

Peritext

Publication multimodale
orientée contexte

Une introduction

MUSEI
WORMIANI
HISTORIA
LUGD-BATAVORUM
EX OFFICINA ELSEVIRIANA
Acad Typis & 1655.

Peritext – publication multimodale orientée contexte

Une introduction

Robin De Mourat

Sommaire

Introduction	7
Le format de données peritext	14
L'écosystème des modules peritext	20
Des systèmes documentaires riches et exploitables	22
Un design adaptable et modulaire	25
Contextualiseurs	27
Contextualiseur bib	28
Contextualiseur glossary	29
Contextualiseur webpage	30
Contextualiseur embed	33
Contextualiseur vidéo	37
Contextualiseur dicto	38
Contextualiseur codefiles	40
Contextualiseur table	41
Contextualiseur vegalite	42
Contextualiseur p5 (processing)	49
Gabarits	50
Générateurs	51
Éditeurs	54
Contribuer à peritext	55
Questions fréquemment posées	57
À propos	61
Références	63
Glossaire	70

Introduction

Peritext est un projet ouvert et libre développé pour supporter des pratiques d'édition savante alliant respect des contraintes de l'édition scientifique et expérimentation sur la forme et sur le contenu des arguments pouvant être développés par les chercheurs et les étudiants.

Peritext permet de fabriquer des documents qui tirent parti de toute la richesse documentaire qu'offre le web, du fichier de données *open data* à l'entretien enregistré dans une vidéo en ligne. Les documents produits avec Peritext sont construits selon une modélisation documentaire qui permet de distinguer la description des ressources convoquées dans un texte de la manière de les mettre en scène et de les présenter.

Peritext permet de fabriquer des documents dits « multimodaux », c'est-à-dire matérialisés simultanément selon plusieurs formats différents, de l'imprimé au site web en passant par l'epub ou la publication automatisée sur réseaux sociaux. Les documents produits dans chacun de ces formats se distinguent par une haute qualité graphique et documentaire.

En quoi consiste Peritext ? par exemple, à produire le document que vous consultez en ce moment !

Qu'est-ce que Peritext ?

Peritext est d'abord une **spécification de format de données** pour la représentation de documents académiques (27). Autrement dit, il s'agit d'un mode d'emploi expliquant comment représenter le contenu d'un document selon une certaine convention de structuration et de description de ses données. Cela permet de fabriquer des applications qui écrivent et qui lisent des documents écrits selon cette convention, et contribue à produire un *format de données* particulier, ouvert et libre, le *format Peritext*¹.

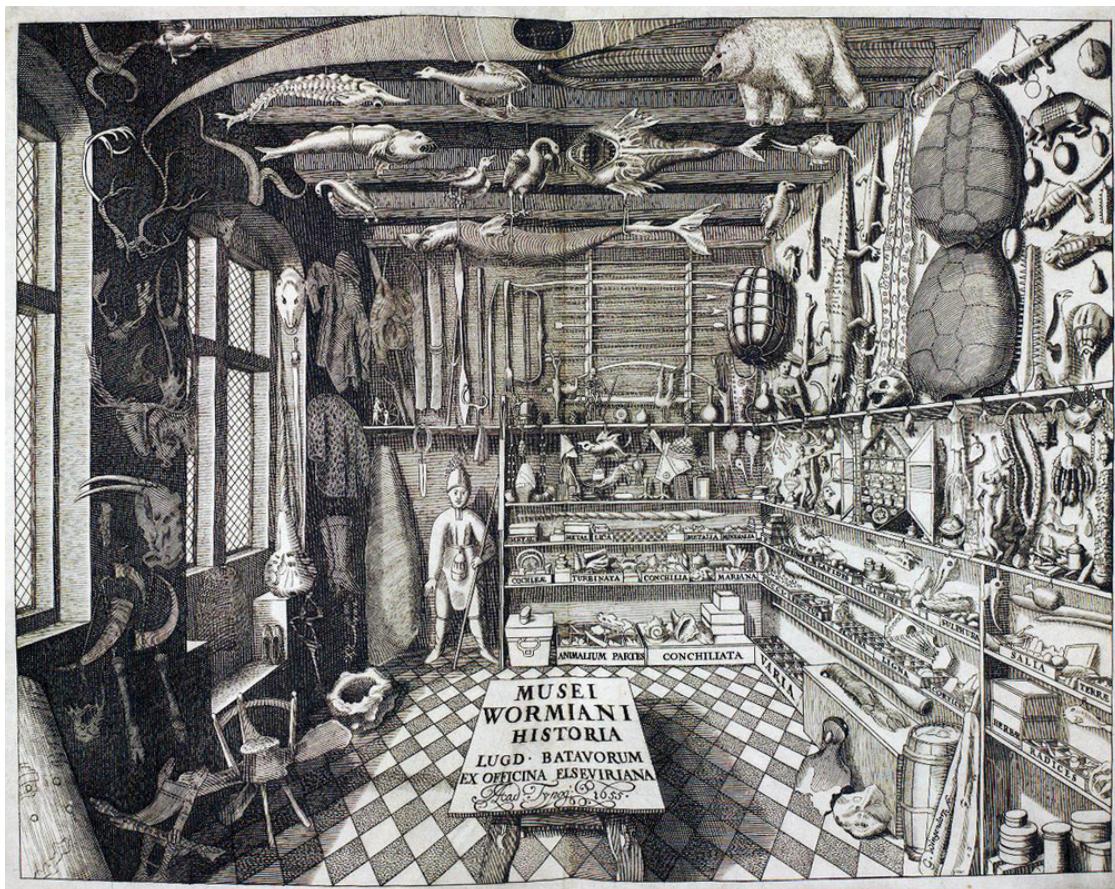
Autour du *format Peritext* s'articule l'ensemble des *modules Peritext* qui sont des briques technologiques et des applications directement utilisables qui en actualisent les potentialités et lui donnent du sens². Ces modules technologiques ont tous en commun le fait qu'ils visent à aider à automatiser certaines tâches relevant des normes documentaires et éditoriales liées à l'édition scientifique contemporaine³, tout en ménageant beaucoup de place et de souplesse pour l'affichage de contenus non-conventionnels, leur mise en scène fine de manière à servir au mieux les arguments de l'auteur, et leur design en fonction des différents supports pris en charge.

1. Le format peritext décrit donc les « données » d'un document, et en aucun cas un type de mise en forme graphique particulier.
2. Tous les modules sont écrits en Javascript et, plus précisément, utilisent intensivement la bibliothèque de rendu open source react (35), qui permet de produire indifféremment du code pour des applications client et des applications serveurs, et pour des environnements interactifs ou des environnements statiques (type générateur de pdf).
3. Des tâches comme la documentation des sources, et la description des métadonnées du document et des documents qu'il cite.

À quoi sert Peritext ?

En amont comme en aval de l'acte de publication, les textes produits et échangés par les chercheurs universitaires convoquent d'une manière de plus en plus intime les environnements numériques (au premier rang desquels le web) dans lesquels ils sont inscrits.

On assiste en effet, dans un large spectre de disciplines comme de sujets de publication, à la prolifération d'une variété d'objets hétéroclites qui participent de manière de plus en plus active aux processus de recherches et sont — contrairement à jadis — directement et pleinement convocables au sein mêmes des publications. En témoignent des exemples aussi divers que l'utilisation croissante d'enregistrements audiovisuels en ligne, l'utilisation des réseaux sociaux et de twitter comme élément de corpus et/ou de restitution des débats en cours, la mobilisation d'archives et de dépôts de sources numérisées rendant possible leur présentation sous forme diagrammatique ou tabulaire, etc.



Le cabinet d'Ole Worm

Reproduction de la gravure conservée à Modène représentant un cabinet de curiosités

Source: *Bibliothèque Estense, Modène.*

Cela dit, dans les formats les plus répandus de l'édition scientifique, les traces du web et autres ressources numériques peinent encore à sortir de la marge des pages : on les remarquera subrepticement au détour d'une note de bas de page (sous la forme d'un discret lien), ou timidement reproduites dans une annexe secondaire, à la fin d'une entrée bibliographique ou dans la légende d'une figure, etc. Ces connexions à des éléments externes désignent des ressources riches, dont la matière contribue aux arguments développés de manière décisive, et est souvent disponible en intégralité en ligne par ailleurs. Elles restent pourtant souvent sur le seuil des échanges et des arguments, notamment du fait qu'elles se retrouvent pour ainsi dire « effacées » des documents une fois traduites et formatées selon les conventions de publication instituées (articles, monographies, actes, mémoires, thèses, ...)⁴.

D'un autre côté, l'essor de la publication en ligne, du mouvement de l'accès ouvert au résultats de la recherche, et de nouvelles pratiques de communication de la part des chercheurs eux-mêmes qui vont de l'usage des réseaux sociaux à l'expérimentation de diverses formes de publication en ligne, ont instauré un régime de production et de circulation des documents académiques extrêmement riche où ces derniers se voient discutés, critiqués, reformatés et éditorialisés par-delà une multitude de mondes scientifiques, sociaux et médiatiques. Le contexte actuel est celui d'une « *diffraction* » de l'*écriture savante*, appelée à être *continuellement traduite et déclinée sur un spectre toujours plus large d'environnements et de médias*.

Il est donc nécessaire d'inventer de nouvelles manières de connecter les modes de production des documents académiques avec les contextes du web. Il s'agit d'opérer une double *recontextualisation* : les ressources numériques doivent être plus significativement contextualisables dans les documents savants, et les documents savants doivent être plus significativement contextualisables dans les environnements numériques.

Pour quels contextes Peritext est-il fait (et pas fait) ?

Peritext ne s'adresse pas à toutes les pratiques d'édition savante, il n'entend en aucun cas proposer une solution globale prenant en charge l'intégralité d'un processus d'édition scientifique, ou couvrir la totalité des contextes de projet scientifiques ou éditoriaux.

4. Pourquoi ne pourrait-on pas avoir accès au contenu complet d'une référence citée dans un texte, s'il est par ailleurs disponible sur le web ? Que dire des vidéos et autres médias ? Ne serait-il pas pertinent de donner ces vidéos à voir directement au lecteur, côte à côte avec ce qui en est dit, voire même d'en pointer des passages précis à différents endroits d'une argumentation ? Quand un jeu de données est mobilisé, pourquoi est-ce si compliqué de le faire parler, par le texte et par l'image ? Ne pourrait-on pas mieux structurer les références aux lieux, personnes et autres éléments de glossaire faites dans un texte en les connectant directement aux ressources qui les concernent ?

Le projet n'est pas conçu comme un système prêt à l'emploi. Il s'agit plutôt d'un écosystème ouvert (c'est-à-dire fait pour se connecter à un archipel de normes, outils, pratiques et projets existants).

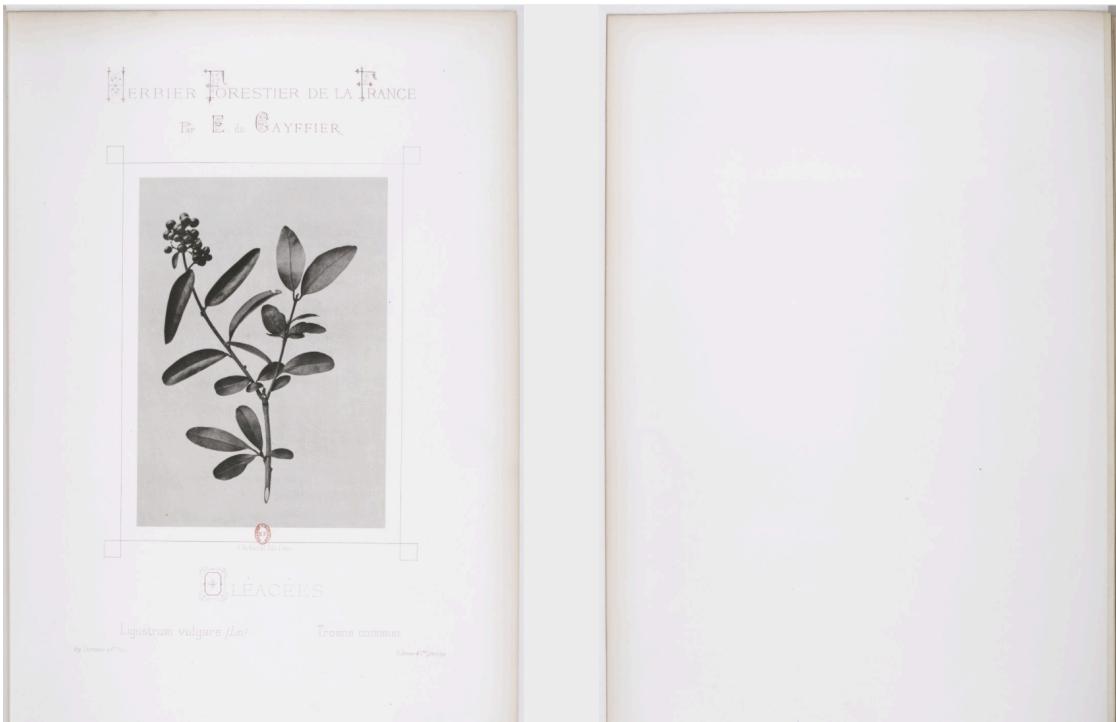
Le projet n'est pas conçu comme une solution à toutes les pratiques d'édition savante. Il a été incubé à partir d'études de cas relevant des « sciences humaines et sociales » et des « humanités numériques », et devrait donc plus naturellement s'aligner avec ces contextes, même s'il est probablement utilisable pour d'autres.

