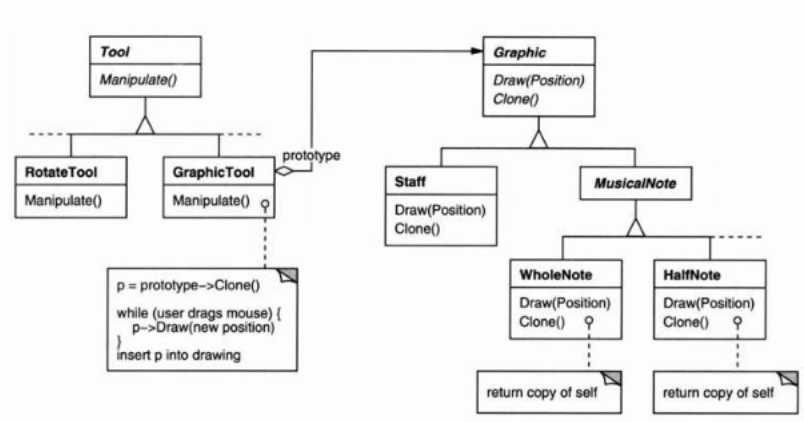
Prototype

Prototype

- Intent
  - Specify the kinds of objects to create using a prototypical instance, and create new objects by copying this prototype.
- Motivation
  - build an editor for music scores by customizing a general framework for graphical editors and adding new objects that represent notes, rests, and staves.

Prototype



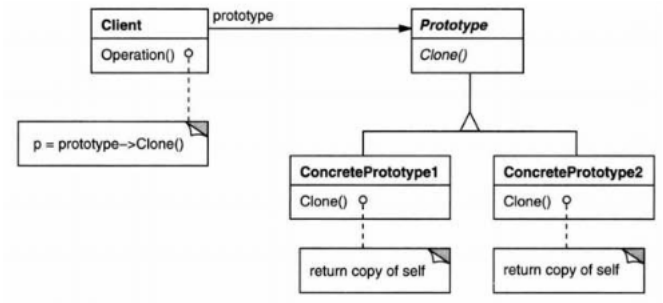
Prototype

- We have separate classes for whole notes and half notes, but that's probably unnecessary.
- Instead they could be instances of the same class initialized with different bitmaps and durations.
- A tool for creating whole notes becomes just a GraphicTool whose prototype is a MusicalNote initialized to be a whole note.
- This can reduce the number of classes in the system dramatically.
- It also makes it easier to add a new kind of note to the music editor.

Applicability

- Use the Prototype pattern when a system should be independent of how its products are created, composed, and represented; and
  - when the classes to instantiate are specified at run-time, for example, by dynamic loading; or
  - to avoid building a class hierarchy of factories that parallels the class hierarchy of products; or
  - when instances of a class can have one of only a few different combinations of state.

Structure



Benefits

- Adding and removing products at run-time.
- Specifying new objects by varying values – not by defining new classes.
  - In fact, cloning a prototype is similar to instantiating a class.

Benefits

- Reduced subclassing.
  - Factory Method often produces a hierarchy of Creator classes parallel to the product class hierarchy.
  - In Prototype, you don't need a Creator class hierarchy at all.
- Configuring an application with classes dynamically.
  - An application that wants to create instances of a dynamically loaded class won't be able to reference its constructor statically.
  - Instead, the run-time environment creates an instance of each class automatically when it's loaded, and it registers the instance with a prototype manager.

Implementation

- Using a prototype manager.
  - When the number of prototypes in a system isn't fixed, keep a registry of available prototypes.
- Implementing the Clone operation.
  - The hardest part of the Prototype pattern is implementing the Clone operation correctly.
  - Shallow or deep?
  - It's particularly tricky when object structures contain circular references.