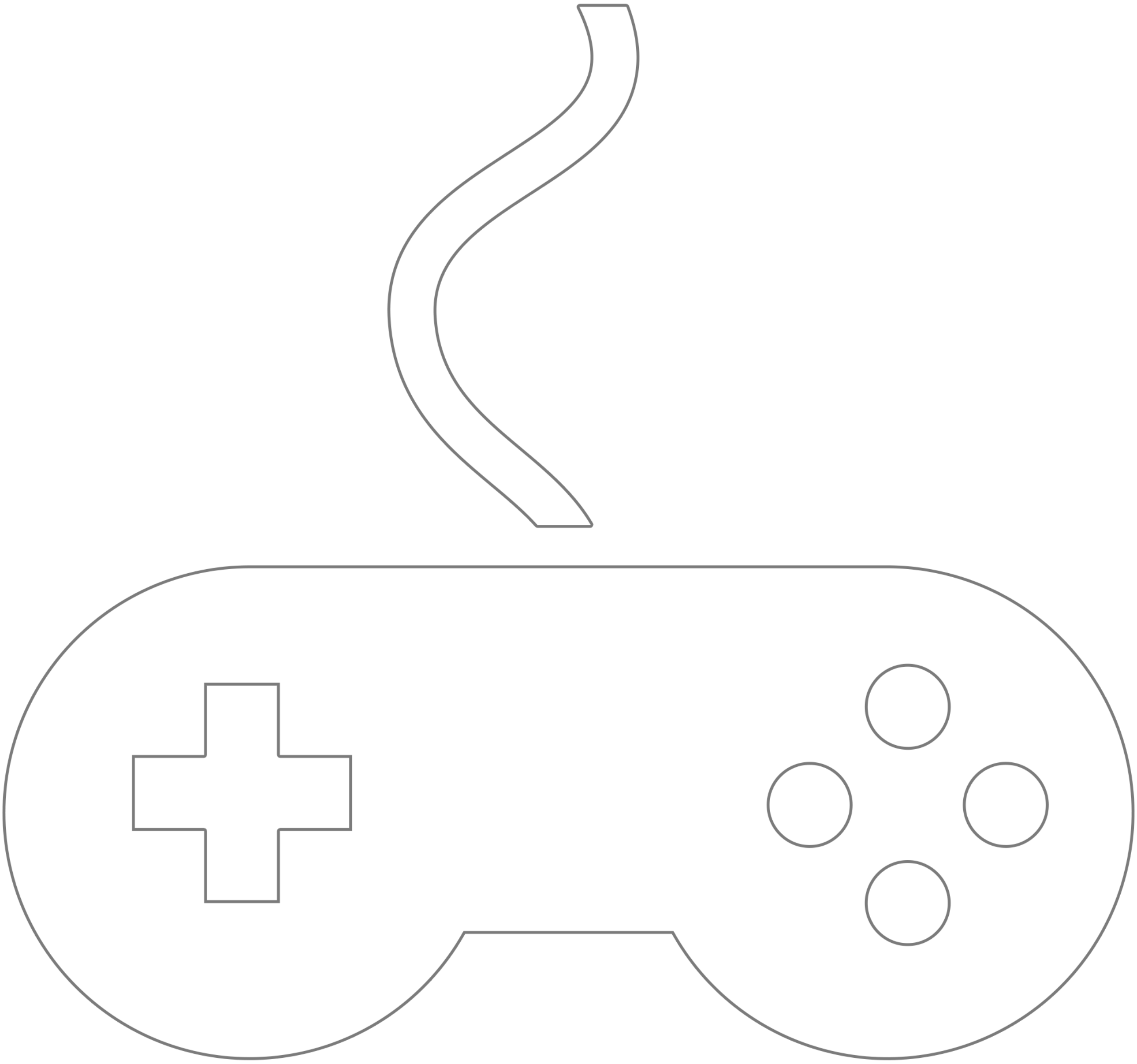


# GAMIFICATION

## But how and why?

Mikkel Lodahl
Lars Tornbjerg

This booklet is developed by

with support from DigitaliseringsBoost

**Ahoot Media**
www.ahoot.dk

**Dania Games**
https://eadania.dk/her-ligger-vi/grenaa/

**Institut for Dansk Spiludvikling**
http://idsu.dk/

**DigitaliseringsBoost**
https://digitaliseringsboost.dk/

April 2021

Magnifying glass icon by Lorc
Cycle icon by Lorc
Uncertainty icon by Lorc
Trophy icon by Lorc
Might force icon by Delapouite
Crafting icon by Delapouite
Retro controller icon by Skoll

