Methods of Development

College 4
OO & Tentamenstof

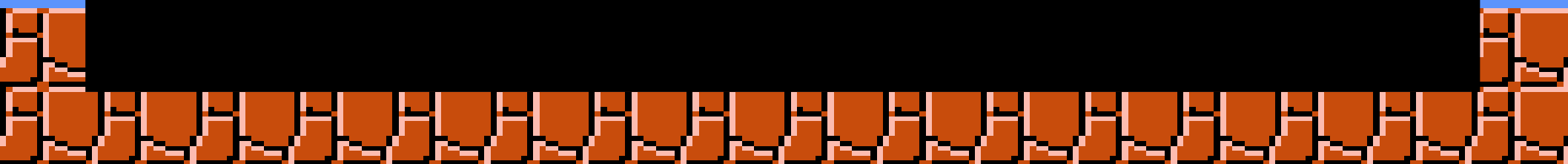
Arjan Scherpenisse
arjan.scherpenisse@kmt.hku.nl
@acscherp

Deze week

- Object-oriëntatie
  - Herhaling vorige week
  - Compositie
- De Opdracht
- Tentamenstof overview
- Q&A


Hello, world!!
Child.naam = ”Arris Thedo Scherpenisse”
Child.roepnaam = ”Arris”

Objecten objecten objecten...



De Interface van een class

Class TROOPER

Attributen:

- naam
- prijs
- positie
- health

Methoden:

+ attack()
+ move()
+ retreat()
+ guard()

Instantiëren maarr..!

position = (10, 10)
health = 100

position = (190, 80)
health = 30

position = (10, 200)
health = 100

“Class” vs. “Object” ???

- Een class is een definitie
  - Er is maar 1 definitie
- Een object is een instantie van een class
  - Er kunnen meerdere instanties zijn
    (meerdere *dezelfde* units, allemaal met verschillende health, positie e.d.)

Instantiëren maarrrr...

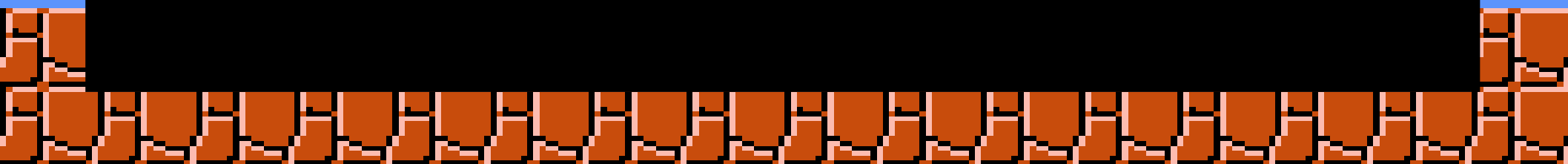
```
SET unitCount = 0
REPEAT
   new Trooper()
   unitCount = unitCount + 1
UNTIL  unitCount = 100
```

Instantiëren maarrrr...

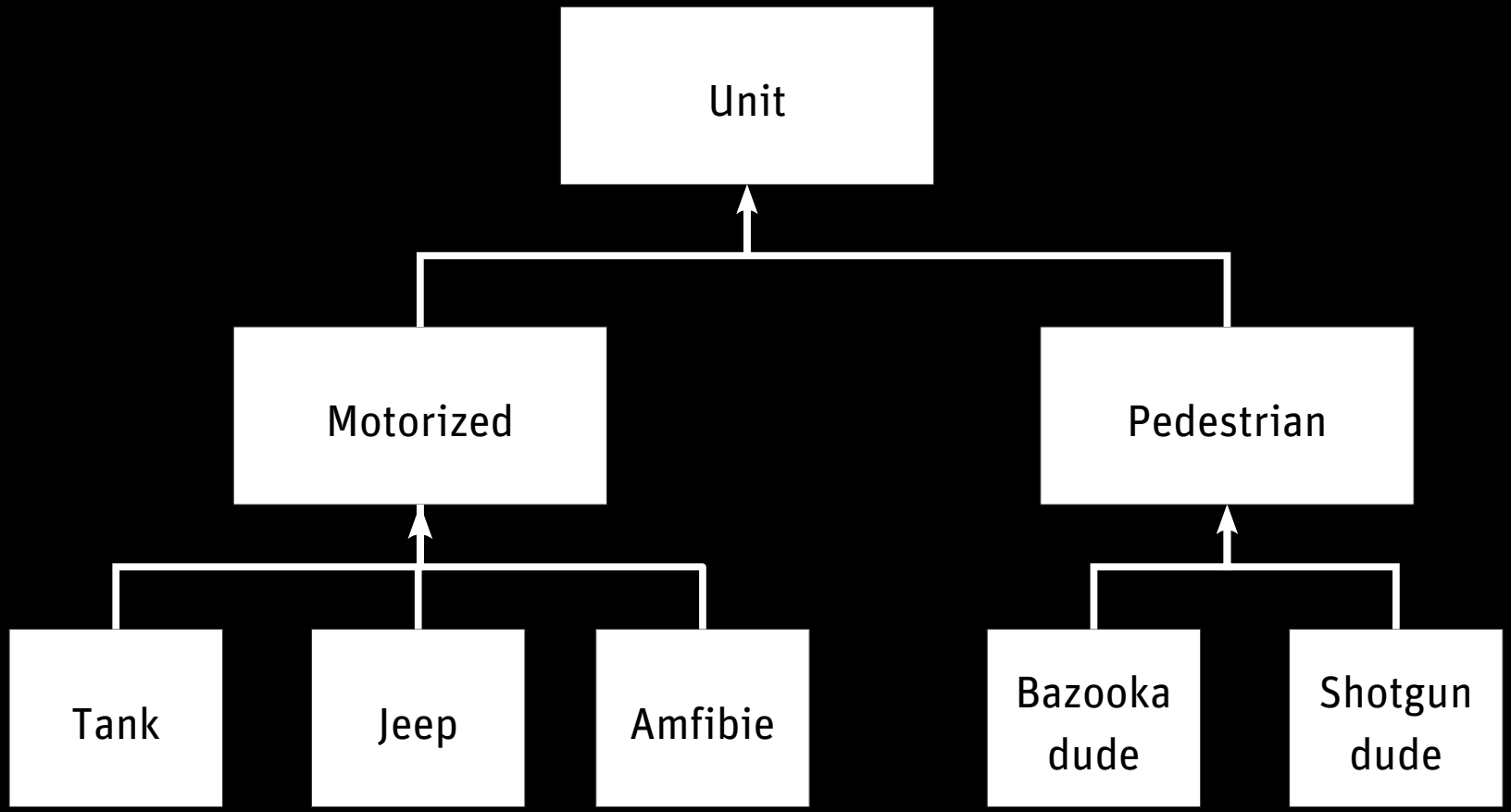
(Als je later iets met de troopers wilt doen, moet je ze bewaren in een aray..)

```
SET unitCount = 0
SET units = [ ]
REPEAT
    mijnUnit = new Trooper()
   add mijnUnit to units
   unitCount = unitCount + 1
UNTIL  unitCount = 100
```

Inheritance



Overerving (inheritance)



Ontwerp van classes

- Stop niet alles in 1 class
- Probeer logische componenten te vinden
- Zijn er dingen te generaliseren?
- Zijn er dingen te herbruiken?
- Compositie
  - Classes kunnen andere classes als “onderdelen” hebben

Compositie

CLASS character
ATTRIBUTEN:

- naam → “Weirdo1990”
- level → 7
- inventory → [dagger, blue gem, magic mushroom]
- controller

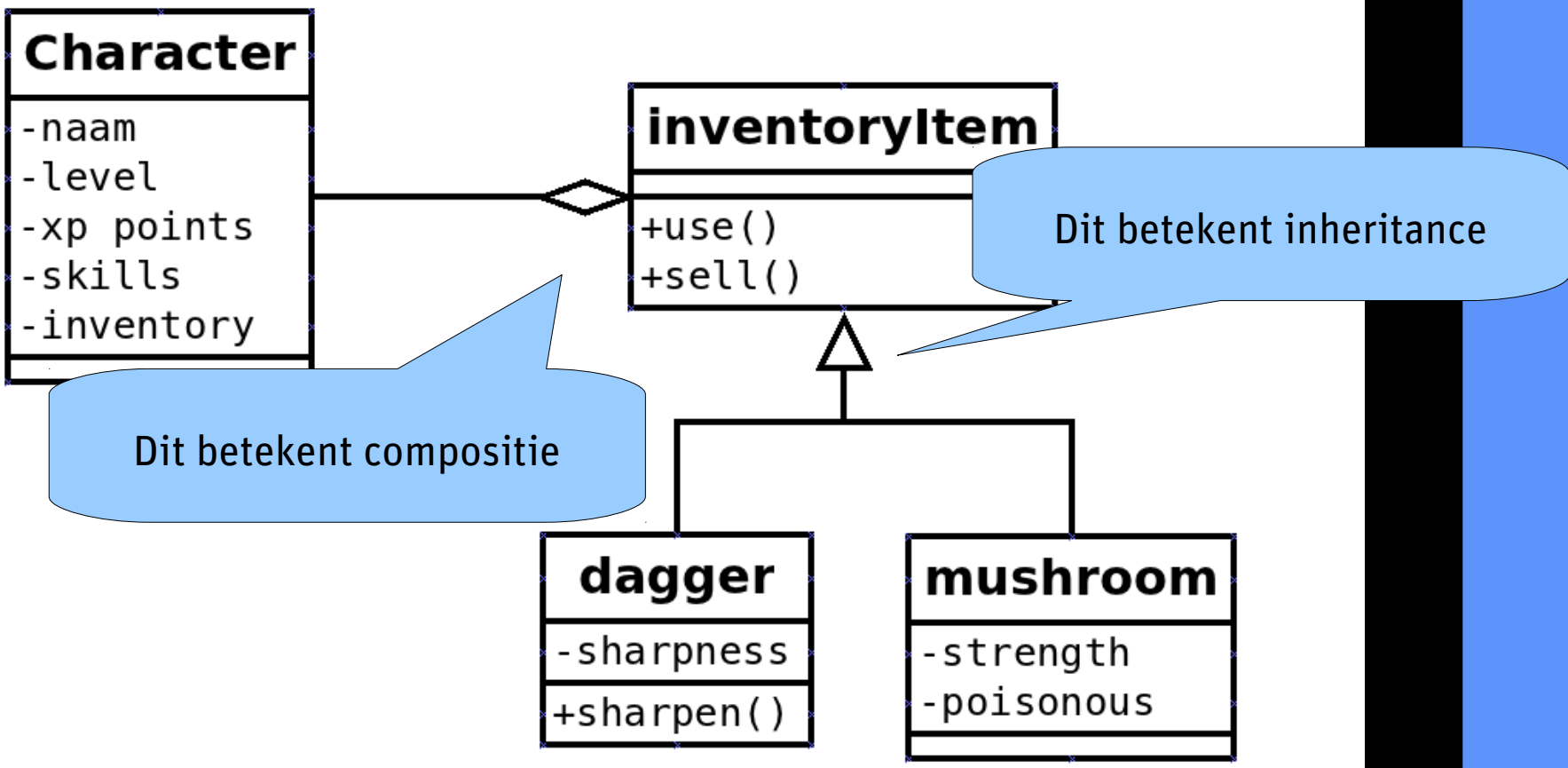
CLASS inventoryItem
ATTRIBUTEN:
- name
- price

