2022

ISSUE 06

REVISTA TRIMESTRAL DE VERYANT E isCOBOL

veryant NEWS

NESTA EDIÇÃO

JUNTE-SE A NÓS NA SCALE 19X EM LOS ANGELES

JULHO 31/28, 2022

A maior conferência de código aberto administrada pela comunidade da América do Norte está de volta e estamos montando nosso estande como um patrocinador.

Nos faça uma visita, e ganhe um brinde gratuito!

Mais informações aqui.

JUNTE-SE A NÓS:

Twitter, LinkedIn, ou Facebook para atualizações do with Veryant's news

Assista nossos vídeos de demonstração e assine nosso canal do YouTube

2022 R1 e mais

2022R1 deu ao isCOBOL um grande passo à frente para manter-se atualizado e relevante em ambientes maiores e modernos

Agora nós suportamos todas as versões LTS (suporte de longo prazo) do Java; Java 1.8, Java 11 e Java 17. Você pode ver a data de lançamento e as datas de término do suporte para as versões Java aqui. O isCOBOL IDE é atualizado para usar a mais nova base estável do Eclipse. O 2022R1 é certificado para ser executado no Windows 11 com Java 11 ou 17. Adicionamos clustering ao WebClient, então agora você pode equilibrar a carga de processamento do WebClient em vários servidores. Procure um vídeo demonstrando como fazer isso em nosso site do YouTube, aqui.

O 2022R1 tem algumas maneiras diferentes de expandir o manuseio do arquivo de configuração. Você pode adicionar um arquivo a outro com iscobol.conf. copy=<filename>. Você pode codificar para redefinir todas as variáveis para seus valores originais ou configurá-las para valores armazenados em um arquivo, com C$CONFIG. Manipular bitmaps e fontes que acompanham seu aplicativo é mais fácil agora com a sintaxe COPY RESOURCE para incluí-los em suas classes.

Evite travar processos em segundo plano definindo iscobol.display.message.timeout para que uma caixa de mensagem seja fechada automaticamente. Como de costume, temos muitos aprimoramentos de controle de GUI, incluindo muitos para o menu de hambúrguer, adicionando pesquisa, seu próprio gerenciador de layout e várias outras especializações a esse novo tipo de menu que funciona bem em telas pequenas. Você pode definir a janela principal para escurecer ao exibir uma janela modal que exija a atenção do usuário. Esses são apenas alguns dos novos recursos em 2022R1. Você pode obter mais informações sobre todos eles na documentação de lançamento e no vídeo.

O que você vai fazer com seu COBOL?

Um argumento para manter seu COBOL ao invés de recomeçar, por Daniel Cárdenas, Business, Development Director da America Latina

Recentemente, muitas pessoas falaram sobre COBOL e aplicativos "antigos", e muitos fora da indústria COBOL não entendem por que algo não foi feito sobre a situação antes da crise atual.

Mas aqueles de nós que usam COBOL todos os dias sabem que é "mais fácil falar do que fazer". Seu aplicativo COBOL não apenas funciona, mas também trabalha duro todos os dias. Pensar em mudar um sistema essencial com décadas de modificações, aplicações específicas do setor, com milhões de linhas de código, não é tarefa fácil. Decidir qual direção seguir é assustador, e eu desafiaria qualquer tomador de decisão a dizer que não sucumbiu à "paralisia da análise" sobre o assunto pelo menos uma vez.

Os tomadores de decisão responsáveis por decidir o que fazer com seus programas COBOL legados estão em uma encruzilhada. Há dois caminhos a seguir: manter o COBOL e atualizá-lo, ou jogá-lo fora e começar de novo com um pacote genérico ou uma reescrita total. Uma vez que você começa por um caminho, é muito difícil mudar sem voltar até a encruzilhada e começar de novo

Meu filho uma vez limpou seu quarto tirando tudo dele e depois colocando tudo de volta. Sua teoria era que ele só devolveria exatamente o que precisasse, e seria mais organizado. E foi basicamente isso que aconteceu. No entanto, ele levou um fim de semana inteiro, então ele teve que dormir no sofá por duas noites durante o processo e a casa inteira estava uma bagunça durante esse período. E, eventualmente, seu quarto ficou tão bagunçado quanto antes, porque as coisas que ele achava que não precisavam voltavam ao seu quarto ao longo do tempo.

