

# ELEVATE 2017

Providing Custom Reporting  
in Production Applications



APPEON®

# 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).

Its contents are protected by US copyright law and may not be reproduced, distributed, transmitted, displayed, published or broadcast without the prior written permission of Appeon. All rights belong to their respective owners.

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.

# Agenda

- Welcome
- Defining the Problem: Delivering applications that are *powerful, flexible, and maintainable*
- Solving the Problem: Using custom libraries and DataWindow objects
- Q & A

# Author Profile



John Hnat



[twitter.com/JohnHnat](https://twitter.com/JohnHnat)



[plus.google.com/+JohnHnat](https://plus.google.com/+JohnHnat)

## Key Skills

- PowerBuilder (v. 5.0 - present)
- Microsoft SQL Server (v. 6.5 - present)

## Recent Projects

- **2017** - Added conditional alerts functionality (automated notification if certain business criteria are met) to application.
- **2017** - Built feature to interface application with external estimating systems.
- **2015** - Implemented ability to populate and generate all government-required forms (both hard copy and soft/XML file format) to satisfy employer requirements under the Affordable Care Act.

# Company Profile



## About

Foundation Software is the developer and direct support of FOUNDATION® — job cost accounting, scheduling and project management software for the construction industry, available on premise, on the cloud and with mobile tools. CEO/Chairman Fred Ode founded the company in 1985 as an ex-teacher and world-traveler turned programmer, and since then Foundation has helped thousands of small and mid-sized contractors to succeed with the business side of their businesses. Over three decades of Ode's leadership, Foundation has developed a unique culture of being entrepreneurial, straight-shooting and fiercely independent. Foundation has won numerous and repeated awards for growth and employee satisfaction, including the Weatherhead 100, Northcoast 99, NEO Success, Crain's 52 and Plain Dealer Top Workplaces Awards.

# Overview

- What this session IS NOT:
  - Demonstration of cutting-edge technology
  - A magic bullet
  - Terribly helpful if you have (a) only one client and (b) limited reporting needs

# Overview (cont' d)

- What this session IS:
  - Demonstration of how to implement custom reporting in your PB applications, without any report-specific coding;
  - Recognition of the several (and often conflicting) demands that we face as application developers;
  - Demonstration of how we can use PB' s ability to call DataWindows dynamically to resolve some of those demands;

# Overview (cont' d)

- What this session IS:
  - Narrative of experience in providing this solution for two separate independent software vendors;
  - Cautionary tale in avoiding the mistakes we made along the way;
  - (hopefully!) A solution that you can take away from this conference and put into action, if you have a need for it.



# Introduction

“Good. Fast. Cheap. Pick two.”

As developers, particularly if we have many clients, we have three main objectives in any solution we provide:

- Powerful - the solution needs to do what clients need it to do
- Flexible - solution must serve the needs of multiple clients
- Maintainable - solution must adapt to future needs easily, and be upgraded/re-deployed without undue

# Introduction - Reports

Viewed in this framework, reporting (a core feature of most business applications) represents a challenge along all three of these dimensions:

- Powerful - the report provides the desired information
- Flexible - the report provides the appropriate information to different clients/audiences
- Maintainable - the report can be changed in the future

# A Tale of Two ISVs

First ISV: Worked 14 years for an ISV in the mortgage banking space; had application deployed at dozens of sites

- Quickly discovered: No two clients did business exactly the same way.
- Different clients would want different reports (e.g., clients in different states with different state reporting requirements)
- Different clients would want different versions of the same report (e.g., different column sets, grouping, sort order,

# Possible Solution #1

“Everything to everyone” : put all desired reports in the delivered application, and include all fields in those reports

## Problems:

- User confusion: would see reports that didn't apply to them (e.g., an OH client may not care about CA reporting requirements)
- Nightmare to train clients ( “ignore these reports / columns” )

## Possible Solution #2

“Spaghetti” code: develop different report DW objects for different clients, then branch to them within the code, so that each client sees just what they need

### Problems:

- Difficult to maintain
- Your testers will hate you forever

## Possible Solution #3

Different EXEs: Build different EXEs for different clients, based upon their desired report sets.

