5. Exploration of the Formative Learning Design Process

Research Questions Nov 2024

```
1. What contradictions arose in participation in this research’s game
coding processes and how were they addressed?
2. How can game design patterns support the development of computational
fluency in novices?
3. How can learners build agency in an evolving community of game makers?
```

Additional information to help reviewers

I am including some brief extracts and diagrams from Chapter 4 - Methodology to help clarify this chapter.

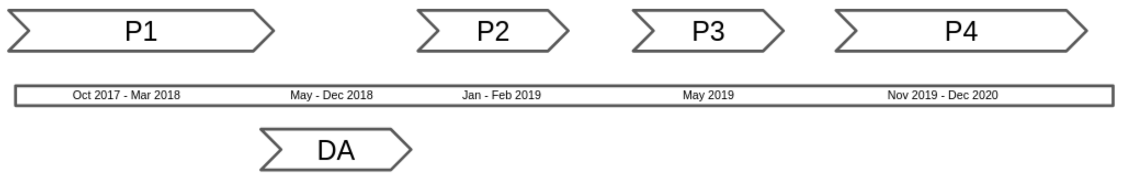


Fig. 4.x Summary of delivery phases and development periods

This chapter contains the names of figures, tables and appendices headings in draft form. These will be replaced with numerical format when the editing process is more complete. This chapter makes extensive references to appendices. While these are still in progress, I will forward a work in progress of the collection of vignettes in particular before the week of the review meeting.

Chapter Introduction

This chapter uses the theoretical concepts explored in Chapter 3 to analyse and describe the transformations between the phases of research outlined in Chapter 4. To do this, I draw upon and adapt a technique from the design-based research community known as *design narrative* (Hoadley, 2002). A design narrative is effective in communicating important design and contextual details, helping to situate the findings of this thesis in a way that allows the design to be understood or replicated by other practitioners and researchers (Hoadley, 2002; Bell, 2004; Brase, 2024). In line with the discussion in Chapter 3, this process is augmented by incorporating an argumentative grammar of formative interventions within the Helsinki School’s interpretation of third-generation activity theory (3GAT).

In the first section, this is achieved by identifying and analysing activity systems at different levels of granularity, specifically examining their system elements and providing a detailed interpretation of the objects of activity at each scope. To ground the reader in concrete detail, this process is illustrated using the context of a vignette included in Appendix 5.toby. Three sections follow, containing in-depth analysis of key areas of contradictions in the emergent activity. The three sections describe different facets of these contradictions and interventions to address them. The first, tackles key transformations in use of primary tools, e.g. those used to directly achieve the task in phase one (P1) and phase two (P2). The second addresses contradictions associated with project navigation and the use help resources and documentation. Finally, other support process to support emerging social coding processes are examined to explore contradictions surrounding participant identity within the game-making activity. Each section advances the design narrative by drawing on technical and social observations from my research journal, additional session notes, and analysis of the created artefacts, including games and documentation. The discussion section of this chapter begins synthesising concrete issues arising in the evolution of the learning design using 3GAT and DBR concepts, specifically: an iterative and mutual design process; double stimulation and its relation to participant agency; and the process of rising to the concrete through the exploration of a germ cell concept in the form of the use of gameplay design patterns.

Vignette and broad analysis using an analysis of activity systems

This section begins analysis of the learning design intervention via the exploration of two scopes of activity system using an illustrative vignette (included as Appendix 5.toby). The broader of the two activity systems explored in this section is represented in two figures below, firstly as a joint activity stemming from the intersection of wider activity systems in Fig 5.x, and secondly as an activity in its own right in Fig 5.x

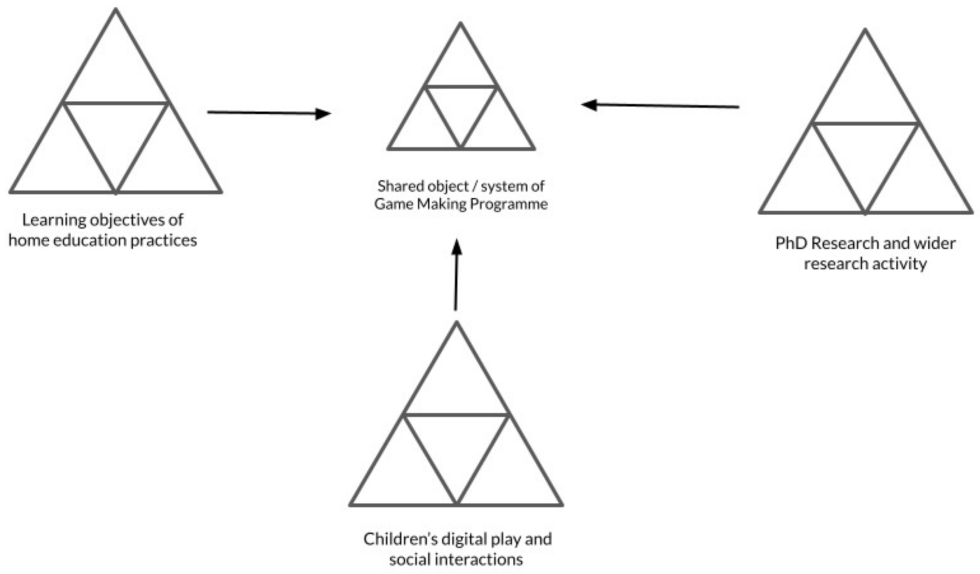

Learning objectives of home education practices
Shared object / system of Game Making Programme
PhD Research and wider research activity
Children's digital play and social interactions

Figure 5.broad - Broad environmental activity systems

The figure above is necessary simplification of the sources of diverse cultural and contextual factors that feed into the new shared activity system. Examples of the influence of those wider systems in the game making activity including playful interactions, the supportive practices of parents, and guidance in practical coding practices from facilitators are explored in Chapters 6 and 7. Turning to the activity system represented below in fig 5.large, the scope of activity here takes as a subject the community of individuals in the room during the game making sessions, namely parents, children, student helpers and myself as a researcher / facilitator.

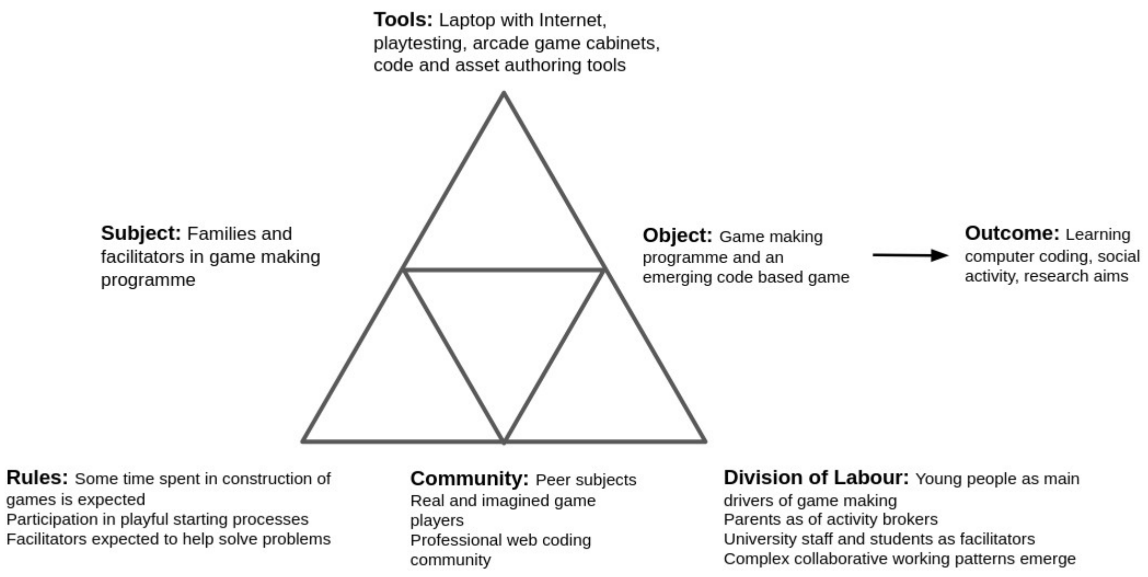

Tools: Laptop with Internet, playtesting, arcade game cabinets, code and asset authoring tools
Subject: Families and facilitators in game making programme
Object: Game making programme and an emerging code based game
Outcome: Learning computer coding, social activity, research aims
Rules: Some time spent in construction of games is expected
Participation in playful starting processes
Facilitators expected to help solve problems
Community: Peer subjects
Real and imagined game players
Professional web coding community
Division of Labour: Young people as main drivers of game making
Parents as of activity brokers
University staff and students as facilitators
Complex collaborative working patterns emerge

