

Introduction to Game Development:

Legend of
Royal
Institution



What makes up a video game

Technical aspects of developing games

Practice

Further reading and recommendations

What
makes up
a video
game

Formulae of a game

Q: What makes game a game?

Game = ??? + ??? + ???



*Images courtesy of Sega Corporation, Xbox Game Studios,
Electronic Arts Inc. (top to bottom)

Gameplay

Describes the experience of play (what players **do**) in the game and defines the connection between a game and a player.

"The structures of player interaction with the game system and with other players in the game."

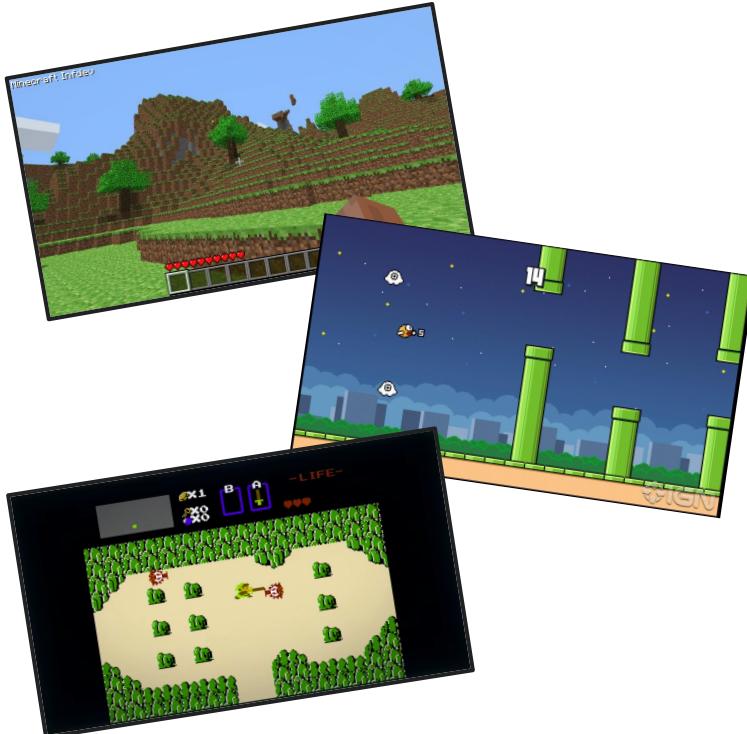
Mario Wonders:

- 2D platformer in an imaginary setting, where the character called Mario, controlled by a player, fights monsters and tries to save Princess Peach.



QQ:
What's your
favourite
game?

Mechanics



The rules, actions, and interactions defined in a game.

Minecraft:

- Player can destroy one world block by clicking on it.
- Monsters spawn at night.

Flappy Bird:

- Tubes collide.
- Game scrolls from one side.

The Legend of Zelda:

- Character moves to another dungeon by stepping over the edge of the screen.
- Character can hit monsters.
- Character takes limited amount of damage.

Analogy: Mechanics are like grammar rules for a language.

QQ:
What are the
game mechanics
your favourite
game
implements?

End goal

QQ: Why does a goal matter?

Crazy Taxi:

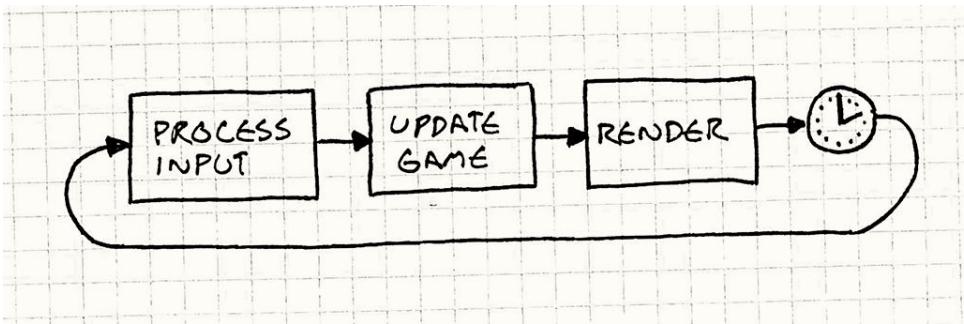
pick up and deliver as many passengers as possible





**Technical
aspects of
developing
games**

Game Loop



At the heart of every game is an infinite loop, often called the **game loop**. Each cycle of this loop is known as a **frame**.

During each frame, the game engine performs three key steps:

1. **Processes input/events** – like keyboard, mouse, or touch input.
2. **Updates the game state** – such as character positions, scores, and collisions.
3. **Renders the scene** – draws the current state to the screen.

Practice

Running activities

tinyurl.com/yzfvbyds

1 Exercise 1: Learn and Play

Welcome to your very first activity!

Before we dive into creating anything, let's take a moment to explore how the game works. In this step, you will play [The Legend of Royal Institution](#) (a maze game) and observe how it behaves.

Put on your game designer hat and figure out how the game works. The following questions will help you to start:

- What are the rules of the maze?
- How does the player move through it?
- What marks the end of the game?

Use the **text boxes** below to jot down any rules, patterns, or interesting game mechanics you notice.

As you explore the game, start thinking about how you could make it more interesting or challenging. What new **game mechanics** could you add? Think about how these elements could change the way the game is played and how they might make it more fun or challenging.

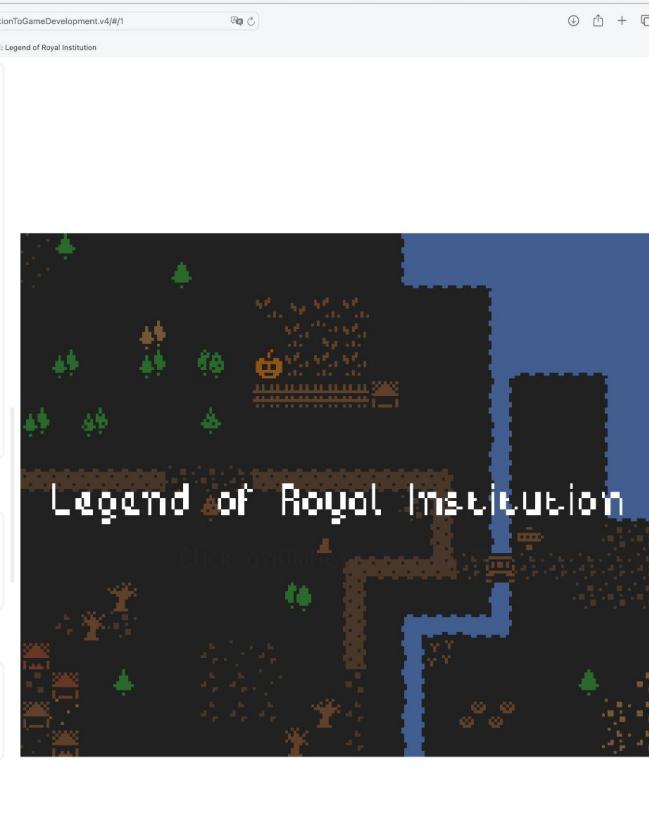
P.S.: There are no wrong answers! This is all about being curious and observant.

