__Warm-up Questions__

1. **Q1: How long have you been working in the game industry?**

1.1.	P1	=>	2.5 years.
1.2.	P2	=>	2.5 years.
1.3.	P3	=>	4 years.
1.4.	P4	=>	5 years.
1.5.	P5	=>	2 years.
1.6.	P6	=>	0.5 years.
1.7.	P7	=>	3.5 years.
1.8.	P8	=>	3 years.
1.9.	P9	=>	3 years.

2. Q2: What’s your favorite part of working in the game industry?

2.1. P1
The creativity or the problem-solving. A mix of both. Not necessarily unique because of game development but technology in general.

2.2. P2
The ability to use my creativity, similar to what I do in my freetime. Feels like I’m doing my hobby at work. I get paid to do what I like.

2.3. P3
It’s fun to solve problems. It’s like a long line with small puzzles. When you solve a puzzle it’s very rewarding.

2.4. P4
Creative problem-solving. To get many interesting and different problems and solve them in specific ways. The collaboration with the different disciplines and solving problems together.

2.5. P5
I like to design and create experiences, and to make interactions. Doing things that are fun to play. Technical parts are also interesting, but mainly game-interaction.

2.6. P6
A lot of things are going on. Many different things all the time. It’s the variation.

2.7. P7
It’s dynamic, a lot of things happening fast. You do something and then test it, but then, nope, that’s not the way you had thought about it from the beginning. You need to be fast and adjust to things. Especially compared to normal software development, where it is more rigid.

2.8. P8
Always had an interest in games. Started studying software at Chalmers. But that wasn’t as fun, I wanted to do frontend that was more focused on the experience.

2.9. P9
To create things. Environments. Create things that look good, to work on bigger and bigger things and many various parts like lightning, materials and create immersion.

3. **Q3: What’s the most challenging part of working in the game industry?**

3.1. P1
Communication and coordination. Many people are working on the same system. No clear architect. Understanding designer desires, and developing things according to the desires of artists.

3.2. P2
If there are problems, like any type of problem, like technical problems, you have to go back several steps or look up things by Googling, which is horrible. Like things that occur in the software Maya.

3.3. P3
It can be hard to know what you are working towards, especially in the beginning of the development. It can be hard to make an estimation of what is fun and what you will be able to make depending on the time. Feature creep is regularly occurring.

3.4. P4
To make the art director happy. I'm just joking. Seriously though, communication is always hard. It's easy to either talk too much and work too little, or the opposite, work too much and talk too little. Talking too little can have bad effects since you're not synced.

3.5. P5
That there are many disciplines. To tie everything together. Communication is the most demanding, that's the part that is most problematic.

3.6. P6
The structural part. It's hard to interpret what processes are urgent and not, and how you should structure the work. It's very chaotic.

3.7. P7
Same as my previous answer. It's both fun and challenging.

3.8. P8
Problems with the disciplines. Some people have poor insight into your discipline and vice versa. Some requests are definitely not possible to do, like requests from publishers. It's fun and annoying, sometimes there is no way to find help regarding very specific things. It can be difficult when someone comes with requests but has no insight in the backend of the project.

3.9. P9
To know many things. It's probably not as straightforward as many other jobs. There is an unending box of things to learn, it's an endless ladder of things you can get better at. It's clear how you can develop yourself, the more you spend time on something, the better you get, especially in the art department.

4. **Q4: What's your work experience prior to working in the game industry?**

4.1. P1 => 10 weeks of software development at Volvo. BSc in game programming.
4.2. P2 => Night guard, truck driver, caretaker. BSc in media/art.
4.3. P3 => Customer service. BSc degree.
4.4. P4 => Painter, craftsman. Started but didn't finish my BSc degree.
4.5. P5 => Technical support. BSc in game design, one year of studies in programming.
4.6. P6 => Truck driver and teaching assistant. BSc in game design.
4.7. P7 => Web developer for 5-6 years. 6 months at Ericsson. BSc in game programming.
4.8. P8 => Summer Jobs at steelworks, ICA.
4.9. P9 => Started here straight after school. BSc in game graphics.

5. **Q5: Which game development discipline would categorize yourself working in? Design, art and/or programming?**

5.1.	P1	=>	Programming.
5.2.	P2	=>	Art.
5.3.	P3	=>	Art/technical artist. Also programming.
5.4.	P4	=>	Art.
5.5.	P5	=>	Design/Production.
5.6.	P6	=>	Design/QA.
5.7.	P7	=>	Programming.
5.8.	P8	=>	Programming.
5.9.	P9	=>	Art.