Illustration 5.x - Larger game making activity system

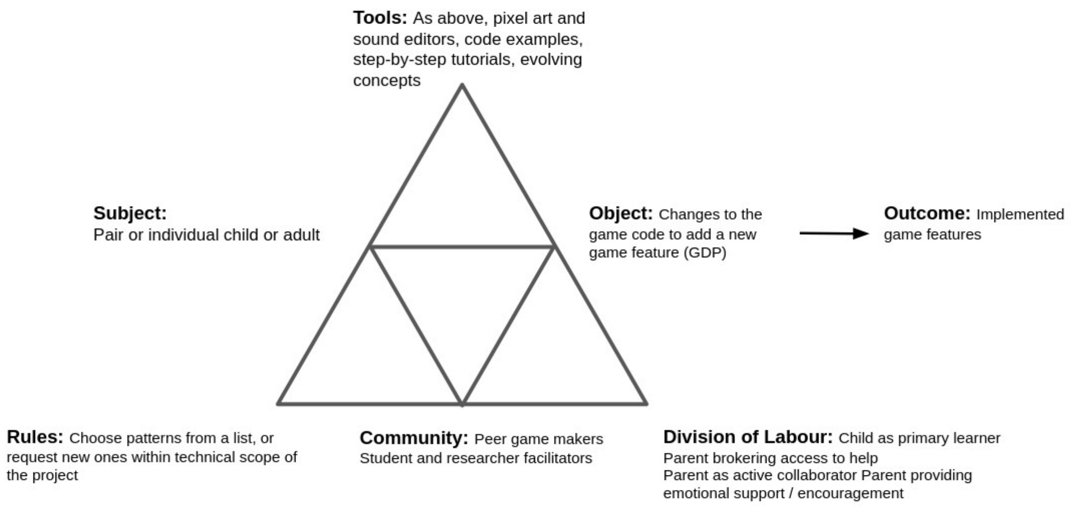
Fig 5.large : Activity system 1 - an emerging community of novice game makers community making games as a learning experience

To situate a description of this system, I draw on illustrative extracts from vignette 5.toby. Before the vignette begins, Toby, a child participant, has been working independently on his emerging platform game for the first five minutes of the session. He is seated next to his two grandparents who are modifying their own games. At the start of the vignette, in my role as session facilitator, I make a brief opening announcement, drawing the participants' attention to the upcoming showcasing of their games to students in the building's foyer. This announcement serves to focus their attention on the completion of their games. The imagined audience of players seems to contribute to Toby's careful attention to the challenge and variety of the game-playing experience, particularly in the specifics of his level design, which is documented in the screenshots and descriptions provided in the vignette. It becomes clear that the more immediate audience of his peers plays an important role in guiding his design decisions. During the session, Toby invites other group members to play his game, initiating and responding to conversations about the difficulty of his game design

Figure.carrot - Final showcase activity in Brooks Building with home made cabinet surrounding a laptop computer.

While my session announcement broadly frames the object of activity as participation in a community endeavour to create a game for an external audience, a 3GAT interpretation reveals several interrelated facets of the object. The object encompasses both digital and physical aspects. Specifically, it includes the use of varied software tools to develop code and game assets, the use of physical computers for coding, and the eventual incorporation of self-made arcade cabinets during the showcase stage. At the community level of activity, the object also includes the complex interaction of motivations driving all members of the community. These motivations, particularly playful and social interactions, became quickly tangible and evident as the session progressed. Participants' engagement with each other, as well as with the task itself, played a significant role in shaping their individual and collective experiences. In addition, during Phase 1 (P1), the object of activity for the community had to expand to include a focus on the evolution of a group working process. This expansion addressed tensions in the fledgling design, highlighting the pedagogical considerations and adjustments necessary to support participants in the process of game development. These collaborative pedagogical advances and how they contributed to the joint understanding of the joint object of activity are explored later in this chapter.

By P2, at the point when this vignette occurs, the implementation of game design patterns had become an important organising principle for myself and participants. Toby's activity in this vignette could be broken into two main actions: one being to alter the existing level design of the game by changing the placement of game elements, the second to add a new game design pattern to his game by adding a new code structure to the starting project. Following Barab and colleagues (Barab et al., 2002), who justify the analysis of smaller activity systems in their study of technology-rich learning environments, I analyse the implementation of these game design patterns as activity systems in their own right, rather than as mere sub-actions within the broader activity. At this smaller-scale activity system, the subject is either an individual or a parent/child pairing, with the driving motivation being to modify specific features within their individual games, as represented in Fig. 5.feature below. One benefit of analysing at this smaller scope is that it allows for a detailed examination of contradictions in the mediating processes involving tool use by individual participants or pairs. These contradictions can lead to expansions in the object of activity, which increases the complexity of the object at the broader community activity scope. This shift moves beyond technical perspective to encompass the diverse cultural elements of the emerging idioculture within the group. The vignette provides evidence of this evolution, demonstrating how variations in the games created correspond with shifts in social patterns of interaction. By analysing these game design pattern implementations as distinct activity systems, it is possible to trace how the changes made at the individual level feed back into the broader community practices, influencing both the technical and social aspects of the learning environment.


Tools: As above, pixel art and sound editors, code examples, step-by-step tutorials, evolving concepts
Subject: Pair or individual child or adult
Object: Changes to the game code to add a new game feature (GDP)
Outcome: Implemented game features
Rules: Choose patterns from a list, or request new ones within technical scope of the project
Community: Peer game makers Student and researcher facilitators
Division of Labour: Child as primary learner Parent brokering access to help Parent as active collaborator Parent providing emotional support / encouragement

Mid-level activity system of game making

Fig 5.feature Activity system 2 - Constructing a game feature by feature

The significance of activity at this scope is reflected in my proposal of the implementation of GDPs as a germ cell concept within the theoretical framework of this thesis. A more granular exploration of this scope of activity helps explore both the developmental and analytical aspects of the germ cell concept. The object of activity at this level is less complex, as the primary motivation is to implement or alter a design pattern in the participant's game on a pair or individual level and therefore incorporates less cultural and social factors. In the course of the vignette, Toby uses a variety of tools to achieve this goal: a web-based coding tool (code playground), a game template that serves as the base for his game project, and a menu of documentation linking to code examples and tutorials. Implementing more complicated gameplay design patterns (GDPs) involved several stages and varied tools, which can be viewed as sub-actions contributing to the broader goal of applying a GDP. Using the terminology of Leontiev (2009), Toby can be seen undertaking chains of processes in a fluid way, indicating that these actions had become operations. The commentary in Vignette 5.toby provides a more detailed description of this process.

Narrative exploration of key contradictions emerging in the game-making learning design

The following sections of this chapter examine three main areas of contradictions within the framework of 3GAT. The first area of contradiction addresses the novel and technical aspects of the learning design, focusing on the shifts in software tool use between P1 and P2. The second section investigates issues related to project navigation and the use of supporting documentation in P2. The third area explores contradictions concerning participant identity, a topic highlighted in research reviewed in the first two chapters as presenting significant barriers to learning to code.

C1 - contradictions involving the use of game programming and asset authoring tools

In this section, I draw on journal notes to describe emerging tensions in kickstarting participant game design activity and the subsequent evolution in tool use between P1 and P2. My initial focus in the early stages was to create a welcoming, low-pressure environment for introducing and exploring the process of making games. To achieve this, I used several activities unrelated to computer coding to scaffold the game design process. Early sessions included playing and discussing retro arcade games, analysing their components, brainstorming game story scenarios, creating pixel-art characters on paper, and making craft collages for game backgrounds. I also provided learners with a laptop they could take home loaded with vintage arcade and console games to help build familiarity with arcade and retro game tropes through home play. Scenes and characters began to take shape during these relatively unstructured planning sessions, with three, mixed age groups of roughly five participants forming. We then discussed, in simple terms, the game features they wanted to include. An example would be to add a bee as the main character, for it to be able to fly around the screen, and for it to collecting honey. Coding tools and processes were introduced in week five, a delay driven by my concern about the potentially alienating effect of engaging with the unfamiliar experience of computer code. My own motivation to build on prior research into novice use of authentic web text languages, such as JavaScript (Chesterman, 2015), rather than simplified block-based code, likely heightened this concern. Four weeks into P1, I drew on documentation to create and introduce a structural JavaScript game making template. For a fuller technical breakdown of the tools used in each phase, see Appendix 5.tech.

As anticipated, the introduction of textual programming with JavaScript into the game-making process contributed to a range of accumulating tensions. These are initially described in broad terms before focusing on those related to tool use. The gap between participants' desired game features and their inability to implement them in code led groups to specialise. Some participants with greater confidence or experience took on coding tasks, while others worked on graphical game assets (both digital and physical), narrative planning, or sound elements. At times, participant expectations exceeded what was realistically achievable within the scope of the game-making project, a tension explored further in Appendix 5.bee. While it was clear that participants needed to develop their coding abilities, the emerging, informal, experimental nature of the learning environment seemed to me to not suit whole-class instruction. This intuition was supported by my reflections on two attempts to deliver whole group presentations of key coding concepts, which I judged in my journal, based on visual cues of engagement, to be unwelcome interruptions to participants' making activities. Instead, I addressed problems on an individual basis. However, as more issues arose, I became overwhelmed by the group dynamics and technical demands on my time. After one session, I emailed participants (see Appendix.narrative2), expressing that I felt daunted by the task of facilitating the integration of the disparate creative elements being produced into a cohesive project. I asked parents for ideas and support in organising and bringing more order to group and planning processes. These developing tensions can be framed as contradictions between the tools available to participants and the object of actualising their ideas and graphical assets in code form. Additionally, my limited time as a facilitator highlights a dysfunctional relationship between the division of labour and the object.

