

Version History for the Rabbit Game

The book is organized around one evolving example, a video game. The game sits atop two layers of software (in C and C++) that isolate it from the vagaries of the underlying operating system. These layers have no version number, since they are not part of the game proper.

Book I: Objects

Foundations

C <code>term_</code> functions	85
C++ class <code>terminal</code>	157

Initial version of the game

1.0 initial version of the game	192
1.1 destructors for classes <code>wolf</code> and <code>rabbit</code>	200
1.2 make the animals impossible to copy	200
1.3 array of rabbits	235
1.4 static data member for classes <code>wolf</code> and <code>rabbit</code>	238
1.5 static member functions for class <code>terminal</code>	244
1.6 initialize the data members of classes <code>wolf</code> and <code>rabbit</code>	264
1.7 constant data member for classes <code>wolf</code> and <code>rabbit</code>	269
1.8 constant data members for class <code>terminal</code>	269

Dynamic Memory Allocation

2.0 list of pointers to dynamically allocated rabbits and a <code>game</code> object to hold it	466
2.1 read the <code>rabbit</code> constructor arguments from an array	470
2.2 pass the array to the constructor for class <code>game</code>	471

Book II: Inheritance, Exceptions, Templates

Inheritance and Exceptions

3.0 single inheritance: derive <code>wolf</code> and <code>rabbit</code> from <code>wabbit</code>	535
3.1 multiple inheritance: mix and match the ancestor classes	563
3.2 mass extinction	571
3.3 a friend of class <code>wabbit</code>	576

xii Version History

3.4	three functions that are neither members nor friends	577
3.5	allow the derived classes to loop through the master list	578
3.6	class <code>visionary</code> : step away from enemies and towards food	578
3.7	private inheritance	582
3.8	throw exceptions	628
3.9	catch the exceptions	629

Templates

4.0	create the four rank classes with one template	693
4.1	create the sixteen grandchild classes with one template	695
4.2	class <code>terminal</code> becomes a template class	740
4.3	let the terminal contain <code>printable_t</code> objects	753
4.4	a map instead of a searching loop	794
4.5	a map instead of a counting loop	795
4.6	a map instead of the big switch	797

Book III: Containers

Second Foundation

	class <code>terminal</code> becomes a container	965
5.0	port the game to the new class <code>terminal</code>	983
	C <code>term0_</code> functions for the ring terminal: a terminal with a different topology	994
	create C++ class <code>terminal0</code> for the ring terminal	999
5.1	port the game to the ring terminal	1000