

Supplementary Materials

Basic Pattern Exercise Example

Step 1: Name a design element.

Jumping

Step 2: Name ten games that use that element—the more different ways the games use it, the better.

Donkey Kong/Jumpman, Q*bert, Super Mario Bros. (Braid as subversion?), Mirror's Edge, Gravity Rush/VVVVVV, Alice/Super Mario World/Crackdown, Guild Wars 2, Tomb Raider (reboot), Prince of Persia (second reboot), Poptropica, Super Meat Boy, Street Fighter/Soulcalibur/Devil May Cry, Doom/Quake/Splosion Man, Tribes, Assassin's Creed, Canabalt, Sonic, Trials HD

Step 3: Describe how each of those games uses the element you chose. Try not to look for a pattern yet. Focus on accurately describing the way each game uses the element you identified.

Jumping is such a fundamental design element that I tried to go back and select games that first used the mechanic in historically significant ways, as well as listing the most modern and innovative uses of the mechanic. Initially, only six examples of significance jumped into my head, so I did a small amount of research to see if there were general opinions on important uses of jumping. The willingness to pause and research a question like this is essential to the pattern development process since we all have played a limited number of games.

- Donkey Kong/Jumpman, Geometry Dash—Jumping is used to avoid enemies and traverse the 2D space.
- Q*bert—A reflex-based puzzle game made in the wake of Pac-Man, it uses jumping as its only movement mechanic.
- Super Mario Bros.—Jumping is used to avoid enemies, traverse 2D/3D space, and as a way to attack enemies.
- Mirror's Edge—This game uses first-person jumping as pure traversal.
- Gravity Rush/VVVVVV—Jumping combines with control of physics.
- Alice/Super Mario World/Crackdown—Jumping with a glide. Also, in-air control?
- Guild Wars 2—Jumping for exploration and as a puzzle. There's little need for it in world traversal, and none in combat.
- Tomb Raider (reboot)—Your ability to jump in this game is superhuman despite the more realistic tone of the game.
- Prince of Persia (second reboot)—This game features assisted jumping where another character helps you jump farther than you can alone. The game is single-player so that may be just a double jump.
- Doom/Quake/Splosion Man/Tribes—Jumping for world traversal. Jumping assisted by the physics of unrelated systems (Rocket Jumping, Ski Jumping, Bunny Hopping).
- Street Fighter/Soulcalibur/Devil May Cry—Jumping for world traversal, jumping as a combat move.
- Poptropica, Super Meat Boy—Jumping for world traversal with very unrealistic physics.
- Trials HD—Jumping in unrealistic environments with very realistic physics.
- Assassin's Creed—Jumping "on rails" for world traversal, jumping to escape enemies.

Step 4: What design problems do the games use the element to solve? Some games may use the element for one purpose while others use it for another. Many games use the elements in more than one way. Describe the problems solved by your element for each of the ten games listed in step 2.

- Navigation through the world space (all games listed)—All the games I chose used jumping as part of world traversal. At a base level, jumping gives you more movement options as a player.
- Creating a sense of autonomy in the player (all games listed)—Jumping increases a character's mobility in the game world, and that can give you a feeling of greater agency as you play. There's some subtlety in how this works, though. In some games, characters can jump in a way that more closely mirrors the real world, which can make you relate to the character more. In

other games, jumping lets the character move through the world in ways you never could in real life, which can still feel empowering because you're controlling the character.