Peritext devrait être pertinent pour les situations suivantes :

- **Les projets éditoriaux dans lesquels les contenus convoquent de manière très importante des documents extérieurs** (tableaux de données, médias, sites web, ...) et consistent entre autre à les commenter en profondeur
- **Les projets éditoriaux qui gagnent à se décliner sur une grande diversité de formats,** au-delà des seules formes imprimées
- **Les projets éditoriaux nécessitant une (très) bonne qualité documentaire,** notamment en termes de métadonnées et d'indexation
- **Les projets éditoriaux nécessitant une (très) bonne qualité graphique** et nécessitant un contrôle fin sur le rendu dans les différents formats

Par ailleurs, Peritext ne sera pas pertinent pour :

- les projets éditoriaux qui misent de manière centrale sur une dimension collective ou réticulaire. Ce point ne devrait pas évoluer dans le futur du projet, mais pour l'instant l'utilisation d'outils tiers tels qu'Hypothes.is ou disq.us est favorisée au niveau des générateurs de type web pour permettre une dimension collaborative minimale au niveau des pratiques d'annotation et commentaire.
- les projets éditoriaux dont le contenu est construit selon des structures rhétoriques qui se départissent de l'organisation traditionnelle des documents académiques, faits de sections (chapitres, parties, ...) se suivant de manière séquentielle. Si votre « matériau de base » ressemble, dans sa structure, à une base de données quelconque ou à un programme interactif (type « histoire dont vous êtes le héros ») plutôt qu'un récit séquentiel traditionnel, alors Peritext n'est pas fait pour vous. Ce point est soumis à une possible évolution ultérieure.
- les projets qui veulent expérimenter de nouvelles formes d'édition savante qui ne prennent pas le texte pour médium principal et structurant pour l'argumentation (organisées autour d'une vidéo ou d'une image par exemple)



**Herbier forestier de la France. Reproduction par la photographie... des principales plantes ligneuses qui croissent spontanément en forêt.
Description botanique...**

Source: *Gallica*

Une écriture et une édition multi-supports

Le format de données Peritext permet de produire une diversité de documents à partir d'une seule et même représentation argumentaire.

Par exemple, ce document même est directement accessible comme un site web, un fichier pdf imprimable, un fichier epub exécutable sur liseuse numérique, une API de manière à ce qu'il soit utilisable par une autre application, des fragments destinés à Twitter, etc.

Peritext rentre dans la catégorie de technologies intitulées « COPE » (« Create Once, Publish Everywhere »⁵) qui consistent à produire une diversité de documents à partir d'un même texte source. Ainsi, avec Peritext les auteur.e.s peuvent écrire une seule fois leur article, chapitre ou livre, et disposer en retour et simultanément de diverses contextualisations de ce contenu pour les différents supports (site web, pdf, epub, ...).

5. « Crée une seule fois, publier partout ». Voir par exemple la technologie Pandoc([40](#)) dans ce registre.

Ils disposent également de paramètres permettant spécifier des règles de présentation spécifiques à un format en particulier, comme par exemple les règles de mise en forme graphique ou les paramètres de présentation d'une ressource particulière pour un format donné.

Un modèle uniifié et souple pour gérer les arguments centrés sur des documents et autres éléments extérieurs

Le format de données Peritext permet de prendre en charge la contextualisation de nombreux éléments extérieurs dans la structure argumentative d'un texte académique (références, données, documents audiovisuels, entités de glossaire, etc.). Pour ce faire, il repose sur un mode de représentation des contenus particulier, centré sur la séparation entre les ressources convoquées dans un récit et les différentes formes que ces dernières peuvent prendre dans le cadre de l'argumentation opérée par un texte⁶.

Respect des normes techniques et éditoriales de l'édition scientifique

Sur le plan documentaire, peritext répond à une série de besoins spécifiques à l'édition scientifique en termes d'indexation et de citabilité⁷.

Adaptation et modularité

Peritext est pensé selon une logique de « boîte ouvrable », c'est-à-dire que ses différents modules fonctionnent avec des paramètres par défaut mais sont par ailleurs profondément paramétrables par les utilisateurs qui le souhaitent.

Outre les paramètres exposés par chacun des modules, peritext ouvre aux utilisateurs avertis la possibilité préciser et de façonnez la présentation graphique des documents grâce aux langages css et css print qui peuvent être utilisés de manière distincte pour chacun des types de formats produits à partir d'un contenu.

Enfin, il est construit selon une logique modulaire qui permet beaucoup de souplesse et de variations dans l'implémentation d'un écosystème éditorial spécifique (en termes de supports visés, de types de documents pris en charge, ...)⁸.

6. Pour en savoir plus, rendez-vous à la section « Le format de données peritext ».

7. Pour en savoir plus, rendez-vous à la section « Des systèmes documentaires riches et exploitables ».

8. Pour en savoir plus, rendez-vous à la section « adaptation et modularité ».

Économie de moyens techniques

La représentation d'un récit peritext est faite au **format json**, ce qui veut dire qu'elle ne nécessite nativement pas de technologies de base de données et peut être stockée dans un simple fichier.

Par ailleurs, le générateur de site web actuel produit un répertoire de fichiers html qui peut être hébergé sur un service d'hébergement gratuit tel que github pages ou surge.

Ainsi, même s'il peut être intégré dans des systèmes d'information ou plateformes de grande envergure, l'écosystème peritext est d'abord pensé pour des contextes d'implémentations légers, décentralisés et peu coûteux en équipement⁹.

9. Pour en savoir plus, rendez-vous par exemple à la section « générateurs ».

Le format de données peritext

Le format de données peritext est fait pour la description de documents savants, qui peuvent correspondre à des formats traditionnels aussi variés que :

- un article de revue académique
- une monographie
- des actes de colloque
- un numéro de revue académique

... ou à des formes ne correspondant à aucun des formats d'édition scientifique établis.

Techniquement, peritext se fonde sur une description au **format json**, qui permet de décrire un objet selon une structuration complexe faite d'éléments imbriqués, et de stocker cette description indifféremment dans de simples chaînes de caractères et autres fichiers de texte ou sous la forme d'un objet dans une base de données.

La spécification exacte du format, actuellement encore en gestation¹⁰, est définie dans un des module du projet intitulé peritext-core (27)¹¹.

Les paragraphes qui suivent décrivent et explicitent les grandes lignes du format peritext. Pour une idée plus précise et technique du schéma, voir le répertoire peritext-core ou sa documentation technique (28), ou se rendre (pour la version web) à la rubriques « autres formats » de ce récit et le télécharger au format json.

Caractéristiques générales du format peritext

Le format peritext est d'abord structuré autour des aspects suivants :

- un récit Peritext est décrit par une série de métadonnées qui couvrent : son titre et sous-titre, son ou ses auteurs, etc.

10. C'est pourquoi tous les modules du projets sont pour l'instant à l'état de version « pre-alpha », c'est-à-dire qu'ils sont instables & non utilisables pour des situations réelles car appelés à évoluer de manière non-compatible avec leur version actuelle.
11. Peritext-core est notamment dédié à décrire le format peritext au moyen de la spécification json-schema développée par l'Internet Engineering Task Force (29)

- un récit Peritext est structuré en *sections*. Une section peut être une partie, un chapitre, une introduction, des remerciements, ... les sections sont assemblées dans un ordre séquentiel pour construire le *sommaire* du récit. Elles peuvent disposer d'un niveau hiérarchique qui permet de définir les sections les plus générales et les sections d'importance moindre.
- chaque section est décrite avec ses propres métadonnées (titre, auteurs, ...)
- chaque section est décrite avec un contenu structuré autour d'un texte principal et d'une série de notes.
- les contenus d'une section ou d'une note en particulier sont représentés sous la forme de blocs de textes annotés avec des styles et des entités spécifiques selon une norme établie¹².
- outre les sections et leur contenu, au niveau d'un récit peritext sont également définis des paramètres (*settings*) qui définissent des informations relatives à l'affichage du récit telles que : le gabarit à utiliser, les règles de citation, des paramètres tels que la position des notes, et enfin le code css à injecter pour chacun des types de supports
- enfin, un récit peritext est décrit par une série d'objets relatifs à la contextualisation d'éléments extérieurs dans l'argument : les ressources, les contextualiseurs et les contextualisations. Ce dernier aspect est décrit en détail ci-après.

Caractéristiques spécifiques liées à l'argumentation fondée sur les documents : le modèle RCC

12. Ce type d'annotation s'adosse à la spécification open source draft-raw développé par l'équipe de développement de l'entreprise Facebook, qui sert à décrire au format json des contenus textuels « riches » pour des applications utilisant la technologie de rendu html intitulée react, qui est utilisée dans tous les modules de l'écosystème peritext. Voir le site du projet draft-js pour plus d'informations [\(25\)](#) .

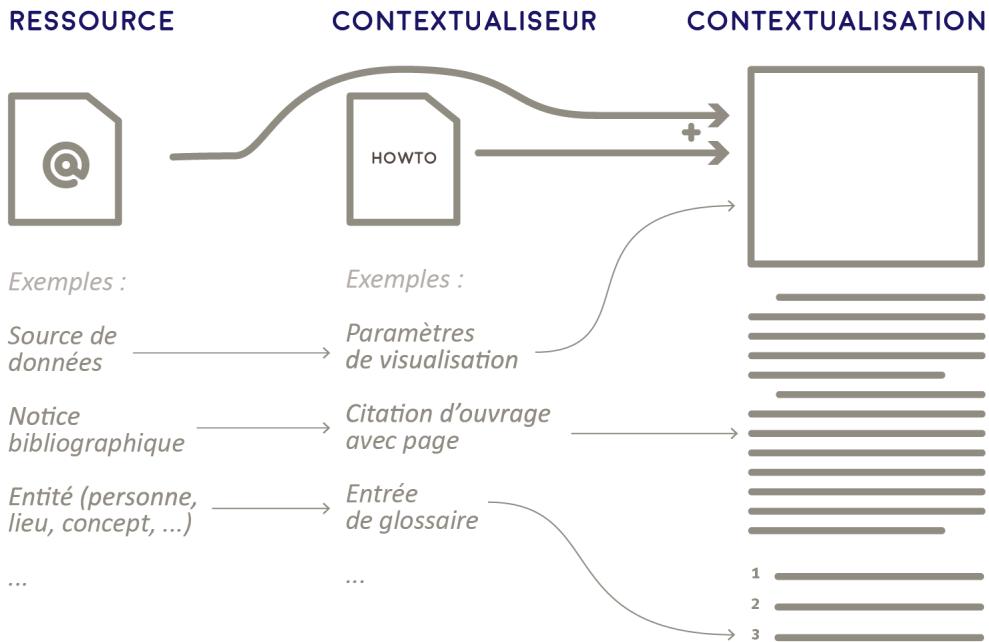


Schéma RCC

Peritext tire sa spécificité de sa vocation à soutenir la convocation de nombreux éléments extérieurs dans la structure argumentative d'un texte académique (références, données, documents audiovisuels, etc.).

