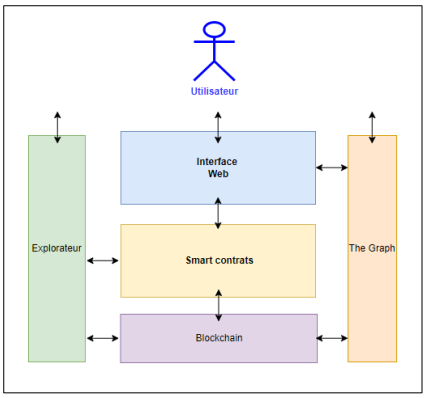
Le YAM, comment ca marche...

En complément du guide utilisateur, pour les curieux qui souhaiteraient comprendre comment ça marche (« DYOR »), nous allons regarder ce qui se passe derrière l'application….
(pas d'inquiétude : aucune compétence en développement n'est nécessaire ;-) ..)

1 - Les composants d'une l'application WEB 3 (comme YAM ou RMM)

Les APPlications distribuées (dAPP) fonctionnent sur une blockchain. Elles sont constituées :

- d'une partie Interface Web (Front-end), que vous accédez via l'url du site et sur laquelle vous vous connectez avec votre wallet,
- de smart contracts, cœur de l'application, qui sont enregistrés et qui s'exécutent sur une blockchain (ici Gnosis),
- et dans le cas présent, d'un service d'accès et d'indexation des données sur la blockchain (The Graph).



L'utilisateur peut aussi accéder aux smart contracts de l'application, directement sans passer par l'interface, au moyen d'un explorateur de blockchain.

Pour la Gnosis Chain, deux explorateurs sont possibles : https://gnosisscan.io/ et https://gnosis.blockscout.com/ . (c'est bien utile d'en avoir deux, car ils sont parfois ponctuellement coincés)

Le service d'indexation de données, The Graph, est accessible par l'utilisateur : c'est assez complexe et une simple utilisation sera présentée en fin de document.

L'accès à l'application, à partir de l'interface, est détaillé dans le guide utilisateur. Dans le présent document, nous allons accéder à l'application sans passer par l'interface et ainsi analyser ce qui se passe aux niveaux inférieurs…

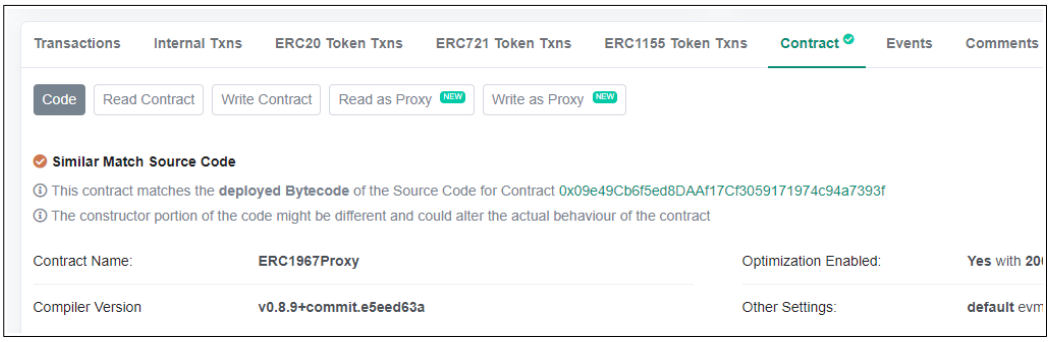
2 – Actions sur le smart contract

Vous pouvez accéder à l'application, sans passer par l'interface et directement auprès des smart contracts : soit pour contourner une défaillance (ponctuelle) de l'interface, soit pour visualiser des informations que l'interface n'affiche pas, soit pour gagner en rapidité (cas des bots..).

Pour accéder a un smart contract avec un explorateur, vous allez avoir besoin de son adresse. Chaque smart contract a une adresse unique, sur la blockchain sur laquelle il est enregistré.