- Creating a sense of danger for the player (Super Mario Bros., Mirror's Edge, Gravity Rush/VVVVVV, Super Meat Boy, Tomb Raider, Prince of Persia, Trials HD)—Jumping can cause death in all of these games. The ratio of how dangerous jumping is to how much it lets you traverse the world directly relates to the amount of power versus fear that you feel when you play. In a game like Super Mario Bros., you may die from jumping incorrectly, but mostly it increases your ability to navigate the world. In a game like Geometry Dash, jumping allows you to progress through the world, but it's also the main thing that causes you to die when you do it incorrectly.
- Adding variety to the ways the player can interact with the world—All games listed except Q*bert and Geometry Dash, in which jumping is the only way you move through the world. But this is particularly true in Guild Wars 2, where jumping isn't a primary world traversal tool and is mostly used in optional jumping exploration puzzles.
- Enabling player mastery through creating complicated, intricate systems that require player skill growth—All games listed. It seems like this connection is stronger the more central jumping is as a mechanic, and the more complicated and subtle the jump mechanics are.
- Enabling player mastery of game systems by creating opportunities for the player to subvert them (Doom/Quake, Tribes)—This is interesting since, in the case of these games, the mechanics were not meant to allow player subversion. Rocket jumping and ski jumping were, on some level, bugs that players found and used to enhance gameplay. The developers, recognizing the value of the bugs, intentionally incorporated them into future games.
- Character building through giving the character abilities the player lacks (Mirror's Edge, Gravity Rush, Alice, Tomb Raider, Poptropica, Assassin's Creed)—Superhuman jumping abilities help make the characters seem superhuman, but not inhuman. Improving a character's most basic movement abilities lets you relate to the character—I can jump, but not like that!—in a way a completely inhuman ability wouldn't.
- Maintaining immersion in the game world by making character abilities and movement match your understanding of how the real world works—Interestingly none of the example games above use jumping in this way, but other games do (e.g., Silent Hill 2, Flashback).
- Enhancing combat by enhancing aggressive player actions (Street Fighter/Soulcalibur, Super Mario Bros., Devil May Cry)—The jumping itself may not be aggressive, but it amplifies the character's aggressive action. A jumping punch to the head is just more impactful than a standing punch to the head.
- Step 5: Look at steps 3 and 4. Are there patterns in the ways the games use the element, and how do those relate to the problems they solve?
- Yes.
- More complex mechanics provide more opportunities for player skill. This taps into basic player needs like autonomy and mastery.
- When power has a cost, it's frightening to use. This would generate a pattern of dangerous jumping.
- Two great things that go great together, such as jumping and punching. This might be a very specific pattern about those two mechanics, or it might generalize to "movement and attack" or even to pairs of mechanics that create a player experience together that is more than either can produce alone.
- She's just like me! vs. I want to be her when I grow up! This would create a pattern about maintaining immersion by creating realistic character abilities vs. character building through superhuman abilities.

Step 5: Pick one of those patterns and describe it using the pattern template.

When power has a cost, it's frightening to use.

Step 6: You may repeat step 6 for each pattern you observed.

For this example, I will only document one pattern. In the next section, I'll show the completed writeup for the pattern I chose. Describing the other three patterns using the Pattern Template is an excellent way to practice before completing the full exercise on your own.

Research:

Creating even simple patterns like this can take a huge amount of both generalized game knowledge and knowledge of specific games. It's easy to feel like the more you know about games and the more games you have played, the better patterns you will see. That's true to a degree, but you shouldn't feel like there's no point in trying to create a pattern because you don't know enough yet. If you're a new designer, the patterns you see may be basic, but they're also probably fundamental. Experienced designers may fail to identify relevant patterns because they're too obvious. That said, when you're constructing a pattern, do try to find at least ten examples. If you can't think of that many games off the top of your head, do some research. The following sources are from the research I did when I was creating this pattern. If you look at the citations, you'll see that I'm referencing a scholarly article, an article on a popular gaming website, a Reddit post, and a fan-made games FAQ website. It's vital to evaluate your research sources and understand how much rigor or opinion there is in what you read. But it's also essential to look beyond academic analysis and consider both media perception and player experience. I do not list the background research for each pattern in the book, but I commonly read a dozen or more sources as I am investigating a possible pattern.

"The Rise of the Jump" (Butler 2014)

"What Was the First Game with a Double Jump and Why Was It Implemented?" (reddit 2016)

"You Say Jump, I Say How High? Operationalising the Game Feel of Jumping" (Fasterholdt, Pichlmair, and Holmgård 2016)

"What Game Do You Think Has Perfect Jumping Mechanics?" (GameFAQs 2018)

Notes on This Pattern

Pattern Name

As stated in the Pattern Template, the pattern name should be an "easy to remember and evocative name." There's a fine line between easy to remember and an inside joke or reference to a fleeting meme. Titles should be evocative, but they shouldn't be a reference that only you or your close friends will understand. Think about who will be reading your pattern and make sure that the title, image, and example games are understandable to the developers who will need to use the pattern.

