



Bâtiment 10A,
15 rue Naudet
CS 10 207,
33 175 GRADIGNAN CEDEX

RÉGIS MARCAHOSSE
6, impasse des marguerites
33 170 GRADIGNAN

Titre Professionnel (niv. III / bac +2) : Développeur web web mobile.
Dossier de projet :

Création du site web du fablab de l'IUT de Bordeaux

4 novembre 2019

Tuteur du projet :
Pierre Grangé-Praderas



Responsable Pédagogique :
Pierre Guillaume

bâtiment C
110, rue Achard
33 300 BORDEAUX

Sommaire

Remerciements	3
Résumé du projet	4
Vue d'ensemble	4
Objectifs	4
Caractéristiques	4
Introduction	5
Contexte du projet	5
Environnement du projet	6
Réalisations	9
Elaboration de la maquette	9
Création des pages web de la partie vitrine	15
Une organisation de travail spécifique	15
Une déformation des développeurs web “moderne” le Javascript	16
Une structure de page en HTML5 et CSS3 “moderne”	21
La création des pages web des projets	22
L’organisation et le fonctionnement de la mise à jour des pages projets.	29
L’organisation des fichiers et méthode de travail	29
Mise à jour des pages projets : l’accès aux données back-end	32
Mise à jour des pages projets : utilisation des données back-end	37
Conclusion générale sur le projet	43
Annexe	44

Remerciements

Je tiens à remercier toutes les personnes qui ont contribué, au succès de ce projet professionnel, que ce soit par les échanges, l'accompagnement ou l'aide qu'ils m'ont apporté.

Tout d'abord, j'adresse mes remerciements à mon tuteur, **Pierre Grangé-Praderas**, pour son accueil, le temps passé ensemble et le partage de son expertise. Sa vision pédagogique de la réalisation de ce projet m'a permis de m'accomplir complètement en tant que développeur web et même au-delà. J'ai pu développer des compétences d'administrateur système, d'administrateur serveur et de maîtrise graphique.

Je remercie également toutes les personnes du fablab. En premier lieu, le fab manager, **Jean-Baptiste Bonnemaïson**, pour son accueil et sa disponibilité. Ensuite, tous les adhérents, bénévoles et chercheurs qui incarnent l'idée du partage et de l'entraide des fablabs.

Résumé du projet

Vue d'ensemble

J'ai effectué un stage de plus de deux mois au Fablab de l'IUT de Bordeaux. Cette association loi 1901, m'a demandé de réaliser leur nouveau site internet (le 3^e).

L'univers particulier du fablab et ses actions pédagogiques, à travers la réalisation de projets concrets avec des outils de fabrication assistée par ordinateur, est très captivant.

Le statut et l'univers de cette structure a orienté les spécifications de leurs besoins de façon originale et stimulante.

Objectifs

1. **Rapatrier les logiciels et composants web sur le serveur.**
2. **Légitimer la pédagogie et l'action du Fablab dans la techno-sphère d'internet.**
3. **Permettre l'utilisation du site à des fins pédagogiques (modification, expérimentation).**
4. **Avoir une facilité de maintenance/ mise à jour importante.**
5. **Avoir la surface d'attaque la plus minime possible.**

Caractéristiques

Le site internet comprendra un site vitrine très majoritairement en CSS3 et HTML5 (environ 20 pages).

Une partie dynamique reprenant et mettant en avant les projets documentés du fablab. Cette partie dynamique est assurée par un script shell Bash qui régénère de façon planifiée toutes les pages web.

Un blog sous WordPress.org pour la "documentation" narrative sur des ateliers réalisés au fablab.

Introduction

Contexte du projet

J'ai effectué ma formation de développeur web web mobile à l'infrep de Bordeaux. Pour cette formation j'ai souhaité réaliser un projet professionnel dans un univers qui m'intéresse à plus d'un titre, les fablabs.

En effet, les fablabs sont des laboratoires de fabrications numériques collaboratifs. On y construit des prototypes ou des objets de façon expérimentale en modifiant l'existant, ou dans un processus d'essai/ erreur / réussite. En utilisant des machines numériques, commandées par ordinateur et avec des fichiers informatiques. Ouvert et collaboratif, les projets peuvent être repris et modifiés entre les milliers de fablabs de par le monde.

Parmi les cinq fablabs de Bordeaux, j'ai choisi celui de Coh@bit liée à l'IUT de Bordeaux.

En effet, le fablab Coh@bit a poussé le partage et l'échange des connaissances vers l'acquisition et la reconnaissance de compétences par les projets réalisés en son sein. Ceci pour un public très varié allant de l'étudiant décrocheur à l'adulte en reconversion. Cette excellence pédagogique est reconnue au point qu'un Diplôme Universitaire est en cours de création. L'obtention de ce diplôme sera possible par l'acquisition de compétences attachées à des micro-référentiels et unités d'enseignement mis en application dans les projets du fablab.

Cela est remarquable pour une structure associative (loi 1901) qui a tout juste cinq ans et deux salariés. Pendant ces cinq ans les outils du fablab ont été construits, hébergés, choisis de façon disparate, en fonction des moyens disponibles. Je reviendrai sur cet aspect dans la partie "environnement du projet".

Ma mission découle directement de cette spécificité et de cet historique. Je dois mettre en place le nouveau site web du fablab (le 3^e). Ce dernier est un outil qui doit refléter leur politique. Légitimer le fablab dans la techno-sphère d'internet. Rapatrier et centraliser les outils. Accompagner la pédagogie effectuée au fablab. Tout cela en permettant la mise à jour des contenus et la maintenance générale du site facilement, et en ayant la surface d'attaque la plus minime possible.

Environnement du projet

Avant tout il est important de rappeler les spécificités du fablab.

Il fait partie de l'association à but non lucratif **creative house** dont les moyens financiers et humains sont limités.

En plus de ses missions de laboratoire numérique collaboratif, il oeuvre dans la médiation numérique (protection des données, alternative aux services capturant les données personnelles...) et le développement des logiciels libres.

Enfin il réalise des projets avec des partenaires comme l'INRA, OpenBeeLab (association d'apiculteur ...).

Les compétences utilisées ou acquises au sein du fablab sont des langages informatiques bas niveau proche du C (projet arduino), des lignes de commande et de programmation bash (linux, raspberry pi) et l'utilisation de logiciels libres.

Les compétences amenées par les adhérents et stagiaires changent régulièrement.

Les outils du fablab ont été construits en fonction des moyens et possibilités. Ainsi le fablab possède :

- un serveur hébergeant le Redmine (outil de gestion de projet) avec un serveur NAS pour sauvegarder toutes les données,
- un calendrier cloud hébergé par Google pour la réservation des machines.
- un calendrier cloud et des outils collaboratifs hébergés par Aquilinet (FDN),
- d'autres outils collaboratifs hébergés par Framasoft (dégooglisons internet),
- un blog WordPress hébergé par BlueHost,
- un site internet WordPress (réalisé en plugins) hébergé par l'IUT de Bordeaux ...

Aujourd'hui le fablab doit recentrer et concrétiser sa nouvelle politique de reconnaissance pédagogique et technique. Pour cela le fablab souhaite réaliser un nouveau site web et recentraliser certains outils.

Ainsi nous avons établi les besoins et les premières spécifications du cahier des charges. En prenant en compte l'existant, le retour d'expérience, les spécificités et les contraintes ainsi que la culture du libre.

Cahier Des Charges Fonctionnel

Besoins:**langues : FR/EN****Communication**

- les projets et les sources
- Informations statiques
 - * horaires ouverture vacances, locaux, matériel, adhésions, charte, statuts, partenaires,
- Évènements :
 - * recrutements pour projets, ateliers, interventions, calendrier évènements, calendrier découpe laser

Documentation

- projets et les sources
- microréférentiels pédagogiques
- plateforme de CVs / profils

Gestion de projets (interne)

- wiki, tickets, gantt, calendriers projets

Administration / gestion de structure

- agendas (Aquilenet, nextcloud)
- Calendrier découpe laser (Google)
- quotidien

architecture web au 29/03/19

cohabit.fr

redirection <https://www.iut.u-bordeaux.fr/cohabit/>
 hébergement : serveur IUT
 NDD: OVH
 CMS : WordPress
 MX: OVH unset

projets.cohabit.fr

hébergement : fablab
 NDD: OVH
 CMS : WordPress
 MX: OVH unset
 gestion: DSI IUT

cplusmoins.org/detournement

hébergement : Bluehost
 NDD: Bluehost
 CMS : WordPress
 MX: OVH unset
 gestion: Association C+/- (pgp)

médias sociaux

facebook
 linkdn
 mastodon
 amstragram
 twitter
 ulab

salon Matrix (riot.im)

- chat

Aquilenet

- agendas / pads

FRAMASOFT

- gitlab,

besoins et architecture web existant

Après 5 années et 2 sites internet, l'infrastructure informatique est morcelée, l'aspect pédagogique et technologique n'est pas mis en avant. Le site actuel présente :

- coté back-end : une grande difficulté de sécurisation, de maintenance et d'adaptation/ mise à jour.
- coté front-end : un manque de lisibilité, un parcours utilisateur difficile à travers les menus et la présence d'informations erronées ou dépassées.

La cause principale est la construction du site web uniquement par plugins dans WordPress et l'absence de toute documentation.

Après ce constat, les premières spécifications générales ont été décidées.

- Utilisation et construction du site sans framework ni langage informatique abstrait. HTML5, CSS3, peu voir pas de Javascript.

Ainsi le besoin pédagogique, c'est-à-dire la modification possible par les adhérents, et la contrainte de compétences fluctuantes sont satisfaits.

- Une sécurité par une surface d'attaque minimale (serveurless) et une compartimentation des outils de nouveau centralisés. le site internet, le Redmine et le blog seront installés sur le serveur dans des répertoires différents et avec des bases de données différentes et non interconnectées (MariaDB pour le blog et PostgreSQL pour Redmine). Pas de base de données pour le site internet, pas de formulaire.

Ainsi on assure la sécurité au niveau d'une intrusion dans le serveur à partir du site internet. On centralise de nouveaux les outils sur le serveur.

- Une mise à jour facilitée, un respect de la culture du libre et des données. Aucune librairie de type bootstrap, JQuery, aucun outil ou plugins avec capture de données personnelles.

Ainsi la sécurité par non mise à jour d'une librairie ou d'un plugin est assurée sans demande de temps et de compétences humaines. La lisibilité et la compréhension sont assurées par l'utilisation Du HTML5 et CSS3, l'utilisation de WordPress pour le blog avec pas ou très peu de plugins.

- Reprise des données par la mise à jour des projets documentés. Utilisation d'un script Shell Bash avec un lancement planifié.

Ainsi la compréhension, la mise à jour et la possibilité de modification par les adhérents sont assurées par un langage informatique qui se retrouve dans les activités et projets du fablab. La sécurité est assurée par une action planifiée et non induite à partir du site internet et donc de l'extérieur.

Ces spécifications se traduisent par la mise en place sur le serveur du fablab d'un site internet comprenant un site vitrine très majoritairement en CSS3 et HTML5 (environ 20 pages). Une partie reprenant et mettant en avant les projets documentés du fablab (par le script shell bash). Un blog (sous WordPress). C'était ma mission.

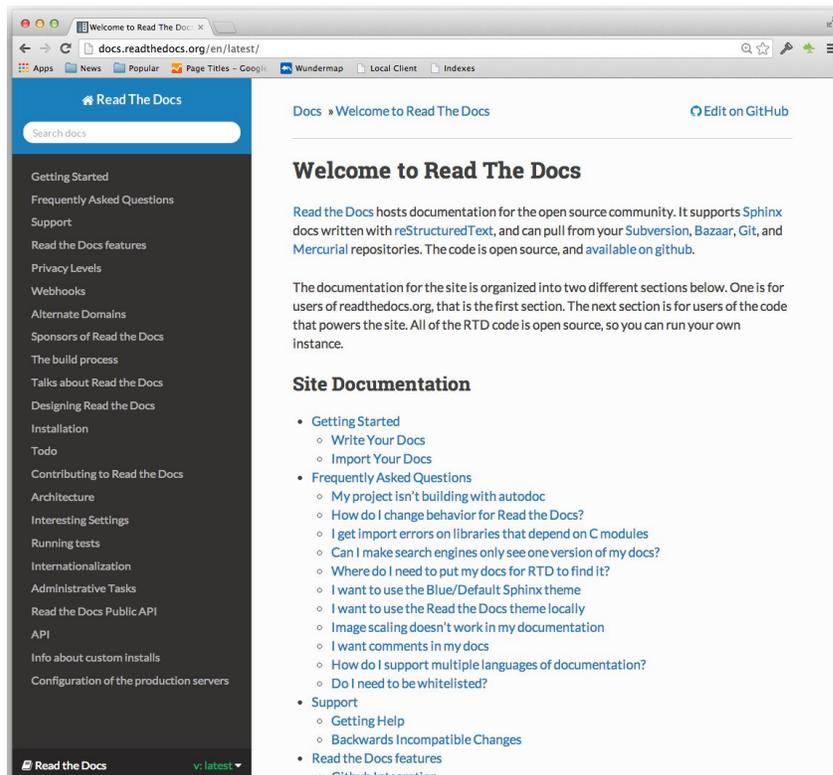
Je vais présenter les étapes suivantes :

- L'élaboration de la charte graphique et de la maquette.
- La création des pages web de la partie vitrine.
- La création des pages web des projets.
- L'organisation et le fonctionnement de la mise à jour des pages projets.

