



Feliz 60vo.
aniversario a
COBOL. ¡Nunca te
has visto mejor!

ESTAMOS MUY EMOCIONADOS CON LAS MEJORAS EN LA VERSION 2020R1.



Puedes

- insertar nuevos controles GUI en tu programa como SCROLL-PANE y estilo TREE-VIEW TABLE-VIEW
- tener un control más preciso sobre la interfaz de usuario y la gestión de eventos del mouse utilizando el estilo NOTIFY-MOUSE compatible con todos los controles
- mostrar miniaturas de página en la vista previa de impresión, lo que permite al usuario saltar rápidamente a una página específica en un informe de varias páginas

Maurizio Girometti, nuestro CEO, dice que isCOBOL, Veryant, comenzó con la misión de "revitalizar COBOL y transformarlo en un lenguaje tecnológico de vanguardia" y con las mejoras en 2019R2 y 2020R1 continuamos con esa misión.

veryant

NEWS

EN ÉSTE NÚMERO

1. Novedades en 2020R1
2. COBOL HOY
3. Code Coverage y Unit Test
3. Ejemplos de Multithreading
4. Relevantes en la Documentación
4. LAF: Seleccionando el diseño básico para tus pantallas GUI
4. Has visto esto?
5. Destacados en la versión 2019 R2
5. Zoom Layout Manager
5. Mensajes de error del compilador
6. JOE scripting language
6. Opciones de Datos
7. Ultima página

LO NUEVO

2020 R1 está aquí

Veryant se complace en anunciar la liberación de la versión más reciente de isCOBOL™ Evolve, isCOBOL Evolve 2020 R1.

isCOBOL Evolve proporciona un entorno completo para el desarrollo, despliegue, mantenimiento y modernización de aplicaciones COBOL

Ahora con la versión 2020R1 de isCOBOL se introducen nuevas capacidades: Code Coverage y Unit Test. Estas características han estado disponibles en otros lenguajes, como Java, y ahora también están disponibles para los desarrolladores de isCOBOL. Le ayudarán a producir pruebas de mejor calidad para aplicaciones COBOL.

Estas nuevas características también están integradas en el IDE, así como el plegado de código personalizado. Hemos agregado algunos nuevos controles GUI, un estilo de un panel de desplazamiento y una vista de tabla al control tree-view que lo ayudarán a mantener sus aplicaciones con un aspecto fresco y actualizado.

Otras mejoras incluyen un manejo más fino del control del mouse, miniaturas de vista previa de impresión y, por supuesto, las constantes mejoras continuas en el rendimiento y la compatibilidad.

Esta edición del boletín incluirá información sobre 2019 R2 también. Normalmente, se introducen nuevas características importantes en las versiones de R1 a principios de año, pero hay algunos cambios en 2019 R2 que son bastante emocionantes, como LM-ZOOM para cambiar el tamaño de la pantalla completa con una variable y la inserción de código para aplicar un estilo a un control utilizando el archivo de configuración en lugar de cambiar su código.

COBOL HOY



Maurizio Girometti, CEO de Veryant, habla sobre sus experiencias con COBOL y cómo encaja en el mundo de hoy

Comencé escribiendo aplicaciones COBOL en 1982 en una minicomputadora Texas TI-990. Fue una gran experiencia después de practicar con Assembler, Fortran y Basic. Encontré COBOL fácil, estable y, por supuesto, absolutamente adecuado para aplicaciones comerciales.

Con el advenimiento de nuevos lenguajes, COBOL comenzó a tener una mala reputación en la comunidad de TI. Al principio existía la ilusión de los lenguajes 4GL: en teoría, cualquier programador podía escribir aplicaciones complejas en minutos. *No es realmente cierto.*

Entonces otros lenguajes prometieron ofrecer tecnología de punta y de hecho lo hicieron. Esto definitivamente contribuyó a aumentar la mala reputación de COBOL, ya que la mayoría del mundo de TI pensó que en este punto COBOL acababa de ser relegado a las viejas aplicaciones de mainframe.

Entonces, ¿es cierto que COBOL es viejo y no soporta las nuevas tecnologías o es solo un "mito urbano"?