Description and Example Games

You may notice that this pattern doesn't have anything specifically to do with jumping. That's not an accident. Many students will see jumping as the pattern itself rather than just a mechanic. Designers tend to want to take the games they've listed in steps 2 and 3 of the exercise and list those as their examples. But it's essential to look at the actual pattern you have generated and find examples of its use that are not examples of the functional element from step 1. If you see many natural examples of wildly different implementations of the pattern you've identified, it's a good indication that you've found a viable pattern. If you find examples that only relate to your starting functional element, then make sure that your pattern description reflects that narrow focus.

Pattern

Name: One of These Days That's Going to Get You Killed

Confidence: 2

Images:



Jumping over a dangerous pit and suffering from a weapon overheating are both examples of this pattern in action.

Author: Chris Barney

Design problem: How do you maintain game balance and create tension when giving the player greater power in their interactions with the game world?

Description: To maintain balance and create tension when designing character abilities, a designer may introduce consequences resulting from using those abilities. The result may be something natural, like falling into a pit of lava you try to jump over, or it may be something mechanical, like weapon heat build-up or a stamina meter.

Games that use this pattern and how:

- Super Mario Bros.—The ability to jump, which increases the character’s ability to move through the world and defeat enemies, also puts him in danger. Failing to jump over dangerous obstacles can result in Mario’s death. Similarly, failing to jump over an enemy results in the enemy killing Mario.
- Sekiro: Shadows Die Twice—Stealth-killing enemies is the easiest way to defeat them, but failing to execute a stealth kill alerts the enemy and nearby enemies and suddenly puts you in a dangerous situation.
- Anthem—Firing weapons increases their heat. Failing to manage that resource, to self-limit the damage you are doing, can result in not being able to fire your gun when you most need it.
- Zelda: Breath of the Wild—Link can climb almost anything, but he has a stamina meter, so if he tries to climb something too high he will fall. He can jump off things and glide, but if he runs out of stamina, he falls to his death.

Seed: Exercise 1: Basic Pattern Exercise—Jumping

Related patterns:

Parent patterns:

Just Look At What You’ve Become (Confidence: 2)—As you apply the pattern One of These Days That’s Going to Get You Killed you are introducing consequences for player actions and character advancement. Use this pattern to ensure that character progression is a meaningful transformation.

I’m Doing It As Hard As I Can (Confidence: 3)—The escalating difficulty introduced by this pattern creates the mechanical need for the character progression systems produced by One of These Days That’s Going to Get You Killed.

Suggested Exercises

Use Exercise 5: Functional Patterns to generate a pattern based on the functional element of mechanical character progression.

Child patterns:

And Now I Guess We Are Doing This (Confidence: 3)—When you use One of These Days That’s Going to Get You Killed to balance increases in character abilities, you create situations where the character is in peril. You can use these situations to force the player to adjust their playstyle using this pattern.

The Risk of Knowing You (Confidence: 2)—When you use One of These Days That’s Going to Get You Killed to balance increases in character abilities, you both place the character in danger and create a sense of risk for the player. Use this pattern to help you use those two effects to create a stronger bond between player and character.

Suggested Exercises

Use Exercise 4: Formal Patterns to generate a pattern based on environmental hazards.

Use Exercise 5: Functional Patterns to generate a pattern based on character stamina.

Use Exercise 5: Functional Patterns to generate a pattern based on limited ammunition.

Keywords: Character Progression, Mechanics, Balance

Figure S1. Completed pattern.

Pattern Language for Game Design

Pattern Library Games Reference

Chris Barney

One of these days that's going to get you killed.

Jumping over a dangerous pit and suffering from a weapon overheating are both examples of this pattern in action.

Author(s): Chris Barney

Groups(s): Pattern Language for Game Design

Pattern: 2

Confidence:

Created On: June 13th 2020, 11:14am

Edited On: November 16th 2020, 9:10am

Seed: Exercise 1: Basic Patterns Exercise — Jumping

Keywords: Balance Character Progression Mechanics

Design Problem: How do you maintain game balance and create tension when giving the player greater power in their interactions with the game world?

Pattern Description: To maintain balance and create tension when designing character abilities, a designer may introduce consequences resulting from using those abilities. The result may be something natural, like falling into a pit of lava you try to jump over, or it may be something mechanical, like weapon heat build-up, or a stamina meter.

