jeu-video, code, fiction-interactive, narration, em

Twine

Twine est accessible sous forme d'application à installer ou directement en ligne. Ce logiciel permet d'écrire des récits interactifs grâce à une interface de programmation par noeud, appelés «passage», les passages peuvent contenir du code (structures conditionnelles, etc.), des macros, des évènements d'interaction (hooks), etc.

Un récit Twine peut aussi être rédigé directement dans un fichier texte avec balises, au format twee (.tw) puis compilé en HTML avec Tweego (il existe aussi d'autres compilateurs)

Twine existe en version 1 (2009-2015) ou 2 (depuis 2015), Dans chacune de ses versions, Twine propose plusieurs «formats d'histoire» différents , chacun comporte des macros qui lui sont propres ainsi qu'un rendu graphique et une documentation. Ces formats sont conçus pour différents usages.

Site du projet : https://twinery.org/
Documentation : https://twinery.org/cookbook/index.html doc complète
Download :

- twine : https://github.com/klembot/twinejs/releases
- tweego : https://github.com/tmedwards/tweego

TL;DR

Pour découvrir Twine, les tutos vidéos francophones de Tourmaline Studio sont parfaits! :
https://www.youtube.com/watch?v=dl_WlMMTt1s&list=PLFGDV4C3TmMGzuRN9In6MQducHbxqwTWI

Formats d'histoires : Harlowe et les autres

chapbook, harlowe, paperthin, snowman, sugarcube : chacun en plusieurs versions

Chaque format d'histoire comprend un format de mise en page différent, un ensemble de macros et certaines fonctionnalités propres programmaées en javascript
cf. https://twinery.org/cookbook/terms/terms_storyformats.html

Harlowe : format par défaut avec Twine 2, conçu pour une facilité d'utilisation et pour les personnes qui découvrent Twine 2

Sugarcube : s'inscrit dans la continuité de Twine 1 en augmentant les macros disponibles. Il a plus de fonctionnalités qu'Harlowe mais nécessite parfois une meilleure connaissance des techniques de programmation et des patterns de développement pour des usages avancés.

Snowman : conçus pour être utilisé avec du javascript et du css "custom". Pas de macros incluses, mais intègre les lbibliothèques javascript suivantes : Underscore.js, Marked et jQuery.

Chapbook : format de 2e génération pour Twine 2, sépare les fonctionnalités entre "inserts" pour l'affichage du texte et "modifiers" pour les diverses fonctionnalités qui s'appliquent au texte.

Harlowe est le format conseillé pour commencer en Twine 2, il est activé par défaut

- Dans le Twine Cookbook, on trouve pas mal d'exemples adaptés à chaque format d'histoires
- Harlowe 2.1.0 doc : https://twine2.neocities.org/2
- Harlowe 3.3.8 doc : https://twine2.neocities.org/
- Sugarcube doc : http://www.motoslave.net/sugarcube/2/
- Snowman :
    - https://videlais.github.io/snowman/#/
    - https://github.com/videlais/snowman
    - Snowman Cheatsheet

Plus d'infos en français ici : https://www.bac-a-sable.eu/twine/quelformatchoisir/
Spécifications des formats d'histoire de Twine 2 : https://github.com/iftechfoundation/twine-specs/tree/master

A ces formats d'histoire proposé par Twine, on peut ajouter des formats préparés par la communauté, comme **trialogue** qui permet de rédiger des dialogues type messagerie instantanée. L'installation d'un nouveau format d'images se fait dans l'interface du logiciel (voir ci-dessous) en important le fichier **format.js** dédié (même principe pour la version en ligne que pour l'application desktop)

Trialogue
Trialogue est un format d'histoire basé sur Paloma, qui à son tour est basé sur Snowman!

- https://github.com/phivk/trialogue
- http://phivk.com/trialogue/
- https://phivk.gitbook.io/trialogue/making-a-scripted-chat-story
- **lien direct vers le** fichier format.js de Trialogue

Paloma
Toute l'histoire à la suite avec le format d'histoire Paloma : https://mcdemarco.net/tools/scree/paloma/

Ajouter un format

Chaque format est défini dans un fichier format.js en javascript (extension .js). Pourinstaller ce format dans l'interface de Twine 2, choisir l'onglet Twine, puis "formats d'histoire", puis "ajouter"

Pour associer un format à une histoire : menu histoire / détails / choisir dans la liste déroulante

Formats de vérification

À ces formats d'histoire s'ajoutent les formats de vérification, comme **Paperthin** (par défaut), qui affiche l'intégralité des passages de manière très simple. Il existe aussi d'autres formats de vérification, comme Illume

Dotgraph
Dotgraph est un format de vérification qui produit un graphe de l'histoire. On peut aussi l'utiliser comme un service web pour schématiser une histoire en ligne.

