Success Story

The Journey to Single Stack – Modernizing Legacy Systems with a New Tool

Background: A 50-Year Legacy in Software Development

Len Krause, President and Chief Technical Officer of an ERP Software Solutions Provider, has been at the forefront of software development since the 1970s, building a thriving business begun by his father. Over the decades, his work has spanned a vast range of technologies, from early Basic Four computers with limited storage and working memory to modern web-based systems. The primary ERP product evolved from character-based systems to graphical interfaces, and eventually, web applications. After nearly 50 years in the industry, Len was approaching retirement but found himself drawn into the new possibilities of a recent BASIS International development tool: Dynamic Web Client (DWC).

The Decision to Embrace DWC: Excitement and Commitment

At a crossroads in his career, Len initially planned to retire to spend more time with his family and reduce his work commitments. However, during a pivotal **BASIS Technical Conference**, Len was introduced to **DWC** – a modern framework that allows BBj (Business Basic for Java, a BASIS International product) developers to manage both the server-side and web client-side of applications within a single-stack environment. Intrigued by the possibilities DWC offered, Len's excitement was immediate. He realized that this tool could fundamentally simplify his work by **eliminating the need for external web client-side technologies like Angular, and instead "do-it-all-in-BBj"**.

> *NOTE: Angular is an open-source framework maintained by Google for building dynamic web applications, especially single-page applications (SPAs). It enables developers to create sophisticated, responsive, and highly interactive user interfaces by managing client-side rendering and application state.*

Len's enthusiasm was so great that he canceled part of a planned vacation to attend DWC training sessions. By the end of training, Len had decided to **extend his retirement by two years** to fully implement DWC and roll it out to his customers. This allowed him to modernize his ERP software offerings and ensure that they remain a valuable asset to his clients and the broader BBj community.

The Challenges with Angular: Complexity and Team Disparity

Before discovering DWC, Len's team had adopted **Angular** for building **dynamic, customer-facing web pages**. While Angular enabled him to create more responsive, modern interfaces, it introduced significant complexity into his development process, particularly due to the separation between **BBj** (server-side) and **Angular** (client-side) development.

Key issues with Angular that Len experienced:

1. **Two Disparate Teams:** Managing two distinct skill sets – BBj for backend development and Angular for

frontend – proved problematic. Len's team, primarily experts in BBj, struggled with the demands of Angular's front-end logic. This led to **frequent delays**, as they had to rely on external contractors for Angular development.

2. **Contracting Challenges**: Initially, Len hired full-time Angular contractors; for the first 5 years, a 3-man full-time team in India, followed by a Ukrainian contractor, and various contracting companies in the United States. However, managing these teams proved difficult. Len faced delivery delays, and evaluating the quality and efficiency of the work was challenging without in-depth Angular expertise. These breakdowns in communication led to much frustration, reduced productivity, mounting costs, and long lead times for the implementation and deployment of new applications and modernizations.

3. **Loss of Control:** Any changes or updates on the server side (BBj) required coordination with the Angular developers, which slowed down development and increased maintenance costs. For every feature or bug fix, the BBj and Angular sides had to sync, adding another layer of complexity. In addition, Len felt he needed more knowledge to determine whether a project should take 3 hours or 3 days and felt a loss of control over both the direction of the UI and the costs thereof.

4. **Limitations in template development:** Angular requires Len to create templates for every new interface or interaction, while DWC offers pre-built web components (such as grids and tree controls) that reduce the need for custom development.

How DWC Solves Len's Problems

DWC (Dynamic Web Client) presented a solution to many of Len's frustrations by **consolidating both server and client-side development into a single stack, BBj-managed environment**. This simplified the development process, eliminated the need for external Angular expertise, and made it easier to implement new features and updates without the back-and-forth between teams.

- **Unified Development Stack:** With DWC, Len could **leverage his team's BBj expertise** to handle both the backend and frontend logic, reducing the complexity of managing two separate development tracks. This eliminates the need for hiring or managing external Angular contractors, solving the problem of disparate teams.

- **Greater Flexibility and Control:** DWC allowed Len's team to easily manage **both client-side and server-side rendering** from within the BBj ecosystem. Unlike Angular, where building user interfaces requires a separate set of skills, DWC lets developers create responsive web interfaces without needing to leave the BBj environment. This makes updates faster and reduces modernization and maintenance costs.

- **Pre-built Web Components:** DWC offers a library of **pre-built web components** like grids, forms, and menus, simplifying UI development. Len no longer needed to worry about creating custom Angular templates for every UI component. This led to a significant **reduction in development time** and allowed the team to focus on core business logic rather than front-end mechanics.

- **Backward Compatibility (Future-Proofing):** One of Len's biggest concerns was ensuring that his company's Dynamo Tools maintained its extensive history of **backward compatibility**, especially for clients who still relied on older versions of the product. DWC, developed by **BASIS International**, ensures that modern applications remain compatible with legacy systems, allowing him to modernize without sacrificing support for long-standing customers.

A Parallel with Java: Introducing webforJ

Just as DWC provided a unified *"do-it-all-in-one"* solution for Len's BBj-based system, BASIS offers **webforJ** for developers working with **legacy backend systems in Java**. Like DWC, webforJ is designed to **modernize legacy Java systems**, allowing developers to create dynamic, responsive web applications while retaining backward compatibility with existing Java infrastructure, all while *"doing-it-all-in-Java"*. This approach ensures that companies with a strong investment in their Java backends can modernize without needing to rewrite their entire codebase.

BASIS: 40 Years of Modernization with Backward Compatibility

BASIS has a long history of helping companies like Len's to transition from legacy systems to modern solutions while ensuring **backward compatibility**. For over 40 years, BASIS has provided tools that allow developers to **modernize their applications** without abandoning their existing technology stack. This has allowed businesses to evolve their software while supporting long-term clients who still rely on older systems.

In Len's case, the **BASIS DWC framework** enabled him to leverage modern web technologies while preserving his products' legacy functionality. This backward compatibility with forward-looking modernization tools makes BASIS a trusted partner for businesses like ERP Software Solutions.

Conclusion: The Path Forward

Len sees the BASIS DWC as a breakthrough that will help him update his company's flagship product, make development more efficient, and guarantee the long-term viability of his applications and tools. By delaying his retirement by two years, Len can concentrate on **transitioning the UI to DWC**, ensuring that his product remains competitive and valuable for years to come.

The success of this transition reflects BASIS International's broader mission: providing developers with the tools they need to **modernize their applications** while ensuring **backward compatibility** with older systems. Whether it's DWC for BBj or webforJ for Java backends, BASIS continues to lead the way in assisting businesses in adapting to new technology while preserving their rich histories, delivering on its Future-Proofing promise - *"do-it-all-in-one"*. ■

About us

BASIS International

We are a global software company – large and stable enough to be a reliable partner for thousands of companies worldwide, yet small and flexible enough to deliver tailored solutions that meet today's business challenges.

For almost 40 years (20+ years of delivering technology built on Java), BASIS International has been a trusted partner, delivering software tools and frameworks to develop and modernize mission-critical business systems for both large international corporations and SMEs. More than 1.2 million users worldwide rely on IT solutions created by BASIS or its customers and partners, specifically tailored to meet individual requirements.

With offices in North America and Europe, as well as partnerships in over 30 countries, BASIS is well-positioned internationally. The "big little software company" is privately owned and independently operated.

BASIS Europe
Nell-Breuning-Allee 6
66115 Saarbrücken
Germany

+49 681 968 14 0
www.webforj.com