Substituir seu COBOL por um pacote genérico ou reescrever totalmente seu aplicativo é um pouco como seu método "cortar e queimar" de limpeza de sala. Sim, você pode acabar com algo brilhante e brilhante. Mas você ainda terá que tocar em cada parte do seu aplicativo COBOL para tentar copiar sua funcionalidade.

E então pague muito dinheiro a um fornecedor externo para modificar seu pacote genérico para se adequar ao seu negócio ou espere anos para que seus programadores escrevam um aplicativo do zero que substituirá algo que levou décadas para ajustar.

Atualizar seu COBOL existente também traz problemas, como quem vai manter o COBOL se todos os seus programadores COBOL estiverem se aposentando? E como você diz a seus clientes que tem um aplicativo de ponta escrito em COBOL? Como você escolhe o fornecedor COBOL certo com o conjunto certo de ferramentas para tornar sua atualização bem-sucedida?

> "Não só sua aplicação COBOL funciona, como trabalha duro todos os dias"

É aí que Veryant se encaixa na decisão. Nosso IDE é baseado em um IDE padrão da indústria – Eclipse, então a maioria dos programadores já está familiarizada com o ambiente. COBOL é uma linguagem relativamente fácil de aprender, e desenhar telas e codificar em COBOL é muito fácil quando eles usam as ferramentas IDE integradas que criamos.

Se você deseja minimizar as interações do seu programador com o COBOL, é fácil criar serviços REST individuais e colocar seu COBOL em uma "caixa preta" em segundo plano e usar uma linguagem mais moderna melhor ajustada a exibições gráficas, como Java, JavaScript ou Python para seu código de interface do usuário.

isCOBOL compila seu COBOL em código Java e, em seguida, compila novamente em uma classe Java. Até onde seus clientes ou usuários podem dizer, você tem um aplicativo Java.

Talvez a decisão mais importante de sua empresa seja o que fazer com seu COBOL. Espero que você considere seriamente mantê-lo. Ligue-nos para uma discussão, demonstração ou análise de código.

Ícones isCOBOL IDE

A IDE usa diferentes ícones nas 3 seções do IDE explorer para representar o status do projeto. Aqui está o seu mapa para entender esses ícones.

Todas Seções


isCOBOL sections


isCOBOL library: lists the version of isCOBOL library used by the project

Visualização Estrutural


isCOBOL screen program


isCOBOL WOW program


Screen designer section for Screen program or Forms list for the WOW program.


Screen designer for Screen program or Form designer for WOW program


Report designer section, containing all the reports of each program.


Report designer


Working storage and Local storage designer


Linkage section designer


File section containing all the datasets (file) of each program.


Dataset (the representation of a file used inside the program)


Event paragraph for Screen program and WOW program

Decoração dos Ícones

Cada ícone pode ser decorado com algum pequeno símbolo para mostrar informações extras.

Um aviso do compilador adicionará um triângulo amarelo no canto inferior esquerdo.

Por exemplo o ícone da fonte torna-se e a pasta.

Um erro do compilador adicionará uma cruz vermelha no canto inferior esquerdo.

Por exemplo, o ícone da fonte torna-se a pasta.

O ícone de erro substituirá o ícone de aviso.

