

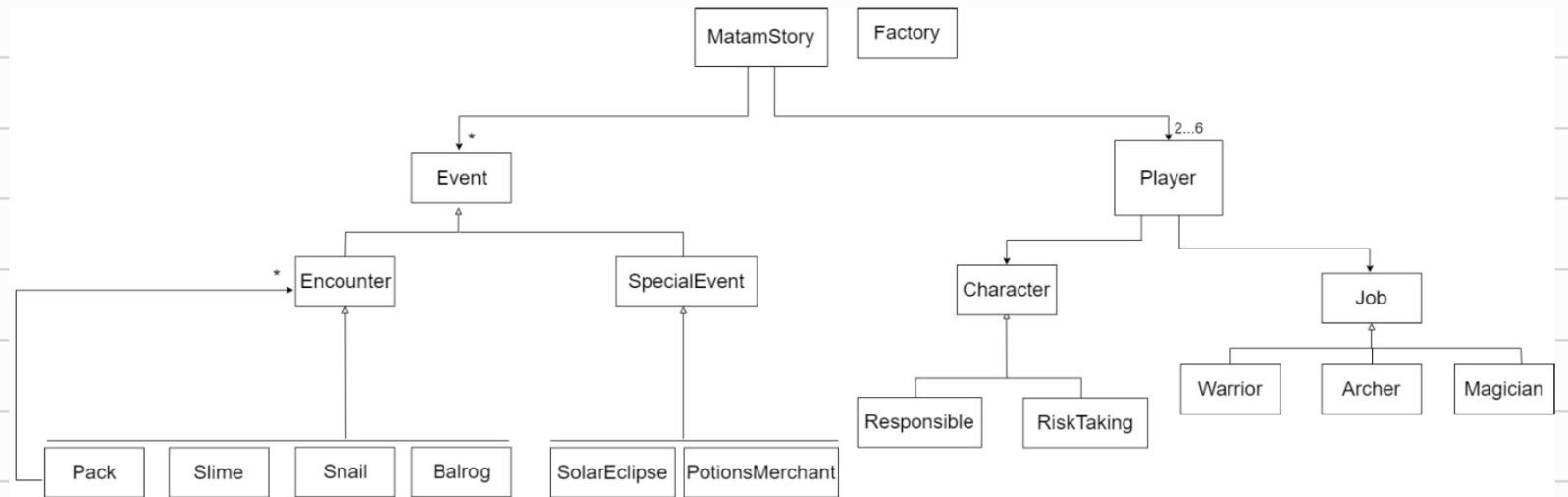
עקבים - נס - גלגלן
318403193 'יכנס' כוונת
208638171 ג'נובה' ג'נובה

שרב להקוחה במדמיה

Pack 2 Snail Snail
Pack 3 Pack 2 Balrog Slime Snail Balrog
Pack 4 Snail Snail Slime Pack 2 Slime Slime
Pack 2 Snail Slime
Pack 2 Pack 2 Pack 2 Balrog Balrog Slime Snail
Pack 3 Slime Snail Snail
Pack 3 Pack 3 Balrog Slime Snail Balrog Balrog
Pack 4 Snail Snail Slime Pack 2 Slime Slime
Pack 2 Snail Slime
2 Pack 3 Pack 2 Balrog Balrog Slime Slime Snail
Pack 2 Snail Snail
Pack 3 Pack 2 Balrog Slime Snail Balrog
Pack 5 Snail Snail Snail Slime Pack 2 Slime Slime
Pack 2 Snail Slime
Pack 2 Pack 2 Pack 2 Balrog Balrog Slime Snail
Pack 2 Snail Snail
Pack 3 Pack 2 Balrog Slime Snail Balrog
מה שכאבי מדמיינים
Pack 2 Snail Slime
imgflip.com



1 Notice



2 Notice

הנתקן בראת מילויים עליונים

Encounter-וּ מילויוֹ רצ' Pack גורם ל - Composite (

הנתקן בראת מילויים עליונים Encounters גורם ל - Composite (

Encounter גורם ריג'ס'ר ו'קונטנ'ט ו'אלט ו'אלט גורם

הנתקן מילויים עליונים מילויים עליונים עליונים - Strategy (2

Player גורם מילויים עליונים גורם ל קון'טנ'ט גורם ל Job -> (k

: מילויים עליונים גורם ל Job -> מילויים עליונים גורם ל קון'טנ'ט

מילויים עליונים גורם ל קון'טנ'ט גורם ל getCombat מילויים עליונים

מילויים עליונים גורם ל קון'טנ'ט גורם ל Job גורם ל קון'טנ'ט גורם ל

מילויים עליונים גורם ל קון'טנ'ט גורם ל getCombat מילויים עליונים

Player. getJob(). getCombat(level, force) :