# Content

# The purpose of this booklet

If you are interested in gamification, this booklet is for you. You will get:

- a short overview of what the term means
- how it can be useful as a design lens
- what it can be useful for
- a set of game mechanics to start you off doing your own gamification designs
- and even an actionable understanding of what games are and how motivation works

If any of this seems useful to you – read on!

(And when you've read it, you might want to return to some of the sections as you do design sprints or workshops incorporating gamification. It's OK to jump around the text a bit.)

# What is Gamification?

As with many of the terms we use both professionally and colloquially, gamification is not exactly a precise term.

Gaining popularity around 2010, the term springs from the serious or applied game research and development communities. An applied game or a serious game[1] is a game that is either designed or used for something other than entertainment or play. For instance, a game designed to teach mathematics is a serious/applied game, as is the use of a game in the popular, Ubisoft-developed historical action series Assassin's Creed by a teacher to illuminate a historical period.

In the early 2010s gamification consultancy firms were on the rise, promising to use tricks from games to make any process more enjoyable and motivating. Often, these consultancies were not based on research or any particularly systematic approach. They were instead populated by people who had studied game design or developed games themselves. This led to a broad range of quality in the solutions on offer, which brought a sort of crash with it, leaving only those consultancies that had developed a strong methodology alive at the turn of the decade.

Usually, these consultancies – who in practice define what gamification is – have relatively broad portfolios of solutions, incorporating applied and serious games as well as what we would like to call gamification in this little book.

For our purposes here, we want to isolate gamification to mean the following:

Gamification is taking design principles from game design

and applying them to other processes than play.

That *sounds* simple enough. The problem is that if you look up game design principles or ask game designers, you will get answers ranging all the way from "make sure the player makes interesting choices" to a one hour semi-academic talk about the consequences of changing the range of a sniper rifle in a first person shooter from 100 to 110 meters. Not all of the lessons learned by game designers are useful in gamification.

Furthermore, it can be difficult to discern exactly what is being talked about, when game designers talk about their craft, especially if you're not well versed in games yourself, but are still interested in gamification as a possible methodology for improving your designs or business processes.

[1] The terms are in practice interchangeable, although there is a tendency to use serious games in Northern Europe and the US, while applied games seems to be popular in middle and southern Europe.

So before we go further, let's establish some very basic principles of games. For example, it would be good to agree at least somewhat on what a game is. This is a surprisingly contentious question in game design, research and consumer circles. However, we would not be completely amiss if we took up the following definition:

A game is a system that

A) Gives you a range of possible goals
B) Gives you a range of actions to take to reach those goals
C) Gives you continuous feedback on how well you are doing in trying to reach those goals

and *playing* a game is accepting that system, setting goals, performing allowed actions to reach them and listening and reacting to the feedback the system provides. All the while, the player must feel the freedom to either stop playing the game or change the rules.

| **Example: Golf** |
|---|
| The example most people use to illustrate this sort of definition is golf. Golf asks you to get a small ball into a small hole very far away as efficiently as you can. This is, of course, ludicrous, but it only gets more ridiculous when you realise you must get the ball into the hole through a less than ideally efficient way: hitting it with a club. You might have a small golf cart right next to you, but you are not allowed to pick up the ball and drive to the hole, since that action is not one of the actions available to you according to the rules. The par system and you counting your strokes provides the feedback, along with the sickening sound of hitting the ball into a lake or a sand pit[2]. |

So here we have some principles that can be used for quite a lot of processes to optimize them. Limit the possible number of goals the participants have. Make it clear which actions they may choose between to achieve goals. Make sure they understand whether a particular action helped or hindered them.

