|  |  |
| --- | --- |
| Program: | GDP1 |
| Course: | For “Graphics 2”, “Game Algorithms *and* Gems”, and “Animation” courses |
| Professors: | Daniel Maclam |
| Project/Test: | 2020 GDP “Game Jam” |
| Weight: | “one project” and possibly “one exam” worth of weight (see details) |
| Due Date: | Wednesday, April 24th or Thursday, April 25th  in the evening   * This is the Contemporary Media “Industry Night”, which invites the same people. * More details, like how the venue will work, is pending. |
| Level Up Showcase | This is the University of Toronto/Ubisoft event Wednesday, April 17th |
|  | <https://levelupshowcase.com/> |
|  | This is in Toronto, and is completely optional |

*Note: This project can be done alone or in groups of up to two   
(but I would strongly suggest that you do it individually).*

Description and Purpose

Now is your time to shine. You have three weeks to create your amazing piece of art/technical demo/game.

Briefly, you will incorporate a number of “technical things” – you know, the things you see in marketing information about games.

Since there are three (3) courses and you need to perform two (2) topics each, you need to incorporate six (6) “technical things” in total. The marks are distributed evenly across all six, but have a different “weight”, depending on the difficulty (more details below).

Critical note: Since it is possible to interpret that the “jam” is a single assignment, worth > 40%, assigned during the last week of classes, you MUST indicate your choice of the following to me PRIOR at least two (2) weeks before the “jam”:

* That you want to do the game jam as a combination of the last project of all three courses (“Graphics 2”, “Game Algorithms and Gems”, and “Animation 2”), in place of the final exam. They would be marked as one item, and the corresponding mark applied to all three courses. In other words, you would be having the jam *instead of* a separate final exam and final project in these courses (aka: no final exam in those three), OR
* The “jam” is simply your final project in each of the courses, but y*ou still want to do a final exam in all three of these courses*. This would mean that you would complete the “jam” (it’s your final project) *and* would complete three final exams for each of “Graphics 2”, “Game Algorithms and Gems”, and “Animation 2”. These would be scheduled during exam week and worth ½ of your “exam” mark (so 20% of your final mark). The single “jam” mark would be applied to the project mark for each of these three courses.

**Another note:**

* You would still be required to complete six (6) technical topics for the jam
* No later than “week 12” (so two weeks before the last day of class):
  + You must communicate your choice (“Jam = project + final” OR “Jam = *only* final project, with separate finals”) no later than “week 12” (so two weeks before the last day of class).
  + You also need to communicate if you are doing this as a group and individually. If a group, the entire group received the same mark, and I *completely “wash my hands” of any group dynamics issues* (i.e. you are marked based on your submission to the jam and how your group behaved or didn’t behave, internally, has absolutely no bearing on the evaluation).

\*\*\*IF YOU DO NOT COMMUNICATE YOUR PREFERENCE TO ME IN WRITING (aka e-mail) BY THIS DEADLINE (two weeks before the last day of class), **YOU ARE AGREEING TO THE 2nd OPTION**:   
*“Jam = project, three final exams”* as per the CIS\*\*\*

Since this is potentially a group based project, everyone in your group must independently communicate their choice, and it must be unanimous; if the decision is *not* unanimous and/or *even one person in your group doesn’t communicate (via e-mail)*, then ***the entire group chooses the “Jam = project, three final exams” option***.

Details

Possible Topics (this is just some suggestions, not a “definitive” list or anything):

* multi-texturing and displacement
  + Decals
  + Normal maps (aka bump maps), displacement mapping, geometry and tessellation (this has to be significant, not just putting a trivial pass-through shader in there)
* Off-screen effects:
  + Blur, Bloom, Depth of field, etc.
  + “Night vision”
  + “XRay” (or something else from Meshlab or Turbo squid)
* Deferred rendering ***FULL*** deferred rendering, not just rendering off-screen, then presenting)
* LOD (Level of Detail)
  + There are many ways to do this:
  + Replacing models (lower resolution versions, like the Stanford Bunny models)
  + Replacing the model with an “imposter” (2D Billboard)
  + Adding an “imposter” that’s not a 2D billboard
* Multiple shaders in scene (or a much more complex “uber” shader)
* GPGPU
  + Tons of options here: rain, particles, depth of field, etc.
  + Trick is: a) getting data into and out of shader and b) using that data in the shader
* Shader stuff:
  + Toon shading
  + Edge detection
  + “fur” or “hair” (geometry shader or adjust model)
  + Deferred rendering (G-buffer)
* Graphical “instancing”
  + There are explicit calls for this (DrawInstanced)
* Particles:
  + Smoke, sparks, flame, etc.
* Something else (transform feedback buffer)
* Threading (this has to show a significant architecture change, usually with a performance increase, either in terms of how “much is done” in a frame and/or a significantly *reduced* CPU load. Just adding something pointless on another thread doesn’t count.
* Another (or “more”) physics things (something James isn’t marking for Physics)
  + Chains, ball-joint constraints
  + Hinge constraints
  + Stacking objects
  + Floating
* Terrain rendering (this is dealing with a lot of data, more than you can usually render)
* Make your game self aware, take over the Internet, then the world, then create robots that go back to the past and attempt to kill Sara Connor. Just an idea.