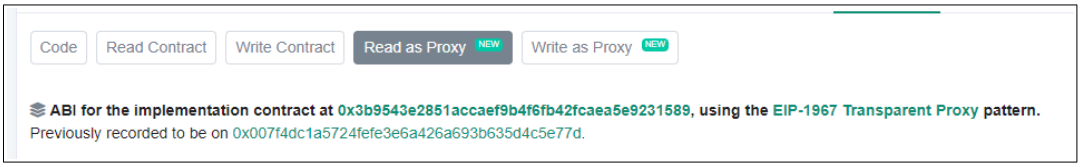
L'adresse du smart contract YAM est : 0xC759AA7f9dd9720A1502c104DaE4F9852bb17C14. Vous pouvez y accéder avec l'explorateur GnosisScan via l'url suivante :
https://gnosisscan.io/address/0xC759AA7f9dd9720A1502c104DaE4F9852bb17C14

L'explorateur donne accès à de multiples informations sur le smart contract : les différents transactions (par type) exécutées, ainsi que le contrat en lui même.
Lorsqu'on sélectionne l'onglet « Contract » les informations suivantes apparaissent : L'accès en lecture ou écriture au contrat ou a son proxy...



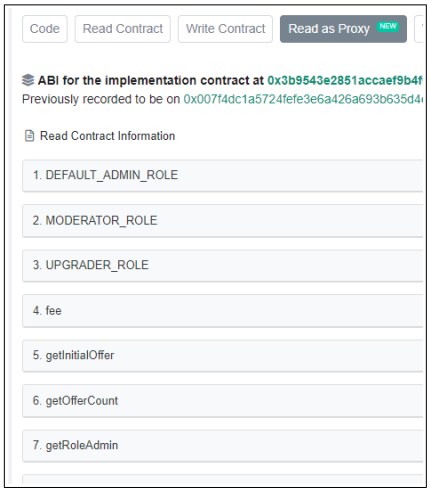
Mais qu'est ce donc qu'un Proxy ?
Les smart contracts, une fois enregistrés à une adresse sur une blockchaine, sont non modifiables. C'est une des forces de la blockchain, mais ce peut être aussi une contrainte lorsqu'on développe et qu'on a besoin de faire des mises à jours. Pour ce faire, on utilise la technique du Proxy, qui permet de modifier la logique du contrat, sans changer son adresse et les valeurs qu'il stocke.
Vous avez alors deux contrats :

- Celui du Proxy : dont l'adresse est fixe et qui stocke les valeurs. Il s'agit ici de l'adresse mentionné ci-dessus. Qui devra être utilisée pour toute action sur le contrat.
- Celui de l'implementation (le logique) du contrat, qui est « masqué » (et accédé) par le Proxy. Son adresse est visible lorsqu'on clique sur Read ou Write « as Proxy », par l'information « ABI for implementation contrat at... » . Cette adresse changeant au fur été à mesure des mises à jour.



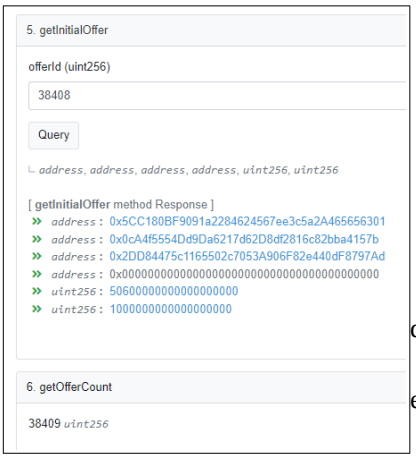
Pour lire la logique du contrat (écrite en Solidity sur Ethereum, ou des blockchaines compatibles comme Gnosis) vous devrez donc aller sur cette adresse.

2.1 - Actions en lecture sur le contrat YAM



L'ensemble des actions possibles, apparaissent en sélectionnant « Read as Proxy ».