Observed mechanics:

- Here are a few ideas for the game rules:
- The character moves when user clicks a button
- Game starts with a dialog
- ...
- 5

Further improvements:

- Spawn the character randomly at some pre-defined start point
- ...
- 4
- 5



The screenshot shows a 2D pixelated game environment titled "Legend of Royal Institution". The scene is a dark, atmospheric maze with various obstacles and collectibles. A character is visible in the center-left area. The title "Legend of Royal Institution" is displayed prominently at the bottom of the screen. A blue path or highlight is visible on the right side of the maze. The overall aesthetic is reminiscent of classic 8-bit video games.

Learn
Play

Data-Driven Game Development

✉️ github.com/soulsmods/Paramdex

In a **config-driven design**, the core game engine is relatively generic, while the rules, content, and behaviors are defined in external data/config files.

Dark Souls (and the other souls-like games):
extremely config/data-driven under the hood.



Dark Souls 1: Config Example

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <PARAMDEF XmlVersion="1">
3      <ParamType>LIGHT_BANK</ParamType>
4      <DataVersion>4</DataVersion>
5      <BigEndian>False</BigEndian>
6      <Unicode>False</Unicode>
7      <FormatVersion>104</FormatVersion>
8      <Fields>
9          <Field Def="s16 degRotX_0">
10         <DisplayName>X角度</DisplayName>
11         <Description>平行光源: 0 </Description>
12         <EditFlags>None</EditFlags>
13         <Minimum>-90</Minimum>
14         <Maximum>90</Maximum>
15         <SortID>1</SortID>
16     </Field>
17     <Field Def="s16 degRotY_0">
18         <DisplayName>Y角度</DisplayName>
19         <Description>平行光源: 0 </Description>
20         <EditFlags>None</EditFlags>
21         <Minimum>-180</Minimum>
22         <Maximum>180</Maximum>
23         <SortID>2</SortID>
24     </Field>
25     <Field Def="s16 colR_0 = 255">
```

Data-Driven Game Development

Minecraft's gameplay is also not just defined by code in the engine: a lot of it lives in external JSON files, configs, and resource packs.

Data Packs

Introduced in **Minecraft 1.13+**, data packs define game mechanics, loot, recipes, and structures through JSON configs, such as, recipes, loot tables, or structures.



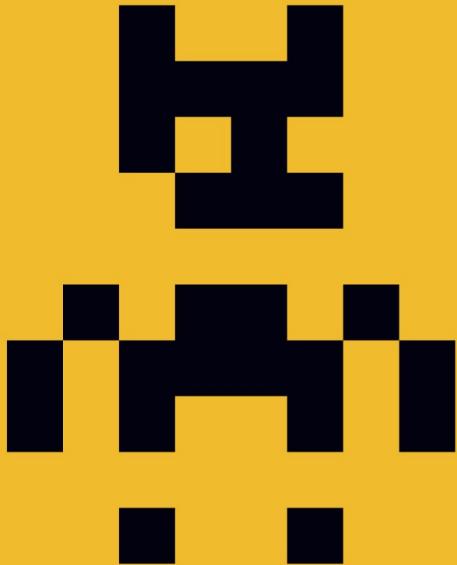
Minecraft: Config Example

```
{  
    "type": "minecraft:crafting_shapeless",  
    "category": "misc",  
    "ingredients": [  
        "minecraft:ender_pearl",  
        "minecraft:blaze_powder",  
        "minecraft:blaze_powder"  
    ],  
    "result": {  
        "count": 1,  
        "id": "minecraft:ender_eye"  
    }  
}
```

```
{  
    "type": "minecraft:crafting_shaped",  
    "category": "misc",  
    "key": {  
        "S": "#minecraft:stone_crafting_materials",  
        "C": "minecraft:coal_block"  
    },  
    "pattern": [  
        "SSS",  
        "SCS",  
        "SSS"  
    ],  
    "result": {  
        "count": 1,  
        "id": "minecraft:furnace"  
    }  
}
```



Data-Driven Game Development



The Legend of Royal Institution also relies on config files.

The configs define almost everything:

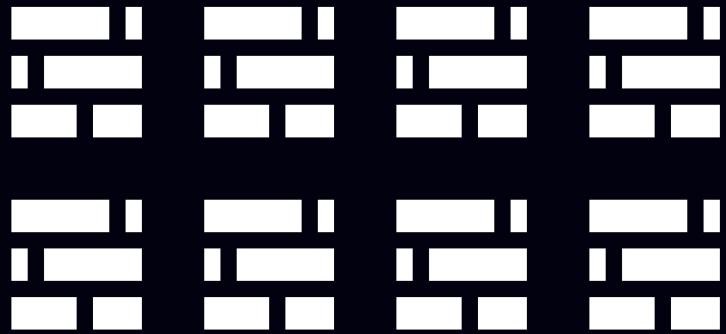
- Character appearance
- Finish (flag) position appearance
- Layout of the level

and many more...

Legend of Royal Institution: Config Example