Note that this will vary depending on the course – i.e. how many projects you have done. For these courses, the “project” portion of the mark (which could be between 2 and 4 projects, usually 2 or 3) is worth 60%, while the “exams” portion is worth 40%

For example:

* Your course has had 1 project (1/2 of 60%) and 1 mid-term (1/2 of 40%).
* This “jam” is essentially “Project 2” and “the final exam”, so worth 50% of your final mark.

Another example:

* Your course has had 3 projects and 1 mid-term.
* The “jam” would be a combination of “Project 4” and the final:
  + Project 4 = ¼ of 60% = 15%
  + Final = ½ of 40% = 20%
  + For a total “weight” of 35% of your final mark

The Game Jam:

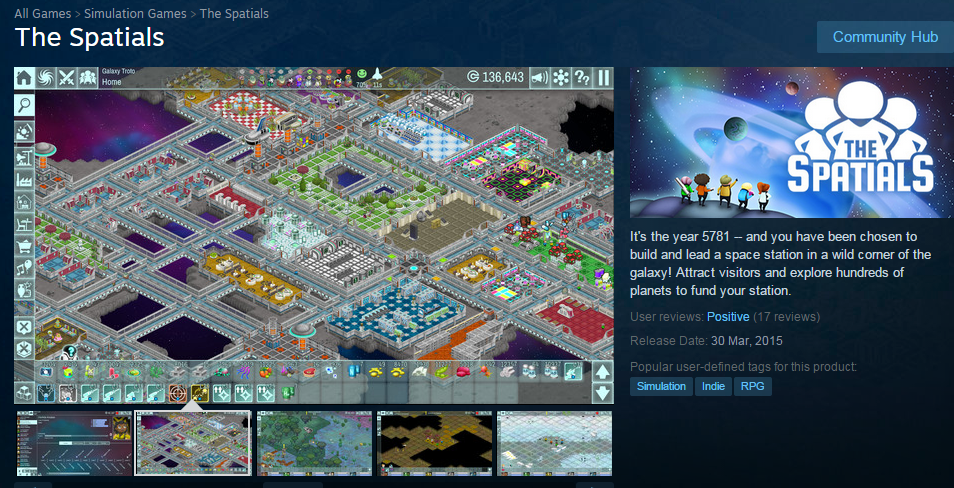
Also note that “one thing” is usually WAY more involved than it seems (like GPU particles, just rendering them is really two things: Compute Shader, then Graphics rendering, really). My point is that some topics are actually more than they seem, and you must execute *all* of these aspects to have incorporated the item – this does *NOT* mean that something like “GPU particles” covers off multiple technical things. If you are unsure, seek clarification beforehand.

**Here’s my restrictions:**

* You can use anything you have “accumulated” during the year, as a student, so anything from class, code you’ve written, etc.
* You can make something on the PS3/PS4 if you’d like. (One year, a student did a basic PS4 game *in addition* to their PC based Game Jam entry)
* *Can’t* be a game engine, like Unity, Unreal, etc., (it’s *theoretically* possible that it can be something you’ve built on, like Cocos or SDL, or something, provided you did a lot of “programming” work, but since the entire course has been a “build your own engine with OpenGL”, that’s going to be a pretty “tough sell” to convince me).
* You will have up to the start of the Game Jam to complete this.
* This *can not* be something you’ve done in another class. While you *can* use code from other classes, you can’t just submit something you’ve already submitted in another class.

