Productivity! Feature Matrix

Features	JBuilderX and Productivity!		JBuilderX
	Std	Pro	
Code Generation Tools			
Delegate.Insight - provides an easy way to generate methods, which implementations are delegated to another object (delegate).		✓	
➔ Ability to use as a delegate a field declared either in currently edited class or in any of its parents.		✓	
➔ Ability to use as a delegate execution result of method without parameters.		✓	
➔ Ability to use as a delegate execution result of method with parameters.		✓	
➔ Ability to pick up changes in the delegate class or interface.		✓	
➔ Utilizing convenient "Insight" approach.		✓	
➔ Ability to view help for found delegates and methods using Help.Insight or JBuilder help.		✓	
➔ Ability to specify search and sorting options.		✓	
➔ Ability to discover current class' context at caret position.		✓	
➔ Ability to change class' context right from the Implement.Insight.		✓	
➔ Ability to work with inner classes.		✓	
➔ Ability to specify placement for generated methods.		✓	
➔ Ability to specify code and parameters' generation options.		✓	
➔ Ability to generate JavaDoc comments during methods generation.		✓	
Introduce.Constructor - allows easy generation of constructors intended to initialize appropriate fields of the class.		✓	
Class.Insight - allows quick finding Java classes with short names matching the word at the cursor position, and inserting the class name found into the cursor position as well as inserting import statement.	✓	✓	
➔ Ability to find a class using package tree.	✓	✓	✓
➔ Ability to find a class using its short name.	✓	✓	✓

➔ Utilizing convenient "Insight" approach.	✓	✓	
➔ Ability to view help for found classes using Help.Insight or JBuilder help.	✓	✓	
➔ Ability to specify classes search and sorting options.	✓	✓	
➔ Ability to exclude not required packages form search result.	✓	✓	
➔ Ability to invoke Smart.Instantiate right from the Class.Insight.	✓	✓	
➔ Utilizing classes' cache to speed up classes finding process.	✓	✓	
Implement.Insight - allows quick finding Java classes with short names matching the word at the cursor position and using them either as a super interface or as a super class.	✓	✓	
➔ Ability to find an interface using package tree.	✓	✓	✓
➔ Ability to find an interface using its short name.	✓	✓	✓
➔ Ability to pick up changes in the interface to be implemented.	✓	✓	
➔ Utilizing convenient "Insight" approach.	✓	✓	
➔ Ability to view help for found interfaces using Help.Insight or JBuilder help	✓	✓	
➔ Ability to specify interfaces search and sorting options.	✓	✓	
➔ Ability to discover current class' context at caret position.	✓	✓	✓
➔ Ability to change class' context right from the Implement.Insight.	✓	✓	✓
➔ Ability to work with inner classes.	✓	✓	✓
➔ Ability to specify placement for generated methods.	✓	✓	
➔ Ability to specify code and parameters' generation options.	✓	✓	
➔ Ability to generate JavaDoc comments during methods generation.	✓	✓	✓
Override.Insight - allows quick finding methods to override with names matching the word at the cursor position or a typed word, and overriding them into the class at the cursor position.	✓	✓	
➔ Ability to select methods to override from one list of all suitable methods.	✓	✓	
➔ Ability to override methods defined in super interface(s) those are directly implemented by the target class.	✓	✓	
➔ Ability to pick up changes in the super classes and interfaces.	✓	✓	
➔ Utilizing convenient "Insight" approach.	✓	✓	

➔ Ability to view help for found methods using Help.Insight or JBuilder help.	✓	✓	
➔ Ability to specify methods search and sorting options.	✓	✓	
➔ Ability to discover current class' context at caret position.	✓	✓	*
➔ Ability to change class' context right from the Override.Insight.	✓	✓	✓
➔ Ability to work with inner classes.	✓	✓	✓
➔ Ability to specify placement for generated methods.	✓	✓	
➔ Ability to specify code and parameters' generation options.	✓	✓	
➔ Ability to generate JavaDoc comments during methods generation.	✓	✓	✓
Constructor.Insight - allows quick "overriding" class constructors.	✓	✓	
Easy.JavaDoc allows easy and convenient generating templates for JavaDoc comments for a particular method or class.	✓	✓	
➔ Ability to generate JavaDoc comment for particular method or class	✓	✓	✓
➔ Ability to generate JavaDoc comments for the several selected methods or classes	✓	✓	
Smart.Instantiate is an additional Class.Insight functionality that allows adding instantiation of a particular class or interface.	✓	✓	
GetSet.Creator is a tool that allows easy creation of accessors and/or mutators for selected fields of a class.	✓	✓	*
➔ Ability to generate accessors and/or mutators for public classes.	✓	✓	✓
➔ Ability to generate accessors and/or mutators for non-public classes.	✓	✓	
➔ Ability to generate accessors and/or mutators for inner classes.	✓	✓	
➔ Ability to generate static methods for static fields.	✓	✓	
➔ Utilizing convenient "Insight" approach.	✓	✓	
➔ Ability to view help for found fields using Help.Insight or JBuilder help.	✓	✓	
➔ Ability to specify fields search and sorting options.	✓	✓	
➔ Ability to discover current class' context at caret position.	✓	✓	
➔ Ability to change class' context right from the GetSet.Creator.	✓	✓	
➔ Ability to specify placement for generated methods.	✓	✓	