Prenons par exemple les actions 5 et 6 :



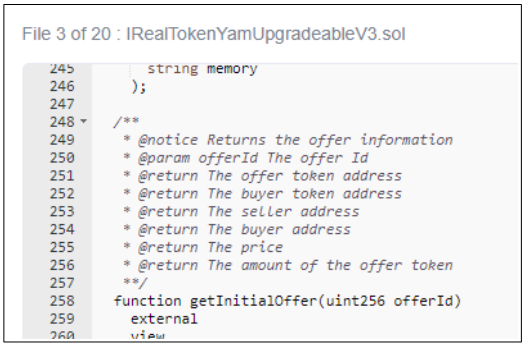
6 – getOfferCount : indique le prochain numéro d'offre disponibles,
5 – getInitialOffer : indique les données de l'offre (en prenant pas exemple la dernière publiée, avec un offer ID = getOfferCount -1 .

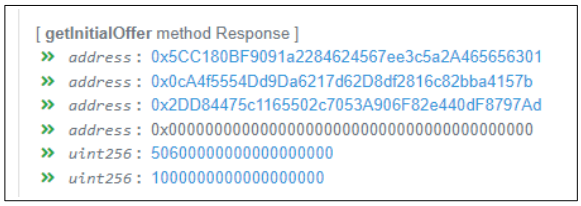
Pour comprendre à quoi correspondent les différentes champs données par un ordre de lecture, il va hélas falloir aller dans le code !… Mais vous allez voir, ce code est très bien écrit et il y a plein de commentaires.

Comme mentionné précédemment, pour accéder au code il faut aller dans un autre smart contract (au moment ou ce document est écrit, il s'agit du contrat ci-après):Il

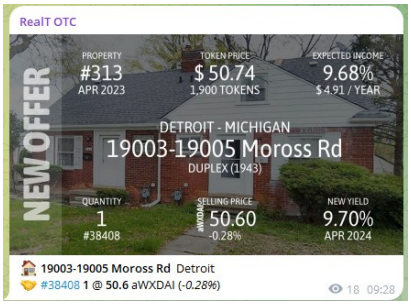
https://gnosisscan.io/address/0x3b9543e2851accaef9b4f6fb42fcaea5e9231589#code

Le smart contrat est constitué de multiples fichiers. celui dans lequel réside les ordres de lecture est le troisième « IRelaTokenbYAMUpgradeableV3.sol ». En cherchant, par exemple, l'ordre de lecture « GetInitialOffer » on trouve à partir de la ligne 249 le commentaire qui indique la signification de chacun des champs :

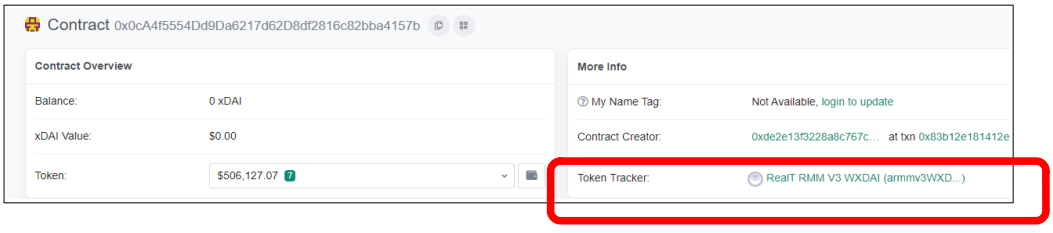




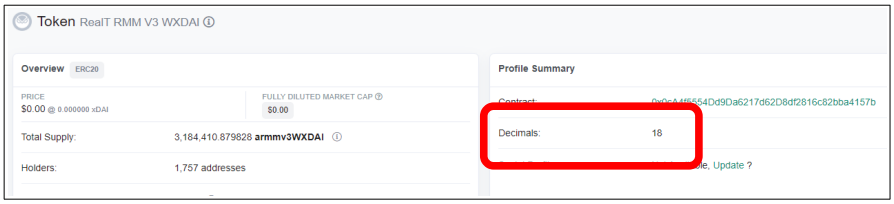
qui correspondent aux informations que vous pouvez, par exemple, retrouvez dans le Telegram des OTC :



La seconde adresse est celle du stablecoin utilisé pour l'offre de vente. Si vous cliquez sur l'adresse l'explorateur vous donnera le nom de ce token : ici armmv3XDAI