Visualização de Arquivo

- Empty project folder
- Project folder with some files in it
- List folder with some files in it
- Copy file
- List file
- Program source code
- Program id: under each source code there is this icon. This represents the generated class
- Class id: this icon has the same meaning as above, but for a class id
- The container of the methods of a class id
- A method of the class-id

Data view

- Dataset. The representation of the dataset (file), one for each file in the program
- File description (FD) designer
- File key designer
- File i-o handling designer
- File EFD designer
- File event paragraph designer

Decorações de ícones

Se o arquivo ou a pasta não estiverem fisicamente presentes na pasta do projeto, mas estiverem vinculados de outro local, um símbolo de link será adicionado no canto inferior direito do ícone. Por exemplo, o ícone de  em torna-se e a  pasta

Estas são as decorações isCOBOL. Você pode ver outros adicionados por plug-ins Ecplise, como SNV.

Você já viu isso?

Novos vídeos do YouTube

Profiler isCOBOL - como funciona e uma demonstração
Utilitário de Cobertura de Código do isCOBOL
O que há de novo em 2022R1 – com demonstração
Tudo sobre o licenciamento isCOBOL
Personalizando isCOBOL
Agrupamento de WebClient

Novos artigos da base de conhecimento (KB)

Como você pode esperar que vários tópicos terminem?
Como usar ícones baseados em fonte em programas isCOBOL GUI
Posso usar o COBFILEIO para dar acesso aos meus arquivos c-tree a um programa Java?
Modernize seu aplicativo de personagem

Tendo um volume persistente para isCOBOL Server em um Docker

O segundo de uma série de três partes sobre isCOBOL e Docker Containers, este artigo mostra como usar dados persistentes em seu contêiner. Aqui está um guia passo a passo do Engenheiro de Suporte Sênior, Valerio Biolchi

Pré-requisitos

Os pré-requisitos em um computador Linux de 64 bits (onde o docker é construído) são:

- O mecanismo Docker (https://docs.docker.com/engine/install/) instalado a partir de um repositório
- Um ambiente isCOBOL SDK licenciado
- Um servidor isCOBOL em execução no Docker com configuração e execução de amostra ISAPPLICATION Veja as instruções no primeiro artigo da série, aqui.

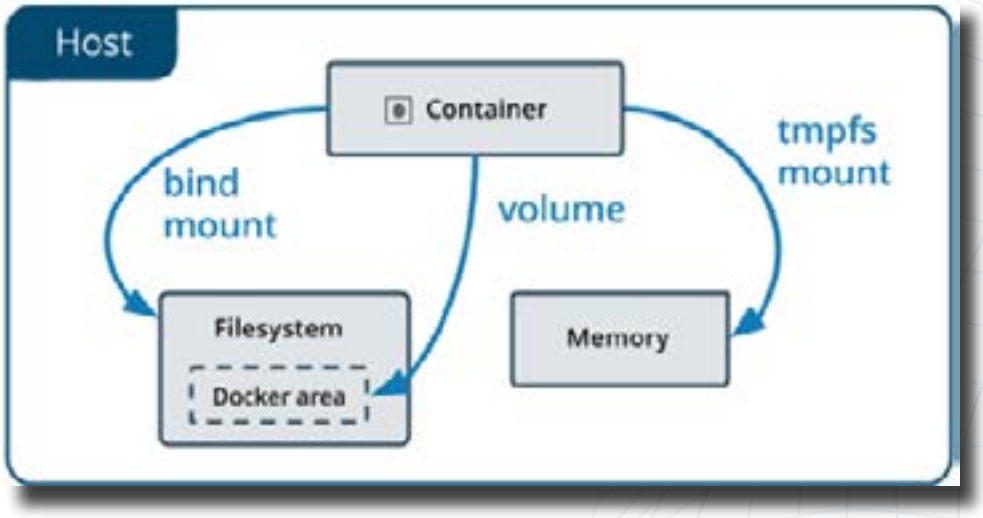
Gerenciar dados em um Docker

Como comportamento padrão, todos os arquivos criados dentro de um container são armazenados em uma camada gravável. Isso significa que os dados não persistem quando o contêiner é removido e é bastante complexo disponibilizar dados para outro processo executado fora do contêiner. Além disso, a camada gravável do contêiner é fortemente acoplada à máquina host em que o contêiner está sendo executado e não é simples mover esses dados para outro host. Por último, mas não menos importante, uma abstração extra adicional reduz o desempenho em comparação com o uso de volumes de dados, que gravam diretamente no sistema de arquivos do host.