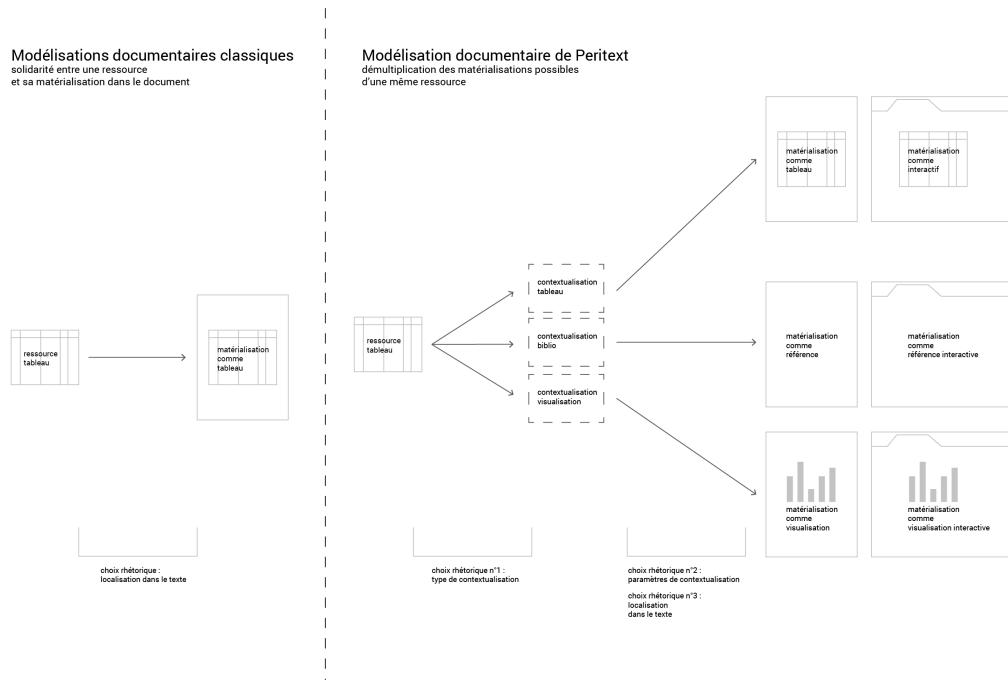
Pour ce faire, dans un récit Peritext, la convocation d'éléments extérieurs dans l'argument du récit s'effectue via trois types d'objets :

- **les ressources** décrivent tous les éléments à convoquer dans le texte, selon un **registre documentaire** indifférent à la forme qu'ils prendront dans le récit. Les ressources peuvent désigner des éléments très variés dans leur nature (références, données, médias, mais aussi entités de glossaire telles que personnes, lieux, thèmes, ou encore morceaux de code, ...)
- **les contextualiseurs** décrivent un ensemble de paramètres permettant d'afficher une ressource d'une certaine manière. Ils agissent sur un **registre rhétorique** pour préciser comment la ressource doit être affichée.
- **les contextualisations** enfin sont la conjonction d'une ressource et d'un contextualiseur à un endroit précis de l'argument. Elles sont par ailleurs assorties d'un titre et d'une légende.

Le modèle de données de Peritext est donc intitulé RCC car il sépare la liste des Ressources utilisées dans le document, les Contextualisations de ces ressources à des endroits spécifiques et les Contextualiseurs qui les font s'afficher d'une manière spécifique.

Cette dimension permet d'envisager des récits académiques qui exploitent bien plus en profondeur les ressources convoquées dans les textes que ce que permettent les formats d'écriture établis.

L'avantage principal de ce modèle est la latitude rhétorique qu'il offre pour les auteurs. Cet atout est rendu viable par le caractère multimodal du modèle RCC qui permet d'envisager des scénarios d'écriture pour plusieurs supports, notamment dans le cas des chercheurs, d'une part pour les supports traditionnels qui demandent un rendu dans un format exclusivement textuel et souvent de type « page » (exemple : au format pdf), et d'autre part pour des formats plus riches permis par le médium web.



Les implications rhétoriques du modèle RCC

Une démonstration d'application du modèle RCC

George, chercheur en archéologie, explique le résultat de sa dernière expédition en Égypte via un tableau représentant la quantité de vases retrouvés sur une liste de sites qu'il a listés par des lettres (« A », « B », « C », ...)

Au moyen d'un éditeur compatible avec Peritext, George stocke d'abord son résultat (consigné avec un autre logiciel type tableur) dans une ressource de type « table », c'est-à-dire un simple tableau de données.

Il est d'abord possible de la contextualiser au moyen d'un contextualiseur de type « table », qui est le contextualiseur par défaut pour ce type de ressource. Ce dernier consiste à représenter la ressource sous la forme d'un tableau. En l'utilisant pour contextualiser sa ressource dans le paragraphe qui suit, George obtient le tableau suivant :

a	b
A	2
B	4
C	6
D	8
E	2
F	3
G	5

(exemple) une ressource contextualisée en table

Les vases de George représentés comme un tableau.

Maintenant, George souhaite à nouveau faire mention de cette ressource sans pour autant l'afficher à nouveau sous la forme d'un encombrant tableau. Pour ce faire, il va utiliser le contextualiseur « bib » qui contextualise la ressource sous la forme d'une référence bibliographique. C'est ce qui va être fait à la fin de ce paragraphe(23).

Enfin, mettons que notre auteur décide maintenant de discuter de la comparaison des valeurs numériques décrites dans les données de cette ressource. Il utilisera alors un troisième contextualiseur de type « vegalite », qui permet de construire des visualisations de données. C'est ce qui va être fait dans le prochain paragraphe :

C'est ainsi que le modèle RCC implémenté par le format Peritext permet :

- de représenter les éléments convoqués dans un argument de plusieurs manières différentes, selon les besoins de l'argumentation.
- de paramétrier finement la manière dont on veut voir affichée une ressource grâce aux paramètres des contextualiseurs et à l'injection de code css via l'éditeur.
- de maintenir une rigueur documentaire qui pourra ensuite être exploitée pour l'indexation du document et pour la génération d'une interface utilisateur exploitant le système hypertexte ainsi créé de manière homogène malgré la diversité des contextualisations possibles.

En effet, malgré la diversité des diverses formes de contextualisations pouvant être appliquées à une même ressource donnée, le lien avec cette dernière restera toujours présent, permettant par exemple de produire une liste de mentions dans la bibliographie. De la même manière, il est à noter que dans le présent gabarit de présentation (en version web) cliquer sur l'icône en dessous de chacune des contextualisations affiche la liste de toutes les autres contextualisations de la même ressource dans le récit et d'y accéder.

Remarque sur les principes du format peritext et sa relation à la forme

Dans une application d'édition suivant le modèle de données Peritext, l'auteur a non seulement la main sur les éléments extérieurs qu'il veut citer dans son texte, mais également sur la manière de les afficher et sur un ensemble de paramètres lui permettant de modeler précisément l'usage qu'il veut en faire dans son argument (par exemple : préciser un passage de texte, mettre en lumière une certaine dimension d'un jeu de données, spécifier un cadrage particulier sur une visualisation, etc.). **Il a ainsi ainsi la possibilité de développer un fil argumentatif pleinement articulé autour de la mobilisation des ressources qu'il convoque.**

Peritext tente de se départir de la traditionnelle distinction entre les modèles de description centrés sur les contenus (*What You See is What You Mean*¹³) et ceux centrés sur la forme propre à un type de support pour le document (*What You See is What You Get*¹⁴). Il s'agit plutôt de modéliser les *intentions argumentatives* attachées à un moment particulier du texte (*What you see is What You Argue*¹⁵), soit l'intersection entre fond et forme. La notion de *représentation intermédiaire* joue ici un rôle crucial, comme nom de cet état intermédiaire des contenus décrits selon le format Peritext.

13. « Ce que vous voyez est ce que vous voulez dire ». Voir des formats comme

14. « Ce que vous voyez est ce que vous obtenez » (à l'impression). Voir des formats comme word dans cette catégorie.

15. « Ce que vous voyez est ce que vous argumentez ». Nous proposons cette expression pour décrire cette troisième voie qui par se référence à la rhétorique ne sépare pas la question de la forme de la question du sens.

L'écosystème des modules peritext

Autour du format Peritext s'articulent un ensemble de modules technologiques qui le comprennent et en exploitent les potentialités. Conceptuellement, ces modules correspondent chacun à l'une des dimensions entrant en jeu dans la mise en forme multimodale d'un récit peritext. Concrètement, ils correspondent à des bibliothèques javascript structurées selon un modèle normalisé qui permet la compatibilité entre les modules de l'écosystème.

Peritext

Modules
constituant
l'écosystème

Générateurs

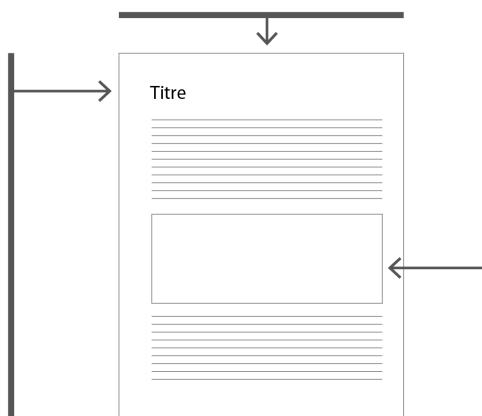
Transforment un récit
peritext en un fichier
(html, pdf, epub) selon
les paramètres fournis.

Gabarits

Définissent les règles
de présentation (thème,
interactivité, ...) d'un récit
pour un type de générateurs
donné.

Contextualiseurs

Produisent
une représentation
d'une ressource selon
un contextualiseur donné
(référence biblio,
visualisation, vidéo, ...).



Les modules de peritext

Les modules génératrices sont des modules qui « consomment » un récit pour le transformer en un document utilisable (un fichier pdf par exemple). Outre le récit à transformer, ils « consomment » également d'autres modules — les gabarits et les contextualiseurs — pour définir comment transformer les contenus et les contextualisations et organiser le document.

Les modules gabarits sont des modules qui définissent le thème visuel, l'organisation des contenus et les mécanismes d'interaction à utiliser par un générateur pour transformer un récit. En somme, ils définissent toute la mise en forme du récit, excepté les contextualisations.

Les modules contextualiseurs sont des modules qui définissent comment interpréter une série de paramètres de contextualisation contenues dans un objet contextualiseur (voir la section « le format de données peritext ») selon le contexte d'implémentation (comme paragraphe, dans un paragraphe) et selon le générateur utilisé.

Des systèmes documentaires riches et exploitables

Le modèle RCC implanté dans Peritext (voir la section « le format de données Peritext ») induit une haute qualité de description des ressources impliquées dans la production des documents peritext. Cette section décrit comment cette richesse est exploitée dans les modules constituant Peritext.

Gestion des références bibliographiques

Peritext s'adosse au standard international Citation Style Language ([\(19\)](#)) pour gérer les références. Il est ainsi capable de gérer la plupart des formats et langues de citation en utilisation. Les modules de l'écosystème permettent par ailleurs de prendre en charge n'importe quel type de ressource Peritext sous la forme d'une citation bibliographique.

L'éditeur démonstrateur Ovide (voir section afférente) propose par ailleurs une interface d'import de références au format BibTeX et une interface expérimentale d'ajout de références par interface graphique.

Métadonnées à l'échelle d'un récit et de ses sections

Dans les gabarits par défaut de l'écosystème de Peritext, les métadonnées renseignées au niveau des récits et des sections d'un document peritext sont mises à profit de la manière la plus complète possible.

Ainsi, les fichiers de sortie html de peritext présentent des métadonnées encodées aux formats DublinCore, Twitter, Open Graph & schema.org.

Métadonnées à l'échelle des contenus

À l'échelle des éléments affichés par les générateurs de Peritext (et notamment les contextualisations de ressource), Peritext entend fournir le meilleur degré de description machine possible.

Pour ce faire, chaque mention de ressource inclue l'imbrication automatique d'un composant CoINS [\(18\)](#) permettant par exemple de l'enregistrer avec le plugin Zotero.

Est prévue dans la feuille de route des modules une implémentation ultérieure des normes d'encodage sémantique au niveau des composants html eux-mêmes (au moyen du format RDFa) afin de permettre la lecture optimale d'un document peritext par une machine.

Systèmes hypertextuels à double-sens

Peritext est attaché à une utilisation optimale de la technologie hypertexte, initialement pensée pour les documents académiques [\(26\)](#). Sans explorer trop avant les possibilités rhétoriques et organisationnelles offertes par cette technologie dans le contexte académique ¹⁶, Peritext tente d'optimiser les formes d'hypertexte existant dans les documents traditionnels (liste de références, glossaire, index d'auteurs) en les opérationnalisant techniquement comme des liens à double sens à tous les endroits où ils sont implantés.

Ainsi, par exemple, une référence à une entrée de glossaire permettra toujours d'accéder à la liste de toutes les mentions de cette entrée, et cette liste permettra dans l'autre sens d'accéder à la localisation de chacune de ces mentions.

Peritext développe ici certaines idées éprouvées dans de précédentes expérimentations en Humanités Numériques¹⁷ mais également dans des expériences plus anciennes telles que le projet Xanadu qui envisageait un système hypertexte permettant de parcourir les liens à double-sens¹⁸

16. Voir le projet Scalar [\(17\)](#), qui expérimente des formes alternatives de structuration des arguments académiques en tirant parti des technologies de base de données relationnelles.

17. Voir par exemple le projet An Inquiry into Modes of Existence [\(8\)](#).

18. Voir notamment le project Xanadu [\(31\)](#) mis en place par Ted Nelson.



Ted Nelson demonstrates Xanadu Space

Ted Nelson présente le logiciel Xanadu Space qui spatialise en trois dimensions un système hypertexte.

