Teaching Coding, Data Science and Building a Community

Stephanie Boyle

@_stephanieboyle

/stephanieboyle9

What should we teach?

How should we teach?

Why should we teach?

< 2008

Working, travelling, faffing

2009

B.Sc. Psychology

2013

Ph.D. Cognitive Neuroscience

2017

Statistical Programmer Pharmaceuticals

2019

Head of Data Delivery CodeClan

What should we teach?

SQL
AGILE
Azure
Power BI
APACHE
Spark
hadoop
aws
Google Cloud Platform

Ethics

Testing

Documentation

Production

Reproducibility

Reviewing

Interpretability

Flexibility

Client Culture

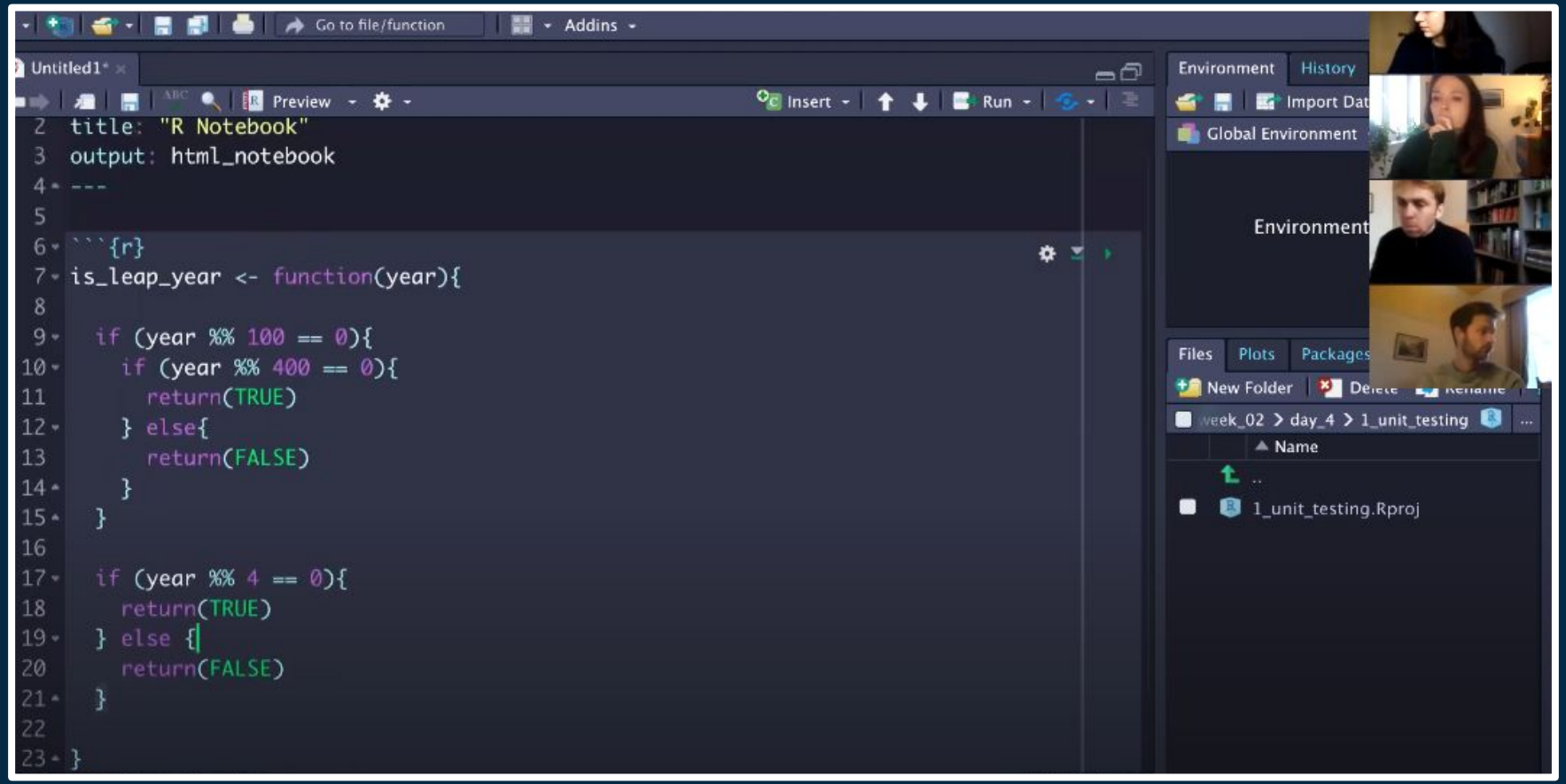
Communication

T-shaped people

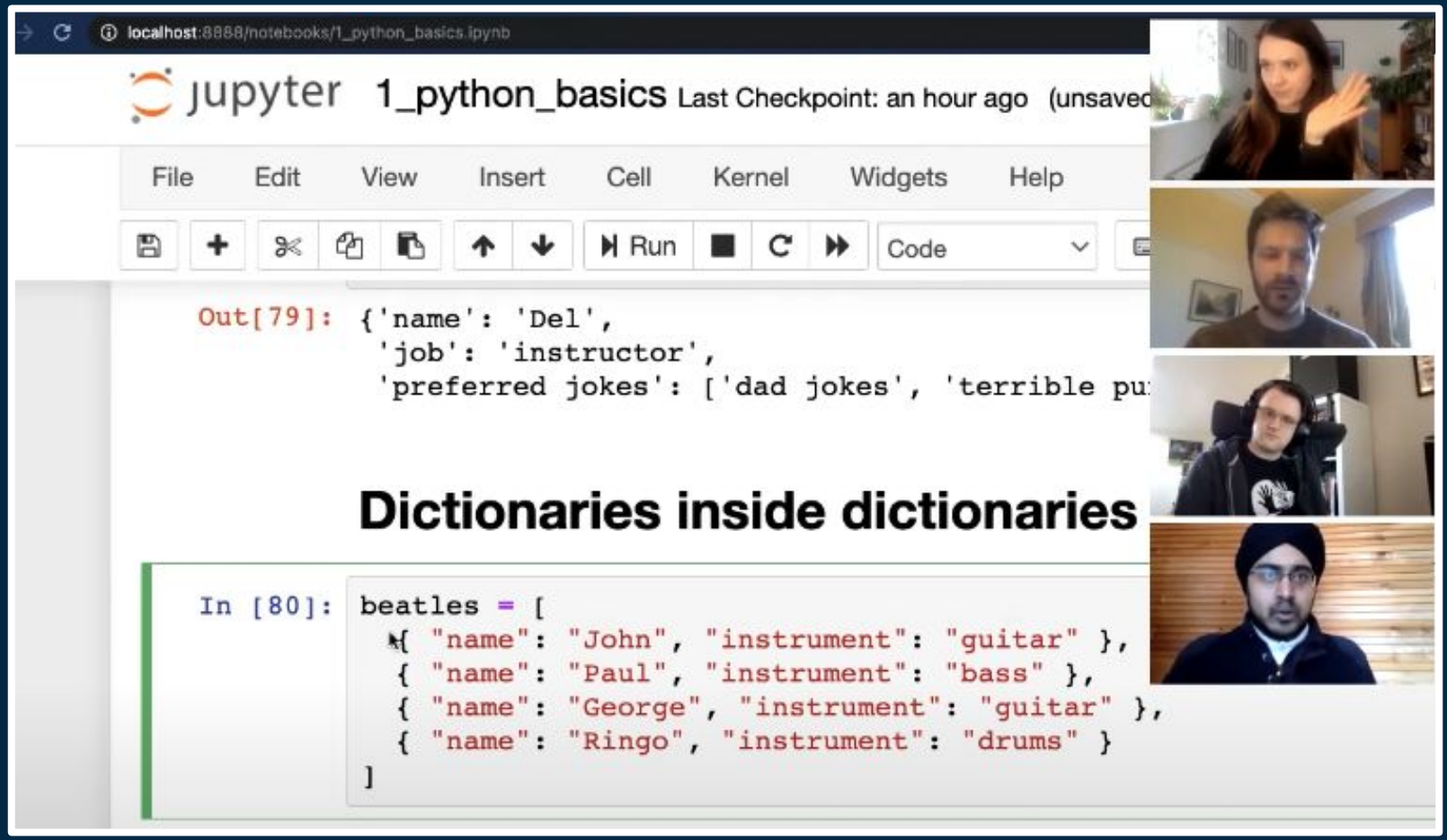
What you want people to know is often different than what people need to know