Example Games:

- Super Mario Bros.**
The ability to jump, which increases the character's ability to move through the world and defeat enemies, also puts him in danger. Failing to jump over dangerous obstacles can result in Mario's death. Similarly, failing to jump over an enemy results in the enemy killing Mario. — [View Game Details](#)
- Sekiro: Shadows Die Twice**
Stealth-killing enemies is the easiest way to defeat them, but failing to execute a stealth kill alerts the enemy and nearby enemies and suddenly puts you in a dangerous situation.
- Anthem**
Firing weapons increases their heat. Failing to manage that resource, to self-limit the damage you are doing, can result in not being able to fire your gun when you most need it.

Figure S2. Pattern as it appears on the patternlanguageforgamedesign.com website. Web interface allows navigation through keywords and display of supporting information such as details about example games.

Proposed Ontology for Pattern Language for Game Design Patterns

The full proposed OWL2 ontology is presented in Manchester syntax, as that is the most human readable and compact. Following the ontology definition, a sampling of individuals is provided. All object properties are `SubPropertyOf: owl:topObjectProperty` and all data properties are `SubPropertyOf: owl:topDataProperty`. These have been omitted from the below ontology for brevity. Ontology header information has been omitted for brevity. Text has been reformatted and ordered for readability.

Classes

Class: example_game	Class: game_release	Class: pattern_seed
Class: exercise	Class: game_type	Class: pattern_states
Class: game	Class: group	Class: pattern_suggested_exercise
Class: game_available_link	Class: group_type	Class: related_pattern
Class: game_developer	Class: keyword	Class: user
Class: game_info_link	Class: pattern	EquivalentTo: author
Class: game_platform	Class: pattern_exercise	Class: author
Class: game_publisher	Class: pattern_related_pattern	EquivalentTo: user

Figure S3. Classes from the pattern language ontology.

Object Properties

ObjectProperty: hasAuthor Domain: pattern Range: user	ObjectProperty: hasGamePlatform Domain: game Range: game_platform	ObjectProperty: hasPatternState Domain: pattern Range: pattern_states
ObjectProperty: hasExampleGame Domain: pattern Range: example_game	ObjectProperty: hasGamePublisher Domain: game Range: game_publisher	ObjectProperty: hasRelatedPattern Domain: pattern Range: related_pattern
ObjectProperty: hasExercise Domain: pattern Range: pattern_exercise	ObjectProperty: hasGameType Domain: game Range: game_type	ObjectProperty: hasRelease Domain: game Range: game_release
ObjectProperty: hasGame Domain: example_game Range: game	ObjectProperty: hasGroup Domain: pattern, user Range: group	ObjectProperty: hasSuggestedExercise Domain: pattern Range: pattern_suggested_exercise
ObjectProperty: hasGameAvailableLink Domain: game Range: game_available_link	ObjectProperty: hasGroupType Domain: group Range: group_type	ObjectProperty: hasUser Domain: pattern Range: user
ObjectProperty: hasGameDeveloper Domain: game Range: game_developer	ObjectProperty: hasKeyword Domain: game, pattern Range: keyword	ObjectProperty: owl:topObjectProperty Domain: related_pattern Range: pattern
ObjectProperty: hasGameInfoLink Domain: game Range: game_info_link	ObjectProperty: hasPatternSeed Domain: pattern Range: pattern_seed	ObjectProperty: relatesTo Domain: related_pattern Range: pattern

Figure S4. Object Properties from the pattern language ontology.**Data Properties**

DataProperty: game_available_link_notes Domain: game_available_link Range: xsd:string	DataProperty: game_release_date Domain: game Range: xsd:dateTimeStamp	DataProperty: pattern_exercise_name Domain: pattern_exercise Range: xsd:string
DataProperty: game_available_link_source Domain: game Range: xsd:string	DataProperty: game_release_name Domain: game_release Range: xsd:string	DataProperty: pattern_exercise_page Domain: pattern_exercise Range: xsd:positiveInteger
DataProperty: game_available_link_url Domain: game Range: xsd:string	DataProperty: game_release_notes Domain: game_release Range: xsd:string	DataProperty: pattern_image Domain: pattern Range: xsd:string
DataProperty: game_description Domain: game Range: xsd:string	DataProperty: game_release_type Domain: game Range: xsd:string	DataProperty: pattern_image_description Domain: pattern Range: xsd:string
DataProperty: game_developer_name Domain: game_developer Range: xsd:string	DataProperty: game_type_name Domain: game_type Range: xsd:string	DataProperty: pattern_name Domain: pattern Range: xsd:string
DataProperty: game_developer_notes Domain: game_developer	DataProperty: game_type_notes Domain: game Range: xsd:string	DataProperty: pattern_seed_description Domain: pattern