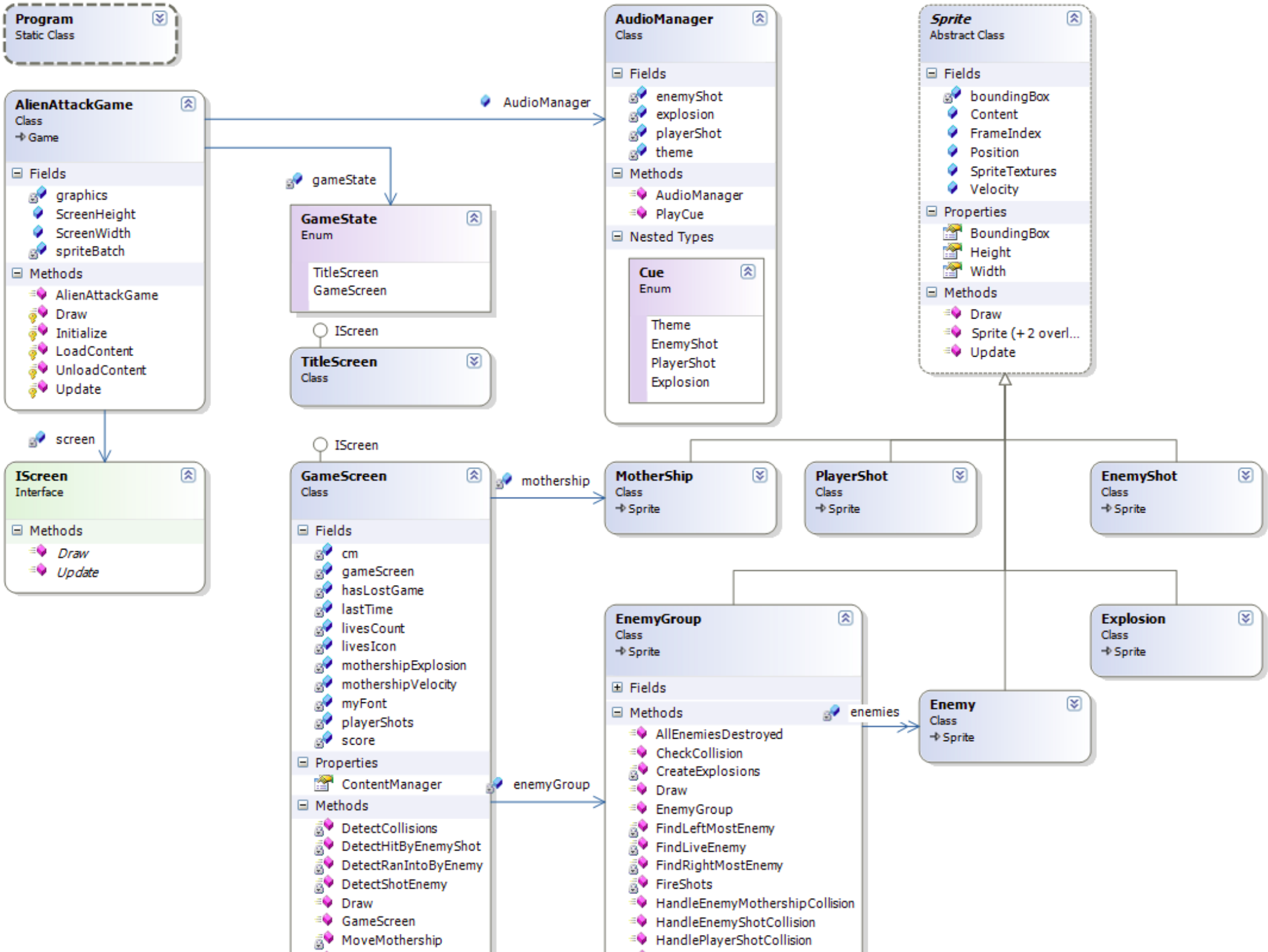
METHODEN:
+ sell()
+ use()

CLASS dagger EXTENDS inventoryItem
ATTRIBUTEN:
- sharpness

CLASS magicmushroom EXTENDS inventoryItem
ATTRIBUTEN:
- strength
- poisonous yes/no

Compositie


Character
-naam
-level
-xp points
-skills
-inventory
inventoryItem
+use()
+sell()
Dit betekent inheritance
Dit betekent compositie
dagger
-sharpness
+sharpen()
mushroom
-strength
-poisonous


Program
Static Class
AlienAttackGame
Class
Game
Fields
graphics
ScreenHeight
ScreenWidth
spriteBatch
Methods
AlienAttackGame
Draw
Initialize
LoadContent
UnloadContent
Update
AudioManager
gameState
screen
AudioManager
Class
Fields
enemyShot
explosion
playerShot
theme
Methods
AudioManager
PlayCue
Nested Types
Cue
Enum
Theme
EnemyShot
PlayerShot
Explosion
GameState
Enum
TitleScreen
GameScreen
IScreen
TitleScreen
Class
Sprite
Abstract Class
Fields
boundingBox
Content
FrameIndex
Position
SpriteTextures
Velocity
Properties
BoundingBox
Height
Width
Methods
Draw
Sprite (+ 2 overl...
Update
IScreen
Interface
Methods
Draw
Update
IScreen
GameScreen
Class
mothership
Fields
cm
gameScreen
hasLostGame
lastTime
livesCount
livesIcon
mothershipExplosion
mothershipVelocity
myFont
playerShots
score
Properties
ContentManager
Methods
DetectCollisions
DetectHitByEnemyShot
DetectRanIntoByEnemy
DetectShotEnemy
Draw
GameScreen
MoveMothership
MotherShip
Class
Sprite
PlayerShot
Class
Sprite
EnemyShot
Class
Sprite
enemyGroup
EnemyGroup
Class
Sprite
Fields
Methods
AllEnemiesDestroyed
CheckCollision
CreateExplosions
Draw
EnemyGroup
FindLeftMostEnemy
FindLiveEnemy
FindRightMostEnemy
FireShots
HandleEnemyMothershipCollision
HandleEnemyShotCollision
HandlePlayerShotCollision
enemies
Enemy
Class
Sprite
Explosion
Class
Sprite

class diagram game

Search

About 2,220,000 results (0.21 seconds)