En cliquant sur le nom du Token, vous allez trouver une autre information importante : le nombre de décimal de ce token : ici 18



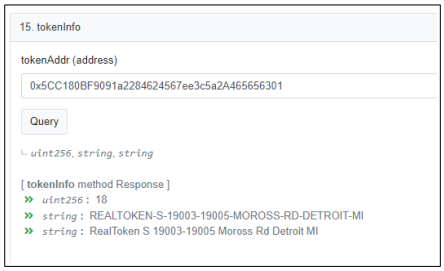
L’avant dernier champ, affiché dans 5 – getInitialOffer, est le montant en armmv3XDAI. Pour convertir le montant affiché (ici : 50600000000000000000), il faut le diviser par 10 puissance 18 (nombre de décimal) ce qui fera 50,6.

Attention, tous les token n’ont pas les même décimales :

- WXDAI : 0xe91D153E0b41518A2Ce8Dd3D7944Fa863463a97d – Decimal : 18
- USDC : 0xDDAfbb505ad214D7b80b1f830fcCc89B60fb7A83 – Decimal : 6
- Armmv3XDAI : 0x0cA4f5554Dd9Da6217d62D8df2816c82bba4157b – Decimal : 18
- ArmmV3USDC : 0xeD56F76E9cBC6A64b821e9c016eAFbd3db5436D1 - – Decimal : 6
- Realtoken : …. Decimal : 18

Parmi tous les ordres de lecture, deux autres sont à voir :

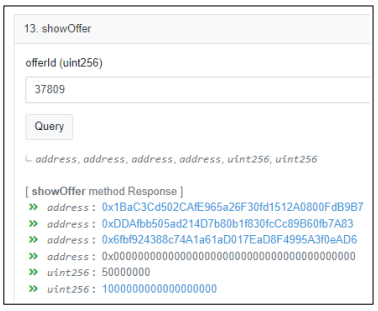
- 13 – showOffer, semblable à 5 – getInitialOffer mais pour l’offre en cours,
- 15 – tokenInfo, qui donne des infos sur l’adresse d’un RealToken



2.1 - Actions en écriture sur le contrat YAM

2-1.1 Achat d’une offre

Prenons comme exemple, l’achat de tout ou partie de l’offre 37809 :



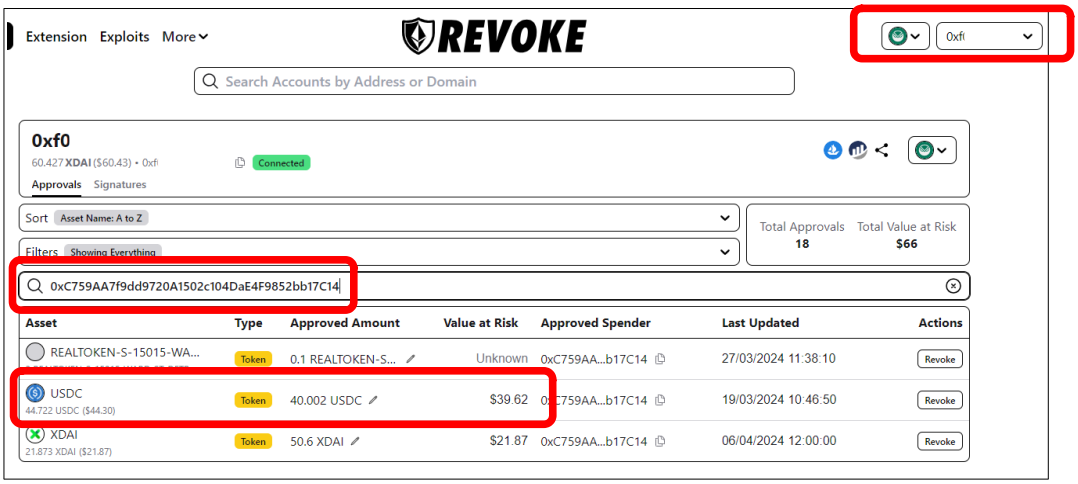
Lors d’un achat, avec l’application front-end, le wallet demande deux autorisations : la première pour que le contrat YAM puisse accéder a vos stablecoin et la seconde pour exécuter l’ordre dans la quantité souhaitée.

