ELEVATE 2024

Open-Source PowerBuilder Foundation Classes

Bruce Armstrong, Integrated Data Services
December 2-3, 2024

DISCLAIMER

This presentation was authored by volunteer(s) in the Appeon community. This is not a work for hire by Appeon. The views and opinions expressed in this presentation are those of the author(s).

Any reference to third-party materials, including but not limited to Websites, content, services, or software, has not been reviewed or endorsed by Appeon. YOUR USE OF THIRD-PARTY MATERIALS SHALL BE AT YOUR OWN RISK.

Appeon makes no warranty of any kind, either express or implied, including but not limited to, the implied warranties of merchantability, fitness for a particular purpose, or non-infringement. Appeon assumes no responsibility for errors or omissions.

Session Agenda

- Why use the Open-Source PowerBuilder Foundation Class (PFC) libraries?
- Why upgrade to the new PFC?
- How to upgrade to the new PFC?
- How to migrate an existing application to use PFC?
- How to contribute to PFC?

Presenter Profile

Bruce Armstrong

https://www.linkedin.com/in/bruceaarmstrong/

Recent Projects

- OpenSourcePFCLibraries - The open-source PowerBuilder Foundation Class libraries

Key Skills

- PowerBuilder
- Oracle PL/SQL
- Java
- C#
- C++

Company Profile

Integrated Data Services

IDS has delivered innovative program and financial management information system solutions to the Federal Government for nearly three decades. Our flagship Enterprise Requirements Management product, CCaR™ is used by the DoD and Federal Agencies including the Air Force, Space Force, Army, USSOCOM, DSCA, DHA, JSF and DOE.

Why use the PFC framework

History

- Originally developed by Sybase and released with PowerBuilder 5.0 in 1995.
- Released by Sybase as an open-source project with the release of PowerBuilder 10 in 2005.
- Originally maintained on Sybase CodeXchange
- Moved to Microsoft CodePlex in 2014 when SAP/Sybase abandoned CodeXchange
- Moved to GitHub in 2017 when Microsoft abandoned CodePlex
- Currently located at https://github.com/OpenSourcePFCLibraries

Why use the PFC framework

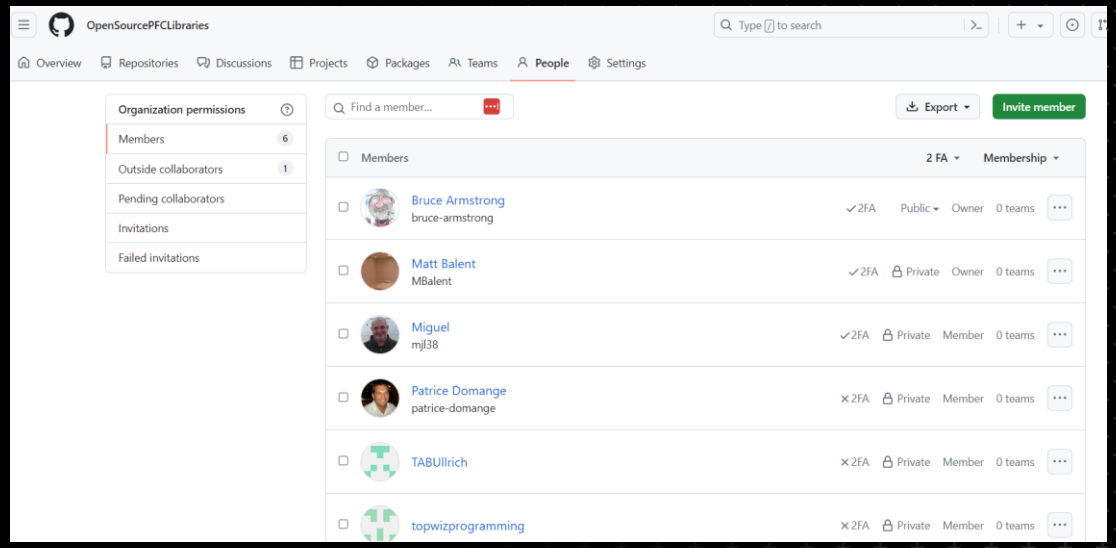
Advantages of open-source software[1]

- Agility
  - You don’t have to wait for the vendor to develop capability, you can add it yourself
- Speed
  - You don’t have to develop the base infrastructure for your project, you build on the infrastructure already provided
- Cost-effective
  - You don’t have to pay your developers to develop the base infrastructure, only for what you build on top of it
- Lower learning curve for new hires
  - They are likely to have used the framework in prior projects
- Share maintenance costs
  - You can report issues you’ve encountered and members of the open-source community will often provide a fix

1 Adapted from https://enterprisersproject.com/article/2015/1/top-advantages-open-source-offers-over-proprietary-solutions

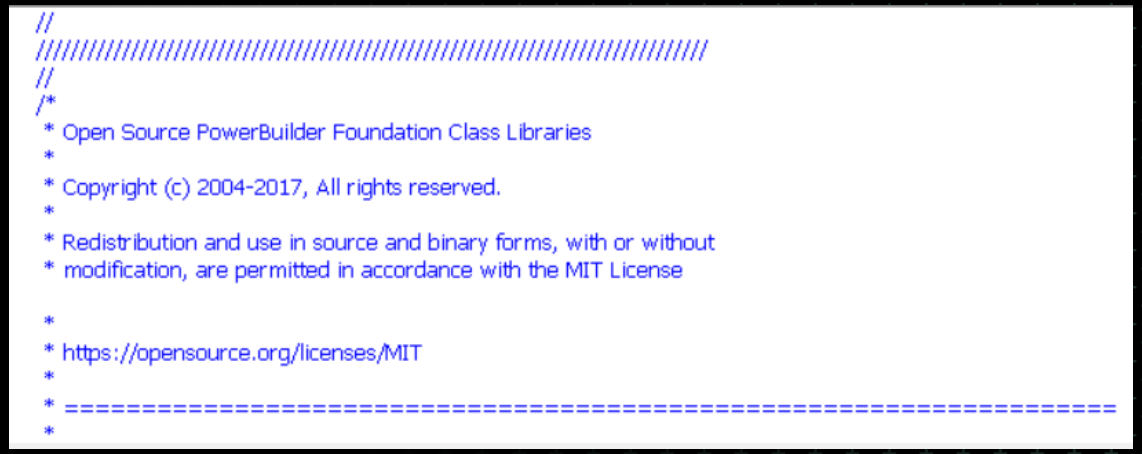
Why use the PFC framework

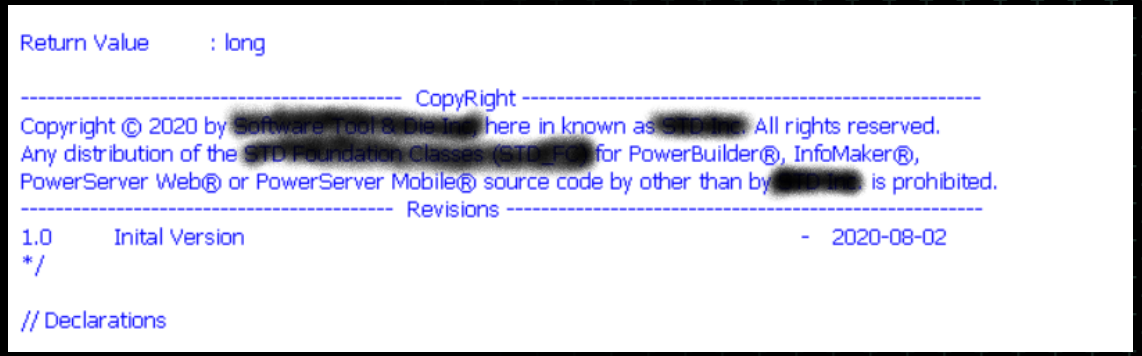
- Supported by a community of developers
  - One of the members of the community leaves the project it is less likely to impact the longevity of the project
  - Open-source projects that are maintained by only one developer are at risk of collapse of that developer stops supporting the project



Why use the PFC framework

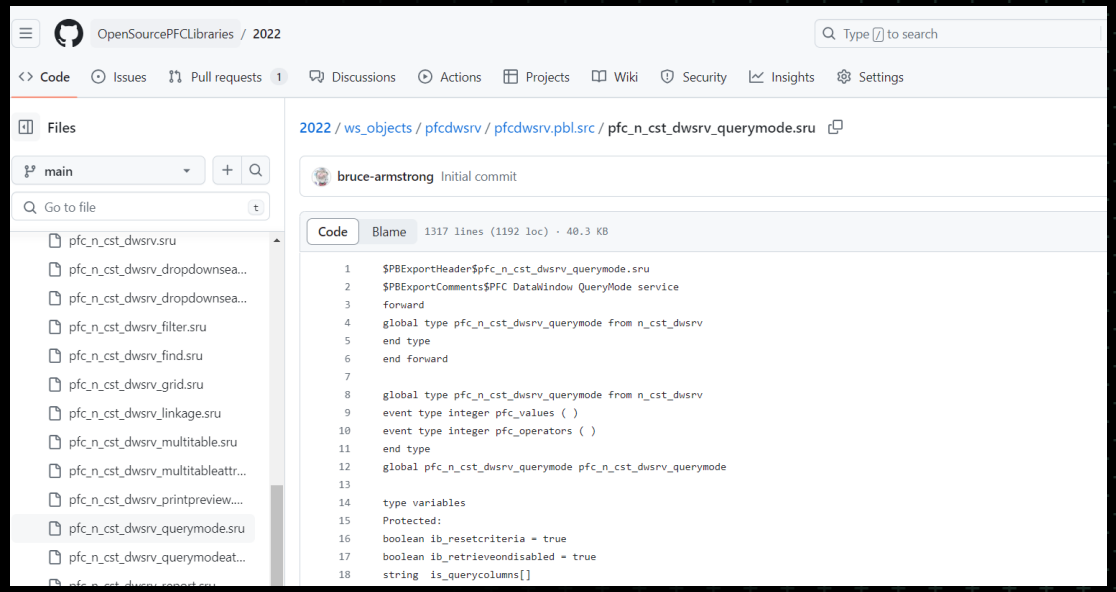
- Permissive licensing
  - The open-source PFC libraries are licensed under one of the most permissive open-source licenses available: The MIT License.
  - Other so called open-source libraries may provide a permissive license that they indicate they are licensed under, but examination of the copyright notices in the source code itself may indicate otherwise.





