

```
let rec learn "ocaml" =  
  "a webapp" :: "written in ocaml" ::  
  "useful to" :: teach "programming"  
and teach ocaml = try learn ocaml with  
this_presentation -> ["tame ocaml!"]
```

: [@ErikMartinDorel](https://twitter.com/ErikMartinDorel)

Équipe ACADIE, Dépt. Fiabilité des Systèmes et des Logiciels, IRIT  
Univ. Toulouse III – Paul Sabatier





UNIVERSITÉ  
TOULOUSE III  
PAUL SABATIER



Mardi 11 Octobre 2022

# Résumé de l'histoire de Learn-OCaml

- Conçu comme moteur du MOOC OCaml (2015, 2017, 2018, 2019, 2020)
  - Supports de cours créés à l'Univ. Paris Diderot (Roberto Di Cosmo, Yann Régis-Gianas, Ralf Treinen...)
-  ● Plateforme développée par OCamlPro (Benjamin Canou, Çağdaş Bozman, Grégoire Henry, Louis Gesbert, Pierrick Couderc...)
  - Basé sur openEDX (parti-pris : évaluation dans le navigateur, côté client)
- 2016 : OCamlPro rend Learn-OCaml indép./openEDX, licence AGPL
-  ● 2018 : © transféré à la Fondation OCaml (OCSF), licence MIT
- 2016-2017 : Première utilisation de Learn-OCaml à l'UT3
- Déc. 2020 : co-mainteneurs := Érik MD et Yann Régis-Gianas
- Au 2022-10-10 : 285 PRs intégrées de 32 contributeurs
- Utilisé notamment en France, au Portugal, Royaume-Uni, Canada, aux États-Unis :

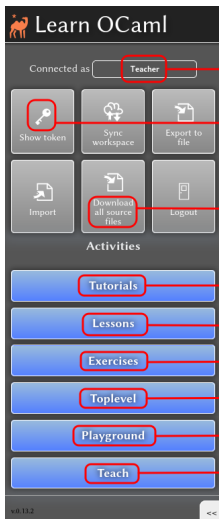
<https://github.com/ocaml-sf/learn-ocaml-places>

# URL démo Learn-OCaml

<https://erikmd.github.io/learn-ocaml-meetup-demo/>



# Page d'accueil de Learn-OCaml: résumé des fonctionnalités



Learn OCaml

Connected as **Teacher**

**Show token** → Token (⇒ BACKEND): unique string (e.g. GX9-HBS-1KS-A1J) serving as login+password (can't be changed!)

**Sync workspace**

**Export to file**

**Import**

**Download all source files** → Download a .zip archive (⇒ BACKEND) with all submitted exercises (i.e., with the Grade button, not just Sync)

**Logout**

Activities

**Tutorials** → "Interactive tutorials": sequence of several topics with snippets that can be automatically copied to a toplevel.

**Lessons** → Lectures: sequence of web slides with -- → arrows and ocaml syntax highlighting.

**Exercises** → (The Main Feature™) Exercises with description, prelude, template, editor, toplevel, and custom graders.

**Toplevel** → Client-side toplevel (less useful than Playgrounds!)

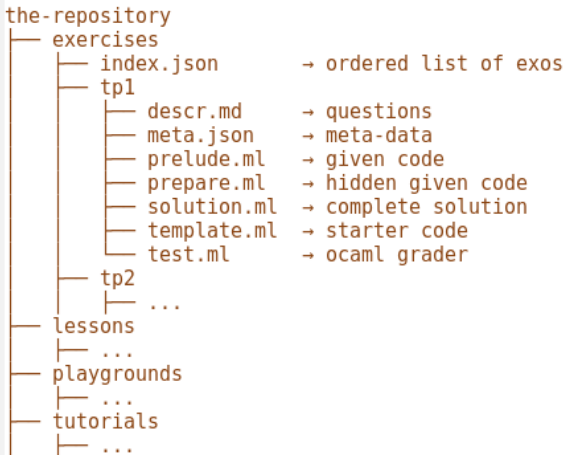
**Playground** → Playgrounds: editor, toplevel, and customizable preludes.

