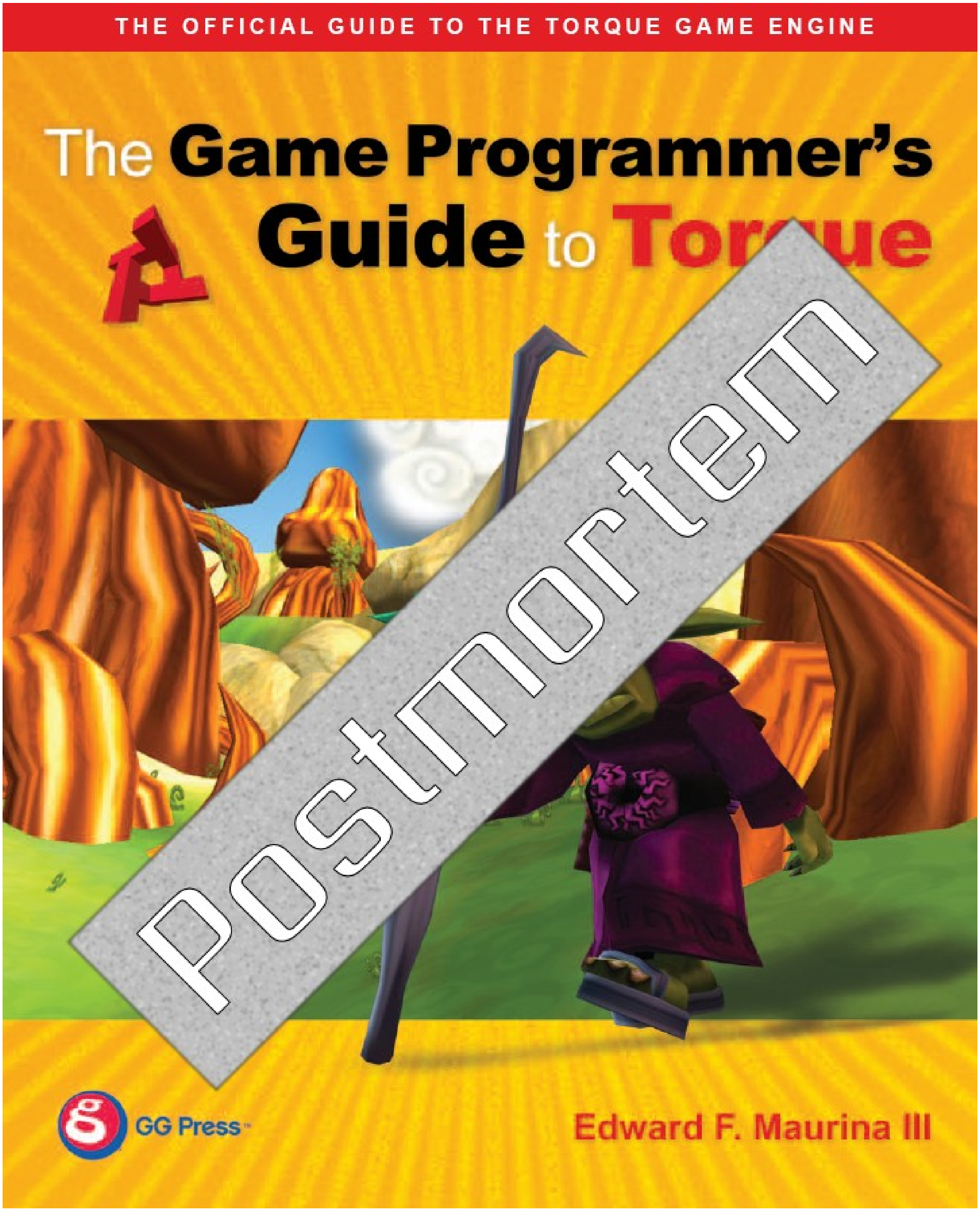

THE OFFICIAL GUIDE TO THE TORQUE GAME ENGINE
The Game Programmer's Guide to Torque
postmortem
GG Press
Edward F. Maurina III

Synopsis Part I

After much deliberation, I have decided to ignore the trend to write a postmortem using the standard 3-3-3 format (three things that went wrong, three things that went right, and three things learned). So, consider yourself warned, this postmortem is a little long.

I am writing this postmortem for the community and for myself. It is my hope that those who read it will be able to take away some useful bits of information.

This postmortem contains the following three sections:

1. **History Of GPGT** – In this first section of the postmortem I traverse the entire history of this project, highlighting the important events that occurred along the way. The purpose of this is to show how a whim and a hobby can turn into something more serious.

2. **Project Analysis** – In this second section of the postmortem I will do a detailed analysis of the project, providing a more formal look at the different parts of writing this book. I will discuss such details as schedule, budget, tools, etc.

3. **Lessons Learned** – In this third and final section I examine some specific things that I learned along the way.

History Of GPGT Part II

The history of “The Game Programmer's Guide to Torque” is both a history of the book and to a large extent, a history of my time (thus far) as an independent game developer.

Like Many Of You

Like many of you, I have played computer games most of my life, and also like many of you, I played around writing my own 'games' of sorts. Up until about 1996, the idea of making games was nothing more than a dream and I pursued it like a hobby.

Pre-Indie Indies

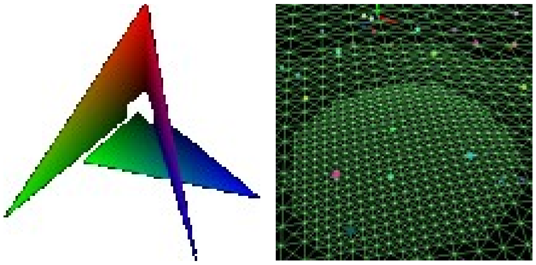
In 1996, I had just finished my degree in Computer Engineering and had also gotten a great job at Intel Corp.

