In this lesson, we're going to dig into building complex systems that you operate yourself, which is the next level of system design.

If you remember from our previous lesson, there are two types of systems: simple systems and complex systems.

In order for a system to qualify as simple, it has to have just one goal or major outcome. It should also be linear in nature, meaning that each step in the system is performed individually and sequentially. And the system should be free of decision-making between steps. No step should be dependent on the outcome of another step.

If a system doesn't meet all three of those criteria, we would call it a complex system.

In a complex system, there could be multiple goals involved. For instance, a major construction project could include the construction of multiple buildings—each one of them being its own goal and having its own steps and processes to complete. Or maybe a road is being installed around the building. The steps of the road construction is part of the overall project, but is also very much it's own major endeavor.

A complex system may also be non-linear, meaning the steps are not necessarily followed in an explicitly defined order or different steps could even happen simultaneously.

The construction project is still a good example here. In a five-building development, the same steps could be happening for two different buildings at the same time.

And an example where different steps are happening concurrently might be doctors and surgeons triaging a patient in the emergency room. If the patient has multiple injuries, many doctors may be working together, but on completely different parts of the patient's body at the same time to stabilize the patient all while a nurse monitors vital signs and reports them to everyone involved.

Finally, a complex system could be one that requires some decision-making throughout its operation because the the outcomes of the steps are not guaranteed. You may need to analyze the results of a step and make a decision to use a different part of the system based on those results.

A really simple but relatable example would be driving your car to the store. You might know the route to take, but lots of things may come up along the way to cause you to change course. If a traffic light changes, you have to make the decision

about what to do. Do you press the gas pedal to accelerate through the intersection, or do you press the brake pedal to stop and wait?

Or what if there's too much traffic on your normal route. There are probably other ways to get to the store that are normally inconvenient but, depending on road conditions, they might become the optimal route.

You could decide that if you're stuck in traffic for more than 5 minutes, you'll turn off the main route and find another one.

This is the most common criteria you'll work with when it comes to building complex systems for yourself, so we'll focus mostly on this one, which is decision making.

And we'll call that decision making "conditional logic." What I mean by conditional logic is that, when the conditions within the system change, different logic needs to be used. So, one set of conditions might dictate that you follow one path of logic or steps, while a different set of conditions would dictate another path.

An investment banker managing a portfolio of investments is another example where a complex system might be required with conditional logic. Their job is to make the most money—or at least lose the least—with the funds they have to manage, but there is no way they could make smart, repeatable decisions with a simple system, because they don't control how the market behaves. They need to make decisions about what to do when the market is going up and also when the market is going down. And those decisions probably look different. A one-size-fits-all approach won't really work.

So, how do you know when you need a complex system?

Sometimes, you'll know from the very beginning that you need a complex system and, if it's truly obvious to you, then you can feel free to pursue that path.

But my advice will always be to start with a simple system if you can, and adjust it into a complex one over time.

I prefer to work that way because it's the fastest way to get started and see results without getting bogged down.

When you're creating a system to try to either speed up your processes or get better or more reliable results, what *actually* needs to go into that system can be deceiving at first.

Things often look more complex than they actually are, and you start down a path of creating variables or planning for situations that may never actually come. It's never a bad thing to be prepared, but there's a real opportunity cost to spending too much time planning when the real value comes from doing.

Sticking to the limitations of a simple system—you know, one goal, linear progress, no conditional logic—is a good way to make sure you actually complete something and are able to quickly test how it works.

Then, when you run into real problems that can't be solved by your simple system, that's when you should look to expand the scope of the system and implement some of the more complex functions like accounting for multiple goals or adding conditional logic between steps.

So, really, the goal is to build the most simple system you can, and then keep it as simple as possible by iterating on it and improving it in small steps so that you're only spending your time fixing real problems that you run into and not spinning your wheels by anticipating every possible problem and trying to engineer them out from the very beginning... even though that can be tempting. That's what often keeps people who *want* to implement systems from actually getting started.

Alright, let's actually walk through how to do this. Let's create a complex system together.

Since we just talked about the importance of starting simple and iterating, let's go back to the simple writing system we built in the last lesson.