```
1   {
2     "gameConfig": {
3       "applyCathodRayTubeEffect": false,
4       "messagesOverwrites": {
5         "character": 0,
6         "intro": "Greetings,\nfearless students!\nEscape the maze...\\nor at least debug it.",
7         "introHeight": 80,
8         "gameOver": "Mistakes happen...\\nGive it another go",
9         "victory": "Great job!\\nYou've completed\\nthe first step.\\nMore challenges\\nlie ahead!",
10        "victoryHeight": 90
11      },
12      "configsOverwrites": {
13        "walls": {
14          "variations": [340, 212, 341, 339, 338, 337, 344, 342, 343, 59]
15        }
16      }
17    },
18    "initialLevelId": 1,
19    "levels": [
20      {
21        "id": 1,
22        "title": "Workshop demo",
23        "levelLayout": [
24          "SP . W0 . . . W0 . . . W0 . . . W0 . W3 . . . W0",
25          ". W0 W2 . W0 W0 . W0 . W0 . W0 . . . . W2 . W0 . .",
26          ". . . . . . . W0 . W0 . W0 W0 W0 . . . . W0 . .",
27          ". W7 W7 W7 W7 . W0 . . . . . . W0 W0 W0 . W4 W0 . .",
28          ". . . . W0 . W0 W0 W0 . W0 W0 . . . . . . . . .",
29          "W0 W0 W0 . W0 . . . . W0 . W0 W0 W0 W0 W7 W3 W7 W0",
30          ". . . . W0 W0 W0 . W0 . W0 . . . . W1 . . . . . .",
31          ". W0 W0 . . . . . W0 . W0 W0 W7 . W0 W0 W0 W0 W0 . .",
32          ". . . W0 W0 W0 W0 . W0 . . . . . W0 . . . . . F0",
33          ". W0 . . . . . W0 W0 W1 . W7 . W0 . W0 W0 . . .",
34          ". W0 W0 W0 W0 . W0 W0 W0 . . . W1 . . . . W2 W0 . .",
35          ". . . . . . . . . . W1 W8 . W0 W0 . . . . F0",
36          ". W3 W3 W3 W3 W3 W3 W3 . . . . . . W0 W2 W0 W0 . .",
37          ". . . . . . . . . . W0 W0 . W0 W0 W0 . . . . . . "
38        ],
39      }
40    ]
41  }
```

Time to
tune
configs



1. Level
Layout

Tilemap

Tile is a small, reusable graphic (usually square or rectangular) that represents a piece of a larger map, environment, or grid in a game.

A **tilemap** is a grid-based system (usually, 2D) that arranges tiles to form game levels, environments, or maps.

There are a lot of games using tilemaps: **Terraria**, **Stardew Valley**, and **Pokemon Ruby**.

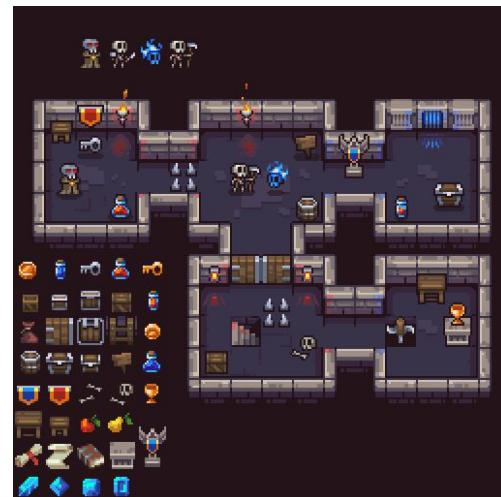


Tiles organisation: tilesets

pixel-poem.itch.io/dungeon-assetpuck



Tileset



Level example

To keep things efficient, games use a **tileset** (a collection of all available tile graphics).

The tilemap stores only the **index or ID** of which tile appears in **each grid cell**.

This allows developers to build large environments without duplicating image data.

Often, tilemaps are arranged in layers: *a background layer for terrain, a collision layer for walls and obstacles, and a decorative layer for details.*

Getting assets and publishing indie projects

 itch.io



Screenshot of the itch.io website showing the search results for "Top free Game assets tagged Tilemap".

The search bar shows the URL: itch.io/game-assets/free/tag/tilemap.

Filter results sidebar:

- Tags:
 - Tilemap
 - Related to the process of arranging game maps using a set of tile graphics (often call tilesets)