Jerry Shaw, a prior classmate of mine, was also hired by Intel. We ended up working together and formed a friendship. Jerry would end up being my cohort in crime for many years to come.

Amazingly, we were both of like mind when it came to making games. So, we decided to team up in our spare time and to work seriously on a 3D game engine.

Once we 'knocked that out', we would move on to making games.

Using our combined programming skills, the results of much research (thanks NeHe, GameDev, GamaSutra, Romaka, Sulaco, OpenGL Org, Flipcode, et cetera!), and a little ingenuity, we quickly cobbled together our first attempt at a 3D game engine.



Wedgie Madness!

Our first attempt was fine as a tech demo, but it wasn't a real game engine.

We accepted this and decided to try again.

Options: “1. Make a Game Engine 2. Make a Game” – Choose One.

Having learned much from our first attempt at developing a 3D game engine, we started over again on “Nascent”. This, we decided, would be our first 'serious' attempt at a complete 3D game engine.

Work on Nascent went on for about a year. We met every 2 or 3 weeks (when we could spare the time), compared our latest advancements, and assigned each other new tasks to work on. The project went well and feature after feature was added to the engine.

At the end of this effort, we had an engine with the following features:

- A fully hierarchical set of game classes.
- An embedded TCL scripting engine with full access to the C++ core components.
- A ROAM based terrain engine, utilizing multi-texturing and pseudo bump-mapping.
- Rudimentary game recording and playback.
- Support for scripted shape rendering (i.e. defined in script), billboard rendering, and full Quake 2 style shape rendering and animation.
- A multi-pass collision detection and response system.
- A limited Physics Engine.
- Hierarchical view-culling.
- Dynamic render scheduling.
- Multiple camera support.
- Etc.

However, we realized that there was still a long ways to go before we could start making a game. Nascent still needed a ton of features. Inevitably, this led us to a very significant conclusion:

“*You can write a game engine, or you can make a game, but you can't do both (with a small team).*”

Decidedly dejected and plain tired of the entire effort, we both returned to our careers and lives, letting Nascent die a quiet death.

We did however continue to play games and quietly aspire to write our own.

GarageGames Breaks The Mold

In 2001, still working at Intel and still playing games (after work of course), my current favorite game was “Tribes 2”. After passing the 100 hour mark playing this addictive game, I learned that the engine used to make this this very cool game had been released under an extremely relaxed and inexpensive license. So, in August of 2001, I signed up and bought a license to the V12 engine.

At this time, the V12 engine (AKA Torque) was quite unique. Certainly, there were other engines available but none of them had V12's combination of history, features, and price. Also, none of them had the stellar and dedicated GarageGames crew behind them.

After getting my copy of the engine, I quickly jumped into the demo and examined the current docs.

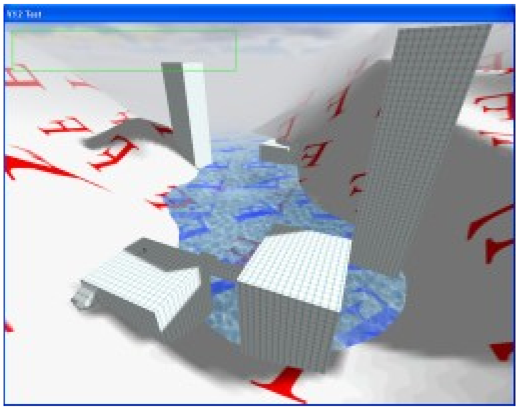
I was both excited and somewhat daunted. This thing was huge!

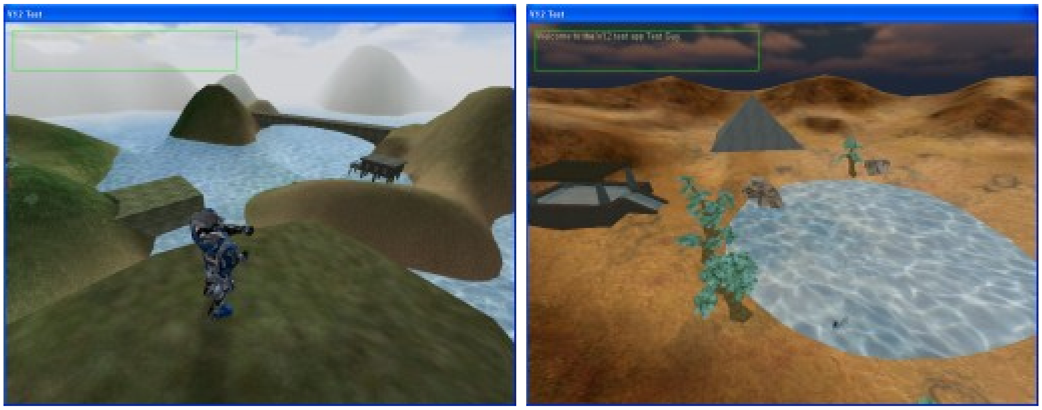
Now, I had worked on projects of this size before, but still, wrapping my head around the engine and grokking it was no mean feat.

