EPILOGUE

Taking Your Next Steps

Sometimes, the best thing about the journey’s end is learning that there are more journeys to come, and this book is no exception. You’ve barely touched the surface of game programming in this book; Appendix A, “Suggested Reading,” will attest to that. There are literally hundreds of books available on game programming. Many, unfortunately, aren’t very good. Even worse, though, the really good books aren’t designed for beginners. Most assume a strong understanding of basic graphics programming techniques and mathematics. Although we’ve tried to skim over such details as best we could, you should take steps to understand as much as possible about the basics. Fortunately, it’s fairly easy to learn such things, even without shelling out lots of money for other books, thanks to the Internet. You could argue that you can find all the material in this book on the Internet, but we like to think our material is quite unique. Besides, where else can you learn about 3-D graphics along with cool trivia like twinkling star fields?

Moving On

In terms of graphics programming knowledge, you should work hard every day to learn these concepts:

- **Linear algebra:** This covers things like vectors, matrix math, etc.

- **Trigonometry:** Because there’s more to trig than SOHCAHTOA.

- **Computer graphics:** Because this field changes every day.

- **The history of computer graphics:** You might be puzzled over this one, but it’s extremely important. You should recognize some of the historic names in computer graphics, like Sutherland, Blinn, Gouraud, Mandlebrot, and others. You should also know what their contributions were (or still are) to the field and how it applies to what you learn.

- **Physics:** Because gravity is your friend.
• *Artificial intelligence:* Unless you like shooting computer opponents that don't shoot back and don't hunt you down, you're probably going to need to learn this.

• *And most importantly... learn what makes a game interesting:* Because great games don't need the latest and greatest graphics techniques; what they need is the undivided attention of the player. Play games and take notes about what you found interesting. Watch other people play games. What did they like or dislike? What gets them excited about a game? What kind of game do they play over and over again, and why? And so on.

**Habits to Build**

Becoming a good game programmer involves making many mistakes along the way. Becoming a great game programmer involves remembering and avoiding previous mistakes (so that you can make newer, more spectacular mistakes). We can't teach you how to be a great programmer, but we can give you some habits to learn that will help you be a good programmer.

**Habit #1: Source Control Is NOT an Option**

Nothing is more painful than making a bunch of changes to your code and discovering you did something wrong, but you no longer have the old version to look back on. Or, worse, your hard drive crashed and you have no backup.

Source control is a very simple habit to have (and one that you'll be forced to adopt if you join a gaming company). The really nice thing about it is that source control at a personal level is either free or very cheap, depending on what you want to do.

If you're using Visual Studio Professional or Enterprise editions, you can use the provided Visual SourceSafe tool for source control. This is a simple tool that allows you to follow a check-in, check-out style of development. This works nicely for a few people, but can get cumbersome for large teams.

You can also opt for the free CVS tool and its companion GUI, WinCVS (http://www.wincvs.org). There's a lot of knowledge about how to set up this tool on the Internet, and it can be relatively straightforward.

Other, more expensive tools are available, such as AlienBrain (http://www.alienbrain.com) and Perforce (http://www.perforce.com). For this book, we used Vault by SourceGear (see the following figure). We like it because