**Design Pattern:**

We discover that SuperTux has different types of design patterns: Template Design Pattern, Facade Design Pattern.

**Template Pattern:**

SuperTux’s development team uses Template design pattern because they implement the skeleton of sprite and sprite\_manager as template to define animations for other objects’, such as brick.cpp, bullet.cpp, and bouncy\_coin.cpp. The template method lets subclasses redefine sprite animation for each individual objects without modifying sprite’s algorithm.

**Facade Pattern:**

One example to show SuperTux’s design pattern is the main.cpp. There are three void functions called init\_tinygettext, init\_video, launch\_game and one int function run inside main.cpp. These functions are used for initiate and run the game; they launch levels ,spawn Tux, and render videos when necessary; which means these functions are “Facade”s of Facade pattern participants. To begin the game, main.cpp needs to call the “Subsystem Classes” of Face pattern participant, such as input\_manager from control folder, sound\_manager from audio folder, gameconfig from supertux folder and more. main.cpp minimizes the communication and dependencies between each subsystem and main.cpp itself is a simplified interface to the more general facilities of a subsystem that unified all the other subsystems required to run the game; which makes the subsystems easier to use.