







Adventures in TwineSpace: An Augmented Reality Story Format for Twine

PS Berge^(✉) , Daniel Cox , Jack Murray , and Anastasia Salter 

University of Central Florida, Orlando, FL 32816, USA
hello@psberge.com

Abstract. Augmented reality games have moved into the mainstream thanks to headline-catching titles such as *Pokémon GO* (2016) and *Harry Potter Wizards Unite* (2019). However, these games represent only a small portion of what is possible in this space, as demonstrated by emerging independent and serious game efforts in this area. Such development is hindered by the challenges facing individual and casual designers in finding AR tools and accessible engines. By exploring the legacy of casual, “low-friction” AR development and text-game design, we argue previous attempts to integrate narrative engines with augmented reality (e.g. Argon.js + Twine, ARIS) and image recognition (Vuforia, AR.js) have yet to meaningfully center casual development. With this in mind, we present a working prototype called TwineSpace combining popular hypertext authoring tool Twine 2 with open-source augmented reality tools A-Frame and AR.js that opens the possibilities for play within the world of spatial narrative. By combining open-source solutions, we hope this new authoring experience can help improve spatial storytelling for indie game designers, teachers, storytellers, and casual creators alike.

Keywords: Twine · Augmented Reality · Story Format · Prototype

1 Introduction

1.1 Augmented Reality for Casual Creators

Augmented reality (or AR) gaming captured the public imagination in 2016. In what has been called the “summer of *Pokémon Go*”, thanks to a moment of global player enthusiasm, upwards of 45 million players engaged with augmented reality at the same time [12, 25]. *Pokémon Go* (2016) trained a wide base of players in a previously obscure genre, creating an incentive for movement and travel as the center of an experience design. This success should have inspired endless imitators in the months and years after 2016, but screen-based augmented reality has grown more slowly. Opportunities for locative play remain limited and tend towards large-scale global productions. *Zombies, Run!* (2012) focuses on movement. *Jurassic World Alive* (2018) brings roaming dinosaurs to the neighborhood. *Minecraft Earth* (2019) offered a short-lived augmented

world focused on crafting. Such experiences have elements of interactive narrative (particularly through quests and environmental design) but lack depth or complexity in their implementation of story experience.

In this paper, we introduce TwineSpace, a story format for creating interactive digital narrative experiences using the authoring tool Twine designed specifically for casual creators interested in making augmented reality projects [7]. We briefly review how others have attempted similar goals, what success they found, and how each project ultimately ended. We then examine what can be learned from these previous attempts and how TwineSpace builds on these lessons. Finally, we examine the potential for digital storytelling with augmented reality and how TwineSpace helps build toward exciting new prospects for engaging the border between physical and virtual materiality. Ultimately, we argue authoring experiences like TwineSpace might enable new kinds of stories and encourage novice designers to experiment with interactive narrative through augmented reality digital storytelling, offering further potential for exploring the experiential and educational usage of the form.

2 Building Interactive Digital Narratives in AR

Twine has an established legacy among independent, student, and hobby creators. Scholars and designers have described how Twine’s ease-of-use has helped marginalized creators build experimental and personal interactive digital narratives [6], and Twine has been pivotal in the classroom for teaching interactive digital narrative design [8]. Merritt K notes how the Twine’s continued success stems from both the accessibility of the platform, and that “Twine has been the site of an incredible artistic flourishing at the intersections of digital games and fiction: a rebirth of hypertext. People who might never otherwise make a videogame make them with Twine” [9]. There have been several previous efforts to incorporate augmented reality into Twine or similar tools; each attempt was marked with limited success and frequently a lack of long-term support. In this section, we provide a brief overview of these previous projects and libraries.

