03/11/2014

Un moteur de jeu 2D en Java

Plan

- **Moteur** 
- Module Game
- Module Platform
- Module Strategy
- Module Network

Moteur

- API « bas niveau »
  - Manipulation des ressources
    - Visuelles (images, sprites, animations, parallaxe)
    - Sonores (sons et musiques)
    - Fichiers (binaires et XML)
    - Périphériques (curseur système / « in-game »)
  - Environnement graphique
    - Résolution écran (+filtrage: bilinéaire, HQ2X, HQ3X)
    - Modes de rendu (fenêtré, plein écran, applet)
    - Gestion du « frame rate »
    - Gestion des séquences (intro, menu, scene…)

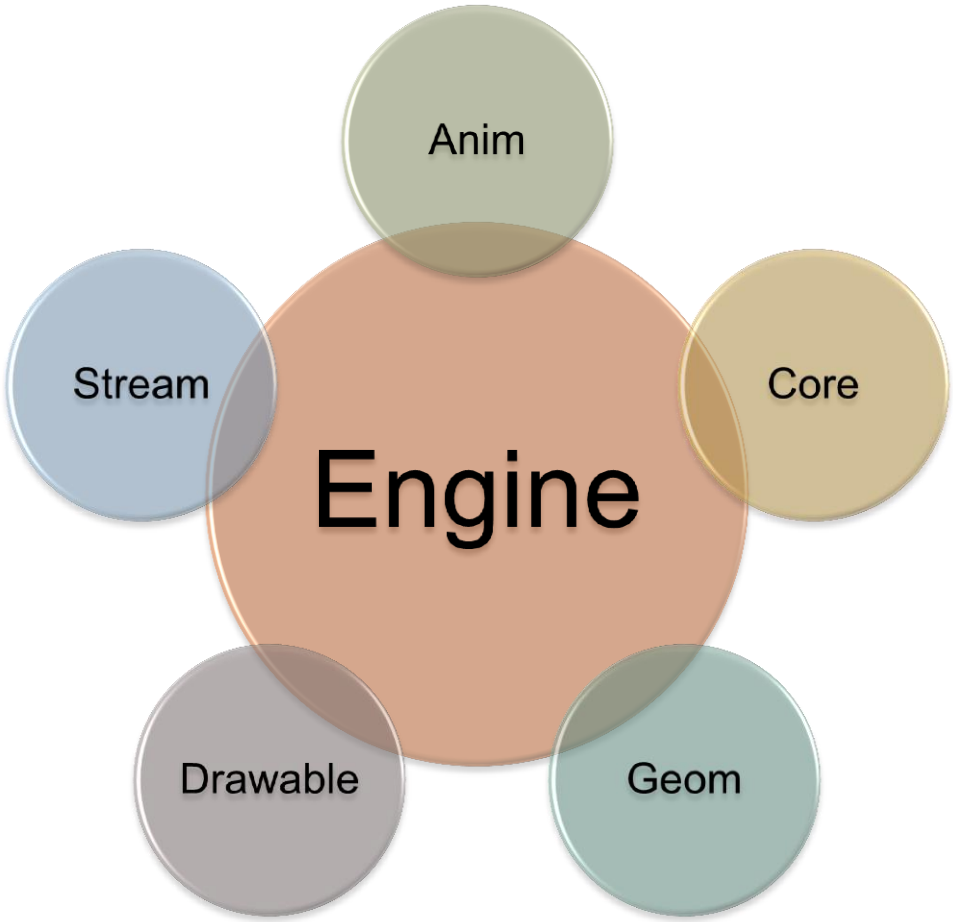
Moteur

- API « haut niveau »
  - Abstraction de premier niveau
    - Classes de base orientées jeux-vidéo généraux
    - Routines de base implémentées et redéfinissables
    - Architecture souple et modulaire
    - Outils standards
  - Abstraction de deuxième niveau
    - Classes de base dédiées à certains types de jeux-vidéo
      - Jeux de Plateforme
      - Stratégie en temps réel `(+pathfinding)`

Moteur

- Situé à partir du package: `com.b3dgs.lionengine`
- Principaux packages / classes
  - anim *(Animator, Animation, AnimState)*
  - core *(Config, Engine, Graphic, Sequence…)*
  - drawable *(Image, Sprite, SpriteTiled, SpriteFont…)*
  - stream *(FileReading, FileWriting, XmlFactory, XmlNode)*
  - geom *(Coord, Rectangle, Polygon…)*