Problems:

- Difficult to maintain
- Your testers will REALLY hate you forever

## Then We Stumbled on a Solution ...

We realized that PB provides the capability to define the library list at run-time

`SetLibraryList()`: our new best friend

Many/most of the differences between clients could be managed via different custom libraries (with different DW objects) deployed to different clients

# Our Delivered Solution

Our delivered solution contained the following elements:

- Report PBLs containing DataWindow objects
  - These reports could be created by Product Development, by other technical staff, or even by the clients themselves
  - As part of deliverable, would license the client a copy of InfoMaker, and install that on a client machine during the process of installing / configuring our solution
  - Gravitated to a standard set of deliverable PBLs that we would deploy for a fresh installation



## Our Delivered Solution (cont' d)

Our delivered solution contained the following elements:

- Setup routine within application
  - Application had several different report menu items - for each, had the ability to specify which PBL would be used
  - Required deploying the PBL in a location accessible to the application
  - During application startup, these PBLs would be added to the library list (using `SetLibraryList()`), and then the report DWs in those PBLs would be available to the application

## Our Delivered Solution (cont' d)

Our delivered solution contained the following elements:

- “Base” report window
  - At run-time, when a user clicked on a report menu routine, they would see a list of all of the report DWs available within that PBL
  - When user would select a report, application would obtain it from that specified PBL (which, remember, had been added to lib list)
  - End result: User sees “their” reports

# Did Our Solution Meet Our Goals?

Think back to the three dimensions of gauging application:

- **Powerful:** The custom report solution delivered what clients needed
- **Flexible:** It allowed us to customize our report offerings for each client, and deliver custom versions of same reports
- **Maintainable:** All code contained in one EXE, with no branching or “spaghetti” code

## So How Well Did This Solution Work?

Generally, worked pretty well. Application deployed with this solution at ~50 client sites.

Clients were happy: they could see the reports they wanted.

Internally, this solution allowed us to off-load report development from Prod Dev to other technical personnel.

Allowed for making income from creating custom reports.

# Problems With Original Solution

Deployment issues: problems with clients being able to deploy PBLs to their network

- It was a routine stumbling block during installations
- It also required clients not to do anything silly, like move or delete files that they did not think were required by the application
- Result: Client frustration, especially during critical installation phase; made installs take longer than necessary

# Problems With Original Solution

Access issues: problems with users not having access to the appropriate network location that stored custom PBLs

- Once upon a time, was not much of an issue; everybody had access to most everything
- As better security practices were implemented, we kept running into this issue with clients
- Result: Client frustration and inadequate use of Client Support resources

# Problems With Original Solution

Security issues: system admins want to limit report deployment among users (not all users should access all reports)

- Solution #1: Deploy different PBLs for different users
  - Was very confusing and led to many more support calls
- Solution #2: Specify report-by-report access per user
  - Difficult to do reliably when you do not know what the report PBL will contain

# Problems With Original Solution

## Security issues: SQL injection concerns

- Custom reports would often require their own database objects (e.g., views, stored procedures)
- Application had a mechanism for running SQL files on an ad-hoc basis; very powerful, but also very dangerous
- As clients became more sophisticated, and started retaining their own DBAs, running SQL files in this manner became more and more difficult; clients would push back



# Problems With Original Solution

## Technology issues: Reliance on InfoMaker

- For a while, became all but impossible to locate vendors, so that our clients could obtain copies of the software
- Client Support now supporting not just our application, but also a third-party application
- By allowing clients to use IM to create own reports, we limited our revenue from creating custom reports for clients
- Clients would not adhere to best practices, then run into