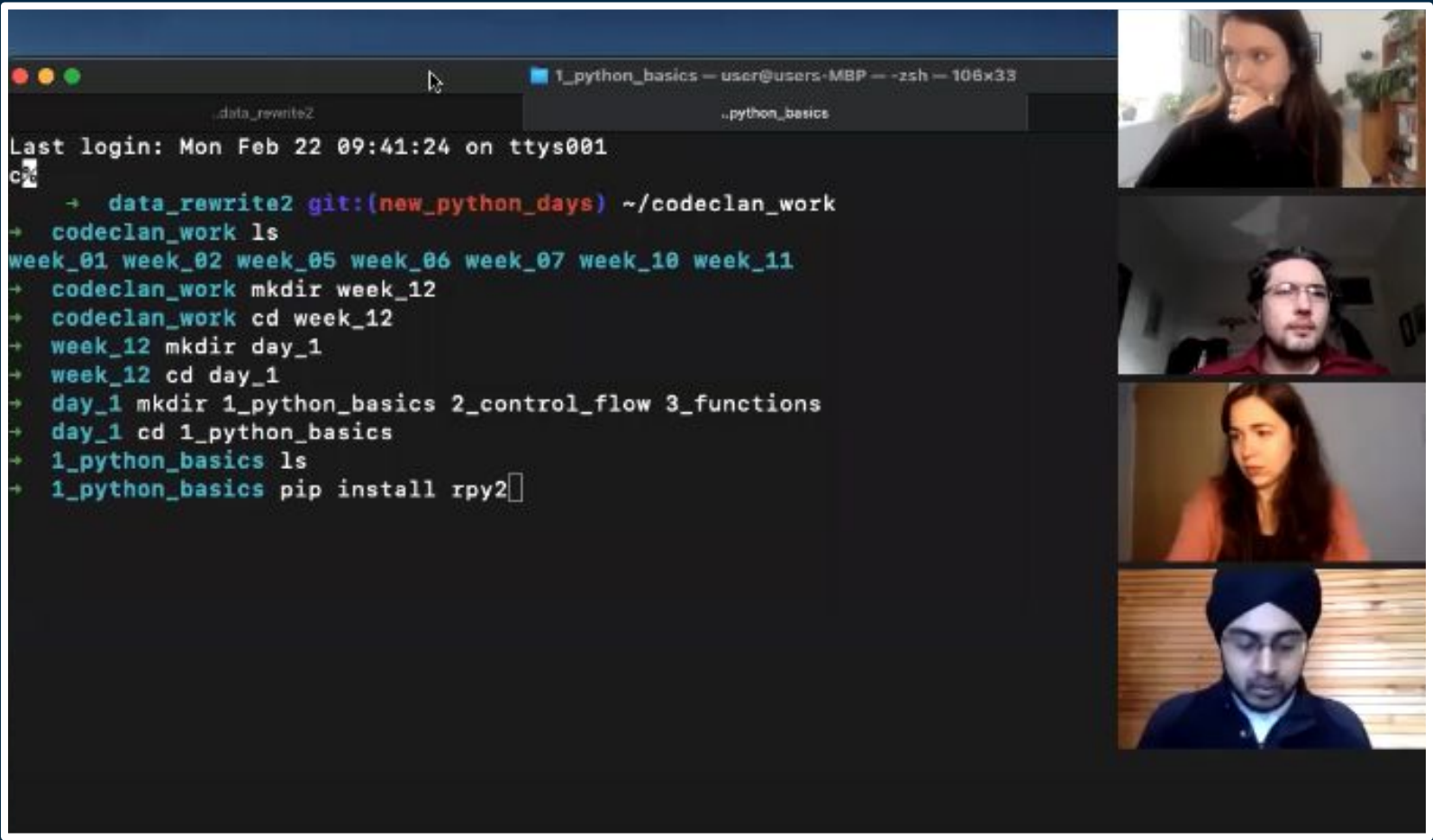
How should we teach?