En 2005 decidimos iniciar Veryant con el propósito de revitalizar COBOL y transformarlo en un lenguaje tecnológico de vanguardia, ya que todavía hay mucho software en el mundo escrito en COBOL. Como resultado, isCOBOL® nació y hoy es utilizado en todo el mundo por miles de desarrolladores.

Así que, dado que isCOBOL proporciona todas las funcionalidades proporcionadas por los lenguajes modernos, incluida una interfaz gráfica de usuario, servicios web, cliente ligero remoto, interfaz web, OOP, un potente IDE basado en Eclipse y mucho más, entonces asumimos que la idea de que "COBOL es un lenguaje antiguo" definitivamente es un "mito urbano".

"En 2005 decidimos iniciar Veryant con el propósito de revitalizar COBOL y transformarlo en un lenguaje tecnológico de vanguardia"

Y, todo esto es portable desde mainframes a dispositivos móviles sin cambios, ya que al final del proceso de compilación lo que obtienes es un objeto Java. Esto significa que si es necesario, puede interactuar con cualquier otro objeto Java, aprovechando fácilmente el poder del mundo Java.

Si quieres evaluar nuestro producto, por favor llama nuestras oficinas y estaremos encantados de proporcionar más información técnica y de ventas, y entregar una copia de evaluación de nuestros productos. Nuestro personal técnico te asistirá durante todo el período de evaluación.

Si fueras agricultor, ¿seguirías arando el suelo de esta manera?

veryant
Evolution without revolution



Code Coverage y Unit Testing

La nueva capacidad Code Coverage le dirá que tanto de su código realmente está siendo ejecutado durante una corrida. Esto es útil cuando configura un entorno de prueba y desea asegurarse de que está probando la mayor cantidad de código posible.

Hemos facilitado la implementación (agregue -coverage a la línea de comando de ejecución) para comprender los resultados con un "Informe global de cobertura" utilizando gráficos para mostrar la cobertura a nivel de programa y párrafo, así como también darle acceso a su código, identificado por colores para mostrar línea por línea lo que se ejecutó y lo que no.

Sabemos que que no todos han podido crear un entorno de prueba automatizado, y también hemos agregado algo para ayudarlo con eso. La función de prueba unitaria permite a los desarrolladores crear suites de pruebas automatizadas, diseñadas para probar que las secciones de

código se ejecutan según lo previsto. El objetivo de Unit Testing es aislar secciones de un programa y asegurarse que funcionen correctamente.

Para configurar esta prueba, debe agregar una declaración ASSERT a las partes del código que desea probar, similar a esto:

```
assert string1 = "my string"
otherwise "Test string
manipulation: Error"
```

Cuando compila y ejecuta su programa con las opciones correctas, se activa la Unit Testing, sin estas opciones se ignoran las declaraciones. La función de Unit Testing genera reportes similares al Code Coverage, y cuando se ejecutan juntos, los informes se combinan.