2.1 Twine and the Geolocation API

In 2015, Shawn Graham documented a process for incorporating location-based play with the SugarCube story format in Twine 2. Using the Geolocation API already available in most web browsers, Graham created a small software library to communicate between the web browser and Twine story to design augmented reality experiences with Twine. As users neared specified coordinates, the game could trigger changes in the passages. Initially developed for a workshop, Graham later documented this approach along with a criteria for building future AR libraries using Twine labeled “low friction” [10]. This criteria included making the creation experience “like a text-editor”, making sure “the coding is minimal,” and allowing the produced artifacts to be “fairly accessible no matter what device the user has” [10]. While one of the earliest documented attempts to combine AR and text-game design in Twine, Graham’s library comes with two major constraints. The first is its reliance on external code: users looking to use the library must manually add the library’s code to an existing story in Twine. The second constraint is

in its purposefully limited design: Graham’s framework was not intended to use the user’s camera or visual dynamics. Rather, it was entirely text-based. Building off the Geolocation API in web browsers, the library can only react to near-exact geolocation positions, requiring a user to have these coordinates ready when working on a story.

2.2 The Argon Browser

In 2017, the Augmented Environments Lab (AEL) at Georgia Tech published Argon, an AR-enabled browser application created for mobile devices. Using the image recognition tool Vuforia, this browser allowed designers to create geolocational experiences and use image and marker tracking. The browser supported image tracking and marker-based experiences for spatial narratives without relying on coordinate-driven geolocation. Like with Graham’s process [10], AEL also built off the SugarCube story format in Twine [11]. By adding Argon-specific functionality (called macros) for SugarCube, the authors were able to create projects using Twine. Using this approach, demonstrations showcased image tracking to build dynamic spatial interactions such as an AR “magic book” projects to display messages and three-dimensional models when the device’s camera detected certain markers [11]. Unfortunately, while documentation on Argon and macros of the SugarCube story format still survive, the mobile browser itself does not, and so these libraries and demonstrations are largely deprecated.

2.3 A-Frame and Twine

In 2018, Ada Rose Cannon published an essay describing a proof-of-concept game combining the virtual and augmented reality web framework A-Frame with the Harlowe story format in Twine [13]. A-Frame is an open-source WebXR framework for building virtual reality and augmented reality experiences in HTML based on JavaScript library Three.js. Unlike other previous solutions, Cannon’s approach was to bypass the story format functionality and to write additional code to listen for both story navigation and user interaction events in the browser. In this approach, the JavaScript library jQuery, built into the Harlowe story format, was used to intercept possible user interactions and manipulate how the story was presented to a user by triggering events in the story format as if a user had done them directly. Using A-Frame to create a three-dimensional scene, a user could click on different elements to prompt an on-screen avatar to navigate through the space between them or move between story segments. While highly specific to the solution created by Cannon, the approach to using A-Frame and manipulating how a user perceived a story establishes a strong foundation for additional work using a similar process.

2.4 Proprietary Solutions

There is also a legacy of AR development tools aimed at casual creators (and especially educators) outside of Twine. One of the earliest efforts to inspire independent AR creators using mobile devices was ARIS, a platform for “mobile games, tours and interactive stories” conceived in 2008. Developed by the Field Day research lab at University of

Wisconsin-Madison, ARIS supported multiple spatial configurations, including geolocation, map-based, and Bluetooth beacons. Its proprietary authoring tool was designed around a scene-based framework with additional support for items, inventories, and other adventure game mechanics [14]. ARIS boasted an impressive base of tens of thousands of creators at its height [14]. Despite its current deprecation, ARIS attracted much scholarly attention about the applicability of augmented reality in narrative game spaces in the classroom [15]. A more recent effort includes Niantic’s 8th Wall, which launched in 2018. Perhaps the closest commercial solution to a tool for casual creativity, it is built on the open-source framework A-Frame. 8th Wall allows users to create interactive experiences including image tracking, three-dimensional model-viewing, face filters, and interactive scenes. While popular with some audiences, 8th Wall requires users to code scenes directly and additionally requires a monthly service fee for the tool and online deployment [16].