Moteur

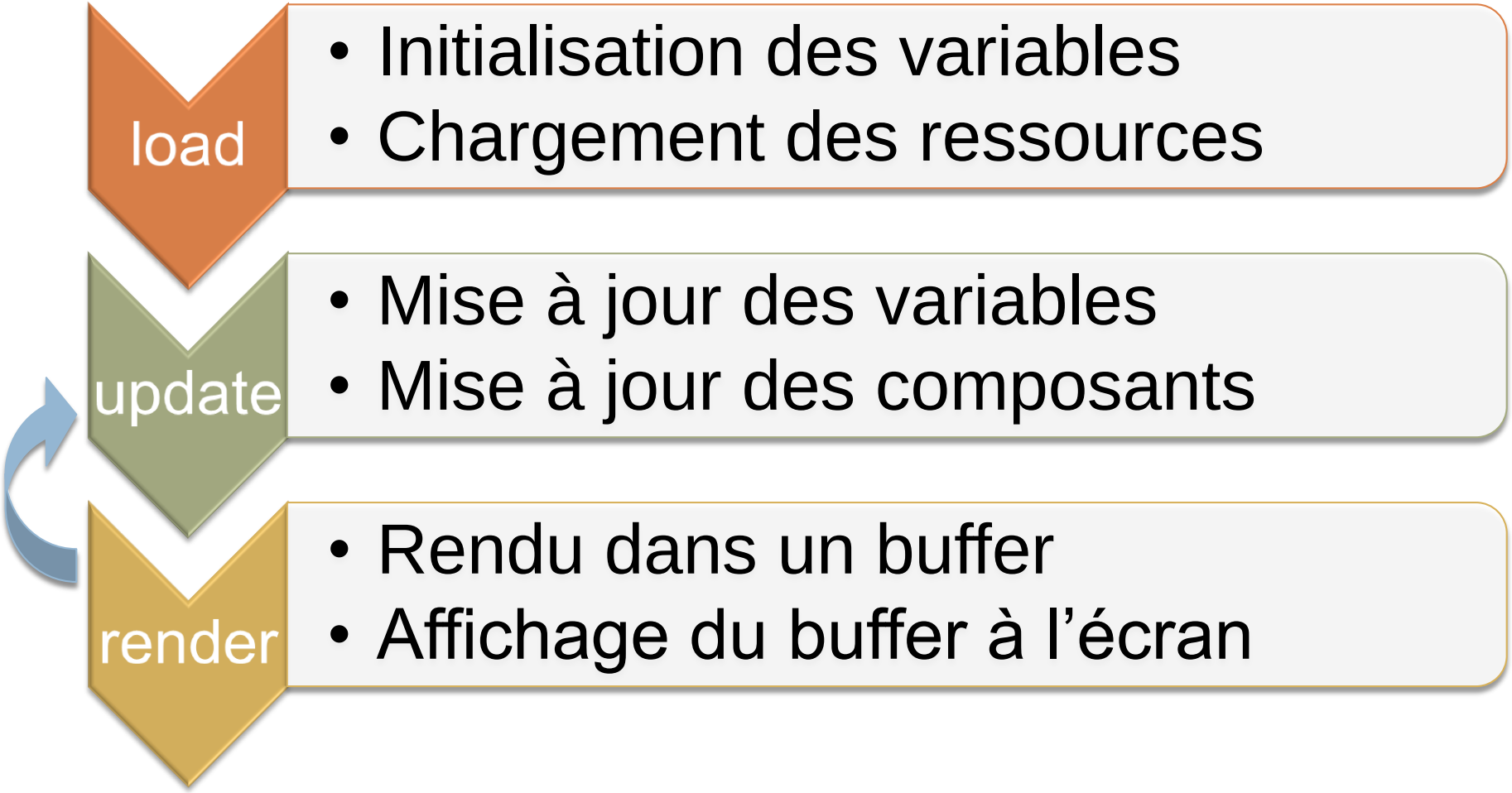


Moteur - Engine

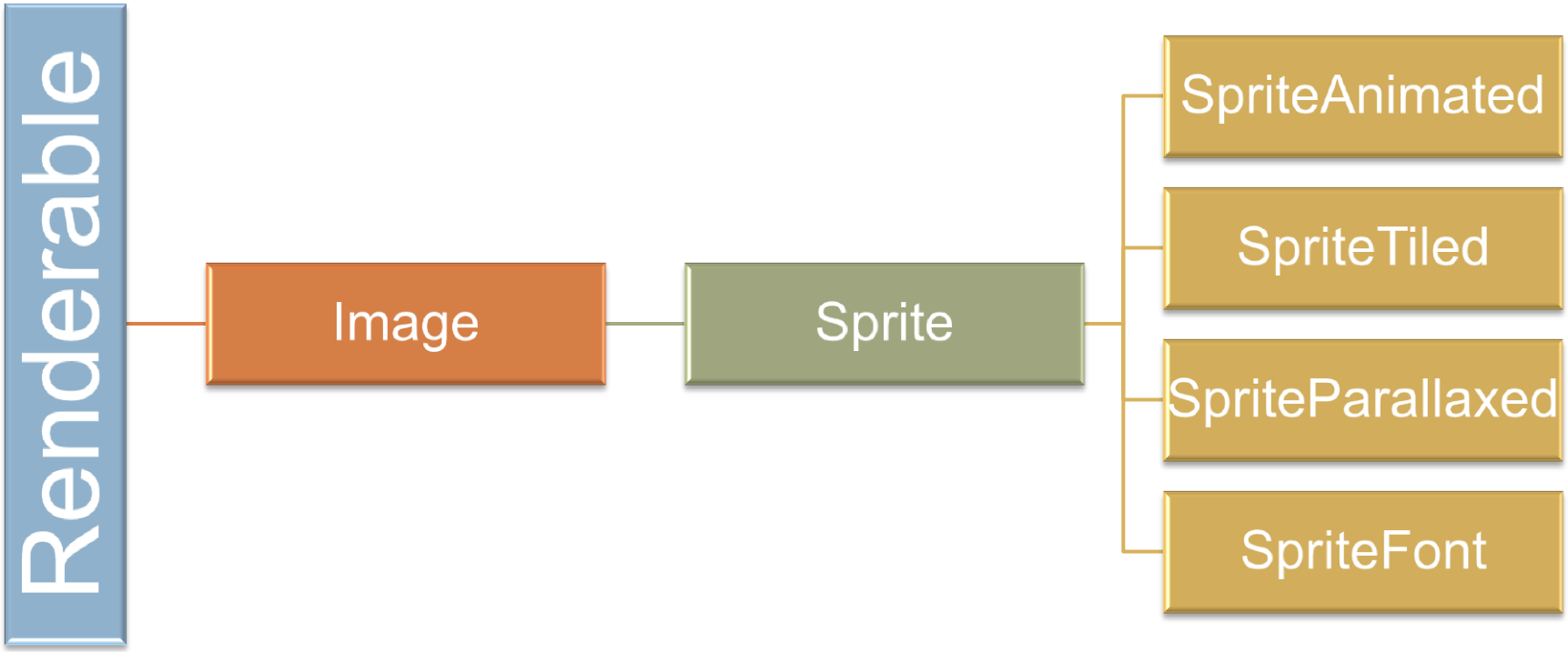
- Squelette de base
  - `load();`
  - `update(double extrp);`
  - `render(Graphic g);`
  - `onTerminate(); // Optionnel`

- Gestion du nombre d'images par seconde
- Gestion de l'extrapolation (`'machine independant'`)
- Modes d'affichage: `plein écran, fenêtré, applet`

Moteur - Engine



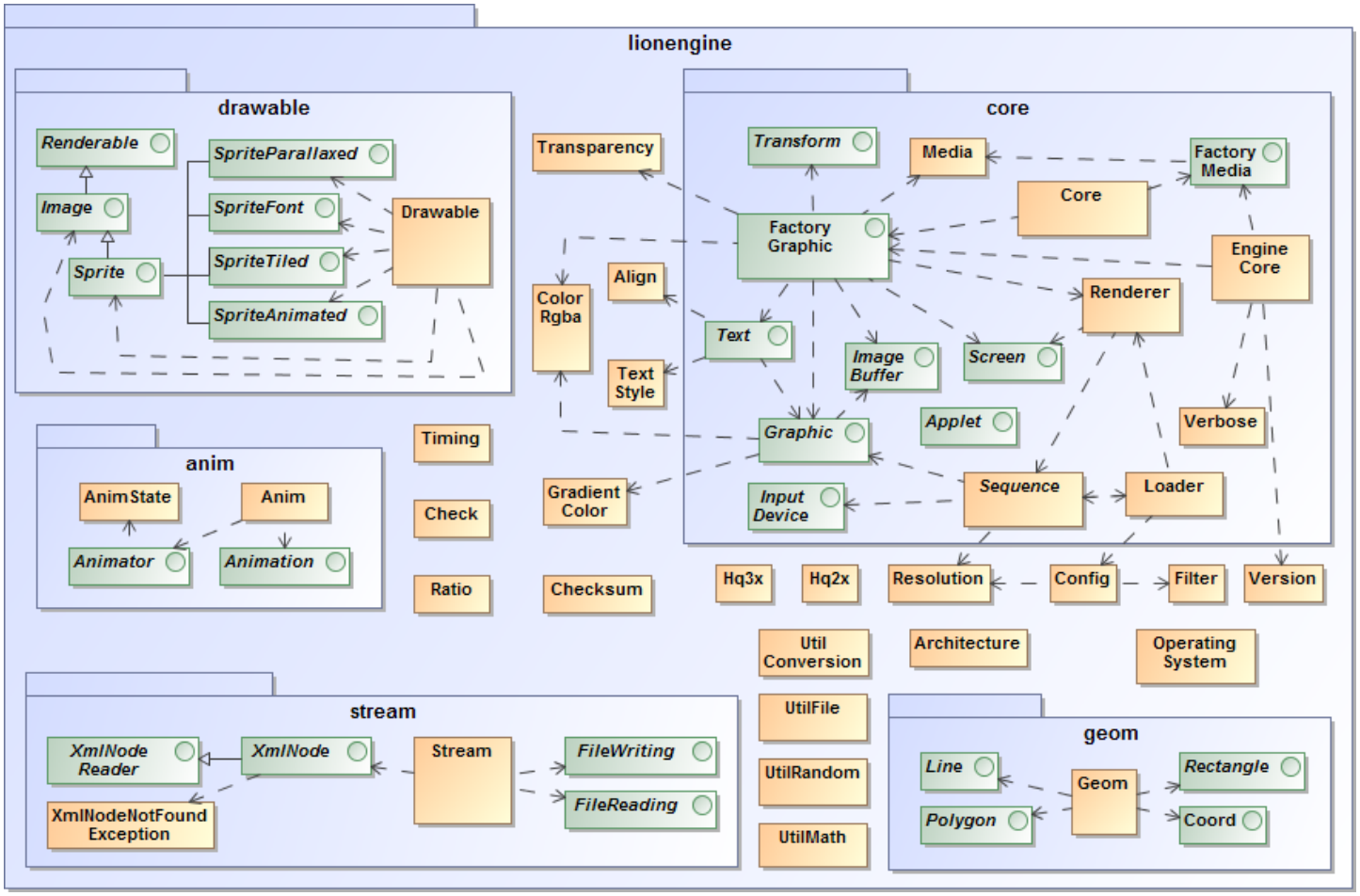
Moteur - Drawable



Moteur - Drawable

- Renderable (élément affichable simplement)
  - Image (surface non modifiable)
    - Sprite (surface modifiable)
      - SpriteAnimated (surface animée)
      - SpriteTiled (surface découpée en rectangles)
      - SpriteParallaxed (surface pour un effet 2.5D)
      - SpriteFont (police d'écriture depuis une image)

Moteur - UML


lionengine
drawable
Renderable
SpriteParallaxed
Image
SpriteFont
Drawable
Sprite
SpriteTiled
SpriteAnimated
core
Transparency
Transform
Media
Factory Media
Core
Factory Graphic
Engine Core
Align
Color Rgba
Renderer
Text
Image Buffer
Screen
Text Style
Timing
Graphic
Applet
Verbose
anim
AnimState
Anim
Gradient Color
Input Device
Sequence
Loader
Check
Animator
Animation
Hq3x
Hq2x
Resolution
Config
Filter
Version
Ratio
Checksum
Util Conversion
Architecture
Operating System
stream
UtilFile
geom
XmlNode Reader
XmlNode
Stream
FileWriting
Line
Rectangle
UtilRandom
Geom
XmlNodeNotFound Exception
FileReading
Polygon
Coord
UtilMath

Moteur - Modules

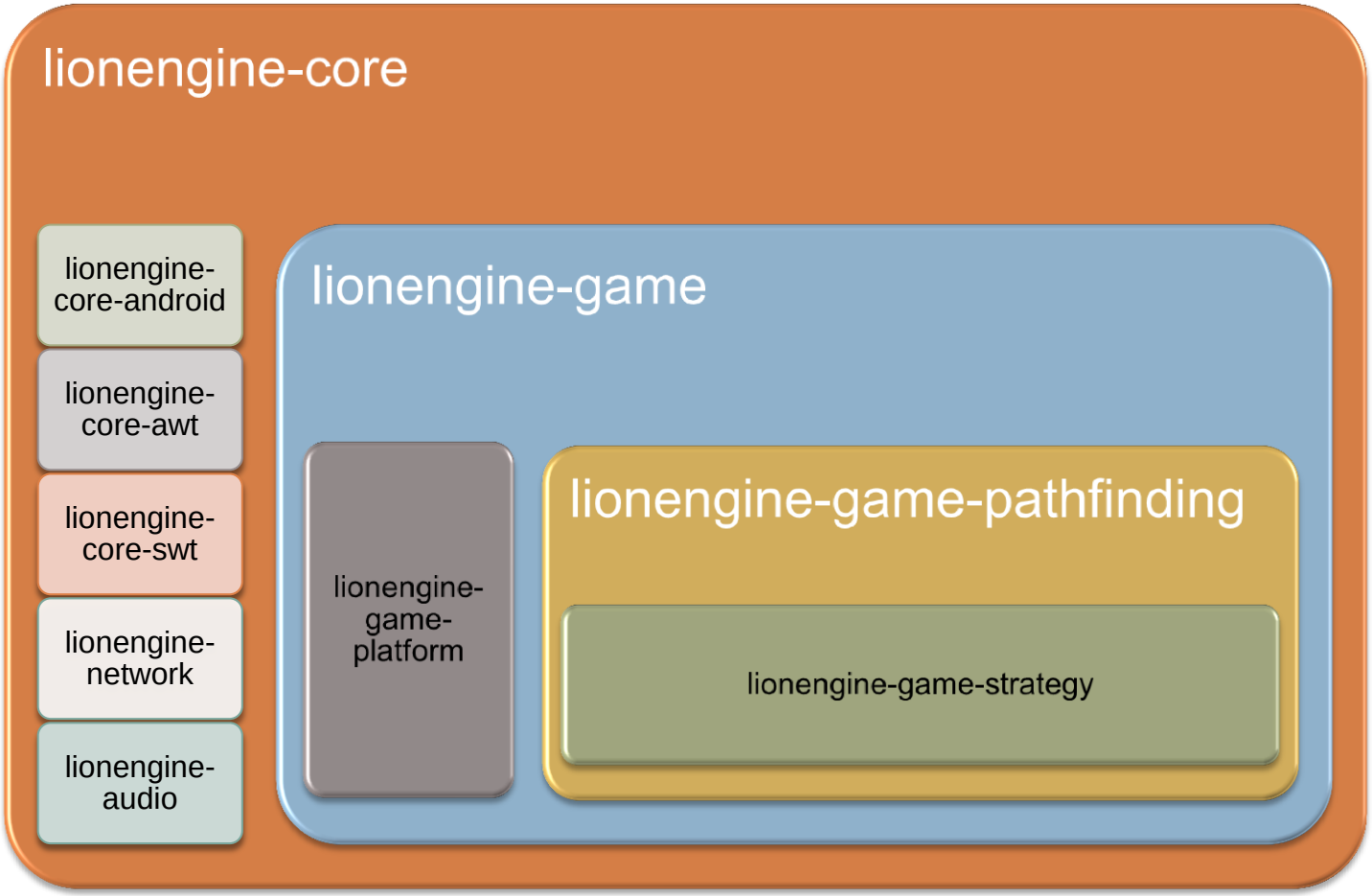
- Le moteur complet est composé:
  - D’une partie centrale
    - Anim
    - Drawable
    - Stream
    - …
  - De modules abstraits
    - Platform
    - Strategy
    - …

