**My tips on how to plan a game, stay motivated and get it finished.**



Introduction

The question of how to motivate oneself or finish a huge project is a very common question here. I wanted to share my personal methods.

This guide was originally posted here on [r/gamedev](https://www.reddit.com/r/gamedev/) 8 months ago. I'm reposting it for those that missed it and to update it with new lessons learned.

**NB:** This primarily applies to very small indie teams or solo developers such as myself, but some of the methods could be applied to specific tasks within a larger team.

**Who am I? Where is the proof that my methods work?** I'm a solo developer with more than 5000 hours of work into my current game, taking it from prototype to a 100+ hour playable game. The game is arriving on Steam in less than two weeks. You can see it [here](http://store.steampowered.com/app/296010/).

\_

Getting Started

Most games revolve around a 10-30 second gameplay experience that is repeated endlessly. I first heard this concept described as a core foundation behind the design of Halo. This is the core of your game and if this isn't fun, it's unlikely the final game will be fun either.

* Decide on the key concepts and gameplay elements that are at the core of your game.
* Don't waste weeks or months planning out every last detail for your game. (I first read this tip from Minecraft's Notch)
* Start work on it now. Don't procrastinate. (Also advised by Notch) Start as you mean to go on. As Nike says 'Just Do It'.

\_

Passion and Dedication

Choose a game genre you fully understand and are passionate about.

* Without passion you will never get it finished.
* Without dedication you will never get it finished. If you struggle with this then practice it in any avenue of life. Learn how to finish things before embarking on bigger things.
* You will have more success making a game for an audience you fully understand that is small, than for a bigger audience that you don't understand. You need to know what your audience wants if you want to have a chance of pleasing them.
* Make a game that you passionately want to play.

\_

The Prototype

The goal of the prototype is to create that core 10-30 second experience to ensure that it's as fun in reality as it was in your imagination.

* If it's not fun, tweak it until it is.
* Don't waste time on fancy graphics at this stage
* If you can't make it fun now don't assume that adding more crap around it will make it fun later. That's highly unlikely.
* Once it's fun, continue to add the core features (in a very rough but functional way) that you believe are essential to your game concept. It's better to make sure the whole concept works now before you waste too much time on a dud.

The prototype can take anything from an hour to a month depending on the complexity of your final game.

**If it's not enjoyable now it's unlikely that it ever will be. Don't build a game on broken foundations.**

\_

The Master Plan

Now that you have a decent prototype it's time to plan out in a very rough way your schedule for the project. If you don't have a schedule you will fumble around endlessly wasting time and never get the thing done.

**List all the core features that your game will need.**

* Don't detail how you will achieve them. You may not even know how you will achieve them at this stage. Just list them. Also, list all the essentials that every game needs eventually - things like save game functionality, a website, rebindable keys possibly.
* Do this in broad strokes. We are not looking to list every minute detail here, just an overview of the big picture - each big job.
* Once you have your list, estimate how long each job in the list will take and write it next to it.
* Total up the time for everything. **Now double it!** Seriously. Even if you are very conservative in your estimates, almost everything is going to take longer than you expect, and you are going to run into endless jobs that you never predicted.
* If the final total is not something that you think you can achieve reduce the scale of your project and repeat the above process until you arrive at something manageable.

\_

Breakdown the Master Plan - Chunks

* Split all the jobs in the master plan into 'Chunks'.
* If your planned game will take two years you may want to break the list down into 'Chunks' that will each take three months each.
* If your planned game will take 3 months in total, break it down into 2 week 'Chunks'
* List your 'Chunks' in the order that they should be completed.

\_

Take the first 'Chunk' and break it down - 'Pieces'

Even if your game is only planned to take 3 months to complete, you are still going to have a lot of work in each 'Chunk'.

Break the first 'Chunk' down into a new list of 'Pieces'. Again, don't get into details here. A 'Piece' might be something like - 'Create a basic GUI Interface' or 'Create assets for game feature X'.

Be sure that you have enough time to complete your 'Pieces' for that 'Chunk' in the timescale you have allocated. If not, you may need to move back up the plan and reduce the scale of your project.