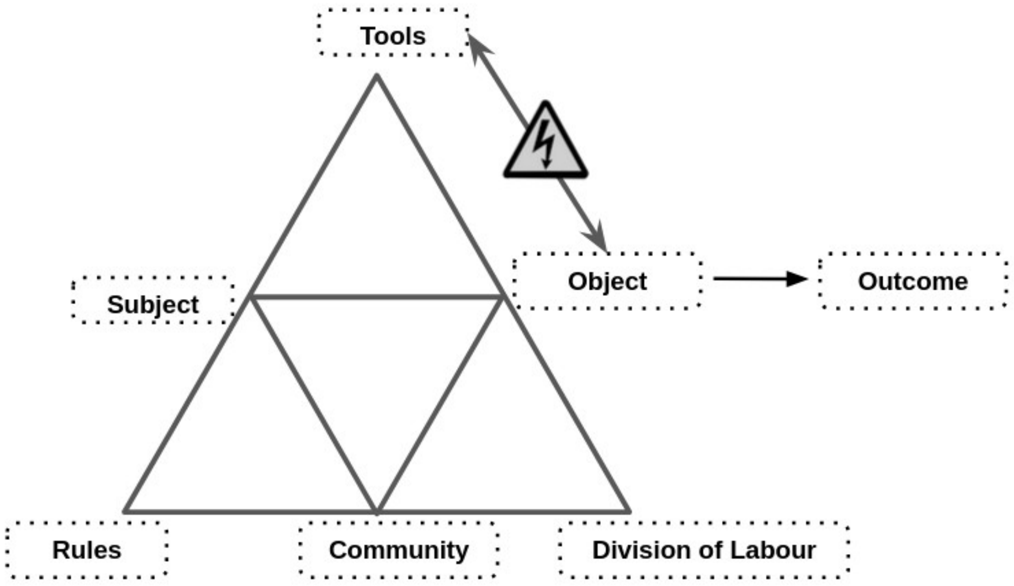


Figure 5.C1: Representation of emerging areas of contradictions in mid-P1 game making process

In response to my email, parents made suggestions including: use of a visible and shared list of game features that are being worked on, documentation to support the implementation their requirements, and more hands on use of tools before beginning the planning process. The group planning process improved, and the self-organisational abilities of parents and tenacity of young people involved developed. Groups expanded their activity by creating wish lists of game features, which acted as mediating tools to coordinate between different team members and to request facilitator support to implement them via code structures. In addition to helping directly via coding with participants, I began to create short stand-alone instruction documents if I thought the requested code would be useful to other groups. Despite these changes which helped to address the area of contradiction outlined above, by this point so many tools and processes and had been introduced, complicating the object of activity at the group process level to such an extent that participants were disorientated and even overwhelmed (see Appendix 5.P1.feedback & Appendix 5.bee.).

On completion of P1, I reflected on the design challenge involved to address the areas of contradiction. The overall learning design should maintain positive motivation towards to the object being worked on and facilitate hands-on coding using authentic tools early in the process to avoid unrealistic expectations. An initial coding activity should help novices build code familiarity without needing explicit instruction. Supporting documentation was needed but should not interfere with the flow of participants game making experience. Finally, the overall learning design should encourage appropriate social learning, peer support and reflective practices but without overly complicating the object of activity or interrupting participant making.

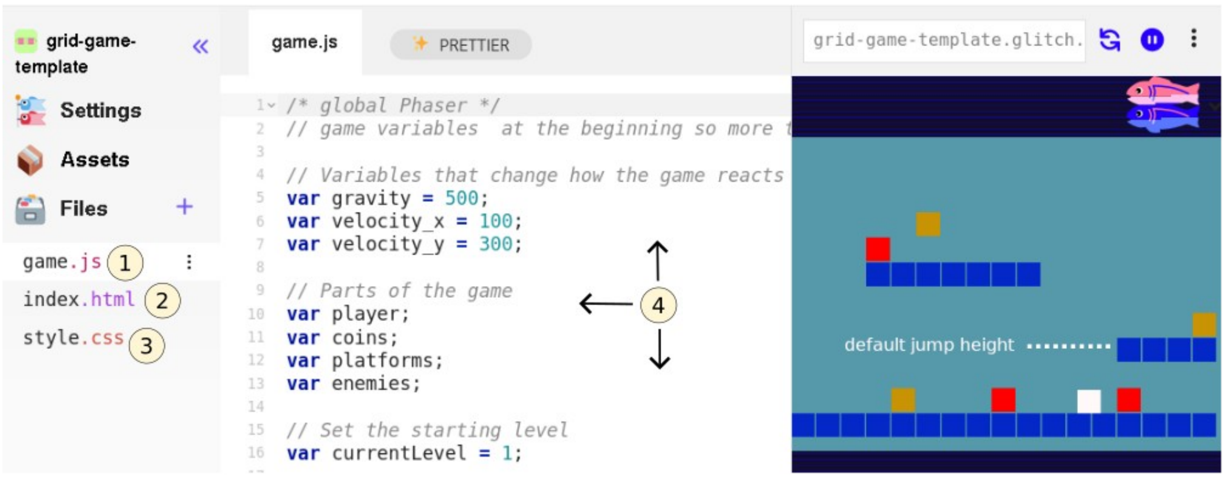
Using a starter game template within a code playground

The use of a code playground and starting template, while technically two separate tools, were experienced by participants as a holistic experience. Code playgrounds, as described in Chapter 2, are online environments used to test, share or invite help from online users on complete or partial code projects or problems, primarily for web-based project involving the technologies of HTML, CSS and variations of JavaScript (for more detail see Appendix 5.tech). The use of code playgrounds by novices can mitigate against some of the initial challenges provided by learning computer coding stemming from unfamiliarity with and potential complexity of code authoring tools and environments (Guzdial, 2004). Tactics to address challenging elements include steps to reduce syntax errors, shielding complexity, facilitating community commenting, sharing, remixing and other forms of collaboration.

In P1, after delays caused by lack of coding knowledge, I had guided participants to begin their games by using a very partial template based on an online tutorial. As the process continued, it became apparent that the starting template greatly shaped the following design possibilities. To address this, in the development period between P1 and P2, I redesigned the starting code template of a game with a greater attention to pedagogical concerns in the following ways: I made changes to the code to reduce obscure syntax where possible increasing code readability and simplifying the structure of inter-related functions to facilitate the process of adding new code structures and code snippets. The choice to pre-select a particular genre, specifically a *platformer* game (see glossary), was a pragmatic response to reducing the tension caused by diverse help requests, which narrowed the range of game features that would be requested. A summary of how these concerns were implemented in the design process follows (see Appendix 5.tech. for a fuller description).

To encourage playful, hands-on experimentation in the early stages, I opted to use a working game template rather than a purely structural approach through the application of a pedagogical technique used in a previous study (Chesterman, 2015), which involved highlighting easily alterable code changes that had a significant impact on the final project. Participants were introduced to the starting template game by playing it, but they were unable to jump onto the first platform. To progress, they needed to alter at least one of the key variables highlighted at the very beginning of the game code (see Figure 5.code below). The underlying game structure initially limited user choice, guiding a shared experience for all participants and facilitating peer learning. Kynigos and colleagues (2018) describe this concept as *half-baked games*, where incompleteness or bugs in behaviour provoke participants to correct or modify them. This approach aligns with the motivations and techniques of the UMC framework explored in the literature review (Lee et al., 2011), particularly the guideline to “create choices that show visible and immediate changes” (Lytle et al., 2019:6). Forcing a rapid transition from the *use* to *modify* stages at an early stage allowed for careful scaffolding of initial coding experiences, aimed at overcoming negative affect towards text-based coding. This alignment with UMC and half-baked design approaches also informed the selection of other key affordances, implemented through simple code changes, which are explored after a summary of the structural design of the game template.

