Interactive Window

The interactive window component enables extensions to offer REPL like experience to its users. VS Code provides the user interface and extensions provide the execution environment, code completions, execution results rendering and so on.

The interactive window consists of notebook editor at the top and regular monaco editor at the bottom of the viewport. Extensions can extend the interactive window by leveraging the notebook editor API and text editor/document APIs:

- Extensions register notebook controllers for the notebook document in the intearctive window through `vscode.notebooks.createNotebookController`. The notebook document has a special notebook view type `interactive`, which is contributed by the core instead of extensions. The registered notebook controller is responsible for execution.
- Extensions register auto complete provider for the bottom text editor through `vscode.languages.registerCompletionItemProvider`. The resource scheme for the text editor is `interactive-input` and the language used in the editor is determined by the notebook controller contributed by extensions.

Users can type in code in the text editor and after users pressing `Shift+Enter`, we will insert a new code cell into the notebook document with the content from the text editor. Then we will request execution for the newly inserted cell. The notebook controller will handle the execution just like it;s in a normal notebook editor.

arch