Source: photonhunter (*youtube: https://www.youtube.com/watch?v=En_2T7KH6RA*)

Pour voir un exemple de mise à profit de ce principe, rendez-vous dans les sections « glossaire » et « références » de ce site web, ou cliquer sur une des citations présentes dans cette page pour voir apparaître sur la colonne de droite une liste des mentions des ressources associées ailleurs dans le document.

Un design adaptable et modulaire

Peritext est un projet ouvert, pensé selon une logique modulaire permettant une multitude de variations et d'adaptations autour d'un même modèle de données interopérable, celui du *format peritext*.

Le projet présente divers niveaux d'ouverture pour son adaptation aux besoins de chacun, qui correspondent aussi à différents niveaux d'expertise. Derrière ces choix de conception réside le projet de permettre un meilleur design des arguments académiques, dans toute leur spécificité et leur complexité, et ce à travers l'ensemble des formats et environnements utilisés par les chercheurs.

Niveau d'adaptation 1 : mise en forme paramétrique et css

L'adaptation de la forme des récits peritext est d'abord permise par un ensemble d'options et de paramètres associés aux documents (exemple: position des notes) et des contextualiseurs utilisés dans un récit. Ceux-ci dépendent du **gabarit** utilisé pour représenter un récit.

Il est par ailleurs possible d'écrire du code css spécifique pour chaque récit, et de différencier les styles pour les supports de type « web » et pour les supports de type « page ». Concernant les supports de type « page », les générateurs en place actuellement dans l'écosystème se plient à la norme css print (3).

Ce premier niveau est accessible à tout type d'utilisateurs, car il est notamment exposé par les éditeurs de contenus Peritext tels qu'Ovide (34).

Niveau d'adaptation 2a : gabarits

Un **gabarit** (ou template) décrit selon quelle forme interpréter les données incluses dans un récit au format peritext.

L'écosystème peritext propose pour l'instant un nombre réduit de gabarits. Il est possible de développer de nouveaux gabarits de mise en forme (sous la forme de composants React (35)) ou de développer des alternatives aux existants, à l'échelle d'un projet en particulier ou comme contribution à l'écosystème général.

Voir la section **gabarits** pour en savoir plus.

Niveau d'adaptation 2b : contextualiseurs

Les générateurs et les éditeurs de l'écosystème peritext utilisent des modules contextualiseurs pour décider comment afficher une contextualisation donnée pour un type de support donné.

Chaque contextualiseur peut exposer jusqu'à 4 composants distincts suivant le type de générateur et la localisation de la contextualisation : comme paragraphe en mode web, dans un paragraphe en mode web, comme paragraphe en mode « page », dans un paragraphe en mode « page ».

La logique modulaire des contextualiseurs permet d'implémenter des configurations spécifiques de Peritext en fonction des besoins de chacun (par exemple : plus orienté vers le partage de codes sources, plus orienté vers les médias vidéos et audio, plus orienté vers la visualisation des données, ...).

Il est possible d'adapter Peritext en développant de nouveaux contextualiseurs ou en développant des alternatives aux existants.

Rendez-vous à la section « contextualiseurs » pour en savoir plus.

Niveau d'adaptation 3 : générateurs

Peritext produit des documents (pdf, epub, site web) au moyen de modules intitulés **générateurs** qui consomment un récit peritext, des **contextualiseurs** et des **gabarits** pour produire un fichier utilisable et partageable. À ce niveau d'adaptation, il est possible de faire à peu près n'importe quoi à partir d'un document encodé au format peritext en écrivant un nouveau générateur.

Rendez-vous à la section « générateurs » pour en savoir plus.

Contextualiseurs

Selon la logique RCC, les générateurs de documents articulés autour de Peritext utilisent des modules *contextualiseurs* pour prendre en charge des ressources courantes dans la publication académique (référence bibliographique, image, élément de glossaire) mais aussi une variété d'autres éléments interactifs (vidéo, production dicto, production quinoa, visualisation vega, ...).

Les modules **contextualiseurs** décrivent comment afficher une ressource dans un contexte particulier et selon les paramètres spécifiques fournis par un object contextualiseur dans un récit peritext. Ce sont en somme des sortes de "mini-programmes « qui *en entrée* lisent une ressource et une série de paramètres et *en sortie* produisent une mise en forme spécifique.

Il est important de comprendre que les contextualiseurs ne sont pas forcément attachés à un type de ressource, puisque c'est là l'une des spécificités principales de Peritext. Certains contextualiseurs sont associés à un seul type de ressource (exemple: dicto), d'autres peuvent être appliqués à plusieurs types de ressources différents (exemple: référence bibliographique)¹⁹.

Les sections suivantes décrivent par le menu les divers modules contextualiseurs de Peritext déjà mis en place et leurs options.

Liste d'idées pour des contributions portant de nouveaux contextualiseurs

- unity : un contextualiseur jouant un code source unity
- three.js : un contextualiseur jouant une géométrie 3D au moyen de la technologie WebGL
- pdf : un contextualiseur affichant un pdf à partir d'une url au moyen d'un module de lecture personnalisé
- facebook: rendre un post facebook selon une mise en forme de qualité ou personnalisée
- twitter : rendre un tweet ou une liste de tweets selon des paramètres personnalisés
- portofolio: un contextualiseur permettant de présenter des séquences de documents & esquisses commentées

19. *Techniquement, ces bibliothèques exposent un ou plusieurs composants react ainsi que des feuilles de style.*

Contextualiseur bib

Le contextualiseur bib (pour bibliographique) affiche une ressource sous la forme d'une référence bibliographique.

Celle-ci peut être une citation courte, c'est-à-dire dans la forme prise dans le corps des textes, comme à la fin de ce paragraphe [\(14\)](#).

Elle peut aussi être longue, c'est-à-dire dans la forme prise normalement dans les bibliographies, comme suit :

^{1.} RICCI, Donato et DE MOURAT, Robin. *An Account of Digital Humanities from the AIME Project* [en ligne]. 2016. Revue échappées. Disponible à l'adresse : <http://echappees.esaprenees.fr/content/2-numeros/3-numero3/5-an-account-of-digital-humanities-from-the-aime-project/echappees-an-account-of-digital-humanities-from-the-aime-project.pdf>

An Account of Digital Humanities from the AIME Project

Le formattage exact de la citation dépend du style et de la locale de citation définie pour le récit Peritext en entier (par exemple : ISO-61, APA, Chicago, ...). Le contextualiseur se met à jour en fonction des données de formattage qui lui sont fournies en amont, ce qui fait que toutes les citations d'un document sont toujours harmonisées et à jour.

Paramétrisation du contextualiseur bib

Il est possible de paramétriser le contextualiseur bib en indiquant pour une citation particulière un préfixe, un suffixe et/ou un localisateur comme un numéro de page. Voici un exemple [\(14, p. 2 version papier\)](#).

Extensibilité du contextualiseur bib

Outre la contextualisation de relevés bibliographiques, le contextualiseur bib peut être utilisé sur n'importe quel type de ressource grâce à un module qui convertit les métadonnées des ressources dans un format standard lisible par celui-ci²⁰.

Voici par exemple la contextualisation bib d'une ressource de type vidéo [\(15\)](#) ou une autre d'une ressource de type site web [\(2\)](#).

20. Il s'agit du format établi par la spécification csl-json [\(19\)](#).

Contextualiseur glossary

Le contextualiseur glossary permet de signaler des entrées de glossaire dans un récit peritext.

Voici un exemple de mention de Michel-Ange .

Il est possible d'utiliser des alias par exemple en citant la même ressource que dans le paragraphe précédent mais en l'appelant Michel-Angelo .

Ce contextualiseur est appelé à évoluer drastiquement, d'abord pour permettre d'attacher du contenu (description, commentaire, image) à chaque entrée de glossaire afin de l'afficher comme un contenu à part entière, ensuite pour s'appliquer de manière plus riche en fonction du type d'entités décrites (personnes, notions, lieux, évènements, ...).

Rendez-vous dans la section « glossaire » de ce document pour voir une liste des mentions de ressources interactive.

Contextualiseur webpage

Le contextualiseur webpage affiche la page web associée à une ressource.

Il peut s'agir d'un lien hypertexte, d'une vignette du site ou encore d'un iframe (site imbriqué) du site.

Page embarquée



Site web hypotheses

Extensibilité du contextualiseur webpage

Outre les ressources webpage, le contextualiseur webpage peut être utilisé sur toutes les ressources qui indiquent une url dans leur métadonnées, ou les ressources bibliographiques qui indiquent une url.

Exemple d'une ressource « dicto » affichée avec le contextualiseur webpage :



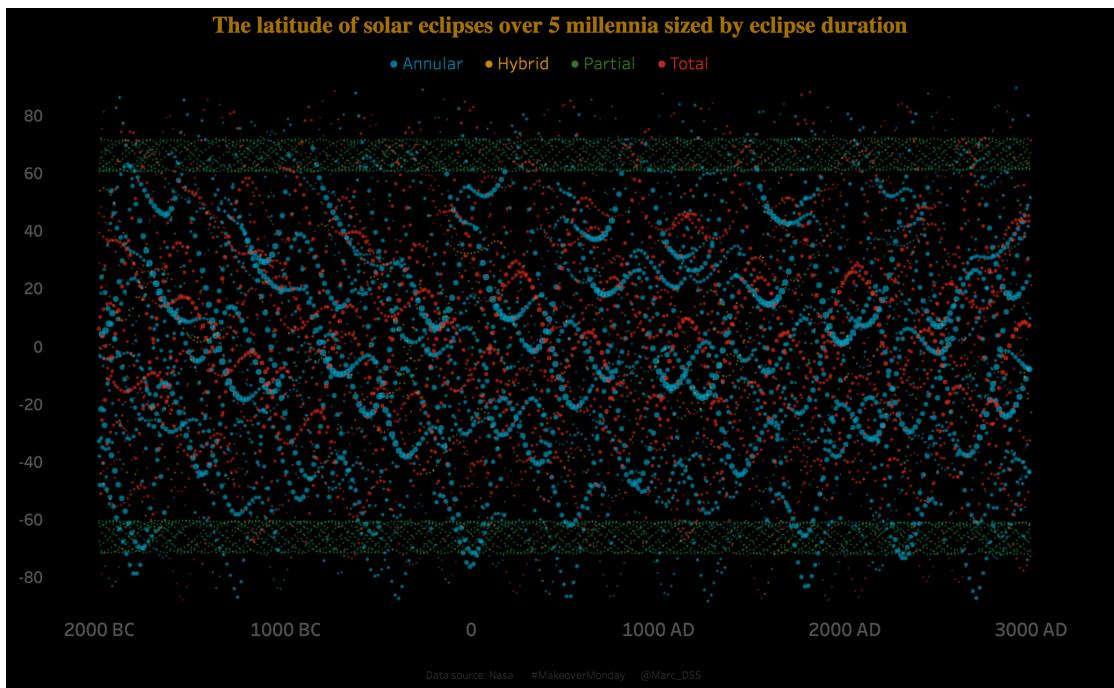
Vectors – une analyse

Analyse par étiquetage des différents articles du journal Vectors

Exemple d'une ressource référence bibliographique affichée avec le contextualiseur webpage (le lien pointe vers fichier pdf, qui sera donc affiché avec les moyens du navigateur — et sa présentation est désactivée pour les supports de type page) :

An Account of Digital Humanities from the AIME Project

Exemple d'une ressource pointant vers une visualisation tableau :



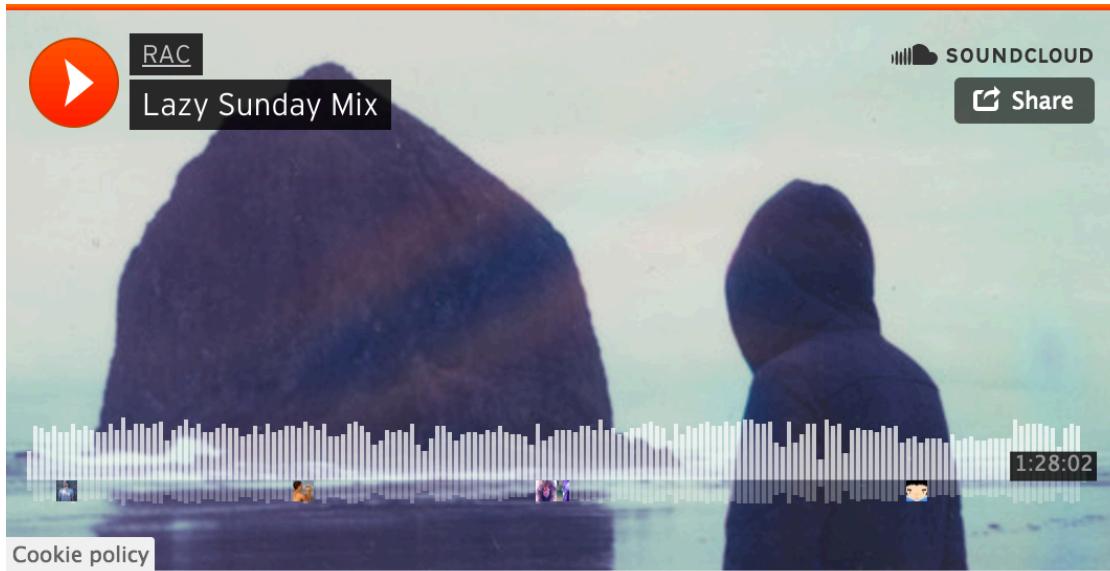
Une visualisation hébergée sur tableau

Accessible à https://public.tableau.com/views/MM34-5MillenniaofSolarEclipses/SolarEclipse?:embed=y&:toolbar=no&:loadOrderID=0&:display_count=yes

Contextualiseur embed

Le contextualiseur embed prend en charge les ressources de type embed, qui ne sont rien d'autre que des portions de code html directement intégrées dans les contenus²¹.