Moteur - Modules

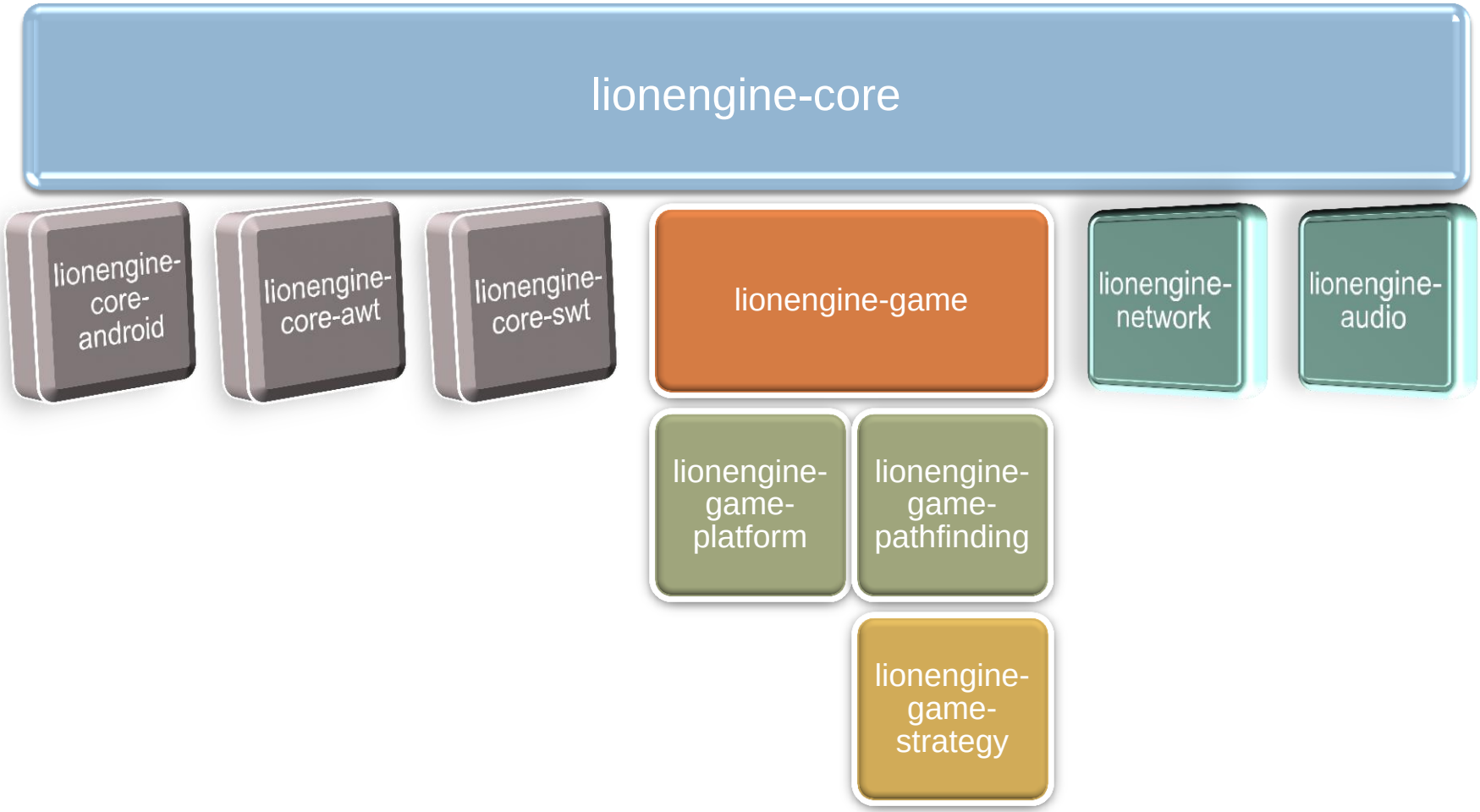
- Un module est présent sous la forme d'un JAR
  - Inclusion aisée des modules sur un projet
  - Nécessité d'inclure la partie centrale d'abord

- Chaque module
  - Dépend du module principal `(lionengine-core)`
  - Propose une base abstraite `(architecture de base)`
  - Est redéfinissable selon les besoins, en tout point
  - Est compatible avec d'autres modules
  - Respecte la même structure que la partie centrale

Moteur - Modules



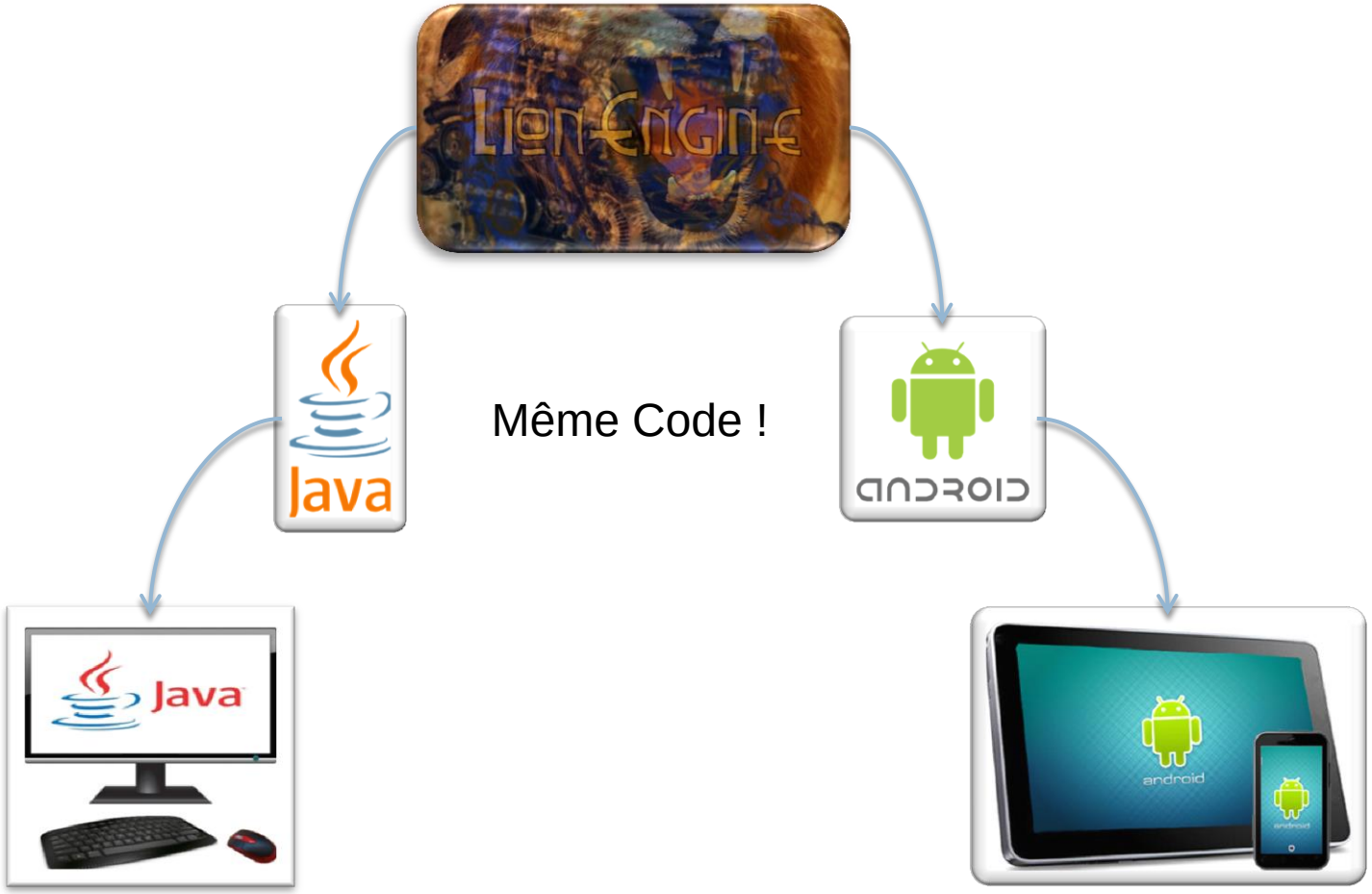
Moteur - Modules



Moteur – Android, AWT & SWT

- Le moteur ne dépend d'aucune implémentation bas niveau directe
- 3 modules au choix
  - lionengine-core-android
    - Dédié aux jeux pour smartphone et tablette, avec Android 2.3.3 au minimum
  - lionengine-core-awt
    - Dédié aux jeux sur PC, fonctionnant en fenêtré, plein écran, et applet en utilisant Java2D
  - lionengine-core-swt
    - Dédié aux jeux sur PC, fonctionnant en fenêtré et plein écran en utilisant SWT