The elusive part is the sense of freedom. If you talk to a teacher getting involved in using games for the first time, they will inevitably utter the phrase "I want my students to spend as much time on homework as they do on playing Fortnite/Minecraft/[whatever game teachers believe their students or pupils play at the moment]". But the magic sauce in games is not really the mechanics – games are just processes like everything else – but the fact that people freely choose to play them.

This is almost impossible to copy in any other process, because almost any process we want to improve and make more motivating are by their very nature not processes the participants would participate in if they had an absolutely free choice with no ramifications. That is to say that if you want to teach someone something and they lose their motivation to learn it, they should ideally

[2] If you're playing it with me, the curses as I repeatedly try and fail to do a put will also provide you feedback. Mainly that you are winning.

just walk away from it and come back later when it makes more sense for them to do so. This is not an option when you work in a system with lesson plans, exams and yearly budgets, or when you answer to management or investors who are counting on optimised processes to hit quarterly goals.

So what *can* we learn from games if not their ultimate magic? All the little tricks and principles – mechanics, game designers call them – they use to make the process of communicating goals, actions and feedback well so that the player becomes *less likely to want* to get up and do something else.

This is easier said than done, but it is easier to do if you define very clearly, what exactly it is you want to achieve from a game. That is what we will be looking into over the next few pages.

# What can you use Gamification for?

The easiest thing to use gamification for is learning. This does not have to be maths or letters, though. It can be learning the correct sequence for starting a car, the way to use a website, or the possibilities in an application or service.

To understand what you could use gamification for, you must ask yourself new questions about your students, users or customers:

- What should they learn?
- Why should they learn it?
- How long do they need to know it for?
- How often will they need what they learn later?
- How precisely should they know what they are being taught?

There is a world of difference between learning to use the remote control for your television, learning to eat well and healthily, and learning the correct path to a certain hospital wing where you're supposed to be a patient.

**Example: TV remote**

If you are learning how to use the remote control for your television, what you need to know right away is how to turn it on and off, how to turn the volume up and down, how to change the channel. You do not immediately need to learn how you adjust the contrast or download an app to the smart-portion of the beast. This is why remote controls have large, distinct buttons at the top that indicate an action to reach your primary goal – turning the TV on – and immediate feedback on that action in a stand-by light on the TV turning off, way before the TV itself is ready to provide any feedback. The software is slow as software inevitably is. Changing channels and volume are usually assigned large buttons in the middle of the remote and given clear feedback once the system has booted up. These are straightforward processes where clarity in indicating success or failure are fairly simply achieved. You also do them again and again, so they do not need to be re-explained to you, only indicated a few times and then you will be doing them without even thinking. Hence the archaic pastime of channel surfing[3], popular among those of us born before 1995.

**Example: Hospital wing**

Learning the correct way to the hospital wing you're looking for is usually a one-time experience. Just as with operating a TV, there is a clear demarcation between success and

[3] Absent-mindedly changing through flow-TV channels

failure, and the signs, markers and indications can be designed in ways that make it easy to identify goals and take actions towards reaching them. Not all hospital wayfinding systems are designed in that way, though. A common, usually effective way is coloured lines running along the floor keyed to a department. Follow the yellow line for the children's wing, the blue line for surgery, the red line for the maternity ward and so on. Unlike using the remote, though, you will not need to remember which colour to follow – because you can just look it up if you ever need to come back to the same wing another time. Thus, some hospitals actually employ or make use of volunteer guides instead. Since there is no need to teach the way to people, they might as well just have a pleasant walk with a nice chat to maybe alleviate some of the hospital stress. As such, the process of finding the right hospital ward may demand clarity, but not recall, and therefore only momentary learning, not habit, is necessary.

### Example: Eating healthily

Now, the tricky part comes in the third example: eating healthily. People try to do this through dietary plans. These are, of course, designed processes, that clearly indicate goals – a certain amount of nutrition divided among several sources – and actions to take, often in the form of approved recipes. In our data driven times, many dietary plans furthermore establish feedback systems ranging from counting calories to intricate self-monitoring devices. However, at some point you grow tired of the recipes. A successfully designed dietary process would then let you yourself expand your possible actions, while being able to discern whether those actions bring you closer to or further from the goal – essentially developing your own feedback system. Very few dietary programs do this. However, if you run around with a lingering sense of having to watch your cholesterol, needing to cut down on carbs or always only consume light products, you may have internalised one set of dietary principles or others…or at least the feedback system of them. Unlike the remote control, you will not have the comfortable design of the buttons constantly reminding you of the correct process. Unlike finding the correct hospital wing, you will never not need to eat healthily – at least not until you won't need to eat at all. You need the system to last even as you abandon the artefacts the designer has made to communicate the system to you. You need to learn.