## A Tale of Two ISVs (Remember, there were two!)

Second ISV: Three years ago, joined a new company; discovered that software did not have similar custom reporting capability

Cognizant that new ISV faced additional challenges:

- Many more clients / new installs
- Existing reporting framework - want to use if possible
- Generally, less technically sophisticated client base
- No dedicated means for executing ad-hoc SQL statements

## Custom Report Solution: Round Two

Decided to proceed with a different approach, one that would provide same functionality, but with following improvements:

- No deliverable PBLs that clients need to deploy
- No need for any file access (other than regular DB access)
- No use of SetLibraryList() for running reports
- Ability to set user access (full, read-only, none) per report
- No reliance on ad-hoc SQL statements to create objects
- No delivery of InfoMaker to clients

# A New Approach to Custom Reports

Made possible by DataWindow flexibility

- Not only can we change the data object at run-time (which made the earlier approach possible); we do not even need to have a data object at all
- If DW definition saved to database, then application could retrieve it at run-time, and use it to create DW object on the fly

## A New Approach to Custom Reports (cont' d)

Step 1: Create DW object and any supporting database objects (such as a stored procedure)

Step 2: Use a packager application (a separate PB application, developed by and used only within our company) to “package” the report, along with an optional criteria DW, and any supporting SQL scripts (e.g., ones that create database objects)

## A New Approach to Custom Reports (cont' d)

Step 3: Within the main application, use a new window to deploy a selected custom report to the current database

Step 4: Use existing application security framework to set each user's access to this custom report

Once these steps are completed, then the user will be able to view the report (assuming they have access); no further actions necessary

## Step 1: Create DW/DB Objects

- Create database objects (same as before)
  - Any stored procedures, views, and/or other database objects needed to retrieve the underlying report data
- Create DataWindow objects
  - Report DW (same as before)
  - Criteria DW, to comply with report framework

## Step 2: Custom Reports Packager

The custom reports packager is a separate application that we developed and that is meant for internal use only; not distributed to any clients

Purpose: Take SQL objects needed for a custom report, and use them to create a “script” object (DW object that stores the database scripts in the DW’s Data property); then write this script object to the same PBL as report/criteria objects



# Custom Reports Packager (cont' d)

Foundation Custom Report Script Packager

Select a Report to be Packaged:

Library File: J:\Projects\Custom Reports\Developed Reports\Custom 401k\cst\_custom\_401k\_report.pbl PBL Browse

Report Name: 401k Report (custom\_401k\_report) ▼

Specify SQL File(s) to be Packaged:

J:\Projects\Custom Reports\Developed Reports\Custom 401k\csp\_rpt\_cst\_401k\_report.sql Add Row

Insert Row

Delete Row

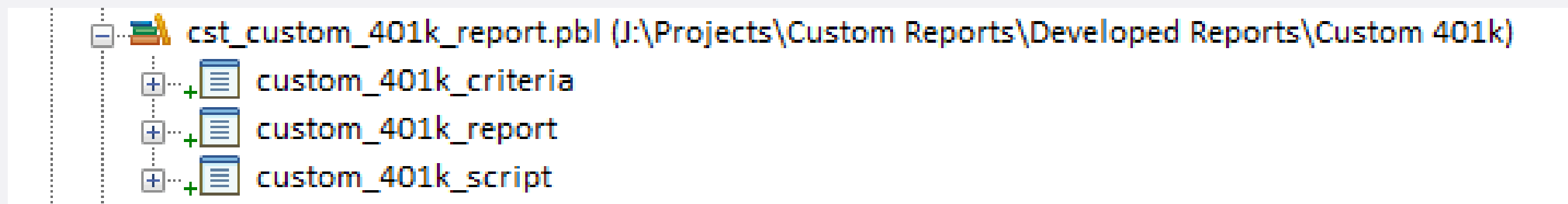
SQL Browse

Package

Close

## Custom Reports Packager (cont' d)

- The generated DW object is a freeform object with one column: a maximum-length (32,767) string column
  - If a database object contains more than 32,767 characters, the packager splits it up into separate lines of data