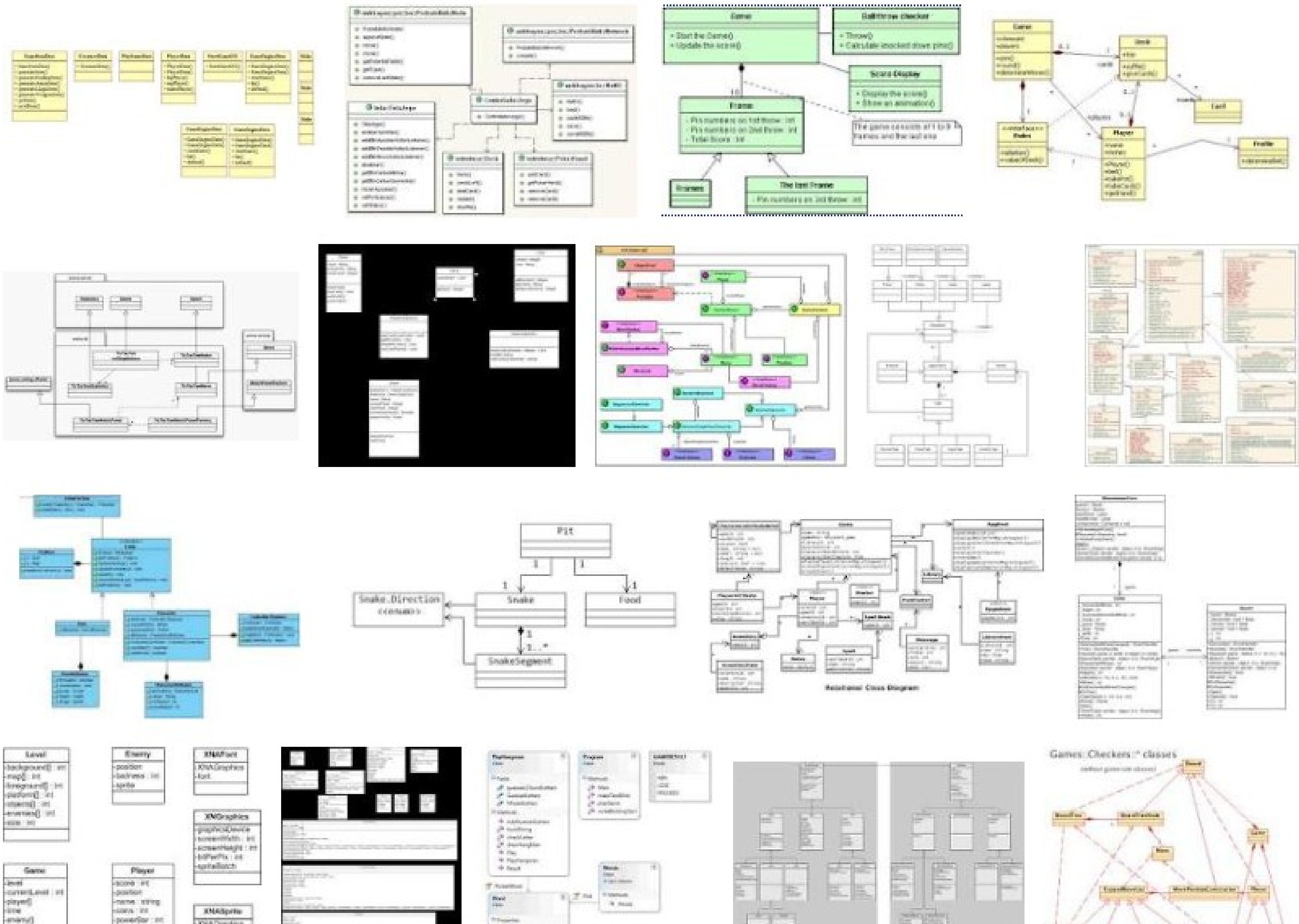
SafeSearch

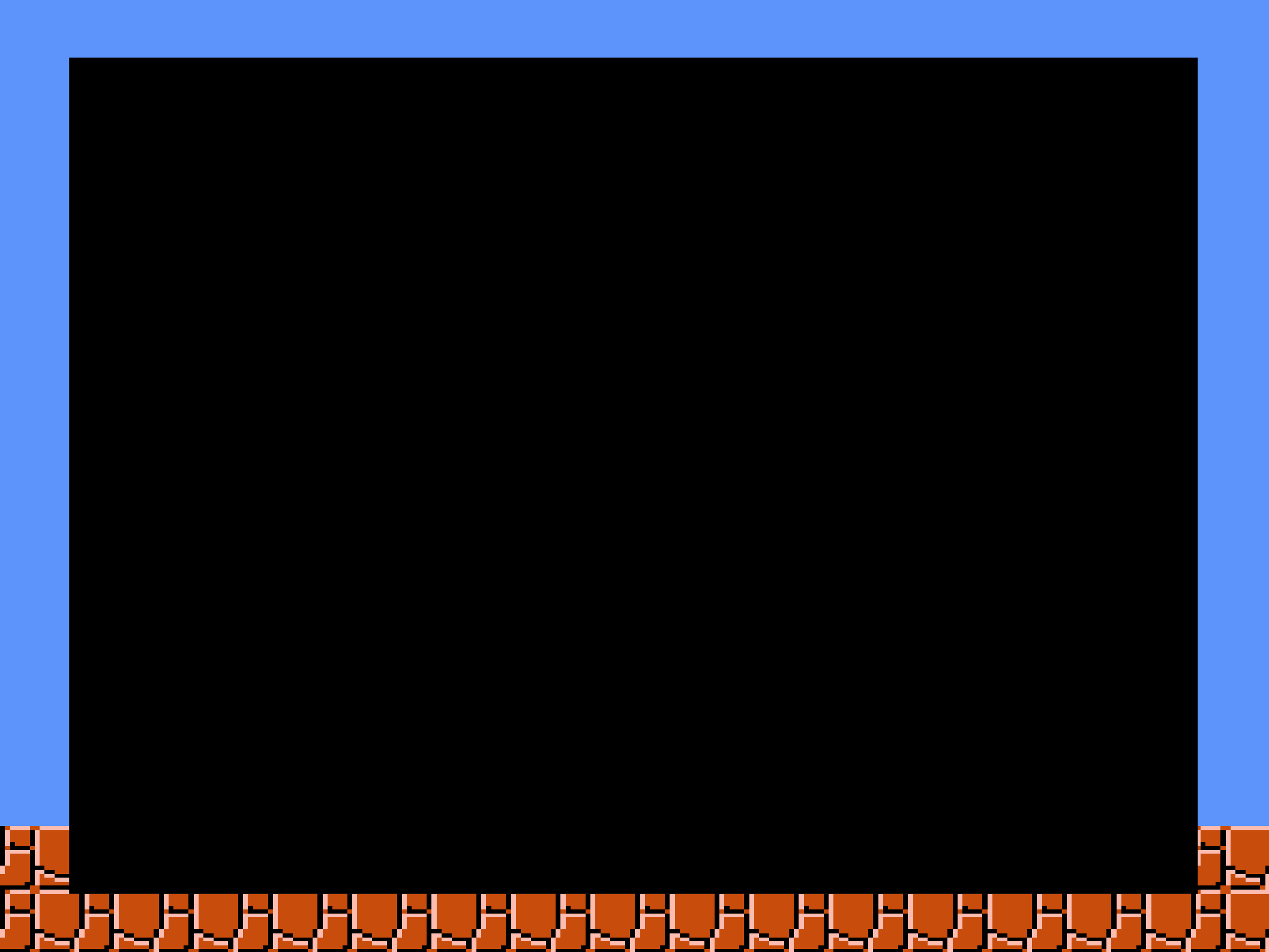
- Everything
- Images
- Maps
- Videos
- News
- Shopping
- More

All results
By subject

Any size
Large
Medium
Icon
Larger than...
Exactly...

Any color
Full color
Black and white



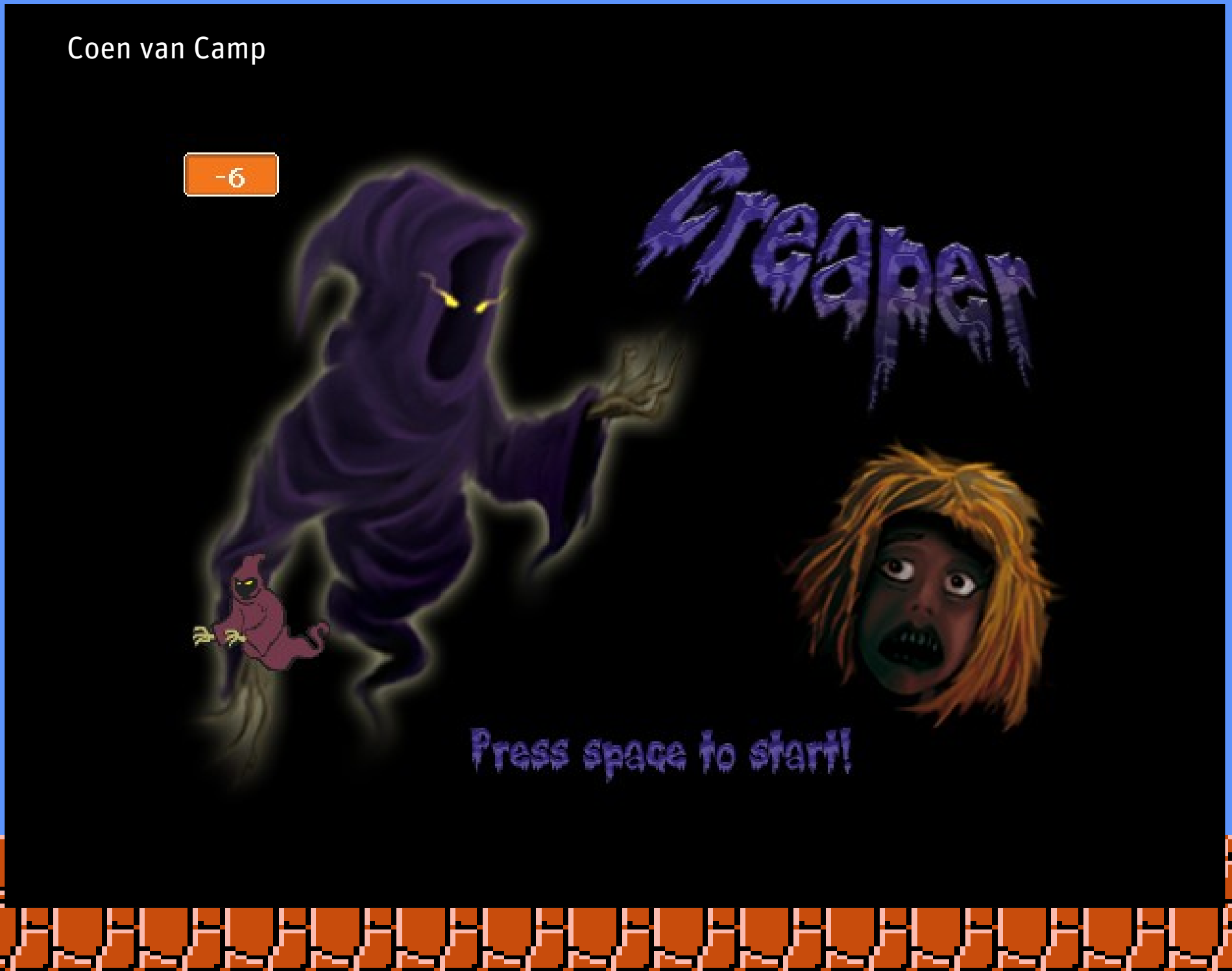


Tussenstand opdracht

- 92 mensen hebben een Scratch game ingeleverd!
- 6 mensen hebben pseudocode gestuurd
- 4 mensen heeft class diagrams gestuurd

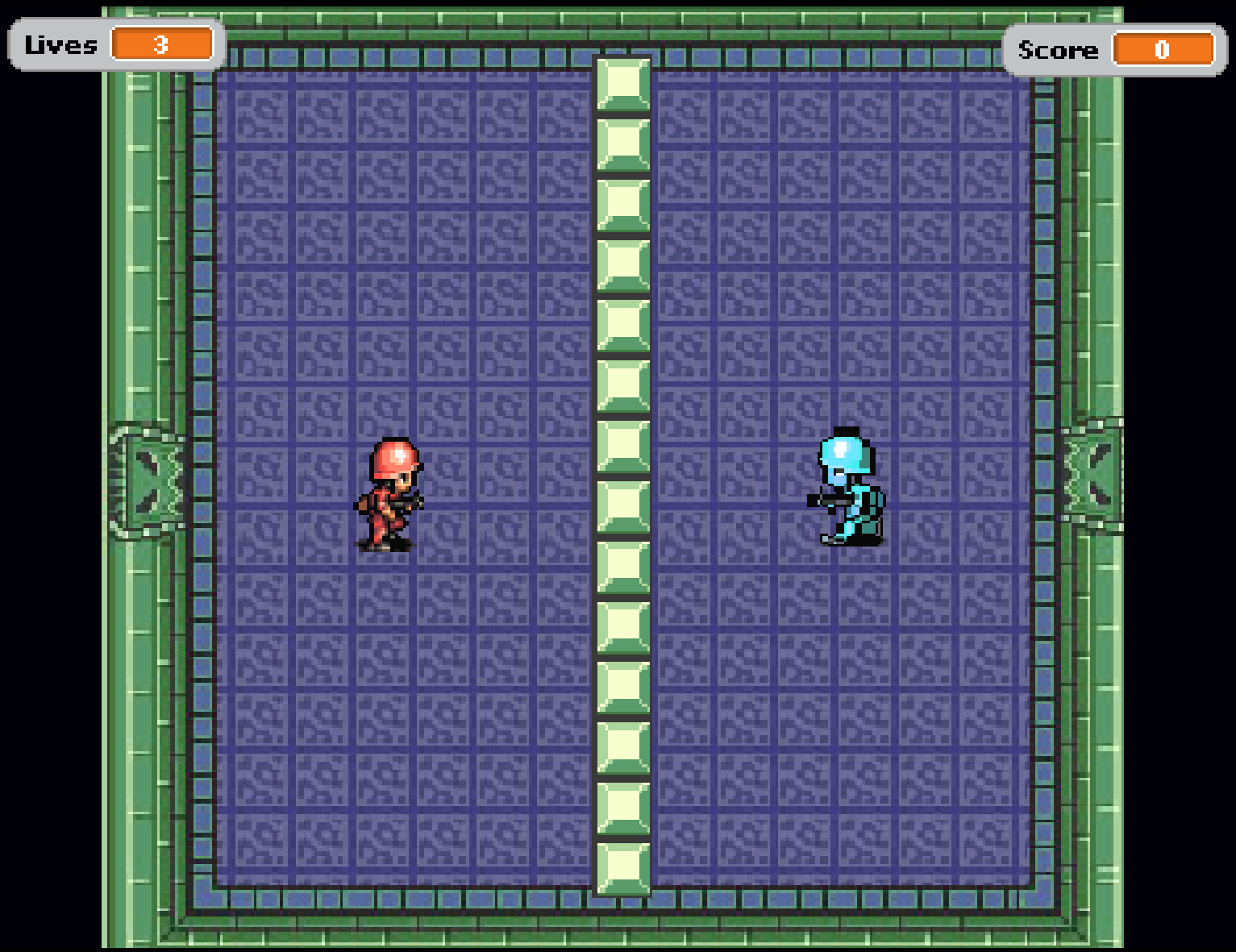
- Hmm....