1. javascript file
2. html file



```
/* global Phaser */
// game variables  at the beginning so more t

// Variables that change how the game reacts
var gravity = 500;
var velocity_x = 100;
var velocity_y = 300;

// Parts of the game
var player;
var coins;
var platforms;
var enemies;

// Set the starting level
var currentLevel = 1;
```

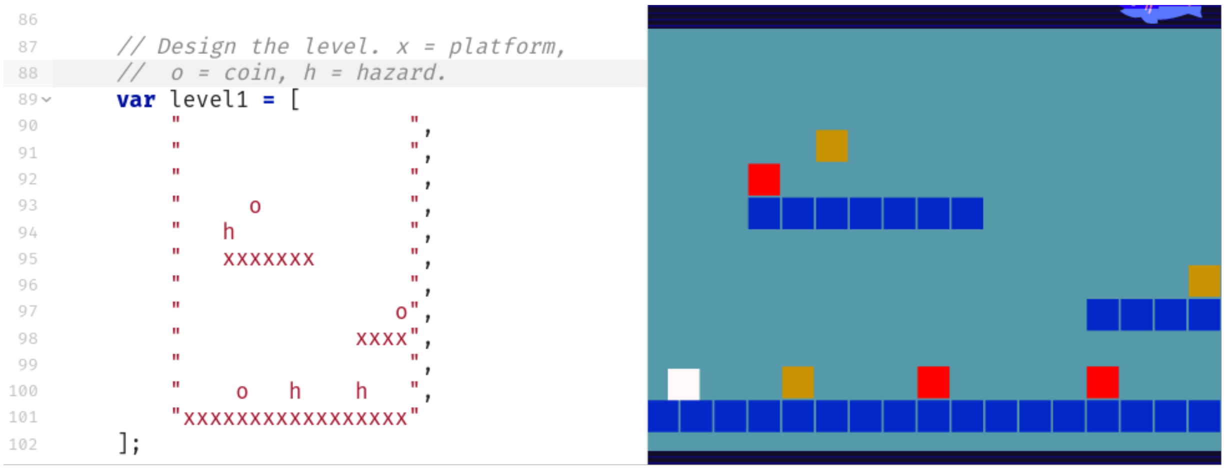
simple game template

Figure 5.code : Summary of the features of the glitch code playground environment and game template - APOLOGIES THIS IS INCOMPLETE CURRENTLY

1. Javascript file which participants alter to make changes to their game
2. Html page within which the game is embeded (not usually altered by participants)
3. Css style sheet (not usually altered by participants)
4. Code editing area
5. Game preview area

The project consists of several interlinked code and asset files (see glossary & Appendix 5.tech). The Glitch.com code editing tool had three main areas. First, on the left, there were options to change project settings, an assets link to manage image and audio files, and a list of project files, including JavaScript, HTML, and CSS. Second, in the central code window, participants could view and edit the code and comments. Third, on the right, a game preview displayed the immediate updates resulting from changes made to the code in the central window.

Beyond the changing of movement variables, two key motivational affordances of the design were the ability to change the platformer level design and replacement of the deliberately simplistic colour block graphics with characters created by participants in the form of pixel art. Addressing level design first, in P1 the process was relatively complex involving changing parameters of functions to change asset location, and spiralling code complexity (see Appendix 5.chapter). Instead, to align with research on the value of a visual approach to coding multi-media projects for novices (Guzdial, 2004; Resnick et al., 2009), in the P2 starting template the use of a graphical grid structure to edit level design shown in Fig 5.grid. A minimal choice of level design elements were represented specifically; platforms to be jumped on; hazards to be avoided; and rewards to be collected.



```
        // Design the level. x = platform,
        //  o = coin, h = hazard.
        var level1 = [
            "                 ",
            "                 ",
            "                 ",
            "     o           ",
            "   h             ",
            "   xxxxxxx       ",
            "                 ",
            "                o",
            "             xxxx",
            "                 ",
            "    o   h    h   ",
            "xxxxxxxxxxxxxxxxx"
        ];
```

Figure 5.grid - Grid based editing of level design with a simple key for hazards, coins, and platforms.

Technically, each level is a JavaScript object consisting of a data array of 12 entries containing 17 characters which representing a matrix layout of the game. Each grid entry can be either black or one of the following: x (platform); h (hazard); o (coin). The structure of text-based array has a strong visual correlation with the resulting game layout and changes to the text based grid in the code area on the left would be immediately seen in the right hand project preview area. This solution abstracts away complexity and repetitive nature of asset placement mirroring a technique called tilemaps (Erhard-Olsson, 2018) used in GUI oriented game making tools (see glossary and Appendix.tech). While my design aim was in part to reduce coding syntax errors and thus reduce learner anxiety, I did not wish not to remove the possibility of learners making mistakes entirely. While even these small changes involved potential syntax errors, as these simple expressions were surrounded by other lines modelling the correct syntax, they could often be corrected by the participants without facilitator support. This chapter's vignette (Appendix 5.toby), where Toby designs many levels as a way of making his game distinct from others, illustrates the impact of this technical adaptation on the evolution of social coding processes a theme which is explored in more detail in a following section.

Turning to the use of graphical assets, the starting template was altered to facilitate and encourage the process of adding designs created by participants. Initial graphical assets consisted of colour blocks, a design choice inviting learners to develop game characters from a clean slate (see Figure 5.px). To help resolve the overly complicated use of multiple asset creation tools, I prioritised the use of the pixel art tool Piskel, as I evaluated it to be intuitive for many younger participants. In P2, participants were guided to make a game on a broadly environmental theme, participants often redesigned sprites to games involving animals. Figure 5.fish shows a whale as a player character and plastic bottles as a hazard and fish as an item to collect. The process of game art and audio creation opportunities seeding narrative and artistic creativity is explored in more detail in Chapter 5.

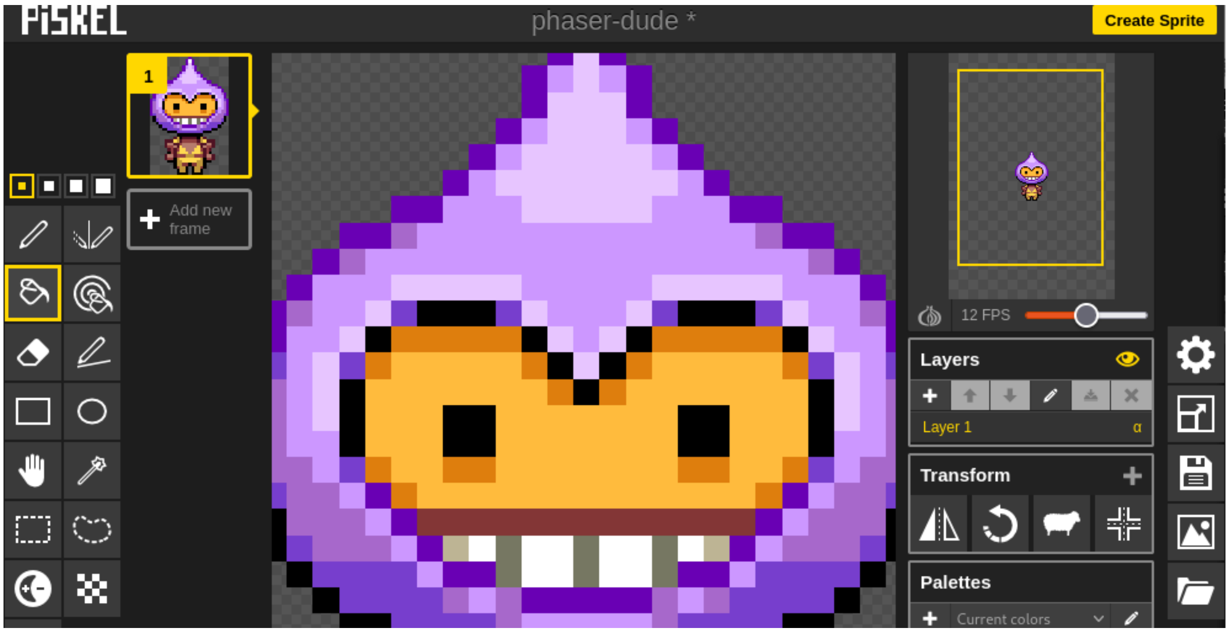


Figure 5.px - Interface of Piskelapp tool

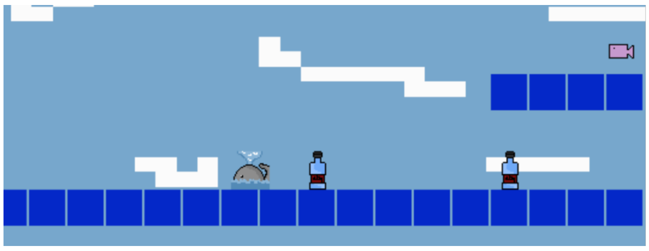


Figure 5.fish - a sample set of assets created in Piskel in the context of a working platform game created by child in P2

While I made several technical adjustments to facilitate the swapping of participant graphical designs (outlined in Appendix 5.tech), the process still required a series of potentially tricky operations. While some novice code authoring tools offer self-contained solutions for audio and graphical asset creation by providing in-built authoring tools and libraries of assets, the code playground Glitch provided neither, thus requiring the use of Piskel as an external asset creation tool, complicating the process. However, this forced choice to use a distributed toolset, rather than a self-contained approach to asset management, led to benefits in developing key digital literacy skills needed for web creation. Some participants became remarkably adept at the complex process of migrating assets from Piskel into their games, transforming the chain of actions involved into fluid operations. This section has focused on the rationale behind the introduction and initial expansion of the design's primary tools, rather than evidencing their subsequent impact on participants, which is explored in Chapter 6.