- http://www.mcdemarco.net/tools/scree/dotgraph/

Syntaxe avec Harlowe

Liens entre passages Les doubles crochets carrés permettent d'insérer des embranchements directement dans le texte des passages, on peut aussi sur le modèle des liens hypertextes utiliser une syntaxe en 2 parties qui permet de différencier la phrase à cliquer du titre du passage :

```
[[phrase proposant le choix->titre du passage]]
```

Commentaires Intégrables dans le texte des passages, Syntaxe comme en HTML

```
<!-- ceci est un commentaire -->
```

Whitespace Quand on insère des lignes de code macro, elles ne sont pas visibles dans le jeu mais les sauts de ligne apparaissent quand même, pour les supprimer, il faut entourer les blocs de macro par des accolades {}

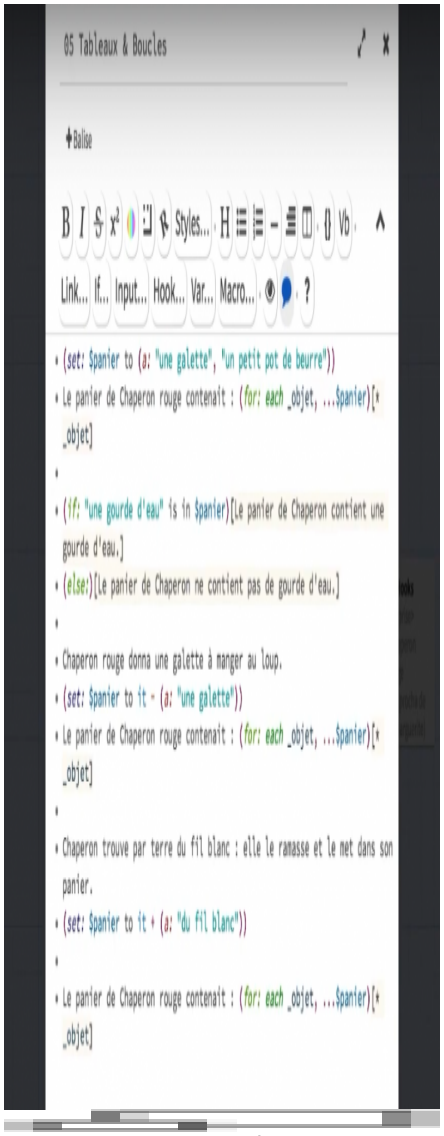
Code Les **macros** permettent de définir des variables, avec $ en préfixe, ce sera une variable globale. Les variables locales sont indiquées avec un underscore en préfixe. On peut typer les vaiables en utilisant des guillemets pour définir une chaîne, nombre sinon

```
$jetons, _essais, $annee, set:$jetons to 12, set:$annee to "2024", set: num-type $age to 19
```

Les **hooks** : identification d'un élément qui pourra être manipulé par du code, on place l'élément entre crochet carré et on peut le nommer, voir exemple ci-dessous (cf. https://www.youtube.com/watch?v=E6zOa1lEJCE)

```
|surprise>[texte à cliquer] (click: ?surprise)[(replace: ?surprise)[nouveau texte qui remplace le précédent]]
```

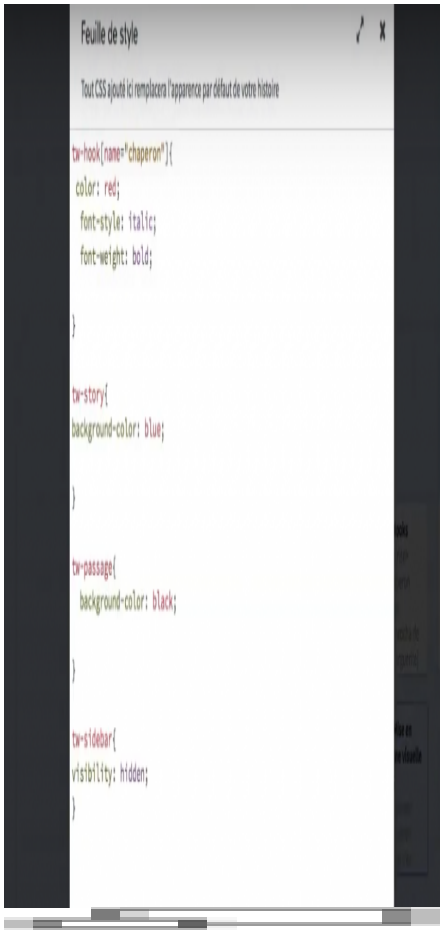
On peut utiliser des **expressions conditionnelles** (if, else, else-if, unless), des tableaux, des boucles



