Gestion de la mémoire

Garbage Collection is BAD !

Gestion de la mémoire

- Principe
- Compteur de références
- ARC

Principe

Bases

- Plateforme mobile = ressources limitées
- Gestion de la mémoire efficace indispensable !
  - Donc pas de Garbage Collection

Bases

- Ne pas se soucier de quand est libéré l’objet
- Indiquer au système quand on se sert et ne se sert plus d’un objet

Évolution

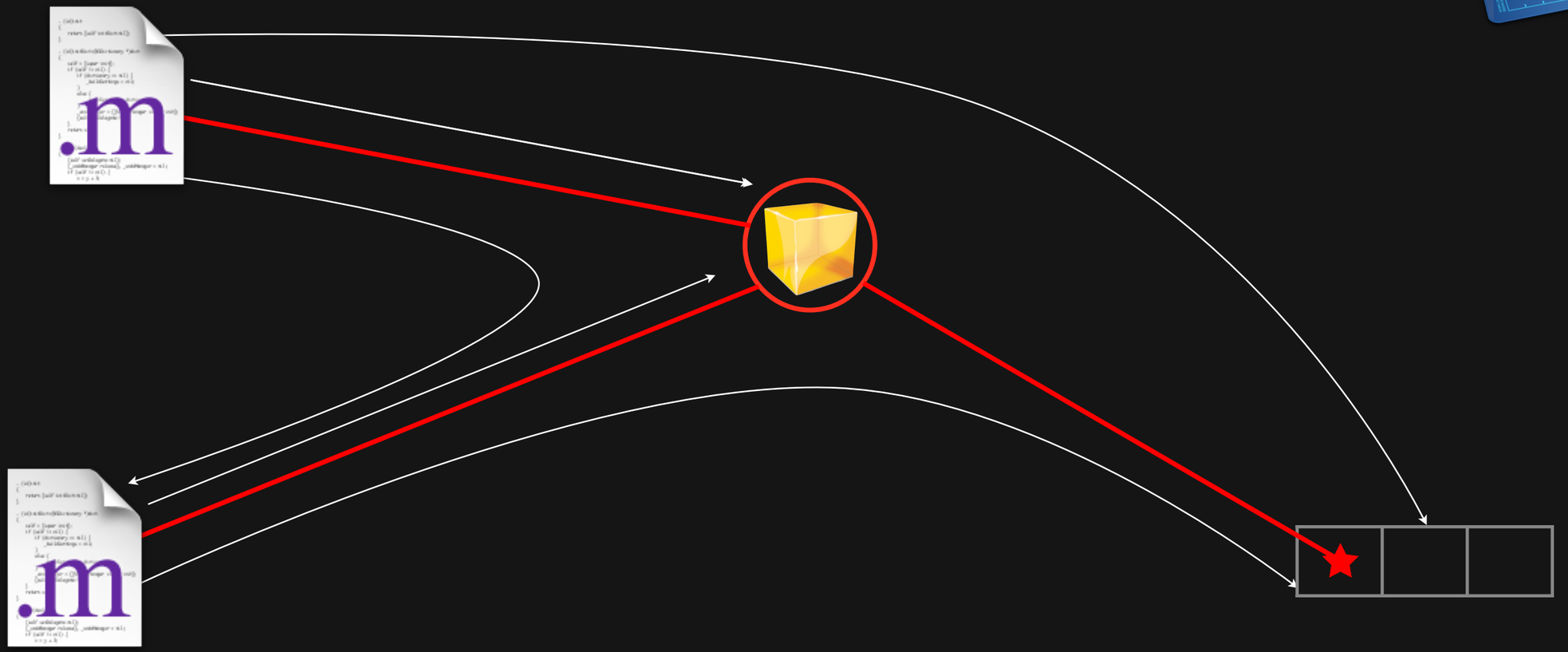
- Historiquement, la gestion était manuelle
  - On indiquait quand on utilisait un objet, et quand on avait fini de s'en servir
  - Cela influait sur un compteur de référence (nombre de référence pointant sur un même objet)
- Depuis iOS 5, il existe un système de gestion automatique (ARC).
  - Le développeur n'a plus à gérer le compteur de référence
  - Les nouveaux projets sont en ARC

Compteur de références

Principe

- Chaque objet possède un compteur de références
  - Quand on utilise l’objet, on incrémente le compteur
  - Quand on n’a plus besoin de l’objet, on décrémente
  - Lorsque le compteur passe à 0, la mémoire est libérée

