Progress®



OpenEdge Frequently Asked Questions

OpenEdge®

Copyright

© 2021 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

These materials and all Progress® software products are copyrighted and all rights are reserved by Progress Software Corporation. The information in these materials is subject to change without notice, and Progress Software Corporation assumes no responsibility for any errors that may appear therein. The references in these materials to specific platforms supported are subject to change.

DataDirect Cloud, DataDirect Connect, DataDirect Connect64, DataDirect XML Converters, DataDirect XQuery, DataRPM, Defrag This, Deliver More Than Expected, DevReach (and design), Icenium, Inspec, Ipswitch, iMacros, Kendo UI, Kinvey, MessageWay, MOVEit, NativeChat, NativeScript, OpenEdge, Powered by Chef, Powered by Progress, Progress, Progress Software Developers Network, SequeLink, Sitefinity (and Design), Sitefinity, Sitefinity (and design), SpeedScript, Stylus Studio, Stylized Design (Arrow/3D Box logo), Styleized Design (C Chef logo), Stylized Design of Samurai, TeamPulse, Telerik, Telerik (and design), Test Studio, WebSpeed, WhatsConfigured, WhatsConnected, WhatsUp, and WS_FTP are registered trademarks of Progress Software Corporation or one of its affiliates or subsidiaries in the U.S. and/or other countries.

Analytics360, AppServer, BusinessEdge, Chef Automate, Chef Compliance, Chef Desktop, Chef Habitat, Chef WorkStation, Corticon.js, Corticon Rules, Data Access, DataDirect Autonomous REST Connector, DataDirect Spy, DevCraft, Fiddler, Fiddler Everywhere, FiddlerCap, FiddlerCore, FiddlerScript, Hybrid Data Pipeline, iMail, JustAssembly, JustDecompile, JustMock, KendoReact, NativeScript Sidekick, OpenAccess, PASOE, Pro2, ProDataSet, Progress Results, Progress Software, ProVision, PSE Pro, Push Jobs, SafeSpaceVR, Sitefinity Cloud, Sitefinity CMS, Sitefinity Digital Experience Cloud, Sitefinity Feather, Sitefinity Insight, Sitefinity Thunder, SmartBrowser, SmartComponent, SmartDataBrowser, SmartDataObjects, SmartDataView, SmartDialog, SmartFolder, SmartFrame, SmartObjects, SmartPanel, SmartQuery, SmartViewer, SmartWindow, Supermarket, SupportLink, Unite UX, and WebClient are trademarks or service marks of Progress Software Corporation and/or its subsidiaries or affiliates in the U.S. and other countries. Java is a registered trademark of Oracle and/or its affiliates. Any other marks contained herein may be trademarks of their respective owners.

July 2021

Product version: Progress OpenEdge 12.4

Updated: 2021/06/29

Table of Contents

Java 11 support for OpenEdge FAQ.....7

Runtime support with .NET Core FAQ.....9

1

Java 11 support for OpenEdge FAQ

OpenEdge 12.2 supports Java 11. The upgrade from Java 8 to Java 11 raises some frequently asked questions that may affect your installation. Questions include JDK compatibility, minimum OS requirements, OpenEdge products and adapters that are affected by this upgrade.

Where can I get a list of compatible JDKs?

For information about JDKs certified to work with OpenEdge, see the OpenEdge 12 Platform and Product Availability Guide.

What is the minimum OS for Solaris to support Java 11?

Solaris 11.4 OS and Oracle Developer Studio 12.6 Compiler are the minimums required.

What is the minimum OS for Linux to support Java 11?

Linux 7.6 OS is the minimum OS required.

Are there release notes for Solaris and AIX regarding missing fonts?

Because Java 11 no longer includes fonts, Solaris and AIX require additional configuration to locate the necessary fonts. For more information, see the OpenEdge 12.2 Release Notes.

Does Aurea Messenger ESB support Java 11?

No, Aurea Messenger ESB, formerly Sonic ESB, requires Java 8. If you use Aurea Messenger ESB, then you need to install JDK 8 during the installation of the adapter.

For more information about installation, see 'Before you install' and 'Install OpenEdge ESB Adapter interactively' in *Install OpenEdge*.

What if my third-party JMS messenger only runs on Java 8?

