[http://www.plantuml.com/plantuml/svg/lLbHR-Cu3dxFhn3Q5nUR5dkuXmK6Wy6cQThaa3H1aaudjnWiL4UDlUDOFab96yprlvz5MQ559oskLzoBRNsaA9AYA8d-NHkkp6QTjGeU\_-0hmKw-fkBfXFriiJZZMZFRl9SgMrg8\_S9cXKgDkEGcJfX-06YiJ29NYXT96dVvMYWMsoUtHEmVtPGWMvMFisE4DajpmpAn5PdbEMTRcIvPh0Gt8e9U3b4YNmeLFMQI6mSx3LV2m8VbVQaRCDyOwVKtrafjd--r1yHHSvgA8QmJ-TJGQIJopSVf1EB-9vskqqnq4ocrK1-ZLbtY-pHxQM5OhkLIP12NUoXEkEAn4GfWnX-qiUtAdgwd0f6TGcmb1NvpOqf9s2wa\_XQq5jXQfah49fMvWnwap0JFcKcKtAoIUvvbGAXSn3IyF8HUUx1F2rUcBXL\_YaAtdjN8QxaLqtILc8YYG\_7O1EUoQ62NqXYv3c2pKNaK65MdBeLTRrhq2yV3rtmbwWnQc6z0UYRe0j1TXThlbJ2JqgbH0rfuo3iorNDq9U7l5mJemCG5DgLvfVK4b4W4pqnIPtYoFex-eMNE\_jAXnMN79LBQp7v-vTb6GFkA6nuEV-NT1XgKCSeiHQX77WisAds-5hdnGKVd4L4oZwTq7X6ZytZwsZo2Fcu5Z6Ik8VxHs2hekpBBKcrDS8dpoJxF6cop4b--pj9HIZvHbgOEsYlXsz4mrUPprVh2rcU4NBK3BkjyP0Z7kKcatZlzfpRSfF4BpDXiCXtCUxF1FtjE972w06upZ5XF878d6T-X2mF1tScySpiOTUQ3yMr0fkFU3sxwyu3ZE-7eUuxpGnppyUGexN8yduz7Hyd3tlLnkLCS5TpXrXK2knhWLsezCWuHBkKJ2iWtwmUX84J0\_CgRrFb7Lt5nasv5i5OrMNuV7zUGavFK74\_BQ2bPVSGIo0DWF0sgkIhrf3550zLnKLyXmSjygUs86N\_OgP8rERmZ2ZygpfgMMRh4VUuPFtVLzcLAWIjcTeLm0ARiJcdsbXi1Me5qeYtYrVTFhz5BvSbA0fsBD\_T7hzPAc7cf6TaWw8wpY4WCUStfDYJ-jU6PZiOFVudOC5cJad2TTELIL6xtCzbR5sRdPn8Qn0L8Q6vvC-jx3x\_g3nnmKZxiLF\_4LSQLIhVY6mKM8DTtYFm9fxq\_tRJhlUkDnLdGzNN5SLqARTASmyeaFdZLFhAtEtMYG-Y2E7-kU86Ey6SLw0YqG8rsZUKCUTuzAqZ7a-PGShKKwa3KKiK5rnibVE3uTjDlIFi8HTyEhtkEYSb-4uqG\_v1W7l5L6llXe476Gy97cF\_cl8nssWYrgyTKB7CZydt204AupnSsuS8ell-1jNOMChpB6BfKkawrwByoyKfAVvg1R-9mnFyV\_XxwxQhI2878tkt--Fx7TsRumn0A0fIZjhs0D-ZsG-Ytuhd9qsRxqyGrZurKksFRUVgsdt4yKBpIIRzxZy6hETPVph52WYsk3RY8xI4bvHeQyyJURfZC1JIz3l56AP6xNjUbSJf1ifjilatpLG0-Y5MQOx65tkW9sFO7G1YZlTSAx\_bEO5G3Chx9uyIfNCD17hQFE3-uzVQ5aj7ngINSrPqYFCqvbKKcAfNTpe47s0fmH5Fo5LAPW-TRl8EcUSLG0-D4n3-6KcfHf-5CyTnUAIEiPiIl1oqUisq4eipNRlyu0rQFY0e1RmBundtibth13YQ-2K-bN5Fqa1IumzVycHGDKQJNH5g-npHVmdaB5djzfcl9EQxXkNWsFkDXSaN4UuOl6srRu6AFDLJphNCVvfthL6dJs4hTYquRzJZA9H0rn-q4OQZXpQEMDQtmVQCkJcnbjYKUHV\_WfDLoHBUHH48NDW9WBVBDL400KfyTzZJDHnMilMoi5XFsIv9QR1U71TaiLDUdKT8FyI3PrfcCB31hlRHgvI1tWwuA2AV7hghrpVFqpOlgwUlr5z20xA\_3qgU-h4jH6YBLui8egIToa4uBh2ZzxGnHIT\_t3QMJPL30L-LKMmrs0yCcuqORRNGQitNnzvdSv4kI8C9D8zK59ZVwlm25MH24bM2z\_cpNcBrt2uK\_Hy1OaLUsdpUreH3C5k3uMsNhlm2\_TF49QHfInqG9hK4zFrYqxnk6\_G-nkv9FaDmRXBlY86plpaiuQuKa8pTnueMrMV0um0dI3\_CvIOglLkL0n-M0ThJaRqklrGvNy\_bSF9VE4HM1VVxCebdxzFhwjzz-\_VMCpT9ybGaZS\_RbIpKhVQwJ4Iy-kv91k-h\_XQLs\_mRNKfahOGQvDZoFNRoS4uQeiis1m6vbHMw4AbnRJ9HntToh8N3n2W75lJhWAdeX1cMy087Q7P5qDxsbqW2a4Ztkf5BKIGOJQ1evOaA\_rrJcVd0rxnEeCno6oCrqV7Tx5MBNWsalHAvwmtadX2QTsQnpG\_X6dPj1btGTZwU4QJ6RzwQB0ptxlSwGgZmPTXOz8k6cFvvHjqt7rBJRopzl1rzxdRjln5UJFubGptvOCmVEfvt9GKAtGxpb2PtfT7mVKW2wlHjTzgRLIgCP\_Fpyv6ydFlz-OebGyX3Fh-sBYtP-uehrmBckSdmBqnh8WbJz2GxmhVr\_MAGm9BdVeKrI\_Qzaa-fJOaN2qJzXMaKBsCN5lx\_i\_z2qoA872GJxnEH6-LJGylQ26B-pEnHKAD7zM2ttGm90uF0GwUsSJJj3XHSx8nHsXdDkrLTaMSiN1n614XaszuMdS\_WTjSUXx8DihYw3RLyNGGJA2DZ4myEv\_QDLXiZ5HNu2jDA7SVcoCzN2o21HXXJNiHbPzRxa16FPiX0vp1sW9AwjcY7Ied7Sie-gxIFMpxv-LVVVHRxShBDMg\_KV](about:blank)

