5 REASONS TO PUT ASIDE THE ‘GREEN SCREEN’

The ISPF Green Screen mainframe interface has served its purpose, but better alternatives are here. Just as technological advances prompted the transition from cards to the green screen, for the next move forward in mainframe productivity it’s well overdue to transition to current interfaces and DevOps pipelines. It’s time to see ISPF for what it really is—stuck along with ‘80s waterfall development practices whose continued existence in the digital age is inhibiting mainframe agility, DevOps and innovation.

Here are five reasons to replace ISPF in your mainframe shop.

1. It’s esoteric

A keyboard is the only means of navigating ISPF. Without icons, dropdowns, or menus, you need to know which command line options to type to get anywhere, and you’ll have to spend quite a bit of time reading a manual written 30 years ago to learn them. When a developer, or even a non-developer, looks at a graphical user interface, they can quickly determine how to get around and start working. Navigation is easy with clickable icons, dropdowns, and menus, all of which are accessible with your mouse.

2. You have to memorize obscure command lines

There’s always the manual to look to, but, hey, why not just Google “ISPF command lines”? There’s a high chance your results will include “ISPF commands cheat sheet”—you’ll need one if you have

little or no mainframe experience and you're in your first job as a COBOL developer. The only people who know these commands by heart are those who have been typing them into the same screen for 35 years.

3. You can't visualize programs

Most COBOL programs were written decades ago, and many haven't been updated in years. These tend to be large, complex programs only a few people understand, if they're still working on them or working at all. While Eclipse gives you the opportunity to see a graphical representation of your code, making these programs more easily understood, ISPF doesn't come close. At most, you get a green and black, or maybe red and yellow, screen, which makes it even harder to analyze and understand lines upon lines of data as you scroll up and down with your "PF8" and "PF7" keys.

4. You can't be agile and multitask

When you're developing with Agile, you need to be nimble enough to do things simultaneously, like test code as you build or work parallel with another developer in the same program. Agility is difficult, sometimes impossible, with ISPF, largely because you can't multitask. You're limited to one 48×80 view—unless you expand to 80×160—so doing more than one thing requires opening another 3270 connection or knowing esoteric commands to split the screen and toggle back and forth.

5. Your next-gen workforce won't use it

ISPF is the quickest way to turn next-gen developers off to the mainframe. True, some millennials have a thing for the '80s, but expecting them to spend 40 or more hours a week reenacting historical coding techniques on something that looks like it's out of *Tron* is too much. Millennials have grown up using graphical technology that requires simple logic to use, not linear tools that require years of practice and studying to master arcane tasks.

While ISPF may have helped lift mainframe shops out of the punched-card era, it's only going to scare away developers who are new to the mainframe today. Fortunately, mainframe shops don't need ISPF; there are the alternatives available, like Eclipse, VS Code, and REST APIs:

- Developers can use modern, Eclipse-based tools that leverage Agile/DevOps best practices, such as BMC Compuware Topaz, to enable quick understanding of outdated, complex, or poorly documented mainframe programs.
- It could be said that sometimes the best interface is no interface. A modern automated Jenkins pipeline using REST APIs and Plugins will provide for higher quality and more output. Newer developers will be able to demonstrate the benefits of automation to those still using ISPF.
- A new integration offers VS Code support for BMC AMI DevX Code Pipeline. This new extension allows developers who are accustomed to working with Git using VS Code to continue being productive. When it comes time to test and deploy on the mainframe, they can easily bring code into ISPW with the extension for a generate or build.

These alternatives don't require comprehensive training or arduous memorization, and they aren't scary to use—you just start using them. Why would a mainframe shop want anything less?

For more information on BMC AMI DevX Code Pipeline *go to this link. Also included in our April 1 release are a new BMC Compuware Topaz command line interface and multiple APIs that give*

developers flexibility in how they connect to the mainframe, lessen dependence on specialized skillsets, and extend the DevOps toolchain. To learn more about the enhancements included in the April release, visit our What's New in Mainframe Solutions webpage.