Admittedly, these solutions exist in the shadow of popular, but high-barrier tools for AR creation connected to game engines like Godot, Unity and Unreal. Plugins for these engines often require moderate coding knowledge and fail the “low friction” criteria. These commercial plugins and libraries frequently cost money for both software and additional hardware devices. For example, Unity’s in-engine AR solution and numerous asset store plugins for AR are comprehensive, but require preexisting knowledge of a complex game engine, higher coding literacy, and potentially paying expensive licensing fees. Such platforms also forefront models and genres of play outside of interactive digital narratives, and instead focus on games with themes of combat or conquest [8]. While we recognize the ubiquity of these solutions for AR development, we argue there exists an important need for more low-barrier tools for encouraging casual creativity, particularly in the interactive narrative design space. As “tools impact the stories created using them in ways that are not obvious,” [17] rethinking our available tools has the potential to reshape how we approach developing AR interactive digital narratives going forward.

3 TwineSpace Overview

Each previous attempt to create a tool for augmented reality projects in Twine has seen success. In particular, the “low friction” criteria established by Graham provides a useful guide for developing new projects. However, each also ended when factors within the design of the projects affected their continued longevity. Users should not have to bypass story format functionality, as was the case with Cannon [13], nor use additional, external code, as was the case with Graham’s solution [10]. Creating an entire web browser dedicated to augmented reality projects like Argon has the greatest potential for many use cases [11], but carries with it an issue of projects no longer working when support for the application ends. There are multiple lessons to draw from these previous efforts:

- Proprietary frameworks, especially those linked to university labs and external funding, often struggle to maintain support and longevity.
- Reliance on private libraries and frameworks frequently contribute to shorter project lifespans.

- There remains an interest in creating AR experiences with Twine.

With these lessons in mind, we present our ongoing work to develop TwineSpace, an open-source Twine story format designed specifically for augmented reality experiences [7]. TwineSpace builds from both Graham’s work on “low friction” augmented reality, while expanding Cannon’s work combining Twine and A-Frame. Our goal with TwineSpace is to bridge two extant casual creator spaces (text-game design and WebXR) while avoiding the pitfalls of previous efforts to build casual AR solutions.

TwineSpace represents a kind of compromise between a dedicated browser and needing to include external code libraries themselves: a new story format specifically designed for augmented reality projects. It builds from previous approaches for creating augmented reality projects in Twine. TwineSpace is a story format based on jQuery, A-Frame, and AR.js (a web library for creating augmented reality projects). Most importantly, it provides a “translator” interface between the text of Twine passages and HTML elements processed by A-Frame.

Instead of asking users to potentially copy and paste code into their stories to enable new functionality, TwineSpace is designed to allow users to create AR experiences from the start. Creators can easily add the story format TwineSpace to Twine by selecting Formats from the story listing, using the Add a New Format tab, and pasting a link from the GitHub repository. TwineSpace incorporates several features akin to other popular story formats such as Harlowe [18] and SugarCube (such as “script” tags for passages) [19], but is based on another story format, Snowman [20]. Our decision to extend the Snowman story format was one of convenience: one of the coauthors of this project (Daniel Cox) designed and maintains the current version of the Snowman story format. Both Graham’s work in 2015 and the Vuforia-based mobile browser Argon in 2017 used SugarCube, a popular story format for authors in Twine [21]. TwineSpace, however, is an entirely new story format based on Snowman—following in the well-defined footsteps of other projects based on, such as Trilogue [22] and BotscripTEN [23]. TwineSpace also takes advantage of the technical details released as part of the Twine Specification repository first published in early 2020 [24].

In this section, we characterize TwineSpace and describe how authors might make use of AR solutions in their projects.

3.1 Macros

Many Twine story formats include named functionality using special syntax or characters. When these appear in the content of a Twine story, the story format packaged with the project performs specific actions such as creating a new variable or changing how content is presented to a user. Following the pattern established by the story format Harlowe, TwineSpace provides macro functionality using parentheses around the use of the macro with a colon after its name. The author can also add additional values associated with the macro after the colon. As TwineSpace is built on A-Frame, this allows users to quickly use macros to represent HTML elements with special meaning to A-Frame such as (box:) to represent the equivalent A-Frame element of <a-box>. At present, TwineSpace supports the macros shown in Table 1.