O sistema Docker fornece duas opções para contêineres armazenarem arquivos na máquina host para que os arquivos sejam persistentes mesmo após a interrupção do contêiner: volumes e montagens de ligação.

O Docker também fornece aos contêineres uma maneira de armazenar arquivos na memória da máquina host chamada ''tmpfs''. Como a memória da máquina host é usada, obviamente esses arquivos não são persistentes, mas ainda são úteis para obter o melhor desempenho ou por motivos de segurança para proteger os dados.

9+



Os volumes são armazenados no sistema de arquivos do host em que o Docker está sendo executado. Por exemplo, em "/var/lib/docker/volumes/." Como uma boa prática, os processos não-Docker devem evitar a modificação desse sistema de arquivos. Os volumes são o mecanismo preferencial para a persistência de dados gerados e usados por contêineres do Docker.

As montagens de ligação eram a opção original para dados persistentes em um Docker. Eles podem ser armazenados em qualquer lugar no sistema de arquivos do host. Os processos que não são do Docker no host do Docker ou em um contêiner do Docker podem modificá-los a qualquer momento. Como uma boa prática, você deve usar volumes sempre que possível, no entanto, em alguns casos, você pode considerar o uso de montagens de ligação entre os processos do Docker e a máquina Host; por exemplo, se você precisar compartilhar um arquivo de configuração entre a máquina host e os contêineres.

Usando Volumes

Podemos criar um volume usando o subcomando "create" e passando um volume "name" como argumento:

```
root@ubuntu# docker volume create myAppVolume
```

O subcomando "ls" mostra todos os volumes conhecidos pelo Docker:

```
DRIVER                              VOLUME NAMES
local                               myAppVolume
```

Para exibir informações detalhadas sobre um ou mais volumes, usamos o subcomando "inspect":

```
root@ubuntu# docker volume inspect myAppVolume
[
    {
        "CreatedAt": "2022-04-21T01:18:45-07:00",
        "Driver": "local",
        "Labels": {},
        "Mountpoint": "/var/lib/docker/volumes/myAppVolume/_data",
        "Name": "myAppVolume",
        "Options": {},
        "Scope": "local"
    }
]
```

Devemos observar que o driver do volume descreve como o host do Docker localiza o volume. Os volumes também podem ser definidos para serem localizados em armazenamento remoto via NFS. No exemplo acima, o volume está no armazenamento local.

Iniciar um contêiner com um volume

Se você iniciar um contêiner com um volume que ainda não existe, o Docker cria o volume para você. O exemplo a seguir monta o volume "myAppVolume" em ":/isapplication/data" no contêiner

```
docker run --name iscobolserver -v myAppVolume:/isapplication/data -p 10999:10999
-d iscobol
```

A opção -v contém três componentes, separados por dois pontos:

- Diretório de origem ou nome do volume
- Ponto de montagem dentro do contêiner
- (Opcional) 'ro' se a montagem for somente leitura

Dentro do container você deve ter o ISAPPLICATION configurado com um arquivo de configuração "config.properties". O arquivo deve incluir iscobol.file_prefix, a propriedade que define o diretório onde o arquivo COBOL é criado e localizado, configurado para o ponto de montagem do volume definido. Por exemplo:

```
iscobol.file_prefix=/isapplication/data
```

Após a execução do contêiner Docker, você encontrará arquivos de dados de índice COBOL dentro da pasta física denominada "/var/lib/docker/volumes/myAppVolume/_data".

