

Tool Assisted Speedrun (TAS)

Si vous avez déjà regardé un speedrun en vous disant "mais ça ressemblerait à quoi si c'était effectué absolument parfaitement ?", eh bien vous avez fait le premier pas dans le monde du Tool Assisted Speedrun.

Cette discipline (communément appelée "TAS"), n'est pas compétitive, contrairement au speedrun. Elle est collaborative. En partie parce que le travail demandé à la réalisation d'un TAS est tellement grand que la compétition n'a pas sa place, mais aussi parce qu'elle n'aurait pas vraiment de sens.

Le principe d'un TAS, c'est de faire un [speedrun](#), mais en remplaçant l'aspect "performance" par un aspect de "travail perfectionniste". Dans un TAS, on peut se permettre d'utiliser des outils externes au jeu pour en avoir une meilleure compréhension, et un meilleur contrôle. Attention néanmoins, il ne s'agit pas de tricher ! Aucune modification préalable du code du jeu n'est tolérée (par des codes de triche ou autres). Le TAS vise à être "vrai", et à pouvoir être exécuté sur n'importe quelle version du jeu, sans intervenant autre que la manette (des fois, plusieurs manettes).

Parfois, le but du TAS n'est pas de finir le jeu le plus vite possible, mais de montrer que quelque chose est possible. On parlera alors de **Tool Assisted Demonstration**, ou TAD.

Le site de référence pour les TAS est tasvideos.org.

Pour faire un TAS, on va utiliser des outils perfectionnés (comme des [émulateurs](#) ultra précis), qui vont nous permettre notamment:

- de ralentir le jeu jusqu'à avancer image par image (et dans certains cas, [encore plus lentement](#))
- de ré-enregistrer des segments autant de fois que nécessaire (retour en arrière)
- de modifier librement la séquence des touches sur lesquelles on va appuyer et leur timing comme on modifierait le script d'une pièce de théâtre
- de voir le contenu de la mémoire interne au jeu (pas la modifier directement, mais la consulter oui)

De plus, le fait d'avoir un contrôle total sur la précision des touches rentrées sur la manette permet généralement l'exécution de tricks normalement absolument impossible à effectuer par des humains. On parle alors de tricks *TAS-only*.

La constitution d'un TAS est un travail de minutie et de patience. Certains projets peuvent prendre plusieurs mois pour être complétés, voir plus, au point que certains vont parfois proposer des **LOTAS** (pour *Low Optimization TAS*) ou **LOTAD** (pour *Low Optimization TAD*), afin d'obtenir un résultat non optimal mais donnant une bonne idée d'un TAS ou d'une démonstration dans un temps de constitution raisonnable.

Un exemple célèbre de TAS est le 1 key de *Super Mario 64*.

<https://www.youtube.com/embed/TkOkjvLKxUY>

Un exemple célèbre de TAD est le no door de *The Legend of Zelda: Ocarina of Time* (qui consiste à faire tout le jeu sans ouvrir de porte):

<https://www.youtube.com/embed/S9dLpmXR0kw>

Arbitrary Code Exectution (ACE)

L'ACE, parfois appelé "Total Control" est dans le domaine du TAS généralement perçu comme le nanan du nanan, le Saint Graal, l'objectif final de tout TASeur sur un jeu.

Il s'agit en fait de pouvoir effectuer une exécution arbitraire de code uniquement en se basant sur le jeu choisi et sur les boutons pressés sur la manette. Cela permet de littéralement recoder ce que vous voulez directement en passant par le jeu, et donc à partir de là faire réellement ce que vous voulez.

La méthode pour exécuter un ACE (lorsque c'est seulement possible) va varier grandement d'un jeu à un autre, donc il est difficile d'écrire une méthodologie. Néanmoins, lorsqu'un ACE est trouvé, il devient généralement très célèbre.

Un exemple très connu d'ACE est celui de *Pokemon Red* qui recode un chat Twitch avec une interface réseau fonctionnelle en utilisant un Super Gameboy (la démonstration démarre vers 6m40) :

<https://www.youtube.com/embed/P28kp66XMw4>