These examples show that you as a process designer need to be very aware of what the aims of your process is. Will the users always be interacting with how you deliver the process? Will they read instructions and then be on their own? Will they need to do the process once, twice, many times, every day?

Quite broadly speaking: if you need your users to go through a process once and under your guidance, you should design for an extrinsic motivation to the process. If you want them to go through the process many times and on their own, you should design for intrinsic motivation. Let's next take a look at what those two types of motivation mean.

# Extrinsic >< Intrinsic motivation

The discipline of self-determination theory[4] (SDT) is the most widely referred to theory of motivation when it comes to game design. In short, it divides motivation into extrinsic and intrinsic parts and then spends a great deal of time trying to chart what intrinsic motivation means. This is because for the purposes of SDT, intrinsic is the only way to go[5]. For our purposes, though, both approaches are useful for different things and reasons.

## Extrinsic motivation

Extrinsic motivation means that you are motivated by something external. This can be tangible like a paycheck for an adult or a promised piece of candy for a child. It can also be intangible like an icon of a medal you get for contributing valuable insights to an online forum or making a series of purchases in an online store. Or it can be somewhere in between, like points you earn for playing a game, which allow you to buy digital goods in the game – like the battle pass in Fortnite – or the stamps you get on your value card when you buy a cup of coffee at a specific coffee shop. The tenth one is free!

Lots of things in our society are extrinsically motivated. One thing that unites them all is that they are not things we would do continuously if we did not get some form of reward for it. On the prevailing theory in society, we would not reliably go to work every day if we did not get an external reward in the form of a paycheck. There is a story about the author Franz Kafka that the reason he finished so few works was that no one wanted to publish them. Why do the tedious job of finishing a piece of writing, if you had already written what was exciting to you in the story, unless someone was asking for it?

Another thing common to extrinsically motivated tasks is that while they may be complicated processes, the outcome is usually very simple. A series of forum posts, a purchase, a clear mission in Fortnite you've either done or not – all of these are very simple outcomes. They are Boolean, to borrow a term from logic, which means that they can be true or false. They can have materialised or not. You can't sort of buy a cup of coffee. You do it or you don't.

If what you are trying to teach is simple – not necessarily to do, but has a simple outcome – then extrinsic motivation may be for you. You should look for mechanics that reward users with something external to the process. The result itself cannot be enough of a reward, because then the users would just do it without needing you to motivate them. It must be something where they'll need the extra punch of a reward to get them motivated for the task.

[4] We won't be doing a literature review here, but here's a website, if you want more: http://selfdeterminationtheory.org

[5] There are somewhat complicated reasons for this, but at the heart of it is that SDT-theorists are concerned with positive psychology – the idea that psychology should not focus on curing illness or identifying negative psychological traits, but instead on promoting positive ones.

The criticism of this approach is simple: what happens when you remove the reward? The answer is simple too: people stop doing the process.

| **Example: Passing an exam** |
|---|
| You probably recognise this most keenly if you think back to your days in school. While we can all agree that education is wonderful and useful, we probably also all remember an exam we studied for only to subsequently forget almost everything committed to memory beforehand. That is because the process – learning the knowledge – was motivated by an external reward – passing the exam. Many of us would probably, if we were to re-take our primary school finishing maths exam today, barely understand most of the questions, let alone be able to produce coherent answers. And I dare you to ask an engineer, who doesn't have a personal preference for literature, analyse a poem twenty years after their last exam requiring such a thing. |

# Intrinsic motivation

So if you want to make sure your users will keep using your process once you take away the reward, you have to do something to make the process itself rewarding. You have to make it intrinsically motivating.