Range: xsd:string DataProperty: game_image Domain: game Range: xsd:string DataProperty: game_info_link_notes Domain: game_info_link Range: xsd:string DataProperty: game_info_link_source Domain: game_info_link Range: xsd:string DataProperty: game_info_link_url Domain: game Range: xsd:string DataProperty: game_name Domain: game Range: xsd:string DataProperty: game_plarform_name Domain: game_platform Range: xsd:string DataProperty: game_platfrom_notes Domain: game_platform Range: xsd:string DataProperty: game_publisher_name Domain: game_publisher Range: xsd:string DataProperty: game_publisher_notes Domain: game_publisher Range: xsd:string	DataProperty: group_auto_join Domain: group Range: xsd:integer DataProperty: group_name Domain: group Range: xsd:string DataProperty: group_type_descrip- tion Domain: group_type Range: xsd:string DataProperty: group_type_name Domain: group_type Range: xsd:string DataProperty: pattern_confidence Domain: pattern Range: xsd:integer DataProperty: pattern_created_date Domain: pattern Range: xsd:dateTimeStamp DataProperty: pattern_description Domain: pattern Range: xsd:string DataProperty: pattern_design_prob- lem Domain: pattern Range: xsd:string DataProperty: pattern_exam- ple_game_description Domain: example_game Range: xsd:string DataProperty: pattern_exercise_de- scription Domain: pattern_exercise Range: xsd:string	Range: xsd:string DataProperty: pattern_seed_name Domain: pattern Range: xsd:string DataProperty: pattern_state_name Domain: pattern Range: xsd:string DataProperty: pattern_suggested_ex- ercise_description Domain: pattern_suggested_exercise Range: xsd:string DataProperty: related_pattern_conf- dence Domain: related_pattern Range: xsd:integer DataProperty: related_pattern_descrip- tion Domain: related_pattern Range: xsd:string DataProperty: related_pattern_type Domain: related_pattern Range: xsd:string DataProperty: video_gameplay Domain: game Range: xsd:string DataProperty: video_trailer Domain: game Range: xsd:string
--	--	--

Figure S5. Data Properties from the pattern language ontology.

Example Individuals Representing a Single Pattern and its Associations

Individual: Autonomy
Types: keyword

Individual: Bethesda_Softworks
Types: game_developer, game_publisher
Facts:
game_developer_name "Bethesda Softworks",
game_developer_notes "Large US based publisher comprising several AAA studios including. Well known for the Elder Scrolls franchise."

Individual: Choice

Types: keyword

Individual: Chris_Barney

Types: user

Facts: hasGroup Northeastern_University

Individual: Exercise_11:_Emergent_Narrative_Patterns

Types: exercise

Facts: pattern_exercise_name "Exercise 11: Emergent Narrative Patterns"

Individual: Exercise_24:_Theoretical_Patterns

Types: exercise

Facts: pattern_exercise_name "Exercise 24: Theoretical Patterns"

Individual: Greater_Choice_Requires_Greater_Motivation

Types: pattern

Individual: Northeastern_University

Types: group

Facts: hasGroupType School

Individual: Open_World_Action_Game

Facts:

game_type_name "Open World Action Game",

game_type_notes "Typically third person action game that focuses on world exploration and providing a variety of player activities."

Individual: Personal_Computer

Types: game_platform

Facts:

game_plarform_name "Personal Computer",

game_platform_notes "Personal Computer running the Windows operating system."

Individual: Published

Types: pattern_states

Facts:

pattern_state_name "Published"

Individual: School

Types: group_type

Facts:

group_type_description "Educational institution, college, or university teaching game design and producing design patterns.",

group_type_name "School"

Individual: Skyrim_Available_Link_1

Types: game_available_link

Facts:

game_available_link_notes "Steam download for the Special Edition release of the game.",

game_available_link_source "Steam",

game_available_link_url "https://store.steampowered.com/app/489830/The_Elder_Scrolls_V_Skyrim_Special_Edition/"

Individual: Skyrim_Info_Link_1

Types: game_info_link

Facts:

game_info_link_notes "",

game_info_link_source "Official Website",
game_info_link_url "https://elderscrolls.bethesda.net/en/skyrim"

