Dictionnaire

- `Cargo.toml` définit les dépendances (les crates) et leur versions minimum à inclure dans le projet, équivalent du `package.json` de NPM
- `crate`: officiellement la plus petite unité de compilation avec cargo, concrètement chaque projet contient un ou plusieurs dossiers avec un `Cargo.toml`
- `crates.io`: le registre officiel des crates publiée pour l'écosystème Rust, l'équivalent de `npmjs.com` pour l'écosystème Javascript, ou `mvnrepository.com` pour Java

Etat de l'art

Format de données humainement éditable existants

Ces recherches se focalisent sur les syntaxes qui ne sont pas spécifique à un domaine ou qui seraient complètement déliée de l'informatique ou de l'éducation. Ainsi, l'auteur ne présente pas Cooklang [1], qui se veut une langage de balise pour les recettes de cuisines, même si l'implémentation du parseur en Rust [2] pourra servir pour d'autres recherches. Contrairement aux languages de programmation qui existent par centaines, les syntaxes de ce genre ne sont pas monnaies courantes. Différentes manières de les nommer existent: language de balise (markup language), format de donnée, syntaxes, langage de donnée, language spécifique à un domaine (de l'anglais Domain Specific Language - DSL), … Les mots-clés utilisés suivants ont été utilisés sur Google, la barre de recherche de Github.com et de crates.io: `data format`, `human friendly`, `human writable`, `human readable`.

KHI - Le langage de données universel

D'abord nommée UDL (Universal Data Language) [3], cette syntaxe a été inventée pour mixer les possibilités du JSON, YAML, TOML, XML, CSV et Latex, afin de supporter toutes les structures de données modernes. Plus concrètement le markup, les structs, les listes, les tuples, les tables/matrices, les enums, les arbres hiérarchiques sont supportés. Les objectifs sont la polyvalence, un format source (fait pour être rédigé à la main), l'esthétisme et la simplicité.

```
{article}:
uuid: 0c5aacfe-d828-43c7-a530-12a802af1df4
type: chemical-element
key: aluminium
title: Aluminium
description: The <@element>:{chemical element} aluminium.
tags: [metal; common]

{chemical-element}:
symbol: Al
number: 13
stp-phase: <Solid>
melting-point: 933.47
boiling-point: 2743
density: 2.7
electron-shells: [2; 8; 3]

{references}:
wikipedi
[-blue]
```

Liste 1. – Un exemple de question à choix multiple tiré de leur documentation [4]. L'option correcte `white` est préfixée par `+` et les 2 autres options incorrectes par `-`. Plus haut, `[!...]` décrit une consigne, `[?...]` décrit un indice.

```
{
    "markup": "[.multiple-choice-1]\n[!What color is milk?]\n[+white]\n[-red]\n[-
blue]",
    "bit": {
        "type": "multiple-choice-1",
        "format": "text",
        "item": [],
        "instruction": [ { "type": "text", "text": "What color is milk?" } ],
        "body": [],
        "choices": [
            { "choice": "white", "item": [], "isCorrect": true },
            { "choice": "red", "item": [], "isCorrect" : false },
            { "choice": "blue", "item": [], "isCorrect" : false }
        ],
        "hint": [ { "type": "text", "text": "Cows produce milk." } ],
        "isExample": false,
        "example": []
    }
}
```

Liste 2. – L'équivalent JSON du choix multiple de Liste 1, tiré de leur documentation [4]

Open Taskpool, projet qui met à disposition des exercices d'apprentissage de langues [5], fournit une API JSON utilisant le JSON data model de Bitmark.

Demander à Open Taskpool des exercices d'allemand vers anglais autour du mot `school` de format `cloze` (texte à trou), se fait avec cette simple requête:
`https://taskpool.taskbase.com/exercises?translationPair=de->en&word=school&exerciseType=bitmark.cloze`.

```
...
"cloze": {
    "type": "cloze",
    "format": "text",
    "instruction": "Gegeben: \"Früher war hier eine Schule.\", schreiben Sie das
fehlende Wort",
    "body": [
        { "type": "text", "text": "There used to be a " },
        {
            "type": "gap",
            "solutions": [ "school" ],
            "answer": { "text": "" }
        },
        { "type": "text", "text": " here." }
    ]
},
...
```

Liste 3. – Extrait simplifié de la réponse JSON, respectant le standard Bitmark [6]. La phrase `There used to be a ___ here.` doit être complétée par le mot `school` en s'aidant du texte en allemand.

Un autre exemple d'usage se trouve dans la documentation de Classtime [7], on voit que le système de création d'exercices est basé sur des formulaires. Ces 2 exemples donnent l'impression que la structure JSON est plus utilisée que le markup. Au vu de tous séparateurs et symboles de ponctuations à se rappeler, la syntaxe n'a peut-être pas été imaginée dans le but d'être rédigée à la main directement. Finalement, Bitmark ne spécifie pas de type d'exercices programmation nécessaire à PLX.