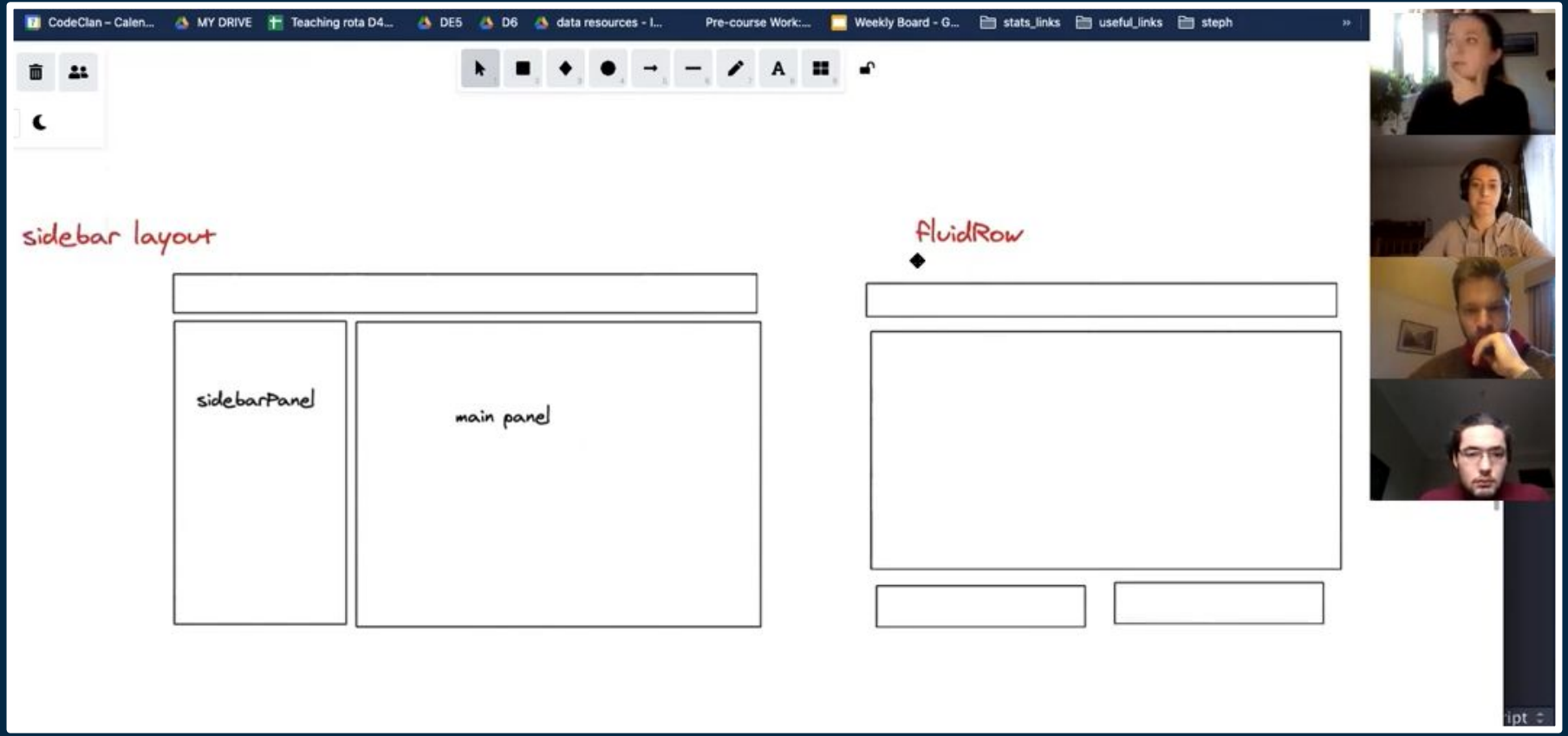
- Gives hands on coding experience to learners. Doing > Talking.
- Shows you make mistakes too. It’s ok not to know something.
- Creates a safe space to fail.
- Teaches the how and why, not just syntax.
- Interactivity leads to questions - which you google. Emphasises that it’s not important to memorise everything.