➔ Ability to specify code and parameters' generation options.	✓	✓	
➔ Ability to generate JavaDoc comments during methods generation.	✓	✓	
Rename Assistant – allows safe renaming of identifiers using "in-place" editing approach.		✓	
The set of assistants those show information about the particular issue and/or list of possible actions to complete.		✓	
➔ Ability to automatically fix type cast errors by adding required type cast.		✓	
➔ Ability to automatically fix unknown class errors by adding appropriate import statement.		✓	
➔ Ability to specify style for error highlight.		✓	
➔ Ability to show Assistant or tool tip when mouse cursor is placed over issue highlight.		✓	✓
➔ Ability to automatically pop up Assistant for the issue nearest to the caret position.		✓	
➔ Ability to enable/disable popping up of Assistants.		✓	
➔ Ability to invoke Assistant using keyboard shortcut.		✓	*
➔ Ability to prompt one or several choices those can fix the issue.		✓	*
➔ Ability to fix the issue using keyboard shortcuts.		✓	*
➔ Ability to navigate to the issue location using keyboard shortcut.		✓	
➔ Assistant for working with Advanced To-Do's.		✓	
The Task List is a tool that allows viewing and managing the list of tasks, code issues and to-do's.		✓	
➔ Ability to have uniform way to work with general tasks, code issues and To-Do's.		✓	
➔ Ability to organize tasks using plain or structured view, filters and ordering criteria.		✓	
➔ Ability to create general tasks and To-Do's.		✓	
➔ Ability to remove general tasks and To-Do's.		✓	
➔ Ability to change attributes of tasks.		✓	
➔ Ability to work with project-wide To-Do's.		✓	
➔ Ability to setup reminders to be notified about tasks need to be done in time.		✓	

➔ Ability to fix code issues right from the Task List.		✓	
The Advanced To Do's tool is further expansion of the To Do comments concept by treating them as tasks with set of attributes like priority, status, owner etc.		✓	
The Smart.Templates is advanced template engine that provides set of sophisticated features, like linked fields, calculated fields; expression and functions supports. It introduces advanced code templates those can be easily adapted to particular coding style coding style.		✓	*
➔ Ability to work with "simple" or "static" templates, which represent simple code fragment to be inserted to the document.		✓	✓
➔ Ability to manage template definitions.		✓	✓
➔ Ability to work with "smart" templates, which represent set of code fragments along with dynamic fields that allows building dynamic and live templates.		✓	
➔ All the template fields of same name are automatically synchronized that allow entering field value only once		✓	✓
➔ Ability to specify default value or initialization expression for a template field. The default value or result of expression execution is automatically assigned to the field on template expansion		✓	
➔ Ability to specify calculated expression for a template field. This allows dynamically calculate field value to reflect changes made by the user in this field or in the other ones		✓	
➔ Ability to specify "change" expression for a template field. This allows making some actions after the field is changed and/or calculated		✓	
➔ Ability to store values entered in the template fields and load previously stored values on the next template invocation. This allows to fill fields automatically during the consecutive template calls		✓	
➔ Ability to distribute templates among groups to allow easy sharing of then between team members.		✓	
➔ Ability to define private templates, which accessible for the particular user only.		✓	
➔ Ability to shorten full-qualified names for classes stated in the template and adding appropriate import statements to the source file.		✓	*
➔ Ability to use preprocessor instructions those allow dynamic building of template code using conditional and iteration statements.		✓	
➔ Ability to format template code according to the current project code style and current indent level.		✓	
➔ Ability to specify supported file types (Java, HML etc) for any template.		✓	*
➔ Ability to specify supported context (e.g. "symbol", "comment" or "string") for any template.		✓	