Table 1. Sample TwineSpace Macros

(box:) (cone:) (ring:) (sphere:) (torus:) (octahedron:) (icosohedron:)	Creates a three-dimensional primitive of the respective type. <i>No arguments required</i>
(circle:) (triangle:) (plane:)	Creates a two-dimensional primitive of the respective type. <i>No arguments required</i>
(image:) (video:) (curvedimage:)	Displays an image or video on a two-dimensional plane. <i>Source required</i>
(text:)	Creates three-dimensional text. <i>Value required</i>
(glTF-model:) (obj-model:)	Displays a three-dimensional model in glTF or OBJ format. <i>Online source required</i>
(videosphere:)	Displays a 360-degree video, projected onto a sphere around the camera. <i>Source required</i>

Macros streamline A-Frame’s HTML-based entities into a quick shorthand for users. These macros allow creators to quickly ‘set the scene’ with custom shapes and models. In each case, these macros can be expanded to contain any of the arguments specified in the A-Frame documentation for the given primitive—but these arguments are not required. For example, to creating a videosphere requires a source video URL as an argument, but it does not require any position or rotation information. Carrying over from A-Frame’s default placement, TwineSpace will also automatically place new objects at the center of the scene unless given a specific position. In this way, authors can determine how much or how little they want to customize these primitives by adding as few or as many macros as they wish.

A-Frame uses the metaphor of a scene to describe a set of three-dimensional entities. In TwineSpace, this metaphor is extended. Each time a user navigates to a new part of a story, its content is parsed. If specific macros are found, a new A-Frame scene is generated, allowing each part of a story, what Twine calls a passage, to be a separate “scene” for both the creator and user experiencing it.

3.2 Embedding Passages

While macros allow creators to quickly generate and customize individual A-Frame elements within passages, we have also developed a solution for pasting entire A-Frame scenes as HTML directly into Twine. This is handled by a unique macro: (*embed-scene: [name]*) where *[name]* is the title of another Twine passage in which the user has placed the complete HTML of an A-Frame scene. Through the (*embed-scene: [name]*) macro,

a creator can define all the elements for a scene as part of one passage and “embed” it in another. For example, one could include the following macro in a text passage:

```
(embed-scene: "Another")
```

When launched, TwineSpace would find a separate passage titled “Another” and treat its content as A-Frame HTML elements where the macro was embedded:

```
<a-scene>
  <a-entity position="-6 -1 0">
    <a-box position="0 0.2 0" color="#4CC3D9"></a-
box>
  </a-entity>
</a-scene>
```

Our hope is this feature will be useful for both established A-Frame users who want to incorporate Twine into their own projects as well as Twine creators interested in using widely available scene templates in their projects. Users unfamiliar with HTML can still generate simple primitives with macros, but embedded scenes provide a more robust solution for those who want to generate complexity (see Fig. 1).

TwineSpace also features the ability to designate certain passages as containing JavaScript code by adding the tag `script` to the passage during story editing. These special passages, like scene passages, are not linked to the story but instead are run before the story started, allowing a more advanced user to define additional functionality in multiple places, if wanted. In this way, advanced JavaScript users who wish to change how A-Frame works more directly can do so by adding code into these special passages. While we expect this to be beyond the scope of what most users will require for their projects, it remains a useful solution for advanced users.

4 Re-imagining Casual AR with TwineSpace

Augmented reality projects invite new examinations of the ways in which movement through space and the digital gaze can complicate the borders between physical and digital materiality. At the intersection between narrative gaming in the form of the hypertext authoring tool Twine and existing open source libraries for creating WebXR experiences, A-Frame and AR.js, TwineSpace seeks to encourage greater explorations of space with image tracking, what Hjorth et al. (2020) have called “a mode of hybridized digital wayfaring” [25]. Based on specific user cases, TwineSpace seeks to meet the demands of a “low friction” design for causal creativity while also hoping to open new possibilities for digital storytelling. In this section, we examine the design goals of the project, its known technical limitations, and how TwineSpace and other solutions point toward new approaches in augmented reality digital storytelling.



Fig. 1. A passage in TwineSpace displaying a pre-built A-Frame demo implemented using the script passage function on mobile. The user can move the camera around to view the 3D scenes while interacting with the narrative elements in Twine.