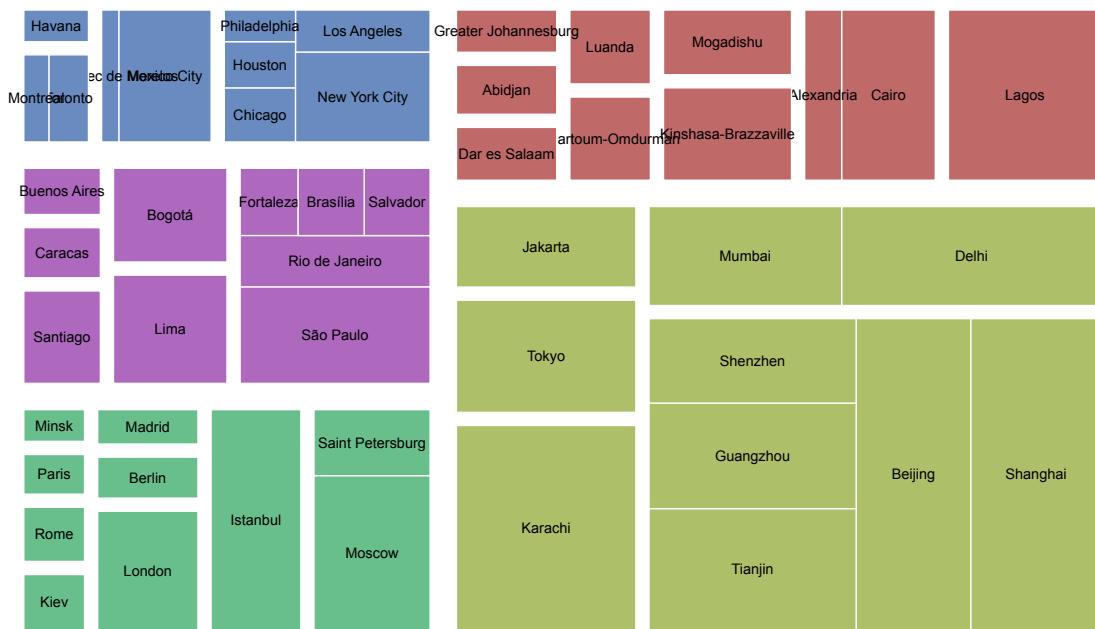
Exemple d'un embed soundcloud :



Un exemple d'embed soundcloud

Un exemple d'embed de code svg, construit avec l'outil raw graphs :

21. Il est à noter que le code javascript contenues dans d'éventuelles balises script injectées selon cette méthode ne sont pas prises en compte.



Exemple d'embed de code svg tiré de raw graphs

 **manovich**
@manovich

Abonné

Stanford professor of Computer Science, Marc Levoy, offers a free course on digital photography. bit.ly/2vDL9mb

À l'origine en anglais



15:35 - 30 août 2017

15 Retweets 25 J'aime

Exemple d'embed basique d'un tweet

Exemple d'embed d'un post facebook :



Centre Pompidou

24 août, 10:53 ·



There's only way to celebrate [#DavidHockney](#)'s birthday: come and see his retrospective at the [Centre Pompidou Paris](#) !

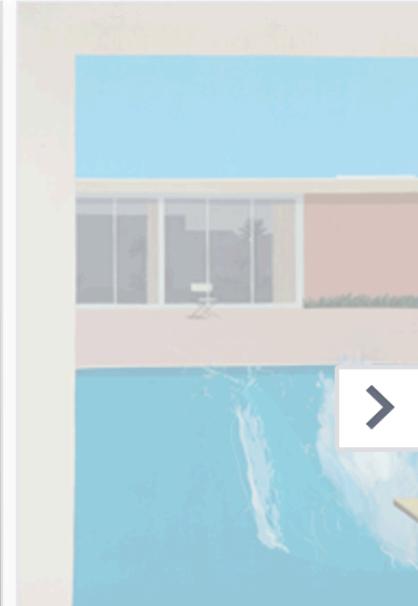
*Buy Your ticket online and enjoy all exhibitions, the Museum, the Kids Gallery and the View of Paris.



Portrait of an Artist

Réserver

by David Hockney



A Bigger Splash

by David Hockney

Exemple d'embed d'un post facebook

Contextualiseur vidéo

Le contextualiseur vidéo joue une ressource sous la forme d'un player vidéo.

Les sources prises en charge pour l'instant sont youtube, vimeo, et une url pointant vers une vidéo html5.

En mode page, le contextualiseur affiche une vignette pointant vers l'url de la vidéo.

En voici un exemple :

Objectified

A peek at the upcoming design documentary "Objectified", by Gary Hustwit, the director of "Helvetica". The trailer features the voices of Jonathan Ive, Andrew Blauvelt, Marc Newson, and Karim Rashid. The song is "I Like Van Halen Because My Sister Says They Are Cool" by El Ten Eleven. Objectified premieres at film festivals and events worldwide starting this March, more info here: <http://www.objectifiedfilm.com>

Source: *Neojaponismo* (youtube: <https://www.youtube.com/watch?v=TyofGn8fiUU>)

Contextualiseur dicto

À propos de Dicto

Dicto est un format de données, un éditeur et un composant de player développés par Robin de Mourat & Donato Ricci dans le cadre d'analyses de transcriptions²².

Dicto permet d'associer des sous-titres (éventuellement taggés) à une vidéo et de produire à partir de ces derniers une application interactive permettant de naviguer dans la vidéo via son texte (et vice versa).

Utiliser le contextualiseur dicto



22. Voir une démonstration d'un format de sortie possible pour dicto ([i](#)) .

Ressource dicto représentée avec le contextualiseur dicto

Analyse par étiquetage des différents articles du journal Vectors

Non-exclusivité du contextualiseur dicto

Une ressource de type dicto est naturellement prise en charge par le contextualiseur dicto. Cela dit, elle peut par ailleurs aussi être prise en charge par le contextualiseur video :



Ressource dicto représentée en vidéo simple

Analyse par étiquetage des différents articles du journal Vectors

Contextualiseur codefiles

Les ressources fichiers de code (codefiles) représentent du code source entré optionnellement divisé en une série de fichiers.

Le contextualiseur fichiers de code affiche une ressource de type fichiers de code (codefiles) sous la forme d'une interface interactive ou statique.

Voici un exemple de contextualisation codefiles :

Contextualiseur table

Le Contextualiseur de type table affiche une ressource de type table (exemple: tirée d'un fichier csv) sous la forme d'un tableau traditionnel.

En version page, il produit un tableau simple.

En version web, il produit un tableau interactif (pagination, recherche, ...)

En voici un exemple :

a	b
A	2
B	4
C	6
D	8
E	2
F	3
G	5

Dataset très simple

Contextualiseur vegalite

Vega (et son petit frère allégé vega lite) est une grammaire de description de visualisations de données interactives et statiques. Il est développé par l'équipe Interactive Data Lab au sein de l'Université de Washington (16) .

Le contextualiseur vegalite permet de contextualiser une ressource de type table sous la forme d'une visualisation de données paramétrée selon la grammaire vegalite.

Utiliser le contextualiseur vega

Soit la ressource suivante :

a	b
A	2
B	4
C	6
D	8
E	2
F	3
G	5

Dataset très simple

Elle est par défaut affichée via le contextualiseur table. La voici désormais affichée avec le contextualiseur vega²³ :

Par défaut, le contextualiseur cherche un état minimal pour afficher la table sous la forme d'une visualisation, pas très intéressant donc.

Ajoutons-lui maintenant le paramètre de spécification suivant :

23. Il est à noter que le module vega rend du svg. Pour une raison inconnue à ce jour cela dit, le module ne fonctionne pas en environnements serveurs ; pour palier à ce problème technique il est pour l'instant possible d'associer une vignette à chaque visualisation afin qu'elle soit utilisée comme image à la place du svg qui devrait normalement s'afficher.

```
{
  "mark": "bar",
  "encoding": {
    "x": {"field": "a", "type": "ordinal"},
    "y": {"field": "b", "type": "quantitative"}
  }
}
```

On obtient :

Visualisations interactives

Outre la réalisation de visualisations simples, vega-lite permet aussi de décrire des interactions et des visualisations composées, plus complexes.

Soit le dataset suivant :

date	price
Jan 1 2000	1394.46
Feb 1 2000	1366.42
Mar 1 2000	1498.58
Apr 1 2000	1452.43
May 1 2000	1420.6
Jun 1 2000	1454.6
Jul 1 2000	1430.83
Aug 1 2000	1517.68
Sep 1 2000	1436.51
Oct 1 2000	1429.4
Nov 1 2000	1314.95
Dec 1 2000	1320.28
Jan 1 2001	1366.01
Feb 1 2001	1239.94
Mar 1 2001	1160.33
Apr 1 2001	1249.46
May 1 2001	1255.82
Jun 1 2001	1224.38
Jul 1 2001	1211.23

date	price
Aug 1 2001	1133.58
Sep 1 2001	1040.94
Oct 1 2001	1059.78
Nov 1 2001	1139.45
Dec 1 2001	1148.08
Jan 1 2002	1130.2
Feb 1 2002	1106.73
Mar 1 2002	1147.39
Apr 1 2002	1076.92
May 1 2002	1067.14
Jun 1 2002	989.82
Jul 1 2002	911.62
Aug 1 2002	916.07
Sep 1 2002	815.28
Oct 1 2002	885.76
Nov 1 2002	936.31
Dec 1 2002	879.82
Jan 1 2003	855.7
Feb 1 2003	841.15
Mar 1 2003	848.18
Apr 1 2003	916.92
May 1 2003	963.59
Jun 1 2003	974.5
Jul 1 2003	990.31
Aug 1 2003	1008.01
Sep 1 2003	995.97
Oct 1 2003	1050.71
Nov 1 2003	1058.2
Dec 1 2003	1111.92
Jan 1 2004	1131.13

date	price
Feb 1 2004	1144.94
Mar 1 2004	1126.21
Apr 1 2004	1107.3
May 1 2004	1120.68
Jun 1 2004	1140.84
Jul 1 2004	1101.72
Aug 1 2004	1104.24
Sep 1 2004	1114.58
Oct 1 2004	1130.2
Nov 1 2004	1173.82
Dec 1 2004	1211.92
Jan 1 2005	1181.27
Feb 1 2005	1203.6
Mar 1 2005	1180.59
Apr 1 2005	1156.85
May 1 2005	1191.5
Jun 1 2005	1191.33
Jul 1 2005	1234.18
Aug 1 2005	1220.33
Sep 1 2005	1228.81
Oct 1 2005	1207.01
Nov 1 2005	1249.48
Dec 1 2005	1248.29
Jan 1 2006	1280.08
Feb 1 2006	1280.66
Mar 1 2006	1294.87
Apr 1 2006	1310.61
May 1 2006	1270.09
Jun 1 2006	1270.2
Jul 1 2006	1276.66

date	price
Aug 1 2006	1303.82
Sep 1 2006	1335.85
Oct 1 2006	1377.94
Nov 1 2006	1400.63
Dec 1 2006	1418.3
Jan 1 2007	1438.24
Feb 1 2007	1406.82
Mar 1 2007	1420.86
Apr 1 2007	1482.37
May 1 2007	1530.62
Jun 1 2007	1503.35
Jul 1 2007	1455.27
Aug 1 2007	1473.99
Sep 1 2007	1526.75
Oct 1 2007	1549.38
Nov 1 2007	1481.14
Dec 1 2007	1468.36
Jan 1 2008	1378.55
Feb 1 2008	1330.63
Mar 1 2008	1322.7
Apr 1 2008	1385.59
May 1 2008	1400.38
Jun 1 2008	1280
Jul 1 2008	1267.38
Aug 1 2008	1282.83
Sep 1 2008	1166.36
Oct 1 2008	968.75
Nov 1 2008	896.24
Dec 1 2008	903.25
Jan 1 2009	825.88

date	price
Feb 1 2009	735.09
Mar 1 2009	797.87
Apr 1 2009	872.81
May 1 2009	919.14
Jun 1 2009	919.32
Jul 1 2009	987.48
Aug 1 2009	1020.62
Sep 1 2009	1057.08
Oct 1 2009	1036.19
Nov 1 2009	1095.63
Dec 1 2009	1115.1
Jan 1 2010	1073.87
Feb 1 2010	1104.49
Mar 1 2010	1140.45

