This page describes the VS Code Smoke test, an automatic and a manual test suite that we execute before we release VS Code to the public on all platforms.

Automated Smoke Test

https://github.com/Microsoft/vscode/blob/main/test/smoke/README.md

Manual Smoke Test

Prereqs

- For the best debugging experience you'll need node version 6.5 (or higher)
- Clone the 'express' smoke test repository:
  - `git clone https://github.com/Microsoft/vscode-smoketest-express.git`
  - `cd vscode-smoketest-express`
  - `yarn`
- In the following steps we'll produce a data directory and extensions directory from the latest **stable** release
  - we want to use **stable** to test the update behaviour from that version to our new version!
  - run `code --user-data-dir <new user data dir> --extensions-dir <new extensions dir> <vscode-smoketest-express-folder>`
  - Set `"update.mode": "none"` to disable the automatic updates if it is available for your platform
  - open some files and untitled files and leave them dirty
  - have some UI state (e.g. panel open, multiple editors)
  - close stable

Data Migration (Electron only)

This task is about running the new version over the same data of the stable released version. Specifically we want to see that the new version is starting up properly, restores all state including dirty files from a previous session.

- Run `code-insiders --user-data-dir <data dir from prereq step> --extensions-dir <extensions dir from prereq step> <vscode-smoketest-express-folder>`
- Verify the new version of VS Code starts properly and all of the UI state is restored from when you used stable in the previous step
- Verify dirty files and untitled are being restored (hot exit)

Data Loss

- Verify the following both with dirty files and untitled files
  - Make sure you get asked and can save when closing the editor itself
  - Make sure you get asked and can save when closing from Opened Editors view
  - Mac & Electron only: Make sure you get asked and can save on window close
  - Electron: Make sure that when you file > quit and reopen, dirty files and untitled restore properly ("hot exit")
  - Web: Make sure that when you close the browser tab and reopen, dirty files and untitled restore properly ("hot exit")

First User Experience (Electron only)

This task is about verifying how a first launch behaves for new users who have never started Code before.

- Delete the user-data-dir and extensions dir from previous steps

- Run `code-insiders --user-data-dir <data dir> --extensions-dir <extensions dir> <vscode-smoketest-express-folder>`
- Verify you see a welcome page and the experience is pleasant

Explorer

- Launch VSCode and open the vscode-smoketest-express folder
- On the desktop create an info.txt file
- Drag it to the explorer then rename it to README.txt
- Move README.txt to another folder and then delete it and verify it ends up in the trash
- Quick open (CTRL+P, ⌘+P) to find the .js files and other files with a similar extension
  - Verify quick open sorting makes sense and it respects fuzzy matching (e.g. 'a.s' should produce 3 results)
- Verify the Opened Editors view reflects your state of opened editors

Configuration and views

- Select File > Preferences > Settings (OS X: Code > Preferences > Settings)
  - Under editor, change line number to off, save the file, verify that line numbers are turned off without refresh.
- Select File > Preferences > Keyboard Shortcuts (OS X: Code > Preferences > Keyboard Shortcuts)
  - Change a binding of an action and verify the new binding works. e.g. Look for command `workbench.action.toggleSidebarPosition` and change its keybinding to `ctrl+u` and verify it works.
- Verify that "Go to line" works (ctrl+g)

Search

- Use Search (CTRL+SHIFT+F, ⌘+SHIFT+F) to find `body`
  - Verify that 16 results in 5 files show up
  - Verify you can run the search filtering for `*.js` files
  - Verify you can dismiss files from search results
  - Verify you can replace search results with a replace term

Multi Root

- Add more than one folder to the workspace
  - Verify you can use the explorer to open files from all folders
  - Verify you can search across all folders

Languages - CSS

- Open file style.css
  - verify quick outline (CTRL+SHIFT+O, (⌘+SHIFT+O)
- Add an empty rule `.foo{}`
  - verify you can see a warning in the editor
  - verify you can see a warning from the Problems view
- Select File > Preferences > Settings (OS X: Code > Preferences > User Settings)
  - change `css.lint.emptyRules` from warning to error
  - Go back to style.css, verify that the warning turns into an error.
  - Verify you get css intellisense

Markdown

- Open `readme.md` and open the preview side by side
- Make changes to the `readme.md` and verify they are reflected live on the preview

JavaScript

- Open `bin/www`
    - Show the quick outline (`CTRL+SHIFT+O`, `⌘+SHIFT+O`) verify that entries show up and make sense
    - From the context select Find All References to `app`
    - From context menu use Rename Symbol to rename a local variable
    - Verify code folding works
- Open app.js
    - Verify go to definition and peek definition work
    - Format the code

Debugging JavaScript

- In VSCode switch to debugging viewlet
- Make sure you see the debug start view
- Click on 'create a launch.json file'
    - make sure that VSCode automatically detects ${workspaceRoot}/bin/www as the 'program' attribute
- Set a breakpoint in index.js:6
- Press `F5` to start debugging. Verify:
    - workbench transforms into "debug mode" - glyph margin and status bar turns orange
- Open browser at http://localhost:3000/
    - verify the breakpoint in index.js gets hit
- Verify step over, step in, continue work
- While stopped, verify:
    - you can hover over values
    - clicking on a stack frame shows variables associated with that stack frame
    - you can add a watch expression
    - you can evaluate a expressions in the debug console
- Verify you can stop the debug session and that workbench exits "debug mode"

Git

- Open git viewlet
- Verify that the changes you made are showing up
    - if not, make some new changes
    - verify the changes in the diff viewer are reflecting the changes you did
    - verify multi select works
- Commit the changes locally
    - verify that an outgoing change shows up in the status bar of VSCode
- Clean up: click on the ellipses in the git viewlet and choose 'Undo Last Commit'

Integrated Terminal

- Open the integrated terminal
- Run a command and verify the output makes sense (e.g. `ls` or `dir`)

Status bar

- Quickly click on all the actions in the status bar and verify they behave as expected
- Execute various commands from the status bar (change language mode, indentation...)

Extensions

- Install an extension that you can easily verify works (e.g. new theme)
  - Verify that the extension gets installed and that on restart of VSCode it works
- Uninstall the extension and verify it is no longer present after VSCode restarts

Accessibility

- Turn on
  - OS X: Voice Over
  - Windows: nvda, command palette > 'Toggle Tab Key Moves Focus' (`CTRL+M`)
  - Linux: n/a
- Tab through the whole workbench and verify what you hear makes sense. Also verify you can tab back into the location from where you started
- Check high contrast theme

Localization (Electron only)

- Install the German Language Pack extension: `ms-ceintl.vscode-language-pack-de`
- Start code from the command line with `--locale=DE`
- Verify all menus and viewlets are in German (or your language of choice)