4 Steps to Successful Participatory Live Coding

Adapted from: *Christina Koch & Greg Wilson (eds): "Software Carpentry: Instructor Training." Version 2016.06,* https://github.com/carpentries/instructor-training*,10.5281/zenodo.57571*

See also https://youtu.be/bXxBeNkKmJE & https://youtu.be/SkPmwe_WjeY

1. Prepare Ahead of Time

✓ Set up your computer — Learners must be able to see, follow, translate, and recreate what you do.
See **Configuring a Computer for Live Coding**

✓ Pre-test all learner actions — Use a fresh, default install, and follow your plan. Even seemingly trivial differences may affect later steps. Avoid teaching what you do not intend or need to.

✓ Put notes elsewhere — Put lesson materials and notes on a separate laptop or tablet, or print them.

✓ Move tables for viewing — Have light, but not right above the presenter or screen. Position tables so learners can see.

2. Create the Right Atmosphere

✓ Keep learners confident — Avoid demotivating statements and actions.
See **Motivation and Demotivation**.

✓ Have fun — Humor is good, even saying ‘that was fun’ after something worked well is a good start.

✓ Embrace & use mistakes — Model a mastery orientation. Learners are more likely to seek help if you are approachable and can relate to their difficulties.

✓ Minimize distractions — Turn off notifications on all devices.

3. Help Learners Keep Up

✓ Narrate everything

Talk about your intentions and actions
See **Narrating Participatory Live Coding**

✓ Start before intros or breaks

Errors and issues are likely to reveal themselves when learners open software, import data, or load packages. Prepare for the next section early to allow time for troubleshooting.

✓ Identify struggling learners

Encourage questions: bring and use colored sticky notes for learners to indicate status.

✓ Bring others to assist

Recruit helpers and have them roam.

4. Be a Good Teacher

✓ Know your audience

Survey your learners to see what they know.

✓ Put your lesson in context

All learners need to know why a particular topic is useful, software is no different.

✓ Control what is covered

One struggling learner may indicate wider problems, but should not alone determine pace, nor should an expert make you cover less-needed skills.

✓ Move to maintain interest

Point to the screen, draw on a board, and walk around (use a microphone if available). Updating drawn diagrams as you cover more skills helps learners understand & organize content.

Motivation and Demotivation

"*None of us go into a workshop with the intention of creating a hostile environment or making the learners hate the tools we're teaching, but we can all accidentally do just that if we don't pay attention to what we say and how we interact with our learners.*"

https://carpentries.github.io/instructor-training/08-motivation

People learn best when they believe they can. Keep learners confident and secure:

- **Be positive**—do not talk disdainfully about any OS, software, or practice.
- **Be genuine**—avoid sarcasm, veiled criticism, taunts or quips.
- **Be patient**—let learners accomplish tasks; do not use their keyboard or mouse.

- **Assume** learners have questions and seek them: "What questions do you have?"
- **Do NOT assume** what a learner knows, how difficult performing a skill is, or how quickly they will learn. It may not be easy, obvious, trivial, or fast.

Narrating Participatory Live Coding

"For every command you type, every word of code you write, every menu item or website button you click, say out loud what you are doing while you do it. Then point to the command and its output on the screen and go through it a second time."

https://carpentries.github.io/instructor-training/14-live

Learners are focused on their own computer and may not see your actions.
Work slowly and always narrate what you are doing so they can follow.

- Say out loud what you are **typing** or **clicking** while you do it.
  - Using keyboard shortcuts is invisible, so be sure to say every time you use one. Ensure that learners understand and can use them, too.
  - Avoid copy-paste; learners cannot follow and need practice typing syntax.
- After you achieve a **result**, point to the projected screen (if possible) and describe again what you did then discuss the output or result.
- If you **make an error**, calmly point out the message or incorrect output. Discuss the error when relevant and narrate all the steps you take to fix it.

Images: Caroline Taylor, editing. Originals by ijmaki: https://safesearch.pixabay.com/illustrations/social-social-networks-1206612 and 1206603

Configuring a Computer for Live Coding

Learners must be able to…

SEE

Even learners with great eyesight can have trouble seeing when they are not right in front, especially with poor lighting. When it is easier to see, it feels easier to do.

- **Maximize** windows to fill the viewable area. Check if the bottom is visible and adjust.
- Increase **contrast** between text and background (e.g., black font, white background).
- Increase **font size** for syntax-based software, but avoid too much line wrapping.
- Make everything **bigger** as needed, but watch for dialog boxes getting cut off.
  - Lower the screen resolution (connecting to the projector may do this); or
  - Increase the scaling in the Accessibility settings.

FOLLOW

Learners must look back and forth between your screen and theirs. To keep up, they must be able to quickly identify your current or most recent action(s).

- Display a **log or history** in syntax-based software. Some are not visible by default.
- Highlight the **cursor**, if used—use special software or increase the size and contrast
- Remove **clutter**—close/hide extra windows or programs, simplify prompts.

TRANSLATE

Learners can more easily copy your actions if your screen looks the same as theirs. Balance this objective with simplification, considering the software and learners.

- Be familiar with differences between **platforms** (Mac, Windows, Linux).
- In general, use the **default** color-scheme with the background and syntax highlighting.

RECREATE

Learners get frustrated if they copy you but get a different result. If you heavily customize, consider having a separate user account for teaching.

- Check that you have the same software **version** as learners, usually the most recent.
- Remove (or avoid) custom keyboard **shortcuts**, text expansion, or command aliases.
- Disable custom **startup** scripts and restore default settings.