 - Suggest updated description
- Price:
 - Free
 - More Options
- Types:
 - Sprites
 - Sound effects
 - Music
 - Textures
 - Characters
 - Tiles
 - Backgrounds
 - Fonts
 - Icons
 - User Interface (UI)
- Styles
- Formats
- Themes
- Tools & Engines
- Misc
- When

Sort by: Popular, New & Popular, Top sellers, Top rated, Most Recent.

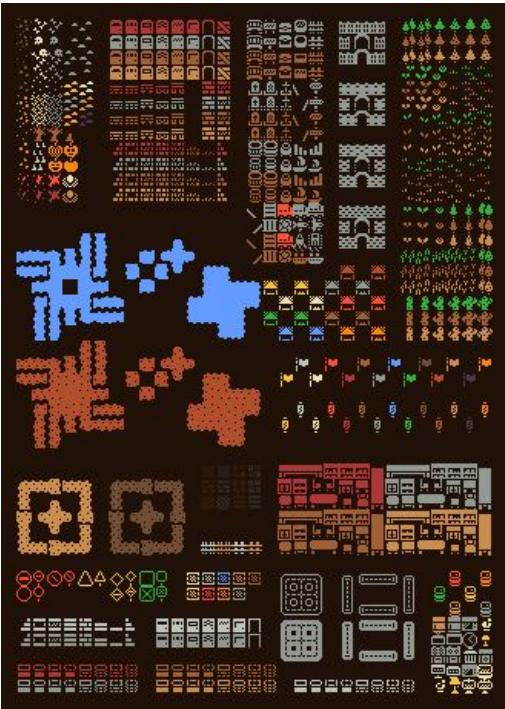
Selected tags: Pixel Art, Tileset, 2D, Fantasy, Sprites, Top-Down, Role Playing, Asset Pack, Platformer, Retro.

Results:

- FREE ASSETS EVERY WEEK** - The Mystery Pack by Backterria
- The Fantasy Tileset - 16x16 pixel art asset pack** by Ventilatore
- Sprout Lands - Asset Pack** by Cup Noodle
- 2D Pixel Dungeon Asset Pack** by Pixel_Poem
- Pixel Art Top Down - Basic** by Cainos
- 16x16 Dungeon Tileset II** by 0x72
- FREE ASSET GIFT** - LEGACY fantasy by Anokolisa
- Free - Hero's Journey - Moon Graveyard** by Anokolisa
- Free Pixel Art Forest** by edemunizz
- PIXEL FANTASY "CAVES"** by Szadi art.
- Kings and Pigs** by Pixel Frog
- GIFIGUE FANTASY "CATACOMBS"** by Szadi art.
- Free - Pixel Art Asset Pack - SideScroller Fantasy - 16x16 Fore...** by Anokolisa
- Free - Complete Cemetery Pixel Art Assets Pack** by edemunizz
- Pixel Fantasy Caves** by Szadi art.
- Rogue Fantasy Catacombs** by Szadi art.
- DUNGEON TILE SET** by Szadi art.

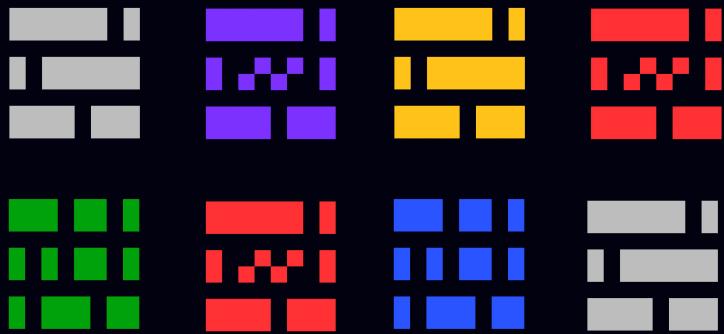
Legend of Royal Institution: tileset

 v3x3d.itch.io/bountiful-bits



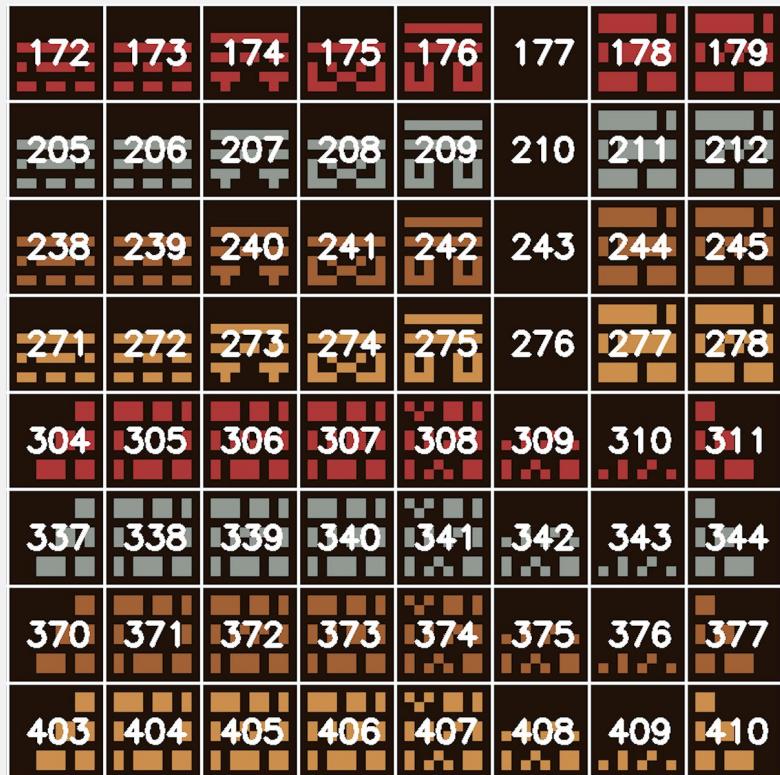
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31							
33	04	15	26	37	48	59	70	81	92	103	114	125	136	147	158	169	180	191	202	213	224	235	246	257	268	279	290	301	312	323								
66	67	78	89	100	111	122	133	144	155	166	177	188	199	210	221	232	243	254	265	276	287	298	309	320	331	342	353	364	375	386								
99	100	101	102	103	104	105	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127	128									
132	133	134	135	136	137	138	139	140	141	142	143	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159	160	161									
165	166	167	168	169	170	171	172	173	174	175	176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191	192	193	194									
217	218	219	220	221	222	223	224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239	240	241	242	243	244	245	246	247								
233	234	235	236	237	238	239	240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255	256	257	258	259	260	261	262	263								
266	267	268	269	270	271	272	273	274	275	276	277	278	279	280	281	282	283	284	285	286	287	288	289	290	291	292	293	294	295	296								
297	298	299	300	301	302	303	304	305	306	307	308	309	310	311	312	313	314	315	316	317	318	319	320	321	322	323	324	325	326	327								
330	331	332	333	334	335	336	337	338	339	340	341	342	343	344	345	346	347	348	349	350	351	352	353	354	355	356	357	358	359	360								
361	362	363	364	365	366	367	368	369	370	371	372	373	374	375	376	377	378	379	380	381	382	383	384	385	386	387	388	389	390									
391	392	393	394	395	396	397	398	399	400	401	402	403	404	405	406	407	408	409	410	411	412	413	414	415	416	417	418	419	420	421								