Intrinsically, human beings are luckily motivated to learn. But only in certain conditions. In SDT these conditions are modelled as autonomy, mastery and relatedness.

Your users must have a degree of freedom in their process. They must be able to make choices themselves. If you design your process so that they must always just follow a step-by-step instruction, they will not get a sense of autonomy and they will stop paying attention. They might complete the process, sure, and that might be enough if it's a one-time process like signing up for a newsletter, but if you want them to learn how to use a complex digital tool like a platform or a spreadsheet, you have to let them mess around a bit.

However, messing around is only fun as long as you can see you get better and better at something. You have to structure the process so the user can start from nothing and learn to get better as they accomplish things in the process. You must partition the process so a sense of mastery occurs.

| **Example: Super Mario** |
|---|
| In games, the classic example is Super Mario, the game where a short plumber runs from one side of the screen, jumping over holes and on enemies and smashes bricks and uses power-ups and finds secret passages and jumps up flagpoles and, and, and…but the player doesn't do all these things right away. First they jump. Then they jump on an enemy. Then they jump up and smash a brick. Then a power-up appears and moves around to entice the player to catch it. Then a hole appears. And so on. Gradually the player becomes better and better at |

controlling the game. They go from beginner to master while always accomplishing some smaller thing in the process, which each accomplishemtn carrying them further towards their overall goal.

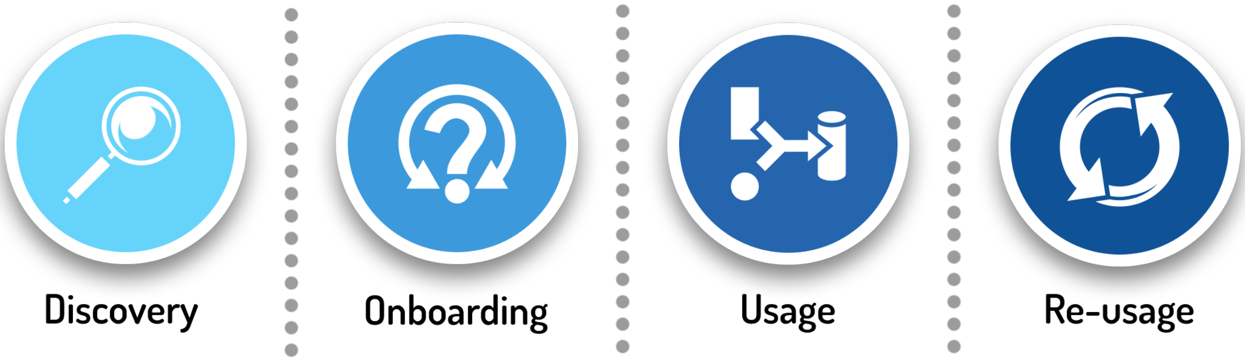
Finally, it should matter that your users are who they are. Your process should be relevant either by a perfect market match – which is, frankly, extremely rare – or by letting the customer influence some part of the process. This is why so many digital apps include things like changing themes or backgrounds. The user must be able to integrate the process and their sense of selves for intrinsic motivation to flourish.

As you might see, intrinsic motivation is connected to a sense of play. If you are trying to teach something that is complex or if the user should *keep running the process* even when your direct design influence has stopped, then you should look for mechanics that work with intrinsic motivation.

So you have asked yourself some questions about what you're trying to tell or teach your user. You have used those questions to start selecting extrinsic or intrinsic motivational mechanics. But there is one more dimension you have to look into. That is the stage of the process you want to gamify.

# Stages of gamified processes

Processes can be divided in many different ways. One rough way of doing so is through phases from the discipline of mapping customer journeys such as: discovery, onboarding, usage, re-usage.



## Discovery

You need something eye-catching to facilitate discovery. Often we default to something visual here, but if you want to think of something mechanical try thinking of describing a play in a football match to someone. You're not going to describe how the players looked, but what they did. You want your product to be described to other people by your user in the same way: here's a great thing you do when you interact with it. However, more often than not this stage is only supported by the product design, not directly a part of it. Other disciplines - marketing, icon design, art direction - rather than game design and gamification are often more useful here.