Moteur - Utilisation



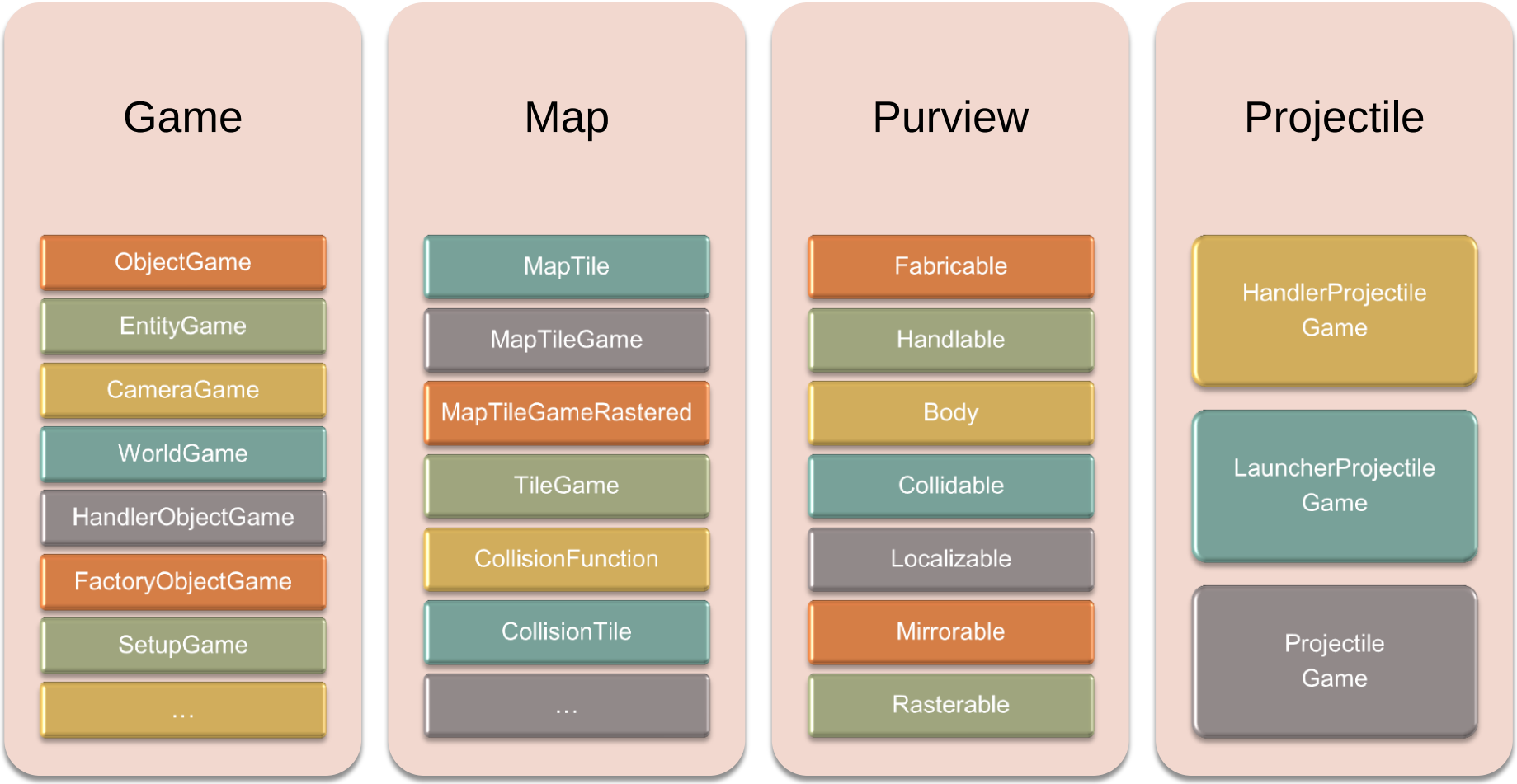
Plan

- Moteur
- **Module Game** 
- Module Platform
- Module Strategy
- Module Network

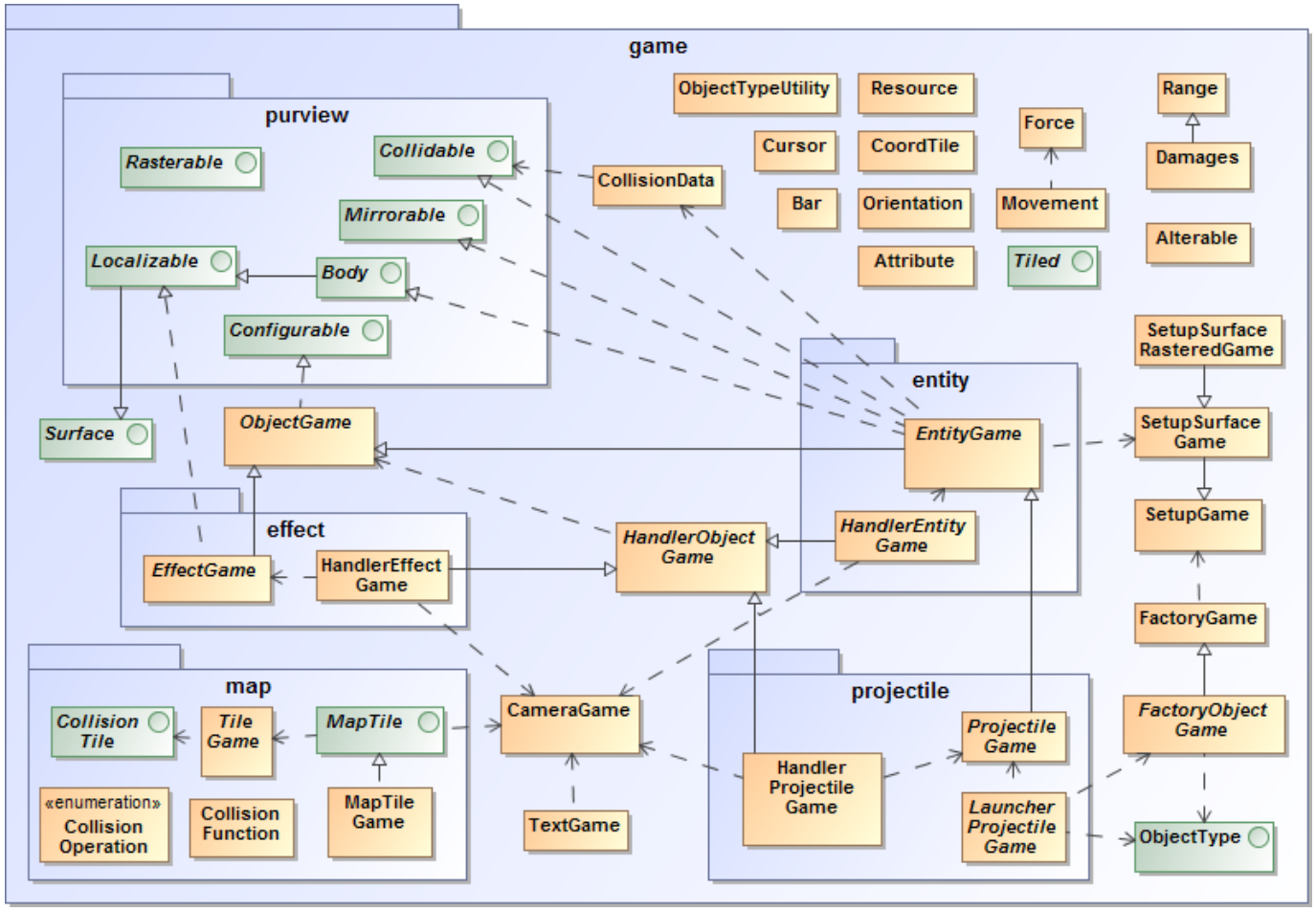
Module Game

- Principal module abstrait
  - Sert de base dans le développement d'un jeu
  - Est utilisé par les modules plus spécifiques
    - Platform
    - Pathfinding
    - Strategy
  - A besoin du moteur principal pour fonctionner