Arquivo Docker

Aqui está o conteúdo do dockerfile usado no artigo anterior do docker, para referência:

a imagem a ser usada para criar o contêiner:

```
# Dockerfile

FROM  openjdk:11
MAINTAINER  Veryant
ENV ISCOBOL=/var/isCOBOL2022R1
ENV ISCOBOL_CLASSPATH=${ISCOBOL}:${ISCOBOL}/isapplication
ENV ISSERVER_OPTS="-c ${ISCOBOL}/isapplication/config.properties"

RUN mkdir ${ISCOBOL}

COPY iscobol.properties ${ISCOBOL}/iscobol.properties
COPY isCOBOL_SDK2022R1/lib ${ISCOBOL}/lib
COPY isCOBOL_SDK2022R1/bin/isserver ${ISCOBOL}/bin/isserver
COPY isCOBOL_SDK2022R1/sample/isapplication ${ISCOBOL}/isapplication

WORKDIR "${ISCOBOL}/isapplication"

CMD ${ISCOBOL}/bin/isserver run
```

Integração COBOL/Web

O poder do componente IWC-PANEL.

O IWC-PANEL é um controle de contêiner que você pode usar em seu aplicativo COBOL para incluir componentes da web em sua tela quando seu programa estiver encapsulado em uma página da web. O controle IWC-PANEL permite que você use os verbos COBOL MODIFY e INQUIRE para interagir com o componente da web. Em geral, um componente do painel iwc na seção da tela se pareceria com o seguinte:

```
03 f-map iwc-panel
            js-name         "f-map"
            line  5         column  2
            size 68 cells   lines 15 cells
            value           fmap-struct
            event procedure FMAP-PROC.
```

A propriedade JS-NAME contém um identificador que será enviado para a página da web na criação, para que o componente da web correspondente possa ser criado. A propriedade VALUE do controle contém a estrutura de mensagem usada para enviar ações para o painel na página da web, então você usaria a instrução MODIFY nessa propriedade para enviar uma mensagem para a página da web. Por outro lado, se a página da web executar uma performAction no painel, o procedimento de evento do seu programa COBOL será chamado e um INQUIRE na propriedade value retornará a mensagem que foi enviada.

INFORME A PÁGINA DA WEB PARA CRIAR O COMPONENTE

Para cada IWC-PANEL em um formulário, um callback na página web é chamado, com os detalhes necessários para realizar a inicialização do componente. O Objeto webclientInstance.options.compositingWindowsListener define retornos de chamada para vários eventos, como a criação do IWC-PANEL e abertura e fechamento de janelas. Uma criação IWC-PANEL acionará o retorno de chamada windowOpened e uma referência ao IWC-PANEL é passada como argumento de função. O retorno de chamada pode verifiar a propriedade .name para detrminar qua controle foi criado e regir de acordo. Aqui está um trechi de código JavaScript mostrando o tratamento windowOpened:

```
compositingWindowsListener:{
   windowOpened: function(win) {
      if (win.name === 'f-map'){
         createMap(win)
      }
   },
},
```

ENVIAR INFORMAÇÕES PARA AS PÁGINAS DA WEB

O trecho de código a seguir mostra como o programa COBOL interage com a página da web. Uma mensagem é criada usando OOP para um jsonStream, depois passada para o JavaScript com uma simples instrução COBOL "modify f-map value …"

```
SHOW-ON-MAP.
    move "selectOffice"        to fmap-action
    move offices(office-index) to selected-office
    set objJsonStream to jsonStream:>
                         new(selected-office, 1);;
    set strbuffer     to string-buffer:>new
    objJsonStream:>writeToStringBuffer(strbuffer)
    move strbuffer:>toString to fmap-data
    modify f-map value fmap-struct.
```

O Javascript recebe e analisa os dados neste snippet:

```
if (actionName === 'selectOffice'){
    let office = JSON.parse(data);
    selectOffice(office);
 }
```

