What the smoke test checks

The **smoke test** is a script in the workspace, `scripts/smoke_test.sh`, that checks whether the EDA tools and the SKY130 PDK resolve inside the container. You'll run it in Lab 03, Part 4. This page explains what each check looks at and what a failure usually means.

Two terms

- Your **`PATH`** is the list of folders your shell searches when you type a command name. A tool "resolves" when the shell finds it on the `PATH` after you type its name.
- A **PDK** (process design kit) is the set of files that describe how one chip fab's manufacturing process behaves, so that tools can simulate your designs and check them against it. The courses use SKY130, an open PDK, which the tools call `sky130A`. Inside the container its files are in `/foss/pdks/sky130A` (`$PDK_ROOT` is `/foss/pdks`).

The environment file

Inside the container, your `workspace` folder appears as `/foss/designs`. The file `/foss/designs/common/.designinit` sets up the shell for the courses: it sets the environment variables the tools read, points the PDK paths at SKY130, and defines the `mod` command. It leaves your `PATH` alone, because the image's own setup already puts the tools on it in every terminal. Lab 03 has you load it with `. /foss/designs/common/.designinit` before you run the smoke test. The leading `.` tells the shell to run the file in your current shell rather than in a new one, so its settings stay in the terminal you're using.

The checks

Check	Role
`ngspice`	Circuit simulation
`xschem`	Schematic capture
`magic`	Layout
`netgen`	LVS (netlist comparison)
`yosys`	RTL synthesis
`sky130A` under `$PDK_ROOT`	The SKY130 PDK is present and is the one in use
`mod`, `add_module.sh`, `ic101_setup`	The course module helpers work: `mod` is defined, `add_module.sh` is present, and `mod` lists and enters this course's `ic101_setup` folder

When every check passes, you'll see a column of `OK` lines, along with an ngspice version line and a `--- module helpers ---` heading, and the last line is `=== All checks passed ===`.

When the PDK check fails

If this check fails, the PDK is almost never missing, because SKY130 ships inside the image at `/foss/pdks/sky130A`. The usual cause is that the image starts on a different PDK, `ihp-sg13g2`, and the tools look in the wrong place until the workspace environment switches everything over to SKY130. Loading `.designinit` is what makes that switch. Troubleshooting has the exact messages and the fix.

Course modules and `mod`

Each course keeps its files in a folder under `modules/` in your workspace, which the container sees as `/foss/designs/modules/`. Once you've loaded `.designinit`, the `mod` command moves you between those folders:

Command	What it does
`mod`	List the course modules
`mod ic101_setup`	Go to this course's scratch folder, `/foss/designs/modules/ic101_setup`
`mod add ad101`	Clone `github.com/uoftasic/ad101` into `modules/ad101` (a later course)
`mod ad101`	Go to `modules/ad101`, once it's cloned

On your laptop, running `./scripts/add_module.sh ad101` from your `workspace` clone does the same thing as `mod add ad101`, because that folder is the one the container mounts as `/foss/designs`, so both commands write into the same `modules/` folder.

`mod add <course>` first prints the address it clones from, then git's own `Cloning into ...` line. If the clone works, the output ends with:

```
OK  module ready: /foss/designs/modules/<course>
    run: mod <course>
cwd: /foss/designs/modules/<course>
```

The `cwd:` line means `mod add` has already moved you into the new folder. If it stops and asks for a GitHub username instead, see Troubleshooting.

Next: Lab 03: Launch the workbench.