Why use the PFC framework

- The source code can be read directly on the internet
  - Such transparency helps ensure that any contributions to the project are not somebody else's intellectual property



Why upgrade to the new PFC

- Sample databases now provided in SQL Anywhere 17 and SQL Server 2022 exports
- Application code has been updated to ensure it works with both of those databases
- Removed obsolete objects intended to support AIX, HPUX, Solaris and 32 bit windows
- Added PFC wrapper objects for new PowerBuilder objects and controls such as CoderObject, CompressorObject, CryptorObject, ExtractorObject, HttpClient, RestClient, Animation control, Datepicker control, MonthCalendar control, RibbonBar control and WebBrowser control

Why upgrade to the new PFC

- n_cst_apppreference – added option to allow application configuration to be stored in INI file, XML file or registry
- n_cst_color – added 50+ additional color constants based on PowerBuilder values
- n_cst_debug – extended to allow DataWindow data to be viewed as JSON for debugging
- n_cst_colorconv – new object which performs conversion of color values between rgb, xya, yxy, hunterlab, cielab, deg2rad, hsl, hal, hsv, rad, hue, cmy and cielch
- n_cst_conversion – extended to handle 240 more PowerBuilder types

Why upgrade to the new PFC

- n_cst_environment – new object to return information about host system
- n_cst_filesrv – extended with methods to create directory trees, get system directory, get short path name, get long path name, get windows directory, move files, get temp file path and get temp file name
- n_cst_inifile – extended with method to create new empty INI file
- n_cst_json – new object for handling JSON text and files (largely obsolete now because of PowerBuilder native JSONParser, JSONPackage and JSONGenerator objects)

Why upgrade to the new PFC

- n_cst_measureconversion – new object to do conversions between metric to English, metric to metric and English to English for distance, weight, temperature, pressure and force
- n_cst_metadata – extended to add methods to return initial values, scripts, functions, events, variables, arguments, etc.
- n_cst_numerical – extended to add methods to do bitwise operations and handle hex, octal and binary numbers
- n_cst_pbunitconversion – new object to handle conversions between inches, twips, pbunits, pixels and centimeters

Why upgrade to the new PFC

- n_cst_platform – extended with methods to get system folder paths, system colors, dpi, system metrics, free memory and physical memory
- n_cst_string – extended with methods to determine if an email address is well formatted, perform a global search and replace, remove punctuation or other special characters and to URL encode a string
- n_cst_resize – extended with options to position controls in center, center top, center bottom, center left and center right
- n_cst_winsrv_style – new object to provide extended styling capability for windows
- n_ds – extended by adding of_printsetup function

Why upgrade to the new PFC

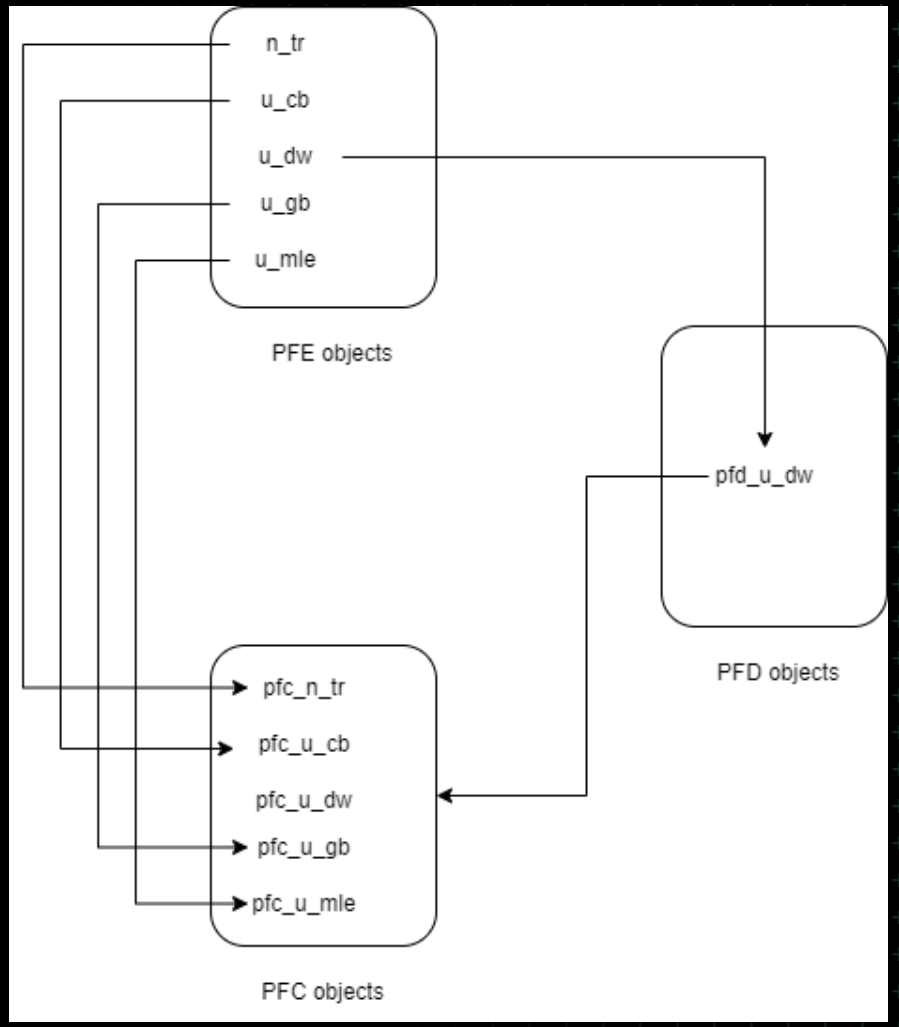
- n_st_splitbar – enhanced to handle new PowerBuilder control types and improve movement handling
- u_htb – extended by adding pfc_positionchanged event that can be scripted to respond to changes in position of the track bar
- u_vtb – extended by adding pfc_positionchanged event that can be scripted to respond to changes in position of the track bar
- w_master – extended by adding pfc_preprint and pfc_postprint events
- w_response – extended by adding user resize capability
- Numerous bug fixes

Why upgrade to the new PFC

- Database files, images, help files moved into separate repositories
- Help files recompiled in CHM and PDF
  - ShowHelp calls in PFC project updated to open CHM files.
- Visual Expert run on the project and the majority of “Critical” and “Major” issues addressed

How to upgrade to the new PFC

- PFC layer objects have all the main source code
  - Your application should not refer to these object and you should not edit them
- PFE (extension) layer objects inherit from PFC layer objects but have no source code.
  - These are the objects that should be referenced in your application code.
- PFD layer objects are created as needed to extend PFC objects
  - This is where you add your framework (not application specific) customizations



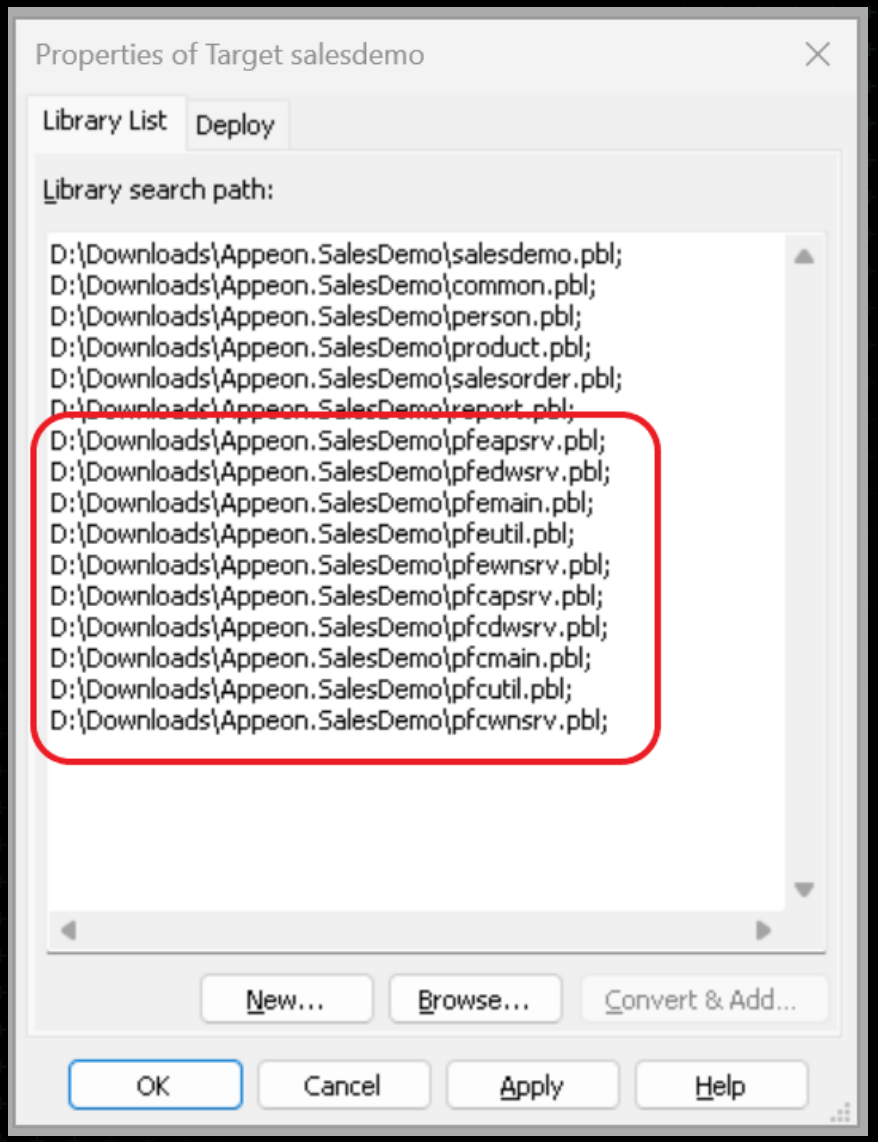
How to upgrade to the new PFC

- If you created a PBD layer
  - Replace the PFC and PFE layers with the current version
  - Modify the objects in the PFE layer for which there are ancestors in PFD
- If you edited the PFE layer
  - Replace the PFC layer with the current version
  - Compare your PFE layer with the current version using a tool such as pblcompare for added/removed/modified objects
- If you edited the PFC layer
  - Use a utility such as pblcompare to compare the PFC layers and apply changes
  - Consider making such changes in the PFD or PFE layer to facilitate later updates from the main Open-Source PFC project

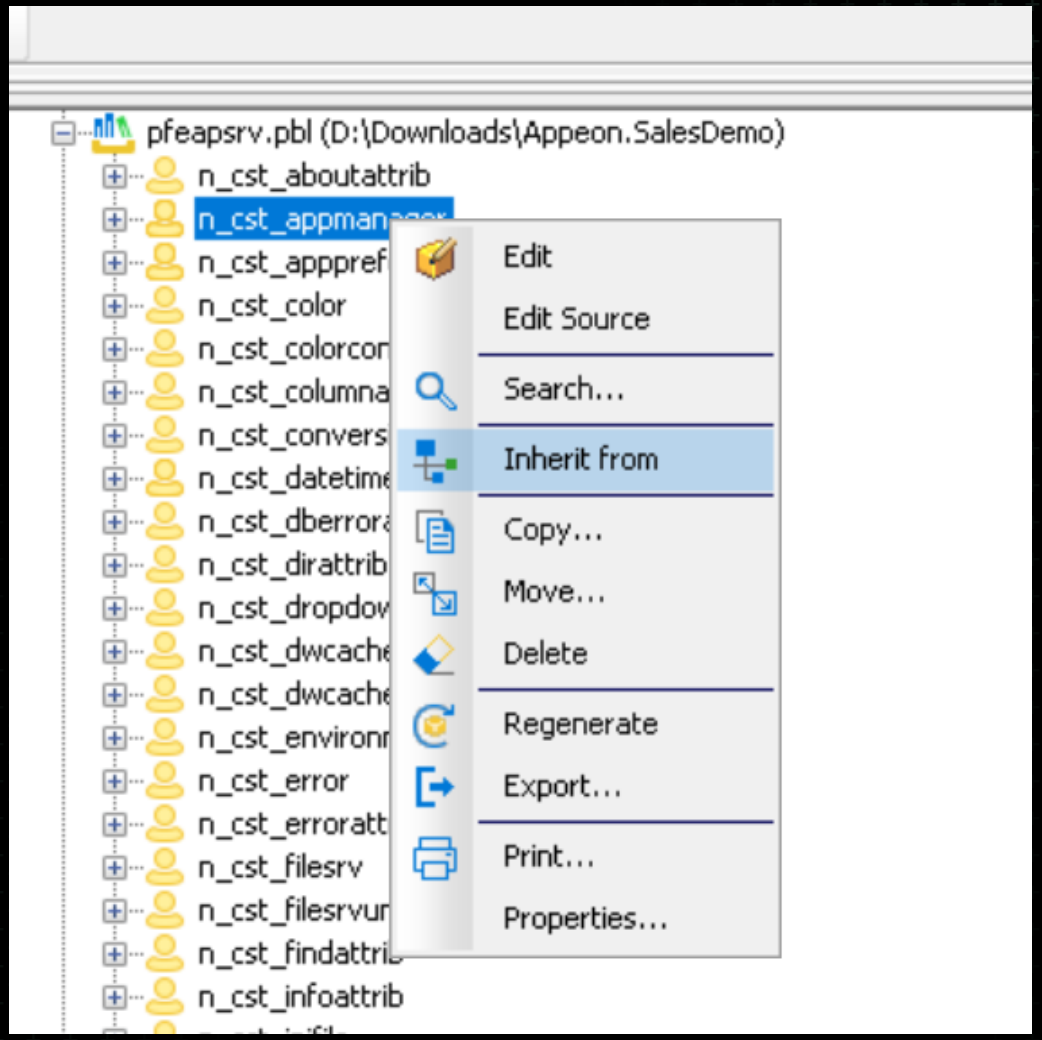
How to migrate an existing application to use PFC?