Images extraites des très bons tutoriels vidéo de Tourmaline Studio

Styles

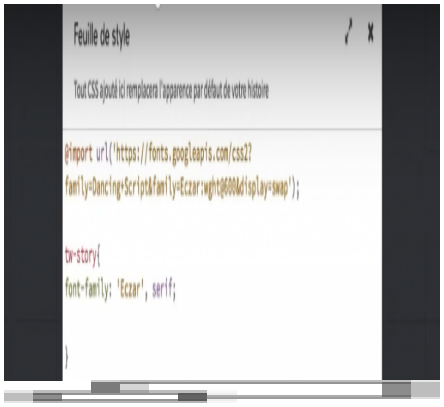
(menu histoire / feuille de style) Mise en forme : directement dans le passage ou par un ensemble regles CSS Balises : on peut ajouter des tags aux passages, par exemple pour les styler différemment en CSS



Images extraites des très bons tutoriels vidéo de Tourmaline Studio

Polices de caractère

import css classique



Images extraites des très bons tutoriels vidéo de Tourmaline Studio

Images

On peut intégrer une image avec la balise html `<img src="image.jpg" width="100%">` pour l'intégrer dans un passage, ou avec la propriétés css background-image associée à un tag pour une image de fond, cf. https://www.youtube.com/watch?v=YEDdbehQL6s L'usage des liens relatifs pose problème car les actions test et play dans Twine renvoient vers un fichier temporaire (dans le répertoire `file:///home/emoc/Documents/Twine/Scratch/`) mais ça fonctionne si on build le jeu. Des personnes ont proposé des solutions (lien symbolique entre les dossiers, cf. https://intfiction.org/t/a-method-for-using-relative-media-links-for-twine-development-on-windows/66637)

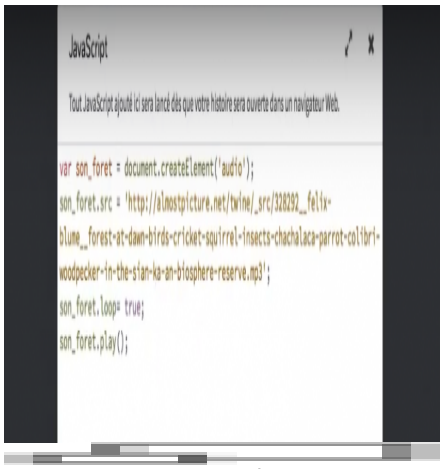


Images extraites des très bons tutoriels vidéo de Tourmaline Studio

Sons

En javascript (menu histoire / javascript de l'histoire)

Sons ambiant pour toute l'histoire



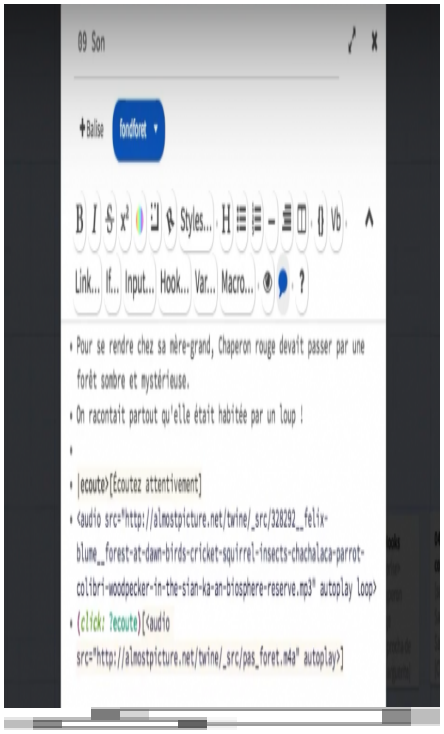
Images extraites des très bons tutoriels vidéo de Tourmaline Studio

