VS Code Debugging

and Features

IntelliSense by VS Code

- A general term for code editing features including:
    - code completion, parameter info, quick info, and member lists
    - Sometimes called "code completion", "content assist", and "code hinting"
- Provided for JavaScript, TypeScript, JSON, HTML, CSS, SCSS, and more
- Supports word based completions for any language
- Have richer IntelliSense by installing a extension
- No built-in support for debugging C programs

IntelliSense Features

- Completions based on language semantics
- Completions based on an analysis of source code
- Suggestions will pop up as you type
    - Continue typing characters, the list is filtered
    - Pressing Tab or Enter will insert
- Can turn off while you type
- Ctrl+Space or click Info icon
    - Quick info for each method
    - Documentation for the method will now expand
    - Documentation will stay and update as you navigate
    - Close pressing Ctrl+Space again or click close icon

VS Code Debugger

- F5 : Start/Continue Debugging
- Ctrl + Shift + D : Show Debug
- Debug Actions
- Logpoints
- Line Breakpoints
- Conditional Breakpoints
- Function Breakpoints
- Data Breakpoints
- Multi-target Debugging

Rename Symbol

- Related to refactoring source code
- Some languages support across files
- Highlight a word and press F2
- Type the new desired name and press Enter

Debug Windows

- Variables Panel : View variable names and values
- Watch Panel : Monitor variables and expressions
- Call Stack : Tracking methods and functions
- Breakpoints : Lists all breakpoints, toggle, clear

Data Inspection

- Variables can be inspected and modified
- Copy Value action to copy the variable's value
- Copy as Expression action to copy expression to access
- Evaluated and watched WATCH section
- Filtered by typing while focus is on Variables section

Start Debugger and Step Through Code

- Select Debugger from the "Play" dropdown

Action	Explanation
Continue / Pause F5	**Continue:** Resume normal execution. **Pause:** Inspect code executing and debug.
Step Over F10	Execute the next method as a single command.
Step Into F11	Enter the next method to follow line-by-line.
Step Out Shift + F11	When inside a method, return to earlier execution context, complete as single command.
Restart Ctrl + Shift + F5	Terminate the current program execution and start debugging again.
Stop Shift + F5	Terminate the current program execution.

Line Breakpoints

- Go to the line where you want to add a breakpoint
  - Click on the left margin or press F9
- Symbol : red circle on the line
- Run the program in debug mode.
- The program will pause execution at the line
- Disabled breakpoints have a filled gray circle

Conditional Breakpoints

- Break execution on line of code when condition is true
    - Expression Condition and Hit Count
- Right-click on an existing breakpoint
- Select Edit Breakpoint and Enter the condition
- Symbol : red circle with a black equals sign inside

Function Breakpoints

- Break execution at beginning of a function
- On Run view right-click inside the Breakpoints section
- Choose Add Function Breakpoint
- Enter the name of the function

Data Breakpoints

- Debug menu, choose New Breakpoint > Data Breakpoint
- Type memory address or variable has a memory address
- Select number of bytes to watch in Byte Count dropdown
- Break execution when value at memory address changes
- Breakpoint hit when value changes/is read/is accessed
- Symbol : a red hexagon

Logpoints

- Variant of a breakpoint that does not "break"
- Set a logpoint by right-clicking line of code
- Select Add Logpoint
- Enter your log message, can include expressions
    - e.g., "Value: {variable}"
- Logs a plain text message to the console
- Symbol : a "diamond" shaped icon
- Can be controlled by a condition and/or hit count
- Supported by VS Code's built-in Node.js debugger

Multi-target Debugging

- For complex scenarios involving more than one process
- Multiply debug sessions are launched
- Individual sessions show up top-level elements
- Debug toolbar shows the currently active session

Debug Actions Widget

- Performed on the active session
- Can be changed
  - Using the dropdown menu in the debug toolbar
  - Selecting a different element in CALL STACK view

Triggered Breakpoints

- Automatically enabled once another breakpoint is hit
- Useful diagnosing failure cases in code
- Set by right-clicking on the glyph margin (lines)
    - Selecting Add Triggered Breakpoint
    - Choosing which other breakpoint enables
- Work for all languages
- Conditional breakpoints may also be used as the trigger

VS Code Shortcuts

User Interface

- Ctrl + B : Toggle Side Bar
- Ctrl + ` : Toggle Integrated Terminal
- Ctrl + W : To close tabs

Files

- Ctrl + P : Quick Open Menu
  - Type for File Search
  - @ Find or Type Symbol
- Ctrl + Up / Down : Scrolling with Keyboard
- Ctrl + \ in Editor : Opens a second window

Typing

- Ctrl + G : Go to line ...
- Ctrl + H : Find and replace
- Ctrl + Shift + P : Type Keyboard Shortcuts
  - Go to "Open Keyboard Shortcuts"
  - List of Shortcuts to learn and rebind if wanted
- Ctrl + Shift + Alt + Up / Down : Copy line up/down
- Ctrl + Enter : Add a line below
- Ctrl + Shift + Enter : Add a line above
- Alt + Up / Down : Move line of code
- Ctrl + [or] : Indent selection
- Ctrl + / : Toggle comment

Extensions	Extensions
Arduino	Polacode
Better Comments	Python
Code Runner	Rainbow CSV
Code Spell Checker	Remote Development
Error Lens	RSS
Git Graph	Time Converter
Jupyter	Todo Tree
Live Share	vscode-pdf
Marp for VS Code	Vim

VS Code Profiles

- Allow users to customized configurations for projects
- Developers install, activate, or deactivate extensions
- Ability to sync profiles across devices

Setup

- Go to File > Preferences > Profile > Create Profiles
    - Type a name for the new profile
    - Select a copy from option
    - Save setup time for preferences
- Switch between profiles based on use

Questions?