- Add PFC libraries to the application library list
- Inherit an application specific application manager from the n_cst_appmanager object in PFC
- Declare the application specific application manager as a application global variable
- Update the application variable types to use PFC classes
- Deal with any object name collisions
- Update window and user object ancestries
- Modify application object events
- Code the application specific application manager constructor and pfc_open events
- Start using PFC services

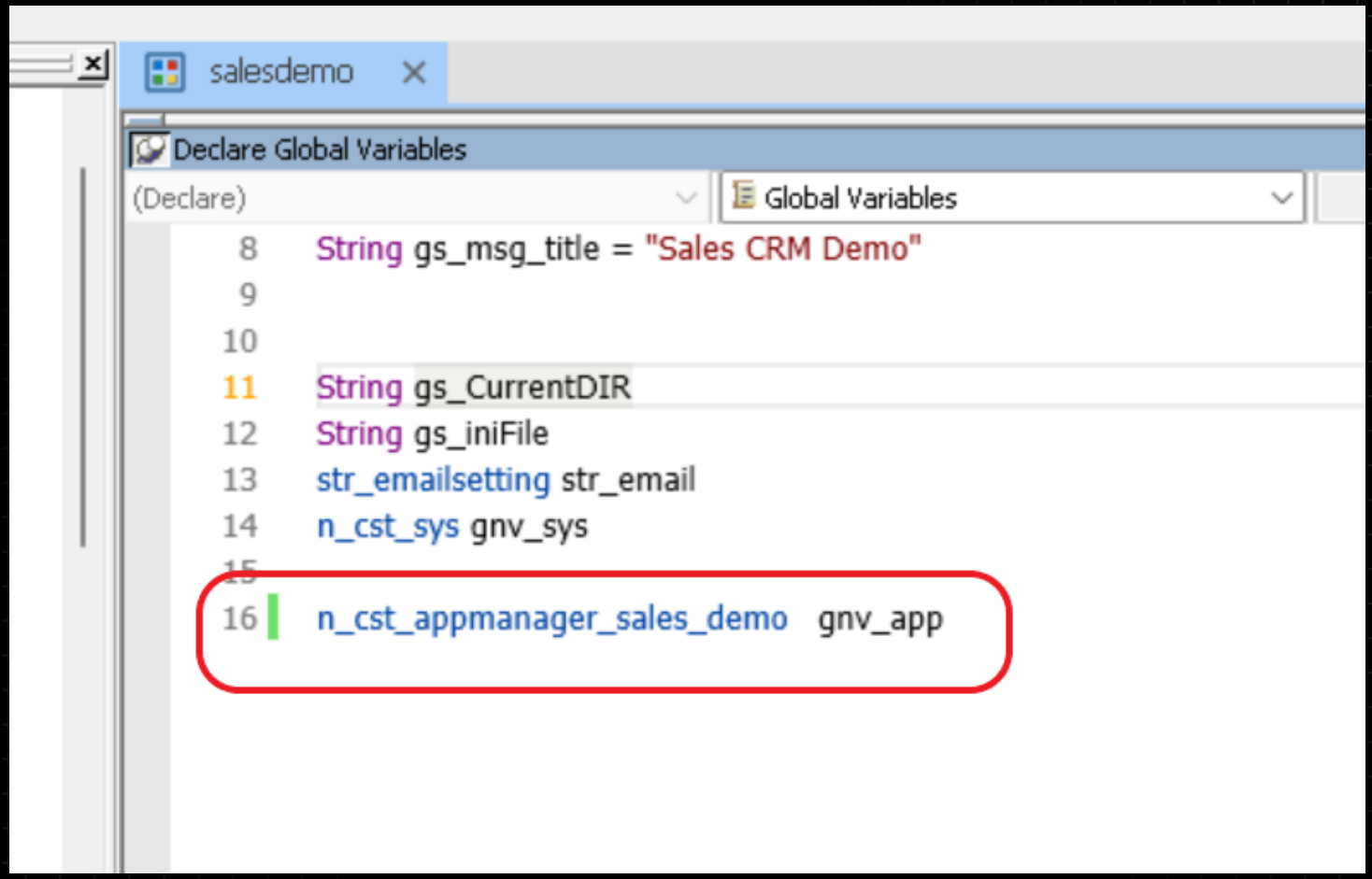
Add PFC libraries to the application library list



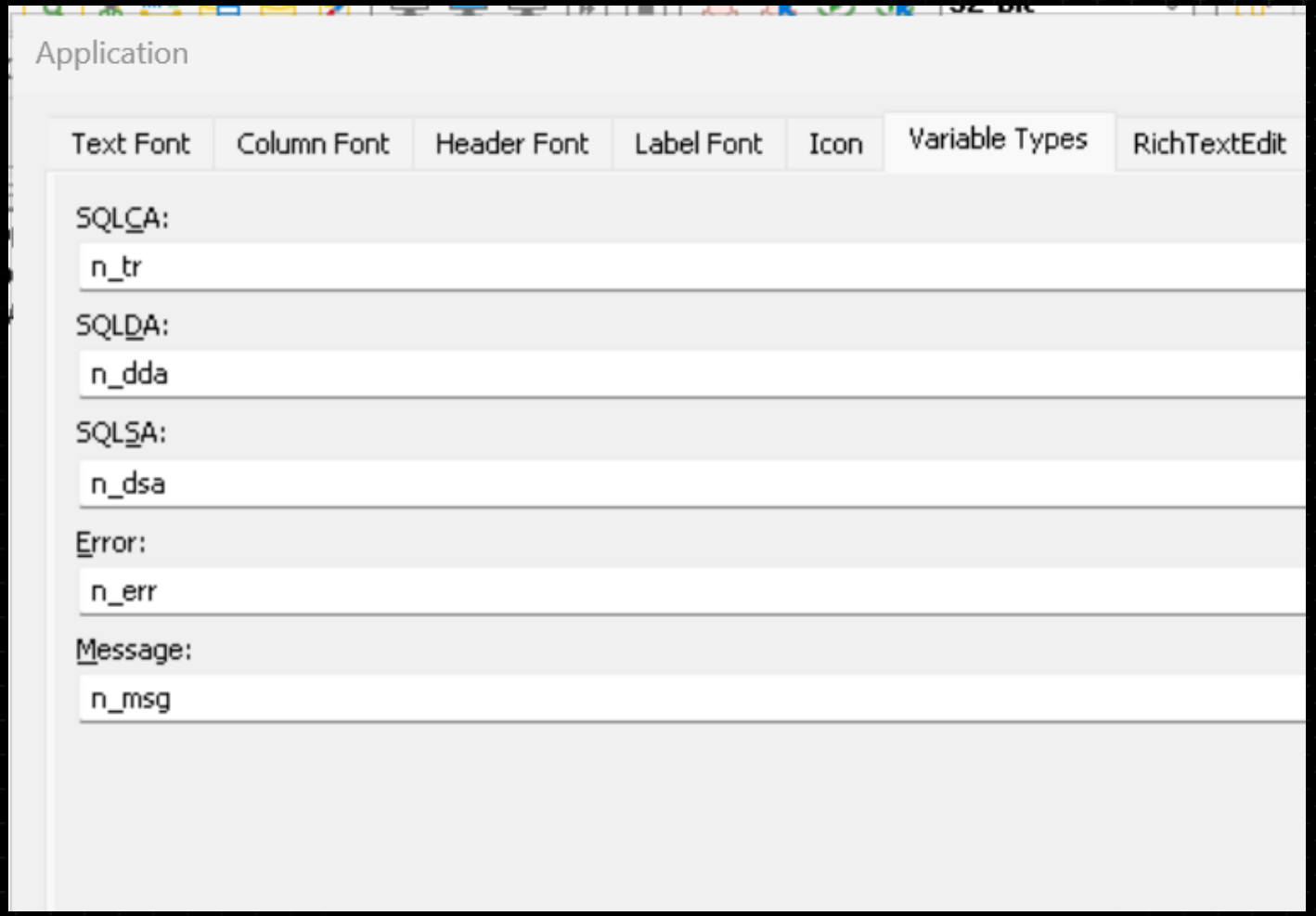
Inherit an application specific application manager from the n_cst_appmanager object in PFC



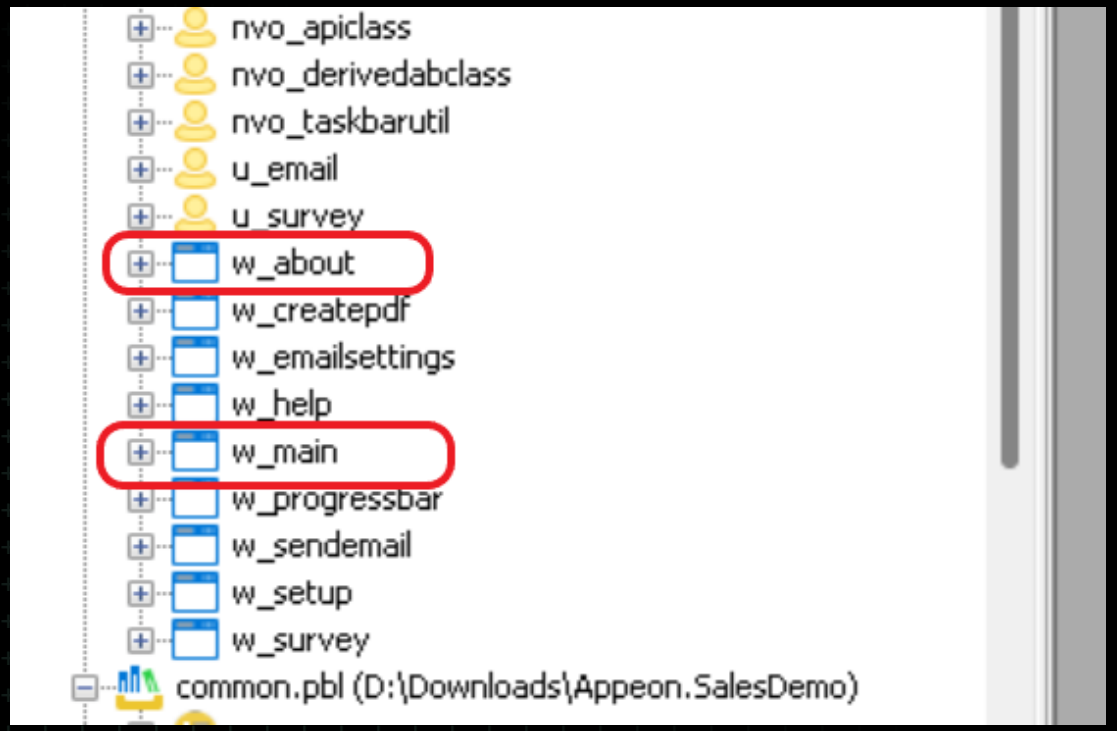
Declare the application specific application manager as a application global variable