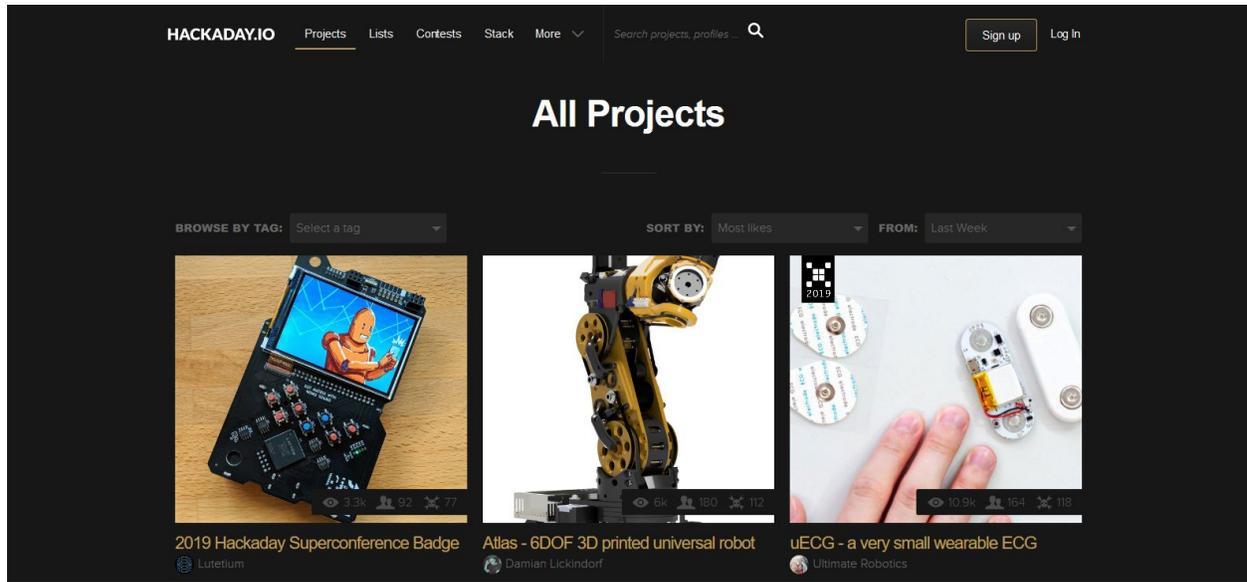
Réalisations

Élaboration de la maquette

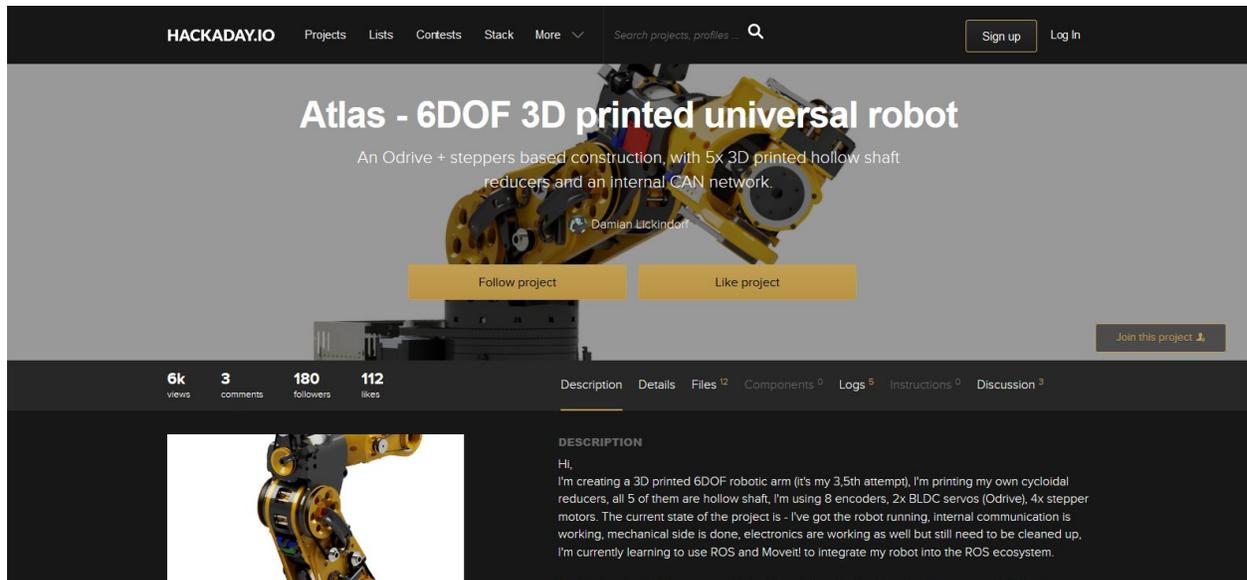
La charte graphique reprend les couleurs du logo vert et blanc et le noir en fond. Pour établir la maquette, je me base sur le besoin “légitimer le fablab dans la techno-sphère d’internet”. Je réalise une étude des sites de documentation et sites de projets partagés.



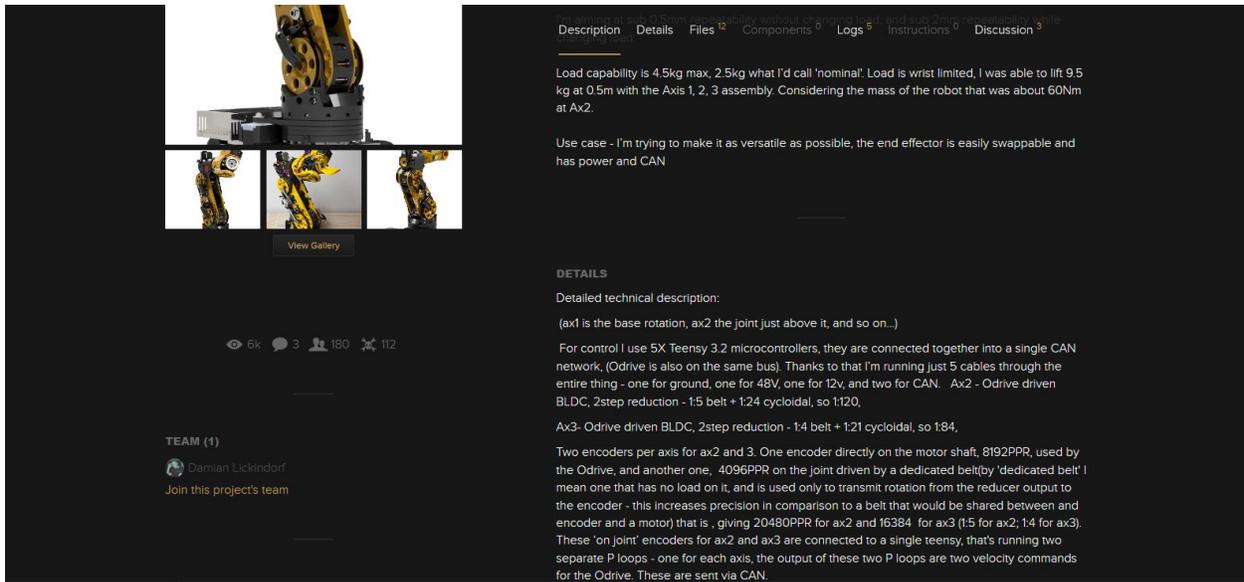
site de documentation



site de projets partagés hackaday.io



détails d'une page projet hackaday.io



Description Details Files 12 Components 0 Logs 5 Instructions 0 Discussion 3

Load capability is 4.5kg max, 2.5kg what I'd call 'nominal'. Load is wrist limited, I was able to lift 9.5 kg at 0.5m with the Axis 1, 2, 3 assembly. Considering the mass of the robot that was about 60Nm at Ax2.

Use case - I'm trying to make it as versatile as possible, the end effector is easily swappable and has power and CAN

DETAILS

Detailed technical description:

(ax1 is the base rotation, ax2 the joint just above it, and so on...)

For control I use 5X Teensy 3.2 microcontrollers, they are connected together into a single CAN network, (Odrive is also on the same bus). Thanks to that I'm running just 5 cables through the entire thing - one for ground, one for 48V, one for 12v, and two for CAN. Ax2 - Odrive driven BLDC, 2step reduction - 1:5 belt + 1:24 cycloidal, so 1:120.

Ax3- Odrive driven BLDC, 2step reduction - 1:4 belt + 1:21 cycloidal, so 1:84.

Two encoders per axis for ax2 and 3. One encoder directly on the motor shaft, 8192PPR, used by the Odrive, and another one, 4096PPR on the joint driven by a dedicated belt(' dedicated belt' I mean one that has no load on it, and is used only to transmit rotation from the reducer output to the encoder - this increases precision in comparison to a belt that would be shared between and encoder and a motor) that is , giving 20480PPR for ax2 and 16384 for ax3 (1:5 for ax2, 1:4 for ax3). These 'on joint' encoders for ax2 and ax3 are connected to a single teensy, that's running two separate P loops - one for each axis, the output of these two P loops are two velocity commands for the Odrive. These are sent via CAN.

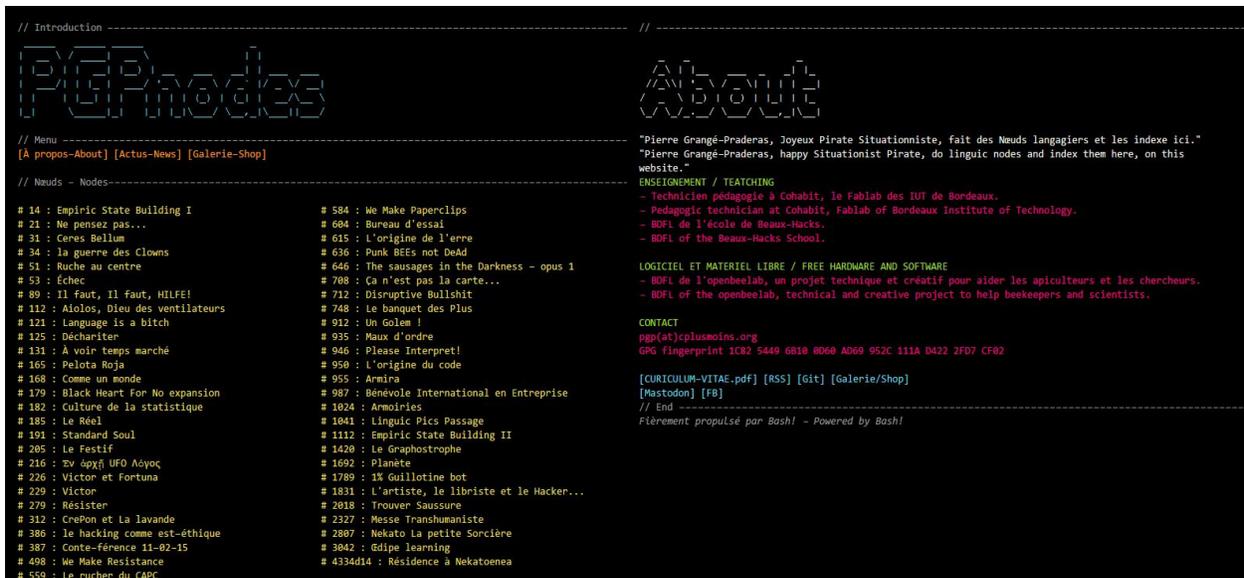
TEAM (1)