@startuml

package "View" {

class ViewWorld {

+ SpriteBatch sb

+ OrthographicCamer camera

+ ViewWorld(Game game)

+ ViewWorld(Game game, int level)

- void create()

+ void render(float)

+ World getWorld()

}

class ViewTuto {

+ Stage stage

+ SpriteBatch sb

+ OrthographicCamer camera

+ ViewTuto(Game game)

+ void render(float)

+ void show()

}

class ViewMenu {

+ Stage stage

+ SpriteBatch sb

+ OrthographicCamer camera

+ ViewMenu(Game game)

+ void render(float)

+ void show()

}

class FileChooser {

+ Stage stage

+ SpriteBatch sb

+ OrthographicCamer camera

+ FileChooser(Game game)

+ void render(float)

+ void show()

}

}

package "model" {

package "character" {

abstract class Character {

# int hp

# int attack

# int posX

# int posY

# int direction

# boolean throughWall

# Sprite sprite

# SpriteBatch batch

+ void draw(SpriteBatch sb)

+ void moveRight()

+ void moveLeft()

+ void moveTop()

+ void moveBottom()

+ void turn(int direction)

+ void decreaseHp(int damage)

+ void setX(int x)

+ void setY(int y)

+ int getPosX()

+ int getPosY()

+ boolean isThroughWall()

+ int getHp()

+ int getAttack()

+ void increaseHP(int health)