En agissant directement sur le smart contract YAM, nous n’allons donner que la seconde autorisation (celle sur le

contrat YAM, puisque c'est sur celui-ci que nous serons connecté).
Il faut donc qu'au préalable, nous ayons donné la première autorisation…

Allowance :

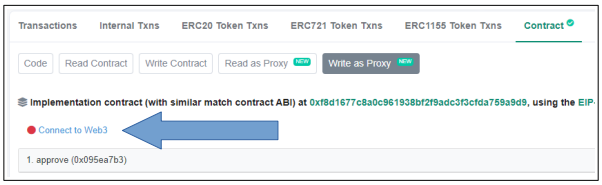
Dans l'exemple, l'offre s'achète en USDC, il faut donc au préalable que le contrat YAM ait été autorisé à dépenser les USDC de votre wallet, pour un montant égal ou supérieur à votre achat.
Pour cela, vous pouvez aller vérifier dans l'application https://revoke.cash/ si l'autorisation est présente :



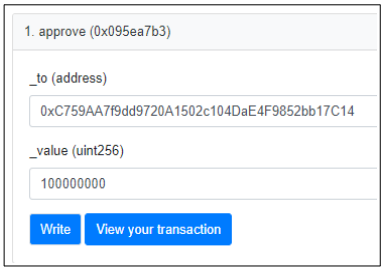
Connecter votre wallet, rechercher les autorisations données au contrat YAM (avec son adresse) et vérifier si l'USDC est listé. Dasn l'image ci-dessus, votre achat ne pourra dépasser 40 USDC.

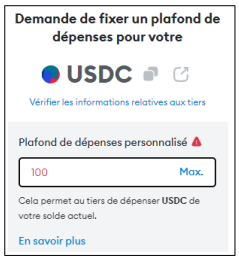
Si aucun USDC n'est listé ou que le montant n'est pas suffisant, il va falloir aller dans le contrat USDC pour faire un approve.

- https://gnosisscan.io/address/0xddafbb505ad214d7b80b1f830fccc89b60fb7a83#writeProxyContract
- Connecter le wallet à autoriser :



- Utiliser l'ordre 1 – approve, avec l'adresse du contrat YAM et la quantité de l'approbation (en décimal 6)

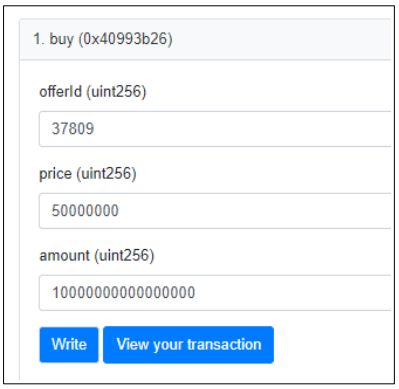




Dans revoke, l'approbation est maintenant passé de 40 à 100

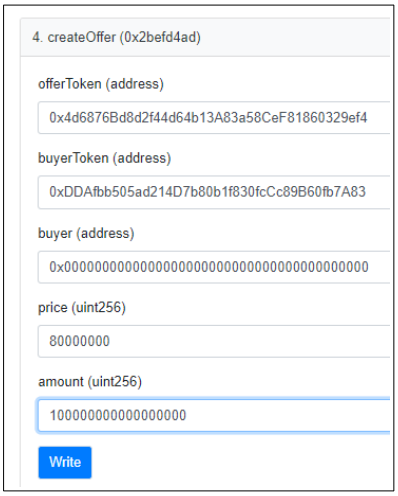
Nous pouvons maintenant aller acheter sur le contrat YAM :