 Damian Lickindorf
Join this project's team

6k 3 180 112

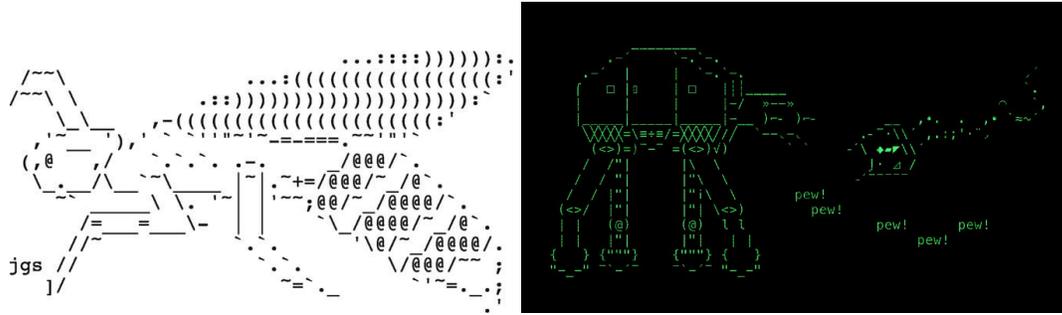
détails d'une page projet hackaday.io

De cette étude est retenue une présentation en vignettes, simple. Mon tuteur trouve que le côté documentation est minimaliste, fait penser à l'ASCII ART et à l'informatique version ligne de commande.



```
// Introduction
-----
P&P nodes
-----
// Menu
[A propos-About] [Actus-News] [Galerie-Shop]
-----
// Neuds - Nodes-----
# 14 : Empiric State Building I          # 584 : We Make Paperclips
# 21 : Ne pensez pas...                 # 604 : Bureau d'essai
# 31 : Ceres Bellum                    # 615 : L'origine de l'erre
# 34 : la guerre des Clowns            # 636 : Punk BEEs not DeAd
# 51 : Ruche au centre                 # 646 : The sausages in the Darkness - opus 1
# 53 : échec                           # 708 : Ca n'est pas la carte...
# 89 : Il faut, Il faut, HILFE!        # 712 : Disruptive Bullshit
# 112 : Aiolos, Dieu des ventilateurs   # 748 : Le banquet des Plus
# 121 : Language is a bitch            # 912 : Un Golem !
# 125 : Déchariter                     # 935 : Maux d'ordre
# 131 : À voir temps marché            # 946 : Please Interpret!
# 165 : Pelota Roja                    # 950 : L'origine du code
# 168 : Comme un monde                # 955 : Armina
# 179 : Black Heart For No expansion    # 987 : Bénévole International en Entreprise
# 182 : culture de la statistique       # 1024 : Armoires
# 185 : Le Réel                        # 1044 : Linguis Pies Passage
# 191 : Standard Soul                  # 1112 : Empiric State Building II
# 205 : Le Festif                      # 1420 : Le Graphostrophe
# 216 : Τη ζωή UFO Αόρατο            # 1692 : Planète
# 226 : Victor et Fortuna              # 1789 : 1% Guillotine bot
# 229 : Victor                         # 1831 : L'artiste, le libraire et le Hacker...
# 279 : Résister                       # 2018 : Trouver Saussure
# 312 : CrePon et La lavande            # 2327 : Messe Transhumaniste
# 386 : le hacking comme est-éthique    # 2807 : Nekato La petite Sorcière
# 387 : Conte-Férence 11-02-15        # 3042 : Edipe Learning
# 498 : We Make Resistance              # 4334d1d : Résidence à Nekatoenea
# 559 : Le rucher du CAPC
-----
// About
-----
About
-----
"Pierre Grangé-Praderas, Joyeux Pirate Situationniste, fait des Neuds langagiers et les indexe ici."
"Pierre Grangé-Praderas, happy Situationist Pirate, do linguistic nodes and index them here, on this
website."
-----
ENSEIGNEMENT / TEACHING
- Technicien pédagogie à Cohabit, le Fablab des IUT de Bordeaux.
- Pédagogic technicien at Cohabit, Fablab of Bordeaux Institute of Technology.
- BDFL de l'école de Beaux-Hacks.
- BDFL of the Beaux-Hacks School.
-----
LOGICIEL ET MATERIEL LIBRE / FREE HARDWARE AND SOFTWARE
- BDFL de l'openbeelab, un projet technique et créatif pour aider les apiculteurs et les chercheurs.
- BDFL of the openbeelab, technical and creative project to help beekeepers and scientists.
-----
CONTACT
pgr(at)plusmoins.org
GPG fingerprint 1C82 5449 6B10 8D69 AD69 952C 111A D422 2F07 CF02
-----
[CURRICULUM-VITAE.pdf] [RSS] [git] [Galerie/Shop]
[Mastodon] [FB]
// End -----
Fièrement propulsé par Bash! - Powered by Bash!
```

Un des sites de mon tuteur



Exemples d'ASCII ART

En conclusion nous déterminons que le site sera sur fond noir, minimaliste avec des menus clairs et rappelant l'informatique avec une organisation et une navigation par vignette. Mon tuteur impose la typo IBM PLEX, qui doit être en local. Enfin, suite à l'étude des besoins en informations et du contenu du site actuel, les différentes catégories de pages web, leurs liens et leurs contenus (ou modification de contenu) sont déterminées avec mon tuteur et le fab manager. Mon tuteur valide ma proposition tant pour la page web de présentation des projets que la page web de présentation d'un projet, en se basant sur le site hackaday.io.

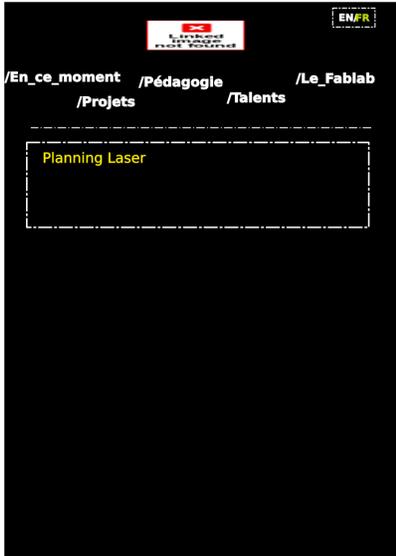
J'apprends à utiliser Inkscape et Gimp pour créer les maquettes graphiques et fonctionnelles, les éléments graphiques, logos...

Je réalise des maquettes en HTML5 et CSS3. J'intègre les demandes de modifications de mon tuteur jusqu'à la validation des maquettes.

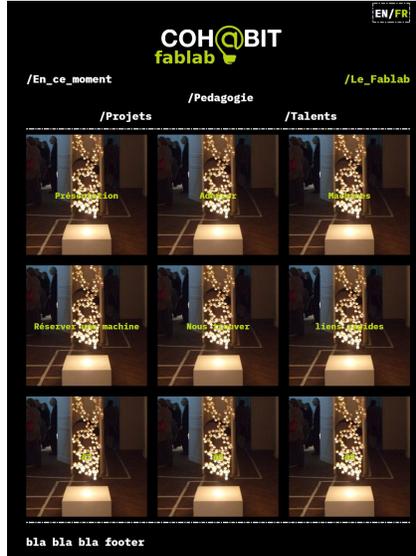
Pour ces nombreux cycles les principales modifications sont :

- Abandon des pointillées (finalement gêne la lisibilité).
- Changement des couleurs des liens actif de vert à jaune (lisibilité).
- Utilisation d'un autre logo du fablab (design).
- Utilisation d'un symbole pour le changement de langue (confusion de lien actif).

La page suivante montre l'évolution de la maquette graphique, la page d'après montre l'architecture des pages web du site, leurs liens et leurs urls.



maquette graphique inkscape



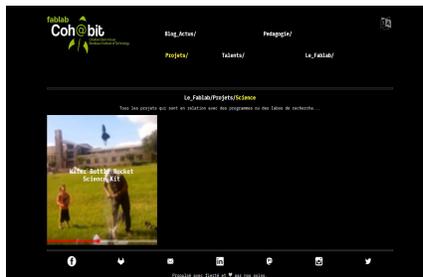
maquette HTML5 CSS3



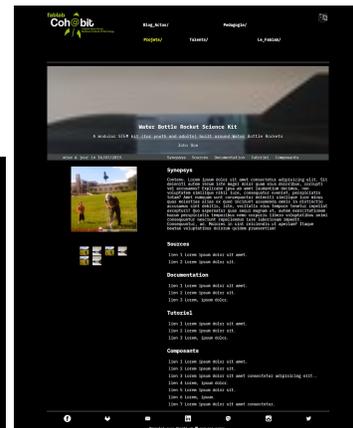
maquette HTML5 CSS3 intermédiaire



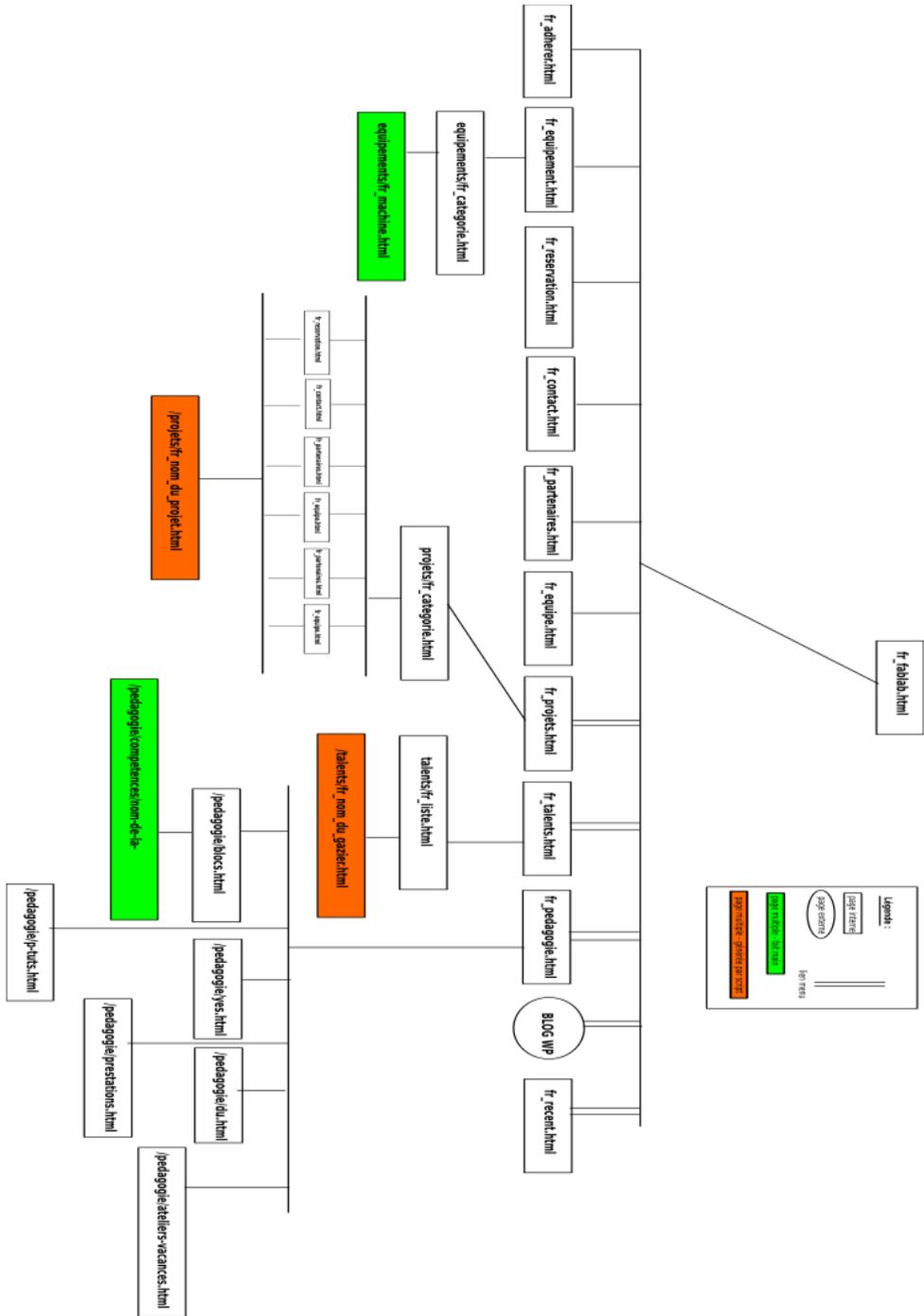
aspect final du site



maquette projets par catégories



maquette projet



maquette fonctionnelle des pages web et de leurs liens

Création des pages web de la partie vitrine

La création des pages web vitrine répondent aux cahiers des charges vu précédemment:

- Charte graphique.
- Menu et navigation en vignette.
- Typo IBM Plex.
- Réalisation en HTML5 et CCS3 sans Js et autres librairies.
- Le site vitrine représente une vingtaine de page web.
- La hiérarchie des pages web est majoritairement sur 3 niveaux.
- Méthode serverless de génération des pages.
- Site en desktop responsive.

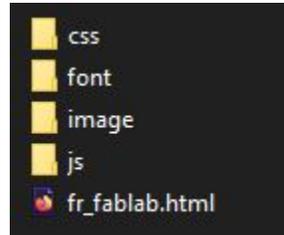
Le contenu et la majorité des éléments graphiques doivent m'être fournis.

Une organisation de travail spécifique

Je déduis de ces éléments une première difficulté. A savoir l'organisation du travail. En effet, toutes ces pages web (vitrine) doivent être réalisées à la main en HTML5 et CSS3. Mon tuteur veut toutes les pages web, sans création ou reprise d'élément par un programme tournant sur le serveur et liant le back-end au front-end. Nous voilà revenus dans les années fin 90-2000.

Je m'organise donc pour valider la page d'accueil complètement avant de la dupliquer pour les autres pages web. En effet, la charte graphique rend la majorité des pages web similaires et je limite le rechercher/remplacer et la reproduction de modification sur près de vingt pages à chaque fois.

J'utilise visual studio code et je crée le répertoire de travail website avec les répertoires et le fichier suivants :



Je travaille sur la page fr_fablab.html, jusqu'à sa validation avant duplication pour créer les modèles des pages de niveau inférieur.

En conclusion cette organisation de travail, s'approche d'une gestion de projet en mode agile. En effet, j'effectue de nombreux cycles courts, (réalisation /approbation /modification /validation) sur la page fr_fablab.html.

Une déformation des développeurs web "moderne" le Javascript

Lors de la création du HTML5 et CSS3 de cette page, j'identifie très vite une deuxième difficulté, ma déformation professionnelle de développeur web "moderne".

En effet, dès la création du menu, je pense à réaliser une toggle class pour la coloration des liens actifs. Cependant du Javascript est nécessaire pour une toggle class, ce qui est interdit par le cahier des charges. Je constate la même difficulté pour réaliser d'autres éléments de la page web ou lors de recherche. En effet, le javascript est la première intention pour les animations de mes vignettes et l'affichage du choix de langue. La recherche sur internet de solution alternative présente un éventail de solution incorporant du javascript, d'où l'importance de bien cibler les termes de recherche (no js / css only ...).