4.1 User Cases

TwineSpace’s design started with three user cases based on our previous research. Many of the previous attempts to bridge AR and Twine focused on three-dimensional models, effects, and other mixed media in their demonstrations. Our own solution, we realized, must also afford similar possibilities with media. This led us to the creation of the first user case:

1. *I am a creator. I want to include AR media in my interactive fiction.*

The inclusion of the (*embed-scene:*) macro allows creators to remix existing A-Frame projects with photospheres, three-dimensional models, and image tracking directly in TwineSpace. Combined with other macros, creators can then add new shapes and models to a project without needing to edit or re-write large amounts of HTML to produce a story in Twine with augmented reality experiences.

Building from the first user case, the second focused on classroom usage around augmented reality. While TwineSpace is not designed exclusively for classroom contexts, we wanted to make sure the approaches we took matched how Twine was already used in pedagogical spaces as seen in previous scholarship and on platforms like ARIS [15]. Building on this, we developed our second user case:

2. I am a teacher. I want to work with my students to create simple AR experiences in a classroom using Twine.
 With this in mind, we wanted to make sure TwineSpace would be effective for similar educational and industry practices, with an attention to the casual creator across fields. This involved a) maintaining our emphasis on low-friction tools for design, b) focusing on marker-based and image-tracking models rather than geolocational models (which are more conducive to building spatial projects on a single campus), and c) ensuring our tools relied on no paid licensing or libraries. At the same time, TwineSpace provides a tool for rapidly outlining or creating a workable digital mockup (or ‘whiteboxing’) of larger AR games prior to more fiscally committed development.
 Finally, our last user case emphasized the importance of image-tracking solutions, as demonstrated by Argon and previously discussed AR games which relied on image tracking:
3. I have a mobile device in a specific space. I want to see locative image markers as part of my interactive fiction experience.
 We anticipated the low-barrier and open-source nature of TwineSpace would have potential for educators, but we also hope physical spaces such as libraries and museums as well as indie artists working in installations and interactive fiction creators could use image-tracking solutions in combination with Twine to create spatial narratives. We believe image-tracking to be a more generous solution for narrative AR than geolocational solutions, which often only work at scale and make distribution difficult.

4.2 Technical Limitations

There remain several technical limitations we are aware of for the TwineSpace story format:

- A-Frame relies on WebGL support to display three-dimensional content. Some mobile devices have limited or no WebGL functionality.
- Versions of the Safari web browser found on iPad and iPhone devices have higher default security restrictions than other web browsers. Some users may not be able to access their camera to process image markers or view certain WebGL content. In these cases, using an alternative web browser such as Chrome on the same device will provide access and better process content.
- TwineSpace currently supports very limited communications between authored content and the current A-Frame scene. While it is possible to setup event handlers and more complex interactions (i.e. touch, timers, locomotion controllers, etc.) all of this would need to be added by a creator themselves using JavaScript.
- TwineSpace inherits another limitation of A-Frame: the lack of a meaningful scene-builder. While the default A-Frame scene inspector can be used to modify existing scenes and templates, there is currently no tool to design and build scenes in a visual inspector.

Finally, it is worth noting how A-Frame can be used as a comprehensive WebXR framework capable of supporting both AR and VR projects. Because of this support,

TwineSpace can be used to build VR projects. However, because such projects frequently require specialized hardware or configurations, we excluded such a focus as it was against the “low-friction” and casual creator uses case behind the design of TwineSpace. VR projects are certainly possible in the future with TwineSpace, but were not within the scope of the current project.

4.3 Built on Open Source

Much of the research on previous approaches on tools for using augmented reality in Twine showed solutions built on proprietary code or commercial libraries. In these cases, once support for the tools ended, so too did the ability for any projects using them to run. Understanding this risk, we followed in the patterns established by solutions from Graham [10] and Cannon [13] for using open-source libraries and documenting not only our code, but the processes around them as well. Active investment in any coding project extends their lives beyond their initial publishing, and this is even more true for open-source projects [26]. Building from established open-source libraries with existing documentation and communities gives us the best chance for TwineSpace’s longevity in the cases of possible future abandonment or a change in developers in the future [27, 28].