453	454	455	456	457	458	459	460	461	462	463	464	465	466	467	468	469	470	471	472	473	474	475	476	477	478	479	480	481	482	483								
495	496	497	498	499	500	501	502	503	504	505	506	507	508	509	510	511	512	513	514	515	516	517	518	519	520	521	522	523	524	525								
549	550	551	552	553	554	555	556	557	558	559	560	561	562	563	564	565	566	567	568	569	570	571	572	573	574	575	576	577	578	579								
581	582	583	584	585	586	587	588	589	590	591	592	593	594	595	596	597	598	599	590	591	592	593	594	595	596	597	598	599	590									
593	594	595	596	597	598	599	590	591	592	593	594	595	596	597	598	599	590	591	592	593	594	595	596	597	598	599	590	591	592									
611	612	613	614	615	616	617	618	619	620	621	622	623	624	625	626	627	628	629	620	621	622	623	624	625	626	627	628	629	620									
625	626	627	628	629	630	631	632	633	634	635	636	637	638	639	640	641	642	643	644	645	646	647	648	649	650	651	652	653	654	655								
661	662	663	664	665	666	667	668	669	670	671	672	673	674	675	676	677	678	679	680	681	682	683	684	685	686	687	688	689	680									
693	694	695	696	697	698	699	690	701	702	703	704	705	706	707	708	709	710	711	712	713	714	715	716	717	718	719	710	711	712									
729	730	731	732	733	734	735	736	737	738	739	740	741	742	743	744	745	746	747	748	749	750	751	752	753	754	755	756	757	758									
759	760	761	762	763	764	765	766	767	768	769	770	771	772	773	774	775	776	777	778	779	780	781	782	783	784	785	786	787	788	789								
791	792	793	794	795	796	797	798	799	800	801	802	803	804	805	806	807	808	809	810	811	812	813	814	815	816	817	818	819	820	821	822							
824	825	826	827	828	829	830	831	832	833	834	835	836	837	838	839	840	841	842	843	844	845	846	847	848	849	850	851	852	853	854								
856	857	858	859	860	861	862	863	864	865	866	867	868	869	860	861	862	863	864	865	866	867	868	869	860	861	862	863	864	865	866	867							
871	872	873	874	875	876	877	878	879	880	881	882	883	884	885	886	887	888	889	880	881	882	883	884	885	886	887	888	889	880									
893	894	895	896	897	898	899	890	891	892	893	894	895	896	897	898	899	890	891	892	893	894	895	896	897	898	899	890	891	892	893								
901	902	903	904	905	906	907	908	909	910	911	912	913	914	915	916	917	918	919	910	911	912	913	914	915	916	917	918	919	910									
919	920	921	922	923	924	925	926	927	928	929	930	931	932	933	934	935	936	937	938	939	930	931	932	933	934	935	936	937	938	939								
941	942	943	944	945	946	947	948	949	950	951	952	953	954	955	956	957	958	959	950	951	952	953	954	955	956	957	958	959	950									
959	960	961	962	963	964	965	966	967	968	969	960	961	962	963	964	965	966	967	968	969	960	961	962	963	964	965	966	967	968	969								
973	974	975	976	977	978	979	980	981	982	983	984	985	986	987	988	989	980	981	982	983	984	985	986	987	988	989	980	981	982	983	984							
991	992	993	994	995	996	997	998	999	990	991	992	993	994	995	996	997	998	999	990	991	992	993	994	995	996	997	998	999	990									
1003	1004	1005	1006	1007	1008	1009	1000	1001	1002	1003	1004	1005	1006	1007	1008	1009	1000	1001	1002	1003	1004	1005	1006	1007	1008	1009	1000	1001	1002	1003	1004							