Pour respecter les cahiers des charges, j'alterne entre des solutions HTML et CSS des années 1990 - 2000 et le CSS3 avancé (animation, keyframes, transitions).

- Couleurs des liens actifs et choix des langue. Après des recherches il est possible de retrouver l'effet toggle d'une class liée avec JS, en utilisant les attributs HTML checkbox et la pseudo-classe css :checked. Il en est de même pour afficher le choix de langues. Voir annexe source mdn <https://developer.mozilla.org/en-US/docs/Web/CSS/:checked>.

Cependant, l'utilisation de checkbox complexifie le HTML5, le rendant moins compréhensible pour un adhérent qui aurait pour mission de le modifier. Ceci est la spécification la plus importante du cahier des charges et c'est de cette dernière que découle la spécification de la limitation des langages utilisés.

C'est pourquoi, je décide pour la couleur des liens du menu d'utiliser une classe "activeLink" sur le nom du menu correspondant à la page active. La couleur et son changement sont réalisés en CSS en utilisant les pseudo-classes :hover :active :focus.

De ce choix, chaque page sera individualisée. Après avoir exposé les avantages et inconvénients à mon tuteur, cette solution est retenue (parce qu'elle oblige à la rigueur et que les estimations en démontrent la pertinence).

Estimation de la modification des 12 headers :

temps = environ 3 min X fréquence de changement 2/an soit 6 min dans l'année avec un besoin de compétences faible.

Estimation de la création d'une automatisation :

temps environ 2 heures avec une compétence élevée et une perte de pédagogie.

De même, pour le choix des langues je choisis l'utilisation des balises nav, ul, li côté html et de gérer l'apparition des choix de langues en CSS avec un display none et la pseudo-classe :hover.

```

<nav class="lang-choice">
  <ul>
    <li>
      <a href="#">
        
      </a>
      <ul>
        <li>
          <a href="#">Anglais</a>
        </li>
        <li>
          <a href="#" class="activeLink yellow">Francais</a>
        </li>
      </ul>
    </li>
  </ul>
</nav>

```

HTML choix des langues

```

/* Menu deroulant choix des langues */

.lang-choice li ul {
  display: none;
  width: 90px;
}

.lang-choice a {
  display: block;
  font-family: "ibm_plex_monoregular", Arial, Helvetica, sans-serif;
  font-weight: normal;
  font-size: 1.3vw;
  padding: 6px;
}

.lang-choice li:hover ul {
  display: block;
  position: absolute;
}

```

CSS choix des langues

- Animation des vignettes. Au passage de la souris sur la vignette le titre monte, le fond s'assombrit et un texte descriptif apparaît. Pour ne pas utiliser le javascript, j'ai choisi d'utiliser les balises HTML `<a>` et leur propriété css `:hover`. Ensuite l'animation est gérée par les propriétés de CSS de transition et de position.

```
<article id="v1" class="vignette">
  <a href="https://projets.cohabit.fr/fablab/fr_adherer.html">
    
    <h2 class="vignette-titre">
      Adhérer
    </h2>
    <div class="vignette-text">
      <div class="text">
        Tout pour adhérer. La charte, le bulletin
        d'inscription, la
        démarche...
      </div>
    </div>
  </a>
</article>
```

Structure HTML des vignettes

```
/*hover style */
.vignette {
  position: relative;
  width: 80%;
  height: auto;
}
.vignette-image {
  opacity: 1;
  display: block;
  width: 125%;
  height: auto;
  transition: 0.5s ease;
  backface-visibility: hidden;
}
```

Gestion de l'effet des vignettes au passage de la souris

```
/* animation text vignette */
.vignette-text {
  transition: 0.5s linear;
  opacity: 0;
  position: absolute;
  width: 100%;
  height: auto;
  top: 60%;
  left: 12%;
  text-align: justify;
}
/* animation titre vignette*/
.vignette-titre {
  transition: 0.5s ease;
  position: absolute;
  width: 100%;
  height: auto;
  top: 40%;
  left: 12%;
  text-align: center;

  color: white;
  font-size: 1.9vw;
  font-weight: normal;
}
```

Gestion de l'animation du titre et apparition du texte descriptif des vignettes

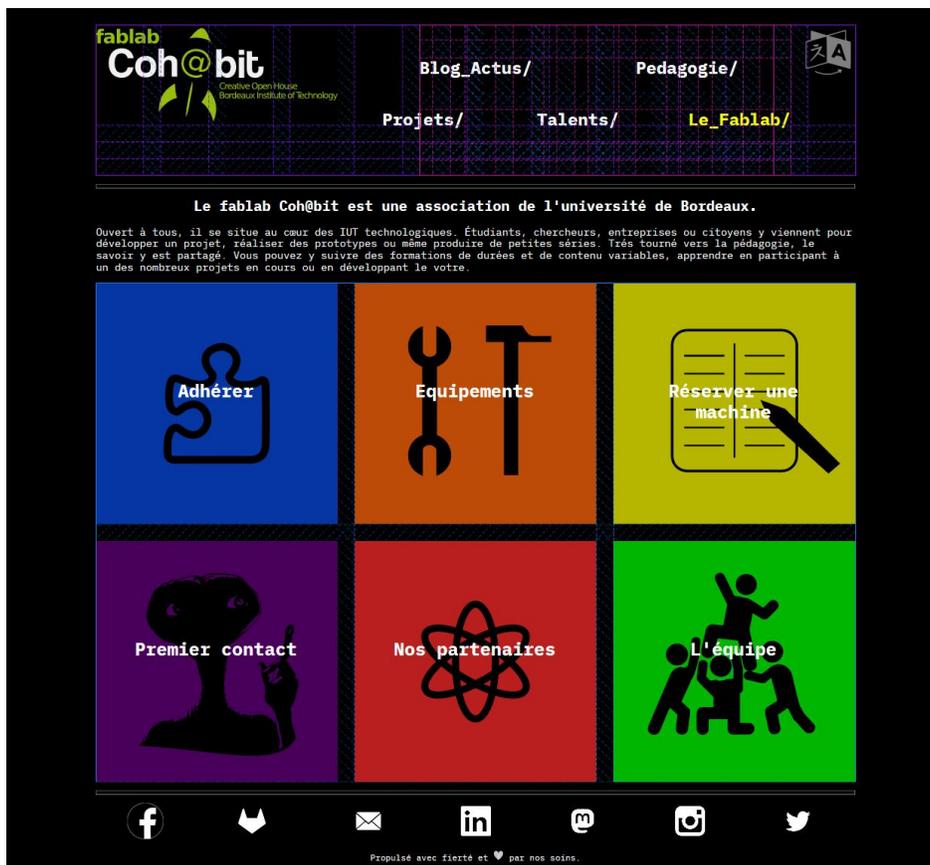
Des essais ont été nécessaires pour déterminer la meilleure valeur pour top et left des titres et textes.

En conclusion, la contrainte d'absence de javascript m'a permis d'améliorer mes compétences en HTML et CSS et de leurs utilisations naturelles. De plus, des études montrent que l'option de désactivation du javascript se généralise dans les navigateurs web et que dans les années à venir les utilisateurs vont utiliser cette option. La maîtrise des effets d'animations par CSS permet de garder le dynamisme et l'interactivité des pages web même avec le javascript désactivé.

Une structure de page en HTML5 et CSS3 "moderne"

J'ai découpé la structure des pages web avec les balises de section de document <header>, <nav>, <section>, <article> et <footer>. Cela me permet d'encapsuler les répétitions de structure comme celle des vignettes. Enfin, étant donné le positionnement du menu et le choix de l'affichage en vignette, j'ai choisi d'utiliser la propriété CSS grid et les propriétés grid-area.

La propriété CSS grid permet un positionnement facile des items et de gérer leurs alignements. Les possibilités de reconfiguration et redimensionnement pour le responsive desktop sont importantes et faciles. Il est possible d'incorporer une grid dans une grid.



CSS grid appliqué au menu et vignettes

Après plusieurs essais, je fixe la taille et le nombre de colonne avec la propriété `grid repeat(nombre de colonne, 1fr)`. Cela permet d'éviter les problèmes de compatibilité de navigateur et l'utilisation des media queries obligatoires avec des propriétés comme `auto-fill`, `auto-fit`, `minmax(200px, 1fr)`.

Après validation, j'ai une page web pour chaque niveau de la partie vitrine. J'ai également créé les fichiers pour les autres pages. Ces fichiers HTML sont vides dans l'attente de la validation de l'arborescence des répertoires et urls prévus sur le serveur. Une fois cette arborescence validée, je duplique la structure des pages pour les toutes les pages vitrines.

Le contenu et les éléments graphiques n'ont pas été créés. Je suis donc obligé d'en créer certains et de générer une page en construction.

Il est décidé avec mon tuteur que la mise à jour de contenu soit des missions pour les prochains adhérents au fablab.

En conclusion, le module CSS grid layout, avec ses propriétés et fonction, est très adapté au rendu souhaité, cela est encore plus vrai pour les pages web concernant les projets. Ce module est très bien documenté (annexe source mozilla https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Grid_Layout).

La création des pages web des projets

La création des pages web projets reprend les mêmes cahiers des charges que les pages vitrines, auxquels sont ajoutées les spécifications suivantes :

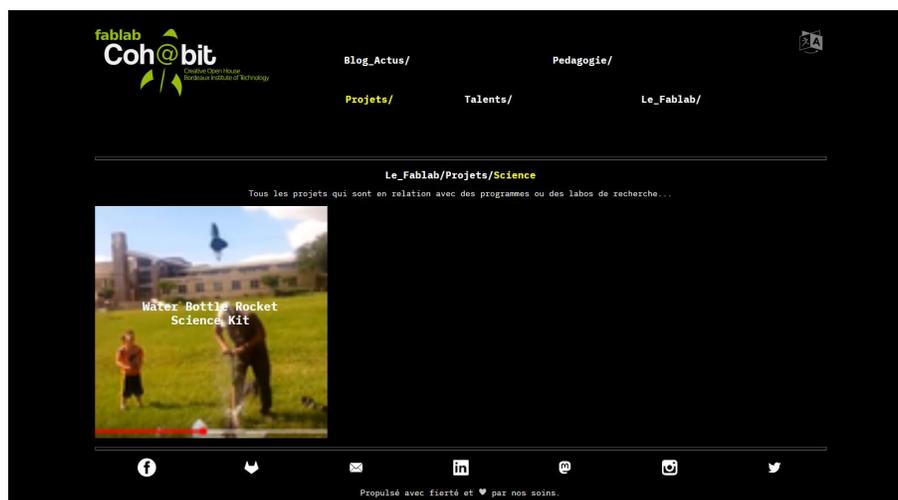
- Une apparence proche de hackaday.io.
- Un script shell bash assure la récupération et l'injection du contenu des projets créés dans le Redmine.

La création de la partie front-end des projets reprend les méthodes des pages web vitrines. Je vais développer que certaines spécificités et méthodes utilisées pour ces pages.

Je suis face à une nouvelle difficulté pour cette partie. En effet, je ne maîtrise pas encore le script shell bash, ses possibilités, son fonctionnement et le lien avec le Redmine et la récupération du contenu. De plus, il n'existe pas encore de projet documenté sur le Redmine, à partir duquel le contenu des pages web projets sera généré. Pour avoir une meilleure vision, je vais interroger à plusieurs reprises mon tuteur, et les utilisateurs du Redmine. J'utiliserai moi-même le Redmine.

Après ces entretiens, je comprends que le script injectera le contenu des projets directement en HTML, en reprenant les balises et structures que j'ai créées.

La sélection d'un projet est réalisée en cliquant sur une vignette avec le même fonctionnement que les autres vignettes. La photo, le titre, le descriptif doivent venir des projets. Ici il est réalisé avec des fausses données.