PotionsMerchant -> מילויים עליונים גורם ל Character גורם ל (2

Character גורם ל קון'טנ'ט גורם ל קון'טנ'ט גורם ל קון'טנ'ט גורם ל

Player. getCharacter.buy(Player)

מילויים עליונים גורם ל Character גורם ל קון'טנ'ט גורם ל קון'טנ'ט גורם ל

מילויים עליונים גורם ל קון'טנ'ט גורם ל קון'טנ'ט גורם ל קון'טנ'ט גורם ל

ו-
.player לשני השחקנים. שחקן הראשון (const) יזכה
(player לשני השחקנים. שחקן השני (const) יזכה)

לפיה שחקן הראשון שחקן strategy-הו WINNER

לפיה שחקן השני שחקן strategy-הו LOSER
ולפיה שחקן השני שחקן strategy-הו WINNER
ולפיה שחקן השני שחקן strategy-הו LOSER
ולפיה שחקן השני שחקן strategy-הו WINNER
ולפיה שחקן השני שחקן strategy-הו LOSER

לפיה שחקן השני שחקן strategy-הו WINNER
ולפיה שחקן השני שחקן strategy-הו LOSER

ולפיה שחקן השני שחקן strategy-הו WINNER
ולפיה שחקן השני שחקן strategy-הו LOSER

ולפיה שחקן השני שחקן strategy-הו WINNER
ולפיה שחקן השני שחקן strategy-הו LOSER

ולפיה שחקן השני שחקן strategy-הו WINNER
ולפיה שחקן השני שחקן strategy-הו LOSER

ולפיה שחקן השני שחקן strategy-הו WINNER
ולפיה שחקן השני שחקן strategy-הו LOSER

ולפיה שחקן השני שחקן strategy-הו WINNER
ולפיה שחקן השני שחקן strategy-הו LOSER

ולפיה שחקן השני שחקן strategy-הו WINNER
ולפיה שחקן השני שחקן strategy-הו LOSER

ולפיה שחקן השני שחקן strategy-הו WINNER
ולפיה שחקן השני שחקן strategy-הו LOSER

ולפיה שחקן השני שחקן strategy-הו WINNER
ולפיה שחקן השני שחקן strategy-הו LOSER

ולפיה שחקן השני שחקן strategy-הו WINNER
ולפיה שחקן השני שחקן strategy-הו LOSER

13) INTRODUCE POLYMORPHISM BY ADDING IS_ABLE WHICH
IS_ABLE HAS A BASE CLASS String WHICH IS_ABLE
FROM WHICH THE DERIVED CLASS (TO WHICH IT REFERS) CAN
ALREADY FIND, CREATE 'IF IT IS A' AND USE
THIS SAME SIGN FOR THIS SAME CODE WHICH IS TO SAY

3. Interface

CLASS Rouge: Job {
public:

Rouge();

Rouge(const Rouge&) = delete;

Rouge& operator=(const Rouge&) = delete;

std::unique_ptr<Job> clone() override; }

THIS IS HOW WE CAN DO IT IN C++ Job IS_ABLE IS_ABLE
WE CAN USE clone('IF IT IS A') OR 'IF IT IS A' WE CAN USE
Rouge FOR THE OTHER METHODS WHICH ARE NOT RELATED TO "IF IT IS A"
WE CAN USE THE SAME SIGN FOR THE OTHER METHODS WHICH ARE RELATED TO "IF IT IS A"
(FOR EXAMPLE WE CAN USE)

ACROSS THE WORLD OF METHODS WHICH ARE RELATED TO "IF IT IS A" (2
.Factory - 2

13) (C++) PRACTICE AGAIN WITH THE SAME SIGN (3
FINDING IN), Job FOR ACROSS THE WORLD OF METHODS WHICH ARE RELATED TO "IF IT IS A"
getCombat() OR Warrior

Rouge - ↗ ۱۵۱۳۰

בנין מילויים ופונקציית getCombat שבודקת אם יש בולט או נזק.