## Step 3: Deploy Custom Report

- Built an interface within our flagship application to deploy custom reports
- Allows user to select a custom report, set its parameters, and then save it to the database

# Deploy Custom Report (cont' d)

Manage Custom Links

Programs Documents **Reports**

Editing Entry: 401k Report

Name: 401k Report

Full Name: 401k Report

Report Group: General

Report Identifier: custom\_401k\_report

Report Modified: 10/06/2016 07:49:21 Criteria Object: Yes

Available in All Companies in this Database

Available to All Users in this Company

Save Changes

Load Report

Delete Entry

Cancel

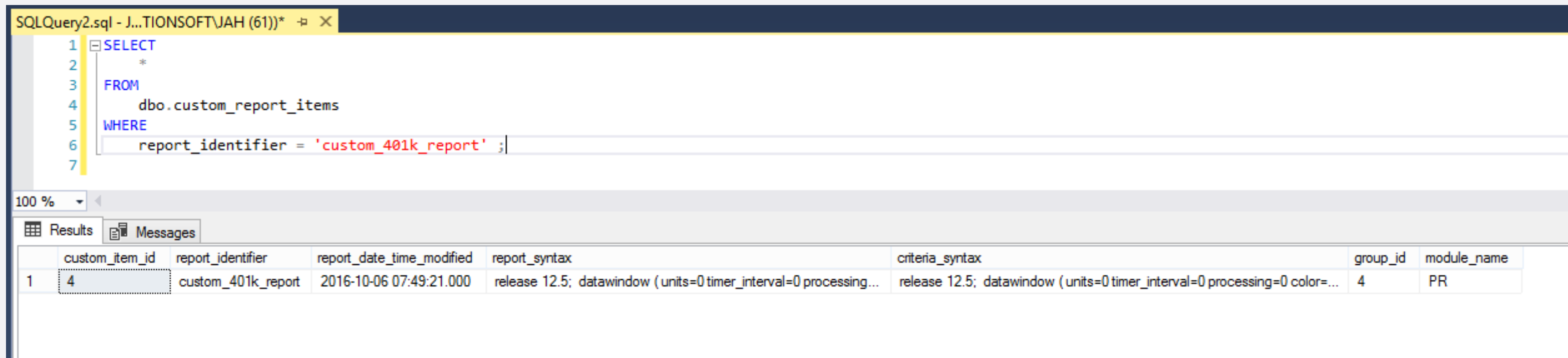
Existing Report Entries. Select to Edit:

Name	Group	Report Identifier	Criteria
401k Report	General	custom_401k_report	Yes
G/L History By Job	G/L	gl_history_by_job_report	Yes
Test Report	Test	test_report	No

Help Close

# Deploy Custom Report (cont' d)

- Note that the DW syntax is saved to the database (see the report\_syntax and criteria\_syntax columns below):



The screenshot shows a SQL query window with the following text:

```
1 SELECT
2 *
3 FROM
4   dbo.custom_report_items
5 WHERE
6   report_identifier = 'custom_401k_report' ;
7
```

Below the query, the results are displayed in a table:

custom_item_id	report_identifier	report_date_time_modified	report_syntax	criteria_syntax	group_id	module_name
4	custom_401k_report	2016-10-06 07:49:21.000	release 12.5; datawindow (units=0 timer_interval=0 processing...	release 12.5; datawindow (units=0 timer_interval=0 processing=0 color=...	4	PR

## Step 4: Set user report security (optional)

- Our application has a security mechanism that controls user access to all menu items - tied custom reports into this existing menu access framework
- May or may not want to do this in your own application, depending on your security needs and existing code

# Running Custom Reports

- Once the custom report has been deployed, it may be executed at will by any users who have access to the report. No further intervention is necessary.
- In our case, custom reports are available from our main application menu, and can be accessed just like any of our regular reports.
- From end user perspective, it is as if we had created this custom report just for that user.

