- Moved test files out into their own package
- Created domain/application split between Builders/interfaces and domain classes
- Used creater pattern on exercise
- Used factory pattern for User
- Used Interface pattern on Trainer/Customer (inherit from User)
- Used Facade Pattern on CLI to handle complex tasks
- Used Creator pattern on Workout
- Used singleton pattern on CLI
- To reduce coupling and improve ve cohesion: Took some cues from MVC and implemented Controllers and Models. Models handled storage, and Domain classes became more like “Types”. Controllers managed updating models and were used by CLI.
- DRY used when refactoring CLI
- **Tried and reverted** Used Creator pattern on Customer
- **Tried and reverted** Used Creator pattern on Trainer
- **Tried and reverted** Used sub creators and moved methods that objects “know about” into the right places, EG Manager for information-expert pattern instead of having a single, “God” CLIController
- Used interfaces for Controller and Model
- Used Abstract Class on user
- Used singleton pattern on models