Les Bitmasks - Fonctionnement et Exemples

Galathil

Important : Ce tutoriel a été rédigé par Nobodie pour le site wow-emu.fr. Je le recopie ici à l'identique, Il se peut que certaines informations soient obsolètes à ce jour. Bonne lecture !

Les Bitmasks - Fonctionnement et Exemples

Prélude : Ce sujet n'est pas original, il a été transféré d'un précédent forum d'émulation en date d'Octobre 2013. J'en suis l'auteur, je le transfère donc pour les besoins de ce forum. Bien qu'il soit en partie ré-écrit, il se peut que certains termes soient inadaptés ou ne soient plus d'actualité.
Ce tutoriel n'a pas été directement écrit par moi-même, mais par un des membres de l'équipe Mist-Eria. ce sujet est posté avec l'accord de son auteur. Des modifications peuvent y être apportées dans un but de clarification.

Bonjour tout le monde ! Dans ce tutoriel je vais vous expliquer ce qu'est un Bitmask, comment ca marche et quelques exemples d'utilisation pour un serveur WoW.

Tout d'abord voici quelques lien utiles mais pas indispensables à la comprehension de ce tutoriel :
http://en.wikipedia.org/wiki/Bit_field
http://fr.wikipedia.org/wiki/Syst%C3%A8me_binaire

1) Qu'est ce qu'un Bitmask ?

Un Bitmask est une donnée traitée bit par bit, elle est stockée sur un entier. Par exemple on peut utiliser un entier de 32 bits pour stocker 32 informations booléenes, chacune sur un bit, correspondant à chaque puissance de 2 entre 2^0 et 2^{31}, dans le cas d'un entier de 32 bits.

La valeur entière du bitmask n'a pas de sens, seul les bits nous intéressent. On utilisera donc les opérations bit à bit pour les manipuler.

Prenons un exemple,

Les npc ont un bitmask appelé "npcflag", ou chaque bit de ce masque représente un flag, qui défini si le npc propose ou non une fonctionalité.
Voici la valeur binaire et décimale de quelques de ces flags :

```
0000 0001 - 1   => Gossip
0000 0010 - 2   => Quest Giver
0001 0000 - 16  => Trainer
1000 0000 - 128 => Vendor
```

Si on veut que notre npc soit à la fois un vendeur et un donneur de quête il suffit de lui mettre le flag 2 et 128 en même temps.
On utilise l'opérateur OU bit à bit :

```
0000 0010  => flag 2
OU 1000 0000  => flag 128
=  1000 0010  => bitmask qui contient le flag 2 et 128
```

Le bitmask résultat a une valeur de 130, et 130 est composé de 128 et 2 =)

Maintenant comment a partir du bitmask retrouver quels flag sont a 1 ?

Prenons un npc qui possède ce mask : 0111 0110. On veux savoir si ce npc est un Trainer (flag 16).

Il suffit pour cela de faire un ET bit à bit entre ces deux valeurs :

```
0111 0110 => mask du npc
ET 0001 0000 => flag 16
=  0001 0000 => résultat
```

Un résultat égal a ce flag indique la présence du flag recherché.

2) Exemples C++ et SQL

Voici quelques exemples d'utilisation des bitmasks.

En SQL :

Nous souhaitons connaitre la liste de tout les npc Vendeurs.

```
SELECT * FROM creature_template WHERE (npcflag & 128) = 128;
```

Nous souhaitons maintenant ajouter ce flag 128 (Vendeur) au pnj 12345.

```
UPDATE creature_template SET npcflag = npcflag | 128 WHERE entry = 12345;
```

Simple comme tout non ?

En C++ :

Nous souhaitons effectuer une action si un npc est un Trainer (flag 16).
En C++ le ET bit à bit est réprésenté par "&". A ne pas confondre avec le ET logique "&&" !

```
if (npcflag & 16)
{
    // Action a effectuer
}
```

Encore une fois, très simple. Nous pouvons aussi ajouter ce flag au bitmask npcflag :

```
npcflag = npcflag | 16;
```

ou

```
npcflag |= 16;
```

Remarque : On préfèrera utiliser des énumérations pour stocker les valeurs des flags, plutôt que d'utiliser directement leur valeur. En effet, ces valeurs changent souvent à chaque MAJ/Extension, cela permet de n'avoir qu'une seule ligne de code a changer par flag.

```
enum NPCFlags // En Hexa dans cet exemple
{
    UNIT_NPC_FLAG_TRAINER              = 0x00000010, // = 16
    UNIT_NPC_FLAG_VENDOR               = 0x00000080, // = 128
    // ...
}
```

Faites attention à la priorité des opérateurs ! (**http://en.cppreference.com/w/cpp/languag...precedence**)

Dans le doute, mettez des parenthèses.

Voila pour ce tuto sur les Bitmask, un concept pas très compliqué mais tellement pratique et surtout **indispensable** dans le domaine de l'émulation (quelle qu'elle soit) !

~~J'espère qu'il vous plaira, n'hesitez pas à poser des questions à la suite~~

---

noc

> **Galathil** Un Bitmask est une donnée traitée bit par bit, elle est stockée sur un entier. Par exemple on peut utiliser un entier de 32 bits pour stocker 32 informations booléenes, chacune sur un bit, correspondant à chaque puissance de 2 entre 20 et 231, dans le cas d'un entier de 32 bits.

En fait un bitmask c'est une donnee binaire, des 1 ou des 0, elle est placee dans une variable numerique qui a un format comme toutes les variables numeriques entieres non signees (sans signe positif ou negatif). Le format depend de ce que l'on a besoin, autrement dit 8 bits, 16 bits, 32 bits, ... d'informations comme une variable numerique entiere non signe classique, de meme que son interpretation ne depend que des besoins de son createur.