Coen van Camp

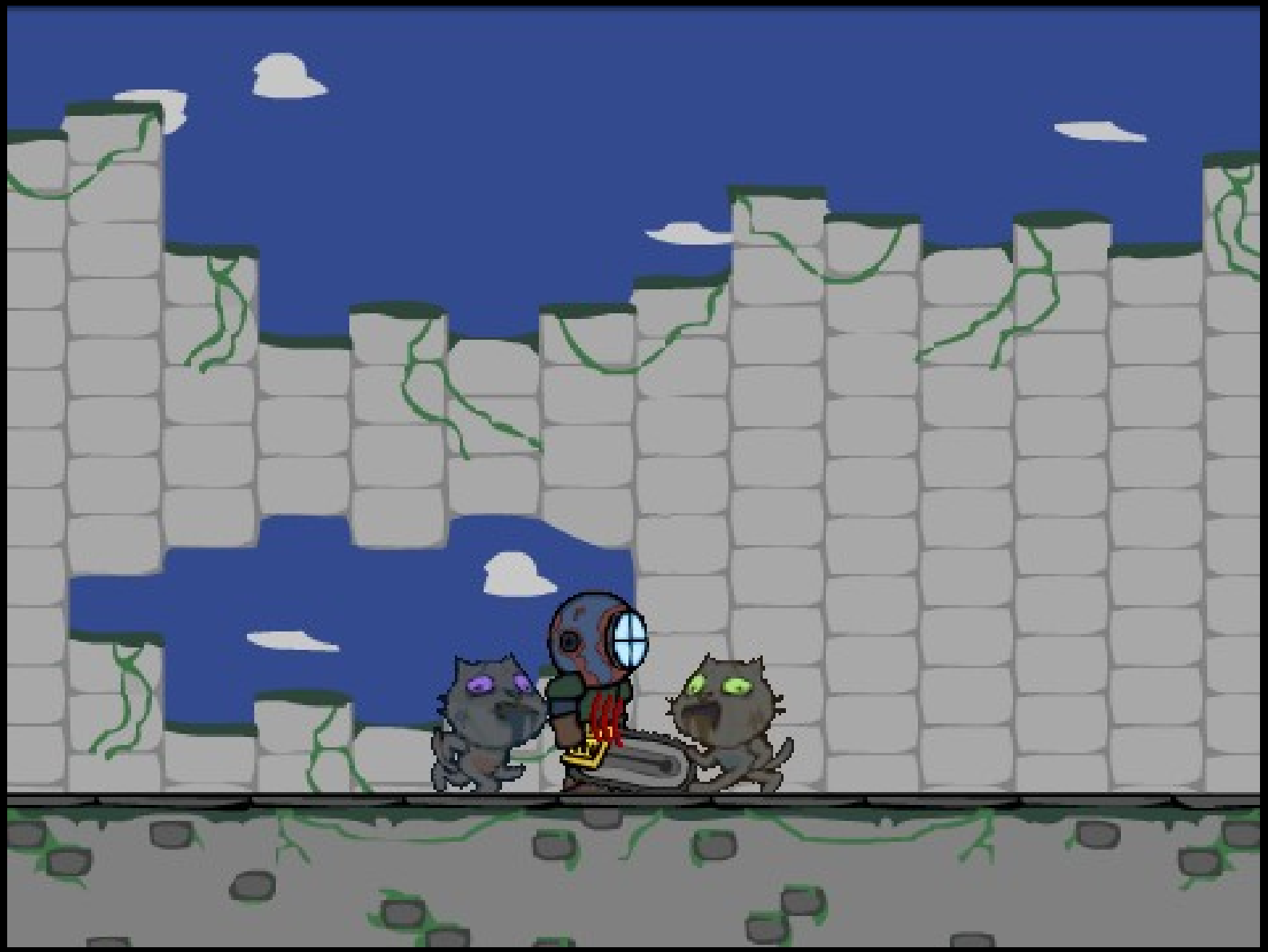


Damir Kalbic

Kevin Nederkoorn



Kasper Smits



Max Plooi


ARJAN
Lv:1
HP
HP: 100
HP: 100
STUDENT
Lv:1
HP
What do you want to do?
THROW PAPER
COMPLAIN
MAKE NOISE
SLEEP
make noise

Download alle 92!

- *klik*

  http://usat1112.hku.nl/wp-content/uploads/2011/11/scratch_games.zip

Ontwerp van software

- Don't Repeat Yourself (DRY)
  - Zijn er dingen te herbruiken?
    Denk in classes, overerving, functies
- Keep It Simple, Stupid (KISS)
  - Maak het in eerste instantie niet moeilijker dan nodig, dat kan later altijd nog
- De 80/20 regel
  - De laatste 20% kost 80% van de tijd...

Design patterns (heel snel)

- “Standaard manieren om dingen te doen”
- Singleton
  - Als je maar 1 instantienodig hebt van een class
- Factory
  - Het uitbesteden van het instantieren van objecten
- Listener/observer
  - Loskoppelen van logica

Komende colleges

- ~~College 1: waar hebben we het over~~
- ~~College 2: imperatief programmeren~~
- ~~College 3: object-orientatie~~
- ~~College 4: vervolg OO, design patterns, Q&A~~
- Tentamen 8 december 10:00 – 11:30

Tentamenstof (1)

- De 6 best practices uit college 1

Tentamenstof (2)

- Pseudocode snappen EN kunnen schrijven
- Pseudocode begrippen
  - If/then/else
  - Loops
  - Variabelen
  - Functies

Tentamenstof (3)

- Object-georienteerd kunnen denken
- Begrippen:
  - Class vs. object
  - Interface vs. implementatie
  - Encapsulatie
  - Overerving (inheritance)
  - Compositie

Beoordeling

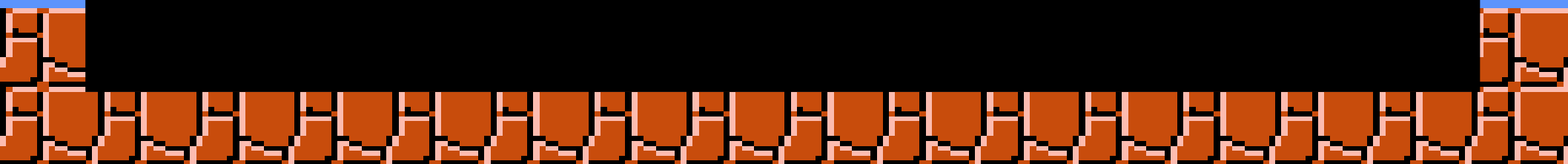
- Tentamen
  - 60% multiple choice
  - 40% open vraag
- Opdracht
  - Opdracht 1 ingeleverd
  - Opdracht 2 voldoende
- **Beoordeling = (opdracht + tentamen) / 2**

Oefenopdracht

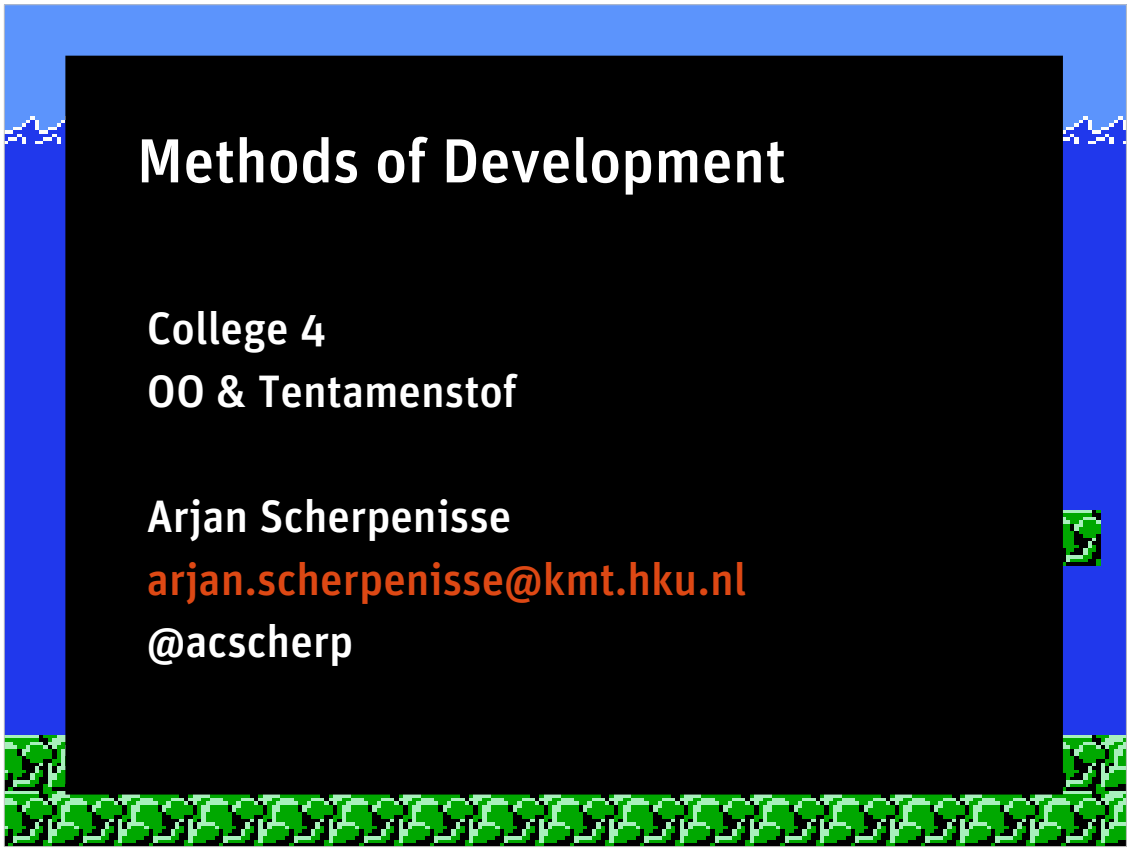
- Bedenk een goede tentamenvraag!
  - Multiple choice
    - 1 goed antwoord, 2 foute antwoorden
  - Open vraag
    - met voorbeelduitwerking
- 2 van de inzendingen komen terug in het *echte* tentamen!

