1. How can I use CrossWay within my Progress OpenEdge development workspace?

Using CrossWay is simple. Just install the provided CrossWay plugin `zip` file into Progress Developer Studio for OpenEdge and the visualize tools like Wayfare's CrossWay Visualizer and/or draw.io installed version of it or draw.io executable. After installation you can generate diagrams and insights directly from your workspace.

2. What types of diagrams does CrossWay generate?

Currently, CrossWay provides three main types of diagrams to help you understand your application architecture:

- **Impact Diagram** reflects complex relationships between project files and reveals how changes to one file or class might affect other connected files, making it easier to analyze dependencies and potential side effects like circular dependencies.
- **Inheritance Diagram** shows class inheritance relationships, visualizing parent–child hierarchies in your code.
- **Interface Diagram** identify where classes use common methods, properties, or structures defined by interfaces, highlighting implementation dependencies.

These diagrams give you both a high-level and a detailed perspective on how your Progress OpenEdge application is organized.

3. Does CrossWay require full source code access?

CrossWay can be configured to take into account all or parts of the ABL source code in order to perform its analysis. It works by analyzing **XREF output** generated by its internal compilation of source code using **OpenEdge 12.8**.

4. Is CrossWay suitable for legacy OpenEdge systems?

Yes, CrossWay supports analysis of legacy-style ABL code as well (e.g. procedural patterns) — making it conceptually suited to work both with legacy OpenEdge systems and latest OpenEdge versions.

However, CrossWay requires for the user to ensure the application code **can be compiled with OpenEdge 12.8**. This means that CrossWay does **not work with pre-compiled r-code** from older runtimes or source code that cannot be compiled under 12.8 due to syntax incompatibilities.

5. Is CrossWay easy to use for non-developers, such as testers or analysts?

Yes — CrossWay is originally designed to be used mostly by OpenEdge developers through the Developer Studio IDE, but the generated diagrams can be also ported out to other systems and machines and they can become accessible to other engineers (e.g. **testers and business analysts, software architects**) that need visibility into how project files interact with each other and how changes can affect an OpenEdge application.

The tool presents complex or simple code relationships through **visual diagrams** that are easy to configure, filter, navigate and understand — even without deep ABL knowledge. These diagrams help non-developers:

- Identify which modules are impacted by a change
- Understand dependencies between files and database tables
- Plan test coverage based on actual code change dependencies
- Reduce guesswork when assessing risk or validating requirements

No need to read source code — **a picture shows the full impact**.

6. What file types and extensions are supported?

CrossWay lets you configure which file types to include in the analysis. By default, it supports standard Progress OpenEdge source files such as **`.i`**, **`.p`**, **`.cls`**, and **`.w`**. You can adjust the supported extensions in the configuration to match your project's needs.

7. What versions of Progress OpenEdge does CrossWay support?

CrossWay currently supports **code that is compiled using Progress OpenEdge 12.8**.

This means that while your application may have been originally developed in earlier OpenEdge versions (such as 10.x or 11.x), CrossWay can analyze it **only if the source code is compatible with the 12.8 compiler** and has been successfully compiled in a 12.8 environment.

- ✓ If you can compile your codebase with OpenEdge 12.8, CrossWay can analyze it — regardless of which version the code was originally written for.
- ✗ CrossWay does **not** support analysis of r-code or source code that cannot be compiled under 12.8.

8. How is database schema information extracted? Does it connect directly to a DB or parse schema files?

CrossWay determines referenced database access information from the XREF output of the compiled file (using the project's own database connections). It does **not** parse exported schema files (like `.df`).

To enable this, before starting to generate any diagrams you will first need to use an "Reset Config" menu option to allow CrosWay to access the Openedge project's database connections.

9. Is the analysis static (code parsing only) or dynamic (runtime behavior tracking)?

CrossWay performs **static analysis only**. It inspects and collects cross-reference information from the source files and related metadata (such as database access) without executing the application. No runtime instrumentation or behavior tracking is used.

10. Does Crossway provide logging or error messages when analysis fails? Where can I find them?

Yes, CrossWay provides detailed logging to help identify issues during analysis. All logs are saved to a text file named **`CrossWay.log`** located in the folder specified by the **`OutputPath`** setting in your `Config.json` file.

This log includes key processing steps and possible compile errors and it is intended to be used for assisting with troubleshooting failures or unexpected behavior during diagram generation or XREF collection.

11. Can CrossWay export diagrams to external formats (PDF, PNG, SVG, JSON)?

Yes. Diagrams generated with CrossWay can be exported in multiple formats depending on the tool used:

- When opened in draw.io, you can export diagrams using its built-in export options (PDF, PNG, SVG, XML/JSON).
- If using the **CrossWay Visualizer**, diagrams can be exported as **PNG** files directly from the interface.

These options allow integration of diagrams into documentation or other analysis tools as needed.

12. Can we customize the visual diagrams (e.g., colors, layout, filters)?

Yes, CrossWay allows extensive customization of visual diagrams through the **UI Config** settings screen but also draw.io and CrossWay Visualizer extends this support with further user options over a generated diagram. You can adjust layout properties such as node size, spacing and initial positioning, as well as define custom colors for different node types (e.g., classes, include files, procedures or ABL UI Design) and relationships (e.g., run, inherits, implements, circular dependencies).

To configure these settings, use the menu option **CrossWay > Settings > UI Config**, and adjust the properties as needed.

For a detailed description of each configurable option, refer to the **UI Config** section of the user manual.

13. How does CrossWay handle code circular dependencies?

CrossWay automatically detects and highlights possible circular dependencies over a generated Impact Diagram using **distinct link colors**. For example, links involved in circular references are typically rendered using the color defined under the **"Circular Dependency Color"** property in the UI configuration.

This visual distinction helps users quickly identify and review cyclic relationships between project files.

Please refer to the UI Config section for more details on how to customize link colors and visibility.

14. My diagram shows incomplete information. Am I missing something?

There are multiple cases and scenarios which may lead you into such a situation, however you might want to review this checklist below in order to ensure CrossWay is correctly running:

- CrossWay configured paths are correctly assigned (e.g Technical Impact Path and Extended Path settings might be wrong or incomplete) - please refer to the **Config Paths** User manual section for more details.
- At the Clean XREF step there were compile errors reported in the Crossway.log
- The Diagram Criteria (Generate Impact Diagram option) is persisting user settings from last use - please review “Simple Links only”, “Depth level”, “Reference Types” checkboxes options

15. Can I use other visual tools for generating diagrams?

No — Currently, CrossWay supports diagram generation through **draw.io** and **CrossWay Visualizer**. These are the only supported rendering tools at this time.

However, we are actively evaluating the possibility of supporting additional diagramming tools or export formats in future versions, based on user feedback and evolving requirements.