Individual: Skyrim_Primary_Release

Types: game_release

Facts:

game_release_date "01/01/2011",
game_release_name "Primary",
game_release_notes "Initial release on Windows PC",
game_release_type "Primary"

Individual: The_Elder_Scrolls_V:_Skyrim

Types: game

Facts:

hasGameAvailableLink Skyrim_Available_Link_1,
hasGameDeveloper Bethesda_Softworks,
hasGameInfoLink Skyrim_Info_Link_1,
hasGamePlatform Personal_Computer,
hasGamePublisher Bethesda_Softworks,
hasGameType Open_World_Action_Game,
hasRelease Skyrim_Primary_Release,
game_description "Skyrim is the fifth instalment in the Elder Scrolls universe. It is an Action RPG in either first or third person, set in an open-world. Sandbox style nonlinear play is exhaustive. Thousands of quests and npc driven mini plots are available, as well as the main storyline quests. The primary character is fully customizable and upgrades in all skills and equipment make for a very wide variety of play experiences. Only Single-player mode is available.",
game_image "[Skyrim Image URL]",
game_name "The Elder Scrolls V: Skyrim",
video_gameplay "JSRtYpNRoN0",
video_trailer "JSRtYpNRoN0"

Individual: Three_Pillars_Relationship_1

Types: related_pattern

Facts:

relatesTo Greater_Choice_Requires_Greater_Motivation,
related_pattern_confidence 2,
related_pattern_description "When you have applied The Three Pillars of Meaning to situations where there are emergent narrative and player choices, then those choices will be meaningful. The more significant you make choices, the more of them your game will be able to support.",
related_pattern_type "Child"

Individual: Three_Pillars_Seed

Types: pattern_seed

Facts:

pattern_seed_description "What makes emergent events narratively meaningful?",
pattern_seed_name "Seed for Three Pillars of Meaning in Emergent Narrative"

Individual: Three_Pillars_Skyrim_Example

Types: example_game

Facts:

hasGame The_Elder_Scrolls_V:_Skyrim,
pattern_example_game_description "The degree to which the three pillars are present for emergent narrative elements in this game varies. At worst, the events are isolated and unrelated to the player or the world: a group of bandits in a cave with no associated NPCs or consequences for "ridding the countryside" of them. At best, all are present: killing an NPC in town results in the guards becoming hostile, the character attracting the attention of the assassins' guild, and the inability to wear holy armor due to your evil actions."

Individual: Three_Pillars_Suggested_Exercise_1

Types: pattern_suggested_exercise
 Facts:
 hasExercise Exercise_24:_Theoretical_Patterns,
 pattern_suggested_exercise_description "Use Exercise 24: Theoretical Patterns to generate a parent pattern based on the theory that The Three Pillars of Meaning in Emergent Narrative is generalizable to narrative in general."

Individual: Three_Pillars_of_Meaning_in_Emergent_Narrative
 Types: pattern
 Facts:
 hasAuthor Chris_Barney,
 hasExampleGame Three_Pillars_Skyrim_Example,
 hasExercise Exercise_11:_Emergent_Narrative_Patterns,
 hasGroup Northeastern_University,
 hasKeyword Autonomy,
 hasKeyword Choice,
 hasPatternSeed Three_Pillars_Seed,
 hasPatternState Published,
 hasRelatedPattern Three_Pillars_Relationship_1,
 hasSuggestedExercise Three_Pillars_Suggested_Exercise_1,
 pattern_confidence 2,
 pattern_created_date "12/22/2020",
 pattern_description "To allow players to construct meaningful emergent narratives, developers should provide players with context, motivation, and consequence for their actions in the game. There are many child patterns that contribute to this pattern; two are listed and 12 other possibilities are provided in the suggested exercise, and there may be more. However, each one contributes to either the context of, motivation for, or consequences of an event that could be part of an emergent narrative.",
 pattern_design_problem "Given the reality of limited resources, when creating design elements that encourage emergent narratives, designers need to maximize the narrative potential of every element. How can designers know if a given game element will contribute to meaningful emergent narratives?",
 pattern_image "pattern_image.jpg",
 pattern_name "Three Pillars of Meaning in Emergent Narrative"

Figure S6. Example Individuals from the pattern language ontology.