OBTENHA INFORMAÇÕES DA PÁGINA DA WEB

Uma mudança no mapa aciona uma performAction na página da web que é capturada pelo programa COBOL. O programa executa o procedimento de evento do controle, FMAP-PROC, que usa uma instrução "inquire f-map value …" para obter o evento O Javascript enviou as informações neste trecho de código:

```
infoWindow.addListener('closeclick', () =>{
   if (mapControl){
      mapControl.performAction(
        {actionName: 'pinClosed',
        data:marker.title});
   }
 });

if (mapControl){
    mapControl.performAction(
    {actionName: 'pinClicked',
     data:marker.title});
}
```

E o programa COBOL o processou no procedimento de evento do controle.

```
FMAP-PROC.
     if event-type = ntf-iwc-event
        inquire f-map value in fmap-struct
        evaluate fmap-action
        when "pinClicked"
             move fmap-datato sel-description
        ...
        when "pinClosed"
        ...
        end-evaluate

     end-if.
```

Na pasta sample/issamples de sua instalação isCOBOL, fornecemos um projeto completo que mostra como integrar um componente de mapa do Google em um aplicativo COBOL e como interagir com ele. Instruções para executar o programa SAMPLES encapsulado em uma página da web – um requisito para usar o IWC-PANEL, está no documento README.md encontrado na pasta issamples.

Destaques da Documentação

Configurações comuns de conexão JDBC

Você pode gerenciar seus dados em todos os bancos de dados compatíveis com JDBC com isCOBOL.

Oferecemos duas maneiras para você fazer isso. Isso pode ser feito escrevendo código SQL embutido (ESQL) em seu programa ou você pode usar nosso produto DatabaseBridge para escrever o ESQL para você.

Em ambos os casos, você precisará adicionar a biblioteca de driver JDBC adequada ao Classpath e configurar a string de conexão JDBC.

Para facilitar isso, a documentação do isCOBOL inclui uma coleção de drivers JDBC e cadeias de conexão para os bancos de dados relacionais mais comuns aqui.

Personalizando ícones padrão

O Framework do isCOBOL inclui uma série de ícones padrão que são usados em vários locais da GUI. Esses ícones são arquivos PNG e GIF armazenados no pacote com.iscobol.gui.client.swing na biblioteca iscobol.jar. É possível personalizar esses ícones adicionando uma biblioteca (ou uma pasta) com o mesmo pacote anterior ao iscobol.jar no CLASSPATH. Suponha que você deseja personalizar o ícone de funil mostrado no cabeçalho da Grade quando o estilo FILTERABLE-COLUMN ou a propriedade FILTER-TYPES está definida.

Vantagens da Fonte do Sistema

Usando os opcodes WBITMAP-LOAD-SYSTEM-FONT e WBITMAP-LOAD-SYSTEM-FONT-EX

As fontes de símbolos são bibliotecas de fontes. Aqui estão algumas das vantagens de usar fontes de símbolos para suas imagens em vez de arquivos de imagem externos separados. Ícones tradicionais perdem qualidade quando você aumenta ou diminui seu tamanho, ou abre e salva em um programa de edição de fotos. As fontes de ícones usadas por esses opcodes W$BITMAP armazenam imagens em formato vetorial e não perdem qualidade. Isso significa que, com fontes de símbolos:

- Se você precisar de mais de um tamanho de ícone (ex: 32 e 16 pixels), não precisará criar e manter dois arquivos de ícone separados para manter a mesma qualidade.
- Quando você adiciona um ícone ao final de uma tira, você mantém a qualidade da tira adicionando código, em vez de ter que abrir a tira em um programa de edição de fotos, degradando a qualidade.
- Se sua aplicação tem dois esquemas diferentes (ex: claro e escuro) você pode definir essas propriedades em seu código, ao invés de criar dois conjuntos de imagens com cores diferentes.
- As fontes Symbol oferecem uma enorme variedade de imagens. Por exemplo Font Awesome, uma das fontes de sistema mais populares, tem mais de 7.000 imagens.