As a reminder, here are the major steps in that system:

First, we choose a topic. Then we do some research to get more familiar with the topic and create an outline of the article we want to write. Next we create a draft by filling in that outline with our major points and arguments. And then we edit it and publish it.

That's basically it. Pretty simple.

But, after operating that simple system for awhile, maybe we're a little frustrated that our articles aren't getting the kind of response we want. Or maybe not that many people are seeing them in the first place, and we want to get more people to read them so that our work has a further reach.

Well, those are both new goals that we want to add to our system. And they're also variables that we're not in direct control of. We can't force more people to read our articles, and we can only control what we write, not how people react to it.

So, that means if we want to build a system to improve those metrics and meets those additional goals, we need to add some more steps to the system and make some guesses about what to do based on the results we get at different points of the process as we operate that system.

It's time to upgrade our simple writing system into a complex writing system!

In order to do that, we'll need to make some guesses about what steps to add that will help us achieve these new goals.

Let's start with getting more people to read our articles. "More people" is kind of a vague goal, so let's make it more specific.

Let's say that, if a week after publishing an article at least 500 people haven't read it, then we should do *something* to increase its reach and get it in front of more readers.

What can we do?

One thing we could try is to find other writers who've written about a similar topic and ask them to promote our article to their audience. That should get us some more readers.

So now we have a complex system with some conditional logic that we need to follow whenever we publish an article.

When we get to the end of our simple system and hit that publish button, we need to do a few additional things.

First, we need to wait a week and see what happens. Then, we look at our stats to see how many people read the article. If it's more than 500, we give ourselves a pat on the back and call it a day. We met our goal! But if it's less than 500, we need to do some more research to find other writers who've written about our topic, find their contact information, and ask them to share our article with their readers.

Now, you could build a whole new system with all kinds of logic around just the promotional side of writing, but this is a good example to show how a simple system can morph into a more complex one to get better results.

Let's go a little bit further.

What if our new promotional steps work really well, and get lots of people to read our article, but we notice that the response to the article is a little lackluster. A lot of people leave comments saying "I already knew all this!" or "Nice try, but I don't agree with you!"

We want our writing to be influential and change how people think, so we have some more work to do. We need to add a little more complexity to the system to try to get the results we want.

So, maybe instead of just picking a topic and writing it, we decide to do some research before we start typing away. That research would be a step we would add at the beginning of the writing system and we would use it to figure out what our audience already knows about a topic and what their current opinions about it are.

Once we take those steps, we would have to make a decision about how to proceed based on what we learned. We'd use some conditional logic here to decide what to write and how to frame the topic so that it would be most interesting and compelling to the people who read it.

And maybe, later on, we realize that the way we do our initial research isn't as good as it could be, so we create another little control where we go back a few steps and start over if we don't get useful results. Now the system is no longer linear—it has a loop built into it—but that way, the foundation is really solid before we start writing.

Over time, we've made a bunch of little adjustments to our simple system that works—but not very well—into a complex system that addresses all the problems we ran into as we tried to use it.

It's a demonstration of how systems evolve over time to get better and stronger.

So, what did we learn here and what did we accomplish?

First we learned what criteria qualifies a system as a complex one. Basically, any time one or more of the three criteria for a simple system aren't met, you have a complex system. That's pretty clear.

We also learned why you might want to use a complex system in place of a simple one. Sometimes, what you're trying to accomplish requires multiple goals or needs analysis between steps in order to get really good results. When that's the case, you'd want to have a complex system instead of a simple one.

But we also learned that, whenever possible, you should *start* building your system with the limitations of a simple one. That way you can build it fast and test it to see what improvements are actually needed. It ensures you don't waste time building unnecessary complexity.

We learned that systems are best when they evolve from a simple one into a complex one based on real, demonstrated demand.

And then we worked through an example of how a simple writing system could evolve into a complex one in order to meet new goals that arise as you operate it and see a need for improvement.

But that's just one example and, as we've seen in previous lessons, this evolution can apply to any system in your life.

Go ahead and take a moment today to think about any systems or routines you might have that aren't working as well as you'd like them to. Then, ask yourself where the breakdowns might be happening so that you can get an idea of what improvements you might need to make to start getting the results from it that you really want to see.

And when you're done with that, I'll see you in the next lesson.