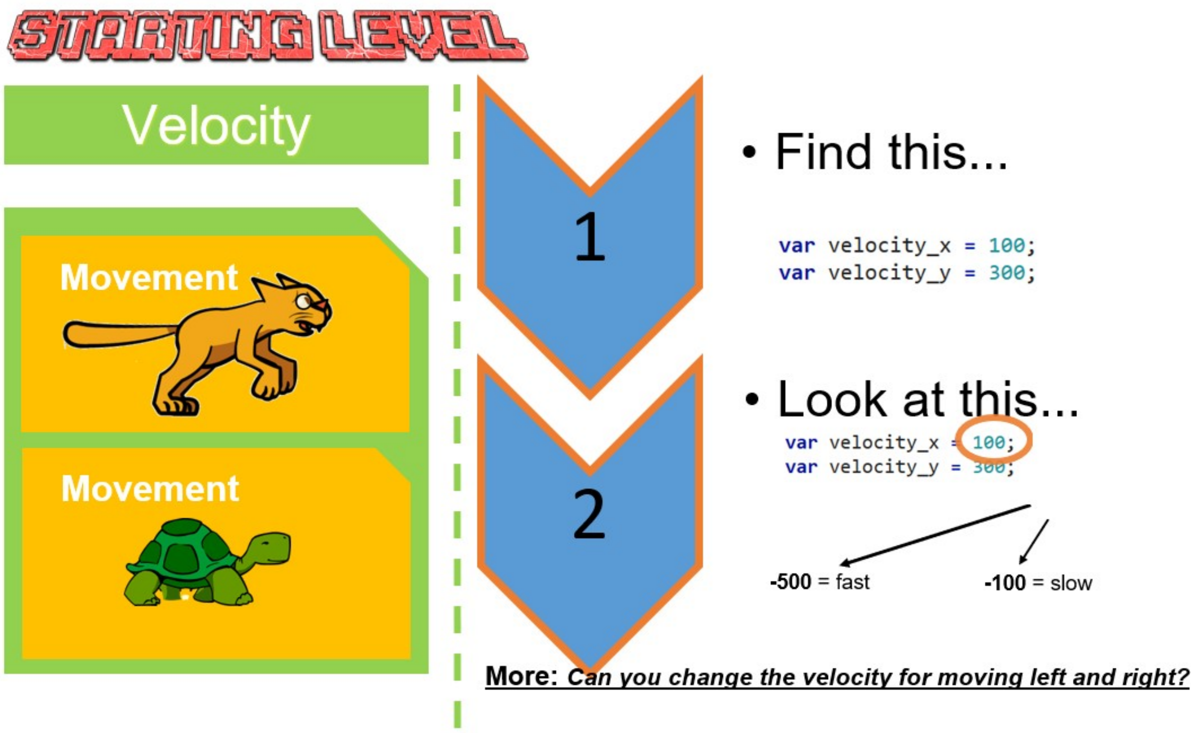
C2 - Contradictions associated with project navigation and use of documentation

The introduction of a half-baked template and the UMC approach had the unintended consequence of reducing the average working group size and also helped by introducing coding tools early, thus avoiding the mismatch between participants' planning and the technical limits of their novice abilities. The ability for individuals and pairs to make quick changes to the games templates via in terms of the affordances described in the previous section appeared to build their ownership over these fledgling games and set a path of working in a smaller group. A negative consequence of smaller group sizes and thus having more games being worked on was the increased demand on my time as a technical troubleshooter for code problems. This section addresses this tension and others related to the introduction of supporting forms of documentation.

While the in-tool scaffolding within the primary tool outlined in the previous section reduced initial reliance on facilitator instruction (Laurillard, 2020), it did not assist participants in implementing new features or adding code structures to their games (e.g., the *Create* stage of UMC). In P1 and early in P2, my guidance and often direct help was needed at this stage, causing delays and participant frustration due to my limited time with each participant or pair. During the development period after P1, I reflected on the tension between learner agency, particularly the role of divergent participant learning paths, and the role of facilitator-led instruction and support, which reappeared consistently in my notes at this time. My reflection was partly driven by two attempts in P1 to deliver whole-class instruction on key coding concepts that I judged to be relevant to the wider group. My reflection on these interventions noted that learner engagement was low, and I surmised that this might be because learners viewed the instruction as not immediately relevant, alien to their favoured hands-on style of working, or as an interruption to the flow of their making activities. This observation aligns with the concept of just-in-time learning approaches within project-based learning (Riel, 1998), where access to supporting documentation is provided based on learner need. During this development period, I struggled to reconcile my desire to support this responsive approach with requests from some parents in the feedback on P1, asking for background concepts and explanations of coding constructs. The design and implementation of supporting resources continued to evolve at a frantic pace at the start of P2, and I finally created three main sources of documentation: quick start cards, written instructions, and code snippets, which are outlined in the following section.

The use of documentation to address contradictions

This section explores the introduction of quick start cards, written instructions, and code snippets to address some of the contradictions in the learning design outlined above. *Quick start cards*, simple printouts highlighting the key affordances of the template involving: game mechanics such as movement, jumping, level design, and the final challenge of swapping out the look of one or more characters by designing pixel art and replacing the line of code that adds the asset to the game. These printed resources which highlighted key lines of code and demonstrated how they could be altered to impact game behaviour (see Figure 5.cards below), became a crucial resource for participants to get started with the starting template game from P2 onwards.



```
var velocity_x = 100;
var velocity_y = 300;
```

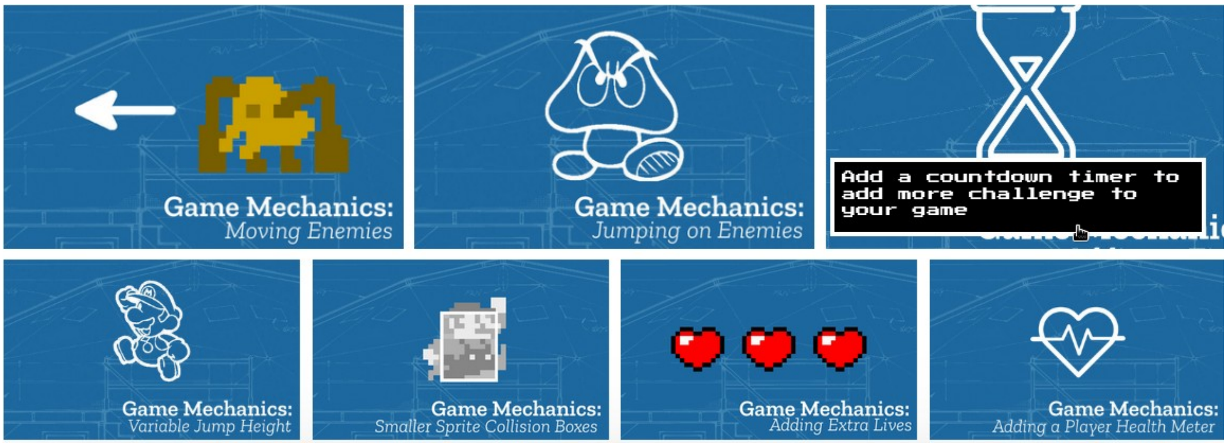
5.cards - Example of a Quick start card

A version of the quick start cards is available online and as part of Appendix.tech. These *quick start cards* supported participants' initial interaction with the code in a way that strongly aligned with the use and modify stages of the UMC framework (Franklin et al., 2020). The starting process addressed the tension of providing documentation to support users in completing coding tasks while also facilitating choice over learner pathways.

Structuring instructional tutorial resources and code snippets

To meet participants' requests for foundational coding knowledge, in the development period after P1, I developed an online manual with step-by-step chapters guiding users to code a core game template structure from first principles. While this linear format did not fully align with my choice-driven, face-to-face sessions, I envisioned it as supplementary reinforcement for learning outside of sessions. Writing these chapters revealed a technique that helped balance structured guidance with choice-driven learning. Each chapter demonstrated a code feature or principle through a playable game project based on the core game template, hosted in an online code playground. While the use of code examples or *snippets* is a common professional practice in problem-solving (Yang et al., 2017), their use by novice learners presents challenges related to relevance, consistency, and accessibility (Treude and Robillard, 2017). Initially, I had encouraged participants to use code snippets from the Phaser website, but their difficulties understanding and applying these abstracted examples prompted me to create more tailored resources.