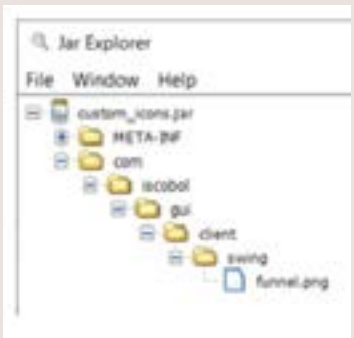
Aqui está um exemplo de código que altera o tamanho e o esquema de cores de uma faixa definida:

```
If use-big-image
   move 32 to imageWidth
else
   move 16 to imageWidth
end-if

If dark-theme
   move -13421772 to imageColor
else
   move -6710886  to imageColor
end-if

CALL "W$BITMAP" USING
      WBITMAP-LOAD-SYMBOL-FONT
                  fontHandle
                  charactersSequence
                  imageWidth
                  imageColor
            GIVING bitmapHandle.
```

Você só precisa colocar um arquivo chamado funnel.png em uma estrutura de pastas chamada com/iscobol/gui/client/swing e criar um jar a partir dele, por exemplo. Copie este jar para o diretório "jars" do seu isCOBOL SDK e, a partir de agora, ao executar um programa COBOL usando o isCOBOL SDK, você verá seu funil customizado nos cabeçalhos da grade. Consulte a documentação online aqui para obter a lista de ícones que você pode personalizar e para obter mais detalhes sobre a criação de seu jar personalizado.



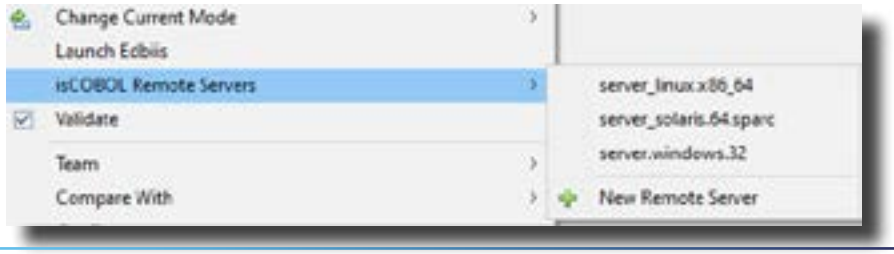
Use a IDE para e executar seus programas batch remotamente

Seu IDE pode compilar e executar seu código local ou remotamente, com o servidor remoto do IDE. Se você tiver processos em lote que precisam ser executados em um servidor remoto, com um pouco de configuração, você pode iniciá-los a partir do IDE em vez de diretamente no servidor.

Depois de iniciar o servidor de aplicativos do isCOBOL com a opção -ide especial no servidor remoto, você cria um projeto e define o servidor remoto no espaço de trabalho do seu IDE e, em seguida, vincula-os ou vinculá-los. Depois de fazer isso, você tem mais modos de compilação e tempo de execução – além do "executar" normal, você encontrará "@<nome do seu servidor>.Run", por exemplo.

Você pode usar o servidor remoto para executar mais de programas em lote. Programas com interface de usuário também funcionam bem, porque o servidor remoto usa tecnologia Thin Client.

Os detalhes estão aqui em nossa documentação e o suporte está a apenas um e-mail de distância!



Onde estão minhas licenças!?!

O runtime framework (e outros produtos isCOBOL) procura por licenças em um arquivo chamado "iscobol.properties" em 4 lugares, e um 5º lugar e nome que você especificar. A licença mais recente encontrada é usada. Aqui está a lista de locais e ordem pesquisada.

Para ver quais licenças o runtime vê, execute

```
iscrun -license
```

Você receberá uma lista de todos os arquivos "iscobol.properties" encontrados e quais licenças foram coletadas e em que ordem.

