---
titre: Des commentaires avec hugo, staticman et heroku
auteur: subversive
date: 05-02-2021

Bonjour !

Tutoriel pour permettre l'écriture de commentaires sur un site internet statique avec hugo cms, staticman et heroku.

Objectif, offrir a vos visiteurs la possibilité d'écrire des commentaires (ou autres) sans passer par un organisme tel que Graphcomment ou Disqus,... Ainsi vous respectez la vie privée de vos utilisateurs.

Tous les tutos disponibles sur le net (que j'ai lu) sont en anglais, ils datent et certains liens ne fonctionnent plus, bref j'ai du faire un mix de tuto pour y arriver, du coup je vous propose un tuto en français.

N'oubliez jamais que ce qui fonctionne est sur les codes sources des sites. En ce sens si vous copiez-collez et qu'il y a une erreur, cherchez la mise à jour dans le code source sur github des sites qui tournent avec.

Edité le 28/08/21, article modifié cf :
- Compte githubBOT
- Protection contre le spam
Compte githubBOT

Requis

- Votre compte github principal.
- Un compte github vérifié qui sera votre bot (ex: githubbot). Conservez les identifiants et mots de passe.
- Un compte vérifié heroku. Conservez les identifiants et mot de passe.
- Heroku CLI.

J'utilise heroku pour ubuntu 20.04, les commandes sont un petit peu différentes mais la procédure reste la même.

Considérant que vous souhaitez installer staticman sur votre site, vous avez donc la capacité de gérer correctement votre site sous hugo (ou autre), de traduire les .toml en .yml et vice versa.

Bien il vous faudra de la patience pour ne manquer aucune étape.

Commençons !

Compte githubBOT

Vous n'avez plus besoin d'un compte githubBOT, faites de même pour utiliser votre compte.

Connectez vous à votre compte GithubBOT, allez dans paramètres du compte (settings du compte) -> Developer settings -> Personal access tokens -> Generate new token.

Nommez le, sélectionnez (select scopes) : repo (tous) + user (tous) => Generate token.
SAUVEGARDEZ bien votre token sur un fichier txt.

Application Heroku

Télécharger staticman via github avec le lien https :

```
git clone https://github.com/eduardoboucas/staticman.git
```

Allez dans le dossier staticman.
Créez un fichier, nommez-le Procfile, insérez-y npm start.

Créez une clé RSA :

```
openssl genrsa -out key.pem
```

Créez un fichier, nommez-le config.production.json, insérez-y :

- notes :
> votretoken ressemble a celui ci : b81f780f9f5f3be15a5595e618379756344cdsdk2
> pour la rsaPrivateKey, collez l'intégralité du fichier key.pem

```
{
  "githubToken": "votretoken",
  "rsaPrivateKey": "-----BEGIN RSA PRIVATE KEY\n-----la clé entière-----\nEND RSA PRIVATE KEY-----",
  "port": 8080
}
```

(toujours dans le dossier staticman)

Connectez votre compte heroku avec :

```
heroku login
```

Toujours dans le dossier cloné plus haut (staticman), créez une application heroku :

```
heroku create nomdevotreapplication
```

Configuration

(votretoken ressemble a celui ci : b81f780f9f5f3be15a5595e618379756344cdsdk2)

```
heroku config:set NODE_ENV="production"
heroku config:set GITHUB_TOKEN="votretoken"
heroku config:set RSA_PRIVATE_KEY="$(cat ./key.pem)"
```

Créez une branche de production :

```
git checkout -b production 55d1430
```

Ajoutez dans le fichier .gitignore pour ignorer le fichier configuration:
```
!config.production.json
```

Déploiement

```
git add .
git commit -m "Set up Staticman v3 for deployment to Heroku"
git push heroku production:master
```

Si tout c'est bien passé, en se rendant sur https://nomdevotreapplication.herokuapp.com/ vous devez lire :

Hello from Staticman version 3.0.0!

Votre app est disponible.

API sur github

Connectons votre githubBOT avec votre compteGITHUBprincipal.

Pour cela, depuis le dépôt de votre site, allez dans "settings" (paramètres de ce dépôt) -> Manage Access -> Invite a colaborator -> yourgithubBOT -> Invite.

Acceptez l'invitation via le mail de votre githubBOT.