\_

Breakdown this first 'Piece' of the first 'Chunk' - Immediate Job List

Each 'Piece' might be still quite complex, and you may not know how to achieve it yet. As an example, our first 'Piece' might be something as broad as 'Implement the user interface' that could take two weeks to achieve. Now break that down into another wide brush stroke list, for example:

* Implement the start screen
* Implement the menu system
* Implement the HUD system
* Etc

Once again don't detail each job yet. Just list the jobs.

\_

Breakdown the first Job in your Immediate Job List

By now you get the idea. Hopefully each job in our 'Immediate Job List' will take no more than a day or two. For smaller projects you will probably be already down to jobs that should only take a couple of hours and you can skip this stage.

So for example we might break down 'Implement Start Screen' into this new list:

* Create the background
* Create the main menu (New Game, Start Game, Options, Quit)
* Implement the code to make the main menu function.
* Add some juicy special effects to make it look nice.

\_

Pick a job and break it down

So let's imagine we have chosen the job 'Implement the code to make the main menu function'.

Finally we are at the micro scale. We now plan in detail how to achieve this single job. Break it down again. List each little piece of the job that needs to be done.

It's a good idea to also now have a rough idea how you will achieve each little piece before getting started. This will help you predict problems that may occur with your chosen method.

This shouldn't take more than 5-10 minutes. Maybe longer if it's a complex problem that you need to do some research on first.

**Now do it!**

Rince and repeat stepping backwards through the processes. Do all the little jobs to complete a job on your 'Immediate Job List' Then pick a new 'Piece' of a 'Chunk' and create a new 'Immediate Job List'

**Then do it!**

\_

The advantages of this method

* It's structured with timescales in place to get it all done.
* The plan is all broad strokes that shouldn't take long to list initially.
* It breaks down massively complex systems into tiny, easy, bite-sized pieces.
* You only get down to details just before actually implementing something that will be finished in a few hours.
* No sooner have you planned the details than you are implementing them while the problem is still fresh in your mind. This keeps motivation levels high and saves time. You aren't trying to remember something you planned six months ago.
* You get to strike jobs off your list quickly. Don't underestimate how motivating this is. You see progression happening in a visual way constantly. There is nothing better than seeing a job list disappear.

\_

Staying Motivated

**Don't procrastinate**

* If you ever allow yourself to think 'I can't be bothered right now', push it out of your mind and get working immediately. Don't give it any time to fester and gain traction.
* We are stupid creatures. We live by habit. If we allow ourselves to not work because 'we can't be bothered' this becomes a habit and it will happen more and more. Don't let it start. Do the opposite - get in the habit of just doing it and then it becomes easy to just do it.
* However, if you are in the habit of just doing it, and you get a strong feeling of 'I can't be bothered' you are likely genuinely tired and need more sleep, or you're overworked. See below.

\_

Don't Overwork - a recommended work schedule.

*Applies only to full-time development*

There is a reason the average working week is 40 hours. This has been proven over time to achieve efficient results. This is especially true in a concentration intensive job like game development.

* Do 8 hour days, 5 days a week
* Take a 5 minute break to get up and walk around once every 45 mins. Give your brain a break.
* Take the weekend off to relax, recharge, and motivate yourself for the following week.
* Get enough sleep. Don't underestimate this tip.
* Take a week off once every 3 months. This is essential or you will burn out. You need time away from thinking about something or you fry your brain so it's of no use to you.
* You will be more efficient and get more done by not overworking. Overwork makes our brain go around in circles while problem solving. If we are fresh, problems get solved very easily.

By not overworking, and getting enough sleep, your motivation levels will always stay high. - Well at least until mid way through Friday :)

The only times I feel like 'I can't be bothered today' is when I'm genuinely physically tired from lack of sleep or because I've overworked recently.

\_

Get it out there

* As soon as you have a playable alpha get it into the hands of your target audience.
* Player feedback is essential to making a great game.
* Listen to all feedback. You may not act on it, but you need a solid reason why you aren't acting on it.
* As your game expands, get more people playing it.
* You might have an ultimate goal of getting it on to Steam, but put it out there on a smaller platform once you have a stable product. Price it so that it offers value for money at that early stage. You are doing this not to make money but for experience and to get motivated from player feedback. You will also find out if you are wasting your time.

\_

Conclusion

I hope this is useful info for some of you. It's not going to work for everyone but it's worked for me and I'm still motivated after nearly three years of development.