4.4 Making New Space for Casual Augmented Reality Storytelling

By building from existing solutions to create augmented reality experiences, the story format TwineSpace provides a flexible template for combining the best of existing storytelling approaches for both Twine and AR. Relying upon the ubiquity of mobile devices, augmented reality storytelling unlocks new and exciting considerations of how the metaphors of space, movement, and gaze can be rethought as understood through the lens of a user’s digital camera. Projects like The Augmented Environments Lab’s image-based ‘storybook’ show how augmented reality can be used with Twine to change the perception of the “physical” page for a reader [11]. Similarly, grave snail games’ *Fish & Dagger* (2021) challenges digital stories as presented on screens, asking readers to find hidden messages within the in-game text by ‘scanning’ the page with their phone, as shown in Fig. 2 [29].

While Twine creators have often found sources of additional interactive content through Bitsy (a casual creator tool for building HTML minigames) [30] and JavaScript [31], AR media offers new possibilities for building additional interactive elements, art, and immersive storytelling. For example, videospheres have proven to be a simple yet effective way to complicate the player’s viewing experience with mobile media: placing the player inside a video (and encouraging them to pan around and explore while reading). In TwineSpace, these can be implemented quickly with an existing video and single line of code, as shown in Fig. 3.

The introduction of AR to Twine games recontextualizes spatiality. Creators can orient (or disorient) users by projecting an interpretive, ‘haptic’ layer onto physical space [32]. This can be done either by mapping these spaces onto real-world spaces (often campuses, landmarks, and natural areas) or by designating locations within a certain proximity of the user, such as in micha cárdenas and the Creative Realities Studios’ AR

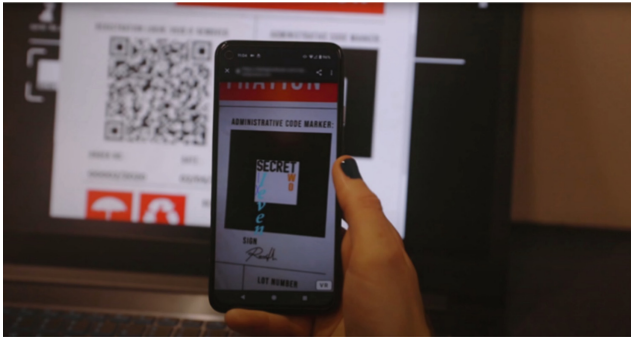


Fig. 2. grave snail games' *Fish & Dagger* includes an image-tracking AR puzzle [24]. This game served as a model for the narrative possibility of marker-based AR in Twine stories.



Fig. 3. A recent build of TwineSpace showing a videosphere with an interactive Twine passage, optimized for mobile viewing. The videosphere plays looping clips of the underside of a waterfall while the user interacts with the narrative in the textbox at the bottom of the screen.

game *Sin Sol* (2018) [33]. The dynamic possibility of displaying rendered content onto images encourages different kinds of navigation: images might be scattered across a

physical space (such as posters or banners in a library or museum) or another kind of paratext (such as a website or book). While Twine games are already about navigating hypertext, our hope is TwineSpace’s AR media and marker tracking affordances further enable designers to play with the dual-navigation of space and narrative [12].

5 Conclusion

While TwineSpace still remains in-development, it is informed by a history of AR interventions and offers new possibilities for the accessible development of augmented reality interactive digital narratives. By combining existing open-source libraries and creating a dedicated story format for Twine, we hope TwineSpace can be a springboard for additional bridges between WebAR communities and interactive digital narrative design. As with other projects emerging from ICIDS and casual creator communities, this framework is intended to be adopted, hacked, and reimaged by any user so inclined. We have drawn extensively from previous efforts to engage AR and narrative gaming, and hope that by relying on open-source tools already popular with casual creators (and educators from this community and elsewhere, including electronic literature and digital humanities spaces), TwineSpace might be better positioned to expand ongoing design work and attract newcomers.

While the TwineSpace story format is released and functional [7], our next steps forward involve engaging with the Twine community and WebAR enthusiasts to receive more feedback and extend functionality. At the same time, we are currently building documentation, tutorials, and examples to hopefully make TwineSpace more accessible to those unaccustomed to AR or who have never used a modified story format in Twine before.