Module Game - Structure



Module Game - UML



Module Game - Game

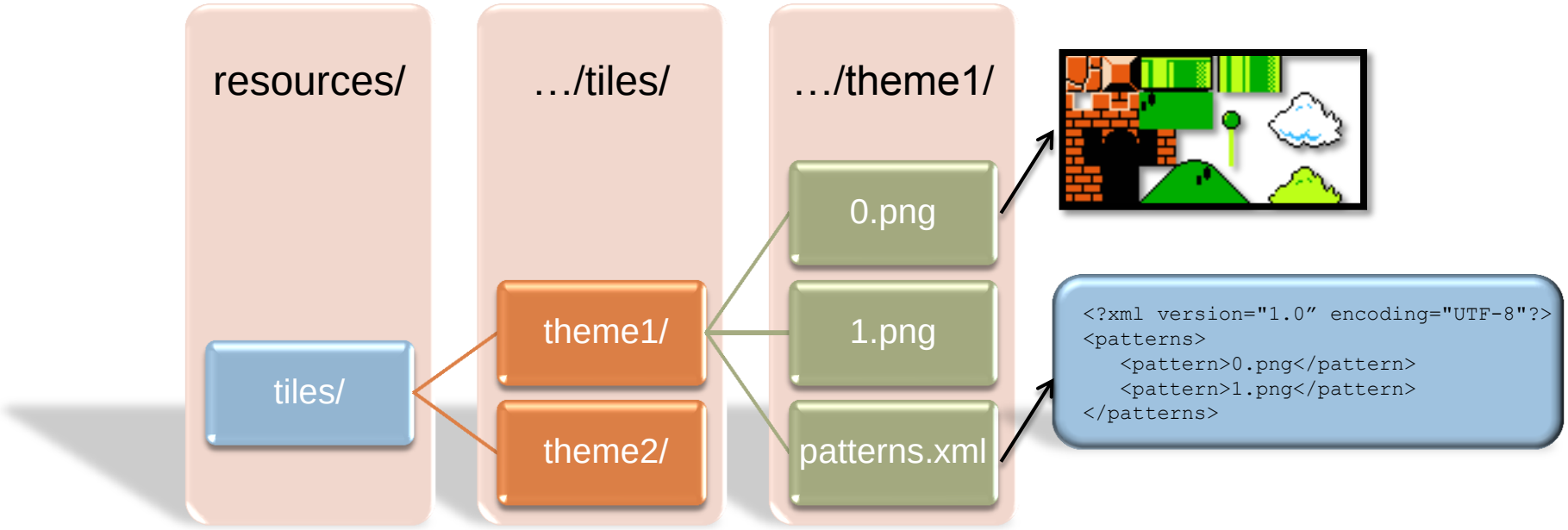
- Dans le package : `com.b3dgs.lionengine.game`
- Propose des types primaires
  - ObjectGame `(représente un objet de base)`
  - CameraGame `(vue du joueur, évoluant dans le jeu)`
  - Damages `(gestion des dégâts, aléatoires ou non)`
  - Alterable `(facilite la manipulation de quantité)`
  - FactoryObjectGame `(chargé de créer les objets)`
  - HandlerObjectGame `(gère une collection d'objet)`
  - WorldGame `(conteneur: handler, map, camera…)`
  - …

Module Game - Map

- Dans le package: `com.b3dgs.lionengine.game.map`
- Propose un type de map standard
  - MapTile (interface décrivant une map à base de tile)
    - MapTileGame (implémentation abstraite de base)
      - Chargement des tiles dans une image (SpriteTiled)
      - Import & export au format binaire
      - Associe les collisions aux tiles à partir d'un fichier externe
      - Génération d'une minimap représentant la map en pixel
    - TileGame (structure de base d'un tile)
      - Numéro de pattern (=N°image d'une tuile)
      - Numéro de tile (=N°tile dans la tuile)
      - Nom de la collision associée
      - Coordonnées sur la map (x, y)

Module Game - Map

- L'architecture impose une structure de rangement
  - Un dossier, placé dans le dossier des ressources, doit contenir un dossier par thème, eux même contenant la liste des tilesheets avec le fichier '`patterns.xml`'
  - Sans ce fichier, toutes les images de type .png seront chargées



Module Game - Map

- La gestion des collisions est effectuée
  - En amont côté Tile
    - Récupère son type de collision depuis un fichier externe
    - Définit les points de collision en fonction d'une localisation
    - Dépend du type de collision du tile
  - En aval côté Entité
    - Teste quel est le premier tile traversé ayant une collision
    - Se place sur le point de collision du tile correspondant
    - Gère plusieurs collisions (sol, zones bloquantes…)

Module Game - Map

- Recherche le premier tile ayant une collision suite à une intersection

- Récupère le tile concerné, et recherche le point de collision associé

- L'entité se place sur le point de collision

Module Game - Map

- `getCollisionTile` permet de récupérer le tile '*raycasté*'
  - Référentiel de l'entité (modulo un offset ajustable)
  - Possibilité de filtrer le type de collision recherché
- Récupération du point de collision du tile
  - L'appliquer à l'entité si il en existe un
- L'entité a récupéré le tile touché, et sa collision
  - Le processus est maintenant complet

Module Game - Purview

- Dans le package: `com.b3dgs.lionengine.game.purview`

- Décrit des capacités supportées par des « Entity »
  - Collidable (gérant l'aspect collision de deux objets)
  - Mirrorable (permet d'avoir son symétrique vertical)
  - Localizable (permet de gérer le placement d'un objet)
  - Rasterable (permet de gérer l'effet raster bar)
  - Body (représente un objet soumis à la gravité)

Module Game - Utility

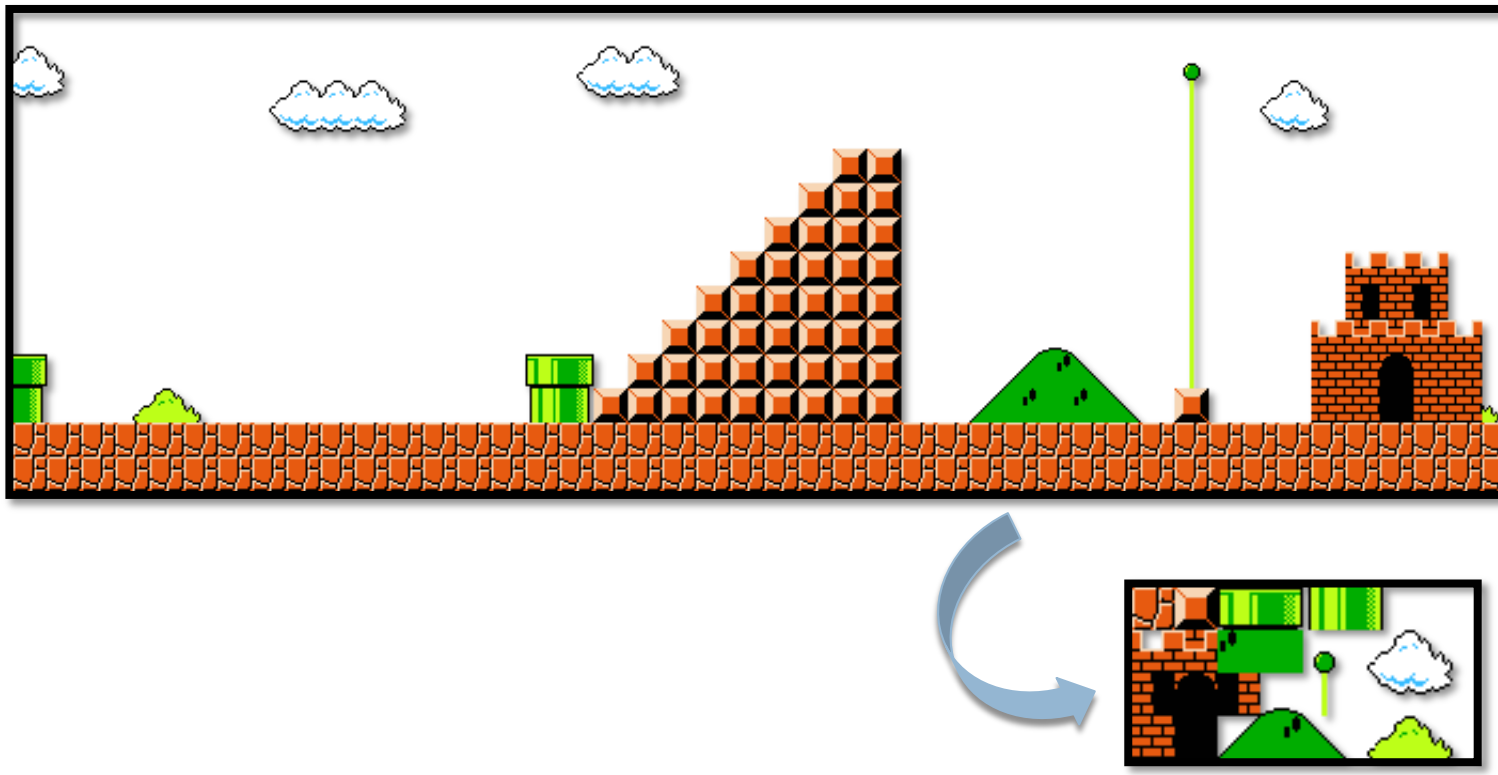
- Dans le package: `com.b3dgs.lionengine.game.utility`

- Conversion d'un « levelrip », en un format de données compatibles MapTile
  - Charge un levelrip (image représentant un niveau entier)
  - Convertit au format MapTile
  - Sauvegarde au format binaire
  - Supporte le multithreading (meilleurs performances)

- Extracteur de tile à partir d'un levelrip
  - Découpe les tiles uniques d'un levelrip et les sauvegarde dans une image en tilesheet

