Organisation:

* Jessica, leader: data structure + gameplay
* Celande: gameplay
* Theophile: graphics

2 weeks

Tools/Plugins :

* Github
* Eclipse
* Maven
* JUnit
* JavaFX

Design pattern : State + Singleon

* Mother
  + Deposite state
  + Stock state
  + Sold out state
  + No coin state

Only 1 vending machine contains :

* 9 different products
* 10 products max for each
* Init : 0€

Different products :

* Price
* Quantity
* Max quantity = 10 (static)
* Id
* Image (see for cache)
* (Name)

Deposite state :

* Usual
  + Machine takes as input the id of the product
  + Customer choose id\*
  + Machine checks on availability\*
  + Machine shows the price\*
  + Customer can enter the price\*
    - Click on coins/bills
      * 0.50, 1, 2, 5
    - Machine decrease the price for each new input
  + Machine validates and give the change
  + Machine gives the product and decrease the quantity
* Cancel (button)
  + Give money back
  + Increase product if has been decrease

\*: can cancel

Stock state:

* Admin can select the quantity to add on
  + X1
  + X5
  + Max
* Admin selects the product
* Machine tries to add the quantity to the product
  + Check if it does not exceed the max quantity
* Take the money: 0€

Sold out state:

* At least 1 product is fully missing
* Have to call for restock

No coin state:

* No change
* The beginning

Switch between the two states?

We don’t manage the change stock.

The customer can order as many products as he wants in the stock limits.