Chapter 1: Object-Oriented Design

Maintainability over the life of the code is more important than optimizing its present state.

Each object encapsulates some state and defines its own behavior.

OOD requires that you shift formthinking of the world as a collection of predefined procedures to modeling the world as a series of messages that pass between objects.- failures in OOD are failures of perspective

Interaction between objects is embodied in the messages - getting the right message to the correct target object requires that the sender of the message know things about the receiver. - creates dependencies between two objects

OOD is about managing dependencies such that objects can tolerate change.

Objects knowing too much is a nightmare because they are resistant to change

The purpose of design is to allow you to do design later and its primary goal is to reduce the cost of change

OOD Principles

Single Responsibility
Open-Closed
Liskov Substitution
Interface Segregation
Dependency Inversion

DRY

LoD - Law of Demeter

OOD has patterns as well - patterns are a simple and elegant solution to specific problems in object oriented software design that you use to make your own designs more flexible, modular, reusable and understandable

Design is the process of progressive discovery that relies on a feedback loop - OO softwre fails when the act of design is separated from the act of programming

OOD is about arranging what code you have so that it will be easy to change

look up ruby metrics
Your Goal when writing software should be to write with the lowest cost per feature.

How much design to do depends on your skill and timeframe.

Objects have behavior(operations) and may contain data, data to which they alone control access(encapsulated).

Objects invoke one anothers behavior by sending each other messages

Its common to desire to manufacture a bunch of objects that have identical behavior that encapsulate different data

Class based OO languages like Ruby allow you to define a class that provides a blue print for the construction of similar objects.

- A class defines methods (definitions of behavior)
- And attributes (definitions of variables)
- Methods get invoked in response to messages

Instantiation is the creation of new instances of an object

Knowing an objects type lets you have expectations about how it will behave.

- In ruby an object can have many types one of which will always come from its class
- Knowledge of its type(s) lets you have expectations about the messages to which it responds to.

OO languages let you invent brand new types of your own tailored to your domain.