Devoxx 2023

Table des matières

Rendons le DDD aux devs (Dorra BARTAGUIZ et Arnaud THIEFAINE)

(notes perdues…)

De Java à Scala 3, des lambdas à la programmation fonctionnelle: un code plus lisible, plus expl...

--- Java à Scala --- (suite)

[RF] github

e.g. jeu de Uno

Carte Uno a un couleur et une valeur
Sauf certaines cartes
=> On peut passer en option la couleur et la valeur

On a plusieurs cartes
=> On peut faire une enum

Il y a plusieurs valeurs
=> Nouvel enum
Idem sur les couleurs

Dans mon enum de cartes je peux me représenter des cartes de type, prend 2 cartes, prends 4 cartes, passe ton tour, etc.

Prendre une carte
=> Fonction
  => Case
On ne peut oublier de cas, il faut tous les définir

Et la partie de Uno...

Etat : avant de jouer, prêt à jouer, pendant
=> Case

Qu'est ce que ça veut dire
- joueurs
- sens de la partie
    Case horaire/anti horaire
- pioche
- talon

Et le joueur...
Id ? Safeuuid (doit assurer l'unicité)

==> Type opaque
Dont on n'a pas accès aux membres ou constructeur du type de façon normale
  e.g. SafeUUID
  Utilité : on ne fausse pas l'id qui doit être unique

Encapsuler les exceptions avec Try()
  Left (convention : erreur)
  Right (convention: success)

SafeUUID est un type opaque riche
Type opaque e.g. une personne avec un nom de type string et et prénom de type string, pour ne pas les interchanger dans l'utilisation, on type le nom par le type nom (string) et prenom avec le type prenom (string), donc on ne peut pas les échanger par erreur
Dans l'exécution on n'a plus besoin de manipuler le type mais la valeur directement (gain de faciliter à l'écriture /réflexion, dans perte dans l'exécution)

La majorité du Scala est de faire du pattern matching

[Q&A] si le SafeUUID vient avec un input ?
Une entrée doit avoir une sortie, si ça ne se passe pas bien, on sort une exception qu'il doit donc traiter

[RQ] L'intérêt est que la validation est portée par le Type (gros pouvoir en termes de maintenance, on ne peut pas perdre la règle par régression du code, car on devrait changer le type pour ça, en java on pourrait facilement enlever le bout de code qui check la validation)

Et les tests...
Et les règles métier...

==> TDD (on écrit les tests avant le code des commandes/instructions)

Rappel : les possibilités doivent être dans le co-domaine (e.g. et si le joueur pose une carte invalide -> le joueur prend 2 cartes)

L'exemple suit le DDD donc indépendant de l'infra, des datas donc testable n'importe où

[RF] https://github.com/simplyscala/fpscala3_uno

L'exemple fonctionne jusqu'au bout (on peut même ajouter des règles métier)

==> Système d'effet

Solution en scala pour gérer les effets
- Ne pas les gérer
- Future[A] ça ne marche pas
- les effets comme valeur (Cats effect, ZIO, en cours)
- en cours plus loin... Capabilities

Qu'est ce qu'un effet
- modèle de contrôle clair
- suppr des erreurs de synchronisation
- parallélisation

Pourquoi ça marche ? Parce que c'est de la composition de fonction

LOOM [RF] photo println

Se rapproche de la notion de machine à états finis ([RQ] d'Erik) :
[web] https://commentouvrir.com/definitions/les-machines-a-etats-finis-expliquees/

Contact @ugobourdon pour recup slides ou questions

Nix : Besoin d'env de développement ultra robuste? On se crée un env immutable et reproductible

https://cfp.devoxx.fr/2023/talk/RHF-6000/Besoin_d'env_de_developpement_ultra_robuste%3F_On_se_cree_un_env_immutable_et_reproductible_avec_Nix_en_quelques_minutes!

Nicolas SAVOIS

Environnement reproductible et fiable

Langage fonctionnel et immutable
Aussi un gestionnaire de packet
Sous MacOS et Linux
Aussi une distribution de Linux

nix-repl

Déclaration des strings
Des variables
Des tableaux
Des structures (format analogue au JSON)
Des fonctions
[RF] photo code

Écrire dans des fichiers

nix eval -f mon fichier.nix

Il n'y a qu'un seul opérateur pour nix, pour en mettre plusieurs "let"

[RF] photo code 2 (en cours code incomplet) voir vidéo quand dispo

nix-shell
C'est une fonction
Qui importe un package

Impure ?
Rien d'installé, uniquement que ce qui déclarer
=> On peut passer des buildInputs [RF] photo avec cowsay

Pure ?
--pure

pkgs
Vient de mon OS (donc dépend de la distribution)

==> Comment "pin" mon paquet ?

1. Importe mon paquet
2. in pkgs.stdenv.mkDerivation
3. nix-build
4. Ajouter src
5. On refait nix-build
6. InstallPhase avec tient un > $out
7. On relance build, et on obtient un fichier de sortie avec ce que j'ai mis sur ma sortie out (e.g. hello devoxx)
8. Avec catimg et pkgs.fetchFromGitHub
9. Il me faut une Rev, c'est quoi ? Le hash du dernier commit de mon repo (dans l'exemple ici le dernier commit du github)
10. Il faut le sha
11. nativeBuildInput (on n'a pas les mêmes entre les utilisataires à la compilation et ce qu'on déclare, sécu)
12. Puis on reexecute et on obtient notre result

[RQ]
Il existe plein de builders
Avec la même logique

[RQ]
Les releases de nix sont toutes construites sur cette même logique, en spécifiant les dépendances et en les fixant (état à un instant T referencé par le commit / sha)

"pinned"
C'est une structure avec un tar, un Sha, un URL
C'est une seule et même version du nix
C'est ça qui rend reproductible l'environnement
On récupère les mêmes packages peut importe d'où on part (peu importe la distribution et version de son OS de travail)

mvn test
eclipse

Qui se lancent dans mon shell !!!
Même version de Java, de node, de python, etc.

