Agile Software Development II
One philosophy, many incarnations

Overview
What’s agile again?
Agile management methods
Organizing code for agility

The agile model

Common agile coding practices

TDD/BDD
Use the tests to develop your thinking about interfaces, and as lightweight spec to guide the development of the actual code.

Code review
Improve your own skills, improve code quality, share knowledge.

Continuous integration and delivery
Ensure that new code is always up to standard.
Keep the path to release clear; make releasing easy.

What are the different incarnations of this model?

Many common coding practices
TDD/BDD
Code review
Continuous integration
Continuous deployment

Plentitude of management and collaboration strategies
Scrum, Kanban, Extreme Programming, etc.

We'll do it live!
Example: Scrum

Roles: Product Owner, Dev Team, (Scrum master)

Sprint Planning
  Retrospective (to encourage continuous improvement)
  User stories from the backlog are prioritized and estimated

The actual sprint
  Daily standup meetings to review progress, remove blockers
  Demo day

Recommended resources

Task management: Pivotal Tracker, Asana
Version control: Github (workflow basics, CLI basics)
Test frameworks: googletest (C++), nosetest (Python), rspec (Ruby), jasmine (Javascript)
Continuous integration: TravisCI, CircleCI
Deployment/Hosting: Heroku

Recap

Agile process defines certain priorities, but leaves a lot of room for specific workflows
Some coding practices are increasingly common in the quest for agility