API ponts et config

Depuis le fichier staticman.yml, j'ai choisi volontairement de désactiver la modération, mais faites à votre guise.

```
moderation: true
```

Cela va créer une "pull request" sur votre branche sélectionnée dans staticman.yml.

A la source (la RACINE) de votre site, créez un fichier staticman.yml, insérez-y :

```
comments:
 allowedFields: ["name", "comment"]
 branch: "master"
 commitMessage: "Nouveau commentaire"
 filename: "comment-{@timestamp}"
 format: "yaml"
 generatedFields:
  date:
   type: date
   options:
    format: "iso8601"
 moderation: false
```

```
 name: "Staticman"
 path: "data/comments/{options.slug}"
 requiredFields: ["name", "comment"]
 transforms:
  email: md5
```

Dans votre fichier config.yml, ajoutez ou complétez :

```
params:
 comments: true
 staticman:
  api: 'https://nomdevotreapp.herokuapp.com/v2/entry/githubprincipal/repositorydusite/master/comments'
```

Ajoutez également un dossier /data/comments/.gitkeep.
(Cela permet de maintenir le git push avec les dossiers. Essayez sans, les dossiers vides ne suivront pas sur github).

La variable "comments" vous permet de désactiver les commentaires depuis le fichier single.html, vous pouvez aussi l'utiliser dans les params d'un article (ou page : comments: false).

Afficher et écrire des commentaires depuis le site

default

Dans layout/_default/single.html insérez :

```
{{- if and ($.Site.Params.comments) (ne .Params.comments false) }}
{{- partial "comments.html" . }}
{{- end }}
```

html

Dans layouts/partials/comments.html (créez le fichier si besoin "comments.html") insérez-y :

```
<!-- On demande s'il ya bien l'API dans config.yml (s2) -->

<!-- Si le commentaire est BIEN parti, on remercie l'utilisateur. -->
<aside aria-label="note" class="note">
<section id="comment-thankyou" style="display: none;">
 <p class="comment-merci">Merci pour le commentaire ! <a rel="noopener noreferrer" title="Rafraîchir la page ?" href="{{ .Permalink }}">Rafraîchir la page ?</a></p>
</section>
</aside>


{{ with .Site.Params.staticman.api }}
<!-- L'API est présent, on peut continuer -->
<div class="classe-commentaire">

<!-- Bouton pour ouvrir l'espace commentaire (Anti-Spam1)-->
<button aria-label="Ouvrir espace commentaire" title="Ouvrir espace commentaire" id="staticman-button" onclick="showComments()">Espace commentaire</button>
```

```
<!-- Section qui permet l'affichage et la lecture des commenaites -->
<section id="staticman-comments" class="js-comments staticman-comments">

  <!-- Récupération des commentaires par fichier dans /data/comments/ -->
  {{ $slug := replace $.File.Path "/" "-" }}

  {{ if $.Site.Data.comments }}
    {{ $comments := index $.Site.Data.comments $slug }}
<!-- Affichage du nombre de commentaires
    {{ if $comments }}
      {{ if gt (len $comments) 1  }}
        <h3>{{ len $comments  }} commentaires</h3>
      {{ else }}
        <h3>{{ len $comments  }} aucun commentaire</h3>
      {{ end }}
    {{ else }}
      <h3>No comments</h3>
    {{ end }}
Fin de l'affichage du nombre de commentaires -->

<!-- Affichage d'un commentaire -->
    {{ $.Scratch.Set "hasComments" 0 }}
    {{ range $index, $comments := (index $.Site.Data.comments $slug ) }}
      {{ if not .parent }}
        {{ $.Scratch.Add "hasComments" 1 }}
        <article id="comment-{{ $.Scratch.Get "hasComments" }}" class="static-comment">
          <div class="comment-author">
            <strong>{{ .name }}</strong>
          </div>
          <div class="comment-timestamp">
            <a href="#comment-{{ $.Scratch.Get "hasComments" }}" title="Lien vers ce
commentaire">
              <time datetime="{{ .date }}">{{ dateFormat "2 Jan 2006 15:04 MST" .date}}</time>
            </a>
          </div>
          <div class="comment-content"><p>{{ .comment | markdownify }}</p></div>
        </article>
      {{ end }}
    {{ end }}
  {{ end }}


<!-- Formulaire pour écrire un commentaire -->
<form class="js-form form" id="commentsform" method="post"
action="{{ $.Site.Params.staticman.api }}">
<fieldset>
    <input type="hidden" name="options[redirect]" value="{{ $.RelPermalink |
absURL }}#comment-thankyou">
    <input type="hidden" name="options[slug]" value="{{ replace $.File.Path "/" "-" }}">
    <input type="hidden" name="options[parent]" value="">
    <label aria-label="Pseudonyme" for="yourname">Pseudonyme:</label>
```

```
  <input name="fields[name]" id="yourname" type="text" placeholder="Votre pseudonyme"
required/>
  <label aria-label="Commentaire" for="yourcomment">Commentaire:</label>
  <textarea name="fields[comment]" id="yourcomment" placeholder="A quoi pensez vous ?"
required></textarea>
  <output id="warningComment">En cliquant sur Envoyer vous acceptez la <a target="_blank"
rel="noopener noreferrer" title="Charte de publication" class=""
href="https://mondomaine.fr/chartedepublication.txt">Charte de publication</a></output>
  <!-- Anti-Spam2 -->
  <input type="hidden" id="youarenotarobot1">
  <!-- Anti-Spam3 -->
  <input aria-label="Résultat du Captcha" name="myCaptcha" class="jCaptcha" type="text"
placeholder="Tapez le résultat s'il vous plaît">
  <button aria-label="Envoyer le commentaire" type="submit" title="Envoyer le commentaire"
class="button" id="submitButton">Envoyer</button>
</fieldset>
</form>

</section>

<!-- Fin class-commentaire -->
</div>
<!-- Fin sécurité -->
{{ end }}
```