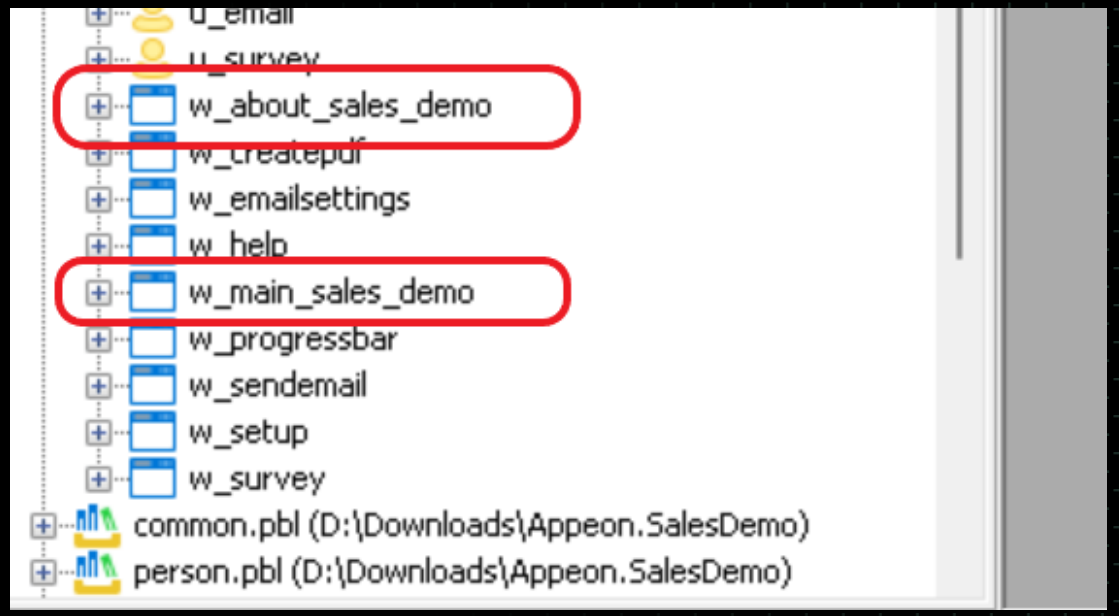


Update the application variable types to use PFC classes

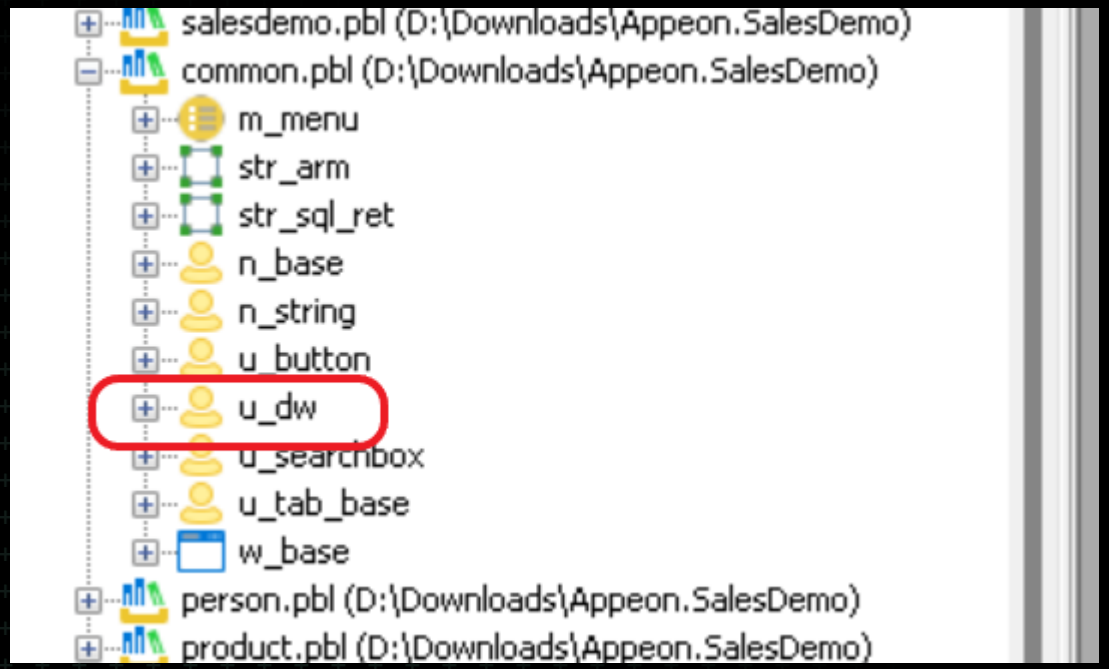


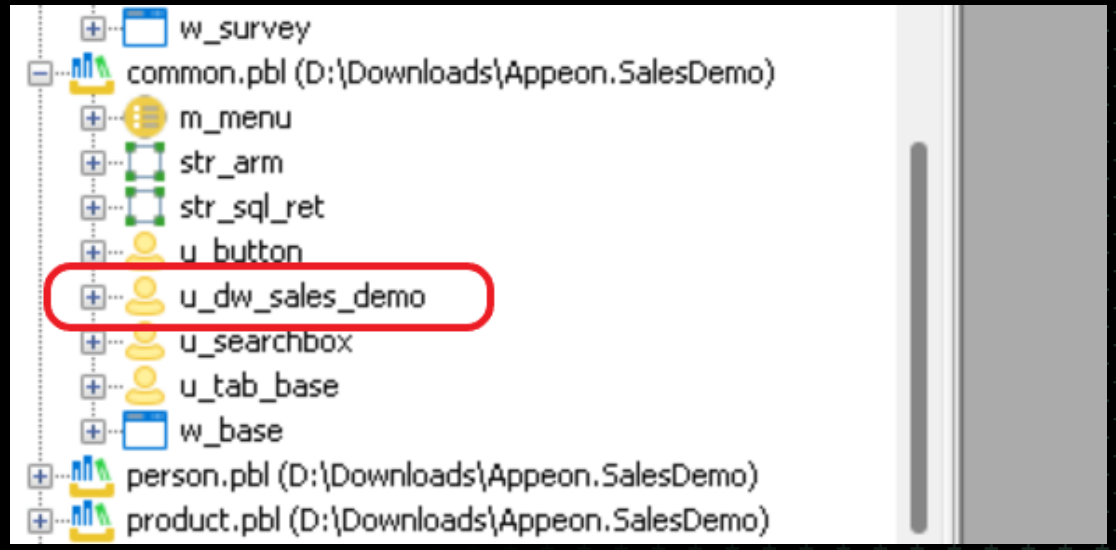
Deal with any object name collisions



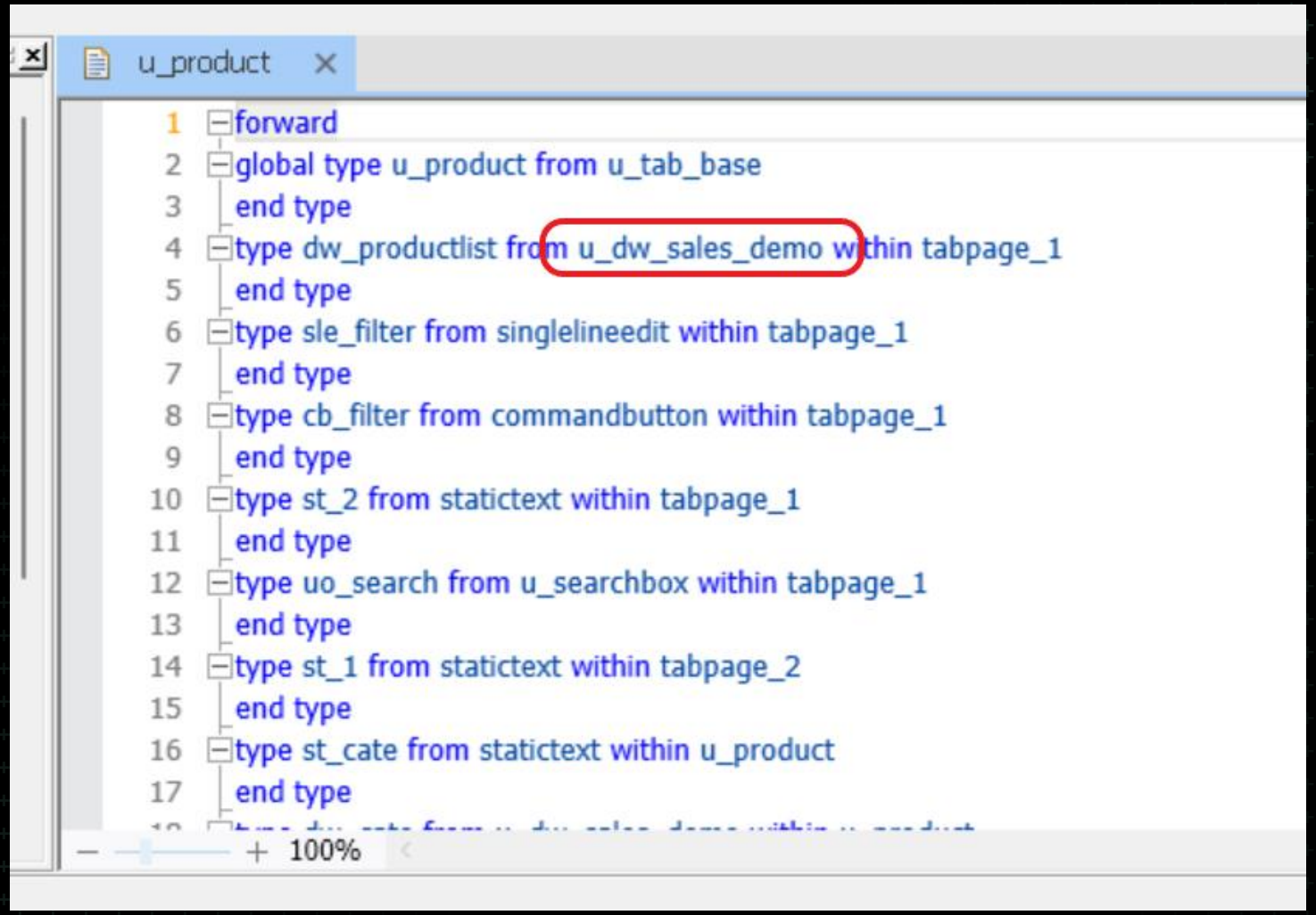