Go to file/function
Addins
Untitled1*
Preview
Insert
Run
title: "R Notebook"
output: html_notebook

```{r}
is_leap_year <- function(year){
if (year %% 100 == 0){
if (year %% 400 == 0){
return(TRUE)
} else{
return(FALSE)
}
}
if (year %% 4 == 0){
return(TRUE)
} else {
return(FALSE)
}
}
Environment
History
Import Dat
Global Environment
Environment
Files
Plots
Packages
New Folder
Delete
week_02 > day_4 > 1_unit_testing
Name
..
1_unit_testing.Rproj
```


localhost:8888/notebooks/1_python_basics.ipynb
jupyter 1_python_basics Last Checkpoint: an hour ago (unsaved
File Edit View Insert Cell Kernel Widgets Help
Run
Code
Out[79]: {'name': 'Del',
'job': 'instructor',
'preferred jokes': ['dad jokes', 'terrible pu
Dictionaries inside dictionaries
In [80]: beatles = [
{ "name": "John", "instrument": "guitar" },
{ "name": "Paul", "instrument": "bass" },
{ "name": "George", "instrument": "guitar" },
{ "name": "Ringo", "instrument": "drums" }
]


1_python_basics — user@users-MBP — -zsh — 106×33
..data_rewrite2
..python_basics
Last login: Mon Feb 22 09:41:24 on ttys001
→ data_rewrite2 git:(new_python_days) ~/codeclan_work
→ codeclan_work ls
week_01 week_02 week_05 week_06 week_07 week_10 week_11
→ codeclan_work mkdir week_12
→ codeclan_work cd week_12
→ week_12 mkdir day_1
→ week_12 cd day_1
→ day_1 mkdir 1_python_basics 2_control_flow 3_functions
→ day_1 cd 1_python_basics
→ 1_python_basics ls
→ 1_python_basics pip install rpy2


CodeClan – Calen...
MY DRIVE
Teaching rota D4...
DE5
D6
data resources - I...
Pre-course Work:...
Weekly Board - G...
stats_links
useful_links
steph
sidebar layout
sidebarPanel
main panel
fluidRow

Why should we teach?

Juniors

- How your organisation works
- Coding != being perfect
- It’s ok to not know everything
- Learn how to learn
- See that googling is option the best (and fastest) option
- Be ok with failure and mistakes

casey
@varcharr

If you want to succeed in tech, learn how to learn.

That's the key. That's how experts are experts.

The "stuff to know" changes too often to depend on knowing one thing or set of things, and different environments need different solutions.

Keep an open mind and keep learning.

5:08 PM · Feb 26, 2021 · Twitter for iPhone

People rarely want to leave jobs where they have room to learn, grow, and thrive.

Invest in people, build them up, and you'll set them (and your organisation) up for success

Elephant in the room...

Training while remote

Totally doable to train remotely

IF YOU INVEST

- Hire people with high motivation to learn
- Move organisational resources to support the teams
- Assign a dedicated mentor for them
- Give the mentor credit for it
- Balance paired tasks / projects / solo work
- Make sure they’re involved in daily stand ups

Gergely Orosz
@GergelyOrosz

Replying to @GergelyOrosz

Every single one of the interns we invested in their first 6-12 months have gone to grow at incredible speeds after: and they are also often the best mentors for new interns.

The kicker? Their mentors grew a lot as well!

If you can't invest, don't expect these rewards.

Those who can, do

Those who can't, teach

NO

Seniors

Those who can, should

You probably already, do

Seniors

- Reinforces knowledge of seniors
- Teaches them how to be mentors/managers
- Teaches them new skills & tech
- Pays it forward
- Lets juniors see the way seniors work
- Creates a strong community

Juniors

Seniors

The difference between is training and experience.

Community is hard to come by,
but it is so worth it

- Create a culture (in your classroom, office, or workforce) where sharing and learning is key
- Build a learning culture where everyone can contribute
- If people feel comfortable, they'll ask
- Don't have strong players dominating - weaker and strong need to co-exist
- You'll have a workforce that can work together

Thanks!

Stephanie Boyle

@_stephanieboyle

/stephanieboyle9