+ void write(Json json)

+ void read(Json json, JsonValue jsonData)

+ int getDirection()

}

interface Movement {

# void moveRight()

# void moveLeft()

# void moveTop()

# void moveBottom()

# void turn(int direction)

}

class Monster {

+ void attackCollision(World world)

+ void attackSword(World world)

+ void attackArrow(World world)

}

class Monsters {

+ Monsters(List<Monster> m)

+ Monsters()

+ List<Monster> getMonsters()

}

class Ghost {

# {static} int SPRITESIZE

+ Ghost(int x, int y)

+ Ghost()

}

class Player {

+ {static} int CPTANIMATION

+ {static} int SPRITESIZEWIGHT

+ {static] int SPRITESIZEHIGHT

- {static] int SPRITETOP

- {static] int SPRITEBOTTOM

- {static] int SPRITELEFT

- {static] int SPRITERIGHT

- int xAttack

- int yAttack

- boolean attackSword

- boolean attackBow

- int numberArrow

+ Player(int x, int y)

+ void increaseHP(int hpGiven)

+ void setDirection()

+ void attackCollision(World world)

+ void attackSword(World world)

+ void drawhit(SpriteBatch sb)

+ void attackBow(int direction)

+ void incrementArrow()

}

class Troll {

+ {static} SPRITESIZE

+ Troll(int x, int y)

+ Troll()

}

}

package "labyrinth" {

abstract class Cell {

- boolean solid

# int x

# int y

# String type

# boolean isActivate

+ Cell(int x, int y)

+ Cell()

+ {abstract} void activate(World world)

+ void draw(SpriteBatch sb)

+ String getType()

+ int getX()

+ int getY()

+ void setActivate()

+ boolean equals(Object o)

+ int hashCode()

}

class Empty {

+ Empty(int x, int y)

+ Empty()

+ boolean isSolid()

+ void activate(World world)

}

class Passage {

- int arriveX

- int arriveY

+ Passage(int posX, int posY, int arriveX, int arriveY)

+ Passage()

+ boolean isSolid()

+ void setDestination(int x, int y)

+ void activate(World world)

+ int getArriveX()

+ int getArriveY()

}

class Trap {

- int damage

+ Trap(int x, int y, int damage)

+ Trap()

+ boolean isSolid()

+ void activate(World world)

+ void CharacterLoarderdraw(SpriteBatch sb)

}

class Treasure {

+ Treasure(int x, int y)

+ Treasure()

+ boolean isSolid()

+ void activate(World world)

+ void activate(World world)

}

class Wall {

+ Wall(int x, int y)

+ Wall()

+ boolean isSolid()

+ void activate(World world)

}

class Magic {

+ Magic(int x, int y)

+ Magic()

+ boolean isSolid()

+ void activate(World world)

+ void draw(SpriteBatch sb)

}

class Mystery {

- int contentActivate

- {static} int print

+ Mystery(int posX, int posY)

+ Mystery()

+ boolean isSolid()

+ void activate(World world)

+ int getContent()

+ void draw(SpriteBatch sb)

}

class Door {

+ Door(int x, int y)

+ Door()

+ boolean isSolid()

+ void activate(World world)

+ void draw(SpriteBatch sb)

}

class Labyrinth {

+ Labyrinth(char[][] tabLevel)

+ Labyrinth()

+ Cell getCell(int x, int y)

+ Cell getNextCell(int x, int y, DirectionFactory direction)

+ void init()

+ void draw(SpriteBatch sb)

+ Cell[][] getCells()

+ boolean equals(Object o)

+ int hashCode()

}

}

class World {

- int level

- int lastLevel

- int room

- Thread one

- boolean currentLevelFinish

- boolean isSaving

- boolean begin

+ World(Game game)

+ void begin()

+ void launchThread()

+ void game()

+ void draw(SpriteBatch sb)

+ void moveHero(DirectionFactory direction)

+ void moveMonster(int direction, int elem)

+ damagePlayer(int damage)

+ teleportPlayer(int x, int y)

+ void winPlayer()

+ void checkLoosePlayer()

+ boolean canMove(Character c, DirectionFactory direction)

+ Player getPlayer()

+ void turn(int direction)

+ void loose()

+ Player getHero()