Substantive Questions

1. **Q1: What is your personal experience/opinion on the used development process?**

1.1. P1

A variant of scrum with a producer, aka scrum master, that plans tasks for the different disciplines. Disciplines have to communicate and correct producer faults. We work in 3-week sprints on the tasks given by the producer. The process is kind of sporadic. The design department can make requests that need to be fixed if there is time for it. Otherwise the tasks are saved and discussed with the producer.

1.2. P2

Sprints of all kinds are great. Keeping track of whatever you do. Syncs are also good, be it weekly or monthly or whatever. Sprints depend on the company. Some companies attempt to just wing it, which results in catastrophes. I think a big part comes down to planning the sprints. Communication is very important as well.

1.3. P3

I think it can be better. It's hard to know in advance what we are working towards. The vision is changed constantly. The art director is the one that knows the specifics of the vision, and it can be hard for him to relay the vision concretely and clearly. It's hard for everyone involved so see the whole vision. Making new content is difficult without seeing the whole picture.

1.4. P4

Generally good. In previous projects and companies though, not so good. Earlier processes were bad, but since then they have been updated and tested a lot. I would really like another VFX guy to talk to about VFX-specific things, now I'm the only VFX guy in the company. But as a whole the current process is good. We get tasks, do what is necessary. Everyone gets a say and can express their opinions.

1.5. P5

It works. Of course there are things that can be better in communication. Better transparency in the planning. This has both pros and cons though. To show the entire plan can be stressful for people to see. For example, they can get overwhelmed and worry that they won't make it in time. On the other hand if you do not show the plan, people get surprised by new tasks that come up. But the communication between the disciplines is quite separated, but works relatively well I'd say. It would be good with directors in other departments than just the art department. A previous project lacked a proper design department and suffered from it, but in the current project we need people dedicated to level and gameplay design.

1.6. P6

Good. I don't have that much to compare with. Feels like an okay process. No big problems.

1.7. P7

Pretty good. It reminds me of my previous jobs. We do sprints, taken from Scrum, done this alot before. It varies more from company to company rather than between traditional software development and game development I'd say. It's very individual in companies how they structure the teams. Compared to how we do it now and before, now it's a lot more individual work now compared to before. Before it was alot more direct collaboration. For example now, one guy is sitting with EOSS specifically. Before it would have been the entire team sitting with it rather than one individual. Varies more from company to company, it's an industry thing.

1.8. P8
It's a bit messy, it hasn't been entirely clear who is responsible for what. It bothers me when not knowing whose responsibility something is. Everyone is equal, only our two bosses are above the rest. Would have preferred a more clear hierarchy. Now you go to the producer, who hasn't got enough insight. It's a lot of freedom, more structure has contributed to better products. Sometimes you gotta crunch. Publishers completely decided on one thing, but the programmers were not in sync. Then our Producer suddenly came with a new request. Then you either gotta just say yes and get it done, or get unfriendly with the boss.

1.9. P9
It's nice to have sprints, you feel where you are in the project plan. It's a clear start and end that can indicate how far into the task you are. Like these things are supposed to be done this sprint. It's a feeling of time moving forward. It can be tiresome to use the 3-week structure if you work on one single task for the entire 3-week period. But generally it's good for keeping track of time. A Lot of opportunities to go back to previous tasks.

2. **Q2: Why do you think the used development process is suitable or not suitable for the game development process?**

2.1. P1
The process fits pretty well because there needs to be some long term planning to have a sense of direction, with milestones and planned sprints. In game development, the specifications and requirements change often from the design department. Some design demands can require big changes in the code architecture, which is inconvenient. But generally the process is pretty flexible.
No opinions on why it would not fit game development.

2.2. P2
Sprints are good. No issues so far. Sprints are important for all types of companies, there is no difference for the game industry.

2.3. P3
Kanban is good. Moving cards and tasks allows an overview of the tasks. Time estimation is difficult. Sometimes you need to wait for other people to finish their task to get started on your own. Also known as blocking. For example, someone doing animations can be waiting for me to set up the rigging.

2.4. P4
It fits because we work in a creative industry where opinions can change often. People can have completely different opinions about things. Everyone can say their opinion and everyone gets heard. It's easy to see what is bad and what is not bad. The iterative approach is suitable because of the changing opinions.

2.5. P5
It fits well because there is a clear pipeline, but it needs to be even more clear. Communication between disciplines needs to improve. In short; we need to be better at what we already do.

2.6. P6
Agile fits very well because you don't really know how the end product is going to end up. The sprint structure fits well because you're kind of working towards a vague goal.

2.7. P7
The sprint-thinking and playtesting fits very well for the game industry. You need the iterative process. You test and tweak depending on what people think. That is especially important for the industry. For games it's hard to know in advance what feels good and what is a good game, sort of.

2.8. P8
It fits very well. Sprints are good. It's alot better with sprints than without sprints. Clear milestones. Good structure since we manage to finish things in time. We can re-prioritize, everyone gets their say. We got more than compared to when it's more open.