**The process:**

* Use the video game name generator to pick a name: <http://videogamena.me/>
* You have to commit to one of them, so you’ll need to send me a screen capture of it **at least two weeks prior to the jam**. Once you commit to a name, you are, well, committed to deliver that (no changing your mind at the last minute)
* You prepare a brief description of what you’re doing, which includes:
  + A very short “elevator pitch” of what the game is (like you see on Steam, see below)
  + A breakdown of the “technical topics” you are implementing. You need “six things” in there:
    - There are three projects (one per course: Graphics 2, Gems, Animation 2)
    - There are three final exams (one per course)
    - You are convincing me that these things are actually implemented (through a quick video walking me through the code that does it)
* Two groups will see your game
  + “Players”, which are usually people in the IT department or students interested in gaming, but not necessarily programmers. They know it’s not some AAA game, but will give you feedback.
  + “Judges”, which are the technical people in the game industry, mostly programmers and producers. They are essentially “people who would be/or be like the people that would hire you”. Virtually all of them are well aware of the GDP program, etc., and are there to ask you tough technical questions about your entry, usually “How did you do this?” or “Why did you do it this way as opposed to that way?” and so on. Keep in mind that they are experts, and are just sort of “probing” what your answers are (and they are genuinely interested, of course).



This is an example of a “pitch” that you will give to your “audience” of technical and play testing people (the text on the right of the image with “*It’s the year 5781…*”)

Grading Scheme

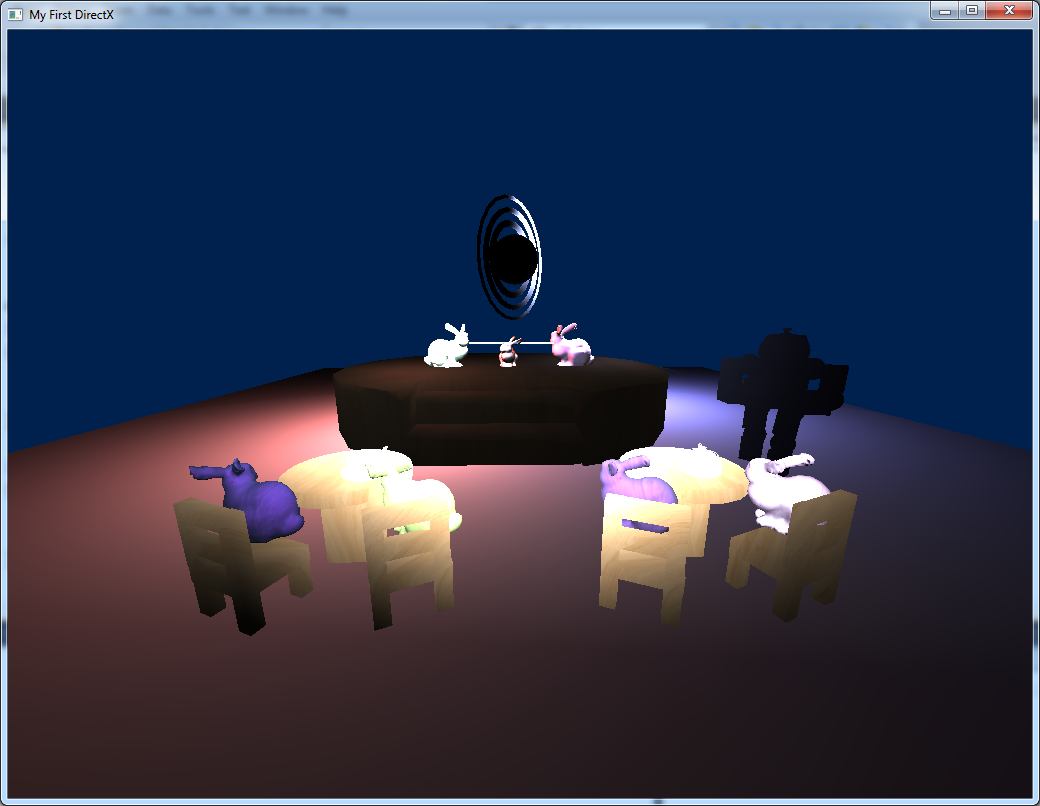
1. Normally a grade of zero will be assigned to any assignment that is submitted late. However, certain rare exceptions apply according to the Fanshawe College Academic Policies and Procedures, specifically 2-C-02: [*https://old.fanshawec.ca/sites/default/files/assets/policies/pdf/2c02.pdf*](https://old.fanshawec.ca/sites/default/files/assets/policies/pdf/2c02.pdf).
   1. Because this game jam involves getting (very busy) external technical resources/judge (people) here for one day, a “missed evaluation” becomes more complex. In that case, I will try to arrange to have at least one external technical resource to evaluate your submission, but if not, your full evaluation will be done by me, using the same format: i.e. an oral defence of your “technical things”.
   2. Also note that because this is a *project* (even though part of the mark is included as an “exam”), you have weeks to complete it; barring an “exceptional circumstance” that lasts the entire time of development (weeks and weeks), *you will be expected to submit your jam entry on the day of the game jam* – you will simply be evaluated at a later time.
2. You must submit your code so that I can see it and compile it.
3. If you code does not even compile, I will not mark it, and it cannot be submitted as game jam entry. No exceptions. This means that your mark for the game jam would be zero.
4. If you code does not build (i.e. linker error) and run (i.e. no crazy run-time crash that is unexpected), I may investigate this further, but only if there is some simple problem and/or slight configuration error.
   1. **PLEASE** include all the libraries, etc., that are required to build your project. Again, I *might* make an attempt to investigate this, but, ***technically, I don’t have to do this, therefore your project “won’t build”, so you would get zero*** (as per 1. and/or 2. above).