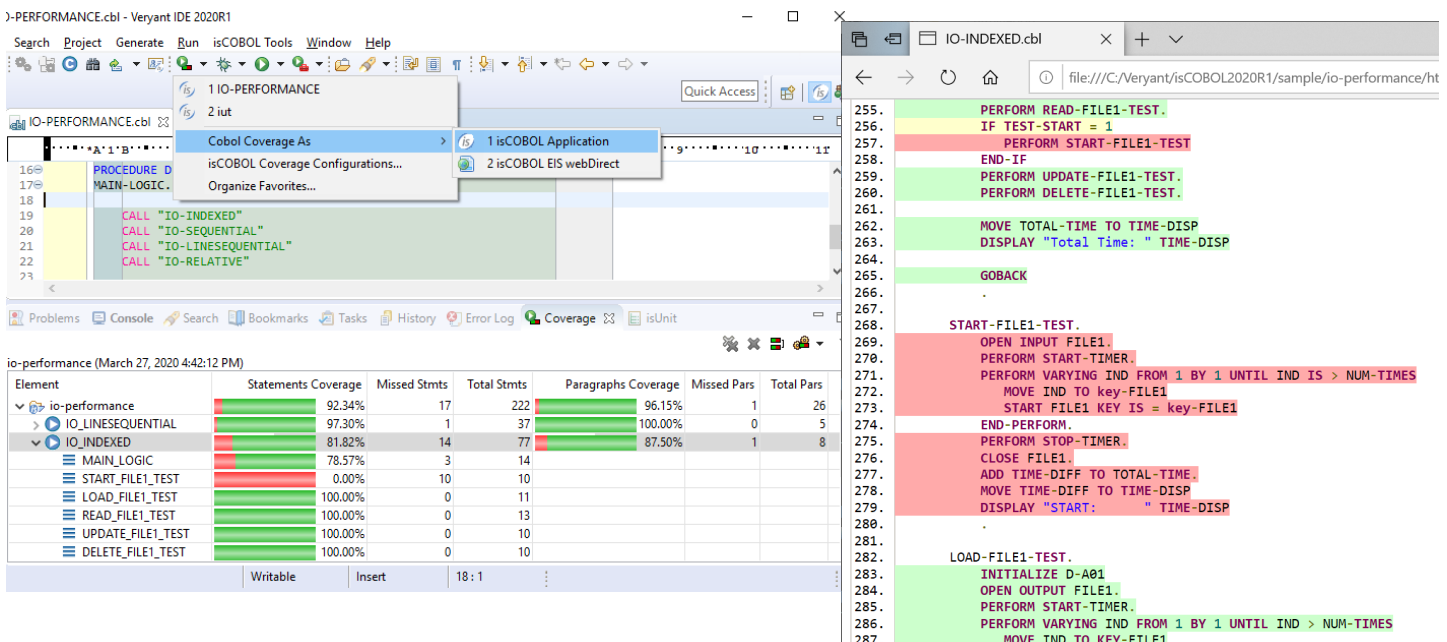
Y lo mejor? Todo esto también está integrado en el IDE . Póngase en contacto con soporte técnico o con su representante de ventas para obtener más información o una demostración.



EJEMPLOS DE MULTI THREADING

Has visto últimamente los ejemplos de subprocesos múltiples? Es una forma interesante y fácil de agregar valor a tu aplicación. Los ejemplos te muestran cómo:

- ejecutar un reloj en segundo plano
- evitar cerrar una ventana antes de otra
- asegurarse de que el usuario no ejecute un programa más de una vez
- contar los programas en ejecución
- Ejecutar multiples programas en el mismo ambiente o en ambientes separados
- Permitir que el usuario interrumpa un proceso en ejecución
- Ejecutar una barra de progreso



DESTACADO EN LA DOCUMENTACION



Nueva Documentación

La documentación de isCOBOL ha sido rediseñada con una nueva apariencia moderna y ahora es accesible para desarrolladores desde dispositivos móviles.

Esto también resuelve el problema que pudo haber notado en la Documentación de 2019: algunos navegadores (como Chrome y Firefox) no mostrarán su documentación instalada debido a una nueva característica de seguridad que introdujeron.

Pruébelo y deje que nuestro departamento de documentación sepa lo que piensa. Simplemente haga clic en el icono de correo electrónico en la parte inferior derecha de cada

Puede accederla [aquí](#).

Has visto esto?

Nuevos videos en YouTube:

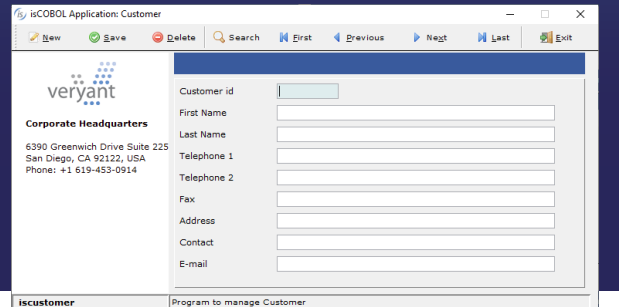
- [REST/SOAP Webservices](#)
- [Installing isCOBOL SDK for beginners](#)
- [isCOBOL 2019R1 new features](#)

Nuevos articulos en la Base de Conocimiento:

[How to use existing FD/SL copybooks in the Screen Programs without generating FD/SL from IDE Data View](#)