Deal with any object name collisions

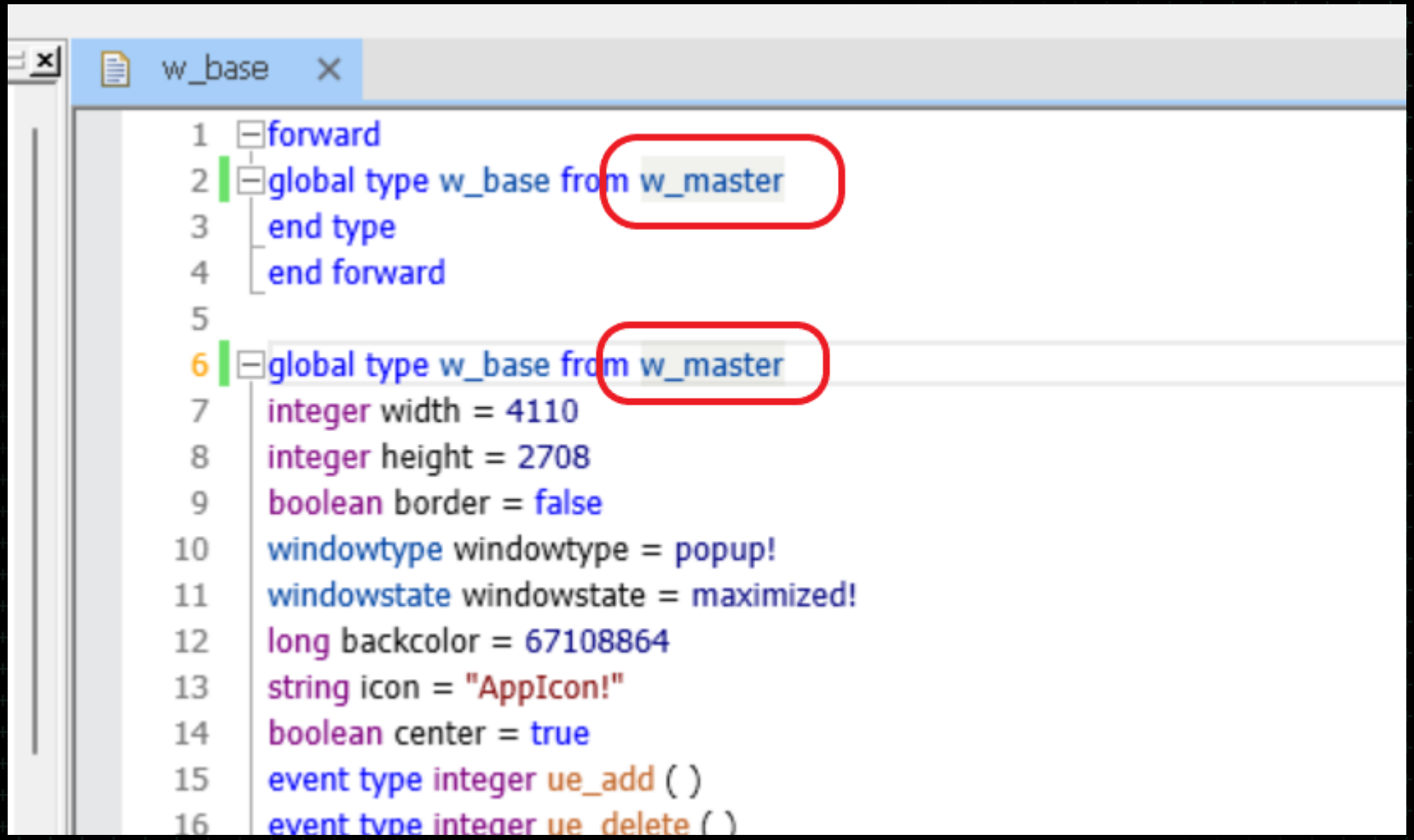




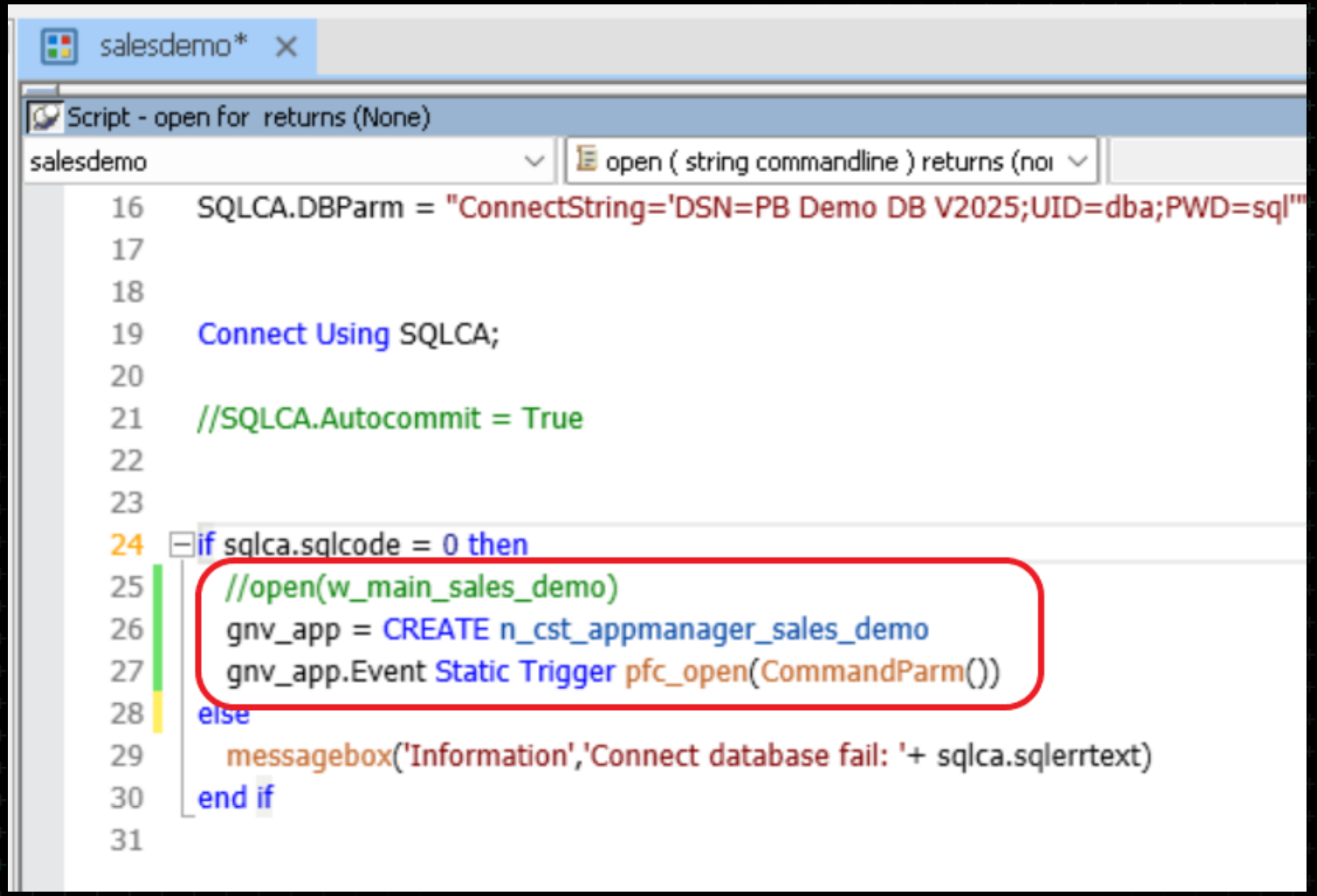
Deal with any object name collisions



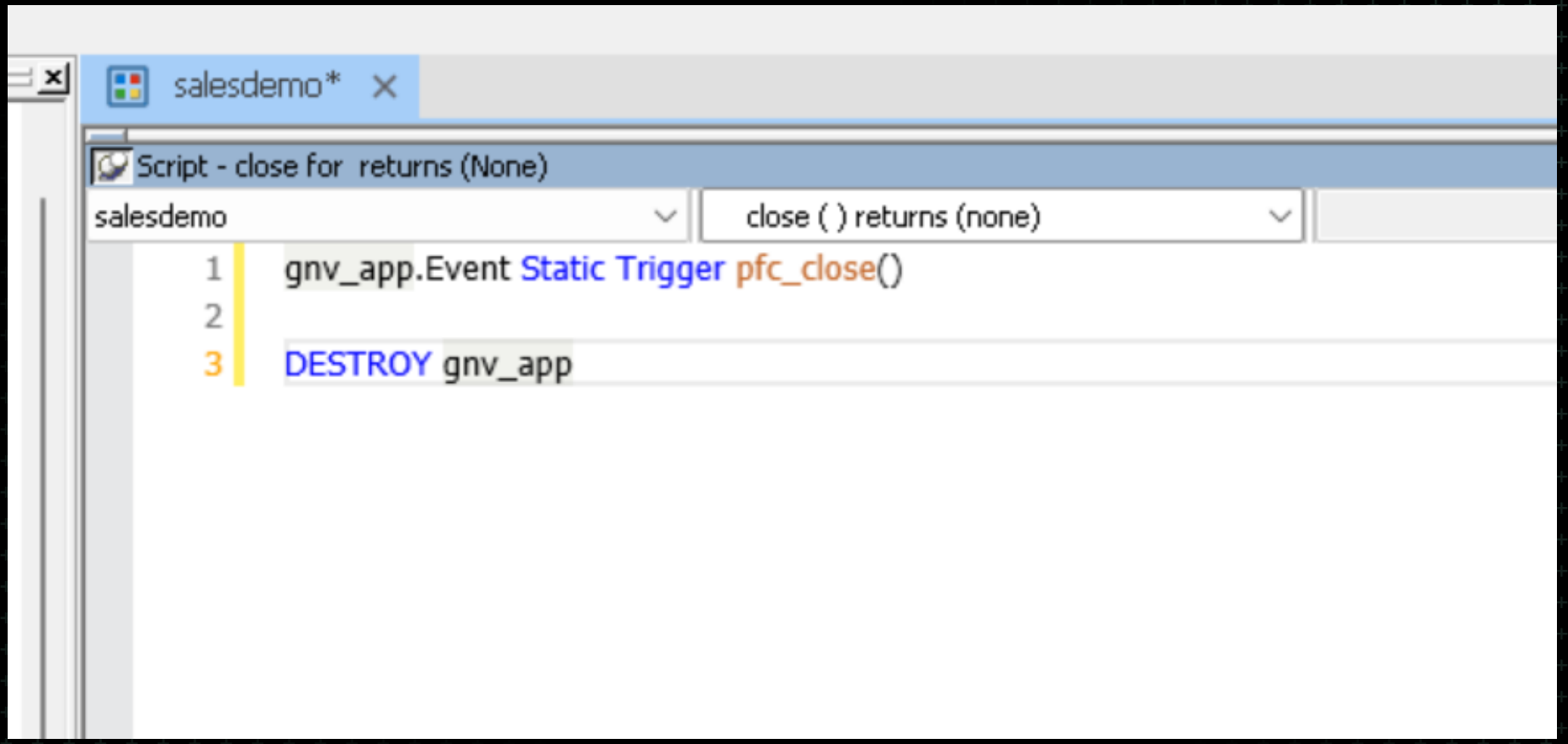
Update window and user object ancestries