## Onboarding

Onboarding is the place where extrinsic gamification really shines. You usually need to teach some very simple concepts very quickly to your users so they can get going. Showing their progress through a progress bar or a steadily more advanced animation or an evolving animal avatar is one way of introducing an extrinsic reward for getting into the basics. However, it can also be useful to think about intrinsic motivation if your onboarding involves processes that will be used again and again by the users.

## Usage

When you move into usage, you're working on a continuous relationship. Depending on what your product is used for, you may need an extrinsic or intrinsic motivator here. For example, if you need telemarketers to log lots of calls, you can reward them a little for each call. But if you need your users to feel comfortable while writing and editing a text document, you need to make it enjoyable to actually do the work so they do that again. You can't very well just reward them a small prize for each use of the letter U.

### Re-usage

Once your users move on, they might come back again. If you designed your usage process for extrinsic rewards, you should expect users to come back when they need the extrinsic reward again, such as when they need to make more money and come back to the telemarketing job after losing the other job they left for in the first place. If you design for intrinsic motivation, you might have customers who come back simply because they enjoy interacting with your text editor, or better yet enjoy the way interacting with the editor makes them feel about themselves.

## Immersion

In games, the process of drawing someone into a game is called immersion. This word obviously draws upon the idea of being surrounded by something. We immerse ourselves in water, for example, when we take a bath. Game designers often refer to immersion as having three levels, that might be useful as analogies when combined with the phases from customer journeys mentioned above.

- Engagement
- Engrossment
- Total immersion

Engagement is the basic access[6] to the game. This includes physical – can the player manipulate the controls? – but also mental – can the player understand what is going on?

Think of the Mario example from earlier. You make the controls of Mario easy to understand and use – a controller for the first Mario game had three main buttons. One multi-directional one for basic movement, one for jumping, one for running and shooting fire. This provides easy physical access. The mental access is provided by clues such as Mario being placed all the way to the left in the starting screen making it obvious for the player to try to move right – in the direction of the not visible goal. Once Mario jumps on an enemy, points are awarded, providing mental access to the rules of the game.

Once the player knows what's happening and how they influence it, you build upon this through engrossment. In Mario, you present the player first with a short hole to jump across, then gradually over the course of a level make them jump over larger and larger holes. This makes the player use the skills learned initially to keep building a sense of mastery.

At some point the player stops thinking that Mario is jumping over the holes and thinks instead: "I'm jumping over the hole!" This is the feeling of total immersion. None of the players actually think that they are in the Mushroom Kingdom jumping over holes and collecting flaming flowers that change the colour of their overalls, but somehow they *do* think that. Or feel it.

Again you can see how motivation is used here. Introducing basic concepts such as stomping on enemies in Mario is rewarded extrinsically with points, while getting better and better at jumping

[6] Access here is not in the same sense as accessibility which deals with ensuring differently abled persons can have access to a game, service or place.

over holes, giving the player several enemies to choose from and having the player feel like *they* are jumping, not Mario, brings mastery, autonomy and relatedness. Intrinsic motivation.

This perspective is particularly valuable if you are working with onboarding. Ideally, you want your users to not feel like the time they spend learning how to use your service is wasted. If you have them make something they can use once they are fully using the app – such as entering their own data and not data you make up for the onboarding exercises – it feels more organic. This combines the extrinsic motivation of learning a useful new skill with the intrinsic motivation of already using it for solving problems from the get go. This resembles how it's fun both to jump in Mario and learning how to jump farther and farther.

In light of all this, you should have a pretty clear idea of what sort of process design you need to do for different parts of your product and different sorts of users.

But how do you know which mechanics to use to further certain design choices? Let's take a look at some mechanics we have collected together as inspiration for you, and how you can use them in your design process.

# Mechanics

Here they are, the *mechanics* we have been talking so much about. Not all of them are exclusively seen in games nor do they all originate from games. As game design has evolved and constructed a canon of its own, some mechanics have been taken from other disciplines, such as psychology or marketing and been adapted to games.

If you speak to any given game designer, you will surely be able to learn of a host of different mechanics than the ones we have chosen here. Think of this as a starter pack, a good beginning. If you get familiar with these, you will have a good grounding and may even start to notice other game mechanics as you observe games out in the wild.