+ List<Monster> getMonsters()

+ void createLevel()

+ void createRoom()

+ void creatMonsters()

+ void healPlayer(int hp)

+ void createMonsters()

+ Point findEmptyCell()

+ void mystery(int c)

+ void nextDoor()

+ int getDoor()

+ void load()

+ void load(File f)

+ void save()

+ Cell getFirstWall(int x, int y, int direction)

+ Player getHero()

+ void setHero()

+ void mystery(int content)

+ void winArrow()

+ Labyrinth getLabyrinth()

+ void evolveMonsters()

+ void nextRoom()

+ boolean isCurrentLevelFinish()

+ int getLevel()

+ void menuReturn()

+ void save()

+ void load()

+ boolean isSaving()

+ setLevel(int level)

+ boolean isBegin()

+ void setBegin(boolean begin)

+ int getLastLevel()

+ void setLastLevel(int lastLevel)

+ void stopThread()

}

interface Attacker {

# void attackCollision(Wold world)

# void attackSword(World world)

# void attackArrow(World world)

}

class LabyrinthLoader {

+ LabyrinthLoader()

+ Labyrinth createLabyrinth(int numLevel)

}

class CharacterLoader {

+ List<Monster> getMonsters()

+ createCharacter(int numLevel)

+ Player getplayer()

}

}

class Game {

+ Music mp3Sound

+ void create()

+ void dispose()

+ World getWorld()

+ Music getSound()

}

package "controller" {

class Listener {

- Music mp3Sound

- Music mp3SoundArrow

- ViewWorld viewWorld

- Boolean turnFirstPress

+ Listener(ViewWorld)

+ boolean keyDown(int)

+ boolean keyUp(int)

+ boolean keyTyped(char)

+ boolean touchDown(int, int, int, int)

+ boolean touchUp(int, int, int, int)

+ touchDragged(int, int, int)

+ boolean(int)

}

}

package "DataFactory" {

class TextureFactory << (S,#FF7700) Singleton >> {

- HashMap<String,Texture> images

+ TextureFactory getInstance()

- TextureFactory()

+ texture getImage(String)

}

enum DirectionFactory {

+ {static} TOP

+ {static} BOTTOM

+ {static} LEFT

+ {static} RIGHT

+ {static} TURNTOP

+ {static} TURNBOTTOM

+ {static} TURNLEFT

+ {static} TURNRIGHT

}

class LabyrinthFactory {

+ {static} HEIGHT

+ {static} WIDTH

+ {static} WALL

+ {static} GROUND

+ {static} FIRE

+ {static} DELTA

+ {static} PASSAGE

+ {static} MAGIC

+ {static} DOOR

+ {static} MYSTERY

+ {static} MAGICHEALTH

+ {static} PLAYER

+ {static} GHOST

+ {static} TROLL

+ {static} NB\_NIVEAUX

+ {static} HP\_PLAYER

+ {static} NB\_MYSTERY

+ {static} MYSTERYTRAP

+ {static} MYSTERYCARE

+ {static} MYSTERYARROW

+ {static} ARROWNUMBER

}

}

CharacterLoader --"1" Player : hero

CharacterLoader --"0,..,n" Monster : monsters

ViewWorld --"1" Game: game

ViewTuto --"1" Game : game

ViewMenu --"1" Game : game

FileChooser --"1" Game : game

Listener --"1" ViewWorld : viewWorld

Character ..|> Movement

TextureFactory -"1" TextureFactory : ourInstance

Monster ..|> Attacker

Monster --|> Character

Ghost --|> Monster

Ghost ..|> Attacker

Player --|> Character

Troll --|> Monster

Troll ..|> Attacker

Empty --|> Cell

Passage --|> Cell

Trap --|> Cell

Treasure -Right-|> Cell

Wall --|> Cell

Magic --|> Cell

Mystery --|> Cell

Door --|> Cell

Labyrinth -Left-"0..n" Cell : cellList

World --"1" Player : hero

World --"0..n" Monster : monsters

World --"1" LabyrinthLoader : labyrinthLoader

World --"1" Labyrinth

World --"1" CharacterLoader : characterLoader

World --"1" Game

Game --"1" ViewMenu

Monsters --"0..n" Monster : monsters

@enduml