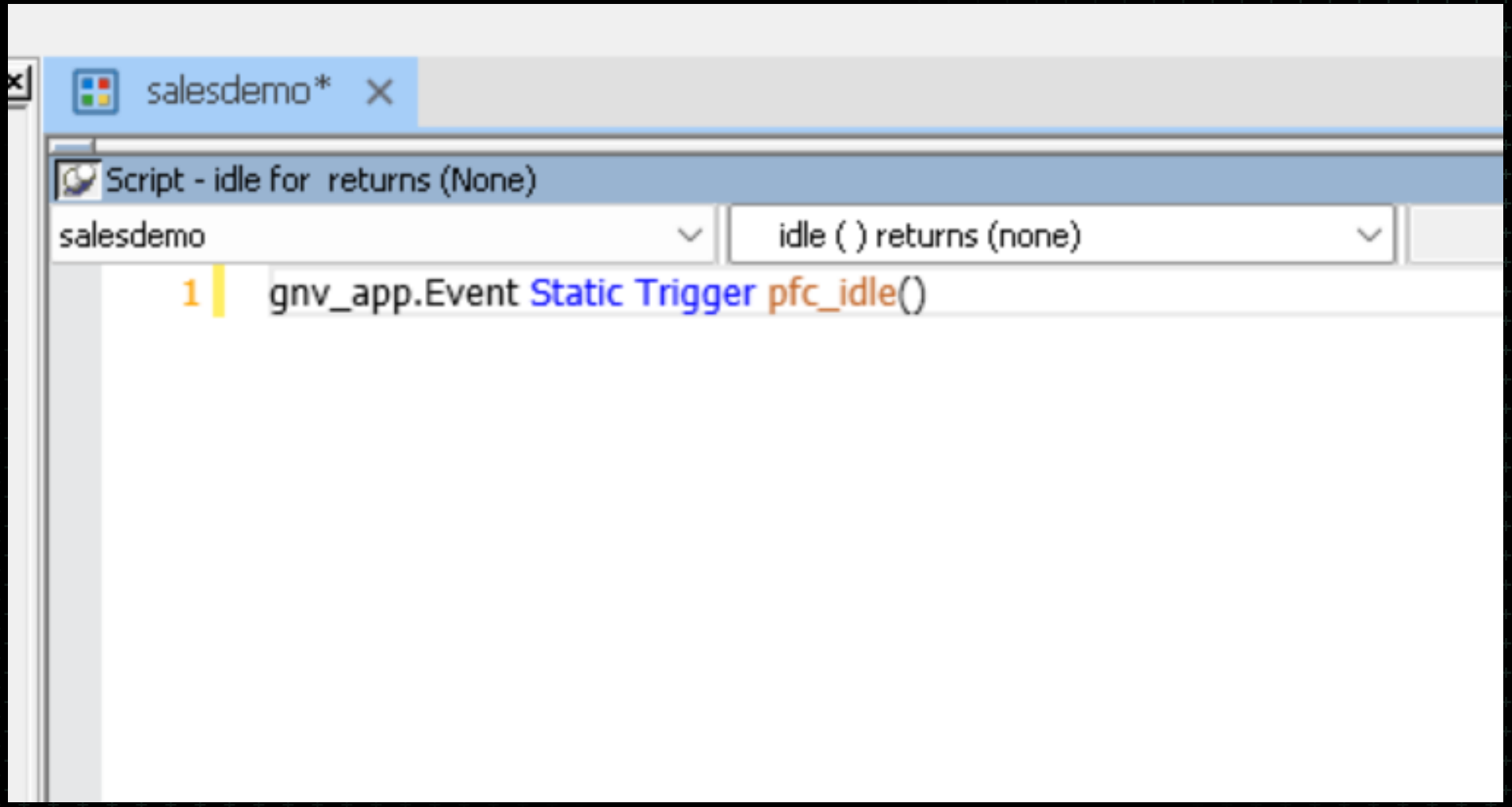
Modify application object events



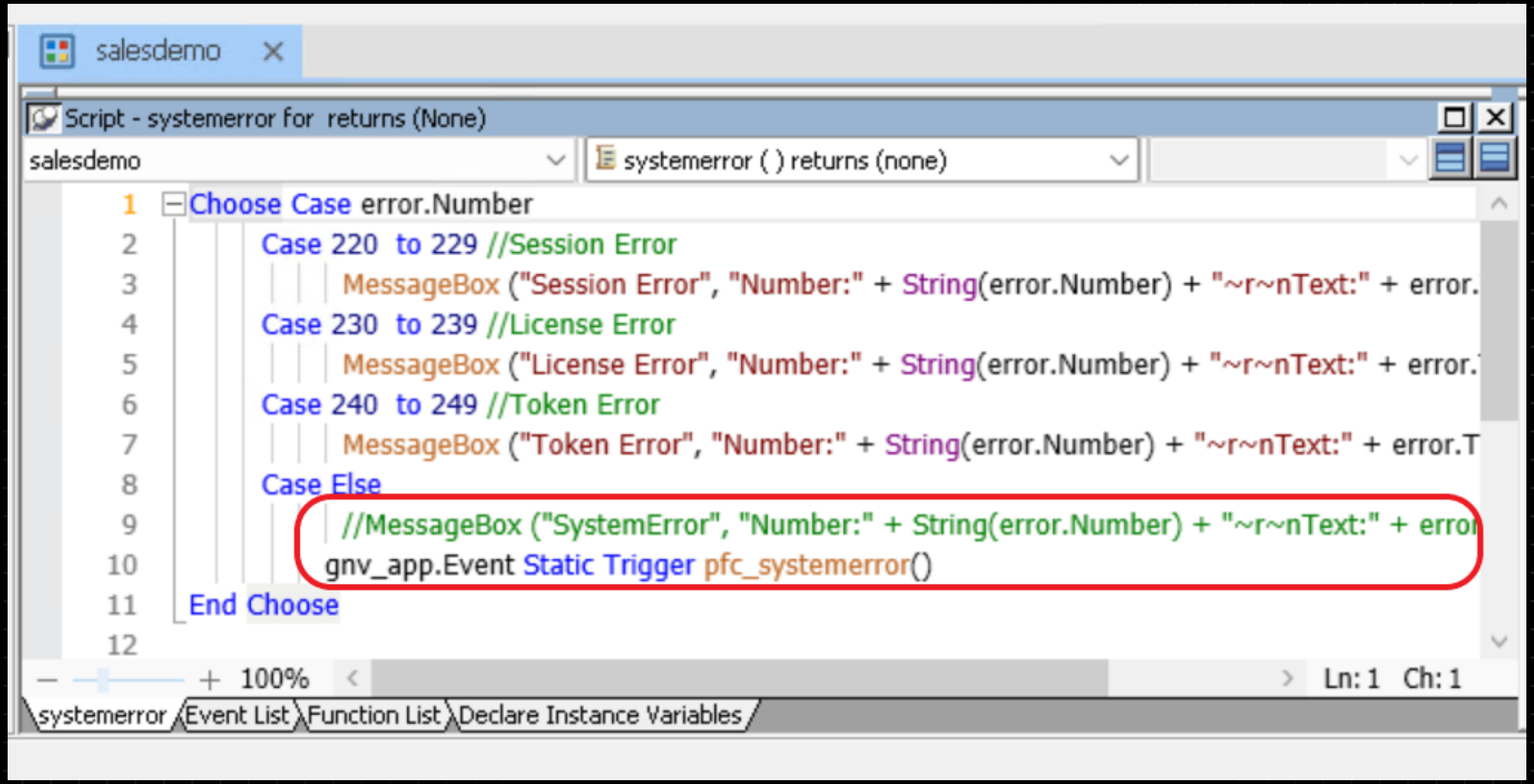
Modify application object events



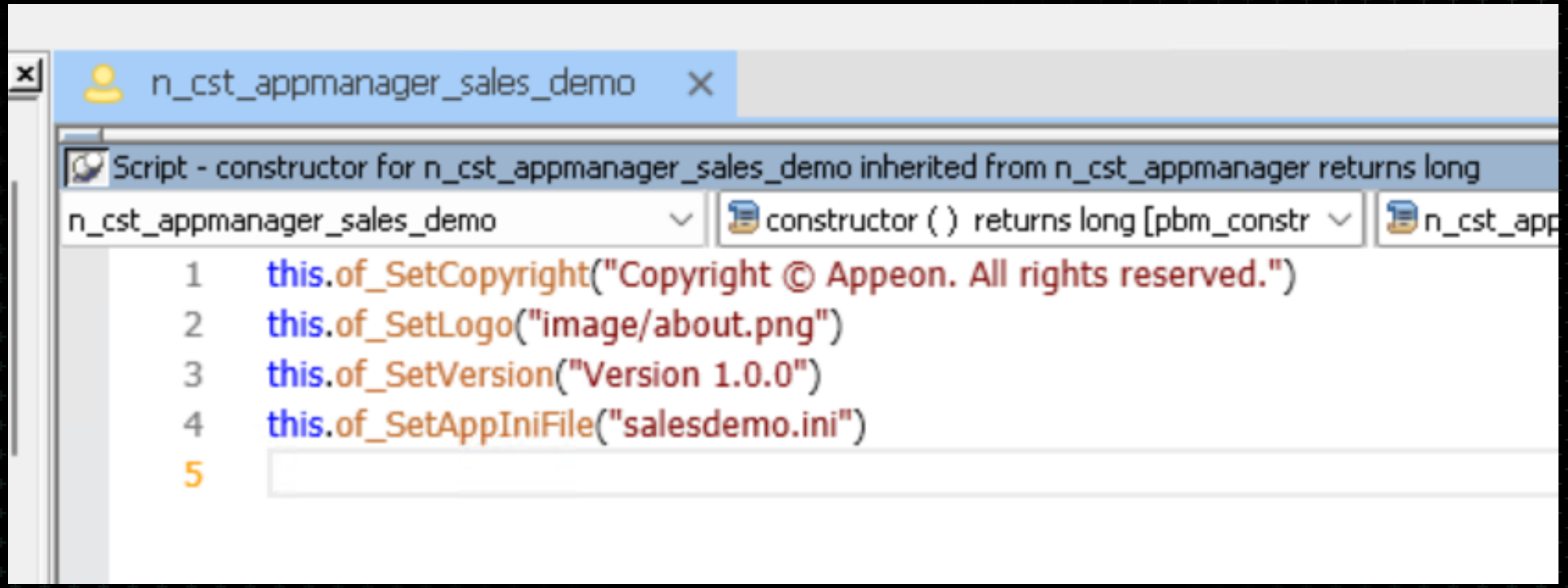
Modify application object events



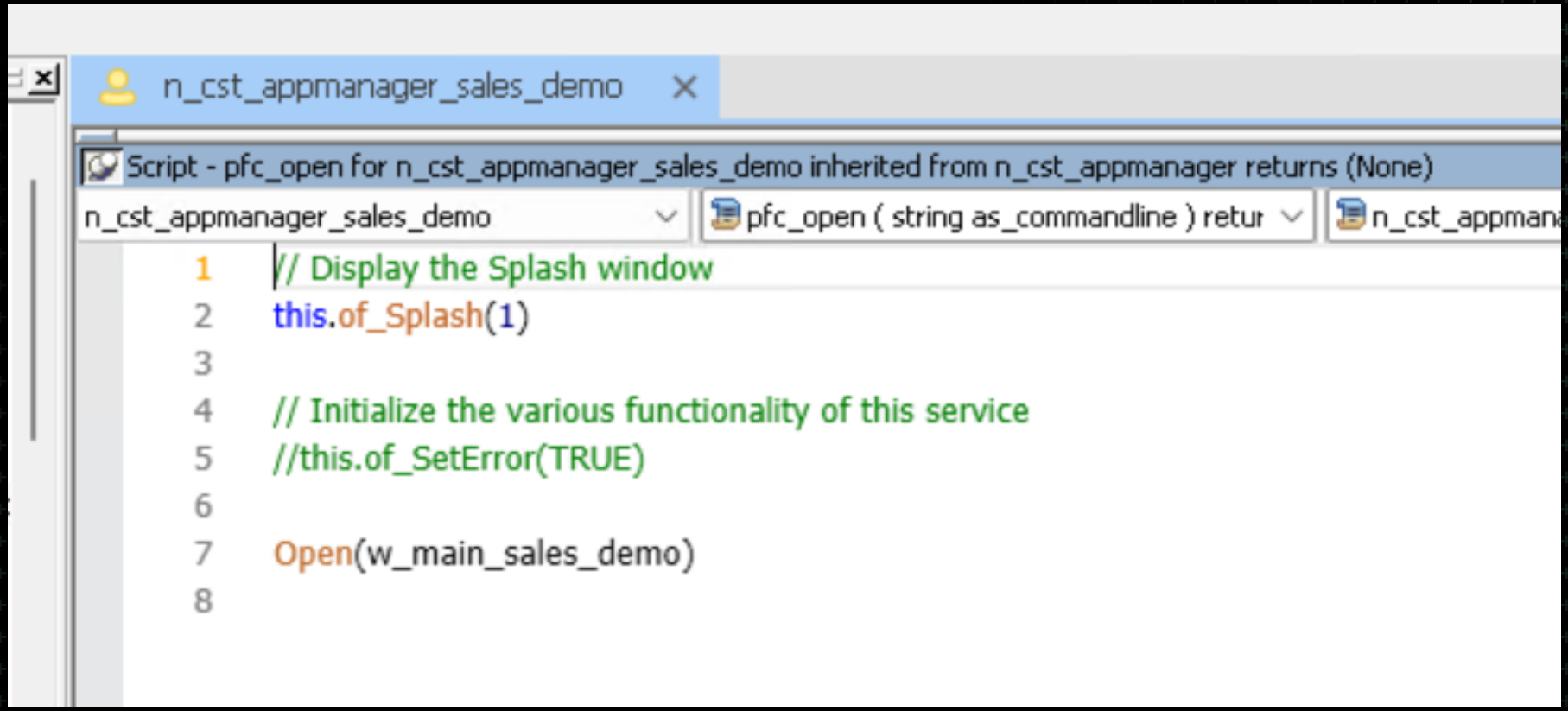
Modify application object events



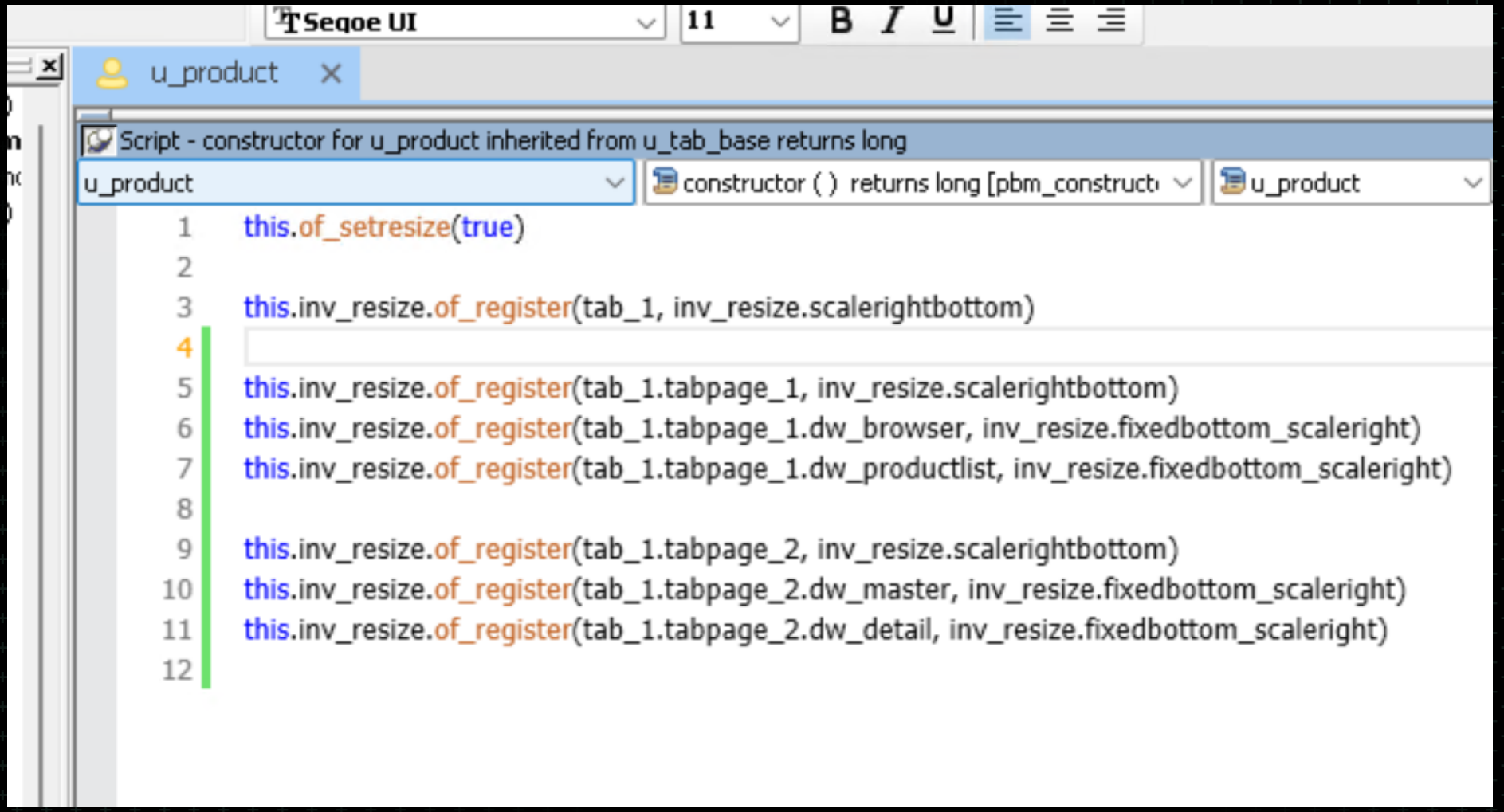
Code the application specific application manager constructor and pfc_open events