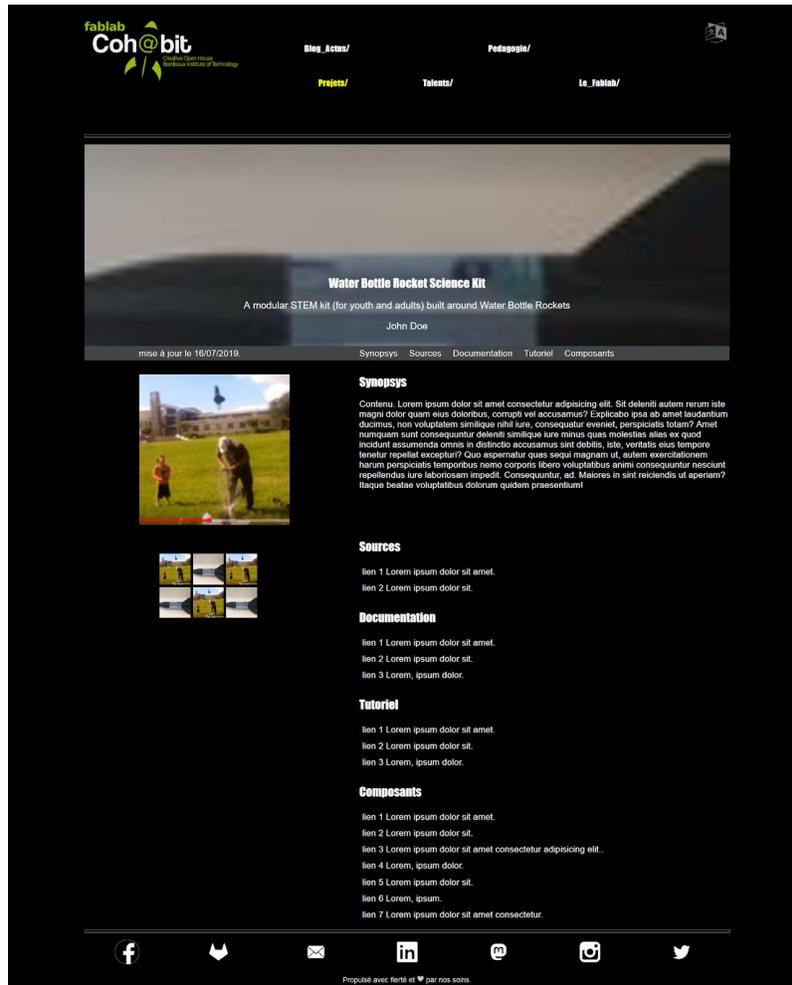
Présentation de tous les projets par catégories

La présentation du projet reprend des éléments dynamiques ou interactifs :

- Un diaporama
- des images qui peuvent s'agrandir
- Un menu sticky, dont les éléments sont ancrés pour arriver directement à la section concernée

La présentation du projet débute par une image illustrative, un titre, un résumé et un auteur.

Je vais rapidement revenir sur cette partie et les 3 éléments précédents pour expliquer les difficultés et les choix réalisés.



Concernant la partie illustrant le projet, il est impossible d'utiliser la propriété CSS background-image. En effet, premièrement, le script reconstruit uniquement des pages HTML en reprenant les balises et structures de mes modèles dans lesquels sont insérés les contenus du projet; deuxièmement, Il est possible de générer également une page CSS, mais cette stratégie est contre productive puisqu'il faudrait un fichier CSS pour chaque projet. Aussi, les images sont

insérées dans le HTML et modifiées (redimensionnées, ajout d'effet) dans le CSS. Dans l'illustration du projet, j'applique les propriétés `object-fit`, `object-position` et `opacity` à l'image.

```
.projet_presentation-image {  
  width: 100%;  
  height: 100%;  
  
  object-fit: cover;  
  object-position: 50% 45%;  
  opacity: 0.7;  
}
```

CSS image illustrant le projet

```
<article class="projet_presentation">  
    
  
  <div class="projet_presentation-descrip">  
    <h1 class="projet_presentation-titre">  
      Water Bottle Rocket Science Kit  
    </h1>  
    <p class="projet_presentation-text">  
      A modular STEM kit (for youth and adults) built around Water Bottle  
      Rockets  
    </p>  
  
    <div class="projet_presentation-talent">  
      <a href="#">  
        John Doe  
      </a>  
    </div>  
  </div>  
</article>
```

HTML Début de la page projet

Concernant le diaporama, je réalise un diaporama qui reprend les images du HTML et applique les propriétés CSS d'animation avec `@keyframes`.

```
/*
#slider {
  width: 100%;
  max-width: 300px;
  overflow: hidden;
}
#slider figure {
  position: relative;
  width: 700%;
  margin: 0;
  padding: 0;
  font-size: 0;
  text-align: left;
  animation: 15s slidy infinite;
}
#slider figure img {
  width: 14.28%;
  /* /* diviser 100% par le nombre de photo*/

  /* height: auto;
  float: left;
}
@keyframes slidy {
  0% {
    left: 0%;
  } ...
  50% {
    left: -200%;
  } ...
  70% {
    left: -200%;
  } ...
  100% {
    left: -400%;
  }
}
```

CSS d'animation avec `@keyframes` pour le diaporama.

Cependant, je dois abandonner cette méthode. En effet pour que le diaporama fonctionne, il faut connaître le nombre photo à l'avance et l'utiliser pour calculer les valeurs des propriétés. Le nombre de photos sera différent dans chaque projet et le script ne modifie pas le fichier CSS.

Aussi, le diaporama sera réalisé avec du javascript. Pour tout le site web, le javascript n'est utilisé exceptionnellement qu'ici.

Pour le menu sticky avec un scroll sur la section concernée, j'utilise le comportement natif des ancres HTML (balise <a> avec un href vers une balise <id> de la section) et j'utilise la propriété CSS position avec la valeur sticky. Pour cette dernière, beaucoup de recherche inclus du javascript qui n'est pas nécessaire à partir du moment où l'on respecte bien l'imbrication des balises HTML. l'élément en position sticky se réfère à son élément HTML le contenant, reste ensuite à déterminer la valeur top (limite à partir duquel l'élément sort du flux de DOM).

Pour réaliser ce menu je me suis appuyé sur la documentation de mdn mozilla (voir annexe mdn mozilla : https://developer.mozilla.org/en-US/docs/Learn/CSS/CSS_layout/Positioning).

Enfin il reste les images qui peuvent s'agrandir au clique et le positionnement de l'ensemble des éléments. Pour les images, leur taille est diminuée en CSS. Leur agrandissement est réalisé en encapsulant les images dans des liens (qui pointe vers l'image à taille réelle) avec l'attribut target=_blank. Le positionnement de l'ensemble des éléments est réalisé en utilisant le module CSS grid (voir page suivante).

En conclusion, j'ai approfondi mes connaissances et ma rigueur en CSS3 et HTML5, permettant d'avoir une page se comportant comme celle de hackaday.io mais sans JQuery, event Js, modale bootstrap ...

J'arrive ici à un point d'étape important de ma mission. En effet, j'ai toutes mes pages modèles vitrines (4 niveaux) et mes pages modèles projets soit le front-end du site web. J'ai également l'architecture du site web, liens des différentes pages et formes des urls. Je fais donc un point d'étape avec mon tuteur, l'objectif est de valider tous les modèles et l'architecture, de déterminer l'organisation des fichiers tels qu'ils seront sur le serveur. Ainsi je peux dupliquer mes modèles aux pages web et je vais commencer le back-end du site web.

The screenshot shows a website layout for a project titled "Water Bottle Rocket Science Kit". The layout is structured using CSS Grid, with a dark background and a light grid overlay. The page includes a header with navigation links, a main content area with a video player, a sidebar with a list of links, and a footer with social media icons and a tagline.

Header: Logo for "fablab Coh@bit Creative Open House! Building a culture of Technology". Navigation links: Blog_Actus/, Pedagogie/, Projets/, Talents/, Le_Fablab/.

Main Content: Video player showing a water bottle rocket launch. Title: "Water Bottle Rocket Science Kit". Subtitle: "A modular STEM kit (for youth and adults) built around Water Bottle Rockets". Author: "John Doe".

Metadata: "mise à jour le 16/07/2019." and a horizontal menu: "Synopsis Sources Documentation Tutoriel Composants".

Synopsis: A large image of a water bottle rocket. Text: "Contenu. Lorem ipsum dolor sit amet consectetur adipisicing elit. Sit deleniti autem rerum iste magni dolor quam eius doloribus, corrupti vel accusamus? Explicabo ipsa ab amet laudantium ducimus, non voluptatem similique nihil iure, consequatur eveniet, perspiciatis totam? Amet numquam sunt consequuntur deleniti similique iure minus quas molestias alias ex quod incidunt assumenda omnis in distinctio accusamus sint debitis, iste, veritatis eius tempore tenetur repellat excepturi? Quo aspernatur quas sequi magnam ut, autem exercitationem harum perspiciatis temporibus nemo corporis libero voluptatibus animi consequuntur nesciunt repellendus iure laboriosam impedit. Consequuntur, ad. Maiores in sint reiciendis ut aperiam? Itaque beatae voluptatibus dolorum quidem praesentium!"

Sources: A vertical list of three small images. Text: "lien 1 Lorem ipsum dolor sit amet. lien 2 Lorem ipsum dolor sit."

Documentation: A vertical list of three small images. Text: "lien 1 Lorem ipsum dolor sit amet. lien 2 Lorem ipsum dolor sit. lien 3 Lorem, ipsum dolor."

Tutoriel: A vertical list of three small images. Text: "lien 1 Lorem ipsum dolor sit amet. lien 2 Lorem ipsum dolor sit. lien 3 Lorem, ipsum dolor."

Composants: A vertical list of seven small images. Text: "lien 1 Lorem ipsum dolor sit amet. lien 2 Lorem ipsum dolor sit. lien 3 Lorem ipsum dolor sit amet consectetur adipisicing elit.. lien 4 Lorem, ipsum dolor. lien 5 Lorem ipsum dolor sit. lien 6 Lorem, ipsum. lien 7 Lorem ipsum dolor sit amet consectetur."

Footer: Social media icons for Facebook, GitHub, Email, LinkedIn, Instagram, and Twitter. Tagline: "Propulsé avec fierté et ♥ par nos soins."

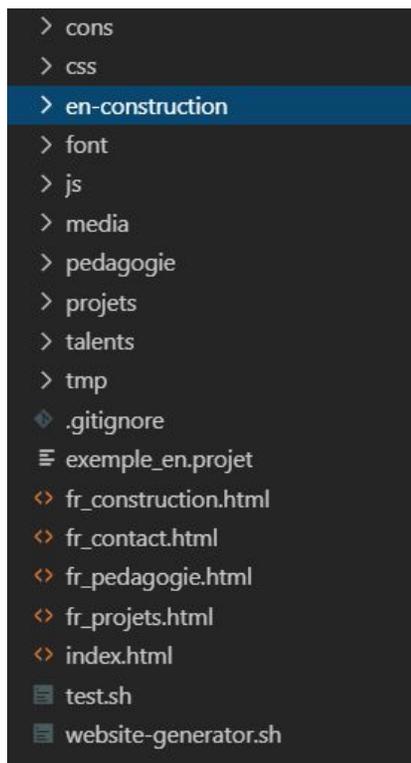
Utilisation de CSS GRID pour le positionnement dans la page projet

L'organisation et le fonctionnement de la mise à jour des pages projets.

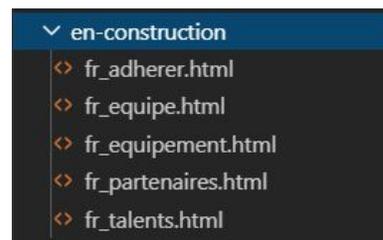
L'organisation des fichiers et méthode de travail

Mon répertoire de travail, miroir de celui du serveur, s'est bien étoffé.

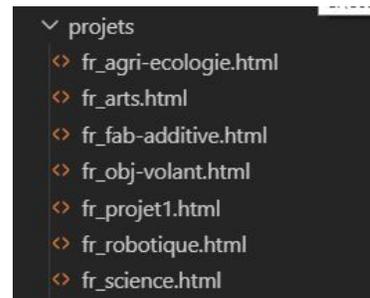
Les pages vitrines de 1er niveau sont à la racine. Les pages de niveau inférieures sont dans les répertoires reprenant le nom des pages "mères". l'ensemble des médias est regroupé dans un répertoire. A la racine de ce dernier est stocké tous les médias des pages vitrines. Enfin, comme le contenu et les éléments graphiques ne m'ont pas été transmis, j'ai créé une page "en construction" et des répertoires regroupant les pages non finies. Pour ces pages j'ai défini une redirection temporaire 302.



répertoire de travail (idem serveur)

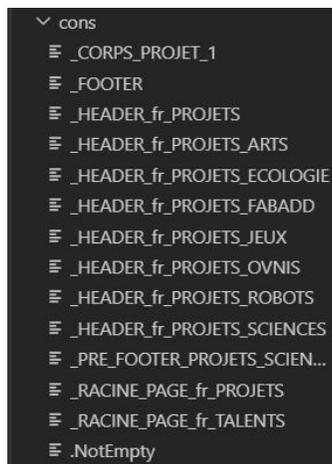


page en construction

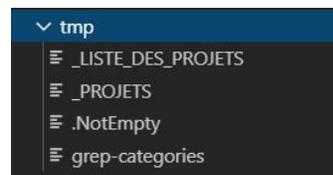


page filles dans les répertoires pages mères

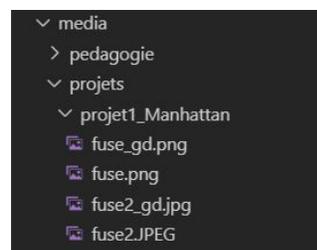
Les pages projets sont générées par un script. Ce script est le fichier shell “website-generator.sh” placé à la racine du répertoire. Ce script crée des fichiers intermédiaires et temporaires pour générer les pages projets. Ces fichiers sont stockés dans le répertoire TMP. Ce script utilise des extraits des pages modèle HTML que j’ai créé. Ces extraits sont enregistrés dans le répertoire CONS. Enfin un sous-répertoire PROJETS dans le répertoire MEDIA permet de regrouper les médias (image, son, vidéo) d’un projet. Ce regroupement de fichiers médias se fait dans un sous-répertoire reprenant le nom du projet. Cette arborescence et les conventions de nommage des fichiers et répertoires ont été définies avec mon tuteur. En effet, cela est très important puisque l’algorithme du script “website-generator.sh” va utiliser ces noms et y faire référence.



fichiers de construction des pages html



fichiers temporaires pour le script



fichiers média pour les projets

```

_HEADER_fr_PROJETS_ARTS X
cons > _HEADER_fr_PROJETS_ARTS
1 <section>
2
3 <article class="presentation prez-text">
4 <h2>
5 <a href="https://projets.cohabit.fr/fablab/index.html">Le Fablab</a><a
6 href="https://projets.cohabit.fr/fablab/fr_projets.html"
7 >Projets</a
8 ></a href="https://projets.cohabit.fr/fablab/fr_science.html" class="activeLink yellow">Science</a>
9 </h2>
10 <p>
11 Tous les projets qui sont en relation avec des programmes ou des
12 labos de recherche...
13 </p>
14 </article>
15
16 <div class="vig-grid">
17

```

exemple de fichiers de construction

Concernant les méthodes de travail, mon tuteur me demande de construire et de faire les tests du script directement sur le serveur de production. Je propose comme alternative d'utiliser notre dépôt git et d'avoir une branche spécifique pour les tests sur le serveur de production.