Zijn Er Nog Vragen?

- Handige aantekeningen PDF binnenkort online!
  - Hou blog in de gaten.



Leer-ze en tot volgende week!


Methods of Development
College 4
OO & Tentamenstof
Arjan Scherpenisse
arjan.scherpenisse@kmt.hku.nl
@acscherp

Deze week

- Object-oriëntatie
  - Herhaling vorige week
  - Compositie
- De Opdracht
- Tentamenstof overview
- Q&A


Hello, world!!
Child.naam = ”Arris Thedo Scherpenisse”
Child.roepnaam = ”Arris”


Objecten objecten objecten...

Objecten hebben een publieke “interface”. Dat is de manier waarop je het object kan gebruiken.

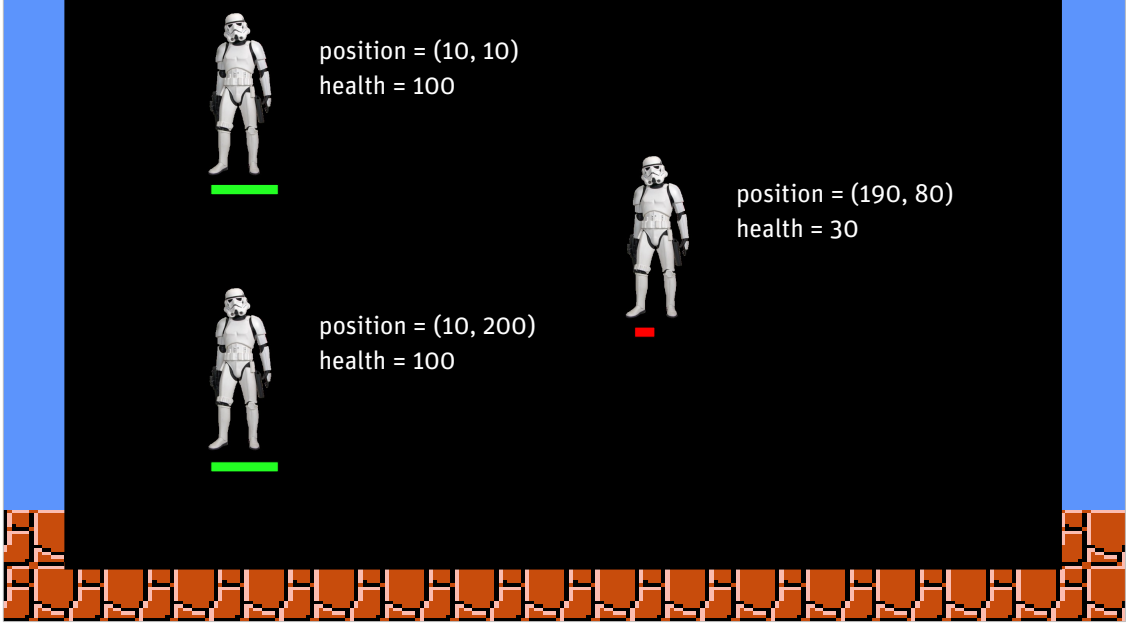
Met zijn interface geeft het object aan wat het kan en hoe het gebruikt dient te worden.

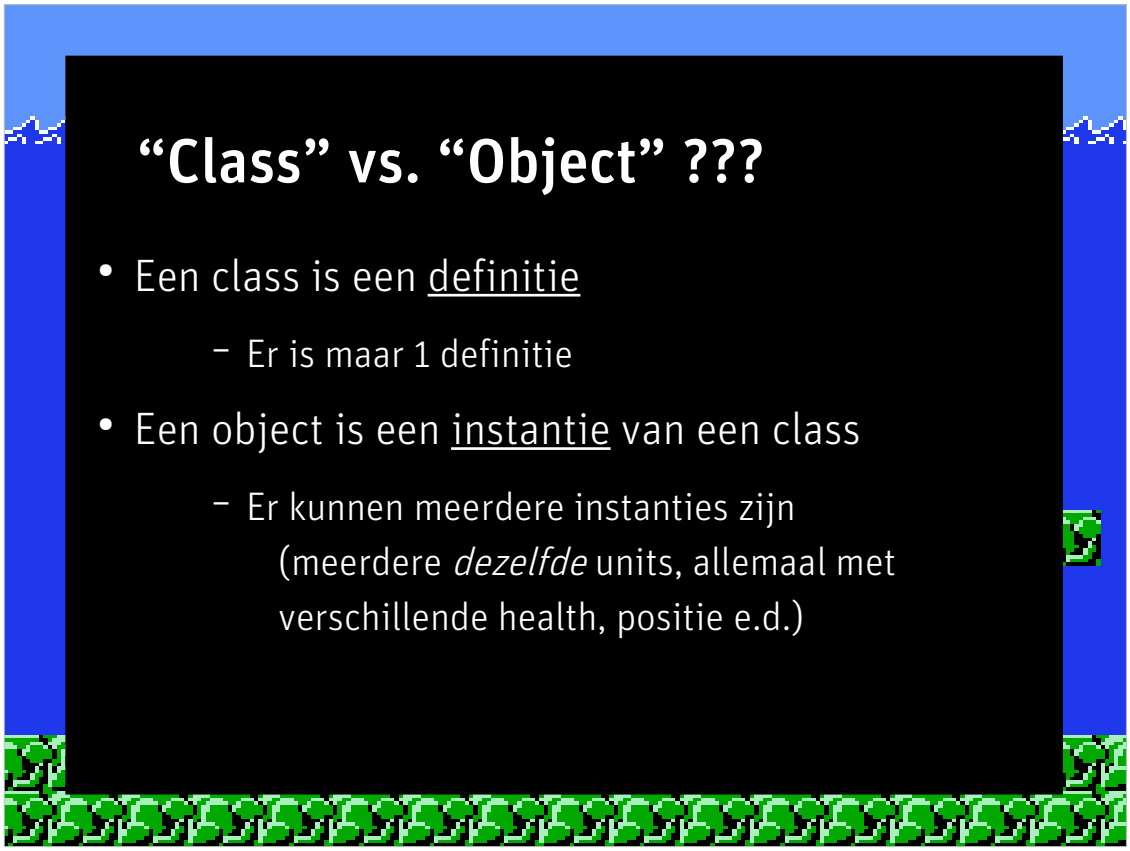
In tegenstelling tot de ***implementatie***, weet je bij de interface niet *hoe* iets is geprogrammeerd. Je gaat er gewoon vanuit dat het werkt zoals er wordt gezegd dat het werkt.

De **implementatie** is de daadwerkelijke code die achter elke methode van een class verborgen zit.

Het woord **interface** gaat zeker te weten terugkomen in een tentamenvraag!

Instantiëren maarr..!





Als je in een RTS game een overzicht hebt van je units die je kunt laten bouwen, dan kijk je eigenlijk naar een lijstje van **classes**.

Als je een unit laat bouwen in de game, dan wordt hij **geinstantieerd** tot een **object**, met zijn eigen naam, health, positie, e.d.

Je programmeert de class maar uiteindelijk leven in je programma de instanties van die class, de objecten.

Instantiëren maarrrr...

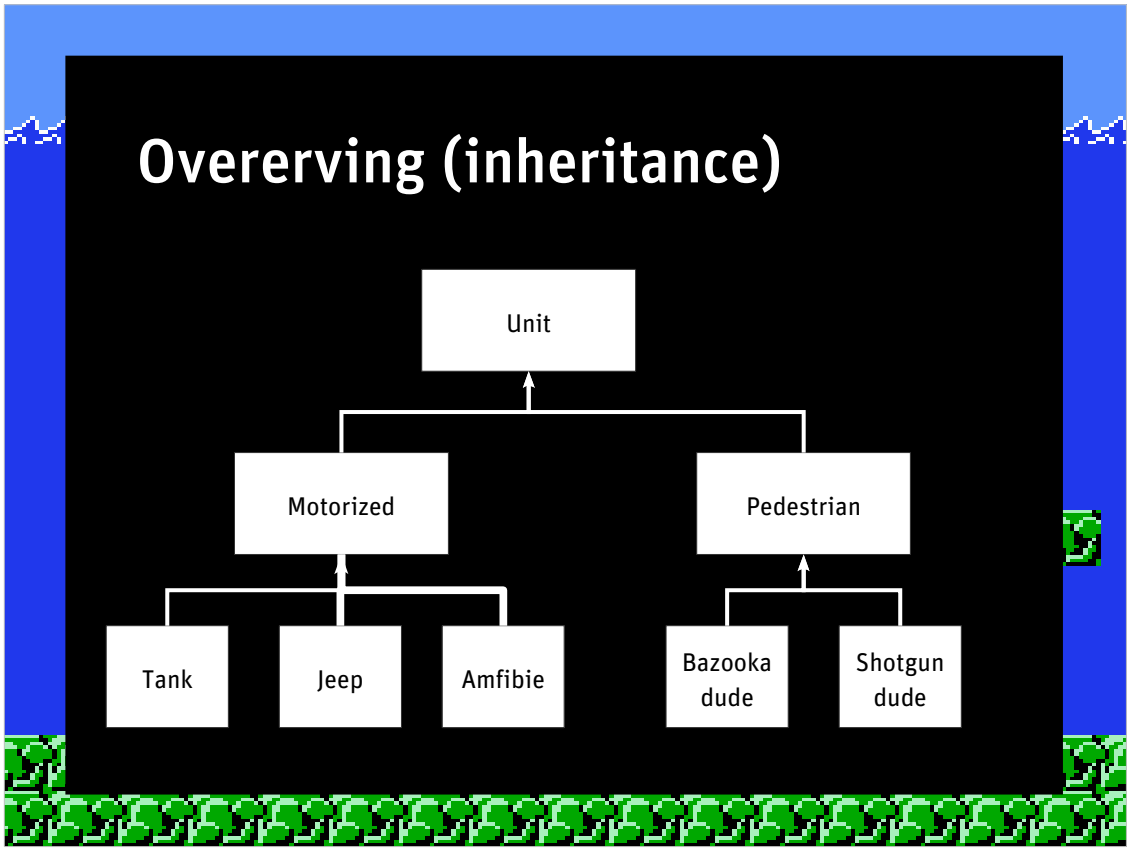
```
SET unitCount = 0
REPEAT
   new Trooper()
   unitCount = unitCount + 1
UNTIL  unitCount = 100
```

Instantiëren maarrrr...