NestedText — Un meilleur JSON

NestedText se veut être human-friendly, similaire au JSON mais pensé pour être facile à modifier et visualiser par les humains. Le seul type de donnée scalaire supporté est la chaîne de caractères, afin de simplifier la syntaxe et retirer le besoin de mettre des guillemets. La différence avec le YAML, en plus des types de données restreint est la facilité d'intégrer des morceaux de code sans échappements ni guillemets, les caractères de données ne peuvent pas être confondus avec NestedText [8].

```
Margaret Hodge:
    position: vice president
    address:
        > 2586 Marigold Lane
        > Topeka, Kansas 20682
    phone: 1-470-555-0398
    email: margaret.hodge@ku.edu
    additional roles:
        - new membership task force
        - accounting task force
```

Liste 4. – Exemple tiré de leur README [8]

Ce format a l'air assez léger visuellement et l'idée de faciliter l'intégration de blocs multi-lignes sans contraintes de charactères réservée serait utile à PLX. Cependant, tout comme le JSON la validation du contenu n'est pas géré directement par le parseur mais par des librairies externes qui vérifient le schéma [9]. De plus, l'implémentation officielle est en Python et il n'y a pas d'implémentation Rust disponible; il existe une crate réservée mais vide [10].

SDLang - Simple Declarative Language

SDLang se définit comme « une manière simple et concise de représenter des données textuellement. Il a une structure similaire au XML: des tags, des valeurs et des attributs, ce qui en fait un choix polyvalent pour la sérialisation de données, des fichiers de configuration ou des langages déclaratifs. » (Traduction personnelle de leur site web [11]). SDLang définit également différents types de nombres (32bit, 64bit, entier, flottant, …), 4 valeurs de booléans (`true`, `false`, `on`, `off`) comme en YAML, différents formats de dates et un moyen d'intégrer des données binaires encodées en Base64.

```
// This is a node with a single string value
title "Hello, World"

// Multiple values are supported, too
bookmarks 12 15 188 1234

// Nodes can have attributes
author "Peter Parker" email="peter@example.org" active=true

// Nodes can be arbitrarily nested
contents {
section "First section" {
paragraph "This is the first paragraph"
paragraph "This is the second paragraph"
}
}

// Anonymous nodes are supported
"This text is the value of an anonymous node!"

// This makes things like matrix definitions very convenient
matrix {
1 0 0
0 1 0
0 0 1
}
```

Liste 5. – Exemple tiré de leur site web [11]

Ce format s'avère plus intéressante que les précédentes de part la densité d'information, par exemple avec l'auteur décrit par son nom, email et un attribut booléan sur une seule ligne. Il est cependant regrettable de voir de les strings doivent être entourées de guillemets et les textes sur plusieurs lignes doivent être entourés de backticks `

Bibliographie

[1] A. Dubovskoy, « Cooklang – Recipe Markup Language ». [En ligne]. Disponible sur: https://cooklang.org/

[2] A. Dubovskoy, « Canonical Cooklang parser in Rust ». [En ligne]. Disponible sur: https://github.com/cooklang/cooklang-rs

[3] Torm, « udl v0.3.1 - Parser for UDL (Universal Data Language) ». [En ligne]. Disponible sur: https://crates.io/crates/udl

[4] bitmark Association, « Quizzes - .multiple-choice, .multiple-choice-1 ». [En ligne]. Disponible sur: https://docs.bitmark.cloud/quizzes/#multiple-choice-multiple-choice-1

[5] Taskbase, « open-taskpool - 12,000 UK 🇺🇦 → DE 🇩🇪 & DE 🇩🇪 → EN🇬🇧 learning tasks ready for you to use. ». [En ligne]. Disponible sur: https://github.com/taskbase/open-taskpool

[6] bitmark Association, « Quizzes - .cloze (gap text) ». [En ligne]. Disponible sur: https://docs.bitmark.cloud/quizzes/#cloze-gap-text

[7] Classtime, « Créer la première question / le premier jeu de questions ». [En ligne]. Disponible sur: https://help.classtime.com/fr/comment-commencer-a-utiliser-classtime/creer-la-premiere-question-le-premier-jeu-de-questions

[8] K. Kundert, « NestedText — A Human Friendly Data Format ». [En ligne]. Disponible sur: https://github.com/KenKundert/nestedtext

[9] Ken et K. Kundert, « NestedText documentation - Schemas ». [En ligne]. Disponible sur: https://nestedtext.org/en/latest/schemas.html

[10] bob22z, « docs.rs - Crate nestedtext ». [En ligne]. Disponible sur: https://docs.rs/nestedtext/latest/nestedtext/

[11] S. Ludwig, « SDLang, Simple Declarative Language ». [En ligne]. Disponible sur: https://sdlang.org/