I later extended this approach by creating stand-alone code snippets that illustrated requested game features as gameplay design patterns, such as *jumping on an enemy to zap it* and *making a moving enemy* (see Appendix.toby for an applied example). In the early stages of P2, learners accessed these projects through a simple Google document with links and brief descriptions. I continued developing new projects and producing printable instructions to support these code snippets, ensuring that each snippet linked to a descriptive chapter and vice versa. To help learners situate the code within the correct structure, all projects used the core game template and included only the new code required for each feature. I also experimented with ways to present these feature choices in an accessible, engaging format, which I describe in detail in Appendix 5.map. Recognising that multiple documentation formats sometimes led to confusion, I created a centralised hub to host both snippets and tutorial chapters, making navigation more intuitive and orienting documentation towards participants' gameplay experience.


Game Mechanics: Moving Enemies
Game Mechanics: Jumping on Enemies
Add a countdown timer to add more challenge to your game
Game Mechanics: Variable Jump Height
Game Mechanics: Smaller Sprite Collision Boxes
Game Mechanics: Adding Extra Lives
Game Mechanics: Adding a Player Health Meter

Pattern Collection

Figure 5.patterns : a screenshot of the hub of GDPs pointing to code snippets and instructional chapters

Chapter 2 reviewed research on using collections of gameplay design patterns to support learning in game design (Bjork and Holopainen, 2005; Holopainen et al., 2007; Holopainen, 2011; Eriksson et al., 2019). Thematic organisation of these patterns shows potential to foster a shared understanding of game-making concepts within a coding community (Holopainen, 2011). For the documentation hub described above, I grouped game design patterns into categories based on academic and professional interpretations of game elements (Tekinbaş and Zimmerman, 2003; Salen et al., 2006; Schell, 2008; Olsson et al., 2014), as well as participants' evolving requests for game features. The final categorisation used in P4 is shown below in Table 5.x.

Game Mechanics	**Game Polish**	**Game Space**	**Challenge Systems**
Add Static Hazard	Add Graphical Effects	Change Design of Levels	Gain Points when Collecting Food
Add an Animated Enemy	Add Sound Effects	Add More Levels	Add a Timer
Jump on Enemy to Zap them	Add a Sound Track (Music)	Change Shape of Levels	Collect all Food before Progressing
Double Jump	Add a Game Story with Messages	Change the Background Image	Power up - Higher Jump
Moving / Patrolling Enemies	Add a Game Story with Messages	Change the Background Image	Power up - Player Speed
Moving / Following Enemies	Animate your Player's Movements	Key and Door	Random Doubling Enemies
	Make Player Immune		

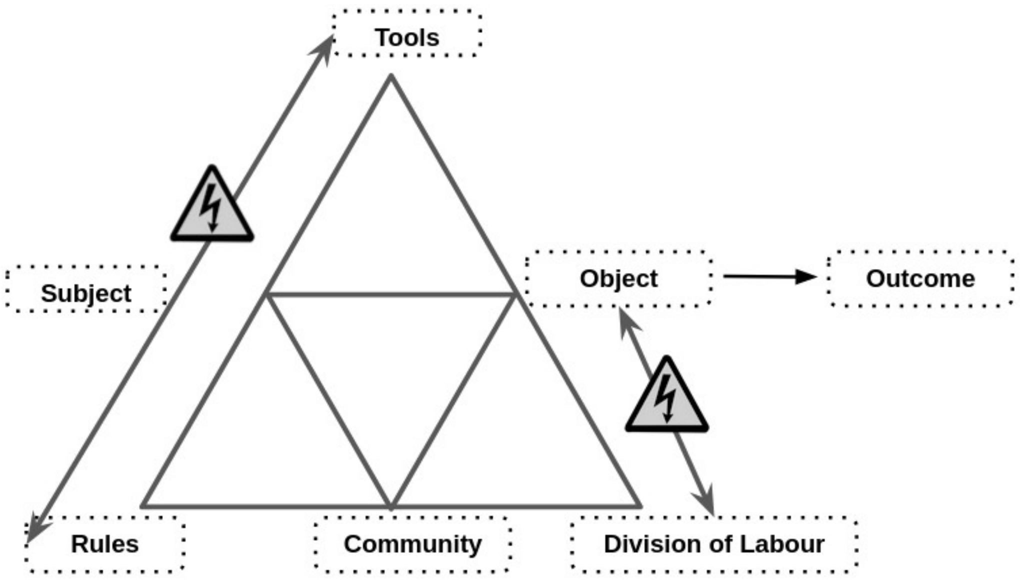
Table 5.x Categorisation of game design patterns used in P4.

This section outlined two key types of secondary support tools developed in response to tensions arising after introducing the primary tools discussed in the previous section. While the code snippets, practical instructions, and chapters on core principles continued to pose some challenges (as seen in Vignette 6.1), the video data analysed in the following chapter shows an increase in participants' practical ability to implement game features, suggesting an enhancement in their instrumental agency. In addition, participants' increase in agency extends beyond just instrumental application. In both P1 and P2, participants made specific requests for new features, which, when deemed suitable by me as the facilitator, were added to the menu of game design patterns. This choice-driven approach can be seen as both relational and authorial agency within the framework discussed in Chapter 3.

C3 - Responding to tensions and barriers in cultural aspects of the game making activity

Some participants experienced alienation during the transition to coding, stemming not only from challenges with tools but also from a sense of exclusion from the culture of coding. This aligns with research on cultural barriers in coding processes, as discussed in Chapter 2 (Kafai and Burke, 2015). In P1, as described above, I incorporated several non-coding techniques to foster positive affect and create an inclusive idioculture, aiming to reduce alienation from unfamiliar coding tools. While participant feedback on these techniques was positive, accumulated tensions led one family, described in the case study Appendix 5.bee, to disengage. Members of this family actively engaged in planning on paper and creating pixel art. However, tensions arose when the introduced coding framework did not support a desired feature, which was a bee design roaming a 3D landscape. During participant feedback, the parent expressed a preference for more hands-on play and exploration of the tools of production before being required to make creative decisions. This case study highlighted several key concerns for the design. Firstly, it underscored the fragility of learners' positive affect during the transition from the planning and sketching phase to the coding phase of the game-making process. Secondly, it revealed the limitations of peer support, as fellow participants were often unable to address more complex issues independently. Lastly, it highlighted a potential imbalance in facilitator attention, as participants more confident in coding sometimes created complex problems requiring significant facilitator time, potentially leaving others feeling less valued.

Though technical challenges related to the use of tools played a role, this area of conflict was compounded by social and cultural factors, ultimately leading to the family's decision to step away from the activity. The tensions arising from the expectation to transition from non-coding creative activities to coding tools are depicted in Fig. 5.x below. This figure illustrates how the family's positioning and alienation from the coding environment, perceived as a space for *hardcore coding* (participants phrase), represent a tension between the division of labour and the object of activity.



tensions with cultural layers

Fig 5.x - Tensions involving cultural aspects of the game making process

My reflections on this conflict prompted several changes in the learning design to address factors situated within the plane of cultural activity. While Chapter 7 focuses on analysing participant reactions to the design from a more cultural perspective in greater depth, this section is limited to exploring the evolution of key learning design features addressing these issues, particularly the identification and fostering of social coding processes.

Social coding processes

The changes in tools and documentation introduced in P2, as described earlier, were initially implemented to address technical issues but also had a significant impact on the cultural dynamics of the game-making group. These adjustments ameliorated some of the systemic conditions that contributed to the conflicts observed in P1. For example, the introduction of a partially completed, half-baked game template had a notable effect on the division of labour within the group. This adjustment led to the formation of smaller, more autonomous groups. In the case study explored in Appendix.bee, a problematic division of labour alienated non-coding team members from the overall process. By allowing participants to work individually or with family members, these barriers to engagement were substantially reduced. The highlighted affordances of the template and the accompanying collection of code examples, as outlined in earlier sections, proved impactful. Despite their relative simplicity, these tools facilitated the creation of games with diverse and complex styles and themes, even at early stages. The provision of a functioning starter game significantly increased the amount of time participants spent playtesting both their own and others' games. Video data captured instances where participants expressed frustration when errors rendered their games unplayable (see Vignette 6.1 for a clear example). In response, I prioritised supporting participants in resolving these issues to ensure their games were functional and playable. By P2, playtesting had evolved into a source of rich, productive peer social interactions. Observing this, I recognised the potential of these interactions and, in preparation for P3, aimed to intervene more strategically and gently to nurture and encourage these emergent patterns of peer collaboration and engagement.