The six topics:

* 60%: the “six things” you have to do, technically. That have been part of one of my classes, or are something outside the class, but *you have vetted through me* ***at least two weeks before the jam***.
* They have to be determined ***at least two weeks before the jam***. This means you can’t just show up and say “I’m doing these things” – you have had to communicate your topics to me, in writing (i.e. an e-mail), at least a week beforehand. This means that if you entry has something “new” (that you haven’t indicated you are doing), it’s *possible* you could get zero on this item even if you implemented it.
  + If you convince me you are a) doing and b) “pulling off” each, you get marks based on the difficulty:
    - (0%) Something trivial. This is something that we’ve “done to death”, is something that has been incorporated for a while, or is something that would take a professional minutes to do – this includes the trivial incorporation of alternative libraries (like assimp instead of ply, or solid instead of texture loading, etc.). This also included anything you’ve done in any other courses (look, I incorporated Bullet Physics! – which I did for Physics 2…)
    - (75%) Easy (something you just coped from class/internet, virtually unchanged)
    - (100%) Hard (something we covered but is somewhat involved in doing and/or incorporating)
    - (115%) Super Hard (something we barely or didn’t cover, so you had to investigate this all yourself, and *also* is technically changing)
  + You get 1/6 of the mark for each “thing” you do. So if you only do 5 of 6, then you can only get 5/6 of the marks (assuming they aren’t all “Super Hard” that is...)
* 20% : the “ranking” of your projects based on the feedback from the players
* 20% : the “six things” + “ranking” based on a technical lead from a local game company
* Note: I reserve the right to “override” these last 40%, which may include (but not limited to):
  + You’re given some extreme mark, like all zeros (it could happen, I suppose).
  + You’ve violated one of the prior rules, like you did completely different “technical things”, but rather than embarrass you/the college/myself, I’ve let you present – but since you technically **didn’t tell me at least two weeks in advance**, you (technically) would get zero.
  + Based on me seeing the code and/or seeing what’s happened in class, etc., I know that you didn’t actually implement this. For example: you used boost; you used a 3rd party library for functionality that you are claiming you did yourself, etc.
  + You are presenting something for the 2nd time and/or are using something from another class. For example, if you’ve already done AABB collision in 1st term (project #2 of Physics 1), but you haven’t changed anything (or trivially changed it), then you can’t submit it again. Same thing for including something from another course, this term, that you have already submitted for a mark in that class.

Additional requirements:

* While you may freely “borrow” mine (or anyone other) code ***but*** your code should be “sufficiently” different. See “70/12-year old ‘squinty eye’ plagiarism test” for more details.  
    
  To clarify this: while I don’t expect you to recreate everything from scratch (although I encourage this), I don’t want to see something that is essentially my in class demos submitted back to me, essentially unchanged. In other words, I need to see that you have put a lot of work into it.  
    
  As a guide of these terms “essentially unchanged” and “a lot of work”, use what me – or one the highest performing student in the program – could do: if what you did would take one of us < ½ an hour to do (like changing the names of the 3D model files, placing them at another place in the scene, etc. – well, that would take, like, 60 seconds, but you get the idea), then it’s *not* “a lot of work”.   
    
