Software Engineering

Data Scientist
- → Research
- → Explore/Model data

SWE
- → Build Products

→ Both code & write programs

→ Part I & II: SWE Practices

→ Clean & Modular code

- → Clean: readable, simple, precise
- → modular: code broken into functions & modules

- Refactoring
  - ↳ Restructuring code to improve internal structure

- Meaningful names
  - ↳ descriptive: guess function from name.

meaningful names cont.

↳ try to make data type clear from name

→ name exceptions depends on audience

- Add whitespace
  ↳ breaks up code & makes it easier to look at

- Modular code
  ↳ Helps you to not repeat yourself
  ↳ Don't overdo it.

→ Efficient code
  ↳ reduce runtime
  ↳ reduce memory space
  } depends on circumstance

  ↳ make use of vectorization & package functions where possible

→ Documentation

- Make comments
- doc strings in function
- README

↳ Docs

"""

<description>

<Args>

<arg>: <type> <description>

Returns:

<ret>: <type> <desc.>

"""

→ Version control

↳ makes it possible to work on multiple features at once.

↳ commit multiple models of a ML model.

→ Testing

↳ Double checking that code does what is expected.

↳ Done via a module that tests each function & prints errors to bring awareness to parts of the code that need work

pytest ← Allows for testing of code

↳ won't stop code at assert statements that fail

↳ names of test module & test funcs. need to start w/ test_

- Test driven development

↳ Create unit tests before writing code, will pass when all tests pass.

→ Logging

Helps to see if why production code crashes.

↳ logging comes at multiple levels

Debug ← normal activity of program
error ← records errors
info ← records user or sys input

→ Code reviews

↳ catch errors & improve readability
↳ Share knowledge

→ Part III: OOP

- Procedural vs OOP paradigms
  - Procedural: executes code in order
  - OOP: executes based on objects

Objects → characteristics, actions

- Classes, methods, objects & attributes

Obj. vs class
- class: is instance of
- an instance of a class will have similar attributes & methods

→ How to create a class

```
class <name>:
    def __init__(self, inputs):
        self.inputs = inputs

    def methods...
```

→ Magic Methods

↳ override normal operations

ie gauss_1 + gauss_2 doesn't mean anything

∴ override '+' w/ def __add__

↑ many such fns exist.

Note that __init__ is also a magic.

__repr__ ← magic for printing an object.

→ Inheritance

↳ Class w/ similar attr. & methods to other classes.

clothing → sock → sock1
sock2

If Gaussian inherits from distribution

then class Gaussian(Distribution)

' in the -- init --, the parent class will also need to be instantiated

→ Web development

↳ components of a web app

HTML] Text, images, etc.

CSS] colors, fonts

Animations] Javascript

server

database

PhP/Java/Ruby

→ Front end

↳ HTML/CSS/Javascript. ⇒ Bootstrap

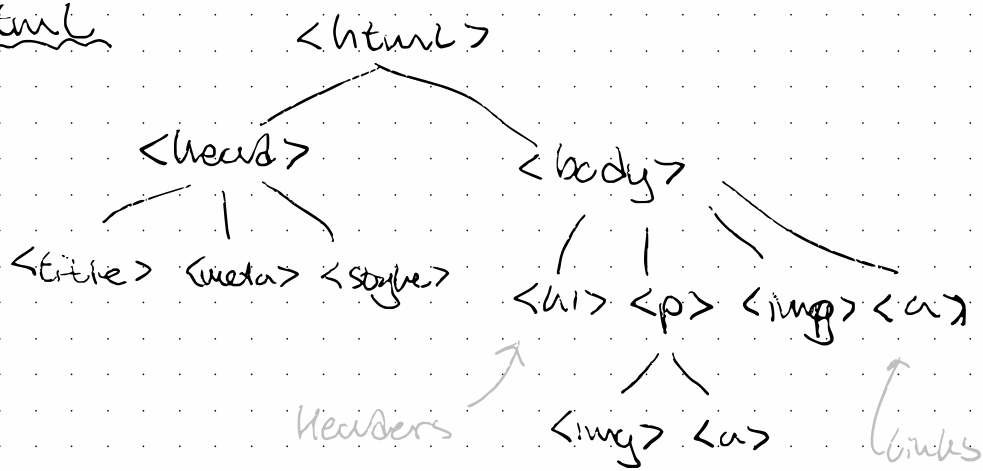
↳ HTML (Hypertext Markup Language)

↳ each section of document defined by a tag <tag>

↳ Note: HTML is not sensitive to tabs & formatting

→ Structure of a page

Html



its possible to add ids to tags for reference. when formatting w/ CSS

ie <body id = 001 class = italix>

↑ CSS added individually

↑ same transform for many sections

CSS (cascading style sheets)

HTML gives structure for text. CSS makes it beautiful.

↳ possible to add "style" w CSS in html tags.

```
<p style = "color: red;">
  Hello world
</p>
```

← Styling individually is tedious

Solution! Style sheet

```
<style>
  # italics { font: italic }
</style>
```

← all css gets added here

↑ usually done in another file

```
<link rel = "style sheet" type = "text/css"
      href = "style.css">
```

Javascript

Makes websites interactive

↳ can add js in src of a tag aswell

↳ Oh god, fuck this

Bootstrap

↳ takes care of css for you so that you only need to write html & css

↳ Divides page into 1 row & 12 cols
Styles CSS & everything else for you

row class ← divide up screen as necessary

div col 4 } 2 cols, 1 w/ 4
div col 8 } 1 w/ 8 } Allows for 12 cols in total

Plotly

↳ takes a js file & makes a plot with it?

variables defined w/ var

↳ lists as in python (dicts too)

↳ possible to define plots you want by checking the documentation.

↳ link to js files & plots will appear in webpage

↳ first define data

↳ then create traces for

Backend

How do you get your website online via internet?

"Flask" ← deploy app to server

will render in browser

How to publish on the internet

↳ Heroku (Salesforce)

alternatives · Amazon Lightsail
· Microsoft azure
· Google Cloud
· IBM Cloud