Use of side missions to encourage varied creative practices

In P3, I introduced a drama-based scenario to guide the game creation process. This scenario revolved around an alien making contact with the group to request the creation of games as a means to judge the worth of the human race (see Vignette 5.alien). Within this fictional framework, participants engaged in side-missions, some of which were public while others remained secret (see Table 5.sidemissions). The introduction of this drama and the accompanying social missions aimed to reinforce the social and cultural dimensions of the activity, building on patterns observed during semi-structured playtesting in earlier phases. These missions were designed to encourage behaviours that fostered community-focused interactions. They included activities not directly related to the game-making process but rather centred on exploring and engaging with others' games and adopting a playful, exploratory approach. I had previously identified such behaviours as crucial for sustaining positive affect and fostering identification with the ongoing group game-making process. The missions were printed on cards, and participants were given one of each type during the first half of the initial two sessions. An sample of the social and secret missions is provided below, with the full table available in Vignette 5.alien.

Your Alien Mission (social)	**Your Secret Alien Mission:**
Find out the names of 3 games that are being made.	Change the variables at the start of someone else's game to make it play in a funny way.
Make a list of characters in two other games being made.	Change of the images in someone else's project to a totally different image and see if they notice.
Find out the favourite computer games of 4 people.	Change the level design of the first level of someone else's project to make it impossible but try to change as little as possible to do that.

Table 5.sidemissions - an extract of side missions given as part of the drama scenario

While this provides an incomplete picture of the missions and drama process, due to space limitations, these side missions are presented in this chapter to offer an overview of the varied tools used. A more detailed exploration of the development of participant agency in relation to these social missions and playtesting is provided in Chapter 7. Within the framework of 3GAT, these missions can be understood as tools used by participants to develop agency through the process of double stimulation, a theme which is continued in the discussion section of this chapter.

Chapter Discussion

This chapter has explored the complexity of the interacting tools and documentation in relation to their evolution in different phases of the formative intervention process. The contradictions explored initially focused on technical and then more social dimensions of the learning design, a process which has surfaced some of the barriers to achieving computational fluency needed to complete the text coding of a multi-media project. The next section synthesises the features of the design narrative, underpinned by existing research, specifically UMC, half-baked approach and constructionist design principles. To do this it employs an argumentative grammar fusing elements of DBR and CHAT concepts: specifically iterative processes, analysis of emerging tensions between systems elements, a focus on affordances and secondary stimuli (particularly in relation to participant agency), and the process of rising to the concrete through exploration of a germ cell concept.

The emergent and mutual nature of this design process aligns with both DBR and CHAT principles. In this chapter and the previous one, I have described my active role in motivating and supporting an emerging community through iterative phases. The design approach was open and risky, requiring me to rapidly generate new tools in response to changing needs. I was fortunate that the tenacious character of the participants in P1 helped mitigate the challenges posed by the incomplete design. The concept of affordances, commonly used in DBR, is present in this design through the careful selection of technical tools and non-technical pedagogical strategies. The term affordances has also been extensively explored in the literature of human-computer interaction (HCI) from a CHAT perspective, defined as “possibilities for human actions mediated by cultural means” (Kaptelinin and Nardi, 2012:927). In 3GAT terminology, the affordances of my design can be framed as a series of secondary stimuli. Sannino (Sannino, 2015) highlights the intersection of the use of secondary stimuli by participants and their volitional action or agency. While subsequent chapters explore shifts in the overall structure of participant activity in alignment with concepts of relational and transformative agency, the evolution of these tools, based on the novice status of the participants, also led to an increase in instrumental agency. This instrumental proficiency was required before the rich set of interactions based on peer learning and other social coding processes could emerge in playtesting practices. Instrumental agency can be seen as the base of a hierarchy of agency, in other words, that other dimensions of agency may need to be built on this one. This foundational agentic level is further explored in relation to other research in the following section.

The alignment of the underlying principles of the learning design with research constructionist design heuristics, half-baked designs, and the use-modify-create (UMC) process merits deeper exploration. The work of Kynigos et al. (Kynigos and Yiannoutsou, 2018) on the concept of half-baked microworlds and games helped me focus on the motivational aspect of incomplete game affordances to drive initial engagement with the *use* and *modify* stages of the UMC approach. Small code changes resulted in potentially large changes in game behaviour, appearance and difficulty aligns with a long standing concept of HCI research that feedback is motivating for system users (Malone, 1982; Bernhaupt et al., 2015). The careful design work on the P2 starting game template facilitated participants in building familiarity with the code structures and use of tools in an accelerated way. In addition, the framework of UMC highlights the benefits of a stepped, stage-based design. In this design, this staged approach is enacted through the interaction of participants with code via increasingly sophisticated documentation. My observations, supported in the following chapter, indicate that the combination of the use of a template and a collection of GDPs accelerates the use of certain coding fluency practices, which, in turn, had a subsequent impact on the increased use of social coding processes.

Turning to constructionist research, a more extensive set of design heuristics is evident in studies of decisions made in creating microworld environments and in the development of Scratch (see Chapter 2 for a summary). Techniques or concepts present in constructionist design research, embodied in this intervention’s toolset, include a simplified code authoring environment with live feedback in a preview window, techniques to avoid and correct syntax errors via the linting capabilities of the code playground (see glossary), and the use of a GDP menu to encourage creative but scaffolded design. A detailed description of these elements is explored in Appendix.tech. Of particular relevance to this design process is the principle *choose black boxes carefully* (Resnick and Silverman, 2005). Resnick et al.(2005) describe black boxes as abstractions that remove potentially problematic areas of the production process, creating a protective, scaffolded learning experience while maintaining a boundary with the authentic challenges of coding in the wild. Black box decisions steer learners toward exploring preferred concepts. Papert’s (1980) work on Turtle control in LOGO exemplifies this approach, using abstraction to focus on powerful mathematical ideas. The use of black boxes is also widespread in professional coding practices, particularly through code libraries and frameworks. These tools provide pre-written, self-contained sections of code to simplify complex operations. In my design, the Phaser game-making code library framework serves this purpose by managing underlying structures for gravity and physics calculations necessary for object collisions. In addition, my starting game template further simplified these processes by allowing participants to modify gravity settings through a single variable, thus combining professional coding language use with an accessible learning experience. Based on Laurillard’s (2020) interpretation of the constructionist design of microworlds as being driven by affordances that structure user experimentation to enable exploration and concept formation without reliance on direct instruction, I propose this design as a form of microworld. When compared to other game coding programs using text languages this learning design can be seen as tightly structured. However, when compared to a constructionist microworld, the structure of my learning design is compromised by its relatively porous borders between this controlled environment and other, more authentic coding ecosystems. I argue that increased scaffolding of learner resources and tools can enhance instrumental agency for novice coders but potentially at the cost of limiting access to the tools and processes of authentic creative communities. The relationship between authentic and scaffolded processes can be understood as an inherent dialectical tension that learning designers should account for rather than attempt to eliminate.

Addressing authenticity in more detail, it is valuable to re-examine the factors driving my choice of JavaScript and Phaser, a professional, open-source game-making framework. Specifically, this choice relates to the potentially empowering impact of exposing participants to authentic technologies and concepts in hands-on, exploratory processes. Ratto (2011) discusses this through critical making, a process that playfully brings forward Latour's (2005; 2008) ideas of shifting from matters of fact to matters of concern by revealing taken-for-granted artefacts as deliberately designed objects. This was evident in conversations between participants where they expressed a sense of inspiration or engagement with previously unknown technologies. For instance, exchanges demonstrated a growing appreciation for the complexity behind professional games, based on their perception of the effort required for their own simple game projects:

```
Pearl: It just shows you what goes into these games.
Student Helper 3: Think about how much effort goes into.
Pearl: You just take things for granted don't you?
```

A possible drawback of using a professional tool for novice learners is the risk of culture shock when participants encounter the unfamiliar and potentially alienating aspects of a professional coding ecosystem. The learning design outlined in this chapter helps address at least the beginning of this process by providing a protective, scaffolded learning environment.

The process of rising to the concrete via a germ cell concept is a powerful analytical and developmental tool (2012). In seeking the germ cell, the researcher explores complex, messy concrete processes over time to discern overarching concerns that can be expressed as a prototypical abstract germ cell concept. For this learning design, the concept and pedagogy of working with game design patterns (GDPs) emerged not only as an object of activity on a more granular scope than is typical but also as a germ cell concept valuable to both participants and facilitators. I continue the analysis of the orienting practice of GDPs as a germ cell concept in the following chapter.

In the learning design outlined in this chapter, GDPs drive stage-based instructional processes. Using terminology from Denner et al. (2019), the shift from P1 to P2 represents a move from a design-build-test to a stepwise, stage-based approach. Stepwise approaches introduce code structures and challenges in an initially simplistic, prototypical format for learners to experience and explore, which are subsequently replaced by more advanced models or complex instances of the task. While the initial use of GDPs in P1 was piecemeal, by P2, a more structured, stepwise approach had been implemented for participants as part of several guiding activities. GDPs were first identified through playing games in an activity adapted from Moveable Game Jams (ensure this is described or linked to) and were used as a list of features serving as a mediating planning tool. In P2, GDPs were incorporated as affordances into a starter game template, initially highlighted via facilitator interaction and later as the guiding principle of *quick start cards* to scaffold modifications to the code template. The structure of documentation, created in response to participant requests for adding new GDPs to their games, evolved from a piecemeal format in P1 to, by the end of P2, a menu of GDP code snippets accompanied by detailed instructional documentation. In line with understandings of DBR research outputs, the identification and use of these patterns as research findings would be expressed as a set of "tentative generalisations" in the form of design heuristics drawn from the analysis of concrete practices (Hoadley and Campos, 2022:215).