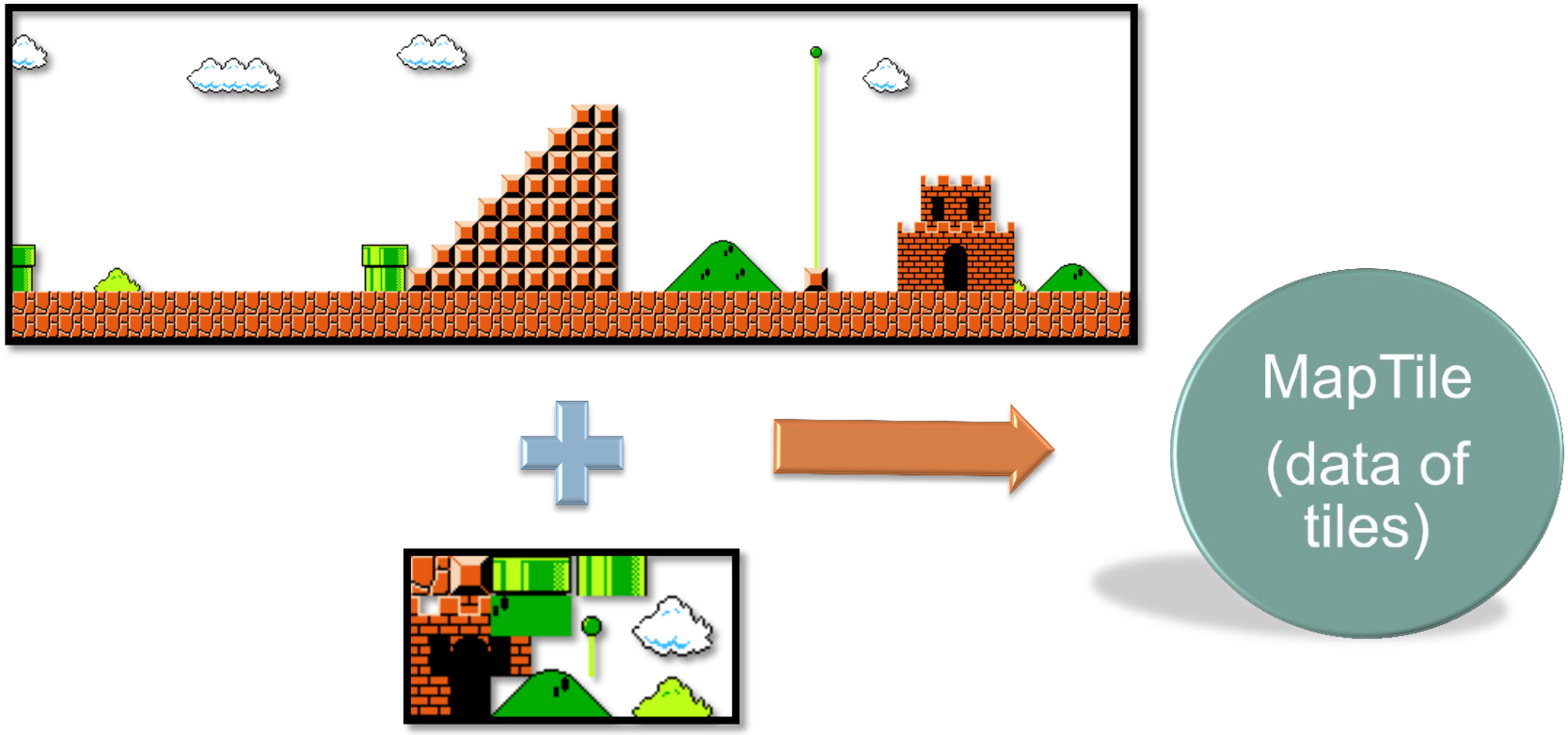
Module Game - Utility

- '*TileExtractor*' en action:



Module Game - Utility

- '*LevelRipConverter*' en action:



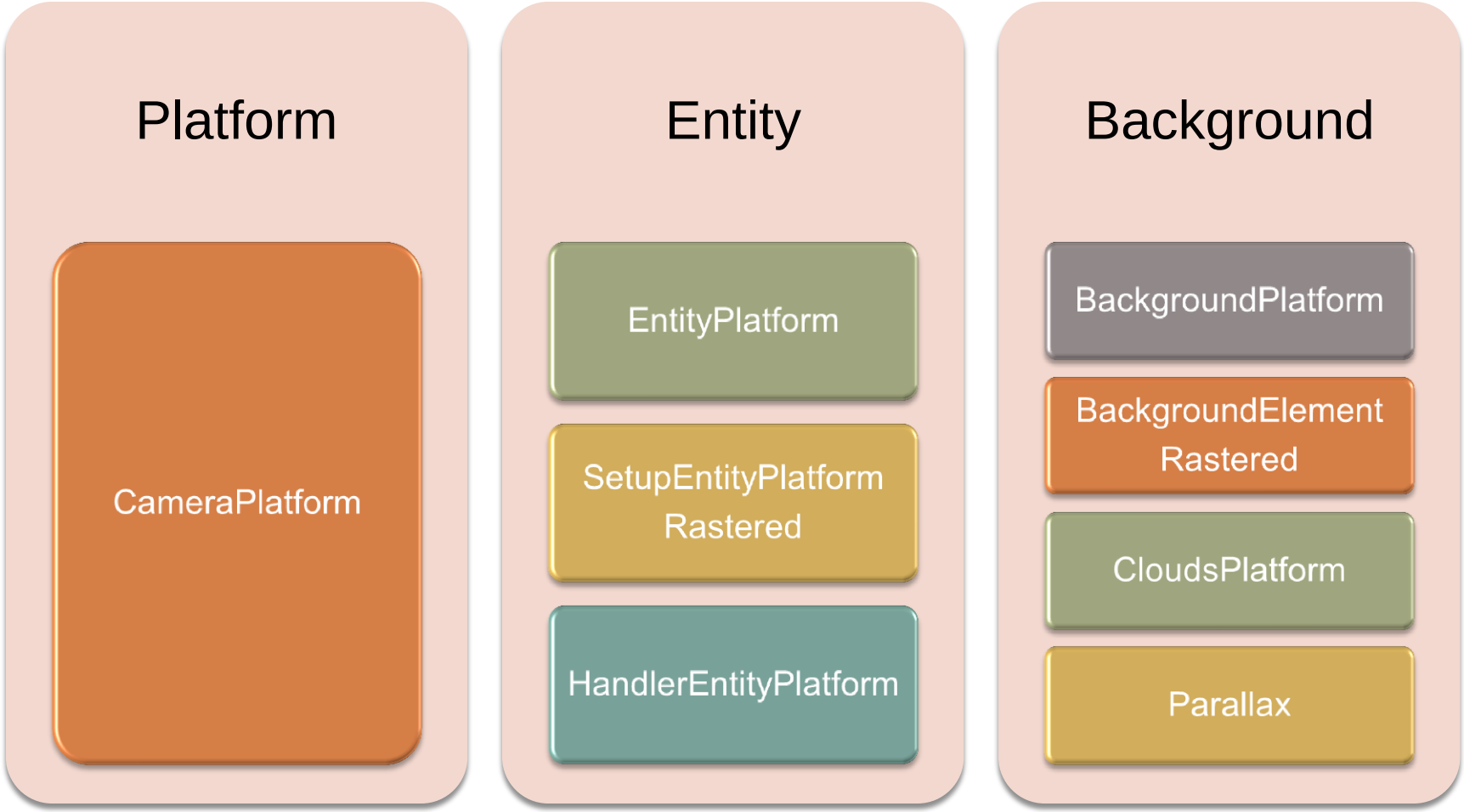
Plan

- Moteur
- Module Game
- **Module Platform** 
- Module Strategy
- Module Network

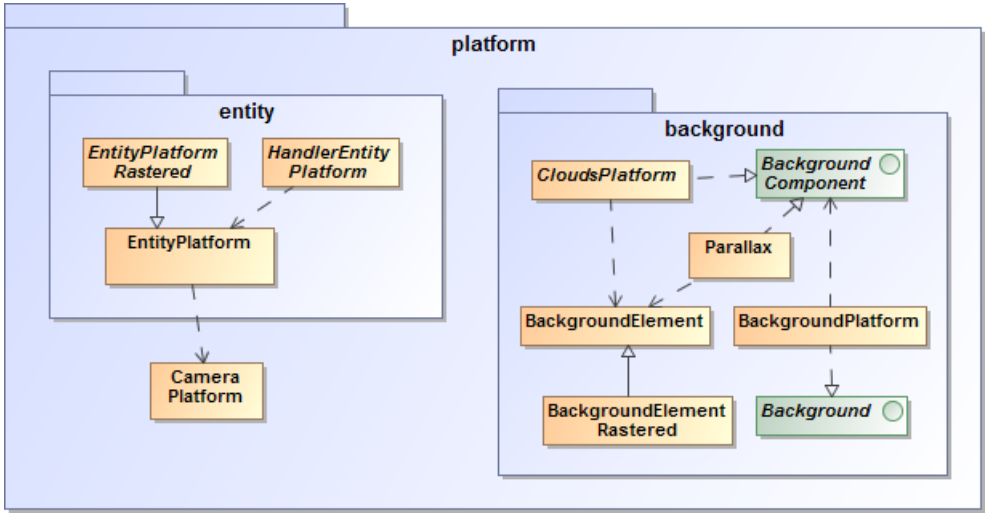
Module Platform

- Module dédié aux jeux de plateforme
  - Propose des types de base
    - EntityPlatform
    - CameraPlatform
    - HandlerEntityPlatform
  - Contient un package complet dédié au background
    - Background, BackgroundRastered, Clouds, Parallax