**Teach** → Teacher Dashboard (⇒ BACKEND): track the progress of students on the exercises (if they clicked on Grade)

Choose an activity.

va 11.2

# Définition d'exercices Learn-OCaml: le format "exodir" I



```
$ learn-ocaml build serve --repo=the-repository
```

# Définition d'exercices Learn-OCaml: le format "exodir" II

```
the-repository/exercises/index.json:
{
  "learnocaml_version": "1",
  "groups":
  { "group1":
    {
      "title": "Some group of exercises",
      "exercises": [ "tp1", "tp2" ]
    }
  }
}
```

## Documentation

[https://ocaml-sf.org/learn-ocaml/exercises\\_format.html](https://ocaml-sf.org/learn-ocaml/exercises_format.html)

# Résumé de l'architecture de Learn-OCaml

- dépendances opam : ocaml, lwt, et js\_of\_ocaml
- notation **côté client** (un "exploit" est toujours possible actuellement)
- déploiement **statique** :
  - un serveur "backend" est facultatif avec learn-ocaml !
  - <https://github.com/erikmd/learn-ocaml-meetup-demo>
- déploiement "full-stack" avec un "backend" :
  - avec docker/docker-compose et une image [ocamlsh/learn-ocaml](#)
  - ou avec [learn-ocaml-essok](#) project
  - ou avec des binaires [learn-ocaml](#) (démon à la fin ?)
  - **fournit** comptes étudiant et enseignant (avec un "dashboard")

# learn-ocaml.el *a.k.a.* learn-ocaml-mode (1/2)

- par Manuel Cabarcos-Baulina, Louis Ayroles, Erik MD
- Project logiciel séparé :  
<https://github.com/pfitaxel/learn-ocaml.el> (MIT license)
- Emacs/Tuareg front-end intégré dans MELPA:  
<https://melpa.org/#/learn-ocaml>
- Utilise `learn-ocaml-client`
- Avantages :
  - un IDE standard OCaml (**Merlin+Eldoc**) pour développer les exercices
  - les élèves stockent leur `exercice.ml` directement sur leur machine
  - **notation** via un serveur Learn-OCaml (TOKEN ou mail/passwd)
  - **retour très rapide** (pas de JavaScript)
- "Bottleneck":
  - PR à intégrer: [#481](#) (lève une limitation majeure)



# learn-ocaml.el *a.k.a.* learn-ocaml-mode (2/2)

learn-ocaml.el : dashboard avec la liste des exercices

```
emacs@presto
File Edit Options Buffers Tools LearnOCaml Exercise-Id: Help
[Icons: Search, Save, Undo, Redo, Refresh]
[Refresh List | TAB / S-TAB : Navigate | q : Close List]
-----
LearnOCaml [directory] (-/) [Change & refresh]

Démo de l'environnement learn-ocaml

Démo de l'environnement learn-ocaml
Commencez par consulter cet exercice
Difficulty: 0/4 id: demo
[Browse subject] [Get template] [Open .ml] [Get last saved version]

PFITA/TP1 : Types et expressions, filtrage et n-uplets

PFITA - Thème 5
Filtrage par motifs
Difficulty: 2/4 id: pfita-theme5
[Browse subject] [Get template] [Open .ml] [Get last saved version]

PFITA - Thème 6 - TPne
Révisions CTD 1-3
Difficulty: 2/4 id: pfita-theme6
[Browse subject] [Get template] [Open .ml] [Get last saved version]

PFITA/TP2 : Filtrage et récursion sur les entiers

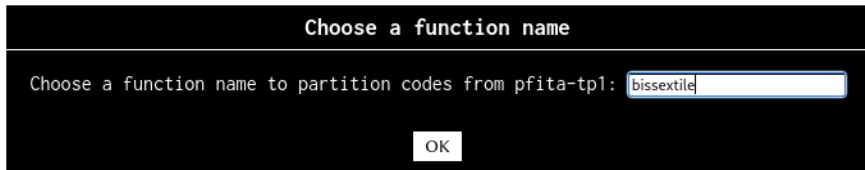
PFITA - Thème 7
Récursion sur les entiers
Difficulty: 2.5/4 id: pfita-theme7
[Browse subject] [Get template] [Open .ml] [Get last saved version]

PFITA/TP3 : Listes et récursion sur les listes

U:%*- *learn-ocaml-exercise-list* Top (1,0) (Fundamental LearnOCaml)
```

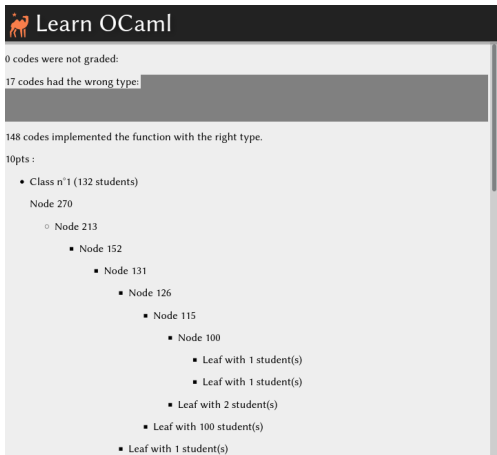
# Dissimilarity analysis (par Alexandre Moine, Yann R.-G.) I

- Fonctionnalité de Learn-OCaml **partition-view**  
Teacher Dashboard → **clic-molette** sur un exo (e.g. pfita-tp1)
- Exemple:



# Dissimilarity analysis (par Alexandre Moine, Yann R.-G.) II

- Exemple:



**Learn OCaml**

0 codes were not graded:

17 codes had the wrong type:

148 codes implemented the function with the right type.

