamic Web Client (DWC), is here to help you do all that and more!

If you've heard rumblings that change is on the way (or even if you have been living in Antarctica and haven't heard anything about it), read on and see what the DWC can do for you!

So What Is the DWC, Anyway?

With the DWC, BASIS introduces a new browser-based client, letting you use today's web technologies to:

- Deliver faster performance and enhanced styling using controls based on Web Components.
- Improve initial load times by minimizing the number of bytes delivered to the browser.
- Facilitate responsive design with the introduction of a new flow layout mode for top-level and child windows.
- Introduce theme support by including light and dark default themes and allowing developers to create their own themes.

The DWC supports sufficient controls, underlying structures, and frameworks for production applications that work well for mobile and desktop clients in all modern web browsers and operating systems.

Why a New Client?

So why did BASIS write a brand new client instead of just improving upon the Browser User Interface (BUI)? Well, to begin with, BUI was created to replicate the look of BBj's Graphical User Interface (GUI), as closely as possible, in a browser. And it does that well, even today. BASIS designed BUI over a decade ago based on the prevailing web standards and browser features from that era. We looked at simply improving BUI and were able to improve initial load times by optimizing the delivery of JavaScript to the client. These changes gave BUI measurable improvements in load time. But there were several desirable enhancements and features that we wanted in the DWC that simply could not be shoehorned into the BUI client, such as:

- Removing the Google Web Toolkit (GWT) from the equation.
- Adding support for flow layouts, integrating CSS layout technologies, and offering responsive and adaptive layouts that are fully contained on the client to avoid making round trips between the client and the remote server that slow down application execution.
- Overhauling the CSS and customization system by completely redesigning the concept of styling from the ground up and adding theming capabilities.

Finally, the web today is a dramatically different place than it was in 2009. It offers advanced technologies, like CSS custom properties, that just were not available when we designed BUI. Today's modern frameworks and design patterns offer improved modularity, reusability, customizability, and future-proofing, making them a vastly superior choice over decade-old web technologies. By utilizing these technologies, developers can create faster, more responsive, and more visually appealing web applications, which can ultimately lead to increased user engagement and satisfaction.

Because of these factors, there are areas where the DWC purposefully breaks ties with the past in order to provide a modern user experience. These changes include:

- Updated layout control: instead of forcing you to use precise, down-to-the-pixel fidelity for control sizing and layout, the DWC offers an optional flow layout that contributes to a dynamic UI.
- Improved font sizing: by default, the DWC does not use the same font size as BUI, as it would be antithetical to today's web standards and would ultimately detract from the user's experience.

But don't worry — we realize that backward compatibility is important, so we added configuration options. These options will help you gradually move your apps into the future. Our See Also section includes a link to find the configuration options and more information about the DWC. But for now, let's talk about how the DWC works.

How Does It Work?

Now that we have recognized the need for a new client with more modern architecture, let's look at how we designed the DWC to realize some of our previously mentioned goals, starting with improving speed.

Improved Load Time

You may have heard the saying, "You never get a second chance to make a first impression," or in web lingo, "The First Contentful Paint is critical." If a web page loads faster, the user will have a better experience. BUI's strategy is to pass everything over to the client that it will ever need. But, the new client takes a distinctively different strategic approach. When the user launches a DWC app, we load just a tiny chunk (only about 7kB) of JavaScript to bootstrap the session. After that, we dynamically download individual messages or chunks of JavaScript, on-demand as the application uses the corresponding functionality. Furthermore, when the server sends the message defining a BBjButton, it also instructs the client's browser to cache the message. This means that the server won't have to transmit the BBjButton message again to that client unless BASIS updates the control or the user clears their cache. So not only are subsequent runs of the same DWC application even faster, but executions of ***any*** DWC application on that client will be faster!

Our data transfer analysis in our browser's Developer Tools showed that eliminating the GWT, along with other optimization strategies, had a significant impact and measurably decreased our app's load time.

Responsive Design

Years ago, computers were primarily desktop machines with monitors that ran in a few standardized resolutions. That worked well with BBj's pixel-oriented positioning because the client's screen space was predictable, but today's users run applications on a variety of devices, operating systems, and screen sizes. They expect to run web apps on whatever device they have handy, whether it is a phone, tablet, or laptop. That is one reason the "Mobile First" principle is so important in product design. And one key to Mobile First design is Responsive Web Design, which describes sites and apps that adapt their layout to the user's browser viewport, like what is shown in **Figure 1**. By making use of flow layouts, CSS Grid, CSS Flexbox, and even media queries, developers can ensure that their applications run well on the user's hardware. This leads to increased user adoption, improved user experience, and ultimately, faster development and easier maintenance of the codebase.

Figure 1. *A Responsive Layout that Changes Dynamically*

By default, the new client supports absolute sizes and positions, preserving the standard BBj behavior from the older BASIS clients such as Visual PRO/5, BBj GUI, and BBj BUI. However, when you are ready to move to a responsive app, you can set a window creation flag to take full control of sizing and positioning with CSS. After that, you can size controls based on the available screen space and even define multiple layout patterns based on the screen size or orientation.

When developers define their desired responsive layout via CSS, the client's browser takes over all the responsibilities of resizing and repositioning the application's windows and controls. The BBj program no longer has to register callbacks for the window's resize events, and all layout changes occur on the client with no communication required between the client and the server. This results in faster layout changes that are not affected by latency and even permits the use of transition effects for smoother animations.

Improved Styling Capabilities

Like BUI, the DWC offers extensive support for external CSS to customize the look and feel of BBjControls in web applications. However, many developers found that writing custom CSS could be a daunting endeavor. Besides needing to learn the CSS language, it is especially challenging to debug CSS when the properties and values that you wrote do not have the desired effect.

Difficulties aside, customers requested the ability to create themes in order to more easily brand their web apps for their target customers. Keeping these matters in mind, BASIS completely redesigned CSS styling from the ground up for the DWC, integrating new concepts such as CSS custom properties and attributes.

With CSS custom properties, developers can set the value of a CSS property once and have that value take effect throughout the app. The DWC's custom properties cover all aspects of an app's look and feel and empower developers to set values to affect their app's colors, typography, spacing, borders, and even animations.

Many of the BBjControls in the DWC also offer extended attributes, which give us even more ways to influence a control. Attributes such as "theme" and "expanse" let us apply predetermined color themes and sizing properties to BBj controls, as seen in **Figure 2**.

Figure 2. *DWC Themer, Displaying a Light and Dark Theme*

Ready to Go!

Now that you have a better idea of what the DWC offers, you are ready to take the new client out for a test drive. The DWC is definitely ready for you!

Looking Back

At the beginning of this article, we discussed BASIS' high-level goals for the DWC, including reducing an app's initial load time, adding support for CSS-based layouts, and improving styling capabilities. We talked about how you can use CSS to define dynamic layouts for a variety of screen sizes and orientations. We also covered how you can exploit the power of CSS custom properties to personalize an application's appearance. The DWC is ready to help you launch your BBj graphical applications into a web browser using the latest and greatest web technologies.

See Also

There's more information available for the DWC, so be sure to check out some of the following links:

- DWC Documentation
- TechCon DWC Video Part 1
- TechCon DWC Video Part 2
- DWC on the BASIS IDE User Group Wiki
- DWC Flexbox Demo (a tool that helps you interactively design a CSS flexbox layout strategy resulting in CSS and BBj code)
- Configuration Options (search for instances of "DWC" on this page, such as `LEGACY_FONTS`, to identify the relevant configuration options)
- BBj DWC Themer