Module Platform - Structure



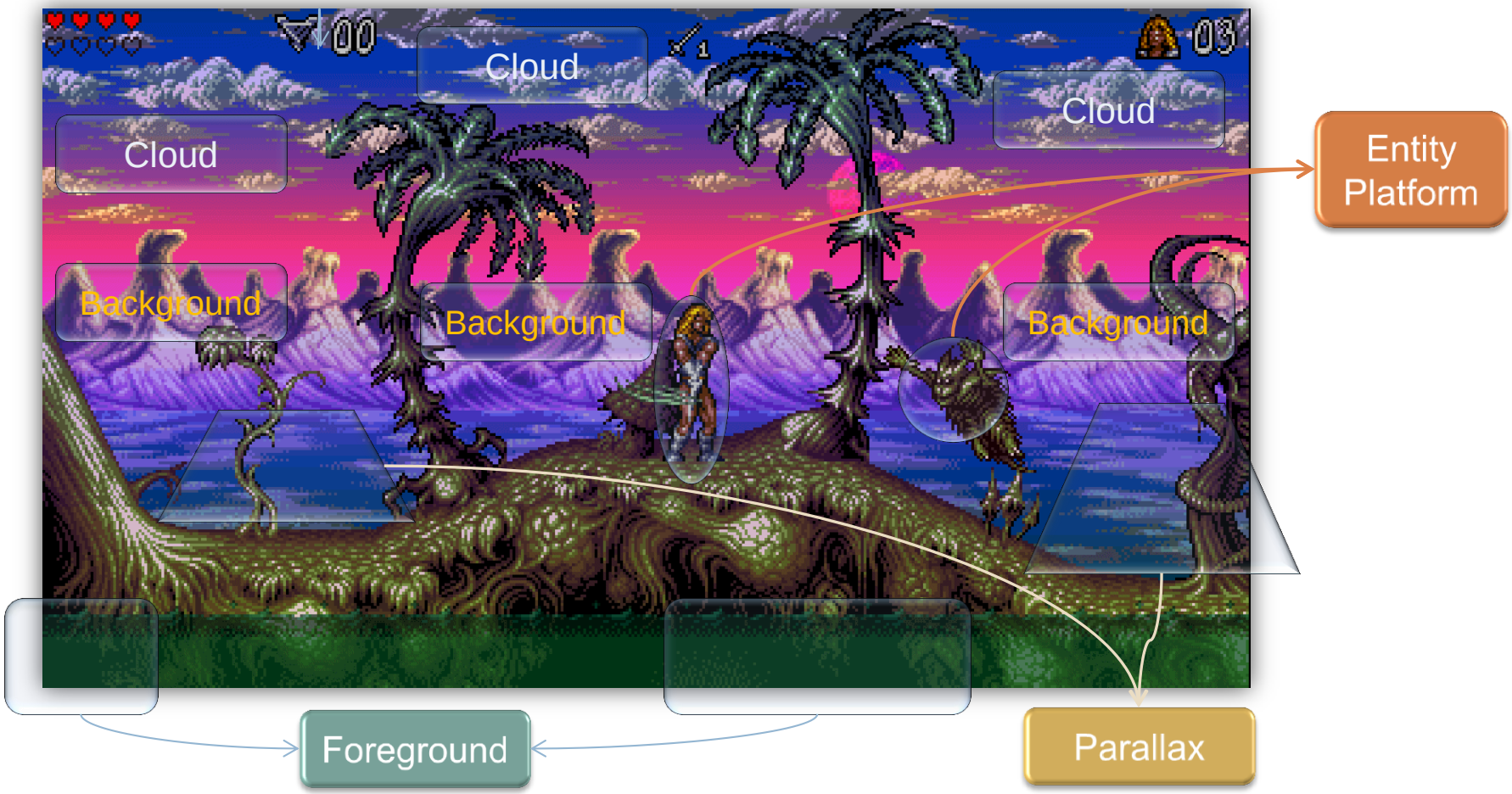
Module Platform - UML



Module Platform - Package

- Dans le package: `com.b3dgs.lionengine.game.platform`
- Propose des types spécialisés « jeu de plateforme »
  - EntityPlatform `(entité spécialisée)`
  - SetupEntityPlatformRastered `(partage des données)`
  - CameraPlatform `(caméra spécialisé)`
  - HandlerEntityPlatform `(handler spécialisé)`
  - BackgroundPlatform `(background orienté scrolling)`
  - Parallax `(effet 2.5D, basé sur une image)`