10pts :

- Class n°1 (132 students)
  - Node 270
    - Node 213
      - Node 152
        - Node 131
          - Node 126
            - Node 115
              - Node 100
                - Leaf with 1 student(s)
                - Leaf with 1 student(s)
                - Leaf with 2 student(s)
                - Leaf with 100 student(s)
                - Leaf with 1 student(s)

# Static binaries as release assets (Louis Gesbert, Erik MD)

- Pour chaque release, des binaires statiques sont dispo pour {Linux, Mac}: <https://github.com/ocaml-sf/learn-ocaml/releases>

## ▼ Assets 9

 <a href="#">learn-ocaml-client-darwin-x86_64</a>
 <a href="#">learn-ocaml-client-linux-x86_64</a>
 <a href="#">learn-ocaml-darwin-x86_64</a>
 <a href="#">learn-ocaml-linux-x86_64</a>
 <a href="#">learn-ocaml-server-darwin-x86_64</a>
 <a href="#">learn-ocaml-server-linux-x86_64</a>
 <a href="#">learn-ocaml-www.zip</a>

- `learn-ocaml-client`: client pour **Tuareg+Merlin**, plus rapide que JS
- `learn-ocaml`: binaire pour servir une instance
- `learn-ocaml-server`: binaire plus rapide que `learn-ocaml` serve

# Support d'exercices basés sur Vg dans Learn-OCaml

- <https://opam.ocaml.org/packages/vg/>
- "Declarative 2D vector graphics in OCaml"
- Support Learn-OCaml: implementation en cours par Étienne Marais, Émile Rolley, Yann R.-G.
- "Bottlenecks":
  - Utiliser le moteur de rendu des navigateurs?  
Réimplémenter un moteur de rendu dans OCaml?
  - Comprendre comment générer un rapport de test aussi utile que possible pour les étudiants?

## use\_passwd/use\_moodle (stagiaires IRIT & Erik MD)

- Authentication forte (par e-mail/password, qui peut être changé)
  - Ajouter `"use_passwd":true` dans `server_config.json`
  - Protocole SMTP: envoi de mail automatique pour chgt mail/password
  - Support dans Tuareg/Merlin/learn-ocaml-mode : OK
- Moodle auth (pas besoin de droits d'admin, just le rôle enseignant) :
  - Ajouter `"use_moodle":true` dans `server_config.json`
  - Protocoles OAuth+LTI: les étudiants se connectent via Moodle
- Tâches restantes:
  - Migrer la branche `oauth-moodle branch` : OCaml 4.05 → 4.12
  - Ajouter la dépendance `irmin` (requiring OCaml > 4.08)
  - Étendre la documentation
  - Livrer `learn-ocaml.el` en même temps

## Pour en savoir plus

- Articles JFLA
- Documentation sur <https://ocaml-sf.org/learn-ocaml/>
- ECM "exodir" (sources de la démo !) :  
<https://github.com/erikmd/learn-ocaml-meetup-demo-repository/tree/master/src/exercises>
- Exposé [Learn-OCaml: How to contribute?](#) (sur ma page web)

# Questions ?



# OCaml



# Questionnaire d'utilisabilité

Sondage en 2019 : 56 étudiants y avaient répondu. 3 parties :

- 1 Questionnaire d'utilisabilité standard en 10 questions, obtenant un score de 77. En pratique, un score  $> 68$  correspond à un système *plus utilisable que la moyenne*.
- 2 Question plus ciblée : «Je trouve que cet environnement de TP est utile pour s'initier à un langage de programmation tel qu'OCaml ?» (échelle de 1 à 5) : score de 86 %.
- 3 Points positifs et suggestions d'améliorations :
  - + portabilité
  - + feedback (erreurs de compilation, tests automatisés)
  - + animations, ergonomie de l'application web
  - - lenteur sur certains PCs (OCaml étant émulé en JavaScript)