I think perhaps that this summarized excerpt from the original 'Known Issues' disclaimer says it best:

- .... ***This is not for the timid***.
- ***GarageGames did not receive any documentation*** ... ***you will have to hone your code perusing skills.***
- ***In the process of removing the Tribes2 intellectual property*** ... ***functionality was impaired***

In addition to the exceptions noted in the original disclaimer, V12 did not come with all of the tools/exporters that we all take for granted now. For me, the real clincher was the lack of a Milkshape Exporter. Being a 'doit-it-on-the-cheap' guy, I didn't have 3DS Max and could not see myself spending that kind of money on what was still, only a hobby.





V12 In All Its Glory!

I quickly realized that I didn't have the time to work with V12 in its current state. So, I put it down for about a year.

Torque Notes Is Born

In early 2002, I came back to the GarageGames site to discover some great things.

Wow! These guys had really made some progress:

- Torque was up to release 1_1_2,
- The site had forums, resources, member home pages, a cool newsletter, **and**
- There was **DOCUMENTATION**.

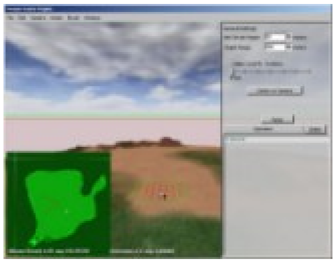
"This is more like it", I thought to myself.

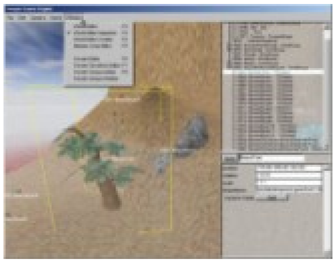
Jumping in head first, I read through the docs, downloaded release 1_1_2, and set to work on a game.

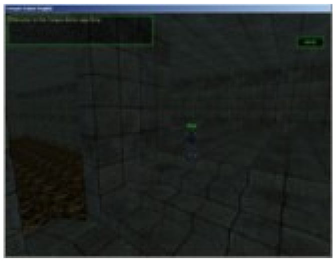
I started work on an MMO* and learned another valuable lesson.

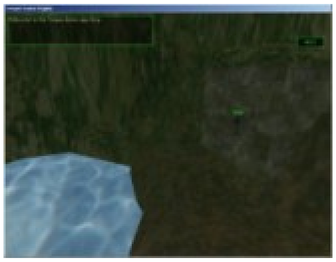
Start small.

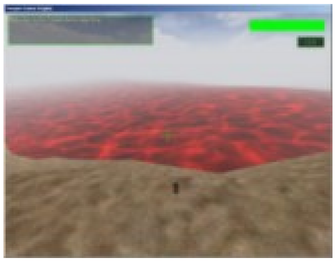
My game idea was way too big for one guy. However, it did force me to learn a lot about Torque.

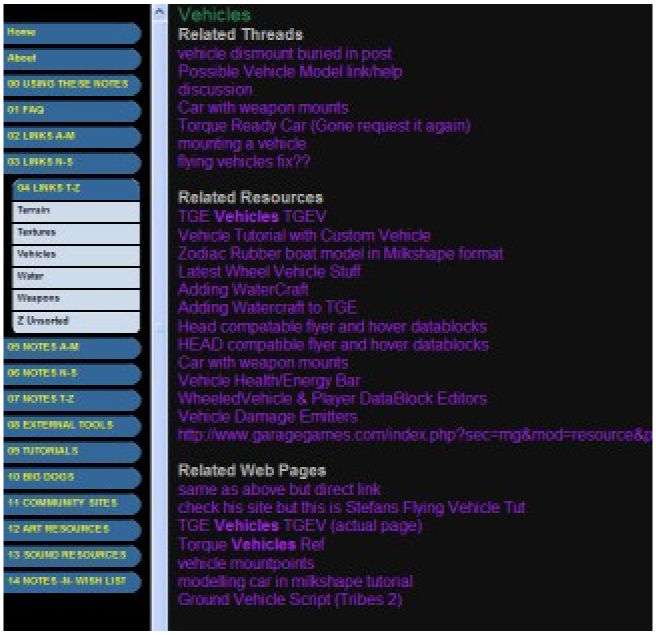












Vehicles

Related Threads

vehicle dismount buried in post
Possible Vehicle Model link/help
discussion
Car with weapon mounts
Torque Ready Car (Gone request it again)
mounting a vehicle
flying vehicles fix??

Related Resources

TGE **Vehicles** TGEV
Vehicle Tutorial with Custom Vehicle
Zodiac Rubber boat model in **Milk**shape format
Latest Wheel Vehicle Stuff
Adding WaterCraft
Adding Watercraft to TGE
Head compatable flyer and hover datablocks
HEAD compatible flyer and hover datablocks
Car with weapon mounts
Vehicle Health/Energy Bar
WheeledVehicle & Player DataBlock Editors
Vehicle Damage Emitters
http://www.garagegames.com/index.php?sec=mg&mod=resource&p

Related Web Pages

same as above but direct link
check his site but this is Stefans Flying Vehicle Tut
TGE **Vehicles** TGEV (actual page)
Torque **Vehicles** Ref
vehicle mountpoints
modelling car in milkshape tutorial
Ground Vehicle Script (Tribes 2)

While I worked on my game idea, I quickly exhausted the easily located documentation and started to read thread after thread, digging deeper and deeper to find information.

Eventually, I had compiled a pretty big collection of notes and Torque related links. So, I decided to share this information with the rest of the community in the form of a monolithic web page.

Thus, 'Torque Notes' was born.

http://www.hallofworlds.com/pages/Torque/TorqueNotes

What About A Book?

Torque Notes was an instant success, getting 1000s of unique hits every month and quickly delivering folks to topics they were seeking.