1008	1009	1010	1011	1012	1013	1014	1015	1016	1017	1018	1019	1010	1011	1012	1013	1014	1015	1016	1017	1018	1019	1010	1011	1012	1013	1014	1015	1016	1017	1018	1019							
1021	1022	1023	1024	1025	1026	1027	1028	1029	1020	1021	1022	1023	1024	1025	1026	1027	1028	1029	1020	1021	1022	1023	1024	1025	1026	1027	1028	1029	1020	1021								
1034	1035	1036	1037	1038	1039	1030	1031	1032	1033	1034	1035	1036	1037	1038	1039	1030	1031	1032	1033	1034	1035	1036	1037	1038	1039	1030	1031	1032	1033	1034	1035	1036	1037	1038	1039			
1045	1046	1047	1048	1049	1040	1041	1042	1043	1044	1045	1046	1047	1048	1049	1040	1041	1042	1043	1044	1045	1046	1047	1048	1049	1040	1041	1042	1043	1044	1045	1046	1047	1048	1049				
1051	1052	1053	1054	1055	1056	1057	1058	1059	1050	1051	1052	1053	1054	1055	1056	1057	1058	1059	1050	1051	1052	1053	1054	1055	1056	1057	1058	1059	1050	1051	1052	1053	1054	1055	1056	1057	1058	1059

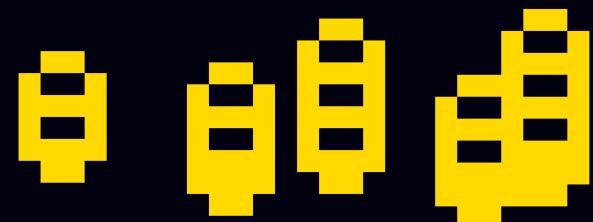
Our game also uses tilesets from [itch.io](#) with non-restrictive license.



2. Walls Appearance

Walls tileset

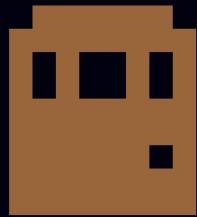




3. Coins

Coins tileset

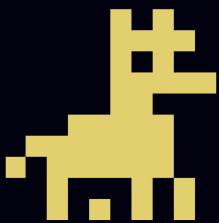




4. Doors

Doors&Keys tilesets





5. Monsters AI

Scripting Language: Bytecode

Game can roughly be divided into a **core/engine** code and a **gameplay** code.

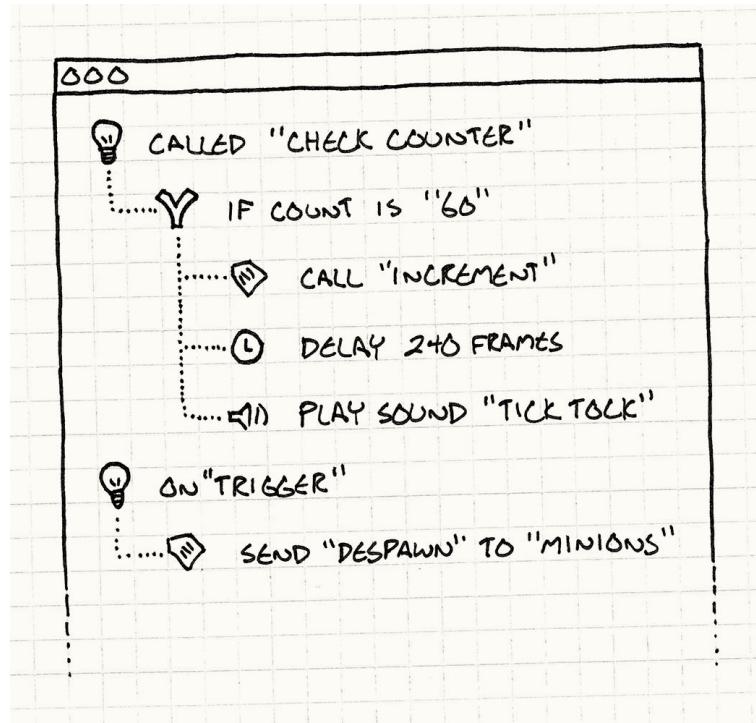
Engine:

- very generic
- extremely complicated
- rendering your game on a specific platform

Gameplay:

- game-specific behavior
- requires constant iteration

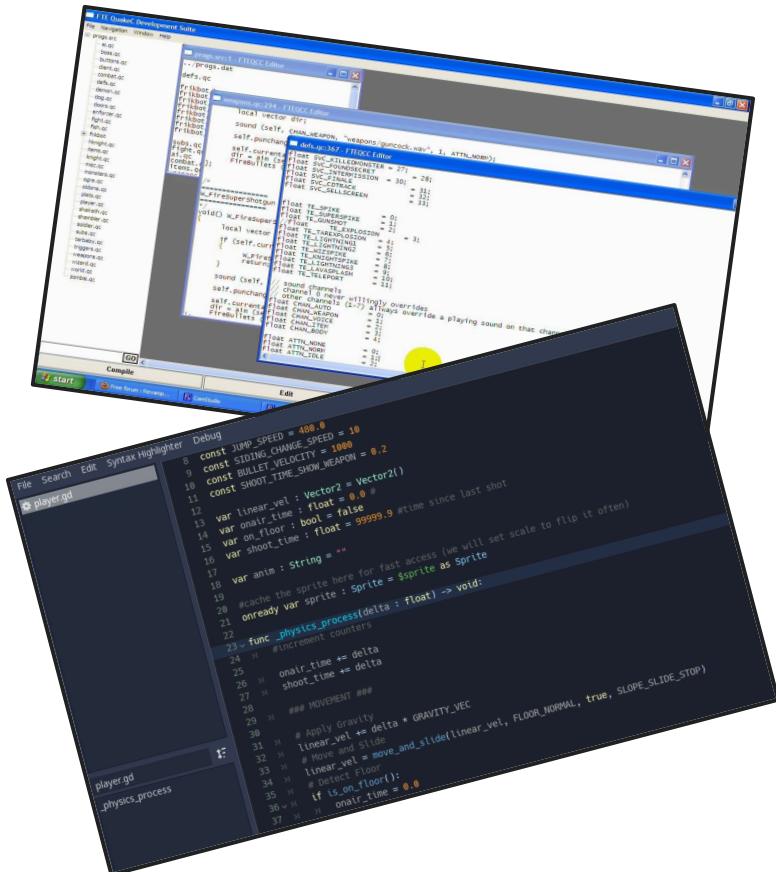
That's why game engines usually abstract gameplay logic into something more *lightweight*.



Scripting Language: Custom Language

The scripting languages can be as complicated as a full-scale language within the game engine.

- QuakeC
 - UnrealScript
 - GDScript
 - Naughty Dog's DC Language

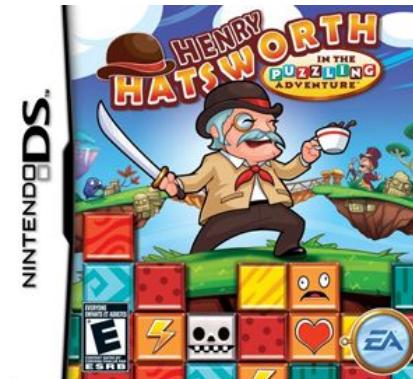


Scripting Language: Custom Instruction Set

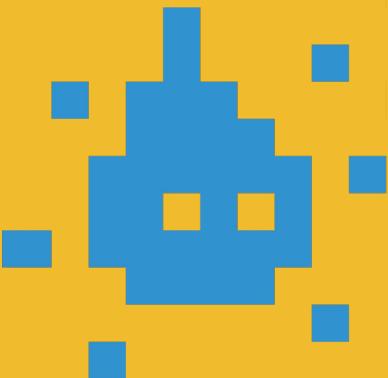
```
LITERAL 0      [0]          # Wizard index
LITERAL 0      [0, 0]        # Wizard index
GET_HEALTH     [0, 45]       # getHealth()
LITERAL 0      [0, 45, 0]    # Wizard index
GET_AGILITY   [0, 45, 7]    # getAgility()
LITERAL 0      [0, 45, 7, 0] # Wizard index
GET_WISDOM    [0, 45, 7, 11]# getWisdom()
ADD           [0, 45, 18]   # Add agility and wisdom
LITERAL 2      [0, 45, 18, 2]# Divisor
DIVIDE         [0, 45, 9]    # Average agility and wisdom
ADD           [0, 54]        # Add average to current health
SET_HEALTH    []            # Set health to result
```

Alternatively, they can be defined as a **custom limited size instructions set**. For example, a spells list for a magic game.

Henry Hatsworth in the Puzzling Adventure for **Nintendo DS** used this approach.



Scripting Language: Custom Instruction Set



The Legend of Royal Institution uses a small markup language for monsters movement system defined by a set of rules.

A monster's movement is defined by 2 parameters:

1. **the update time (in milliseconds):** how often the monster will move to the next position.
2. **a movement list:** a set of instructions defining the specific path/trajectory of the monster.

Legend of Royal Institution: Movement System

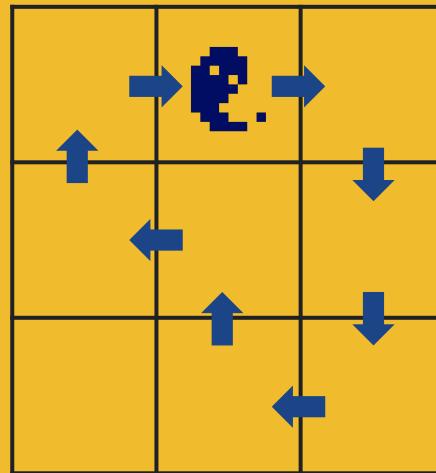
Every move command is either:

1. A direction;
2. A direction followed by **colon** and a **repeat counter**.

Allowed directions are: **left, right, up, down.**

A few tips:

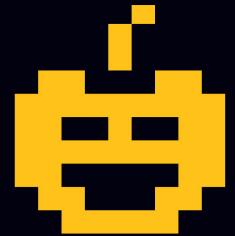
- Every path should **start** and **end** at **the exact same point**.
- Monsters can move **through the walls**.



Path is: right down:2 left up left up right

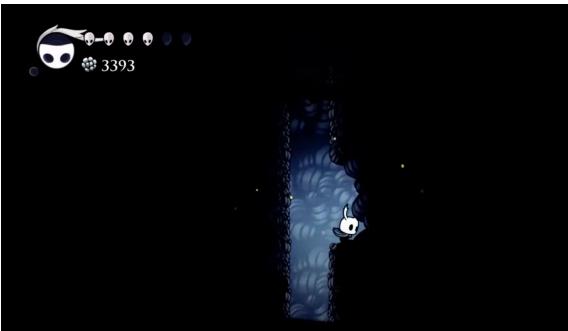
Creatures tileset





6. Garnitures

Easter Eggs: Hidden Locations



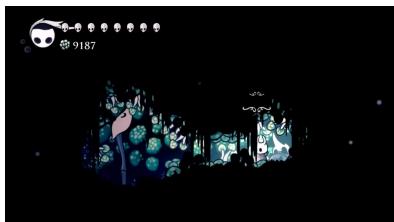
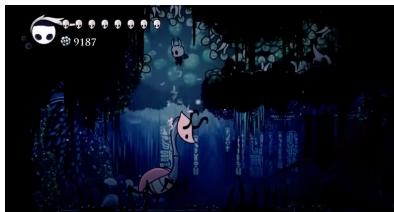
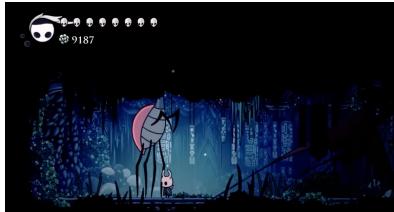
Easter eggs are secret features deliberately placed by developers to surprise and reward curious players.

Examples:

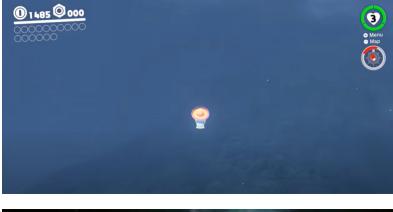
- **hidden rooms**
- references to other games
- humorous messages
- powerful items (sometimes imbalanced!)

Encourage exploration and create memorable “aha!” moments.

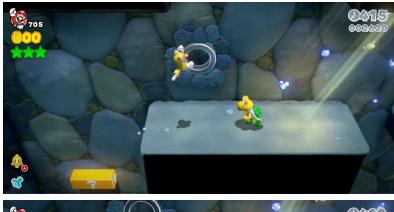
Easter Eggs: Hidden Locations



Hollow Knight



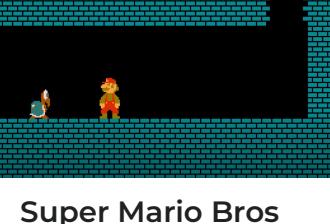
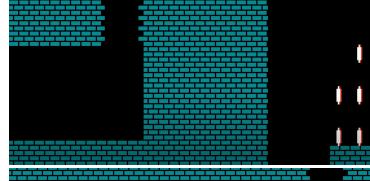
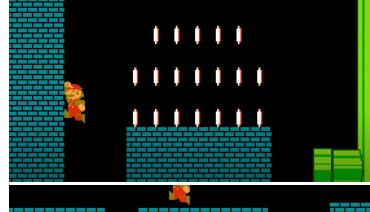
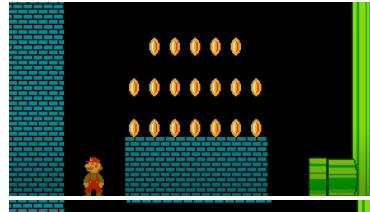
Mario Odyssey



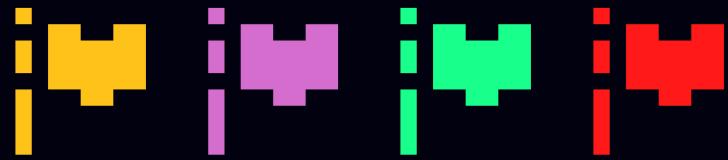
Mario 3D World



Legend Of Zelda



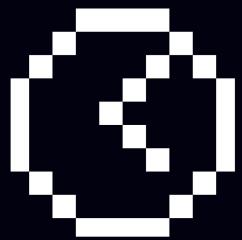
Super Mario Bros



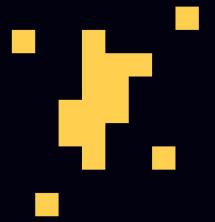
7. Flag Appearance

Flag tileset

777	778	779	780	781	782	783	784	785	786	787	788	789	790
810	811	812	813	814	815	816	817	818	819	820	821	822	823
843	844	845	846	847	848	849	850	851	852	853	854	855	856
876	877	878	879	880	881	882	883	884	885	886	887	888	889
909	910	911	912	913	914	915	916	917	918	919	920	921	922

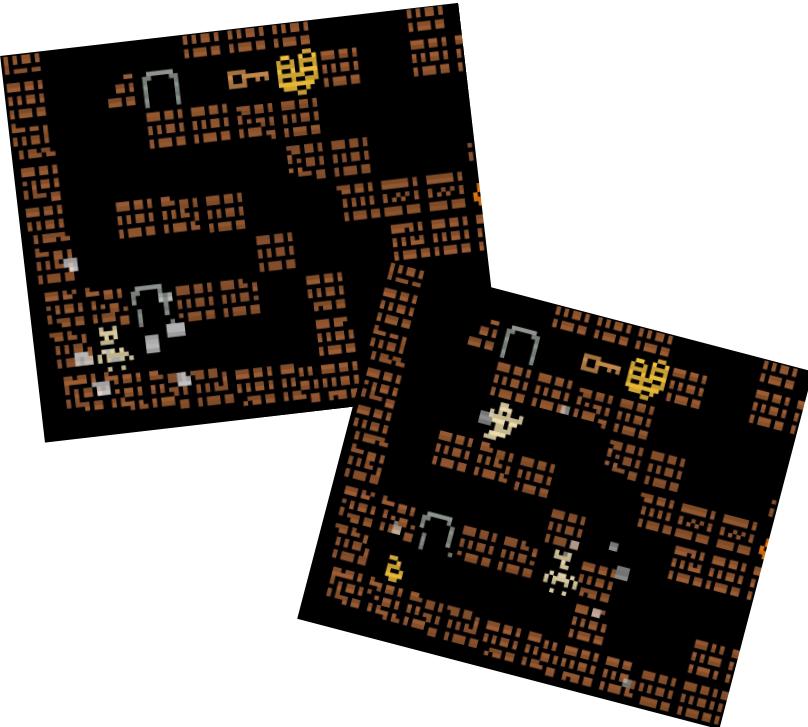


8. Level Constraints



9. Special effects

Particles



Particles are tiny graphical elements used to simulate complex visual effects such as **fire, smoke, sparks, rain, magic, etc...**

Instead of manually animating every detail, a particle system spawns and controls **hundreds** of small sprites or points, each with its own properties like size, color, lifetime, and movement.

Particles: Examples



Minecraft



Portal

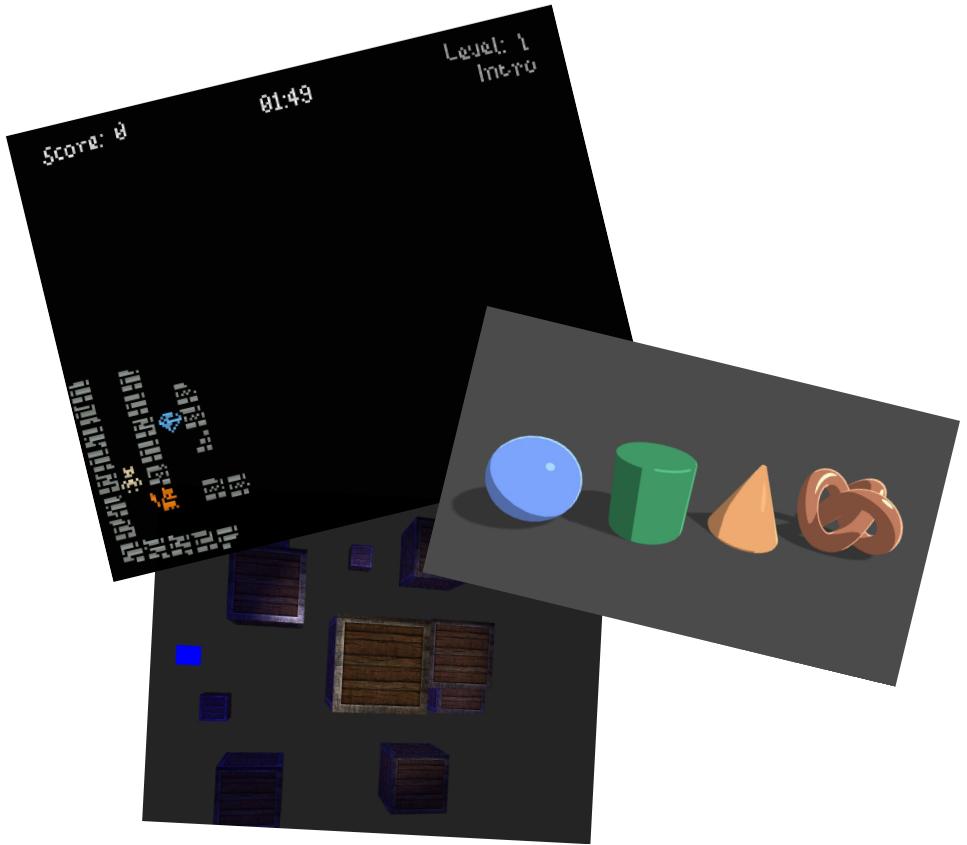


Pokemon Fire Red

Shaders

Shaders are small programs that run on the graphics card to control how objects are drawn on the screen.

Developers can create effects like: reflections, glowing surfaces, rippling water, or dramatic shadows.



Shaders: Examples



Jet Set Radio

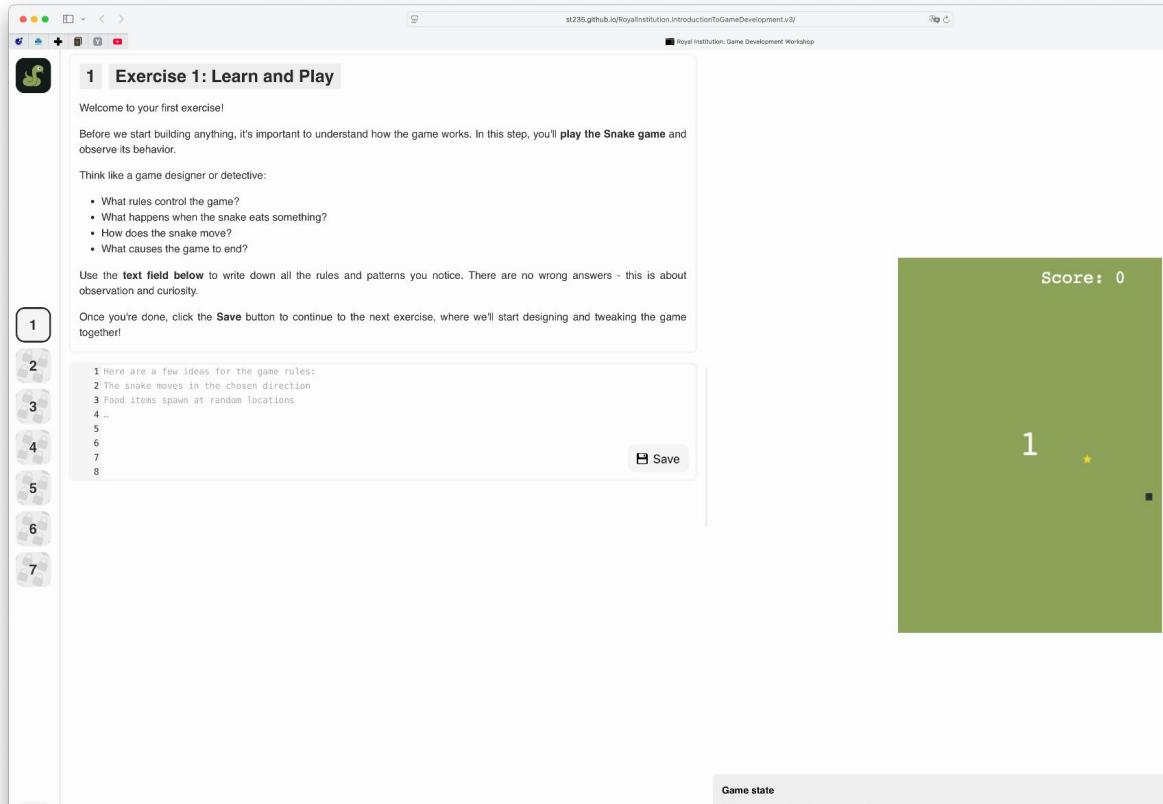


Pokemon Fire Red

Final thoughts and reflections

Explore other activities: Snake Game Design

 st235.github.io/RoyalInstitution.IntroductionToGameDevelopment.v3