➔ Ability to assign keyboard shortcut for any template. Shortcut assignments for different key maps are supported too.		✓	
➔ Ability to utilize selected block of code as field value.		✓	
The Smart.JavaDoc tool is additional viewer for Java file node that offers JavaDoc authoring in mode that is very close to WYSIWYG one.		✓	
➔ Close to WYSIWYG mode of JavaDoc comment editor (according to HTML output provided by standard JavaDoc doclet)		✓	
➔ Rich HTML editing capabilities. Though, Smart.JavaDoc does not support whole set of HTML tags, however, the supported set is quite enough for creation of professional quality documentation.		✓	
➔ Ability to discover and visually highlight JavaDoc conflicts and errors such as unsupported tags, parameters and throws conflicts, missing tags etc.		✓	
➔ Ability to easily correct found JavaDoc conflicts and errors.		✓	
➔ Rich functionality of JavaDoc comment structure management (adding specific tags, adding all required tags, removing all invalid and unused tags, tags renaming may be performed in couple of clicks).		✓	
➔ Convenient functionality for navigation and instant accessing members for which JavaDoc comments should be created.		✓	
➔ Ability to filter members used for documentation creation via advanced Java Structure Component.		✓	
➔ Ability to perform two-way editing of JavaDoc comment – using both Smart.JavaDoc and usual Java source editor.		✓	
➔ Ability to browse source code while editing JavaDoc.		✓	
➔ Ability to instantly get preview of JavaDoc comment will be generated.		✓	
➔ Sophisticated multilevel undo/redo support.		✓	
Editor Enhancements			
The Smart.Braces is a tool that allows easy creation of closing braces while you are typing.	✓	✓	✓
Ability to show scope lines for all scopes of entire file that can simplify code reading and understanding in many cases.		✓	
Code Folding Enhancements			
➔ Ability to fold Java Doc comments.		✓	
➔ Ability to collapse all Java Doc comments.		✓	
Classes Highlight - allows highlighting classes used in the code.		✓	
The Thumbnail Gutter represents additional gutter placed near vertical scrollbar of editor pane and allows showing gutter marks for the whole source		✓	

file as well as navigate to them.			
➔ Ability to show gutter marks with the colors corresponding to issue priority.		✓	
➔ Ability to view description for the issue using tool tip window.		✓	
➔ Ability to navigate to the issue location.		✓	
➔ Ability to view arrangement of the class at the caret.		✓	
➔ Ability to easily understand whether there are errors or warnings in the file using appropriate icon at the top of the gutter.		✓	
The Smart.Clipboard represents several tools used for provide more efficient clipboard operations. These tools include clipboard management, swapping context of clipboard with current selection, inserting pasted Java code with correct indent along with required import statements etc.		✓	
➔ Ability to align code block on paste.		✓	✓
➔ Ability to insert import statements for classes contained in the block to paste.		✓	
➔ The Swap action allows swapping the content of the clipboard with currently selected block of code.		✓	
➔ The Pop Paste action allows consecutive popping and pasting of code fragments from the local clipboard history in the LIFO order.		✓	
➔ The Clipboard.Insight tool allows viewing of local clipboard queue and pasting one or several selected fragments in the editor.		✓	
The Auto.Indent tool allows indentation of a line at caret position or whole selected block according to the current indent level.		✓	✓
The Smart.Gutter is additional gutter placed near standard JBuilder editor gutter and is used for showing various hints concerning corresponding code in editor by arranging appropriate gutter marks.		✓	
➔ Ability to show gutter marks specific to particular case.		✓	
➔ Ability to view description of the case using tool tip window.		✓	
➔ Ability to easily navigate to the code related to the case.		✓	
The Smart.Braces.Highlight tool providers matching braces highlight and navigation operations as well as showing code fragment that corresponds to appropriate brace.		✓	
➔ Ability to highlight matching brace using background color.		✓	✓
➔ Ability to show matching brace path in the gutter.		✓	✓
➔ Ability to view matching brace code when it's has been scrolled out of view. The special popup window shows the matching brace code in the top		✓	✓