Job:: getCombat(unsigned int force, unsigned int level, unsigned int enemyCombat = 0);

רָגְבָּה לְ Co^{mbat} - הַיְלֵן לְ נִכְחָה, הַיְלֵן יְנֻסָּה

• پیشنهاد ۲ پیشنهاد معرفت اکتشافی که ریاضیات اندیشه است

הנתקה מכם נסב למשך ימים אחדים.

רְאֵבָן וְאַבְנָה אֲבָנָה (אֶנְבָּנָה רְכָב רְכָבִין כְּבָשׂוֹן)

جیلیز پرنسپال نیک پینکنیک

Encounter :; fight

‘What is the best way to get rid of the problem?’, he said.

1) *Neur* se nsof *respon*, se combat \rightarrow *Il* *re* *pro**te**n* *re**ac*

רֹאשׁוֹת אַיִלָּה כְּבָדָה וְעֵינָיו כְּבָדָה. Rouge:: getCombat (1) כְּבָדָה

UnSigned int Rouge :: getCombat (. . . level . . . force . . . enemyCombat) {

unsigned int combat = level + force;

```
if (enemyCombat >= 2 * combat) {
```

```
return enemyCombat + 1;  
}
```

return combat;

• **אֶת־בְּנֵי־עַמּוֹ** אָמַר־יְהוָה אֱלֹהִים־אֲלֵיכֶם

```
    void Encounter::Play(Player& p) {  
        if (p.getHealth() <= 0) {  
            cout << "The player has died." << endl;  
            return;  
        }  
        cout << "The player is fighting the enemy." << endl;  
        cout << "The player attacks the enemy." << endl;  
        cout << "The enemy attacks the player." << endl;  
        cout << "The player wins the fight." << endl;  
        cout << "The player's health is now " << p.getHealth();  
        cout << " and the enemy's health is " << getHealth();  
        cout << endl;  
    }
```

void Encounter::Play (Player &) {

 Player.getJob().Fight(*this, Player &);
}

void Job::Fight (Encounter & encounter, Player & player) {
 Job job;

 if (job == "Rouge") {
 cout << "Job: Rouge" << endl;

 if (job == "Rouge") {
 cout << "Job: Rouge" << endl;
 cout << "Job: Rouge" << endl;

 cout << "Job: Rouge" << endl;

 cout << "Job: Rouge" << endl;

 } else if (job == "Mage") {
 cout << "Job: Mage" << endl;

 cout << "Job: Mage" << endl;

 cout << "Job: Mage" << endl;

 }
}

 if (encounter.getCombat >= 2 * this->getCombat()) {
 cout << "Job: Mage" << endl;

 cout << "Job: Mage" << endl;

 }
}

 else {
 cout << "Job: Mage" << endl;

 }
}

 if (!Strategy) {
 cout << "Job: Mage" << endl;

 }
}

 if (Strategy) {
 cout << "Job: Mage" << endl;

 cout << "Job: Mage" << endl;

 cout << "Job: Mage" << endl;

Class DivineInspiration : SpecialEvent {

public:

Divine Inspiration());

Divine Inspiration (const Divine Inspiration &) = delete;

DivineInspiration& operator= (const DivineInspiration&) = delete;

Void play (Player &) override;

{

Factory → Monolithic → Microservices → API Gateway → Bus (2)

```
Void Player::setJob (uniqueptr<Job> newJob) {
```

this → Job = std::move(newJob);

{

מגניב לנו. Factory::getRandomJob() מילא לנו גיבוב (4
随机选取一个，返回一个随机的 Job 对象) 但
Size -> 如果所有随机选取的 Job 对象的
大小 |> N 那么返回的 Job 的 size 就是
随机的。如果所有 Job 的 uniqueptr < Job > 亂
Factory-ה מילא לנו גיבוב Job, NGEN

void DivineInspiration::Play() {
 cout << "Divine Inspiration!" << endl;
}

Vold Divine Inspiration :: Play (Player & player) {

```
std::unique_ptr<Job> newJob = Factory::getRandomJob();
```

```
player.setJob(std::move(newJob)),
```

```
print DivineInspirationMessage( Player, player.getJob().getName());
```

{

1151