Nous utilisons un dépôt git pour le projet depuis le début. Pendant la phase de création du front-end, suite à une mauvaise méthodologie et une erreur de mon tuteur, j'ai passé deux jours à remettre en état les fichiers en incluant les modifications préconisées par mon tuteur. Suite à cela, j'ai revu la méthodologie de travail avec le git, multiplié les branches et réalisé une copie miroir du dépôt.

The screenshot shows the GitHub interface for the repository 'Website Cohabit' (Project ID: 46826). It displays the repository name, a 'Clone' button, and statistics: 223 Commits, 4 Branches, 0 Tags, and 22.3 MB Files. The current branch is 'update'. A recent commit by 'pgp' is shown, titled 'Merge branch 'update' of framagit.org:cohabit-fablab/website-cohabit into update', with a commit hash of 87588d44. Below this, a table lists files and their last commit details:

Name	Last commit	Last update
cons	correction pb icone footer	1 month ago
css	icone rezo flexible et responsive	2 weeks ago

Dépôt git du projet (html 65.8% css 17.3% shell 16.6% js 0.4%)

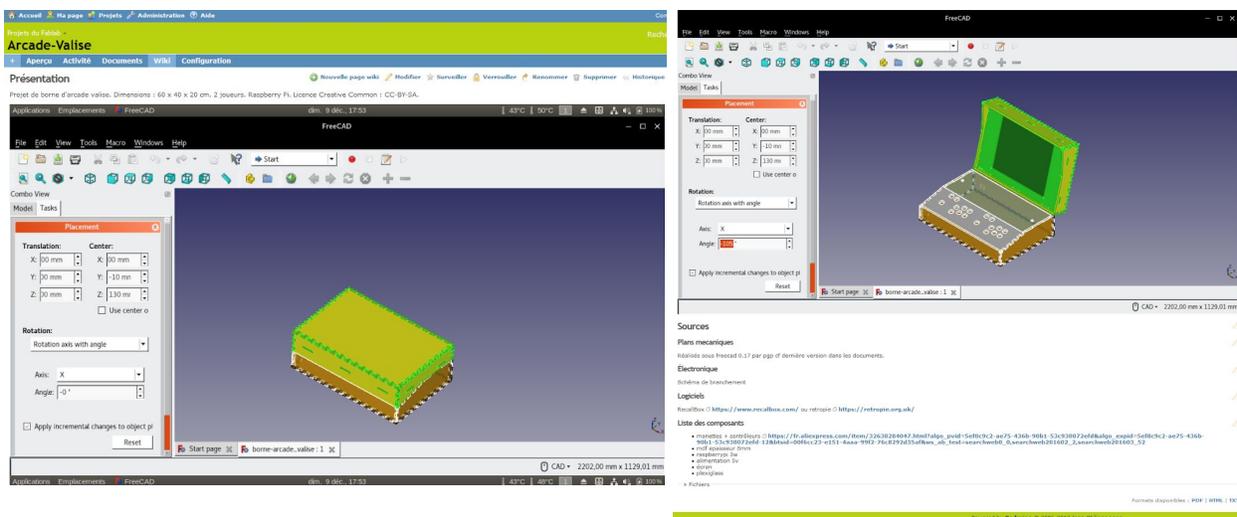
Mise à jour des pages projets : l'accès aux données back-end

Pour la mise à jour des pages projets le script va chercher le contenu des éléments dans le Redmine. Je vais présenter cet outil, les essais et choix retenus pour récupérer les données.

Redmine est un logiciel web de gestion de projet open source (<https://www.redmine.org/>). Il est très complet avec gestions des droits, calendrier gants, gestion de tickets, wiki...

Ce dernier est installé sur le serveur du fablab et il est associé à une base de données PostgreSQL. Il affiche l'ensemble des projets documentés et les ressources documentées du fablab (voir page 34).

Pour chaque projet une partie regroupant les documents et un Wiki est disponible. Ainsi un projet documenté est un projet avec ses sources, les fichiers informatiques pour les machines numériques, les explications et tous les médias permettant de le reproduire.



Exemple de Wiki d'un projet



Détail d'un projet onglet documents



Détail des documents "page-web"

Nous avons donc ici, l'ensemble des données à récupérer. Mes premiers essais sont réalisés par requête PostgreSQL. Pour cela je n'ai pas accès à une interface graphique de gestion de base de données. Je me forme donc à l'utilisation de PostgreSQL en ligne de commande, notamment en utilisant la documentation officielle

<https://www.postgresql.org/docs/9.1/app-psql.html>. Bien que j'arrive à afficher la structure et les données de la base et des tables, déterminer la requête permettant de trouver un projet précis, et par la suite, toutes ses données liées est compliqué.

Pour faciliter, les essais et les tests le projet arcade-valise est créé, ainsi que le fichier arcade-valise.projet.

Je réalise une recherche sur stackoverflow pour trouver une requête me permettant de chercher dans l'ensemble de la base de données le nom de ce fichier. Je tape : How to search a specific value in all tables (PostgreSQL)? ainsi je trouve une fonction psql qui me permet de trouver le champ et la table pour une valeur bien précise, à savoir le nom de mon projet. La fonction psql et son utilisation se trouve en annexe

<https://stackoverflow.com/questions/5350088/how-to-search-a-specific-value-in-all-tables-postgresql>.

Accueil
Ma page
Projets
Administration
Aide

Projets
Activités
Demandes
Temps passé
Gantt
Calendrier
Annonces

Recherche:
Allez à un projet...

Voir les projets fermés
 Nouveau projet

Fablab Coh@bit

Accueil
Si vous venez pour la première fois, ou pour vous repérer, cliquez sur accueil et sur wiki.

Administration du fablab

Autres Stagiaires
Tous stagiaires qui ne retire pas dans le cadre des stages YES.
Communication - graphisme
Les outils de communication du fablab et les éléments de charte graphique.
Evénements

Local Fablab
toutes les informations sur le réseau, les locaux du fablab

Design fablab

PAKI

PERSEVERONS
services civiques

Ateliers
tous les ateliers du fablab

Adobe Photoshop et Illustrator
Atelier sur les logiciels de la suite Adobe: Photoshop CS2 et Illustrator CS2

Atelier Documentation

Atelier Freedrad

Atelier WEB & bash

Atelier SpeakerPhat

Atelier Bluetooth
Atelier jeune, comment fabriquer une petite enceinte bluetooth de 2x5W avec différents composants.

Install Party Linux

Lampe à gravité

Lecteur Ampli mp3

pirateBoxzero

Thermomètre

Documentation logiciels

freedrad

Git

Inkscape

KICAD

Documentation Machines

brodeuse

Charly_Robot

Découpeuse laser

Hack RF

Imprimante 3D

Imprimantes 3D (maintenance)

Imprimante SLA

Imprimante AnyCubic Photon

Laser-V-2000

plotter

Hall Noir

Pédagogie

Projets des adhérents

Horloge Lixie
horloge à base de leds ws2812. Plaques de plexi et esp3286

IMB -cours de billes
quel est le chemin le plus rapide pour qu'une bille aille d'un point A à un point B ?

Préce pour le projet Drones&cat
pièce imprimée en 3D pour un drone de labo d'aérophysique de Bordeaux

portofolio

Projets du Fablab

Arcade-Valise

Astronomie
Banc en palètes pour la jardin permaculture

Construction d'un banc en palettes pour le jardin en permaculture

Banc Test

Biopointe
boite de rangement pour les composants électroniques

Borne d'arcade

Bras Robot

Capteur COV

Capteurs&spores&TV

Raspberry Pi

Cité SGN
Reproduction d'une cité en plastique pour une amoree en SGN

concentrateur solaire

Entree conductrice
le but est d'essayer de produire de l'entree conductrice à moindre coût et fabriquer maison.

Galatee

Gravure de PCB

hélioat

Jardin connecté

communication rechaire

Jardin Permaculture

Low Bot

Main Myoélectrique
Prothèse de main Myoélectrique, sous la tutelle de Micrombarrier

Mali
faire essayer / éfablab coh@bit au Mali:
serveur moodle

Onirisation

openbeelab

Pharmatronic Box

pluive-Radio-Streamer

PrirateBox

Projets découpe laser

petits projets sur la découpeuse laser

Pied à coulisse

Puissance 4
Un puissance 4 en mdf 3mm

Projets impression 3D

Fusee-Ileda

Révoit Arduino Eloise

[Robot Agricole] destacheur

Robot Innove

Robots Insectes

Serveuse automate

tentacle
coulé en epoxy et Nico
licences libres hardware libre
thermo-bibill

The fighter
Projet de robotique
février 2015

Système d'information

YES

Mes projets

Formats disponibles: Atom

ensemble des projets et ressources documentées du redmine

Après concertation avec mon tuteur, nous choisissons de ne pas intégrer de requête psql dans le script shell. Nous décidons de prévoir une feature pour cadrer et normer les projets documentés dans le Redmine et ainsi à termes pouvoir utiliser les requêtes psql et la base de données du Redmine. En attendant je crée un document type "exemple.projet", qui reprendra toutes les données du projet dans des variables et qui sera stocké sur le Redmine (onglet document, page-web). Ainsi avec l'utilisation des filtres et des fonctions avancées de recherche du shell bash, nous retrouverons notre liste de projets, leurs emplacements et leurs données.

```
exemple.projet.txt - Bloc-notes
Fichier Edition Format Affichage Aide
# fichier projet : ce fichier vous permet de publier votre projet sur le site public de cohabit
# pour cela il vous suffit :
# 1 - de copier ce fichier type dans votre projet redmine
# 2 - de remplacer les expressions notées "valeur" par vos valeurs sans les guillemets.
# W - pas d'espace dans le NOM DU PROJET
# 3 - si il y a plusieurs images refaites une ligne par image, idem pour les autres medias.
# 4 - pour la langue c'est FR ou EN, pour faire les deux il faut faire deux fichiers .projet
# 5 - résumée: c'est une phrase ou 2 qui présente le projet_tutoriel
# 6 - Catégories : effacer les mentions inutiles, une seule catégorie par ligne

_AUTEUR|"valeur"|
_NOM_DU_PROJET|"valeur"
_LANGUE|FR
_RESUMEE|"valeur"
_PRESENTATION|"valeur"
_SOURCES|"valeur"
_SOURCES|"valeur"
_DOCUMENTATION|"valeur"
_TUTORIEL|"valeur"
_COMPOSANTS|"valeur"
_CATEGORIE|sciences
_CATEGORIE|jeux
_CATEGORIE|arts
_CATEGORIE|ovnis
_CATEGORIE|robots
_CATEGORIE|fabrication_additive
_CATEGORIE|ecologie
_TAGS|"valeur"
_SON|"valeur"
_VIGNETTE|"valeur"
_VIDEO|"valeur"
_IMAGE|"valeur"
```

fichier fr_exemple.projet

```

arcade-valise.projet - Bloc-notes
Fichier Edition Format Affichage Aide
# fichier projet : ce fichier vous permet de publier votre projet sur le site public de cohabit
# pour cela il vous suffit :
# 1 - de copier ce fichier type dans votre projet redmine
# 2 - de remplacer les expressions notées "valeur" par vos valeurs sans les guillemets.
# W - pas d'espace dans le NOM DU PROJET
# 3 - si il y a plusieurs images refaites une ligne par image, idem pour les autres medias.
# 4 - pour la langue c'est FR ou EN, pour faire les deux il faut faire deux fichiers .projet
# 5 - résumé: c'est une phrase ou 2 qui présente le projet_tutoriel
# 6 - Catégories : effacer les mentions inutiles, une seule catégorie par ligne

_AUTEUR|pgp
_NOM_DU_PROJET|Arcade-Valise
_LANGUE|FR
_RESUMEE|Une borne d'arcade transportable à base de raspberryPi, découpée à la laser et qui se replie pour faire une valise.
_PRESENTATION|Une borne d'arcade transportable à base de raspberryPi, découpée à la laser et qui se replie pour faire une valise. Lorem ipsum dolor...
_SOURCES|<a href=https://projets.cohabit.fr/redmine/documents/179>plans mécaniques freecad</a>
_SOURCES|<a href=https://projets.cohabit.fr/redmine/documents/98>fichiers de découpe</a>
_DOCUMENTATION|<a href=https://projets.cohabit.fr/redmine/projects/arcade-valise/wiki>WIKI</a>
_TUTORIEL|"valeur"
_COMPOSANTS|<a href=https://projets.cohabit.fr/redmine/documents/180>Liste des composants</a>
_CATEGORIE|sciences
_CATEGORIE|jeux
_CATEGORIE|arts
_CATEGORIE|ovnis
_CATEGORIE|robots
_CATEGORIE|fabrication_additive
_CATEGORIE|ecologie
_TAGS|
_SON|"valeur"
_VIGNETTE|arcade-valise.jpg
_VIDEO|"valeur"
_IMAGE|"valeur"

```

exemple appliqué au projet arcade-valise

En conclusion, avec ce contournement, nous avons notre composant back-end qui accède aux données. Reste maintenant à utiliser ces données et les injecter dans les pages web projets coté front-end.

