C# scripting with Roslyn can indeed be set up to be quite powerful and interactive, though it may not have all the built-in features of Jupyter Notebooks. Here’s how you can expand on what I’ve provided and some insights into making it more robust:

**Expanding C# Scripting with Roslyn**

1. **Custom REPL Environment**: You can build a custom REPL environment using Roslyn’s scripting APIs. This involves creating a loop that reads input, compiles it, executes it, and displays the results.

**C#**

using Microsoft.CodeAnalysis.CSharp.Scripting;

using Microsoft.CodeAnalysis.Scripting;

var scriptState = await CSharpScript.RunAsync("Console.WriteLine(\"Hello, World!\");");

while (true)

{

Console.Write("> ");

var input = Console.ReadLine();

scriptState = await scriptState.ContinueWithAsync(input);

}

AI-generated code. Review and use carefully. [More info on FAQ](https://www.bing.com/new#faq).

1. **Adding Features**:
   * **Syntax Highlighting**: Integrate with a text editor that supports syntax highlighting for a better coding experience.
   * **Error Handling**: Enhance error handling to provide more informative feedback.
   * **Variable Inspection**: Implement features to inspect and modify variables on the fly.
   * **Code Completion**: Use Roslyn’s APIs to provide code completion and IntelliSense-like features.

**Standard Features to Include**

* **Interactive Execution**: Ability to execute code snippets interactively.
* **Variable State Management**: Inspect and modify variable states during execution.
* **Error Reporting**: Clear and informative error messages.
* **Code Completion**: Basic code completion to assist with coding.
* **Documentation**: Inline documentation and help features.

**Is This an “Expert Skill”?**

Creating REPLs and interactive scripting environments is a valuable skill, especially for developers working in fields that require rapid prototyping, data analysis, or educational tools. While it may not be essential for all developers, it is certainly beneficial for those looking to deepen their understanding of language internals and interactive programming.

**Comparison with Jupyter Notebooks**

* **Jupyter Notebooks**: Offer a rich, out-of-the-box interactive environment with support for multiple languages, visualizations, and narrative text.
* **C# Scripting with Roslyn**: More lightweight and customizable, but requires additional effort to match the feature set of Jupyter Notebooks.

**Practical Use Cases**

* **Prototyping**: Quickly test and iterate on code snippets.
* **Learning and Teaching**: Create interactive coding exercises and tutorials.
* **Data Analysis**: Perform data analysis and visualization (though Jupyter is more commonly used for this).

[Would you like more detailed steps on setting up any specific features, or do you have a particular use case in mind?1](https://github.com/dotnet/interactive)[2](https://dev.to/kenakamu/time-to-learn-c-with-notebook-2gfj)

[1](https://github.com/dotnet/interactive): [GitHub - dotnet/interactive](https://github.com/dotnet/interactive)[2](https://dev.to/kenakamu/time-to-learn-c-with-notebook-2gfj): [Time to learn C# with Notebook in VS Code - DEV Community](https://dev.to/kenakamu/time-to-learn-c-with-notebook-2gfj)