(Als je later iets met de troopers wilt doen, moet je ze bewaren in een aray..)

```
SET unitCount = 0
SET units = [ ]
REPEAT
    mijnUnit = new Trooper()
   add mijnUnit to units
   unitCount = unitCount + 1
UNTIL  unitCount = 100
```


Inheritance

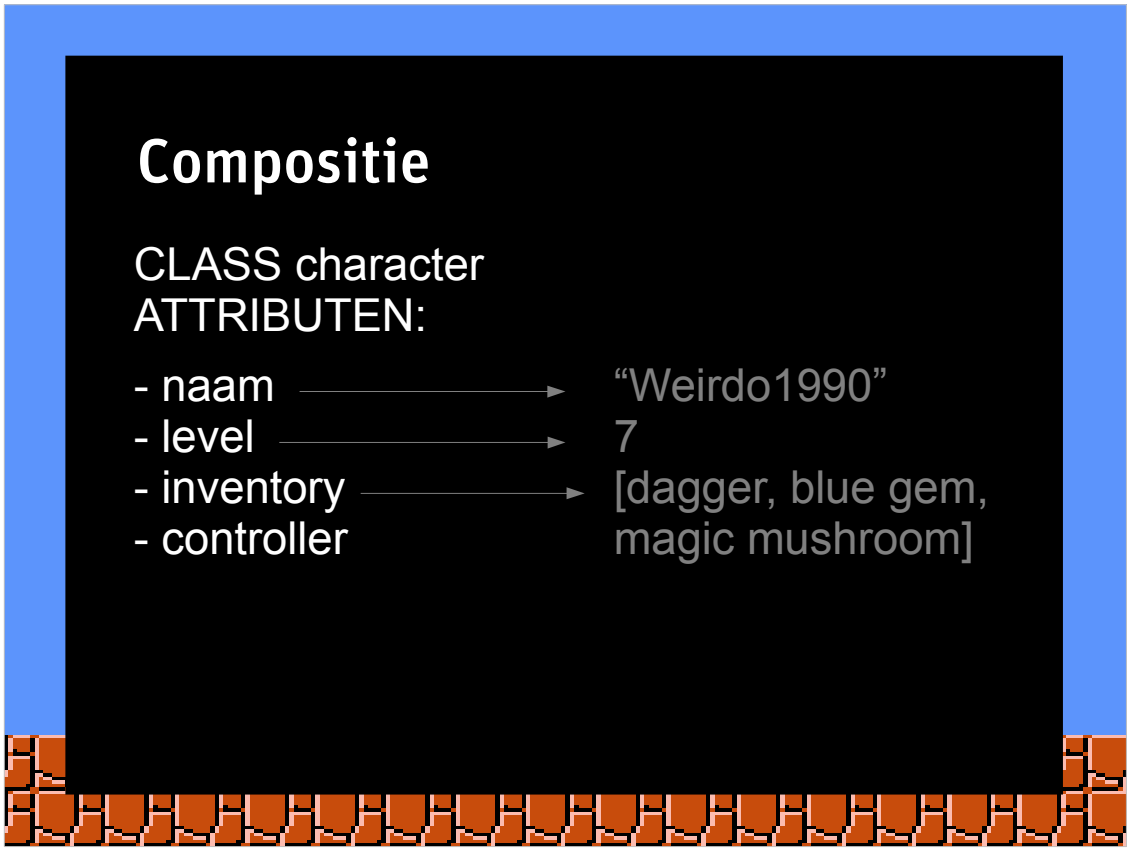


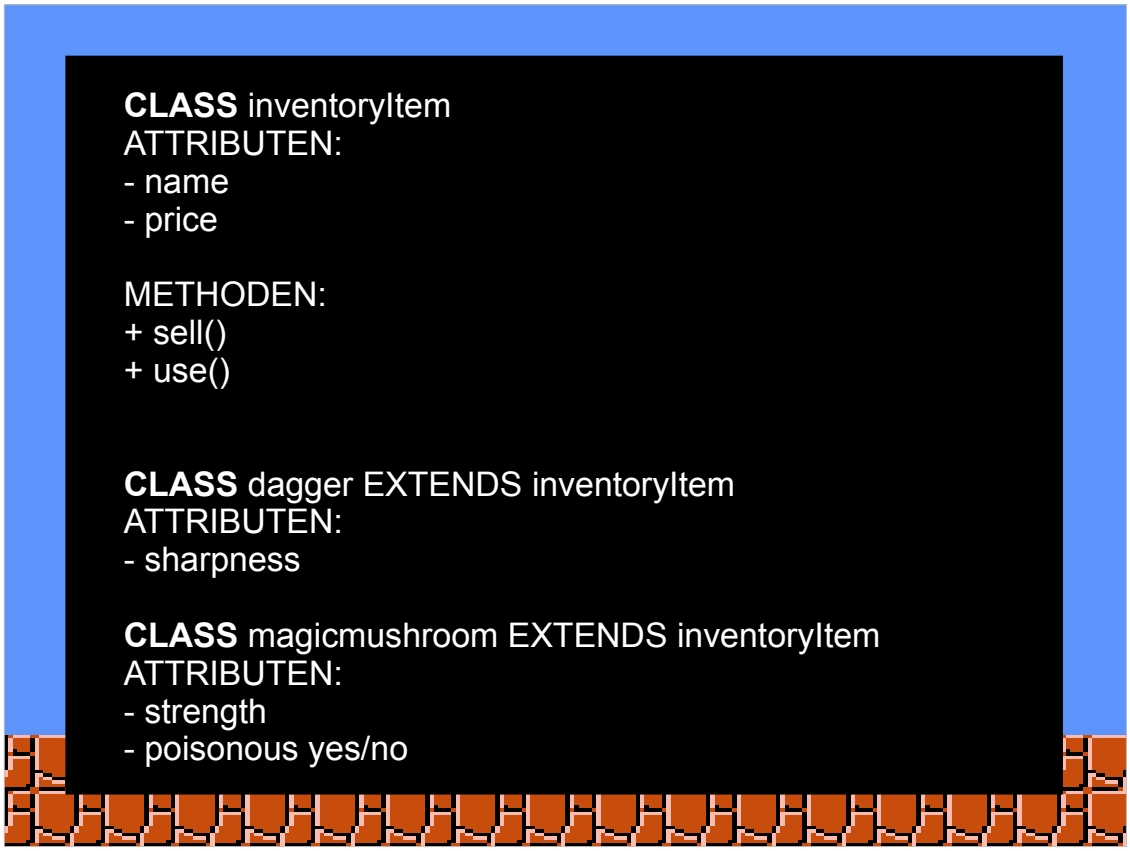
Classes kunnen eigenschappen van elkaar overnemen.

Ze “delen” eigenschappen. Alle units in een RTS hebben bv. Een positie, maar ze kunnen niet allemaal schieten.

Ontwerp van classes

- Stop niet alles in 1 class
- Probeer logische componenten te vinden
- Zijn er dingen te generaliseren?
- Zijn er dingen te herbruiken?
- Compositie
  - Classes kunnen andere classes als “onderdelen” hebben

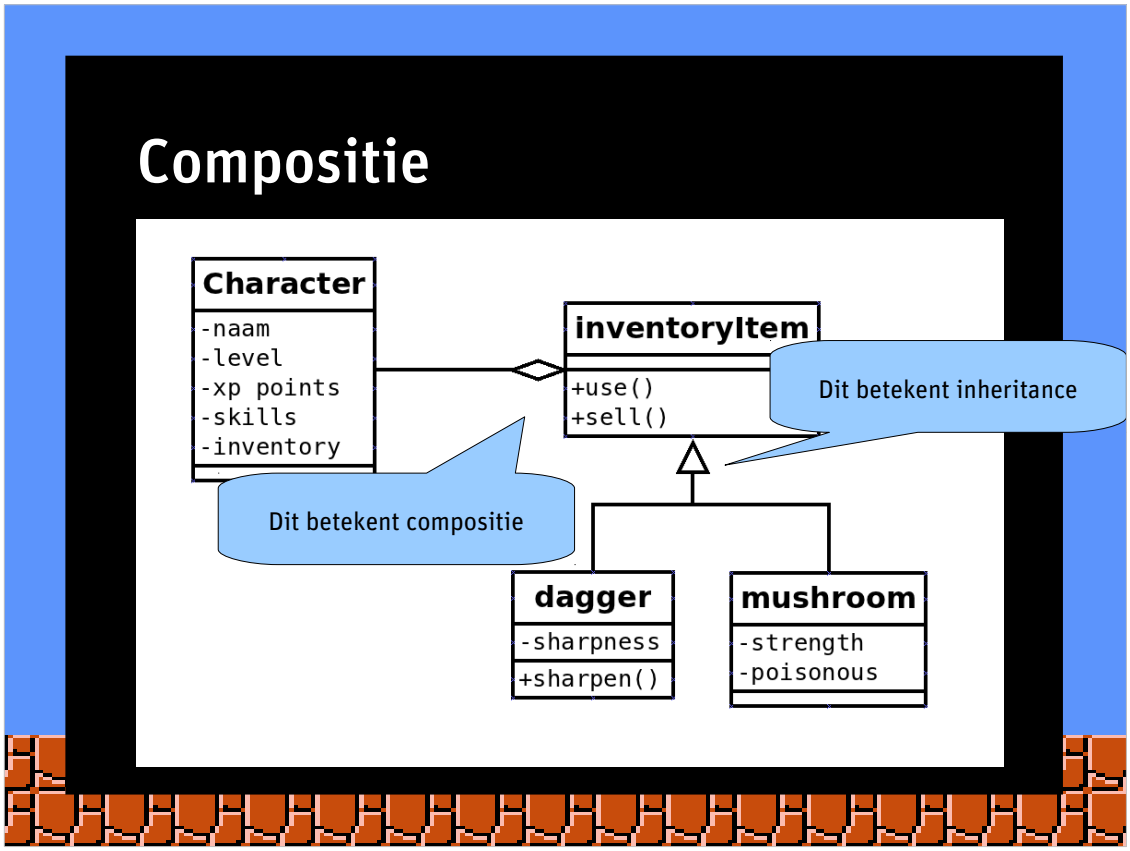

Compositie
CLASS character
ATTRIBUTEN:
- naam
- level
- inventory
- controller
“Weirdo1990”
7
[dagger, blue gem,
magic mushroom]



Je ziet dat inventoryItem een methode heeft “use”.
Maar wat die methode precies doet, is afhankelijk van de subclass!!