Compteur de références



Le deuxième «bout de code» n'a plus besoin de l'objet

ARC

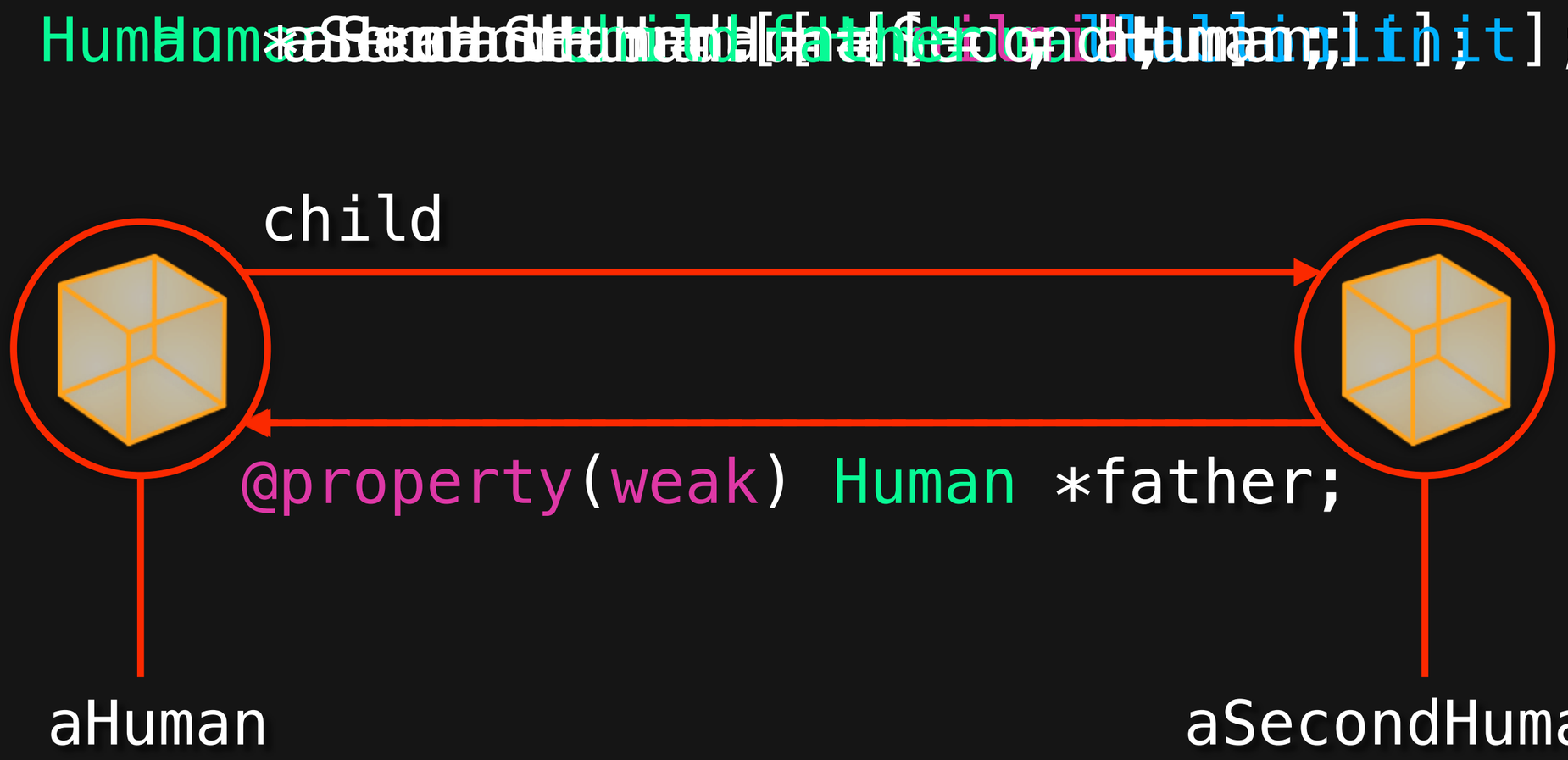
ARC

- Fonctionnalité au niveau compilateur
- Compteur de référence géré par le compilateur lors de la compilation
- Implique de suivre quelques règles pour bien fonctionner

- Ne pas influer sur le compteur manuellement (pas de **retain**, **release**, **autorelease** ou **dealloc**)
- Ne pas stocker de pointeurs objet dans des structures C
- Ne pas faire de cast direct entre types objets et non-objets
- Ne pas utiliser d’objets **NSAutoReleasePool**
- **Penser à son graphe d’objets !**

```
Human *aHuman = [[Human alloc] init];
Human *aSecondHuman = [aHuman retain];
```

Cycle de références



- Quantifieurs de durée de vie
  - Pour les *properties*
    - `strong` : l'objet doit rester vivant tant qu'on pointe vers lui **(par défaut)**
    - `weak` : l'objet restera vivant tant qu'un pointeur `strong` pointera vers lui
  - Pour les variables
    - `__strong` **(par défaut)**
    - `__weak`

Pour aller plus loin…

- Advanced Memory Management Programming Guide
- Transitioning to ARC