2.9. P9
It's about production. Compared to other jobs, they have a flow of similar tasks, such as a cashier with a flow of shoppers. It's a delivery of food all the time. Here we work towards a goal that needs to be divided into subtasks. Like we're not going to make 100 unique weapons. It's not like tech support. Working towards a product is more simple.

3. **Q3: What's your experience collaborating with the other disciplines involved in the development process?**

3.1. P1
There is too little communication. Someone makes a model or design mockup with the assumption that the other disciplines can adapt to it. For example, if the design department wants cool water physics, they assume that the programming department will handle it, but in reality it is a complex problem. Programmers can misinterpret requests from the design team, and develop the wrong system. It is fun to collaborate though, and insightful. It is fun to see the work of others.

3.2. P2
Communication is mostly within the art department, such as with animation. Redoing tasks because of changes can come from design or other people in the art department. It is good if the higher ups that are demanding the changes can be united in an opinion. It's difficult to satisfy different opinions.

3.3. P3
Design can request new enemies, such as the Golem Boss, and how it will work when it moves. This becomes a bit of a back and forth process since design can change their opinions regularly. Design asks how hard it can be to implement these ideas.

3.4. P4
Very varying. It depends on the person, some people have strong opinions and are more difficult to work with. But in the end we all want to work towards the same end goal. But generally this process is actually fun. Collaboration is good. Different disciplines say different things, and you have to solve it. Personally I arrive late to the most recent updates and have to adapt to the designers afterwards and ask how things are supposed to work. For example what is supposed to happen when the enemy slams his hand in the ground. Questions about details.

3.5. P5
It works well. Communication is often lacking though. If not everyone participates it won't end up well. Design is the pillarstone that tells everyone else how everything should be. It works well. We really want to know if there are things missing from the other disciplines. They should let us know.

3.6. P6
It's fun to work with the other disciplines.With the competencies in the various fields that you can rely on. Some people can have a solution and completely different thoughts on how to solve a problem.

3.7. P7

Usually you end up talking with the design department a lot. Personally, there isn't that much collaboration with the art department. It’s because my tasks are more backend-oriented, while gameplay programmers are more communicative with design etc.

3.8. P8

It’s good with nice people. Some people don’t really understand problems, they say “Oh but that’s not really a problem”. Yes, it is, they have to understand that. They don’t take it seriously and leave it unfixed. Months ahead the problem is still there. Optimizing graphics needs to be done. As a programmer I can spend more time on getting insight into other disciplines’ problems, but then my own sprint will suffer from it. Shared knowledge is something the other disciplines don’t understand. It’s hard to share knowledge, they don’t understand various parts. It usually ends up with a discussion with the design department. When publishers say something, then that's the way it is. They can threaten to withdraw the contract. So we released a product, and they were not satisfied. They’re usually like “we want it like this and that, or we’ll withdraw”. People don’t like when you go around poking in other disciplines. We're making a game, big egos don’t work.

3.9. P9

Before, if we needed things from other disciplines or vice versa, it was through direct communication between the disciplines, now it all goes through the manager. It was good before too, but it would not have worked out with this many people that we have now. It depends on how you look at it. It’s boring with less direct collaboration between disciplines, but it’s more effective.

4. **Q4: How do you handle changing requests/suggestions/demands from other disciplines?**

4.1. P1

It depends. If done iteratively, it is not that bad. Rewriting the same thing due to design decisions going back and forth is tiring. But needs to get done. It’s tiring to do the same iteration many times.

4.2. P2

By contacting the responsible higher ups and letting them know about the varying opinions of other higher ups, then let them discuss which is actually the best option, so that they are united in their opinion. If not, then I use “blockins”, which are kind of like drafts, and present them to the higher ups, so that I don’t spend too much time on a piece that will later be discarded or changed. Redoing art assets can take a long time.

4.3. P3

Just get it done. It’s a balancing act. Is it worth the time it will take to apply the changes? These changes are most often from the design department, and rarely from programming, but it happens.

4.4. P4

I make sure that things are saved that might be used later when designers want to switch to a different variant of things, so that I don’t have to redo a lot of work. If the changes are extreme you have to question why they want it that way. I leave comments like “do not remove in case design may want to use this”. I have alot of control in VFX on my own though.

4.5. P5

Nah, it can happen sometimes. You need to think like this so you know what to do in animation. Works pretty well. Not really that much in the design department.

4.6. P6

It depends on how urgent the requests are and where my resources are needed at that time. You have to balance how important it is that it gets done soon versus how big an effect it will have on the project. For example, right now I’m working on an old project, but if a crash report occurs in the new project, I need to tend to that immediately.