- https://gnosisscan.io/address/0xC759AA7f9dd9720A1502c104DaE4F9852bb17C14#writeProxyContract
- connecter le wallet, et compléter l'ordre 1 – buy :
  - avec le numéro de l'offre,
  - le prix (afin d'assurer qu'il n'y a pas eu de modification juste avant l'achat)
  - la quantité en décimal 18
    Attention a ne pas faire d'erreur sur le nombre de 0, car un de plus c'est 10 fois plus acheté !..
    (sur l'image l'achat est de 0,1 tokenisés

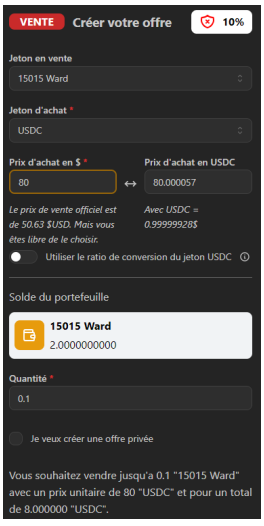


2.1.2 - Création d'une offre

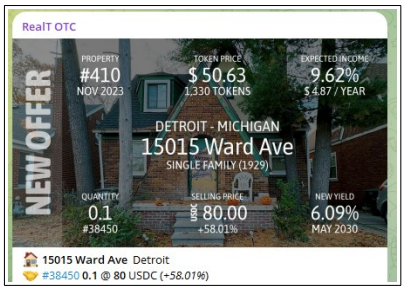
- Ordre : 4 – CreateOffer, avec
- l'adresse du token vendu
- l'adresse du stablecoin d'échange,
- l'adresse de l'acheteur (pour un achat en mode privé), sinon 0
- le prix
  Attention comme le prix est en USDC , il est exprimé en decimal 6 (ici 80 USDC)
- le nombre de token vendu
  Là c'est exprimé en decimal 18 (ici 0,1 token)



A titre de comparaison, sur l'application Front-End, la création de la même offre se présenterait comme cela :
C'est évidement plus simple et plus sécurisé (grâce aux deux protections)

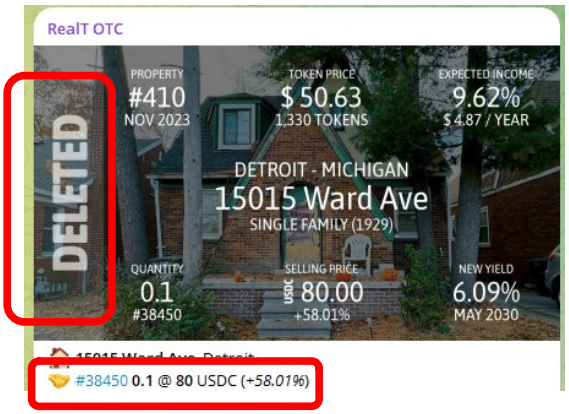


Une fois approuvé, l'offre est visible sur le Telegram OTC



2.1.3 Effacer une offre

Là rien de plus simple, puisqu'il y a juste à indiquer le numéro de son offre

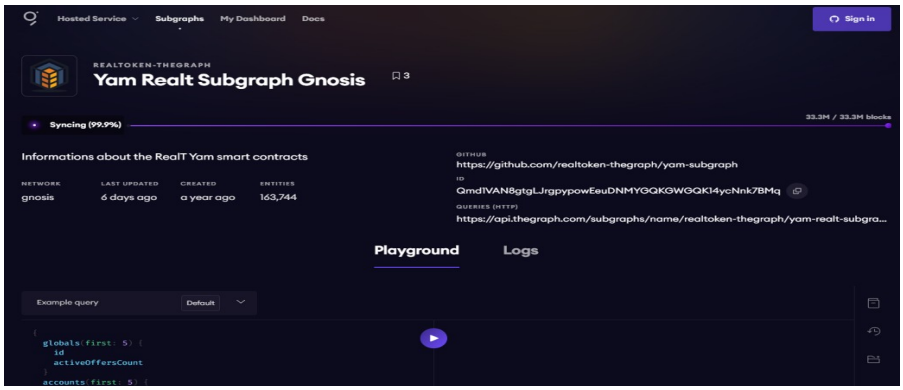


Les autres types d'ordres sont sur le même principe et ne seront pas détaillés.

