

# DEVELOPMENT WORKFLOWS

How a list of goals get worked on

- Past: Waterfall
- Recent: Agile

# **WATERFALL MODEL**

Each step done before next starts, back up if problem

Common key: All requirements in advance

- Fantastic for "known" tasks
- Great for hard requirements:
  - features
  - resources (incl people)
  - deadlines
- Generally terrible for most software
  - slow
  - inaccurate

# AGILE DEVELOPMENT

<https://agilemanifesto.org/>

Often struggle to actually adopt

- Companies want hard accuracy (it is a lie)

Common ideas:

- produce working code very frequently (no guess)
- cannot "make up", if behind, change one of:
  - resources
  - features
  - deadlines

# SPRINTS

Two common styles:

- "Sprints"
  - 1wk/2wk/1month runs
  - work assigned at start
  - expected to be done at end
- "Kanban"
  - constant flow of work
  - no task pauses long (~1-2 days)

Both involve small, defined, concrete tasks

# TASKS / STORIES

- something worth tracking if done
  - depends on company
- has some meaningful result
- not too large (2 days or less)
  - break up bigger tasks

# ESTIMATION

**We are terrible at estimating software dev times**

Minimize impacts

- Small tasks
- Regular check-ins to confirm status
- Keep repo in usable state

Establish "velocity"

- Consistently bad estimation

# CODE REVIEW

- NOT judgment of you
- Ideally not a rubber-stamp
- chance to improve
- consider longer-term impacts
- consider wider issues
  - consistency
  - compatibility
- should be done promptly
  - don't block others

# STANDUPS

Intended to be a fast meeting

- Share status
- Identify problems

Dos and Don'ts:

- Don't take up extra time just because you feel you need to justify your time
- Do listen for things to learn
- Do listen for issues others might not see
- Do identify your potential problems