However, the page began to sprawl and to become very dated. In fact, it soon became completely unmanageable. So, I made a hard choice and stopped maintaining it.

Interestingly, by this time, a new thought was starting to tickle my brain,

"Hmmm, there might be a book in this somewhere."

After thinking about this for some time, I decided that some kind of short reference would be a great idea. I figured,

"A 100 page quick reference ought to be real popular."

“Essential Guide to the Torque Game Engine” Announced

In September 2002 I officially announced that I was writing a reference guide for the engine and gave it the working title: “Essential Guide to the Torque Game Engine” (EGTGE).

Nobody took any notice, but I was inspired and that was enough.

EGTGE Hits 350 Pages

March 2003 came and went. EGTGE hit and passed the 350 page mark.

At this time, I was still thinking that I would be done soon.

The work continued. In fact, the rest of 2002 and a good part of 2003 came and went, and the work still continued.

"Hmmm. This is harder than I thought it would be."

Ken (Finney) Releases His Book

In May of 2004, Ken Finney's book “3D Game Programming All In One” was released. In my mind, this effectively quashed my plans for EGTGE.

Severely disappointed, I released the lite-edition of EGTGE as a community resource and called it quits.

I took a long delayed vacation, thinking that it was time to move onto a new project. But, I couldn't have been more wrong. It seems, I had not accounted for the fact that Torque Notes, and the promise of EGTGE had started to draw a following.

The community response was rapid and overwhelming. People took turns, either lambasting me or encouraging me, but the gist of the message was the same from everyone:

"*Keep working on your book!*".

A few days later, GarageGames contacted me and offered to publish my book if I would just finish it.

"You're Not Going To Get Rich"

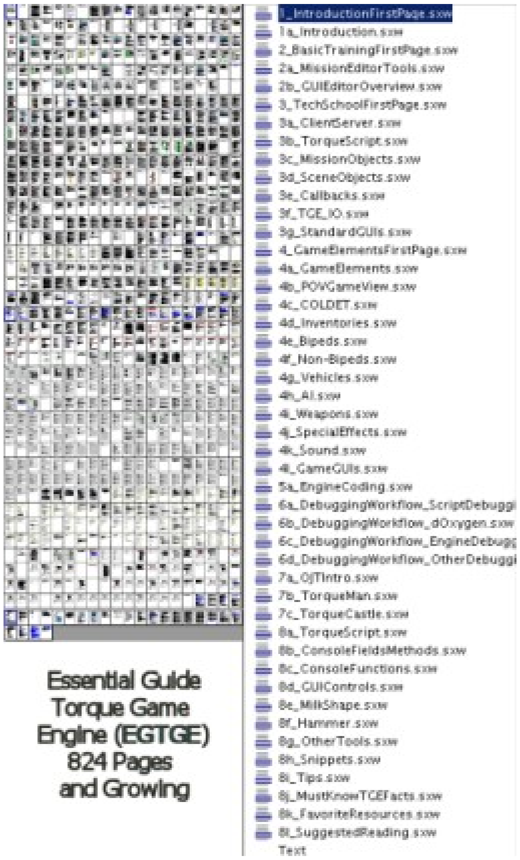
September 2004 came rolling along. The book was at the 824 page mark. Somewhere along the way, Josh and I had decided that the book would need to be printed in two volumes.

Great! Now, I had two books to finish. This was getting out of hand.

It felt like it should be done, but I didn't feel like everything was in place yet.

Soon, IGC'04 arrived and it was time to head to Eugene. During the conference, Jeff Tunnell and I huddled up for a talk about EGTGE. I no longer remember everything that was said, but one thing did stick. Jeff told me (paraphrased):

"*Ed, you're not going to get rich making this book. So, just finish it up and get it out*".



```
1_IntroductionFirstPage.sxw
1a_Introduction.sxw
2_BasicTrainingFirstPage.sxw
2a_MissionEditorTools.sxw
2b_GUIEditorOverview.sxw
3_TechSchoolFirstPage.sxw
3a_ClientServer.sxw
3b_TorqueScript.sxw
3c_MissionObjects.sxw
3d_SceneObjects.sxw
3e_Callbacks.sxw
3f_TGE_IO.sxw
3g_StandardGUIs.sxw
4_GameElementsFirstPage.sxw
4a_GameElements.sxw
4b_POVGameView.sxw
4c_COLDET.sxw
4d_Inventories.sxw
4e_Bipeds.sxw
4f_Non-Bipeds.sxw
4g_Vehicles.sxw
4h_AI.sxw
4i_Weapons.sxw
4j_SpecialEffects.sxw
4k_Sound.sxw
4l_GameGUIs.sxw
5a_EngineCoding.sxw
6a_DebuggingWorkflow_ScriptDebuggi
6b_DebuggingWorkflow_dOxygen.sxw
6c_DebuggingWorkflow_EngineDebugg
6d_DebuggingWorkflow_OtherDebuggi
7a_OJTIntro.sxw
7b_TorqueMan.sxw
7c_TorqueCastle.sxw
8a_TorqueScript.sxw
8b_ConsoleFieldsMethods.sxw
8c_ConsoleFunctions.sxw
8d_GUIControls.sxw
8e_MilkShape.sxw
8f_Hammer.sxw
8g_OtherTools.sxw
8h_Snippets.sxw
8i_Tips.sxw
8j_MustKnowTGEFacts.sxw
8k_FavoriteResources.sxw
8l_SuggestedReading.sxw
Text
```

This was excellent advise. I just didn't I know how to follow it. At the time, I was seriously wondering how I would ever finish the book, that is, the books.