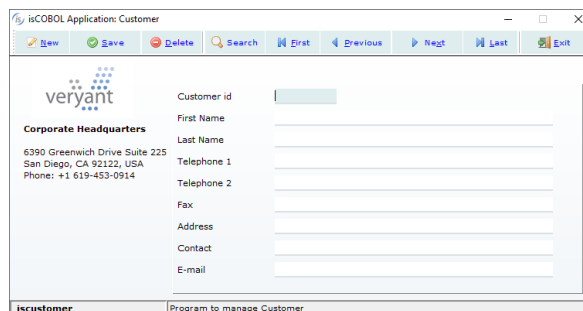
LAF: Eligiendo un diseño básico para sus pantallas GUI

Pantalla Original:



Compilado con estas variables de configuración:

```
iscobol.compiler.gui.window.defaults=gradient-color-1 rgb x#ffffff, gradient-color-2 rgb x#F2F6F9  
iscobol.compiler.gui.label.defaults=transparent  
iscobol.compiler.gui.tool_bar.defaults=Background-Color rgb x#FFFFFF,Foreground-Color rgb x#F2F6F9  
iscobol.compiler.gui.push_button.defaults=flat, Background-color -14675438, Foreground-color x#F2F6F9  
iscobol.compiler.gui.entry_field.defaults=border-color rgb x#dae1e5, border-width (0 0 2 0 )
```



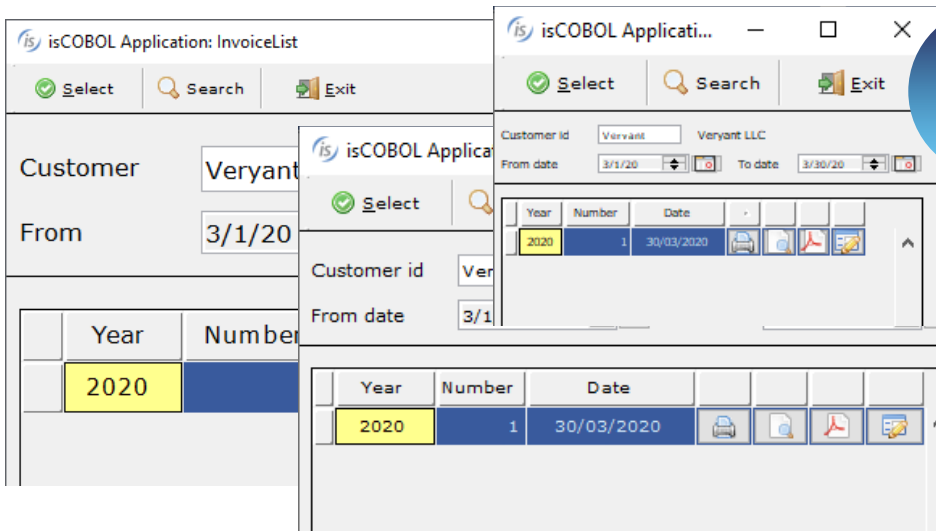
El mismo programa compilado con las variables de configuración anteriores

Un lenguaje de diseño es un conjunto de estilos y principios de diseño para proporcionar una apariencia coherente en toda una aplicación o plataforma. [Apple Human Interface guideline](#), introducido a finales de 1977 con la Apple II, [Google's Material Design language](#), desarrollado en 2014, y Microsoft's [Fluent Design System](#) introducido en 2017 son ejemplos comunes de lenguajes de diseño, y describen la apariencia y funcionamiento que los usuarios ven todos los días.

Si desea que su aplicación se vea familiar y actualizada, probablemente terminará imitando uno o más de estos lenguajes de diseño.

En 2019R2 hemos agregado algunas características para ayudarlo a mantener sus pantallas GUI frescas: bordes personalizados para los campos de entrada para que pueda copiar el diseño del elemento de entrada de Google con solo un borde inferior visible, y un botón plano y una pestaña de acordeón mejorada.

Y para facilitarlo, puede aplicar estos y otros estilos desde el archivo de configuración en lugar de agregarlos individualmente a la descripción de cada control en cada programa.



DESTACADOS EN 2019 R2

Algunos de ellos:

- Insertar estilos de GUI en su programa desde el archivo de configuración, sin cambiar su código
- Hemos mejorado el manejo dinámico de variables
- Hemos aumentado el rendimiento de las declaraciones CALL y las operaciones c-treeRTG.
- Nuestra compatibilidad con RM/COBOL ha aumentado para soportar v8 o superior
- La administración de la consola de WebClient se ha vuelto más informativa y segura con grabación automática de sesiones y notificaciones push al cliente.



