# Code patterns

1. Template method pattern

* **Code snippet:**   
  **GPCalendarBase:**  
  **WeekendCalenderImpl:**

**AlwaysWorkingTimeCalendarImpl:**



* **Location:**   
  biz.ganttproject.core/src/main/java/biz/ganttproject/core/calendar/GPCalendarBase.java

biz.ganttproject.core/src/main/java/biz/ganttproject/core/calendar/WeekendCalendarImpl.java  
  
biz.ganttproject.core/src/main/java/biz/ganttproject/core/calendar/AlwaysWorkingTimeCalendarImpl.java

* **Reasoning:** This is an example of the template method pattern. We have 2 classes (AlwaysWorkingTimeCalendarImpl, WeekendCalenderImpl) that extend from another one GPCalendarBase. The two classes that extend from the base class are enforced to apply the behavior set by the base.

1. Builder pattern

* **Code snippet:**   
  ProjectDatabase.kt  
    
  TaskImpl.kt:



* **Location:**   
  ganttproject/src/main/java/net/sourceforge/ganttproject/storage/ProjectDatabase.kt  
  ganttproject/src/main/java/net/sourceforge/ganttproject/task/TaskImpl.kt
* **Reasoning:** In these code snippets we can see an example of the builder pattern, by having multiple functions that allow the object to be built. Highlighted in yellow are the calls to the functions that build the object, the parts are assembled when needed.

1. Singleton

* **Code snippet:**   
  
* **Location:**   
  ganttproject/src/main/java/net/sourceforge/ganttproject/language/GanttLanguage.java
* **Reasoning:** This is an example of a singleton. The object can only be instantiated once, since no one has access to its constructor (private). It can only exist one instance of this object and a global point of access to this instance is provided by the class.