Les secrets internes de Spring (Carl Azoury)

Inversion du contrôle
Couplage vs interface

Trop de dépendances, complique les tests et les evo

Limiter l'implémentation en passant par l'interface

Spring permet
D'instancier pour nous
Nous détacher de la tuyauterie

e.g. ApplicationContext [RF] photos x2

Intérêt, jouer sur la définition des beans

e.g. RSS feed [RF] photos x2

2 notions importantes
- BeanFactoryPostProcessor
- BeanPostProcessor
[RF] photo

Télétravail asynchrone (Benoît Prioux)

REX sur retour personnel (chez Alan, c'est une mutuelle)

Nouvel environnement technique
Startup
100% remote

Culture d'entreprise
- culture de l'écrit
- transparence radical
- pas de réunion
- pas de manager
- responsabilité distribuée et autonome
- travail flexible

Communication asynchrone
On attend pas forcément une réponse directe

Comment dans son exp c'est facilité : par l'écrit, transparent et public pour tous, et emboarding fort

Outil: Slack (utilisé uniquement en communication asynchrone dans son exp)
Aucun message privé
Pour diffuser l'info, ne pas la bloquer, même si on s'adresse à qqn en particulier, donc n'importe qui peut répondre
Nécessite des normes/format que tout le monde adhère : format, ping des bonnes personnes
Le daily du matin est remplacé par un canal avec un formulaire pour dire si on va bien, qu'est ce qu'on, qu'est ce qu'on attend
Intérêt on n'est pas arrêté par l'attente des autres

HPFO Highlights progress fires objectives
Un résumé de la semaine, en pingant pour par exemple avoir de l'aide, on peut aussi s'abonner

Thread pour soi même mais ouvert au public
Intérêt : journal perso, donne de l'info aux autres, permet aux autres de reprendre son travail

Team retro
Canal d'humeur,

Team praise
Canal de félicitations remerciements
Pourquoi: parce qu'en asynchrone on n'a plus les échanges naturels directs plaisant, il faut donc le mettre en avant

La prise de décision
Comment sans réunion ? (Bon sachant qu'une réunion en soi c'est pas toujours voire souvent dans la culture de réunion inefficace)
Quelle solution dans son exp
Via GitHub :
On propose une décision, en l'argumentant, avec une deadlines, en pingant les collaborateurs
C'est celui qui ouvre qui va conclure, en mettant en avant la conclusion/décision et donc décider à partir des retours des autres
Nécessite toujours un format/un template
=> Toute décision devient une discussion

Retour d'XP ce n'est pas si simple, ça prend du temps à préparer, parfois plus qu'une réunion, mais le résultat est plus travaillé, car chacun a pu préparer / organiser son temps de réflexion

Pas de réunion mais des One One
Le but n'est pas d'avoir de décision, c'est plus personnel

Une décision doit être documenter, outil utilisé est Notion qui decrira une vérité du moment
Ça permet l'emboarding (ça reste lourd mais c'est centralisé)

Comment ?
Nécessite d'aller chercher les infos, les aides, etc.
Apprendre à être autonome
Falling into the pit of succes
Formattage du code
L'inter pour les [GP]
Vetif de la sécurité

Les tests unitaires
Permet aussi de vérifier un format de fichier
Respect architecturer
Remplacer les TODO [TODO] kezako

Développement de l'xp

Automatisation
Ex bot d'astreinte (qui est d'astreinte, comment faire, etc) pour ne pas à avoir besoin de se faire des visio

Assistance par l'IA
OpenAI
ChatGPT

Résumé :
Communication publique et écrite
Processus de décision asynchrone
Autonomie

Intérêt : efficacite, flexibilité, responsabilité

Attention : lié à la culture d'entreprise, demande de l'engagement personnel (dans son xp a eu besoin d'un an pour être à l'aise et épanoui) et un engagement de tout un chacun
Besoin d'initiative de lien social

[Q&A]
Qui contrôle si qqn ne joue pas le jeu ?
  Ça se régule tout seul, ça se voit, la personne n'est pas épanouie donc peut voir de lui même
  Un coach suit et peut détecter ce cas
Comment si on part d'une très grosse structure
  Baby step on ne peut pas y aller d'un coup
  Prendre des choses qui peuvent s'appliquer comme la culture écrire e.g.
Les gens transverses comment font ils ?
  Ça arrive, les gens autour sont là pour t'aider à apprendre à dire non, c'est de le culture d'entreprise et d'introspection personnel

Devoxx France 2023

Mercredi 12 avril :
Vérifier les salles dans les 48h avant

University / Hands on lab

9h30 - 12h30
1. Rendons le DDD aux devs (Neuilly 252AB)
2. De A à Z cloud Open Source (Neuilly 251)
3. De la Terre à la Lune NestJS (hol Hall RdC Paris sud 202-203)

13h30 - 16h30
1. De Java à Scala 3 (Paris 241)
2. Apprenez le Rust (hol Hall RdC Paris sud 202-203)
3. Automatisez vos tests avec Cypress (hol mezza Neuilly sud 231-232M)

Tools in action :

17h00 - 17h30
1. Env de développement avec Nix (Paris 241)
2. NodeJS (Neuilly 251)

17h45 - 18h15
1. Les secrets de Spring (Amphi bleu)
2. Playwright tests e-t-e (Neuilly 252AB)

18h30 - 19h00
1. Travail asynchrone (Amphi bleu)
2. REX d'un projet open source (Paris 242AB)