The screenshot shows a web-based exercise interface for game development. At the top, a navigation bar includes a logo, a search bar, and a link to "st235.github.io/RoyalInstitution.IntroductionToGameDevelopment.v3". The main content area has a title "1 Exercise 1: Learn and Play" with a green icon. Below it, a welcome message and instructions for playing the Snake game are displayed. A list of questions for observation is provided:

- What rules control the game?
- What happens when the snake eats something?
- How does the snake move?
- What causes the game to end?

Below this, a text field for notes is shown with numbered ideas:

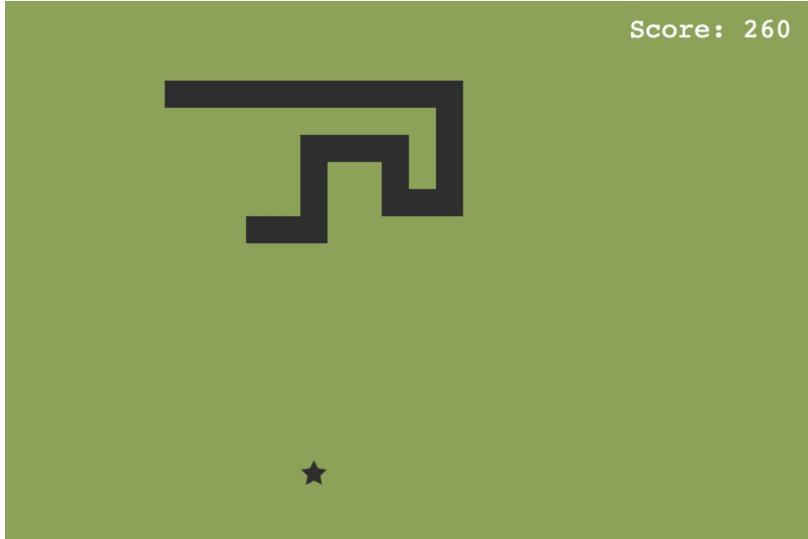
- 1 Here are a few ideas for the game rules:
- 2 The snake moves in the chosen direction
- 3 Food items spawn at random locations
- 4 -
- 5 -
- 6 -
- 7 -
- 8 -

A "Save" button is located next to the text field. To the right, a preview window titled "Game state" shows a green playfield with a yellow star-shaped food item and a small black square representing the snake's head. The score is listed as "Score: 0".

Explore other activities: Developing Snake Game

 st235.github.io/RoyalInstitution.IntroductionToGameDevelopment.v2

 https://docs.google.com/presentation/d/1Mjba1zi_cBy_Yj6le7shPmTOuNbAomML247daykqZMo/edit

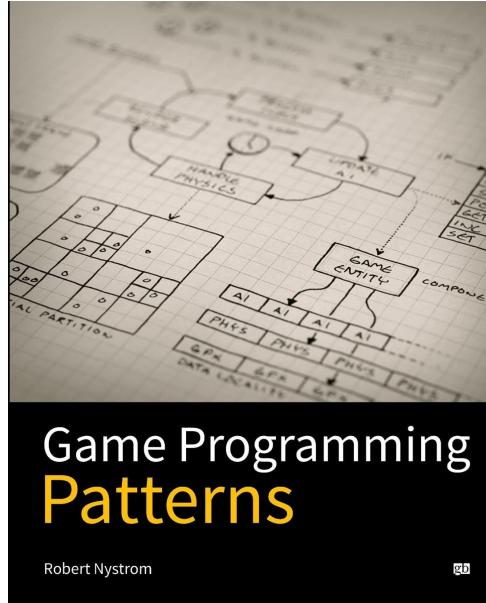


**Further
reading**

“Game Programming Patterns” by Robert Nystrom

gameprogrammingpatterns.com

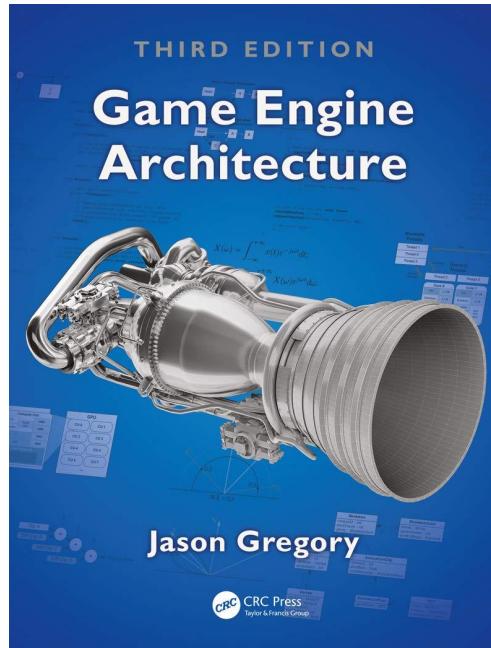
Game Programming Patterns is a collection of patterns and advice the author found in games that make code cleaner, easier to understand, and faster.



“Game Engine Architecture” by Jason Gregory

gameenginebook.com

Game Engine Architecture covers both the theory and practice of game engine software development, bringing together complete coverage of a wide range of topics. The concepts and techniques described are the actual ones used by real game studios like Electronic Arts and Naughty Dog.



CS50G: Introduction to game development

cs50.harvard.edu/games/2018/

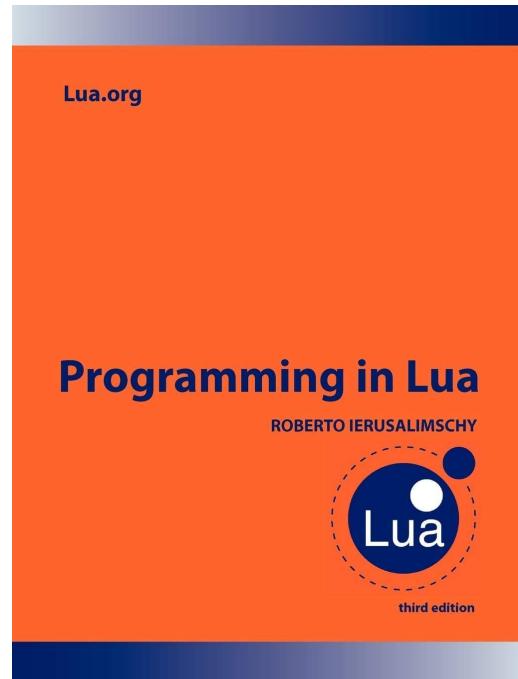


Students explore the design of such childhood games as Super Mario Bros., Legend of Zelda, and Portal in a quest to understand how video games themselves are implemented. Via lectures and hands-on projects, the course explores principles of 2D and 3D graphics, animation, sound, and collision detection using frameworks like Unity and LÖVE 2D, as well as languages like Lua and C#.

Lua programming language

 lua.org

Lua is a powerful, efficient, lightweight, embeddable scripting language. It supports several programming styles: procedural, object-oriented, functional, data-driven, and data description.



Phaser

phaser.io

Phaser is a free, open-source **JavaScript** framework for making 2D games that run in a web browser. It's especially popular for **prototyping**.



*Image is a courtesy of Phaser Studio Inc

**Thank you
for
attention!**

Alex Dadukin

Software Engineer

**For any inquiries or questions contact Ben and he will kindly
redirect them to me*