javascript

Le javascript est nécessaire dans cette config, à vous de le personnaliser en fonction de vos désirs.

Pour des soucis de performance, le script se trouve dans un autre fichier js. le voici :

```
/* Saticman + Anti-Spam
--------------------------------------------------------------------------------*/

// Permet d'afficher l'espace commentaire.
function showComments() {
// Remove button
var staticmanButton = document.getElementById('staticman-button');
staticmanButton.parentNode.removeChild(staticmanButton);
// Un-hide comments
var staticmanComments = document.getElementById('staticman-comments');
staticmanComments.style.display = 'block';
};

// Permet de calculer le captcha.

(function (root, factory) {
if (root === undefined && window !== undefined) root = window;
if (typeof define === 'function' && define.amd) {
// AMD. Register as an anonymous module unless amdModuleId is set
define([], function () {
  return (root['jCaptcha'] = factory());
```

```
});
} else if (typeof module === 'object' && module.exports) {
// Node. Does not work with strict CommonJS, but
// only CommonJS-like environments that support module.exports,
// like Node.
module.exports = factory();
} else {
root['jCaptcha'] = factory();
}
}(this, function () {

"use strict";

{
var generateRandomNum = function generateRandomNum() {
num1 = Math.round(Math.random() * 8) + 1;
num2 = Math.round(Math.random() * 8) + 1;
sumNum = num1 + num2;
};
//
// @param {Object}
// @param {Object}
// @param {Boolean}


var setCaptcha = function setCaptcha($el, options, shouldReset) {
if (!shouldReset) {
  $el.insertAdjacentHTML('beforebegin', "<canvas class=\"".concat(options.canvasClass, "\"\n
width=\"").concat(options.canvasStyle.width, "\" height=\"").concat(options.canvasStyle.height,
"\">\n              </canvas>\n          "));
  this.$captchaEl = document.querySelector(".".concat(options.canvasClass));
  this.$captchaTextContext = this.$captchaEl.getContext('2d');
  this.$captchaTextContext = Object.assign(this.$captchaTextContext, options.canvasStyle);
}

this.$captchaTextContext.clearRect(0, 0, options.canvasStyle.width, options.canvasStyle.height);
this.$captchaTextContext.fillText("".concat(num1, " + ").concat(num2, "
").concat(options.requiredValue), 0, 0);
};
// @param {Object}



var jCaptcha = function jCaptcha() {
var options = arguments.length > 0 && arguments[0] !== undefined ? arguments[0] : {};
this.options = Object.assign({}, {
  el: '.jCaptcha',
  canvasClass: 'jCaptchaCanvas',
  requiredValue: '=',
  resetOnError: true,
  focusOnError: true,
  clearOnSubmit: true,
  callback: null,
```

```
  canvasStyle: {}
}, options);

this._init();
};

var sumNum, num1, num2;
var numberOfTries = 0;
;
jCaptcha.prototype = {
_init: function _init() {
  this.$el = document.querySelector(this.options.el);
  generateRandomNum();
  setCaptcha.call(this, this.$el, this.options);
},
validate: function validate() {
  numberOfTries++;
  this.callbackReceived = this.callbackReceived || typeof this.options.callback == 'function';

  if (this.$el.value != sumNum) {
    this.callbackReceived && this.options.callback('error', this.$el, numberOfTries);
    this.options.resetOnError === true && this.reset();
    this.options.focusOnError === true && this.$el.focus();
    this.options.clearOnSubmit === true && (this.$el.value = '');
  } else {
    this.callbackReceived && this.options.callback('success', this.$el, numberOfTries);
    this.options.clearOnSubmit === true && (this.$el.value = '');
  }
},
reset: function reset() {