Draft Version Completed

In May 2005, I finished the draft version of the GPGT volume 1. I also had much of GPGT volume 2 ready to go. In total, I had accumulated:

- Approximately 1400 pages of written material.
- A finished single-player game prototype.
- A kit containing 24 GUI samples, 8 functionally complete interface samples, 3 example HUDs, 23 3D lessons, ...

I turned this in to GarageGames and they sent it to AK Peters.

GPGT Jam Session #1

By mid-September 2005, AK Peters had reviewed the draft and was asking for some changes. So, I hopped in my car and drove to Eugene to participate in a copy-editing and writing marathon (otherwise know as *GPGT jam session #1*).

Over Saturday September 10th and part of the 11th, Josh Williams, Ben Garney, Matt Fairfax, and I read and reviewed the entire guide. We also discussed the various changes that AK Peters had requested.

Basically the way this worked was that everyone got a chapter and read it. Then, they sat down with me (and sometimes with Josh) to review any issues and to have me answer any questions about questionable sections.

When we got done with the session I had the following list of things to do:

- Incorporate fixes for all 14 chapters.
- Replace the original 'Torque Man' (a 3D Pac-Man clone) with 'Maze Runner', thereby avoiding any possibility of legal issues with Namco Limited.
- Remove the game prototype creation chapter and instead append 'lessons' to the end of chapters, thereby building the game as the reader progresses through the guide.
- Put together a description to go on the back cover.
- Incorporate professional art for a finished version of the game (AKA Maze Runner Advanced).

All of this work needed to be done in about a week. So, I drove home and got to work immediately.

Josh did me a huge favor and wrote the back-cover text. Then, he hooked me up with Christophe Canon of *FroGames* to put the pro-art into Maze Runner Advanced.

Note: *Christophe was a real pleasure to work with.*

http://www.frogames.net

GPGT Jam Session #2

Exactly three months later, in mid-December 2005, AK Peters had re-examined the book. As a result of their re-review, they supplied a new list of things that needed fixing, and they sent a hard-copy with hand written notations back to GarageGames for the GarageGames staff and I to update. It was time for another jam session.

Over Saturday December 10th and most of the 11th, Josh Williams, Justin DuJardin, Ben Garney, Matt Langley, Jay Moore, Julie Moore, and I did another complete review.

It is worth noting that this sessions was a little different than our prior one. We had no schedule leeway for completing the fixes. They all needed to be in by the following Monday to stay on schedule.

It was painful, but we were successful!

Please Make It Stop!

After the second Jam session, I thought AK Peters would be taking over and I would get some time to breath.

Wrong!

In the period between December 27th 2005 and February 3rd 2006, AK Peters did the final review and layout work. Josh and I ended up being heavily involved with this process.

The way this worked was as follows:

1. Josh would receive the 'current chapter' from AK Peters and forward it to me. This chapter came in layout form. That is, it now closely approximated the final look. In addition to the chapter, we also received a list of 20 to 40 questions and comments from the copy-editors and layout staff.

2. Josh and I then (independently) re-read the entire chapter, looking for any errors or technical issues.

3. Next, I would handle all of the questions/requests that came with the chapter. These were usually such things as:

- Spelling Questions

- Missing Table/Image Captions
- Path Questions
- Illustration Revision Request

4. Once I was done writing all of my own corrections down, providing responses to the questions and comments, and re-doing any artwork, I shipped all this to Josh.

5. Lastly, Josh would packages everything up, adding his own inputs, and ship it off to AK Peters.

This went on for the next 39 days. During this time, I re-read every chapter at least once, I updated or re-createe 373 images, I wrote between 18,000 and 20,000 words of feedback in the form of 898 unique fixes and responses to questions, and I created an Index through a combination of concordance files and a whole lot of hand-editing.

In short, this experience was enlightening, but exhausting.

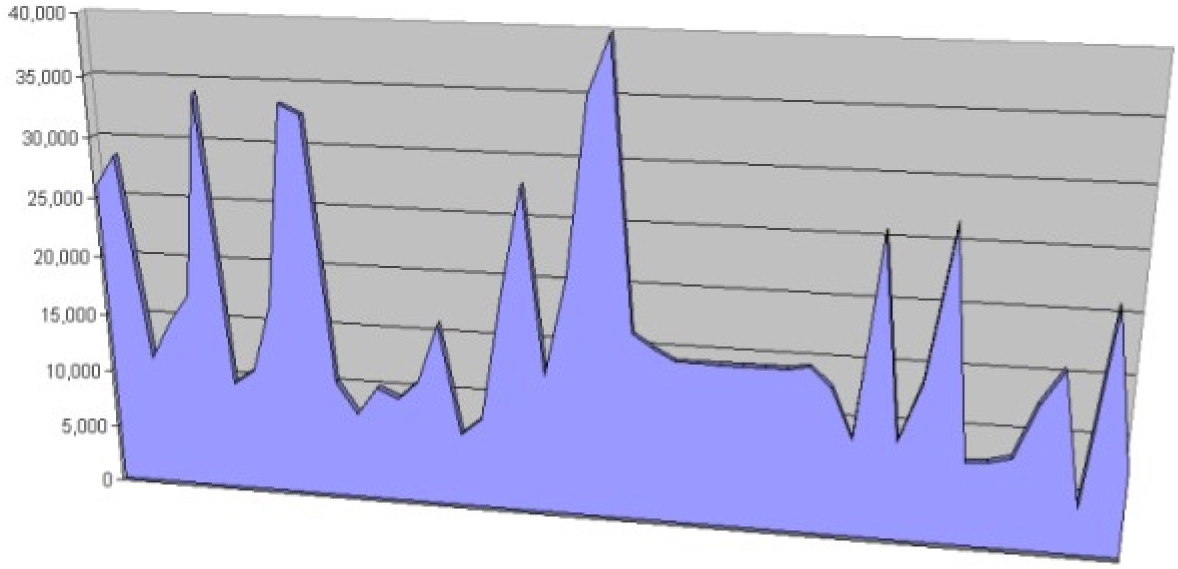
First Printing

In March 2006, AK Peters ran the first printing and took 90 copies with them to GDC '06.