4.7. P7

I think I handle it pretty well. It changes a lot and it changes often. It's fun and challenging, you have to think about when you do your things. You have to think about whether it will probably require change later on and then you have to make your system flexible for changes. You need to be more open. If you know more clearly in advance what your requirements are, then you know how to design your systems. You need to have that kind of mindset and preparation that down the line things are going to be changed.

4.8. P8

I hate it, I don't like it. I want clear specifications. Everytime I get a specification, I walk through it from point to point and bring up potential problems with the design department. If they don't like it, it can contribute to making it harder to change later on. 6 months later, it suddenly needs overhauling. Navmesh has had to be revamped 3 times now. If it had been clear from the beginning I would have saved like 6 weeks of work. It's not a problem to redo things, it's just that it contributes to bad productivity. To change design later on. I prefer clear things from the beginning. Would have preferred prototyping.

4.9. P9

Personally I don't care. It's stuff that happens. Get it done, just do it.

5. **Q5: In what ways does collaboration with other disciplines affect the quality of your work?**

5.1. P1

It's nice to delegate the work between different disciplines. I am used to the time in school or freetime where you do everything by yourself, but delegating the work to experts in the different disciplines allows more time to spend on your own discipline.

Delegate work to other disciplines allows more focus on ex programming and making things correctly. The collaboration allows more time for quality work on code etc.
Feedback loops with other perspectives allows better work quality, communication, and less iteration.
Changing requirements/tight deadlines can have a negative effect, many iterations, crunch burnout, but I attempt to work against burnout and crunch time. Duct tape solutions do not hold up in the long run. Take it as it comes. If you can't make it in time then so be it. No compromise for crunch time.

5.2. P2

It can take time. It depends on how fast the higher ups want a task to be finished. More time might be required for higher ups to approve the work. Again, varying opinions can affect the quality of the assets I need to create. However, technology in the Unreal Editor allows me to work quite flexibly so that I can for example quickly apply new textures to models without having to go through tedious procedures in the art software. But yes, changing requirements can definitely affect the quality of work in a bad way.

5.3. P3

Mostly positive. It depends on who you are collaborating with. Sometimes you can ask for help from, for example, the programming department. If I'm working on a new feature, I can just go and ask them. Whatever design wants they will get it. This is both positive and negative, but also quite fun.

5.4. P4

The programming department can bottleneck me pretty hard if I want to do cool stuff in VFX, for example if there is a lack of information that I need. For example the bullets in the game are not removed, they are pooled, which greatly affects the way I create VFX for bullets. The design department is more like "make it work". Design changes their opinions a lot from the original idea. The collaboration with the other disciplines contributes to better quality in my work. If I just sit on my own, my work gets isolated and may not fit into the general context later on. By talking to the other disciplines, I get feedback to make sure the work fits into the context. It's a little bit irritating sometimes, but is better in the long run.

5.5. P5

If you can’t collaborate then my ideas won’t be visualized in the game. Sometimes I get back a response like: “no, that won’t work”, or: “the specs won’t work”. Feedback from other disciplines can affect how the design department works. This affects the quality of the final result. Sometimes, after playtest, when things are not working as intended, we can get feedback like: “but that’s the way you wanted it”. Non-iterative and strong opinions can affect it negatively. Flexibility is necessary.

5.6. P6

If I find something weird then I will be able to get an explanation. I can see that lightning takes 20ms on one frame, and I will talk to the art director. You can get a completely different detail level on what to do. For example when we had spikes when VFX were spawned, then talk to this person, who will look at it, maybe we should pool this thing. Then we have other things that we pool, and async and stuff. We can then identify and log a problem and present solutions. Like, we need to do this to avoid the problem. The collaboration with the other departments increases the quality as long as there is open communication.

5.7. P7

Hard to say. Neither positive nor negative. It can have an effect. But you do it from the start. Opinions changing 180 degrees of course will have an effect. Refactorings can affect it. Optimally you would have rewritten it from the start, but due to time it’s not optimal to rewrite, so the quality gets lower.

5.8. P8

Most often negatively. If I would have done it myself, I would never have done it this way. But in the long run I think that the end result gets better by collaboration because I cannot have all the knowledge myself. It gets better by inputs from the different disciplines. Less work for myself with a team. In the end I think with a team you get a better product to sell.

5.9. P9

It depends on the time. Different things take different amounts of time. If new changes need to be done faster then we will lose quality. But we are making a game from the ground up, so design opinions have more prioritization. But even if my own tasks or projects get worse it fits better in the whole picture. Listen to the design department because it is the game as a whole that is the important thing. It depends, personally I think some things are bad but I still accept what design says. Everyone has an opinion, as an artist I tend to listen more to the designers than the programmers. Also I tend to listen more to senior developers than new recruits.