While AR continues to be defined by headline-catching, global-scale collection-games, we hope this project emphasizes the important potential for new, low-friction AR game development tools supporting casual creativity. While we recognize TwineSpace is merely a tool, and that bridging hypertext and AR does not imply novel narrative structures, we also recognize the ways previous expansions of Twine’s framework have galvanized creators and ushered forth new hypertext [34] and digital art [35]. What kinds of stories might independent designers tell with the affordances of AR media? Perhaps more importantly, how might augmented reality—still largely associated with gigantic, global gaming franchises—be transformed by the possibilities of casual creation? It is our hope that TwineSpace might support these new inquiries as it supports the projects exploring new approaches to interactive digital narrative design, allowing us to envision the next opportunities and challenges at this “milestone moment” in the development of the form [36].

References

1. Bonsignore, E.M., Hansen, D.L., Toups, Z.O., Nacke, L.E., Salter, A., Lutters, W.: Mixed reality games. In: Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work Companion, pp. 7–8. Association for Computing Machinery, New York, NY, USA (2012). <https://doi.org/10.1145/2141512.2141517>

2. Donald, N., Gillies, M.: Novel dramatic and ludic tensions arising from mixed reality performance as exemplified in better than life. In: Schoenau-Fog, H., Bruni, L.E., Louchart, S., Baceviciute, S. (eds.) ICIDS 2015. LNCS, vol. 9445, pp. 297–308. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-27036-4_28
3. Arenas, L., Zarronandia, T., Díaz, P., Aedo, I.: A platform for supporting the development of mixed reality environments for educational games. In: Zaphiris, P., Ioannou, A. (eds.) LCT 2015. LNCS, vol. 9192, pp. 537–548. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-20609-7_51
4. Compton, K., Mateas, M.: Casual creators. In: Proceedings of the Sixth International Conference on Computational Creativity, pp. 228–235 (2015)
5. Green, D., Hargood, C., Charles, F.: Contemporary issues in interactive storytelling authoring systems. In: Rouse, R., Koenitz, H., Haahr, M. (eds.) ICIDS 2018. LNCS, vol. 11318, pp. 501–513. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-04028-4_59
6. Anthropy, A.: Rise of the videogame zinesters: how freaks, normals, amateurs, artists, dreamers, drop-outs, queers, housewives, and people like you are taking back an art form. Seven Stories Press (2012)
7. Cox, D., Berge, P., Murray, J., Salter, A.: TwineSpace (2022). <https://github.com/videlais/twine-space>
8. Barbara, J.: Twine and DooM as authoring tools in teaching IDN DESIGN of LudoNarrative dissonance. In: Bossler, A.-G., Millard, D.E., Hargood, C. (eds.) ICIDS 2020. LNCS, vol. 12497, pp. 120–124. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-62516-0_11
9. Kopas, M. (ed.): Videogames for humans: twine authors in conversation. Instar Books (2015)
10. Graham, S.: Tutorials for Augmented Reality & Archaeology (2021). <https://github.com/shawngraham/ar-archaeology/blob/42a93b0a930b17184d4012f10ed800c2887f3bfa/workingshop%20materials/Hacking%20Twine%20to%20make%20a%20location-based%20game.md>
11. Understanding AR (2022). <https://github.com/argonjs/understanding-argon-twine>
12. e Silva, A.D.S. and Glover-Rijkse, R. (eds.): Hybrid Play: Crossing Boundaries in Game Design, Player Identities and Play Spaces. Routledge, London (2020). <https://doi.org/10.4324/9780367855055>
13. Cannon, A.R.: How I built a game in a week. <https://medium.com/samsung-internet-dev/how-i-built-a-game-in-a-week-5810b1197686>. Accessed 07 Feb 2022
14. Holden, C.L., Gagnon, D.J., Litts, B.K., Smith, G.: ARIS: an open-source platform for widespread mobile augmented reality experimentation. In: Neto, F.M.M. (ed.) Technology Platform Innovations and Forthcoming Trends in Ubiquitous Learning, pp. 19–34. IGI Global, Hershey, PA, USA (2014). <https://doi.org/10.4018/978-1-4666-4542-4.ch002>
15. Perry, B.: ARIS: a tool to promote language learning through AR gaming. *CJ* **35**, 333–342 (2018). <https://doi.org/10.1558/cj.36318>
16. 8th Wall | Pricing and Plans - WebAR, AR, WebVR Developer Tools and SDK. <https://www.8thwall.com/pricing>. Accessed 21 July 2022
17. Kitromili, S., Jordan, J., Millard, D.E.: How do writing tools shape interactive stories? In: Rouse, R., Koenitz, H., Haahr, M. (eds.) ICIDS 2018. LNCS, vol. 11318, pp. 514–522. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-04028-4_60
18. Hansen, M.: modality/harlowe (2022). <https://github.com/modality/harlowe>
19. Edwards, T.M.: SugarCube v2 (2022). <https://github.com/tmedwards/sugarcube-2>
20. Cox, D.: Snowman (2022). <https://github.com/videlais/snowman>
21. Cox, D.: Static Echoes: Exploring the Life and Closing of the Free Twine Hosting Service, philome.la. Electronic Literature Organization Conference 2020 (2020)
22. Kemenade, P. van: Trialogue (2022). <https://github.com/phivk/trialogue>
23. Heuser, J.: BotscripTEN (2022). <https://github.com/jakobo/botscripTEN>

