Présentation du langage

Nous avons choisi le langage Uiua. C'est un langage de programmation orienté liste et basé sur l'utilisation d'un stack. Il est fortement influencé par APL et BQN. Le langage est à la fois interprétable et compilable. L'interpréteur et le compilateur sont écrit en Rust et est disponible sur crates.io. Il est aussi intégrable en tant que librairie dans un projet Rust.

Il est très concis et permet d'effectuer des opérations sur des listes d'éléments plutôt que sur des éléments individuels.

```
Xy \leftarrow ^{\circ} \otimes \boxplus \exists . \div 2: - \div 2, \uparrow .500
Rgb \leftarrow [:^{\circ} \boxminus \times .Xy \not \leftarrow \triangle \vdash Xy0.5]
u \leftarrow ! < 0.2: > 0.7. + \times 2 \times .:^{\circ} \boxminus Xy
c \leftarrow > \checkmark / \mathbb{C} \quad Xy
\otimes \mathbb{C}: \neg u \quad c1 \quad + 0.1 \quad \downarrow x \in 0.95 Rgb
```

Logo de Uiua et code utilisé pour le générer

La plupart des fonctions intégrées à Uiua sont représentées par des symboles, ce qui rend la lecture du code plus difficile mais permet de réduire la taille du code. Les symboles sont toutefois choisis pour repésenter au mieux la fonction (example \sin est associé à \sim).

Afin de ne pas avoir à utiliser un clavier spécial, les fonctions sont aussi accessibles via leur nom. L'outil de compilation permet d'automatiquement remplacer les noms par les symboles correspondants.



Différents symboles et leur nom

Malgré les symboles, l'intégration aux environnements de développement permet de voir les symboles sous forme de texte ainsi qu'une partie de la documentation associée.

```
find p
p find | 2
Find the occurences of one array in another

Documentation

A 1 marker will be placed the the start of each occurence of the first array in the second array.

p 5 [1 8 5 2 3 5 4 5 6 7]

[0 0 1 0 0 1 0 1 0 0]
```

Intégration dans VSCode

Types de données

Uiua ne possède que quelques types de données: les nombres, les charactères, les listes et les "box".

Les nombres peuvent être des entiers, des flottants et des nombres complexes. Les caractères sont des caractères unicode.

Les listes sont des listes d'éléments de même type. Et les "box" permettent de stocker des éléments de différents types dans une même liste.

Une grande partie des fonctions sont dites "pervasives" afin de pouvoir manipuler une liste de box sans avoir besoin de les déballer.

Code	Output
1	1
1.1	1.1
€ 3 4	4+3i
[1 2 3]	[1 2 3]
[1_0 0_1]	11001
@a	@a
"abc"	"abc"
□1	1
□[1 0]	[0 1]

Représentation des types de données

Les chaines de caractères sont représentées par des listes de charactères mais du sucre syntaxique est disponible pour les déclarer. Une liste de box peut aussi être déclarée de manière plus lisible.

```
[@H @e @l @l @o] "Hello"
```

Sucre syntaxique pour les chaines de caractères et les listes de box

Fonctions

Les fonctions sont définies par des symboles ou des noms. Il n'a pas de paramètres nommés: ils sont directement extraits du stack et les valeur de retour sont placées sur le stack. La plupart du temps la signature est inférée par le compilateur mais il est parfois nécessaire de spécifier le nombre de paramètres attendus et retournés (par example pour les fonctions récursives).

Fonctions système

Uiua possède un certain nombre de fonctions système qui permettent d'accéder à des fonctionnalités de bas niveau (système de fichiers, sockets, appel de commandes) ainsi que des fonctions pour différent types de médias (images, sons et gifs).

Foreign Function Interface

Il est aussi possible d'appeler des fonctions définies dans des librairies externes. Cela permet d'étendre les fonctionnalités de Uiua en utilisant des librairies écrites en Rust, C ou C++ par exemple.

Motivation du choix

Le paradigme est totalement différent de ce que nous avons l'habitude de voir ce qui force un changement drastique de réflexion. Les possibilités avec les types de médias nous ont aussi beaucoup intéressé pour le projet.

Projet

Notre projet consiste en une application graphique permettant de générer des images de manière paramétrique. L'interface graphique sera réalisée en utilisant la librairie raylib et permettra de démontrer les capacitées de la FFI. La génération d'image permet de montrer les capacités de manipulation de données multi-dimensionelles du language, les fonctions de manipulation de format d'image ainsi que les fonctions système pour la sauvegarde de fichiers.

Aperçu du cahier des charges

- Interface graphique simple
 - o Modification en temps réel des paramètres
- Génération d'images paramétriques
 - o Formes pré-définies (cercles, carrés, ...)
 - Courbes paramétriques (Bézier)
- Sauvegarde des images générées
 - Choix du format (PNG, JPEG, BMP, ICO, QOI)
 - Choix du nom