They sold out.

Amazon, Barnes & Noble, GarageGames, and AK Peters all started taking and filling orders.

Over the next three some odd months, the Amazon Ranking would look like this:



Summary Of Effort

Over the past four years, I have spent about 2500 hours working on this book. In retrospect, I am somewhat surprised to find that writing was only about half of the work. Besides writing, I did all of the following tasks:

- Learned the engine thoroughly,
- Found and fixed engine bugs,
- Wrote tens-of-thousands of lines of script and C++ code
- Created well over 1000 unique art assets (mostly illustrations, but also textures for the GPGT lesson kit and the game prototype),
- Designed and wrote two complete single-player games, and
- Participated in review and proofing sessions till I was ready to cry.

At the end of the day, I believe that the effort was worthwhile. In addition to working with lots of wonderful people in the community, I also got to know the staff of GarageGames much better. GarageGames, the community, and all three of the Torque engines rock!

Mission Accomplished?

The bold among you may ask,

"So, was Jeff right, or did you get rich?"

Yes, Jeff was right, and No, I did not get rich. However, I didn't start this project with that in mind.

My original motivations for writing this book were:

1. Help grow the Torque community,
2. Help others so they won't have to go through the growing pains I had to go through,
3. Get my own company started. i.e. Build some recognition for myself and hopefully for the Hall Of Worlds Logo.
4. Hopefully make some money,

I accomplished #4, but did I accomplish the other goals?

You tell me.

Project Analysis Part III

Project Data

Project Name: The Game Programmer's Guide to Torque (GPGT)

Design Overview: Originally, GPGT, known first as the "Essential Guide to the Torque Game Engine" (EGTGE), was intended to be a short (~100 page) quick reference discussing some tools, scripting, and a few tips and tricks.

I intended to have two versions:

- EGTGE Lite (a free version used as a 'loss leader')
- EGTGE Full (a for pay version)

> "*This will be a multi-month project, culminating in a guide containing documentation on many significant Torque topics with full references and many accompanying examples. The guide will be fully illustrated and come with an accompanying resource disk. Currently, I have not decided if this will be printed or an e-book.*"
>
> - Original EGTGE Description From Torque Notes Page

Intended Customer: New Game Developers, Veteran Torque Users, Teachers and Students.

Product's Purpose: Teach folks how to use Torque to make games and other products.

Project Start / Finish Dates (Estimated): I officially started writing in November of 2002. I estimated I would be done between June and August of 2003.

Project Start / Finish Dates (Actual): I did in fact start November 2002, but the book did not actually go to press until March of 2006.

Budget (Estimated): None.

Budget (Actual) : By the end of the project I ended up making several purchases related to the book, coming to a grand total of about $2000 dollars.

Release Date: The official (bookstore) release was May 2006.

Release Format: Soft cover (600 pp) with accompanying CD containing 580 (pp) reference, samples, and a lesson kit. The kit was designed to run on all Windows platforms, OSX, and in a degraded mode on Linux systems.

Lines of Code: I ended up writing/generating about 42,000 lines of new script.

Art Assets: > 1000 unique pieces

Development Software

- Windows 2000
- Torque Game Engine
- Torque Show Tool
- Open Office versions 1.1 through 2.0 Beta.
- Visual Studio 7
- Paint Shop Pro 7, 8, 9
- Various Flaming Pear Filters
- Milkshape
- GameSpace
- Unwrap 3D
- Audacity
- Fraps
- Hammer

Development Hardware:

- 1.8 GHz Pentium 4 with 1GB memory and a late model nVidia video card.
- Laser Printer

Other Development Materials:

- Book: "*Books, Typography, and Microsoft Word: How to Get High-Quality Type for Desktop Book Publishing, Self Publishing, and Print on Demand, or Tips on Type*"
- Book: "*Print-On-Demand Book Publishing: A New Approach to Printing and Marketing Books for Publishers and Authors*"
- Book: "*The Self-Publishing Manual*"

Known Bugs:

About 50 errata have been discovered to date. More details can be found here:

http://gamers.hallofworlds.com/support/gpgt/

, and here:

http://www.garagegames.com/mg/forums/result.forum.php?qf=178

Known Compatibility Issues:

There were recently some issues on OSX systems. These have been resolved.

Team Members and Responsibilities:

- Edward F. Maurina III – Writer
- Teresa Tse – Business Manager

Project Ratings 1 (poor) to 10 (outstanding)

- **How well did the project achieve its goals? 8**
  - I wanted to cover more in volume 1 than was possible due to page constraints.

- **Was the project developed in a reasonable amount of time? 3**
  - No. I was way over estimate.

- **Was this a cost-effective project? 7**
  - I'm still in the negative, but will recover my expenses eventually when my royalty check comes in.
  - To be completely realistic, if this project were only about making money, then the rating would be a **1**. Why? Well, if I considered my time worth only $50/hr (which is low), then 2500 hours x $50 comes to $125,000. It is highly unlikely that I will earn that kind of money from this book. Fortunately, money wasn't my primary motivation.