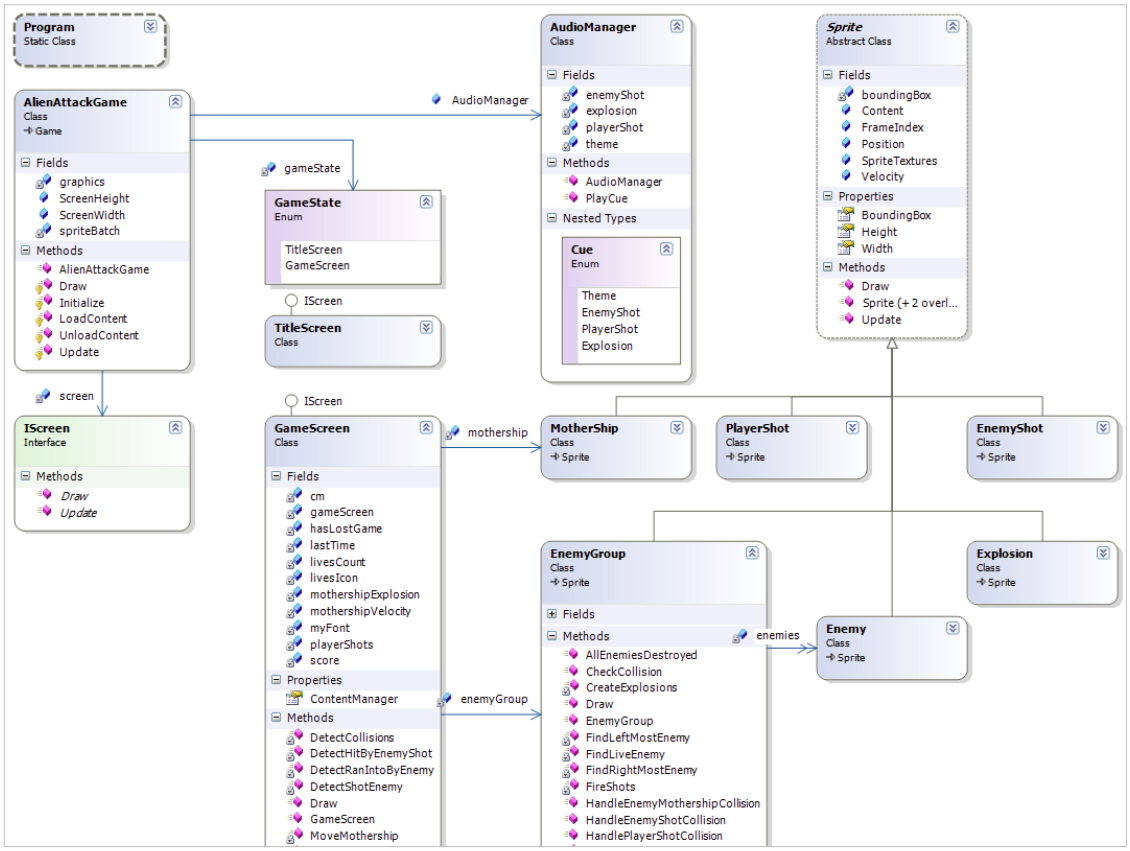
Het is dus 1 methode met *meerdere implementaties*, eentje per subclass. (Dit fenomeen heet “polymorphisme”.)

Maar de methode “sell” is voor beide subclasses identiek. Of je nu een dagger of een mushroom wilt verpatsen, dat gaat op dezelfde manier. (je raakt het item kwijt en krigjt er geld voor terug)



De class inventoryItem is **abstract**.

Je instantieert nooit direct een inventoryItem, maar

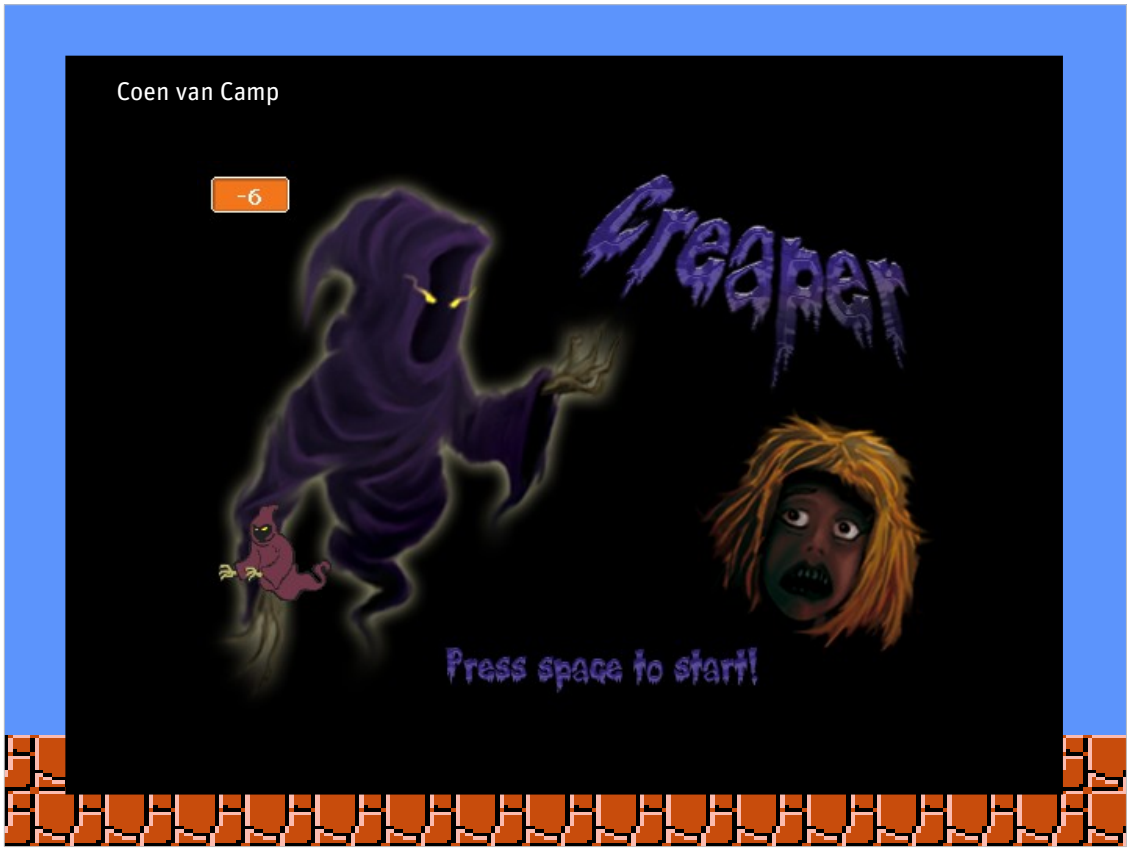


Games zijn vaak hele ingewikkelde dingen met veel classes. Deze classes hebben allemaal een relatie tot elkaar.

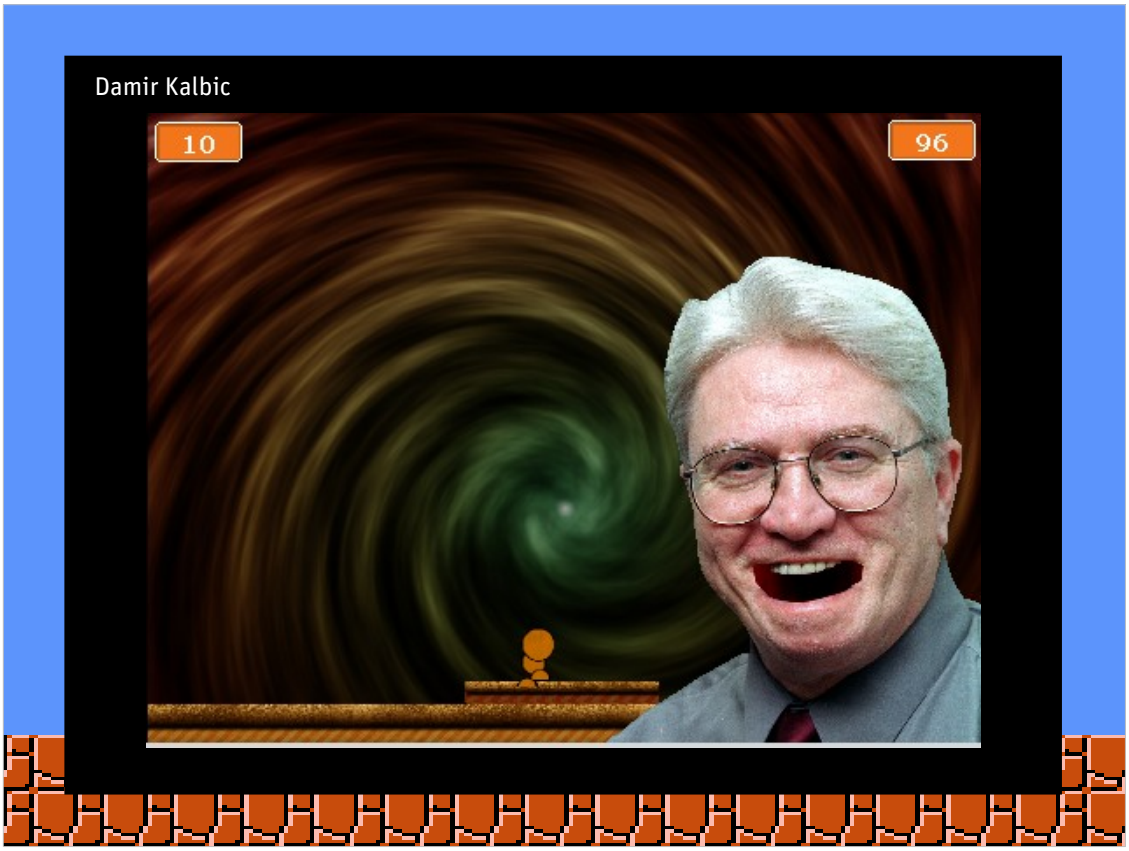
Google maar eens op “class diagram game” om meer voorbeelden te zien.