Zoom Layout Manager

En el último boletín hablamos sobre el Responsive Layout Manager, que le permite aplicar el cambio de tamaño a cada control en su pantalla para responder en diferentes ambientes de despliegue.

En 2019R2 – ampliamos esto con el lm-zoom layout manager style. Usando esta variable en el archivo de propiedades file:

iscobol.gui.layout_manager=lm-zoom

automáticamente habilita su aplicación para adaptarse a múltiples dispositivos,

tamaños y resoluciones de pantalla. Permite al usuario cambiar el tamaño de una ventana de aplicación o rotar una pantalla, un comportamiento requerido en la era del desarrollo de aplicaciones para los dispositivos móviles.

Sus ventanas se pueden cambiar automáticamente de tamaño: la propiedad de tamaño de los controles se ajusta con el ancho de la ventana y la propiedad de tamaño de fuente de los controles se ajusta con la longitud de la ventana.

Tipos de mensajes en errores de compilación

iscc prog1.cbl

--S: #36 Ambiguous identifier A; file = prog1.cbl, line = 17, col 11
 --E: #154 End statement required END-PERFORM; file = prog1.cbl, line = 21, col 19
 --W: #254 Procedure name same as data name: A; file = prog1.cbl, line = 23

Sabes lo que significan S, E y W arriba? Cuando compilas tu código, hay tipos de mensajes. S es un error grave, por lo que tu código no se compilará en una clase. E es un error regular y W es una advertencia: ambos no impedirán que el compilador compile y cree su .class.

Si, por ejemplo, no deseas ver la advertencia anterior cada vez que compilas, puedes crear una propiedad de configuración para ocultarla:

iscobol.compiler.message level.254 = 0

[lee mas acerca de este tema en nuestra documentación.](#)



JOE - El poder detrás del lenguaje de scripting de Veryant

Tradicionalmente, los desarrolladores de software han combinado de programación con utilitarios de lenguajes de script para unir sus módulos de aplicación. Estos lenguajes de script (como los lenguajes de script de shell) ofrecen una estructura más relajada, sin fase de compilación y una forma flexible y simple de interactuar con el sistema operativo que los lenguajes de desarrollo más formales suelen carecer. A menudo, las aplicaciones heredadas combinan el desarrollo de COBOL con lenguajes de script basados en el sistema operativo y algunas funciones clave de las aplicaciones pueden depender en gran medida de él. Si la plataforma de desarrollo objetivo careciera del equivalente de ese lenguaje de script, complicaría severamente la transición. Teniendo en cuenta todo eso, Veryant ofrece nuestro propio lenguaje de secuencias de comandos, llamado JOE.

Además de los beneficios del lenguaje de programación estándar, JOE también puede:

- Interactuar directamente con clases de Java
- Interactuar con las clases isCOBOL
- Acceder y actualizar archivos ISAM
- Acceda y actualice cualquier RDBMS a través del driver JDBC

y SQL

- Acceder y fijar variables de ambiente
- Ofrece un flujo lógico básico pero potente y manejo de expresiones
- Interactúe con aplicaciones de Office (como Excel) a través de clases de POI de Apache o similares
- Trabaje sin problemas con la arquitectura del servidor de aplicaciones

Y como cualquier otro lenguaje de script, JOE:

- Es fácil de entender, usar y extender (más allá de la fase de migración)
- Se puede personalizar para manejar operaciones frecuentes
- Se asemeja a cualquier lenguaje de secuencias de comandos en términos de capacidad y legibilidad

Por todo lo anterior, JOE es la respuesta perfecta a su situación de migración de qué hacer con su lenguaje de procedimiento heredado.