Without further ado, here is a non-exhaustive list of game mechanics you can use in your process:

| **Game mechanics** | | | |
|---|---|---|---|
| Flow | Tutorial | Progress bars | Larger narrative |
| Game feel | Customization | Achievement badges | In-game economy |
| Scarcity | Loss aversion | Triggering | Micro/macro/meta |
| Traces | Endowment effect | Competition | Empowered progress |
| Collaboration | Quest lines | Pliability | |

Remember that not all mechanics are suited for all kinds of processes, as each process is unique, you'll need to evaluate if the mechanic achieves the desired outcome. Sometimes they might even cause the opposite of what you intended.

For example, scarcity is the principle that you can direct what players want and do by limiting their access to a resource. This can be a resource in the game, such as certain powers or monies, or it can be something outside the game such as time. One good example of this is the very popular mobile game Wordfeud, where you compete with others in a game of Scrabble, setting the longest and most complex words on a board. Each player has 72 hours to make each move. If they fail to act within 72 hours, they lose the game. In theory, this keeps players coming back again and again, ensuring that the Wordfeud developers have an audience for the ads their revenue depends on. But it can also have the opposite effect - if the player loses too many times from timing out, they stop being interested in the game at all, concluding that the mechanic simply showed them that they do not have space for Wordfeud in their lives anyway.

Furthermore, it's not about adding as many different mechanics as possible, but adding a few and doing these really well. Having too many different mechanics will make them difficult to relate to for the users, and they might lessen the impact or even cancel each other out. This is also true if you over-use a single mechanic.

The game Assassin's Creed II follows the exploits of an assassin climbing over the rooftops of various Italian renaissance cities to plan and carry out hits on various historical figures. For the game, the developers constructed several very large, open areas modeled on Florence, Venice, Forli and part of Tuscany. This took considerable effort, but a player might only see a fraction of the game area through following the narrative, so to drive players to see more, a range of achievements were built into the game, requiring the player to explore more parts of the map. In theory, this is a good idea. In practice, two different design decisions lessened the impact. One was including too many achievements for mundane interactions. One of these occurs right as the character is being born, where the player must press the different buttons on the controller to make the newborn baby she is controlling move to indicate life. This gets you an achievement, just for pushing three buttons. Later, the player is charged with finding 100 small feathers spread throughout the game, necessitating thorough exploration of almost the entire game world. But who does that? The inclusion of too many and too demanding achievements lessen the impact of the achievements as a mechanic to control player behaviour.

## Extrinsic >< Intrinsic mechanics

A mechanic will typically either motivate extrinsically or intrinsically. And sometimes it's about how you implement the mechanics that decides if it's one or the other.

Whether you want to appeal to extrinsic or intrinsic motivation is tied both to the types of mechanics you want or are able to implement, as well as to what process (take a look at the section on the two motivations earlier in the booklet to learn more about this.

Sometimes, though, it's good to have a mix of both. Take a good look at the process you're trying to make more motivating and ask yourself if one type of motivation is dominating the other. Then see if you can add some of the opposite mechanics to balance it out. We've taken the list of mechanics above and sorted them for you in the motivations they are typically best suited to be used to appeal to:

| Extrinsic | Intrinsic |
|---|---|
| Progress bars | Flow |
| Larger narrative | Tutorial |
| Empowered progress | Game feel |
| Achievement badges | Customization |
| Questlines | Collaboration |
| Scarcity | Traces |
| Loss aversion | Endowment effect |

| | |
|---|---|
| Triggering | Pliability |
| Micro/macro/meta | |
| Competition | |

# Most appropriate stages

All mechanics can be used in any stage of the gamified proces, but you might find certain mechanics more appropriate or powerful in certain stages. Below, we've divided the mechanics into the different stages, skipping over discovery, since that stage is much more ripe for other kinds of design. Don't feel too bound by our sorting, though. Using the same mechanics across multiple stages can often be used to bridge the gap between the stages, and can increase the sense of mastery felt by the user.