- **Did development proceed smoothly? 6**
  - The project had many jumps and starts, as well as a few complete stops. It was an enlightening but rough ride.

- **How well did the project meet its estimated budget? 5**
  - Not at all, but then the original budget was non-existent which was unreasonable and unrealistic.

- **How well was change managed during this project? 6**
  - I ended up doing major re-writes of sections of the guide for the following reasons:
    - Torque 1.2 to Torque 1.3 transition
    - Torque 1.3 to Torque 1.4 transition
    - False start on sample game: 'Torque Man'
    - Writing while learning

What went right?

1. **I learned a lot about the engine and making games.** Remember, there is always more you can know.

2. **I learned a lot about writing and publishing a book.** This is big understatement, but this will become clear a bit later when I discuss some writing/publishing specific topics.

3. **I hit the target (audience) sweet spot.** I feel that I hit the right level of usability to enable both new and experienced users to benefit from this book.

4. **Well received.** The book has been well received and looks as if it may be adopted by some universities and colleges as part of their game development programs.

5. **Built reputation.** The book and associated efforts have helped build my reputation, however humble it may be, in the gaming community.

   Note: I joke, but this isn't about humility or vanity, but rather reputation and name recognition. Both of these are valuable commodities if you intend to do business in a distributed environment like the Internet.

6. **Is improving Torque awareness.** The book has helped and is helping to build further awareness of Torque and the GarageGames community. This is a good thing. For members of the community to have the highest likelihood of success in their endeavors, it is important that they be part of a vibrant community with a constant infusion of new members. The community doesn't necessarily need to grow, but it must not stagnate. The GarageGames community has definitely avoided stagnation.

7. **Is helping new users.** The book has helped and is helping new users get into using the Torque Game Engine more quickly than would otherwise be possible. Struggling for struggling's sake is no fun.

1. **The project failed to meet its estimated time line.** This was due to my lack of experience as a writer and because I didn't (initially) treat the project as a job. I didn't create a schedule or a plan until I was well into the project.

2. **The project was way over budget.** Again, I had no idea when I got into this, just what would be involved in writing a book, and I didn't plan for any expenses.

3. **Art in wrong format for printing.** All the art had to be massaged into grayscale. Also, by the time I had to do this, several original pieces were lost and and to be re-produced.

4. **The first 'sample' game had to be canceled due to poor legal planning (on my part).** Originally, I created a Pac-Man like game in 3D. We later decided that although this was probably safe, it would be better to not take chances. So, we changed the game to 'Maze Runner', which is a pseudo-generic 3D platformer.

5. **I assumed that AK Peters would 'magically' produce the index for me.** Instead I ended up having to write it at the last moment. Therefore, it was written as an, "I thought you were going to do it? Oh crap!”, kind of effort. I think this shows in the final version.

6. **Failure to back up data.** I didn't back up my data properly in the early part of the project and suffered a complete hard-drive failure in early 2004. I ended up having to use a data recovery service to get the book back. It cost me about $300 to retrieve less that 50MB of data.

7. **Insidious and silent content corruption.** Early on, Open Office was a little buggy. I discovered this during a review when I found data and images both missing and corrupted. Although I stayed with Open Office, I took steps to reduce the likelihood of a repeat of this issue in the future. Among these steps were: upgraded to latest Open Office version, backed up before, after, and hourly while I wrote, checked my documents at the end of the day for any corruption.

- **Note:** After version 1.1 I never experienced this issue again.

8. **Did not use a spell checker until after the technical review.** Because there are so many non-English words (i.e. Datablock names, etc.) in the guide, I turned off the spell checker at an early date and never turned it back on. So, although I spell quite well, we did end up having to find a few misspelled words during copy-edit sessions. This was a big waste of valuable time.

9. **Technical Reviews were hard work and somewhat unexpected.** I learned that you should not count on the publisher to do the work for you. When it comes to writing Technical books, you have to do all the technical editing yourself. In other words, don't expect copy-editors at a publishing house to be capable of catching technical issues with your book. This is your job. This sounds obvious, but when we started the process of reviewing I was still quite naïve about which party would be responsible for what part of the review.

10. **Too many chapters to review in too short a time for the final technical review.** Although we did get through it and there have been relatively few errata, I could have done better. The primary problem is the fact that I ended up doing a final technical review of an entire chapter along with all fixes and re-submissions of art (sometimes) in a single evening.

11. **Last minute changes to directory structure of kit caused some mismatches between book and disk.** At the very last moment I discovered that in some cases, a zipped file will not extract properly if the path is too long. So, I quickly renamed a few files and directories, completely failing to recognize that this caused a mismatch with some of the 'Maze Runner' lessons.

Lessons Learned Part IV

This final section will be a free-form list of things that I learned while writing the guide.

Limit Your Scope

If you are writing a book or other document. Limit your scope from the start and be sure to keep to that limit. I started GPGT as an open-ended project and got bitten by major feature creep.

You're Not Going To Get Rich

Boy. Jeff was so right. No single book is going to make you rich. The trick is to create a steady flow of products, each of which contributes to your bottom line incrementally. If you can only afford the time to work on one project, and if you need to make your living from that project, then I don't suggest writing a book. You'll probably starve long before you finish it. However, if you can afford the investment of writing a book, then by all means do so. Why? Because, books have the following nice properties:

- **Relatively long life.** If you can refresh a book and publish new revisions, you can benefit from your initial investment of time over the long term.
- **Can Be Turned Into Franchise.** No, I don't mean a literal franchise. What I mean is, often written material can be adopted to other purposes. Thus, you may be able to re-use your written material to create other products.
- **Name Recognition.** Most people (not all) still tend to recognize that writing a book is no simple task. So, finishing and publishing one cannot hurt your reputation (as long as you do a good job of it), and a good reputation is one of the pillars of success.