  For example, this was created by a fourth semester CPA student (i.e. *NOT* in the GDP class) for a final project in the *FIRST* DirectX class I delivered in summer 2011 (DX10):



* The bunnies in the back skip rope.
* The other bunnies bounce in their chairs
* You can move the waiter around
* There are three separate lights
* It’s DirectX 10 (summer 2011)
* I’m expecting more from you.
* You also *cannot* simply use an existing game engine (or part of a game engine, Ogre3D for instance) to complete this assignment; it should be either completely new of significantly modified.

GDP Game Jam 2018: Wednesday, April 25th, 2018

|  |  |
| --- | --- |
| Student name(s): | |
|  | |
|  | |
|  | |
| Game name (from the name generator): |  |

Description:

|  |
| --- |
|  |

Feedback, ranking, and recommendations (0-5: think “0 stars”, “5 stars”, etc.):

Keep in mind that these developers are:

* Here for programming, not art
* Are \*not\* using a “game engine” – everything here is implemented “from scratch”
* The only exception would be things like the Havok Physics engine, but even that is a huge struggle to integrate
* Think of these as early “Early Alphas”, “Early ‘Green Light’”, or “Proof of concept” sort of designs, not complete, polished products ready to hit store shelves...

|  |  |
| --- | --- |
| Fun? | |
| Close to what you expected (from name+desctiption)? | |
| Game play (nightmarish, awkward, good, etc.)? | |
| Appealing? | |
| Like to see more of it when it’s “done”? | |
| Anything else (Balance, etc.)? | |
| Evaluated by: |  |
| GDP Game Jam 2018: Wednesday, April 25th, 2018  Student name(s): | |
|  | |
|  | |
|  | |

|  |  |
| --- | --- |
| Game name (from the name generator): |  |

Description:

|  |
| --- |
|  |

Technical “things” implemented (see next page for a suggested list):

|  |  |
| --- | --- |
| 1. | |
| Easy (75%), Hard (100%), or Super Hard (115%)? | Mark (could be zero): |
| 2. | |
| Easy (75%), Hard (100%), or Super Hard (115%)? | Mark (could be zero): |
| 3. | |
| Easy (75%), Hard (100%), or Super Hard (115%)? | Mark (could be zero): |
| 4. | |
| Easy (75%), Hard (100%), or Super Hard (115%)? | Mark (could be zero): |
| 5. | |
| Easy (75%), Hard (100%), or Super Hard (115%)? | Mark (could be zero): |
| 6. | |
| Easy (75%), Hard (100%), or Super Hard (115%)? | Mark (could be zero): |
| (7.) | |
| Easy (75%), Hard (100%), or Super Hard (115%)? | Mark (could be zero): |

|  |  |
| --- | --- |
| Evaluated by: |  |

**Possible Topics** (*not* an exhaustive list)**:**

* multi-texturing and displacement
  + Decals
  + Normal maps (aka bump maps), displacement mapping, geometry and tessellation (this has to be significant, not just putting a trivial pass-through shader in there)
* Off-screen effects:
  + Blur
  + Bloom
  + “Night vision”
  + “XRay” (or something else from Meshlab or Turbo squid)
* Deferred rendering ***FULL*** deferred rendering, not just rendering off-screen, then presenting)
* LOD (Level of Detail)
  + There are many ways to do this:
  + Replacing models (lower resolution versions, like the Stanford Bunny models)
  + Replacing the model with an “imposter” (2D Billboard)
  + Adding an “imposter” that’s not a 2D billboard
* Multiple shaders in scene (or a much more complex “uber” shader)
* GPGPU
  + Tons of options here: rain, particles, depth of field, etc.
  + Trick is: a) getting data into and out of shader and b) using that data in the shader
* Shader stuff:
  + Toon shading
  + Edge detection
  + “fur” or “hair” (geometry shader or adjust model)
  + Deferred rendering (G-buffer)
* Graphical “instancing”
  + There are explicit calls for this (DrawInstanced)
* Particles:
  + Smoke, sparks, flame, etc.
* Something else (transform feedback buffer)
* Threading (this has to show a significant architecture change, usually with a performance increase, either in terms of how “much is done” in a frame and/or a significantly *reduced* CPU load. Just adding something pointless on another thread doesn’t count.
* Another (or “more”) physics things (something James isn’t marking for Physics)
  + Chains, ball-joint constraints
  + Hinge constraints
  + Stacking objects
  + Floating
* Terrain rendering (this is dealing with a lot of data, more than you can usually render)
* Make your game self aware, take over the Internet, then the world, and then create robots that go back to the past and attempt to kill Sara Connor. Just an idea.