Since the 40-page manual is probably overkill, I used ChatGPT to help design an outline for a screencast that was more user friendly. This was the result.... Version 10-25-2025

AutoCorrect2 & Hotstring Helper 2 – Full Demo Script and Narrative

This guide is designed to accompany a screencast or live training session demonstrating the AutoCorrect2 system, Hotstring Helper 2 (hh2), and related tools like MCLogger. It's intended for beginners and intermediate users who want to build an evolving, efficient, and personalized hotstring-based autocorrect system.

1. Why AutoCorrect2?

Many of us spend time retyping the same text, fixing common typos, or repeating phrases. Hotstrings are tiny shortcuts that automatically expand into text — reducing keystrokes, fixing errors, and increasing speed. AutoCorrect2 builds on AutoHotkey to make hotstrings easy to create, manage, analyze, and optimize. Over time, your hotstring library becomes a living system that adapts to your typing habits.

Introduce the concept of typing efficiency and automation. Set the stage by explaining how this tool saves time and reduces errors.

2. What Are Hotstrings?

A hotstring is a text trigger that automatically expands into a predefined replacement. For example, typing 'btw' followed by a space can automatically turn into 'by the way.' Unlike hotkeys, hotstrings require no special key combinations — just regular typing. They work across most Windows applications, making them powerful and universal.

Demo idea: Type ::btw::by the way into a simple AHK script, run it, and show how it expands.

3. Hotstring Helper 2 – Core Components

Hotstring Helper 2 (hh2) is the graphical interface of AutoCorrect2. It allows rapid creation and editing of hotstrings without hand-editing script files. Main UI elements:
- Options Box (hotstring behavior)
- Trigger Box (the shortcut)
- Replacement Box (expanded text)
- Append Button (save to your library)
- Spell Button (fix typos using LanguageTool)
- Exam Pane (advanced word analysis).

Keep this section visual and slow-paced. Show the interface and hover over each element.

4. Creating Boilerplate Hotstrings

Boilerplate entries are hotstrings that insert large chunks of text—like signatures, contact info, or standard replies. To create one:
1. Select the desired text.
2. Press Win + H to open hh2.
3. hh2 auto-generates an acronym trigger.
4. Press Append.
5. Type the trigger in any app to expand it.

Tip: Demonstrate how prefixes (e.g., semicolons) prevent unwanted triggers.

5. Creating Autocorrect Hotstrings

Autocorrect entries fix common typos automatically. Steps:
1. Select the typo in any document.
2. Press Win + H.
3. Use Spell to fix the replacement text.
4. Append the hotstring.
5. hh2 can auto-enter the corrected word immediately.

This is where users experience the 'magic moment'—show a live example.

6. Multi-Match Optimization and Confounding Mistypings

One of the most powerful features of hh2 is the Exam Pane. It allows trimming of trigger strings to cover more words. For example, trimming 'combonation' to 'combin' lets a single hotstring fix dozens of words with similar roots.

But this power comes with a trade-off:
- Shorter triggers = higher coverage but more ambiguity.
- Longer triggers = more precision but fewer matches.

A 'confounding mistyping' occurs when an over-generalized trigger unintentionally changes an unrelated word—for example, a trigger meant to fix 'combonation' might accidentally modify 'trombone.' The Exam Pane shows both Fix and Misspell counts, and web frequency to help you find the right balance.

Use real examples and show how trimming affects the number of Fixes and Misspells. Emphasize practical judgment.

7. Validity Checks & Conflicts

hh2 validates your hotstrings before adding them. It detects:
- Invalid option characters
- Duplicates
- Subset/superset conflicts (nullification)
- Previously removed strings
- Potential misspellings

This keeps your library clean and functional.

Show an intentional conflict, such as two overlapping triggers. Explain nullification vs actual conflict.

8. Helpful Extras & Tools

Beyond basic hotstring creation, hh2 includes:
- Suggester Tool for exploring better trigger variants
- WordNet / GCIDE dictionary lookups
- Secret Control Panel for advanced features
- Integration with word frequency lists for smarter decisions

Encourage users to explore these gradually rather than all at once.

9. MCLogger – Evolving Your Hotstring Library

MCLogger passively logs promising typos and phrases as you type throughout your day. Later, you can review these logs and convert the useful entries into actual hotstrings with hh2. MCLogger works even outside of AutoCorrect2, making it useful for building vanilla hotstring lists too.

Position MCLogger as a 'field research' tool. It helps your library grow organically rather than all at once.

10. Library Evolution: Growth and Pruning

Over time:
- New entries are added through hh2, MCLogger, and other sources.
- Old or unfit entries are removed using ACLogAnalyzer.
- High-utility entries stay, weak ones are culled.

This creates a continuously evolving and efficient hotstring system.

Draw parallels to biological evolution: variation, selection, retention.

11. Best Practices for Long-Term Use

• Favor multi-match entries where appropriate, but avoid overgeneralizing.
• Separate boilerplate and autocorrect lists.
• Protect uncommon but important words.
• Validate regularly and analyze logs.
• Keep your library well-organized for quick maintenance.

This is the 'philosophy' slide—reinforce habits that keep the tool useful over years.

12. Resources & Next Steps

• GitHub: github.com/kunkel321/AutoCorrect2
• AHK Docs: autohotkey.com/docs/v2/Hotstrings.htm
• LanguageTool.org API
• Kaggle word frequency dataset
• Practice: Build five useful hotstrings today and test them in real-world typing.

Wrap up with encouragement to take action immediately.