Collaborate Using a Private Website

Early on in the project I created a private site where I could post the latest versions of chapters and supporting materials. This allowed any reviewers with access to download the materials when they wanted to. It also served as a pseudo-backup mechanism. I ended up having more than one site by the end of the project, each one dedicated to a different set of reviewers/consumers of the draft GPGT materials.

Create Your Own Base 'Kit'

If you are writing about a programming topic or another topic that requires a starting point, then provide that starting point. In my case, I ended up relying on the 'FPS' Starter Kit as a starting point and this got me into real trouble at the end of the project. This happened because the kit itself changed from version 1.2 to 1.3 to 1.4, **AND** because there are slight variances in the OSX and Windows versions of the Starter Kit (as supplied in the demo).

Hindsight tells me that I should have created my own starter kit and used it exclusively.

GPGT Lesson Kit

Although I didn't create my own base kit, I did have the GPGT lesson kit. I created this application so that I might create samples in these categories: GUI, Interface, Scripting, and 3D lessons. The kit itself was designed with a tool kit of scripts and predefined lesson file formats. This allowed me to easily create a new lesson and simply drop it into the kit, knowing that it would be properly loaded and displayed to the user. This had the added benefit of allowing me to provide new lessons after printing.

Know What You're Going To Write Before You Write It. #1

By this, I simply mean know your topic matter **BEFORE** you write. I started this project on a whim and while I was still new to the engine. Thus, I ended up re-writing sections later when I learned of new or different engine features which required a different approach to documenting them. In some ways, this actually helped this particular document, but the overall effect was a significant slow-down.

Know What You're Going To Write Before You Write It. #2

No, I'm not really repeating myself.

This time, I mean you should decide what you want to write about **BEFORE** you write it. This may sound counter-intuitive at first, but what I am really saying is, to plan your book out before you commit any words to paper. This is not an easy process and you should expect to change your mind about organization later, but if you start without a plan, you will regret it.

Stick To Your Plan.

I just said that you should expect change and that is still true, but be sure that you only accept small changes. If you make broad changes to your plan, you will likely waste a significant amount of time and effort.

Know Your Tools.

Be sure to learn how your writing tools work before getting too far into your project. For example, if you are using Open Office, learn how to use Master Documents and Templates. Also learn how the formatting system works. Although I don't encourage to you to pursue final formatting decisions in the beginning of your effort, I do suggest that you set up a custom set of formatting rules in the form of a template and then use that template for all of your documents. Later, when you want to reformat your documents, you can do so by modifying the template instead of manually modifying every chapter.

Also, if you intend to use Open Office to generate your Table Of Contents and Index, learn how these features and the concordance import features work **BEFORE** you start writing. You can certainly do this later, but learning how to set up for an index in advance will save you a ton of time later.

About The Table Of Contents (TOC)

If you are using Open Office, I do suggest that you use the built in indexes and tables feature to generate your TOC. This feature works perfectly well. One thing though. Be sure to generate them with the 'Protect against manual changes' feature disabled.

About The Index

Regardless of what tool you are using to write your book I have these suggestions:

1. Do not use an external tool to generate your index or a concordance file. The effort required to do this is equal to or greater than doing it manually.
2. Do not use a concordance file import mechanism. In all but the simplest cases, this will generate a over-sized index that is next to impossible to trim down.
3. Plan for your index as your write. By this, I mean mark keywords and sections for inclusion in your index as your write, or during your final copy-edit session. Either will work, but the prior is best. It is best because when you are writing about a topic you are often in the proper mindset for generating the tag to go with your topic. If you do it later, you will have to ask yourself, "When someone is looking for this information, how will they be thinking about it? What is the best way to label this so they can find it?"

I tried all of the above methods for creating my index and found the latter to be the most sensible. However, because I did create my index last, it suffered for my lack of imagination at the time, and for the extreme time crunch I was under.

Believe it or not, it took me over a week to create the current GPGT index.

Use Test Readers From The Start

I was fortunate enough to have an entire class of test readers (Michael Rogers CS481 class at Millikin University) reviewing my material while trying to use it. This was extremely useful. The bottom line is, there is no way, after reading and re-reading your own materials for the 15th time that you will catch the same errors as a new reader.

About Art and Assets

I got some good advice from Joe Maruschak and from others while writing my book and while working on the game prototypes that go with it. A summary of this advise would be:

- Use placeholder art until the very end. - In other words, do not bother creating superb art assets (or illustrations) until the underlying book/game is done. Why? Because, changes in the book/game may very well require changes to the art, thus wasting effort.
- Use grayscale for all assets until the asset is finalized. - This is similar to the placeholder art idea, but the differences is that by using grayscale textures/illustrations for all non-finalized assets you can clearly see what parts of your book/game are not done yet, or at least have not made it through final approval. Also, and Joe pointed this out, when you show a demo or prototype and the art is grayscale, folks will be more accepting of changes that come later. i.e. They don't form attachments to specific assets and they expect change.

Lastly, I have discovered that while draft writing, it is best to skip art entirely, and in those cases where a placeholder is absolutely needed, just sketch something by hand, scan it, and import it. Unless you're a whiz artist (which I'm not). Don't even touch your art program(s).