  generateRandomNum();
  setCaptcha.call(this, this.$el, this.options, true);
}
};
}

return jCaptcha;

}));

// Permet d'afficher/déployer un Captcha.
// Option, choisissez un nombre d'essai maximum pour valider le captcha.
   const maxNumberOfTries = 5;

   // Captcha SETUP Initial
   var myCaptcha = new jCaptcha({
      el: '.jCaptcha',
      canvasClass: 'jCaptchaCanvas',
      canvasStyle: {
   // properties for captcha stylings
   width: 100,
   height: 15,
   textBaseline: 'top',
```

```
  font: '15px Segoe UI bolder',
  textAlign: 'left',
  fillStyle: 'red',
    },

    // set callback function
    callback: function (response, $captchaInputElement, numberOfTries) {

      if (maxNumberOfTries === numberOfTries) {
        // nombre maximum de tentatives atteintes!
          // e.g. disable the form:
          form.removeEventListener('submit', formSubmit);
          $captchaInputElement.classList.add('disabled');
          $captchaInputElement.placeholder = 'Maximum attempts reached!';
          $captchaInputElement.setAttribute('disabled', 'true');
          document.getElementById("submitButton").setAttribute('disabled', 'true');
          return;
      }

      if (response == 'success') {
        // réponse valide !
          $captchaInputElement.classList.remove('error');
          $captchaInputElement.classList.add('success');
          $captchaInputElement.placeholder = "Succès !";

          // Donc maintenant on continue l'envoi du formulaire!

        // pseudonyme vide
        if(form.yourname.value == ""){
           form.warningComment.style.display = 'block';
           form.warningComment.innerText = "Veuillez saisir votre pseudonyme";
           return false;
        }
        // commentaire vide
        else if(form.yourcomment.value == ""){
           form.warningComment.style.display = 'block';
           form.warningComment.innerText = "S'il vous plait tapez un commentaire";
           return false;
        }
        // input caché n'est pas vide -> pas humain
        else if(form.youarenotarobot1.value != ""){
          return false;
        }
        // tout est bon !
        else{
           form.submitButton.disabled = true;
           form.warningComment.style.display = 'none';
           form.warningComment.innerText = "";
           form.submitButton.innerText = "Envoi...";
           document.getElementById("commentsform").submit();
           return true;
         }
      }
```

```
        if (response == 'error') {
          // le résultat n'est pas le bon
            $captchaInputElement.classList.remove('success');
            $captchaInputElement.classList.add('error');
            $captchaInputElement.placeholder = "S'il vous plaît essayez à nouveau !";
            return false;
        }

    }

  });

// Valider le JsCaptcha depuis un envoi du formulaire.
function formSubmit(e) {
  e.preventDefault();
  // myCaptcha validate
  myCaptcha.validate();
};

// J'appelle la fonction formSubmit si je clique sur submit depuis le formulaire
Id="commentsform".
var form = document.getElementById("commentsform");
form.addEventListener('submit', formSubmit);

// Le commentaire bien parti, script pour afficher le message merci!
if (/comment-thankyou/.test(window.location.href)) {
document.getElementById('comment-thankyou').style.display = 'block';
}else{
document.getElementById('comment-thankyou').style.display = 'none';
}

/* FIN Staticman + Anti-Spam section */
```

> Notes: Dans le partial footer, mettez votre script en bas des pages pour éviter les erreurs.