Dataset « sp500 »

Source: Exemples associés au site officiel vegalite

Et la spécification suivante :

```
{
  "$schema": "https://vega.github.io/schema/vega-lite/v2.json",
  "vconcat": [
    {
      "width": 480,
      "mark": "area",
      "encoding": {
        "x": {
          "field": "date",
          "type": "temporal",
          "scale": {"domain": {"selection": "brush"}},
          "axis": {"title": "", "labelAngle": 0}
        },
        "y": {"field": "price", "type": "quantitative"}
      }
    },
    {
      "width": 480,
      "height": 60,
      "mark": "area",
      "selection": {
        "brush": {"type": "interval", "encodings": ["x"]}
      },
      "encoding": {
        "x": {
          "field": "date",
          "type": "temporal",
          "axis": {"format": "%Y", "labelAngle": 0}
        },
        "y": {
          "field": "price",
          "type": "quantitative",
          "axis": {"tickCount": 3, "grid": false}
        }
      }
    }
  ]
}
```

On obtient la visualisation suivante, permettant de sélectionner dans la visualisation du dessous une plage temporelle à visualiser dans le graphique du dessus :

Apprendre à utiliser la grammaire vegalite

Les possibilités offertes par vega sont très importantes et couvrent une grande partie des besoins courants en termes de visualisation de données. La documentation proposée sur le site de vega permet de s'en faire une première idée ([16](#)) .

Pour prévisualiser et tester une spécification vegalite avant de l'intégrer dans un contextualiseur, rendez-vous sur l'outil de développement proposé par ses créateurs ([33](#)) .

Contextualiseur p5 (processing)

Le projet p5.js

Processing est un langage de programmation, une bibliothèque Java, un éditeur et une communauté dédiées au développement d'outil de programmation pour les artistes visuels.

C'est aujourd'hui un langage utilisé et enseigné dans nombre de contextes artistiques et éducatifs.

p5.js est un portage de la syntaxe mise en place par Processing pour le langage javascript.

L'outil permet de créer diverses formes d'images mouvantes et interactives 2D et 3D, et optionnellement d'utiliser les options du navigateur (micro, caméra, ...) pour créer des expériences numériques merveilleuses.

Utiliser le contextualiseur p5

Le contextualiseur p5.js interprète une ressource de type fichiers de code contenant un *sketch* p5 pour en faire une application interactive, stockée dans une balise canvas.

Prenons la ressource de type fichiers de code suivante :

Choisissons maintenant le contextualiseur p5 pour l'afficher. On obtient l'image suivante :

On obtient un canvas dans lequel est représenté un programme processing interprété à partir du code de la ressource ²⁴.

Les possibilités offertes par ce contextualiseur sont nombreuses.

Pour en savoir plus sur les capacités de p5.js, rendez-vous sur le site officiel du projet [\(24\)](#).

²⁴. Pour l'instant, les contextualisations de ce type ne sont pas visibles sur les documents de type codex directement. Il est donc possible de spécifier une image de vignette au niveau du contextualiseur qui sera utilisée en remplacement.

Gabarits

Les gabarits définissent la manière selon laquelle doivent être affichés les contenus d'un récit. Ils concernent à la fois le code html/css de chaque portion de contenu, et l'organisation même des pages/sections pour les documents codex ou des routes pour les documents web.

Par exemple, le présent document est mis en forme au moyen des gabarits *peritext-template-web-garlic* et *peritext-template-codex-garlic*.

Les gabarits sont attachés à un type de générateurs particulier avec lequel ils sont compatibles (pour l'instant les deux types disponibles sont page et web — voir la section générateur).

Gabarits existants

Gabarit Web Garlic

Il s'agit d'un gabarit pour générateurs de type web très simple, fortement calqué sur la structure rhétorique d'un livre académique traditionnel ou d'une thèse.

En ce qui concerne l'interface des contenus, le gabarit présente une colonne principale déroulant les contenus de manière linéaire, et une colonne optionnelle permettant de naviguer dans les différentes contextualisations d'une même ressource.

Il présente également un ensemble de « vues de coupe » dans le document calquées sur les formats imprimés traditionnels, comme une bibliographie/liste de ressources et un glossaire interactifs permettant de prévisualiser et de naviguer dans les contextualisations d'une même ressource.

Ce gabarit est pour l'instant testé avec le générateur suivant : peritext-generator-next.

Gabarit Codex Garlic

Il s'agit d'un gabarit pour générateurs de type page très simple plutôt pensé pour les pdf interactifs et les epub, utilisant les polices Roboto et Merryweather et présentant une table des matières interactives, un glossaire, un index des auteurs, et des références interactives.

Ce gabarit est pour l'instant testé avec les générateurs suivants : peritext-generator-pdf, peritext-generator-epub.

Générateurs

Les générateurs produisent un *output* exploitable pour dans un format donné donné à partir d'un récit peritext et d'un ensemble d'options²⁵.

Ils se divisent en trois grandes familles :

- les générateurs de type page sont dédiés à tous les formats de sortie structurés autour de pages à tourner. Ils peuvent correspondre à des rendus par impression ou sur écran, indifféremment. Techniquement, ces formats de sortie n'intègrent pas de code javascript²⁶, mais peuvent intégrer des hyperliens.
- les générateurs de type web produisent un site web interactif ou applications assimilées (application mobile, application bureau, ...)
- les générateurs de type fragment produisent des contenus pouvant être consommées dans des environnements de type réseaux sociaux, basés sur la publication étalée dans le temps de plusieurs petits fragments des contenus.

Générateur pdf (famille page)

Ce générateur produit un fichier pdf interactif et imprimable, avec le moteur PrinceXML([2](#)) qui permet de gérer toutes les règles CSS print²⁷.

Générateur epub (famille page)

Ce générateur produit un fichier epub pour liseuses, prenant en charge les hyperliens internes et acceptant du code css personnalisé. Ce générateur est appelé à être grandement amélioré à l'avenir.

Générateur pandoc (famille page)

Ce générateur produit des fichiers correspondant à des formats « vernaculaires » couramment utilisés pour décrire les documents.

25. *Techniquement, ces bibliothèques exposent une fonction qui consomme un récit au format json et des modules de contextualisation et de gabarit, pour produire en sortie un output utilisable (e.g. « fichier.pdf »).*
26. Des exports de type page tels que epub3 permettraient la prise en charge du javascript, mais ceci est pour l'instant laissé de côté dans un souci de simplicité.
27. Note: prince xml est la seule technologie non libre utilisée dans un module peritext. Le choix de Prince XML est motivé par sa prise en charge hors pairs de la spécification w3c pour le css print([3](#)). Les autres technologies de conversion pdf en javascript, basées sur webkit ou phantom, ne prennent pas (ou très mal en charge) ces spécifications.

Comme il se fonde sur la technologie pandoc ([40](#)) pour convertir une représentation « page » du document, ce générateur prend en charge la conversion vers un large spectre de formats de fichiers :

- odt (open office)
- docx (word)
- icml (indesign)
- latex
- context
- rst
- tei
- ...

Il est à noter que ce générateur est expérimental et fournit pour l'instant des fichiers de qualité variable selon les formats.

Générateur next (famille web)

Ce générateur produit un site web statique (série de fichiers html) au moyen du framework next.js ([4](#)) . Le framework next.js a la particularité de générer des applications *isomorphiques* : l'intégralité du contenu de chaque page est écrit dans des fichiers html (pour l'usage des machines d'indexation, et des navigateurs interdisant javascript), mais le comportement et l'affichage est aussi pris à charge via une application javascript pour les clients qui le peuvent afin de profiter d'un maximum d'interactivité et de subtilité dans la présentation. Cette technologie permet donc d'allier indexabilité des contenus et interactivité des interfaces.

Ce générateur permet également d'utiliser le site statique qu'il produit comme API, puisqu'il expose les différentes parties du récit qu'il génère sous la forme de fichiers json séparés pouvant être requêtés par une application tierce (exemple d'une vue sur les ressources du présent document : [\(32\)](#)).

Générateur twitter (famille fragment)

Ce générateur produit des extraits du document aptes à être publiés sur twitter sous la forme de couples texte-image. L'image représente un extrait de contenu, et le texte peut contenir, au choix de l'auteur, des hashtags dérivés du contenu de l'extrait, un lien vers une page de mise en contexte, et du texte personnalisé. En complément, l'écosystème peritext propose un exemple d'implémentation de ce générateur avec l'application peritext-server-twitterbot.

Rendez-vous sur le compte twitter @peritext-ecosystem pour voir ce générateur en action.

Tweets **Tweets & réponses** **Médias**

 **rob th 38** @testtwitroth · 2 min
Doc excerpt mysite.com/sections/4a51c... # [#peritext](#)-twitter
🌐 À l'origine en anglais

Cela dit, Peritext se distingue une première fois de ses voisins dans la mesure où il permet également aux utilisateurs de se faire "éditeurs-designers" et disposer de divers moyens très détaillés pour mettre en forme leur récit pour chacun des supports disponibles, en fonction de leur corpus, de leur argument et de leur projet intellectuel.

Peritext - publication multimodale orientée contexte – Introduction

Reply Retweet Like Share

Exemple de tweet produit par le générateur twitter

Liste d'idées pour des contributions au niveau des générateurs

- Générateur (côté client) de code pouvant être embarqué dans un blog wordpress
- Générateur d'extraits sous des formats consommables par les APIs de facebook, instagram, & telegram ...
- Générateur de site *monopage* adapté aux articles scientifiques et devoirs d'étudiants
- Générateur de livre audio via une API de text-to-speech
- Générateur d'application de « spatialisation » de récits avec Unity ou mobilizing.js
- ...

Éditeurs

De la même manière que les modules de Peritext sont censés être assemblés différemment selon différents cas d'usage, les applications permettant d'éditer des documents peritext sont censées être multiples et adaptées à des contextes spécifiques.

Ovide est éditeur compatible avec l'écosystème Peritext. Il a vocation à faire office de « preuve de concept » plutôt que d'application en soi, étant donné que sa configuration technique est davantage tournée vers l'expérimentation que vers une utilisation à grande échelle comme un outil « sur étagère »²⁸.

Pour les besoins de la démonstration, les fonctionnalités d'export en pdf, epub, et web sont pourvues par un serveur équipé des modules générateurs correspondants.

Une version de démonstration d'Ovide sera peut-être mise en ligne à terme.

^{28.} Ainsi, Ovide utilise par exemple l'intégralité des contextualiseurs à disposition dans l'écosystème, ce qui ne devrait pas être le cas pour des applications réelles, qui seraient configurées en fonction des besoins spécifiques.

Contribuer à peritext

L'intégralité du code de peritext est vouée à être placée sous licences libres LGPL-3 & CECCIL-B. L'ensemble du projet est déposé sur [github](#), au sein de l'organisation [peritext](#) (30) .

Pour l'instant, les modèles de données associés aux ressources, contextualiseurs, gabarits de peritext ne sont pas stabilisés, car une utilisation prolongée est nécessaire pour les gérer et les fixer de manière pertinente²⁹. Il en est de même pour l'API des modules, qui évolue en fonction de l'expérience tirée des premières implémentations du projet.

Ainsi, les possibilités de contribution à peritext sont amenées à s'élargir petit à petit, une fois le projet stabilisé et apte à fonctionner comme une plateforme pour d'autres initiatives.

Créer/hacker un contextualiseur

Un contextualiseur doit exposer l'ensemble des éléments suivants :

- une série de composants exposés au propriétés BlockStatic, BlockDynamic, InLineStatic ou InLineDynamic pour chacun des cas de figure pris en charge par le contextualiseur
- une propriété metadata présentant les propriétés du contextualiseur, les modules disponibles, et les options qu'il propose à l'utilisateur
- une propriété defaultCss qui expose le code css par défaut à injecter quand le contextualiseur est implémenter

Exemple de l'API du contextualiseur `vegalite` :

... plus d'indications bientôt.

