Chapter 11 - Systems

Separe Constructing a System from using It

Software Systems should separate the startuo process, when the application objects are constructed and the dependencies are "wired" thogether, from the runtime logic that takes over after startup

Separation from main

One way to separate construction from use is simply to move all aspects of construction to `main`, or modules called by `main`, and to design the rest of the system assuming that all objects have been created constructed and wired up appropriately.
The Abstract Factory Pattern is an option for this kind of approach.

Dependency Injection

A powerful mechanism for separating construction from use is Dependency Injection (DI), the application of Inversion of control (IoC) to dependency management. Inversion of control moves secondary responsibilities from an object to other objects that are dedicated to the purpose, thereby supporting the Single Responsibility Principle. In context of dependency management, an object should not take responsibility for instantiating dependencies itself. Instead, it, should pass this responsibility to another "authoritative" mechanism, thereby inverting the control. Because setup is a global concern, this authoritative mechanism will usually be either the "main" routine or a special-purpose *container*.