of the editor.			
➔ Ability to navigate to matching brace code using only one keyboard shortcut.		✓	
➔ Ability to work on any side of brace.		✓	
The Matching.Code.Highlight tool performs highlighting of code matching to one at caret position as well as displaying appropriate code fragment in the popup window.		✓	
➔ Ability to highlight the break target statements for return, break, labeled break (break <label>), continue, and labeled continue (continue <label> statements)		✓	
➔ Ability to highlight matching code using dotted path with arrow on the Gutter		✓	
➔ Ability to view matching code when it's has been scrolled out of view. The special popup window shows the matching code in the top of the editor		✓	
➔ Ability to navigate to matching code using keyboard shortcut.		✓	
The Changes Highlight Highlights changed lines if Java source on the gutter.		✓	
The Methods and Classes Separator tool visually separates classes and methods from each other by painting horizontal line at the top of declaration.		✓	
The Current Line Highlight tool highlights the line under cursor in the current editor with appropriate background color.		✓	✓
The Smart.Selection tool offers sophisticated code selection functionality that is based on n structure of Java program. It allows expanding/narrowing selection incrementally using appropriate code elements as well as quickly selection of whole statement, code block, method or class.		✓	
The Advanced Text View Status Bar represents a replacement of standard component that allows viewing of class and method for current caret position, caret offset, and lines count.		✓	
IDE Improvements			
Project View Synchronizer provides functionality for synchronizing of currently active file with the corresponding node in the JBuilder Project View.		✓	
Java Structure Synchronizer allows synchronizing the Java Structure View with current caret position in the editor.		✓	
The Change.ReadOnly allows easy viewing and managing the read-only status for file nodes.		✓	
➔ Ability to view "read-only" status for open files.		✓	✓
➔ Ability to view "modified" status for open files.		✓	
➔ Ability to view "read-only" status for project tree nodes.		✓	

➔ Ability to view "modified" status for project tree nodes.		✓	
➔ Ability to change "read-only" status.		✓	
Navigation Tools			
The Persistent.Bookmarks tool offers advanced bookmarks functionality. These bookmarks are persistent between sessions of JBuilder, are associated not only with editor, but also with file and JBuilder project.			
➔ General bookmarks functionality.		✓	✓
➔ Ability to link bookmarks to the file and project.		✓	✓
➔ Ability to restore bookmarks after JBuilder restart.		✓	✓
➔ Ability to assign description to bookmarks.		✓	✓
➔ Ability to view bookmarks' description using tool tip window.		✓	✓
➔ Ability to manage bookmarks using context menu one the gutter.		✓	
➔ Ability to manage bookmarks using the Manage Bookmarks dialog.		✓	✓
➔ Ability to easily navigate to bookmarks using the Persistent.Bookmarks.Navigate insight.		✓	
The View Navigator tool allows quick navigation by source elements (classes, methods, fields), issues/errors, editing points or search results.		✓	
The Navigator.Insight is specialized insight used for quick controlling of View Navigator.		✓	
The Browse.Insight tool allows quick finding Java classes with short names matching the word at the cursor position and browsing them or the appropriate help topics.	✓	✓	*
➔ Ability to find a class using package tree.	✓	✓	✓
➔ Ability to find a class using its short name.	✓	✓	✓
➔ Utilizing convenient "Insight" approach.	✓	✓	
➔ Ability to view help for found classes using Help.Insight or JBuilder help.	✓	✓	
➔ Ability to specify classes search and sorting options.	✓	✓	
➔ Ability to exclude not required packages form search result.	✓	✓	✓
➔ Utilizing classes' cache to speed up classes finding process.	✓	✓	
The Browse.Members tool allows quick finding members belonging to the current discovered context and browsing them.	✓	✓	

➔ Utilizing convenient “Insight” approach.	✓	✓	
➔ Ability to view help for found members using Help.Insight or JBuilder help.	✓	✓	
➔ Ability to specify members search and sorting options.	✓	✓	
➔ Ability to discover current class’ context at caret position.	✓	✓	
➔ Ability to change class’ context right from the Browse.Members.	✓	✓	
➔ Ability to work with inner classes.	✓	✓	
Hyperlink.Navigate is a tool that allows easy and convenient navigation through symbols definitions basing on the concept of hyperlinks.	✓	✓	
Search Results and References Highlight is a tool is intended to highlight in the editor various things found during search or find references operations.		✓	
Local References Highlight - it allows finding local references of the symbol under caret and highlighting them.		✓	
Information Tools			
Help.Insight allows easy viewing help topics, if any, for the identifier at the cursor position. Also, it provides quick help for items shown in JBuilder built-in Member Insight and Productivity! insights.	✓	✓	
Hyperlink.Help allows easy and convenient viewing help topics for particular symbols.	✓	✓	
Context.Insight is a tool that allows you to check context of the current cursor position. Context.Insight collects information about all classes and methods and shows it using the insight popup window.	✓	✓	