The Progress OpenEdge JMS adapter can operate with any JMS-compliant vendor such as Aurea Messenger MQ (formerly SonicMQ), IBM WebSphereMQ, and ActiveMQ. It is possible to use the adapter with Java 8. During installation, supply a Java 8 JDK. For more information about installation, see 'Install the Progress OpenEdge JMS Adapter' in *Use the Progress OpenEdge JMS Adapter*.

How do I install a new JDK in an existing OpenEdge environment?

After you move or install a new JDK, you must make the JDK available to your OpenEdge installation. For more information about using a new JDK, see 'Upgrade the JDK after Installation' in *Install OpenEdge*.

Are there changes required for Java Open Clients to support Java 11?

If you use Java Open Clients, then you need to include an extra JAR file (`messages.jar`) in your Java applications. Additionally, the path names have been updated in 12.2. For more information on the required JAR files and updated path names, see 'Java Open Client runtime package' in *Java Open Clients*.

Are there release notes for the JAVA 11 memory leak on Solaris when using encryption/TLS with PAS for OpenEdge?

The underlying security provider for Oracle Java 11 on Oracle Solaris 11 causes memory leaks. These leaks may be an issue for PAS for OpenEdge instances that are using encryption or TLS. For more information, see the OpenEdge 12.2 Release Notes.

2

Runtime support with .NET Core FAQ

Which versions of .NET Core are now supported in OpenEdge 12.4?

Open Client for .NET supports .NET Framework 4.8, .NET Core 3.1, and .NET Core 5.0.

What platforms can I run .NET Core Open Client applications on?

Open Client applications can be run on any platform that supports .NET standard 2.0 targeted libraries. This means that they can be run on Linux or Windows using .NET core 3.1, and on Windows using .NET framework 4.8.

What .NET Core libraries do I need for my Open Client applications?

In addition to the Progress provided `Progress.Messages.dll` and `Progress.o4glrt.dll`, .NET Open Client applications now require the `System.Configuration.ConfigurationManager.dll` version 4.7 package. This can be automatically retrieved from NuGet by including a reference to this package in your application project.

How do I build an existing Open Client application to target .NET Core?

You can use `msbuild.exe` from the .NET Software Development Kit or from Visual Studio or the .NET Core Software Development Kit. You cannot use the old .NET Framework Software Development Kit, which makes use of the csc compiler.

What changes to ProxyGen have been made to support .NET Core?

The csc compiler is no longer called by ProxyGen to compile proxy libraries. ProxyGen requires the use of the MSBuild compiler. You will need to have the .NET Desktop Development Component of Visual Studio or MSBuild Tools installed. If you have an existing `.xpxg` file that uses a custom compiler, you will need to open the `.xpxg` file in the GUI ProxyGen tool and update the compiler settings to use MSBuild. If you do not migrate the `.xpxg` file to MSBuild, bproxygen will fail. If the `.xpxg` file uses the default compiler, then ProxyGen will attempt to locate MSBuild automatically.

Proxygen 12.4 now generates .NET standard targeted libraries. Progress recommends that in your ProxyGen `.xpxg` file you make use of the strong-named and signed libraries, as this recommendation follows Microsoft's recommended best practice. Proxygen generates and uses standard Microsoft Visual Studio SDK style project files for compilation of the proxygen libraries.

The Open Client libraries (`Progress.o4glrt.dll`, and `Progress.Messages.dll`) and the proxy libraries generated by ProxyGen are configured to target MSIL compatible bytecode. These libraries can be used with either .NET core applications or .NET framework applications on 32-bit or 64-bit .NET runtimes.

The custom compiler page in the GUI ProxyGen tool has some user interface changes to support MSBuild instead of csc. Visual Studio 15.3 is the minimum version needed to compile .NET standard libraries. For more information see the .NET standard overview.

If I continue using the OpenClient Toolkit with Microsoft .NET Framework, what changes do I need to make?

If your .NET application is targeting the .NET Framework, you do not have to change anything. You can update your app by including library references to the updated Open Client libraries and any generated proxy libraries.

If your .NET application is targeting .NET Core, then you will need to ensure your project has a package reference to the `System.Configuration.ConfigurationManager` version 4.7. You can add this package reference to your applications `csproj` file from within Microsoft Visual Studio or the .NET software development kit. Download the required package here.

How do I install .NET Core 3.1 in an existing OpenEdge environment?

You can download .NET Core from the .NET download site.