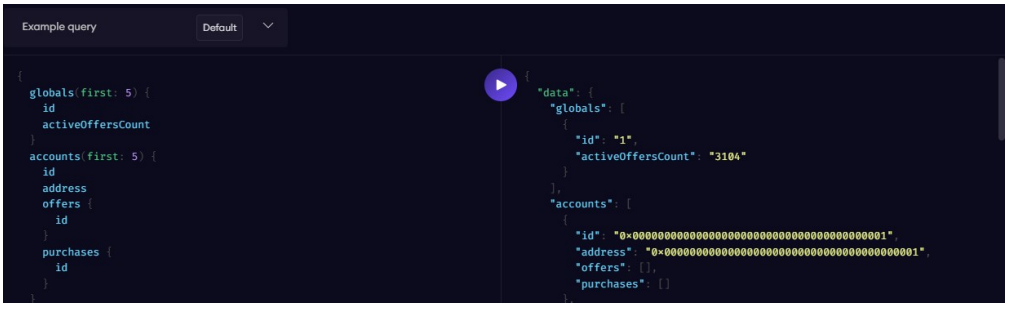
3 - Informations complémentaires, à partir de The Graph

The Graph est un protocol (/service) pour simplifier l'accès aux données des blockchains. RealT a mis en place un Subgraph, spécifique pour les données YAM. Ces données sont accessibles soit via une API,soit via un explorer :

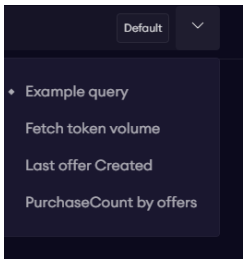
https://thegraph.com/hosted-service/subgraph/realtoken-thegraph/yam-realt-subgraph-gnosis

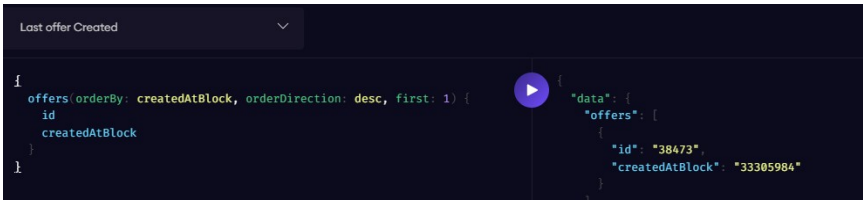


L'interrogation des données se fait au travers de requêtes en GraphQL :
Dans l'explorer elles sont écrites sur la gauche, en cliquant sur la flèche central le résultat apparaît sur la droite.

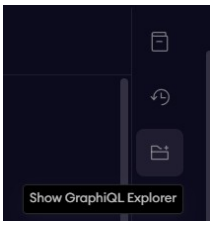


Quelques requêtes sont prédéfini



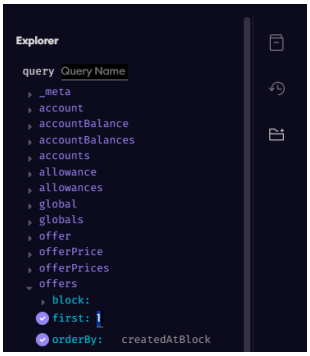


A droite, le dernier icone permet d'ouvrir l'explorateur des données



Vous voyez alors apparaître l'ensemble des données interrogeables (celles cochées correspondent à la requête existante).
La requête se constitue donc au fur et à mesure que vous cochez ou décochez les données souhaitées.

Toute la difficulté étant de savoir à quoi correspondent chacune des données pour le YAM (il n'existe hélas pas de dictionnaire ….).



Les résultats sont présentés en format JSON (plutôt dédié à des programmes, qu'a des humains…)

Pour une lecture simplifiée de ces données, il faut : soit faire un développement, soit se servir d'un programme qui sait lire ce format : par ex Google Sheet (avec l'extension API Connector) .

Dans les « Hosted Service » de The graph (https://thegraph.com/hosted-service) , deux Subgraph sont disponibles pour YAM :