1. /etc/iscobol.properties
2. <userhome>/iscobol.properties
3. <java classpath>/iscobol.properties
4. -c <any path and name passed from the command line>
5. $ISCOBOL/iscobol.properties

veryant

JANELAS "NÃO DECORADAS"

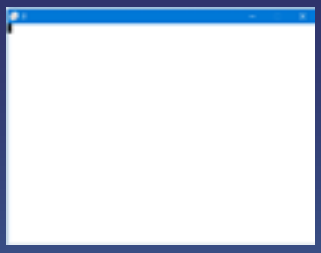
O que são e quando faz sentido usá-los.

Quando o estilo "não decorado" é definido, as decorações nativas, como o quadro e a barra de título, não são exibidas.

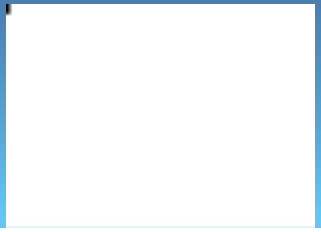
Este estilo de janelas é útil no ambiente WebClient, onde você pode querer remover a barra de título da janela para que a tela do programa pareça um formulário HTML comum ou seja incorporada em uma página.

Você pode usar esse estilo em janelas gráficas flutuantes, independentes e iniciais (padrão). Não tem efeito nas janelas de encaixe e MDI, e as janelas de notificação já não são "decoradas" por padrão.

Aqui está uma janela "decorada":



E aqui está a mesma janela com a propriedade "não decorada". Observe que a barra de título e a borda azul (quadro) desapareceram..



Evolution, without revolution

Contate-nos

Para clientes com suporte, envie-nos um e-mail para support@veryant.com

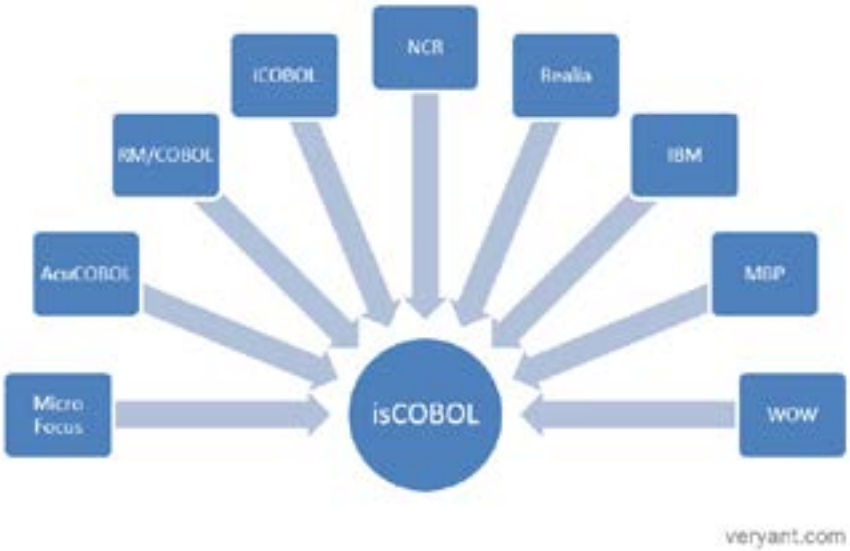
Se você quiser que a Veryant entre em contato com você para agendar um briefing técnico do produto, envie-nos um e-mail para info@veryant.com

Se você quiser que a Veryant entre em contato com você para uma cotação especial ou para os assistentes de vendas, envie-nos um e-mail para sales@veryant.com

Sede da Empresa
6390 Greenwich Dr., Suite 225
San Diego, CA 92122 - USA
Tel (English): +1 619 797 1323
Tel (Español): +1 619 453 0914
info@veryant.com

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

Como sempre, 2022R1 contém várias adições de compatibilidade – à medida que continuamos a tornar seu processo de conversão o mais suave, rápido e sem problemas possível.



Follow **Veryant** on

veryant.com