Créer/hacker un gabarit/template

L'API des gabarits varie selon qu'ils se destinent à des générateurs de type « page » ou « web ».

Pour les deux, un gabarit doit exposer :

- une propriété metadata
- un ensemble de composants correspondant aux différentes vues/sections d'un récit (à normaliser)

29. Voir la section « issues » du répertoire [github peritext-core](#) (27) pour voir la liste des modifications prévues. Les propositions d'autres modifications sont d'ailleurs ouvertes !

- un composant centralisateur permettant d'afficher tout le récit (par exemple pour une utilisation du module comme aperçu)

Exemple de l'API du contextualiseur web-garlic :

... indications plus précises bientôt

Créer/hacker un générateur

Un générateur est un module qui prend en argument un récit et des modules à consommer (un module gabarit, des modules contextualiseurs, une liste de traductions de termes) et produit un fichier.

Voici par exemple l'API du générateur epub :

... indications plus précises bientôt

Questions fréquemment posées

Puis-je utiliser Peritext pour mon projet éditorial actuel ?

En cet instant, non. Ou alors à vos (terribles) risques et périls ! Peritext est pour l'instant à l'état expérimental, et la spécification de son format est appelée à évoluer de manière non-compatible avec la version actuelle. Cela signifie que les modules du projet, qui évoluent en fonction de l'état de la spécification Peritext, ne seront à un certain point plus compatibles avec les documents développés aujourd'hui dans l'écosystème. Ce n'est donc en aucun cas une technologie à utiliser dans un contexte de production.

D'autre part, les outils et modules de code de peritext ne sont pas encore testés pour fonctionner dans le monde réel et devraient ne pas supporter certains cas d'usage non pris en compte (exemples : documents très gros, fichiers de données ou images très importantes, ...).

Une communication officielle devrait être organisée pour la sortie de la version stabilisée de Peritext.

Quelle est la différence entre Peritext et LaTeX ?

Le projet [LaTeX](#) (36) vise à fournir des outils pour une production documentaire très bonne qualité typographique, et se fondant sur l'abstraction entre les contenus et leur forme. Il bénéficie par ailleurs d'une importante communauté qui développe des modules intitulés *packages* venant augmenter la syntaxe initiale et permettant de fournir des services aussi variés que la gestion bibliographique, la création d'un glossaire, l'intégration d'images ... voici un exemple d'extrait de document écrit selon la syntaxe établie par LaTeX :

LaTeX présente l'avantage d'être un projet ayant une grande profondeur temporelle et une communauté d'utilisateurs extrêmement importante.

Cela dit, les besoins suivants ont motivé le projet de développer peritext comme une alternative à des projets tels que LaTeX :

- insuffisance de la balise \cite pour prendre en charge de manière fine et circonstanciée l'intégration d'éléments extérieurs dans un document scientifique

- meilleure prise en compte des formats de rendus autres que le format « page » (pdf, ...)
- meilleure gestion du système documentaire plus fine

Peritext reste moins puissant et performant que LaTeX sur beaucoup de plans — notamment sur le plan typographique —, mais il s'adapte peut-être mieux aux cas d'usage précis de publication multimodale de textes enrichis.

Quelle est la différence entre Peritext et des outils de conversion de documents ?

Pandoc ([40](#)) est un fantastique outil de conversion de documents, depuis une myriade de formats de fichiers vers une autre myriade de formats de fichiers. Html-2-print ([41](#)), quant à lui, est un très intéressant projet centré sur l'écriture d'un document en html pour un rendu ultérieur en pdf ou impression.

Ces projets offrent de très bonne solutions pour la conversion de documents classiques d'un format vers un autre. Cela dit, ils se fondent sur un mode de description des documents qui n'induit pas d'abstraction au niveau des contenus permettant de distinguer l'usage de ressources et leur mode de contextualisations. De ce fait, ils ne profitent pas des possibilités documentaires & hypertextuelles induites par le modèle mis en place par Peritext et ne permettent pas la prise en charge éléments complexes et interactifs mis en oeuvre via le modèle RCC.

Peritext propose néanmoins un générateur basé sur Pandoc pour produire de tels documents « vernaculaires » (markdown, html statique, word, ...) à partir de récits peritext.

Il est également prévu à terme de prendre en charge l'import de documents écrits dans un logiciel d'édition de texte courant pour constituer une base de document peritext.

Quelle est la différence entre peritext et des outils de gestion de chaîne éditoriale ?

Les chaînes éditoriales sont des « outils qui accompagnent la production documentaire de masse » en se fondant sur « une mise en évidence des structures documentaires présentes dans un corpus » ([42, p. 24](#)). Ce concept s'incarne dans une diversité de projets de recherche menés notamment au sein de l'Université de Technologies de Compiègne et de produits dont principalement la suite scenari([43](#)) . On trouve dans cette suite une multitude de logiciels destinés à des domaines aussi variés que des supports de cours, des dictionnaires, des documentations techniques, etc.

Peritext a été développé en partie avec les principes établis par les recherches sur les chaînes éditoriales en tête. Il se base aussi sur une structuration documentaire particulière pensée spécifiquement pour la publication multimodale de textes enrichis.

De ce point de vue, on pourrait plus ou moins dire que Peritext est dédié à la mise en place de chaînes éditoriales spécialisées dans l'édition de textes enrichis académiques, journalistiques ou pédagogiques. Un niveau d'abstraction moindre, et des choix technologiques différents au niveau des technologies utilisées (javascript) et du format de stockage des données json) diffèrent tout de même peritext assez fortement de technologie comme scenari.

Concernant les visualisations, pourquoi ne pas utiliser des formats existants pour la description de récits agrémentés de visualisations ?

Aujourd'hui, la prolifération des données numériques dans nos sociétés, et conséquemment la demande croissante d'outils de « narration de données » dans des domaines comme le journalisme ou la recherche universitaire, a motivé la fabrication d'outils et de syntaxes dédiées à l'écriture de textes agrémentés de visualisation de données. Parmi ceux-ci figurent des formats tels que rmarkdown⁽³⁷⁾³⁰ ou, plus récemment, idyll⁽³⁸⁾³¹. Ces derniers permettent d'écrire « en ligne », dans le corps même du texte, des descriptions de visualisations qui seront ensuite transformées en images au moment de la compilation.

Peritext ne se fonde pas sur l'une de ces syntaxes parce qu'elles se focalisent sur la question de la visualisation et ne permettent pas d'établir une logique documentaire globale, permettant de traiter les visualisations comme des « contextualisations » parmi d'autres. De plus, elles demandent l'apprentissage d'une syntaxe assez complexe (ou la maîtrise par ailleurs d'un langage de programmation comme Javascript ou R), ce qui en fait des solutions ne convenant qu'à une partie des personnes ayant besoin de ce type d'outils.

Par rapport à ces dernières, on pourrait dire que Peritext correspond à des usages à la fois plus « lourds » — multimodaux et dans lesquelles la question documentaire est importante — et plus « légers » — sans recours d'emblée à des codes pour l'écriture.

30. fondé sur une combinaison des langages markdown pour les contenus, et r pour les visualisations.

31. fondé sur une extension du langage markdown.

Quelle est la différence entre Peritext et des plateformes d'édition de récits basés sur des jeux de données ?

Silk est une plateforme commerciale qui permet d'exposer un dataset sur un site web en fournissant divers moyens de le visualiser, et en produisant des textes qui le commentent et le présentent selon plusieurs perspectives [\(39\)](#). Cela en fait un outil très intéressant pour des publications centrées sur la production d'un jeu de données nouveau.

Peritext se démarque de ce type de plateformes de par sa nature ouverte et modulaire, et non réduite à la seule question de la visualisation de données, et enfin par sa dimension multimodale.

Pourquoi les récits Peritext sont-ils structurés de manière séquentielle, et pas comme des bases de données ?

Peritext est fondé sur le choix critique d'organiser ses contenus selon une liste de *sections* organisées linéairement en fonction d'un *sommaire*. On aurait pu opter pour des modes d'organisation beaucoup plus libres et délinéarisés, permettant par exemple de proposer plusieurs cheminement à partir des mêmes contenus, ou de constituer les contenus comme un réseau, sans qu'une « séquence » parmi d'autre ne prévale dans l'interface proposée aux lecteurs³².

Ce choix est basé sur le pari que les formats conventionnels de l'édition savante, formés par les technologies de l'impression et du codex, comme la « monographie », « l'article scientifique », « le chapitre », vont perdurer encore longtemps — et conserver une dimension séquentielle même s'ils sont présentés sur des sites web ou autres environnements numériques.

Le projet de peritext est donc de permettre à un auteur de produire du contenu pour ces formats courants sans effort particulier, tout en permettant — sans effort particulier non plus — d'expérimenter des formes alternatives de mise en scène. Peritext fait donc le choix d'une structure séquentielle dans l'écriture des contenus pour convenir à la fois à des formats conventionnels et à des formats non conventionnels, et se calant sur les contraintes structurelles des premiers.

Ce choix est fait au prix d'une restriction dans la liberté d'organisation des documents pour les nouveaux formats, qui peut cela dit être contournée par le design de gabarits et de générateurs précis interprétant la « séquence » d'un document selon des règles alternatives, ou même ne prenant pas du tout en compte cette dernière.

32. Voir à ce propos la plateforme Scalar qui introduit la notion de « cheminement » (« *path* ») pour proposer un format de récit académique complètement délinéarisé [\(17\)](#) .

À propos

Peritext est un projet libre développé dans un contexte de recherche en design et esthétique des médias numériques. Il a également beaucoup à voir avec les sciences de l'information et de la communication, les sciences de la documentation et l'informatique. Il s'agit à l'origine du volant pratique d'une recherche doctorale menée par Robin de Mourat sur les formats d'édition savante³³.

Le projet est né dans le contexte de la construction des humanités numériques françaises d'une part, et d'un intérêt croissant des institutions et des étudiants de design français vis-à-vis de la recherche académique d'autre part. Ces deux milieux (humanités numériques et recherche en design) partagent la nécessité de mettre en scène et discuter de manière élaborée une diversité de documents hétéroclites impliqués dans les recherches concernées.

Histoire du projet / remerciements

- 2013-2016 : Robin de Mourat bénéficie d'un contrat doctoral de 3 ans de la part du Ministère de l'Enseignement Supérieur et de la Recherche via l'ENS Cachan pour conduire des activités de recherche et d'enseignement auprès de Nicolas Thély à l'université Rennes 2, dans le cadre du groupe de recherche MONADE (21). La naissance de Peritext doit beaucoup à ce contrat et à cet environnement bienveillant et stimulant.
- Janvier 2014 — Mars 2015: la recherche doctorale conduit à une phase d'observation participante sur le projet « Enquête sur les Modes d'Existence » (7) , investigation philosophique soutenue par un écosystème d'instances imprimées et de sites web complémentaires. La conduite d'entretiens commentés et d'une enquête numérique(8) sur le projet donne un aperçu contextualisé des problématiques et des potentialités de la publication multimodale dans le contexte académique. Une partie importante des principes de conception de Peritext découlent de cette rencontre et des nombreux échanges qu'elle a occasionné.
- 2015 : rédaction par Donato Ricci & Robin de Mourat d'un texte de recherche en design portant sur le projet EME, dont émerge la centralité du concept de *contexte* dans les enjeux actuels de l'édition savante (13).
- Juin 2015 : conduite du workshop « Open AIME », dédié à la confrontation de l'écosystème EME (ses composantes techniques et processus de design) aux cas d'étude d'autres chercheurs (11) — qui permet de cerner les perspectives de recherche qui seront développées par peritext.
- Janvier 2016 : Publication collective de l'article « Clues, Anomalies, Understanding »(12) qui sur la forme propose une première itération de peritext, via la réalisation d'un site web généré à partir de contenus écrits selon une syntaxe markdown+json (9).

33. Thèse actuellement en cours de rédaction ...

- Été 2016 : Première spécification & implémentation de Peritext sous la forme d'une bibliothèque javascript unique utilisant des données encodées en markdown & BibTeX et stockées dans un dossier de fichiers textes [\(10\)](#)
- Septembre 2016 : présentation du projet Peritext à la Maison des Sciences de l'Homme de Lille
- 2016–2017 : développement de Peritext dans le cadre du projet Quinoa pris en charge par le médialab Sciences Po [\(5\)](#) pour le programme FORCCAST [\(6\)](#) — les spécifications du format sont simplifiées et rationalisées à cette occasion
- Mai–Septembre 2017 : exposition du projet Peritext au festival international de design graphique de Chaumont lors de l'exposition « une cartographie de la recherche en design graphique » [\(22\)](#)
- Juin–Août 2017 : refactorisation du projet comme un écosystème ouvert de modules, écriture de la spécification opérationnelle du format peritext en json-schema, début de développement de l'éditeur Ovide.