# What Advantages Have We Seen?

- Ability to deliver desired reports to clients
  - Previously: Might not get them at all, depending on development priorities and available resources
  - Now: Can respond to all client report needs
- Ability to deliver a more tailored application for client
  - Previously: If we chose to include a new report for a client, then all clients would get that report, whether desired or not
  - Now: Can deliver specific reports to specific clients, without application being cluttered with undesired reports for other users



## What Advantages Have We Seen? (cont' d)

- Quicker delivery of desired reports
  - Previously: All reports included in EXE; as such, at mercy of our development cycle (could take months, even years – and that's if we decided to include the report in the application)
  - Now: Custom reports are not part of the EXE; can develop them, test them, and deliver them outside of development cycle
- On-ramp for including new reports in mainline application
  - Previously: Had to include all reports in EXE
  - Now: Can “test” new reports with clients; once they get to critical mass, can move them to the main application

## What Advantages Have We Seen? (cont' d)

- Off-load report development to data specialists
  - Previously: All reports developed by programmers; took them away from more critical tasks
  - Now: Many/most custom report tasks can be done by data specialists (non-programmers)
- On-ramp for training of new programmers
  - Previously: had to recruit new programmers from Support/Training, and they would start from scratch
  - Now: Data Specialist role is transition period/” proving ground” for eventual move into programming

## What Advantages Have We Seen? (cont' d)

- Economies of scale
  - At this point, have a “library” of custom report solutions
  - Takes less time to deploy at 2<sup>nd</sup> client than at 1<sup>st</sup>
- Overall effect on bottom line
  - Not a huge money-maker for us - we often do not charge
  - But it enhances client retention
    - We respond quickly to specific client requests
    - The more entrenched we are at a client site, the more difficult it is going to be for us to be replaced