Es la elección perfecta para la modernización y el desarrollo continuo para implementar operaciones frecuentes con una poderosa simplicidad y elegancia, uniendo los componentes de su aplicación.

```

1  /** isCOBOL exec JOE */
2
3  checkSurname := {a_value.
4    a_surname := a_value toString ; toUpperCase ; trim.
5    if ((a_surname = "DOE") or (a_surname = "SMITH")),{
6      "Yes, he's a member of Veryant staff".
7    },{
8      "Veryant staff only allowed here!".
9    }
10 }.
11
12 checkCompany := {a_value.
13   a_company := a_value toString ; toUpperCase ; trim.
14   if (a_company = "VERYANT",{
15     "Yes".
16   },{
17     "Only Veryant is allowed!".
18   }
19 }.
  
```



OPCIONES DE ALMACENAMIENTO DE DATOS

Con isCOBOL, tiene una amplia gama de opciones de almacenamiento de datos, que incluyen:

JISAM

Este es el formato de archivo indexado predeterminado en el runtime de isCOBOL.

c-tree

Esta es otra opción incluida con el runtime. En lugar de usar directorios compartidos para sus archivos de datos, como lo hace JISAM, c-tree es un servidor de archivos, que agrega velocidad y seguridad a sus datos. c-tree puede expandirse para ser una base de datos y permitir el acceso a sus datos con SQL.

RDBMS con sintaxis COBOL

Nuestro producto Database Bridge, también conocido como EasyDB, convierte sus declaraciones de E/S de archivos COBOL en sentencias SQL y se conecta con su sistema de base de datos relacional

RDBMS con ESQ

El soporte de ESQ está integrado en el compilador isCOBOL, sin necesidad de de un precompilador externo

File Connectors

Acceda a sus datos en archivos Vision, c-tree, DBMaker o Micro Focus ISAM sin convertirlos.

Base de datos DBMaker y Pervasive

El DCI y Btrieve file handler están integrados en isCOBOL, por lo que puede almacenar sus datos en estos sistemas de bases de datos sin EasyDB u otros costos adicionales.



Evolution without revolution

Veryant LLC

El equipo de soporte técnico dedicado y experimentado de Veryant está comprometido a proporcionar los más altos niveles de atención al cliente

Para clientes con soporte, envíenos un correo a support@veryant.com

Si desea que Veryant se comunique con usted para programar una reunión técnica sobre el producto, envíenos un correo a info@veryant.com

Si desea que Veryant se comunique con usted para solicitar un presupuesto especial o asistencia de ventas, envíenos un correo a sales@veryant.com

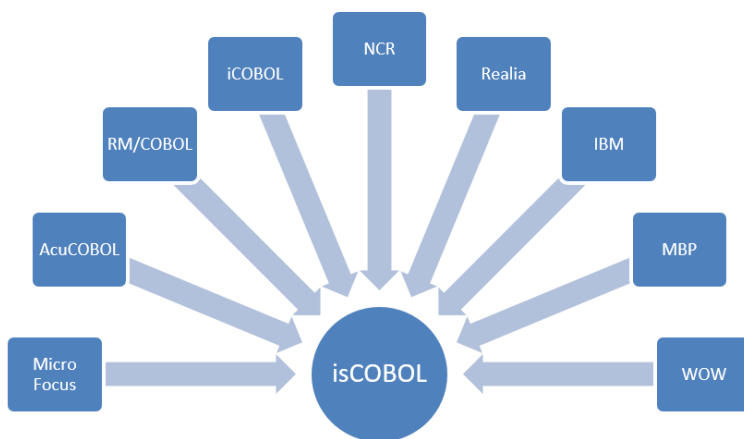
Corporate Headquarters

6390 Greenwich Dr., Suite 225
San Diego, CA 92122 - USA
Tel (English): +1 619 797 1323
Tel (Español): +1 619 453 0914

European Headquarters

Via Pirandello, 29
29121 - Piacenza - Italy
Tel: +39 0523 490770
Fax: +39 0523 480784
emea@veryant.com

Como siempre, 2020R1 contiene múltiples adiciones de compatibilidad, a medida que continuamos haciendo que su proceso de conversión sea lo más fluido, rápido y sin problemas posible.



veryant.com

**Ofreciendo opciones, flexibilidad
y soluciones rentables para
organizaciones con valiosos activos de
COBOL**

**CONTACTENOS PARA
UNA DEMOSTRACION**

CLICK HERE



Cool Business Oriented Language

Follow **Veryant** on



[**veryant.com**](http://veryant.com)