Code the application specific application manager constructor and pfc_open events

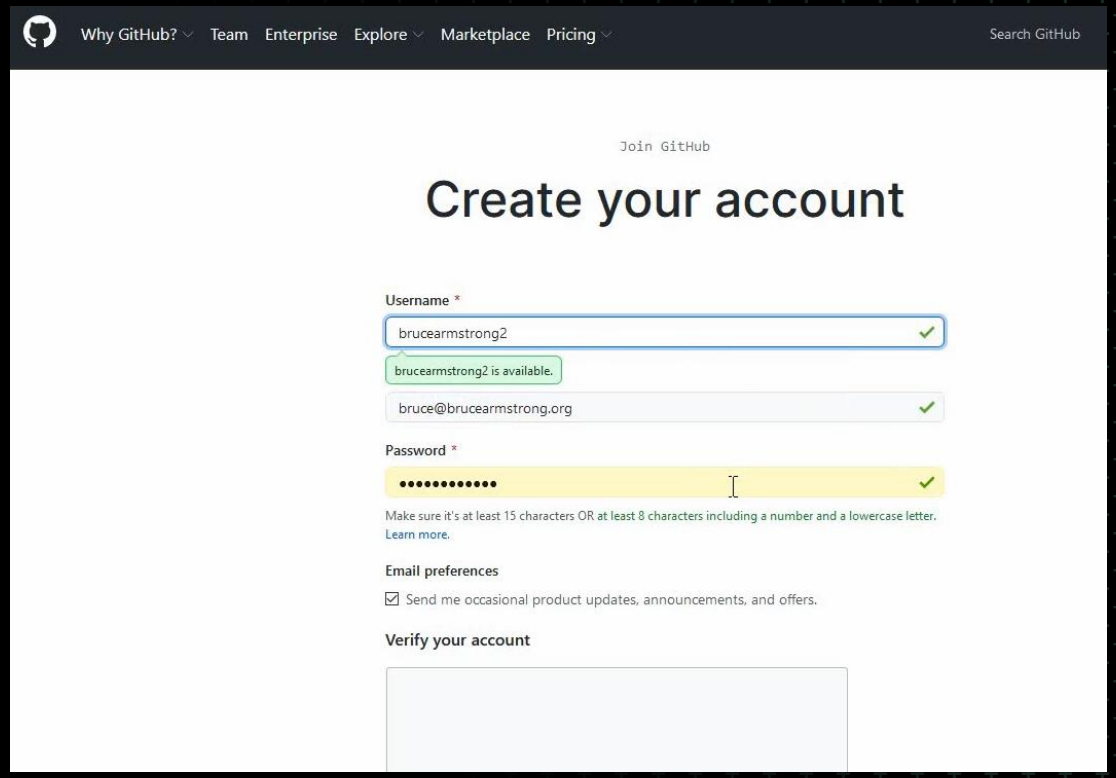


Start using PFC services



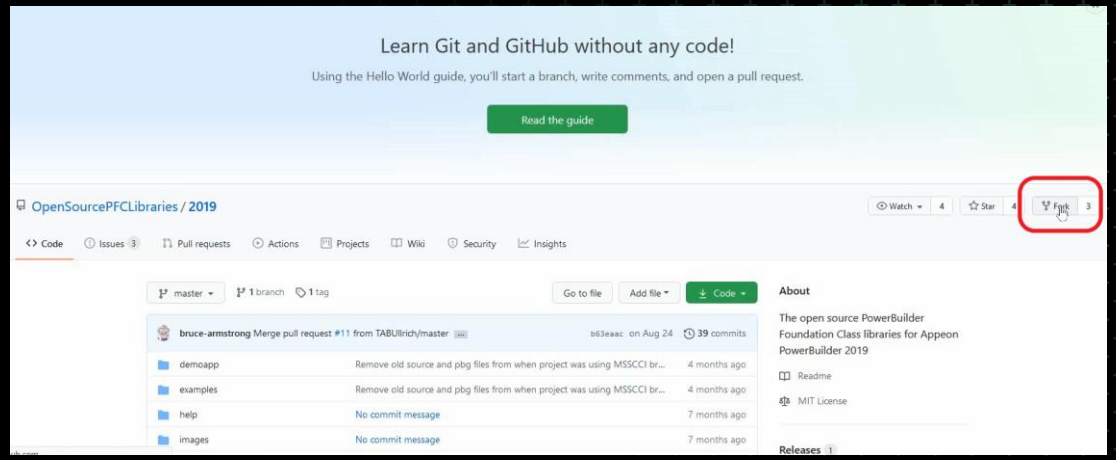
How to contribute to PFC

- Create a GitHub account if you don’t already have one



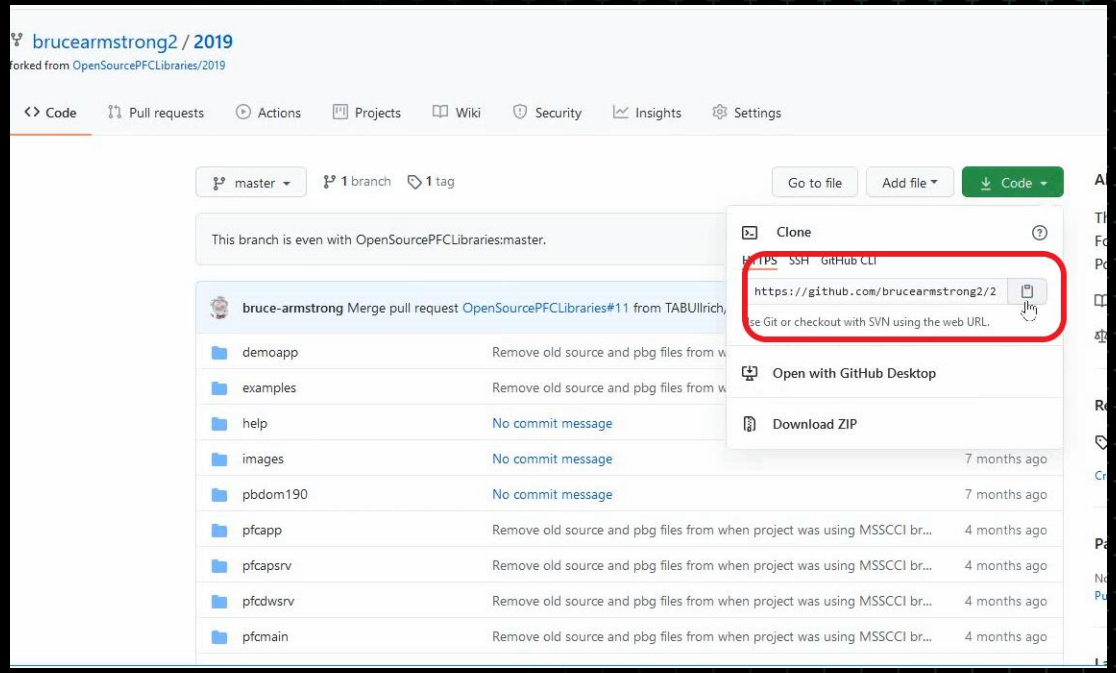
How to contribute to PFC

- Do a Fork from the Open Source PFC Library project in GitHub to create a copy in your own workspace in GitHub