24. Twine Specifications (2022). <https://github.com/iftechfoundation/twine-specs>
25. Hjorth, L., Richardson, I.: Pokémon GO: Mobile media play, place-making, and the digital wayfarer. *Mob. Media Commun.* **5**, 3–14 (2017). <https://doi.org/10.1177/2050157916680015>
26. Avelino, G., Constantinou, E., Valente, M.T., Serebrenik, A.: On the abandonment and survival of open source projects: An empirical investigation. In: 2019 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM), pp. 1–12. IEEE, Porto de Galinhas, Recife, Brazil (2019). <https://doi.org/10.1109/ESEM.2019.8870181>
27. Aberdour, M.: Achieving Quality in Open-Source Software. *IEEE Softw.* **24**, 58–64 (2007). <https://doi.org/10.1109/MS.2007.2>
28. Kaur, R., Kaur, K.: Insights into developers' abandonment in FLOSS projects. In: Nagar, A.K., Jat, D.S., Marín-Raventós, G., Mishra, D.K. (eds.) *Intelligent Sustainable Systems. LNNS*, vol. 333, pp. 731–740. Springer, Singapore (2022). https://doi.org/10.1007/978-981-16-6309-3_69
29. Fish & Dagger by grave snail games. <https://gravesnail.itch.io/fish-and-dagger>. Accessed 27 July 2022
30. The Tower by communistsister. <https://communistsister.itch.io/the-tower>. Accessed 27 July 2022
31. my father's long, long legs by ztul. <https://ztul.itch.io/mfilll>. Accessed 27 July 2022
32. Pozo, T.: Queer Games After Empathy: Feminism and Haptic Game Design Aesthetics from Consent to Cuteness to the Radically Soft. *Game Studies*, p. 18 (2018)
33. Sin Sol / No Sun | micha cárdenas. <https://michacardenas.sites.ucsc.edu/sin-sol-no-sun/>. Accessed 25 Apr 2022
34. Millard, D.E., Hargood, C.: Hypertext as a lens into interactive digital narrative. In: Mitchell, A., Vosmeer, M. (eds.) *ICIDS 2021. LNCS*, vol. 13138, pp. 509–524. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-92300-6_51
35. Salter, A., Moulthrop, S.: *Twining: Critical and Creative Approaches to Hypertext Narratives*. Amherst College Press (2021). <https://doi.org/10.3998/mpub.12255695>
36. Murray, J.H.: Research into interactive digital narrative: a kaleidoscopic view. In: Rouse, R., Koenitz, H., Haahr, M. (eds.) *ICIDS 2018. LNCS*, vol. 11318, pp. 3–17. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-04028-4_1