Son ambiant pour un passage



Images extraites des très bons tutoriels vidéo de Tourmaline Studio

Son évènementiel : en associant le déclenchement d'un son à une macro



Images extraites des très bons tutoriels vidéo de Tourmaline Studio

Si on peut utiliser du js, ça pourrait aussi être sous cette forme intégrée dans le texte :
http://soundcite.knightlab.com/#make-clip-options

Vidéo

En utilisant la balise html video (à tester)

Préparer des récits Twine en format texte (twee)

L'application Twine est composé de 5 parties principales qui composent le projet :

1. le visualisateur d'histoire : permet de voir les différents passages d'une histoire et les liens entre eux.
2. l'éditeur de passage qui permet de rédiger les passages et de définir les liens entre eux.
3. l'éditeur javascript
4. l'éditeur CSS
5. le compilateur : il assemble le texte des différents passages qui composent l'histoire, le javascript et le css dans un template d'histoire pour les combiner dans un unique fichier HTML (= publication)

Un projet Twee est composé de 3 parties :

1. le compilateur Twee : un utilitaire en ligne de commande qui combine les différents éléments du projet (textes, javascript, css) dans un template d'histoire. Il en existe plusieurs et TweeGo est le plus utilisé
2. Un éditeur de texte pour rédiger les différents fichiers du projet (VSCode / Codium propose une extension pour Twee)
3. les différents fichiers : js, css, un ou plusieurs fichiers twee (.tw), etc.

Un intérêt de Twee est qu'il permet de modulariser des passages pour les utiliser dans différents projets. Twee permet aussi de travailler à plusieurs plus facilement en séparant des sections de passages que chaque personne travaille. On peut lui associer un gestionnaire de version (type git)

À noter : on peut exporter un fichier twee depuis L'application Twine

Notation Twee

- https://twinery.org/cookbook/terms/terms_twee.html
- http://www.motoslave.net/tweego/docs/#twee-notation

Tweego :

- install : https://github.com/tmedwards/tweego
- http://www.motoslave.net/tweego/

Export / publication

Depuis l'application, l'histoire est exportable en un fichier HTML unique ou associée à un dossier si elle contient des assets (images, vidéos, sons). Il est possible de réaliser une application exécutable à partir de ces fichiers en utilisant NW.js ou Electron :

- Web2Executable (nw.js) : https://github.com/jyapayne/Web2Executable
- Electrify (Electron) : https://github.com/jyapayne/Electrify

Application android (apk) avec PhoneGap , voir
https://whatbinder.com/2018/06/01/converting-twine-2-stories-to-android-apk-apps-for-the-google-play-store/

Ressources

- **très complet, ressources décrites** https://twinelab.net/twine-resources/
- https://github.com/vorpalhex/awesome-twine
- https://twinery.org/cookbook/
- https://twinery.org/questions/
- des jeux réalisés avec Twine
  https://www.reddit.com/r/twinegames/comments/wuzem0/the_most_impressive_twine_game_youve_played/
- tutos vidéos spécifiques https://www.youtube.com/@DanCox/videos
- The Twine Grimoire (pdf) : https://gcbaccaris.itch.io/grimoire-one
- Introduction to Twine (Harlowe) : https://docs.wixstatic.com/ugd/d1fd96_4cea140f718a483689aa26a67981671c.pdf

Chat-like discussions

- trialogue (voir plus haut) est très bien pour simuler des conversations messagerie instantanée, mais basé sur le format d'histoire snowman
- chat like discussion in harlowe : http://twinery.org/forum/discussion/7994/chat-like-conversations-for-harlowe / css voir https://codepen.io/2ne/pen/AXMLpj
- sur le même sujet une vidéo un peu datée... (2017) : https://www.youtube.com/watch?v=kjKRegikbYQ

Récits exemple

- un exemple en Harlowe : https://www.bac-a-sable.eu/agneau/

Tutoriels

une série de 10 tutos vidéos par Tourmaline Studio, très bien faits pour découvrir Twine 2 (2021) :
https://www.youtube.com/watch?v=dl_WlMMTt1s&list=PLFGDV4C3TmMGzuRN9In6MQducHbxqwTWI
(en) vidéo d'intro très bien : https://www.youtube.com/watch?v=lhn39SPETMM

Article extrait de : https://lesporteslogiques.net/wiki/ - **WIKI Les Portes Logiques**
Adresse : **https://lesporteslogiques.net/wiki/ressource/logiciel/twine/start?rev=1737993909**
Article mis à jour: **2025/01/27 17:05**