Demo

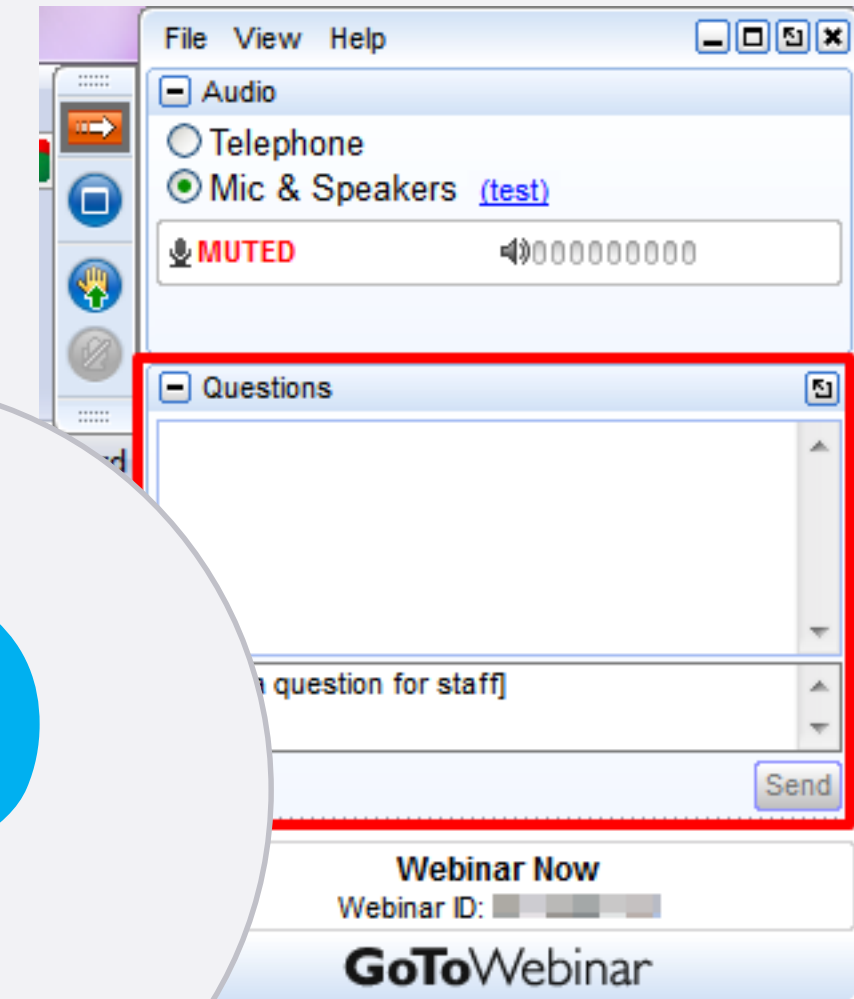
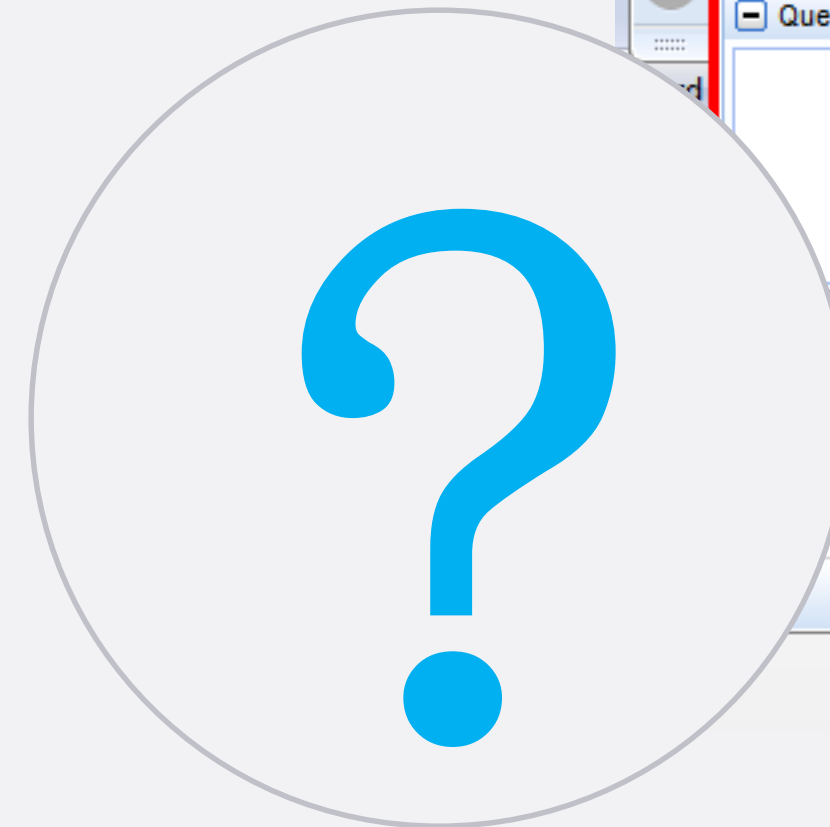


# Conclusion

- Good, fast, cheap: pick ALL THREE
- Custom report solution is a way for PB solution developers to deliver powerful, flexible, and maintainable applications

# Ask the Expert

We will do our best to answer all questions now or later by email.



# Connect with the Appeon Community



[community.appeon.com](https://community.appeon.com)

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



[facebook.com/AppeonPB](https://facebook.com/AppeonPB)

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



[twitter.com/AppeonPB](https://twitter.com/AppeonPB)

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



[linkedin.com](https://linkedin.com)

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



[youtube.com/c/AppeonHQ](https://youtube.com/c/AppeonHQ)

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



[google.appeon.com](https://google.appeon.com)

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



Thank You