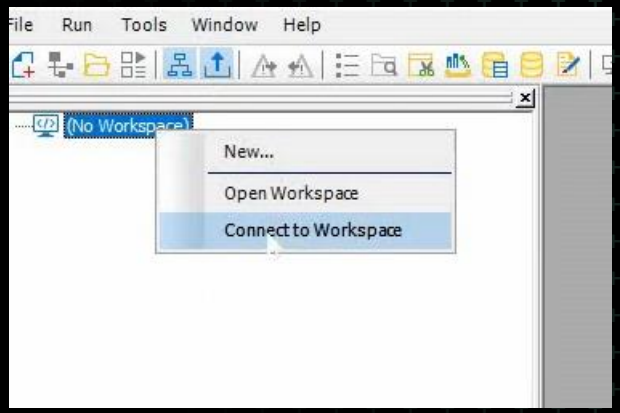
How to contribute to PFC

- Copy the URL for your copy of the project



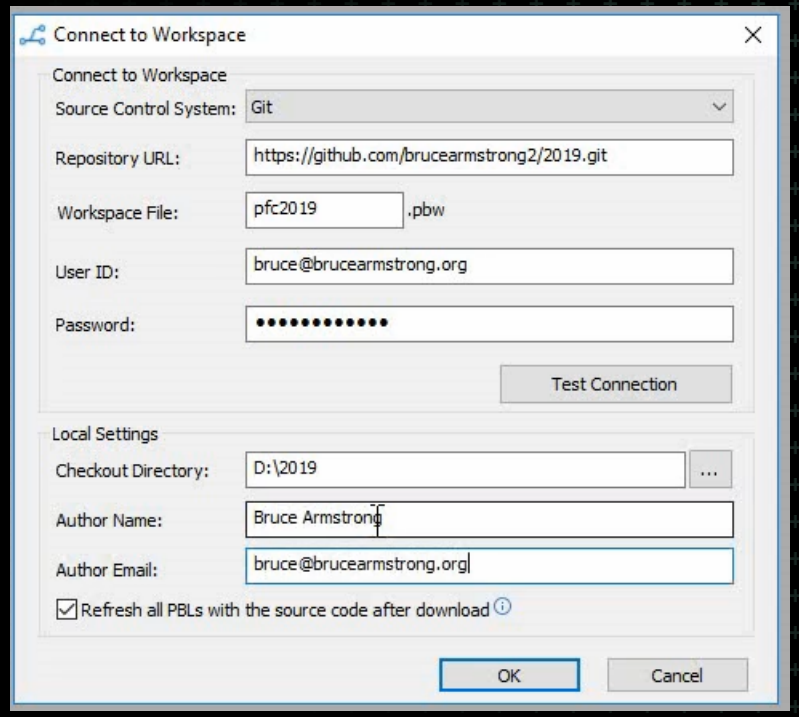
How to contribute to PFC

- Do a Connect to Workspace in the PowerBuilder IDE



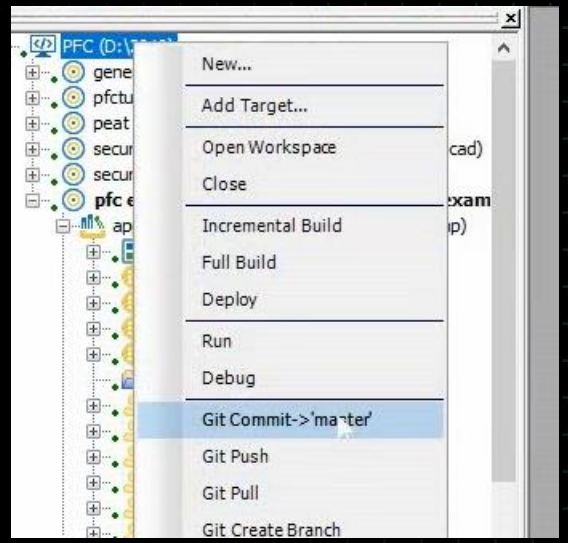
How to contribute to PFC

- Paste in the GitHub URL and provider your GitHub credentials
- This will create the workspace, target and PBLs in a repository on your local machine



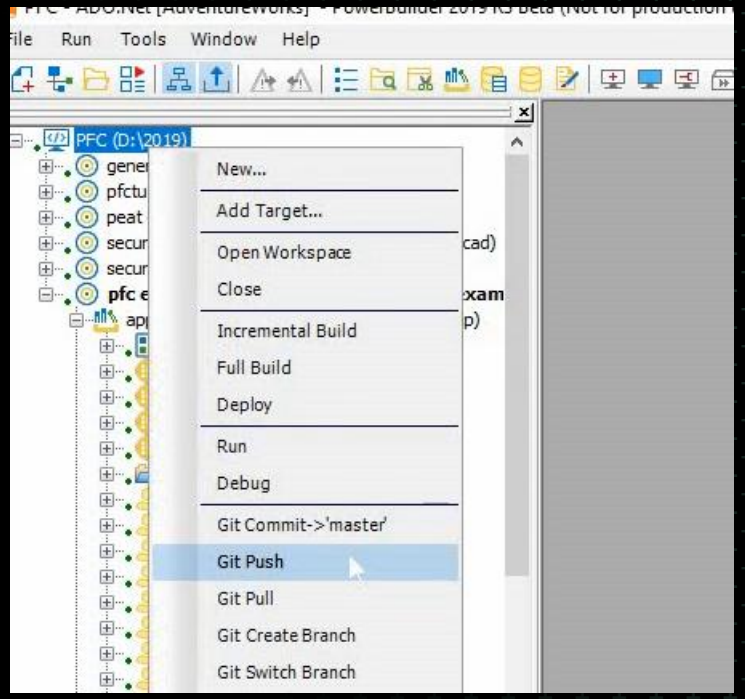
How to contribute to PFC

- After making the changes you want to contribute, do a Commit to your local repository



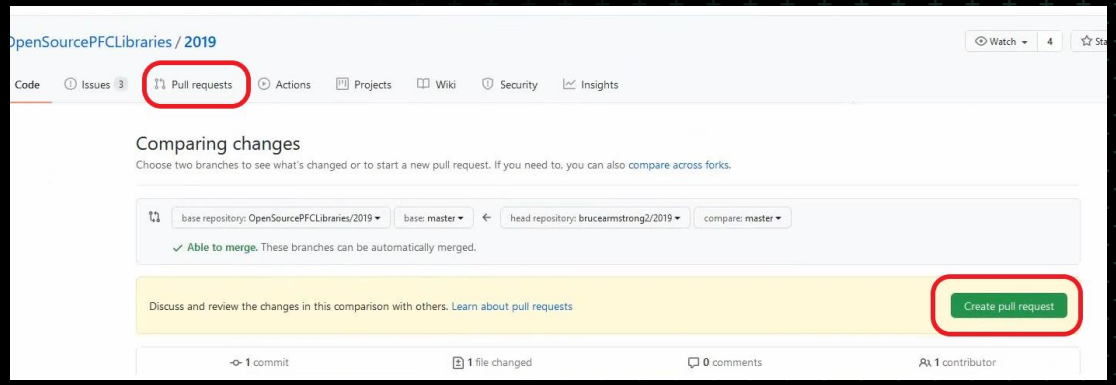
How to contribute to PFC

- The do a push to the project in your GitHub workspace



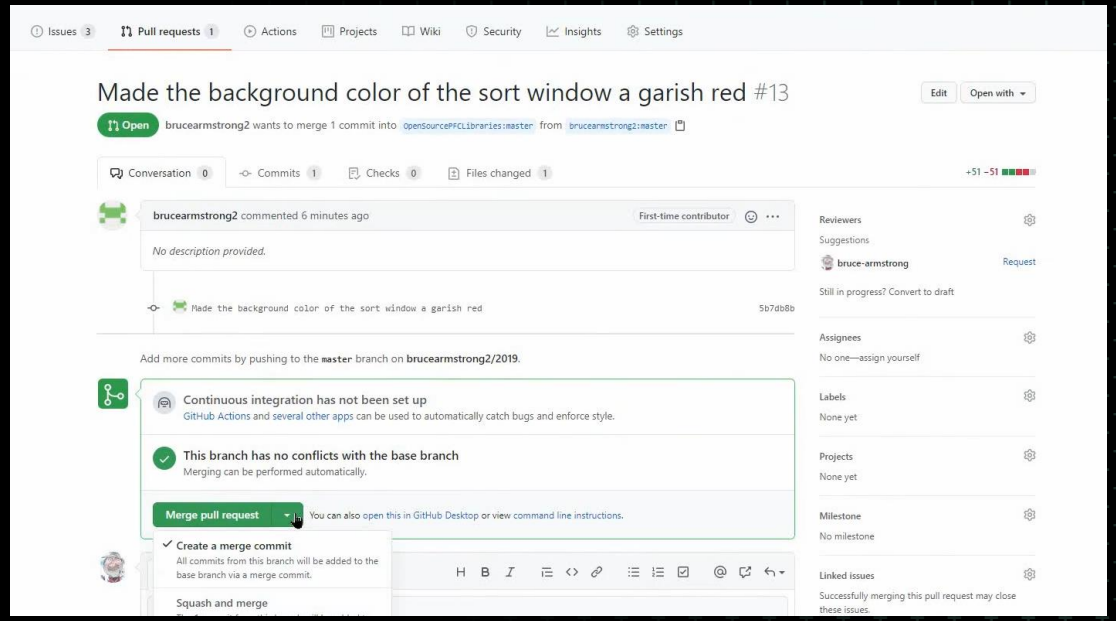
How to contribute to PFC

- Create a Pull Request in your GitHub workspace
- This submits the change to the main PFC project



How to contribute to PFC

- One of the admins in the main PFC project can now review the change, approve it or request changes to it



Session Agenda (recap)

- Why use the Open-Source PowerBuilder Foundation Class (PFC) libraries?
- Why upgrade to the new PFC?
- How to upgrade to the new PFC?
- How to migrate an existing application to use PFC?
- How to contribute to PFC?

Connect with Us

community.appeon.com

Discussions, tech articles and videos, free online training, and more.

facebook.com/AppeonPB

Encourage us with a “like”, see cool pics, and get notified of upcoming events.

twitter.com/AppeonPB

Follow Appeon and community members to get the latest tech news.

linkedin.com

Build up your career profile, and stay in contact with other professionals.

youtube.com/Appeon

Share important Appeon videos with others; no account registration required.

Thank you

Q&A Time