This chapter has explored the iterative development of the learning design, focusing on resolving barriers to coding and aligning with existing research principles, particularly the use-modify-create (UMC) process and constructionism. It emphasized the potential and challenges of leveraging professional tools within scaffolded environments, underscoring the pivotal role of learner-led design decisions in motivating coding practices and fostering social collaboration. Two key lines of inquiry emerged and are explored in subsequent chapters. Chapter 7 examines the impact of emerging community processes on participants' agency, while the next chapter employs video data to describe and analyse the diverse applications of GDPs. This analysis further illuminates in particular the cultural and social dimensions of game-making as a collaborative activity.

Barab, S. A., Barnett, M., Yamagata-Lynch, L., Squire, K. and Keating, T. (2002) 'Using Activity Theory to Understand the Systemic Tensions Characterizing a Technology-Rich Introductory Astronomy Course.' *Mind, Culture, and Activity*, 9(2) pp. 76–107.

Bell, P. (2004) 'On the Theoretical Breadth of Design-Based Research in Education.' *Educational Psychologist*, 39(4) pp. 243–253.

Bernhaupt, R., Isbister, K. and De Freitas, S. (2015) 'Introduction to this Special Issue on HCI and Games.' *Human–Computer Interaction*, 30(3–4) pp. 195–201.

Bjork, S. and Holopainen, J. (2005) *Patterns in game design*. 1st ed, Hingham, Mass: Charles River Media (Charles River Media game development series).

Brase, A. K. (2024) 'Knowledge generation between design, data and theory: Argumentation in design-based research.' *EDeR. Educational Design Research*, 8(1).

Chesterman, M. (2015) *Webmaking using Javascript to promote student engagement and constructivist learning approaches at KS3*. [Online] [Accessed on 19th March 2016] https://web.archive.org/web/20180831105358/http://digitalducks.org/blog/project-report-webmaking/.

Denner, J., Campe, S. and Werner, L. (2019) 'Does Computer Game Design and Programming Benefit Children? A Meta-Synthesis of Research.' *ACM Transactions on Computing Education*, 19(3) pp. 1–35.

Engeström, Y., Nummijoki, J. and Sannino, A. (2012) 'Embodied Germ Cell at Work: Building an Expansive Concept of Physical Mobility in Home Care.' *Mind, Culture, and Activity*, 19(3) pp. 287–309.

Erhard-Olsson, N. (2018) *Procedural Generation of 2D Levels for a Motion-based Platformer Game in Unity with Large Amount of Movement*.

Eriksson, E., Baykal, G. E., Björk, S. and Torgersson, O. (2019) 'Using Gameplay Design Patterns with Children in the Redesign of a Collaborative Co-located Game.' *In Proceedings of the 18th ACM International Conference on Interaction Design and Children*. Boise ID USA: ACM, pp. 15–25.

Franklin, D., Coenraad, M., Palmer, J., Eatinger, D., Zipp, A., Anaya, M., White, M., Pham, H., Gökdemir, O. and Weintrop, D. (2020) 'An Analysis of Use-Modify-Create Pedagogical Approach's Success in Balancing Structure and Student Agency.' *In Proceedings of the 2020 ACM Conference on International Computing Education Research*. Virtual Event New Zealand: ACM, pp. 14–24.

Guzdial, M. (2004) ‘Programming Environments for Novices.’ *In* Fincher, S. and Petre, M. (eds) *Computer Science Education Research*. Taylor & Francis.

Hoadley, C. and Campos, F. C. (2022) ‘Design-based research: What it is and why it matters to studying online learning.’ *Educational Psychologist*, 57(3) pp. 207–220.

Hoadley, C. P. (2002) ‘Creating Context: Design-based Research in Creating and Understanding CSCL.’

Holopainen, J. (2011) *Foundations of gameplay*. phd. Blekinge Institute of Technology.

Holopainen, J., Bjork, S., Kuittinen, J., Mayer, I. and Mastik, H. (2007) ‘Teaching gameplay design patterns.’ *Organizing and Learning through Gaming and Simulation, Proceedings of ISAGA*.

Kafai, Y. and Burke, Q. (2015) ‘Constructionist gaming: understanding the benefits of making games for learning.’ *Educational Psychologist*, 50(4) pp. 313–334.

Kaptelinin, V. and Nardi, B. (2012) ‘Affordances in HCI: toward a mediated action perspective.’ *In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. New York, NY, USA: Association for Computing Machinery (CHI ’12), pp. 967–976.

Kynigos, C. and Yiannoutsou, N. (2018) ‘Children challenging the design of half-baked games: Expressing values through the process of game modding.’ *International Journal of Child-Computer Interaction*, April.

Latour, B. (2008) ‘A cautious Prometheus? A few steps toward a philosophy of design (with special attention to Peter Sloterdijk).’ *In Proceedings of the 2008 annual international conference of the design history society*, pp. 2–10.

Laurillard, D. (2020) ‘The significance of Constructionism as a distinctive pedagogy.’ *CONSTRUCTIONISM 2020*, 29.

Lee, I., Martin, F., Denner, J., Coulter, B., Allan, W., Erickson, J., Malyn-Smith, J. and Werner, L. (2011) ‘Computational thinking for youth in practice.’ *ACM Inroads*, 2(1) pp. 32–37.

Leontiev, A. N. (2009) ‘Activity and Consciousness.’ *Revista Dialectus*, (4).

Lytle, N., Catete, V., Isvik, A., Boulden, D., Dong, Y., Wiebe, E. and Barnes, T. (2019) ‘From “Use” to “Choose”: Scaffolding CT Curricula and Exploring Student Choices while Programming (Practical Report).’ *In Proceedings of the 14th Workshop in Primary and Secondary Computing Education*. New York, NY, USA: Association for Computing Machinery (WiPSCE’19), pp. 1–6.

Malone, T. W. (1982) ‘Heuristics for designing enjoyable user interfaces: Lessons from computer games.’ *In Proceedings of the 1982 Conference on Human Factors in Computing Systems*. New York, NY, USA: Association for Computing Machinery (CHI ’82), pp. 63–68.

Olsson, C. M., Björk, S. and Dahlskog, S. (2014) ‘The conceptual relationship model: understanding patterns and mechanics in game design.’ *In Digital games research association (DiGRA), snowbird, utah, USA (2014)*. DiGRA, pp. 1–16.

Papert, S. (1980) *Mindstorms: children, computers, and powerful ideas*. 2nd ed, New York: Basic Books.

Ratto, M. (2011) ‘Critical Making: Conceptual and Material Studies in Technology and Social Life.’ *The Information Society*, 27(4) pp. 252–260.

Resnick, M. and Silverman, B. (2005) ‘Some reflections on designing construction kits for kids.’ *In Proceedings of the 2005 conference on Interaction design and children*. ACM, pp. 117–122.

Resnick, M., Silverman, B., Kafai, Y., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., Millner, A., Rosenbaum, E. and Silver, J. (2009) ‘Scratch: programming for all.’ *Communications of the ACM*, 52(11) p. 60.

Riel, M. (1998) ‘Education in the 21st century: Just-in-time learning or learning communities.’ *In fourth annual conference of the emirates center for strategic studies and research, Abu Dhabi*.

Salen, K., Tekinbas, K. S. and Zimmerman, E. (2006) *The Game Design Reader: A Rules of Play Anthology*. MIT Press.

Sannino, A. (2015) ‘The principle of double stimulation: A path to volitional action.’ *Learning, Culture and Social Interaction*, 6, September, pp. 1–15.

Schell, J. (2008) *The Art of Game Design: A book of lenses*. 1 edition, Amsterdam ; Boston: CRC Press.

Tekinbaş, K. S. and Zimmerman, E. (2003) *Rules of play: game design fundamentals*. Cambridge, Mass: MIT Press.

Treude, C. and Robillard, M. P. (2017) ‘Understanding Stack Overflow Code Fragments.’ *In 2017 IEEE International Conference on Software Maintenance and Evolution (ICSME)*, pp. 509–513.

Weibel, P. and Latour, B. (2005) *Making Things Public : Atmospheres of Democracy*. Weibel, P. and Latour, B. (eds). MIT Press.

Yang, D., Martins, P., Saini, V. and Lopes, C. (2017) ‘Stack Overflow in Github: Any Snippets There?’ *In 2017 IEEE/ACM 14th International Conference on Mining Software Repositories (MSR)*, pp. 280–290.