Références

1.

DE MOURAT, Robin. Vectors — une analyse [en ligne]. Disponible à l'adresse : <http://dicto-formats.herokuapp.com/share/transcription/vectors-une-analyse?display=fullscreen&showtags=1&showrailway=1&leftcolumn=50>
Analyse par étiquetage des différents articles du journal Vectors

Mentions: , p. 38, p. 39, p. 31

2. Site web hypotheses. [en ligne]. Disponible à l'adresse : <http://hypotheses.org/>

Mentions: , p. 30

3.

DE MOURAT, Robin et RICCI, Donato. Dicto — site web. [en ligne]. 2017. Disponible à l'adresse : <http://dicto-playground.herokuapp.com/>
Site web de démonstration de l'outil Dicto

Mentions: , p. 38

4.

HUSTWIT, Gary. Objectified. [en ligne]. Disponible à l'adresse : <https://www.youtube.com/watch?v=TyofGn8fiUU>
A peek at the upcoming design documentary « Objectified », by Gary Hustwit, the director of « Helvetica ». The trailer features the voices of Jonathan Ive, Andrew Blauvelt, Marc Newson, and Karim Rashid. The song is « I Like Van Halen Because My Sister Says They Are Cool » by El Ten Eleven.

Mentions: , p. 37, p. 28

5. *Dataset très simple*.

Mentions: , p. 41, p. 42, p. 0, p. 0, p. 18, p. 18, p. 0

6. Prince XML. [en ligne]. Disponible à l'adresse : <https://www.princexml.com/>

Mentions: , p. 51, p. 28

7.

Spécification W3C pour le css print. [en ligne]. Disponible à l'adresse : <https://www.w3.org/TR/css-print/>

Mentions: , p. 51, p. 25

8. Framework javascript next.js. [en ligne]. Disponible à l'adresse : <https://zeit.co/blog/next3>

Mentions: , p. 52

9.

Médialab Sciences Po. [en ligne]. Disponible à l'adresse : <http://www.medialab.sciences-po.fr/fr/>

Mentions: , p. 62

10. Programme FORCCAST. [en ligne]. Disponible à l'adresse : <http://controverses.org/>

Mentions: , p. 62

11. An Inquiry Into Modes Of Existence. [en ligne]. Disponible à l'adresse : <http://modesofexistence.org/>

Mentions: , p. 61

12. AIME, expérimentation de quelques instruments d'écoute. [en ligne]. Disponible à l'adresse : <http://modesofexistence.org/-recherche-en-design-re-penser-les-normes-de-la-production-scientifique/>

Mentions: , p. 61, p. 23

13. *Un exemple d'embed soundcloud.*

Mentions: , p. 33

14. *Exemple d'embed de code svg tiré de raw graphs.*

Mentions: , p. 34

15. *Exemple d'embed basique d'un tweet.*

Mentions: , p. 35

16. *Exemple d'embed d'un post facebook.*

Mentions: , p. 36

17. Clues, Anomalies, Understanding — digital companion. [en ligne]. 2015. Disponible à l'adresse : <http://modesofexistence.org/anomalies/>

Mentions: , p. 61

18. Première implémentation de Peritext. [en ligne]. Disponible à l'adresse : <https://github.com/peritext/peritext>

Mentions: , p. 62

19. Compte-rendu du workshop «open aime». [en ligne]. 2015. Disponible à l'adresse : <http://modesofexistence.org/open-aime-workshop-11-12-june-paris/>

Mentions: , p. 61

20. RICCI , Donato, DE MOURAT , Robin, LECLERO, Christophe et LATOUR, Bruno. Clues, Anomalies, Understanding. Detecting underlying assumptions and expected practices in the Digital Humanities through the AIME project. [en ligne]. 2015. Visible Language. Disponible à l'adresse : <http://visiblelanguagejournal.com/issue/172/article/1172>

Mentions: , p. 61

21.
RICCI , Donato, DE MOURAT, Robin et BOULANGER , Pierre-Laurent. AIME: opening the context of a Humanities inquiry. [en ligne]. 2015. Disponible à l'adresse : <http://echappees.esapyprenees.fr/content/2-numeros/3-numero3/5-an-account-of-digital-humanities-from-the-aime-project/echappees-an-account-of-digital-humanities-from-the-aime-project.pdf>

Mentions: , p. 61

22.
RICCI, Donato et DE MOURAT, Robin. An Account of Digital Humanities from the AIME Project [en ligne]. 2016. Revue échappées. Disponible à l'adresse : <http://echappees.esapyprenees.fr/content/2-numeros/3-numero3/5-an-account-of-digital-humanities-from-the-aime-project/echappees-an-account-of-digital-humanities-from-the-aime-project.pdf>

Mentions: , p. 31, p. 28, p. 28, p. 28

23.Vega lite. [en ligne]. Disponible à l'adresse : <https://vega.github.io/vega-lite/docs/>

Mentions: , p. 43, p. 48

24.*Exemple de ressource codefiles.*

Mentions: , p. 0

25.*Mon beau réseau roi des forêts.*

Mentions:

26.Scalar. [en ligne]. Disponible à l'adresse : <http://scalar.usc.edu/>

Mentions: , p. 23, p. 60

27.
Ted Nelson demonstrates Xanadu Space. [en ligne]. Disponible à l'adresse : https://www.youtube.com/watch?v=En_2T7KH6RA
Ted Nelson présente le logiciel Xanadu Space qui spatialise en trois dimensions un système hypertexte.

Mentions: , p. 24, p. 23

28.
La norme OpenURL et la technologie Context Objects in Span. [en ligne]. Disponible à l'adresse : <http://blog.stephanepouyllau.org/116>

Mentions: , p. 23

29.CitationStyles.org. [en ligne]. Disponible à l'adresse : <http://citationstyles.org/>

Mentions: , p. 22, p. 28

30.*Ressource processing js.*

Mentions: , p. 0, p. 0

31.ManyLines. [en ligne]. Disponible à l'adresse : <http://tools.medialab.sciences-po.fr/manylines>

Mentions:

32.DE MOURAT, Robin. *Exemple de présentation SVG*.

Mentions:

33.DE MOURAT, Robin. *Histoire du Fort et des combats de 1870-1871. Une histoire du fort d'Issy*

Mentions:

34.DE MOURAT, Robin. *Histoire de la visualisation.*

Cette présentation traite de l'histoire de la visualisation

Mentions:

35.*Le modèle de document Peritext.*

Mentions:, p. 16

36.

Projet MONADE. [en ligne]. Disponible à l'adresse : https://www.mshb.fr/projets_mshb/monade/2289/

Mentions:, p. 61

37.

LE SIGNE. Une cartographie de la recherche en design graphique — Biennale de design graphique 2017. [en ligne]. Disponible à l'adresse : <http://www.centrenationaldugraphisme.fr/fr/cig/page/biennale-de-design-graphique-2017/expositions/une-cartographie-de-la-recherche-en-design-graphique>

Mentions:, p. 62

38.

Une visualisation hébergée sur tableau. [en ligne]. Disponible à l'adresse : https://public.tableau.com/views/MM34-5MillenniaofSolarEclipses/SolarEclipse?:embed=y&:toolbar=no&:loadOrderID=0&:display_count=yes

Mentions:, p. 32

39.

COMMUNAUTÉ P5.JS. Site officiel de p5.js. [en ligne]. Disponible à l'adresse : <https://p5js.org/>

Mentions:, p. 49

40.

Conventions de conversion de données pour le projet draft-js. [en ligne]. Disponible à l'adresse : <https://draftjs.org/docs/api-reference-data-conversion.html>

Mentions:, p. 15

41.

BUSH, Vannevar et OTHERS. As we may think. *The atlantic monthly*. 1945. Vol. 176, n° 1, pp. 101-108.

Mentions:, p. 23

42. *Le cabinet d'Ole Worm. 1655.*

Reproduction de la gravure conservée à Modène représentant un cabinet de curiosités

Mentions: , p. 8

43.

DE MOURAT, Robin. Répertoire peritext-core sur github. [en ligne]. Disponible à l'adresse : <https://github.com/peritext/peritext-core>

Mentions: , p. 14, p. 7, p. 55

44.

Peritext core — documentation technique. [en ligne]. Disponible à l'adresse : <https://peritext.github.io/peritext-core/>

Mentions: , p. 14

45.

INTERNET ENGINEERING TASK FORCE. Spécification JSON Schema. [en ligne]. Disponible à l'adresse : <http://json-schema.org/>

Mentions: , p. 14

46.

L'organisation peritext sur github. [en ligne]. Disponible à l'adresse : <https://github.com/peritext/>

Mentions: , p. 55

47. API du générateur epub.

Mentions: , p. 0

48. API du contextualiseur vegalite.

Mentions: , p. 0

49. API du gabarit << web-garlic >>.

Mentions: , p. 0

50.

Exemple d'utilisation du générateur next comme API. [en ligne]. Disponible à l'adresse : <http://peritext.robindemourat.com/static/generated/resources.json>

Mentions: , p. 52

51.

EUGÈNE DE, Gayffier. Herbier forestier de la France. Reproduction par la photographie... des principales plantes ligneuses qui croissent spontanément en forêt. Description botanique... [en ligne]. 1868. Disponible à l'adresse : <http://gallica.bnf.fr/ark:/12148/btv1b86260659/f19.double?>

Mentions: , p. 11

52.

Outil et de prévisualisation d'édition de la grammaire vega. [en ligne]. Disponible à l'adresse : <https://vega.github.io/editor/#/edited>

Mentions: , p. 48

53.

DE MOURAT, Robin. Éditeur Ovide. [en ligne]. Disponible à l'adresse : <http://ovide.robindemourat.sh/>

Mentions: , p. 25

54. *Les modules de peritext.*

Mentions: , p. 20

55.

FACEBOOK. Site web de présentation de la technologie open source react. [en ligne]. Disponible à l'adresse : <https://facebook.github.io/react/>

Mentions: , p. 7, p. 25

56.

Projet LaTeX — site officiel. [en ligne]. Disponible à l'adresse : <https://www.latex-project.org/>

Mentions: , p. 57

57. *Exemple de document écrit avec la syntaxe LaTeX.*

Mentions: , p. 0

58.

Dataset « sp500 » [en ligne]. Disponible à l'adresse : https://vega.github.io/editor/#/examples/vega-lite/overview_detail

Mentions: , p. 43, p. 0

59. rmarkdown. [en ligne]. Disponible à l'adresse : <http://rmarkdown.rstudio.com/>

Mentions: , p. 59

60. idyll. [en ligne]. Disponible à l'adresse : <https://idyll-lang.github.io/>

Mentions: , p. 59

61.

Silk — exemple d'une histoire interactive. [en ligne]. Disponible à l'adresse : <http://google-self-driving-car-incidents.silk.co/>

Mentions: , p. 60

62. Pandoc. [en ligne]. Disponible à l'adresse : <https://pandoc.org/>

Mentions: , p. 58, p. 52, p. 11

63. *Les implications rhétoriques du modèle RCC.*

Mentions: , p. 17

64.

OPEN SOURCE PUBLISHING. html 2 print. [en ligne]. Disponible à l'adresse : <http://osp.kitchen/tools/html2print/>

Mentions: , p. 58

65.

ARRIBE, Thibaut. *Conception des chaînes éditoriales: documentariser l'activité et structurer le graphe documentaire pour améliorer la maîtrise de la rééditionnalisation*. 2014.

Mentions: , p. 58

66. Communauté Scenari.org. [en ligne]. Disponible à l'adresse : <https://scenari.org/>

Mentions: , p. 58

67. *Exemple de tweet produit par le générateur twitter*.

Mentions: , p. 53

68. Ovide - screencast. [en ligne]. Disponible à l'adresse : <https://vimeo.com/237039631>

Mentions: , p. 0

Glossaire

contextualiseur

Mentions: , p. 26, p. 26, p. 27, p. 27

CSS

Mentions: , p. 25, p. 25

format json

Mentions: , p. 13, p. 14

gabarit

Mentions: , p. 26, p. 25, p. 25, p. 50, p. 50, p. 50, p. 50, p. 25, p. 25

générateur

Mentions: , p. 26, p. 26, p. 50

LaTeX

Mentions: , p. 57, p. 57

Michel-Ange

Mentions: , p. 29, p. 29

Peritext - publication multimodale orientée contexte

Une introduction

Robin De Mourat

Peritext est un format de données et un écosystème ouvert de briques logicielles qui permettent de fabriquer des documents enrichis de visualisations, de vidéos et autres ressources numériques, et de les publier simultanément dans des formats très différents tels que imprimé, web, pdf, epub, twitter ...