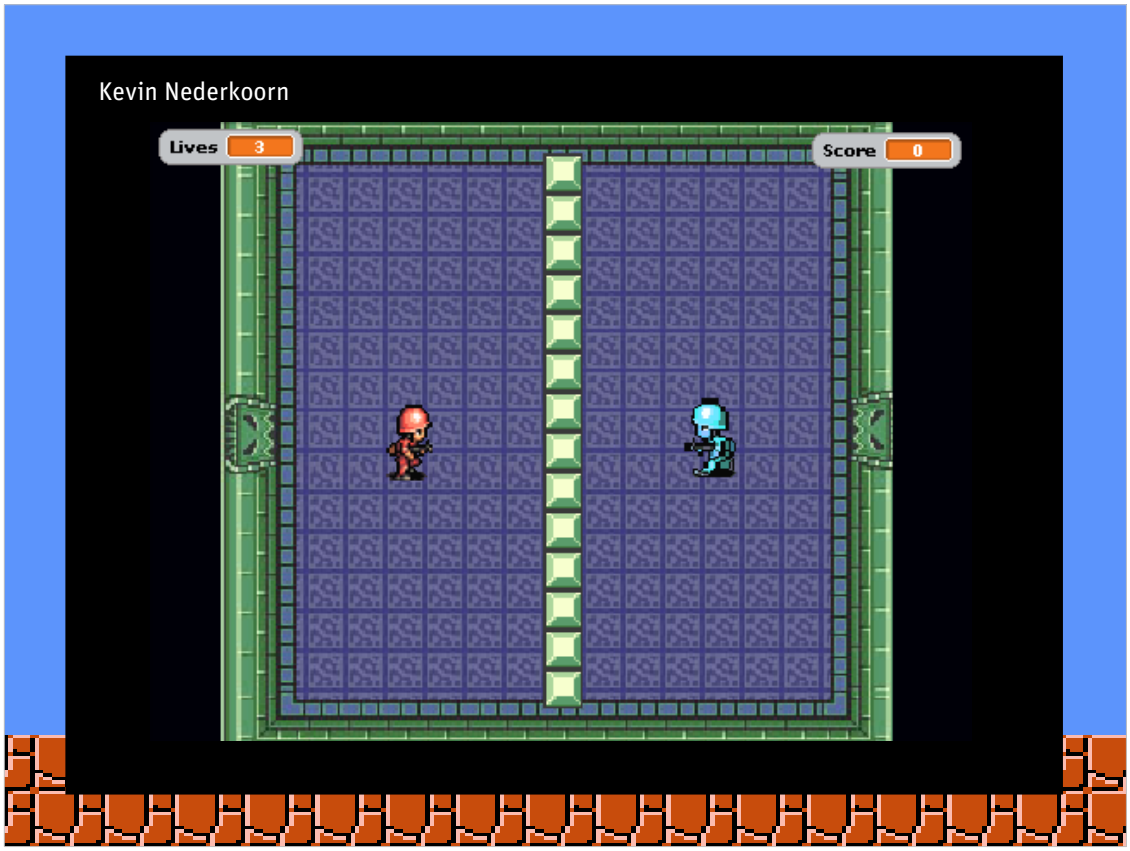
Tussenstand opdracht

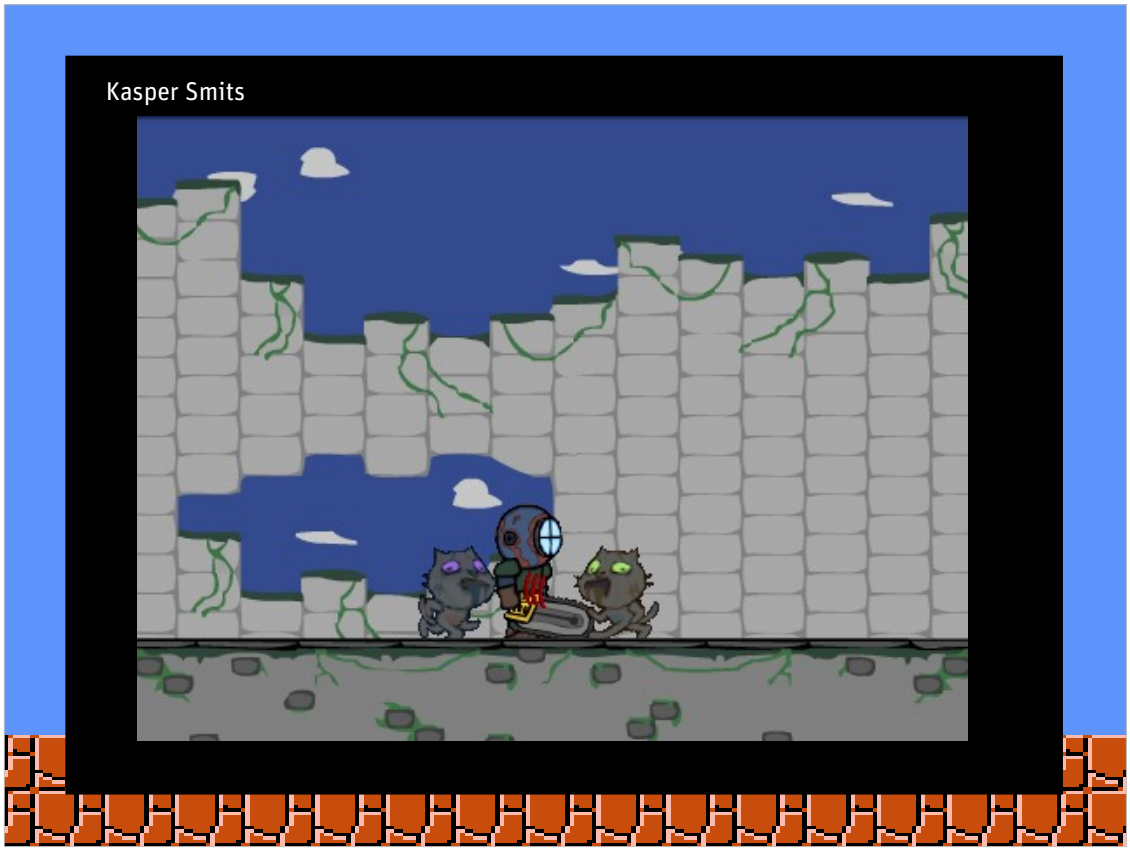
- 92 mensen hebben een Scratch game ingeleverd!
- 6 mensen hebben pseudocode gestuurd
- 4 mensen heeft class diagrams gestuurd
- Hmm....


Coen van Camp
-6
Greaper
Press space to start!

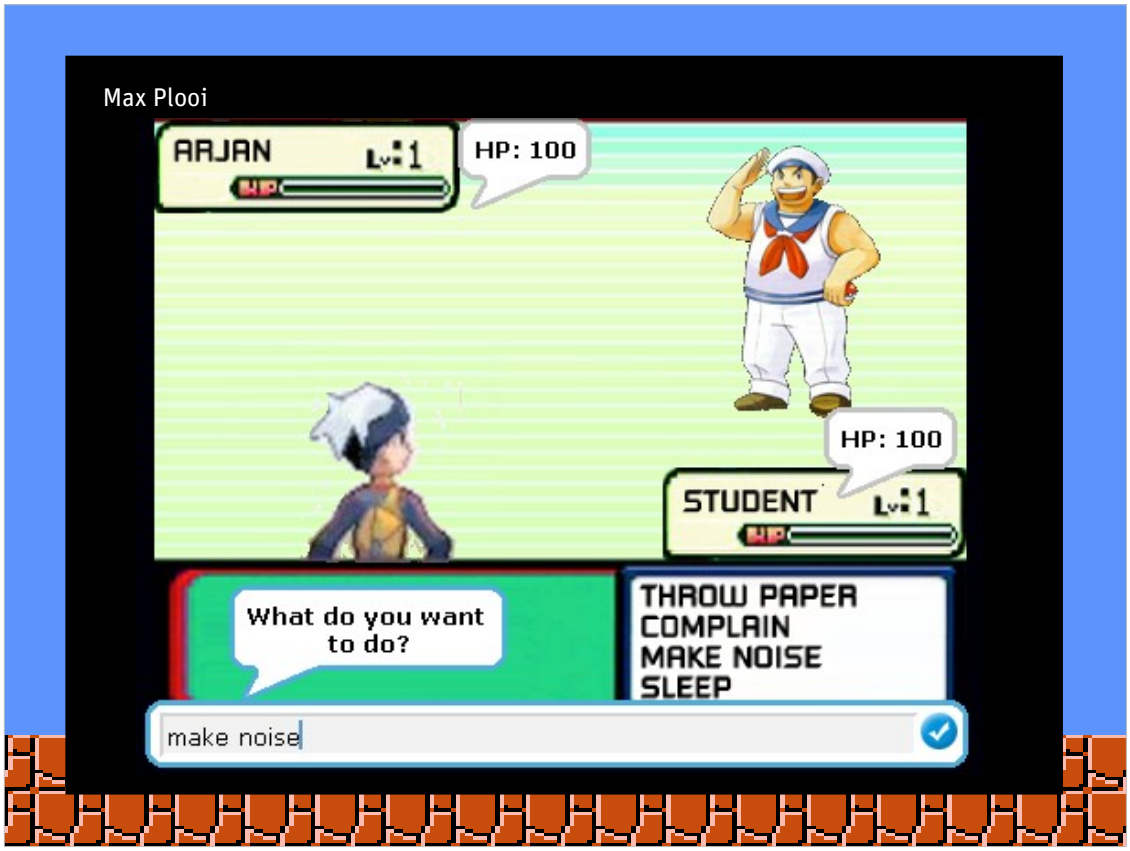
Damir Kalbic




Kevin Nederkoorn
Lives 3
Score 0


Kasper Smits

Max Plooi


ARJAN
Lv:1
HP
HP: 100
HP: 100
STUDENT
Lv:1
HP
What do you want to do?
THROW PAPER
COMPLAIN
MAKE NOISE
SLEEP
make noise

Download alle 92!

- *klik*

  http://usat1112.hku.nl/wp-content/uploads/2011/11/scratch_games.zip

Ontwerp van software

- Don't Repeat Yourself (DRY)
  - Zijn er dingen te herbruiken?
    Denk in classes, overerving, functies
- Keep It Simple, Stupid (KISS)
  - Maak het in eerste instantie niet moeilijker dan nodig, dat kan later altijd nog
- De 80/20 regel
  - De laatste 20% kost 80% van de tijd...

Design patterns (heel snel)

- “Standaard manieren om dingen te doen”
- Singleton
  - Als je maar 1 instantienodig hebt van een class
- Factory
  - Het uitbesteden van het instantieren van objecten
- Listener/observer
  - Loskoppelen van logica

Komende colleges

- ~~College 1: waar hebben we het over~~
- ~~College 2: imperatief programmeren~~
- ~~College 3: object orientatie~~
- ~~College 4: vervolg OO, design patterns, Q&A~~
- Tentamen 8 december 10:00 – 11:30

Tentamenstof (1)

- De 6 best practices uit college 1

Tentamenstof (2)

- Pseudocode snappen EN kunnen schrijven
- Pseudocode begrippen
  - If/then/else
  - Loops
  - Variabelen
  - Functies

Tentamenstof (3)

- Object-georienteerd kunnen denken
- Begrippen:
  - Class vs. object
  - Interface vs. implementatie
  - Encapsulatie
  - Overerving (inheritance)
  - Compositie

Beoordeling

- Tentamen
  - 60% multiple choice
  - 40% open vraag
- Opdracht
  - Opdracht 1 ingeleverd
  - Opdracht 2 voldoende
- **Beoordeling = (opdracht + tentamen) / 2**

Oefenopdracht

- Bedenk een goede tentamenvraag!
  - Multiple choice
    - 1 goed antwoord, 2 foute antwoorden
  - Open vraag
    - met voorbeelduitwerking
- 2 van de inzendingen komen terug in het *echte* tentamen!

Zijn Er Nog Vragen?

- Handige aantekeningen PDF binnenkort online!
  - Hou blog in de gaten.


Leer-ze en tot volgende week!