| **Onboarding** | |
|---|---|
| **Extrinsic motivation** | **Intrinsic motivation** |
| Progress bars | Flow |
| Larger narrative | Tutorial |
| Empowered progress | |

| **Usage** | |
|---|---|
| **Extrinsic motivation** | **Intrinsic motivation** |
| Achievement badges | Game feel |
| Questlines | Customization |
| Scarcity | Micro/macro/meta |
| Loss aversion | Pliability |
| Triggering | |
| | |

| Re-usage | |
|---|---|
| **Extrinsic motivation** | **Intrinsic motivation** |
| Competition | Collaboration |
| | Traces |
| | Endowment effect |

Once you set about designing processes using these gamification mechanics, you may find that you are overwhelmed. Even once the mechanics are sorted into different motivations and stages, it may be helpful to print the mechanics out on cards or large pieces of paper to work with the mechanics physically during your design workshops. You may also be wondering right now, why we haven’t described most of the mechanics at all.

Well, the answer to both of these concerns are to be found in the set of printable cards we have published alongside this booklet. They can be found in a handy pdf at the IDSU webpage (IDSU.dk) or in non-editable Google Slide form, courtesy of Ahoot, here: Gamification cards

Pick out the mechanics from the deck that you want to work with, either because of the stage or motivation they are related to, shuffle them about and draw them out at random to direct your design process. Or hang them up on a large whiteboard and start brainstorming, drawing mindmaps and scribbling notes alongside them. The important this is for you to start talking about the mechanics, about which motivations they work with and how they can be used to enhance your particular process or product.

# Conclusion and further reading

Here we are, at the end of this booklet.

Maybe you read all the way here, maybe you skipped to the end to see if there was a butler lurking around back here. If so, we must disappoint you.

However, we'd like to take the time to first remind you of the questions we wrote about above to determine which sort of gamification is needed to spice up a particular process:

- What should they learn?
- Why should they learn it?
- How long do they need to know it for?
- How often will they need what they learn later?
- How precisely should they know what they are being taught?

But you know, these are not the only questions you could ask yourself. You could also ask questions such as:

- What games do my audience play?
- What mechanics are in those games?
- Can any of those mechanics be useful in my own design, making my audience already feel familiar with my product from the get go?

Or you could ask yourself questions about the mechanics:

- Do we already have processes that seem to use some of these mechanics without intending to?
- Are any of the mechanics particularly interesting to me?
- Are any of the mechanics I'm considering in opposition to each other? Why and how?

Whatever you do, you should know that reading this booklet is not enough. We have done our best to give you a clear understanding of the challenges and opportunities in the field of gamification.

But any game designer could tell you that knowing is less than half the battle. You don't really know what works until you've tried it. And you may not know what works next time, until you've tried it again. Fun is unpredictable, and game design is only a range of techniques of making fun a little more predictable.

The same is true for motivation and gamification - and you need to get out there and do the work to figure out how exactly your process can match with which gamification techniques.

Good luck!

# Further reading

Though, maybe reading more *is* the next step for you? Below, we have made a short, curated list of further reading below, if you would like to dig deeper into the world of gamification. There is a lot of material out there, some of it better than other, but the following are not bad places to go from here:

Andreas Lieberoth's article Shallow Gamification:
https://journals.sagepub.com/doi/10.1177/1555412014559978

Jane McGonigal's book Reality is Broken:
https://www.saxo.com/dk/reality-is-broken_jane-mcgonigal_paperback_9780099540281

Yu-kai Chou's Octalysis framework:
https://yukaichou.com/gamification-examples/octalysis-complete-gamification-framework/

Andrzej Marczewski's book Even Ninja Monkeys Like To Play:
https://www.goodreads.com/book/show/27411927-even-ninja-monkeys-like-to-play

Elisa Mekler's article on pro-social game mechanics:
https://www.researchgate.net/publication/326663352_Does_a_prosocial_decision_in_video_games_lead_to_increased_prosocial_real-life_behavior_The_impact_of_reward_and_reasoning

Lee Sheldon's book The Multiplayer Classroom:
https://www.saxo.com/dk/multiplayer-classroom_lee-sheldon_pdf_9781000039009

James Paul Gee's book Good Video Games and Good Learning:
https://www.saxo.com/dk/good-video-games-and-good-learning_james-paul-gee_paperback_9781433123931