Mise à jour des pages projets : utilisation des données back-end

L'utilisation du langage de programmation shell bash a été choisie, car il correspond à une compétence transverse aux projets réalisés au fablab (console linux, raspberry pi...). Afin de réaliser ce script, j'augmente mes compétences, en suivant le mooc "Maîtriser le shell bash", pour pouvoir utiliser les filtres, les fonctions et la syntaxe de programmation de ce langage. Ces nouvelles compétences m'aideront également pour la création des requêtes sql et psql en ligne de commande.

Avant de voir le script shell bash en lui-même, voici son process:

1. Récupération de tous les fichiers avec une extension `.projet` enregistrés dans le répertoire du Redmine.

1.1. Écriture de la liste des adresses de ces fichiers dans `/tmp/_LISTE_DES_PROJETS`

2. Pour chaque adresse (boucle `while`):

2.1. Lecture du fichier et enregistrement des données dans des variables (`_NOM_DU_PROJET`, `_AUTEUR`, `_VIGNETTE` ...).

2.2. Création du sous-répertoire pour les médias du projet et copie des médias.

2.3. Formatage des dimensions des médias.

2.4. Création de la balise HTML pour chaque projet en fonction de la catégorie `/tmp/_CORPS_PROJETS_"CATEGORIES"`.

3. Création de la page complète par concaténation des extraits HTML de `/CONS` et des balises créés en 2.4.

4. Sauvegarde par `rsync` des modifications sur un dépôt git spécifique pour la gestion des logs.

Le lancement du script est planifié en tant que tâche cron sur le serveur.

Pour expliquer le script, tout en gardant de la clarté j'explique la création de la vignette projet pour la catégorie science.

```

website-generator.sh
1  #!/bin/bash
2  #générateur statique de site web du fablab cohabit
3  set -o verbose
4  set -o nounset
5
6  LISTE_DES_PROJETS="/tmp/_LISTE_DES_PROJETS"
7
8  PAGE_PROJETS_SCIENCES="/projets/fr_science.html"
9
10 CORPS_PROJETS_SCIENCES="/tmp/_CORPS_PROJETS_SCIENCES"
11
12 CORPSPROJET="/tmp/_CORPS_PROJET"
13
14 HEADER_fr_PROJETS="/cons/_HEADER_fr_PROJETS"
15
16 HEADER_fr_PROJETS_SCIENCES="/cons/_HEADER_fr_PROJETS_SCIENCES"
17
18 PRE_FOOTER_PROJETS_SCIENCES="/cons/_PRE_FOOTER_PROJETS_SCIENCES"
19

```

Un script bash commence toujours par le shebang `#!`. Cela spécifie que ce fichier est un script et non un binaire, il est suivi de l'interpréteur utilisé, ici `bash`. la fonction `set` sert à l'affichage d'informations si un bogue se produit pendant l'exécution du script.

Nous avons ensuite la déclaration des variables et leurs affectations. Ici il s'agit de déclarer et d'affecter tous les fichiers en respectant les conventions de nommage et l'arborescence des répertoires déterminées (voir "L'organisation des fichiers et méthode de travail" p 30).

```
22 function checkValues() {
23     | grep -w "$1" "$2" | cut -d '|' -f 2
24 }
25 function checkMedia() {
26     | grep -w "$1" "$2" | cut -d '|' -f 1
27 }
28
29 # on enregistre l'IFS actuel
30 R=$IFS
31
32 # on change l'IFS, pour être un retour à la ligne
33 IFS='
34 '
35 # RAZ des listes tempo
36 rm $LISTE_DES_PROJETS
37 rm $CORPS_PROJETS_SCIENCES $CORPS_PROJETS_OVNIS $CORPS_PROJETS_ARTS $CORPS_PROJETS_FABADD $CORPS_PROJETS_ECOLOGIE $CORPS_PROJETS_ROBOTS
38 > $CORPS_PROJETS_SCIENCES $CORPS_PROJETS_OVNIS $CORPS_PROJETS_ARTS $CORPS_PROJETS_FABADD $CORPS_PROJETS_ECOLOGIE $CORPS_PROJETS_ROBOTS
39 $CORPS_PROJETS_JEUX
```

Suive les déclarations de fonction. Ces fonctions utilisent le filtre grep avec l'option -w. Cela permet de sélectionner dans un fichier les lignes avec les mots correspondants aux paramètres \$1 et \$2 lors de l'appel à ces fonctions. Une fois cette ligne sélectionnée, on l'envoie pour traitement (pipe |) par la commande cut. On coupe la ligne (cut) au niveau du délimiteur '|' (-d '|') et on garde la partie dans le champ qui suit immédiatement (-f 2).

L'IFS est le séparateur interne. On le définit pour être sûr d'avoir un retour à la ligne lors de l'interprétation des espaces, par exemple entre les variables ou les noms de fichiers.

Le lancement du script, commence par une remise à zéro des fichiers temporaires. On efface tous les fichiers en utilisant les variables déclarées au début. Pour cela on utilise la commande rm, remove avec un seul nom de variable ou plusieurs noms de variable mais avec une redirection simple ">" vers les mêmes noms de variables entraînant leur écrasement (variables venant d'être déclarées et vides).

```

40  ### générateur des pages projet fr et en ###
41
42  # générer la liste des Projets
43  find /opt/redmine/redmine-3.4.4/files -type f -name "*.projet" | sort -V > $LISTE_DES_PROJETS

```

Étape 1 du process. Grâce au travail avec les requêtes psql, je connais le chemin du répertoire où sont stockés les documents fr_exemples.projet. Il s'agit de /opt/redmine/redmine-3.4.4/files. On utilise la commande find avec les options -type f et -name ainsi qu'une expression régulière dans ce répertoire. Ainsi on trouve (find) tous les fichiers de type régulier (-type -f) dont le nom (-name) se termine par .projet ("*.projet"). Le pipe "|" permet d'envoyer tous les résultats trouvés à la commande de tri "sort" avec l'option -V, qui correspond à un tri naturel avec prise en compte des nombres dans le texte. Ces lignes de résultats triés sont envoyées (redirection simple >) dans le fichier _LISTE_DE_PROJETS par la variable ad hoc.

```

45  # lecture des fichiers de la listes des projets et extraction des données (boucles while)
46  cat $LISTE_DES_PROJETS | while read ligne
47  do
48  # Lecture du fichier .projet et enregistrement des variables
49  NOM_DU_PROJET=`checkValues "_NOM_DU_PROJET" $ligne`
50  RESUMEE=`checkValues "_RESUMEE" $ligne`
51  VIGNETTE=`checkValues "_VIGNETTE" $ligne`
52  CATEGORIE=`checkValues "_CATEGORIE" $ligne`
53

```

Étape 2 et 2.1 du process. Ouverture de la boucle while. On affiche la liste des projets (chemin des fichiers) que l'on passe (pipe "|") en lecture (fonction read()) avec en paramètre le numéro de ligne) pour chaque ligne de la liste et tant qu'il y a des lignes (while) . Ainsi pour chaque ligne (\$ligne) on ouvre le fichier se terminant par ".projet" et on fait (do) une extraction des données et une affectation des variables en utilisant les fonctions définies en début du script checkValues().

```

54  #gestion des médias du projet création du répertoire et du lien symbolique
55  mkdir ./media/projets/"$NOM_DU_PROJET"
56  LIEN_VIGNETTE=`find /opt/redmine/redmine-3.4.4/files -type f -name "$VIGNETTE"``
57  #création du lien symbolique vignette
58  echo "$LIEN_VIGNETTE" >> ./tmp/_test_lien_vignette
59  # copie lien symbolique vignette
60  cp "$LIEN_VIGNETTE" ./media/projets/"$NOM_DU_PROJET"/"$VIGNETTE"
61
62  # conversion de l'image en 300x300 px sans déformation
63  convert -resize 300 ./media/projets/"$NOM_DU_PROJET"/"$VIGNETTE" ./media/projets/"$NOM_DU_PROJET"/resize-"$VIGNETTE"
64  convert -background white -gravity center -crop 300x300+0+0 ./media/projets/"$NOM_DU_PROJET"/resize-"$VIGNETTE" ./media/projets/
"$NOM_DU_PROJET"/vignette-"$VIGNETTE"

```

Étape 2.2 et 2.3 du process. Gestion des médias du projet. Nous sommes toujours dans la boucle while. Avec la commande `mkdir` on crée le sous-répertoire “nom du projet” dans le répertoire `./media/projets/`. On utilise ici la valeur de la variable `$NOM_DU_PROJET` affectée plus haut. Grâce aux requêtes `psql`, je sais que les médias sont enregistrés dans le même répertoire que les fichiers “.projet”. De plus, je sais que le Redmine les stocke en les préfixant d’une série de chiffre et un underscore . C’est pourquoi, j’utilise la même méthode que pour l’étape 1 en changeant juste l’expression régulière. À savoir, “*”`$VIGNETTE` avec “*” pour aucun ou plusieurs caractères alphanumériques (préfixe) et la variable `$VIGNETTE` entre “” pour qu’elle soit substituée par son nom de fichier. La commande `cp` permet de copier l’image et de la renommée sans son préfixe.

Ayant réalisé des tests avec d’autres projets, je suis de nouveau confronté au problème de non normalisation des projets dans le Redmine. Cette fois-ci sur les dimensions des médias, donc je passe par une étape de conversion des images en utilisant `ImageMagick` en ligne de commande (doc officielle <https://imagemagick.org/>).

```

66 # remise à zero de la liste des catégories
67 rm -rf ./tmp/grep-categories
68 > ./tmp/grep-categories
69
70 #générer la listes des catégories d'un projet et affectation de variable en fonction des catégorie
71 echo -n "$CATEGORIE" >> /var/www/html/fablab/tmp/grep-categories
72 GREPSCIENCES=`grep -w "sciences" ./tmp/grep-categories`
73

```

```

74
75 # crée une balise html, texte avec lien pour chaque page projet étiquetée en fonction de la catégorie ici simplification science
76 # et les enregistre dans le fichier /tpm/_CORPS_PROJETS_"catégorie" ici simplification SCIENCES
77
78 if [ "$GREPSCIENCES" = "sciences" ]
79 then echo "<article id=\"v1\" class=\"vignette\"><a href=\"\"$NOM_DU_PROJET\".html\"><img width=\"200\" src=\"\"https://
projets.cohabit.fr/fablab/media/projets/\"$NOM_DU_PROJET/\"$VIGNETTE\"\" alt=\"\"$NOM_DU_PROJET\"\" class=\"vignette-image\"/><h2
class=\"vignette-titre vig-titre-projet\">\"$NOM_DU_PROJET\"</h2><div class=\"vignette-text\"><div class=\"text
vig-text-projet\">\"$RESUMEE\"</div></div></a></article>" >> $CORPS_PROJETS_SCIENCES
80 fi
81 done

```

Étape 2.4 du processus. Création du HTML en fonction des catégories. Nous sommes toujours dans la boucle while. Chaque projet peut avoir plusieurs catégories, les données devront donc être injectées dans une ou plusieurs balises html en fonction des CATÉGORIES. Pour gérer cela, je crée un fichier temporaire `grep-catégorie` que j’efface et recrée sans informations au début du

script. J'écris (echo) les valeurs de la variable \$CATEGORIE que j'ajoute (redirection >>) à la fin de mon fichier temporaire grep-catégorie.

Ensuite je crée des variables, ici GREPSCIENCES, qui me sert de clé pour la réalisation des balises html conditionné par un IF. Je reprends ma structure HTML correspondante à la vignette dans laquelle j'injecte les variables spécifiques à chaque projet. Ces variables sont entre " " pour permettre leur substitution par les valeurs affectées en début de la boucle. Je ferme la boucle while.

```

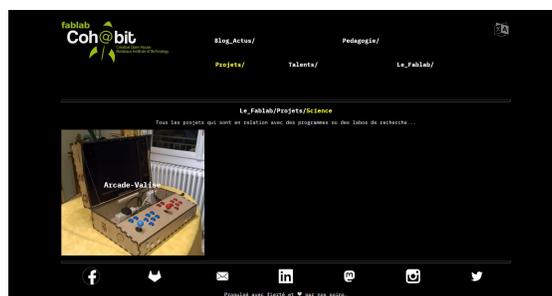
82
83 # génération de la page des projets sciences #
84 cat $HEADER_fr_PROJETS $HEADER_fr_PROJETS_SCIENCES $CORPS_PROJETS_SCIENCES $PRE_FOOTER_PROJETS_SCIENCES $PIEDS > $PAGE_PROJETS_SCIENCES
85
86 #on rétablit l'IFS
87 IFS=$R
88
89 #synchronisation avec le dépôt git pour les logs (exemple de commande)
90 #rsync -avz -e "ssh -i /home/pgp/.ssh/id_rsa -p 55555" /media/pgp/disque2/Websites/pgp-web-static-cli/* root@web.openbeelab.org:/var/www/pgp/
91
92 exit 0
93

```

Étape 3 et 4. Il ne reste plus qu'à concaténer (cat) les extraits de HTML de /CONS avec le HTML construit dans la boucle \$CORPS_PROJETS_"catégories". Cela est fait pour toutes les pages projets et catégories. Puis on écrase l'ancienne page HTML PROJETS avec la nouvelle (> \$PAGE_PROJETS_"catégories").