css

Pour styliser le formulaire et l'affichage avec les boutons c'est comme cela :

```
/* Styles for Staticman de lespace commentaire
-------------------------------------------------------------------------------------------------------
*/
/* Bouton pour ouvrir l'espace commentaire */
#staticman-button{
  color: var(--content);
  margin-top: var(--gap);
  border: 1px solid var(--border);
  padding-inline-start: var(--radius);
  border-radius: var(--radius);
  display: flex;
  overflow-x: auto;
```

```
  font-size: 1em;
  width: 100%;
  padding: 0.25em 0.75em;
  margin-bottom: 1rem;
}
/* Section staticman-comments Ou js-comments */
#staticman-comments {
  display: none;
}
.staticman-comments {
  color: var(--content);
  margin-bottom: 2rem;
}
/* Article seul */
.static-comment{
  color: var(--content);
  margin-top: var(--gap);
  border: 1px solid var(--border);
  padding-inline-start: var(--radius);
  border-radius: var(--radius);
  display: flex;
  overflow-x: auto;
}
/* Article auteur + content + date */
.staticman-comments .comment-author, .comment-content, .comment-timestamp {
  margin-top: 0;
  color: var(--content);
  line-height: 1.6;
}
.staticman-comments article{
  display: grid;
  grid-template-columns: 0.5fr 0.5fr;
}
.staticman-comments .comment-author {
  grid-column: 1;
  grid-row: 1;
  font-size: 0.8em;
}
.staticman-comments .comment-timestamp {
  grid-column: 2;
  grid-row: 1;
  text-align: right;
  margin-right: 5px;
  font-size: 0.8em;
}
.staticman-comments .comment-timestamp a{
  box-shadow: 0 1px 0;
}
.staticman-comments .comment-timestamp a:hover {
  font-weight: bolder;
}
.staticman-comments .comment-content{
  margin-top: 0.5rem;
```

```
    grid-column: 1/3;
    grid-row: 2;
    font-size: 1em;
}
.comment-content a{
    color: var(--content);
    border-bottom: 1px solid var(--content);
}
.comment-content a:hover{
    font-weight: bolder;
}
.staticman-comments .comment-content p {
    padding: 5px 1rem 5px 1rem;
}
/* Formulaire */
.js-form{
    color: var(--content);
    margin-top: var(--gap);
    border: 1px solid var(--border);
    padding-inline-start: var(--radius);
    border-radius: var(--radius);
    display: flex;
    overflow-x: auto;
}
fieldset {
    padding: 15px;
}

/* les Inputs du Formulaire (pseudo+jscaptcha) */
.staticman-comments input {
    color: var(--content);
    border: 1px solid var(--border);
    width: 100%;
    font-family:inherit;
    font-size: 0.85rem;
    margin-top: 0.5rem;
    margin-bottom: 1rem;
}
/* Les placeholder ^^ */
.js-form input::placeholder{
    color: var(--content);
        font-size: 1em;

}
.js-form textarea::placeholder{
    color: var(--content);
        font-size: 1em;
}
/* le textarea du formulaire (comment) */
.staticman-comments textarea {
    color: var(--content);
    border: 1px solid var(--border);
    vertical-align: top;
```

```
    height: 10em;
    width: 100%;
    font-family:inherit;
    font-size: 0.85rem;
    margin-top: 0.5rem;
    margin-bottom: 1rem;
}
/* label et bouton */
.staticman-comments label, .staticman-comments button {
    margin-top: 0.5rem;
    font-size: 0.85rem;
    font-size: 1em;
}
/* warning comment ouput */
#warningComment{
    color: var(--content);
    margin-top: 1rem;
    margin-bottom: 1rem;
    width: 100%;
    display: block;
    font-family:inherit;
    font-size: 0.85rem;
}
#warningComment a{
    color: var(--content);
    border-bottom: 1px solid var(--content);
}
#warningComment a:hover{
    font-weight: bolder;
}

/* bouton envoyer */
#submitButton{
    color: var(--content);
    border-bottom: 1px solid var(--content);
    font-size: 1em;
}
#submitButton:hover{
    font-weight: bolder;
}
/* post envoi du commentaire  aside note + section id thankyou + p*/
.note{
    color: var(--content);
}
.comment-merci{
    color: var(--content);
    display: flex;
    overflow-x: auto;
    width: 100%;
    padding: 0.25em 0.75em;
    margin-bottom: 1rem;
}
.comment-merci a{
```

```
  border-bottom: 1px solid var(--content);
}
.comment-merci a:hover{
  font-weight: bolder;
}

/* Anti-Spam limite d'essai atteinte css*/
.jCaptcha.disabled~.button {
    cursor: not-allowed;
}
/* JS NoScript Message*/
.no-script-text{
    color: var(--content);
    margin-top: var(--gap);
    border: 1px solid var(--border);
    padding-inline-start: var(--radius);
    border-radius: var(--radius);
    display: flex;
    overflow-x: auto;
    width: 100%;
    padding: 0.25em 0.75em;
    margin-bottom: 1rem;
}

/* écran < 800 */
@media screen and (max-width: 800px) {
  .static-comment .comment-content p{
    padding: 2.5px 0.5rem 2.5px 0.5rem;
  }
  fieldset {
    padding: 5px;
  }
}
```

