Time	~ 1 hour
Learning Goals	• Write rspec tests for the public interface of a given class

A bank account provides a useful example for discussing object-oriented design because the needs for security and privacy are imminent and obvious. A bank account should not reveal certain things about itself so easily, like the account number, for example.

In this challenge, you will write the examples needed to thoroughly test the public interface of the `Account` class.

The public interface of the class represents its contract with the rest of the application: each public method is agreeing to provide some behavior and/or return value given the right inputs and state.

The purpose of a good test suite is to ensure that the contract is honored.

Objectives

Grab the files from the gist

Fork and clone the repo to your computer. Read through them until you understand the code.

Write your examples

There are 6 public instance methods for the `Account` class:

1. `initialize`
2. `transactions`
3. `balance`
4. `account_number`
5. `deposit!`
6. `withdraw!`

Before you actually implement any specs, go through each of these methods and answer the following questions:

- What are the valid inputs, if any, for this method?
- What constitutes expected return values?
- What constitutes unexpected return values?
- Does the method cause changes to the state of the program?
- What defines a happy path scenario? What about an unhappy path scenario?

Remember to ensure that your spec is actually testing what you want by ensuring that it fails when the corresponding code is commented out. If it passes even when the code it is supposedly testing is commented out, then it is a broken test.

For a quick reference of the RSpec API, check out the cheat sheet.