Pour finir on rétablit l'IFS et on lance une synchronisation pour sauvegarder les modifications (dépôt git spécifique de log).

En conclusion, le script permet d'injecter et de mettre à jour les données back-end du Redmine dans le front-end du site web. Le script est encore en cours d'écriture et de validation pour les pages projets détaillées.



Conclusion générale sur le projet

Je retire quatre enseignements importants de ce projet.

- Premièrement, j'ai grandement amélioré mes compétences front-end et responsive. J'ai une meilleure vision des structures et de l'enchaînement des actions pour réaliser un front-end. Je compte me constituer un recueil des bonnes pratiques, des étapes, snippet est process pour mes prochaines réalisations.
- Deuxièmement, j'ai grandement amélioré mes compétences dans la gestion d'un projet web. Ayant des compétences en gestion de projet et conduite du changement, je n'arrivais pas au terme de ma formation à visualiser la gestion du projet web, ces étapes, par où commencer. Aujourd'hui je suis plus serein de ce côté et je me sens mieux assuré.
- Troisièmement, les vicissitudes du projet, l'environnement moins professionnel, le choix d'utiliser des techniques entraînant beaucoup de duplication et reprise m'ont permis de palier à mon habitude de vouloir trop bien faire. De ne pas avoir peur de refaire et refaire mon code.
- Enfin, quatrièmement, la pédagogie et le choix des technologies m'ont permis de développer beaucoup d'autres compétences sysOps, devOps et graphisme. Il m'a également permis de faire le point concernant l'apport des framework, des bibliothèques externes et de pouvoir à l'avenir faire des choix avisés en fonction des projets.

Pour terminer, mon aventure sur ce projet et bien d'autres au fablab est toujours d'actualité puisqu'à la suite de mon travail, je suis maintenant un adhérent bénévole.

Annexe



Toggling elements with a hidden checkbox

This example allows for (optional) pseudo-classes to let the user toggle content based on the state of a checkbox, all without using JavaScript.

HTML

```

1 <input type="checkbox" id="toggle" />
2
3 <div>
4   <div style="display: inline-block; width: 400px; height: 40px; border: 1px solid black;">
5     </div>
6   <div>
7     <div class="expandable" style="display: inline-block; width: 100%; height: 100%; border: 1px solid black;">
8       <div style="border: 1px solid black; padding: 5px; width: 100%; height: 100%; background-color: #f0f0f0;">
9         <div style="border: 1px solid black; padding: 5px; width: 100%; height: 100%; background-color: #f0f0f0;">
10          <div style="border: 1px solid black; padding: 5px; width: 100%; height: 100%; background-color: #f0f0f0;">
11          <div class="expandable" style="display: inline-block; width: 100%; height: 100%; border: 1px solid black;">
12          <div class="expandable" style="display: inline-block; width: 100%; height: 100%; border: 1px solid black;">
13          </div>
14        </div>
15      </div>
16    </div>
17  </div>
18 </div>
19 </div>
20 </div>
21 </div>
22 </div>
23 </div>
24 </div>
25 </div>
26 </div>
27 </div>
28 </div>
29 </div>
30 </div>
31 </div>
32 </div>
33 </div>
34 </div>
35 </div>
36 </div>
37 </div>
38 </div>
39 </div>
40 </div>
41 </div>
42 </div>
43 </div>
44 </div>
45 </div>
46 </div>
47 </div>
48 </div>
49 </div>
50 </div>
51 </div>
52 </div>
53 </div>
54 </div>
55 </div>
56 </div>
57 </div>
58 </div>
59 </div>
60 </div>
61 </div>
62 </div>
63 </div>
64 </div>
65 </div>
66 </div>
67 </div>
68 </div>
69 </div>
70 </div>
71 </div>
72 </div>
73 </div>
74 </div>
75 </div>
76 </div>
77 </div>
78 </div>
79 </div>
80 </div>
81 </div>
82 </div>
83 </div>
84 </div>
85 </div>
86 </div>
87 </div>
88 </div>
89 </div>
90 </div>
91 </div>
92 </div>
93 </div>
94 </div>
95 </div>
96 </div>
97 </div>
98 </div>
99 </div>
100 </div>

```

CSS

```

1 /* Hide the toggle checkbox */
2 #toggle {
3   display: none;
4 }
5
6 /* Hide expansion content by default */
7 .expandable {
8   visibility: collapse;
9   background: #fff;
10 }
11
12 /* Style the button */
13 #toggle {
14   display: inline-block;
15   width: 20px;
16   height: 20px;
17   border: 1px solid black;
18   background-color: #fff;
19 }
20
21 /* Show hidden content when the checkbox is checked */
22 #toggle:checked ~ .expandable {
23   visibility: visible;
24 }
25
26 /* Style the button when the checkbox is checked */
27 #toggle:checked {
28   background-color: #fff;
29 }
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

```

Result



Alternance des éléments avec une case à cocher caché.

cet exemple utilise la pseudo-classe `:checked` permettant l'usage d'un contenu alternant basé sur l'état d'une case à cocher, le tout sans utiliser Javascript.

<https://developer.mozilla.org/en-US/docs/Web/CSS/:checked>



https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Grid_Layout

Le modèle de disposition en grille est un module de spécification CSS il permet de créer des mises en page en divisant l'espace d'affichage en régions principales ou en définissant des relations de taille, de position, de calques et d'empilement entre les éléments HTML primitifs.

Comme les tableaux, la grille permet d'aligner des éléments sous forme de colonnes et de lignes mais à la différence des tableaux, la grille n'a pas de structure de contenu. Ainsi, on peut créer de nombreuses mises en page qui n'auraient pas été possibles avec les tableaux. Ainsi, les éléments fils d'un conteneur en grille peuvent être positionnés afin qu'ils se chevauchent ou qu'ils se comportent comme des éléments positionnés.

Positioning allows you to take elements out of the normal document layout flow, and make them behave differently; for example sitting on top of one another, or always remaining in the same place inside the browser viewport. This article explains the different `position` values, and how to use them.

Prerequisites: HTML basics (study [Introduction to HTML](#)), and an idea of How CSS works (study [Introduction to CSS](#).)

Objective: To learn how CSS positioning works.

We'd like you to follow along with the exercises on your local computer, if possible — grab a copy of `0_basic-flow.html` from our GitHub repo ([source code here](#)) and use that as a starting point.

Introducing positioning

The whole idea of positioning is to allow us to override the basic document flow behavior described above, to produce interesting effects. What if you want to slightly alter the position of some boxes inside a layout from their default layout flow position, to give a slightly quirky, distressed feel? Positioning is your tool. Or if you want to create a UI element that floats over the top of other parts of the page, and/or always sits in the same place inside the browser window no matter how much the page is scrolled? Positioning makes such layout work possible.

There are a number of different types of positioning that you can put into effect on HTML elements. To make a specific type of positioning active on an element, we use the `position` property.

défilement de la page : https://developer.mozilla.org/en-US/docs/Learn/CSS/CSS_layout/Positioning

le positionnement est l'outil qu'il vous faut, car il rend de tels agencements possibles. Il y a de nombreux types différents de positionnement dont vous pouvez profiter sur les éléments HTML. Pour rendre effectif un type en particulier de positionnement sur un élément, nous utilisons la propriété `position`.

Le positionnement permet de sortir les éléments du cours normal de la composition du document, et leur permettre de se comporter différemment, par exemple de se placer sur un autre, ou de toujours rester à la même place dans le cadre d'affichage du navigateur.

Introduction au positionnement

L'idée globale du positionnement est de permettre le contrôle du cours normal de la mise en page classique pour produire des effets intéressants. Mettons que vous voulez modifier légèrement le placement de quelques boîtes par rapport à leur position par défaut dans le cours normal de mise en page du document pour donner une touche d'excentricité, ou bien, créer un élément d'interface utilisateur flottant au-dessus d'autres parties de la page tout en restant à la même place à l'intérieur de la fenêtre du navigateur quel que soit le point de

position: sticky

There is another position value available called `position: sticky`, which is somewhat newer than the others. This is basically a hybrid between relative and fixed position, which allows a positioned element to act like it is relatively positioned until it is scrolled to a certain threshold point (e.g. 10px from the top of the viewport), after which it becomes fixed. This can be used to for example cause a navigation bar to scroll with the page until a certain point, and then stick to the top of the page.

https://developer.mozilla.org/en-US/docs/Learn/CSS/CSS_layout/Positioning

position: sticky (collé)

Une autre valeur de positionnement disponible est position : sticky. Elle est récente. Il s'agit d'un mixte entre position relative et position fixe : l'élément à positionner est en positionnement relatif jusqu'à un certain seuil (par ex. 10px du haut de la fenêtre), seuil au-delà duquel il est en positionnement fixe. Ce positionnement s'utilise par exemple pour faire défiler une barre de navigation avec la page jusqu'à un certain point et ensuite coller en haut de la page.

The screenshot shows the Stack Overflow interface. At the top, there is a navigation bar with the Stack Overflow logo, links for 'Products', 'Customers', and 'Use cases', and a search bar. Below the navigation bar is a dark banner with the text: 'We're rewarding the question askers & reputations are being recalculated! [Read more](#).' The main content area features a question titled 'How to search a specific value in all tables (PostgreSQL)?' with a 'PUBLIC' tag. The question metadata indicates it was 'Asked 8 years, 8 months ago', 'Active 2 months ago', and 'Viewed 98k times'. The answer text reads: 'Here's a `pl/pgsql` function that locates records where any column contains a specific value. It takes as arguments the value to search in text format, an array of table names to search into (defaults to all tables) and an array of schema names (defaults all schema names). It returns a table structure with schema, name of table, name of column and pseudo-column `ctid` (non-durable physical location of the row in the table, see [System Columns](#))'.

```

CREATE OR REPLACE FUNCTION search_columns(
    needle text,
    haystack_tables name[] default '{}',
    haystack_schema name[] default '{}'
)
RETURNS table(schemaname text, tablename text, columnname text, rowctid text)
AS $$
begin
    FOR schemaname,tablename,columnname IN
        SELECT c.table_schema,c.table_name,c.column_name
        FROM information_schema.columns c
        JOIN information_schema.tables t ON
            (t.table_name=c.table_name AND t.table_schema=c.table_schema)
        WHERE (c.table_name=ANY(haystack_tables) OR haystack_tables='{}')
            AND (c.table_schema=ANY(haystack_schema) OR haystack_schema='{}')
            AND t.table_type='BASE TABLE'
    LOOP
        EXECUTE format('SELECT ctid FROM %I.%I WHERE cast(%I as text)=%L',
            schemaname,
            tablename,
            columnname,
            needle
        ) INTO rowctid;
        IF rowctid is not null THEN
            RETURN NEXT;
        END IF;
    END LOOP;
END;
$$ language plpgsql;

```

EDIT: this code is for PG 9.1 or newer. Also, you might want the version on [github](#) based on the same principle but adding some speed and reporting improvements.

Examples of use in a test database:

- Search in all tables within public schema:

```

select * from search_columns('foobar');
schemaname | tablename | columnname | rowctid
-----
public    | s3       | username   | (0,11)
public    | s2       | rename     | (7,29)
public    | w        | body       | (0,2)
(3 rows)

```

- Search in a specific table:

```

select * from search_columns('foobar','w');
schemaname | tablename | columnname | rowctid
-----
public    | w        | body       | (0,2)
(1 row)

```

- Search in a subset of tables obtained from a select:

```

select * from search_columns('foobar', array(select table_name::name from information_schema.tables
schemaname | tablename | columnname | rowctid
-----
public    | s2       | rename     | (7,29)
public    | s3       | username   | (0,11)
(2 rows)

```

jects Customers Use cases Search...

```

select * from public.w where ctid='(0,2)';
title | body | tsv
-----
toto | foobar | 'foobar':2 'toto':1

```

Variants

- To test again a regular expression instead of strict equality, like grep, this part of the query:

```
SELECT ctid FROM %I.%I WHERE cast(%I as text)=%L
```

may be changed to:

```
SELECT ctid FROM %I.%I WHERE cast(%I as text) ~ %L
```

- For case insensitive comparisons, you could write:

```
SELECT ctid FROM %I.%I WHERE lower(cast(%I as text)) = lower(%L)
```

share improve this answer

edited Aug 29 at 23:09

answered Apr 12 '14 at 21:37



Daniel Vérité
42.6k • 11 • 95 • 114