Module Platform - Exemple



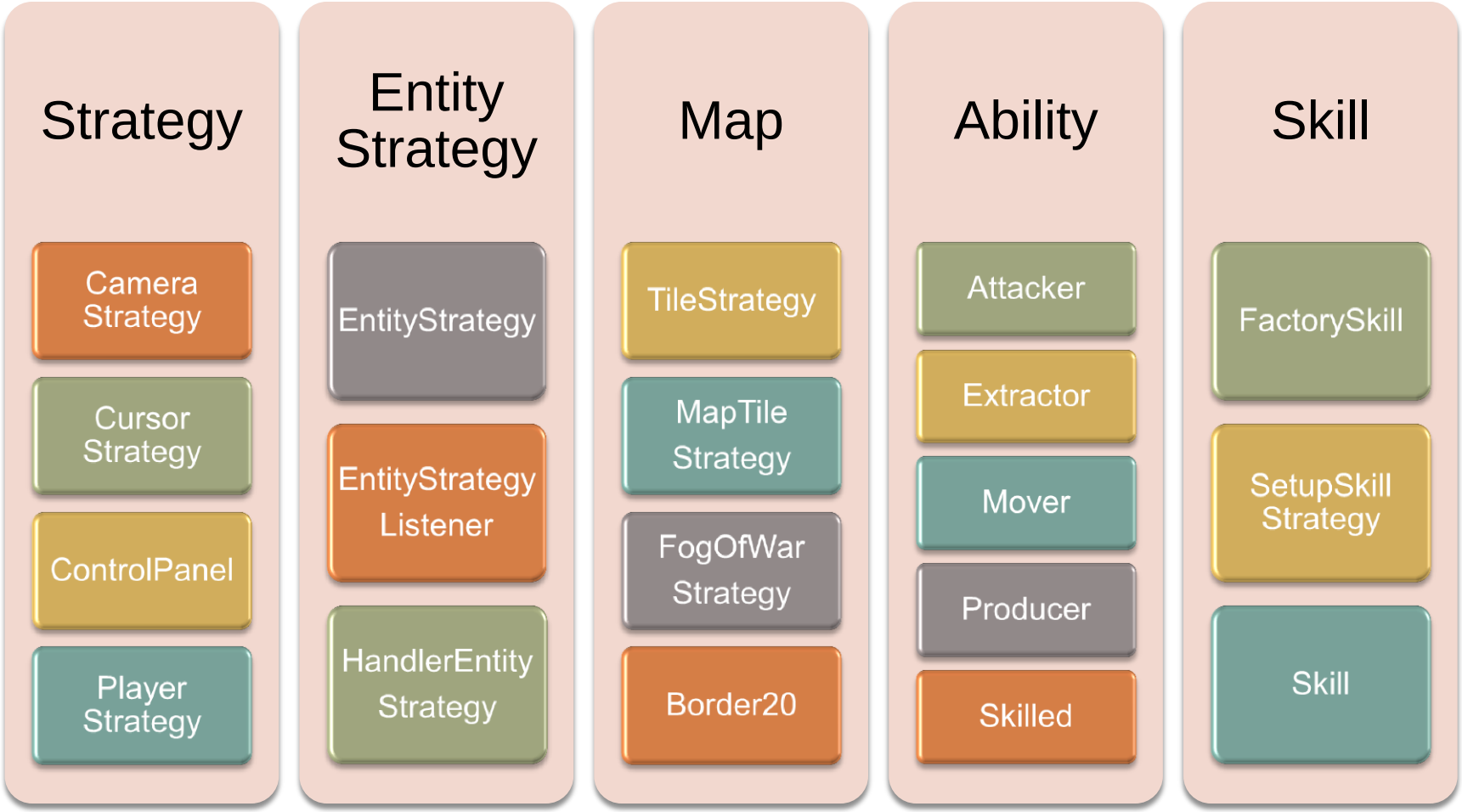
Plan

- Moteur
- Module Game
- Module Platform
- **Module Strategy** 
- Module Network

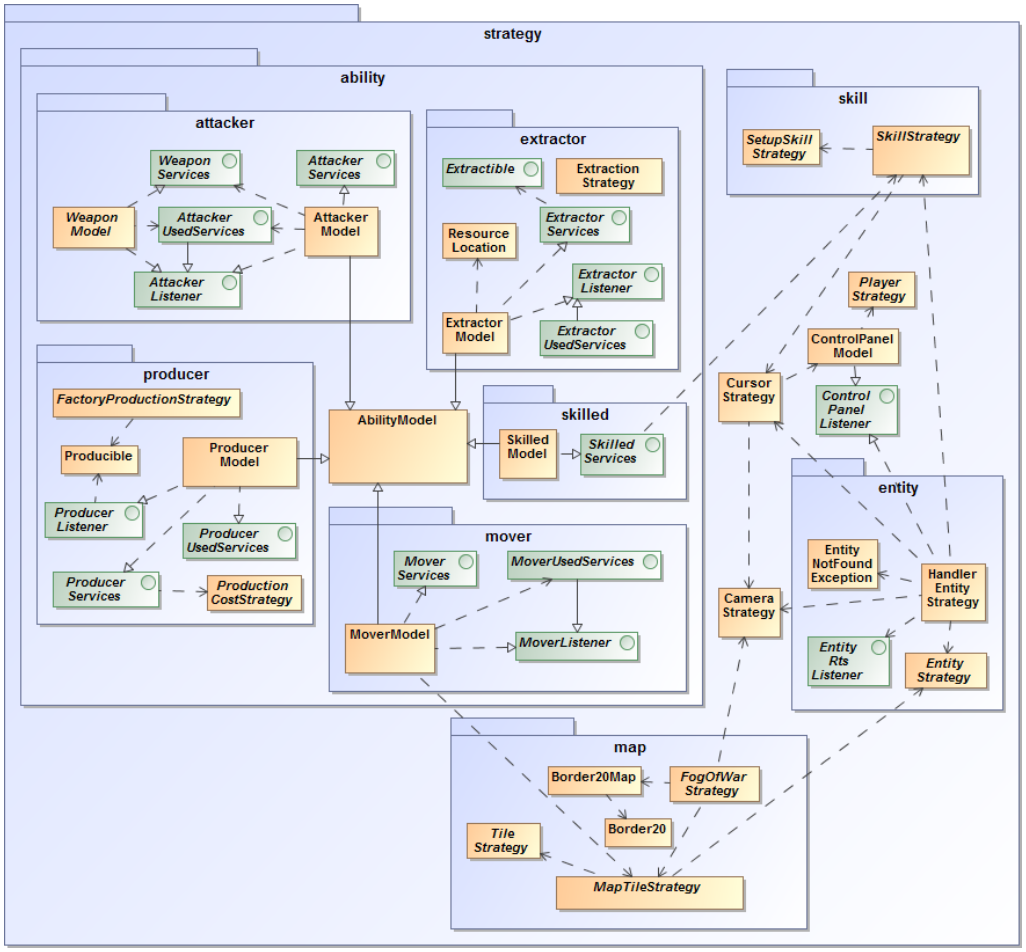
Module Strategy

- Module dédié aux jeux de stratégie
  - Propose des types de base
    - EntityStrategy
    - CameraStrategy
    - CursorStrategy
    - SkillStrategy
  - Permet une gestion plus avancée des maps
    - MapTileStrategy
  - Contient un package complet dédié au pathfinding

Module Strategy - Structure



Module Strategy - UML



Module Strategy - Package

- Dans le package: `com.b3dgs.lionengine.game.strategy`
- Propose des types spécialisés « jeu de stratégie »
  - EntityStrategy `(entité spécialisé)`
  - CameraStrategy `(caméra spécialisé)`
  - ControlPanel `(panneau de contrôle)`
  - HandlerEntityStrategy `(handler spécialisé)`
  - SkillStrategy `(base abstraite d'une compétence)`
  - PlayerStrategy `(représentation de base d'un joueur)`

Module Strategy - Exemple

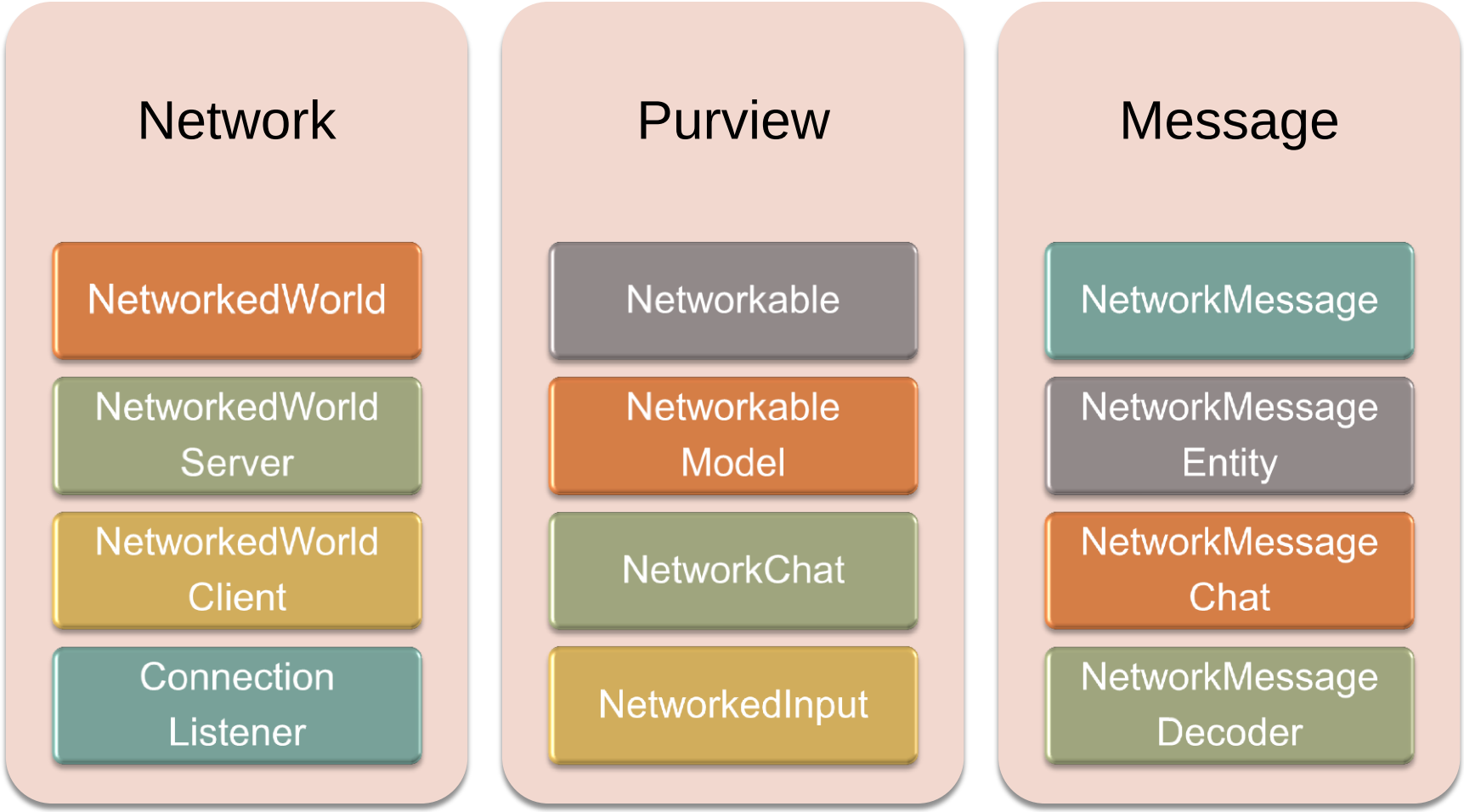
Plan

- Moteur
- Module Game
- Module Platform
- Module Strategy
- **Module Network** 

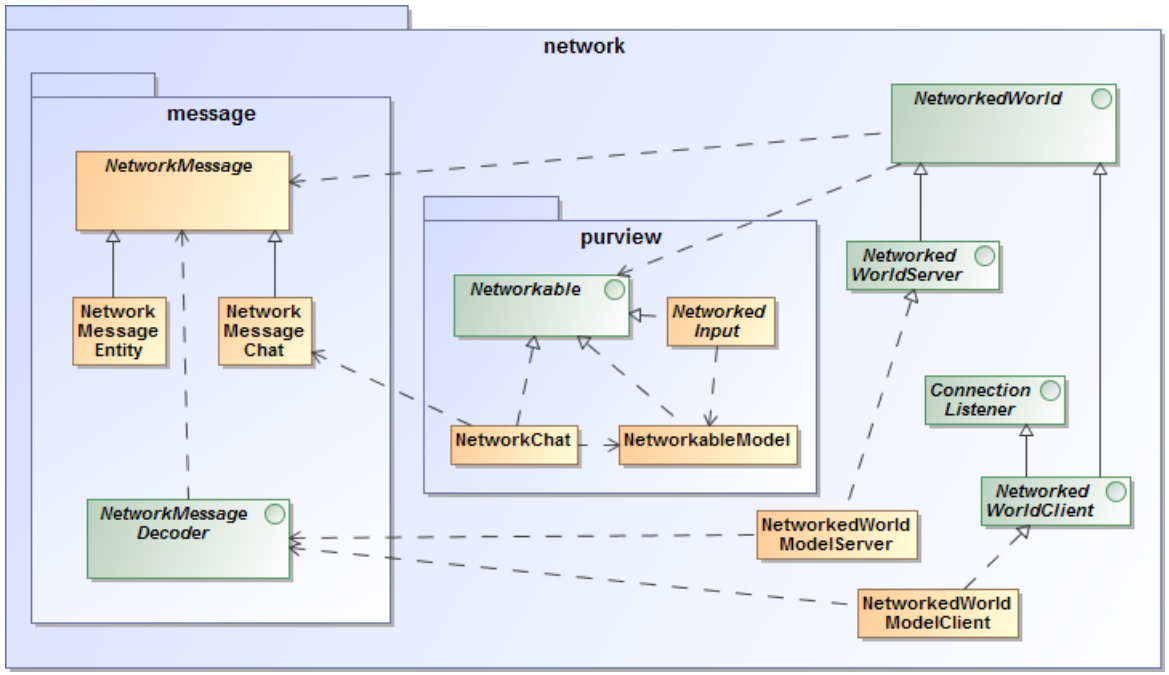
Module Network

- Module dédié aux jeux en réseau
  - Propose des types de base
    - Networkable
    - NetworkMessage
    - NetworkMessageDecoder
    - NetworkedWorldServer
    - NetworkedWorldClient
    - NetworkChat

Module Network - Structure



Module Network - UML



Module Network - Package

- Dans le package: `com.b3dgs.lionengine.network`
- Propose des types spécialisés « network »
  - NetworkMessage (`message standard`)
  - NetworkMessageDecoder (`décode un message`)
  - Networkable (`support pour le réseau`)
  - NetworkedWorldServer (`monde côté serveur`)
  - NetworkedWorldClient (`monde côté client`)
  - NetworkChat (`chat standard`)

Module Network - Exemple