A vous de choisir en fonction de votre architecture css, en fonction du thème choisi.

Pour styliser le captcha c'est plus complexe cherchez `// Captcha SETUP Initial` dans votre javascript.

Protection contre le spam

A la recherche de l'utilité, de la vie privée et de la performance.

Bon l'utilité, a part gêner le handicap...
Tout se fait côté client donc tout peut se casser.

- Buster: Captcha Solver for Humans

Les CAPTCHA diminuent les performances de vos sites web/apps.

Le REcaptcha favorise a outrance chromium et les comptes google.

Il faut que la réponse soit sur un serveur distant, sinon le robot (peut) lire la réponse.

Inutile si peu de visiteurs, car le spam est un marché, qui va ouvrir un commerce où il n'y a personne ?

J'utilise 3 protections (toutes dépassées):

- Pas de javascript = pas de commentaire. Il faut javascript pour afficher les commentaires car il faut cliquer sur le bouton "ouvrir espace commentaire".
- Un input caché `<input type="hidden" id="youarenotarobot1">`, le visiteur(non-robot/humain) ne le voit pas mais le robot oui, si il le remplit alors ce n'est pas un humain.
- `js-captcha de robiveli`.

Après quelques semaines d'essais. Un robot a cassé le pseudo anti-spam. La seule solution est l'autre solution.

autre solution

Posséder un serveur, y ajouter staticman + visual captcha.
Vous pouvez également interdire les url dans le contenu des commentaires.

C'est la seule solution FIABLE.

Notes importantes

Ces notes dates, certains passages sont a ne pas prendre en compte.

- Il n'y pas de protection contre le spam vraiment active, si jamais lisez l'article Tuto en anglais de 2020.
- Il n'y a pas de modération avant affichage.
- Ne pas hésiter à observer le code source du site
- Cette version est très simple, sans réponse entre commentaires, avec trois champs requis seulement.le code source du site
- Pour observer où se stocke un commentaire.

Astuce

- Je maîtrise mal le vocabulaire git mais avant de push -u vers le dépôt github, pensez à utiliser `git pull` pour « rapatrier » les nouveaux commentaires sur votre dépôt local.

- Dans la version (v3) de staticman, il n'y aurait plus besoin du Personal access tokens. Mais je ne sais comment procéder.

- L'on peut passer directement via le bouton deploy to heroku, avec ce commentaire vous devriez pouvoir y arriver sans passer par heroku CLI.

Problèmes rencontrés

- L'utilisation de `{{ $slug := replace $.RelPermalink "/" "-" }}` avec `name="options[slug]"` ne permet l'affichage des commentaires sur les articles où les titres ont des accents. Solution trouvée avec `{{ replace $.File.Path "/" "-" }}`.
- J'ai supprimé du script de js-captcha, je ne sais a quoi il sert. `Il ne fallait pas le faire`.
- Pour utilsier ReCaptcha avec staticman, le script n'est pas dispo sur les fichiers de ce site.
- J'ai un souci de cache-control depuis que les commentaires sont disponibles `[résolu]`.

- Il faut reconstruire le site pour afficher un commentaire, ce qui peu prendre du temps, surtout si le dino (herokuapp) dort.. C'est faux !
- Il faut patienter le temps que le dino (votreapp) envoi les données vers le dossier /data/, rafraîchir la page, r-ouvrir l'espace commentaire et hop. La données dans le dossier /data/ s'affichent sans latence.

Sources

tutos

Tuto en anglais de 2018
Tuto en anglais de 2019
Tuto en anglais de 2020
Tuto en anglais de 2020 2

fichiers

Ceux au complet dont je me suis servi.

staticman-js-common.js
staticman-styles.css
staticman.html
js-captcha-index.js
js-captcha-index.html